Bielefeld University
Faculty of Technology
Biodata Mining & Applied Neuroinformatics Group

Looking Inside Ensembles of negatively correlated Self-Organizing Maps

Der Technischen Fakultät der Universität Bielefeld vorgelegte

D i s s e r t a t i o n

zur Erlangung des akademischen Grades Dr.-Ing. von

Alexandra Scherbart

January 9, 2011

Supervisor: JunProf. Dr. Tim W. Nattkemper

Contents

List of Figures

List of Tables

Glossary

α	Parameter for trimmed-mean calculation, 26
b	Bias or intercept, 21
β	Exponent reweighting the predictors for aggregation, 102
$\mathcal{C}$	Output function of LLM, 20
C	Regularization parameter for SVM, 22
d_{in}	Number of features or columns in $\mathbf{X}$, 12
$\bar{e}$	Ensemble prediction error, 48
ϵ^{A}	Learning step width for supervised adaptation of $\mathbf{A}$, 20
ϵ^{in}	Learning step width in unsupervised adaptation, 16
ϵ^{out}	Learning step width for supervised adaptation of $\mathbf{w}^{\text{out}}$, 20
η	Parameter for reducing error variables for GNG, 18
$\bar{f}$	Aggregated ensemble prediction, 48
f_m	Single predictor output, 48
G	Number of rows (assessed feature relevances) of $\mathbf{Z}$, 39
γ	NCL parameter controlling the penalty strength, 52
h	Single hypothesis, 45
h_σ	Gaussian neighborhood function, 16
K	Size of randomly sampled subspaces, see RSM., 52
κ	Index of winning node, $\kappa \in \{1, \ldots, n_l\}$, 16
λ	NCL strength parameter, 54
M	Number of ensemble predictors, 48
μ	Fuzzification parameter for FCL, 19
N	Number of samples or rows in $\mathbf{X}$, 12
n_l	Number of nodes of VQ-based approach, e. g. SOM, 15
Ω	Term connecting mean predictor error and diversity, 50
Φ	Transformation of samples into $\mathcal{F}$ for SVM, 22
ϕ	Internal (non)linear functions, 54
π	Parameter for SVR, 22
r	Pearson's correlation coefficient, 24
σ	Width of Gaussian neighborhood function, 16
t	Number of learning epochs, 16
θ	Identically distributed random vectors of samples, 38

Υ	NCL parameter, defining type of penalty function, 54
ε	Parameter of insensitive loss function for SVR, 22
w_m	Weight of ensemble predictor m, 48
ξ	Slack variables for SVM, 22
ζ	Parameter for interpolating new nodes for GNG, 18
$\mathbf{A}$	Locally trained linear mappings from input to output space, 19
$\mathbf{K}_m$	Subset of features randomly chosen for predictor m, 52
$\boldsymbol{p}$	Permutation of a vector, 109
$\mathbf{T}$	Matrix of training samples, 12
$\mathbf{T}_m$	Training data for ensemble predictor m, 51
$\mathbf{U}$	Matrix of degree of membership of samples to prototypes, 19
$\mathbf{v}$	Node of VQ-based approach, e. g. of a SOM, 15
$\mathbf{w}^{\text{in}}$	Vector used to build prototypes of VQ, 15
$\mathbf{w}^{\text{out}}$	Vector approximating the distribution of the target values, 19
$\mathbf{X}$	Matrix of input data with N rows and d_{in} columns, 12
$\mathbf{x}$	Input vector, samples, 12
$\mathbf{y}$	Real-valued output vector, target variables, 12
$\mathbf{Z}$	Matrix of feature weightings as estimations of feature relevance, 39
$\mathcal{F}$	High-dimensional feature space for SVM, 22
$\mathcal{H}$	Base hypothesis space, 45
P	Parameter space, 24
$\mathcal{X}$	Input space, $\mathbb{R}^{d_{\text{in}}}$, 12
Bagging	Bootstrap aggregating, 51
BT	Bagged Trees, 39
CV	10-fold Cross-Validation, 24
DIV	Diversity along the ensemble predictors, 50
$\text{DIV}_{\text{inter}}$	Inter-SOM diversity, 63
$\text{DIV}_{\text{intra}}$	Intra-SOM diversity, 63
EL	Ensemble Learning, 45
FCL	Fuzzy C-means clustering, 18
GNG	Growing Neural Gas, 17
Lasso	Least absolute shrinkage and selection operator, 107
LERRANCO	Local Linear Ensembles for Regression with Resampling And NCL, 58
LLM_{FCL}	Local Linear Map based on FCL, 20

LLM_{GNG}	Local Linear Map based on GNG, 20
LLM	Local Linear Map, 19
LLM_{NG}	Local Linear Map based on NG, 20
LLM_{SOM}	Local Linear Map based on SOM, 20
LM	Linear Least Squares, 38
MLP	Multi-Layer Perceptron, 65
MSE_{Map}	Mean squared error per single ensemble predictor, 50
MSE_{Test}	MSE on test data to assess generalization performance, 63
MS	Mass Spectrometry, 13
NCL	Negative Correlation Learning, 52
NG	Neural Gas, 16
ν-SVR	ν-Support Vector Regression, 23
OOB	Out-of-bag data, see Bagging, 51
PIP	Peak Intensity Prediction, 12
PLS	Partial Least Squares, 38
RF	Random Forests, 38
RMSE	Root Mean Square Error, 25
RSM	Random Subspace Method, 52
RWSM	Random Weighted Subspace Method, 107
SOM	Self-Organizing Maps, 15
SoR	Total sum of assessed relevance $\mathbf{z}$, 40
SVM	Support Vector Machine, 20
SVR	Support Vector Regression, 22
VQ	Vector Quantization, 12
WTA	Winner-takes-all rule, 20

1 Introduction

Data from life science is an application representative of domains for which a high number of variables are available. As it is often the case in bioinformatics applications, the dimensionality of data exceeds the number of samples available, resulting in a low cardinality of samples and a sparse representation of the underlying process that generated the data. Moreover, the functions to be approximated, where the samples are generated or recorded from, are very complex. This puts new challenges in application areas as e. g. meta-genome expression, micro-array analysis or Mass Spectrometry, as well as in research, development and evaluation of task-adequate techniques. Machine Learning offers a promising way to cope with these key issues.

Obstacles arising in these domains are four-fold: (a) finding an adequate and suitable predictor, (b) improving the performance of prediction, (c) reducing the number of variables by feature selection for lowering time as well as the costs of computation and (d) providing a basis for data mining purposes to get further insights into the data generating process.

Since these high-dimensional vectors are supposed to contain redundant and interdependent information, methods are needed, that are able to cope with this data and tools that are capable to reduce dimensionality of feature space. The task of finding a suitable subset of features is well-known as feature or variable selection and is one of the central problems in machine learning [Blum and Langley, 1997]. Focusing on relevant features that contribute to model the underlying task offers simpler models, scalability and better generalization [Guyon and Elisseeff, 2003]. In addition, it reduces computational costs.

1.1 Prediction of Peptide Peak Intensities

One challenging task in life sciences, more precisely in bioinformatics, is the prediction of spectrum peak intensities from pre-computed molecular features, which would pave the way to a better understanding of spectrometry data and improved spectrum evaluation. This issue serves as an illustrative example throughout this thesis.

Mass Spectrometry (MS) is one of the key techniques in proteomics and life sciences for the analysis and identification of proteins and peptides. Matrix-assisted laser desorption ionization (MALDI) is one of the most often used technique for the analysis of whole cell proteomes in high-throughput experiments.

In proteomics, proteins in a complex sample are aimed to be quantitatively characterized or protein abundances in cells between different environmental states are compared. The characterization of proteomes would be helpful for a better understanding of the underlying cellular mechanisms. Opposed to labeling methods, label-free methods directly use peak heights to estimate peptide abundances.

There are different applications of MALDI-MS where the prediction of peak heights (referred to as intensities) in the spectrum are needed for further improvements: Commonly, proteins are digested into smaller fragments, the peptides, which are then analyzed by MS providing mass information. The peaks are detected at a certain mass and provided as a list with their corresponding peak heights. These depend on the ionizability of peptides and on the physiochemical properties. Protein identification is commonly done by comparing the peak's masses from a spectrum - the so called protein mass finger print (PMF) - to theoretical PMFs in a data base, generating a score for each comparison. Different tools are available for this purpose. For an overview see Shadforth et al. [2005]. These tools rarely use peak intensities, because there is no model to calculate the theoretical PMFs directly. The use of peak intensities could improve the reliability of protein identification without lowering the error rate, as was shown by Elias et al. [2004] for tandem MS.

For the prediction of MALDI PMF there has been one study so far by Gay et al. [2002], who applied different regression and classification algorithms. Tang and et al. [2006] used multi-layer neural networks to predict peptide detectabilities (i. e. the frequency with which peaks occur in spectra) in LC/MS ion trap spectra, which is a related problem.

An algorithmic approach for peak intensity prediction is a non-trivial task because of several difficulties: The extraction of PMF from spectra is a signal processing task which can not be done perfectly. Data from this domain is always very noisy and contains errors introduced by preprocessing steps in the wet lab as well as in signal processing. Misidentifications may even lead to wrong sequences. Intensity values can be distorted due to the unknown scale of spectra. It is nearly impossible to come by a large enough data set from real proteins where the content is known and allows for absolute quantification. There is no perfect gold standard, because of the not reproducible and non-unique relation between the observed peak height and the peptide concentration in the sample.

To overcome these obstacles for the prediction of peak intensities in MALDI-TOF spectra, a completely data-driven and model-free approach is applied. Techniques from the field of machine learning and artificial neural networks are used in order to enhance existing model-based approaches and provide therewith a higher reliability of protein identification and label-free quantification. Based on a pre-selected training set of peptide/peak intensity pairs, the goal is to model the non-linear relationship between peptides and peptide peak heights only using the peptide sequence information and the chemical properties. To apply an algorithmic approach, the given peptide string representations are preprocessed and transformed into a numerical description of the chemical characterizations. Using these as input samples for training, the machine learning techniques are applied to find a mapping of the peptides to their numerical value describing the intensity. These models may provide a basis for further analysis and facilitating interpretation of the underlying data generating process.

1.2 Bias and Variance and the Trade-Off

From a machine learning point of view, the problem can be stated as follows: Given a training set $\mathbf{T} = \{(\mathbf{x}, y)_n, n = 1, \ldots, N\}$, consisting of input-output pairs: peptide samples $\mathbf{x}_n$, which are elements of feature space $\mathcal{X} = \mathbb{R}^{d_{\text{in}}}$, and real-valued outputs, i. e. intensities, $y_n \in \mathbb{R}$ inducing a regression problem. If labels are categorical or binary in contrast to continuous targets, the learning task is classification. In some sense, every classification task can be transformed into a regression problem, as an estimation of the posterior class probabilities. Both are instances of supervised learning problems, which are stated as "the search for algorithms that reason from externally supplied instances to produce general hypotheses, which then make predictions about future instances." [Kotsiantis, 2007]. In unsupervised learning samples are unlabeled or no label is accessible. The task is to hopefully discover meaningful clusters in which the characteristics of the underlying data can be separated well. The majority of clustering techniques requires a specification of the number of clusters a priori. For a lack of any prior knowledge about the underlying data distribution, an optimal number of clusters representing the data best has to be estimated e. g. by Silhouette Coefficient [Kaufman and Rousseeuw, 1990] or the Gap Statistics introduced by Tibshirani et al. [2000].

The complex relationship between the specific tasks and the wide range of possible architectures has led to vast research in designing and adapting problem-specific algorithms, in the majority of cases resulting in single highly-complex predictors. However, single learners typically suffer from computational, statistical as well as representational problems [Dietterich, 2002]. The single learner may fail in finding a successful hypothesis when sufficient size of training data compared to the size of hypothesis space is absent. A large amount of data makes it hard and inefficient for one learner to approximate and requires the construction of batch algorithms. The computational drawbacks arise from the difficulty in finding the best hypothesis. Many algorithms employ local search or greedy heuristics and therefore may get stuck in local optima. In some cases of applications it is nearly impossible to represent the true hypothesis by the learning algorithm.

As it is well known from the theory of machine learning, every learner can be decomposed into a bias and a variance term. The bias-variance trade-off is deduced as the error accounting for the model complexity. The bias-variance decomposition is formulated based on Hastie et al. [2001]. Assume the data arising from a model

$y = f(\mathbf{x}) + \epsilon$, where ϵ is the random error with $E\{\epsilon\} = 0$ and $var\{\epsilon\} = \sigma^2$.

$$\begin{aligned} E\{(f(\mathbf{x}) - y)^2\} &= (E\{f(\mathbf{x})\} - y)^2 + E\{(f(\mathbf{x}) - E\{f(\mathbf{x})\})^2\} + var\{\epsilon\} \\ &= bias(f(\mathbf{x}))^2 + var(f(\mathbf{x})) + var\{\epsilon\} \end{aligned}$$

The last term is the *irreducible* error and hence cannot be avoided or controlled. The first term, the bias component, measures how much the expected value of the estimate differs from the true mean target y. The second component is the variance, the expected squared deviation of $f(\mathbf{x})$ around its average. Minimizing both bias and variance and controlling the trade-off is a hard task: Low bias can only be achieved at the price of high variance [Geman et al., 1992] and vice-versa, but can be optimized by means of regularization or cross-validation. The estimator's model is built only on the training samples, but desired to be capable to predict future, unseen samples from the same source, namely to generalize well. This *generalization* ability is an essential requirement and directly coupled to the "model complexity" or "capacity" expressed as the variance. An estimator which approximates very closely every of the training samples may fail badly in predicting new samples and hence lead to bad generalization. This undesired behavior called *over-fitting* with too complex approximating estimators can be controlled by imposing a restriction to model complexity and variance.

Kotsiantis [2007] covers the best-known classification architectures in a review from the vast amount of existing approaches. The algorithms are classified into six categories: Logic-based algorithms as e. g. Decision Trees [Quinlan, 1986], Perceptron-based techniques [Rosenblatt, 1962; Rumelhart et al., 1987], statistical learning algorithms, instance-based learning e. g. k-nearest neighbor (kNN) [Cover and Hart, 1967], and Support Vector Machines (SVMs) [Vapnik, 1995]. All of these exhibit their own strengths and weaknesses, and there does not exist one *universal* learning algorithm for *every* purposes: Instead one has to select a learner appropriately to find a good compromise between the limiting factors accuracy, the number of free parameters, the speed of learning and prediction, the tolerance to missing values or redundant variables and the transparency. In terms of statistical inference, Wolpert and Macready [1997] show that for both static and time-dependent optimization problems, the average performance of any pair of algorithms across all possible problems is identical. They introduce the theorem under the name of "no free lunch", which means if an algorithm performs superior to some particular problems, the reverse must be true over the set of all other problems.

1.3 Why apply Vector-Quantization Based Self-Organizing Maps?

In the context of bioinformatics, the data puts new challenges and frequently occurs sparse, noisy, unbalanced, with missing values and may be high-dimensional [Seiffert et al., 2006]. From this point of view, the use of Self-Organizing Maps (SOMs) as a representative of prototype based classifiers and learning vector quantization [Kohonen, 1982] in this field of application is rarely recommended. Neural networks based on gradient descending techniques may get stuck in local minima. Difficulties may arise from the number of parameters to tune and to be set before training a single SOM. For example the number of nodes supplied, are far from being uniform. As a rule of thumb, the relation of the two grid sizes should be equal to the relation of the first two principal components. On the other hand, the dimension of nodes along one grid axis should be of the same order of magnitude as the number of variables.

Other architectures, as for example the SVM, are known to perform more robust and insensitive against outliers, have excellent generalization abilities and show their strength with high-dimensional input data [Burges, 1998]. SVMs are a class of learning algorithms that are designed to implicitly transfer input feature vectors into a high-dimensional space and calculate the optimal linear separating hyperplane. However, it is difficult to interpret their models and parameter-tuning can be time-consuming.

But, in contradiction to the black box SVM, their transparency redounds to SOMs advantage when one is more interested in comprehensibility than in the best learning accuracy. Against this background, traditionally, decision trees are widely accepted and applied in medicine and biology [Seiffert et al., 2006].

The prototype based method of SOM-type provides a promising approach to find a set of clusters and characteristic prototypes representing the samples best according to the statistical properties of the data. The SOM is one of the most popular and widely used artificial neural network algorithm for clustering, classification, data mining and visualization purposes. They are very efficient, allow fast training and adaptation to additional data as well as low memory-usage and offer transparency. The formation of SOMs as topographic maps is inspired by the organization of early sensory processing in the cerebral cortex [Kohonen, 2001]. Similar functionalities or processing of external stimuli are allocated in neighboring regions on the planar organized feature map. They are designed to facilitate a mapping from a potentially high-dimensional input space onto a lower-dimensional output space for dimension reduction. The topology preserving projections offer enhanced visualizations [Ultsch, 1993, 2003a,b; Himberg, 2000; Vesanto, 1999] and references therein. They preserve topology in that sense that if two nodes are located next to each other on the defined grid structure, they will be caused to response to samples in input space that are likewise close to each other. There are a lot of variations proposed extending the classical architecture of SOMs. Martinetz et al. [1993] introduced the vector quantizing *Neural Gas* (NG), neglecting the neighborhood preserving property of SOMs. The neurons are adapted according to the ranking of the neurons depending on their distance to the presented sample in the input space.

The task of mass spectrometry prediction and peptide prototyping corresponds to the task of unsupervised clustering as well as classification and supervised prediction. After assigning every input point to a prototype, a prediction of a real-valued output

$\mathbf{y}$ has to be done. To solve this task, the SOM is extended by an additional layer of nodes. Local Linear Maps (LLMs) [Ritter, 1991], a type of artificial neural net, combine an unsupervised vector quantization (VQ) algorithm based on SOMs with supervised linear learning principles. The LLM achieves global non-linearity by fitting a set of *local* linear functions to the training data.

1.4 Why Ensemble Learning?

The drawbacks and difficulties in designing an adequate and appropriate predictor, has led to change of paradigms in the recent years: Multiple learners supersede single learners and solve the task at hand by dividing it into several sub-tasks in terms of a divide-and-conquer approach. This new direction provides a new concept of improving single predictors by combining them to an *ensemble* or committee of separately trained simple machines.

Ensemble Learning (EL) was paid much attention in the literature recently. The output of several learning algorithms is combined to build powerful learning machines. This concept has been proven to increase predictive power over single base learners for a great variety of classification and regression problems [Hansen and Salamon, 1990; Krogh and Vedelsby, 1995; Breiman, 1996a; Dietterich, 2002]. Ensembles voting from a set of different hypotheses cope with the drawbacks of statistical, computational and representational problems [Dietterich, 2002] typically arising with single learning algorithms. Given a set of observations $\mathbf{X} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$, the problem in regression ensembles is to select a set of appropriate predictors $H = \{f_1, \dots, f_M\}$ from the base hypothesis space $\mathcal{H}$ and to aggregate the outputs of the M base learners to one ensemble prediction $\bar{f}(\mathbf{x})$ as a convex combination of the ensemble predictors $\{f_1, \dots, f_M\}$.

Krogh and Vedelsby [1995] proved that the ensemble error is guaranteed to be less than or equal to the average quadratic error of the individual predictors. The expected quadratic error of the ensemble can be decomposed into three terms similar to the bias-variance decomposition for one single learner leading to the bias-variance-covariance decomposition [Geman et al., 1992]. For regression tasks, the ensemble error rate is given by simple averaging over the set of $m = 1, \dots, M$ predictor outputs f_m, as in Equation (1.1):

$$\bar{e}(\mathbf{x}) \;=\; (\bar{f}(\mathbf{x}) - y)^2 \;=\; \sum_{m=1}^{M} \frac{1}{M}(f_m(\mathbf{x}) - y)^2 - \sum_{m=1}^{M} \frac{1}{M}(f_m(\mathbf{x}) - \bar{f}(\mathbf{x}))^2 \tag{1.1}$$

On the right side of Equation (1.1), the first term is the average error of individual predictors. The second term measures how much each single ensemble member diverges from the ensemble output $\bar{f}$, the so called *diversity* or *ambiguity*. It is determined by both variance and covariance [Brown et al., 2005c]. The higher the diversity, the lower is the ensemble error rate, if the mean component error is fixed. This leads to two potentially conflicting targets: Balancing diversity against single predictor accuracy, is the most critical and challenging task for ensemble methods. Maximizing only diversity, may worsen prediction performance of every single learner. On the other hand, focusing only on the minimization of the ensemble predictors, may lead to very similar nets with low prediction error each, but also low ambiguity. In this worst case, even a high quantity of very similar ensemble predictors brings no gain compared to just one single learner. The success of ensembles is mainly brought with a shift of diversity or complexity of the learning function from one learner to many. The function to predict is approximated by a *set* of hypotheses rather than by just one single hypothesis. As a consequence, in contrast to "classical" learning architectures, the concept of ensemble learning is rather based on forcing *different* hypotheses.

Regarding SOMs, training an ensemble or, more generally, a set of SOMs has been proposed quite early by Fritzke [1994] allowing growing structures, hierarchically arranged structures [Miikkulainen, 1990] or a combination of both, as proposed by Dittenbach et al. [2000], or by Multi-SOMs [Goerke et al., 2001]. EL offers by contrast to these approaches a non-competing strategy of training the single networks. In the light of new theory behind EL, in particular Negative Correlation Learning (NCL) [Liu and Yao, 1999], the question arises if SOM EL can benefit from non-independent learning when individual learning stages are interlinked by *inter-SOM diversity* error rates. NCL allows to balance between single network accuracy and diversity controlled by cooperation of neural networks, thereby dispensing with a sub-local accuracy for a higher overall generalization ability.

1.5 Potential of SOM Ensemble Learning

In a first brief study [Scherbart and Nattkemper, 2008], we were able to show that EL applied to SOMs has clear potential. SOM ensembles are found superior to other VQ-algorithms in a regression application. We have proposed to take emphasis on the sources of *implicit* diversity by combination of the two re-sampling methods Bagging (bootstrap aggregating) [Breiman, 1996a] and Random Subspace Method [Ho, 1998] to enforce differences between the ensemble predictors, which is referred to as *inter-SOM diversity*.

One interesting feature of the SOM architecture is that a single SOM can be seen as an ensemble itself, where each node represents one classifier and a locally acting function approximator. As a consequence, each SOM has a diversity feature representing the variances between the nodes and I refer to this as *intra-SOM diversity*. The *intra*-SOM diversity is mainly controlled by the width of the Gaussian neighborhood function. With a weak connectivity, one can expect a diverse SOM, but with high generalization error. A strongly intra-connected SOM usually offers non-ambiguous nets with low *intra-SOM diversity*. However, with SOMs applied as a classifier and reconsidered as an ensemble itself, we gain completely new perspectives in the context of diversity and EL: If the quantification of *intra-SOM diversity* is accomplished, how far is the intra-SOM diversity between the nodes inside each SOM affected by boosting the inter-SOM diversity? How well do SOM ensembles perform at all with NCL? And at last, do the prediction performances compare to other existing reference learners?

To shed light on that subjects, the new EL theories are introduced into the field of SOM ensemble learning and I propose an ensemble architecture supplying LLMs, namely the *L*ocal Linear *E*nsembles for *R*egression with *R*esampling *A*nd *N*egative *CO*rrelation Learning (LERRANCO). Nevertheless, in the context of EL, the base precondition for the success of any ensemble is the application of *weak* learners. Typical representatives of this category are decision trees, neural networks, nearest neighbors. I can show that ensemble predictors of SOM-type with a small number of nodes also fulfill the requirements. Moreover, with a committee of *weak* SOM learners the demand for individually fine-tuned parameters fades into the background.

The proposed LERRANCO ensemble architecture offers simpler models, scalability, efficiency and good generalization for regression as well as classification purposes. This will be demonstrated for several regression benchmark datasets applied and one challenging, high-dimensional, small and noisy dataset from proteomics. Focusing on the SOMs inside an ensemble, the question arises how far the interplay between SOMs and within SOMs succeeds. I will discuss to which extend boosting diversity affects the prediction performance and give insights of the interrelation between diversity and the sub-local accuracy inside SOMs.

To the best of our knowledge, there have been two works recently published on this topic: Minku et al. [2009] demonstrated the usefulness of NCL in *incremental learning* applied to neural networks of fixed size as well as to Self-Organizing Neural Grove [Inoue and Narihisa, 2003] as an extension of the SOM. Prudhomme and Lallich [2008] applied SOMs of size 20×20 for knowledge discovery purposes and NCL for ensemble predictor selection, but without a randomization in their training samples.

1.6 Outline

This thesis is organized as follows:

Chapter 2 introduces to the main issues at hand in the context of peak intensity prediction and describes the background and computational methods applied. The evaluation and analysis is presented towards a prediction of peak intensities with machine learning methods in a *single learner setup*. Special focus is laid on the issue of using relevant features in modeling the non-linear regression task.

In Chapter 3 the concepts of Ensemble Learning are explained and how diversity along ensemble predictors e. g. by Negative Correlation Learning, can be achieved. Parameterized penalty functions are derived, which *explicitly* boost the diversity along the ensemble predictors. Chapter 4 introduces to the proposed ensemble architecture LERRANCO based on SOMs. The categorization of strategies to force accurate and diverse ensemble predictors is subdivided into *implicit* and *explicit* ones, as well as *between* and *inside* the ensemble predictors. A quantification of the diversity inside each predictor is proposed.

Chapter 5 answers the question, if and how SOMs are suited as ensemble predictors. The evaluation features special consideration placed on the factors which *implicitly* force the diversity along the ensemble predictors. These general factors include the type of training algorithm, the topology of the nets, the training data and the initial conditions e. g. Gaussian neighborhood width.

In Chapter 6 the evaluation of the negatively correlated SOM ensembles as part-adapting structures is presented. It will be discussed to which extend boosting diversity *explicitly* by NCL affects the prediction performance and the sub-local accuracy inside the SOMs. I can show how well SOM ensembles perform at all with NCL and that the prediction performances are comparable or even superior to other existing reference learners.

In Chapter 7 the challenging problem of aggregation is discussed, i. e. how to combine predictors and on which to rely on.

Chapter 8 covers the challenges arising in the context of Ensemble Learning and Feature Selection. With the objective of assessing the feature relevance or importance, two approaches are examined, dependent either on an a priori or a posteriori assessment of feature relevance.

This thesis is closed by Chapter 9 with a summary and discussion of the results.

1.7 Publications

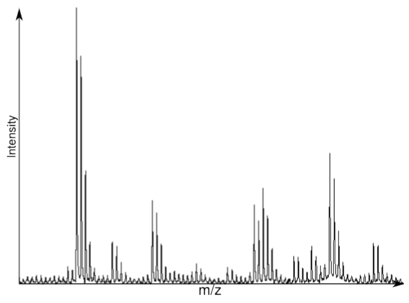
Parts of this thesis have been published or are under review.
As first author:

- A. Scherbart, T. W. Nattkemper. Looking Inside Self-Organizing Map Ensembles with Resampling and Negative Correlation Learning. Neural Networks, Volume 24, Issue 1, pages 130–141, 2011.
- A. Scherbart, T. W. Nattkemper. The Diversity of Regression Ensembles Combining Bagging and Random Subspace Method. In: Proc. of International Conference on Neural Information Processing, ICONIP 2008, Part II, LNCS 5507, pages 911–918.
- A. Scherbart, W. Timm, S. Böcker, T. W. Nattkemper. Improved Mass Spectrometry Peak Intensity Prediction by Adaptive Feature Weighting. In: Proc. of International Conference on Neural Information Processing, ICONIP 2008, Part I, LNCS 5506, pages 513–520.
- A. Scherbart, W. Timm, S. Böcker, T. W. Nattkemper. SOM-based Peptide Prototyping for Mass Spectrometry Peak Intensity Prediction. In: Proc. of Workshop on Self-Organizing Maps, WSOM 2007.
- A. Scherbart, W. Timm, S. Böcker, T. W. Nattkemper. Neural network approach for mass spectrometry prediction by peptide prototyping. In Proc. of International Conference on Artificial Neural Networks, ICANN 2007, Part II, LNCS 4669, pages 90–99.
- A. Scherbart. Contribution to OpenMS - An open-source framework for mass spectrometry: Class PeakIntensityPredictor added for peak intensity prediction since version 1.3 (February 13th, 2009). `http://open-ms.sourceforge.net/`

Co-authorship:

- W. Timm, A. Scherbart, S. Böcker, O. Kohlbacher, T. W. Nattkemper, Peak intensity prediction in MALDI-TOF mass spectrometry: a machine learning study to support quantitative proteomics., BMC Bioinformatics 9 (2008) 443, doi:10.1186/1471-2105-9-443,
URL `http://www.biomedcentral.com/1471-2105/9/443`.

