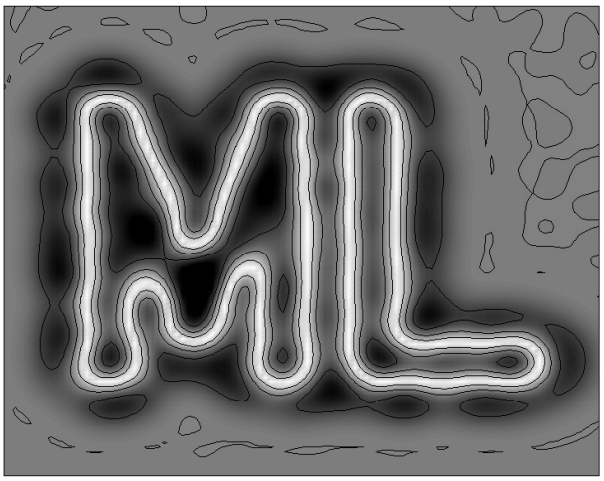
Discriminative Dimensionality Reduction: Variations, Applications, Interpretations

Alexander Schulz



Dissertation

vorgelegt zur Erlangung des Grades
Doktor der Naturwissenschaften (Dr. rer. nat.)

Disputation am 31.03.2017

Universität Bielefeld, Technische Fakultät

First Examiner: Prof. Dr. Barbara Hammer, Bielefeld University, Germany
Second Examiner: Prof. Dr. Paulo Lisboa, Liverpool John Moores University, United Kingdom

Printed on non-aging paper according to ISO 9706.

Bielefeld University – Faculty of Technology
P.O. Box 10 01 31
D-33501 Bielefeld, Germany

Alexander Schulz
Machine Learning research group
CITEC – Cognitive Interaction Technology Center of Excellence
Inspiration 1, D-33619 Bielefeld, Germany
http://www.cit-ec.de/tcs
aschulz@techfak.uni-bielefeld.de

Abstract

The amount of digital data increases rapidly as a result of advances in information and sensor technology. Because the data sets grow with respect to their size, complexity and dimensionality, they are no longer easily accessible to a human user. The framework of dimensionality reduction addresses this problem by aiming to visualize complex data sets in two dimensions while preserving the relevant structure. While these methods can provide significant insights, the problem formulation of structure preservation is ill-posed in general and can lead to undesired effects.

In this thesis, the concept of discriminative dimensionality reduction is investigated as a particular promising way to indicate relevant structure by specifying auxiliary data. The goal is to overcome challenges in data inspection and to investigate in how far discriminative dimensionality reduction methods can yield an improvement. The main scientific contributions are the following:
(I) The most popular techniques for discriminative dimensionality reduction are based on the Fisher metric. However, they are restricted in their applicability as concerns complex settings: They can only be employed for fixed data sets, i.e. new data cannot be included in an existing embedding. Only data provided in vectorial representation can be processed. And they are designed for discrete-valued auxiliary data and cannot be applied to real-valued ones. We propose solutions to overcome these challenges.
(II) Besides the problem that complex data are not accessible to humans, the same holds for trained machine learning models which often constitute black box models. In order to provide an intuitive interface to such models, we propose a general framework which allows to visualize high-dimensional functions, such as regression or classification functions, in two dimensions.
(III) Although nonlinear dimensionality reduction techniques illustrate the structure of the data very well, they suffer from the fact that there is no explicit relationship between the original features and the obtained projection. We propose a methodology to create a connection, thus allowing to understand the importance of the features.
(IV) Although linear mappings constitute a very popular tool, a direct interpretation of their weights as feature relevance can be misleading. We propose a methodology which enables a valid interpretation by providing relevance bounds for each feature.
(V) The problem of transfer learning without given correspondence information between the source and target space and without labels is particularly challenging. Here, we utilize the structure preserving property of dimensionality reduction methods to transfer knowledge in a latent space given by dimensionality reduction.

Acknowledgments

I would like to thank my friends and family for providing great support during the work on my thesis.

I also wish to thank all my colleagues from whom I learned various important things. These are Andrej Gisbrecht, Babak Hosseini, Bassam Mokbel, Benjamin Paaßen, Benoît Frénay, Christina Göpfert, Daniela Hofmann, Frank-Michael Schleif, Jeffrey Queißer, Johannes Brinkrolf, Kerstin Bunte, Lukas Pfannschmidt, Lydia Fischer, Markus Lux, Viktor Losing, Witali Aswolinskiy and Xibin Zhu.

Finally, my special thanks go to Barbara Hammer for being an excellent supervisor and for creating an incredibly warm working environment.

Contents

1. Introduction **1**

1.1. Motivation 1
1.2. Scientific contributions and structural overview 4
1.3. Publications in the context of this thesis 6

2. Discriminative dimensionality reduction **9**

2.1. Motivation 9
2.1.1. Scientific contributions and structure of the chapter 11
2.2. Kernel t-SNE 12
2.2.1. T-distributed stochastic neighbor embedding (t-SNE) 13
2.2.2. Assessing the quality of dimensionality reduction mappings . . 13
2.2.3. Parametric extension of dimensionality reduction 14
2.2.4. Illustration 15
2.3. Definition of the Fisher metric 17
2.3.1. Metrics 17
2.3.2. Fisher metric as a special case of the Riemannian metric 17
2.3.3. Approximation of the shortest paths 20
2.3.4. Example 21
2.4. Discriminative dimensionality reduction for classification tasks 22
2.4.1. Approximation of the probabilities 23
2.4.2. Example 24
2.5. Discriminative dimensionality reduction in kernel space 24
2.5.1. Kernelization 25
2.5.2. Experiments 26
2.5.3. Conclusion 29
2.6. Discriminative dimensionality reduction for regression tasks 30
2.6.1. Gaussian Processes for regression 31
2.6.2. Estimating the Fisher matrix based on a Gaussian Process 32
2.6.3. Justification for discriminative DR 33
2.6.4. Experiments 34
2.6.5. Conclusion 38
2.7. Discussion 39

3. Visualization of functions in high-dimensional spaces **41**
3.1. Motivation . . . 41
3.1.1. Scientific contributions and structure of the chapter . . . 45
3.2. Dimensionality reduction techniques . . . 45
3.3. Inverse dimensionality reduction . . . 47
3.4. General framework . . . 48
3.4.1. Naive approach . . . 49
3.4.2. Main procedure . . . 51
3.4.3. Evaluation . . . 52
3.5. Experiments with classification functions . . . 53
3.6. Experiments with regression functions . . . 68
3.7. Discussion . . . 74

4. Interpretation of data mappings **75**
4.1. Motivation . . . 75
4.1.1. Scientific contributions and structure of the chapter . . . 76
4.2. Estimating interpretable components for nonlinear DR . . . 77
4.2.1. Neighborhood Retrieval Optimizer . . . 79
4.2.2. Feature selection for DR . . . 80
4.2.3. Relevance learning for DR . . . 81
4.2.4. Metric learning for DR . . . 81
4.2.5. Experiments . . . 83
4.3. Valid interpretation of feature relevance for linear data mappings . . . 90
4.3.1. Definition and measure of feature relevance . . . 92
4.3.2. Linear bounds . . . 97
4.3.3. Metric learning as linear data transformation . . . 101
4.3.4. Experiments for linear regression . . . 102
4.3.5. Experiments for metric learning . . . 105
4.4. Discussion . . . 113

5. Dimensionality reduction for transfer learning **115**
5.1. Motivation . . . 115
5.1.1. Scientific contributions and structure of the chapter . . . 116
5.2. Transfer learning without given correspondences . . . 116
5.2.1. Shared linear embedding . . . 117
5.2.2. Shared nonlinear embedding . . . 117
5.3. Experiments . . . 119
5.4. Discussion . . . 121

6. Conclusion **123**

A. Mathematical derivations **127**
A.1. The Fisher information matrix for a discrete auxiliary variable . . . 127
A.2. The Fisher information matrix for a continuous auxiliary variable . . . 128

B. Publications in the context of this thesis **131**

Bibliography **135**

List of Tables

2.1. Average 1-NN classification errors in percent with standard deviations; sum of the negative EVs in relation to the summed absolute values of the EVs. 29
2.2. Prediction errors in different data spaces using the nRMSE over 10 runs. The standard deviation is given in brackets. 36

3.1. Classification accuracies of the three SVMs, each trained on a different label assignment. 59
3.2. Visualization qualities for the regression models, as measured by the Pearson correlation. 72

4.1. Feature ranking induced by the different techniques for set2 and set3. Fisher t-SNE is abbreviated via F t-SNE. 85
4.2. 1-NN errors in various data spaces of the data sets USPS and Adrenal. . 86
4.3. Classification error rates ranging between 0 and 1 for all data sets. If not specified differently, the classification model is GMLVQ. 108

5.1. Mean classification accuracies with a linear SVM for the experiments. . 120

List of Figures

2.1. T-SNE projection of a subset of the usps data set (left) and its out of sample extension computed with kernel t-SNE (right). 15
2.2. Evaluation of a kernel t-SNE and parametric t-SNE projection of the usps data set. 16
2.3. Two-dimensional toy data (left) and three-dimensional ball data (right). 17
2.4. Estimation of $p(c = 1|\mathbf{x})$ (left) and $p(c = 2|\mathbf{x})$ (right) for the toy data set using the Parzen window estimator. 19
2.5. Parzen window estimation of $\max(p(c = 1|\mathbf{x}), p(c = 2|\mathbf{x}))$. The right plot shows the same figure viewed from above together with the eigenvectors of the Fisher matrices scaled with their according eigenvalues and the straight path approximation together with a minimal path. . . . 21
2.6. Visualization of the ball data set with t-SNE (left) and out of sample extension with kernel t-SNE (right). The Fisher metric is utilized in the top row, the Euclidean metric in the bottom row. 25
2.7. Unsupervised t-SNE projections in rows one and three of the data sets Aural Sonar, Patrol, Protein, Voting, Java Programs and Sonatas. Rows two and four contain the according supervised Fisher t-SNE projection. 28
2.8. The three-dimensional sphere data set (left). Evaluation of the preservation of neighborhoods for the two projections of this data set with t-SNE and Fisher t-SNE (right). Area under the curve value is shown in the legend. 35
2.9. Two projections of the sphere data set are shown: the unsupervised projection (left) and the supervised projection (right). 36
2.10. Two embeddings showing the housing data set: unsupervised t-SNE embedding (left) and discriminative Fisher t-SNE embedding (right). . . 37
2.11. Two embeddings depicting the diabetes data set: t-SNE embedding (left) and Fisher t-SNE embedding (right). 38
2.12. A Fisher t-SNE projection of the diabetes data set with different colorings according to the target variable (top), feature 1 (bottom left) and feature 3 (bottom right). 39

3.1. Principled procedure how to visualize a given data set and a trained classifier. The example displays a SVM trained in 3D. 50
3.2. Illustration of our proposed approach to visualize a regression model (in this case a Decision Tree). 52

3.3. Toy data set 1 (left). Note the potential outlier point of class 1 in the upper right part of the data set. The right image shows toy data set 2. . 55
3.4. Visualization of two different SVMs trained on data set 1 with PCA. . . 56
3.5. Visualization of two different SVMs trained on data set 1 with SOM. . . 56
3.6. Visualization of data set 2 with PCA (left) and the according inverse projected samples (right). 57
3.7. Visualization of data set 2 with SOM (left) and the according SOM map (right). 57
3.8. Visualization of data set 2 with t-SNE (left) and the according inverse projected samples (right). 58
3.9. Visualization of data set 2 with Fisher SOM (left) and the according inverse projected samples (right). 58
3.10. Visualization of SVMs trained on the 10-dimensional data set with the labels l_i^1 (left), l_i^2 (middle) and l_i^3 (right). 60
3.11. Empirical comparison of different DR techniques with and without supervision. 61
3.12. Visualization of the phoneme data set with the methods t-SNE, Fisher t-SNE, Isomap, Fisher Isomap, MVU and Fisher MVU. 62
3.13. Visualization of the phoneme data set with the methods SOM, Fisher SOM, GTM and Fisher GTM. 63
3.14. The three-dimensional data set 3 shown from two different perspectives. 64
3.15. Projection of data set 3 with t-SNE (left). Data set 3 together with the prototypes of the trained RSLVQ model (right). 65
3.16. Two visualization of the same RSLVQ classification model: The projection methods Fisher t-SNE based on the original labeling (left) and Fisher t-SNE based on the labels from the trained classifier (right) are applied. 66
3.17. Two Fisher SOM visualization of the same Classification Tree classier. The left visualization is based on labeling provided by the classifier and the right on the original labels. 67
3.18. Fisher SOM visualization of the Classification Tree where the data points are labeled according to the classifier. The same projections as shown in Fig. 3.17 are utilized. 68
3.19. Visualization of the RSLVQ classifier with Fisher SOM (left) and Fisher t-SNE (right). Both projections are based on the Fisher information as defined by the labels of the classifier (but the original labeling is shown). 69
3.20. Two toy data sets: data set1 (left) and data set2 (right). 70
3.21. Four different visualizations of the same regression model. These are based on (from top left to bottom right): GTM, Fisher MDS, Fisher GTM, Fisher t-SNE. 71

3.22. A Fisher GTM induced visualization of the SVR (left) and Decision Tree (right) with data set1. The continuous surfaces depict the prediction of the regression models. 72
3.23. A Fisher GTM (left) and a Fisher t-SNE (right) visualization of a SVR model trained on the diabetes data set. 73

4.1. Left: Data set1. Right: Relevance profile of the Adrenal data set. Green marks indicate that these 9 dimensions are also the top ones in [17]. . . 81
4.2. Artificial multimodal data (left), projection by LDA (middle), projection by Fisher t-SNE (right) 83
4.3. Relevance determination for data set1 using λ_{NeRV} (left), λ_{forward} (middle) and $\lambda_{\text{backward}}$ (right). 84
4.4. Relevances Ω_{ii} obtained by the proposed method for the LDA projection (left) in dependency of the choice k of the cost function $E_k(\Omega)$, for the projection by Fisher t-SNE (right) 87
4.5. T-SNE projection of the diabetes data set (left), quality for the t-SNE mapping for the standard Euclidean metric versus the transformed data with relevance matrix for neighborhood range 10 (middle) and 50 (right). 87
4.6. Projection of the adrenal data using t-SNE (left) and Fisher t-SNE (middle). The latter can be used to learn the relevant factors for this discriminative visual display (right). 88
4.7. Projection of the linearly transformed adrenal data using t-SNE (left). Projection onto the two main eigenvectors of the learned linear transformation (right). 88
4.8. Lower and upper bounds of feature relevance given by Alg. 1 and Alg. 2 for the diabetes dataset. c is the mean square error of a linear regression. 96
4.9. Lower and upper bounds of feature relevance for the diabetes dataset. Results are based on Alg. 1 and Alg. 2 (left) and on the linear programming method (right). 102
4.10. Lower and upper bounds of feature relevance for a toy dataset. The left figure shows the results of the generic approach, the right one for the LP method. 103
4.11. Lower and upper bounds of feature relevance for a Boston Housing dataset. The left figure shows the results of the generic approach, the right one for the LP method. 104
4.12. Lower and upper bounds of feature relevance for a Poland Electricity Consumption dataset. The left figure shows the results of the generic approach, the right one for the LP method. 104
4.13. Lower and upper bounds of feature relevance for a Santa Fe Laser dataset. The left figure shows the results of the generic approach, the right one for the LP method. 105

4.14. Two relevant features of the xor data set (left). Average classification error rates of GMLVQ with regularized metrics for the xor data set (right). 105
4.15. Spectra of the data sets wine (left) and tecator (right). 106
4.16. Results of our proposed approach for the xor data set. The first row shows the original linear mappings, the second row depicts the resulting upper (in black) and lower bounds (in white). 107
4.17. Employing the xor data set, estimates of the coefficients for different values of the L1 norm (x-axis) are shown. The methods lasso (left), elastic net (middle) and ridge regression (right) are utilized. 108
4.18. Average classification error rates of GMLVQ with regularized metrics for the wine (left) and tecator (right) data set, both for set S. 109
4.19. Results of our proposed approach for the wine data set. The first row shows the original linear mapping, while the second row depicts the resulting upper relevance bounds. The lower bounds are all zero, in this case. 110
4.20. Results of our proposed approach for the tecator data set. First two columns: The first row shows the original linear mapping, the second row depicts the resulting upper and lower relevance bounds. The last column shows the summed lower and upper bounds. 110
4.21. Absolute values of the original mapping (top row) together with the absolute value of the averaged regularized mappings (bottom row). . . 111
4.22. Average classification error rates of GMLVQ (left) and LMNN (right) with regularized metrics for the adrenal data set. 111
4.23. Relevance bounds for a GMLVQ model (top) and a LMNN model (bottom), both trained on the adrenal data set. 112

5.1. Examples of images from the Coil data set: the top row contains images from the source data while the bottom row shows the according target images. 118
5.2. The linear alignment of source and target data for the Iris data set is shown left. Both data sets are shown individually with their according labeling middle (source) and right (target after transfer). 119
5.3. A linear (top three) and nonlinear (bottom three) alignment of source and target data for the Coil data set is shown left. Both data sets are shown individually with their according coloring middle (source) and right (target). 120

Chapter 1.

Introduction

Chapter overview *This chapter provides an informal introduction into the research topics investigated in this thesis. Consequently the structure and the major contributions are presented.*

1.1. Motivation

Due to developments in sensor technology and storing capacity, the availability of digital data is growing vastly [65], thereby getting bigger as concerns their size, complexity, and dimensionality. Accordingly, big data has been proclaimed as one of today's major challenges in the digital society [79, 32]. Computational intelligence and machine learning techniques offer a fundamental approach to tackle a few of the involved problems [179, 74, 61]. In almost all settings, however, data analysis is not fully automated, but the human has to decide on the suitability of the used techniques, often in an interactive way. Hence, it is vital to establish an intuitive access to digital data and the possible outcomes of algorithmic steps for the practitioner. Since decades, visual data inspection offers one premier interface in this setting, since it relies on one of the most powerful human senses as well as the astonishing cognitive capabilities of instantaneous visual grouping and feature detection [144, 171].

There exists a variety of classical machine learning tools which aim for intuitive visual data inspection such as the self-organizing map (SOM) [83], generative topographic map (GTM) [19], Autoencoder [66], independent component analysis (ICA) [71], or the Oja and Sanger learning rules [113, 139]. At their core, these methods rely on a low-dimensional representation of the data. In the research branch of visual analytics, low-dimensional embeddings also constitute one popular mode for data analysis, often realized by techniques such as scatter plots, tour methods or (mostly) linear projections. Scatter plots offer one of the most prominent techniques to directly inspect data visually: here, data are displayed in two or three dimensions such that their neighborhood relationships can directly be inspected. Phenomena such as clusters, complex grouping, or outliers can easily be observed. For example, scatter plots constitute an essential part in the pipeline to identify cell populations through gating in flow cytometry [114]. Another example is the interactive analysis of complex metage-

nomic data [91]. For higher dimensionality, scatter matrices, parallel coordinates, tour methods, glyphs and similar approaches have been proposed [144]. The field of visual analytics employs such techniques, often together with suitable interactive settings, to visually analyze data [78]. However, their applicability is limited in case of high-dimensional data, since not all information available in the different dimensions and their correlation can easily be integrated based on these simple methods.

In this context, dimensionality reduction plays a major role, referring to the task of mapping high-dimensional vectors to low-dimensional counterparts such that as much structure as possible is preserved. These techniques have a large history of successful applications in various areas including biomedical analysis, network visualization, image analysis, text mining, and so on [77, 116, 137, 135]. The abstract concept to preserve the structure of the original data, however, has led to a plethora of mathematical formalizations and resulting methods [94, 26, 77, 53]. Besides common and popular linear techniques [16], more complex and nonlinear methods have been developed, including manifold learning [135, 150] and neighbor embedding algorithms [159, 163].

One problem of unsupervised dimensionality reduction for data visualization consists in the fact that this setting is inherently ill-posed: unless data are intrinsically low-, i.e. two-dimensional, which is usually not the case for realistic signals, any smooth embedding of the data into the visual plane necessarily sacrifices some of the information present in the signals. This observation has been pointed out in the overview [160]. Fortunately, data inspection and visualization is usually integrated into a scenario with a specific underlying purpose: as an example, in medical data visualization, the medical expert is interested in an exploration of the given data concerning important aspects of a given disease, or a response to a specific treatment. In this setting, there exists a very clear, albeit abstract guideline about which information contained in the data can be abandoned by a dimensionality reduction technology, namely all irrelevant aspects of the data regarding the given disease or response to the treatment, respectively. This identifies a clear objective of what should be visualized by the dimensionality reduction method and what should be discarded, albeit an exact mathematical realization of this idea is difficult. In dimensionality reduction this observation has caused a line of research which is often put under the term of discriminative dimensionality reduction (DiDi): given auxiliary information about the data specifying which aspects of the data are interesting in a user-centered way, visualize only those aspects of the data which are of relevance to this user specified objective. Examples where this principle has been investigated include the work [35, 94, 103, 8, 50, 163, 26, 27].

Despite many advances in this area, the application of **discriminative dimensionality reduction in complex settings** poses technical challenges, which is one focus in this thesis. How can these algorithms be reformulated such that they are applicable to streaming data? These techniques rely on non-parametric projection techniques, thereby directly assigning low-dimensional coordinates to each data point without using an explicit parametric function. Hence, a challenging problem is: How can these

techniques be extend such that they are suitable for subsequently arriving data? An additional crucial problem is the application to complex data: Currently, most DiDi approaches are designed for data represented by vectors, only. For complex data, however, it is often more convenient to define proximity measures on data instances directly, instead of engineering a feature-based representation [51, 34]. Therefore, it is unclear how these techniques can be employed for non-vectorial representations. Another complex setting occurs if auxiliary information is no longer discrete, but continuous. Nonlinear DiDi approaches usually assume the former, which is the case e.g. in classification settings. This poses the question: How can continuous auxiliary data be integrated into successful DiDi techniques, occurring e.g. in regression scenarios? These questions are addressed in chapter 2 of this thesis.

Apart from using DiDi tools for interactive data exploration, a promising alternative way is to increase the interpretability of supervised machine learning models. These are fit to a given data set in order to solve tailored tasks such as classification or regression. During the optimization process, these models gather knowledge which allows them to make qualified decisions, but it is often not possible to access the reasoning behind these decisions. This raises the question of model interpretability. Several possible remedies have been proposed, including relevance learning, feature selection techniques, and sparse model descriptions, for example [117, 162, 138, 141, 67]. However, these methods focus on specific properties of the respective models and, thus, allow to interpret only these aspects. None of them satisfactorily answers the question: How can we **visualize high-dimensional classifiers or regression models**, i.e. the core underlying function of these models? A key question in this context is how to extend dimensionality reduction techniques such that they can also be employed to visualize such functions. We target these questions in chapter 3.

Although nonlinear dimensionality reduction techniques constitute powerful tools to embed high-dimensional data in a low-dimensional space, linear mappings are still often preferred in practical applications [16]. One major reason for this is that linear mappings provide information about the importance of the features for the given projection. However, linear mappings are restricted in their flexibility and, hence, often yield inferior embeddings as compared to nonlinear non-parametric methods. Thus, a central question in this context is: How can we **determine relevant features for nonlinear methods**? Chapter 4 deals with this question.

Linear mappings constitute a prominent element, not only in the context of dimensionality reduction, but in basically all fields of machine learning including regression, classification or metric learning. One of the striking properties of linear models is that they seemingly allow an interpretation of the relevance of input features by inspecting their corresponding weighting; in a few cases, such techniques have led to striking semantic insights of the underlying process [5]. Recent results, however, have shown that the interpretation of linear weights as relevance terms can be extremely misleading in particular for high-dimensional data [149]. Hence, an important question is: How can we **extract a valid relevance profile from linear mappings**? This requires to

distinguish between strictly required features and features which can be replaced by others but do carry relevant information. These aspects are addressed in chapter 4.

A core property of dimensionality reduction is that it preserves the intrinsic structure of the given data while projecting it to a low-dimensional space, thereby removing noise. This characteristic makes it well suited to transfer knowledge from one domain to another, i.e. if the same task should be performed in a different domain, e.g. due to a sensor change. Successful applications in this context exist [20, 120, 143], but they rely either on correspondence information between the different spaces or label information. The particularly interesting task of transfer learning without labels and correspondence information has been barely investigated. Or stated differently: How can we **transfer knowledge in an unsupervised setting without correspondence information**? Chapter 5 of this thesis deals with this question.

1.2. Scientific contributions and structural overview

After having provided a basic motivation and raised relevant questions in the context of discriminative dimensionality reduction and interpretability, we will address the latter by proposing novel algorithms. The following gives a summary of the scientific contributions of this thesis.

Discriminative dimensionality reduction in complex settings For the methodology of discriminative dimensionality reduction based on the Fisher metric, we propose three extensions which enable the application of DiDi techniques in more complex scenarios in chapter 2.

- In section 2.2 we propose a parametric extension for nonlinear dimensionality reduction. Since most modern DR and DiDi techniques are non-parametric, they are restricted to a fixed data set and have to be recomputed if additional data become available. Our contribution allows to process also sequentially arriving data and to project large parts of a data set in linear time.
- After recalling the basic concepts for DiDi with the Fisher metric in sections 2.3 and 2.4, we propose a reformulation of the Fisher metric based DiDi framework in section 2.5. It enables applications to complex data provided only by similarities. This opens the way towards computing discriminative projections of structured data for which a vectorial representation is difficult to obtain. Examples include musical pieces and graphs.
- In section 2.6, we present a novel technique to compute DiDi mappings for real-valued auxiliary information based on the Fisher metric. This approach is based on the idea to augment the computation of Fisher distances by allowing also real-valued auxiliary information. This enables the computation of powerful nonlinear embeddings of a data set with real-valued information emphasizing the important structure.

Visualization of functions in high-dimensional spaces In chapter 3, we propose a framework to visualize a high-dimensional function together with a data set in two dimensions. We apply this scheme to visualize the underlying functions of classification and regression models. At its core, this framework is based on computation of DR and inverse DR projections. This framework is general in the sense that it allows to visualize any classification or regression model and to employ any DR technique, including DiDi methods. In the experiments, we demonstrate that the proposed framework benefits from DiDi methods and that it allows to solve identified user tasks such as: How complex are the decision boundaries of a classifier or the prediction function of regression model in a specific region of the data space? Does overfitting/underfitting behavior appear?

Interpretation of data mappings In chapter 4, two concepts for interpretation in the context of data mappings are proposed.

- In section 4.2, we present a novel technique to estimate interpretable components for nonlinear DR techniques. This methods creates a connection between the information provided by the neighborhood structure in a nonlinear embedding and the role of the original features. We evaluate this approach using data with known ground truth and demonstrate its suitability for real world data from the biomedical domain.
- In section 4.3, we propose a method which estimates valid relevance bounds for a given linear mapping. This provides an estimation of feature relevance even for high-dimensional and correlated features. We demonstrate this approach for linear mappings occurring in regression and metric learning.

Dimensionality reduction for transfer learning In chapter 5, we employ the structure preservation property of DR methods to develop a novel technique for transfer learning. This method is able to transfer knowledge from a source data space to a target data space without requiring label or correspondence information. We demonstrate this approach on artificial data and on a data set consisting of images.

1.3. Publications in the context of this thesis

The following peer-reviewed articles have been published in the context of this thesis: (More detailed references are provided in Appendix B on page 131.)

Journal articles

[J17] A. Schulz, J. Brinkrolf, and B. Hammer. Efficient kernelization of discriminative dimensionality reduction. *Neurocomputing*, 268(C): 34–41, 2017.

[J15b] A. Schulz, A. Gisbrecht, and B. Hammer. Using Discriminative Dimensionality Reduction to Visualize Classifiers. *Neural Processing Letters*, 42(1): 27–54, 2015.

[J15a] A. Gisbrecht, A. Schulz, and B. Hammer. Parametric nonlinear dimensionality reduction using kernel t-SNE. *Neurocomputing*, 147: 71–82, 2015.

Conference and Workshop articles

[C16b] C. Prahm, B. Paaßen, A. Schulz, B. Hammer, and O. Aszmann. Transfer Learning for Rapid Re-calibration of a Myoelectric Prosthesis after Electrode Shift In *ICNR 2016*, pages 153–157, 2016.

[C16a] A. Schulz, and B. Hammer. Discriminative dimensionality reduction in kernel space. In *ESANN 2016*, pages 123–128, 2016.

[C15f] A. Schulz, B. Mokbel, M. Biehl, and B. Hammer. Inferring feature relevances from metric learning. In *SSCI CIDM 2015*, pages 41–48, 2015.

[C15e] [1] B. Mokbel, and A. Schulz. Towards dimensionality reduction for smart home sensor data. In *NC*^{2} *2015*, pages 41–48, 2015.

[C15d] [2] A. Schulz, and B. Hammer. Visualization of regression models using discriminative dimensionality reduction. In *CAIP 2015*, pages 437–449, 2015.

[C15c] A. Schulz, and B. Hammer. Discriminative dimensionality reduction for regression problems using the Fisher metric. In *IJCNN 2015*, pages 1–8, 2015.

[C15b] P. Bloebaum, A. Schulz, and B. Hammer. Unsupervised dimensionality reduction for transfer learning. In *ESANN 2015*, pages 507–512, 2015.

[C15a] A. Schulz, and B. Hammer. Metric learning in dimensionality reduction. In *ICPRAM 2015*, pages 232–239, 2015.

[C14d] B. Frenay, D. Hofmann, A. Schulz, M. Biehl, and B. Hammer. Valid interpretation of feature relevance for linear data mappings. In *SSCI CIDM 2014*, pages 149–156, 2014.

[C14c] A. Gisbrecht, A. Schulz, and B. Hammer. Discriminative dimensionality reduction for the visualization of classifiers. In *ICPRAM 2013 Selected Papers*, pages 39–56, 2014.

[C14b] P. Bloebaum, and A. Schulz. Transfer learning without given correspondences. In *NC*^{2} *2014*, pages 42–51, 2014.

[C14a] A. Schulz, A. Gisbrecht, and B. Hammer. Relevance learning for dimensionality reduction. In *ESANN 2014*, pages 165–170, 2014.

[1] Winner of the *Best Poster* award at NC^2 2015.
[2] Winner of the *Best Poster* award at CAIP 2015.

[C13c] A. Schulz, A. Gisbrecht, and B. Hammer. Classifier inspection based on different discriminative dimensionality reductions. In *NC*^{2} *2013*, pages 77–86, 2013.

[C13b] A. Schulz, A. Gisbrecht, and B. Hammer. Using nonlinear dimensionality reduction to visualize classifiers. In *IWANN 2013*, pages 59–68, 2013.

[C13a] [3] B. Hammer, A. Gisbrecht, and A. Schulz. Applications of discriminative dimensionality reduction. In *ICPRAM 2013*, pages 33–41, 2013.

[C12b] B. Hammer, A. Gisbrecht, and A. Schulz. How to visualize large data sets? In *WSOM 2012*, pages 1–12, 2012.

[C12a] A. Schulz, A. Gisbrecht, K. Bunte, and B. Hammer. How to visualize a classifier? In *NC*^{2} *2012*, pages 73–83, 2012.

Funding acknowledgments

The following institutions and associated grants are gratefully acknowledged:

- The *Cluster of Excellence Cognitive Interaction Technology* (CITEC) (EXC 277), funded by the *German Science Foundation* (DFG).
- The project *Discriminative Dimensionality Reduction* (DiDi) funded by the *German Science Foundation* (DFG) under grant number HA 2719/7-1.
- A travel scholarship by the *German Academic Exchange Service* (DAAD).

[3] Winner of the *Best Paper* award at ICPRAM 2013.

Chapter 2.

Discriminative dimensionality reduction

Chapter overview *This chapter presents the general idea of computing discriminative dimensionality reduction mappings with the help of distances computed on a Riemannian manifold. This approach employs concepts from the information geometry literature to shape the metric of the data space such that it emphasizes directions important for the auxiliary information. We reformulate this framework such that it is applicable to proximity data. We extend it for the case of real-valued auxiliary information, and we propose a methodology to compute an out-of-sample extension.*

Parts of this chapter are based on:

[J15b] A. Schulz, A. Gisbrecht, and B. Hammer. Using Discriminative Dimensionality Reduction to Visualize Classifiers. *Neural Processing Letters*, 42(1): 27–54, 2015.

[J15a] A. Gisbrecht, A. Schulz, and B. Hammer. Parametric nonlinear dimensionality reduction using kernel t-SNE. *Neurocomputing*, 147: 71–82., 2015.

[C16a] A. Schulz, and B. Hammer. Discriminative dimensionality reduction in kernel space. In *ESANN 2016*, pages 123–128, 2016.

[C15c] A. Schulz, and B. Hammer. Discriminative dimensionality reduction for regression problems using the Fisher metric. In *IJCNN 2015*, pages 1–8, 2015.

2.1. Motivation

In the era of big data, efficient tools are required to make many data instances intuitively accessible to the user at the same time. Dimensionality reduction methods play a major role in this context. Dimensionality reduction (DR) refers to the problem of mapping high-dimensional data points to few dimensions such that as much structure as possible is preserved. Starting with classical methods such as principal component analysis (PCA), multidimensional scaling (MDS), or the self-organizing map (SOM), it offers a visual data analysis tool which has been successfully used in diverse areas such as social sciences or bioinformatics since decades [83, 175]. In the last years, a huge variety of diverse alternative DR techniques has emerged, including popular algorithms such as the generative topographic map (GTM), locally linear embedding (LLE), Isomap, Isotop, maximum variance unfolding (MVU), Laplacian Eigenmaps, neighborhood retrieval visualizer (NeRV), maximum entropy unfolding (MEU), t-distributed stochastic neighbor embedding (t-SNE), and many others

[135, 150, 173, 9, 159, 163], see e.g. [160, 163, 94, 26] for overviews. These methods belong to nonlinear DR techniques, enabling the accurate visualization of data which lie on curved manifolds or which incorporate clusters of complex shape, as is often the case for real-life examples, thus opening the way towards a visual inspection of nonlinear phenomena in the given data.

Many classical techniques such as PCA and SOM belong to the class of parametric methods. These techniques specify an explicit parametric mapping. Most recent dimensionality reduction methods, on the other side, belong to the class of non-parametric techniques: they provide a mapping of the given data points only, without specifying an explicit parametric function. This choice has the benefit that it equips the techniques with a high degree of flexibility: no constraints have to be met due to a predefined form of the mapping, rather, depending on the situation at hand, arbitrary restructuring, tearing, or nonlinear transformation of data is possible. Hence, these techniques carry the promise to arrive at a very flexible visualization of data such that also subtle nonlinear structures can be spotted.

Although nonlinear DR methods constitute powerful tools in the context of data exploration, the general goal of structure preservation is ill-posed: If the intrinsic data dimensionality is larger then the projection space (which is usually 2 for the purpose of visualization), the methods have to deal with information loss. Thereby, the decision which information to preserve can depend on several factors such as the mathematical formalization or even random aspects of the method. One possible remedy is to specify auxiliary information indicating which changes of the data are important and which can be neglected. Thereby, this information reduces the relevant dimensionality of the data and enables a meaningful visualization. Class labels in a supervised setting can be considered as one example for such auxiliary data. Techniques employing auxiliary information for data visualization are called *discriminative* (or sometimes *supervised*) dimensionality reduction methods. Techniques for discriminative dimensionality reduction (DiDi) aim to preserve that structure of the data which is particularly relevant for the specified auxiliary data.

A variety of different classical discriminative dimensionality reduction techniques has been proposed, such as the Fisher's linear discriminant analysis (LDA), partial least squares regression (PLS), informed projections [35], global linear transformations of the metric [57, 27], or kernelization of such approaches [103, 8]. More modern discriminative DR techniques include unsupervised DR based on the Fisher metric [123], multiple relational embedding (MRE) [107], colored maximum variance unfolding (CMVU or MUHSIC) [148], supervised Isomap (s-isomap) [52], parametric embedding (PE) [72], and neighborhood component analysis (NCA) [57].

The recent paper [163] has conducted a study in order to compare these modern methods for discriminative dimensionality reduction. They come to the result that unsupervised neighbor embedding methods together with the discriminative Fisher metric obtain a superior performance. This approach is based on the general idea to locally modify the metric [123, 54]. A Riemannian manifold is defined which takes

into account auxiliary information of the data and which measures the effect of data dimensions in the feature space on this auxiliary information. Additionally, such a formulation of DiDi is particularly elegant, since it can be integrated in any unsupervised DR technique which requires distances, only. This can be done by replacing the commonly used Euclidean distance by the Fisher distances, as has been done for the SOM, NeRV and t-SNE in [123, 163] [J15a]. In this thesis, we will employ discriminative dimensionality reduction based on the Fisher metric since (i) it outperforms other DiDi techniques as concerns their capability to embed the data while focusing on the specified auxiliary information, (ii) it constitutes a general concept to incorporate auxiliary information into any distances based DR technique and such allows to find a suitable combination for the task at hand, (iii) it is based on the formal mathematical framework of information geometry, combining the concepts of Riemannian geometry together with information theory and, such, providing a well founded mathematical formulation instead of heuristic approaches.

However, the application of DiDi methods in complex domains requires to solve open questions, such as the application to data given only by similarities or dissimilarities, the utilization of auxiliary information in form of a continuous variable and the computation of an out-of-sample extension mapping. The scientific contributions to these topics are detailed in the next section.

2.1.1. Scientific contributions and structure of the chapter

In this chapter, we present three core contributions to the methodology of discriminative dimensionality reduction.

Kernel t-SNE In section 2.2, we present a general approach to compute an out-of-sample extension for an arbitrary non-parametric DR technique. We demonstrate its performance for an unsupervised mapping in section 2.2 and for discriminative projections in section 2.4.

DiDi in kernel space In section 2.5, we reformulate the Fisher metric framework, thus, allowing to compute Fisher distances from similarity based data, only. This enables us to compute DiDi projections based on proximity data only, i.e. without requiring a vectorial representation.

DiDi for regression In section 2.6, we propose a DiDi framework based on the concept of the Fisher metric for continuous auxiliary data. This technique relies on a Gaussian process to estimate the conditional density.

This chapter is organized as follows: We start by proposing the method kernel t-SNE in section 2.2 which allows to compute an out-of-sample extension for a given non-parametric DR method. For this purpose, we exemplarily introduce the method t-SNE, recall the state of the art method to evaluate the quality of DR mappings, introduce the kernel t-SNE methodology and demonstrate it on a benchmark scenario.

Consecutively, we introduce the central concept of the Fisher metric in section 2.3. We will use this framework later as a core step to compute discriminative distances for DiDi projections. This section briefly recaps the information geometrical concepts required to compute distances based on the Fisher metric and presents the common approximation schemes to path integrals in this context.

Section 2.4 builds on section 2.3 and recalls a complete scheme to compute DiDi mappings using the Fisher metric. This requires in particular to estimate the conditional probability density from the data.

Section 2.5 presents the reformulation of DiDi based on Fisher distances, which allows to apply these techniques to data given by similarities, only. The framework can be reformulated exactly in terms of inner products.

Finally, section 2.6 presents a new scheme to compute DiDi based on the Fisher metric for continuous auxiliary information. This approach utilizes a Gaussian process to estimate the conditional density.

2.2. Kernel t-SNE

In the following, we assume to have N vectorial data $\mathbf{x}_i \in \mathcal{X} = \mathbb{R}^D$ in a D-dimensional vector space which is potentially high-dimensional. For every point $\mathbf{x}_i$, DR methods determine coefficients $\boldsymbol{\xi}_i \in \Xi = \mathbb{R}^d$ with $D > d = 2$, usually. In cases of other assumptions, we specify them in the beginning of the corresponding section.

While parametric mappings provide an explicit functional form, non-parametric mappings such as t-SNE, MVU, or Isomap have in common that no direct out-of-sample extension is available. However, non-parametric methods seem to be particularly successful in embedding data sets truthfully. This is the result of the evaluation in the recent review [125]. These methods often take a simple cost function based approach: the N data points $\mathbf{x}_i \in \mathcal{X} = \mathbb{R}^D$ constitute the starting point; for every point $\mathbf{x}_i$, projections $\boldsymbol{\xi}_i$ are determined such that the characteristics of these points mimic the characteristics of their high-dimensional counterpart. Thereby, the characteristics differ among the various method, they are e.g. pairwise distances of data points, the overall data variation, locally linear relations of data points, or local probabilities induced by the pairwise distances, to name a few examples [26].

However, one major challenge of non-parametric DR methods is that they do not provide a direct out-of-sample extension and, hence, are not directly applicable to streaming data, for instance.

To solve this problem, we present the new method *kernel t-SNE* which provides a parametric mapping that allows to compute out-of-sample extensions directly. This method is applicable to any non-parametric DR technique because it relies on a kernel projection, trained retrospectively after the original projection. In particular, it is also applicable to DiDi methods, which will be demonstrated in section 2.4.2. It also allows to project large data sets by applying the core method to a subset of the data and projecting the remained with kernel t-SNE in linear time.

The rest of this section is structured as follows: we exemplarily introduce the method t-SNE as one of the most popular non-parametric techniques in the following subsection 2.2.1, we address the issue of quality assessment in dimensionality reduction in section 2.2.2 and present our contribution kernel t-SNE in section 2.2.3. Finally, section 2.2.4 provides a short evaluation of the proposed technique.

2.2.1. T-distributed stochastic neighbor embedding (t-SNE)

The recent review [125] has compared many prominent non-parametric approaches and the popular method t-distributed stochastic neighbor embedding (t-SNE) [159] has performed very competitive. In the following, we will explain t-SNE in more detail since we will use it in our experiments. However, any DR method which works on distances can be augmented by our proposed kernel approach.

In t-SNE, probabilities in the original space are defined as $p_{ij} = (p_{(i|j)} + p_{(j|i)})/(2n)$, where

$$p_{j|i} = \frac{\exp(-0.5\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma_i^2)}{\sum_{k,k\neq i}\exp(-0.5\|\mathbf{x}_i - \mathbf{x}_k\|^2/\sigma_i^2)}$$

depends on the pairwise distances of points; the parameter σ_i is replaced by an other parameter, the effective number of neighbors, frequently termed perplexity: σ_i is adapted such that each data point has this priorly specified number of neighbors.

In the projection space, probabilities are induced by the Student t-distribution

$$q_{ij} = \frac{(1 + \|\boldsymbol{\xi}_i - \boldsymbol{\xi}_j\|^2)^{-1}}{\sum_k \sum_{l,l\neq k}(1 + \|\boldsymbol{\xi}_k - \boldsymbol{\xi}_l\|^2)^{-1}}$$

to avoid the crowding problem by using a long tail distribution. The goal is to find projections $\boldsymbol{\xi}_i$ such that the difference between p_{ij} and q_{ij} becomes small as measured by the Kullback-Leibler divergence

$$Q^{\mathrm{t-SNE}}(\mathbf{X}, \Xi) = \sum_i \sum_{j\neq i} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}. \tag{2.1}$$

t-SNE relies on optimization of (2.1) using a gradient based technique.

2.2.2. Assessing the quality of dimensionality reduction mappings

A popular tool to evaluate the quality of a given data projections was proposed by Lee and colleagues [92, 93]. The key idea is to measure the preservations of neighbors instead of distances. More formally, for each data point $\mathbf{x}^i$ and its low-dimensional counterpart $\boldsymbol{\xi}^i$, the functions $\mathsf{N}_k(\mathbf{x}^i)$ and $\mathsf{N}_k(\boldsymbol{\xi}^i)$ measure the indices of their k nearest neighbors. Then the average preservation of the k nearest neighbors can be written as

$$Q_k^{\mathrm{nx}}(\mathbf{X}, \Xi) = \frac{1}{Nk}\sum_{i=1}^{N}\left|\mathsf{N}_k(\mathbf{x}^i) \cap \mathsf{N}_k(\boldsymbol{\xi}^i)\right|, \tag{2.2}$$

where we sometimes refer to this quantity simply as *quality*. This can also be formalized in terms of the co-ranking framework, thereby summarizing alternative evaluation measures [92]. Since it is usually not known which neighborhood size k is most important, $Q_k^{\text{nx}}(\mathbf{X}, \Xi)$ is typically evaluated for each possible $k \in \{1, 2, \ldots, N-1\}$.

For a random projection, the average value of Q_k^{nx} is $k/(N-1)$, which corresponds to the diagonal in a quality plot and is usually treated as a baseline. It is possible to remove this baseline from Q_k^{nx} by subtracting it and rescaling the resulting term such that it is again between 0 and 1:

$$Q_k^{\text{nx\_n}}(\mathbf{X}, \Xi) = \frac{(N-1)Q_k^{\text{nx}}(\mathbf{X}, \Xi) - k}{N-1-k}. \tag{2.3}$$

This is referred to as R_{NX} in the literature [93] and is often plotted using a logarithmic scale on the axis depicting the neighborhood size k. This emphasizes that local neighborhoods are usually treated as the most important ones.

Using this scheme, it is additionally possible to obtain a scalar quality value by calculating the area under the $Q_k^{\text{nx\_n}}$ curve plotted with a logarithmic scale. This makes use of a strong emphasis on local neighborhoods.

2.2.3. Parametric extension of dimensionality reduction

In the paper [J15a], we propose a general way how to extend the prescriptions of non-parametric DR methods to a parametric form by means of an interpolation by Gaussian kernels. We specify a functional form π_{pm} of the mapping as follows:

$$\mathbf{x} \mapsto \pi_{\text{pm}}(\mathbf{x}) = \frac{\sum_j \boldsymbol{\alpha}_j k_j(\mathbf{x}, \mathbf{x}_j)}{\sum_l k_l(\mathbf{x}, \mathbf{x}_l)} \tag{2.4}$$

where $\boldsymbol{\alpha}_j \in \Xi$ are parameters corresponding to points in the projection space and the data $\mathbf{x}_j$ are taken as a fixed sample, usually j runs over a small subset $\mathcal{X}'$ sampled from the data $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$. k is the Gaussian kernel parameterized by the bandwidth $\sigma_j^{\mathbf{x}}$:

$$k_j(\mathbf{x}, \mathbf{x}_j) = \exp(-0.5\|\mathbf{x} - \mathbf{x}_j\|^2/(\sigma_j^{\mathbf{x}})^2) \tag{2.5}$$

The idea is to determine the parameters of this mapping such that the data $\mathbf{x}_i$ and their projections $\boldsymbol{\xi}_i$ obtained with the considered projection technique are matched as far as possible. Note that the mapping has a generalized linear form such that training can be done in a particularly simple way provided a set of samples $\mathbf{x}_i$ and $\boldsymbol{\xi}_i$ is available. The parameters $\boldsymbol{\alpha}_j$ can be analytically determined as the least squares solution of the mapping: Assume $\mathbf{A}$ is the matrix of parameters $\boldsymbol{\alpha}_j$, $\mathbf{K}$ is the normalized Gram matrix with entries

$$(\mathbf{K})_{i,j} = k_j(\mathbf{x}_i, \mathbf{x}_j) / \sum_l k_l(\mathbf{x}_i, \mathbf{x}_l) \tag{2.6}$$

and Ξ denotes the matrix of projections $\boldsymbol{\xi}_i$. Then, a minimum of the least squares error

$$\sum_i \|\boldsymbol{\xi}_i - \pi_{\text{pm}}(\mathbf{x}_i)\|^2 \tag{2.7}$$

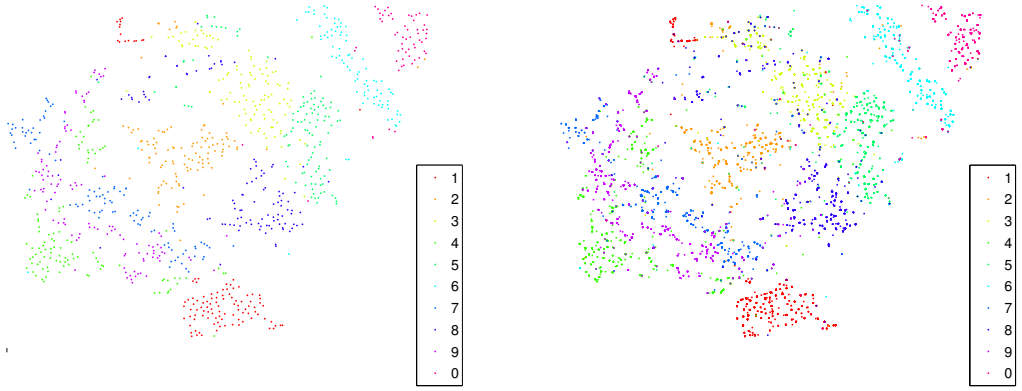


Figure 2.1.: T-SNE projection of a subset of the usps data set (left) and its out of sample extension computed with kernel t-SNE (right).

with respect to the parameters $\boldsymbol{\alpha}_j$ has the form

$$\mathbf{A} = \boldsymbol{\Xi} \cdot \mathbf{K}^{-1} \tag{2.8}$$

where $\mathbf{K}^{-1}$ refers to the pseudo-inverse of $\mathbf{K}$.

The bandwidth $\sigma_i^{\mathbf{x}}$ of the mapping constitutes a critical parameter since it determines the smoothness and flexibility of the resulting kernel mapping. We use a principled approach to determine this parameter as follows: $\sigma_i^{\mathbf{x}}$ is chosen as a multiple of the distance of $\mathbf{x}_i$ from its closest neighbor in $\mathcal{X}'$, where the scaling factor is typically taken as a small positive value. We determine this factor automatically as the smallest value in such a way that all entries of $\mathbf{K}$ are within the range of representable numbers (respectively a predefined interval). This technique allows us to extend any given non-parametric mapping to an explicit parametric form.

2.2.4. Illustration

In this section, we briefly demonstrate the performance of kernel t-SNE using one example. A more thorough evaluation has been performed in the paper [J15a] and can be looked up there.

For this purpose, we utilize the usps data set [45]. It consists of images of the handwritten digits 0 to 9, where each image is encoded with 16×16 gray scale pixels. The data set contains 1,100 images of each class, resulting in 11,000 images.

We preprocess the data set by projecting it to 30 dimensions with PCA and apply t-SNE on a subset of size 1,100 to obtain the training set for kernel t-SNE. Then we apply kernel t-SNE to compute the out-of-sample extension for the remaining data points. Both projections are displayed in Figure 2.1, where the t-SNE projection is shown on the left side and the out-of-sample extension on the right.