2 Peak Intensity Prediction in a Single Learner Setup



Any protein is a compound of organic molecules, which consists of a very long chain or sequence of amino acids. Peptides are short chains of amino acids. The primary structure of any protein as a polypeptide chain is described by its unique string representation over the alphabet of 20 amino acids.

Mass spectrometry (MS) is a key technique for the analysis and identification of proteins. In proteomics, proteins in a complex sample are aimed to be quantitatively characterized or protein abundances in cells between different environmental states are compared. Spectra provide mass, more precisely mass-to-charge ratio (m/z), information as shown as an example in the figure on the right. The peaks are detected at a certain mass and recorded as a list with their corresponding peak heights. After some preprocessing steps, these peaks' masses are used as input for a given task-specific database. By comparing the list of peaks' masses from the spectrum to theoretical known patterns of proteins, the entire best-matching protein can be identified by a database search. For details of the preprocessing and the peak extraction, I refer to Timm et al. [2008]. Every matched peak corresponds to a peptide, which is a substring of the best-matching protein sequence. The peptide peak heights, referred to as intensities, depend not only on their abundance, but on their ionizability as a function of their physiochemical properties and environment. A prediction of spectrum peak intensities from pre-computed molecular features would pave the way to a better understanding of spectrometry data and improving the reliability of protein identification and label-free protein quantitation.

To this end, one way to build a predictor is to model the non-linear relationship between peptides and peak intensity values in mass spectra, only using the peptide's sequence information and its derived chemical properties. By the application of machine learning methods, the function which maps the peptide to an intensity value does thus not demand an explicit formula.

In a supervised learning context, machine learning methods are applied to build models that are trained to predict the correct output for a given input. The problem can be stated as the search for algorithms that *learn from data* to find general hypotheses with insufficient knowledge about the data generating system. A predictor is trained by presenting pairs of input-output data. From a machine learning point of view, the regression problem of peak intensity prediction can be stated as follows: Given an input dataset $\mathbf{X} = \{(\mathbf{x}_n, y_n), n = 1, \dots, N\}$, consisting of N available input-output pairs: peptide samples $\mathbf{x}_n$, which are elements of feature space $\mathcal{X} = \mathbb{R}^{d_{\text{in}}}$, and real-valued outputs, i. e. the extracted corresponding intensities, $y_n \in \mathbb{R}$. In order to assess the performance of the prediction model, the input dataset $\mathbf{X}$ is split into two portions: the first portion $\mathbf{T}$ is used to train the predictor. The second portion is used for testing and contains those samples $\{\mathbf{x}_n \in \mathbf{X} : \mathbf{x}_n \notin \mathbf{T}, \forall n \in \{1, \dots, N\}\}$, which have not taken part in training. The quality of the resulting model is related to the prediction capability on unseen *test* samples and is termed the *generalization* performance.

For learning methods to be applicable, the peptide sequences, which are non-vectorial, need to be embedded into a numerical vector space. For each peptide sequence $n = 1, \dots, N$, numerical vectors $\mathbf{x}_n$ are calculated by the physico-chemical information about the amino acids constituting the peptide solely derived from the peptide's string representation. Several paradigms exist to derive such feature or variable vectors $\mathbf{x}_n$. To describe an entire peptide sequence by a vector of chemical properties of fixed length d_{in}, the amino acid specific values have to be cumulated to one entry. Hence, the total amount in matters of a specific chemical feature is regarded for each peptide sequence. Two different feature vectors are built in this work so as to reflect the chemical characteristics of the peptide, namely Heur and AAindex. Amino acid frequencies, typically used in bioinformatics, in conjunction with chemical features of the peptides are used to create the heuristically selected d_{in} =18-dimensional feature space Heur. The second feature space, called AAindex, consists of $d_{\text{in}} = 531$ features. With these two feature sets, two different numeric descriptions of the peptides' sequences are available as input data matrix $\mathbf{X}$ for the learning procedure. Often, the number of features d_{in}, i. e. the number of columns of $\mathbf{X}$, is also called the *dimensionality* of $\mathbf{X}$.

In this chapter, I give a brief introduction to the problem at hand, and describe the single learner setup to peak intensity prediction (PIP). The foundations are laid according to the machine learning methods and specifications of the problem. I propose to apply the Local Linear Map (LLM) as a vector-quantization (VQ)-based approach (see 2.3.1 on page 15) to overcome the obstacles and derive transparent models for peptide prototyping. Successful predictors are obtained by fitting sets of local linear functions approximated by local experts (2.3.5 on page 26). Separately trained transparent predictors based on SOMs can give good generalization results and come even close in accuracy to other reference learning algorithms. The evaluation of experiments are given as a proof of concept with comparable results to those obtained by ν-Support Vector Regression (see ν-SVR, 2.3.2 on page 20). Given an estimation of feature relevance a priori, an adaptation of feature space facilitates an improved peak intensity prediction as proposed in 2.4.3 on page 40 for the LLM predictor. Having performed evaluations with regard to types of learning architectures, different setups of training sets and feature sets, the priority objective is to decide which type of feature set has greatest potential therewith in a general learner setup.

2.1 MS data

In this study one peptide dataset A of MALDI mass spectra is used. It consists of 66 spectra of 29 different proteins, with 16 of these proteins being present in multiple spectra. Peak extraction steps include soft filtering, baseline correction, peak picking and isotopic deconvolution in the corresponding raw spectra. The resulting list of peaks is matched against masses derived from a theoretical tryptic digestion. For preprocessing, normalization of the intensities is necessary, because spectra do not have the same scale. For a MALDI spectrum the exact amount of protein sample that leads to it is not known, nor it is possible to scale spectra belonging to the same protein by the same amount.

These steps for A result in 857 matched peaks corresponding to 415 different peptides. For further reading and details of preprocessing, I refer to Timm et al. [2008]. Subsequently, the natural logarithm of the intensities is applied to compute the final intensity output values, s. t. $\mathbf{y}' = ln(\mathbf{y} + 1)$. The distorted intensity values become approximately normal distributed and the error additive.

2.2 Benchmark Datasets

In this thesis, I use five different (synthetic as well as real-world) benchmark datasets to allow a general comparison to other regression and ensemble architectures applied to these problems. These are Friedman, Boston, Forestfires [Asuncion and Newman, 2007], NO2 and two challenging small and noisy (high-dimensional) datasets from proteomics named Heur and AAindex [Scherbart et al., 2007b; Timm et al., 2008]. All datasets are of small size and low-dimensional, except for the last one, AAindex, where the dimensionality exceeds the number of available samples as it is often the case in bioinformatics applications. Aside from Friedman dataset, randomly selected 10% of the input samples are put aside for testing and the remaining 90% are used to build the predictors. Only for the synthetic dataset of type Friedman, 200 randomly generated samples are used to build the predictors and 2000 for testing. All features are centered and normalized by variance prior to training. To overcome the distortion of target values as it is the case in the Forestfires data, the response variable $\mathbf{y}$ is transformed following Cortez and Morais [2007] according to $\mathbf{y}' = ln(\mathbf{y} + 1)$.

In the following, the benchmark datasets are presented briefly and a survey of the size and the training/testing ratios is given in Table 2.1.

Friedman This artificial data set has ten independent variables uniformly distributed over [0,1], while only five out of these ten are used to define y [Friedman, 1991].

$$y = 10\sin(\pi \cdot x_1 \cdot x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + e, \text{ where } e \text{ is } N(0,1).$$

Boston This data set consists of 506 input samples and 13 input variables, concerning housing values in suburbs of Boston [Asuncion and Newman, 2007].

Forestfires This data set is collected to provide a basis for a prediction of the burned area of forestfires, in the northeast region of Portugal, by using meteorological and other data of 12 variables and 517 instances [Cortez and Morais, 2007].

NO2 The data are a subsample of 506 observations from a data set with seven variables, that originate in a study where air pollution at a road is related to traffic volume and meteorological variables, collected by the Norwegian Public Roads Administration [Vlachos, 2005].

AAindex It consists of 372 (input,output) pairs corresponding to peptides and their recorded intensity values in mass spectrograms. For all peptide sequences, numerical representations are derived by physico-chemical information about the amino acids constituting the peptide yielding a 531-dimensional input feature space. The feature vectors are attributes taken from the amino acid index database [Kawashima et al., 1999] extended by peptide length, mass, and numbers and fractions of acidic, basic, polar, aliphatic and arginine residues.

Heur As for AAindex, it consists of the same 372 samples corresponding to peptides and their observed peptide peak intensities. It is built by amino acid frequencies, together with different types of characterization that are assumed to be relevant for MALDI ionization and additional features that are chosen in an ad hoc feature forward selection. For details about the selected features and the feature selection process, see [Timm et al., 2008].

Dataset	**# variables**	**# Training**	**# Test**
Friedman#1	10	200	2000
Boston	13	506	10%
Forestfires	12	517	10%
NO2	7	450	10%
AAindex	531	372	10%
Heur	18	372	10%

Table 2.1: Dataset Survey

2.3 Towards Peak Intensity Prediction with Machine Learning methods

Given the datasets of peptide characterizations, several machine learning algorithms can be applied to solve the task of peak intensity prediction. The ones employed here are Local Linear Maps (2.3.1) - with different VQ-algorithms supplied as (Growing) Neural Gas (GNG, NG), Self-Organizing Maps (SOMs) as well as Fuzzy C-means- clustering (FCL) - and ν-Support Vector Regression (see 2.3.2) for comparison purposes. The evaluation process is described in 2.3.3 including model selection and assessment. Based on the issues in handling data prior to learning covered in 2.3.4, the results (see 2.3.5) of the different learning algorithms are evaluated in terms of these issues to model the non-linear relationship between peptide and corresponding peak heights.

2.3.1 Local Linear Map (LLM) - VQ-based approach

Local Linear Maps (LLMs) [Ritter, 1991] are a representative of prototype based classifiers and learning vector quantization [Kohonen, 1982]. They provide an attractive option for the type of task at hand as Sharkey et al. [1997] underlined for neural networks in general, when data is noisy, where explicit knowledge about the task is not available, when fast and incremental learning is demanded. The prototype based method of VQ-type provides a promising and transparent approach to find a set of clusters and characteristic prototypes best representing the samples according to the statistical properties of the data.

For determining peptide prototypes and learning the mapping into the output, i. e. intensity, space, I use the Local Linear Map (LLM)-architecture. The LLM, as a mixture of linear experts, combines unsupervised vector quantization algorithm for computing a voronoi tessellation of the input space $\mathcal{X}$ and derives implicit models for characterizing peptides and feature analysis. The second step consists of supervised adaptation of the neural network and prediction of peaks' intensities. A LLM is composed of n_l nodes, where each node $\mathbf{v}_i, i = 1, \ldots, n_l$ consists of the prototype vectors $\mathbf{w}_i^{\text{in}} \in \mathbb{R}^{d_{\text{in}}}$, which are used to adapt to the input data $\mathbf{x}_n \in \mathbb{R}^{d_{\text{in}}}, n = 1, \ldots, N$.

Vector quantization with SOM A *Self-Organizing Map (SOM)* consists of a set of n_l regular ordered nodes $\mathbf{v}_i, i = 1, \ldots, n_l$, which are connected to each other via a two-dimensional grid structure, defining a neighborhood between the nodes and a topology in feature space.

In the unsupervised training phase of SOMs, the learning procedure changes the weights of the prototype vectors $\mathbf{w}_i^{\text{in}}$ according to a Gaussian neighborhood function

h_σ with width σ decreasing over grid distance $r_i(\mathbf{x}_n, n_l)$ around the winning node $\mathbf{v}_\kappa$:

$$\begin{aligned} \Delta \mathbf{w}_i^{\text{in}} &= \epsilon^{\text{in}} \cdot h_\sigma\left(r_i(\mathbf{x}_n, n_l)\right) \cdot \left(\mathbf{x}_n - \mathbf{w}_i^{\text{in}}\right) \\ h_\sigma(r_i(\mathbf{x}_n, n_l)) &= \exp\left(-\tfrac{r_i(\mathbf{x}_n, n_l)}{2\sigma^2}\right). \qquad (2.1) \\ r_i(\mathbf{x}_n, n_l) &= d_p(\kappa, i) \\ \kappa &= \arg\min_i \left\{\|\mathbf{x}_n - \mathbf{w}_i^{\text{in}}\|\right\}. \qquad (2.2) \end{aligned}$$

The grid distance $d_p(\kappa, i)$ is defined between two indices along the grid structure, where p indicates the type of metric used. With $p = 1$, applied in this work, the distance used is Manhattan- or Minkowski distance as a common choice, while with $p = 2$, it corresponds to the Euclidean distance. The learning rate ϵ^{in} and the size of the neighborhood σ should decrease with training steps [Kohonen, 2001]. Depending on the number of epochs t_{max} the learning rate ϵ^{in} is decaying exponentially from initial value ϵ_i^{in} to a final value ϵ_f^{in} and the size of the neighborhood σ decays from σ_i to σ_f:

$$\epsilon^{\text{in}}(t) = \epsilon_i^{\text{in}} \cdot \left(\frac{\epsilon_f^{\text{in}}}{\epsilon_i^{\text{in}}}\right)^{\frac{t}{t_{max}}}, \qquad \sigma(t) = \sigma_i \cdot \left(\frac{\sigma_f}{\sigma_i}\right)^{\frac{t}{t_{max}}}. \qquad (2.3)$$

Good results were found for rapid learning with $\epsilon_i^{\text{in}} = 0.5$, $\epsilon_f^{\text{in}} = 0.01$, $\sigma_i = 2.0$ and $\sigma_f = 0.4$ for $t \leq t_{max}$ and keeping $\epsilon^{\text{in}} = \epsilon_f^{\text{in}} = 0.01$, $\sigma = \sigma_f = 0.4$ fixed for the rest of the learning process ($t_{max} \leq t \leq 2 \cdot t_{max}$). A simulation sequence of a SOM for a ring-shaped uniform probability distribution is shown in Figure 2.1.

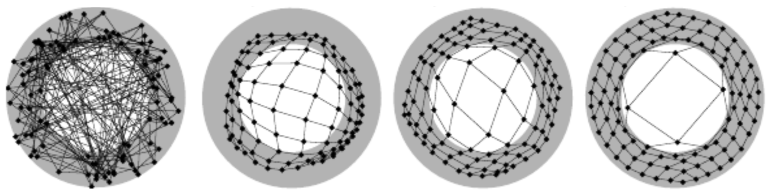


Figure 2.1: Self-Organizing Map simulation sequence for a ring-shaped uniform probability distribution. Image is taken from Fritzke [1997] for 0, 2500, 10000 and 40000 training epochs.

I also applied Neural-Gas (NG) clustering [Martinetz et al., 1993], Growing Neural Gas (GNG) [Fritzke, 1994], and Fuzzy C-means (FCL) [Bezdek, 1981] clustering instead of SOM in the input space for comparison.

Vector quantization with NG With NG, the distance between input pattern $\mathbf{x}_n$ and prototype $\mathbf{v}_i$ yields a ranking among the neural gas nodes. The learning procedure changes the weights according to the ranking of the prototypes[1] $r_i(\mathbf{x}_n, n_l)$, such that

$$\begin{aligned} \Delta \mathbf{w}_i^{\text{in}} &= \epsilon^{\text{in}} \cdot h_\lambda\left(r_i(\mathbf{x}_n, n_l)\right) \cdot \left(\mathbf{x}_n - \mathbf{w}_i^{\text{in}}\right) \\ \text{with} \quad h_\lambda\left(r_i(\mathbf{x}_n, n_l)\right) &= \exp\left(-\tfrac{r_i(\mathbf{x}_n, n_l)}{\lambda}\right). \end{aligned}$$

Following the proposal of Martinetz et al. [1993] the parameters $\epsilon_i^{\text{in}} = 0.5$, $\epsilon_f^{\text{in}} = 0.01$, $\lambda_i = 10$ and $\lambda_f = 0.5$ were used to reach a good adaption and convergence of the nodes to the data. As for SOMs in Equation (2.3), ϵ^{in} is decaying from ϵ_i^{in} to ϵ_f^{in} for $t \leq t_{max}$ and kept constant $\epsilon^{\text{in}} = \epsilon_f^{\text{in}} = 0.01$, if $t_{max} \leq t \leq 2 \cdot t_{max}$. λ decays from accordingly from an initial value λ_i to λ_f:

$$\lambda(t) = \lambda_i \cdot \left(\frac{\lambda_f}{\lambda_i}\right)^{\frac{t}{t_{max}}}.$$

The simulation sequence of a NG for a ring-shaped uniform probability distribution is shown in Figure 2.2.

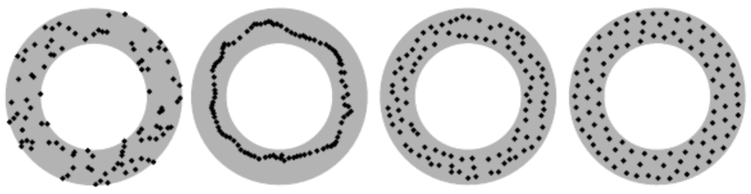


Figure 2.2: Neural Gas simulation sequence for a ring-shaped uniform probability distribution. Image is taken from Fritzke [1997] for 0, 2500, 10000 and 40000 training epochs.

Vector quantization with GNG GNG includes a growth process of successively inserting nodes according to local error measures. Starting initially with two nodes, an edge is initialized connecting the winner node and the second nearest node. The edges between the nodes define a direct topological neighborhood $\mathcal{N}$ of the nodes. Iteratively, the prototype vectors $\mathbf{w}_\kappa^{\text{in}}$ of the winner node, according to Equation (2.2), and its direct topological neighbors are updated by fractions ϵ^b and ϵ^n respectively.

$$\begin{aligned} \Delta \mathbf{w}_\kappa^{\text{in}} &= \epsilon^b \cdot \left(\mathbf{x}_n - \mathbf{w}_\kappa^{\text{in}}\right) \\ \Delta \mathbf{w}_i^{\text{in}} &= \epsilon^n \cdot \left(\mathbf{x}_n - \mathbf{w}_i^{\text{in}}\right) \ \forall i \in \mathcal{N}_\kappa. \end{aligned}$$

[1]The parameter λ is used here for traditional reasons. This is a different λ than in NCL 3.3.2 on page 52. For sake of clarity, it will be used as long as it is unambiguous in context.

Edges older than a certain number of training steps a_{max} are removed. For every node, a local error variable E_i is hold to store the squared distance between input sample $\mathbf{x}_n$ and the best-matching prototype vector $\mathbf{w}^{\text{in}}_{\kappa}$. A new node r is generated after a predetermined number of learning steps τ as shown below in Figure 2.3. The node q is the one with the maximum accumulated error after τ steps. The new node r is inserted and interpolated between the node q and the node p with maximum accumulated error in the direct neighborhood of q. The squared distance of the new inserted node r is then interpolated and reduced by a factor of ζ:

$$\begin{aligned} \mathbf{w}^{\text{in}}_r &= \frac{(\mathbf{w}^{\text{in}}_q + \mathbf{w}^{\text{in}}_p)}{2} \\ E_r &= \|\mathbf{x}_n - \mathbf{w}^{\text{in}}_r\|^2 = (1-\zeta)\frac{(E_q - E_p)}{2} \end{aligned}$$

In addition, the edge between q and p is replaced by the two edges connecting r with q and p. The error variables of all nodes are reduced by a factor η:

$$E_i = (1-\eta)E_i.$$

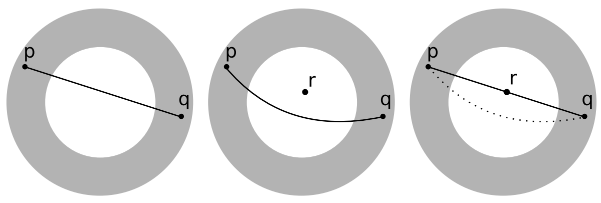


Figure 2.3: Growing Neural Gas: Insertion of a new node r in GNG network

This process is repeated until a stopping criterion is fulfilled. The parameters as proposed by Fritzke [1994] are adopted to $\epsilon^b = 0.1$, $\epsilon^n = 0.001$, $\zeta = 0.5$, $\eta = 0.0005$, deletion of edges after a number $a_{max} = 88$ of training steps and $\tau = 600$ constant over the number of learning epochs $t = 40$.

The images of the simulation sequences for a ring-shaped probability distribution as given for GNG in Figure 2.4 are taken from Fritzke [1997]. This work is recommended to the interested reader for a comprehensive and illustrative comparison and discussion of the before-mentioned VQ-based learning methods.

Vector quantization with FCL As in fuzzy logic, one assumes "soft" partitions instead of hard partitions. Hence, each sample has a certain degree of membership belonging to each node or assigned cluster. The degree of membership is formalized as a matrix $\mathbf{U} \in \mathbb{R}^{N \times n_l}$ with N rows and n_l columns. Every entry u_{ni} of $\mathbf{U}$ assigns each sample

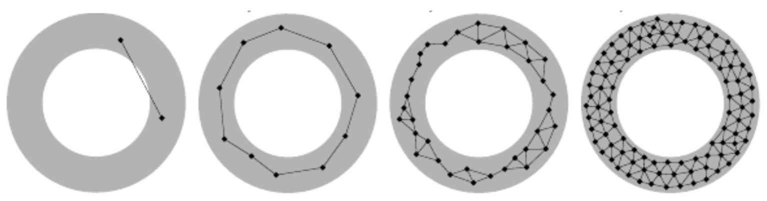


Figure 2.4: Growing Neural Gas simulation sequence for a ring-shaped uniform probability distribution. Image is taken from Fritzke [1997] for 0, 2500, 10000 and 40000 training epochs.

$\mathbf{x}_n, n = 1, \ldots, N$ its membership to cluster $i \in (1, \ldots, n_l)$. The degree of fuzzification for each sample equals 1 and furthermore all clusters are non-empty:

$$\sum_{i=1}^{n_l} u_{ni} \; = \; 1, \; \forall n \; \text{ and } \; \sum_{n=1}^{N} u_{ni} \; > \; 0, \; \forall i.$$

Similar to k-means [Macqueen, 1967; Hartigan and Wong, 1979], the nodes are pulled towards the centroid of a cluster, but the centroid is determined as the weighted mean of all samples according to their degree of membership. u_{ni} is thus inverse to the distance to the centroid $\mathbf{w}_i^{\text{in}}$ according to fuzzification parameter $\mu > 1$ and normalized:

$$\begin{aligned} \mathbf{w}_i^{\text{in}} &= \frac{\sum_{n=1}^{N} u_{ni}^{\mu} \mathbf{x}_n}{\sum_{n=1}^{N} u_{ni}^{\mu}} \\ u_{ni} &= \frac{1}{\displaystyle\sum_{j=1}^{n_l} \left(\frac{\|\mathbf{x}_n - \mathbf{w}_i^{\text{in}}\|}{\|\mathbf{x}_n - \mathbf{w}_j^{\text{in}}\|} \right)^{\frac{2}{\mu-1}}} \end{aligned}$$

The degree of membership is weighted by the exponent $\mu > 1$. Good results are found with $1 < \mu < 2.5$ [Pal et al., 1996].

Supervised training of local linear mappings After unsupervised adaptation and tessellation of the input space, the VQ-based approaches, for example the SOM, are extended to a LLM [Ritter, 1991] by an additional supervised trained layer of nodes, such that each node consists of a triple $\mathbf{v}_i = (\mathbf{w}_i^{\text{in}}, \mathbf{w}_i^{\text{out}}, \mathbf{A}_i)$. The vectors $\mathbf{w}_i^{\text{out}} \in \mathbb{R}^{d_{\text{out}}}$ approximate the distribution of the target values $\mathbf{y} \in \mathbb{R}^{d_{\text{out}}}$. The matrices $\mathbf{A}_i \in \mathbb{R}^{d_{\text{in}} \times d_{\text{out}}}$ are locally trained linear maps from the input to the output space. An input sample $\mathbf{x}_n$ is mapped to an output by the corresponding local expert $\mathbf{v}_\kappa$

according to the winner-takes-all (WTA)-rule:

$$\begin{aligned}
\mathcal{C}(\mathbf{x}_n) &= \mathbf{w}_\kappa^{\text{out}} + \mathbf{A}_\kappa^T \left(\mathbf{x}_n - \mathbf{w}_\kappa^{\text{in}}\right), \\
\Delta \mathbf{w}_i^{\text{out}}(\mathbf{x}_n) &= \epsilon^{\text{out}} \cdot h_{i\sigma} \cdot (y_n - \mathcal{C}(\mathbf{x}_n)) && (2.4) \\
\Delta \mathbf{A}_i(\mathbf{x}_n) &= \epsilon^{\text{A}} \cdot h_{i\sigma} \cdot (y_n - \mathcal{C}(\mathbf{x}_n)) \cdot \tfrac{(\mathbf{x}_n - \mathbf{w}_i^{\text{in}})^T}{\|\mathbf{x}_n - \mathbf{w}_i^{\text{in}}\|^2}. && (2.5)
\end{aligned}$$

The weights $\mathbf{w}_i^{\text{out}}$ and the linear maps $\mathbf{A}_i$ are changed iteratively by the gradient descent learning rules given by Equations (2.4),(2.5). The learning step widths $\epsilon^{\text{in}}, \epsilon^{\text{A}}, \epsilon^{\text{out}} \in [0;1]$ for updating neighbors and σ are decreased during training.

Good learning results are achieved with $\epsilon_i^{\text{out}} = 0.3$, $\epsilon_i^{\text{A}} = 0.3$, both decreasing exponentially over the number of learning epochs to a final value of $\epsilon_f^{\text{out}} = \epsilon_f^{\text{A}} = 0.01$ analogously to the unsupervised training and keeping $\epsilon_f^{\text{out}} = \epsilon_f^{\text{A}} = 0.01$ fixed for the rest of the learning process ($t_{max} \leq t \leq 2 \cdot t_{max}$).

A schematic representation of the LLM is shown in Figure 2.5.

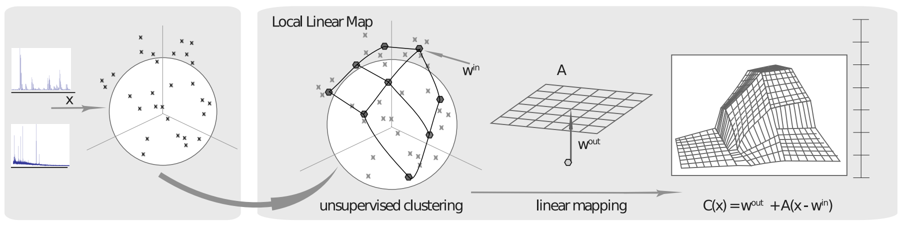


Figure 2.5: Schematic representation of SOM and supervised trained linear mappings to output space by LLM. The prototype vectors approximate to the input data while unsupervised clustering. Subsequently, the local linear mappings from input to output space are trained.

The concept of approximating nonlinear functions by fitting simple models to localized subsets of the data is related to other regression approaches like Locally-Weighted Regression (LOESS) [Cleveland and Devlin, 1988] and to radial basis functions (RBF) [Millington and Baker, 1990]. Hastie et al. [2001] demonstrated the usefulness of locally linear function fitting as well.

In the remainder, the type of VQ the LLM is based on is indicated by LLM_{SOM}, LLM_{NG}, LLM_{GNG} and LLM_{FCL}.

2.3.2 ν-Support Vector Machine

Support Vector Machines (SVMs) are a class of learning algorithms that are designed to implicitly transfer input feature vectors into a high-dimensional space and calculate the optimal linear separating hyperplane [Vapnik, 1995]. For a classification task, SVMs minimize the generalization error by maximizing the "margin" as the largest possible distance between the hyperplane separating the samples on either side of the two induced classes. The non-trivial problem of finding an optimal separating hyperplane is illustrated in Figure 2.6. In the case of linear separable data, samples

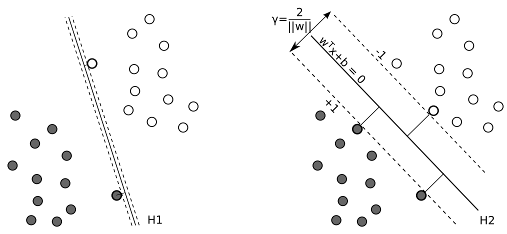


Figure 2.6: Two classes separating hyperplanes H1 and H2 for linear separable data. The dots are samples of two classes as indicated by the color (black/white). As shown in this example, finding a linear separating hyperplane may be a trivial task, but far from being uniform. Both hyperplanes, H1 as well as H2, denoted as black solid lines are optimal in that sense that the two classes are separated well. Only H2 maximizes the margin, i. e. the largest possible distance between the hyperplane separating the samples on either side. The support vectors lying at the border of the margin are marked bold and describe the hyperplane.

are classified according to which side of the separating hyperplane $\mathbf{w}^T\mathbf{x} + b = 0$ they lie, i. e. the decision is done by $f_{\mathbf{w},b}(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}^T\mathbf{x} + b)$, where $\mathbf{w}$ is a vector of weights and b the intercept. Its solution to the optimum separating hyperplane can be found by solving a convex quadratic programming problem:

$$\begin{aligned} \text{minimize} \quad & \tfrac{1}{2}\|\mathbf{w}\|^2 \\ \text{subject to} \quad & y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1,\ n = 1, \ldots, N \end{aligned}$$

The solution $\hat{\mathbf{w}}$ is completely described as a linear combination of a subset of input training instances $\mathbf{x}_n$, which lie at the border of the margin and are called the *Support Vectors* (SVs). Hence, the model complexity of an SVM is independent of the dimensionality of training data.

In some cases, the convex optimization problem is not feasible or one wants to allow for some relaxation of errors. This is addressed by the "soft margin" loss function introducing slack variables ξ_n and leading to the formulation:

$$\begin{aligned} \text{minimize} \quad & \tfrac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N}\xi_n \\ \text{subject to} \quad & y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1 - \xi_n,\ n = 1, \ldots, N \\ \text{and} \quad & \xi_n \geq 0 \end{aligned}$$

C is a regularization parameter controlling the trade-off between the size of the margin and the tolerance for accounting errors. This optimization problem can be solved in its dual formulation by quadratic programming, such that a classification can be performed by:

$$\begin{aligned} f(\mathbf{x}) &= \operatorname{sgn}\left(\langle \mathbf{w}, \mathbf{x} \rangle + b\right) \\ &= \operatorname{sgn}\left(\sum_{n=1}^{N} \pi_n y_n \langle \mathbf{x}_n, \mathbf{x} \rangle + b\right) \\ \text{where } \mathbf{w} &= \sum_{n=1}^{N} \pi_n y_n \mathbf{x}_n \end{aligned}$$

where sgn indicates the sign function and $< .,. >$ the dot product.

Nonlinear classification is achieved by applying the so called "kernel trick" [Aizerman et al., 1964]: The training samples are transformed by a map $\Phi : \mathbb{R}^{d_{\text{in}}} \rightarrow \mathcal{F}$ into a high- (possibly infinite) dimensional feature space, where a linear separation corresponds to a nonlinear separation in the original input space. For any training data, this separation is possible as long as the transformed feature space is of sufficient dimensionality. The algorithm only depends on the dot products $\langle \mathbf{w}, \mathbf{x} \rangle$ in $\mathcal{F}$. Without explicitly calculating nor knowing the mapping Φ itself, the product is replaced by a class of functions, the *kernel functions*, such that $K(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}')$. There exist a broad variety of kernels, which fulfill the conditions for kernels as characterized by Mercer [1909]. Examples of popular kernels are:

$$\begin{aligned} K(\mathbf{x}, \mathbf{x}') &= \langle \mathbf{x}, \mathbf{x}' \rangle^d && \text{(Polynomial kernel)} \\ K(\mathbf{x}, \mathbf{x}') &= \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right) && \text{(Gaussian kernel)} \\ K(\mathbf{x}, \mathbf{x}') &= \exp\left(-\gamma_{\text{RBF}} \|\mathbf{x} - \mathbf{x}'\|^2\right) && \text{(Radial-Basis Functions kernel)} \\ K(\mathbf{x}, \mathbf{x}') &= \tanh(a \langle \mathbf{x}, \mathbf{x}' \rangle + c), a > 0, c < 0 && \text{(Sigmoid kernel)} \end{aligned}$$

Support Vector Regression (SVR) Similar to the "soft margin" loss function, Vapnik [1995] introduced the ε-SV regression, which corresponds to the ε-insensitive loss function:

$$|y - f(\mathbf{x})|_\varepsilon = |\xi|_\varepsilon = \begin{cases} 0 & \text{if } |\xi| \leq \varepsilon \\ |\xi| - \varepsilon & \text{otherwise} \end{cases}$$

This way, the function $f(\mathbf{x})$ is approximated by allowing errors smaller than ε and the function is as flat as possible [Smola and Schölkopf, 2003]. Figure 2.7 shows an example of an ε-insensitive SVR. The prediction performance may crucially depend on the choice of ε. Schölkopf et al. [1999] proposed a modification of ε-SVR that automatically

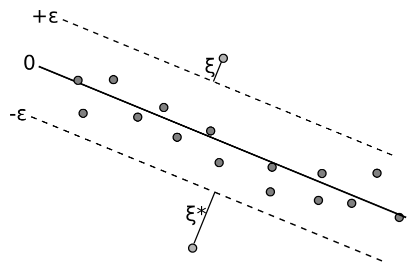


Figure 2.7: Example shows an ε-insensitive SVR. Errors which are smaller than ε are ignored, i. e. samples that lie inside the tube with width ε around the approximated linear function. Samples that are outside the tube, have errors of $|\xi| - \varepsilon$.

minimizes ε such that at most a fraction of instances lie outside the tube with radius ε, namely the ν-SVR. By including ε as a variable of the optimization problem, the tube is allowed to adapt automatically to the data. The primal optimization problem is reformulated with an extra term for $\nu \geq 0, C \geq 0$:

$$\begin{aligned} \text{minimize} \quad & \tfrac{1}{2}\|\mathbf{w}\|^2 + C \cdot \left(\nu\varepsilon + \tfrac{1}{N}\sum_{n=1}^{N}(\xi_n + \xi_n^*) \right) \\ \text{subject to} \quad & (\mathbf{w}^T\mathbf{x}_n + b) - y_n \leq \varepsilon + \xi_n \\ & y_n - (\mathbf{w}^T\mathbf{x}_n + b) \leq \varepsilon + \xi_n^* \\ & \xi_n^* \geq 0, \varepsilon \geq 0. \end{aligned}$$

Hence, with $0 \leq \nu \leq 1$ the number of SVs can be pre-specified, as the ν is an upper bound on the fraction of errors as well as a lower bound on the fraction of SVs. It offers good robustness to outliers, for it is a generalization of an estimator for the mean of a random variable, which throws away the largest and smallest examples (a fraction of at most $\nu/2$ of either category). It estimates the mean by taking the average of the two extremal ones of the remaining examples [Schölkopf et al., 1999].

In this work, the evaluations of SVR are grounded on the package `e1071` [Dimitriadou et al., 2009] available for R [R Development Core Team, 2008] used as interface to the widely used SVM library `libsvm` [Chang and Lin, 2001].

2.3.3 Evaluation

About 10% of the centered and normalized data are used for validation and are put aside. The remaining dataset is used to train the LLM and to find the best parameter set using 10-fold Cross-Validation (CV). Therefore, the remaining dataset is split into ten portions and one set is used for testing performance of the selected model. It was ensured that peptides from one spectrum as well as peptides occurring in more than one spectrum are found in only one of the portions.

Model selection A grid search over the parameter space $P = \left(n_l, \epsilon^{\mathrm{A}}\right)$ is performed to determine the optimal parameters for learning. The remaining learning parameters for the LLM are set to initial values $\epsilon^{\mathrm{out}} = 0.3, \epsilon^{\mathrm{in}} = 0.5$ decreasing over training steps. A 10-fold-CV is done for each parameter set. For every point in the parameter space the prediction accuracy for every training/test set is determined by the squared Pearson-correlation coefficient r^2 and the root mean square error (RMSE) of the test set. The choice of the best parameter set is made by the best mean r^2 over all 10 test sets while training the learning algorithm.

Pearson's correlation coefficient The Pearson correlation coefficient $(R_{YY'})$ is defined between two random variables Y and Y' measuring similarity and strength of the linear relationship of two series of samples.

$$R_{YY'} = \frac{cov(Y, Y')}{\sqrt{var(Y)var(Y')}}$$

where *cov* is the covariance and *var* the variance. $R_{YY'}$ is estimated by:

$$\begin{aligned}\mathrm{r} &= \frac{\sum_i (Y_i - \overline{Y})(Y'_i - \overline{Y'})}{\sqrt{\sum_i (Y_i - \overline{Y})^2 \sum_i (Y'_i - \overline{Y'})^2}} \\ &= \frac{1}{N-1} \sum_{i=1}^{N} \frac{(Y_i - \overline{Y})(Y'_i - \overline{Y'})}{s_Y \cdot s_{Y'}}\end{aligned}$$

where $\overline{Y}$ or $\overline{Y'}$ denotes the mean of Y and Y',

$$\text{and } s_Y = \frac{1}{N-1} \sum_{i=1}^{N} (Y_i - \overline{Y})^2, \quad s_{Y'} = \frac{1}{N-1} \sum_{i=1}^{N} (Y'_i - \overline{Y'})^2$$

as a scaled version of covariance between Y and Y'. For the correlation coefficient r applies $\mathrm{r}^2 \leq 1$. With $\mathrm{r} = 1$ $(\mathrm{r} = -1)$ the random variables are perfectly (negative) correlated, and with r next to 0, the series of samples Y and Y' indicate a weak linear relationship and are orthogonal to each other.

Root Mean Squared Error (RMSE) The RMSE is an error measure for the mean squared deviation of samples from the diagonal fit:

$$RMSE(\mathbf{y}, \mathbf{y}') = \sqrt{\frac{1}{N} \sum_{n=1}^{N} (y_n - y'_n)^2}$$

Model assessment The final model with the optimal parameters is chosen. To validate its prediction (generalization) error on new data, the validation set is used, which has not taken part in training.

2.3.4 Issues in Data Handling Prior to Learning

In many application areas, obstacles have to be overcome with regard to uncertain samples, missing values, high variance in the features or samples, etc. Jointly, these may have a dramatical impact on the prediction performance and put new challenges in the application of machine learning algorithms. In the context of the prediction of peak intensities, the major issues can be specified arising mainly from the imperfect identification routines.

Missing values The identification process in preprocessing is followed by the extraction of peptides, where the best matching proteins constituting to the extracted peptides are identified. Under this presumption of having identified each protein correctly, the resulting list of peptides is matched against theoretically derived peptides. Hence, there exist non-observed peptides in spectra where no intensity value is extractable. Missing values in features or target values is an often occurring problem in machine learning. One possible solution would be to discard the affected samples, thereby reducing the size of the training set. But especially for small datasets, this limits the search space of the predictor in finding the true hypothesis. Depending on context, other strategies are possible like replacing the missing values with zero or according to some aggregation. In this study, both approaches are examined: First, all of the non-observed peptides are discarded and second, the missing peptides are incorporated solely in the unsupervised vector quantization step.

Multiple recordings/samples Most of the peptides in the data set occur multiple times in different spectra with different intensity values, which means the real-valued samples $(\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j)$ in the input space, $\exists i, j,\ \text{s.t.}\ \mathbf{x}_i = \mathbf{x}_j : y_i \neq y_j$. In the remainder, I refer to them as "duplicates". Due to limitation of the training data, outliers (potential noisy peptides) are eliminated by mapping each peptide to one unique value, the α-trimmed mean of all intensities per distinct peptide with $\alpha = 0.25$. The α-trimmed mean is defined as the mean of the center $100\% - 2\alpha \cdot 100\%$ of an ordered list. In the case of less than four peptides in the list, a simple mean is taken. This way, a more reliable approximation is hopefully achieved.

Under-sampled instances Peptides, which do not carry much information or being noisy, should not be regarded in the learning process or at least should be less weighted. At about 50% of the peptides are represented only once in the data. These are referred to as "singles". In fact, their intensities show a wide spread distribution and may be not valuable regarding signal-to-noise-ratio. Hence, one might come up with another competing strategy and regard the singles as samples being under-sampled. The LLM-predictor is capable to perform an alternative mapping of the duplicate samples for the α-trimmed mean. It seems to be a reasonable strategy to take the duplicates more into account for the learning process.

Variance of features As features typically differ in their variance, one input with a large variance will have a higher impact on the resulting predictor compared to others with low variance. When distance functions, e. g. Euclidean distance, are applied, it is essential to relate the variance of features to their relevance. The features $d = 1, \dots, d_{\text{in}}$ should be therefore rescaled and standardized to zero mean $\overline{\mathbf{x}}_d$ and standard deviation $\sigma(\mathbf{x}_d)$ of one:

$$\mathbf{x}_d^{std} = \frac{\mathbf{x}_d - \overline{\mathbf{x}}_d}{\sigma(\mathbf{x}_d)}$$

2.3.5 Results

A series of experiments are performed in order to measure the performance of the previously described regression architectures under various aspects. First, the prediction capability is assessed to test how successful the prediction of peak intensities with machine learning methods can be performed when supplying the 18-dimensional selected subset of features Heur. The four VQ-based approaches of LLM-type, LLM_{GNG}, LLM_{NG}, LLM_{SOM} and LLM_{FCL}, are applied in direct comparison. The performance is assessed also for the high-dimensional dataset with 531 features. The ν-SVR and the linear least squares model (LM) (see Section 2.4.1.1 on page 38) are used for comparison purposes.

The following results are evaluated for two subsets of the entire data set A supplied with variable set Heur. The first set, denoted by Heur_{TM}, contains the peptides mapped by α-trimmed mean, whereas the second set, Heur_{Dup}, contains all peptides including these ones, which occur in multiple spectra, referred to as duplicates. The validation set of Heur_{TM} consists of 44 items and for Heur_{Dup} of 73 items. One advantage of the LLM is that is capable of dealing with the duplicates straightforward.

2.3.5.1 Peptide Prototyping

A display of the prototype vectors resulting from the NG training allows a profiling of the peptides. In Figure 2.8, a resulting parallel coordinates plot for multivariate data is used to plot the six prototype vectors in case of Heur_{Dup}. A set of parallel axes is drawn for each feature, where for each feature the range of values (from min to max, from bottom to top respectively) covered by the prototypes is shown.

The correlation of input space is reflected in the prototype distribution of Figure 2.8. We can see that some of the features ('VASM830103','WILM950102','FINA770101',

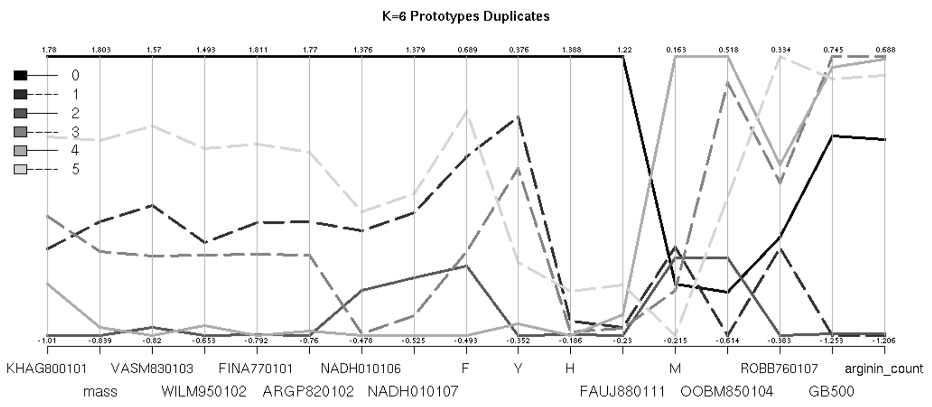


Figure 2.8: Parallel coordinates plot for six prototypes in case of Heur_{Dup} dataset. For every feature the range of values covered by the prototypes is shown.

'ARGP820102') show a correlation to the mass, while no such tendency can be observed for the other features. 'OOBM850104', 'ROBB760107' and 'M' show the least similarity to any other feature. If we look at prototype 0 and 4, it can be seen that they cover contrary outmost areas in data space in almost all features except the three mentioned ones. The prototypes 2 and 4 are near to each other for almost all features, except for the three mentioned features and 'arginin_count' as well as 'GB500' (the last two being highly correlated), where they split up to almost the extremes. Similar behavior can be observed for the prototypes 1 and 3.

The six prototypes take three to five levels for each feature, two or more prototypes sharing the same region. 'OOBM850104' and 'KHAG800101' show the most even distribution of prototypes. Thus the prototypes share ranges in certain features and split up for others, achieving a separation in data space: For a data point that is similar to two prototypes sharing their space for a set of features, other features decide to which prototype it is assigned to.

2.3.5.2 Predicting Peaks' Intensities

For comparison purposes, the prediction capabilities of the LLM_{NG} for the two datasets Heur_{TM} and Heur_{Dup} are evaluated as described in Section 2.3.3 on page 23. A grid search is performed over all parameters $P = \left(n_l, \epsilon^{\text{A}}\right)$ and that parameter set is chosen, which yields the best mean r^2 of the training/test sets. For the exact results of r^2 and RMSE see Table 2.2. From the results it is clear that peak intensities can be characterized and predicted by the use of the heuristically selected feature set with high prediction accuracy. A minimum RMSE on the *test set* is achieved with $n_l = 3$ nodes

		Test				Valid	
Dataset	$\mathbf{n_l}$	r^2	ave(r^2)	$\sigma^2(r^2)$	**RMSE**	r^2	**RMSE**
	3	0.6773	**0.4341**	0.1811	**1.14**	0.1611	1.748
$\mathrm{Heur}_{\mathrm{TM}}$	6	0.6318	0.4391	0.1421	1.173	0.1148	1.838
	10	0.6003	0.3845	0.1881	1.19	0.157	1.774
	3	0.647	**0.433**	0.1462	1.429	0.1972	1.4516
$\mathrm{Heur}_{\mathrm{Dup}}$	6	0.6168	0.4286	0.1356	1.576	0.2195	**1.4294**
	10	0.5659	0.3599	0.1549	1.72	0.137	1.5169

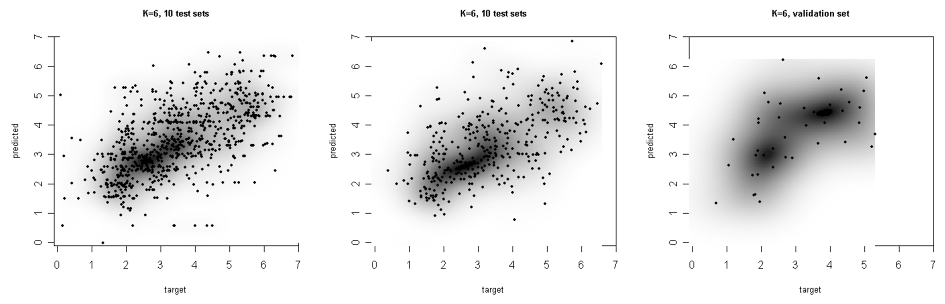
Table 2.2: Comparison of capabilities of the $\mathrm{LLM}_{\mathrm{NG}}$ for variable set Heur in predicting intensities for the studied datasets $\mathrm{Heur}_{\mathrm{TM}}$ and $\mathrm{Heur}_{\mathrm{Dup}}$. The evaluation was done for $K = 3, 6$ and 10 prototypes. The resulting performances are given by the best r^2, the mean (r^2) and $\sigma^2(r^2)$ of all 10 test sets and the corresponding error RMSE, as well as for the validation set.

for $\mathrm{Heur}_{\mathrm{TM}}$, while the RMSE on the validation set is minimized by a model trained on $\mathrm{Heur}_{\mathrm{Dup}}$ with $n_l = 6$ nodes. It can be observed that considering the entire data set including duplicates instead of a α-trimmed mean mapped data set yields higher correlations and a better generalization performance only on the small validation set. The issue of a meaningful evaluation of this small validation set is addressed later in the Discussion, see 2.3.6 on page 33. The LLM provides an alternative way of mapping the α-trimmed means by mapping each of the duplicate intensities to one unique target value. An example of the prediction of duplicates is given in Figure 2.9a.

2.3.5.3 Comparison Prediction Performance of SOM to Neural Gas

For a comparison of the prediction and generalization performance of $\mathrm{LLM}_{\mathrm{SOM}}$ and $\mathrm{LLM}_{\mathrm{NG}}$, the studied dataset $\mathrm{Heur}_{\mathrm{TM}}$ is evaluated. Due to the difference of topology of these learning methods, iteration is performed over a number of nodes specified by n_g. The SOMs topology is kept fixed as $n_l = n_g \times 2$, and its performance is compared to the one of NG with $n_l = n_g \cdot 2$ nodes.

In Figure 2.10 on page 30 the results of the prediction accuracy determined by r^2 and RMSE with $\mathrm{LLM}_{\mathrm{SOM}}$ and $\mathrm{LLM}_{\mathrm{NG}}$ for data set $\mathrm{Heur}_{\mathrm{TM}}$ are summed up. Both, $\mathrm{LLM}_{\mathrm{SOM}}$ and $\mathrm{LLM}_{\mathrm{NG}}$, yield a similar behavior with respect to the prediction performance. As the number of nodes increases ($n_l > 4$, where the optimum is reached), the clustering error for both learning paradigms decreases, while the prediction error and correlation for the ten test sets also decrease and increase for the validation set. Furthermore, it can be observed that though a worse clustering error of SOM-learning, the prediction error for the ten test sets as well as for validation is smaller than that of NG-learning and yields better prediction performance. The number of nodes n_l is a critical size due to over-fitting of the training data.



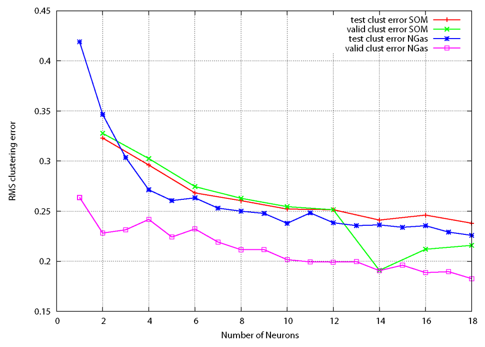
(a) NG, trained on Heur_{Dup} test set prediction (b) NG, trained on Heur_{TM} test set prediction (c) NG, trained on Heur_{TM} validation set prediction

Figure 2.9: Scatterplots of LLM_{NG} trained on dataset Heur with $n_l = 6$ nodes. The first two plots show the predicted 10 test sets, while for (a) the NG model was trained accounting duplicates, i. e. Heur_{Dup}. The second model in (b) was trained on Heur_{TM}, while for (c) the same model is used to predict the small validation set.

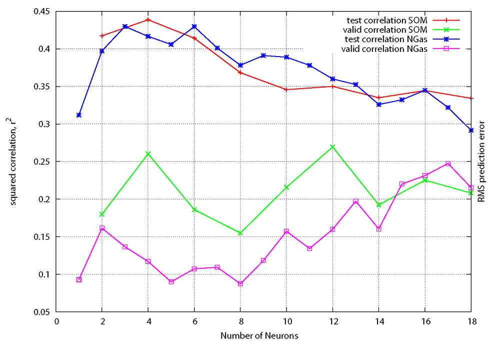
2.3.5.4 Subsampling of Peptides

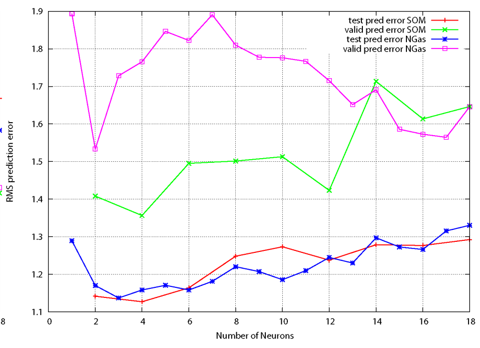
Up to now, the missing peptides, i. e. those for which no intensity value is extracted, have not been accounted for. The entire set of peptides is subsampled by excluding those missing peptides. By mapping the duplicate samples to one target value by the α-trimmed mean the size of the available training samples is yet further reduced. The LLM is capable of dealing with the duplicates straightforward, and, moreover, allows to incorporate those peptides with missing target values. For these peptides solely the vectorial description in the input space is available. To account these for, the training sets used to build the LLM models can be extended by the missing peptides in the unsupervised clustering step. In order to explore if the subsampling of training set size is beneficial, evaluations are performed over the training sets induced by the *inclusion of the missing* values as well as by the *inclusion of the duplicates*. Thus, training in this setup is performed on the entire set of extracted peptides *without subsampling*. The generalization performance of the derived predictors trained in this manner is compared to those trained on the α-trimmed mean target values (*with subsampling*). Furthermore, each of the resulting models is used to predict the α-trimmed mean target values in either case.

As an example, the resulting scatterplots are shown in Figure 2.11 on page 32. For the corresponding left figures the target values are plotted against the predicted values for the Heur as well as the AAindex dataset. These are compared to those models resulting from a training without subsampling of peptides. In both cases of feature



(a) Quantization error.





(b) Correlation of Prediction. (c) Prediction error.

Figure 2.10: Results of the prediction accuracy for $\mathrm{Heur}_{\mathrm{TM}}$. Iteration is done over $n_l = 1, \ldots, 18$ number of nodes, with $n_l = 2 \times n_g$ ($\mathrm{LLM}_{\mathrm{SOM}}$), and $n_l = 2 \cdot n_g$ ($\mathrm{LLM}_{\mathrm{NG}}$) nodes respectively. (a): For every evaluation the results of the mean performance of the test sets is plotted as well as the performance of the validation set. The best mean test correlation is yielded by a 2×2-$\mathrm{LLM}_{\mathrm{SOM}}$. While the prediction accuracy for the ten test sets decreases, the prediction accuracy for validation set increases proportional to the number of nodes. (b): It can be stated that the clustering error for SOM tends to be worse than for NGas. (c): The prediction error increases for $\mathrm{LLM}_{\mathrm{SOM}}$ and $\mathrm{LLM}_{\mathrm{NG}}$ in case of test sets to the same degree, whereas the prediction error of the validation set for the $\mathrm{LLM}_{\mathrm{SOM}}$ is much smaller than that for $\mathrm{LLM}_{\mathrm{NG}}$.

sets, though, the extension of the training sets leads to a deterioration in terms of generalization performance and correlation.

If trained on the duplicates, the LLM predictor models perform an alternative mapping of the duplicate samples for the α-trimmed mean. It seems to be reasonable to take the duplicates more into account for the learning process. To test to which extend the under-sampled peptides are of importance with regard to prediction performance, an additional factor to the learning algorithm is introduced

$$\epsilon^* = \epsilon \cdot f(peptide_{\text{occ}}), \quad \text{where} \quad \epsilon \in \left\{\epsilon^{\text{in}}, \epsilon^{\text{out}}, \epsilon^{\text{A}}\right\}$$

such that single occurring peptides are penalized and the duplicates are more heavily weighted, ensuring that not the whole learning process itself is slowed down. The function of the peptide occurrence is evaluated with regard to the parameters s_1, s_2, s_3, such that $f(peptide_{\text{occ}}, s_1, s_2, s_3) = -\exp(-peptide_{\text{occ}}/s_1)^{s_2} + s_3$. Several combinations of s_1, s_2, s_3 are tested and the best evaluated function of peptide occurrence is found with $s_1 = 0.3, s_2 = 0.25$ and $s_3 = 1.2$.