We evaluate the quality of the out-of-sample extension obtained with kernel t-SNE using the quantity $Q_k^{\text{nx-n}}$ detailed in section 2.2.2. We additionally employ a subsamling strategy in order to save computational time and in oder to be able to compare

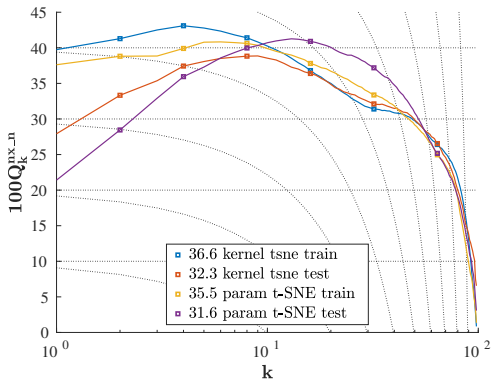


Figure 2.2.: Evaluation of a kernel t-SNE and parametric t-SNE projection of the usps data set.

both results despite their different sample sizes (see [J15a] for more details). The resulting quality values for each neighborhood k are depicted in Figure 2.2. Furthermore, this Figure also shows the quality values for an out-of-sample projection obtained with parametric t-SNE [156]. From this evaluation, two results can be concluded: the embedding quality of the training set for kernel t-SNE is higher then the quality of the training set for parametric t-SNE for local neighborhoods. This is plausible because, for kernel t-SNE, we utilize the standard non-parametric formulation of t-SNE to compute an embedding of the training set. For parametric t-SNE, however, the parametrized function is utilized to obtain a projection of the training set, hence, potentially restricting the mapping. A second observation is that kernel t-SNE obtains higher quality values for small neighborhoods while parametric t-SNE obtains higher values for large neighborhoods. A common concept in evaluating the quality of dimensionality reduction mappings, however, is that the preservation of local neighborhoods are usually considered to be more important. One consequence of this concept, for instance, is that the quality evaluation methodology based on the coranking framework [93] puts an emphasis on local neighbors by using a logarithmic scaling of the x axis.

A more extensive evaluation, including more comparisons to parametric t-SNE, can be found in [J15a]. Here it is further demonstrated that, since the deep architecture used for parametric t-SNE is a powerful model with many parameters, it requires many training instances to learn the required mapping and fails if these are not available. In this context, the parametrization of kernel t-SNE seems to be a good compromise between flexibility of the mapping and complexity of the function.

In the following, we can employ t-SNE to project a small part of the data set and utilize kernel t-SNE to project the remainder. This strategy is particularly useful for methods, which have a high computational cost, as e.g. for DiDi methods such as Fisher t-SNE. In order introduce the latter, we require the notion of the Fisher metric which is illustrated in the following.

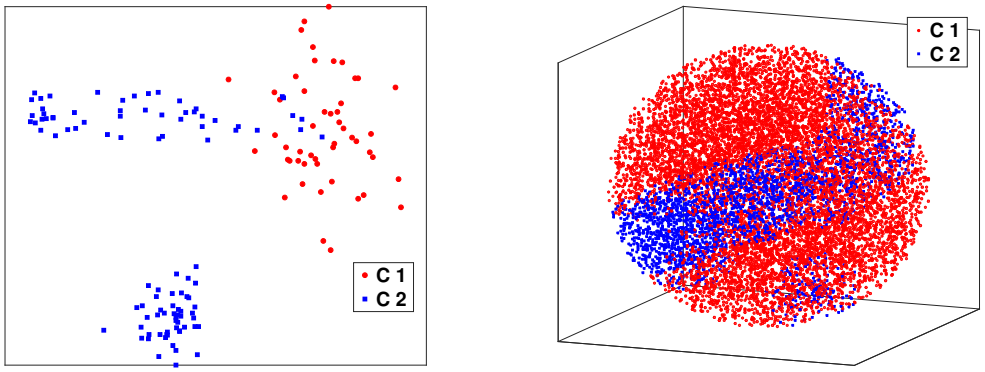


Figure 2.3.: Two-dimensional toy data (left) and three-dimensional ball data (right).

2.3. Definition of the Fisher metric

This section provides some mathematical background on the concept of Fisher metrics which we employ to compute Fisher distances. It also details common approximation schemes for path integrals required for distance computations and illustrates the major steps for computing Fisher distances using an example. We will employ these concepts as one core step to compute discriminative dimensionality reduction mappings.

In order to discuss Fisher metrics, we will first recap the basic concepts of a metric.

2.3.1. Metrics

A *pseudometric* d is a distance function defined on a set $\mathcal{X}$ with $d : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}^+$ and it satisfies the following four properties:

$$
\begin{aligned}
&d(\mathbf{x}_i, \mathbf{x}_j) \geq 0 && \text{non-negativity} \\
&d(\mathbf{x}_i, \mathbf{x}_i) = 0 && \\
&d(\mathbf{x}_i, \mathbf{x}_j) = d(\mathbf{x}_j, \mathbf{x}_i) && \text{symmetry} \\
&d(\mathbf{x}_i, \mathbf{x}_k) \leq d(\mathbf{x}_i, \mathbf{x}_j) + d(\mathbf{x}_j, \mathbf{x}_k) && \text{triangle inequality}
\end{aligned}
$$

where $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k \in \mathcal{X}$. A *metric* further requires the property $d(\mathbf{x}_i, \mathbf{x}_j) = 0 \Leftrightarrow \mathbf{x}_i = \mathbf{x}_j$.

These properties allow to identify metrics in a real vector space. In order to incorporate auxiliary information into the metric, we require the concept of Fisher metrics.

2.3.2. Fisher metric as a special case of the Riemannian metric

Concepts from information geometry provide an elegant tool to incorporate auxiliary information into the metric by making use of probability density functions [123]. The next paragraph reviews the required concepts of Riemannian metrics and Fisher metrics as used widely in the information geometry literature. The second paragraph of section 2.3.2 illustrates how these concepts can be used to integrate auxiliary information into the metric.

Classical viewpoint in the Information geometry literature

In the field of Information geometry, a family of probability distributions $S = \{p(\mathbf{x}|\boldsymbol{\theta})\}$ is treated as a statistical model, where $p(\mathbf{x}|\boldsymbol{\theta})$ is a probability density function parameterized by $\boldsymbol{\theta}$ which is defined on an open subset of $\mathbb{R}^n$ and $\mathbf{x} \in \mathcal{X}$ is a random variable. An example for such a function is the probability density of the normal distribution. In order to measure geometric properties between these distributions, such as distances, the structure of a *Riemannian manifold* can be introduced for S. The Riemannian manifold induced by S consists of a Riemannian metric together with a differentiable manifold [4]. For reasonable parameterizations θ and sensible choices of p, S is a differentiable manifold, or short manifold.

A *Riemannian metric* relies on an inner product $g_{\boldsymbol{\theta}}$, which is defined on element pairs from the tangent space at position $\boldsymbol{\theta}$ on a manifold [4, 3]. This inner product can be used to compute the length of an element $\mathbf{v}$ from the tangent space by $\sqrt{g_{\boldsymbol{\theta}}(\mathbf{v}, \mathbf{v})}$. Elements from the *tangent space* are derivatives $\gamma'(t)$ of curves $\gamma(t)$ defined on the manifold, where we write $\gamma'(t)$ as a short form for $\frac{\partial}{\partial t}\gamma(t)$. Also, for infinitesimal close points $\boldsymbol{\theta}$ and $\boldsymbol{\theta} + d\boldsymbol{\theta}$, $d\boldsymbol{\theta}$ is an element from the tangent space.

This notion can be extended to compute the length of a path $\gamma(t)$ along the manifold from point $\boldsymbol{\theta}_i$ to $\boldsymbol{\theta}_j$, where $\gamma : [0, 1] \to \mathbb{R}^n, \gamma(0) = \boldsymbol{\theta}_i, \gamma(1) = \boldsymbol{\theta}_j$ and γ is differentiable with respect to its parametrization t. The length of a path can be computed by

$$\|\gamma\| := \int_0^1 \sqrt{g_{\gamma(t)}(\gamma'(t), \gamma'(t))} dt \tag{2.9}$$

and the distance between the coordinates $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_j$ is then

$$d_R(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \inf_{\gamma} \|\gamma\|, \tag{2.10}$$

where the minimum is taken over all differentiable paths from $\boldsymbol{\theta}_i$ to $\boldsymbol{\theta}_j$.

The *Fisher metric* is a Riemannian metric, which measures distances between infinitesimally close points $\boldsymbol{\theta}$ and $\boldsymbol{\theta} + d\boldsymbol{\theta}$ using the Kullback-Leibler divergence between their according probability densities:

$$D_{\mathrm{KL}}\left(p(\mathbf{x}|\boldsymbol{\theta}), p(\mathbf{x}|\boldsymbol{\theta} + d\boldsymbol{\theta})\right) \propto d\boldsymbol{\theta}^{\mathsf{T}} \mathbf{J}(\boldsymbol{\theta}) d\boldsymbol{\theta}. \tag{2.11}$$

This relations follows from the Taylor expansion of the Kullback-Leibler divergence with respect to $d\boldsymbol{\theta}$ around 0 [90]. The right hand side of equation (2.11) computes the length of the tangent vector $d\boldsymbol{\theta}$ using the inner product

$$g_{\boldsymbol{\theta}}(\mathbf{u}, \mathbf{v}) = \mathbf{u}^{\mathsf{T}} \mathbf{J}(\boldsymbol{\theta}) \mathbf{v}, \tag{2.12}$$

where $\mathbf{u}$ and $\mathbf{v}$ are elements from the tangent space of the data manifold at $\boldsymbol{\theta}$ and $\mathbf{J}(\boldsymbol{\theta})$ is the local *Fisher information matrix*

$$\mathbf{J}(\boldsymbol{\theta}) = \mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta})} \left\{ \left(\frac{\partial}{\partial \boldsymbol{\theta}} \log p(\mathbf{x}|\boldsymbol{\theta}) \right) \left(\frac{\partial}{\partial \boldsymbol{\theta}} \log p(\mathbf{x}|\boldsymbol{\theta}) \right)^{\mathsf{T}} \right\}. \tag{2.13}$$

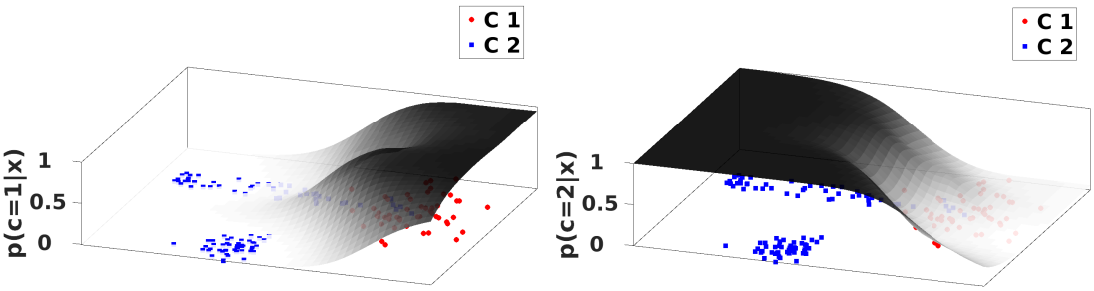


Figure 2.4.: Estimation of $p(c = 1|\mathbf{x})$ (left) and $p(c = 2|\mathbf{x})$ (right) for the toy data set using the Parzen window estimator.

$\mathbf{J}(\boldsymbol{\theta})$ is based on derivatives of $p(\mathbf{x}|\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$, i.e. it emphasizes directions of $\boldsymbol{\theta}$ along which $p(\mathbf{x}|\boldsymbol{\theta})$ changes the strongest. $\mathbf{J}(\boldsymbol{\theta})$ is also referred to as a *metric tensor*.

This is a short and strongly abbreviated summary of the according sections in [4, 3, 96]. More details can be found therein.

The Fisher metric in this thesis

This framework is usually used in the parameter space in the context of parameter estimation $\boldsymbol{\theta}$. Here we will adopt a different purpose: we follow ideas from [123, 122] and employ the Fisher metric in the data space. We replace the probability density function $p(\mathbf{x}|\boldsymbol{\theta})$ of a random variable $\mathbf{x}$ given certain parameters $\boldsymbol{\theta}$ by $p(\mathsf{aux}|\mathbf{x})$, where $\mathbf{x}$ is a position in the data space and aux the value of an auxiliary variable. Thereby, the auxiliary variable aux is supposed to indicate particularly important aspects in the data. An example for this can be the label in a classification scenario, as utilized in [123], or the value of a continuous regression variable. In [123], this concept is referred to as *learning metrics* because it enables to define a new metric, the Fisher metric, from the given data and the corresponding auxiliary variable. So it is learned from the data.

In our setting the Fisher Information matrix is given by

$$\mathbf{J}(\mathbf{x}) = \mathbb{E}_{p(\mathsf{aux}|\mathbf{x})}\left\{\left(\frac{\partial}{\partial \mathbf{x}}\log p(\mathsf{aux}|\mathbf{x})\right)\left(\frac{\partial}{\partial \mathbf{x}}\log p(\mathsf{aux}|\mathbf{x})\right)^{\mathsf{T}}\right\}, \tag{2.14}$$

which again results from the local Kullback-Leibler divergence [137, 145].

Employing this setting has the effect that the local positive semidefinite matrix $\mathbf{J}$ amplifies directions along which the auxiliary variable changes. Dimensions which are locally irrelevant for aux do not contribute.

Combining these aspects, global distances in the Fisher metric framework are defined by

$$d_M(\mathbf{x}_i, \mathbf{x}_j) = \inf_{\gamma} \|\gamma\| = \inf_{\gamma} \int_0^1 \sqrt{\gamma'(t)^{\mathsf{T}}\mathbf{J}(\gamma(t))\gamma'(t)}dt, \tag{2.15}$$

where γ is again a path on the manifold, with $\gamma : [0,1] \to \mathcal{X}$, $\gamma(0) = \mathbf{x}_i$ and $\gamma(1) = \mathbf{x}_j$ are fixed.

Additionally, if the utilized auxiliary information is noisy or other aspects in the data should not be neglected completely, the Fisher metric can be regularized by combining it with the Euclidean metric via setting the inner product $g_{\mathbf{x}}$ to

$$g_{\mathbf{x}}(\mathbf{u}, \mathbf{v}) = \mathbf{u}^{\mathsf{T}} \left(\mathbf{J}(\mathbf{x}) + \lambda \mathbf{I}\right) \mathbf{v}, \tag{2.16}$$

where $\mathbf{I}$ refers to the identity matrix having ones on the diagonal and zeros elsewhere and λ is the weighting factor for the Euclidean metric, which is usually small.

This Fisher metric emphasizes changes of the auxiliary variable. It can be used for interpretable data analysis [115] or to replace the Euclidean metric in dimensionality reduction methods to obtain discriminative dimensionality techniques.

2.3.3. Approximation of the shortest paths

In order to compute the distance on the Riemannian manifold (we will sometimes refer to this entity as *Fisher distance*, in the following) between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ as defined by equation (2.15), minimal path integrals need to be computed. However, this is usually computationally intractable and, hence, approximations are required. We repeat the most common ones here for convenience [123].

Local approximation The most simple approximation is to assume that the shortest path between two points is the straight line and to utilize the inner product $g_{\mathbf{x}_i}$:

$$d_1(\mathbf{x}_i, \mathbf{x}_j) = g_{\mathbf{x}_i}(\mathbf{x}_i - \mathbf{x}_j, \mathbf{x}_i - \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^{\mathsf{T}} \mathbf{J}(\mathbf{x}_i)(\mathbf{x}_i - \mathbf{x}_j). \tag{2.17}$$

This is only exact if $\mathbf{x}_i$ and $\mathbf{x}_j$ are infinitesimally close to each other. Since the Fisher information matrix is only computed on the position $\mathbf{x}_i$, it neglects change of the auxiliary variable in other regions and, hence, constitutes a very crude approximation.

Straight line approximation An extension of this is still to assume that the shortest path is the straight line and to approximate the length of this line by T piecewise constant terms induced by equidistant points on the line from $\mathbf{x}_i$ to $\mathbf{x}_j$. Define points $\mathbf{x}_t$ on this line as convex combinations of $\mathbf{x}_i$ and $\mathbf{x}_j$: $\mathbf{x}_t = \mathbf{x}_i + (t-1)/T \cdot (\mathbf{x}_j - \mathbf{x}_i)$, with $t \in \{1, \ldots, T\}$. Using the inner product based approximation for consecutive points $\mathbf{x}_t$ and $\mathbf{x}_{t+1}$, the exact distance on the manifold d_M can be approximated by

$$d_T(\mathbf{x}_i, \mathbf{x}_j) = \sum_{t=1}^{T} \sqrt{(\mathbf{x}_{t+1} - \mathbf{x}_t)^{\mathsf{T}} \mathbf{J}(\mathbf{x}_t)(\mathbf{x}_{t+1} - \mathbf{x}_t)}. \tag{2.18}$$

This approximation is accurate if the shortest path is indeed close to the straight line and a sufficiently large number T is used. This way to compute the length of a straight line leads to asymmetric distances. These can either be symmetrized after the computation or by a symmetric sampling along the line.

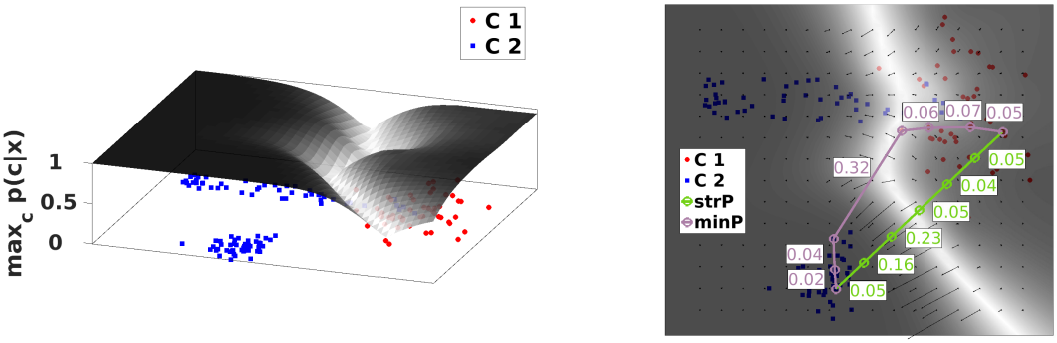


Figure 2.5.: Parzen window estimation of max $(p(c=1|\mathbf{x}), p(c=2|\mathbf{x}))$. The right plot shows the same figure viewed from above together with the eigenvectors of the Fisher matrices scaled with their according eigenvalues and the straight path approximation together with a minimal path.

Graph-based approximation A more precise approximation removes the assumption of shortest paths being straight lines and searches for the shortest path in a graph: Assume a fully connected graph where each pair of data points is connected by an edge. The length of each edge is computed using the straight line approximation d_T. Now, standard graph search algorithms such as Floyd's algorithm can be employed to search for the shortest path in this graph. Finding all shortest paths has cubic complexity in the number of nodes, i.e. data samples in this case.

Although this graph-based approach is the only approximation guaranteed to be a pseudometric, the results from [163, 123] show that the straight line approximation d_T (2.18) works reasonably well in practice and can be beneficial since it reduces the computational complexity from cubical to squared.

2.3.4. Example

In order to illustrate the definition of the Fisher information matrix we adopt a classification scenario, where the auxiliary variables aux take the role of class labels c. For this purpose we construct a toy data set which is shown on the left of Figure 2.3. This data set is two-dimensional, class 2 consists of two modes and one of them overlaps with points of class 1.

Since the Fisher matrix depends on the class-density $p(c|\mathbf{x})$, we show this function in Figure 2.4. This is a two-class problem and, hence, there exist actually two functions $p(c=1|\mathbf{x})$ and $p(c=2|\mathbf{x})$. For each position in the data space $\mathbf{x}$, the former expresses the probability of $\mathbf{x}$ belonging to class 1 and the latter the probability of $\mathbf{x}$ belonging to class 2. For the visualizations we estimate $p(c|\mathbf{x})$ from the data using the Parzen window estimator described in section 2.4.1.

For illustrational purposes, we can combine $p(c=1|\mathbf{x})$ and $p(c=2|\mathbf{x})$ by plotting $max\,(p(c=1|\mathbf{x}), p(c=2|\mathbf{x}))$ for every position $\mathbf{x}$. This is shown in the left plot of

Figure 2.5, where the right visualization constitutes the same plot, viewed from above. Here, the region of highest uncertainty is clearly visible. Further, the right hand side also displays the Fisher matrices: The black arrows show the eigenvectors of the Fisher matrices scaled with the according eigenvalues. They are largest where the class-density changes, corresponding to the derivative of $p(c|\mathbf{x})$.

Further, for every position $\mathbf{x}$ in the data space, there is only one direction of change, i.e. the Fisher matrices are one-dimensional. This is analytically clear because the Fisher matrix consists of an expectation, which reduces to a sum with as many elements as the number of classes, i.e. two in this case. Further, since $p(c = 1|\mathbf{x})$ and $p(c = 2|\mathbf{x})$ are linearly dependent (they sum to 1), the Fisher matrices for two-class problems have actually rank one.

Figure 2.5 also displays the distance along two paths between two selected points, using different approximations: the straight line approximation is shown in green using 5 intermediate points on the line and distances between these line segments are printed in the same color. The pink curve results from the shortest path search in the fully connected graph consisting of the data points. The straight line and the graph search approximations yield the distances 0.578 and 0.562, respectively, i.e. they are similar in this case. In the worst case, there can exist a large difference. However, as investigated in [123], the difference is usually not pronounced for real life data and closer neighborhood sizes, i.e. the straight line approximation constitutes a valuable choice for the purpose of discriminative dimensionality reduction.

2.4. Discriminative dimensionality reduction for classification tasks

In this section we will explain how to compute DiDi mappings using the following general idea: Modify the metric locally [123, 116] by defining a Riemannian manifold. The latter takes the auxiliary information of the data into account by measuring the effect of the data dimensions in the feature space for this auxiliary information. Then we can employ any distance based DR method on top of this modified metric to obtain a discriminative dimensionality reduction technique. In particular, we replace the Euclidean metric by the Fisher metric.

We have already introduced the basic concepts of the Fisher metric in section 2.3, and the relevant modifications to apply them for DiDi in section 2.3.2. This enables us to compute Fisher distances based on the data $\mathbf{X}$ and auxiliary information aux. Now, we can detail the procedure for the case of discrete auxiliary information, occurring for instance in classification scenarios. In this case, we can use the Fisher metric by treating the class labels c of the classification problem as auxiliary data aux. Then we can, as described before, replace the Euclidean metric by the Fisher metric in distance based DR methods, and such, obtain discriminative DR techniques. Many popular dimensionality reduction methods can be extended in such a way to become discriminative dimensionality reduction techniques. These include MDS, SOM, GTM, Isomap,

MVU, SNE, t-SNE and NeRV just to name the most popular ones. We refer to DiDi methods based on the Fisher metric by adding the prefix *Fisher* to their name, e.g. Fisher t-SNE for t-SNE applied to Fisher distances.

To this end, we need to compute Fisher matrices, which boils down to estimating a probability density model $p(c|\mathbf{x})$ and computing its gradient with respect to $\mathbf{x}$. In the following, we will discuss feasible techniques for doing so and demonstrate the advantage of DiDi tools for a labeled toy data set.

2.4.1. Approximation of the probabilities

A central part of this modified distance computation consists in the estimation of the probability $p(c|\mathbf{x})$ of information c given a data point $\mathbf{x}$. In the setting of a classification scenario, there are two essentially different possibilities how to choose this information:

(a) We can use the given class labels $c_i := l_i$ for data point $\mathbf{x}_i$, respectively, as provided in the training set. This choice emphasizes the given 'ground truth' of the data.

(b) We can use the labeling as provided by a trained classifier $c := f(\mathbf{x})$, if such a model is available. This choice emphasizes aspects of the data which are regarded by the classifier as interesting. Hence, those aspects of the data can be visualized which influence the trained classification.

Apart from the different semantic meaning, this choice has consequences on the possibilities how to compute the probability $p(c|\mathbf{x})$. The Fisher matrix is based on the local change of the probability distribution $p(c|\mathbf{x})$, the latter of which is usually unknown and needs to be approximated. A common way to do this is to use the Parzen window non-parametric estimator as proposed in [123]. Essentially, computation takes place by estimating class probabilities as

$$\hat{p}(c|\mathbf{x}) = \frac{\sum_i \delta_{c,c_i} \exp(-0.5\|\mathbf{x}-\mathbf{x}_i\|^2/(\sigma^p)^2)}{\sum_j \exp(-0.5\|\mathbf{x}-\mathbf{x}_j\|^2/(\sigma^p)^2)}, \tag{2.19}$$

where the sum is over all data points and σ^p constitutes the bandwidth parameter. The Fisher information matrix based on the Parzen window estimator [123] becomes

$$\mathbf{J}(\mathbf{x}) = \frac{1}{(\sigma^p)^4} \mathbb{E}_{\hat{p}(c|\mathbf{x})} \left\{ \mathbf{b}(\mathbf{x},c)\mathbf{b}(\mathbf{x},c)^\mathsf{T} \right\} \tag{2.20}$$

where

$$\mathbf{b}(\mathbf{x},c) = \mathbb{E}_{\zeta(i|\mathbf{x},c)}\{\mathbf{x}_i\} - \mathbb{E}_{\zeta(i|\mathbf{x})}\{\mathbf{x}_i\} \tag{2.21}$$

$$\zeta(i|\mathbf{x},c) = \frac{\delta_{c,c_i} \exp(-0.5\|\mathbf{x}-\mathbf{x}_i\|^2/(\sigma^p)^2)}{\sum_j \delta_{c,c_j} \exp(-0.5\|\mathbf{x}-\mathbf{x}_j\|^2/(\sigma^p)^2)} \tag{2.22}$$

$$\zeta(i|\mathbf{x}) = \frac{\exp(-0.5\|\mathbf{x}-\mathbf{x}_i\|^2/(\sigma^p)^2)}{\sum_j \exp(-0.5\|\mathbf{x}-\mathbf{x}_j\|^2/(\sigma^p)^2)}. \tag{2.23}$$

$\mathbb{E}$ denotes the empirical expectation, i.e. weighted sums with weights depicted in the subscript. If large data sets or out-of-sample extensions are dealt with, a subset of the data only is usually sufficient for the estimation of $\mathbf{J}(\mathbf{x})$.

Appendix A.1 details how the definition of the Fisher information matrix (2.14) and the Parzen window estimator give rise to the result provided by eqs. (2.20) to (2.23).

In principle, any differentiable estimator for $p(c|\mathbf{x})$ can be employed, such as a multilayer perceptron [116]. The advantage of the parzen window estimator, however, is that it yields a correct estimation of the probability density in the limit. A disadvantage is its computational cost, $\mathcal{O}(N^2)$ for N data points. In case this is too demanding, a fixed subset can be used. Suitable choices to estimate the bandwidth parameter from the data are provided in the literature, e.g. by $\hat{h}_{rot}$ [153].

As an alternative, provided the class labels $f(\mathbf{x})$ given by a classification function are of interest, it is often possible to rely on the explicit functional form of f as provided by the classifier if the latter yields a probabilistic output for the class labels.

2.4.2. Example

In this section, we demonstrate a DiDi mapping based on the Fisher metric. We utilize the scheme from section 2.4.1 to compute Fisher distances and use these together with t-SNE. We apply this scheme to an artificially generated data set constituting a three-dimensional filled ball. A non-linear tube is contained inside this ball defining two classes. This data set is shown in the right image of Figure 2.3.

A projection of this data set to 2 dimensions is shown in Figure 2.6, where t-SNE using the Fisher metric (top left) and t-SNE together with the Euclidean metric (bottom left) is applied to a subset. The remaining data set can be projected using kernel t-SNE. The according out-of-sample extensions are shown in the right column.

While the unsupervised t-SNE projection distorts the structure of class 2 in the projection, the Fisher t-SNE projection preserves the tube-like structure. Additionally, a noisy region (on the left) in the projection is visible which is indeed present in the original data set (there it is visible in the bottom).

2.5. Discriminative dimensionality reduction in kernel space

Although the discriminative dimensionality reduction techniques which are based on the Fisher metric have shown to perform well in practice [163], they are, like most DiDi methods, restricted to vectorial data. Hence, they are not applicable whenever complex, non-vectorial data structures are dealt with. In this section, we propose an extension of the Fisher metric to kernel spaces, thereby enabling powerful DiDi technologies for general data structures described in terms of pairwise relations, the kernel matrix, only. Following section 2.4, we compute DiDi mappings by applying DR methods on distances computed with the Fisher metric. For this purpose, we rely on the formalization of Fisher distances presented in section 2.3.2 and estimate the required densities using the Parzen window estimator, as discussed in section 2.4.1.

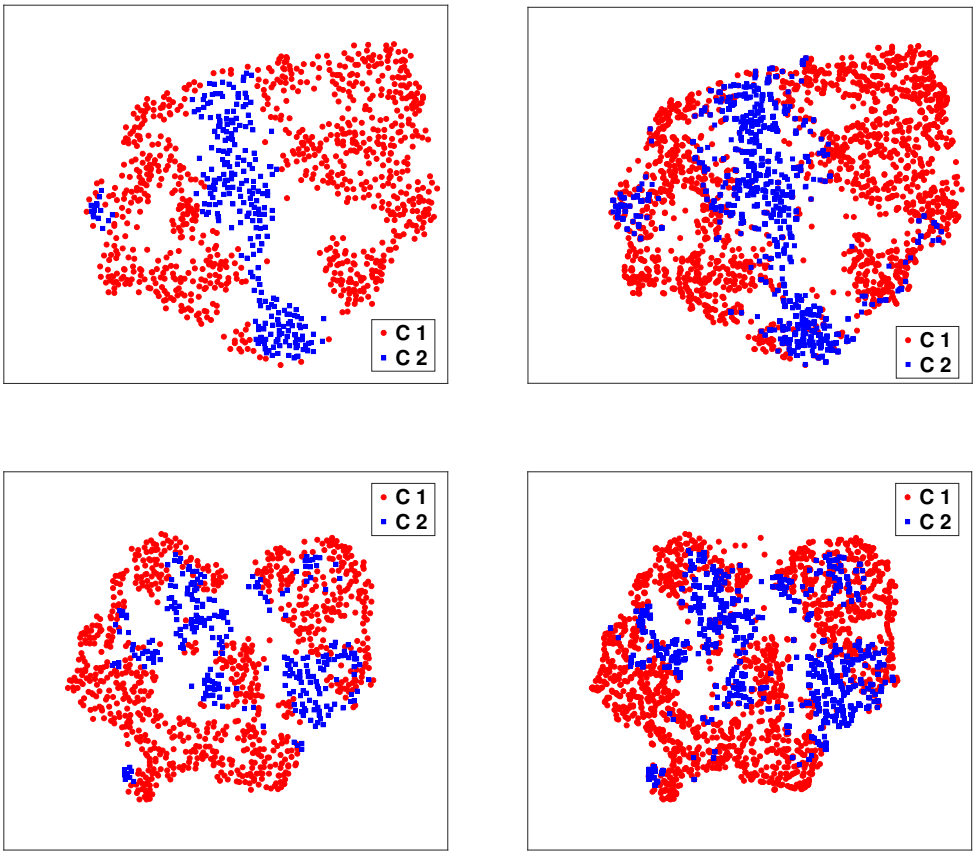


Figure 2.6.: Visualization of the ball data set with t-SNE (left) and out of sample extension with kernel t-SNE (right). The Fisher metric is utilized in the top row, the Euclidean metric in the bottom row.

We further employ the straight line approximation for the minimum path integrals as discussed in section 2.3.3.

We demonstrate the feasibility of the approach for several benchmarks, including complex structured data from the domains of music and java programming.

2.5.1. Kernelization

We assume that data are characterized in terms of pairwise similarities only, i.e. a matrix $\mathbf{S} \in \mathbb{R}^{N \times N}$ is given with N being the number of data, entries are denoted as s_{ij}. We assume symmetry of $\mathbf{S}$, such that an implicit vectorial embedding exists [62]. Further, we require non-negativity of the values to guarantee a valid probability distribution. In particular, this covers the case of structure kernels for complex data structures [105]. However, we will see in experiments that the Fisher metric also provides reasonable results for general matrices. We denote data in kernel space as $\mathbf{x}_i$ where $s_{ij} = \mathbf{x}_i^\top \mathbf{x}_j$. Since we utilize the straight line approximation for shortest paths on

the manifold, we require to calculate coordinates on the line between two points and compute distances between them. Equidistant points on the line from $\mathbf{x}_i$ to $\mathbf{x}_j$ have the form $(1-\alpha)\mathbf{x}_i + \alpha\mathbf{x}_j$ where $\alpha = (t-1)/T$ for $t \in \{1, \ldots, T+1\}$, hence, differences of consecutive points have the form $(\mathbf{x}_j - \mathbf{x}_i)/T$. Thus, denoting $\mathbf{x}(t) := (1-\alpha)\mathbf{x}_i + \alpha\mathbf{x}_j$, distances $d_T(\mathbf{x}_i, \mathbf{x}_j) \cdot (T\sigma^2)$ consist of terms of the form

$$\sigma^4(\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{J}(\mathbf{x}(t))(\mathbf{x}_i - \mathbf{x}_j) = \sum_c \hat{p}(c|\mathbf{x}(t)) \left(\mathbf{x}_i^\top \mathbf{b}(\mathbf{x}(t), c) - \mathbf{x}_j^\top \mathbf{b}(\mathbf{x}(t), c)\right)^2 \tag{2.24}$$

where

$$\mathbf{x}_i^\top \mathbf{b}(\mathbf{x}(t), c)) = \sum_l \left(\zeta(l|\mathbf{x}(t), c) \cdot \underbrace{\mathbf{x}_i^\top \mathbf{x}_l}_{s_{il}} - \zeta(l|\mathbf{x}(t)) \cdot \underbrace{\mathbf{x}_i^\top \mathbf{x}_l}_{s_{il}}\right) \tag{2.25}$$

The terms $\hat{p}(c|\mathbf{x}(t))$, $\zeta(l|\mathbf{x}(t), c)$, and $\zeta(l|\mathbf{x}(t))$ are defined in section 2.4.1 and can be expressed in terms of Gaussians with the argument

$$\|\mathbf{x}(t) - \mathbf{x}_l\|^2 = (1-\alpha)^2 s_{ii} + \alpha^2 s_{jj} + s_{ll} + 2(1-\alpha)\alpha s_{ij} - 2(1-\alpha)s_{il} - 2\alpha s_{jl}, \tag{2.26}$$

hence, the full computation can be kernelized.

2.5.2. Experiments

Our reformulation of Fisher distance computations in terms of kernels does not rely on approximations and, hence, is equivalent to the vectorial computation if the similarity matrix $\mathbf{S}$ is given by a standard scalar product. Hence, we do not present comparisons to the vectorial case, here.

Instead, we evaluate the method for six benchmark data sets that are only given as similarity matrices and are not necessarily Euclidean.

Aural Sonar [126]: Data consist of 100 returns from a broadband active sonar system, their similarity is evaluated by human experts. Two classes (target of interest versus clutter) are distinguished.

Patrol [31]: 241 members of seven patrol units are characterized by (partially faulty) feedback of unit members naming five colleagues each.

Protein [68]: 226 globin proteins are compared based on their evolutionary distances, four classes of different protein families result.

Voting [31, 45]: 435 either republican or democrat candidates are characterized by 16 nominal attributes which characterize the key votes identified by the Congressional Quarterly Almanac. The value difference metric is used for comparison.

Java Programs [118, 119]: 64 Java programs which implement bubble sort or insertion sort, respectively, have been retrieved from the internet. They are compiled with the Oracle Java Compiler API and compared by alignment.

Sonatas [55]: 1068 sonatas in MIDI format from the online collection *Kunst der Fuge* are transformed to graph structures and compared with the normalized compression distance of their paths, labeling is given by one of 5 composers from the classical / baroque era.

A more detailed description of the data can be found in [31, 55].

Each data set is characterized in terms of a symmetrized similarity matrix $\mathbf{S}$. All data are projected to two dimensions based on t-distributed stochastic neighbor embedding (t-SNE), see section 2.2.1 for a description. We compare the result of a projection of t-SNE, which is directly applied to the dissimilarity matrix as induced by $\mathbf{S}$, and to the dissimilarity matrix computed from the Fisher metric. We denote the former step as t-SNE and the latter as Fisher t-SNE, for short. Note that some of the data matrices $\mathbf{S}$ do not relate to valid kernels, i.e. have negative Eigenvalues (EVs). Therefore, we compare the result achieved with plain data $\mathbf{S}$ and its clip-based eigenvalue correction [31, 62]. Notably, the Fisher metric does not encounter numerical difficulties when addressing the plain data, while t-SNE does.

Besides the visual impression, we compare the methods by a 1-nearest neighbor (1-NN) classification in the projection space. Thereby we also report the result which we obtain when applying Fisher t-SNE to data with randomly permuted labels, which corresponds to the quality which is merely due to statistical effects of the data. We refer to the 1-NN error in this setting as a baseline. Note that it is not reasonable to evaluate the projections by the quality framework [95] since we do not aim to preserve neighborhoods based on Euclidean distances.

For the computation of distances in the Fisher metric, the parameter σ for the Parzen window estimate has to be specified. In order to find an appropriate value, we compute bandwidths using the perplexity based idea as in [158], and average those to obtain a single bandwidth value.

If all data points are utilized for the Parzen window estimator, Fisher t-SNE sometimes tends to overfit the data. Such a behaviour could be monitored by a low baseline value. In order to prevent such a behaviour, we use 60% - 100% of the data for the Parzen window estimator.

Since t-SNE is not deterministic, we run the t-SNE algorithm 10 times on the respective distance matrix. The averaged leave-one-out 1-NN errors for the six data sets are displayed in Table 5.1, with standard deviations depicted in brackets. If clipping is applied, this is stated behind the method name. For the clipped Eigenspectrum of $\mathbf{S}$, the 1-NN errors of both t-SNE and Fisher t-SNE are comparably low (see e.g. [31]). Further, the discriminative projections have an even lower classification error, on average. The high baseline error indicates that Fisher t-SNE does not neglect the intrinsic structure of the data when the task is to embed a random class distribution.

Based on the clipped Eigenspectrum, an instance of each embedding is shown in Fig. 2.7. For each data set, a t-SNE projection is shown in rows one and three, a Fisher t-SNE mapping in rows two and four.

In addition to the numerical evaluation, these visualizations show that the Fisher

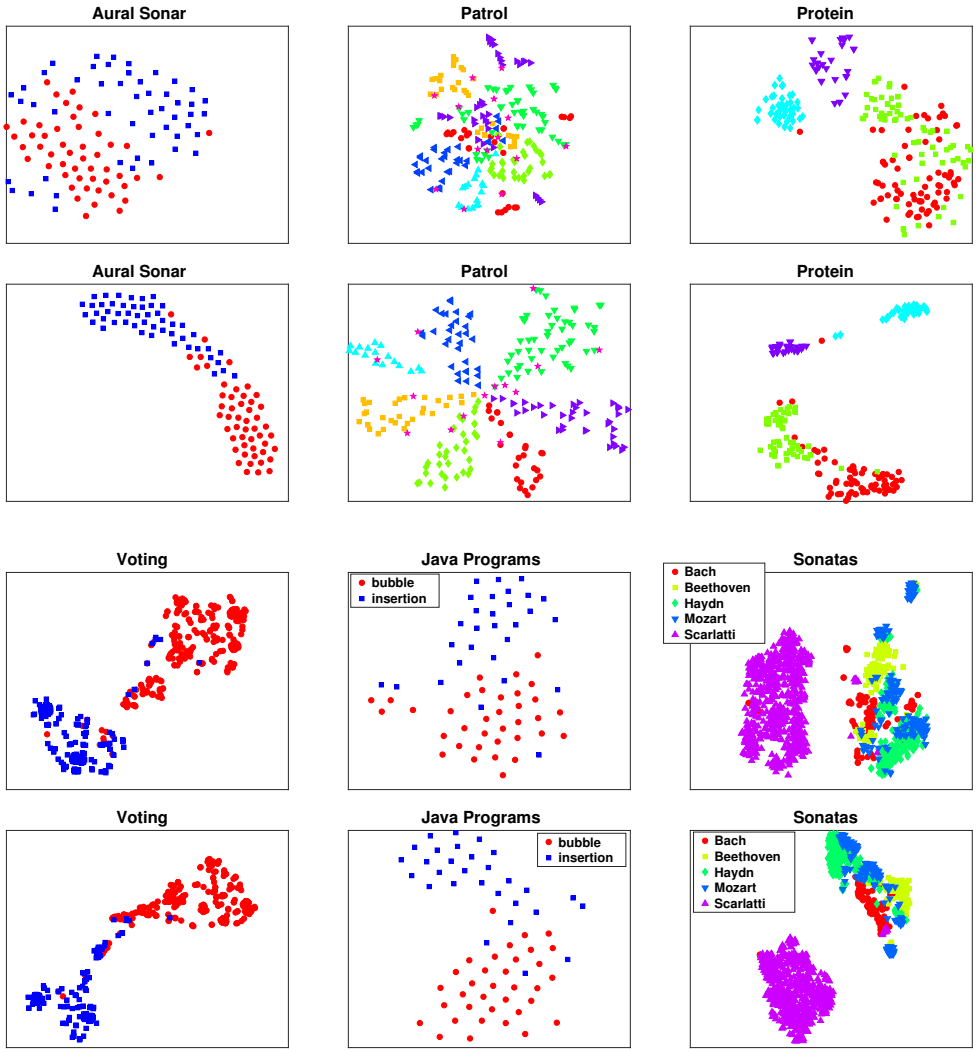


Figure 2.7.: Unsupervised t-SNE projections in rows one and three of the data sets Aural Sonar, Patrol, Protein, Voting, Java Programs and Sonatas. Rows two and four contain the according supervised Fisher t-SNE projection.

Table 2.1.: Average 1-NN classification errors in percent with standard deviations; sum of the negative EVs in relation to the summed absolute values of the EVs.

	AuralS	Patrol	Protein	Voting	Java	Sonatas
original data (clip)	17	19	10	6	11	11
t-SNE (clip)	15 ($\pm$2)	16 ($\pm$1)	8 ($\pm$1)	7 ($\pm$1)	13 ($\pm$2)	9 ($\pm$1)
Fisher t-SNE (clip)	9 ($\pm$1)	11 ($\pm$1)	3 ($\pm$1)	4 ($\pm$1)	11 ($\pm$2)	6 ($\pm$1)
original data	21	7	77	6	14	13
t-SNE	18 ($\pm$2)	87 ($\pm$1)	31 ($\pm$6)	7 ($\pm$1)	15 ($\pm$2)	10 ($\pm$1)
Fisher t-SNE	10 ($\pm$3)	15 ($\pm$1)	4 ($\pm$0)	6 ($\pm$1)	14 ($\pm$2)	6 ($\pm$1)
baseline (clip)	40	81	48	43	45	49
perc. negative Eigs	21	50	20	0	8	2

information based projections have a clearer class separability and, hence, enable the user to get a better understanding of the data. The unsupervised projection of the Protein data set, for instance, suggests that two classes are strongly overlapping. Here, the discriminative visualization, which emphasizes local directions that are relevant for class separation, shows that both classes have only a few overlapping points. Another example constitutes the Patrol data set, where the Fisher t-SNE embedding shows a clear class structure with only few noisy points coming from a specific class.

Another interesting aspect in Table 5.1 is the classification performance on the original data, without clipping. While t-SNE suffers from a large accuracy loss for the Patrol and Protein data sets, Fisher t-SNE obtains stable results with only a slight performance decrease. Particularly the Patrol data set has large negative eigenvalues, as can be seen in Table 5.1.

Fisher t-SNE seems to be particularly robust towards a indefinite similarity matrix. A reason for this could be that the Fisher metric sets a focus on changes with respect to the class labels, while neglecting other changes. This can lead to a reduction of the dimensionality and, hence, to an easier problem for the embedding, in particular, if the neglected changes are along the negative directions of pseudoeuclidean embedding.

2.5.3. Conclusion

In this section we have reformulated one particularly popular approach for discriminative dimensionality reduction such that it is applicable to non-vectorial data only given by kernel values or more general. We evaluated this method with six data sets from this domain and obtained a clear improvement as compared to unsupervised projections in many cases. The robustness of Fisher t-SNE towards indefinite proximities seems interesting and requires further investigation.

2.6. Discriminative dimensionality reduction for regression tasks

The current formulation of DiDi based on the Fisher metric is restricted to discrete auxiliary variables. This restricts the application of these methods and prevents them from being applied to real-valued auxiliary data. However, such data occur quite frequently as for instance in the class of regression problems. An example of a real-valued auxiliary variable can be the continuously measured progress of a disease.

The field of discriminative dimensionality reduction with real-valued auxiliary data is by far less investigated than it is the case for discrete auxiliary information. In the literature, there exist basically two families of approaches which are particularly formulated for the task of regression. These are methods using the concept of inverse regression (IR) [99, 176, 80, 100, 33] and of kernel dimension reduction (KDR) [50, 110]. The latter rely on a measure of conditional independence between the regression target and the data conditioned on the projection space, following the idea that the projection space should contain all relevant information to predict the target. The former approaches rely on the idea of inverse regression, meaning that they try to predict the data from the target variable. The image of the resulting function characterizes the low-dimensional projections space. However, these techniques comprise mostly linear methods which provide less flexible mappings as compared to nonlinear projections. One notable exception constitutes the method manifold kernel dimensionality reduction (mKDR) [110] which combines the linear KDR (the kernel is used to estimate conditional independence while the DR is performed with a linear mapping) together with Laplacian eigenmaps. It constitutes a linear DiDi projection applied on top of an unsupervised nonlinear projection, i.e. the nonlinear part is performed in an unsupervised way. Another interesting exception is the covariance operator inverse regression (COIR) [80] which uses the IR concept together with the kernel trick, similarly as in kernel PCA, and the concept of covariance operators in RKHS.

In contrast, in the context of DiDi with discrete auxiliary information, methods that build on top of the Fisher metric have been shown to perform particularly well [163]. Following this research line, we propose a new DiDi framework in the setting of real-valued auxiliary variables which enables us to incorporate such auxiliary information into the metric. This discriminative metric can then be combined with any distance based DR method to yield a DiDi technique. Essentially, we will answer the question: How can we use the framework discussed in section 2.3 and reformulate it such that it is applicable to real-valued auxiliary information? We will employ the methodology of Gaussian processes for this purpose.

More precisely, we assume that auxiliary information is specified as a real-valued variable. E.g. the user specifies relevant variation due to prior knowledge, one variable y is singled out or y constitutes a regression target variable. Then the DiDi task refers to the goal of visualizing only that information in $\mathcal{X}$ which is relevant for y.

We have already discussed in section 2.1 that the incorporation of label information

effectively reduces the intrinsic data dimensionality, hence the ill-posed DR problem becomes more well-posed by incorporating this explicit label information. A similar effect holds when incorporating real-valued data. The relation of the data and the real-valued auxiliary information can often be phrased in terms of a function $y = f(\mathbf{x})$, where f is smooth. Hence this equation defines a manifold of co-dimension one, which means that, locally, all but one dimension of the data $\mathbf{x}$ can be neglected as concerns changes in the value $f(\mathbf{x})$. Hence, taking directions which do influence this manifold only, data are regularized towards a locally one-dimensional representation. The visualization problem becomes one of visualization a one-dimensional manifold, only! We will later make this argument more precise by referring to the corresponding Fisher matrix. Note that, the concept of visualizing only the information in $\mathbf{x}$, which influences f, preserves important characteristics of the data such as the question whether there is a pronounced relation of $\mathbf{x}$ and y at all, what is the number of modes, do there exist outliers. etc. Due to this observation, it is apparent that supervised dimensionality reduction is a task which is different from a direct prediction of the auxiliary variable y: aspects in the data $\mathbf{x}$ which are relevant for y are visualized, whereby the auxiliary information is used as a regularization only about which data aspects are of particular relevance.

In this chapter, we propose to use the concepts from section 2.3 and reformulate them such that they can be used together with a real-valued auxiliary variable. We formalize this in section 2.6.2 for the computation of the Fisher matrix. Since we employ the framework of Gaussian processes in this context, we review this methodology in advance in section 2.6.1. Having these concepts, we can use the shortest path approximations from 2.3 to define Fisher distances and plug these into any distances based DR technique, such as t-SNE.

2.6.1. Gaussian Processes for regression

In the following, we will recall the formal concepts of a Gaussian Process [131]. A Gaussian process defines a distribution directly over the space of functions; model inference can be done by conditioning this probability based on observed values. This abstract concept becomes tractable based on the observation that a distribution over the space of functions can be linked to a family of distributions over the output values of finite sets of inputs only.

More formally, assume data points $\mathbf{x}_i \in \mathcal{X} = \mathbb{R}^D$ in a high-dimensional feature space are given together with observed targets $y_i \in \mathbb{R}$. The matrix $\mathbf{X}$ contains all the data points $\mathbf{x}_i$ in the columns and the vector $\mathbf{y}$ all the targets.

Following [131], we write a Gaussian Process (GP), a collection of random variables having a joint Gaussian distribution, as

$$f(\mathbf{x}) \sim GP(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')) \tag{2.27}$$

which is specified by its mean function and covariance function

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})] \tag{2.28}$$
$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))]. \tag{2.29}$$

The random variables are the values $f(\mathbf{x})$ in the target space. For the covariance function we employ the squared exponential (or Gaussian) kernel

$$k(\mathbf{x}, \mathbf{x}') = \exp(-\beta \|\mathbf{x} - \mathbf{x}'\|^2) + \sigma_{GP}^2, \tag{2.30}$$

where σ_{GP}^2 models the variance of the additive i.i.d. noise ϵ of the targets, i.e. we assume $y = f(\mathbf{x}) + \epsilon$.

We can construct the joint Gaussian distribution $\mathcal{N}$ of the observed target values $\mathbf{y}$ and the unobserved target value y_* at the position $\mathbf{x}_*$:

$$\begin{bmatrix} \mathbf{y} \\ y_* \end{bmatrix} = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_{GP}^2 \mathbf{I} & \mathbf{k}_* \\ \mathbf{k}_*^\mathsf{T} & k_* \end{bmatrix}\right), \tag{2.31}$$

where $[\mathbf{K}]_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$, $[\mathbf{k}_*]_i = k(\mathbf{x}_i, \mathbf{x}_*)$, $k_* = k(\mathbf{x}_*, \mathbf{x}_*)$ and mean 0 is usually assumed for the targets. The latter is not a restriction of the method, since the mean can easily be subtracted from the target data.