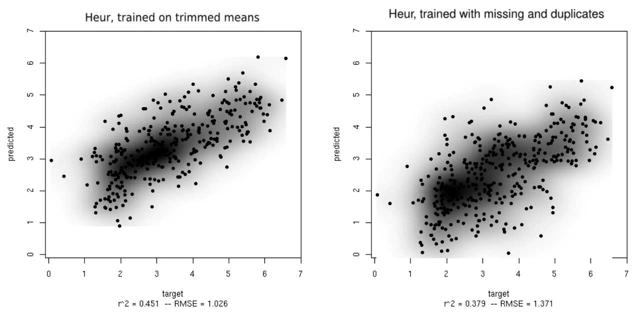
2.3.5.5 Comparison Prediction Performance of Feature Sets

The influence of the two feature sets Heur and AAindex is compared for the LLM_{GNG}, LLM_{NG}, LLM_{SOM} and LLM_{FCL} to the reference learning architectures ν-SVR as well as LM. In Table 2.3 the resulting generalization performances are listed for the two feature sets available. These are given with regard to the training based on the intensities mapped by α-trimmed mean as compared to the entire set of duplicates in case of the LLM-variants.

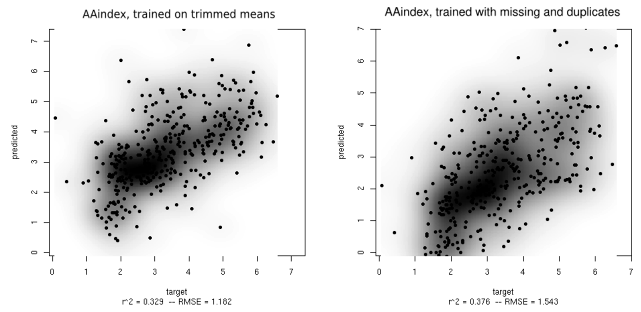
	LLM based on					
feature set	**GNG**	**NG**	**SOM**	**FCL**	**ν-SVR**	**LM**
$\text{AAindex}_{\text{TM}}$	12.97	1.2	1.16	1.91	**1.03**	1.11
Heur_{TM}	1.65	1.06	**1.03**	1.10	1.06	1.16
$\text{AAindex}_{\text{Dup}}$	23.26	1.19	1.18	1.48	**1.09**	1.15
Heur_{Dup}	3.57	1.41	1.40	1.66	**1.07**	1.18

Table 2.3: Prediction accuracy in terms of testing RMSE of applied regression architectures LLM, ν-SVR and LM - trained on AAindex feature space (531 dim). Results are also given for the Heur feature space (18 dim).

It can be observed that the applied learning architectures differ strongly in their performance. This trade-off may be due to two main reasons: First of all, the number of variables (531) even exceeds the number of available peptides in the dataset (372).



(a) Heur dataset trained with (left) and without (right) subsampling.



(b) AAindex dataset trained with (left) and without (right) subsampling.

Figure 2.11: Scatterplots of LLM prediction on (a) Heur and (b) AAindex dataset. Notice that prediction is done over all ten testing sets for 10-fold CV. The left column of figures corresponds to the prediction of models trained *with* subsampled peptides by the α-trimmed mean. The figures on the right show the predictions of models trained *without* subsampling peptides and incorporating the entire set of samples including missing values and the duplicates.

Best prediction and generalization performance is observed for the SVR, while the LLM shows only a slight worse accuracy regarding Heur feature space. The linear model (LM) shows a clear over-fitting and lacks of the generalization on new peptides. A comparison between the peptide and feature representations is summarized by the following order according to the RMSE:

$$\text{Heur}_{\text{TM}} \; < \; \text{AAindex}_{\text{TM}} \; < \; \text{AAindex}_{\text{Dup}} \; < \; \text{Heur}_{\text{Dup}}$$

These observations suggest that the relationship between the peptide feature vectors and the intensity target values is apparently nonlinear.

2.3.6 Discussion

The results show that the LLM_{SOM}-approach combining data mining and supervised learning yields similar results in the prediction accuracy compared to the ν-SVR. Thus, the LLM benefits from supplying SOMs compared to NG due to their topology preserving characteristic. A low number of local experts and locally trained linear mappings facilitate to successfully model the non-linear relationship between the peptides and their peak heights. The SOM topology characteristic allows for a two-dimensional representation of the input space. The visual inspection of the prototypes reveals that the peptides can be grouped around a set of approximately six profiles. Those seem to have individual mappings to the peak intensity, which can be discussed with biochemical experts.

From the results it is clear that the peptide intensities can be modeled and predicted by the use of the heuristically selected subset of features with a high prediction accuracy. It can be observed that considering the entire data set including the duplicates instead of the α-trimmed mean mapped dataset yields lower correlations and a worse generalization performance. The alternative mapping for the α-trimmed mean by the prototypes is non-beneficially distorted due to the noise with high variance and outliers. This is reflected for both features spaces, $\text{AAindex}_{\text{Dup}}$ and Heur_{Dup} including the duplicates, in a decrease in accuracy, while the former has a slight advantage. A mapping of the non-linear relationship from the peptides' numerical representations to the target values presupposes some characteristics or variables, which seem to be underrepresented in the Heur description.

A direct quantitative comparison of the prediction performance of the duplicates and the α-trimmed mean predictor models in terms of test RMSE is non-feasible. As the duplicate samples are mapped to one unique value by the α-trimmed mean procedure, the duplicate entries in target values are still accounted for when evaluating errors of models used to predict the duplicates. Due to the small size of samples available, there is no separated test set applicable for all variations of the peak intensity dataset. Therefore, the performance of models is assessed via cross-validation. The duplicates' errors in predictions intrinsically exhibit a significant higher amount of deviations of the target values as the α-trimmed means. Only a qualitative comparison is allowed here but no quantitative, as this is a critical task due to the incoherency of the results when evaluating the errors. The correlations between the peptide intensity values and their α -trimmed means, as depicted in Figure 2.12, may be interpreted as an "upper bound" for the accuracy achievable.

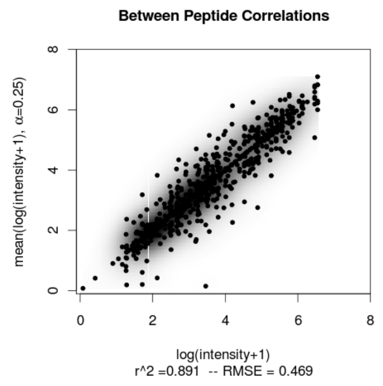


Figure 2.12: Between peptide correlations: Scatterplot of duplicates ($\log(\mathit{intensity}+1)$) against trimmed mean target values ($\mathit{mean}(\log(\mathit{intensity}+1), \alpha = 0.25)$).

The investigation of the treatment of single occurring peptides, which amount 50% of the entire data, did not issue a precise statement. Therefore a comparison to data sets with more multiple spectra per protein would be helpful.

An assessement in terms of generalization performance of models used to predict the validation set might not be meaningful since it contains 43 peptide samples with even high variance. The fact that some test sets performance is worse (especially for dataset $\mathrm{Heur}_{\mathrm{TM}}$) than the performance of the chosen validation set, can be explained by the static choice of the set. The different portions of $\mathrm{Heur}_{\mathrm{TM}}$ set yield a wide spread of correlation, resulting in high standard deviation of r^2 over all the portions. There are test sets that seem significantly worse in prediction performance over all training sets. There exists a positive correlation to the number of test set samples. To this end, I proceed with an exclusion of the validation set and concentrate on assessing the model generalization performance by averaging over the test sets' performance.

2.3.7 Conclusions

The Local Linear Map based on SOM-VQ is considered for prediction of peak intensities with comparable results [Scherbart et al., 2007b] to those obtained by ν-Support

Vector Regression (SVR) [Timm et al., 2006]. In addition, Neural Gas clustering instead of SOM in the input space is applied for comparison [Scherbart et al., 2007a] providing a basis for peptide prototyping and visualization. Other than for example SVR, the LLM can be used for data mining once adapted in a straightforward manner. Both architectures have been proposed to model the non-linear relationship between peptide and peak intensities. This method is still in an early phase: A proof of concept has been conducted and published in [Timm et al., 2008]. Peak intensities can be predicted with significant correlations, but application tests are yet to come.

It turns out that the performances strongly depend on the choice of the feature space. The non-linear relationship between peptides and their peak intensities is modeled with a minimum RMSE of 1.03 in case of the 18-dimensional selected subset of features chosen in an ad-hoc forward feature selection. Due to the known curse of dimensionality and to the VQ-based approach acting in Euclidean space, the LLM shows difficulties in high-dimensional spaces. For the high-dimensional dataset AAindex, the ν-SVR is the only learning architecture, which is capable to extract useful information to build successful predictor models with high prediction performance.

The priority objective is to decide, which type of feature set has greatest potential therewith in a general learner setup. I believe that, based on the high-dimensional feature space named AAindex, an improved prediction is possible even for the $\mathrm{LLM}_{\mathrm{SOM}}$. Consequently, I concentrate in the remaining of this thesis on the AAindex features and the target intensity values calculated by the α-trimmed mean.

2.4 Improved Peak Intensity Prediction by Adaptive Feature Weighting

Predictors may strongly benefit from a reduction of features, thereby defying the curse of dimensionality, offering reduced time and space complexity as well as an improved performance. Conversely, features being highly redundant and correlated to others, can considerably degrade the process of learning. The task of deciding which features to use to build successful models is one of the central problems in machine learning [Blum and Langley, 1997]. The selection of relevant features contrasts with the selection of useful variables[2], as many redundant, but relevant variables, may be excluded [Guyon and Elisseeff, 2003]. Blum and Langley [1997] give a good overview and discuss the various definitions and notions of usefulness and relevance.

To explore the potential from focusing on relevant features that contribute to the non-linear peptide/peak intensity relation, a regression based combination of feature weightings and a linear predictor is proposed in [Scherbart et al., 2008]. The regression approach of LLM is extended to provide simpler models and better generalization if trained on the high dimensional dataset AAindex.

[2]The terms 'features' and 'variables' are used here without distinction and interchangeably.

The two-step regression approach includes an adaptive regression based feature weighting by four learning architectures individually, therewith four different model specific estimations of feature relevance are obtained. Subsequently, the estimated feature weightings of the d_{in} features are used for scaling the input vectors in a re-training step with an independent predictor, i. e. with the LLM. The performance when accounting for the assessed weighting of features and subsequent re-training is compared to an approach integrating weighting and filtering of features. I show that the overall performance is improved as compared to using the entire feature set. The LLM-regression benefits from a re-weighting of features according to the model specific estimation of feature relevance.

A schematic description of the proposed two-step regression architecture is depicted in Figure 2.13.

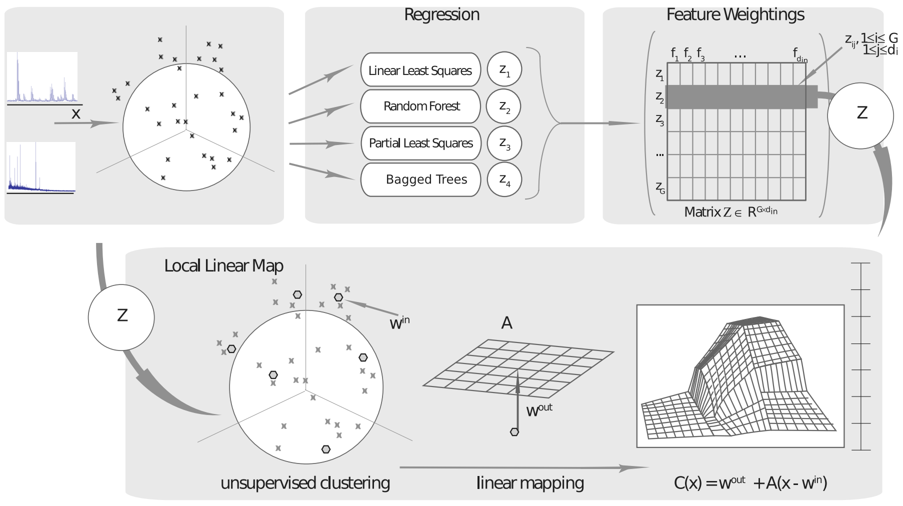


Figure 2.13: Graph depicting the proposed architecture schematically. Given the MS data, the proposed architecture is based on a two-step regression approach: Four different models (Linear Least Squares, Random Forest, Partial Least Squares, Bagged Trees) are derived to be evaluated according to their corresponding model specific metric for estimating the contribution of each feature. Each of the model specific estimations of feature relevance is integrated into a common data structure **Z** for the application of subsequent filtering of features. These scaled feature weightings **Z** are then used as input to form a new prediction model of LLM-type.

2.4.1 Assessing the Feature Relevance

Feature Selection covers the problem of selecting an appropriate subset of features, which are most likely to be relevant to predictor models. Classic feature selection steps include heuristic search (forward selection, backward elimination, stepwise selection) and successively adding or eliminating attributes by an adequate strategy. The approaches can be divided into filters and wrappers. Filter approaches use general characteristics (e. g. correlation) of the data provided to select a subset of features, independently of the chosen learner. Wrappers score subsets of features by a metric according to the estimated accuracy of a given learning machine.

To reflect feature relevance as a numerical value, an estimation of the feature relevance is done by four different regression models, which are trained on the entire feature space. Each of the four applied architectures comprises an internal model dependent metric as measure of quality. The feature weights are set proportionally to the assessed change in accuracy (via correlation, MSE), e. g. to the decrease in error when permuting the features. The greater the decrease in performance when leaving a certain feature out, the higher is the assigned degree of relevance of the feature. The model dependent metrics evaluation to assess the feature relevance is offered by the R package `caret` [Kuhn, 2008]. Each of the $G = 4$ estimations is interpreted as the feature relevance $\mathbf{Z} \in \mathbb{R}^{G \times d_{\text{in}}}$ as assessed by Linear Models, Partial Least Squares, Random Forests and Bagged Trees. The feature relevance is calculated according to:

Linear Models (LM) Absolute value of t-statistic for each model parameter [Kuhn, 2008].

Partial Least Squares (PLS) The feature relevance measure is based on the weighted sums of the absolute regression coefficients. The weights are a function of the reduction of the sums of squares across the number of PLS components and are computed separately for each outcome. Therefore, the contribution of the coefficients are weighted proportionally to the reduction in the sums of squares [Kuhn, 2008].

Random Forest (RF) For regression, the MSE is computed on the out-of-bag data (see 3.3.1.1 on page 51) for each tree, and then the same is computed after permuting a variable. The differences are averaged and normalized by the standard error. If the standard error is equal to 0 for a variable, the division is not done [Liaw and Wiener, 2002].

Bagged Trees (BT) The same methodology as a single tree is applied to all trees and the total relevance is returned.

2.4.1.1 Linear Least Squares

Linear least squares is the most widely used method for regression. A linear model is fit of the form

$$f(\mathbf{x}) \;=\; b_0 + \sum_{i=1}^{d_{\text{in}}} \mathbf{x}_i b_i \;=\; \mathbf{X}^T \mathbf{b}$$

where $\mathbf{x}_i$ are the variables, b_i are called the coefficients and b_0 the intercept. Written in matrix notation, $\mathbf{X}$ is here a $(N+1) \times d_{\text{in}}$ matrix with the first row filled up with ones, i. e. the column i of $\mathbf{X}$ corresponds to $(1, x_{1_j}, \ldots, x_{N_j})$. Linear Least Squares [Wilkinson and Rogers, 1973; Chambers and Hastie, 1992] finds a solution to the fit, i. e. coefficients b_i, $i = 0, \ldots, d_{\text{in}}$, such that the residual sum of squares (RSS) is minimized:

$$\begin{aligned} \arg\min_{\mathbf{b}} RSS \;&=\; \arg\min_{\mathbf{b}} \sum_{i=1}^{N} (y_i - f(\mathbf{x}_i))^2 \;=\; \arg\min_{\mathbf{b}} \|y - f(\mathbf{x})\|^2 \\ \text{where } \hat{\mathbf{b}} \;&=\; \left(\mathbf{X}^T\mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{y} \end{aligned}$$

2.4.1.2 Partial Least Squares

Partial Least Squares (PLS) was first introduced by Wold [1966]. PLS regression searches for a set of components (called *latent* vectors) that perform a simultaneous decomposition of $\mathbf{X}$ and $\mathbf{y}$ with the constraint, that these components explain as much as possible of the *covariance* between $\mathbf{X}$ and $\mathbf{y}$ [Abdi, 2007]. The input data matrix $\mathbf{X} \in \mathbb{R}^{N \times d_{\text{in}}}$ is decomposed as follows:

$$\mathbf{X} \;=\; \mathbf{Q}\mathbf{P}^T \text{ with } \mathbf{Q}^T\mathbf{Q} \;=\; \mathbf{I}$$

with $\mathbf{I}$ the identity matrix, $\mathbf{Q}$ the so called loading matrix and the columns of $\mathbf{Q}$ are the latent vectors. $\mathbf{y}$ is estimated as:

$$\hat{\mathbf{y}} \;=\; \mathbf{Q}\mathbf{B}\mathbf{C}^T$$

where $\mathbf{B}$ is a diagonal matrix corresponding to the “regression weights” and $\mathbf{C}$ is the “weight matrix” of the dependent variables. This way, with $\mathbf{Q}$ a generalization to principal component analysis is performed. In this work, the function `plsr` of the R package `pls` [Wehrens and Mevik, 2007] is used.

2.4.1.3 Random Forests

Breiman [2001] proposed Random Forests (RF) as a combination of decision tree predictors for classification as well as regression tasks. Random Forests are defined as “a collection of tree-structured classifiers $\{f(\mathbf{x}, \theta_m),\ m = 1, \ldots\}$ where the $\{\theta_m\}$ are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input $\mathbf{x}$" [Breiman, 2001].

A Random Forest is built by several random trees, each of them constructed independently. The nodes of each tree are split by the best presented variable of a randomly selected subset of variables. Typically, only with a third of the entire variables is randomly sampled for each node. RFs are explained and discussed in more details in 3.3.1.3 on page 52 in the context of Ensemble Learning (Chapter 3) and Bagging (3.3.1.1 on page 51). The evaluation of RFs is based on the implementation in the `randomForest` package by Liaw and Wiener [2002].

2.4.1.4 Bagged Trees

Bagged Trees (BT) are built by a bundle of decision trees similar to Random Forests. Bagging was proposed by Breiman [1996a] to get more stable trees. Each of the trees is constructed independently of randomly drawn samples of the size N, the so called bootstrapped samples, thereby approximately containing 63% of the original instances. The outputs of the trees are then aggregated by majority vote in case of categorical outputs (classification) or by simple averaging the numerical outputs in case of regression.

The decision trees in this function are computed using the implementation in the `rpart` R-package [Therneau et al., 2009].

2.4.2 Evaluation

Performance assessment is done similarly to the previous evaluations. In the remainder, the validation set is completely excluded from the evaluation due to the static choice of the set and in order to keep the results comparable and consistent.

Model selection A grid search over the parameter space is performed to determine the optimal parameters for learning. For every point in the parameter space the prediction accuracy for every training/test set is determined by the RMSE of the test sets over all portions of 10-fold cross-validation. From the set of input vectors, a randomly sampled subset of instances of appropriate size, e. g. 10%, is excluded from the training set and is used to built up the test set for each iteration. The choice of the best parameter set is made by the best mean performance over 20 iterations.

Feature Weighting The G final models LM, RF, PLS, BT are evaluated by the corresponding model specific metric for estimating the contribution of each feature. Each of these G methods contributes one vector or row $\mathbf{z}_g, g = 1, \dots, G$ to the entire matrix $\mathbf{Z}$ of feature weightings with d_{in} columns.

Filtering of Feature Weightings A filtering approach discards feature weightings below a certain threshold. The cut-off value is determined by the $c\%$ most relevant features, where $c \in \{95, 90, 70\}$. These features are chosen, which make up $c\%$ of the total sum of relevance (SoR), where the SoR is given as $SoR(g) = \sum_{l=1}^{d_{\text{in}}} \mathbf{z}_{gl}$. Having sorted each row $\mathbf{z}_g$ of $\mathbf{Z}$ in descending numerical order, these features $1 \ldots, j$ are selected, where j is determined by $\arg\min_j \sum_{l=1}^{j} \mathbf{z}_{gl} \geq 0.9 \cdot SoR(g)$.

Retraining of LLM model These feature weightings $\mathbf{z}_g$ (scaled up to be in range of $[0, 1]$) are then used to scale the input variables to $\tilde{\mathbf{x}} = \mathbf{z}_g \cdot \mathbf{x}$. A new model of LLM-type is built based on the optimal parameters determined in the model selection step with the input vectors $\tilde{\mathbf{x}}$.

2.4.3 Results

The prediction and generalization performance of the LLM predictor is compared when applying different approaches of weighting and filtering of the features. The degree of feature relevance $\mathbf{z}_g$ is estimated by the four regression architectures LM, PLS, RF and BT. As an example, the resulting assessed relevance of features is shown for AAindex data in Figure 2.14, when a selection of features is performed to account for the 25 most relevant features on average.

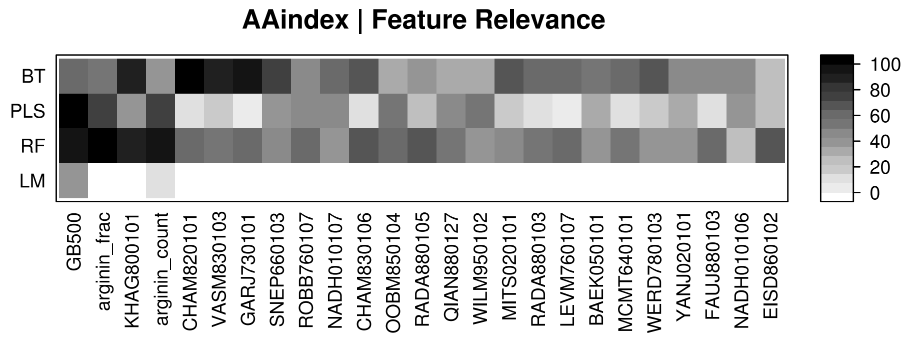


Figure 2.14: Levelplot of AAindex feature relevance, which are estimated by the underlying model specific metrics by LM, RF, PLS and BT. For the graphical representation, a selection is performed to account for the 25 most relevant features on average. Darker gray values indicate a higher feature relevance. The LM assessed relevance of features apparently takes on a special position.

Evaluation is performed for the AAindex dataset as well as for the smaller Heur dataset. To make the results comparable, exactly one predictor is used as reference learning architecture, namely the LLM_{SOM}, for retraining on the derived feature space. For clarity of notation, I will only refer to LLM for the LLM_{SOM} in the remaining.

2.4.4 Weighted Feature Space

The regression models of the learning architectures are evaluated in terms of generalization performance (MSE) as well as their estimation of the contribution of features to the corresponding final model taking the entire, non-weighted feature space as input. The reference results are summarized in Table 2.4 for both the AAindex and Heur feature spaces and the regression architectures LM, LLM and ν-SVR given in the last three columns.

The estimated contributions of features $\mathbf{z}_g$ from the G model specific metrics are used to re-build the model of LLM-type with the input vectors $\tilde{\mathbf{x}}$ based on the optimal parameters determined in the model selection step. A comparison in Table 2.4 of the resulting performances when introducing a feature weighting shows clearly the increase in accuracy for all derived estimations of feature relevance applied to the LLM and AAindex feature space. While an improvement of prediction performance can be observed for the $\text{LLM}_{.}$ compared to standard LLM trained on the entire 531 dimensional AAindex feature space in all cases, this improvement does no longer hold true for the small feature set Heur. In this case, a re-weighting of features by any specific model is disadvantageous. Using the linear predictor LM as a feature weighting method, the estimation of features contributing leads to a decrease in accuracy of the LLM_{LM} trained on the re-weighted AAindex feature space. The feature weighted-LLM

	$\mathbf{LLM_{LM}}$	$\mathbf{LLM_{RF}}$	$\mathbf{LLM_{PLS}}$	$\mathbf{LLM_{BT}}$	**LLM**	**SVR**	**LM**
AAindex	1.12	**1.01**	1.03	**1.01**	1.33	1.03	1.11
Heur	1.22	1.03	1.09	1.04	1.03	1.06	1.16

Table 2.4: Prediction accuracy in terms of testing RMSE of regression models of LLM-type - retrained on AAindex feature space (531 dim) incorporating the input vectors scaled by the feature weightings. The three last columns are given as reference accuracy in feature spaces without scaling. Results are also given for the Heur feature space (18 dim).

trained on the high-dimensional AAindex feature space beats the non-weighted models of LLM-type trained on AAindex feature space in general. It outperforms the ν-SVR in terms of generalization performance for the test set. The LLM_{RF} and LLM_{PLS} models trained on AAindex feature space yield the major improvement of prediction performance.

2.4.5 Weighted and Filtered Feature Space

In an effort to integrate the individual filter algorithms they are combined into a meta-filtering approach. Analogously to the ensemble learning approach, one is interested in a skillful combination of multiple predictors generated. The integration of the different estimations of feature relevance may happen in multiple ways as a meta-filtering, by taking the minimum, the maximum, the product or the mean for each feature over all considered embedded methods. Having a closer look at the distribution of the feature weightings, there is a small number of features accounted as highly relevant, while a few features are estimated to contribute to the predictor model to very low degree. They may be regarded as less important and carrying not much information. It seems reasonable to apply a filtering on the resulting estimated feature weightings prior to retraining the LLM, therewith discarding feature weightings below a certain threshold. The cut-off value is determined by the $c\%$ most relevant features, where $c \in \{95, 90, 70\}$.

As an example, the filtering approach for $\mathbf{z}_{BT}$ with respect to the different values of $c \in \{95, 90, 70\}$ discards the first 225 features if $c = 95\%$, the first 286 in case of $c = 90\%$, and a total of 384 corresponding to $c = 70\%$ if sorted in descending numerical order. Hence, the number of dimensions of the input samples used to form the LLM_{BT} predictor, is reduced to a quantity of $\{306, 245, 147\}$. The precedent filtering of feature weights leads to a slight worse prediction performance of the subsequent applied LLM_{BT}. The corresponding results are given regarding the LLM_{BT} for in terms of testing RMSE by $\{1.013, 1.034, 1.064\}$. With a rising filtering threshold, the prediction performance keeps constant, while the generalization performance decreases for any of the prior applied feature relevance estimators.

2.5 Discussion

One of the most important questions in conjunction with finding a model for predicting peak intensities is the representation of the peptides. A suitable feature space is the precondition for the success of any machine learning method. An a priori given estimation of the feature relevance might provide a helpful ranking of the features. The two-step regression approach benefits from the precedent weighting of the euclidean input space as offered by the model specific metrics. RFs and BTs combine decision trees using Bagging and are members of a complementary class of algorithms opposed to the class of neural networks, the LLM belongs to. The assessed relevance of features contributing to the models of RFs and BTs is used to re-scale the features as input for a subsequent training with the LLM. This facilitates a promising and suitable way of improving the peak intensity prediction.

The statistics of the variable relevance estimation based on the model specific metrics by LM, RF, PLS and BT, are evaluated and listed in Table 2.5. To summarize the statistics and the distribution of the feature relevances, a few features are contributing to the linear model LM and only 28 are used. In contrast to the LM, the non-linear regression models RF, PLS as well as BT, rank many features as contributing less to their built model. These observations mainly rely on the mean relevance and the num-

Statistics	**LM**	**RF**	**PLS**	**BT**
# features > 0	28	530	530	393
Entropy	4.26	7.57	8.32	8.06
Total Sum of Relevance (SoR)	718.976	1753.31	3936.29	6121.71
# features: SoR > 0.9	20	232	318	245

Table 2.5: Statistics on feature relevance estimated by LM, RF, PLS, BT specific metrics, scaled up in the interval [0,100]. The sum of relevance over all 531 features (SoR) is calculated as $\sum_{l=1}^{d_{\text{in}}} \mathbf{z}_{gl}$ for a certain model $g \in \{LM, RF, PLS, BT\}$. The last row accounts for the number of most relevant features, which make up 90% of SoR, i. e. $\arg\min_j \sum_{l=1}^{j} \mathbf{z}_{gl} > 0.9 \cdot SoR$.

ber of features, which make up approximately 90% of the total sum of relevance (SoR). As stated in Guyon and Elisseeff [2003], a variable that is completely useless by itself can provide significant performance improvement when taken with others. Two variables that are useless by themselves can be useful together.

Our results indicate that combined decision trees models, as by RF or BT, give a beneficial hint of the usefulness of the variables.

2.6 Conclusions

In this chapter, special focus is laid on the issue of using relevant features in modeling the non-linear relationship between peptides and the peptide peak heights. The regression architecture of LLM-type is extended to account for a priori given assessments of degrees of feature relevance. These are estimated based on their contributions to different predictor models. The regression based combination of estimated feature weightings and a linear predictor provides simpler models, better generalization and reduced computational costs. A comparison between the two peptide feature representations shows better performance for the high-dimensional AAindex feature space in general. We got the major improvement in the performance of regression models when retraining is performed on the accounted estimated feature relevance by Partial Least Squares and Bagged Trees. These model dependent feature weightings methods perform a skillful scoring of the features in combination with the LLM. Though many features were supposed to be relevant to low degree, integrating a filtering of the feature weightings prior to the retraining of the LLM leads to a decrease in the prediction accuracy. Hence, a vast subset of features accounts less for successful peak intensity prediction models, but still carry that much useful information that discarding them deteriorates the prediction performance. The most relevant AAindex features found amongst others were estimated gas-phase-basicity, fractions of arginine residues, acidic, basic and polar.

2.7 Contribution to OpenMS - An Open-Source Framework for Mass Spectrometry

OpenMS is an open-source framework for mass spectrometry, available at `http://open-ms.sourceforge.net`. We contributed a published module named *PeakIntensityPrediction* (PIP) to the project. The implemented classes, `PeakIntensityPredictor` and `AAindex`, transform the peptide sequence into the corresponding vectorial chemical feature space. Based on these features, a mapping is performed by the incorporated model that has been adapted with the LLM (1×3 nodes) and returns the predicted peptide intensity. For a detailed overview of the contributed module and OpenMS in general, I refer to `http://www-bs2.informatik.uni-tuebingen.de/services/OpenMS-release/html/tutorial_pip.html`.

3 Ensemble Learning

"For every complex problem, there is a solution that is simple, neat, and wrong." Henry Louis Mencken (1880-1956).

Ensembles have been studied in a wide range of areas and found consideration under various names like committee, collections, mixture of experts, classifier systems or multiple classifiers amongst others. Numerous methodologies are still part of the active research for increasing the precision and accuracy by an adequate integration or combination of predictors over single predictors.

In contrast to "classical“ learning architectures, the function to predict is approximated by a *set* of hypotheses rather than by just one single hypothesis. The concept of ensemble learning (EL) is based on voting from a set of different solutions or hypotheses. Ensembles cope with the drawbacks of statistical, computational and representational problems typically arising with single learning algorithms Dietterich [2002]. Given a set of samples $\mathbf{X}$, the problem in regression ensembles is to select a set of appropriate predictors $H = \{f_1, \dots, f_M\}$ from the base hypothesis space $\mathcal{H}$ and to aggregate the outputs of the M base learners by a convex combination to one ensemble prediction. Every learning algorithm corresponds to one point in $\mathcal{H}$. The hypothesis space $\mathcal{H}$ is made up by all possible hypotheses, and the task is to identify the best hypothesis h^* in $\mathcal{H}$.

The construction of accurate and diverse hypotheses overcomes the statistical problems arising from the limited access to the underlying distribution of data. The aggregation of a set of hypotheses reduces these restrictions. A sufficient number of different starting initializations reduces the risk to get stuck in local minima and hence the computational issue. By a linear combination of the hypotheses, the representational drawback is coped with an expansion of the space of representable functions. However, ensembles bare some shortcomings or weaknesses as the demand for an increasing amount of storage and computation and the loss of simple and comprehensible structures.

How can we construct ensembles? From the various methods that have been developed many can be applied to any kind of predictor in general. The common issues in ensemble system design are:

How to generate the training sets? One straightforward way to manipulate the set of hypotheses accessible to a learner is to supply each learner with a slightly altered different training set in an effort to generate diverse ensemble predictors. Diverse hypotheses are explored by the (re-)sampling methods: Cross-validation (CV) like *sampling* methods can be described as dividing the training set into several disjoint subsets and leaving out one portion at a time. *Resampling* methods form predictor models with different training subsets of the available samples. Bagging is one method to build such, where each training set is drawn randomly, but with replacement, from the original training set of samples. Boosting [Freund and Schapire, 1996] directly reassesses the distribution of the samples by incrementally re-weighting, and hence emphasizing on, those training samples that were mis-classified by subsequent models. Other re-sampling methods supply each predictor with all N of the original samples, but each with a different subset of variables as with the Random Subspace Method (RSM).

Which learning algorithm to employ? The choice of learning algorithm is a crucial step and contrary to the credo for single predictors: Unstable and weak learners are necessary to achieve improved accuracies.

How many predictors are sufficient? To make it short, there is no unique answer. This issue strongly depends on the type of algorithm applied and on how diverse the ensemble predictors are. An ensemble of highly redundant or correlated predictors may be replaced by one single predictor without losings.

How to assess models? Learning models may vary in their assessment and algorithmic-specific methods applied. Examples are the distance measure used, the type of kernel or the type of learning function and the direct quantification of diversity as in Negative Correlation Learning (NCL).

How to combine the predictors' output? Having trained the predictors, the strategies employed for combining the outputs of the single predictors to form an ensemble output can be sub-divided into either an unweighted aggregation, i. e. simple averaging as proposed for Bagging, or by a weighted aggregation, e. g. by some measure of accuracy or variance. Other strategies are gating networks [Jordan, 1994] or stacked generalization [Wolpert, 1992].

3.1 Reasons for the Success of Ensembles

Forcing different hypotheses is a common issue in ensemble learning. As already mentioned, balancing the diversity against the predictor accuracy, is the most critical and challenging task for ensemble methods. Maximizing only diversity, may worsen the prediction performance of every single learner. The success of the ensembles is mainly brought with a shift of diversity or complexity of the learning function from one learner to many.

3.1.1 Weak Learners

The base precondition for the success of any ensemble are accurate and diverse predictors [Hansen and Salamon, 1990]. They prove the necessary condition of *weak* learners. Under the presumption of independent errors, a classifier is called *accurate*, if the chances of each classifier to predict new values correctly are slightly better than random guessing. This can be formally described as the probability of k out of M classifiers voting wrong, given a likelihood of $(1-p)$ for the correct classification:

$$\binom{M}{k} p^k (1-p)^{M-k}.$$

By induction it can be shown, that provided $p < 1/2$, with an increasing M the ensemble error rate decreases.

3.1.2 Unstable Learners

Breiman [1996a] gives a formalization when bootstrap aggregation works well: He denotes this as *instability* of learning algorithms, when small changes in the training set cause large changes in predictions. Examples of unstable procedures are neural networks, decision trees, regression trees and subset selection in linear regression, while k-nearest neighbors is stable. The variance is reduced, while the bias is left unchanged.

These characterizations of weak and unstable learners have helped to understand why and when ensembles improve generalization performance compared to any of the individual predictors.

3.1.3 History of Ensembles

The first work on ensembles and combining classifiers might be published by Dasarathy and Sheela [1979] to an ensemble system dividing the input space supplying several classifiers. Hansen and Salamon [1990] introduced ensembles of neural networks, which benefit from similar configured neural networks thereby reducing the variance among networks. Schapire [1990] showed the strength of weak learnability, in which the learner is only required to perform slightly better than guessing. In probably approximately correct (PAC) sense, a weak learner is as strong as a model in which the learner's error can be made arbitrarily small. This work has helped to establish ensemble systems and Schapire's boosting procedure paved the way to one of the most popular family of algorithms in machine learning - AdaBoost [Freund and Schapire, 1995, 1996].

The first study of regression ensembles with neural networks was published by Perrone and Cooper [1993] showing a guaranteed improvement by averaging in functional space. Hashem [1994] derived optimal linear combinations of neural networks. The first experimental study on combining regressors using Boosting was published by Drucker [1997]. Avnimelech and Intrator [1999] extended the Boosting algorithm to regression problems. Carney and Cunningham [1999] proposed to estimate the set of weights and the number of hidden units in neural network ensembles based on out-of-bag error estimates.

Empirical studies and comparisons of the popular ensemble methods Bagging and Boosting can be found in Quinlan [1996]; Maclin [1997]; Bauer and Kohavi [1998]; Opitz and Maclin [1999]; Dietterich and Fisher [2000] or Breiman [1998]. In 1997, Dietterich [1997] points out Ensemble Learning as one of the four current directions in research activities. A good and comprehensive overview of the recent methods in ensemble learning is provided by Polikar [2006]. In the year 2000 the first workshop on "Multiple Classifier Systems" was held, which meanwhile has well-established as a forum for the issues in ensemble system design.

3.2 Assessing Ensemble Error

For regression tasks, the ensemble output is given by a weighted combination of the single predictors' output $f_m, m = 1, \ldots, M$:

$$\bar{f}(\mathbf{x}) = \sum_m w_m f_m(\mathbf{x}), \quad \text{where } w_m \geq 0, \sum_m w_m = 1, \quad \forall m = 1, \ldots, M.$$

Analogously to the case of one predictor, the ensemble error rate is defined as the squared error between the desired target value y and the predicted value $\bar{f}(\mathbf{x})$ yielding the ensemble prediction error:

$$\bar{e}(\mathbf{x}) = (\bar{f}(\mathbf{x}) - y)^2.$$

The expected error in predicting a single sample $\mathbf{x}$ can be decomposed into a bias- and a variance term. This decomposition is well known as the bias-variance trade-off when controlling the model complexity and also holds true for the ensemble vote $\bar{f}$[1]:

$$\begin{aligned} E\{(\bar{f} - y)^2\} &= (E\{\bar{f}\} - y)^2 + E\{(\bar{f} - E\{\bar{f}\})^2\} \\ &= bias(\bar{f})^2 + var(\bar{f}). \end{aligned}$$

Often, a simple averaging over the ensemble predictors' outputs f_m is used according to $w_m = 1/M$. This is used interchangeably - if unambiguous.

3.2.1 Bias-Variance-Covariance Decomposition

The expected quadratic error of the ensemble can be decomposed into three terms similar to the bias-variance decomposition for one single learner leading to the *bias-*

[1]In the remainder, I will omit the input vectors $f(\mathbf{x})$ to f.

variance-covariance decomposition [Geman et al., 1992]:

$$\begin{aligned} E\{(\bar{f}-y)^2\} &= (E\{\bar{f}\}-y)^2 + E\{(\bar{f}-E\{\bar{f}\})^2\} \\ &= bias(\bar{f})^2 + var(\bar{f}) \\ &= \overline{bias}^2 + \frac{1}{M}\overline{var} + \left(1-\frac{1}{M}\right)\overline{covar} \end{aligned} \tag{3.1}$$

With this concept the expected error of the ensemble is composed of a fraction of the variance as well as of the *covariance* along the ensemble predictors, when the following three components are defined as:

$$\overline{bias} = \frac{1}{M}\sum_m (E\{f_m\}-y) \tag{3.2}$$

$$\overline{var} = \frac{1}{M}\sum_m E\{(f_m - E\{f_m\})^2\} \tag{3.3}$$

$$\overline{covar} = \frac{1}{M(M-1)}\sum_m \sum_{j\neq m} E\{(f_m - E\{f_m\})(f_j - E\{f_j\})\}. \tag{3.4}$$

The components are the averaged bias (3.2), the averaged variance (3.3) and the covariance (3.4) between the ensemble predictors.

3.2.2 Ambiguity Decomposition

Krogh and Vedelsby [1995] first presented the decomposition of the ensemble error rate and gave the definition of the *ensemble ambiguity*. They proved that the ensemble error rate is less or equal to the mean squared error e_m of the individual predictors. The *ensemble ambiguity* $\bar{a}$ on a certain sample is determined by

$$\begin{aligned} \bar{a} &= \sum_m w_m (y-f_m)^2 - (y-\bar{f})^2 \\ &= \sum_m w_m e_m - \bar{e}, \quad \text{where } e_m = (y-f_m)^2 \\ \bar{e} &= (y-\bar{f})^2 \\ &= \sum_m w_m e_m - \bar{a}. \end{aligned} \tag{3.5}$$

On the right side of Equation (3.5), the first term is the average error of individual predictors. The term $\bar{a}$ measures how much each single ensemble predictor diverges from the ensemble output $\bar{f}$, the so called *diversity* or *ambiguity*. The higher the diversity, the lower is the ensemble error rate, if the mean predictor error is fixed.

To find out about the relation of the bias-variance-covariance decomposition (3.1) to the ambiguity decomposition (3.5), an additional term is introduced. Brown [2004] introduced the term Ω, which connects both components, diversity and mean predictor error:

$$\begin{aligned}
E\{(\bar{f}-y)^2\} &= bias(\bar{f})^2 + var(\bar{f}) \pm \Omega \\
&= E\{\frac{1}{M}\sum_m (y-f_m)^2 - \frac{1}{M}\sum_m (f_m-\bar{f})^2\} \pm \Omega \\
&= \overline{bias}^2 + \frac{1}{M}\overline{var} + \left(1-\frac{1}{M}\right)\overline{covar} \pm \Omega \\
\Omega &= \overline{var} + \frac{1}{M}\sum_m (E\{f_m\} - E\{\bar{f}\})^2\} \\
E\{\frac{1}{M}\sum_m (y-f_m)^2\} &= \overline{bias}^2 + \Omega \\
E\{\frac{1}{M}\sum_m (f_m-\bar{f})^2\} &= \Omega - \left[\frac{1}{M}\overline{var} + \left(1-\frac{1}{M}\right)\overline{covar}\right] \qquad (3.6) \\
&= \left(1-\frac{1}{M}\right)(\overline{var} - \overline{covar}) + \frac{1}{M}\sum_m (E\{f_m\} - E\{\bar{f}\})^2\}
\end{aligned}$$

The diversity is determined by both variance and covariance. The Ω term reflects the strong interaction between the diversity and the remaining parts of the error [Brown, 2004].

3.3 Assessing Diversity

In the previous section, I have listed different definitions how to explicitly *quantify* diversity (Equation (3.5) and (3.6)) - all arising from the decomposition of the ensemble error rate to the two terms:

$$\bar{e} = (y-\bar{f})^2 = \sum_{m=1}^{M} w_m (y-f_m)^2 - \sum_{m=1}^{M} w_m (f_m-\bar{f})^2 \qquad (3.7)$$

$$= \mathrm{MSE}_{\mathrm{Map}} - \mathrm{DIV}. \qquad (3.8)$$

The first, $\mathrm{MSE}_{\mathrm{Map}}$, is the mean squared error per every single predictor, and second, DIV, measures how much each single ensemble predictor diverges from the ensemble output $\bar{f}$, the *diversity*.

3.3.1 Strengthening the Effect by Diverse Predictors

Brown [2004] studied where the diversity arises from and formalized a taxonomy for creating diversity. The approaches encouraging diverse hypotheses differ in the way the ensemble error diversity is quantified and integrated in the training process - implicitly or explicitly. One way to manipulate the set of hypotheses accessible to a learner is to supply each learner with a slightly altered different training set to generate diverse ensemble predictors. While Bagging (3.3.1.1) is one implicit method to build altered learners, other re-sampling methods supply each predictor with all N patterns, but with different subset of variables as in RSM (3.3.1.2). Every learner is presented only a subtask of the entire learning task and the learning is restricted to the information presented. Boosting directly reassesses the distribution of the samples and rather could be classified as a heuristically acting explicit method. A typical example for an *explicit* method is NCL, see Section 3.3.2. Only NCL directly takes the quantified ensemble error diversity into account.

3.3.1.1 Bagging

> ”Bagging goes a way toward making a silk purse out of a sow's ear, especially if the sow's ear is twitchy.“ Breiman [1996a]

One well-studied and intuitive way of combining the ensemble predictors is simple averaging ($w_m = 1/M$). The method of Bagging, as the short form of ***bootstrap aggregating***, was introduced by Breiman [1996b]. He proposed aggregating a set of predictors generated from bootstrapped samples $\mathbf{T}_1, \ldots, \mathbf{T}_M$. From the original set of N training samples, N samples are randomly selected ***with replacement***. These M bootstrap training sets $\mathbf{T}_m$ $(m = 1, \ldots, M)$ are the training bases for the constructed predictors $f(\mathbf{x}, \mathbf{T}_m) = f_m(\mathbf{x})$ that are used to form the bagged predictor $\bar{f}$. He showed empirically as well as theoretically that Bagging improves the generalization performance for any kind of predictor, which fulfills instability. There exists evidence that this kind of aggregation, i. e. averaging from bootstrapped samples, constrains the complexity, moreover, reduces the variance while having same bias. The training sets $\mathbf{T}_m$ contain about a fraction $(1 - 1/M)^M \approx 0.63$ of the original N samples. Therefore about one-third of the patterns are left out in every bootstrapped sample and are called "out-of-bag" (OOB) data. The OOB estimates are given by the aggregation over the predictors f_m, which do not contain sample $(\mathbf{x}, y) \in \mathbf{T}_m$. These OOB estimates tend to overestimate the test error rate, since the prediction is based on only one-third of the M predictors. The error rate decreases with an increasing number of predictors in the ensembles. The OOB data can be used for form accurate estimates for important quantities like error, strength, correlation and feature relevance as discussed by Breiman [1996b].

3.3.1.2 Random Subspace Method

The Random Subspace Method (RSM) [Ho, 1998] was proposed for classifiers of multiple trees constructed in randomly chosen subspaces of size K. Each predictor $m = 1, \dots, M$ is presented a subset $\mathbf{K}_m$ of size K of the entire set of features $\{1, \dots, d_{\text{in}}\}$. This way, for each predictor a randomly sampled vector $\mathbf{K}_m$ is constructed independent of the past random vectors $(\mathbf{K}_1, \dots, \mathbf{K}_{m-1})$.

3.3.1.3 Random Forests

Breiman [2001] proposed Random Forests (RF) as a combination of decision tree predictors for classification as well as regression tasks. Random Forests are defined as "a collection of tree-structured classifiers $\{f(\mathbf{x}, \theta_m),\ m = 1, \dots\}$ where the $\{\theta_m\}$ are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input $\mathbf{x}$" [Breiman, 2001]. RFs are designed to minimize the correlation between the predictors while maintaining strength. The first portion of randomness is injected by the bootstrap aggregating, i. e. Bagging, such that the predictors $\{f(\mathbf{x}, \theta_m)\}$ are built independent of the past random vectors $\theta_1, \dots, \theta_{m-1}$. With Bagging, the random vectors $\theta_1, \dots, \theta_M$ are the bootstrapped training sample sets $\mathbf{T}_1, \dots, \mathbf{T}_M$. A RF is built by several random trees, each of them constructed independently. Second, the nodes of each tree are split by the best presented variable of a randomly drawn subset of variables for each node, as proposed for the RSM. Typically, each node is presented only a third of the entire variables. For classification, the aggregated output is taken as a majority vote. The average over all predictor outputs is taken for regression.

This class of procedures gives good accuracy, is robust to outliers and noise, faster than Bagging with all variables and can easily be scheduled in parallel.

3.3.2 Negative Correlation Learning

Liu and Yao [1999] introduced the term *Negative Correlation Learning* (NCL). In NCL, the error function e_m of the ensemble predictors is extended by an additional penalty term to balance between the accuracy of individual predictors and the quantified diversity:

$$e_m = \tfrac{1}{2}(f_m - y)^2 - \gamma(f_m - \bar{f})^2.$$

γ is a parameter controlling the penalty for a high correlation of the individual predictors errors and thereby enforcing the diversity or *negative correlation* in errors. One varies between the extremes of a single predictor ($\gamma = 0$) and a fully connected ensemble ($\gamma = 1$), where each predictor is influenced by each of the others. Thus, with the penalty strength parameter γ, the degree of diversity within the model is specified, which could be referred to as the *model complexity*. For a good overview on the theory behind NCL, I recommend Brown et al. [2005c].

3.3.3 Parameterizing Penalty Functions

There are several penalty functions discussed in the literature, though the connection between the often used and competitive $\bar{f}$**-penalty** and the y**-penalty** is "not as well understood" as noted by Brown et al. [2005b]. These penalties are given by:

$$\begin{aligned}
\bar{f}\textbf{-penalty:}\quad e_m^{(1)} &= \tfrac{1}{2}(f_m - y)^2 + \gamma(f_m - \bar{f}) \sum_{j \neq m} (f_j - \bar{f}) && (3.9)\\
&= \tfrac{1}{2}(f_m - y)^2 - \gamma(f_m - \bar{f})^2 && (3.10)\\
y\textbf{-penalty:}\quad e_m^{(2)} &= \tfrac{1}{2}(f_m - y)^2 + \gamma(f_m - y) \sum_{j \neq m} (f_j - y) && (3.11)
\end{aligned}$$

Equation (3.9) can be transformed to Equation (3.10), for the following holds true:

$$\sum_{j \neq m} (f_j - \bar{f}) \;=\; -(f_m - \bar{f}).$$

The gradients of the penalty functions are after some transformations given by:

$$\begin{aligned}
\bar{f}\textbf{-penalty:}\quad \frac{\partial e_m^{(1)}}{\partial f_m} &= (f_m - y) - \gamma \left[2 \left(1 - \frac{1}{M} \right) (f_m - \bar{f}) \right] && (3.12)\\
&= (f_m - y) - \lambda(f_m - \bar{f})\\
&= (f_m - y) - \lambda(f_m - \bar{f}) + \lambda y - \lambda y\\
&= (f_m - y) + \lambda \sum_{j \neq m} (f_j - \bar{f})\\
&= (1 - \lambda)(f_m - y) + \lambda(\bar{f} - y)\\
\frac{\partial^2 e_m^{(1)}}{\partial f_m^2} &= 1 - \lambda(1 - \frac{1}{M})
\end{aligned}$$

$$\begin{aligned}
y\textbf{-penalty:}\quad \frac{\partial e_m^{(2)}}{\partial f_m} &= (f_m - y) + \lambda \sum_{j \neq m} (f_j - y) && (3.13)\\
&= (f_m - y) + \lambda[(\sum_{j=1}^{M} (f_j - y)) - (f_m - y)]\\
&= (f_m - y) + \lambda[M(\bar{f} - y) - (f_m - y)] && (3.14)\\
&= (1 - \lambda)(f_m - y) + \lambda M(\bar{f} - y)\\
\frac{\partial^2 e_m^{(2)}}{\partial f_m^2} &= 1
\end{aligned}$$

where

$$\lambda \;=\; 2\gamma\left(1-\tfrac{1}{M}\right)$$

is the strength parameter and relates the ensemble size M to the penalty. To derive Equation (3.14) from its previous line, I just use the fact that uniform weights are assumed. Therefore, the following holds true:

$$\sum_j (f_j - y) = M \cdot (\bar{f} - y).$$

Both penalty functions can be evaluated separately. But they are connected through a simple weighting in the derived gradients. I join the both decompositions by introducing the parameter Υ and define the **Υ-penalty**:

$$\textbf{Υ-penalty:}\quad \frac{\partial e_m^{\Upsilon}}{\partial f_m} \;=\; (1-\lambda)(f_m - y) + \lambda\Upsilon(\bar{f} - y). \tag{3.15}$$

The upper equations can be derived by setting Υ to $\Upsilon = 1$ (3.12) or $\Upsilon = M$ (3.13) and thus varying between the extremes of $\bar{f}$- and the y-penalty. We are now given a *family of parameterized functions*, aside from λ, depending on the new parameter Υ.

Any predictor $f_m = \sum_q w_{qm}\phi_{qm}$ is given as a linear combination of internal (non)-linear functions ϕ. According to the argumentation of Brown et al. [2005c], one can derive an upper bound for the strength parameter λ: It has to be ensured that every entry of the diagonal corresponding to the qth weight of the mth predictor, $\frac{\partial^2 e_m}{\partial w_{qm}^2}$, has to be positive-valued. Otherwise all of the entries of the corresponding Hessian matrix could not be positive-definite. Applied to the Υ-penalty, this leads to the following derivations:

$$\begin{aligned}
0 \;<\; \frac{\partial^2 e_m^{\Upsilon}}{\partial w_{qm}^2} \;&=\; \left[\frac{\partial}{\partial w_{qm}}\frac{\partial e_m^{\Upsilon}}{\partial f_m}\right]\frac{\partial f_m}{\partial w_{qm}} + \left[\frac{\partial}{\partial w_{qm}}\frac{\partial f_m}{\partial w_{qm}}\right]\frac{\partial e_m^{\Upsilon}}{\partial f_m} \\
&=\; \left[\frac{\partial}{\partial w_{qm}}(1-\lambda)(f_m - y) + \lambda\Upsilon(\bar{f} - y)\right]\frac{\partial f_m}{\partial w_{qm}} + \left[\frac{\partial}{\partial w_{qm}}\frac{\partial f_m}{\partial w_{qm}}\right]\frac{\partial e_m}{\partial f_m} \\
&=\; \left(\phi_{qm} - \lambda\phi_{qm} + \lambda\frac{\Upsilon}{M}\phi_{qm}\right)\phi_{qm} \\
&=\; \left[\phi_{qm} - \lambda\left(\phi_{qm} - \frac{\Upsilon}{M}\phi_{qm}\right)\right]\phi_{qm} \\
&=\; \left[1 - \lambda\left(1 - \frac{\Upsilon}{M}\right)\right]\phi_{qm}^2 \\
&=\; \frac{\partial^2 e_m^{\Upsilon}}{\partial f_m^2}\phi_{qm}^2
\end{aligned}$$

$$\begin{aligned}
\Leftrightarrow \quad 0 &< \left[1-\lambda\left(1-\frac{\Upsilon}{M}\right)\right]\phi_{qm}^2 \\
0 &< \phi_{qm}^2 - \lambda\phi_{qm}^2\left(\frac{M-\Upsilon}{M}\right) \\
\phi_{qm}^2 &> \lambda\phi_{qm}^2\left(\frac{M-\Upsilon}{M}\right) \\
\lambda &< \frac{\phi_{qm}^2}{\phi_{qm}^2\left(\frac{M-\Upsilon}{M}\right)}
\end{aligned}$$

$$\Leftrightarrow \quad \lambda < \frac{M}{M-\Upsilon} \tag{3.16}$$

To the upper bound of λ does apply:

$$\lambda < \frac{M}{M-\Upsilon} = \begin{cases} \infty & \text{if } \Upsilon = M \\ \frac{M}{M-1} & \text{if } \Upsilon = 1 \\ \frac{M^2}{M^2-1} & \text{if } \Upsilon = \frac{1}{M} \\ 1 & \text{if } \Upsilon = 0 \end{cases} \tag{3.17}$$

Recall the ensemble error rate as

$$e_{\text{ens}} = (\bar{f}-y)^2 = (\frac{1}{M}\sum_m f_m - y)^2.$$

With $\lambda = 1$ in a fully connected ensemble, a single-unit-like ensemble, we achieve:

$$\frac{\partial e_m^{\Upsilon}}{\partial f_m} = \begin{cases} \frac{\partial e_{\text{ens}}}{\partial f_m} = \frac{1}{M}\left(\bar{f}-y\right) & \text{if } \Upsilon = \frac{1}{M} \\ \frac{\partial e_m^{(1)}}{\partial f_m} = M\frac{\partial e_{\text{ens}}}{\partial f_m} & \text{if } \Upsilon = 1 \\ \frac{\partial e_m^{(2)}}{\partial f_m} = M^2\frac{\partial e_{\text{ens}}}{\partial f_m} & \text{if } \Upsilon = M \end{cases} \tag{3.18}$$

What does this mean for Υ and λ? The corresponding situation depends on the chosen value of parameter Υ giving the derived upper bound for λ as illustrated in Figure 3.1. The more Υ is next to zero, the smaller is the possible range of λ. If Υ is next to M, the possible range of λ gets unbounded. The discussed problem that arises when the parameter Υ actually goes to M, is mainly a computational one. It can no longer be guaranteed to find an optimal strength parameter λ, when theoretically no upper bound ($\lambda < \infty$) is given. However, as Brown et al. [2005b] already stated for $\Upsilon = M$ corresponding to the y-**penalty** function in Equation (3.13), a larger Υ leads to an increase in speed and allows a faster convergence due to the steepness of the error landscape. This behavior can be explained due to:

$$\begin{aligned}\frac{\partial^2 e_m^{\Upsilon}}{\partial f_m^2} &= 1 - \lambda\left(\frac{M-\Upsilon}{M}\right)\\ &= \begin{cases} 1-\lambda\left(1-\dfrac{1}{M^2}\right) & \text{if } \Upsilon = \dfrac{1}{M}\\ 1-\lambda\left(1-\dfrac{1}{M}\right) & \text{if } \Upsilon = 1\\ 1 = \dfrac{\partial^2 e_m^{(2)}}{\partial f_m^2} = \dfrac{\partial^2 e_m^{(1)}}{\partial f_m^2}_{\;(\lambda\,=\,0)} & \text{if } \Upsilon = M \text{ or } \lambda = 0\end{cases}\end{aligned}$$

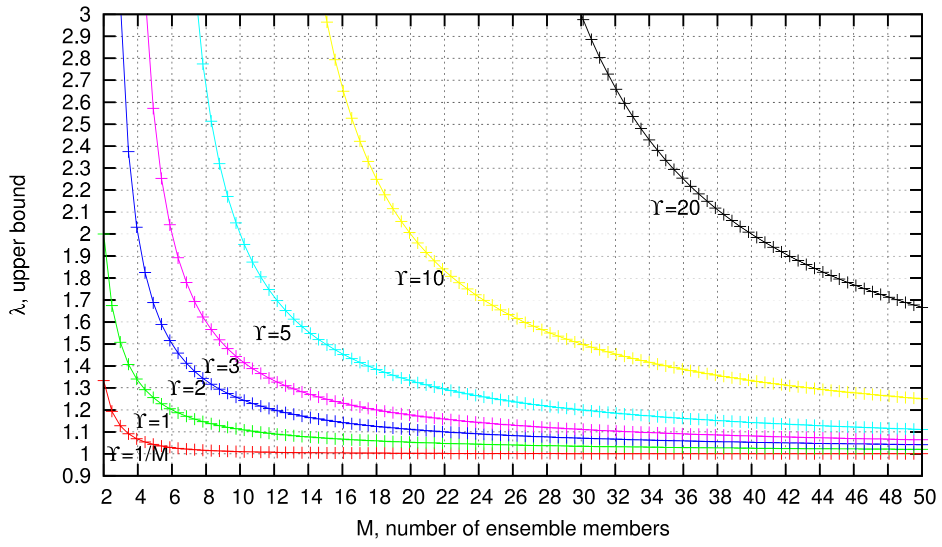


Figure 3.1: Upper bound of λ given by M and $\Upsilon \in \{\frac{1}{M}, 1, 2, 3, 5, 10, 20\}$.