By a-posteriori conditioning this Gaussian prior distribution on the observed data, we can obtain the predictive distribution

$$y_* | \mathbf{X}, \mathbf{y}, \mathbf{x}_* \sim \mathcal{N}(\bar{y}_*, \mathrm{cov}(y_*)), \quad \text{where} \tag{2.32}$$
$$\bar{y}_* = \mathbf{k}_*^\mathsf{T} \boldsymbol{\alpha}, \quad \boldsymbol{\alpha} = (\mathbf{K} + \sigma_{GP}^2 \mathbf{I})^{-1} \mathbf{y} \tag{2.33}$$
$$\mathrm{cov}(y_*) = k_* - \mathbf{k}_*^\mathsf{T} (\mathbf{K} + \sigma_{GP}^2 \mathbf{I})^{-1} \mathbf{k}_*. \tag{2.34}$$

2.6.2. Estimating the Fisher matrix based on a Gaussian Process

Following the concept of learning metrics based on the Fisher metric in 2.3.2, we assume for the regression scenario that each observed instance $\mathbf{x}$ is accompanied by auxiliary information in form of a target value $y \in \mathbb{R}$. Using the latter to define a metric tensor, we obtain the Fisher information matrix

$$\mathbf{J}(\mathbf{x}) = \mathbb{E}_{p(y|\mathbf{x})} \left\{ \left(\frac{\partial}{\partial \mathbf{x}} \log p(y|\mathbf{x}) \right) \left(\frac{\partial}{\partial \mathbf{x}} \log p(y|\mathbf{x}) \right)^\mathsf{T} \right\}. \tag{2.35}$$

In order to compute these Fisher matrices, we need (I) to estimate $p(y|\mathbf{x})$, (II) to compute the derivative thereof and (III) to calculate the expectation.

(I) We assume that $p(y_*|\mathbf{x}_*)$ is induced by a Gaussian Process. Formally, we prefer to write $p(y_*|\mathbf{X}, \mathbf{y}, \mathbf{x}_*)$ in this case. Following equation (2.32), we have

$$p(y_*|\mathbf{X}, \mathbf{y}, \mathbf{x}_*) = \frac{1}{\sqrt{2\pi \mathrm{cov}(y_*)}} \exp\left(-\frac{(y_* - \bar{y}_*)^2}{2 \cdot \mathrm{cov}(y_*)} \right), \tag{2.36}$$

where $\bar{y}_*$ and $\mathrm{cov}(y_*)$ are defined in equations (2.33) and (2.34).

(II) The derivative of $p(y_*|\mathbf{X}, \mathbf{y}, \mathbf{x}_*)$, where both $\bar{y}_*$ and $\mathrm{cov}(y_*)$ depend on $\mathbf{x}_*$, is given by:

$$\frac{\partial}{\partial \mathbf{x}_*} \log p(y_*|\mathbf{X}, \mathbf{y}, \mathbf{x}_*) = \frac{\beta}{\mathrm{cov}(y_*)} \left(2(y_* - \bar{y}_*)\mathbf{t}_1 + \left(\frac{(y_* - \bar{y}_*)^2}{\mathrm{cov}(y_*)} - 1 \right) \mathbf{t}_2 \right) \tag{2.37}$$

with auxiliary variables

$$\mathbf{t}_1 = \sum_i (\mathbf{x}_i - \mathbf{x}_*)[\mathbf{k}_*]_i [\boldsymbol{\alpha}]_i \tag{2.38}$$

$$\mathbf{t}_2 = \sum_i [\mathbf{k}_*]_i \left(\sum_j (\mathbf{x}_i + \mathbf{x}_j - 2\mathbf{x}_*)[\mathbf{k}_*]_j [(\mathbf{K} + \sigma_{GP}^2 \mathbf{I})^{-1}]_{j,i} \right) \tag{2.39}$$

(III) For the continuous case of a regression, the expectation in equation (2.35) is an integral. It can be solved analytically yielding the following expression for the Fisher information matrix

$$\mathbf{J}(\mathbf{x}_*) = \frac{2\beta^2}{\mathrm{cov}(y_*)} \left(2\mathbf{t}_1 \mathbf{t}_1^\mathsf{T} + \frac{1}{\mathrm{cov}(y_*)} \mathbf{t}_2 \mathbf{t}_2^\mathsf{T} \right). \tag{2.40}$$

A more detailed derivation is given in Appendix A.2.

As for the case of discrete auxiliary data, it is advisable to regularize the Fisher information matrix in practice. This can be done by adding a small fraction of the identity matrix to prevent degeneration. This has the effect that dimensions which are irrelevant for the prediction are not completely neglected and, such, other dominant structure can be preserved as well.

2.6.3. Justification for discriminative DR

The aim of unsupervised DR is to project a high-dimensional data set $\mathbf{X}$ to a low-dimensional space while preserving the structure of the data. Such a definition is, however, ill-posed due to two reasons: 1) the term *structure preservation* is not a precise concept and can be formalized in different ways; 2) if the intrinsic dimensionality of $\mathbf{X}$ is higher then the dimensionality of the projection space d, a structure preserving embedding is not possible and the DR technique has to select which information to neglect and which to embed.

Aspect 1) is usually addressed in that moment that the user selects an existing DR technique: Each DR technique is based on a different formalization of structure preservation [26] and by selection of such a method, the user specifies a mathematical formalization of structure preservation. However, this choice mainly influences the visualization result for those parts of the data where the DR is not unique due to a high intrinsic data dimensionality, i.e. as soon as 2) is addressed, the diverse DR techniques have much less freedom to provide qualitatively different solutions for 1).

Hence, in this section, we will discuss how the concept of discriminative DR can help to alleviate aspect 2) (thus, to a large part also 1)): The goal of supervised DR

is to embed data $\mathbf{X}$ in Ξ such that structure which is relevant for the target variable $y = f(\mathbf{x}) \in \mathbb{R}$ is preserved. Usually, f is smoothly differentiable. Further, specifying y does usually only make sense if there is a clear relationship of $\mathbf{x}$ and y. The Fisher information relies on the tangent space of f by referring to its derivative $\partial f / \partial \mathbf{x}$. Directions which are orthogonal to this space, locally, do not affect the function value: this is obvious if one resorts to the Taylor expansion for linearization at a point $\mathbf{a}$:

$$f(\mathbf{x}) \approx f(\mathbf{a}) + (\mathbf{x} - \mathbf{a})^\top \frac{\partial}{\partial \mathbf{x}} f(\mathbf{a}). \tag{2.41}$$

Hence, the function value is not changed if directions orthogonal to the derivative are added to $\mathbf{a}$. If we assume a deterministic function, the probability $p(y|\mathbf{x})$ is peaked at the function value $y = f(\mathbf{a})$, i.e. also the Fisher matrix boils down to a matrix of rank one as determined by the tangential space of f at the considered data point, and path integrals vanish apart from the direction spanned by this derivative. This implies that, locally, a linear projection to a one-dimensional subspace, as given by $\frac{\partial}{\partial \mathbf{x}} f(\mathbf{a})$, is computed, i.e. only a one-dimensional subspace is relevant for prediction. This is easy to visualize; thus if f is a deterministic prediction, the problem of discriminative DR is well posed locally due to a locally only one-dimensional manifold, i.e. local manifold preservation is possible using DR.

For the full Fisher matrix, a probabilistic view is taken, i.e. $\mathbf{J}(\mathbf{a})$ is an infinite sum over rank one matrices induced by the tangent vector of $\log p(y|\mathbf{a})$ for all possible outcomes y at position $\mathbf{a}$. Provided the relation of $\mathbf{a}$ and y is random, this yields a full rank matrix, hence no regularization of the DR problem takes place. However, as soon as a relationship of y and $\mathbf{a}$ is present such as dimensions which are irrelevant for y, these do no longer contribute to the Fisher matrix, and DR techniques regard these directions as invariant. In the limit case of a deterministic relationship (or a peaked distribution which centres at around this value), the rank of the Fisher matrix approximates one, and the DR problem becomes more and more regularized.

2.6.4. Experiments

In this section, we evaluate our proposed scheme of computing discriminative dimensionality reduction mappings with various data sets. For each dat set, we apply the DR method t-SNE and project the data set to two dimensions. In addition, we compute distances on the Riemannian manifold as induced by the Gaussian Process based on the target variable. For this purpose we employ the straight line approximation (2.18) with $T = 4$ on the Riemannian manifold. The results in [123] have shown that $T = 4$ is a reasonable choice and that results don't improve with larger values (in [123], $T = 5$ equals $T = 4$ for our formalization). Finally, we apply t-SNE based on these supervised distances. Again, we will term the whole procedure *Fisher t-SNE*.

We utilize t-SNE since it is very popular and achieves often competitive results (see e.g. [125]). However, any approach that works on distances only, such as for instance MVU, Isomap, NeRV and so on, can be applied here.

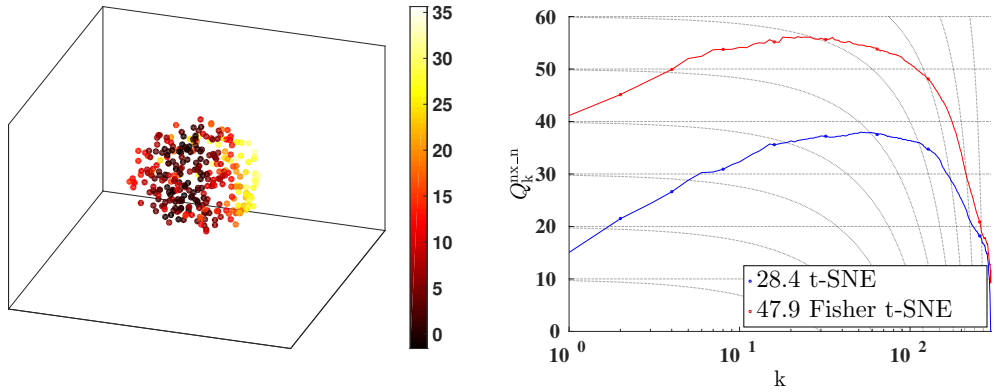


Figure 2.8.: The three-dimensional sphere data set (left). Evaluation of the preservation of neighborhoods for the two projections of this data set with t-SNE and Fisher t-SNE (right). Area under the curve value is shown in the legend.

The remaining of this section discusses an evaluation scheme for the experiments and shows results obtained on artificial and real life Benchmark data.

Empirical evaluation scheme

In the following, we compare unsupervised visualizations with discriminative visualizations for real-valued auxiliary data. For this purpose, we compute the prediction error in both projections, using a leave-one-out k-nearest neighbor regression scheme: the prediction of a datum $\mathbf{x}$ is the weighted average prediction value of the k nearest training points. We use $k = 5$ in the following. Further, we utilize the normalized root mean squared error

$$\text{nRMSE} = \frac{\sqrt{1/n \sum_i (y_i - f_{knn}(\mathbf{x}_i))^2}}{\text{std}(\mathbf{y})}$$

to measure the prediction error. Here, $\text{std}(\mathbf{y})$ refers to the standard deviation of $\mathbf{y}$. Since the dimensionality reduction technique we employ here is non-deterministic, we compute it ten times and evaluate the mapping in each run.

Additionally, in some cases it is useful to compare in how far neighborhoods from the original data space have been preserved in the projection. For this purpose, we utilize a scaled version of the average agreement $Q_k^{\text{nx\_n}}(\mathbf{X}, \Xi)$ from equation (2.3) [93], which measures in how far the current preservation for each neighborhood k is better than in a random mapping, as discussed in section 2.2.2.

Artificial data

As a first illustration, we utilize an artificially generated data set which is a three-dimensional filled sphere (Fig. 2.8). The target function is a second degree polynomial in the first two dimensions, i.e. feature three is irrelevant for predicting the target. The latter is encoded in the color of the points.

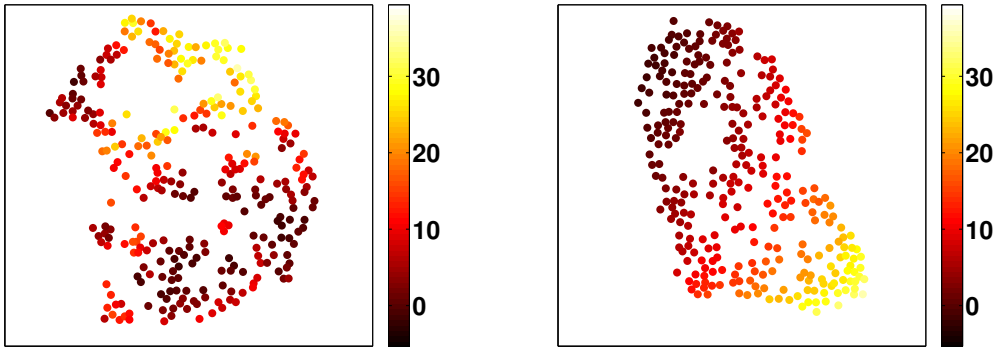


Figure 2.9.: Two projections of the sphere data set are shown: the unsupervised projection (left) and the supervised projection (right).

Obviously, the three-dimensional sphere cannot be embedded accurately in two dimensions such that all neighborhood relationships are preserved. However, since the target mapping depends only on two dimensions, finding an embedding which is accurate with respect to the supervised information in the data is possible.

We expect that an unsupervised projection has to make compromises between the three features equally, and such, would distort the continuous structure of the target variable. This would make a prediction task more difficult and would not show an accurate visualization of the original structure. A discriminative projection, however, will emphasize those dimensions that are relevant for the prediction and such yield a more accurate visualization with respect to the target function.

Fig. 2.9 (left) depicts an unsupervised visualization of the sphere data set and Fig. 2.9 (right) displays a supervised projection. As expected, the discriminative visualization shows the continuous change in the target variable while the unsupervised mapping does not. The numerical evaluation based on the nearest neighbor error yields the same conclusion: the average prediction error decreases from 0.357 to 0.099 in the discriminative visualization (see Table 2.2). In particular, the error is smaller than in the original data space.

For this artificial data set, we can employ an additional evaluation since we know that the first two dimensions contain the discriminative information. We compare the neighborhoods in the t-SNE and Fisher t-SNE projections to those defined in the first

Table 2.2.: Prediction errors in different data spaces using the nRMSE over 10 runs. The standard deviation is given in brackets.

	orig space	t-SNE	Fisher t-SNE
sphere	0.255	0.357 ($\pm$0.025)	0.099 ($\pm$0.013)
housing	0.440	0.471 ($\pm$0.004)	0.207 ($\pm$0.002)
diabetes	0.785	0.814 ($\pm$0.012)	0.506 ($\pm$0.006)

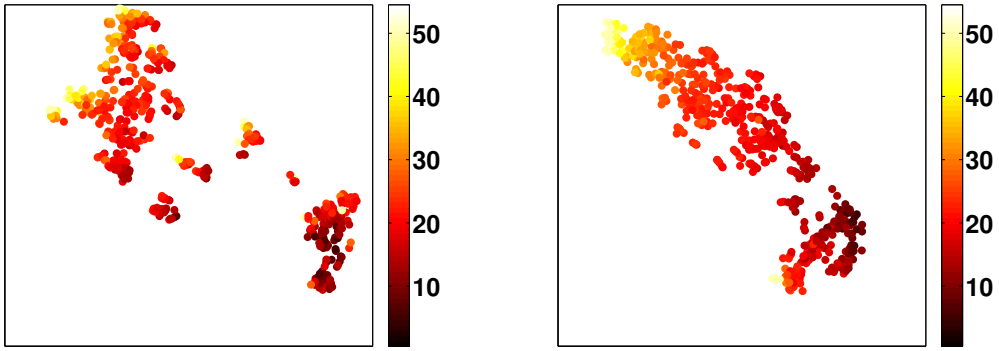


Figure 2.10.: Two embeddings showing the housing data set: unsupervised t-SNE embedding (left) and discriminative Fisher t-SNE embedding (right).

two dimensions of the data set. The resulting logarithmically scaled curve in Fig. 2.8 (right) shows the values $Q_k^{\text{nx-n}}$, which measure how well the neighborhoods of size k have been preserved. Fisher t-SNE achieves here superior results, as well.

Real world benchmark data

In the following, we employ two real world data sets for the evaluation of our approach:

- The housing data set [76] consists of 506 instances that describe houses in the suburbs of Boston. The goal is to predict the housing values based on 13 features. The data set can be obtained from the UCI Machine Learning Repository [45].

- The diabetes data set consists of 10 features for 442 patients. The goal here is to predict a measure of the diabetes progression one year after recording the features. This data set has been used in the original LARS paper [41].

Similarly as in the previous section, we compute unsupervised t-SNE projections and supervised mappings which are based on Fisher distances and calculate prediction rates. Repeating this procedure ten times yields the average prediction errors shown in Table 2.2. In both cases, the error in the discriminative case is lower and the standard deviation is small.

Exemplarily, an unsupervised and a discriminative embedding of the housing data set is depicted in Fig. 2.10. While the global structure seems to be similar in both projections, differences can be observed locally in particular for the target values around 40.

Fig. 2.11 shows a t-SNE and a Fisher t-SNE projection of the diabetes data set. Again, the global structure agrees while many differences can be observed, locally.

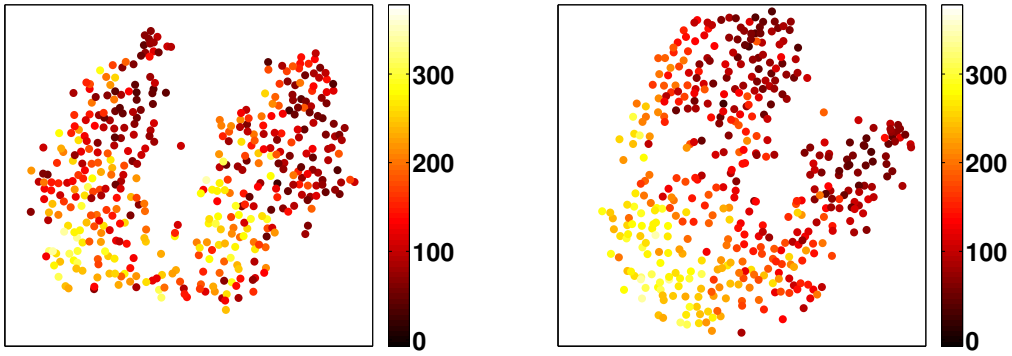


Figure 2.11.: Two embeddings depicting the diabetes data set: t-SNE embedding (left) and Fisher t-SNE embedding (right).

Interpretation of feature selection results

One possible application of Fisher t-SNE in the context of regression is to visualize the correlation of single features with the target variable. For instance if a feature selection approach has found particular relevant features, it might be of interest to analyze in which regions of the data space a feature is particularly useful to predict the target and in which this is not the case.

For the purpose of a short illustration, we utilize the diabetes data set, where it is known that feature 3 is particularly useful for prediction while this is not the case for feature 1 [41]. Fig. 2.12 shows three times the same Fisher t-SNE projection, where the targets (left), feature 1 (middle) and feature 3 (right) are used for labelling.

The bottom left image in Fig. 2.12 shows structural information but it is clearly not helpful in predicting the target. The bottom right image, however, shows a similar gradient as the top image in the most regions (except the top left and top right part of the data).

2.6.5. Conclusion

We have investigated the question of how to shape dissimilarity based dimensionality reduction techniques for data visualization according to auxiliary information in case the latter is real valued. Building on the very successful strategy as introduced in [163, 56] and discussed in section 2.3, we have proposed to change the metric in the data space according to the given information based on the Fisher information. This results in an intrinsically low-dimensional manifold for which dimensionality reduction is much easier (and well-posed) based on standard methods such as t-SNE. Unlike approaches [163, 137] focusing on auxiliary discrete label information as discussed in section 2.4, we have addressed the setting of real valued data and we have proposed a very robust scheme how to approximate the Fisher metric based on a Gaussian process model of the data. This has the advantage of a highly flexible scheme for which the

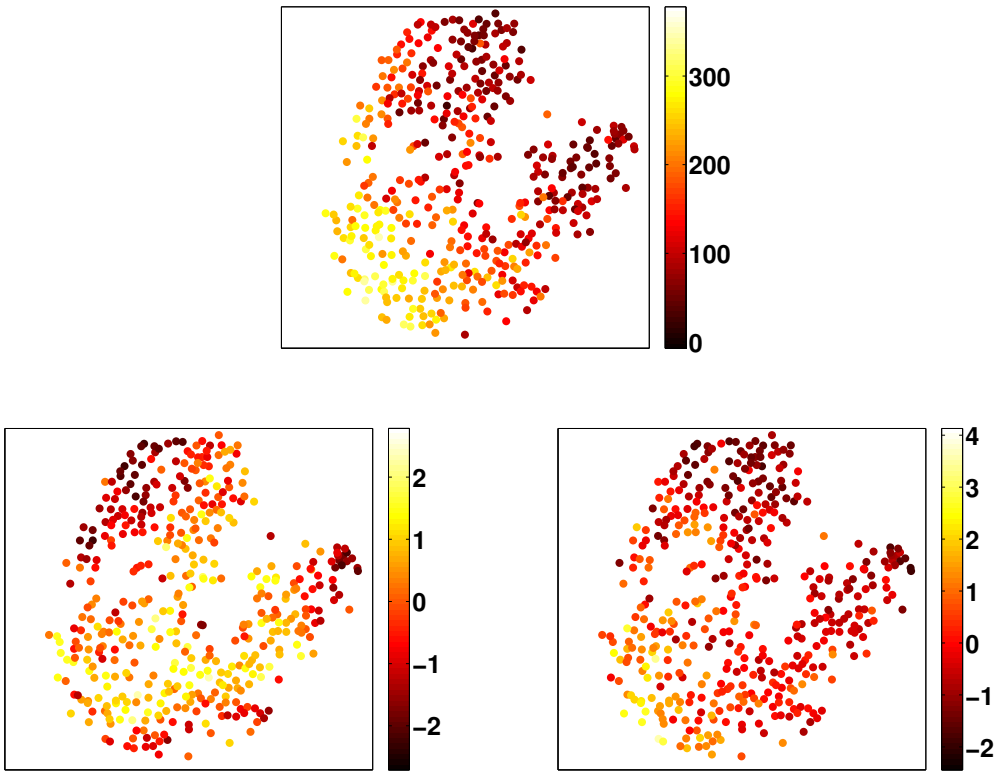


Figure 2.12.: A Fisher t-SNE projection of the diabetes data set with different colorings according to the target variable (top), feature 1 (bottom left) and feature 3 (bottom right).

mathematical derivative and the integral can efficiently be computed. Based on artificial and real life benchmarks, the superiority of the proposed approach in contrast to a mere unsupervised dimensionality reduction becomes apparent again: the proposed method opens a very intuitive way to shape the ill-posed problem of data visualization according to the aspects which are considered as relevant by the applicant. Since real labels constitute a universal set in which to embed features or measurements, the technology comes with a widespread applicability.

2.7. Discussion

In this chapter, we have proposed three contributions to DiDi methods based on the Fisher metric which enable to apply these techniques in different complex scenarios. Nevertheless, there are quite a few interesting aspects for further research: So far, minimum path integrals of the Fisher metric have been approximated by sampling along a straight line. Albeit this crude approximation yields good results, the question occurs whether an explicit analytical solution or a variational approximation is possi-

ble in the context of GPs or for approximations by e.g. piecewise constant or piecewise linear functions. Further, in particular for big data, the seamless integration of its computation into the Barnes Hut approximation of the dimensionality reduction method [177, 157] itself could be of great potential as regards computational complexity. We have proposed a scale up by subsampling and a subsequent extension towards further data by means of the kernel t-SNE technology as introduced in section 2.2.3; this is particularly suited in the case of streaming data which arrive over time, but it bears the risk of loosing information present in the full data. Here, speed-up such as offered by Barnes-Hut enables a better controllable approximation which is tailored to t-SNE and similar technologies. Apart from these algorithmic issues, the question occurs whether the proposed method induces auxiliary semantics which could be of use in an interactive data exploration scheme. The Fisher information constitutes a natural way to quantify the local relevance of a given data feature for the task at hand. This information is usually of high relevance for the user, and its value could be integrated into the data visualization scheme relying on a suitable display of this information. Here, ideas from information visualization as well as according studies to investigate their efficiency and effectiveness could be very valuable.

Having investigated extensions of discriminative dimensionality reduction towards out-of-sample extensions, non-vectorial data representations in the form of kernels, and real-valued auxiliary information, we now turn to a first application domain where DiDi plays a major role: the visualization of classifiers together with the decision boundary instead of a finite data set only.

Chapter 3.

Visualization of functions in high-dimensional spaces

Chapter overview *This chapter presents a novel framework to visualize a high-dimensional function together with a data set in two dimensions. We employ this concept to visualize classification and regression models, thus, enabling us to directly inspect characteristics of the trained model such as overfitting/underfitting behavior or the handling of multi-modal data. We demonstrate this approach for various classification and regression models and show that it highly benefits from the use of discriminative dimensionality reduction.*

Parts of this chapter are based on:

[J15b] A. Schulz, A. Gisbrecht, and B. Hammer. Using Discriminative Dimensionality Reduction to Visualize Classifiers. *Neural Processing Letters*, 42(1): 27–54, 2015.

[C15d] A. Schulz, and B. Hammer. Visualization of regression models using discriminative dimensionality reduction. In *CAIP 2015*, pages 437–449, 2015.

3.1. Motivation

An increasing complexity of data as concerns its size, dimensionality, or heterogeneity poses strong challenges on automated data analysis. Often, it is no longer possible to specify a dedicated learning task in advance. Rather, complex settings cause the need of an interactive data analysis: humans interactively process and interpret large, heterogeneous, and high-dimensional data sets, specifying the learning goals and appropriate data analysis tools based on the obtained findings [171, 69, 144, 132]. In this realm, interpretability of the models and data visualization play a major role since they offer an intuitive interface to the data and its analysis tools for the human practitioner [162, 94, 138]. Hence a trained classifier is no longer judged by its classification accuracy only, rather, the question moves into the focus based on which rationale the classifier makes it decision, what are problematic regions of the classification task where refinement would be valuable, and which data correspond to outliers or noise.

Possible remedies to the challenge of model interpretability are offered by relevance learning, feature selection techniques, or sparse model descriptions, for example [117, 162, 138, 141, 67, 11]. Roughly speaking, such techniques aim for a model shape which is directly interpretable by humans e.g. by means of sparsity. Further, visualization

plays a major role, since it addresses one of the most powerful senses of humans.

Visualization of data constitutes a well-investigated research topic with a plethora of different visualization techniques having been proposed in the machine learning context. Besides classical methods such as linear projections and nonlinear extensions, a variety of (often non-parametric) dimensionality reduction (DR) techniques has been proposed in the last decade, such as t-SNE, NeRV, or MVU, see section 2.1 for references on DR techniques. Often, however, these methods are used to visualize a given data set in two dimensions only. Such, they do not yet answer the question how to visualize the relation of these data in connection to a given supervised model, such as a classification or regression model. The possibility to also visualize decision boundaries as provided by a given classifier or the learned function of a regression model would allow us to extract information beyond the mere accuracy of the model addressing questions such as: is the model particularly complex in certain regions of the data space, is it too simplistic in others, how are noisy regions and outliers treated, how does the model extrapolate, is the data multimodal, etc.

In this chapter we propose a framework to visualize high-dimensional functions together with a data set in two dimensions. In particular, we will apply it to decision boundaries of arbitrary classification models and to prediction functions of arbitrary regression models. Such a visualization then allows us to answer questions such as formulated above. Since a visualization of these functions basically displays the central functionality of the respective models, we will sometimes refer to our concept as classifier and regression model visualization.

Generally speaking, the general framework for classification and regression model visualization, as proposed in this chapter, relies on an identification of a given data manifold and a two-dimensional projection. Note that, for two-dimensional input data, function or classifier visualization constitutes a classical tool of data analysis: the plane is sampled, and its function values are directly displayed at the corresponding position in the plane in terms of a colormap or contour plot. A bijective mapping between the original data manifold and a low-dimensional projection enables us to directly transfer this procedure: we can identify points in low and high dimensions, and display the color or contour of these points in the low-dimensional space according to their function value in the high-dimensional space.

This naive approach, however, has a few drawbacks: (i) Many powerful DR methods do not provide an explicit mapping, rather they provide a non-parametric projection of the given data points only. Here, we will employ the parametric out-of-sample extension proposed in section 2.2, if necessary. (ii) It is infeasible to sample the usually high-dimensional feature space; still we have to somehow obtain an explicit description of the high-dimensional function, e.g. of the decision boundaries of an arbitrary classifier. We solve this problem by a trick: we sample in the low-dimensional projection space rather than the feature space itself, and use the inverse projection of these sampled data to determine an explicit description of the high-dimensional function in the data manifold. (iii) Unless the data manifold is intrinsically two-dimensional,

however, there cannot exist a bijection of the data manifold and a low-dimensional projection, hence no valid back-projection. More generally, the question what to visualize in a reasonable way is not clear due to the usually high data dimensionality. Hence the task of classifier or regressor visualization is essentially ill-posed. We will rely on discriminative DR to circumvent this problem.

More precisely, we will point out the necessity to integrate auxiliary information to the DR technique to make the DR problem well-posed. For classifier visualization, we do not want to visualize all aspects of the data, rather we are interested in the positioning of the data as concerns the class boundaries. For the visualization of regression models we are not interested in areas of the data space where the regression function is constant but where it shows interesting variation. As discussed in chapter 2, there exists a very natural way to enhance many DR techniques with auxiliary information: instead of the original data and its underlying distance measure, we rely on a distance measure which is induced by the Fisher information metric for the given auxiliary variable. This way, those aspects of the data are emphasized which are relevant for the given classifier or the regression model, rather than e.g. directions parallel to the classifier's decision boundaries. As discussed in section 2.6.3, the Fisher information matrix is low-dimensional in the regression case, making the function under the Fisher metric locally low-dimensional, as well. Similar holds for the classification case, because the decision boundary correspond to a one-co-dimensional topological manifold. Hence the data set measured in the Fisher metric, which essentially restricts to directions orthogonal to the boundary, is locally approximately one dimensional (in the vicinity of class boundaries), and the task to visualize such data in two dimensions is well defined.

We will elaborate on this issue in the following and demonstrate the beneficial effect of taking auxiliary information into account. Actually, there exist two different reasonable choices for the auxiliary information: the ground truth which is the auxiliary information provided by the data, and the prediction which is provided by the trained classification or regression model. In particular for models with low accuracy, a scenario where inspection might be particularly interesting, these predictions do not coincide. In such settings, we would like to 'see' what causes the problems of the model. We will discuss that both possible choices provide different visualizations of the models, focusing on different aspects of the setting and different insights into the model behavior. We will demonstrate this aspect in the following in examples. In summary, a powerful high-dimensional function visualization framework results which we will test for different classifier types, different regression models and for different DR techniques, including t-SNE, SOM, GTM, and MVU.

Moreover, we identify typical user tasks that can be performed with the help of our proposed method. These include for the case of classifier visualization the following.

1. Is there multimodality in the data, i.e. are there certain classes which fall into multiple modes and how does the classifier handle them?

2. How does the classification model deal with potential outliers in the data?

3. Is there overlap in the data and how do the class boundaries look in those regions?
4. How complex are the class boundaries of the trained model? Do they potentially overfit the data?
5. If the model contains interpretable components such as data prototypical instances: What is their location in relation to the data and how do they contribute to the class boundary?

In the case of the visualization of regression models, the user tasks can be slightly different. We summarize them in the following.

1. How complex is the learned function? Does it overfit/underfit some regions?
2. Is the data multi-modal, i.e. are clusters present in the data and how does the regression model deal with those? What is the prediction for the regions in-between the clusters?
3. Are specific aspects of the selected model visible (such as local linear functions) and are these suited for the data at hand.
4. Are there potential outliers in the data and how does the model treat these?

We address these user tasks in the sections 3.5 and 3.6.

Literature overview for the visualization of classifiers At present, visualization in the context of classifiers is rather limited: visualization is often restricted to the training procedure, e.g. providing interfaces to set certain parameters or to inspect the area under the curve (AUC) results [63]. Other methods analyze the class topology in a projection space [37] and in the original data space [6]. There exists relatively little work to visualize the underlying classification function itself, including interactive tour methods [29], nomograms [73], linear projection techniques on top of the distance to the decision boundary [128], or graphs emphasizing those regions where class affiliation changes [106]. Very few nonlinear techniques exist, one notable approach being proposed for the visualization of support vector machine (SVM) using self-organizing maps (SOM), resulting in a technique dubbed support vector machine visualization (SVMV) [170].

Literature overview for visualization of regression models Besides approaches to judge the quality of trained regression models with quantitative estimates [28], there exists only little work which aims to visualize the regression function itself. For the special case of Decision Trees, a direct inspection is possible through the special tree structure of the model. However, these models can get unclear with increasing size and data complexity. More general approaches such as Breheny and Burchett [21] try to analyze the relationship between the target and a single explanatory variable

by visualizing the predictions of the model for different values of this variable while keeping the others fix. However, this approach treats the explanatory variables independently (or a small subset simultaneously) and thus cannot find information that is present in many dependent features.

In this chapter, we propose a general approach which generalizes the standard display of two-dimensional classifiers or functions in terms of contour plots towards high-dimensional data by means of discriminative dimensionality reduction techniques.

3.1.1. Scientific contributions and structure of the chapter

This chapter presents the following core contributions.

Framework for visualizing classification and regression models In section 3.4, we propose a general framework for the visualization of classification and regression models enabling to visualize any such model that provides a real valued output (such as the certainty measure of the decision, in case of a classifier). We demonstrate this for a support vector machine (SVM), support vector regression (SVR), learning vector quantization (LVQ) and a decision tree model.

Influence of DiDi In section 3.5, we highlight the necessity and investigate the effect of using discriminative DR in this context.

User tasks demonstration In sections 3.5 and 3.6, we present how the identified user tasks can be addressed with our proposed framework.

The remainder of the chapter is structured as follows: Section 3.2 describes the DR techniques investigated for the task of high-dimensional function visualization. Afterwards, section 3.3 discusses the task of computing an inverse DR mapping, thereby dealing with the fact that DR is typically many to one. Section 3.4 describes our main approach for the visualization of classifiers and regression models, by employing the concepts of DR and inverse DR. Finally, we demonstrate and evaluate the proposed approach by visualizing classification models in section 3.5 and regression models in section 3.6. Here, we also show exemplarily how the proposed user tasks can be addressed. Section 3.7 concludes the chapter.

3.2. Dimensionality reduction techniques

As already mentioned before, dimensionality reduction techniques are concerned with the following problem: given data points $\mathbf{x} \in \mathcal{X} = \mathbb{R}^D$ in a high-dimensional feature space, how to map these points to low-dimensional counterparts $\pi(\mathbf{x}) = \boldsymbol{\xi} \in \Xi = \mathbb{R}^2$ in the two-dimensional plane such that as much structure as possible is preserved. As described in the recent overview [53] for example, one can distinguish parametric and non-parametric DR techniques.

Parametric techniques specify a functional form $\pi_{\text{pm}} : \mathcal{X} \to \Xi, \mathbf{x} \mapsto \boldsymbol{\xi} = \pi_{\text{pm}}(\mathbf{x})$ (we employ the subscript $_{\text{pm}}$ to emphasize that the mapping is parametric) with

free parameters which determine the form of the mapping. Given a set of examples $\mathbf{x}_1, \ldots, \mathbf{x}_N$, training takes place by optimizing these parameters such that the examples are mapped as accurately as possible. Popular methods include:

- *Principle component analysis (PCA)* defines $\pi_{\mathrm{pm}}(\mathbf{x}) = \mathbf{W}^\mathsf{T}\mathbf{x}$ as a linear mapping with $\mathbf{W} \in \mathbb{R}^{D\times 2}$. The linear parameters $\mathbf{W}$ are determined such that the squared reconstruction error of the given data is minimized, resulting in an eigenvalue problem with an explicit solution. Due to the particularly simple form, an approximate inverse is offered by the mapping $\pi_{\mathrm{pm}}^{-1} : \boldsymbol{\xi} \mapsto \pi_{\mathrm{pm}}^{-1}(\boldsymbol{\xi}) = \mathbf{W}\boldsymbol{\xi}$.
- One popular non-linear alternative is offered by the *self-organizing map (SOM)*, which maps the data to a two-dimensional regular grid (consisting of nodes $\mathbf{c}_k$) by means of a winner-takes-all function, where each position j in the grid is associated with one position $\mathbf{w}_j \in \mathbb{R}^D$ in the feature space. Training takes place by Hebbian learning, thereby also respecting the neighborhood of the lattice. This way, a locally constant projection of the data to two dimensions, the lattice position of the winner, is defined: $\pi_{\mathrm{pm}}(\mathbf{x}) = \mathbf{c}_k$ with $k = \arg\min_j d(\mathbf{x}, \mathbf{w}_j)$. By means of local interpolation, this mapping can easily be turned into a smooth function. By construction, an inverse mapping is offered by mapping a position j in the lattice to the position $\mathbf{w}_j$ of the associated place in the feature space: $\pi_{\mathrm{pm}}^{-1}(\boldsymbol{\xi}) = \mathbf{w}_k$ with $k = \arg\min_j d(\boldsymbol{\xi}, \mathbf{c}_j)$. Again, this simple function is locally constant, but can easily be turned into a smooth mapping by means of local interpolation.
- We will also consider the *generative topographic mapping (GTM)* as a probabilistic counterpart of SOM. Essentially, GTM relies on data being generated by a constraint mixture of Gaussians. The centers of the Gaussians are generated by a smooth mapping from regular lattice positions in a two-dimensional latent space which can be used for data visualization. GTM training can be derived from a maximization of the data log likelihood function. Due to its probabilistic modeling which allows to compute probabilities of lattice points having generated a given data, a smooth mapping of data to its low-dimensional projection $\pi_{\mathrm{pm}}(\mathbf{x}) = \sum_k \mathbf{c}_k p(\mathbf{c}_k|\mathbf{x})$ and vice versa $\pi_{\mathrm{pm}}^{-1}(\boldsymbol{\xi}) = \sum_j \mathbf{w}_j \phi(\boldsymbol{\xi})$ (where the basis function ϕ are often Gaussian kernels with predefined centers) is directly provided by GTM.

In contrast to these parametric techniques, non-parametric mappings rely on a mapping of a given set of data $\mathbf{x}_i$ to their low-dimensional counterparts $\boldsymbol{\xi}_i$ only, but no explicit functional form $\pi_{\mathrm{pm}} : \mathcal{X} \to \Xi$ is priorly specified. Training takes place by tuning the projections $\boldsymbol{\xi}_i$ such that a certain criterion is optimized: usually, the structure in the data space as defined by $\mathbf{x}_i$ and the structure of the projections $\boldsymbol{\xi}_i$ are measured and compared using some suitable cost function. An overview about a generic formalization of different popular non-parametric DR techniques as cost function optimization can be found in [26]. We will exemplarily investigate the following four popular techniques:

- The goal of *Multidimensional scaling (MDS)* is to embed the data such that the distances in $\mathcal{X}$ and in Ξ agree. If these distances are Euclidean, MDS is equivalent to PCA. However, other metrics can be integrated directly.
- *Isomap* is based on the objective to preserve distances in the data space and the projection space as measured in a least squares error. Thereby, the distances in the original data space are taken along the data manifold as so-called geodesic distances. Since the exact manifold is not available, a simple numeric approximation scheme is taken: local neighborhoods of a given data point to its closest k neighbors are approximated by the euclidean distance; on a global scale, shortest paths in this neighborhood graph are considered.
- *Maximum variance unfolding (MVU)* relies on a similar idea, by first constructing a local neighborhood graph connecting every point to its k closest exemplars. Then, projection takes place by unfolding the data as much as possible in two dimensions (i.e. maximizing its covariance) thereby respecting the neighborhood structure of the constructed graph.
- *T-distributed stochastic neighbor embedding (t-SNE)* has already been detailed in section 2.2.1. For convenience, we recall the core concept in the following. T-SNE defines local neighborhoods in a probabilistic sense by using Gaussians based on pairwise distances in the feature space and student-t distributions induced by euclidean distances in the projection space. Training takes place by a minimization of the error in between these distributions as measured by the Kullback Leibler divergence. Unlike MVU, the resulting cost function can have local optima resulting in different possible visualizations. See 2.2.1 for more details.

With the exception of PCA, all these methods rely on distances to the data only and, hence, can be equipped with the Fisher metric, see section 2.3. Employing one of these methods together with the Fisher metric renders them discriminative and, hence, we will refer to such a combination with the method name preceded by the prefix *Fisher*.

3.3. Inverse dimensionality reduction

In addition to projecting data down to two dimensions, we will also require a mapping to project data back to the original data space. However, note that in general a direct inversion of a projection mapping $\pi : \mathcal{X} \to \Xi$ is not possible since the projection π is usually many to one. Hence, we are interested in finding a mapping $\pi^{-1} : \Xi \to \mathcal{X}$ such that

$$\pi(\pi^{-1}(\boldsymbol{\xi})) = \boldsymbol{\xi} \tag{3.1}$$

and π^{-1} maps into the data manifold. Nevertheless, we will use the term *inverse dimensionality reduction* for this concept, similarly as the term inverse kinematics is used in robotics for finding an inverse mapping of a many to one map (this is often the case e.g. for robot arms having an elbow up and elbow down solution).

Many parametric techniques nevertheless provide explicit inverse mappings which find a suitable inverse of the projections to the data manifold, such as discussed for PCA, SOM, and GTM above. For non-parametric mappings a picewise linear mapping is developed in [40], where the parameters have to be recomputed for each point. We propose a similar trick as before which is based on a similar interpolation idea as in [40] but having a global analytical functional form which is smooth.

We assume that points $\mathbf{x}_i \in \mathcal{X}$ and projections $\pi(\mathbf{x}_i) = \boldsymbol{\xi}_i \in \Xi = \mathbb{R}^2$ are available. For an inverse projection, we assume the following functional form

$$\pi^{-1} : \Xi \to \mathcal{X}, \boldsymbol{\xi} \mapsto \frac{\sum_j \boldsymbol{\beta}_j k_j(\boldsymbol{\xi}, \boldsymbol{\xi}_j)}{\sum_l k_l(\boldsymbol{\xi}, \boldsymbol{\xi}_l)} \tag{3.2}$$

where $\boldsymbol{\beta}_j \in X$ are parameters of the mapping and $k_j(\boldsymbol{\xi}, \boldsymbol{\xi}_j) = \exp(-0.5\|\boldsymbol{\xi} - \boldsymbol{\xi}_j\|^2/(\sigma_j^{\xi})^2)$ constitutes a Gaussian kernel with bandwidth determined by σ_j^{ξ}. The bandwidth is determined in the same way as in the kernel t-SNE approach, see section 2.2.3. Summation is over a random subset Y' of the given data projections $\boldsymbol{\xi}_i = \pi(\mathbf{x}_i)$, or over codebooks resulting from a previously run vector quantization on the $\boldsymbol{\xi}_i$.

One particular problem is given by the fact that the inverse mapping π^{-1} of π is not well defined if obtained from minimizing the standard Euclidean error: Since the intrinsic data dimensionality is usually larger than two, the inverse $\mathbf{x}$ of a given projection $\boldsymbol{\xi}$ is ambiguous. Data dimensions which are not relevant for the projection π can be treated arbitrarily. Moreover, equation (3.1) cannot be used directly for optimization in case that π is non-paramteric. Thus, a challenge is the task to find a suitable inverse projection of π which tolerates such invariances.

We solve this problem by optimization of the following costs with respect to the parameters $\boldsymbol{\beta}_j$

$$E = \sum_i \left(d_1 \left(\mathbf{x}_i, \pi^{-1}(\boldsymbol{\xi}_i) \right)^2 \right) = \sum_i \left(\mathbf{x}_i - \pi^{-1}(\boldsymbol{\xi}_i) \right)^{\mathsf{T}} \mathbf{J}(\mathbf{x}_i) \left(\mathbf{x}_i - \pi^{-1}(\boldsymbol{\xi}_i) \right) \tag{3.3}$$

where the matrix $\mathbf{J}$ refers to the Fisher information matrix, see section 2.3. In contrast to a standard Euclidean error function, this function has the advantage that those dimensions in $\mathcal{X}$ which are locally relevant for the classification are emphasized. Invariances of the projection π due to the given class labeling are tolerated in the inverse projection. We utilize the distance d_T (2.18) with $T = 1$ in order to save computational time. This local approximation works usually well since the points $\mathbf{x}_i$ and $\pi^{-1}(\boldsymbol{\xi}_i)$ will get close to each other in the course of optimization. Minimization of these costs with respect to the parameters $\boldsymbol{\beta}_j$ takes place by gradient descent.

3.4. General framework

In this section we present a new general framework for the visualization of classification and regression functions. We assume the following scenario: a data set including points $\mathbf{x}_i \in \mathcal{X}$ is given. Every data point is accompanied either with a label $l_i \in L$

belonging to a finite set of different labels L or with a target variable $y_i \in \mathbb{R}$. In addition, either a classification model $f : \mathcal{X} \to L$ or a regression model $f : \mathcal{X} \to \mathbb{R}$ has been trained on the given training set, such as a support vector machine or a decision tree. For a classifier, we additionally assume that the label $f(\mathbf{x})$ is accompanied by a nonnegative real value $r(\mathbf{x}) \in \mathbb{R}$ which scales with the distance from the closest class boundary. As an example, this could be the activation of a linear classifier such as SVM, or it could be the class probability if a probabilistic classifier such as robust soft learning vector quantization or a Bayesian classifier is considered. Note that most classifiers offer a natural way to equip the mere class output with a smooth value which correlates to the distance to the decision boundary. Since we do not assume a specific scaling of this output, any such value will do.

A visualization of the given data set and the trained model would offer the possibility to visually inspect the prediction results and to address user tasks as formulated in section 3.1. We propose a general framework to visualize a trained classification or regression model together with a given data set such as the training set of the model.

3.4.1. Naive approach

In the following, we assume that a nonlinear dimensionality reduction method is given. As already said before, a naive pipeline to directly extend common practice for two-dimensional classifier visualization to high dimensions could be like follows:

- Sample the full data space $\mathcal{X}$ by points $\mathbf{z}_i$.
- Project these points nonlinearly to two-dimensional points $\pi(\mathbf{z}_i)$ using some nonlinear dimensionality reduction technique.
- Display the data points $\pi(\mathbf{x}_i)$ and the contours induced by the sampled function $(\pi(\mathbf{z}_i), r(\mathbf{z}_i))$, the latter approximating the boundaries of the classifier or the prediction of a regression model.

This method, however, fails unless $\mathcal{X}$ is low-dimensional because of two reasons:

- Sufficiently sampling $\mathcal{X}$ requires an exponential number of points, hence it is infeasible for high-dimensional $\mathcal{X}$.
- It is impossible to map a full high-dimensional data set faithfully to low dimensions, hence topological distortions would be unavoidable.

The problem lies in the fact that this procedure tries to visualize the function in the full data space $\mathcal{X}$. It would be sufficient to visualize only those parts which are relevant for the given training data $\mathbf{x}_i$ and the underlying function behavior as measured using the Fisher metric.

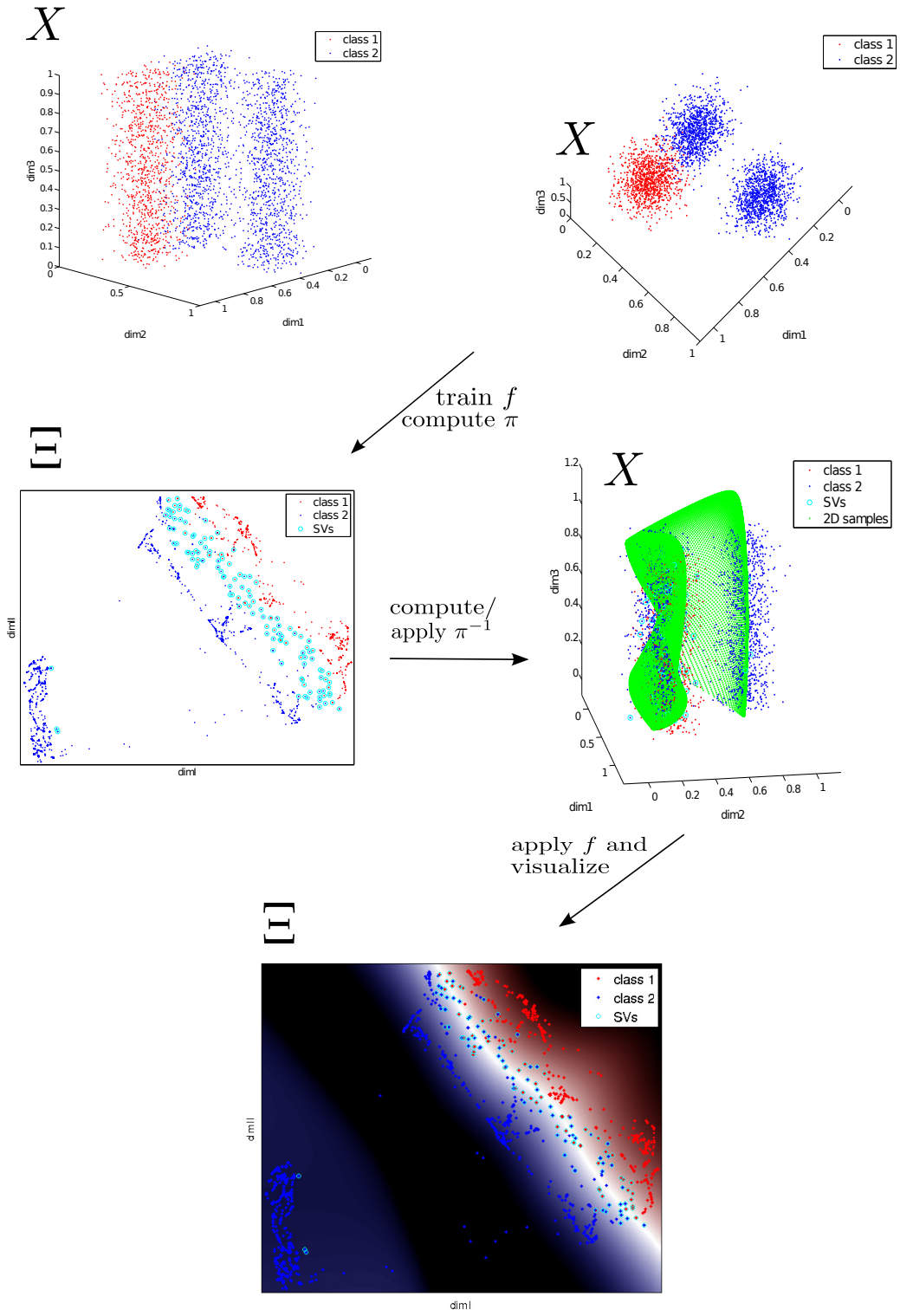


Figure 3.1.: Principled procedure how to visualize a given data set and a trained classifier. The example displays a SVM trained in 3D.

3.4.2. Main procedure

Therefore, we propose to sample in the projection plane instead of the original data manifold, and we propose to use a discriminative DR technique to make the problem of data projection well-posed (since the Fisher metric makes the data space locally low-dimensional the projection can find a compromise, at least locally). Together with the techniques presented in the last chapter, this leads to the following feasible procedure for visualization of high-dimensional functions:

Dimensionality reduction Project the data $\mathbf{x}_i$ using a nonlinear discriminative DR technique (for instance utilizing Fisher distances calculated with (2.18)) leading to points $\pi(\mathbf{x}_i) \in \Xi$.

Inverse dimensionality reduction Sample the projection space Ξ in a regular grid leading to points $\{\mathbf{z}'_i\}_{i=1}^n$. Determine points $\mathbf{z}_i$ in the data space $\mathcal{X}$ which are projected to these points $\pi(\mathbf{z}_i) \approx \mathbf{z}'_i$ by applying an inverse mapping π^{-1} (if not provided by the DR method, it can be obtained by optimizing (3.3)), relying on the Fisher metric to make it well posed.

Visualization Visualize the training points $\pi(\mathbf{x}_i)$ together with the given function which is induced by $(\mathbf{z}'_i, f(\mathbf{z}_i))$ as contours. In the case of classification, utilize also $(\mathbf{z}'_i, r(\mathbf{z}_i))$, where the function value r is provided by the classifier f. In this case, we depict $f(\mathbf{z}_i)$ as color values and $r(\mathbf{z}_i)$ as the intensity. For regression, it can be useful to plot $f(\mathbf{z}_i)$ over a third axis.

This corresponds to applying the function $f \circ \pi^{-1}$ for every position $\mathbf{z}'$ in the projection space. An illustration of this procedure is shown exemplarily in Figure 3.1 for classification and in Figure 3.2 for regression. More information on the data set used for the latter are detailed in section 3.6.

Unlike the naive approach, sampling takes place in $\mathbb{R}^2$ only and, thus, it is feasible. Further, only those parts of the space $\mathcal{X}$ are considered which correspond to the observed data manifold $\mathbf{x}_i$, i.e. the prediction function is displayed only as concerns these training data.

Parameter choices

The proposed framework is general in the sense that it allows to visualize a large set of functions, using any DR technique. Hence, there are a few choices which have to be made. These are illustrated in the following.

Visualized model The most obvious choice is to select the model or functions which should be visualized. Here, we will often demonstrate our approach using the support vector machine as a state of the art technique.

Discriminative mapping A principle choice which has to be made is whether the utilized DR technique should make use of auxiliary information or not. We will investigate this question in the experiments.

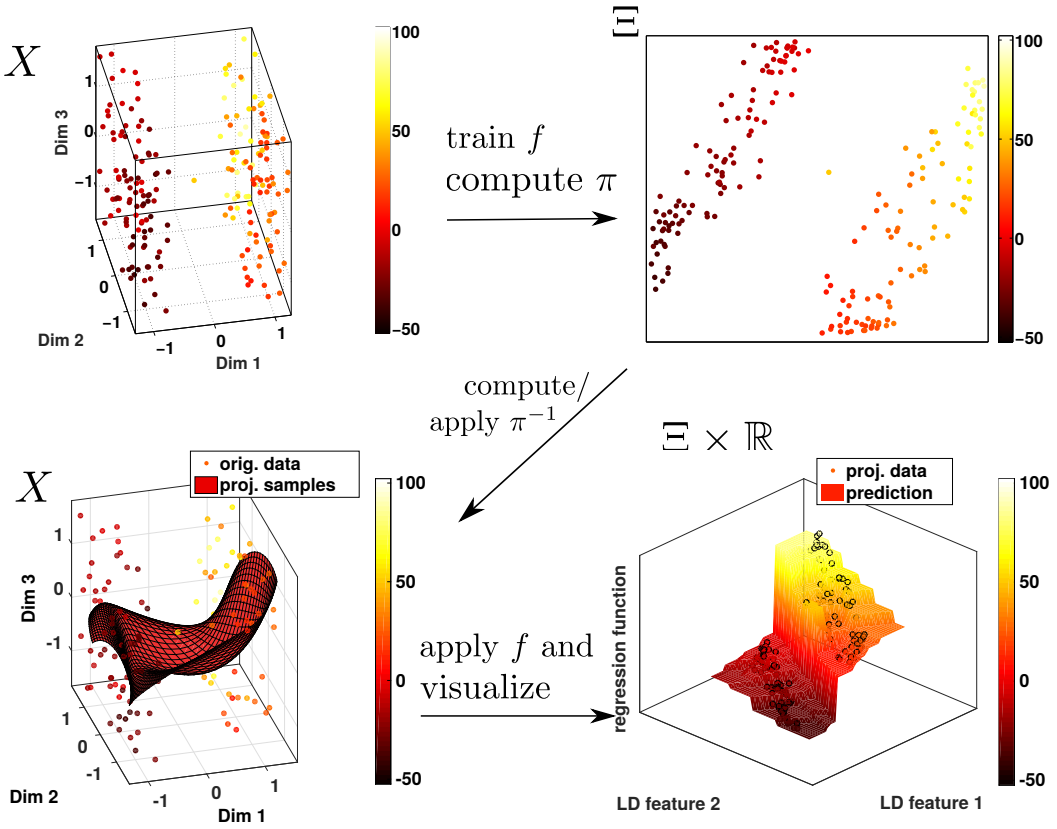


Figure 3.2.: Illustration of our proposed approach to visualize a regression model (in this case a Decision Tree).

Choice of DR In both supervised and unsupervised DR cases, there is a large pool of DR techniques from which one can be selected. We will evaluate their suitability for the task of the visualization of classification and regression models in the following. The method t-SNE will be a default choice.

Type of auxiliary information Two types of auxiliary data can be used in this context: the prediction as provided by the function f (i.e. the classification or regression function), or the auxiliary data directly from the training set, i.e. the ground truth. Depending on which information is used to determine π and its inverse π^{-1}, we obtain a visualization of the function which respects invariances of the underlying ground truth, or which respects invariances of the observed function, allowing different insights into its behavior as we will demonstrate in the following. Per default, we will refer to the original labels unless stated otherwise.

3.4.3. Evaluation

In the following we introduce a scheme which allows to evaluate the quality of the visualized function in the vicinity of the data.

The key idea is to compare the predictions of the visualized function for the visu-

alized data $(f \circ \pi^{-1} \circ \pi)(\mathbf{x})$ to the predictions of the original function for the original data $f(\mathbf{x})$.

In the case of a classification model, these two functions $f(\mathbf{x})$ and $(f \circ \pi^{-1} \circ \pi)(\mathbf{x})$ yield discrete values so a direct comparison is possible. Furthermore, since we assume that the classifier also provides a function r measuring the certainty of the model, we can also use this function for evaluation.

More formally, we evaluate the visualization of classification functions with the first two measures and the visualization of regression models with the third measure.

1. Compute the accordance of the classifications in the projection and in the original data space by calculating the percentage of points $\mathbf{x}^i$ for which $(f \circ \pi^{-1} \circ \pi)(\mathbf{x}^i)$ and $f(\mathbf{x}^i)$ agree.

2. Compute the accordance of the certainty values by computing the Pearson correlation between $(r \circ \pi^{-1} \circ \pi)(\mathbf{x}^i)$ and $r(\mathbf{x}^i)$:

$$\frac{\mathbb{E}\left\{(r(\mathbf{x}) - \mathbb{E}\left(r(\mathbf{x})\right)) \cdot \left((r \circ \pi^{-1} \circ \pi)(\mathbf{x}) - \mathbb{E}\left((r \circ \pi^{-1} \circ \pi)(\mathbf{x})\right)\right)\right\}}{\sqrt{\mathbb{E}\left\{\left(r(\mathbf{x}) - \mathbb{E}\left(r(\mathbf{x})\right)\right)^2\right\} \cdot \mathbb{E}\left\{\left((r \circ \pi^{-1} \circ \pi)(\mathbf{x}) - \mathbb{E}\left((r \circ \pi^{-1} \circ \pi)(\mathbf{x})\right)\right)^2\right\}}} \tag{3.4}$$

3. Compute the accordance of the predictions by computing the Pearson correlation between $(f \circ \pi^{-1} \circ \pi)(\mathbf{x}^i)$ and $f(\mathbf{x}^i)$, similarly as in equation (3.4).

These criteria do not measure in how far π^{-1} is the exact inverse of π. Obtaining an exact inverse mapping is impossible for most data sets. Instead, these measures evaluate the suitability of π^{-1} with respect to f (and also r in the classification case), i.e. errors along directions where f doesn't change are not accounted as such. This way, only directions in the data space are considered which are relevant for the prediction. Or more generally speaking, these measures compare in how far the induced function $(f \circ \pi^{-1})$ and the original function f agree on the pairs $(\mathbf{x}_i, \xi_i)$ as provided by π.

For the computation, we utilize only those points $\mathbf{x}$ which were not utilized to train the mapping π^{-1}. Further, we approximate $(f \circ \pi^{-1} \circ \pi)(\mathbf{x})$ (and for classification also $(r \circ \pi^{-1} \circ \pi)(\mathbf{x})$) by the prediction value of the closest sampled point $\mathbf{z}'$ of $\pi(\mathbf{x})$, simply because we have already computed $f \circ \pi^{-1}$ for these points.

3.5. Experiments with classification functions

In this section we demonstrate our approach for various data sets and scenarios. In the first experiments, we exemplarily visualize SVMs while later we also apply our approach to probabilistic LVQ models and classification trees.

In 3.5.1, we utilize two data sets addressing the user cases 2 and 4 and we investigate the influence of DR techniques on the visualization of classifiers. First, we apply PCA (being the most simple and straight forward method) and show it's limitations. Further, we compare the SOM (suggested in the literature [170]) to non-parametric

projections. In section 3.5.2, we perform a sanity check by visualizing classifiers based on different labellings of the same data. Furthermore, we also address the previously specified user cases 1 and 3. In section 3.5.3 we empirically analyze the effect of including supervised information and in section 3.5.4 we consider two types of supervision: given by the original labeling and by the labels assigned to by the classifier. Additionally, we investigate properties of the prototype based classifier, thus addressing user case 5. In the last section 3.5.5 we visualize two other classifiers: a classification tree and a Robust Soft LVQ model.

Now follows a short description of the classifiers we utilize in our experiments.

- The Support Vector Machine (SVM) [161] trains a maximal margin linear classifier in feature space. The decision function has the form $f(\mathbf{x}) = \text{sign}\left(\mathbf{w}^\mathsf{T}\phi(\mathbf{x}) + b\right)$, where $\mathbf{w}$ and b are optimized by the method. For the SVM, we can directly compute the distance from the decision boundary by $r(\mathbf{x}) = (\mathbf{w}^\mathsf{T}\phi(\mathbf{x}) + b)/\sqrt{\mathbf{w}^\mathsf{T}\mathbf{w}}$.

  Originally, the SVM solves only two-class problems. If more classes are available we employ a "one versus one" classification scheme (i.e. training a two-class SVM for each pair of two classes) with a subsequent majority vote for classification. For this approach, the class boundaries of the resulting SVM mostly coincide with the boundaries of the two-class SVMs, which is not the case for the "one versus all" scheme (see [87] for more details). Hence, in the case of more then two classes we specify the overall value $r(\mathbf{x})$ to be the minimum distance of $\mathbf{x}$ to the class boundary of each two-class SVM containing the class of $\mathbf{x}$. The "one versus one" scheme is also implemented in the LIBSVM toolbox [30] which we utilize in the following.

- The Robust Soft LVQ (RSLVQ) classification scheme [142] learns a prototype based probabilistic model for the data such that the likelihood of correct classification is optimized. A Gaussian mixture is employed as the probabilistic model, which directly provides probability estimates for $r(\mathbf{x})$.

- Classification Trees divide the input space into several regions, thereby using axis aligned decision boundaries. They typically work in a greedy way, subdividing regions if these contain too many points form different classes. For this splitting step of cells we use the Gini index. See [84] for an review of Classification Trees. A probabilistic output for the certainty of the classification can be provided using the distribution of data points inside such cells.

3.5.1. Toy data examples with different DR mappings

We utilize two three-dimensional artificial data sets in order to provide an example for our approach and to demonstrate the user tasks 1,2 and 4 as defined in the introduction. Both data sets consist of two classes and are shown in Fig. 3.3. For both data sets we train Support Vector Machines.

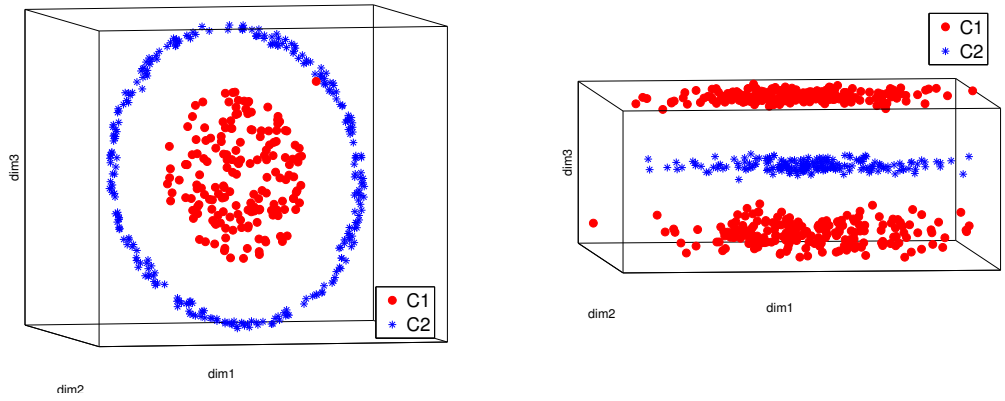


Figure 3.3.: Toy data set 1 (left). Note the potential outlier point of class 1 in the upper right part of the data set. The right image shows toy data set 2.

Data set 1 (left) is intrinsically two-dimensional and consists of a plate surrounded by a circle. Each object represents a class. Note that one point of class 1 lies apart from the other samples of that class and close to samples of class 2. We train two SVM models for this data set: a complex one with small RBF kernels and one with larger RBF kernels. We apply the proposed framework to visualize these classifiers, thereby employing π and π^{-1} as provided by the PCA. In this case, PCA is suitable because the class-relevant structure of the data is described well by the variance. Indeed, the evaluation schemes estimate the accordance to 0.99. The visualizations of the data together with the underlying classifier are shown in Fig. 3.4.

The left image depicts the complex SVM. It can be directly observed that the class boundaries are rather complex and that the outlier is classified correctly, yielding potential generalization disadvantages. The right hand side of the figure shows the less complex SVM. Here, the rather smooth class boundaries are directly visible and a good generalization can be expected due to the large margin - at the cost of one misclassification, however. Observing the complexity of the class boundaries might be very interesting, for instance if one is addressing the bias variance dilemma of the classifier. This is an example how the user tasks 2 and 4 can be addressed with our proposed framework.

SVMV [170] uses the SOM for dimensionality reduction and yields a very similar result as can be seen for both SVMs in Fig. 3.5. The two SVMs can be distinguished here as well, although, the margin of the classifier is not displayed so well. This is an effect of the SOM since it is related to vector quantization and, hence, usually doesn't place nodes in regions without data (except it has to due to the neighborhood function, which is the reason for the class boundary being shown in this example at all).

The quality of the visualization as measured by the accordance of the class labels assigned to points by the classifier and the labeling that would be assigned to by the visualization of the classifier amounts to 100% for all visualizations shown in Figures 3.4 and 3.5. The evaluation of the contours describing the certainty of the classifier

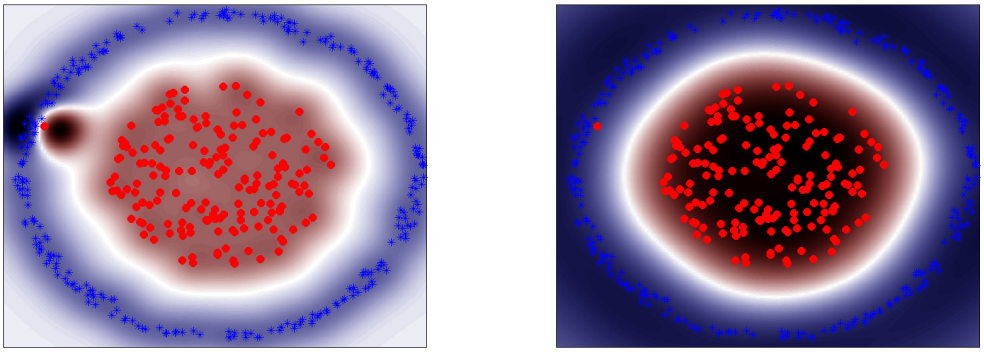


Figure 3.4.: Visualization of two different SVMs trained on data set 1 with PCA.

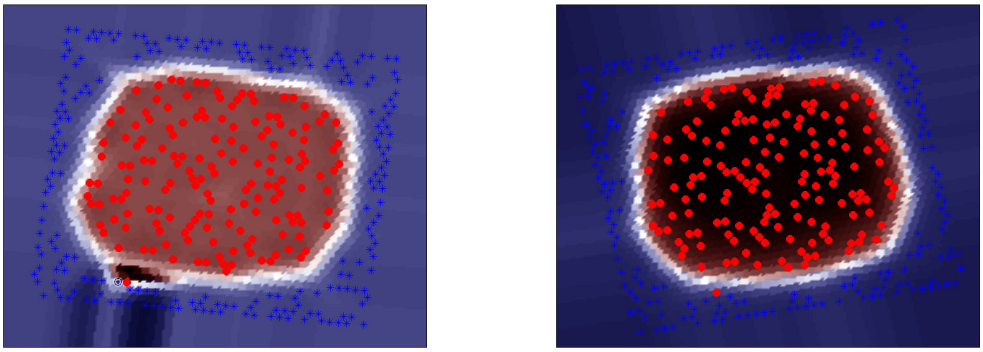


Figure 3.5.: Visualization of two different SVMs trained on data set 1 with SOM.

amounts to over 0.99.

Data set 2 consists of three clusters. One cluster corresponds to class 2 and it is surrounded by the other two clusters belonging to class one (see Fig. 3.3, right). The topmost two clusters are flat disks while noise is added to the lowest one, yielding that the lowest two clusters are closer to each other. Furthermore, for all clusters, the variance along the first two dimensions is higher then along the third one. This data set is an example for user task 1.

We use this data set to show the drawbacks of PCA and SOM visualizations. We train a SVM classifier and visualize it with PCA in Fig. 3.6. The accuracy of this visualization as concerns the labels amounts only to 42%, accordance of the contours only to 0.04. As can be verified in Fig. 3.6 (left), PCA maps the three clusters on top of each other, making a proper visualization of the classifier impossible. In these visualizations, we mark the points for which the classifier is displayed incorrectly with white circles. The right image shows the projections of the samples from the two-dimensional space into the original data space (this image is zoomed in on the Z-axis). In this case, the points are mapped to the first two principal components showing also

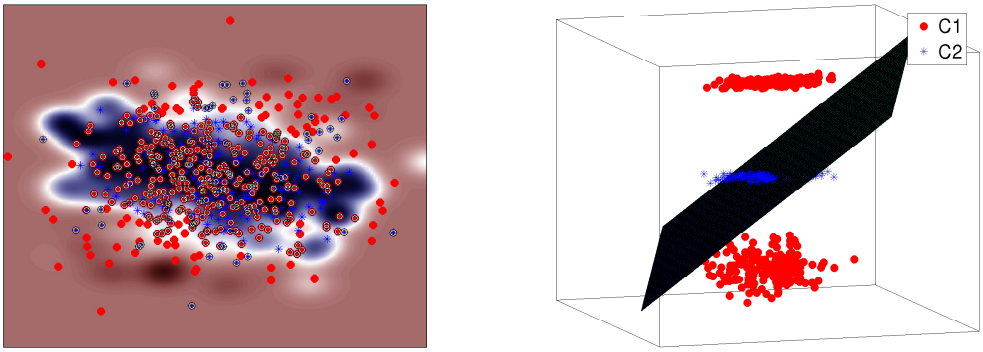


Figure 3.6.: Visualization of data set 2 with PCA (left) and the according inverse projected samples (right).

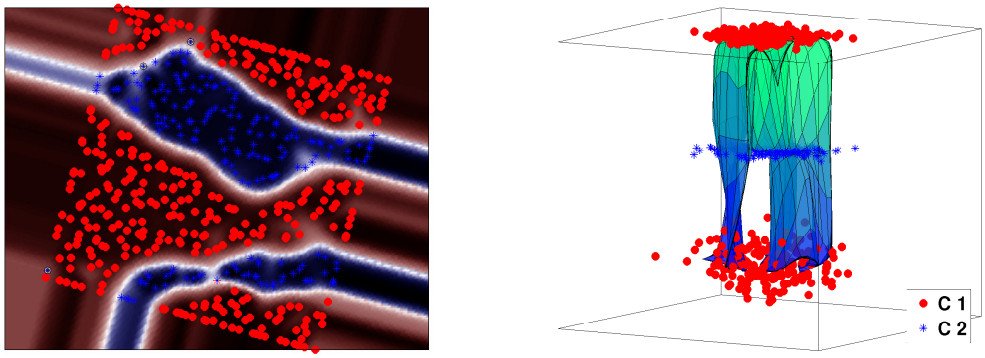


Figure 3.7.: Visualization of data set 2 with SOM (left) and the according SOM map (right).

how the dimension reduction from three to two dimensions has worked.

The same classifier is visualized by the SOM in Fig. 3.7. Although, the projection is much better (99% of the data points are assigned to the correct class by the map and contours agree to the value of 0.98) it fails to show the three distinct clusters. On the contrary, it suggests that there exist two clusters of class 2. The right hand side of Fig. 3.7 shows again the inverse samples. Due to the fact that the inverse mapping π^{-1} for the SOM is the assignment of a point to a high-dimensional prototype, these shown samples coincide with the location of the self-organizing map. The position of this SOM grid explains how the projection of the data emerged.

Using t-SNE we obtain the visualization shown in Fig. 3.8. Here, the three distinct clusters are visible and, further, it is shown that the class boundary between the blue cluster and one of the red clusters is more complex. The quality of the visualization of the classifier amounts to 99% and 0.98 for the labels and contours, respectively. The right hand side of Fig. 3.8 shows again the projected samples. The shown manifold

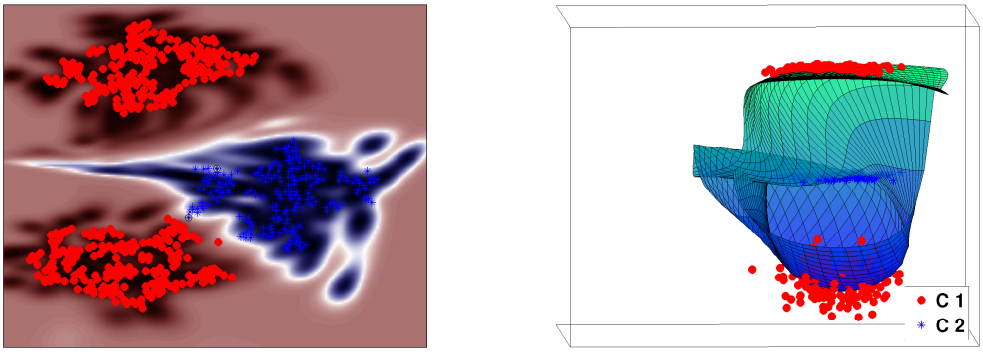


Figure 3.8.: Visualization of data set 2 with t-SNE (left) and the according inverse projected samples (right).

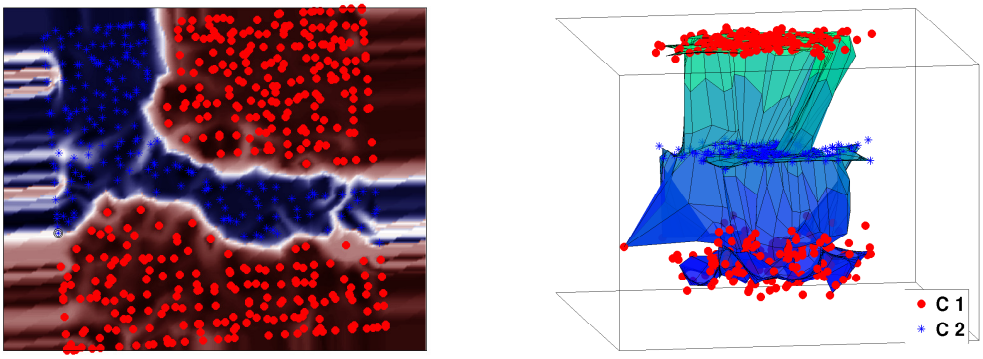


Figure 3.9.: Visualization of data set 2 with Fisher SOM (left) and the according inverse projected samples (right).

lies smoothly in the data clouds.

Calculating the SOM on the Fisher metric (we use the relational batch SOM [60] in our experiments for this purpose) we obtain the visualization shown in Fig. 3.9. The projection displays much better the original data characteristics hence it shows that two regions of class one are separated by samples from class two. So for this data set, the integration of the Fisher metric yields a major improvement to the approach SVMV. However, the margin of the classifier is still not visible. The quality evaluation yields the values 99% and 0.99 for the accordance of labels and certainty.

Replacing the standard Euclidean metric by the Fisher metric seems to be advantageous for showing the class sensitive properties of the data. Whether this generalizes also to other data sets and whether it is also beneficial for the visualization of classifiers is investigated in section 3.5.3.

Table 3.1.: Classification accuracies of the three SVMs, each trained on a different label assignment.

	l^1	l^2	l^3
training set	100%	96.5%	51.5%
test set	99.2%	95.2%	48.0%

3.5.2. Visualizing classifiers for different class distributions of the same points

In this section we demonstrate the suitability of our approach for another artificial setting: We randomly generate data $\{\mathbf{x}_i\}_{i=1}^N$ for $N = 500$ on a three-dimensional filled cube and generate three sets of labels for two-class problems. With these experiments, we address the user tasks 1 and 3. The labels are generated as follows:

1. The first class distribution consists of two clearly separated classes defined by a linear plane. We refer to the according labels by $\{l_i^1\}_{i=1}^N$.

2. For the second labeling $\{l_i^2\}_{i=1}^N$ we employ two parallel separation planes.

3. Here we utilize a random assignment with labels $\{l_i^3\}_{i=1}^N$. An overlapping class structure originates, thus addressing user task 3.

One thing that all these scenarios have in common is that this data set is intrinsically three-dimensional and impossible to visualize adequately in two dimensions. However, the class relevant structure is locally one-dimensional, i.e. at each position in the data space only one direction is relevant for classification. This holds for all two-class problems, as discussed in section 2.3.4.

Additionally, we project this data set with a random matrix to 10 dimensions. In this 10-dimensional data space, we train one SVM for each set of labels. The classification accuracies of these three classifiers are depicted in Table 3.1.

We utilize our approach to visualize these classifiers. Thereby, we rely on Fisher t-SNE to project the data set $\{\mathbf{x}_i\}_{i=1}^N$, while each time employing different labels and hence yielding different visualizations of the data. The three resulting visualizations of the classifiers are depicted in Figure 3.10.

The accuracy of the three visualizations as measured by the method introduced in section 3.4.3 based on the labels yields an accordance of 98.6% for the set $\{l_i^1\}_{i=1}^N$ (the left visualization), 96.0% for $\{l_i^2\}_{i=1}^N$ (middle) and 100% for set $\{l_i^3\}_{i=1}^N$ (right). The quality based on the certainty yields 0.91 (left), 0.90 (middle) and 0.82 (right).

In addition to the high accordance of the labelling regions, in this case we know the underlying class structure and, hence, can judge the visualization qualitatively. The structure of the projected points and of the class regions agrees largely to the labelling of the asociated case, i.e for case 1 two coherent structures are present, for case 2 there are 4 coherent regions while for case 3 the labelling does not have any structure.

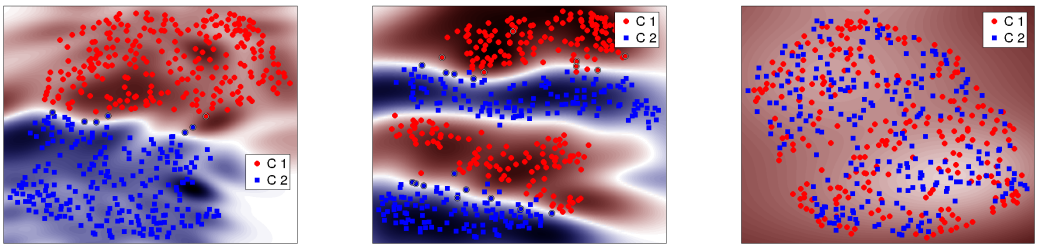


Figure 3.10.: Visualization of SVMs trained on the 10-dimensional data set with the labels l_i^1 (left), l_i^2 (middle) and l_i^3 (right).

3.5.3. Evaluating discriminative dimensionality reduction techniques for classifier visualization

In this section, we compare the DR techniques t-SNE, Isomap, MVU, SOM and GTM to visualize classifiers. Exemplarily, we use the SVM here (other classifiers are visualized in sections 3.5.4 and 3.5.5). We apply these methods on the Euclidean and on the Fisher metric and we use the prefix "Fisher" in front of the DR name to indicate the latter.

In order to evaluate the effect this change of the metric has, we utilize three benchmark data sets. Similarly as in [163], we use a randomly chosen subsample of 1500 samples for each data set to save computational time.

- The *letter recognition* data set (referred to as letter) comprises 16 attributes of randomly distorted images of letters in 20 different fonts. The data set contains 26 classes and is available at the UCI Machine Learning Repository [45].

- The *phoneme* data set (denoted phoneme in the following) consists of phoneme samples which are encoded with 20 attributes. 13 classes are available and the data set is taken from LVQ-PAK [82].

- The *U.S. Postal Service* data set (abbreviated via usps) contains 16×16 images of handwritten digits, and hence comprises 10 classes. It can be obtained from [134]. This data set has been preprocessed with PCA by projecting all data samples on the first 30 principal components.

As described previously, we employ SVMs with a "one versus one" classification scheme for the following data sets with more than two classes.

For each data set we apply the ten DR methods to project all points from that set. Afterwards, we utilize a ten-fold cross-validation scheme to evaluate the inverse mapping π^{-1}: The data set is randomly divided into ten parts, where nine subsets are used to train π^{-1} and the remaining subset is used for evaluation with our scheme proposed in section 3.4.3. This procedure is repeated ten times yielding a mean and standard deviation shown in Fig. 3.11 for all methods and all data sets.

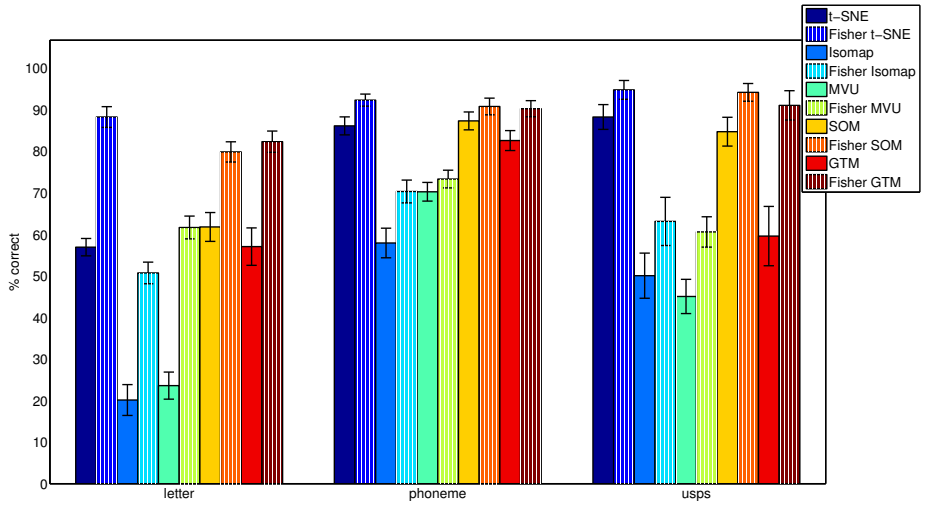


Figure 3.11.: Empirical comparison of different DR techniques with and without supervision.

For each data set and each DR projection the supervised variant achieves a better performance for the purpose of classifier visualization. This also holds for the SOM projection, yielding an improvement to the SVMV method. Further, the methods Fisher t-SNE, Fisher SOM and Fisher GTM yield the best results in our experiments.

Example visualizations of the SVM trained on the phoneme data set are shown in Fig. 3.12 and 3.13. In both, the left column displays the unsupervised visualizations and the right one the supervised ones. In the right column, the cluster structure is better visible and, hence, allows a better visualization of the class boundaries.

3.5.4. Utilizing supervised information based on a trained classifier

In this section we apply supervised projections based on the Fisher information metric which is induced by different conditional class probabilities $p(c|\mathbf{x})$. We illustrate two different variants for the estimation of the conditional class probability. Thereby, $p(c|\mathbf{x})$ is utilized twice in our proposed algorithm: for the calculation of the DiDi projection and for estimation of its inverse.

More precisely, we investigate the difference between estimating $p(c|\mathbf{x})$ from the given labeling $c := l$ (i.e. from the ground truth) and estimating $p(c|\mathbf{x})$ from the labels of the classification model $c := f(\mathbf{x})$, i.e. the difference between using $p(l|\mathbf{x})$ and $p(f(\mathbf{x})|\mathbf{x})$ for the estimation of the local Fisher information matrix. Both can be done with the Parzen window estimator. If a probabilistic model is available and if it provides differentiable probabilities $p(f(\mathbf{x})|\mathbf{x})$, however, an alternative for the latter is to utilize $p(f(\mathbf{x})|\mathbf{x})$ directly to compute the local Fisher information matrices.

In this section we do the latter, and for this purpose utilize the Robust Soft LVQ (RSLVQ) classifier which has been briefly summarized in the beginning of this sec-

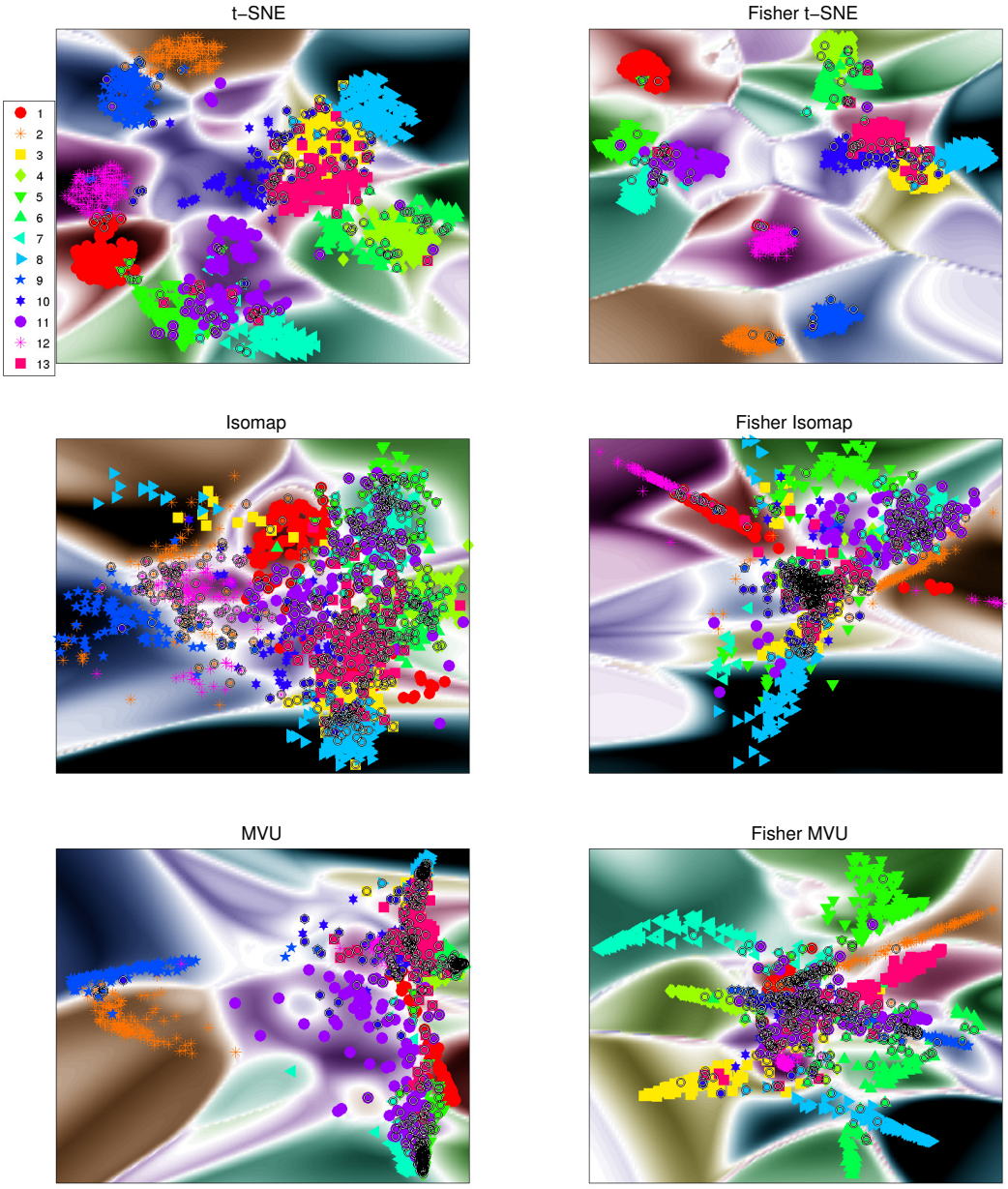


Figure 3.12.: Visualization of the phoneme data set with the methods t-SNE, Fisher t-SNE, Isomap, Fisher Isomap, MVU and Fisher MVU.

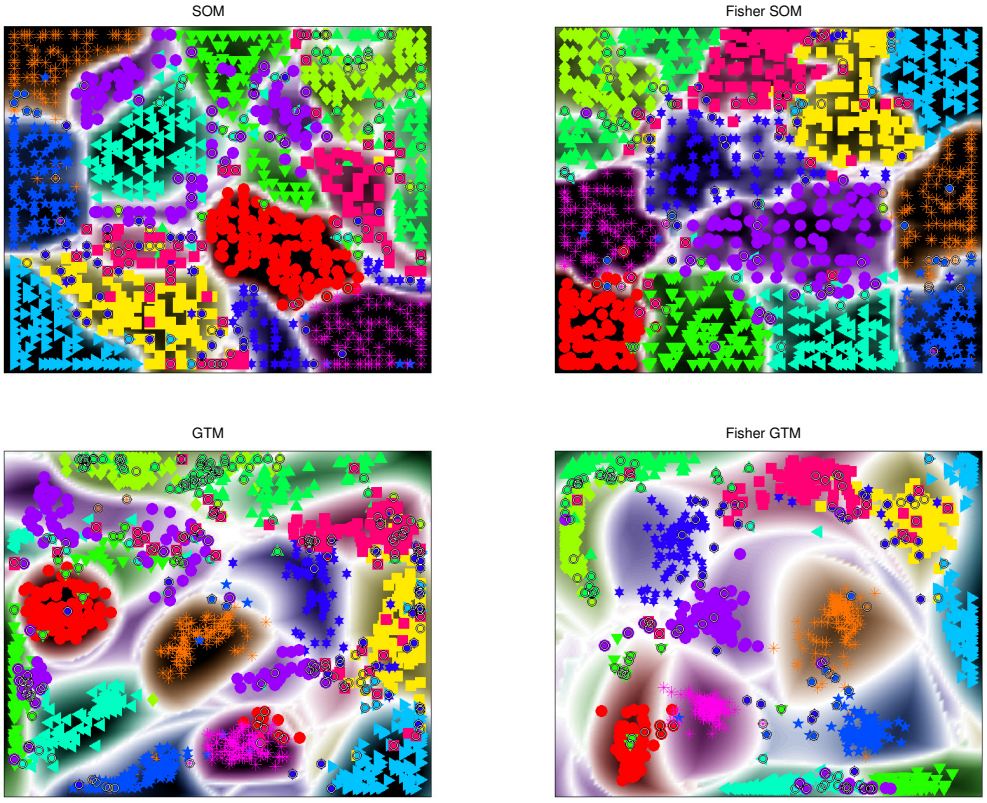


Figure 3.13.: Visualization of the phoneme data set with the methods SOM, Fisher SOM, GTM and Fisher GTM.

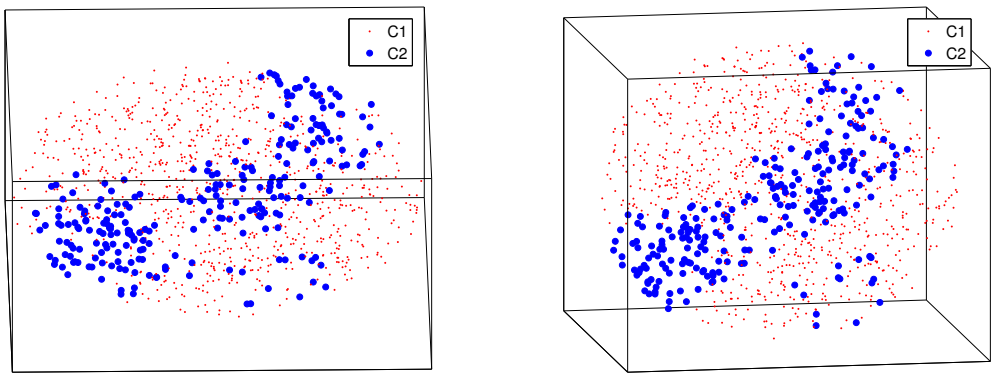


Figure 3.14.: The three-dimensional data set 3 shown from two different perspectives.

tion. With this classifier, we can demonstrate the user task 5, i.e. how did the RSLVQ algorithm choose the prototype positions in order to solve the task.

We create an artificial data set (referred to as data set 3) which is intrinsically three-dimensional and, hence, cannot be projected to two dimensions without information loss. The data points are uniformly sampled in a filled ball. A posterior labeling is assigned to them such that a nonlinear class structure emerges. This set is shown in Fig. 3.14 from two perspectives. Class two (shown in blue) consists of a continuous tube which is, however, separated by a gap. Further, there is a distinct noisy region.

An unsupervised projection of this data set with t-SNE is shown in the left image of Fig. 3.15. As expected, the projection distorts the continuous class structure since in an unsupervised scenario no information about the labeling is available. This illustrates that unsupervised visualization techniques might not always be well suited if intrinsically high-dimensional data should be projected to low dimensions. In this example, the displayed information looks almost arbitrary.

For the training of the classifier, we use only four prototypes per class, which is a small number considering the complexity of the data set. The trained classifier achieves a classification accuracy of 90%. Now, a typical use case for the classifier visualization method occurs: How did the classification method solve this problem? Which simplifications of the data did the classifier use and which data points are regarded as similar by the classifier?

In order to answer these questions we visualize the classifier using Fisher t-SNE built on the original class labels l_i on the one hand and on the provided classification $f(\mathbf{x}_i)$ on the other hand. We build the visualization of the classifier on top of these two projections. The two resulting visualizations are depicted in Fig. 3.16. The left visualization is based on the Parzen window estimator for the class labels $p(l|\mathbf{x})$: Basically, two clusters of points from class blue are shown and these are distinct from each other. The visualization quality of the classifier amounts to 92%. Interestingly, albeit this is not yet perfect, the visualization looks much more reasonable than direct

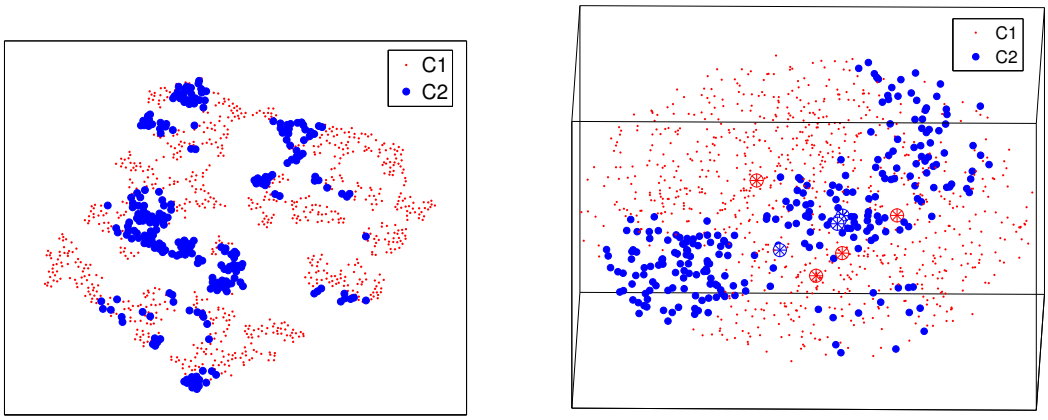


Figure 3.15.: Projection of data set 3 with t-SNE (left). Data set 3 together with the prototypes of the trained RSLVQ model (right).

unsupervised t-SNE on the data. The right visualization shows the same classifier, but this time based on the discriminative projection obtained by using the probabilities $p(f(\mathbf{x})|\mathbf{x})$ of the classifier itself. The data from class two form again two clusters, but this time, they are close to each other. The quality is estimated to 95%. Furthermore, the shape of the class boundaries resembles more the expected shape of the classifier, the latter usually being related to convex regions. In the visualization based on the ground truth, the original spherical shape of the data is much more pronounced.

The Parzen window estimator used in the left visualization estimates the probability density accurately and finds the gap in the blue class tube. In this part of the data space, the class distribution changes rapidly and, therefore, the distances in this region grow large, which can directly be observed in the visualization. The prototype distribution does not fit very well to the visualized classifier, since in one region of the blue class there are three prototypes of that class on top of each other and in another region there are none. But since the visualized class distribution is correct as concerns a large part of the points, we can see from this visualization that the largest part of the blue class tube is classified correctly.

In the right visualization which is based on the labeling of the classifier, the two parts of the tube lie close together. This suggests that the labeling of the classifier does not change much in this region, i.e. that the data lying in this gap of the tube are classified incorrectly. This can also be seen directly in the visualization. For few points, the visualization of the classifier is inaccurate, but these lie close to the class boundary, i.e. imply only small inaccuracies. The most points (95%) are displayed in the correct region of the classifier. This time, the location of the prototypes is plausible in relation to the data: the prototypes of the blue class are surrounded by those of the red class. Such a constellation is plausible in the original data space.

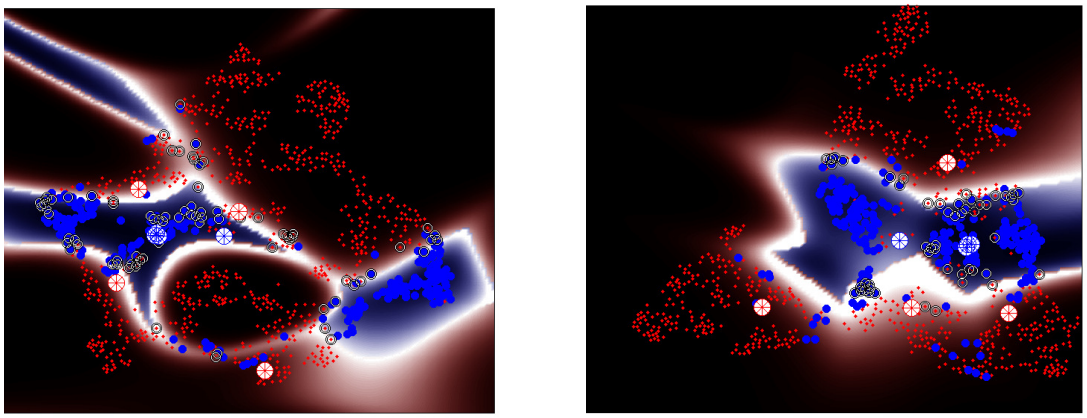


Figure 3.16.: Two visualization of the same RSLVQ classification model: The projection methods Fisher t-SNE based on the original labeling (left) and Fisher t-SNE based on the labels from the trained classifier (right) are applied.

From the latter visualization we can deduce more information as regards potential errors as compared to the previous one; we see directly the source of the remaining classification error: the classifier is not powerful enough and is not able to classify this gap in the data correctly. Furthermore, there are a few points from the blue class which lie in the cluster of points from the red class. From the perspective of this visualization we would deduce that these are either overlapping regions or too complex regions for our classifier (both aspects are probably correct: in the high-dimensional data we can see that there is indeed a region of overlapping classes).

For this toy example we can verify our interpretation by visualizing the positions of the prototypes in the original data space. The right image in Fig. 3.15 depicts the original data set in conjunction with the prototypes of the classifier. The same positioning of the prototypes as in the low-dimensional visualization emerges: the prototypes of the blue class are surrounded by those of the red class.

3.5.5. Visualize different classification models

In this section we apply our approach on the real world benchmark data set USPS for the two classifiers Robust Soft LVQ and Classification Tree, in order to demonstrate that it also works for other classifiers than the SVM.

Visualization of a Classification Tree We train a Classification Tree on the USPS data set used in the previous section. The resulting classifier obtains a classification accuracy of 89% on the training set and 66% on the test set.