4 LERRANCO Architecture

This chapter introduces to the proposed ensemble architecture LERRANCO using Local Linear Maps (LLMs) based on SOMs. LERRANCO is the acronym for *L*ocal Linear *E*nsembles for *R*egression with *R*esampling *A*nd *N*egative *CO*rrelation Learning.

Since in many application areas the issue of learning speed is as important as the issue of prediction accuracy [Jordan and Jacobs, 1998], I propose to combine SOMs to an ensemble structure. Regarding Self-Organizing Maps (SOMs), training an ensemble or, more generally, a set of SOMs has been proposed quite early by Fritzke [1994], allowing growing structures, hierarchically arranged structures [Miikkulainen, 1990] or a combination of both, as proposed by Dittenbach et al. [2000], or by Multi-SOMs [Goerke et al., 2001]. Ensemble Learning offers by contrast to these approaches a non-competing strategy of training the single networks.

4.1 Related Work on SOM Ensemble Learning

Ensembles of SOMs have primarily been studied in the context of ensemble methods for clustering and are still part of active research. SOMs have been shown to be one of the most popular tool for clustering, classification, data mining and visualization purposes due to its topology preserving property. Though, with SOM *ensembles* one loses the comprehensible and interpretable structures, facilitating analysis and enhanced visualizations, as it is for any other learning architecture applied. We are faced with the problem of finding an appropriate consensus scheme for clustering of prototypes in ensembles. We can not hope for a consistent common clustering over the independently trained predictors, which would just require some aligning or re-labeling of clusters. This issue can be handled by mapping the total set of resulting prototypes onto a subsequently applied SOM approximating as a secondary clustering. But this process increases computational costs and we cannot rely on a comprehensible map or visualization.

What it makes so hard or even impossible to cope with are indeed some stochastic elements imposed by the design of SOMs: First of all, the prototype vectors are randomly initialized either linear in input space or along the first principal components. The second element is the random order, in which the samples are presented while training. However, this is a necessary condition for successfully applying SOMs in general. The commonality of clusterings is hampered arguably the most by the techniques, which boost the diversity along the ensemble predictors. Some ways of overcoming this dilemma between commonality and essential diversity in SOM ensembles are discussed in the literature. Petrakieva and Fyfe [2003] examine how some constraints on the initial conditions of SOMs help to give clusters, which are comparable from net to net. They investigate to force the commonality by *incrementally* constraining the centers of the second and subsequent SOMs to the positions of the centers learned by previous SOMs. An alignment procedure by matching pairs of clusters, whose number of overlapping samples is the largest, is examined by [Jiang and hua Zhou, 2004; Zhou and Tang, 2006]. Georgakis and Li [2005] fused several SOMs into one final map incrementally by merging the nodes of one SOM onto previous SOMs at each time. Doing so, any topology and neighborhood along the single SOMs is destructed.

4.2 Proposed LERRANCO Architecture

Nevertheless, in all of these approaches the set of single predictors has been acting independently of each other. In the light of new theory behind EL, in particular NCL, the question arises if SOM EL can benefit from non-independent learning when individual learning stages are interlinked by *inter-SOM diversity* error rates.

In order to transfer the common core issues in ensemble construction into the special context of LERRANCO architecture, its necessary to pick up the general context as mentioned at the beginning of the previous Chapter 3:

How to generate the training sets? Inspired by the advantages of the RF architecture, the ideas of building powerful estimators with part-adapting structures are adopted. The proposed ensemble architecture LERRANCO integrates the resampling methods Bagging (3.3.1.1) and RSM (3.3.1.2).

Which learning algorithm to employ? The initial issue was to model the non-linear relationship between peptides and their intensities in mass spectra. The SOM has been shown to be a valuable tool for this modeling purpose with major advantage due to its topology preserving characteristic, its fast training and its low memory-usage.
The answers to the two preceding core questions are evaluated and discussed in details in Chapter 5.

How to assess models? NCL allows to balance between the single predictor accuracy and diversity, controlled by the cooperation among the SOMs, for improved generalization performance. With the NCL strength parameter λ, one varies between the extremes of independently acting predictors and a fully-connected ensemble. The evaluation of the integration of the NC penalty term is addressed in Chapter 6.

How to combine the predictors' output? The aggregation by averaging is probably the most simple and widely used method to form an ensemble predictor and is covered in Chapter 7.

A schematic description of the proposed LERRANCO architecture is depicted in Figure 4.1. It includes the following steps according to the listed Algorithm 1:

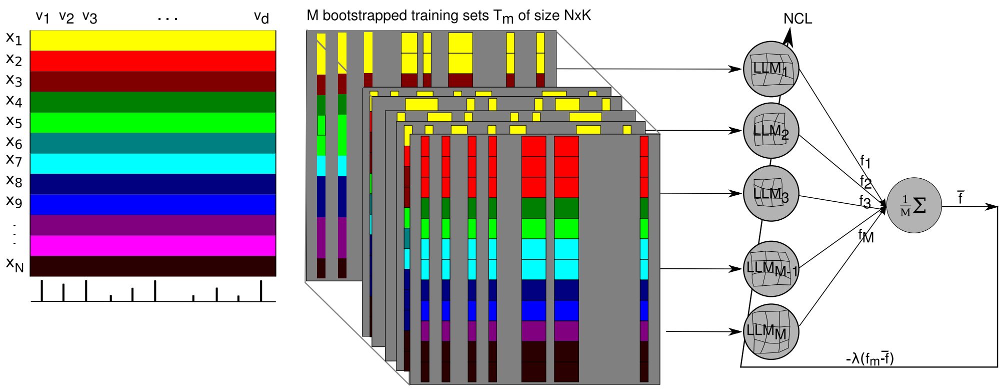


Figure 4.1: Schematic overview of proposed LERRANCO ensemble architecture. First step is the training set generation (sub-sampling). The second component is the training of the SOMs. The LLM performs a fitting of locally trained linear mappings based on the M SOMs. Third component is the explicitly quantification of diversity between the predictors taken into account by NCL. The outputs are combined to form the ensemble prediction $\bar{f}$.

LERRANCO($\mathbf{T}$, $M = 100$, $K < d_{\text{in}}$, $\Upsilon = 1$, $\lambda = 1$, VQ("SOM", 2×5, $\sigma = 2.0$)).

The architecture requires the number M of LLM predictors. For each network $m = 1, \ldots, M$, the bootstrapped samples are created with replacement of size N from training data $\mathbf{T} \in \mathbb{R}^{N \times d_{\text{in}}}$. In addition, a random subset of features $\mathbf{K}_m$ of size K (without replacement) is taken. With $K < d_{\text{in}}$, the entire learning task given by $\mathbf{T} \in \mathbb{R}^{N \times K}$ is sub-sampled and each predictor acts in a randomly sampled subspace of size K. Each of the M LLM models are used to carry out the presented sub-task $\mathbf{T}_m$ of size $N \times K$ given by all the samples $(\mathbf{x}_n, y_n)$ contained in $\mathbf{T}_m$. The LLM predictors may be based on any of the VQ-based approaches, like GNG, NG, FCL or SOM. The weights of each node are updated according to the gradient descent procedure. The width of the step is determined in accordance to NCL. The remaining two parameters Υ and λ concern NCL (3.3.2 on page 52 and 3.3.3 on page 53) and define the type and strength of NCL. The ensemble predictors are evaluated and aggregated to form the ensemble prediction by simple averaging.

A publicly available R package providing my implementation of the LERRANCO architecture is found here Scherbart [2009].

Algorithm 1 LERRANCO($\mathbf{T} \in \mathrm{I\!R}^{N\times d_{\text{in}}}$, $M = 100$, $K < d_{\text{in}}$, $\Upsilon = 1$, $\lambda = 1$, VQ("SOM", 2×5, $\sigma = 2.0$))

```
Require: Training set $\mathbf{T} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ with $d_{\text{in}}$ features
Require: $M$, the number of predictors in the ensemble
Require: $K$, the size of random subspaces ($K < d_{\text{in}}$)
Require: $\Upsilon$, the parameter inducing the type of penalty function for Negative Correlation Learning (NCL), default is $\Upsilon = 1$
Require: $\lambda$, the parameter for NCL, depends on $\Upsilon$ ($0 \leq \lambda \leq 1$)
 1: for $m = 1$ to $M$ for each network do
 2:     Create bootstrapped samples $\mathbf{T}_m$ with replacement of size N from the training data $\mathbf{T}$
 3:     Take a random subset of features $\mathbf{K}_m$ of size $K$ (without replacement)
 4: end for
 5: Train LLM models $f_1, \ldots, f_M$ with $(\mathbf{T}_1, \mathbf{K}_1), \ldots, (\mathbf{T}_M, \mathbf{K}_M)$:
 6: for $n = 1$ to $N$ for each training sample $(\mathbf{x}_n, y_n) \in \mathbf{T}$ do
 7:     for $m = 1$ to $M$ for each network if and only if $(\mathbf{x}_n, y_n) \in \mathbf{T}_m$ do
 8:         Find winner node $\kappa \in \{1, \ldots, n_l\}$ which is located closest to sample $\mathbf{x}_n$:
                $\kappa = \arg\min_i \left\{\|\mathbf{x}_n - \mathbf{w}_i^{\text{in}}\|\right\}$
 9:         Map sample $\mathbf{x}_n$ to output value by winner node $\kappa$ as a *winner-take-all* rule:
                $f_m(\mathbf{x}_n) = \mathcal{C}_\kappa(\mathbf{x}_n) = \mathbf{w}_\kappa^{\text{out}} + \mathrm{A}_\kappa\left(\mathbf{x}_n - \mathbf{w}_\kappa^{\text{in}}\right)$
10:         Evaluate ensemble members and aggregate to ensemble prediction by averaging:
                $\bar{f}(\mathbf{x}_n) = \frac{1}{M}\sum_m f_m(\mathbf{x}_n)$
11:         Determine mean error (MSE) for each ensemble member:
                $e_m = (f_m(\mathbf{x}_n) - y_n)^2$
12:         Determine gradient descent step in accordance to NCL:
                $\frac{\partial e_m}{\partial f_m}(\mathbf{x}_n) = (1-\lambda)(y_n - f_m(\mathbf{x}_n)) + \lambda\Upsilon(y_n - \bar{f}(\mathbf{x}_n))$
13:         Perform a single update for the weights of nodes $\mathbf{v}_i = (\mathbf{w}_i^{\text{in}}, \mathbf{w}_i^{\text{out}}, \mathbf{A}_i)$:
                $\Delta\mathbf{w}_i^{\text{in}}(\mathbf{x}_n) = \epsilon^{\text{in}} \cdot h_{i\sigma} \cdot \left(\mathbf{x}_n - \mathbf{w}_i^{\text{in}}\right)$
                $\Delta\mathbf{w}_i^{\text{out}}(\mathbf{x}_n) = \epsilon^{\text{out}} \cdot h_{i\sigma} \cdot \left[\frac{\partial e_m}{\partial f_m}(\mathbf{x}_n)\right]$
                $\Delta\mathbf{A}_i(\mathbf{x}_n) = \epsilon^{\mathrm{A}} \cdot h_{i\sigma} \cdot \left[\frac{\partial e_m}{\partial f_m}(\mathbf{x}_n)\right] \cdot \frac{(\mathbf{x}_n - \mathbf{w}_i^{\text{in}})^T}{\|\mathbf{x}_n - \mathbf{w}_i^{\text{in}}\|^2}.$
14:     end for
15: end for
```

4.3 Accurate and Diverse Ensemble Predictors

Sharkey [1996] subdivides the factors in efforts to obtain diverse ensembles of neural networks into initial conditions, the training data, the topology of the nets and the training algorithm. However, I agree with Brown et al. [2005a] that the taxonomy is valid for the majority of methodologies emphasizing on diversity and allows to categorize them. But with NCL, none of the above mentioned categorizations can be applied. The categorization of diversity strategies is adopted and extended in this thesis as follows:

1. In this approach, the hypothesis space is directly connected to the positions of the nodes $\mathbf{v}_i = (\mathbf{w}_i^{\text{in}}, \mathbf{w}_i^{\text{out}}, \mathrm{A}_i)$. The diversity along the ensemble predictors in my architecture arises first of all from the random starting initialization of the nodes $\mathbf{v}_i$. Hence, the *starting point in hypothesis space* is determined for time $t = 0$: $\mathbf{w}_0 \models (\mathbf{T}_m, \mathbf{K}_m)$, where $\mathbf{K}_m$ is the subset of variables randomly chosen.
2. The *set of accessible hypotheses* is restricted to the information accessible to the learner. This is enforced by aggregation of the resampling techniques Bagging and RSM, thereby altering the training sets and presenting each single network a separate subtask of the entire learning task.
3. The *intra-member* or *intra-SOM* diversity is a intrinsic property of the SOM, as a single SOM can be seen as an ensemble itself. Though the weights are not equally distributed, they rather depend on the distance of sample $\mathbf{x}_n$ to its closest prototype $\mathbf{w}_\kappa^{\text{in}}$. That is,

$$\begin{aligned} f_m(\mathbf{x}_n) &= \sum_i w_{im}\phi_{im}(\mathbf{x}_n) = \sum_i h_{i\sigma}\mathcal{C}_{im}(\mathbf{x}_n) \\ \text{with } \sum_i h_{i\sigma} &= 1 \text{ and } h_{i\sigma} \geq 0 \; \forall i = 1, \ldots, n_l. \end{aligned} \tag{4.1}$$

When varying the width σ of the Gaussian neighborhood function, $h_{i\sigma}$, which refers to the short form of Equation (2.1) on page 16, the connectivity of nodes is changed. With a weak connectivity, one can expect diverse SOM nodes, inducing a VQ similar to k-Means, but with high prediction error. On the other hand, a strongly intra-connected SOM should offer non-ambiguous nodes with a low *intra-SOM diversity.* The nodes while supervised training are adapted according to:

$$\begin{aligned} \Delta\mathbf{w}_i^{\text{out}}(\mathbf{x}_n) &= \epsilon^{\text{out}} \cdot h_{i\sigma} \cdot \left[\frac{\partial e_m}{\partial f_m}(\mathbf{x}_n)\right] \\ \Delta\mathbf{A}_i(\mathbf{x}_n) &= \epsilon^{\mathrm{A}} \cdot h_{i\sigma} \cdot \left[\frac{\partial e_m}{\partial f_m}(\mathbf{x}_n)\right] \cdot \frac{(\mathbf{x}_n - \mathbf{w}_i^{\text{in}})^T}{\|\mathbf{x}_n - \mathbf{w}_i^{\text{in}}\|^2} \end{aligned}$$

$$\text{where} \quad \frac{\partial e_m}{\partial f_m}(\mathbf{x}_n) = (y_n - f_m(\mathbf{x}_n))$$
$$\text{and} \quad e_m = \tfrac{1}{2}(y_n - f_m(\mathbf{x}_n))^2$$

$\epsilon^{\text{out}}, \epsilon^{\text{A}}$ denote the learning step widths.

4. The methods that explicitly encourage diversity can be interpreted as a directed ***traversal of hypotheses space***. In case of NCL the ensemble members interact and are forced to follow different trajectories in hypotheses space. I emphasize on noting this to be a *inter-member* strategy:

$$\frac{\partial e_m}{\partial f_m}(\mathbf{x}_n) = (1-\lambda)(y_n - f_m(\mathbf{x}_n)) + \lambda\Upsilon(y_n - \bar{f}(\mathbf{x}_n))$$

Introducing a strong focus on the explicitly quantifying and boosting diversity along the SOMs enforces the variances between the ensemble predictors and will be referred to as *inter-SOM diversity* ($\text{DIV}_{\text{inter}}$) in the remainder.

The diversity arises over a combination of several factors: We have to discriminate between *implicit* and *explicit*, as well as between *inter*- and *intra*-SOM diversities. The first is the *explicit* diversity forcing impact by NCL on *inter*-SOM level. By the combination of the two resampling methods Bagging and RSM the diversity *between* SOMs is enforced *implicitly*. One interesting feature of the SOM architecture is that a single SOM can be seen as an ensemble itself, where each node represents one classifier and a locally acting function approximator. As a consequence, each SOM has a diversity feature representing the variances between the nodes and I refer to this as the *intra-SOM diversity* ($\text{DIV}_{\text{intra}}$). The third factor is the *implicit* diversity *inside* the SOMs controlled by the width σ of the Gaussian neighborhood function corresponding to the connectivity of nodes. The *intra*-SOM diversity is affected by the *explicit* forcing of the diversity along the ensemble predictors by NCL.

This distinction of diversity boosting factors is outlined in the following Table 4.1.

Diversity	**implicit**	**explicit**
inter-SOM	Bagging, RSM	NCL
intra-SOM	initial conditions, weights, topology, σ	inter-SOM diversity due to NCL

Table 4.1: Diversity boosting factors in the LERRANCO architecture.

The factors promoting diversity in the proposed ensemble architecture are of major influence on the trajectory through the hypothesis space schematically depicted in Figure 4.2. The entire hypothesis space (dashed line) is restricted to the set of accessible hypotheses indicated by the solid line. Every SOM-predictor corresponds to a point in the space $\mathcal{H}$ approximating the true hypothesis with an extended region of coverage (intra-SOM diversity). The traversal through $\mathcal{H}$ is forced by the learning procedure incorporating NCL, such that the aggregation of hypotheses finds the best approximating ensemble hypothesis.

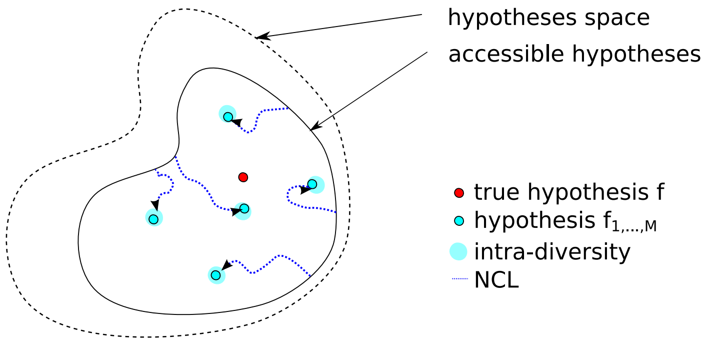


Figure 4.2: Schematic representation of the hypothesis space and the diversity.

4.4 Quantification of Intra-SOM Diversity

To address the issue, if and to which extent the diversity between the SOMs is propagated to the nodes of each single SOM, an auxiliary quantification is required. Under the presumption of an aggregation by the winner-take-all rule (see Equation (2.2) on page 16), only the entire error rate $\mathrm{MSE}_{\mathrm{Test}}$ or $\mathrm{MSE}_{\mathrm{Map}}$ is accessible. The exact amount of *intra*-SOM diversity cannot be assessed directly as proposed for an ensemble to quantify the $\mathrm{DIV}_{\mathrm{inter}}$ (see Equation (3.7) on page 50), unless the requirement of equally distributed weights ($w_i = 1/n_l$) is imposed. This allows to quantify the *intra*-SOM diversity for one single SOM after decomposing the error rate of each SOM according to the ensemble error rate for all nodes $i = 1, \ldots, n_l$ of SOM number m:

$$\mathrm{DIV}^m_{\mathrm{intra}} = \frac{1}{n_l}\sum_{i=1}^{n_l}(\mathcal{C}_{im} - f_m)^2 = \frac{1}{n_l}\sum_{i=1}^{n_l}(y - \mathcal{C}_{im})^2 - (y - f_m)^2 \tag{4.2}$$

$$\mathrm{DIV}_{\mathrm{intra}} = \frac{1}{M}\sum_{m=1}^{M}\mathrm{DIV}^m_{\mathrm{intra}} \tag{4.3}$$

The estimated amount of *intra*-SOM diversity can now be assessed via Equation (4.3) on page 63 representing the variances between the nodes.

4.5 Related Work on SOM Ensemble Learning with NCL

There have been two works recently published on this topic: Minku et al. [2009] demonstrate the usefulness of NCL applied to neural networks with focus on *incremental learning* algorithms. They compare the performance of ensembles of fixed size to an incrementally growing ensemble as well as to Self-Organizing Neural Grove [Inoue and Narihisa, 2003] as an extension of the SOM. In case of the growing networks, initially starting with just one network, a new network is inserted per each incoming data set. The strength parameter γ for NCL was chosen in their study to be 0.390625, based on preliminary executions and uniform over all data sets.
Prudhomme and Lallich [2008] use NCL for ensemble selection and apply SOMs of size 20×20 for knowledge discovery purposes, but without a randomization in their samples while training. A randomization in samples after each epoch while training is the most probable way to overcome the shortcomings of learning the samples by heart. For classification purposes, a population of 100 maps is trained each with a random subset of variables of size $\sqrt{d_{\text{in}}}$. From this population, they employ genetic algorithms to find the subsets of maps, which optimize the ensemble error rate, resulting in best subsets of size $M = 35$ to $M = 45$.

4.6 Conclusion

In this chapter, I propose a neural network ensemble architecture based on Self-Organizing Maps (SOMs), namely the *L*ocal Linear *E*nsembles for *R*egression with *R*esamp-ling *A*nd *N*egative *CO*rrelation Learning (LERRANCO). I introduce the concepts of Negative Correlation Learning (NCL) into the field of SOM ensemble learning. NCL allows to balance between the single network accuracy and the diversity controlled by the cooperation along the ensemble networks thereby dispensing with a sub-local accuracy inside each SOM. The categorization of diversity strategies is adopted and extended in this thesis to distinguish the diversity in SOM ensembles arising over a combination of four factors: The first factor is the *explicit* diversity forcing impact by NCL on *inter*-SOM level. By combining the two resampling methods Bagging and RSM the diversity *between* SOMs is enforced *implicitly.* The third factor is the *implicit* diversity *inside* the SOMs controlled by σ corresponding to the connectivity of nodes. The *intra-SOM diversity* is affected implicitly by the inter-SOM diversity due to NCL.

5 SOMs as Accurate and Diverse Ensemble Predictors

Given the bias-variance-covariance decomposition, diversity is the characteristic factor, since the ensemble error decreases as the diversity increases. If SOMs are suited at all to fulfill the theoretically posed requirements of weak and unstable algorithms as a basis for ensemble learning, will be verified in this chapter.

The characterizations of weak and unstable learners help to understand why and when SOM ensembles improve the prediction performance compared to any of the other VQ-based approximations in input space as basis for Local Linear Maps, like (Growing) Neural Gas and Fuzzy-C-means clustering (see 2.3.1 on page 15). I show which parameters are of major influence on the ensemble error rate and the diversity in SOM ensembles. Using the proposed method the resulting accuracies are comparable to those obtained by other reference architectures. When employing the re-sampling methods, it is expected that every single predictor adapts to different parts of the same learning task.

In this chapter, special consideration is placed on the factors which *implicitly* promote the diversity along the ensemble predictors. The evaluation is subdivided into the factors as recommended by Sharkey [1996]:

Training algorithm The type of VQ-algorithm (GNG, SOM, NG and FCL) the Local Linear Map is based on, is addressed in Section 5.2.

Topology of the nets In Section 5.3, the ensemble size M is varied in order to explore, how many predictors are sufficient.

Training data The application of resampling techniques Bagging and RSM by a variation in the size K of the randomly subsampled feature spaces is examined in Section 5.4.

Initial conditions In Section 5.5, the effect of the grid size of each SOM and of the Gaussian neighborhood width on the ensemble prediction performance is evaluated.

The resulting generalization performances of the proposed ensemble are compared to Random Forests, Multi Layer Perceptron (MLP)-based ensembles as well as to the ν-SVR for several benchmark datasets.

5.1 Evaluation

I use five different benchmark datasets, synthetic as well as real-world ones to allow a general comparison to other ensemble architectures. These datasets are introduced in Section 2.2 on page 13. LLMs based on GNG, NG, SOM and FCL are applied for comparison purposes of the prediction performance. The output of $M = 100$ predictors are combined and aggregated by simple averaging and the reported results in terms of the mean squared error ($\mathrm{MSE}_{\mathrm{Test}}$) for the accounted test sets are averaged over 50 repetitions. The test sets have not taken part in training. Except for the Friedman dataset, 10% of the samples are randomly selected and put aside for testing. The remaining 90% are used to build the predictors. For the Friedman dataset, 2200 samples are newly generated in each iteration. Out of these, 200 randomly selected samples are used to build the predictors and 2000 for testing. All features are centered and normalized (see Section 2.3.4 on page 26) prior to training.

The single predictors are built and combined by a call to LERRANCO($\mathbf{T}$, $M = 100$, $K < d_{\mathrm{in}}$, $\Upsilon = 1$, $\lambda = 0$, VQ(.)). M bootstrap training sets $\mathbf{T}_m$ $(m = 1, \ldots, M)$ are the training bases for the constructed predictors $f(\mathbf{x}, \mathbf{T}_m, \mathbf{K}_m) = f_m(\mathbf{x})$ that are used to form the bagged ensemble predictor $\bar{f}$. Any predictor is constructed with uniform parameters, e. g. VQ("SOM", 2×5, $\sigma = 2.0$). The internal parameters of the LLM algorithm were set to $\epsilon^{\mathrm{in}} = [0.5, 0.01], \epsilon^{\mathrm{out}} = \epsilon^{\mathrm{A}} = [0.3, 0.01]$ decreasing exponentially over $t_{max} = 40$ learning epochs. The prototype vectors are initialized to lie along the first principal components and the samples in $\mathbf{T}_m$ are randomly shuffled while training. The degree of fuzzification in case of FCL is set to $\mu = 2$.