Fig. 3.17 shows two Fisher SOM visualizations of this classifier: For the left we employ the Fisher information defined by the labels of the classifier and for the right

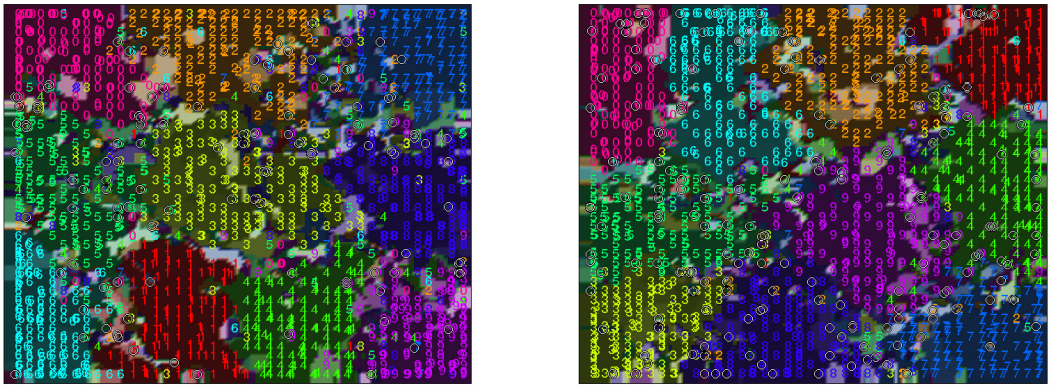


Figure 3.17.: Two Fisher SOM visualization of the same Classification Tree classier. The left visualization is based on labeling provided by the classifier and the right on the original labels.

one we utilize the original labels for the Fisher information (we use the Parzen window estimator in both cases). Due to this choice the left visualization rather shows the '"view of the classifier" on the data while the right one shows the true distribution. However, the first one can be better suited to interpret the trained classification model. In this case the quality of the left visualization of the classifier is 92.3% and the quality of the right one is 87.8%.

In the left visualization we can see that in the region of class 9 some instances of class 8 are mixed. In the right visualization this is not the case. Therefore, we can deduce that the separation of class 8 and 9 is particularly hard for the given classifier. Further, the classes 5 and 3 seem to overlap (left visualization). However, these two classes only have very little overlap in the right visualization. This indicates that the classifier is not complex enough in this region of the data space, as well.

Furthermore, we re-plot both visualizations from Fig. 3.17 in Fig. 3.18 with the labeling assigned to by the classifier. The visual impression of the two images shown in Fig. 3.18 agrees with the result of the formal evaluation measure suggesting that the left one visualizes the classifier more accurately.

Visualization of a Robust Soft LVQ model As a next step, we train a RSLVQ classifier with two prototypes per class on the USPS data set. The trained model obtains a classification accuracy of 97,2% on the training and of 87.3% on the test set.

Using the Fisher information as defined by the labels of the classifier and the Fisher SOM technique, the visualization shown in Fig. 3.19 (left) results. This visualization of the classifier has an accordance of 97.9%. The high classification accuracy can be observed in this visualization, as well. In addition, we can see directly which classes are mixed up the most time. For example, there are a few instances of class 7 classified

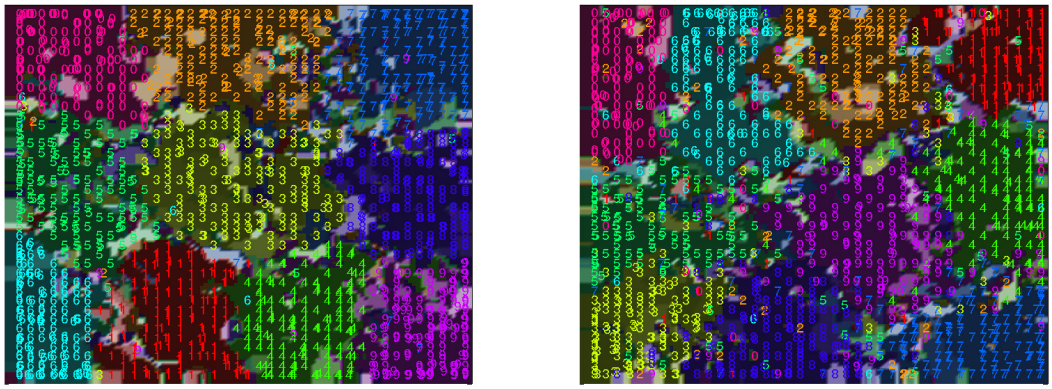


Figure 3.18.: Fisher SOM visualization of the Classification Tree where the data points are labeled according to the classifier. The same projections as shown in Fig. 3.17 are utilized.

as class 9. Having this knowledge, we could improve our classifier by increasing the complexity of the class boundary between these two classes (in this case we could employ more prototypes for these classes). On the other hand, the visualization suggests that the classes are unimodal. Furthermore, some prototypes of the same class seem to be located close to each other (e.g. those of class 0).

In order to obtain another view on the data, we project the classifier also with Fisher t-SNE (shown on the right of Fig. 3.19). This method tends to show clustering information (in contrast to the SOM, which doesn't show gaps between clusters). The Fisher t-SNE projection indicates further that the complexity of the model could be reduced without loosing much accuracy, since many prototypes lie on top of each other. More precise, for all except three classes (1,3 and 5) the two prototypes are positioned on top of each other. We examine this hypotheses by training a RSLVQ classifier with only one prototype per class. Indeed, this model has only a slight accuracy loss: the model classifies 95,1% of the training set and 86.9% of the test set correct (using the same training/test set partition as before).

3.6. Experiments with regression functions

In this section we demonstrate our proposed approach with artificial and real life data sets. We employ the popular Support Vector Machine for regression and the Decision Tree scheme as the models that we interpret. Furthermore, since we do not assume any particular property of the regression model, any regression scheme could be visualized in the same way. A description of the models follows.

- The Support Vector Machine for regression (SVR) [161] employs a linear function $f(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{w}$ in the feature space for prediction. Errors are penalized linearly,

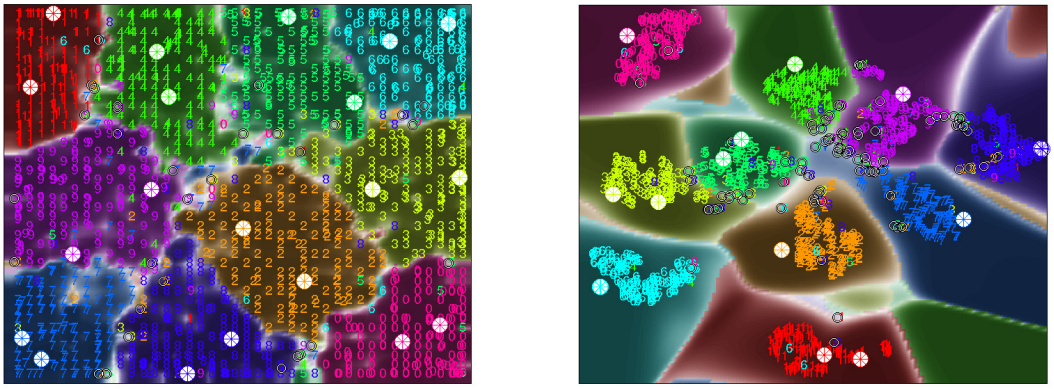


Figure 3.19.: Visualization of the RSLVQ classifier with Fisher SOM (left) and Fisher t-SNE (right). Both projections are based on the Fisher information as defined by the labels of the classifier (but the original labeling is shown).

where small errors, i.e. predictions lying in an ϵ-tube around the target, are not penalized. Since the whole approach can be formulated using scalar products of the data only, kernels can be employed. In the experiments, we utilize the Gaussian kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\beta\|\mathbf{x}_i - \mathbf{x}_i\|^2\right)$. We use the implementation provided by the libsvm [30].

- Decision Trees (DecTree) [22] for regression partition the data space X, where the prediction value in each partition is the mean of the points lying in the according partition. Splits are optimized such that the mean squared error is minimized. We utilize the Matlab implementation here.

In the following, we demonstrate how the user tasks described in section 3.1 can be tackled with our proposed approach, what effects the choice of the employed dimensionality reduction can have and we apply our presented approach to a real world data set. In the following, we briefly characterize the utilized data sets.

- *Data set1* is depicted in Fig. 3.20 (left) and consist of three two-dimensional clusters positioned above each other. One of these clusters (the bottom one) has additional noise in the third dimension. The prediction function is encoded in the color and is a squared function of dimension three.

- *Data set2* consists of two three-dimensional clusters with an outlier in-between these two clusters. Fig. 3.20 (right) depicts this set, where the color indicates the target variable of the regression task which is a linear function for the left cluster and a squared function for the right one. In both cases, the target function depends only on the first two dimensions.

- The *diabetes* [41] data set describes 442 patients by the 10 features age, sex, body

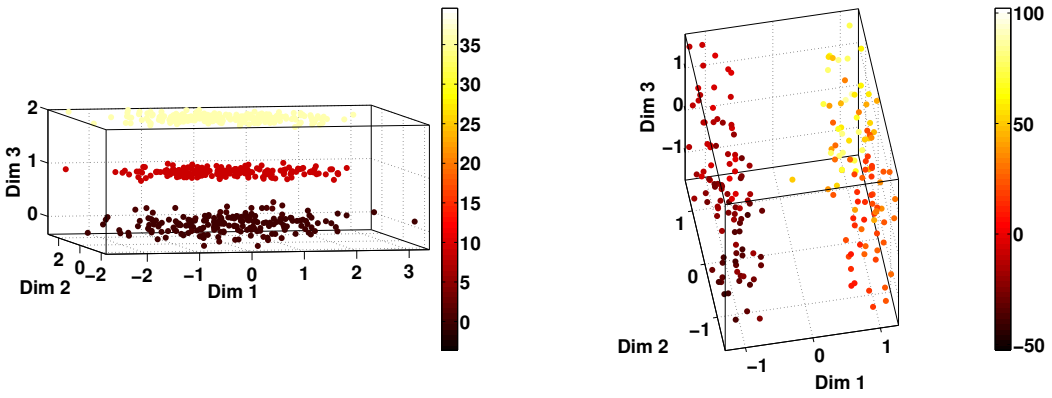


Figure 3.20.: Two toy data sets: data set1 (left) and data set2 (right).

mass index, blood pressure and 6 blood serum measurements. The target variable is a measure of the progression of the diabetes disease one year after feature acquisition.

We employ our proposed scheme for visualization of high-dimensional functions in the following. In contrast to the visualization of classifiers, however, we plot the resulting visualization over the third axis of the plot, such that the curvature of the visualized function can be observed better.

3.6.1. Effect of the selected dimensionality reduction technique

One key ingredient in our proposed approach is the DR projection. However, since any DR technique can be applied, we discuss in this section effects of the selected methods. We take a look at unsupervised DR techniques and we investigate the performance of DiDi methods in this context. For this purpose, we train a SVR model on data set1 and visualize it with different techniques.

The most common visualization approach is PCA. However, the latter is driven only by the variance of the data and neglects other structure. Hence, using PCA for data set1 yields to overlapping clusters and consequently to a bad visualization of the underlying regression model: the accordance computed by the evaluation procedure of section 3.4.3 is 0.21, i.e. the visualized model has only a small correlation to the original one.

In a scenario where the structure of the data is not known, more powerful nonlinear DR methods can be necessary. We investigate here the two methods GTM as a generative model and t-SNE as a neighborhood embedding technique.

Applying our regression model visualization approach to the trained SVR using the GTM yields a visualization with a quality of 0.95 (as summed up in Table 3.2). The visualized model is depicted in the top left corner of Fig. 3.21. Although, the accordance of the visualized prediction model with the original one is high, the visualization tears the cluster structure apart. So, more powerful methods for DR can increase the

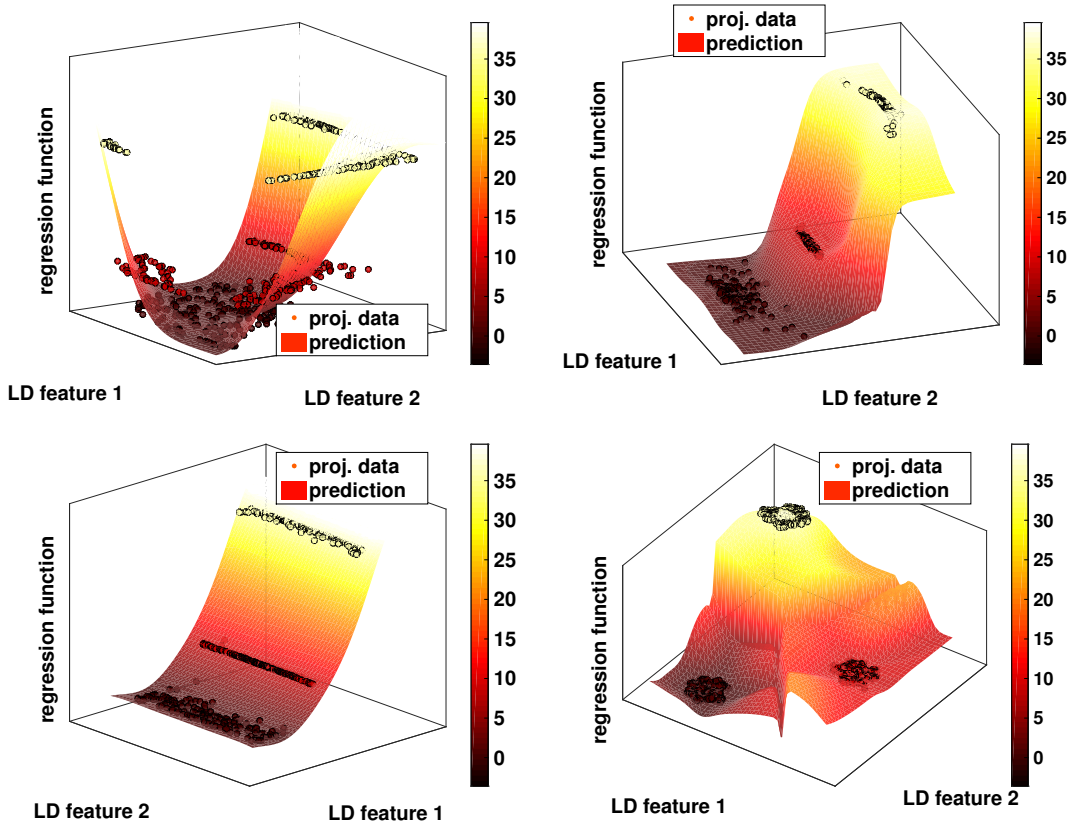


Figure 3.21.: Four different visualizations of the same regression model. These are based on (from top left to bottom right): GTM, Fisher MDS, Fisher GTM, Fisher t-SNE.

visualization quality. However, there still might be undesired effects, especially if the approaches act in an unsupervised way. Another option, besides choosing more powerful methods, is to utilize supervised ones, as already done for the visualization of classifiers. This can be done, as discussed earlier with the use of the Fisher metric.

To demonstrate the effect of such a supervised visualization, we apply our regression model visualization approach using Fisher MDS, Fisher GTM and Fisher t-SNE. Applying these techniques, we obtain three different visualizations of the same regression model. We evaluate them and obtain a quality of 0.99 for each visualization (summed up in Table 3.2). The visualizations (in Fig. 3.21) of Fisher MDS (top right) and Fisher GTM (bottom left) agree largely, while the Fisher GTM based visualization shows the shape of the squared polynomial target function without any distortions. In the Fisher t-SNE projection, the squared prediction for the single clusters can be observed, but it is not so clear as in the Fisher GTM mapping. One reason for this is that t-SNE often tears clusters apart since it has a high focus on local neighborhood preservation.

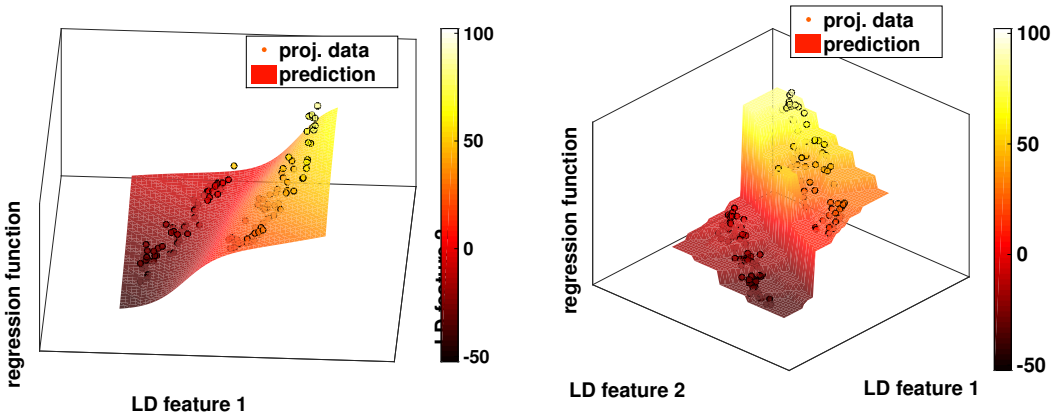


Figure 3.22.: A Fisher GTM induced visualization of the SVR (left) and Decision Tree (right) with data set1. The continuous surfaces depict the prediction of the regression models.

3.6.2. Illustration of potential user tasks

We utilize data set2 to illustrate the identified user tasks. For this purpose, we train a SVR and a Decision Tree using this data set. For the DR, we employ the supervised technique Fisher GTM - an unsupervised approach would try to embed this intrinsically three-dimensional data set in two dimensions and, hence, might result in an embedding not well suited to visualize the target function.

Using these ingredients, we can visualize the two regression models with our proposed approach. The numerical evaluation scheme in 3.4.3 implies a quality of 0.99 for both visualizations, as measured by the Pearson correlation. I.e. the regression model is shown accurately at least at the positions of the data. The evaluation results for all experiments are summed up in Table 3.2.

The resulting visualized models are shown in Fig. 3.22. The left plot depicts the SVR and the right one the Decision Tree. In both cases, the first two coordinate axes encode the two-dimensional embedding space of the data. The target variable is encoded both by the third axis and by the coloring. The surface depicts the prediction of the respective regression model.

We exemplarily address the user tasks for these visualizations. Considering user task 1, the complexity of the prediction functions can be observed directly in the vi-

Table 3.2.: Visualization qualities for the regression models, as measured by the Pearson correlation.

	PCA	GTM	Fisher MDS	Fisher GTM	Fisher t-SNE
data set1,	0.21	0.95	0.99	0.99	0.99
data set2, SVR	–	–	0.99	0.99	0.99
data set2, DecTree	–	–	0.99	0.99	0.99
diabetes	–	–	–	0.94	0.92

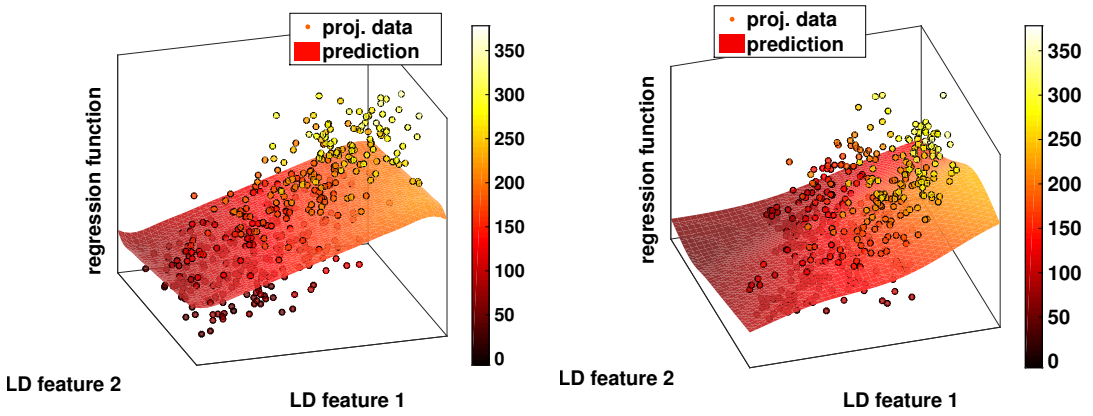


Figure 3.23.: A Fisher GTM (left) and a Fisher t-SNE (right) visualization of a SVR model trained on the diabetes data set.

sualizations: the SVR instance shows a smooth predictive function while the Decision Tree instance is very complex. This is particularly the case for the cluster with the squared function: the trained SVR might be considered underfitted, here.

Dealing with task 2, the user can observe that the complexities of the target functions are quite different in the two present clusters. The user could prefer to train two independent local models on these clusters, for instance. The extrapolation between these clusters is smooth in the left image but very steep in the right one, which might lead to bad predictions if future data are expected to lie also between the clusters.

In the right visualization, the piecewise constant regions are well visible which is typical for Decision Tree models (user task 3).

Considering user task 4, the visualizations directly imply how both models treat the outlier point: the SVR ignores it and the Decision Tree overfits it. Having this insight, the user can judge which model handles the data point of interest better, depending on his estimation of the regularity of this point.

3.6.3. Applying the proposed framework to real world data

For the diabetes data set, we train the SVR model by splitting the data set multiple times randomly in a training and a test set in order to estimate a good parameter value for the kernel of the SVR.

In the previous subsections we have argued that discriminative nonlinear DR methods are best suited for the visualization of regression models. Hence, we apply two such methods, i.e. Fisher GTM and Fisher t-SNE to the SVR model trained on the diabetes data set.

The evaluation based on 3.4.3 yields a quality value of 0.94 for the visualization based on Fisher GTM and a value of 0.92 for the Fisher t-SNE induced visualization. Both are shown in Fig. 3.23.

Interestingly, both visualizations agree in that sense that they show an almost linear prediction function. We have validated this by training a linear model and have

obtained a similar error on the test data.

3.7. Discussion

The general framework presented in this chapter allows to visualize nonlinear supervised models trained on potentially high-dimensional data sets. This framework makes it possible to visualize arbitrary classification and regression models, with the only restriction that the classifiers have to provide some measure of certainty. In the case of classification, we demonstrated this for Support Vector Machines, Classification Trees and probabilistic LVQ classifiers. For regression models, we employed Support Vector Machines for regression and Decision Trees.

This framework is general, in the sense that it allows to combine arbitrary classification and regression models with arbitrary projection methods. In oder to demonstrate this generality, we utilized ten dimensionality reduction methods to visualize the models and stated experimentally that among them, supervised DR techniques are particularly well suited for this task. Due to its generality, this framework also includes methods from the literature as a special case, for instance the SVMV. We have, moreover, extended this approach by utilizing the Fisher SOM in this context.

Further, we demonstrated that a visualization of trained supervised models can give insights into their prediction process. Hence, it could also be used to help improving the process of fitting models.

The evaluation of these visualized models is currently based on the prediction and certainty accordance of the projected and original model. Although we also evaluate the generalization of such visualizations to new points, other properties of the model are not evaluated, yet. Such properties include the topological structure of the function and the size of the margin in case of classification [104]. Hence, a more extensive evaluation of the obtained visualizations constitutes an open problem.

Chapter 4.

Interpretation of data mappings

Chapter overview *This chapter presents two methods for improving the interpretability of data mappings. The first proposal aims to estimate interpretable components for nonlinear dimensionality mapping, such enabling to access the relevance of the original features for a given mapping. The second approach deals with linear mappings, in general. Although a direct interpretation seems easily possible for them, it can actually be misleading, in particular for high-dimensional data. We propose a framework which estimates a valid relevance profile for a given linear data mapping.*

Parts of this chapter are based on:

[C15f] A. Schulz, B. Mokbel, M. Biehl, and B. Hammer. Inferring feature relevances from metric learning. In *SSCI CIDM 2015*, pages 41–48, 2015.

[C15a] A. Schulz, and B. Hammer. Metric learning in dimensionality reduction. In *ICPRAM 2015*, pages 232–239, 2015.

[C14d] B. Frenay, D. Hofmann, A. Schulz, M. Biehl, and B. Hammer. Valid interpretation of feature relevance for linear data mappings. In *SSCI CIDM 2014*, pages 149–156, 2014.

[C14a] A. Schulz, A. Gisbrecht, and B. Hammer. Relevance learning for dimensionality reduction. In *ESANN 2014*, pages 165–170, 2014.

4.1. Motivation

Machine learning (ML) methods constitute core technologies in the era of big data [32]: successful applications range from everyday tasks such as spam classification up to advanced biomedical data analysis. Further, today's most significant machine learning models are supported by strong theoretical guarantees such as their universal approximation capability and generalization ability. Still, it is a long way to enable the direct use of advanced ML technology in complex industrial applications or settings where a human has to take responsibility for the results. Most popular ML models act as black boxes and do not reveal insight into why a decision has been taken [136]. Hence the accuracy on the given data is the sole information based on which practitioners can decide to use a model. Despite strong theoretical results under idealized assumptions, this can be extremely problematic, since these assumptions are usually not met in practice. Further, black box models are restricted to a mere functional inference. Auxiliary information is not extracted, albeit often aimed for e.g. in biomedical

data analysis. These facts have caused a strong interest in interpretable ML models, with first promising results in specific domains such as biomedical data analysis [12, 23, 24, 101, 138, 151, 162].

In the last chapters, we have addressed one possibility to inspect data in a human understandable way, namely visualization of data together with a given classifier. In this chapter, we would like to take yet another point of view, focusing on the interpretation of a given mapping in terms of its original, usually meaningful input features.

In machine learning models, data mappings constitute a frequent operation. They appear in basically every algorithm, such as in regression, classification, dimensionality reduction or metric learning. These data mappings vary a lot as concerns their complexity, ranging from linear and locally linear to nonlinear mappings. In this chapter, we propose tools to improve the interpretability of nonlinear mappings for dimensionality reduction and of linear mappings in general.

While there exist many complex and successful DR approaches which allow to project data in a nonlinear way [26, 94, 159, 163, 53], linear methods such as the principal component analysis (PCA) are often preferred by practitioners. One reason for this is the lack of interpretability for nonlinear methods: While linear mappings provide parameters which directly weight the according features, nonlinear methods provide either parameters which interact in a complex nonlinear way with the features, or they don't provide a parametric mapping at all. The latter are usually referred to as the class of non-parametric DR techniques and these are often particularly flexible and well suited for data projection, as we have seen in previous chapters. One example constitutes the class of discriminative dimensionality reduction mappings discussed in chapter 2. Considering these nonlinear DR methods, we propose an algorithm which provides an estimation of how relevant the data features are for the obtained projections in section 4.2 .

Linear mappings constitute a core part in many algorithms, including ridge regression, metric learning, or principal component analysis. Such methods have a very broad area of application, while being particularly valuable in the context of high-dimensional data [147]. In addition to excellent generalization properties and efficient learning procedures, these methods seem to be especially well suited for interpretability since they assign weights to each feature. These weights are then often interpreted as relevance scores for the according feature. Recent results have, however, shown that a direct interpretation of the weights of linear mappings can be exceedingly misleading, in particular for high-dimensional and correlated data. In this context, we propose a technique to extract valid feature relevance from linear mappings.

4.1.1. Scientific contributions and structure of the chapter

This chapter presents the following contributions:

Interpretation of nonlinear DR In section 4.2, a novel approach is proposed which estimates the relevance of the original features for a given projection. Thereby, it

defines a linear mapping on the data, with the goal of making them as similar as possible to the given, possible nonlinear, projection. This similarity is measured by a nonlinear function which evaluates the neighbor preservation. The weights of the linear mapping can then be employed for interpretation.

Valid interpretation of linear mappings Based on the notion of equivalence for linear mappings, section 4.3 presents a new framework which allows a valid interpretation of feature relevance, closely related to the concepts of strongly and weakly relevance.

4.2. Estimating interpretable components for nonlinear DR

One very common classical dimensionality reduction method is offered by principal component analysis (PCA), which constitutes the by far most popular data visualization technique in diverse application domains [16]. However, being a linear technique, it is severely restricted as concerns its capability to capture non-linear structures and clustering effects. In recent years, a huge variety of non-linear dimensionality reduction techniques has been proposed, see e.g. the overviews [26, 94, 159, 163, 53]. Many techniques can be accompanied by guarantees that they are capable of extracting the true, possibly non-linear underlying data manifold [135, 150, 53]; however, these techniques are not well suited to visualize data provided the underlying manifold structure cannot be preserved in only two dimensions due to a higher intrinsic data dimensionality [160]. A few powerful alternatives rely on the notion of neighborhood structures, with the neighborhood retrieval visualizer (NeRV), for example, explicitly realizing an information retrieval perspective, and allowing a suitable compromise of the amount of information which is preserved in the visualization [159, 163]. These techniques provide excellent results in application scenarios, and they mirror what is currently accepted as state of the art as a suitable cost function of non-parametric dimensionality reduction techniques [95, 163]. In this section, we will mostly be concerned with NeRV as theoretically well-founded method and one of the most powerful nonlinear data visualization techniques available today. Quite a few extensions of NeRV, or the very similar technique t-SNE proposed in [159] exist to cope with the problems of efficient implementation, integration of prior knowledge, an extension of the non-parametric technique to an explicit mapping prescription, or extensions to alternative cost measures [177, 93]. We have dealt with the topics of integration of prior knowledge and extension to an explicit mapping in chapter 2.

One severe problem of techniques such as t-SNE, NeRV and its discriminative counterparts, lies in the fact that they are non-parametric nonlinear techniques for which the obtained visual data display, unlike linear counterparts such as PCA, cannot easily be linked to semantically meaningful information: The two-dimensional projection coordinates have no direct meaning and they are not linked to feature dimensions of the data, unlike linear projections such as PCA, where the projection axes can be expressed as weighted combinations of the original data dimensions. For non-parametric pro-

jections, the relative location of data points is the only relevant information preserved in the mapping. As a consequence, it is not easy to judge which data dimensions are particularly important for the visual display, and which correlations of the data dimensions contribute to the mapping. In particular, invariances of a visualization such as orthogonal transformations cannot easily be dealt with due to this missing alignment. Since data visualization is an unsupervised and inherently ill-posed task, this fact leads to a severe risk of interpreting the visual display in a wrong way, if its interpretation is possible at all [162, 138]. Further, an interactive manipulation of the data by means of the visual display is not easily possible.

Recently, a few approaches have been proposed which try to overcome this gap and which accompany visualization techniques with methods to more easily interpret the display and manipulate the data representation based thereon [25, 43, 124]. These techniques propose to change the data metrics based on a given visual display, whereby different techniques are involved, ranging from heuristic model updates up to Bayesian learning of the data metric. In this contribution, we will follow these first steps which change the metric of the data based on a given visual display. By incorporating recent insights from the fields of metric learning and supervised machine learning, we will arrive at a very simple and intuitive metric adaptation scheme which offers insight into the visual display. Furthermore, this scheme directly provides the possibility to manipulate the data representation accordingly.

Metric learning constitutes a very powerful scheme which is well-known in machine learning, and a variety of techniques has been proposed in the context of supervised learning, see e.g. [13, 27, 57, 108]. Mostly, a global or local general quadratic form of the distance[1] is adapted in these settings, such that the underlying goal (usually classification) is improved as much as possible. Besides an increased model accuracy, these techniques enable auxiliary insight into the task by providing a relevance weighting of the data dimensions, which indicates the contribution of these data dimensions to the task at hand. Furthermore, by means of the linear transformation underlying the quadratic form, a new data representation is defined, which can even directly be used to inspect the data in some cases.

Here, we will transfer a particularly elegant metric learning scheme to the field of unsupervised dimensionality reduction [17]. This scheme will allow us to learn a global quadratic form which mirrors the neighborhood relationships as provided by the visual display. The metric allows a direct interpretation of the relevance of the feature dimensions for the given mapping; further, since it can be linked to a linear data transformation, it enables a change of the data representation based on the visual display. Hence, it makes it possible to impose external information on the data in a very simple form. We will demonstrate this latter principle by referring to discriminative dimensionality reduction settings.

In the following subsection 4.2.1, we explain the neighborhood retrieval visualizer and its relation to a quantitative evaluation of dimensionality reduction techniques.

[1] Sometimes also referred to as Mahalanobis distance

Subsequently, we propose three different schemes for estimating the feature relevance for a given visual data display: In 4.2.2, we describe a simple approach based on ideas from feature selection. In 4.2.3, we propose a relevance learning scheme and, as an extension in 4.2.4, a powerful metric learning scheme based on NeRV, which enables the efficient learning of relevance matrices by a superposition of a cost optimization and suitable regularization. Thereby, all these schemes can be used independently of the technique which is underlying the visual display. We demonstrate the suitability and efficiency of the approaches in several benchmarks.

4.2.1. Neighborhood Retrieval Optimizer

We will exemplarily consider NeRV [163] as a differentiable objective function to measure the similarity of the original data and a given projection of the former. This cost function is particularly well suited for our purpose, since it can be linked to neighborhood preservation in an information retrieval sense. NeRV can also be used directly as a method for DR, but we will mainly utilize it as a cost function measuring the quality of an embedding. This way, we do not have to make any assumptions on the way the projected data points have been obtained.

Similarly as t-SNE, NeRV relies on measuring the probability for two points being neighbors. In order to compute such a measure, we assume the availability of a distances measure d in the data space $\mathcal{X}$. We define

$$p_{j|i} = \frac{\exp(-0.5 d(\mathbf{x}_i, \mathbf{x}_j)^2 / (\sigma_i^x)^2)}{\sum_{k \neq i} \exp(-0.5 d(\mathbf{x}_i, \mathbf{x}_k)^2 / (\sigma_i^x)^2)} \tag{4.1}$$

as the probability of two points being neighbors in the data space, and

$$q_{j|i} = \frac{\exp(-0.5 \|\boldsymbol{\xi}_i - \boldsymbol{\xi}_j\|^2 / (\sigma_i^\xi)^2)}{\sum_{k \neq i} \exp(-0.5 \|\boldsymbol{\xi}_i - \boldsymbol{\xi}_k\|^2 / (\sigma_i^\xi)^2)} \tag{4.2}$$

as the probability of two projections being neighbors in the projection space. Thereby, the standard deviation σ_i^x in the data space is chosen such that a fixed effective number of neighbors k (with default $k = 10$) is reached and then the standard deviation σ_i^ξ is set to the same value. NeRV optimizes the costs

$$Q_k^{\mathrm{NeRV}}(\mathbf{X}, \Xi) = \gamma \sum_i \sum_{j \neq i} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}} + (1 - \gamma) \sum_i \sum_{j \neq i} q_{j|i} \log \frac{q_{j|i}}{p_{j|i}} \tag{4.3}$$

corresponding to the deviation of the two probability distributions. $\gamma \in [0, 1]$ weights the relevance of obtaining a good recall, corresponding to the first summand, and a good precision, corresponding to the second summand; per default, a compromise $\gamma = 0.5$ is chosen. Optimization is commonly done by a stochastic or conjugate gradient descent. There exist very similar alternative methods such as t-NeRV, which uses the student-t distribution instead of Gaussians for the low-dimensional embedding, to better prevent the so-called crowding problem. Another similar alternative constitutes the method t-SNE, which has been discussed in section 2.2.1. The single difference to t-NeRV is that t-SNE optimizes only one summand of these costs [160].

Interestingly, the NeRV costs can be interpreted as a smoothed version of the crisp costs which evaluate the degree of neighborhood preservation for a given DR display, as formalized in the frame of the co-ranking framework proposed in [92] and discussed in section 2.2.2. Assume a fixed neighborhood range k, the average overlap of neighborhoods of size k in the projection space and the original data space are counted, leading to the quality $Q_k^{\text{nx}}(\mathbf{X}, \Xi)$ as defined in equation (2.2). The neighborhood degree k is crisp, while the NeRV costs consider a smooth version induced by the Gaussian, but still emphasizing a certain neighborhood range by means of a fixed choice of the bandwidth.

Since NeRV is a non-parametric approach, we obtain projection co-ordinates of the given data only. The axes of the projection are widely arbitrary, and no semantic meaning is attached to the visual display. By incorporating relevance or metric learning, we aim at complementing the visual display by a link to the original data dimensions, such that the display can be accompanied by a semantic meaning in terms of the original (usually interpretable) data dimensions.

4.2.2. Feature selection for DR

Note that nonlinear DR techniques such as t-SNE provide a non-parametric mapping $\mathbf{x}_i$ to $\boldsymbol{\xi}_i$ for which an interpretation is not clear. In particular, it is not clear how relevant a given feature X_l is for the mapping. We are interested in ways to enhance nonlinear DR by a relevance weighting for the features $l \in \{1, \ldots, D\}$ of $\mathcal{X}$. As a first approach, we treat this problem as a feature selection problem. That means, we deal with the question: Which features are particularly relevant for the given DR function (which is given only implicitly)?

The definition of explicit evaluation functions for nonlinear DR allow us to directly transfer classical feature selection techniques [39]: We can apply one forward or backward selection step regarding one feature for these evaluation functions. This yields our first two relevance determination techniques. Assume a nonlinear projection $\mathbf{X} \to \Xi$ is given.

- $\lambda^k_{\text{forward}}(l) := Q_k^{\text{nx}}(\mathbf{X}|_l, \Xi)$ where $\mathbf{X}|_l$ considers only feature X_l, i.e. the points $(\mathbf{x}_i)_l \in \mathbb{R}, \forall i$. This induces an ascending relevance ranking of the features.

- $\lambda^k_{\text{backward}}(l) := Q_k^{\text{nx}}(\mathbf{X}|_{\neg l}, \Xi)$ where points $((\mathbf{x}_i)_1, \ldots, (\mathbf{x}_i)_{l-1}, (\mathbf{x}_i)_{l+1}, \ldots, (\mathbf{x}_i)_D) \in \mathbb{R}^{D-1}, \forall i$ are considered. This induces a relevance ranking in descending order.

These techniques provide a suggestion for feature selection based on a fixed neighborhood k. The effect of this parameter is investigated in the experiments. These measures yield a qualitative evaluation of the relevance since they only consider the extreme cases of a feature being present or not. Thus, they do not allow a more fine grained view.

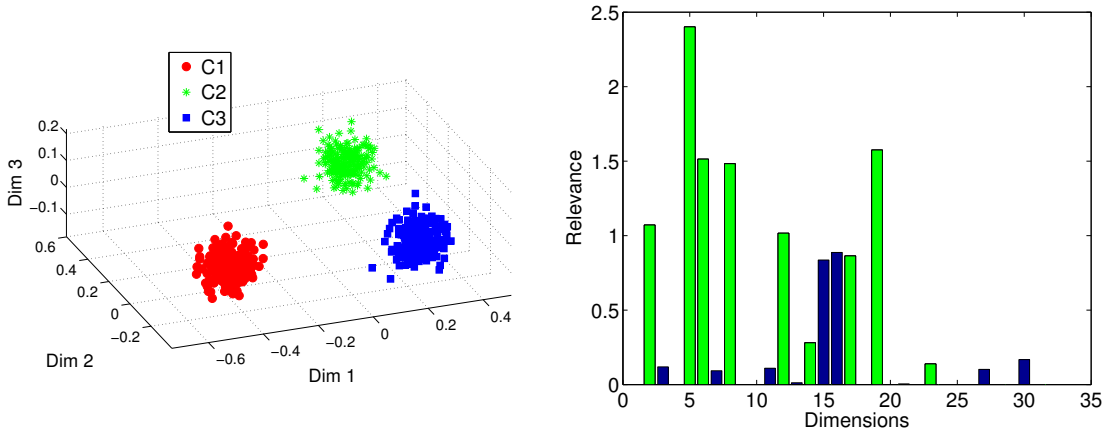


Figure 4.1.: Left: Data set1. Right: Relevance profile of the Adrenal data set. Green marks indicate that these 9 dimensions are also the top ones in [17].

4.2.3. Relevance learning for DR

For quantitative measures, we consider the smooth quality function Q_k^{NeRV}. The idea is to change the metric in $\mathcal{X}$ such that it takes into account the relevance of the dimension l: $d(\mathbf{x}_i, \mathbf{x}_j)^2 = \sum_l ((\mathbf{x}_i)_l - (\mathbf{x}_j)_l)^2$ becomes $\sum_l \lambda_l^2 ((\mathbf{x}_i)_l - (\mathbf{x}_j)_l)^2$. This corresponds to a feature transformation $\mathbf{X}_\lambda = \{(\lambda_1(\mathbf{x}_i)_1, \ldots, \lambda_D(\mathbf{x}_i)_D) \mid i\}$ of $\mathbf{X}$. We are interested in relevance terms λ such that the transformed feature space $\mathbf{X}_\lambda$ is as close as possible to the projection Ξ as measured by quality evaluation measures. This yields to the following objective:

- $\lambda_{\mathrm{NeRV}}^k(l) := \lambda_l^2$ where λ_l optimizes $Q_k^{\mathrm{NeRV}}(\mathbf{X}_\lambda, \Xi) + \delta \sum_l \lambda_l^2$.

$\delta > 0$ weights the sparsity constraint. Since the projection points $\boldsymbol{\xi}^i$ are fixed, we set σ in both spaces such that the fixed neighborhood size k is reached. To compute $\lambda_{\mathrm{NeRV}}^k(l)$, we optimize this objective L1 regularized quality $Q_k^{\mathrm{NeRV}}(\mathbf{X}_\lambda, \Xi) + \delta \sum \lambda_l^2$ with respect to λ_l^2. We use a gradient technique similar to well known algorithms from neural network optimization [133]. Strictly speaking, the result is not necessarily unique due to possible local optima; in practice, we did not observe problems.

4.2.4. Metric learning for DR

While feature selection only selects a set of features in a greedy way, relevance learning aims for a more subtle weighting of the features. Yet, it disregards possible feature dependencies by its decomposition in terms of a simple linear combination of features. This drawback can be overcome when considering a full matrix.

Again, assume a fixed data projection $\mathbf{X} \mapsto \Xi$ is given. A straightforward extension to the approach from above is to replace $\lambda_{\mathrm{NeRV}}^k(l)$ by a more general form of metric. The idea is to change the metric of the data representation in $\mathbf{X}$ such that the chosen metric best resembles the information which is inherent in this given non-parametric dimensionality reduction mapping.

We consider a global quadratic form for $\mathbf{X}$

$$d_{\mathbf{\Lambda}}(\mathbf{x}_i, \mathbf{x}_j)^2 = (\mathbf{x}_i - \mathbf{x}_j)^\mathsf{T} \mathbf{\Lambda} (\mathbf{x}_i - \mathbf{x}_j) \tag{4.4}$$

with a positive semidefinite matrix

$$\mathbf{\Lambda} = \mathbf{\Omega}^\mathsf{T} \mathbf{\Omega} \tag{4.5}$$

The goal is to learn $\mathbf{\Lambda}$ (or equivalently $\mathbf{\Omega}$) such that it best resembles the given visual display. Provided this metric change captures the relevant information of the visual display, it enables two things:

- It is possible to judge the relevance of the data dimensions for the given display by inspecting the relevance terms

  $$\Lambda_{ii} = \sum_j \Omega_{ji}^2 \tag{4.6}$$

  and hence gives a semantic interpretation of the display by linking it to the most relevant data dimensions (the ones with largest Λ_{ii}).

- It is possible to transform the data

  $$\mathbf{X} \mapsto \mathbf{\Omega X} \tag{4.7}$$

  to obtain data representations which more closely resemble the projections of the data in two dimensions; this opens the possibility to imprint information on the data based on the visual interface.

How can we obtain a suitable matrix $\mathbf{\Lambda}$? Mimicking the successful approach of relevance learning which has been established in supervised machine learning [17], we optimize $\mathbf{\Lambda}$ such that the objective as imposed by NeRV is optimized by an adjustment of $\mathbf{\Lambda}$, together with a suitable regularization:

$$E_k(\mathbf{\Omega}) = Q_k^{\mathrm{NeRV}}(\mathbf{\Omega X}, \mathbf{\Xi}) + \delta \cdot \mathrm{trace}(\mathbf{\Lambda}) \tag{4.8}$$

where $\delta > 0$ constitutes a small positive value which enforces solutions with a small norm for regularization. As before, we set σ in both spaces such that the fixed neighborhood size k is reached. While optimization with a gradient technique is possible, we again use an adaptive step size similar to well-known algorithms from neural network optimization [133]. Note that the derivatives of the costs $E(\mathbf{\Omega})$ can be computed based on the derivative of NeRV itself [163] using the following equality and symmetry of NeRV with respect to data points and projections

$$\frac{\partial E_k(\mathbf{\Omega})}{\partial \Omega_{ij}} = \sum_l \frac{Q_k^{\mathrm{NeRV}}(\mathbf{\Omega X}, \mathbf{\Xi})}{\partial (\mathbf{\Omega} \mathbf{x}_l)_i} \cdot (\mathbf{x}_l)_j + 2 \cdot \delta \cdot \Omega_{ij} \tag{4.9}$$

The transformation matrix $\mathbf{\Omega}$ is not unique since the costs are invariant with respect to orthonormal transformations of the matrix. This does not affect its trace (and hence the relevance terms which will be interpreted), however. Further, the result is not necessarily unique due to possible local optima of the costs which are inherent in NeRV; in practice, we did not observe problems.

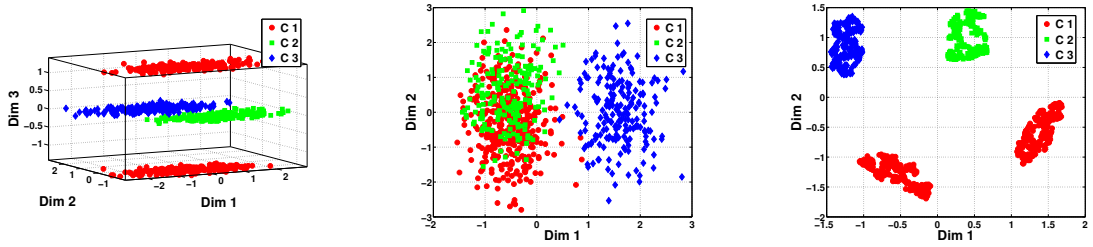


Figure 4.2.: Artificial multimodal data (left), projection by LDA (middle), projection by Fisher t-SNE (right)

4.2.5. Experiments

Experiments using feature selection and relevance learning

In this section we evaluate the concepts for estimating the relevance of features for a given visual display based on feature selection 4.2.2, and on relevance learning 4.2.3. All relevance measures yield a ranking of the dimensions according to their relevance for the visualization at hand, but only λ_{NeRV} can also be employed as a metric for the original data. In the following, we i) demonstrate how the methods work for a simple toy scenario, ii) we compare the rankings of the dimensions qualitatively for different projection types and iii) we show that the metric induced by λ^k_{NeRV} improves the similarity of the original and projected data. We also compare one of our relevance profiles to one from the literature and observe a large accordance. In all experiments we set $\gamma = 0.5$ and $\delta = 1$.

We utilize the following data sets in our experiments.

Data set1 contains three clusters with 20 points each in three dimensions, see Fig. 4.1. The third dimensions does not contain cluster information.

Data set2 contains three two-dimensional Gaussians arranged above each other along dimension 3. Although this dimension has the smallest variance, it is relevant for cluster separation.

Data set3 consists of ten features with three classes in the first two dimensions. The other dimensions contain increasingly noisy copies.

USPS refers to a data set of images [45], as introduced already in section 3.5. These show the handwritten digits from 0 to 9 and their size is 16×16. We randomly select 200 images per class.

Adrenal refers to a data set containing 147 patients characterized by 32 features [17, 5], which are various steroid markers. The data describe two different kinds of adrenal tumors.

Proof of concept As stated before, our proposed approaches are not specific to a certain DR technique which is used for computing the projections. Hence, we exemplarily

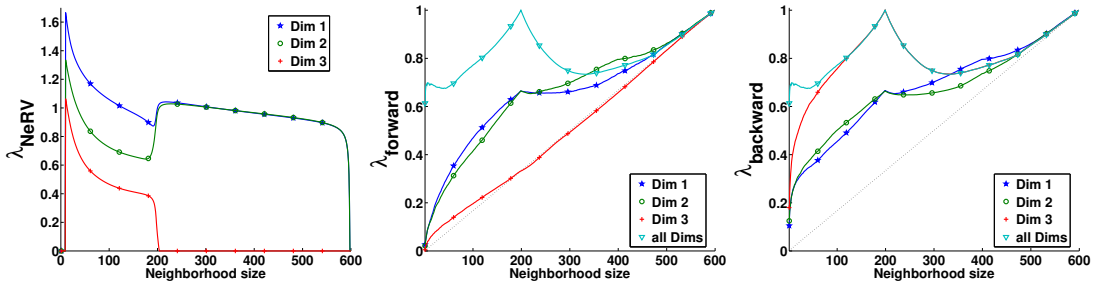


Figure 4.3.: Relevance determination for data set1 using λ_{NeRV} (left), λ_{forward} (middle) and $\lambda_{\text{backward}}$ (right).

apply t-SNE in order to project data set1 to two dimensions, yielding well-separated clusters. Applying our proposed relevance learning approach results in relevance profiles λ which are shown in Fig. 4.3 for varying k. As mentioned before, λ_{forward} results in a ascending feature relevance ranking while $\lambda_{\text{backward}}$ produces an descending one. The ranking of the dimensions induced by λ is identical for all techniques. λ_{NeRV} mirrors the irrelevance of dimension 3 as soon as the neighborhood exceeds the cluster size. Locally, all dimensions carry information because of the isotropic cluster shapes.

While the baseline for irrelevant dimensions in case of λ_{NeRV} is zero, the baseline for λ_{forward} is given by the diagonal, and the baseline for $\lambda_{\text{backward}}$ depends on the data and is given by the quality of the projection. As can be seen from Fig. 4.3 (middle and right), also the forward and backward relevance selection methods clearly mark dimension 3 as unimportant. Unlike λ_{NeRV}, these relevance schemes do not indicate the local importance of all dimensions.

Qualitative comparison for different mapping characteristics We compare the relevance ranks induced by the three schemes using different data and projection characteristics: We employ data set2 and set3. For the former, PCA and t-SNE lead to different map characteristics: PCA ignores dimension 3 because of the small variance while t-SNE emphasizes it. For data set3, projections of PCA, t-SNE and Fisher t-SNE are similar, with Fisher t-SNE better emphasizing the cluster structure which is mostly apparent in the first two dimensions.