For each of the datasets, three non-linear regression architectures, RandomForests (RF), MLP-based ensembles as well as the ν-SVR are also applied for comparison. The ν-SVR (see 2.3.2 on page 20) generalizes an estimator for the mean of a random variable discarding the largest and smallest samples (a fraction of at most $\nu/2$ of either category). Other parameters that have to be chosen are the regularization parameter C and parameter γ_{RBF} controlling the width of the radial basis kernel function. For the ν-SVR, a grid search runs over $C \in \{e^{-3}, e^{-1}, \ldots, e^{13}\}$ and $\gamma_{\mathrm{RBF}} \in \{e^{-13}, e^{-11}, \ldots, e^{5}\}$ in steps of e^2, and $\nu \in \{0.2, 0.3, \ldots, 0.8\}$ in steps of 0.1 to determine the best parameter set using 10-fold cross-validation. The RF performance is evaluated with $M = 500$ trees grown and $d_{\mathrm{in}}/3$ randomly sampled features as candidates for each node split (see 3.3.1.3 on page 52). The MLP results are given by $M = 10$ with six hidden nodes and a linear kernel.

5.2 Training Algorithm

The applied VQ-based clustering algorithm is supposed to be the key element for the success of ensembles. The VQ-algorithms differ inherently in their objectives of approximating the distribution of the samples in input space. For example, with GNG successively new nodes are inserted, if necessary, according to local error measures. The distribution of the training samples in input space is not inevitably related to the distribution in output space. Hence, the chance of over-fitting may be increased as opposed to a predetermined, fixed number of nodes as for NG, SOMs or FCL. On the other hand, this may be advantageous if different subtasks of the learning task demand and benefit from an adaptation with different numbers of nodes. Larger nets are not expected to be classified as weak learners as the variance is increased.

At first the number of nodes n_l or the size of the grid in case of the SOM is varied, while the remaining parameters are kept fixed with default values. For NG or FCL, a grid search runs over reasonable network sizes $2 \leq n_l \leq 15$. The two-dimensional grid of each SOM is expanded from (1×2) up to (5×7). The optimal network size is determined by the minimum test error rate over 20 repetitions. Next to the number of nodes n_l approximating the training data presented, the parameter grid search runs over the size K of the randomly sampled feature subsets by RSM. Apart from the high-dimensional dataset AAindex, K is varied from about one third of the entire dataset dimensions d_{in} incrementally in steps of 1 up to $K = d_{\text{in}}$. In case of AAindex, the grid search runs over $K \in \{10, 15, 20, 25, 30, 50, 100, 531\}$. Randomly selected variables uniformly distributed build up the training sets $(\mathbf{T}_m, \mathbf{K}_m)$ of size $N \times K$ as re-sampled bases for the regressors $f(\mathbf{x}, \mathbf{T}_m, \mathbf{K}_m) = f_m(\mathbf{x})$. Every ensemble predictor is generated with only a subset of training samples and a feature subset of size K.

We are interested in further insights into the interdependencies between the evaluated parameters of the proposed LERRANCO-architecture combining Bagging and RSM and the resulting test error rate as well as in the diversity. It is worth to recall that the ensemble error rate $\bar{e}$ is given by the difference between the mean error per map and the diversity, denoted by $\bar{e} = (y - \bar{f})^2 = \text{MSE}_{\text{Map}} - \text{DIV}_{\text{inter}}$.

The results are listed in Table 5.1, where the test error is given in terms of MSE_{Test} for the evaluated four LLM-variants based on GNG, NG, SOMs and FCL in direct comparison to RF, MLP-ensembles and the ν-SVR as reference. The best parameter set with minimum MSE_{Test} over grid search parameters (n_l, K) in case of the LLM and over the ν-SVR parameters $(\nu, \gamma_{\text{RBF}}, C)$ is found in Table 5.2. In Table 5.1 per each dataset the corresponding optimal accuracy of the generalizing learning architecture is marked bold. If comparing solely the different LLM-variants, the SOM-based ensemble algorithm apparently outperforms the others. In case of Friedman, Forest-fires and AAindex the LERRANCO regression ensembles based on SOMs yield the best generalization performance compared to every other applied ensemble method. Only the ν-SVR has an advantage over the LERRANCO models for the Friedman and AAindex data. For the other data sets the LERRANCO based on SOMs performs with comparable results to the reference learning architectures. To this end, the evaluations in the remainder of this chapter are given for LERRANCO ensembles based on SOMs.

MSE_{Test}	**LERRANCO based on**						
Dataset	**GNG**	**NG**	**SOM**	**FCL**	**RF**	**MLP****	**ν-SVR**
Friedman	6.2	5.4	5.2	7.6	5.7*	7.7	**5.1**
Boston	11.7	11.7	11.3	14.87	**10.2***	23.2	10.5
Forestfires	4.2	4.1	**4.0**	4.1	4.6	4.3	4.2
NO2	0.28	0.27	0.26	0.26	**0.22**	0.30	0.25
AAindex	1.37	1.05	1.04	1.09	1.26	1.20	**1.03**

Table 5.1: Test error (MSE), averaged over 50 iterations. * Results are taken from Breiman [2001]. ** The MLP results are given by $M = 10$ with six hidden nodes.

	LERRANCO based on						
	GNG	**NG**	**SOM**	**FCL**		**ν-SVR**	
Dataset	$\mathbf{n_l}$	$\mathbf{n_l}$	$\mathbf{n_l}$	$\mathbf{n_l}$	ν	γ_{RBF}	C
Friedman	14	5	2×6	2	0.2	-5	9
Boston	32	5	2×5	4	0.2	-3	3
Forestfires	31	6	1×2	4	0.2	3	11
NO2	32	5	1×4	6	0.2	-3	1
AAindex	23	7	2×2	3	0.6	-11	9

Table 5.2: Optimal parameter set for LERRANCO, i. e. the number of nodes with minimum test error (MSE), averaged over 50 iterations. For GNG, the number of nodes is averaged over the predictors.

5.3 Topology of Networks

In this section, the focus is on the "complexity" of ensembles. The ensemble size M is varied in order to explore how many predictors are sufficient. Given the previously determined optimal grid sizes for the SOM algorithm in Table 5.2, they are kept fixed as constants.

If predictors in an ensemble share common properties of the learning task at hand, i. e. they exhibit only low diversity or high similarity, no gain in accuracy is anticipated when increasing the quantity of ensemble predictors as compared to just one single learner. If the individual ensemble predictors are diverse at a sufficient level, they are expected to cover different sub-tasks and hence a growing number of ensemble predictors is expected to yield an appreciable improvement in ensemble accuracy.

In Figure 5.1 the $\mathrm{MSE}_{\mathrm{Test}}$ is plotted against the number of ensemble predictors $M \in \{1, 2, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ for Friedman, Boston, NO2 and AAindex data. Note that the feature vectors are subsampled by RSM for each ensemble predictor with $K \in \{5, 10\}$ for the Friedman data (a), $K = 8$ for the Boston (b), $K = 5$ for the NO2 data (c) and $K = 25$ in case of the AAindex data (d), as denoted in the header title of the plots. Having trained LERRANCO ensembles on the Friedman data with all $K = 10$ available features, the dashed curve in (a) indicates that a growing number $M > 1$ of predictors has little impact on the accuracy in terms of $\mathrm{MSE}_{\mathrm{Test}}$. This is explained by the low diversity between the individual predictors trained with only small deviations in the training sets induced by Bagging. In contrast to that, if K is set to $K < d_{\mathrm{in}}$, the error rate curves of $\mathrm{MSE}_{\mathrm{Test}}$ with regard to M show that the most significant reduction is achieved with ensemble sizes of up to $M = 20$. Beyond this point, the slope of $\mathrm{MSE}_{\mathrm{Test}}$ is still negative with a growing number of ensemble predictors. Apart from some statistical deviations, the lowest ensemble $\mathrm{MSE}_{\mathrm{Test}}$ is achieved for $M \to 100$.

Therefore, the number of ensemble predictors $M > 1$ is necessary but not sufficient for ensembles succeeding over single predictors. One can rather deduce gain in diversity, e. g. by RSM, as premise for improving the accuracy of ensembles with $M \gg 1$.

5.4 Training Data

With Bagging, an implicit diversity enforcing function approximation is applied. Traversing through the hypothesis space, only one of the targets accuracy and diversity can be tuned, which is in contrast to NCL, where one tries to find explicitly the "right" balance between these targets. The amount of diversity, which is brought into the ensembles only due to training with the bootstrapped samples, is not yet considered.

The effect of the diversity implicitly promoting method on the $\mathrm{MSE}_{\mathrm{Test}}$ is evaluated. To approve Bagging as a suitable training set generation method, the resulting ensemble generalization capabilities are compared to those obtained *without* bootstrapping,

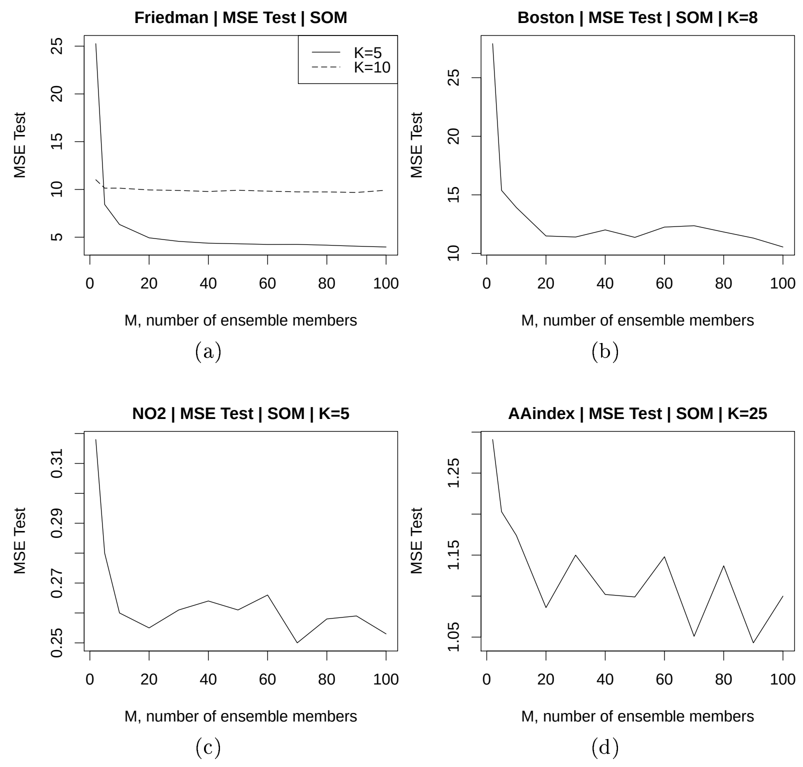


Figure 5.1: $\mathrm{MSE}_{\mathrm{Test}}$ for LERRANCO models based on SOMs when varying number of ensemble members M.

i. e. $\mathbf{T}_m = \mathbf{T}_j,\ \forall j, m \in \{1, \ldots, M\}$. The resulting performances are given in Figure 5.2 on page 72, where the $\mathrm{MSE}_{\mathrm{Test}}$ is shown with regard to a varying size K of randomly selected subspaces for each ensemble predictor. The dashed lines correspond to the ensemble versions trained without bootstrapping. Even if the non-bootstrapped ensembles error rate curves are close to those including bootstrapping in some cases, the application of Bagging induces a clear advantage due to the gain in diversity.

5.4.1 Size of Random Subspaces

As stated in Section 5.3 on page 69, the diversity along the ensemble predictors and the implicitly forced negative correlation in errors is a premise for the success of ensembles. One substantial component of the LERRANCO architecture is the RSM, which forces the diversity by sub-sampling the features available for each ensemble member. To address the issue of the relation between MSE_{Test} and the number K of randomly selected subsets of features, the following Figure 5.2 is given. The relationship of K and the MSE_{Test} for each of the datasets and ensembles of size $M = 100$ is analyzed. Apart from the NO2 data, ensembles have considerable advantage from splitting the entire task into several sub-tasks by RSM. The aggregation of part-adapting predictors leads to a better adaptation and a lower MSE_{Test} in general. The ensembles error rate curves exhibit the lowest MSE_{Test} for $K \ll d_{\text{in}}$. If the number K of randomly sampled features is too small though, the individual predictors are no longer capable to extract the relevant information for modeling the sub-task appropriately.

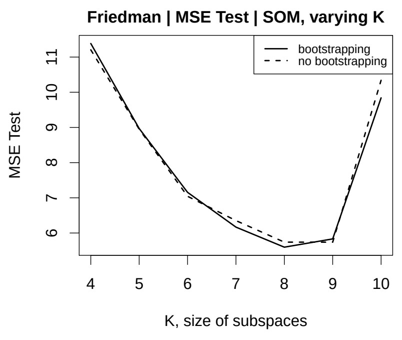
5.5 Initial Conditions

The application of the SOM requires several parameters to be set and some difficulties may be encountered from tuning them. Among the grid size, the type of initialization of prototype vectors, the number of epochs, the initial and final value of learning rates as well as of the Gaussian neighborhood width, the size of the parameter space is even enlarged when applying the LLM. In the context of EL, the question crops up, if the complexity of the search over the parameters is raised. A fine-tuning of each parameter individually for each SOM inside an ensemble would mean an infeasible effort needed in practice. Actually I can show that the demand for individually fine-tuned parameters fades into the background with ensembles of SOMs. Moreover, the SOMs profits from the necessary requirement of *weak* learners in EL. Improved prediction accuracies are obtained with the parameters set according to rules of thumb. This will be illustrated for the two main parameters, the grid size and the Gaussian neighborhood width, σ, and their impact on MSE_{Test} and the diversity.

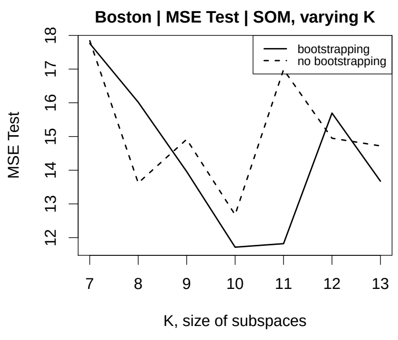
5.5.1 Grid Size

To determine the effect of grid sizes, the number of nodes of each SOM is varied while the other parameters of the SOM as well as of the architecture are kept fixed. The call to LERRANCO is done by ($\mathbf{T}$, $M = 25$, $K < d_{\text{in}}$, $\Upsilon = 1$, $\lambda = 0$, VQ("SOM", n_l, $\sigma = 2.0$)). Reasonable network sizes are tried and the two dimensional SOM-grid is expanded from (1×2) up to (5×7). The optimal network size is determined by the best test error rate over 10 iterations. Evaluation is done over varying K and grid sizes $n_l = 2 \times i, i = 1, \ldots, 6$ as in Figure 5.3 as an example given for the Friedman and AAindex data. Best ensemble accuracy is obtained by $K = 9$ and $n_l = 2 \times 5$ in case of Friedman data (a) and by $K = 50$, $n_l = 2 \times 2$ for AAindex data (b). Note that σ was set to the initial value of 2.0 for all $M = 25$ ensemble predictors uniformly.

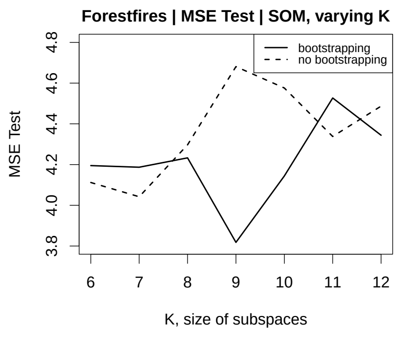
Increasing the size of the networks leads to a loss in generalization performance, since every net gets too specialized to the presented subtask. As expected, powerful



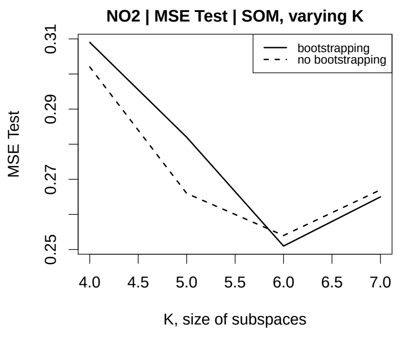
(a)



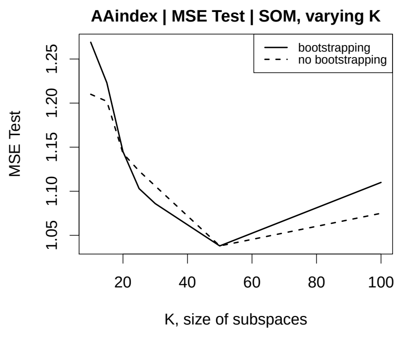
(b)



(c)

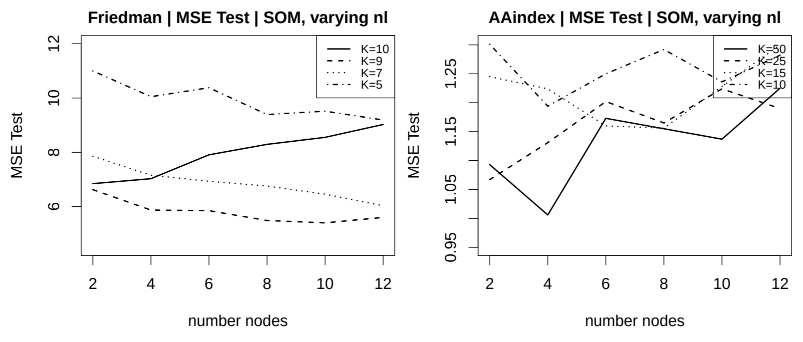


(d)



(e)

Figure 5.2: $\mathrm{MSE}_{\mathrm{Test}}$ for LERRANCO models based on SOMs when varying size K of randomly selected subspaces for each ensemble member.



(a) (b)

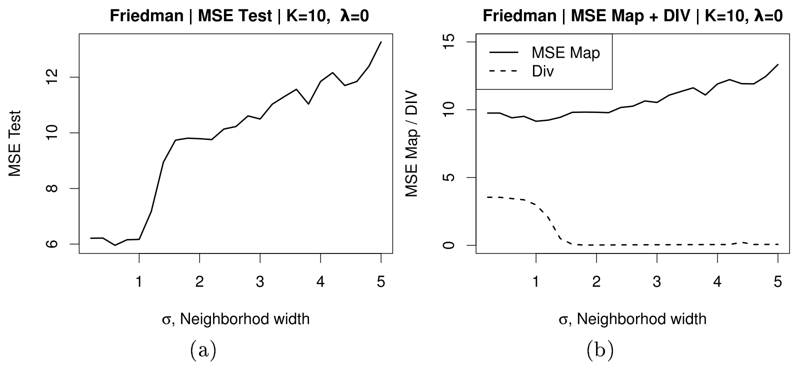
Figure 5.3: Test error rate ($\mathrm{MSE}_{\mathrm{Test}}$) when varying K and the number of nodes n_l, i. e. grid size for SOMs. Best ensemble accuracy is obtained by $K = 9$ in case of the Friedman data and with $K = 50$ for the AAindex data. Note that σ was set to an initial value of 2.0 for all ensemble predictors uniformly.

ensembles are generated with small nets of low complexity, i. e. a low number of nodes ($n_l = 3$ up to ≈ 10). At least for the small data sets, roughly speaking, this means that the gain in diversity is abrogated by the increase in $\mathrm{MSE}_{\mathrm{Map}}$.

5.5.2 Gaussian Neighborhood Width

To apportion the effect of σ to the $\mathrm{MSE}_{\mathrm{Test}}$, the initial connectivity of nodes σ is varied from 0.4 to 5.0. σ decays from σ_i to σ_f exponentially with the training steps. Towards the end of the training process, only slight locally adaptations are done according to the small learning rate and a variation of the final σ_f has low effect.

In Figure 5.4 the results are given for the dataset Friedman (and AAindex) in terms of $\mathrm{MSE}_{\mathrm{Test}}$ for $K = 10$ ($K = 25$). For the Friedman data, there exist clear minima with $\sigma \approx 0.4, \ldots, 1.0$, where the error rates $\mathrm{MSE}_{\mathrm{Test}}$ (a) and $\mathrm{MSE}_{\mathrm{Map}}$ (b) show its lowest values. This is explained by the increase in diversity between the ensemble predictors (b). For $\sigma > 1.5$ the $\mathrm{MSE}_{\mathrm{Map}}$ basically constitutes to the $\mathrm{MSE}_{\mathrm{Test}}$. In case of AAindex data (c), a relation between $\mathrm{MSE}_{\mathrm{Test}}$ and σ is not apparent. With $\sigma > 2$, a state of saturation is reached due to the small number of nodes supplied in this case (2×2) and hence an initial $\sigma = 2.0$ should be a suitable choice.



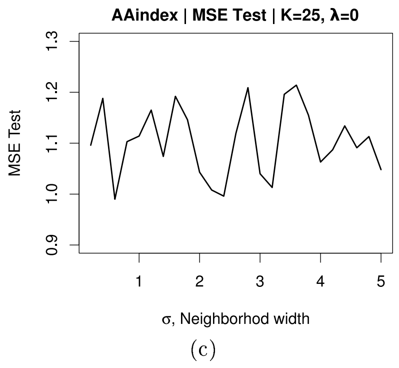


Figure 5.4: Test error rate ($\mathrm{MSE}_{\mathrm{Test}}$) when varying intra-SOM diversity by initial width of Gaussian neighborhood function $\sigma \in [0.4, 5]$ if $\lambda = 0$, i. e. no NCL.

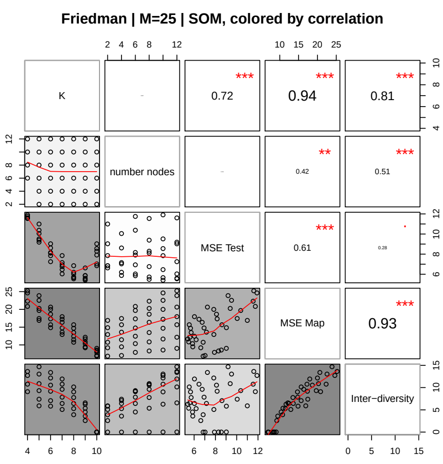
5.6 Discussion

With the concept of EL, ensembles of SOMs provide considerable advantage from the aggregation of part-adapting structures. If and only if the ensemble predictors are diverse enough, the MSE_{Test} may be improved when adding more ensemble members. It turns out that a quantity of $M = 100$ is a good compromise between the resulting ensemble test error rate and the effort needed for additional storage capacities. The drawbacks arising from the increased capacities needed to store the M ensemble predictor models, as compared to single predictors, are reduced by actually lower capacities of each individual ensemble predictor.

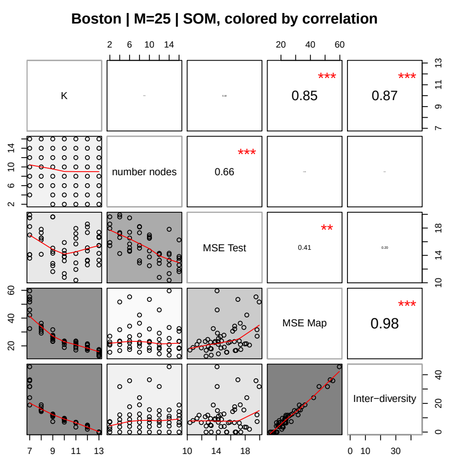
By the application of the re-sampling methods, Bagging and RSM, the size or complexity of the training sets are reduced significantly, see Figure 5.2 on page 72. An exact quantitative evaluation of the complexity and of the computation time for the LERRANCO architecture is discussed later in Section 6.8 on page 95. The effective reduction in the number of SOM internal parameters is very attractive in practice. The computational effort is reduced dramatically with the parameters set uniformly over all ensemble predictors and the prediction capabilities are enhanced at the same time. The critical parameters left are merely the grid size and the number of randomly sampled subspaces used for training. If the number of randomly sampled features K is too small though, the predictors are no longer capable of extracting the relevant information for modeling the particular sub-task at hand. Subject to the restrictions in that situation, the application of Bagging, i. e. leaving out approximately 36% of the samples, even more stresses the lack of information as well as the effect of noise.

Each model's complexity is tweaked to a low number of nodes, i. e. small grid sizes, conform to the postulation of weak and unstable learners (see Figure 5.3). With a small width of neighborhood function, the connectivity between the nodes inside of each SOM is reduced and the individual nodes are forced to specialize locally. This way, the ensemble accuracy is improved by forcing the diversity along each individual predictor (see Figure 5.4).

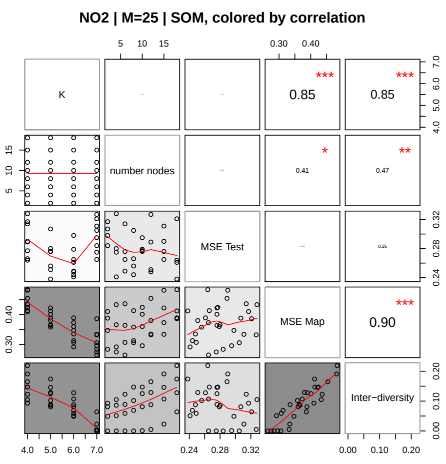
In Figure 5.5 on page 76 scatterplot-matrices are depicted with regard to the parameters K and n_l. The resulting MSE_{Test}, MSE_{Map} and $\text{DIV}_{\text{inter}}$ are reported for the datasets Friedman, Boston, AAindex and NO2. Over all evaluated datasets, roughly speaking, the size K of subspaces randomly sampled is a counterpart to the number of nodes n_l with regard to MSE_{Map} and the diversity among the ensemble predictors. When forcing the locality of the approximating predictors by *lowering* K, the increase in $\text{DIV}_{\text{inter}}$ comes at the cost of the sub-local accuracy in terms of MSE_{Map}. The opposite holds true for the number of nodes n_l supplied: By a *higher* number of nodes, the commonality along the predictors is reduced, i. e. $\text{DIV}_{\text{inter}}$ is boosted, when forcing the local specialization of approximating SOMs. For an improvement of ensemble performance, the parameters K and n_l have to be balanced against each other in order to compensate the deterioration of MSE_{Map} for an increase of diversity among the ensemble predictors. The specialization of the ensemble predictors locally adapting to



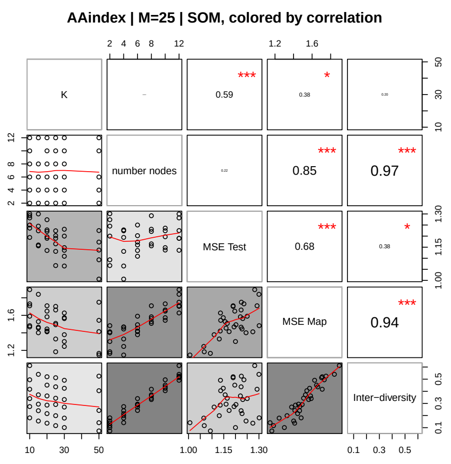
(a)



(b)



(c)



(d)

Figure 5.5: Scatterplot-matrices for datasets Friedman, Boston, NO2 and AAindex. The terms MSE_{Test}, MSE_{Map} and $\text{DIV}_{\text{inter}}$ are recorded regarding the size K of features sampled and the number of nodes n_l each SOM is trained with. The fields of the matrix are colored with gray values, reflecting the correlation between the according measures, K and n_l. Correlation values $\mathrm{r} \in [0, 1]$ are mapped to gray values (white (no correlation) to light gray($|\mathrm{r}| = 1$)). The upper panels show the exact r. The lower K, the higher is the resulting $\text{DIV}_{\text{inter}}$. The increase in $\text{DIV}_{\text{inter}}$ comes at the cost of the sub-local accuracy in terms of MSE_{Map}.

the sub-tasks facilitates the ensemble to achieve a higher overall generalization ability.

5.7 Conclusion

The output of several LLMs based on SOMs is aggregated to build powerful part-adapting learning machines. This concept has been shown to increase the predictive power over single base learners. The proposed ensemble architecture LERRANCO is applied to different benchmark data sets and compared to Random Forests, MLP-based ensembles as well as to the ν-SVR regarding the generalization performance with comparable results. It turns out that LLM ensembles supplying SOMs are appropriate and succeeding ensemble predictors.