We report the feature ranking of the most relevant features for different neighborhood sizes k (medium $\widehat{=} 0.8 \cdot$ cluster size, large $\widehat{=} 1.2 \cdot$ cluster size) in Table 4.1. The features of set2 in case of the PCA projection are ranked equally by all three approaches: dimension three plays only a minor role. This is plausible, since the PCA projection almost ignores this dimension. The ranks derived from the t-SNE projection of set2 also agree. Here the third dimension plays the major role since it separates the clusters, which is also done by the projection. For set3, the ranks of the dimensions assigned to by different methods vary. The high redundancy of this data set might be the reason for this. However, dimensions one and two have the noise-free class information and should be preferred in a discriminative setting. This is indeed the case

Table 4.1.: Feature ranking induced by the different techniques for set2 and set3. Fisher t-SNE is abbreviated via F t-SNE.

neighb.	medium	large	medium	large	medium	large
set2	λ_{NeRV}		λ_{forward}		$\lambda_{\text{backward}}$	
PCA	$(1,2) \gg 3$	$(1,2) \gg 3$	$(1,2) \gg 3$	$(1,2) \gg 3$	$(1,2) \gg 3$	$(1,2) \gg 3$
t-SNE	$3 \gg (1,2)$	$3 \gg (1,2)$	$3 \gg (1,2)$	$3 \gg (1,2)$	$3 \gg (1,2)$	$3 \gg (1,2)$
set3	λ_{NeRV}		λ_{forward}		$\lambda_{\text{backward}}$	
PCA	$3 > 2 \gg 1$	$(2,3) \gg 1$	$1 > 3 > 2$	$1 > 3 > 2$	$2 > (1,3,4)$	$2 > (1,3,4)$
t-SNE	$1 > 2 \gg 9$	$1 \gg 2 \gg 9$	$1 > (2,3)$	$1 > (2,3)$	$2 > 9 > 3$	$2 > (1,3,9)$
F t-SNE	$1 > 2 \gg 3$	$1 > 2 \gg 3$	$1 > (2,3)$	$1 > (2,3)$	$2 > (1,3,4)$	$2 > (1,3,4)$

for the Fisher-t-SNE projection. This redundancy cannot be accounted for by forward or backward selection, while λ_{NeRV}, optimizing simultaneously for all features, breaks ties in favor of the less noisy features 1 and 2. This result of forward/backward selection can be explained by its greedy nature which does not allow to take correlation adequately into account.

So far, we have seen that the proposed schemes can provide an interpretable relevance profile for a given DR projection. I.e. the methods provide a (gradually) sparse model which includes only few of the original data features, thus providing information about the original features. However, a direct interpretation of the coordinate axes of a low-dimensional projection is still not possible.

Suitability of induced feature transformation Finally, we demonstrate the suitability of the metric induced by λ_{NeRV} to imprint the information of the projection Ξ to $\mathbf{X}$. We evaluate this property by a comparison of the nearest neighbor (NN) error of the data in the projection space and the original data space $\mathcal{X}$ or its transformation, respectively. Thereby, we learn λ_{NeRV} based on a Fisher t-SNE mapping which also takes the available label information into account. We expect that the NN error improves in the latter setting for the transformed representation $\mathbf{X}_\lambda$ of $\mathbf{X}$. Further, we expect that the classification is also improved if standard t-SNE is applied to the data $\mathbf{X}_\lambda$.

We use the two data sets USPS and Adrenal for this purpose.

The results using λ_{NeRV}, which is learned on the Fisher t-SNE mapping, are reported in Table 4.2. The first three columns display the error in the original data space as well as in the t-SNE and the Fisher t-SNE projection. In columns four and five, the data are scaled with λ. The classification error reduces, if $\mathbf{X}$ is projected to two dimensions using t-SNE because of the elimination of noise, and even more so if Fisher t-SNE is used, i.e. the class information directs what is considered as noise. Interestingly, the classification improves when transforming the data according to the learned relevance from λ_{NeRV}, albeit only a linear transformation of the data takes place this way. This behavior is also preserved if a standard t-SNE projection is used on top of the feature transformation. Hence the results substantiate the possibility to change the data representation based on visual information this way, albeit the method is still limited to a

Table 4.2.: 1-NN errors in various data spaces of the data sets USPS and Adrenal.

neighb.				medium	large	medium	large
data sets	$\mathbf{X}$	t-SNE($\mathbf{X}$)	F-t-SNE($\mathbf{X}$)	$\mathbf{X}_\lambda$		t-SNE($\mathbf{X}_\lambda$)	
USPS	7.3%	6.7%	0.0%	2.2%	2.7%	3.1%	3.5%
Adrenal	10.9%	8.8%	0.7%	7.5%	6.8%	7.5%	6.8%

global linear weighting.

For the Adrenal data, we compare the relevance profile λ_{NeRV} based on our method, with relevances from [17], obtained differently. Interestingly, there is a large overlap of these two results as shown in Fig. 4.1 (right). Unlike [17] we can obtain this profile in one run of the algorithm making repetitions and thresholding as used in [17] superfluous.

Experiments using metric learning

Using the concepts of section 4.2.4, we investigate the possibility to substantiate a given visual display of data by metric learning, leading to relevance factors which allow a meaningful insight into the relevance of the data dimensionalities for the display, and leading to a more suitable representation of the data which imprints the information as provided by the visual display. While we can evaluate the former with a reference to the gained semantic insight, we evaluate the latter by the coranking framework which compares the neighborhood structure induced by the data representation and the visual display, respectively [93]. We consider the following three data sets:

Multimodal data refers to an artificially generated data set with known ground truth. Data are three dimensional, belonging to 3 classes, whereby one class is multimodal, see Fig. 4.2 (left). Dimension 1 is irrelevant for the cluster formation, dimension 2 discriminates the classes, dimension 3 discriminates the two modes in class 1.

Diabetes data refers to a data set describing 442 patients by 10 features (age, sex, BMI, blood pressure, 6 measurements taken from blood serum) with a labeling according to diabetes progression after one year. The data set has been used in [41], where a modern feature selection technique has marked three of the criteria as particularly relevant for the prediction task.

Adrenal As described in the previous section.

Artificial multimodal data We project the given data to two dimensions in two different ways: on the one hand, the discriminative linear discriminant analysis (LDA) is used, which projects the data linearly to the plane, preserving classes as indicated by the labels as much as possible. Since it relies on a unimodal Gaussian for every class,

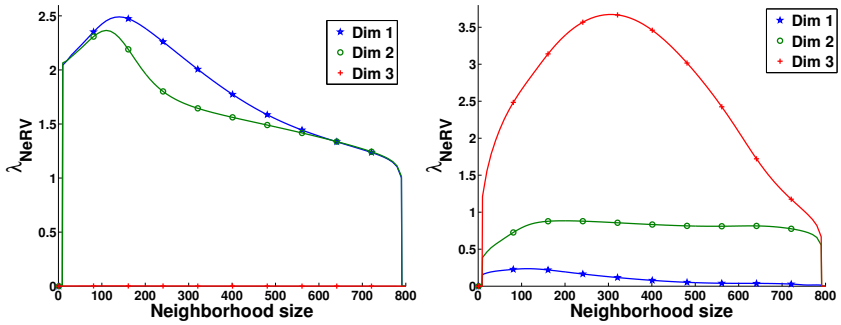


Figure 4.4.: Relevances Ω_{ii} obtained by the proposed method for the LDA projection (left) in dependency of the choice k of the cost function $E_k(\Omega)$, for the projection by Fisher t-SNE (right)

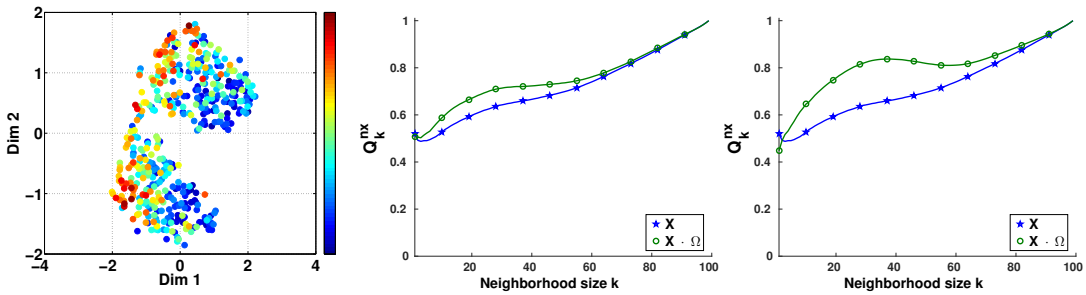


Figure 4.5.: T-SNE projection of the diabetes data set (left), quality for the t-SNE mapping for the standard Euclidean metric versus the transformed data with relevance matrix for neighborhood range 10 (middle) and 50 (right).

LDA is not capable of preserving the multi modality of class one, resulting in an overlap of classes one and two. In comparison, we use the non-linear projection technique t-SNE which is applied to the data as characterized by the Fisher information metric to take the label information into account (as discussed in section 2.4). The Fisher information metric curves the space locally such that the information most relevant to the given labeling is emphasized. On top of this curvature, t-SNE emphasizes the cluster structure and finds a corresponding two dimensional projection, displaying all four modes present in the data set (see Fig. 4.2).

We employ the proposed method from 4.2.4 to learn a global quadratic form, whereby we report the obtained results for different values of the neighborhood size k for the cost function E_k. The relevance terms Λ_{ii} for $i \in \{1, 2, 3\}$ and the two different projections are depicted in Fig. 4.4. The relevance terms clearly confirm the expectations if one interprets these two projections: LDA ignores the separation induced by the third dimension, treating the remaining two dimensions as equally important; this results in the failure to separate classes one and two. Fisher-t-SNE, in contrast, neglects the first dimension, which does not contain structure, but emphasizes the other two, such that all data modes are preserved. The relevance terms mirror this interpretation for

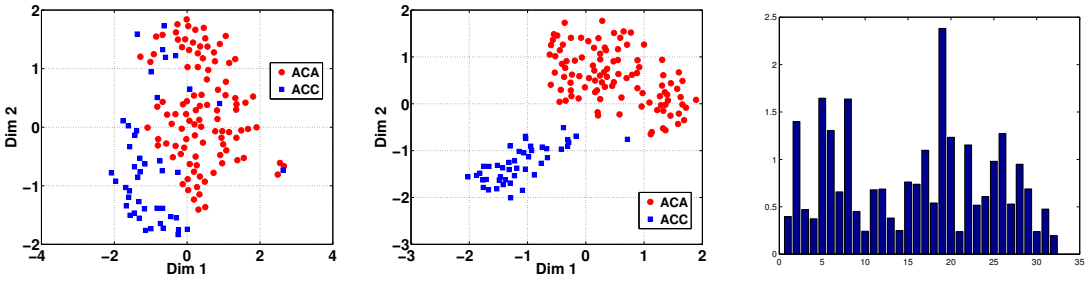


Figure 4.6.: Projection of the adrenal data using t-SNE (left) and Fisher t-SNE (middle). The latter can be used to learn the relevant factors for this discriminative visual display (right).

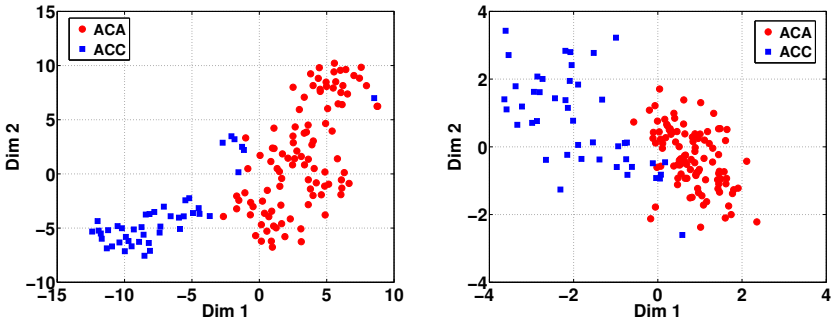


Figure 4.7.: Projection of the linearly transformed adrenal data using t-SNE (left). Projection onto the two main eigenvectors of the learned linear transformation (right).

all but extremal choices of the neighborhood degree k.

This example also elucidates the fact that matrix learning for a given visual display is different from classical feature selection: rather than emphasizing factors relevant for a given labeling, the proposed framework identifies factors which best explain the given visual display. These factors can coincide with the factors identified by feature selection provided the visual display emphasizes the given class labeling, but in general, this is not the case.

Diabetes data We project the given data using t-SNE to two dimensions. One can observe a correlation of the output and one projection axes, which is overlaid by a two cluster structure orthogonal to the output label (see Fig. 4.5 (left)). The t-SNE projection displays a reasonable quality as evaluated by the co-ranking framework (see Fig. 4.5 (middle/right)). In comparison, we transform the data according to the learned quadratic form for a neighborhood 10 and 50, respectively. As can be seen via the coranking framework, the transformed data, albeit relying on a linear transform only, much better resembles the information shown in the visual display. This confirms the possibility of imprinting information from the visual display to the given data representation for this medical data set.

Adrenal data For the adrenal data, we consider a projection of the original data by Fisher t-SNE, compared to a projection of the data by standard t-SNE (see Fig. 4.6). Interestingly, the 1-nearest neighbor classification error of the original data set is 10.9%, as also mirrored in the t-SNE projection which displays quite some overlap of the data, while the error drops down to only 0.7% for the Fisher t-SNE projection. We can imprint the information available in this discriminative projection to the data by means of relevance learning, as before. We learn a quadratic form with neighborhood range $k = 10$ of the costs, resulting in relevance factors which strongly resemble the findings as described in the publication [18]. This profile is very consistent for different choices of neighborhood range k (we tested values $k \in \{10, 20, 40\}$ which lead to qualitatively the same result). As before, we can imprint this information onto the original data by means of an according data transformation. The t-SNE projection of the linearly transformed data is depicted in Fig. 4.7, the 1-nearest neighbor error reduces to 3.4% (as compared to trice as much for the original data). Note that, unlike the Fisher information metric, the data are subject to a simple linear data transform only as regards its representation, followed by the non-parametric t-SNE mapping. Interestingly, the obtained linear data transformation even suggests a linear data display with almost the same quality: Fig. 4.7 also displays the linear projection to the first two eigenvectors of the learned data transformation. The 1-nearest neighbor error is 2.7% only, enabling a very efficient representation of the data which mirrors the underlying label information. For both cases, one point is clearly indicated as an outlier (possibly corresponding to a mislabeling of the data point, as also discussed in the publication [18]). Due to its possibility to follow strong nonlinearities caused by its non-parametric nature, the Fisher information metric itself tends to overfit in this region, such that this outlier is much less pronounced in the Fisher t-SNE mapping (Fig. 4.6).

Resumee

We have investigated seven data sets as concerns the possibility to link their visual displays to explicit relevance terms which link the displayed points to a semantic meaning, and which open an interface towards imposing this information to the data representation by means of a linear transform. The tasks at hand being unsupervised, the evaluation of these possibilities it not straightforward. In our experiments, we demonstrated the claims in the following way:

- We evaluated the relevance and matrix learning frameworks for artificial data with known relevances for the given visual displays. The found relevances confirm the expectation in these settings.
- We evaluated the possibility to imprint the information shown in the visual display to the data by means of a linear data transformation by using the co-ranking framework for data visualization for a real life data set.
- We evaluated the possibility to imprint the information as shown in the visual display by a reference to the nearest neighbor error in the case of an initial su-

pervised dimensionality reduction. Here, the transformed data clearly allow to achieve a better nearest neighbor error, i.e. a data transformation as learned from the initial discriminative visual display of the data enables us to obtain an alternative data representation which better resembles this important aspect. Thereby, due to the linearity of the transformation, a semantic interpretation of the axes is still possible.

So far, by restricting to a global quadratic form, the induced data transformation is linear. Note that, similar to proposals in supervised metric learning, a generalization of the approach to locally quadratic forms (and hence a globally non-linear data projection) would be possible [13]. However, the choice of the tesselation of the data space is not trivial.

4.3. Valid interpretation of feature relevance for linear data mappings

Linear (or locally linear) data transformations constitute a particularly prominent element in machine learning which seemingly combines efficient and well founded training algorithms with interpretable model components. Global linear models such as ridge regression, linear discriminant analysis, or principal component analysis constitute premier techniques in many application domains in particular if high data dimensionality is involved [147]. Besides, the very active field of metric learning usually aims for an adaptive quadratic form, which essentially corresponds to a linear transformation of the data. Many different successful approaches have recently been proposed in this context, see e.g. [13, 17].

One of the striking properties of linear models is that they seemingly allow an interpretation of the relevance of input features by inspecting their corresponding weighting; in a few cases, such techniques have led to striking semantic insights of the underlying process [5]. Thus, these models carry the promise of fast and flexible learning algorithms, which directly address a simultaneous, quantitative, and interpretable weighting of the given features, provided linear data modeling is appropriate.

Recent results, however, have shown that the interpretation of linear weights as relevance terms can be extremely misleading in particular for high-dimensional data [149]: those data likely display correlations of the features, hence relevance terms can be high due to purely statistical effects of the data. Conversely, highly correlated but very important features can be ranked low due to the fact that they share their impact. In the contribution [149] a first cure to partially avoid these effect by a L_2 regularization has been proposed. In particular in the case of feature correlations, however, the approach still fails to provide efficient bounds for the minimum and maximum feature relevance; hence it offers a partial solution of the problem only. Here, we propose a L_1 regularization instead, which allows an efficient formalization of the minimum and maximum feature relevance as a linear programming problem. Since many recent

datasets are characterized by their high dimensionality, this constitutes a crucial step for feature relevance interpretability in many modern domains.

Very high data dimensionality is becoming more and more prominent. For example, in omics studies, many genes are simultaneously considered [15, 109]. Even if having more information may seem beneficial at first glance, this wealth of features can also be problematic. Indeed, machine learning in high-dimensional space suffers from the curse of dimensionality [14, 165], also known as the empty space phenomenon. This is due to the fact that the size of a dataset should scale exponentially with its dimensionality, what cannot be achieved in practice. Other counterintuitive phenomena like the concentration of distances [44] occur, what causes distances to be less useful in high-dimensional spaces. Eventually, high-dimensional data are harder to analyze and to visualize for human experts. As argued above, direct feature ranking in linear maps can easily loose its interpretability in this situation.

Feature selection [59] is a common preprocessing for high-dimensional data, and we will compare our modeling to classical feature selection. Feature selection consists in selecting a few relevant features which allow reaching good prediction performances with easy-to-interpret models. For example, least angle regression (LARS) [41, 64] obtains sparse feature subsets for linear regression. Many methods have been proposed for non-linear models, based e.g. on mutual information [7, 166, 140, 38, 47, 46, 164]. Such solutions improve the performances of subsequently used machine learning algorithms. In our setting, we are not so much interested in a sparse linear representation, rather we address the question, given a linear mapping, what is the relevance of features for the given mapping, taking into account all possible invariances inherent in the data. Concerning this question, classical feature selection, though very powerful, is not entirely satisfying when it comes to interpretability. Indeed, most feature selection algorithms only provide either a unique subset of features or a path of feature subsets of increasing size. This leaves out an important part of the information. For example, if two relevant features are linearly dependent, the LARS algorithm may arbitrarily include any of them in the feature subset. This may incorrectly suggest that the other feature is irrelevant. Also, most feature selection methods do not specify which features are strictly necessary, what may be interesting to understand the system under study.

These limitations of feature selection can be alleviated using the concept of strong and weak relevance [75, 81, 111]. Strongly relevant features provide new information, even if all other features are already used. Weakly relevant features may provide new information, but only if certain features (e.g. redundant ones) are not simultaneously considered. In general, the determination of weakly relevant features requires exhaustive search over all feature subsets [111]. In this paper, we restrict to linear mappings only, ignoring possible nonlinear effects. We are interested in the relevance of the features for the given mapping, aiming at both, strong and weak feature relevance. We do not strictly follow the formal definition of strong and weak feature relevance for linear settings, but we will use a different formalization which is inspired by these terms but

allows efficient modeling. Essentially, we will consider two weight vectors of a given mapping as equivalent, if they have the same (or a similar) classification behavior and the same (or similar) length of the weight vector, thus accounting for a similar signal to noise ratio or generalization ability, respectively. Then we propose a measurement similar to weak and strong feature relevance by the minimum and maximum weight of a feature in this equivalence class. These bounds give an interpretable interval for the feature relevance.

This section is organized as follows. First, section 4.3.1 discusses the problem of weak and strong relevance for linear relationships. The concept of bounds for feature relevance is introduced, as well as a simple, generic reference algorithm. Section 4.3.2 proposes a new algorithm to find strongly and weakly relevant features for linear models (and the corresponding feature relevance bounds), while section 4.3.3 describes how the concept of metric learning can be reframed in terms of linear mappings. Experiments are performed in section 4.3.4.

4.3.1. Definition and measure of feature relevance

This section defines the concept of feature relevance and discusses a simple algorithm to quantify it, aiming at approximations of the formal concept of weak and strong feature relevance. For linear mappings, a similar mathematical definition is proposed in section 4.3.2 which resembles the underlying ideas but directly gives rise to an efficient solution.

Feature relevance

The question what means feature relevance has been extensively discussed, see e.g. the survey [10] and the approaches [152, 178]. The notion of strong and weak feature relevance has been defined in [75, 81, 111]. Assume the task is to predict a target Y based on D features $X_1 \dots X_D$, which can be either continuous (regression) or discrete (classification). A variable Y is *conditionally independent* of a variable X_j given a set of variables S, if $P(Y|X_j, S) = P(Y|S)$. This is denoted as $Y \perp\!\!\!\perp X_j|S$. A feature X_j is *strongly relevant* to predict Y iff

$$Y \not\perp\!\!\!\perp X_j|X_{(j)} \tag{4.10}$$

where $X_{(j)}$ is the set of all features except X_j. Strongly relevant features are strictly necessary to achieve good prediction, since they contain some information which is not provided by any other feature. Finding theses features is particularly interesting to understand the studied process, since these features are likely to play a key role.

A feature X_j is defined as *weakly relevant* to predict Y iff it is not strongly relevant and

$$Y \not\perp\!\!\!\perp X_j|S \tag{4.11}$$

for some feature subset $S \subset X_{(j)}$. A weakly relevant feature is not necessarily useful, since it provides information which is also contained in other features. Indeed, $Y \perp\!\!\!\perp X_j|X_{(j)}$ holds if the feature X_j is not strongly relevant (first part of the definition).

This can occur if X_j is redundant with other features, for example. Nonetheless, experts are often still interested in such features: some weakly relevant features are often necessary for a good model accuracy, albeit the choice is not necessarily unique. Further, weakly relevant features are often crucial to understand the complex relationships between the features and the target. One example is explained in [111]: in gene expression analysis, experts '*are primarily interested in identifying all features (genes) that are somehow related to the target variable, which may be a biological state such as "healthy" vs. "diseased"*' [58, 146].

Searching for relevant features

Under reasonable assumptions, generic (but potentially time consuming) algorithms are proposed in [111] to find strongly and weakly relevant features. We recall this procedures for convenience. Strongly relevant features can be found by selecting all features whose removal lowers the prediction performance. Assume there is given a classifier with prediction error $c(S)$ based on the feature set S. Then these features corresponds to the subset $\left\{X_j | c\left(X_{(j)}\right) > c(X) + \epsilon\right\}$ where the parameter $\epsilon > 0$ controls the trade-off between prediction and recall [111]. This backward procedure is efficient, since this criterion must only be estimated D times.

Weakly relevant feature are much harder to find. When directly testing the definition, one has to consider the $\mathcal{O}\left(2^D\right)$ possible feature subsets $S \subset X_{(j)}$ for the conditional dependence $Y \not\perp\!\!\!\perp X_j | S$. In practice, such an exhaustive search is not affordable and one has to rely on heuristics to find weakly relevant features. For example, the recursive independence test (RIT) algorithm [111] first finds the features X_j satisfying $Y \not\perp\!\!\!\perp X_j$. Then, it recursively adds all the other features $X_{j'}$ which are pairwise dependent with respect to those features, i.e. $X_j \not\perp\!\!\!\perp X_{j'}$. For each step, a (specific) statistical independency test is required.

Bounds for feature relevance

The algorithms described in the previous paragraph find sets of relevant features, whereby weakly relevant features can only approximately be determined efficiently. We are interested in a yet different setting: on the one hand, we do not necessarily consider a clear objective such as the classification error, rather our goal is to interpret the relevance of features for a given linear mapping and data set. In addition, we are not only interested in qualitative results, indicating a feature as relevant or irrelevant, respectively. Rather, we would like to identify an interval for every feature, which quantifies the minimum and maximum relevance the feature might have for the given mapping. Thus, such bounds should not only indicate whether features are strongly or weakly relevant, but also *how much* they are relevant. A non-zero lower bound indicates that a feature is strongly relevant, whereas a large upper bound points out that the feature is at least weakly relevant.

In the following, we will focus on linear relationships, which are common in biomedicine or social sciences, and particularly interesting for the case of high data dimensionality, i.e. a potentially large number of correlated features. In the following paragraph, inspired by the formal notion of strong and weak feature relevance, we propose a generic approach which is suitable for low dimensionalities and which can serve as a basic comparison. Afterwards, in section 4.3.2, we propose an efficient method to compute feature relevance bounds. These are tested in section 4.3.4.

Generic approach to compute feature relevance bounds

Using the same idea as the algorithm in [111] which finds strongly relevant features (see section 4.3.1), the following algorithm computes lower bounds for the feature relevance.

Algorithm 1 Compute lower bounds for feature relevance

Input: criterion c and dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1\ldots N}$
Output: lower bound l_j for each feature X_j

compute $c(\mathcal{D})$
for $j = 1 \ldots D$ **do**
  $l_j \leftarrow c\left(\mathcal{D}_{X_{(j)}}\right) - c(\mathcal{D})$
end for

Here, $\mathcal{D}_{X_{(j)}}$ is the dataset restricted to the features $X_{(j)}$ and c measures the error of a feature subset to predict Y. Hence, the difference $c\left(\mathcal{D}_{X_{(j)}}\right) - c(\mathcal{D})$ can be interpreted as the minimum contribution of X_j to the total relevance. This quantity is used as a lower bound l_j to the relevance of feature X_j. It is non-zero if X_j is strongly relevant.

For upper bounds, an exhaustive search would be necessary, but intractable in practice. Instead, a greedy forward-backward search is used in the following algorithm. Here, $\mathcal{C}$ and $\mathcal{S}$ are the subsets of candidate and selected features, respectively. If c is the mean square error, the quantity $c(\mathcal{D}_\emptyset)$ is defined as the target variance. Also, NB_FB_STEPS is the number of backward and forward steps which are performed. Using greedy algorithms like the following forward-backward search is a standard approach in feature selection. Even if it is not optimal, it often gives good results. The particularity of this greedy search is that the search criterion is the upper bound itself. In other words, the algorithm searches for the feature subset which allows a given feature to be as useful as possible. The number of steps is deliberately limited because (i) weakly relevant features are unlikely to be highly relevant when a lot of other features are simultaneously considered and (ii) the estimation of c is often less reliable when the dimensionality increases. Also, computing the upper bounds with Alg. 2 requires to evaluate $\mathcal{O}(D^2 \times \text{NB\_FB\_STEPS})$ times the criterion c. It is therefore necessary to use a small value for NB_FB_STEPS. Here, we use NB_FB_STEPS $= 6$ as a compromise between accuracy and efficiency.

Algorithm 2 Compute upper bounds for feature relevance

Input: criterion c, dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1\ldots N}$
lower bounds l_j for every feature X_j
Output: upper bound u_j for each feature X_j

compute $c(\mathcal{D}_\emptyset)$
for $j = 1 \ldots D$ **do**
// initialise upper bound
$u_j \leftarrow \max\left(l_j, c(\mathcal{D}_\emptyset) - c\left(\mathcal{D}_{X_j}\right)\right)$
$\mathcal{C} \leftarrow \{1 \ldots D\} \setminus \{j\}$
$\mathcal{S} \leftarrow \emptyset$

// forward search steps
for $s = 2 \ldots$ NB_FB_STEPS **do**
// find next feature to add to $\mathcal{S}$
for $k \in \mathcal{C}$ **do**
$\Delta c_k = c\left(\mathcal{D}_{X_{\mathcal{S}\cup\{k\}}}\right) - c\left(\mathcal{D}_{X_{\mathcal{S}\cup\{j,k\}}}\right)$
end for
$k^* \leftarrow \arg\max_{k\in\mathcal{C}} \Delta c_k$
$u_j \leftarrow \max\left(u_j, \Delta c_{k^*}\right)$

$\mathcal{C} = \mathcal{C} \setminus \{k^*\}$
$\mathcal{S} = \mathcal{S} \cup \{k^*\}$
end for

// backward search steps
for $s =$ NB_FB_STEPS $\ldots 2$ **do**
// find next feature to remove from $\mathcal{S}$
for $k \in \mathcal{S}$ **do**
$\Delta c_k = c\left(\mathcal{D}_{X_{\mathcal{S}\setminus\{k\}}}\right) - c\left(\mathcal{D}_{X_{\mathcal{S}\setminus\{k\}\cup\{j\}}}\right)$
end for
$k^* \leftarrow \arg\max_{k\in\mathcal{C}} \Delta c_k$
$u_j \leftarrow \max\left(u_j, \Delta c_{k^*}\right)$

$\mathcal{S} = \mathcal{S} \setminus \{k^*\}$
end for
end for

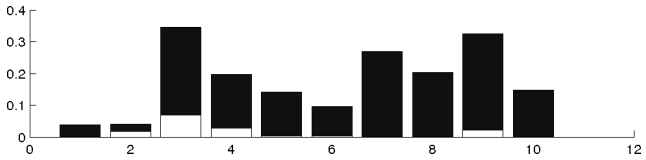


Figure 4.8.: Lower and upper bounds of feature relevance given by Alg. 1 and Alg. 2 for the diabetes dataset. c is the mean square error of a linear regression.

Fig. 4.8 shows the lower and upper bounds obtained for the diabetes dataset used in the original LARS paper [41]. The 10 features for the 442 patients are the age, the sex, the body mass index (BMI), the blood pressure (BP) and 6 blood serum measurements $X_5 \dots X_{10}$. The goal is to predict a measure Y of diabetes progression one year after feature acquisition. Fig. 4.8 shows that the BMI X_3, the BP X_4 and the serum measurement X_9 are particularly informative; this is confirmed by the results of LARS obtained by Efron et al. [41].

Notes on the error criterion and the generic algorithms

In this paper, c is the mean square error, since we focus on linear regression. However, the above discussion and the two suggested algorithms remain valid for non-linear regression using e.g. a kNN like in [111]. Also, other criteria can be used, like the (estimated) conditional entropy $c\left(\mathcal{D}\right) = \hat{H}(Y|X)$. The difference $c\left(\mathcal{D}_{X_{(j)}}\right) - c\left(\mathcal{D}\right)$ becomes the (estimated) conditional mutual information $\hat{I}\left(X_j; Y|X_{(j)}\right) = \hat{I}\left(X_{(j)} \cup \left\{X_j\right\}; Y\right) - \hat{I}\left(X_{(j)}; Y\right)$, i.e. the additional information in X_j about Y. Entropies can be estimated with the Kozachenko-Leonenko estimator [85, 86, 140, 38]. Similar approaches exist in feature selection [129, 112], but they do not derive bounds.

The above algorithms have several drawback. First, the criterion c has to be computed for each feature subsets. Second, when the number of features D increases, the lower bounds tend to zero because of overfitting. Third, the used algorithm for the upper bounds is a heuristic, since forward-backward search is not exhaustive. Eventually, the overall computational cost is quadratic w.r.t. the dimensionality D. However, these two algorithms can still provide excellent points of comparison in section 4.3.4 due to their strong resemblance of the weak and strong relevance of features.

Classical linear feature selection

In the context of a linear or generalized linear mapping f, a popular technology for feature selection is offered by LASSO and variants [49]. Assume $f(\mathbf{x}) = \boldsymbol{\omega}^\mathsf{T}\mathbf{x}$. Then LASSO relies on a L1-regularized optimization of the mapping parameters

$$\min \frac{1}{2} \cdot \sum_{i=1}^{N} (f(\mathbf{x}_i) - y_i)^2 \text{ such that } \sum_{j=1}^{D} |\omega_j| \leq s \tag{4.12}$$

for a sparsity constant $s > 0$. The constraint can be integrated into the objective as a penalty term with a fixed weighting. Ridge regression penalizes the L2 instead of L1 norm, and Elastic Net addresses a mixture of both objectives [49], where, depending on the weighting of the penalty, different degrees of sparsity can be enforced. It is possible to infer the relevance of a given feature X_j from the size of the resulting weight $|\omega_j|$, whereby a varying penalty also can shed some light on the question whether the feature is weakly / strongly relevant.

Inspired by this observation, we will use L1 regularization for the valid interpretation of relevance terms of a given mapping. Thereby, we separate the question of how to train the mapping and how to interpret the feature relevance. This strategy, together with a slight reformulation of the regularization, enables us to derive intervals for the possible relevance range of a given feature.

4.3.2. Linear bounds

We are interested in the interpretation of a given linear mapping $f(\mathbf{x}) = \boldsymbol{\omega}^\mathsf{T}\mathbf{x} \in \mathbb{R}$ with $\boldsymbol{\omega} \in \mathbb{R}^D$, which we assume to map to a one-dimensional space, for simplicity. Generalizations to higher dimensions such as present in metric transformation, for example, are immediate (i.e. treat each one-dimensional mapping independently and aggregate the results). We assume that this mapping either comes from a regression or classification task such as ridge regression, LARS, LASSO, or it arises from a quadratic metric adaptation method which corresponds to a linear transformation of the data space (further discussed in 4.3.3). For a given linear mapping, the value $|\omega_j|$ is often taken as a direct indicator of the relevance of feature X_j provided the input features have the same scaling, i.e. the values delivered by a linear mapping are directly interpreted. As pointed out in [149], this is highly problematic: For correlated features, which is often the case, in particular for high-dimensional data, the absolute value of ω_j can be very misleading. The approach [149] formalizes this observation based on a mathematical treatment in the form of mapping invariances.

First, we define the central notion of invariance. This concept will serve as criterion based on which feature ranking will be derived, i.e. it will take the role of the criterion c from the previous section. Given a mapping $f(\mathbf{x}) = \boldsymbol{\omega}^\mathsf{T}\mathbf{x}$ and data $\mathbf{X}$ consisting of a matrix with data vectors $\mathbf{x}_i$, we define that $\boldsymbol{\omega}$ is *equivalent* to $\boldsymbol{\omega}'$ iff

$$\boldsymbol{\omega}^\mathsf{T}\mathbf{X} = (\boldsymbol{\omega}')^\mathsf{T}\mathbf{X} \tag{4.13}$$

i.e. the mapping of the data is not changed when substituting $\boldsymbol{\omega}$ by $\boldsymbol{\omega}'$. Unlike a pre specified criterion c such as the accuracy, this notion directly relates to the behavior of the mapping on the given data only. The approach [149] exactly characterizes under which condition $\boldsymbol{\omega}$ is equivalent to $\boldsymbol{\omega}'$: two vectors $\boldsymbol{\omega}$ and $\boldsymbol{\omega}'$ are equivalent iff the difference vector $\boldsymbol{\omega} - \boldsymbol{\omega}'$ is contained in the null space of the data covariance matrix $\mathbf{X}\mathbf{X}^\mathsf{T}$. The covariance matrix has eigenvectors $\mathbf{v}_i$ with eigenvalues $\lambda_1 \geq \ldots \geq \lambda_I > \lambda_{I+1} = \ldots = \lambda_D = 0$ sorted according to their size, whereby I denotes the number of non zero eigenvalues.

In [149] it is proposed to choose one canonic representation $\boldsymbol{\omega}'$ of the equivalence class induced by a given $\boldsymbol{\omega}$ before interpreting the values: one considers the vector $\boldsymbol{\omega}'$ which results by dividing the null space of the covariance matrix; $\boldsymbol{\omega}$ becomes $\boldsymbol{\omega}' = \Psi\boldsymbol{\omega}$ where

$$\Psi = \mathbf{I} - \sum_{i=I+1}^{D} \mathbf{v}_i \mathbf{v}_i^\mathsf{T}$$

denotes the matrix which corresponds to the projection of $\boldsymbol{\omega}$ to the eigenvectors with non zero eigenvalues only induced by the eigenvectors $\mathbf{v}_i$ of the matrix $\mathbf{X}\mathbf{X}^\mathsf{T}$. Hence the eigenvectors with eigenvalue zero are divided out. It has been shown in the approach [149] that this choice of a representative corresponds to the vector in the equivalence class with smallest L_2 norm.

This has the result, that it is no longer possible to assign a high value ω_j to an irrelevant feature based on random effects of the data, i.e. strongly relevant features are identified and spurious relevances which are solely due to data correlations are no longer possible. While providing a unique representative of every equivalence class, this choice is problematic as concerns the direct interpretability of the values: Weakly relevant features share the total relevance of the features uniformly. Hence a feature which is highly correlated to a large number of others is always weighted low, independent of the fact that the information provided by this feature (or any equivalent one) might be of high relevance for the linear mapping prescription. Thus, assuming a practitioner interprets such a relevance profile, there is a high risk that all such features are discarded, albeit they are of high importance, but mutually redundant. In the following, we propose an alternative to choose representatives which are equivalent to $\boldsymbol{\omega}$ but which allow a direct interpretation of the weight vector. Essentially, we will not consider the representative with smallest L_2 norm, but use the L_1 norm instead. Unlike the former, the latter induces a set of equivalent weights which have minimal L_1 norm. We can infer the minimum and maximum relevance of a feature by looking at the minimum and maximum weighting of the feature within this set. Then, a practitioner can choose the best suited one among weakly relevant sets based on additional criteria (such as the costs which arise by measuring this feature).

Formalizing the objective

Given a parameter vector $\boldsymbol{\omega}$ of a linear mapping, we are interested in equivalent vectors, i.e. vectors of the form

$$\boldsymbol{\omega}' = \boldsymbol{\omega} + \sum_{i=I+1}^{D} \alpha_i \mathbf{v}_i \tag{4.14}$$

for real valued parameters α_i which add the null space of the mapping to the vector $\boldsymbol{\omega}$. We want to avoid random scaling effects of the null space, therefore we choose minimum vectors only, similar to the approach [149] . Unlike the L_2 norm, however, we use the L_1 norm:

$$\mu \leftarrow \min_{\boldsymbol{\alpha}} \left\| \boldsymbol{\omega} + \sum_{i=I+1}^{D} \alpha_i \mathbf{v}_i \right\|_1 . \tag{4.15}$$

The value of the minimum μ is unique per definition. This is not the case for the corresponding vector $\boldsymbol{\omega} + \sum_{i=I+1}^{D} \alpha_i \mathbf{v}_i$. A very simple case illustrates this fact: assume identical features $X_i = X_j$ and a weighting ω_i and ω_j. Then any weighting $\omega_i' = t \cdot \omega_i + (1-t)\omega_j$ and $\omega_j' = (1-t)\omega_i + t\omega_j$ yields an equivalent vector with the same L_1 norm.

This observation enables us to formalize a notion of minimum and maximum feature relevance for a given linear mapping: the *minimum feature relevance* of feature X_j is the smallest value of a weight $|\omega_j'|$ such that $\boldsymbol{\omega}'$ is equivalent to $\boldsymbol{\omega}$ and $|\boldsymbol{\omega}'|_1 = \mu$. The *maximum feature relevance* of feature X_j is the largest value of a weight $|\omega_j'|$ such that $\boldsymbol{\omega}'$ is equivalent to $\boldsymbol{\omega}$ and $|\boldsymbol{\omega}'|_1 = \mu$. In mathematical terms, this corresponds to the following optimization problems:

$$\begin{aligned} \underline{\omega}_j \leftarrow \min_{\boldsymbol{\alpha}} & \left| \omega_j + \sum_{i=I+1}^{D} \alpha_i (\mathbf{v}_i)_j \right| \\ \text{s.t.} & \quad \left\| \boldsymbol{\omega} + \sum_{i=I+1}^{D} \alpha_i \mathbf{v}_i \right\|_1 = \mu \end{aligned} \tag{4.16}$$

and

$$\begin{aligned} \overline{\omega}_j \leftarrow \max_{\boldsymbol{\alpha}} & \left| \omega_j + \sum_{i=I+1}^{D} \alpha_i (\mathbf{v}_i)_j \right| \\ \text{s.t.} & \quad \left\| \boldsymbol{\omega} + \sum_{i=I+1}^{D} \alpha_i \mathbf{v}_i \right\|_1 = \mu. \end{aligned} \tag{4.17}$$

where $(\mathbf{v}_i)_j$ refers to component j of $\mathbf{v}_i$. This framework yields a pair $(\underline{\omega}_j, \overline{\omega}_j)$ for each feature X_j indicating the minimum and maximum weight of this feature for all equivalent mappings with the same L_1 norm. This strongly resembles the notion of strong and weak feature relevance in the special case of linear mappings and the mapping invariance as objective.

Note that this framework does not exactly realize the notion of strong and weak feature relevance in a strict sense due to the following reason: we aim for scaling terms as observed in the linear mapping, which are subject to L_1 regularization. This has the consequence that two features which have the same information content but which are scaled differently are not treated as identical by this formalization. Rather, the feature with the better signal to noise ratio which corresponds to a smaller scaling of the corresponding weight is preferred. Qualitative feature selection would treat such variables identically.

There exist natural relaxations of this problem as follows: In Eq. (4.14), we can incorporate eigenvectors which correspond to small eigenvalues, thus enabling an only approximate preservation of mapping equivalence. Further, we can relax the equality in Eq. (4.15) to allow values which do not exceed $\mu + \epsilon$ instead of μ for some small $\epsilon > 0$. Such relaxations with small values ϵ are strongly advisable for practical applications to take into account noise in the data. We will use these straight-forward approximations in experiments.

Reformalization as linear programming problem

For an algorithmic solution, we rephrase these problems as linear optimization problems (LP). We reformulate problem (4.16) as the following equivalent LP where we introduce a new variable $\tilde{\omega}_k$ for every k which takes the role of $|\omega_k + \sum_{i=I+1}^{D} \alpha_i (\mathbf{v}_i)_k|$:

$$\begin{aligned} \underline{\omega}_j \leftarrow \min_{\tilde{\boldsymbol{\omega}},\boldsymbol{\alpha}} & \tilde{\omega}_j, \\ \text{s.t.} \quad & \sum_{i=1}^{D} \tilde{\omega}_i \leq \mu \\ & \tilde{\omega}_k \geq \omega_k + \sum_{i=I+1}^{D} \alpha_i (\mathbf{v}_i)_k, \forall k \\ & \tilde{\omega}_k \geq -\left(\omega_k + \sum_{i=I+1}^{D} \alpha_i (\mathbf{v}_i)_k \right), \forall k, \end{aligned} \tag{4.18}$$

where μ is computed in (4.15) and the variables $\tilde{\omega}_i$ must be non negative due to the constraints. For the optimum solution, we can assume that equality holds for one of the two constraints for every k; otherwise, the solution could be improved due to the weaker constraints and the minimization of the objective. For problem (4.17), we use the equivalent formulation

$$\begin{aligned} \max_{\tilde{\boldsymbol{\omega}},\boldsymbol{\alpha}} & \left| \omega_j + \sum_{i=I+1}^{D} \alpha_i (\mathbf{v}_i)_j \right|, \\ \text{s.t.} \quad & \sum_{i=1}^{D} \tilde{\omega}_i \leq \mu \\ & \tilde{\omega}_k \geq \omega_k + \sum_{i=I+1}^{D} \alpha_i (\mathbf{v}_i)_k, \forall k \\ & \tilde{\omega}_k \geq -\left(\omega_k + \sum_{i=I+1}^{D} \alpha_i (\mathbf{v}_i)_k \right), \forall k, \end{aligned} \tag{4.19}$$

where, again, new variables $\tilde{\omega}_k$ are introduced. Again, these take the role of the absolute value $|\omega_k + \sum_{i=I+1}^{D} \alpha_i (\mathbf{v}_i)_k|$: any solution for which equality does not hold for one of the constraints can be improved due to the weaker constraints and maximization as the objective. This is not yet a LP since an absolute value is optimized. For its solution, we can simply solve two LPs where we consider the positive and negative value of the objective:

$$\overline{\omega}_j^{\pm} \leftarrow \max_{\tilde{\boldsymbol{\omega}},\boldsymbol{\alpha}} \pm \left(\omega_j + \sum_{i=I+1}^{D} \alpha_i (\mathbf{v}_i)_j \right),$$

and we add the corresponding non negativity constraint

$$\pm \left(\omega_j + \sum_{i=I+1}^{D} \alpha_i (\mathbf{v}_i)_j \right) \geq 0$$

At least one of these LPs has a feasible solution, and the final upper bound can be derived thereof as the maximum value

$$\overline{\omega}_j = \max\{\overline{\omega}_j^+, \overline{\omega}_j^-\}$$

This approach requires to solve LP problems containing $2D$ constraints and $I+1$ variables. Standard solver can be applied.

4.3.3. Metric learning as linear data transformation

Metric learning has been introduced in distance based machine learning models as a means to autonomously adjust the underlying distance measure to the given task at hand [13, 89]. Here, we focus on two popular metric learning schemes only. Since the proposed technique for metric interpretation is separated from the metric learning step itself, the proposed regularization for feature relevance determination can be used for every metric adjustment scheme which arrives at a general quadratic form, as utilized in the following. This includes the metric learning for DR scheme which has been proposed in section 4.2.4.

We rely on a distance measure which is given by a general quadratic form

$$d : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}, (\mathbf{x}_i, \mathbf{x}_j) \mapsto (\mathbf{x}_i - \mathbf{x}_j)^\mathsf{T} \mathbf{\Lambda} (\mathbf{x}_i - \mathbf{x}_j) \tag{4.20}$$

with the positive semi-definite matrix $\mathbf{\Lambda} = \mathbf{\Omega}^\mathsf{T}\mathbf{\Omega}$. Optimizing $\mathbf{\Omega}$ increases not only the performance of metric learning methods, but it also offers an interpretation of the feature relevance in terms of its diagonal $\Lambda_{ii} = \sum_j \Omega_{ji}^2$ or related terms, since the metric (4.20) corresponds to the linear data transformation

$$\mathbf{x} \mapsto \mathbf{\Omega}\mathbf{x}. \tag{4.21}$$

Hence interpretation of the matrix relevance terms reduces to the interpretation of this linear mapping.

Examples of popular metric learners employing such a parametrization are as follows:

- Large margin nearest neighbor (LMNN) [174] adjusts this matrix in such a way that the k-NN error induced by this distance is optimized. More precisely, it fixes the k nearest neighbors for every given data point, and adjusts the matrix $\mathbf{\Omega}$ such that points with the same label in this neighborhood are close, while points with a different label are separated by a distance term with a margin at least one.

- Generalized matrix learning vector quantization (GMLVQ) relies on a prototype-based winner-takes-all scheme rather than lazy learning [141]. Together with the prototype locations, the matrix $\mathbf{\Omega}$ is adjusted such that the distance of a given data point with correct labeling versus its distance to a prototype with incorrect labeling is minimized.

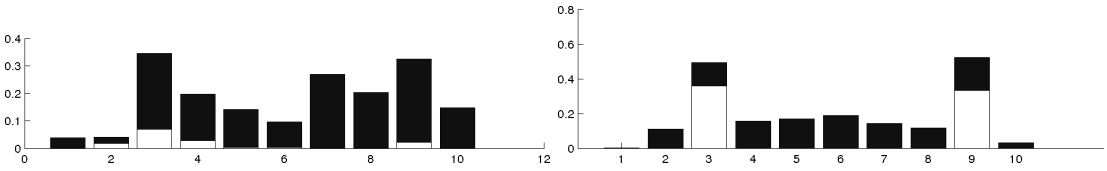


Figure 4.9.: Lower and upper bounds of feature relevance for the diabetes dataset. Results are based on Alg. 1 and Alg. 2 (left) and on the linear programming method (right).

In the following, we will employ the unique eigendecomposition of the matrix $\mathbf{\Lambda}$ as the linear data mapping $\mathbf{\Omega}$, i.e. the eigenvectors scaled with the square root of the eigenvalues.

Given the parameters $\mathbf{\Omega}$ that define a linear mapping $\mathbf{\Omega x}$ of a general quadratic form (4.20), we are interested in the interpretation of the mapping parameters $\mathbf{\Omega}$. First, we decompose the problem into one-dimensional mappings based on the following observation: Each row $\boldsymbol{\omega}$ of $\mathbf{\Omega}$ constitutes an independent mapping of the data into a one-dimensional subspace. Hence we can interpret each of these rows independently. After having obtained relevance bounds for the individual mappings $\boldsymbol{\omega}$, we can sum the absolute values of them in order to obtain relevance bounds for the whole mapping $\mathbf{\Omega}$.

4.3.4. Experiments for linear regression

In this section, results accomplished by the linear bounds method and the generic approach are compared. For both methods, data are normalized beforehand to have zero expectation and unit variance. Further, we consider a relaxed LP, allowing a bound of $1.1 \cdot \mu$ instead of μ for numerical reasons. Further we incorporate eigenvectors also with eigenvalues close to zero into the null space. For the data used in the following, a natural choice can be made by inspecting the eigenspectrum. We report the used number of eigenvectors for every data set.

Note that the methods investigated in this experiment do not reveal the strong and weak relevance, but they rely on the quantitative scaling instead. Still, upper and lower bounds allow us to distinguish three settings:

1. A feature is irrelevant: this corresponds to a small upper bound.
2. A feature is relevant for the mapping but can be substituted by others: this corresponds to a small lower bound and large upper bound.
3. A feature is relevant and cannot be substituted: this corresponds to a large lower bound.

Albeit cases 2) and 3) are not equivalent to weak and strong feature relevance in the strict sense, we will refer to these setting by these terms in the following. In the following, we will depict lower bounds as white bars and upper bounds in black.

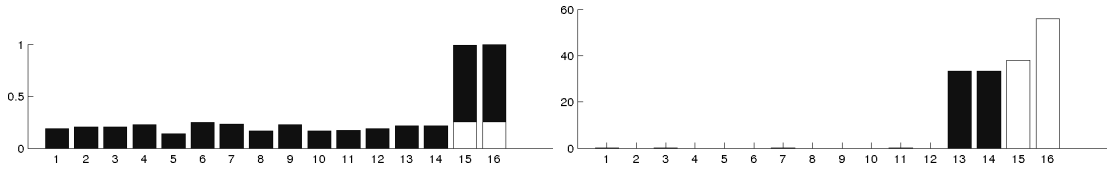


Figure 4.10.: Lower and upper bounds of feature relevance for a toy dataset. The left figure shows the results of the generic approach, the right one for the LP method.

As a first illustration, we display the feature relevances of the LP approach generated on the diabetes dataset as discussed in Section 4.3.1 in Fig. 4.9. Here, we utilize the smallest 3 eigenvalues. The features X_3 and X_9 are indicated as strongly relevant. Otherwise, features display similar upper bounds as predicted before, with small differences: the strongly relevant features X_2 and X_4, as detected by the baseline, are not highlighted by the LP technique. This is due to the fact that the resulting map can slightly be changed since noise due to small eigenvectors is accepted. Under these conditions, the features are no longer mandatory to explain the mapping. Further, X_1 vanishes for the LP method, which can be attributed to the fact that the same effect to the mapping can be achieved with another feature which has a better signal to noise ratio, i.e. L_1 norm would increase when incorporating X_1.

Difference between methods

To show a major advantage of the LP method, a toy dataset was generated: unlike iterative feature selection, the LP technique simultaneously judges the relevance of all features. Hence it can better handle settings where a large number of noisy features masks weakly relevant information. In this example, the first twelve dimensions are noisy and only slightly correlated with the target, features X_{13} and X_{14} are useful but redundant, and the last two dimensions are necessary and independent. The objective for the task is to predict the sum of the last three dimensions. We choose the dimensionality 1 for the approximated null space.

Results for both methods are displayed in Fig. 4.10. The generic method finds the two necessary and independent dimensions. It does not single out the weak relevance of the previous two features. Better results can be obtained with the linear programming approach which disregards the first dimensions completely, shows a full lower bound for the last two features, and correctly indicates the potential relevance of the other two dimensions.

Benchmarks

We utilize several benchmark data sets from [1, 45].

Boston Housing The Boston Housing dataset [76] concerns housing values in suburbs of Boston with the median value of owner-occupied homes as target. The dimensionality of the null space is picked as 3. Like displayed in Fig. 4.11, features

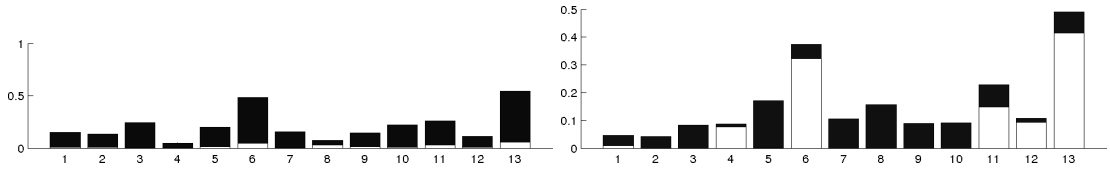


Figure 4.11.: Lower and upper bounds of feature relevance for a Boston Housing dataset. The left figure shows the results of the generic approach, the right one for the LP method.

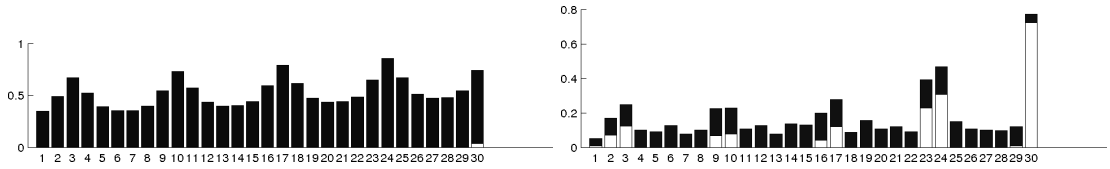


Figure 4.12.: Lower and upper bounds of feature relevance for a Poland Electricity Consumption dataset. The left figure shows the results of the generic approach, the right one for the LP method.

X_6 and X_{13} which correspond to the average number of rooms per dwelling and the percentage of lower status of the population are identified as most relevant. The same holds for X_4, X_{11} and X_{12} but to a lesser degree. Interestingly, the relevance of features like X_9 (index of accessibility to radial highways) can play an important role, but this information can also be gathered from other features.

Poland Electricity Consumption This dataset [97, 98] is a time series monitoring the electricity consumption in Poland based on time windows of size 30. We choose the zero space dimensionality as 3 corresponding to the extremely high correlation observed in this time series data. Fig. 4.12 shows that the last feature is identified by LP as the most relevant one. This is expected due to the smoothness of the time series. For the LP technique, the feature is marked as strongly relevant since its substitution would require a too large weighting. Further, for both methods, the cyclicity of the time series is clearly observable, whereby the basic method does not identify any feature as strongly relevant but the last one. Interestingly, the LP technique identifies two consecutive features as relevant for every cycle, since two values allow the estimation of the first-order derivative for better time series prognosis [48].

Santa Fe laser This dataset [70, 172] is a time series monitoring the physical process related to a laser with time windows of size 12; the dimensionality of the null space is chosen as 2. Interestingly, a result which is very similar to the previous one can be obtained. The features X_6 and X_{12} as well as their immediate predecessors are picked by the LP technique as strongly relevant. As can be seen in Fig. 4.13 both methods identify the last two features as relevant, but the LP method shows a clearer profile as concerns the past values, which coincides with findings from [48].

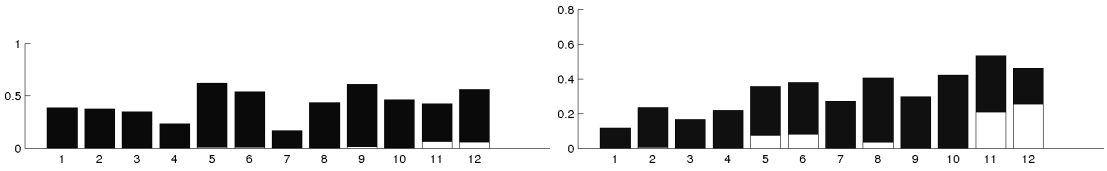


Figure 4.13.: Lower and upper bounds of feature relevance for a Santa Fe Laser dataset. The left figure shows the results of the generic approach, the right one for the LP method.

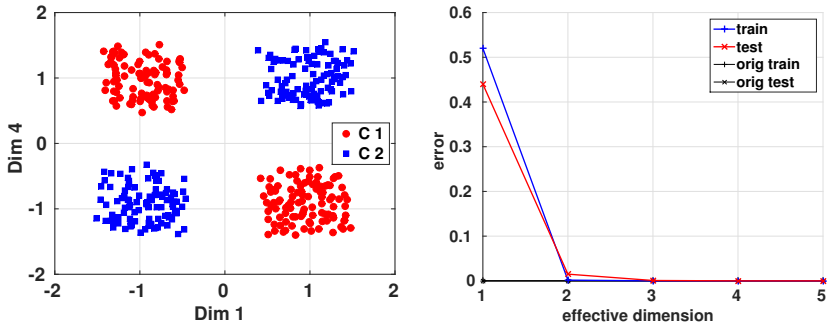


Figure 4.14.: Two relevant features of the xor data set (left). Average classification error rates of GMLVQ with regularized metrics for the xor data set (right).

4.3.5. Experiments for metric learning

In this section we apply our proposed methods to four data sets from different domains. Thereby we employ two high-dimensional spectral data sets. Due to their large dimensionality, they constitute particular interesting examples. After describing the data, we explain the experimental setup and, finally, depict the results. For the evaluation, we employ the following data sets.

- The xor data set is artificially generated and consists of 4 clusters belonging to 2 classes constituting the XOR problem. One dimension is present 3 times with the addition of noise (features 1-3) and two identical irrelevant features are included (5-6). An image with features 1 and 4 is depicted in Fig. 4.14 (left).

- The wine data set consists of 256 features which are near-infrared spectra measuring the alcohol content of 124 wine samples [154]. The set is split into 94 training and 30 test samples, where samples number 34, 35 and 84 are discarded as outliers, similar to [88]. Additionally, we switch the role of training and test set to obtain a more challenging problem in terms of interpretation. Since this is originally a regression problem, we transform it into a classification problem by binning alcohol levels into 3 classes of similar size.

- The tecator data set [2] consists of 100 features that represent absorbances deduced from a spectometer. The goal is to predict the fat content of 215 meat samples. The set is split into 172 samples for training and 43 samples for evalua-

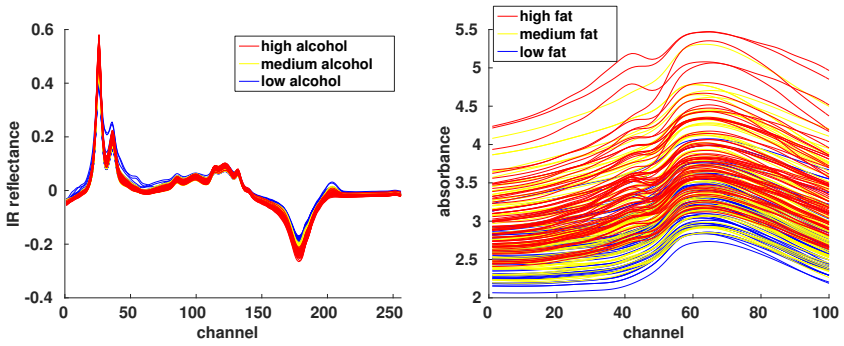


Figure 4.15.: Spectra of the data sets wine (left) and tecator (right).

tion, where we again switch the role of training and test set. Similarly as for the wine data, we bin the target variable into three classes to obtain a classification problem.

- We employ the adrenal data set [5, 18] as described previously in section 4.2.5.

Experimental setup

As a pre-processing step, we apply a z-score transformation to all our data sets, by removing the mean and standard deviation of the training data from each data set. It is important that the features in each data set have the same scaling so that we can interpret the weights of linear mappings, and in particular the result of our approach.

We train the GMLVQ model always using one prototype per class, except for the xor data set, where we use two. For the LMNN model, we use the parameters suggested by a parameter search procedure provided by the original authors.

A crucial parameter in our framework is the size of the assumed null space of the data. In order to obtain a sensible choice for this parameter, we first train a metric learning algorithm and then utilize the following scheme:

1. Create a set S of candidate values for the size of the null space. This can simply be all possible values, or a guess based on the eigenspectrum of the data.

2. For each element in S, apply our proposed interpretation framework to the previously learned metric, resulting in $2D$ relevance mappings for each row of the trained metric.

3. Compute the classification accuracy on the train and test set for each of these $2D$ mappings and average them. Select the size of the null space as the one with a small test error along with the largest null space.

We also employ the term *regularized relevance profile* when we refer to the resulting relevance bounds of our approach.

Since the null space is often large, we will recall in the following the size of the *effective dimension* which is the number of dimensions minus the size of the null space.

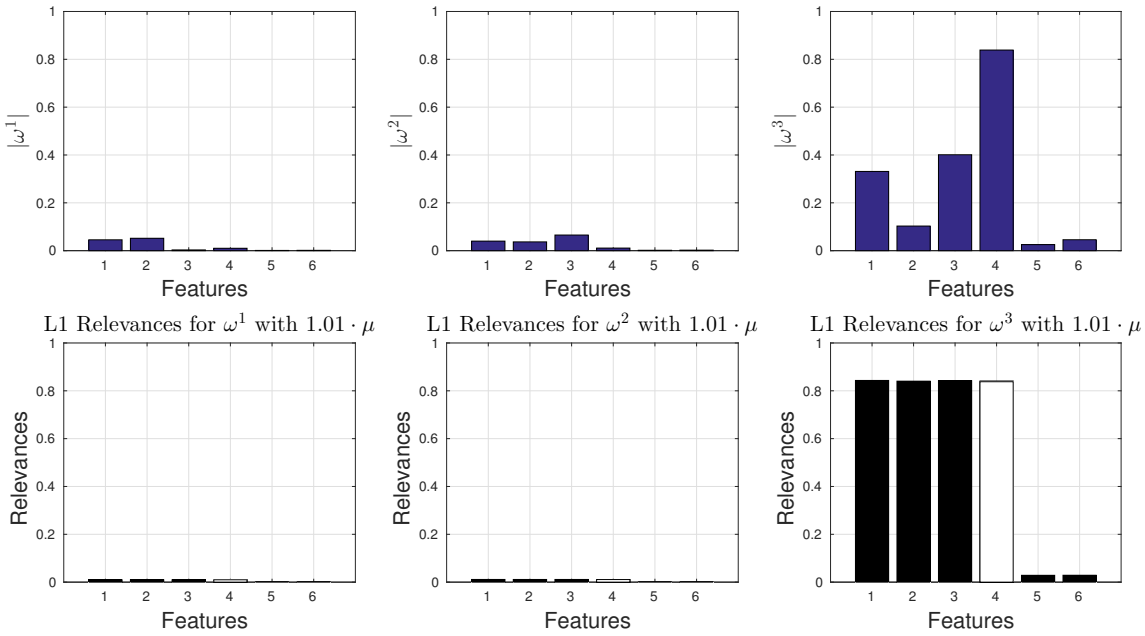


Figure 4.16.: Results of our proposed approach for the xor data set. The first row shows the original linear mappings, the second row depicts the resulting upper (in black) and lower bounds (in white).

Additionally, due to noise, we soften the minimum norm conditions in (4.18) and (4.19) by allowing solutions smaller then $1.01 \cdot \mu$, in the following.

As concerns the complexity of the metric learning scheme for high-dimensional data, the computation of a full rank matrix $\mathbf{\Lambda} \in \mathbb{R}^{D \times D}$ can be costly. However, $\mathbf{\Lambda}$ can be forced to have a low rank [27]. This can be done by defining $\mathbf{\Lambda} = \mathbf{\Omega}^\mathsf{T}\mathbf{\Omega}$ with $\mathbf{\Omega} \in \mathbb{R}^{l \times D}$, where $l \leq D$ restricts the rank.

Synthetic data

In order to demonstrate the problem of directly interpreting linear weights of a trained metric as relevances, we employ the synthetic data set xor.

We train a GMLVQ method that results in a zero prediction error on the training and test set. The resulting three mappings of the metric with the largest scaling are depicted in the first row of Fig. 4.16. Basically, only one of these mappings has a high scaling so that the classification model uses approximately a one-dimensional subspace to solve the classification task.

A direct interpretation of this linear vector $|\boldsymbol{\omega}^3|$ would suggest that feature 4 is the most important one, features 1 and 3 have only half the relevance and features 2, 5 and 6 are not useful for the task. However, for this data set we know that features 1-3 have the same explanatory power and if considered alone, each of them is as important as feature 4. It follows that, for this example, a direct interpretation of the linear weights is misleading, particularly for the weakly relevant features.

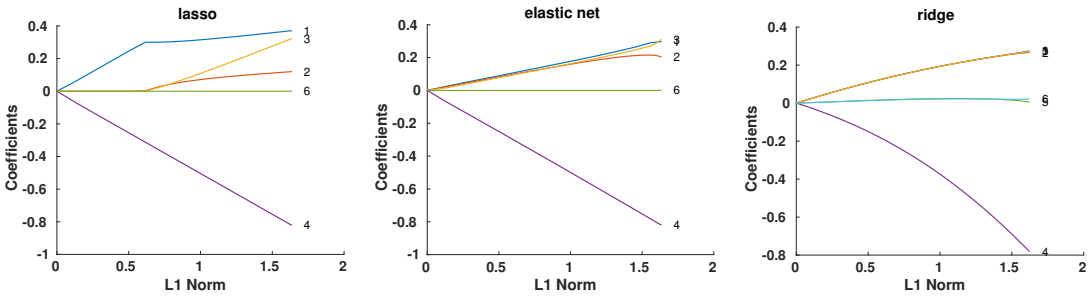


Figure 4.17.: Employing the xor data set, estimates of the coefficients for different values of the L1 norm (x-axis) are shown. The methods lasso (left), elastic net (middle) and ridge regression (right) are utilized.

In order to obtain a valid interpretation for the relevance of the features, we apply our proposed framework. We estimate the classification accuracy of the regularized mappings for all possible sizes of the null space as described in subsection 4.3.5. The resulting curves are depicted in Fig. 4.14 (right). It is apparent from the Figure, that the smallest effective dimension size with a zero test error is 3, although the test error for 2 dimensions is only slightly larger. Nevertheless, we employ 3 for our proposed framework. The resulting relevance bounds are shown in the second row of Fig. 4.16, where black bars depict weakly and white bars strongly relevant features.

The results show that the bounds for the first two one-dimensional mappings $|\omega^1|$ and $|\omega^2|$ have vanished. Formally speaking, this implies that the same mappings can be obtain with an almost zero L1 norm, meaning that these two mappings map the training data to zero. More interestingly are the resulting bounds for $|\omega^3|$: The framework has identified feature 4 as a strongly relevant feature and has found that features 1-3 can be replaced but that each of them can explain as much of the target variable as feature 4. This explanation is precisely how we generated the data. Features 5 and 6 have almost 0 upper bounds, merely reflecting noise.

In order to have a comparison to relevance interpretation in literature, we apply the methods Lasso, Elastic Net and Ridge Regression to our resulting mapping by defining $\hat{y}_i = \omega^\top \mathbf{x}_i$. Then, we can apply the formulation in equation (4.12) for Lasso and according ones for Elastic Net and Ridge Regression to obtain an interpretation for the feature weights based on these methods. The results are depicted in Fig. 4.17 for the Lasso (left), Elastic Net (middle) and Ridge Regression (right). We interpret only $|\omega^3|$ this way, since, as we saw previously, this mapping contains the relevant

Table 4.3.: Classification error rates ranging between 0 and 1 for all data sets. If not specified differently, the classification model is GMLVQ.

	xor	wine	tecator	GMLVQ on adrenal	LMNN on adrenal
train error	0.00	0.00	0.07	0.04	0.00
test error	0.00	0.29	0.16	0.03	0.05

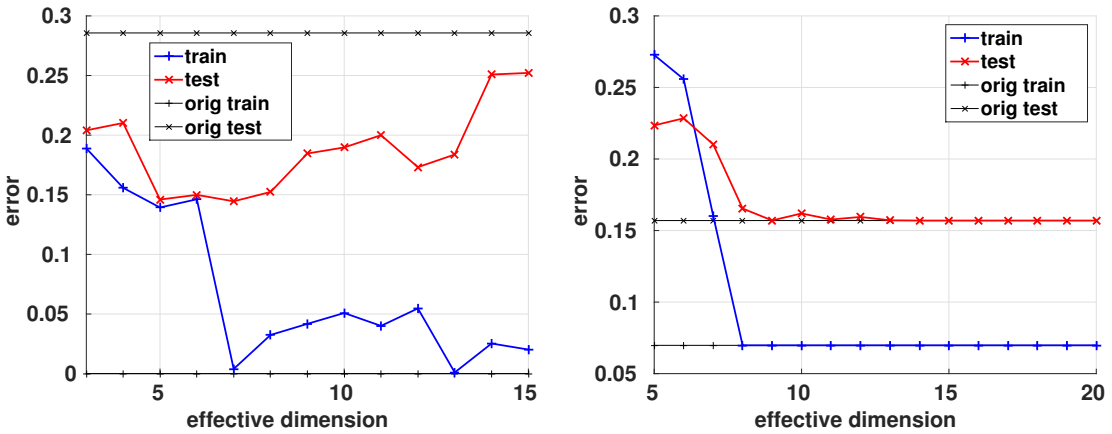


Figure 4.18.: Average classification error rates of GMLVQ with regularized metrics for the wine (left) and tecator (right) data set, both for set S.

information for discriminating the classes.

The progress of the coefficient weights for all three frameworks implies that feature 4 is particularly relevant. However, the results differ for the three weakly relevant features 1-3: Elastic net and Ridge regression require all three features equally weighted and hence do not show that each of them can actually be neglected. The Lasso identifies feature 1 as particularly important, followed by feature 3 and 2. This order seems arbitrary and is an artifact of the noise which was added to all three features. Hence, we argue that these frameworks cannot provide the same information as our formalization.

Near-infrared spectral data

Spectral data have many correlated features and hence a large null space. For such data, it can be particularly misleading to directly interpret the weights of linear mappings. Hence, our method should be well suited in this case.

First of all we train a GMLVQ model for each of the two data sets wine and tecator where we restrict the rank of the matrix $\mathbf{\Omega}$ to two since GMLVQ tends to utilize only a low rank matrix in the end, usually. This does not harm the training accuracy as can be observed in Table 4.3 but tends to improve the generalization. Subsequently, we apply our approach to compute relevance bounds for the according learned metric. As previously, we determine a suitable size of the effective dimension using the scheme described in subsection 4.3.5. The corresponding images are shown in Fig. 4.18: left for the wine and right for the tecator data set.

The smallest test error is obtained with an effective dimensionality of 7 for the wine data set and with an effective dimensionality of 9 for the tecator data set. It is particularly interesting, that for the wine data set the regularized metric achieves a better performance then the original metric: while the training error stays the same, the test error drops from 0.29 to 0.14, which is a factor 2. The resulting relevance bounds for

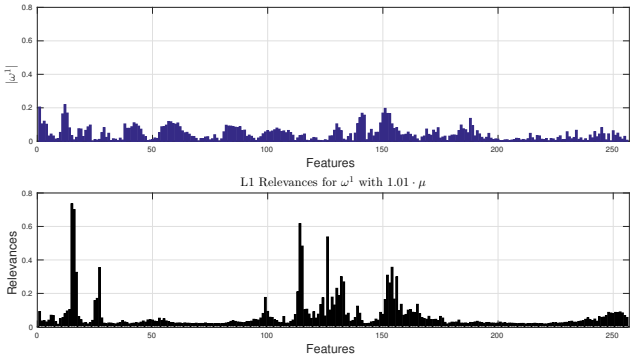


Figure 4.19.: Results of our proposed approach for the wine data set. The first row shows the original linear mapping, while the second row depicts the resulting upper relevance bounds. The lower bounds are all zero, in this case.

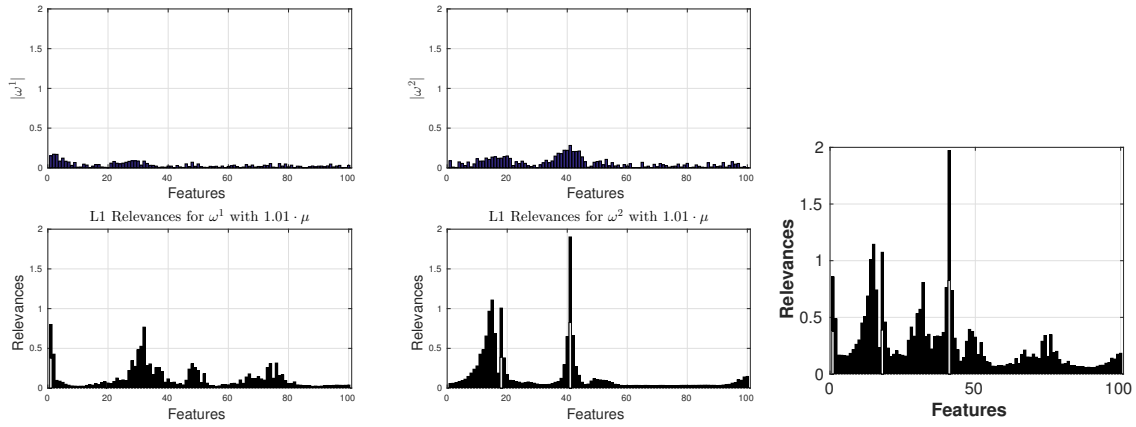


Figure 4.20.: Results of our proposed approach for the tecator data set. First two columns: The first row shows the original linear mapping, the second row depicts the resulting upper and lower relevance bounds. The last column shows the summed lower and upper bounds.

the wine data set are depicted in Fig. 4.19 and for the tecator data set in Fig. 4.20. For the wine data, the training procedure of the classifier yielded a rank one matrix, hence we use only a one-dimensional mapping for interpretation, in this case. For the tecator data set, both mappings are utilized, and the resulting relevance bounds for both mappings are displayed in Fig. 4.20 (last column).

Interestingly, the relevance bounds for both data sets contain only very few irreplaceable features, while the upper bounds are peaked, which implies that a few features can already explain the mapping to a large extent. Particularly for the wine data set, much noise is removed from the original mapping, i.e. many features have a low upper bound. Fig. 4.21 displays the original mapping (top) and the averaged mapping over all $\overline{\omega}, \underline{\omega}$ (bottom). Here it is apparent that, while the original mapping has many non-zero values, the averaged regularized profile is extremely sparse. Furthermore, the classification error with the averaged map accounts to 0 on the training and to 0.14

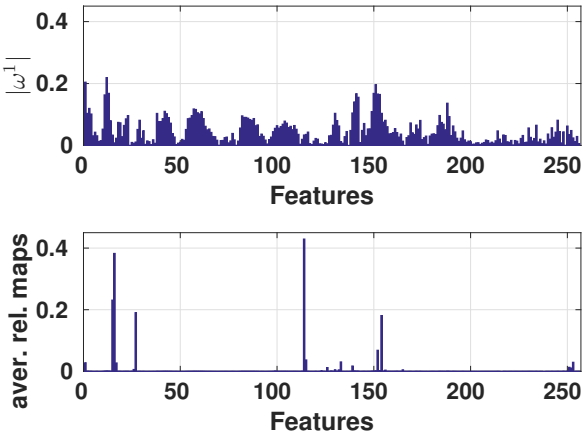


Figure 4.21.: Absolute values of the original mapping (top row) together with the absolute value of the averaged regularized mappings (bottom row).

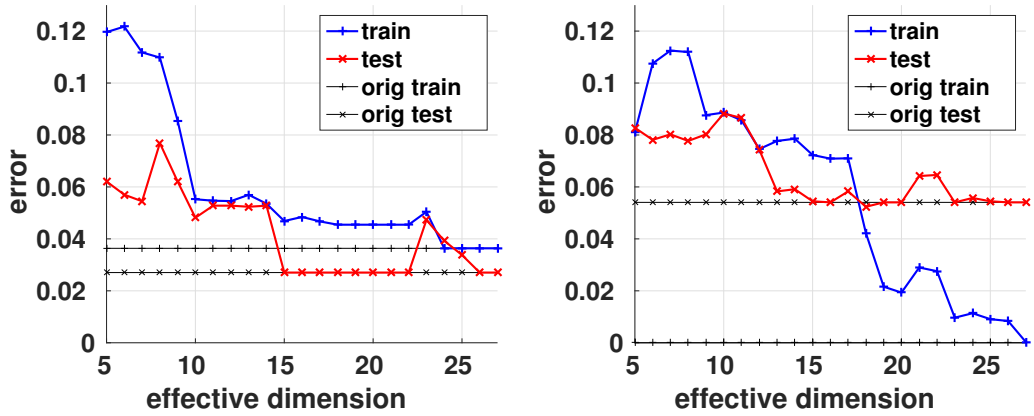


Figure 4.22.: Average classification error rates of GMLVQ (left) and LMNN (right) with regularized metrics for the adrenal data set.

on the test set, which is comparable to the averaged error of the regularized mappings $\overline{\omega}, \underline{\omega}$.

Biomedical data

We utilize the adrenal data set to compare two metric learning approaches: The GMLVQ and LMNN. Both use the same functional form for computing distances, hence, we can apply our approach to both trained relevance matrices. First, we train both models with the restriction to rank two relevance matrices. This restriction did not harm the classification performance in our experiments, as compared to training without this restriction. The classification errors are depicted in Table 4.3. Both approaches achieve a comparable performance, where the LMNN model is better on the training data while GMLVQ is superior on the test data.

For these models, we compute the classification errors based on different sizes of

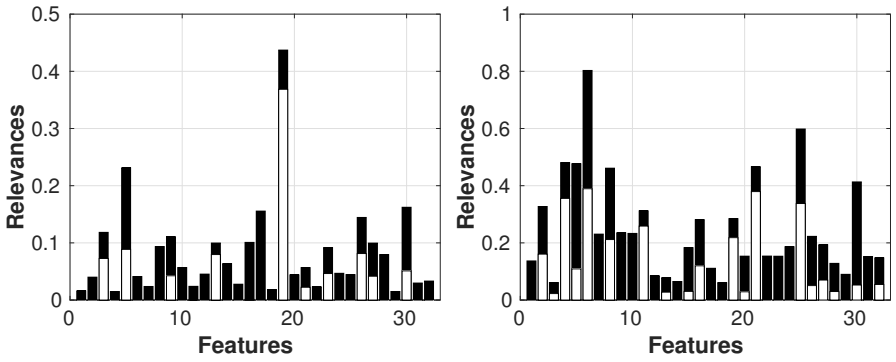


Figure 4.23.: Relevance bounds for a GMLVQ model (top) and a LMNN model (bottom), both trained on the adrenal data set.

the effective dimension. These results are depicted in Fig. 4.22. Good performances are achieved with an effective dimensionality of 15 for the GMLVQ model and of 20 for the LMNN algorithm. The according relevance bounds for both relevance matrices are shown in Fig. 4.23.

Both trained distance metrics agree on a few features, such as 19, while they emphasize often different ones. This might explain the different error curves in Fig. 4.22 and the different classification performance on the training and test set. One explanation for these different metrics is that the underlying classification models work differently. While GMLVQ is usually applied with few prototypes, the LMNN model searches among all stored training points. Hence, the latter acts much more locally in the classification. Also, the optimized GMLVQ parameters are not deterministic because the utilized cost function is not convex.

Resumee

We have addressed the question in how far weights which arise from a linear transformation such as a linear classification, regression, or metric learning, allow a direct interpretation of the weighting terms as relevances. We have discussed that this is usually not the case in particular for high-dimensional data, a setting with particular importance e.g. for spectral data analysis or the biomedical domain. Inspired by previous work which addresses the null space of the observed data, and the notion of weak and strong feature relevance, we have developed a framework which yields to an efficient quantitative evaluation of the minimum and maximum feature relevance for a given linear mapping. This framework is based on the hypothesis that the objective is the output of the given mapping for the given data, and only weights which are minimum in L_1 norm are of interest. Then, linear programming enables a polynomial deterministic technique to estimate these relevance intervals.

We have compared the techniques to a corresponding baseline which is directly based on forward-backward feature selection. It becomes apparent that the techniques closely resembles the notion of weak and strong feature relevance; unlike iterative

methods, it does not face problems when dealing with high-dimensional data and many irrelevant features, still being capable of distinguishing this information from mere noise. However, this concept is currently restricted to linear mappings.

4.4. Discussion

In the first section, we have introduced relevance learning into dimensionality reduction as an efficient concept to accompany a given visual display by the possibility to judge the relevance of data dimensions for the given mapping. Besides a better interpretability of the mapping, we have shown how this framework can be used as an interface to change data representations by means of visual displays, e.g. by incorporating label information into the pipeline. This opens the way for future work in particular in two aspects: on the one hand, extensions to local matrix variants are possible, which allow a richer representation of globally non-linear dependencies, and its corresponding visual display. On the other hand, the proposed framework can be integrated into an interactive pipeline, where online adaptation of the display according to a new metric is a central demand.

Following up, we have discussed the interpretability of weights from linear mappings. Since a direct interpretation can be misleading, in particular for correlated features, we have proposed a novel method that allows valid interpretation by providing relevance bounds for each feature. This way, properties related to the concepts of strongly and weakly relevance can be investigated. So far, we have demonstrated the techniques for various benchmarks with very promising results. An interesting opportunity for future work is to test the suitability of this technique for interactive data exploration e.g. in biomedical applications where relevance intervals will be checked by medical experts.

Having investigated how to enhance dimensionality reduction techniques by interpretable relevance profiles of their features, and having discussed the uniqueness of such relevance profiles for general linear functions in particular for high-dimensional data, we now turn to a particularly interesting use of DR techniques for the task of transfer learning.

Chapter 5.

Dimensionality reduction for transfer learning

Chapter overview *This chapter presents an approach for transfer learning in the particularly difficult scenario of unlabeled data and no available correspondence information between instances in the source and target space. This approach is based on the central idea of using a low-dimensional representation provided by dimensionality reduction techniques, since a core property of the latter is to preserve structure while removing noise.*

Parts of this chapter are based on:

[C15b] P. Bloebaum, A. Schulz, and B. Hammer. Unsupervised dimensionality reduction for transfer learning. In *ESANN 2015*, pages 507–512, 2015.

[C14b] P. Bloebaum, and A. Schulz. Transfer learning without given correspondences. In *NC^2 2014*, pages 42–51, 2014.

5.1. Motivation

A crucial property of every successful machine learning model is its generalization ability from the known training data to novel settings, with statistical learning theory offering powerful mathematical tools for establishing formal guarantees for valid generalization [167]. One core assumption underlying the classical setting is that of data being independent and identically distributed (i.i.d.): the training scenario and future application areas are qualitatively the same, differences result from different sampling from the same distribution only. Transfer learning addresses the setting that source and target data are qualitatively different because they follow a different underlying distribution or they are even contained in different spaces [121].

Models which reliably follow a trend have become increasingly important in the context of big data, distributed systems, and life-long learning, as demonstrated e.g. by quite some recent successful approaches [127]. In this contribution, we will focus on the second problem of data being contained in different spaces. This setting occurs e.g. when the same objects are measured using sensors with different characteristics, a sensor is exchanged in a system (e.g. by a more sensitive one), the same objects are described in different languages, etc. One promise of such transfer consists in a plug-and-play technology for novel sensors or representations, without the need of costly

retraining of the underlying models.

A few approaches have been proposed in this context, such as a common feature representation [130, 42], a coupled embedding of data in a low-dimensional space [20, 120, 143], or a combination of representation learning and classification [102]. In this contribution, we are interested in the potential of modern unsupervised dimensionality reduction to induce a common representation of data for transfer learning. For this purpose, we will address two problems: How to linearly embed source and target in a common domain such that the resulting characteristics are shared as much as possible? We will rely on a probabilistic modeling of the target domain which induces an explicit embedding mapping via an expectation maximization (EM) [36] approach. How to extend this framework to nonlinear mappings by incorporating modern nonlinear DR techniques which are better capable of capturing nonlinear characteristics of the data? We will rely on t-SNE [159] as a method which is particularly suited to reliably capture cluster structures, and its recent extensions to kernel mappings to allow for an integration into the transfer pipeline, as discussed section 2.2.3.

Unlike our approach, most manifold learners rely on explicit correspondences or equivalent information [168]. One rare exception is the approach [169], where local characteristics are directly extracted from the manifold to provide a local fingerprint. However, it is computationally exponential in the neighborhood size and, further, it does not resolve ambiguities due to local self similarity.

5.1.1. Scientific contributions and structure of the chapter

This chapter presents the following core contributions.

Linear transfer learning In section 5.2.1, a novel transfer learning technique is presented which is capable of transferring knowledge from a source space into a target space without requiring labeled data or correspondence information. Thereby, it relies on a common linear embedding of both spaces.

Nonlinear transfer learning In section 5.2.2, the previous approach is extended allowing nonlinear transfer learning. A strong regularization is employed for this purpose.

5.2. Transfer learning without given correspondences

We assume N source data $\mathbf{x}_i \in \mathcal{X}$ and K target data $\mathbf{z}_j \in \mathcal{Z}$ with different spaces $\mathcal{X}$ and $\mathcal{Z}$ but shared underlying information are present. We will model the fact that these two data sets share their structure by embedding both simultaneously in a low-dimensional vector space Ξ where we assume a common distribution of the data sets. This will provide an explicit embedding $\mathbf{x}_i \mapsto \boldsymbol{\xi}_i^x \in \Xi$ of the source data and $\mathbf{z}_j \mapsto \boldsymbol{\xi}_j^z \in \Xi$ of the target data. The question how suitable embeddings can be found will be the subject of sections 5.2.1 and 5.2.2.

Provided such an embedding is present, knowledge transfer is immediate: Assume for instance, source labels $l(\mathbf{x}_i)$ are present. This enables us to learn a classifier on the embedding space based on the training data $(\boldsymbol{\xi}_i^x, l(\mathbf{x}_i))$. By means of the mapping $\mathbf{z}_j \mapsto \boldsymbol{\xi}_j^z$, this classifier can be directly extended to the target data.

5.2.1. Shared linear embedding

For simplicity, we first assume that data can be embedded linearly into a low-dimensional space Ξ. For the mapping $\mathbf{x}_i \mapsto \boldsymbol{\xi}_i^x$ we can simply rely on a PCA embedding which captures the most relevant linear structure of the source data. Note that it is easily possible to exchange this embedding by any other suitable mapping such as LDA in case of auxiliary labels or a nonlinear map as we will do in section 5.2.2. The target embedding should aim for a match with the source distribution in the latent space Ξ. We consider a parametrized linear mapping

$$f_{\mathrm{pm}} : \mathcal{Z} \to \Xi, \mathbf{z} \mapsto \boldsymbol{\xi}^z = \mathbf{W}\mathbf{z} \tag{5.1}$$

which induces a mixture of Gaussians

$$p(\boldsymbol{\xi}_i^x | \mathcal{Z}, \mathbf{W}) \sim \sum_j \theta_j \exp\left(-\|\boldsymbol{\xi}_i^x - \mathbf{W}\mathbf{z}_j\|^2/(2\sigma^2)\right) \tag{5.2}$$

in the latent space, where we introduce hidden variables h_{ij} and the $\mathbf{h}_i$ have one non-zero entry 1. To enforce a shared distribution of source and target distribution in the latent space, we optimize the expected value of the log likelihood

$$\sum_i \sum_j \mathbb{E}(h_{ij}) \left(\log \theta_j + \log \mathcal{N}(\boldsymbol{\xi}_i^x | \mathbf{W}\mathbf{z}_j, \sigma)\right). \tag{5.3}$$

Its optimization can rely on an EM approach [36] with the expectation of the hidden variables (E-step)

$$\gamma_{ij} := \mathbb{E}(h_{ij}) = \exp\left(-\|\boldsymbol{\xi}_i^x - \mathbf{W}\mathbf{z}_j\|^2/(2\sigma^2)\right) / \sum_l \exp\left(-\|\boldsymbol{\xi}_i^x - \mathbf{W}\mathbf{z}_l\|^2/(2\sigma^2)\right) \tag{5.4}$$

and a direct minimization of the following term with respect to $\mathbf{W}$ (M-step):

$$\sum_{i,j} \gamma_{ij} \|\boldsymbol{\xi}_i^x - \mathbf{W}\mathbf{z}_j\|^2. \tag{5.5}$$

It is often useful to apply a standard regularization in this step. In order to initialize the mapping $\mathbf{W}$, a PCA projection can be utilized. For the bandwidth σ, a deterministic annealing scheme can be employed [155].

5.2.2. Shared nonlinear embedding

It has been emphasized in [159, 94] that linear DR does not allow a reliable characterization of central data characteristics for many modern data sets; in such cases, a

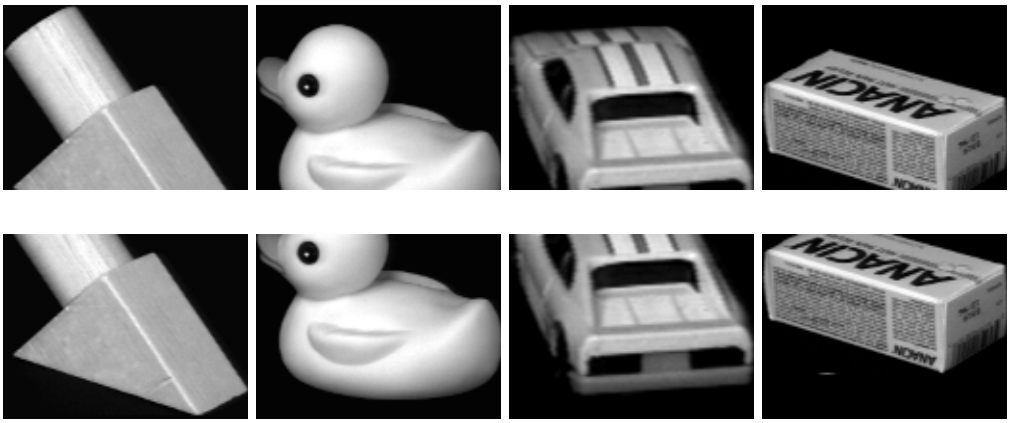


Figure 5.1.: Examples of images from the Coil data set: the top row contains images from the source data while the bottom row shows the according target images.

shared linear representation is clearly not sufficient to provide an informative shared representation of source and target domain. We are interested in how far modern nonlinear dimensionality reduction techniques allow us to solve this problem. More specifically, we will rely on t-SNE as a particularly powerful embedding technique in case of clustered data [159]. Obviously, it is easily possible to exchange a PCA embedding $\mathbf{x}_i \rightarrow \boldsymbol{\xi}_i^x$ by any given nonlinear embedding such as t-SNE for the source data. For the target domain, we aim for an explicit mapping and, therefore, rely on the kernel extension of t-SNE as introduced in section 2.2.3. The linear mapping (5.1) becomes

$$f_{\text{pm}} : \mathcal{Z} \rightarrow \Xi, \mathbf{z} \mapsto \boldsymbol{\xi}^z = \sum_j \mathbf{w}_j \cdot k(\mathbf{z}, \mathbf{z}_j) / \sum_l k(\mathbf{z}, \mathbf{z}_l) \tag{5.6}$$

with Gaussian kernel $k(\mathbf{z}, \mathbf{z}_j) = \exp(-0.5\|\mathbf{z} - \mathbf{z}_j\|^2/\sigma_j^2)$ where σ_j is adjusted according to the effective neighbors of $\mathbf{z}_j$, and $\mathbf{w}_j \in \Xi$ comprises the parameters of $\mathbf{W}$. Since, in our setting, we aim for a match of the target and source distribution, we optimize the data likelihood by substituting the M-step in equation (5.5) by the optimization of

$$\sum_{i,j} \gamma_{ij} \|\boldsymbol{\xi}_i^x - f_{\text{pm}}(\mathbf{z}_j)\|^2 + \lambda Q^{\text{t-SNE}}\left((\mathbf{z}_1, \ldots, \mathbf{z}_K), (f_{\text{pm}}(\mathbf{z}_1), \ldots, f_{\text{pm}}(\mathbf{z}_K))\right), \tag{5.7}$$

with respect to parameters $\mathbf{W}$. To better deal with the non-linearity, we add the regularization term $Q^{\text{t-SNE}}$ which enforces structure preservation of the target data during optimization. We utilize the t-SNE cost function to measure the latter.

For an initialization, the parameters $\mathbf{W}$ are adjusted such that $f_{\text{pm}}(\mathbf{z}_i)$ approximates the t-SNE projection of the (target) data $\mathbf{z}_i$, i.e. we calculate the kernel t-SNE mapping. The sum in equation (5.6) can either be over all points or over a subset, only. We use the latter, mainly for regularization purposes (see 2.2.3 for more details).

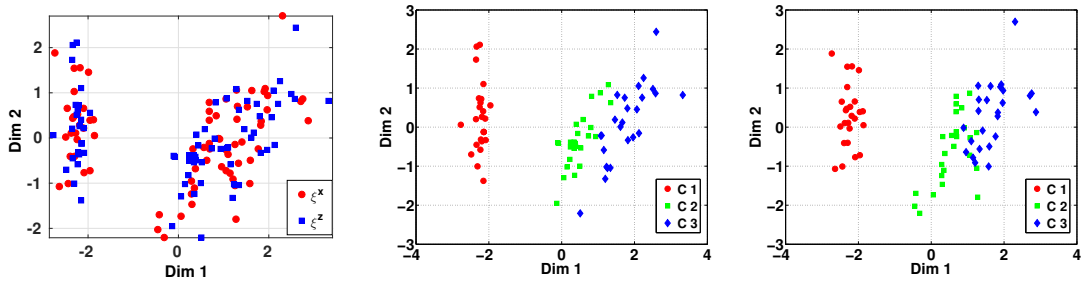


Figure 5.2.: The linear alignment of source and target data for the Iris data set is shown left. Both data sets are shown individually with their according labeling middle (source) and right (target after transfer).

5.3. Experiments

In the following, we evaluate the linear and nonlinear transfer learning techniques exemplarily with two data sets. For transfer learning, we will always assume that only the source data are accompanied by labels while this is not the case for the target data. This means, that we cannot use class information for the transfer learning. However, we will use this information in order to evaluate the quality of the transfer learning: In the embedding space we utilize the source data $(\boldsymbol{\xi}_i^x, l(\mathbf{x}_i))$ to train a linear Support Vector Machine (employing the one versus one scheme for data with more than two classes). Subsequently, we classify the projected target data $\boldsymbol{\xi}_i^z$ and compute the accuracy by comparing to the labels $l(\mathbf{z}_i)$. Note that the latter labels are not used for the transfer learning but for evaluation purposes, only. The classification accuracy for the embedded target data allows to judge in how far the transfer of information was successful.

We employ the following two benchmark data sets in our experiments.

- The Iris data set utilizes four features to describe three classes of iris plants. 150 instances are available in this data set.
- The Coil data set consists of images of objects that are rotated around their own axes. We employ four items from this data set in our experiments.

We create a transfer learning scenario for the Iris data set utilizing the following scheme: We split the data set randomly into two parts, using one for the source and the other for the target data. The latter are additionally mapped with a random matrix to ten dimensions. Note that for this data set, the source and target data don't have any common instances.

For the Coil data set, we cut each image in order to obtain source and target data: We utilize the top 3/4 of each image for the source data and the lower 3/4 for the target data. Such, 1/2 of the information overlaps for both sets. One example object of each class is shown in Fig. 5.1.

Evaluation of TL with linear embeddings: We apply our approach to the Iris data set. We use a two-dimensional embedding space for the source data created by the

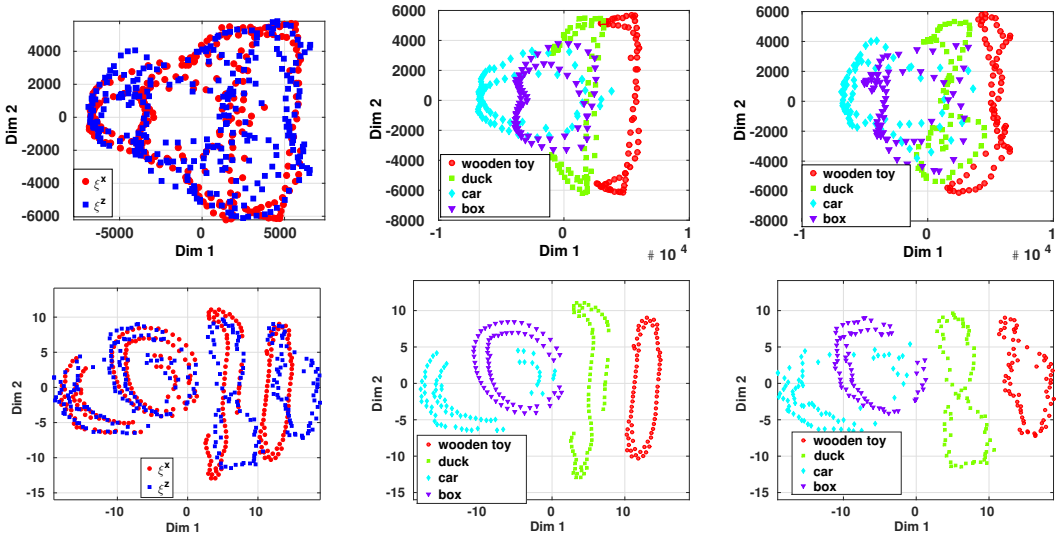


Figure 5.3.: A linear (top three) and nonlinear (bottom three) alignment of source and target data for the Coil data set is shown left. Both data sets are shown individually with their according coloring middle (source) and right (target).

PCA mapping. Fig. 5.2 shows an alignment result of the method. The left image depicts the source and target data in the embedding space while the middle and right image show the source and target data individually. This procedure was iterated ten times yielding the mean classification accuracy of 91% for the source data and of 83% for the target data.

We also apply our approach to the Coil data set. As previously, we iterate the procedure ten times yielding the accuracies 70% and 66% for the source and target domain (see Table 5.1 for an overview). An exemplary alignment is shown in Fig. 5.3 (top). The reason for the drop of the accuracy is visible here: Due to the linear mapping, the classes overlap and, hence, an accurate classification is not possible. This holds in particular also for the classification of the source data.

Evaluation of TL with nonlinear embeddings: In order to obtain a nonlinear embedding, we utilize the non-parametric method t-SNE. Applying the scheme from 5.2.2 yields the mean accuracy after ten runs of 92% for the source and 83% for target data. An exemplary alignment is shown in Fig. 5.3 (bottom). In the middle figure, the advantage of nonlinear mappings is visible: The classes are well separated which allows a successful consequent transfer learning.

Table 5.1.: Mean classification accuracies with a linear SVM for the experiments.

Embedding	linear		nonlinear
Data sets	Iris	Coil	Coil
Error Source	91%	70%	92%
Error Target	83%	66%	83%

5.4. Discussion

We have introduced an approach to perform Transfer Learning via mapping source and target data into a common embedding space. For this purpose, we have proposed the two possibilities to use linear and nonlinear embeddings. The linear embeddings have proven to be very stable but they do not allow an accurate transfer if linear projections cannot embed the class structure adequately. Nonlinear methods can improve the transfer of information in this case.

In this chapter we have utilized only two dimensionality reduction techniques, i.e. PCA and t-SNE. Other approaches such as DiDi methods or manifold embeddings could be particularly useful here and the investigation of their applicability is subject to future work.

Chapter 6.

Conclusion

Summary In this thesis, we discussed the general problem of making large amounts of digital data accessible for humans. To address this challenge, we focused on dimensionality reduction as a particular promising technique for providing an intuitive interface to such data. Since dimensionality reduction is an ill-posed problem in the case of intrinsically high-dimensional data, however, we relied on discriminative dimensionality reduction, i.e. on incorporating auxiliary data into dimensionality reduction to provide a clear specification for relevant structure. Inside this frame, we made use of the Fisher metric as a particularly flexible and mathematically well grounded concept to enable discriminative dimensionality reduction.

To circumvent the disadvantages of such nonparametric mappings, we proposed a parametric extension termed kernel t-SNE. This technique enables applying the mapping for new data points which have not been available for training. An example for such a scenario are streaming data. This method also enables to project large parts of a data set in $\mathcal{O}(N)$ time, which can be crucial for large data sets. We have reformulated the computation of distances inside the Fisher metric framework, thus, making it applicable to data given by similarities, only. This opens the way towards DiDi projections for complex data where vectorial representations are difficult to obtain. These include for instance musical pieces, graphs or structured data in general, as demonstrated in the experiments. Further, we have proposed a definition of the Fisher metric enabling to integrate also real-valued auxiliary information. This allows to consider regression targets as auxiliary data. We have demonstrated it in artificial and real world benchmarks.

In addition to employing DR for data visualization, a valuable tool for accessing data can be to interpret machine learning models. In this context, we have presented a framework that allows to visualize high-dimensional functions, such as those of classification or regression models, in two dimensions. We have performed numerical evaluations to judge the quality of our embeddings and demonstrated the usefulness for specified user tasks with various models and data sets. Methods for DiDi have shown to be particularly useful in this application.

Further, we have proposed a method which gives insights into the relevance of features for a nonlinear data projection. In addition to the benefits of nonlinear DR techniques, such as providing intuitive access to the neighborhood structure of a given

data set, this technique augments nonlinear DR methods by providing information about the original features. Furthermore, this method enables the user to imprint structural information on a potentially high-dimensional data set by specifying such structure in the projection space.

Moreover, we presented a novel technique for valid interpretation of linear mappings. In the presence of many or correlated features, this method provides intervals for the relevance of features. Thereby, it allows to identify whether features are strictly relevant or can be replaced by others. We demonstrated this approach for linear mappings implemented by linear regression and metric learning.

The structure preserving property of DR methods enabled us to tackle the particular difficult transfer learning problem where no label and correspondence information is available. Our proposed method uses only the structure of the manifolds to align the data from the source and the target space in a latent embedding space. We demonstrated our approach for two data sets and presented the possibility to use nonlinear transfer functions.

Outlook Although this thesis has addressed challenging questions and presented solutions, there exist still many open questions, some of which are already under investigation. We summarize a few of them in the following.

Discriminative dimensionality reduction based on the Fisher metric has proven to be a powerful approach which is applicable in many scenarios such as with discrete and real-valued auxiliary information or to data described by similarities, only. However, the run time complexity is still a problem for large data sets: Currently the implementations range from $\mathcal{O}(N^3)$ to $\mathcal{O}(N^2)$, which makes it impossible to apply these algorithms to larger data sets. A possible cure might be applying the concepts used for efficient neighborhood embeddings [177, 158] to distance computation with the Fisher metric.

In this thesis, we have presented a formulation of the Fisher metric, which allows to compute Fisher distances from data given only as similarities. However, this concept is restricted to discrete-valued auxiliary information. An extension would allow to compute DiDi projections also for similarity data together with real-valued auxiliary information. Such a method would be relevant in complex data domains where a vector based representation is not easily found.

Further, we have proposed a framework to visualize high-dimensional functions such as classification and regression functions. This approach is evaluated by investigating the inverse DR function on the positions of the data points, providing an estimate of the visualization quality on the positions of the data. This evaluation could be augmented by geometrical properties such as the preservation of curvature or function specific characteristics such as the margin for classifiers. Even more interesting would be the incorporation of such aspects directly into the optimization.

A further open question can be found in the context of interpretability of nonlinear DR. The proposed method utilizes a globally linear mapping, which is beneficial

for interpretation. However, it is possible that the relevance of features depends on the positions in the data space and that it could change as the position changes. A locally linear mapping could measure such effects, hence, constituting an interesting extension.

The concept for the valid interpretation of linear mappings depends heavily on the chosen notion of equivalence among linear mappings. While the notion introduced in this thesis presents a rather general formulation, more model specific formalizations are possible. Such could be for instance given by the constrains of a linear SVM model.

The achievements of this thesis set the stage for further research in the exiting topic of representation learning and its theoretical foundations.

Appendix A.

Mathematical derivations

A.1. The Fisher information matrix for a discrete auxiliary variable

We demonstrate in the following how to obtain the analytical form of the Fisher information matrix shown in eqs. (2.20) to (2.23), in case of a discrete auxiliary variable and the Parzen window estimator. This result was already reported in [123]. Nevertheless, we provide here the derivation for the convenience of the reader.

The Fisher information matrix with a discrete auxiliary variable is defined by

$$\mathbf{J}(\mathbf{x}) = \mathbb{E}_{p(c|\mathbf{x})}\left\{\left(\frac{\partial}{\partial \mathbf{x}}\log p(c|\mathbf{x})\right)\left(\frac{\partial}{\partial \mathbf{x}}\log p(c|\mathbf{x})\right)^{\mathsf{T}}\right\}, \tag{A.1}$$

where we estimate $p(c|\mathbf{x})$ by

$$\hat{p}(c|\mathbf{x}) = \frac{\sum_i \delta_{c,c_i}\exp(-0.5\|\mathbf{x}-\mathbf{x}_i\|^2/(\sigma^p)^2)}{\sum_j \exp(-0.5\|\mathbf{x}-\mathbf{x}_j\|^2/(\sigma^p)^2)}. \tag{A.2}$$

Then, the derivative can be decomposed into

$$\begin{aligned}\frac{\partial}{\partial \mathbf{x}}\log p(c|\mathbf{x}) =& \frac{\partial}{\partial \mathbf{x}}\log\left(\sum_i \delta_{c,c_i}\exp(-0.5\|\mathbf{x}-\mathbf{x}_i\|^2/(\sigma^p)^2)\right)\\ &-\frac{\partial}{\partial \mathbf{x}}\log\left(\sum_j \exp(-0.5\|\mathbf{x}-\mathbf{x}_j\|^2/(\sigma^p)^2)\right),\end{aligned} \tag{A.3}$$

with

$$\begin{aligned}\frac{\partial}{\partial \mathbf{x}}\log\left(\sum_i \delta_{c,c_i}\exp(-0.5\|\mathbf{x}-\mathbf{x}_i\|^2/(\sigma^p)^2)\right) &= \frac{-1}{(\sigma^p)^2}\sum_i \zeta(i|\mathbf{x},c)(\mathbf{x}-\mathbf{x}_i)\\ &= \frac{-\mathbf{x}}{(\sigma^p)^2} + \frac{1}{(\sigma^p)^2}\sum_i \zeta(i|\mathbf{x},c)\mathbf{x}_i\\ &= \frac{-\mathbf{x}}{(\sigma^p)^2} + \frac{1}{(\sigma^p)^2}\mathbb{E}_{\zeta(i|\mathbf{x},c)}\{\mathbf{x}_i\}\end{aligned} \tag{A.4}$$

and

$$\begin{aligned}\frac{\partial}{\partial \mathbf{x}} \log \left(\sum_j \exp(-0.5\|\mathbf{x}-\mathbf{x}_j\|^2/(\sigma^p)^2) \right) &= \frac{-1}{(\sigma^p)^2} \sum_i \zeta(i|\mathbf{x})(\mathbf{x}-\mathbf{x}_i) \\ &= \frac{-\mathbf{x}}{(\sigma^p)^2} + \frac{1}{(\sigma^p)^2} \sum_i \zeta(i|\mathbf{x})\mathbf{x}_i \\ &= \frac{-\mathbf{x}}{(\sigma^p)^2} + \frac{1}{(\sigma^p)^2} \mathbb{E}_{\zeta(i|\mathbf{x})}\{\mathbf{x}_i\}. \end{aligned} \tag{A.5}$$

Thereby,

$$\zeta(i|\mathbf{x},c) = \frac{\delta_{c,c_i} \exp(-0.5\|\mathbf{x}-\mathbf{x}_i\|^2/(\sigma^p)^2)}{\sum_j \delta_{c,c_j} \exp(-0.5\|\mathbf{x}-\mathbf{x}_j\|^2/(\sigma^p)^2)} \tag{A.6}$$

$$\zeta(i|\mathbf{x}) = \frac{\exp(-0.5\|\mathbf{x}-\mathbf{x}_i\|^2/(\sigma^p)^2)}{\sum_j \exp(-0.5\|\mathbf{x}-\mathbf{x}_j\|^2/(\sigma^p)^2)}. \tag{A.7}$$

Then we see that

$$\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x}) = \frac{1}{(\sigma^p)^2} \left(\mathbb{E}_{\zeta(i|\mathbf{x},c)}\{\mathbf{x}_i\} - \mathbb{E}_{\zeta(i|\mathbf{x})}\{\mathbf{x}_i\} \right) =: \frac{1}{(\sigma^p)^2} \mathbf{b}(\mathbf{x},c) \tag{A.8}$$

and the Fisher information matrix is given by

$$\mathbf{J}(\mathbf{x}) = \frac{1}{(\sigma^p)^4} \mathbb{E}_{\hat{p}(c|\mathbf{x})} \left\{ \mathbf{b}(\mathbf{x},c)\mathbf{b}(\mathbf{x},c)^\mathsf{T} \right\}. \tag{A.9}$$

A.2. The Fisher information matrix for a continuous auxiliary variable

This section depicts the mathematical derivation for computing the Fisher information matrix in case of a continuous auxiliary variable. In particular, we utilize the Gaussian Process methodology and show how to obtain the result (2.40) given in section 2.6.2.

Starting from the definition of the Fisher information matrix

$$\mathbf{J}(\mathbf{x}_*) = \mathbb{E}_{p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*)} \left\{ \left(\frac{\partial}{\partial \mathbf{x}_*} \log p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*) \right) \left(\frac{\partial}{\partial \mathbf{x}_*} \log p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*) \right)^\mathsf{T} \right\}. \tag{A.10}$$

we need to do three things in order to compute this matrix: (I) estimate $p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*)$, (II) compute its derivative with respect to $\mathbf{x}_*$ and (III) calculate the expectation.

(I) For estimating $p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*)$, we can use a Gaussian Process, as mentioned in section 2.6.2, giving us

$$p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*) = \frac{1}{\sqrt{2\pi \mathrm{cov}(y_*)}} \exp\left(-\frac{(y_* - \bar{y}_*)^2}{2 \cdot \mathrm{cov}(y_*)} \right), \tag{A.11}$$

with

$$\bar{y}_* = \mathbf{k}_*^\mathsf{T} \boldsymbol{\alpha}, \quad \boldsymbol{\alpha} = (\mathbf{K} + \sigma_{GP}^2 \mathbf{I})^{-1} \mathbf{y} \tag{A.12}$$

$$\mathrm{cov}(y_*) = k_* - \mathbf{k}_*^\mathsf{T} (\mathbf{K} + \sigma_{GP}^2 \mathbf{I})^{-1} \mathbf{k}_* \tag{A.13}$$

in agreement with the Gaussian Process literature [131].

(II) The derivative can be decomposed into

$$\frac{\partial}{\partial \mathbf{x}_*} \log p(y_*|\mathbf{X}, \mathbf{y}, \mathbf{x}_*) = -\frac{1}{2}\frac{\partial}{\partial \mathbf{x}_*}\frac{(y_* - \bar{y}_*)^2}{\mathrm{cov}(y_*)} + \sqrt{2\pi \mathrm{cov}(y_*)} \cdot \frac{\partial}{\partial \mathbf{x}_*}\frac{1}{\sqrt{2\pi \mathrm{cov}(y_*)}} \tag{A.14}$$

with

$$\frac{\partial}{\partial \mathbf{x}_*}\frac{(y_* - \bar{y}_*)^2}{\mathrm{cov}(y_*)} = \left(\frac{\partial}{\partial \mathbf{x}_*}(y_* - \bar{y}_*)^2\right) \cdot \frac{1}{\mathrm{cov}(y_*)} + \left(\frac{\partial}{\partial \mathbf{x}_*}\frac{1}{\mathrm{cov}(y_*)}\right) \cdot (y_* - \bar{y}_*)^2, \tag{A.15}$$

where

$$\frac{\partial}{\partial \mathbf{x}_*}(y_* - \bar{y}_*)^2 = -4\beta(y_* - \bar{y}_*)\mathbf{t}_1, \tag{A.16}$$

$$\frac{\partial}{\partial \mathbf{x}_*}\frac{1}{\mathrm{cov}(y_*)} = \frac{2\beta}{\mathrm{cov}(y_*)^2}\mathbf{t}_2 \tag{A.17}$$

and

$$\frac{\partial}{\partial \mathbf{x}_*}\frac{1}{\sqrt{2\pi \mathrm{cov}(y_*)}} = \frac{\sqrt{\mathrm{cov}(y_*)}}{2\sqrt{2\pi}} \cdot \frac{\partial}{\partial \mathbf{x}_*}\frac{1}{\mathrm{cov}(y_*)}. \tag{A.18}$$

Thereby,

$$\mathbf{t}_1 = \sum_i (\mathbf{x}_i - \mathbf{x}_*)[\mathbf{k}_*]_i [\boldsymbol{\alpha}]_i \tag{A.19}$$

$$\mathbf{t}_2 = \sum_i [\mathbf{k}_*]_i \left(\sum_j (\mathbf{x}_i + \mathbf{x}_j - 2\mathbf{x}_*)[\mathbf{k}_*]_j [(\mathbf{K} + \sigma_{GP}^2 \mathbf{I})^{-1}]_{j,i}\right). \tag{A.20}$$

Combining these derivatives, we obtain

$$\frac{\partial}{\partial \mathbf{x}_*} \log p(y_*|\mathbf{X}, \mathbf{y}, \mathbf{x}_*) = \frac{\beta}{\mathrm{cov}(y_*)}\left(2(y_* - \bar{y}_*)\mathbf{t}_1 - \left(\frac{(y_* - \bar{y}_*)^2}{\mathrm{cov}(y_*)} - 1\right)\mathbf{t}_2\right). \tag{A.21}$$

(III) Inserting the result of the derivation into equation (A.10), we obtain

$$\begin{aligned}\frac{\mathrm{cov}(y_*)^2}{\beta^2}\mathbf{J}(\mathbf{x}_*) = \; & 4\mathbf{t}_1\mathbf{t}_1^\mathsf{T}\mathbb{E}_{p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*)}\left\{(y_* - \bar{y}_*)^2\right\} \\ & + \mathbf{t}_2\mathbf{t}_2^\mathsf{T}\mathbb{E}_{p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*)}\left\{\left(\frac{(y_* - \bar{y}_*)^2}{\mathrm{cov}(y_*)} - 1\right)^2\right\} \\ & - 2\left(\mathbf{t}_1\mathbf{t}_2^\mathsf{T} + \mathbf{t}_2\mathbf{t}_1^\mathsf{T}\right)\mathbb{E}_{p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*)}\left\{(y_* - \bar{y}_*) - \frac{(y_* - \bar{y}_*)^3}{\mathrm{cov}(y_*)}\right\}.\end{aligned}$$

Since $0 = \mathbb{E}_{p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*)}\{y_* - \bar{y}_*\} = \mathbb{E}_{p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*)}\left\{-\frac{(y_* - \bar{y}_*)^3}{\mathrm{cov}(y_*)}\right\}$ (first and third moment of a centered Gaussian), the last term vanishes. Finally, because $\mathbb{E}_{p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*)}\left\{(y_* - \bar{y}_*)^2\right\} = \mathrm{cov}(y_*)$ and $\mathbb{E}_{p(y_*|\mathbf{X},\mathbf{y},\mathbf{x}_*)}\left\{\left(\frac{(y_* - \bar{y}_*)^2}{\mathrm{cov}(y_*)} - 1\right)^2\right\} = 2$, the Fisher information matrix can be written as

$$\mathbf{J}(\mathbf{x}_*) = \frac{2\beta^2}{\mathrm{cov}(y_*)}\left(2\mathbf{t}_1\mathbf{t}_1^\mathsf{T} + \frac{1}{\mathrm{cov}(y_*)}\mathbf{t}_2\mathbf{t}_2^\mathsf{T}\right). \tag{A.22}$$

Appendix B.

Publications in the context of this thesis

Journal articles

[J17] Alexander Schulz, Johannes Brinkrolf, and Barbara Hammer. Efficient kernelisation of discriminative dimensionality reduction. *Neurocomputing*, 268(Supplement C):34 – 41, December 2017. Advances in artificial neural networks, machine learning and computational intelligence.

[J15b] Alexander Schulz, Andrej Gisbrecht, and Barbara Hammer. Using discriminative dimensionality reduction to visualize classifiers. *Neural Processing Letters*, 42(1):27–54, August 2015.

[J15a] Andrej Gisbrecht, Alexander Schulz, and Barbara Hammer. Parametric nonlinear dimensionality reduction using kernel t-sne. *Neurocomputing*, 147:71 – 82, January 2015. Advances in Self-Organizing Maps Subtitle of the special issue: Selected Papers from the Workshop on Self-Organizing Maps 2012 (WSOM 2012).

Conference and Workshop articles

[C16c] Benjamin Paaßen, Alexander Schulz, and Barbara Hammer. Linear Supervised Transfer Learning for Generalized Matrix LVQ. In Barbara Hammer, Thomas Martinetz, and Thomas Villmann, editors, *Proceedings of the Workshop New Challenges in Neural Computationn, NC^2 2016, Hannover, Germany, September 12, 2016*, number 4, pages 11–18, 2016.

[C16b] Cosima Prahm, Benjamin Paaßen, Alexander Schulz, Barbara Hammer, and Oskar Aszmann. Transfer Learning for Rapid Re-calibration of a Myoelectric Prosthesis after Electrode Shift. In Jaime Ibáñez, José González-Vargas, José María Azorín, Metin Akay, and José Luis Pons, editors, *Converging Clinical and Engineering Research on Neurorehabilitation II: Proceedings of the 3rd International Conference on Neural Rehabilitation, ICNR 2016, Segovia, Spain, October 18-21, 2016*, pages 153–157, Cham, 2016. Springer International Publishing.

[C16a] Alexander Schulz and Barbara Hammer. Discriminative Dimensionality Reduction in Kernel Space. In *Proceedings of the 24th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2016, Bruges, Belgium, April 27-29, 2016*. i6doc.com, 2016.

[C15f] Alexander Schulz, Bassam Mokbel, Michael Biehl, and Barbara Hammer. Inferring feature relevances from metric learning. In *IEEE Symposium Series on Computational Intelligence, SSCI 2015, Cape Town, South Africa, December 7-10, 2015*, pages 1599–1606, 2015.

[C15e] [1] Bassam Mokbel and Alexander Schulz. Towards Dimensionality Reduction for Smart Home Sensor Data. In Barbara Hammer, Thomas Martinetz, and Thomas Villmann, editors, *Proceedings of the Workshop New Challenges in Neural Computation, NC^2 2015, Aachen, Germany, October 10, 2015*, number 3, pages 41–48, 2015.

[C15d] [2] Alexander Schulz and Barbara Hammer. *Computer Analysis of Images and Patterns: 16th International Conference, CAIP 2015, Valletta, Malta, September 2-4, 2015, Proceedings, Part II*, chapter Visualization of Regression Models Using Discriminative Dimensionality Reduction, pages 437–449. Springer International Publishing, Cham, 2015.

[C15c] Alexander Schulz and Barbara Hammer. Discriminative dimensionality reduction for regression problems using the fisher metric. In *International Joint Conference on Neural Networks, IJCNN 2015, Killarney, Ireland, July, 12-17, 2015*, pages 1–8, July 2015.

[C15b] Patrick Blöbaum, Alexander Schulz, and Barbara Hammer. Unsupervised Dimensionality Reduction for Transfer Learning. In Michel Verleysen, editor, *Proceedings of the 23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2015, Bruges, Belgium, April, 22-24, 2015*, pages 507–512. i6doc.com, 2015.

[C15a] Alexander Schulz and Barbara Hammer. Metric learning in dimensionality reduction. In *Proceedings of the International Conference on Pattern Recognition Applications and Methods, ICPRAM 2015, Lisbon, Portugal, January, 10-12, 2015*, pages 232–239, 2015.

[C14d] Benoît Frénay, Daniela Hofmann, Alexander Schulz, Michael Biehl, and Barbara Hammer. Valid interpretation of feature relevance for linear data mappings. In *Computational Intelligence and Data Mining, CIDM 2014, IEEE Symposium Series on Computational Intelligence, SSCI 2014, Orlando, Florida, USA, December, 9-12, 2014*, pages 149–156, 2014.

[C14c] Andrej Gisbrecht, Alexander Schulz, and Barbara Hammer. *Pattern Recognition Applications and Methods: International Conference, ICPRAM 2013 Barcelona, Spain, February 15-18, 2013 Revised Selected Papers*, chapter Discriminative Dimensionality Reduction for the Visualization of Classifiers, pages 39–56. Springer International Publishing, Cham, 2015.

[C14b] Patrick Blöbaum and Alexander Schulz. Transfer Learning without given Correspondences. In Barbara Hammer, Thomas Martinetz, and Thomas Villmann, editors, *Proceedings of the Workshop New Challenges in Neural Computation, NC^2 2014, Münster, Germany, September, 2, 2014*, pages 42–51, 2014.

[C14a] Alexander Schulz, Andrej Gisbrecht, and Barbara Hammer. Relevance learning for dimensionality reduction. In Michel Verleysen, editor, *Proceedings of the 22nd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2014, Bruges, Belgium, April, 23-25, 2014*, pages 165–170. i6doc.com, 2014.

[C13c] Alexander Schulz, Andrej Gisbrecht, and Barbara Hammer. Classifier inspection based on different discriminative dimensionality reductions. In Barbara Hammer, Thomas Martinetz, and Thomas Villmann, editors, *Proceedings of the Workshop - New Challenges in Neural Computation, NC^2 2013, Saarbrücken, Germany, September, 3, 2013*, pages 77–86. TR Machine Learning Reports, 2013.

[1] Winner of the *Best Poster* award at NC^2 2015.
[2] Winner of the *Best Poster* award at CAIP 2015.

[C13b] Alexander Schulz, Andrej Gisbrecht, and Barbara Hammer. *Advances in Computational Intelligence: 12th International Work-Conference on Artificial Neural Networks, IWANN 2013, Puerto de la Cruz, Tenerife, Spain, June 12-14, 2013, Proceedings, Part I*, chapter Using Nonlinear Dimensionality Reduction to Visualize Classifiers, pages 59–68. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[C13a] [3] Barbara Hammer, Andrej Gisbrecht, and Alexander Schulz. Applications of discriminative dimensionality reduction. In *Proceedings of the 2nd International Conference on Pattern Recognition Applications and Methods, ICPRAM 2013, Barcelona, Spain, February, 15-18, 2013*, pages 33–41, 2013.

[C12b] Barbara Hammer, Andrej Gisbrecht, and Alexander Schulz. *Advances in Self-Organizing Maps: 9th International Workshop, WSOM 2012 Santiago, Chile, December 12-14, 2012 Proceedings*, chapter How to Visualize Large Data Sets?, pages 1–12. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[C12a] Alexander Schulz, Andrej Gisbrecht, Kerstin Bunte, and Barbara Hammer. How to visualize a classifier? In *Proceedings of the Workshop - New Challenges in Neural Computation, NC^2 2012, Graz, Austria, August, 28, 2012*, pages 73–83. Machine Learning Reports, 2012.

[3] Winner of the *Best Paper* award at ICPRAM 2013.

Bibliography

[1] Environmental and industrial machine learning group. `http://research.ics.aalto.fi/eiml/datasets.shtml`.

[2] Tecator meat sample dataset.

[3] S. Amari. *Differential geometrical methods in statistics*, volume 28 of *Lecture notes in statistics ; 28*. Springer, Berlin [u.a.], 1985.

[4] S. Amari and H. Nagaoka. *Methods of Information Geometry*. Translations of mathematical monographs. American Mathematical Society, 2007.

[5] W. Arlt, M. Biehl, A. E. Taylor, S. Hahner, R. Libe, B. A. Hughes, P. Schneider, D. J. Smith, H. Stiekema, N. Krone, E. Porfiri, G. Opocher, J. Bertherat, F. Mantero, B. Allolio, M. Terzolo, P. Nightingale, C. H. L. Shackleton, X. Bertagna, M. Fassnacht, and P. M. Stewart. Urine steroid metabolomics as a biomarker tool for detecting malignancy in adrenal tumors. *J Clinical Endocrinology and Metabolism*, 96:3775–3784, 2011.

[6] M. Aupetit and T. Catz. High-dimensional labeled data analysis with topology representing graphs. *Neurocomputing*, 63:139–169, 2005.

[7] R. Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5:537–550, 1994.

[8] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12:2385–2404, 2000.

[9] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15:1373–1396, 2003.

[10] D. Bell and H. Wang. A formalism for relevance and its application in feature subset selection. *Machine Learning*, 41(2):175–195, 2000.

[11] V. V. Belle and P. Lisboa. Research directions in interpretable machine learning models. In *21st European Symposium on Artificial Neural Networks, ESANN 2013, Bruges, Belgium, April 24-26, 2013*, 2013.

[12] V. V. Belle and P. Lisboa. White box radial basis function classifiers with component selection for clinical prediction models. *Artificial Intelligence in Medicine*, 60(1):53–64, 2014.

[13] A. Bellet, A. Habrard, and M. Sebban. A survey on metric learning for feature vectors and structured data. *CoRR*, abs/1306.6709, 2013.

[14] R. E. Bellman. *Adaptive control processes - A guided tour*. Princeton, New Jersey, U.S.A., 1961.

[15] T. D. Bie, L.-C. Tranchevent, L. M. M. van Oeffelen, and Y. Moreau. Kernel-based data fusion for gene prioritization. In *ISMB/ECCB (Supplement of Bioinformatics)*, pages 125–132, 2007.

[16] M. Biehl, B. Hammer, E. Merényi, A. Sperduti, and T. Villmann. Learning in the context of very high dimensional data (dagstuhl seminar 11341). volume 1, pages 67–95, 2011.

[17] M. Biehl, B. Hammer, P. Schneider, and T. Villmann. Metric learning for prototype based classification. In M. Bianchini, M. Maggini, and F. Scarselli, editors, *Innovations in Neural Information – Paradigms and Applications*, Studies in Computational Intelligence 247, pages 183–199. Springer, 2009.

[18] M. Biehl, P. Schneider, D. Smith, H. Stiekema, A. Taylor, B. Hughes, C. Shackleton, P. Stewart, and W. Arlt. Matrix relevance lvq in steroid metabolomics based classification of adrenal tumors. In *ESANN*, 2012.

[19] C. M. Bishop, M. Svensén, and C. K. Williams. Gtm: The generative topographic mapping. *Neural computation*, 10(1):215–234, 1998.

[20] J. Blitzer, S. Kakade, and D. P. Foster. Domain adaptation with coupled subspaces. In *AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, pages 173–181, 2011.

[21] P. Breheny and W. Burchett. Visualization of regression models using visreg, 2013.

[22] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Statistics/Probability Series. Wadsworth Publishing Company, Belmont, California, U.S.A., 1984.

[23] S. Briesemeister. *Interpretable Machine Learning Approaches in Computational Biology*. PhD thesis, University of Tübingen, 2011.

[24] S. Briesemeister, J. Rahnenführer, and O. Kohlbacher. Going from where to why - interpretable prediction of protein subcellular localization. *Bioinformatics*, 26(9):1232–1238, 2010.

[25] E. T. Brown, J. Liu, C. E. Brodley, and R. Chang. Dis-function: Learning distance functions interactively. In *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, pages 83–92. IEEE, 2012.

[26] K. Bunte, M. Biehl, and B. Hammer. A general framework for dimensionality reducing data visualization mapping. *Neural Computation*, 24(3):771–804, 2012.

[27] K. Bunte, P. Schneider, B. Hammer, F.-M. Schleif, T. Villmann, and M. Biehl. Limited rank matrix learning, discriminative dimension reduction and visualization. *Neural Networks*, 26:159–173, 2012.

[28] A. C. Cameron and F. A. G. Windmeijer. An r-squared measure of goodness of fit for some common nonlinear regression models. *Journal of Econometrics*, 77(2):329–342, Apr. 1997.

[29] D. Caragea, D. Cook, H. Wickham, and V. Honavar. Visual methods for examining svm classifiers. In Simoff et al. [144], pages 136–153.

[30] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[31] Y. Chen, E. K. Garcia, M. R. Gupta, A. Rahimi, and L. Cazzanti. Similarity-based classification: Concepts and algorithms. *JMLR*, 10:747–776, 2009.

[32] Committee on the Analysis of Massive Data, Committee on Applied and Theoretical Statistics, Board on Mathematical Sciences and Their Applications, Division on Engineering and Physical Sciences, and National Research Council. *Frontiers in Massive Data Analysis*. National Academic Press, 2013.

[33] R. D. Cook. SAVE: A method for dimension reduction and graphics in regression. *Communications in Statistics-Theory and Methods*, 29(9):2109–2121, 2000.

[34] G. Da San Martino and A. Sperduti. Mining structured data. *Computational Intelligence Magazine, IEEE*, 5(1):42–49, Feb 2010.

[35] D.Cohn. Informed projections. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, pages 849–856. MIT Press, 2003.

[36] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38, 1977.

[37] I. S. Dhillon, D. S. Modha, and W. S. Spangler. Class visualization of high-dimensional data with applications. *Computational Statistics & Data Analysis*, 41(1):59–90, November 2002.

[38] G. Doquire and M. Verleysen. A comparison of multivariate mutual information estimators for feature selection. In *ICPRAM (1)*, pages 176–185, 2012.

[39] G. Doquire and M. Verleysen. Mutual information-based feature selection for multilabel classification. *Neurocomputing*, 122:148–155, 2013.

[40] E. P. dos Santos Amorim, E. V. Brazil, J. D. II, P. Joia, L. G. Nonato, and M. C. Sousa. ilamp: Exploring high-dimensional spacing through backward multidimensional projection. In *IEEE VAST*, pages 53–62. IEEE Computer Society, 2012.

[41] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

[42] G. Elidan, B. Packer, G. Heitz, and D. Koller. Convex point estimation using undirected bayesian transfer hierarchies. *CoRR*, abs/1206.3252, 2012.

[43] A. Endert, P. Fiaux, and C. North. Semantic interaction for visual text analytics. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 473–482. ACM, 2012.

[44] D. Francois, F. Rossi, V. Wertz, and M. Verleysen. Resampling methods for parameter-free and robust feature selection with mutual information. *Neurocomputing*, 70(7-9):1276–1288, 2007.

[45] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[46] B. Frénay, G. Doquire, and M. Verleysen. Is mutual information adequate for feature selection in regression ? *Neural Networks*, 48:1–7, 2013.

[47] B. Frénay, G. Doquire, and M. Verleysen. Theoretical and empirical study on the potential inadequacy of mutual information for feature selection in classification. *Neurocomputing*, 112:64–78, 2013.

[48] B. Frénay, M. van Heeswijk, Y. Miche, M. Verleysen, and A. Lendasse. Feature selection for nonlinear models with extreme learning machines. *Neurocomputing*, 102:111–124, 2013.

[49] J. H. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2 2010.

[50] K. Fukumizu, F. R. Bach, and M. I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *J. Mach. Learn. Res.*, 5:73–99, Dec. 2004.

[51] T. Gärtner. A survey of kernels for structured data. *SIGKDD Explor. Newsl.*, 5(1):49–58, July 2003.

[52] X. Geng, D.-C. Zhan, and Z.-H. Zhou. Supervised nonlinear dimensionality reduction for visualization and classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 35(6):1098–1107, 2005.

[53] A. Gisbrecht and B. Hammer. Data visualization by nonlinear dimensionality reduction. *WIREs Data Mining and Knowledge Discovery*, accepted.

[54] A. Gisbrecht, D. Hofmann, and B. Hammer. Discriminative dimensionality reduction mappings. In J. Hollmén, F. Klawonn, and A. Tucker, editors, *IDA*, Lecture Notes in Computer Science, pages 126–138. Springer, 2012.

[55] A. Gisbrecht, B. Mokbel, and B. Hammer. Relational generative topographic mapping. *Neurocomputing*, 74(9):1359–1371, 2011.

[56] A. Gisbrecht, A. Schulz, and B. Hammer. Parametric nonlinear dimensionality reduction using kernel t-sne. *Neurocomputing*, 147:71–82, 2015.

[57] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In *Advances in Neural Information Processing Systems 17*, pages 513–520. MIT Press, 2004.

[58] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.

[59] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *JMLR*, 3:1157–1182, 2003.

[60] B. Hammer and A. Hasenfuss. Topographic mapping of large dissimilarity datasets. *Neural Computation*, 22(9):2229–2284, 2010.

[61] B. Hammer, H. He, and T. Martinetz. Learning and modeling big data. In M. Verleysen, editor, *ESANN*, pages 343–352, 2014.

[62] B. Hammer, D. Hofmann, F. Schleif, and X. Zhu. Learning vector quantization for (dis-)similarities. *Neurocomputing*, 131:43–51, 2014.

[63] J. Hernandez-Orallo, P. Flach, and C. Ferri. Brier curves: a new cost-based visualisation of classifier performance. In *International Conference on Machine Learning*, June 2011.

[64] T. C. Hesterberg, N. H. Choi, L. Meier, and C. Fraley. Least angle and l1 penalized regression: A review. *Statistics Surveys*, 2008.

[65] M. Hilbert and P. López. The world's technological capacity to store, communicate, and compute information. *Science*, 332(6025):60–65, 2011.

[66] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[67] D. Hofmann, F.-M. Schleif, B. P. en, and B. Hammer. Learning interpretable kernelized prototype-based models. *Neurocomputing*, revised, 2013.

[68] T. Hofmann and J. M. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(1):1–14, 1997.

[69] T. W. House. Big data research and development initiative, 2012.

[70] U. Hübner, N. B. Abraham, and C. O. Weiss. Dimensions and entropies of chaotic intensity pulsations in a single-mode far-infrared nh3 laser. *Phys. Rev. A*, 40:6354–6365, 1989.

[71] A. Hyvarinen, J. Karhunen, and E. Oja. *Independent component analysis*. Adaptive and learning systems for signal processing, communications, and control. John Wiley, New York, Chichester, Weinheim, 2001.

[72] T. Iwata, K. Saito, and N. Ueda. Parametric embedding for class visualization. In *Neural Information Processing Systems (NIPS) 17*, 2004.

[73] A. Jakulin, M. Možina, J. Demšar, I. Bratko, and B. Zupan. Nomograms for visualizing support vector machines. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD '05, pages 108–117, New York, NY, USA, 2005. ACM.

[74] Y. Jin and B. Hammer. Computational intelligence in big data [guest editorial]. *IEEE Comp. Int. Mag.*, 9(3):12–13, 2014.

[75] G. H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *ICML'94*, pages 121–129, 1994.

[76] D. H. Jr. and D. L. Rubinfeld. Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management*, 5(1):81 – 102, 1978.

[77] S. Kaski and J. Peltonen. Dimensionality reduction for data visualization [applications corner]. *IEEE Signal Process. Mag.*, 28(2):100–104, 2011.

[78] A. Kerren, J. T. Stasko, J.-D. Fekete, and C. North. *Information Visualization : Human-Centered Issues and Perspectives*. Lecture Notes in Computer Science. Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg, 2008.

[79] T. Khalil. Big data is a big deal. White House, Sep 2012.

[80] M. Kim and V. Pavlovic. Dimensionality Reduction using Covariance Operator Inverse Regression. In *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[81] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artif. Intell.*, 97(1-2):273–324, 1997.

[82] T. Kohonen, J. Hynninen, J. Kangas, J. Laaksonen, and K. Torkkola. LVQ_PAK: The Learning Vector Quantization program package. Report A30, Helsinki University of Technology, Laboratory of Computer and Information Science, Jan. 1996.

[83] T. Kohonen, M. R. Schroeder, and T. S. Huang, editors. *Self-Organizing Maps*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 3rd edition, 2001.

[84] R. Kothari and M. Dong. Decision trees for classification: A review and some new results.

[85] L. F. Kozachenko and N. Leonenko. Sample estimate of the entropy of a random vector. *Problems Inform. Transmission*, 23:95–101, 1987.

[86] A. Kraskov, H. Stögbauer, and P. Grassberger. Estimating mutual information. *Phys. Rev. E*, 69:066138, 2004.

[87] U. H.-G. Kreßel. Advances in kernel methods. chapter Pairwise classification and support vector machines, pages 255–268. MIT Press, Cambridge, MA, USA, 1999.

[88] C. Krier, D. Francois, F. Rossi, and M. Verleysen. Feature clustering and mutual information for the selection of variables in spectral data. In *15th ESANN 2007, Bruges, Belgium, April 25-27, 2007, Proceedings*, pages 157–162, 2007.

[89] B. Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2013.

[90] S. Kullback. *Information theory and statistics*. Wiley series in probability and mathematical statistics. Wiley, 1959.

[91] C. C. Laczny, N. Pinel, N. Vlassis, and P. Wilmes. Alignment-free Visualization of Metagenomic Data by Nonlinear Dimension Reduction. *Scientific Reports*, 4, Mar. 2014.

[92] J. Lee and M. Verleysen. Quality assessment of dimensionality reduction: Rank-based criteria. *Neurocomputing*, 72(7-9):1431–1443, 2009.

[93] J. A. Lee, E. Renard, G. Bernard, P. Dupont, and M. Verleysen. Type 1 and 2 mixtures of kullback-leibler divergences as cost functions in dimensionality reduction based on similarity preservation. *Neurocomput.*, 112:92–108, July 2013.

[94] J. A. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer, 2007.

[95] J. A. Lee and M. Verleysen. Scale-independent quality criteria for dimensionality reduction. *Pattern Recognition Letters*, 31(14):2248–2257, 2010.

[96] J. M. Lee. *Riemannian manifolds : an introduction to curvature*. Graduate Texts in mathematics. Springer, New York, 1997.

[97] A. Lendasse, J. A. Lee, V. Wertz, and M. Verleysen. Time series forecasting using CCA and kohonen maps - application to electricity consumption. In M. Verleysen, editor, *ESANN 2000, Bruges (Belgique)*, pages 329–334, April 2000.

[98] A. Lendasse, J. A. Lee, V. Wertz, and M. Verleysen. Forecasting electricity consumption using nonlinear projection and self-organizing maps. *Neurocomputing*, 48(1-4):299–311, 2002.

[99] K.-C. Li. Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association*, 86(414):316–327, 1991.

[100] K. C. Li. On Principal Hessian Directions for Data Visualization and Dimension Reduction: Another Application of Stein's Lemma. *Journal of the American Statistical Association*, 87(420):1025–1039, 1992.

[101] P. J. G. Lisboa. Interpretability in machine learning - principles and practice. In F. Masulli, G. Pasi, and R. R. Yager, editors, *WILF*, volume 8256 of *Lecture Notes in Computer Science*, pages 15–21. Springer, 2013.

[102] M. Long, J. Wang, G. Ding, S. J. Pan, and P. S. Yu. Adaptation regularization: A general framework for transfer learning. *IEEE Trans. Knowl. Data Eng.*, 26(5):1076–1089, 2014.

[103] B. Ma, H. Qu, and H. Wong. Kernel clustering-based discriminant analysis. *Pattern Recognition*, 40(1):324–327, 2007.

[104] M. Maillot, M. Aupetit, and G. Govaert. *Extraction of betti numbers based on a generative model*, pages 537–542. i6doc.com publication, 2012.

[105] G. D. S. Martino and A. Sperduti. Mining structured data. *IEEE Comp. Int. Mag.*, 5(1):42–49, 2010.

[106] O. Melnik. Decision region connectivity analysis: A method for analyzing high-dimensional classifiers. *Machine Learning*, 48(1-3):321–351, 2002.

[107] R. Memisevic and G. Hinton. Multiple relational embedding. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 913–920. MIT Press, Cambridge, MA, 2005.

[108] B. Mokbel, B. Paassen, and B. Hammer. Adaptive distance measures for sequential data. In M. Verleysen, editor, *ESANN*, pages 265–270, 2014.

[109] Y. Moreau and L.-C. Tranchevent. Computational tools for prioritizing candidate genes: boosting disease gene discovery. *Nat Rev Genet*, 13(8):523–536, 2012.

[110] J. Nilsson, F. Sha, and M. I. Jordan. Regression on manifolds using kernel dimension reduction. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 697–704, New York, NY, USA, 2007. ACM.

[111] R. Nilsson, J. M. Pena, J. Björkegren, and J. Tegnér. Consistent feature selection for pattern recognition in polynomial time. *Journal of Machine Learning Research*, 8:589–612, 2007.

[112] J. Novovicová, P. Somol, M. Haindl, and P. Pudil. Conditional mutual information based feature selection for classification task. In *CIARP'07*, pages 417–426, 2007.

[113] E. Oja. Simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 15(3):267–273, Nov. 1982.

[114] K. O'Neill, N. Aghaeepour, J. Å pidlen, and R. Brinkman. Flow cytometry bioinformatics. *PLoS Comput Biol*, 9(12):1–10, 12 2013.

[115] S. Ortega-Martorell, I. Olier, T. Delgado-Goni, M. Ciezka, M. Julià-Sapé, P. J. G. Lisboa, and C. Arús. Semi-supervised source extraction methodology for the nosological imaging of glioblastoma response to therapy. In *2014 IEEE Symposium on Computational Intelligence and Data Mining, CIDM 2014, Orlando, FL, USA, December 9-12, 2014*, pages 93–98, 2014.

[116] S. Ortega-Martorell, H. Ruiz, A. Vellido, I. Olier, E. Romero, M. Julia-Sape, J. D. Martin, I. H. Jarman, C. Arus, and P. J. G. Lisboa. A novel semi-supervised methodology for extracting tumor type-specific mrs sources in human brain data. *PLoS ONE*, 8(12):1–14, 12 2013.

[117] C. Otte. Safe and interpretable machine learning: A methodological review. In C. Moewes and A. Nürnberger, editors, *Computational Intelligence in Intelligent Data Analysis*, volume 445 of *Studies in Computational Intelligence*, pages 111–122. Springer Berlin Heidelberg, 2013.

[118] B. Paaßen. Java Sorting Programs, doi: 10.4119/unibi/2900684, 2016.

[119] B. Paaßen, B. Mokbel, and B. Hammer. Adaptive structure metrics for automated feedback provision in Java programming. In M. Verleysen, editor, *Proceedings of the ESANN*, 2015.

[120] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2011.

[121] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Trans. on Knowl. and Data Eng.*, 22(10):1345–1359, Oct. 2010.

[122] J. Peltonen. *Data exploration with learning metrics*. Citeseer, 2004.

[123] J. Peltonen, A. Klami, and S. Kaski. Improved learning of riemannian metrics for exploratory analysis. *Neural Networks*, 17:1087–1100, 2004.

[124] J. Peltonen, M. Sandholm, and S. Kaski. Information retrieval perspective to interactive data visualization. In M. Hlawitschka and T. Weinkauf, editors, *Proceedings of Eurovis 2013, The Eurographics Conference on Visualization*. The Eurographics Association, 2013.

[125] D. H. Peluffo-Ordóñez, J. A. Lee, and M. Verleysen. Recent methods for dimensionality reduction: A brief comparative analysis. In *22th European Symposium on Artificial Neural Networks, ESANN 2014, Bruges, Belgium, April 23-25, 2014*, 2014.

[126] S. Philips, J. Pitton, and L. Atlas. Perceptual feature identification for active sonar echoes. In *OCEANS 2006*, pages 1–6, Sept 2006.

[127] R. Polikar and C. Alippi. Guest editorial learning in nonstationary and evolving environments. *IEEE Trans. Neural Netw. Learning Syst.*, 25(1):9–11, 2014.

[128] F. Poulet. Visual svm. In C.-S. Chen, J. Filipe, I. Seruca, and J. Cordeiro, editors, *ICEIS (2)*, pages 309–314, 2005.

[129] P. Pudil, J. Novovicová, and J. Kittler. Floating search methods in feature selection. *Pattern Recogn. Lett.*, 15(11):1119–1125, Nov. 1994.

[130] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *(ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, pages 759–766, 2007.

[131] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[132] P. E. Rauber, R. R. O. d. Silva, S. Feringa, M. E. Celebi, A. X. Falcao, and A. C. Telea. Interactive Image Feature Selection Aided by Dimensionality Reduction. In E. Bertini and J. C. Roberts, editors, *EuroVis Workshop on Visual Analytics (EuroVA)*. The Eurographics Association, 2015.

[133] M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The rprop algorithm. In *Proceedings of the IEEE International Conference on Neural Networks*, pages 586–591. IEEE Press, 1993.

[134] S. Roweis. Machine learning data sets, 2012. Available at http://www.cs.nyu.edu/ roweis/data.html.

[135] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE*, 290:2323–2326, 2000.

[136] C. Rudin and K. L. Wagstaff. Machine learning for science and society. *Machine Learning*, 95(1):1–9, 2014.

[137] H. Ruiz, T. A. Etchells, I. H. Jarman, J. D. Martin, and P. J. Lisboa. A principled approach to network-based classification and data representation. *Neurocomputing*, 112:79 – 91, 2013. Advances in artificial neural networks, machine learning, and computational intelligenceSelected papers from the 20th European Symposium on Artificial Neural Networks (ESANN 2012).

[138] S. Rüping. *Learning Interpretable Models*. PhD thesis, Dortmund University, 2006.

[139] T. D. Sanger. Optimal Unsupervised Learning in a Single-Layer Linear Feedforward Neural Network. *Neural Networks*, 2:459–473, 1989.

[140] E. Schaffernicht, R. Kaltenhaeuser, S. Verma, and H.-M. Gross. On estimating mutual information for feature selection. In *Artificial Neural Networks – ICANN 2010*, volume 6352, pages 362–367. Springer Berlin Heidelberg, 2010.

[141] P. Schneider, M. Biehl, and B. Hammer. Adaptive relevance matrices in learning vector quantization. *Neural Computation*, 21:3532–3561, 2009.

[142] S. Seo and K. Obermayer. Soft learning vector quantization. *Neural Computation*, 15(7):1589–1604, 2003.

[143] S. Si, D. Tao, and B. Geng. Bregman divergence-based regularization for transfer subspace learning. *IEEE Trans. Knowl. Data Eng.*, 22(7):929–942, 2010.

[144] S. J. Simoff, M. H. Böhlen, and A. Mazeika, editors. *Visual Data Mining - Theory, Techniques and Tools for Visual Analytics*, volume 4404 of *Lecture Notes in Computer Science*. Springer, 2008.

[145] J. Sinkkonen. *Learning metrics and discriminative clustering*. Dissertations in computer and information science. Report D; 2. Helsinki University of Technology; Teknillinen korkeakoulu, 2003-11-21.

[146] D. K. Slonim. From pattern to pathways: gene expression data analysis comes of age. *Nature Genetics Supplement*, 32:502–508, 2002.

[147] G. K. Smyth. *Limma: linear models for microarray data.*, pages 397–420. Springer, New York, 2005.

[148] L. Song, A. Smola, K. Borgwardt, and A. Gretton. Colored maximum variance unfolding. *Advances in neural information processing systems*, 20:1385–1392, 2008.

[149] M. Strickert, B. Hammer, T. Villmann, and M. Biehl. Regularization and improved interpretation of linear data mappings and adaptive distance measures. In *IEEE SSCI CIDM 2013*, pages 10–17. IEEE Computational Intelligence Society, 2013.

[150] J. Tenenbaum, V. da Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[151] J. Tikka. *Input Variable Selection Methods for Construction of Interpretable Regression Models*. TKK Dissertations in information and computer science. Helsinki University of Technology, 2008.

[152] I. Tsamardinos and C. F. Aliferis. Towards Principled Feature Selection: Relevancy, Filters and Wrappers. In *in Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.

[153] B. A. Turlach. Bandwidth Selection in Kernel Density Estimation: A Review. In *CORE and Institut de Statistique*, pages 23–493, 1993.

[154] UCL. Spectral wine database, 2007. Provided by Prof. Marc Meurens.

[155] N. Ueda and R. Nakano. Deterministic annealing em algorithm. *Neural Netw.*, 11(2):271–282, Mar. 1998.

[156] L. van der Maaten. Learning a parametric embedding by preserving local structure. *Journal of Machine Learning Research*, 5:384–391, 2009.

[157] L. van der Maaten. Barnes-hut-sne. *CoRR*, abs/1301.3342, 2013.

[158] L. van der Maaten. Accelerating t-sne using tree-based algorithms. *Journal of Machine Learning Research*, 15(1):3221–3245, 2014.

[159] L. van der Maaten and G. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[160] L. van der Maaten, E. Postma, and H. van den Herik. Dimensionality reduction: A comparative review. Technical report, Tilburg University Technical Report, TiCC-TR 2009-005, 2009.

[161] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[162] A. Vellido, J. Martin-Guerrero, and P. Lisboa. Making machine learning models interpretable. In *ESANN*, 2012.

[163] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *Journal of Machine Learning Research*, 11:451–490, 2010.

[164] J. R. Vergara and P. A. EstÃ©vez. A review of feature selection methods based on mutual information. *Neural Computing and Applications*, 24(1):175–186, 2014.

[165] M. Verleysen. Learning high-dimensional data. *Limitations and Future Trends in Neural Computation*, 186:141–162, 2003.

[166] M. Verleysen, F. Rossi, and D. Francois. Advances in feature selection with mutual information. In *Similarity-Based Clustering*, volume 5400, pages 52–69. 2009.

[167] M. Vidyasagar. *A Theory of Learning and Generalization*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2nd edition, 2002.

[168] C. Wang and S. Mahadevan. Manifold alignment using procrustes analysis. In *ICML '08*, pages 1120–1127, New York, NY, USA, 2008. ACM.

[169] C. Wang and S. Mahadevan. Manifold alignment without correspondence. In *IJCAI'09*, pages 1273–1278, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[170] X. Wang, S. Wu, X. Wang, and Q. Li. Svmv - a novel algorithm for the visualization of svm classification results. In J. Wang, Z. Yi, J. Zurada, B.-L. Lu, and H. Yin, editors, *Advances in Neural Networks - ISNN 2006*, volume 3971 of *Lecture Notes in Computer Science*, pages 968–973. Springer Berlin / Heidelberg, 2006.

[171] M. Ward, G. Grinstein, and D. A. Keim. *Interactive Data Visualization: Foundations, Techniques, and Application*. A. K. Peters, Ltd, 2010.

[172] A. Weigend and N. Gershenfeld. Results of the time series prediction competition at the santa fe institute. In *Neural Networks, 1993., IEEE International Conference on*, pages 1786–1793 vol.3, 1993.

[173] K. Q. Weinberger and L. K. Saul. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *Unfolding, Proceedings of the 21st National Conference on Artificial Intelligence*. AAAI, 2006.

[174] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009.

[175] D. M. Witten and R. Tibshirani. Supervised multidimensional scaling for visualization, classification, and bipartite ranking. *Comput. Stat. Data Anal.*, 55(1):789–801, Jan. 2011.

[176] H.-M. Wu. Kernel sliced inverse regression with applications to classification. *Journal of Computational and Graphical Statistics*, 17(3):590–610, 2008.

[177] Z. Yang, J. Peltonen, and S. Kaski. Scalable optimization of neighbor embedding for visualization. In S. Dasgupta and D. Mcallester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, volume 28, pages 127–135. JMLR Workshop and Conference Proceedings, May 2013.

[178] L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *J. Mach. Learn. Res.*, 5:1205–1224, 2004.

[179] Y. Zhai, Y.-S. Ong, and I. Tsang. The emerging "big dimensionality". *Computational Intelligence Magazine, IEEE*, 9(3):14–26, Aug 2014.