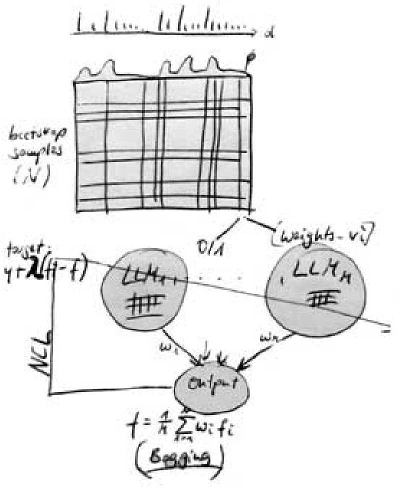
In this Chapter, a special focus is placed on the factors, which *implicitly* force the diversity along the ensemble predictors. It is explored how the ensemble prediction performance is affected by these factors, which include the type of training algorithm, the topology of ensembles, the resampling method for the training set construction and the initial conditions like the grid size. Powerful ensembles are shown to be generated with small nets of low complexity.

The number of ensemble predictors $M \gg 1$ is necessary but not sufficient for improving the accuracy of ensembles over single base learners. A quantity of $M = 100$ is a good compromise between the resulting ensemble test error rate and the effort for additional storage capacities. The usefulness of combining Bagging with Random Subspace is demonstrated in terms of the diversity and the MSE_{Test}. It turns out that the factors mostly increasing the diversity and predictive power are the supplied VQ-method, among the number of variables building Random Subspaces, followed by the number of ensemble members. A small width of neighborhood can be identified to be a significant factor in order to boost the diversity implicitly inside each SOM predictor.

When aggregating part-adapting structures, the gain in diversity should be the main focus of attention as a premise for successful ensembles. The diversity along the ensemble predictors exhibits a high correlation to the mean error per predictor (MSE_{Map}). Consistent to the observations made by Zenobi and Cunningham [2001], the most successful ensembles are found each presented only a subset of features. Therewith, the predictors are forced to locally specialize or to discriminate in these local regions of the input space. These characterizations of *locally specialized learners* help to understand how the SOM ensembles succeed.

The potential of SOMs to Ensemble Learning is mainly exploited by an effective reduction in the models' complexity according to the number of parameters, the size of each SOM and the dimensionality of input space for each SOM.

6 Negatively Correlated SOM Ensembles



Up to now, I examined SOM ensembles acting independently of each other. The single predictors successfully are forced to follow different trajectories implicitly by the re-sampling methods Bagging and RSM. With the LERRANCO architecture by employing NCL, a shift is performed towards interlinking individual learning stages including a term penalizing correlation in errors. NCL allows to balance between the single network accuracy and the diversity controlled by the cooperation of predictors, thereby dispensing with a sub-local accuracy for a higher overall generalization ability. Boosting the diversity *explicitly* this way, the formed bagged predictor $\bar{f}$ ranges from an aggregation of independent predictors to a fully connected network as a single, complex learning unit. Focusing on the SOMs inside an ensemble, the question arises, how far the interplay between SOMs and within SOMs succeeds. To this end, an explicit distinction between the inter- and intra-SOM diversity is carried out.

6.1 LERRANCO Evaluation

The prediction performance of the proposed SOM-based ensemble architecture LERRANCO is evaluated as described in Section 5.1 on page 66. For each of the benchmark datasets, also two complementary non-linear regression architectures, RF as well as ν-SVR are applied for comparison. The RF performance is evaluated with $M = 500$ trees grown and $d_{\text{in}}/3$ randomly sampled variables as candidates for each node split.

Unless otherwise stated, the parameters of LERRANCO are set uniformly over all single ensemble predictors to LERRANCO($\mathbf{T}$, $M = 100$, $K < d_{\text{in}}$, $\Upsilon = 1$, $\lambda = 1$, VQ(.)). Any LLM predictor is constructed with uniform parameters of VQ("SOM", n_l, $\sigma = 2.0$). The internal parameters of the LLM algorithm were set to $\epsilon^{\text{in}} =$

$[0.5, 0.01], \epsilon^{\text{out}} = \epsilon^{\text{A}} = [0.3, 0.01]$ and $\sigma = [2.0, 0.4]$ decreasing exponentially over $t_{max} = 40$ learning epochs. In preliminary experiments (see Table 5.2 on page 68), I determined LERRANCO performing best, if the training is based on SOMs with n_l nodes according to the grid sizes of 2×6 for Friedman, 2×5 for Boston data, 1×2 for Forestfires, 1×4 for NO2 and 2×2 for the AAindex dataset. M resampled training sets $(\mathbf{T}_m, \mathbf{K}_m), m = 1, \ldots, M$, are the training bases for the constructed predictors. The outputs of the $M = 100$ predictors are combined and aggregated by simple averaging.

The presented results are limited to the $\bar{f}$-penalty, (Equation (3.9) on page 53) corresponding to $\Upsilon = 1$ to restrict the upper bound for NCL. $\lambda \in [0.0, 1.0]$ is sampled coarsely to intervals of 0.1. In the regions of special interest, i. e. $\lambda \in [0.9, 1.0]$, the search is refined to intervals of 0.01. The generalization performance is measured in terms of mean squared test error (MSE_{Test}) for the accounted test sets, which have not taken part in training.

6.2 Results

All of the reported results of the prediction performance of the SOM ensemble architecture LERRANCO and the reference regression architectures are averaged over 50 repetitions. The evaluation of the prediction performance in terms of MSE_{Test} is summarized in Table 6.1 for the corresponding optimal parameters K, λ and σ. The results in Table 6.1 marked with a $(^{+})$ are taken from Breiman [2001], who trained RFs on transformed versions of the Friedman and the Boston dataset. He proposed to transform the original training data to a randomly linear combined feature space of higher dimension ($d'_{\text{in}} = 25$). This way, every newly derived variable is made up of two others, such that e. g. variable $\tilde{\mathbf{x}}^3$ is combined by $0.2 \cdot \mathbf{x}^5 + 0.5 \cdot \mathbf{x}^2$, where d denotes a column of the training set $\mathbf{T}$ with N samples, i. e. $\mathbf{x}^d = (x_{1d}, \ldots, x_{Nd})^T$ with $d \in \{1, \ldots, d_{\text{in}}\}$.

Without NCL (i. e. $\lambda = 0$), the LERRANCO ensembles based on SOMs outperform RFs regarding test error rate except for the Boston and NO2 data. With NCL ($\lambda > 0$), the generalization performance is further improved and hence forcing a fully connected ensemble. LERRANCO exhibits comparable or even superior generalization ability to RF or ν-SVR, while only in case of NO2 dataset the RF MSE_{Test} is smaller. The resulting scatterplots of LERRANCO predictions on the evaluated (benchmark) datasets and the corresponding estimated density distributions are shown in Figure 6.1. The densities were estimated by Fast Fourier Transform and a Gaussian kernel density estimation (for details see **?**). These predictions are performed with the optimal parameters of (K, λ), which turn out to be $(5, 1.0)$ in case of the Friedman data, $(8, 0.97)$ for the Boston, $(9, 0)$ for Forestfires, $(6, 0.9)$ for NO2 and $(25, 0.9)$ for the AAindex data.

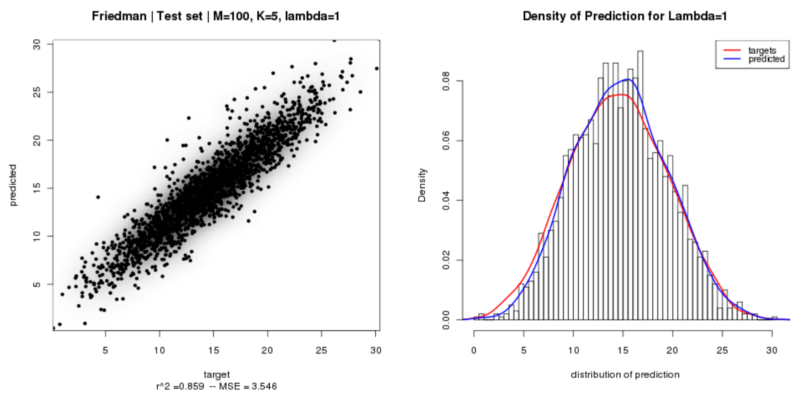
Next to NCL, the size $K < d_{\text{in}}$ of the randomly sampled feature subsets set appropriately has a major effect on the decrease in MSE_{Test}. In Figure 6.2 the exact

	LERRANCO			**RandomForest**		**ν-SVR**
MSE$_{\text{Test}}$	$\lambda = 0$	$\lambda \to 1$	p-value			
Friedman	5.2	**3.9**	***/***	7.1	(5.7^{+})	5.1
Boston	11.3	**10.1**	/	10.7	(10.2^{+})	10.5
Forestfires	**4.0**	4.1	*/	4.6		4.2
NO2	0.26	0.24	·/	**0.22**		0.25
AAindex	1.04	**1.02**	*/	1.16		1.03

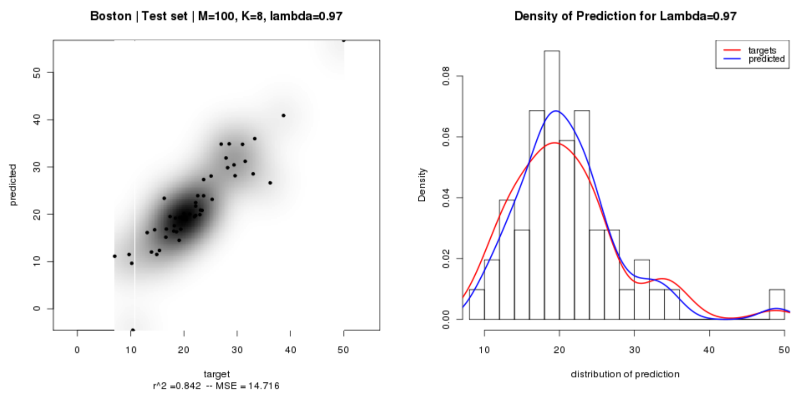
Table 6.1: Mean Squared Test error (MSE$_{\text{Test}}$), averaged over 50 iterations for LERRANCO, RF and ν-SVR. Results marked with ($^{+}$) are taken from Breiman [2001] with randomly linear combined feature space of higher dimension ($d'_{\text{in}} = 25$). The corresponding best results are listed for LERRANCO ensembles of size $M = 100$, $K < d_{\text{in}}$, $\lambda = 0$ (no NCL) or $\lambda \to 1$. A two-sided t-test is applied to show the statistical significance of results regarding LERRANCO compared to (RF/ν-SVR). The p-value (confidence level of 0.95) is indicated by (*/*), where the symbols ("***", "**", "*", ".", " ") correspond to cut-points $(0, 0.001, 0.01, 0.05, 0.1, 1)$.

resulting error rates for the datasets Friedman, Boston, Forestfires, NO2 and AAindex are given when varying values of strength parameter λ with NCL and the size K of the randomly selected subspaces. The LERRANCO ensemble models exhibit a sharp decrease in MSE$_{\text{Test}}$ when boosting the diversity explicitly by an increased λ over all datasets, except for the Forestfires dataset. Thus, the ensembles based on SOMs demonstrate a pronounced improvement of the generalization performance with NCL. Especially the prediction of the Friedman test sets benefits from NCL in combination with RSM: While a priori only five of the available $d_{\text{in}} = 10$ variables are used to define the target value $\mathbf{y}$, the smallest MSE$_{\text{Test}}$ is actually found with $K = 5$ and $\lambda = 1$, showing an enormous impact of boosting the diversity explicitly. The sole exception from that improvement due to NCL is the dataset Forestfires. LERRANCO ensembles based on SOMs with grid sizes of 1×2 each outperform every other diversity encouraging approach.

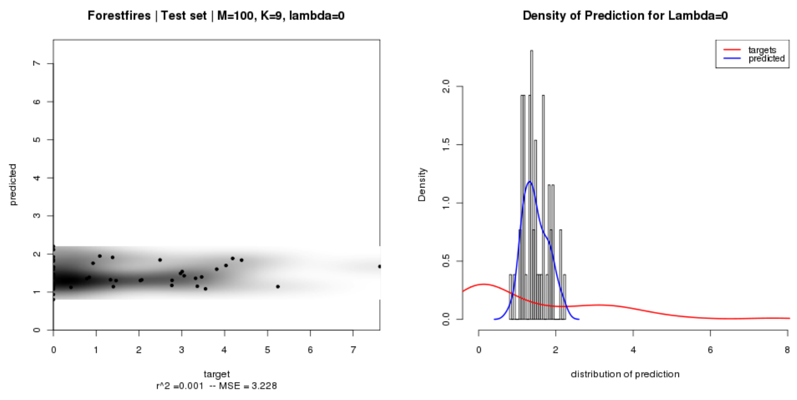
The influence of NCL-parameter λ regarding *inter-SOM* diversity is discussed in more details in the next Section. The choice and the role of σ affecting the intra-SOM diversity is analyzed in Section 6.4.



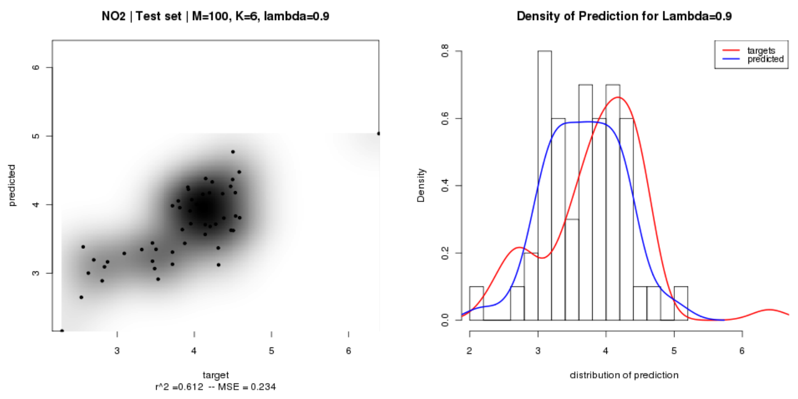
(a) Friedman



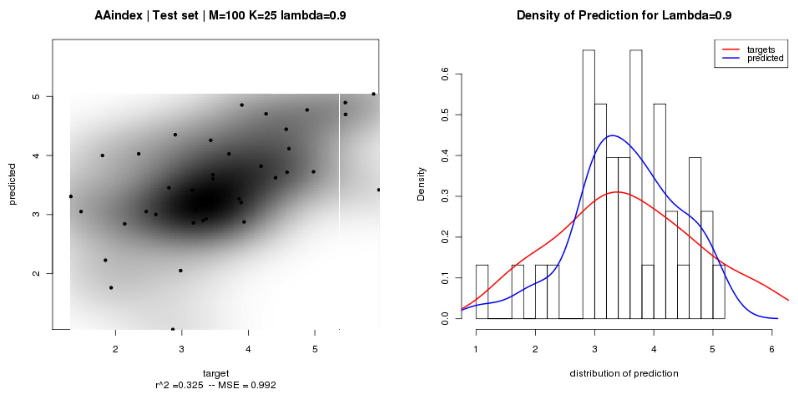
(b) Boston



(c) Forestfires



(d) NO2



(e) AAindex

Figure 6.1 *(previous page)*: Scatterplots of evaluated benchmark datasets predicted by LERRANCO, i. e. the test target values $\mathbf{y}$ are plotted as points against the predicted vectors for Friedman, Boston, Forestfires, NO2 and AAindex data. The right columns show the corresponding estimated density distributions.

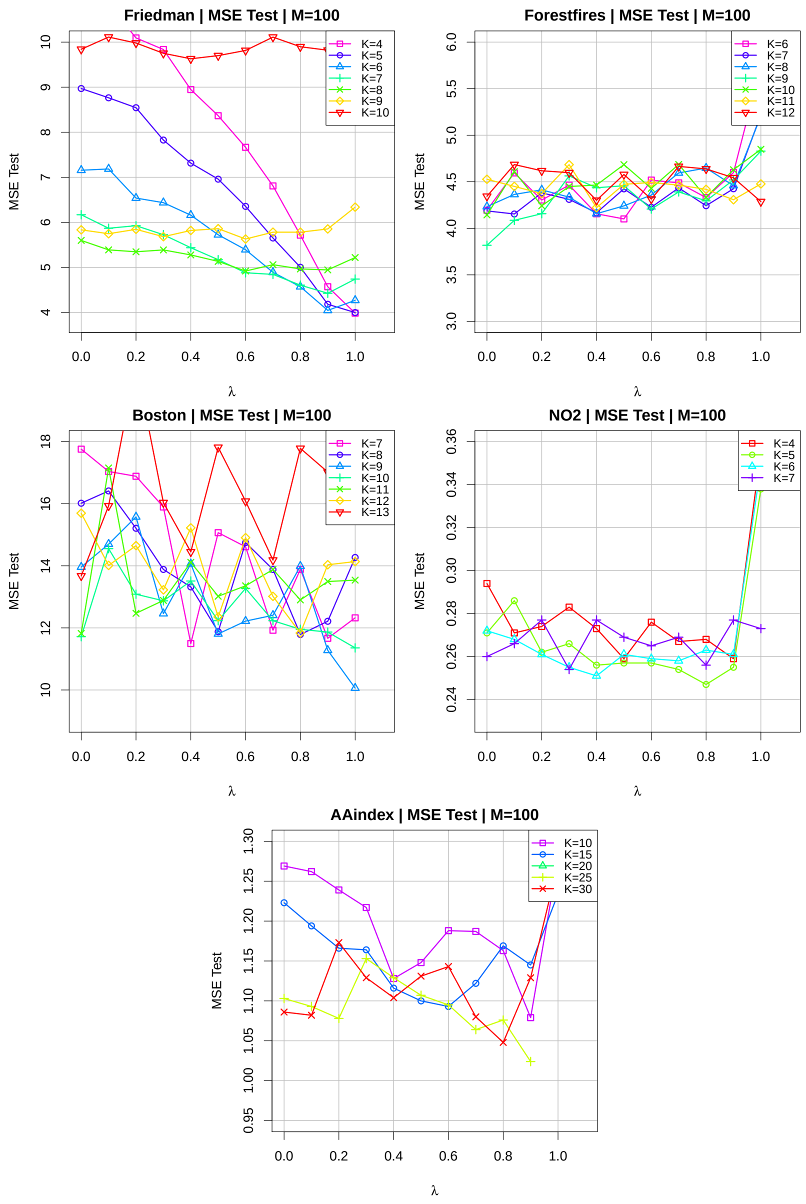


Figure 6.2: Test error rate (MSE_{Test}) when varying λ as well as the size K of uniformly selected subspaces plotted for Friedman, Boston, Forestfires, NO2 and AAindex dataset.

6.3 Inter-SOM Diversity

With $\lambda > 0$, the positive correlations in the errors are penalized hence diverse ensemble members are enforced. In fact, boosting the NCL strength i. e. $\lambda \to 1$ significantly improves the prediction performance for all datasets except for the Forestfires data. This states a high correlation between the quantified diversity and the reduction of MSE_{Test}. In Table 6.2, the exact correlation coefficients between the quantified *inter*-SOM diversity ($\text{DIV}_{\text{inter}}$) and λ (MSE_{Test} respectively) are denoted. The Pearson correlation coefficients (r) corroborate the theory behind NCL of forcing the diversity of the single ensemble predictors except for the NO2 dataset. Here, the low correlation between diversity and MSE_{Test} appears due to a high variation in MSE_{Test} with λ.

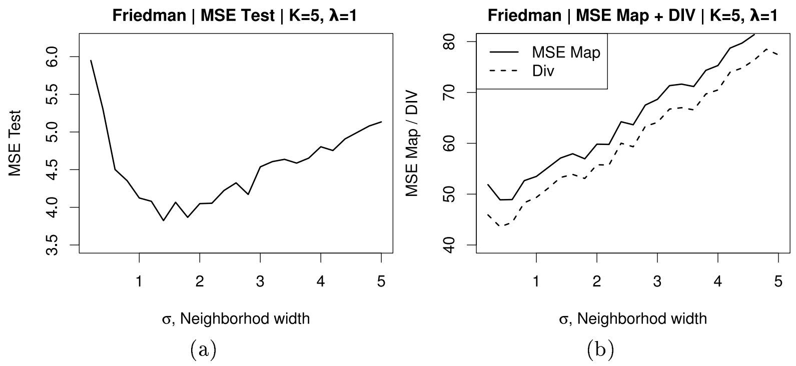
Correlation r	**r(λ,$\text{DIV}_{\text{inter}}$)**	**r($\text{DIV}_{\text{inter}}$, MSE_{Test})**
Friedman	0.83	−0.84
Boston	0.80	−0.67
Forestfires	0.75	0.43
NO2	0.79	−0.06
AAindex	0.81	−0.63

Table 6.2: Table of Pearson correlation coefficients r between $\text{DIV}_{\text{inter}}$ and NCL strength parameter λ or MSE_{Test} showing a high absolute correlation. The higher λ, the higher is the reduction of MSE_{Test} due to the boosted diversity between the individual ensemble predictors.

When enforcing the inter-SOM diversity along the ensemble predictors, the question crops up, to which extent this effects the nodes inside each single SOM. In the next subsection, the issue of the relationship between the *inter*-SOM and *intra*-SOM diversity is addressed.

6.4 Intra-SOM Diversity

The *intra*-SOM diversity ($\text{DIV}_{\text{intra}}$) arises mainly over the combination of two factors: The first factor is the explicit diversity forcing impact by NCL, while the implicit diversity inside the SOMs is not under control of NCL, but controlled by the width σ of the Gaussian neighborhood function. To apportion the effect of this *intrinsic intra*-SOM diversity to the test error rate, I vary the initial connectivity of nodes σ from 0.4 to 5.0. In Figure 6.3 the results are given for dataset Friedman (a),(b) (and AAindex in (c)) in terms of MSE_{Test} for $K = 5$ ($K = 25$) and $\lambda = 1$ ($\lambda = 0.9$). For the Friedman data, there exist clear minima with $\sigma \approx 1.0, \ldots, 2.5$, where test error rates show its lowest values (a). The inter-SOM diversity $\text{DIV}_{\text{inter}}$ and MSE_{Map} are apparently related to σ (b). For the AAindex dataset, the generalization error rate is mostly far from being influenced by the connectivity of nodes (c). With $\sigma > 2$, a state of saturation is reached due to the small number of nodes supplied in this case (2×2).



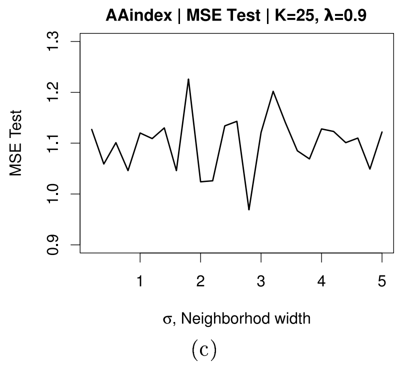


Figure 6.3: Test error rate ($\mathrm{MSE}_{\mathrm{Test}}$) when varying intra-SOM diversity by width of Gaussian neighborhood function $\sigma \in [0.4, 5]$ for ensembles of size $M = 100$

To explore, if and to which extent the $\mathrm{DIV}_{\mathrm{inter}}$ is propagated to the nodes within each single SOM, the $\mathrm{DIV}_{\mathrm{intra}}$ for one single SOM is assessed according to Equation (4.3) on page 63. The estimated amount of $\mathrm{DIV}_{\mathrm{intra}}$ represents the variances between the nodes. A positive absolute relation between λ, the *inter*- and *intra*-SOM diversity can be figured out as shown in Figure 6.4. The given scatterplot-matrices present the correlations between λ, $\mathrm{MSE}_{\mathrm{Test}}$, $\mathrm{MSE}_{\mathrm{Map}}$, $\mathrm{DIV}_{\mathrm{inter}}$ as well as $\mathrm{DIV}_{\mathrm{intra}}$ for the evaluated LERRANCO ensembles of size $M = 100$. The correlation between each pair of combination is reported by the exact correlation values r in the upper panels. In the lower panels, the r values are mapped to gray values. White panels correspond to no correlation and dark gray panels indicate highly correlated combinations. As a consequence of boosting the diversity on the inter-SOM level, the $\mathrm{MSE}_{\mathrm{Map}}$ is increased as the term directly connecting $\mathrm{DIV}_{\mathrm{inter}}$ and $\mathrm{DIV}_{\mathrm{intra}}$. The evaluation of Boston data regarding intra-SOM diversity exempts from the positive correlation between λ and $\mathrm{DIV}_{\mathrm{intra}}$, as well as the $\mathrm{MSE}_{\mathrm{Test}}$ of Forestfires data does.

6.5 Dynamics of Boosted Negatively Correlated SOMs

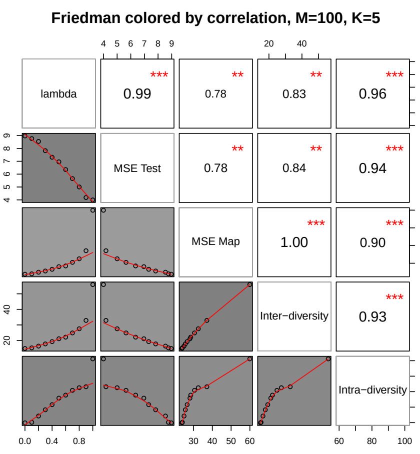
In contrast to many other learning architectures, the SOM delivers a deeper insight into the operations and approximations of the input and output space. It facilitates the inspection of the prototype vectors, the error rates while training, the distributions, the locally trained linear mappings and so on. In general, ensemble architectures provide no such way and lack more or less of comprehensibility. Nonetheless, an assessment of the relevance of features is feasible as mentioned for an example for RFs in Section 2.4.1 on page 37.

But in context of NCL, several questions are left open about the dynamics among the predictors. Liu and Yao [1999] as well as Brown [2004] give thankful impulsions and investigations on how to include the quantified diversity term in the learning process. Not only experimentally, but also from a theoretically point of view, NCL has proven to be a valuable concept in ensemble learning. What actually happens inside the ensemble is to some extend reflected by the proposed quantification of $\mathrm{DIV}_{\mathrm{intra}}$. However, the effective dynamics on the interplay among the ensemble predictors over the progress of training and over the strength of NCL is not yet considered. In the following, it will be explored how the distributions of $\mathrm{MSE}_{\mathrm{Test}}$ and the internal diversity measures, $\mathrm{DIV}_{\mathrm{inter}}$ and $\mathrm{DIV}_{\mathrm{intra}}$, among the predictors are affected by λ.

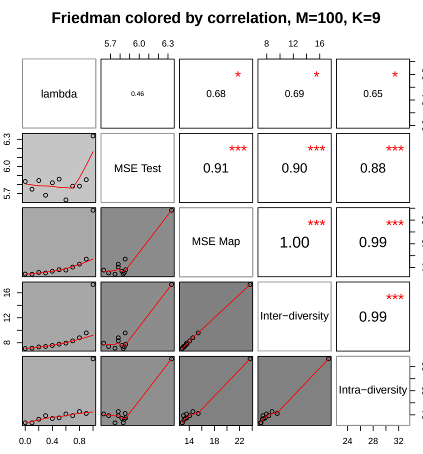
6.5.1 Dynamics in Time

Eastwood and Gabrys [2007] exploit the dynamics and the behavior of predictors of MLP-type when applying NCL. They derive an optimal value for the NCL-strength parameter in terms of $\gamma^* = \frac{1}{2}(1-\frac{1}{M})^{-1}$. This is a varied expression of the upper bound $\gamma = 2 \cdot \lambda(1 - \frac{1}{M})$ [Brown et al., 2005a] corresponding to $\lambda = 1$. The upper bound of λ does not protect against over-fitting, because the introduced complexity may not be appropriate for the particular problem [Eastwood and Gabrys, 2007]. They captured the deviance when lambda exceeds the upper bound and the learning exaggerates the variance in predictors as forced by $\lambda > \lambda_{\mathrm{upper}}$.

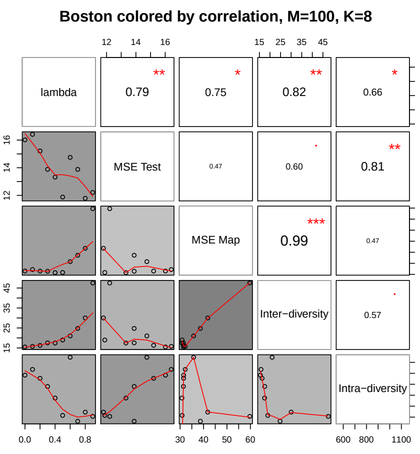
Similar to their analysis, I report the evaluations of the internal error measure $\mathrm{MSE}^{m}_{\mathrm{Map}}$ recorded while the training proceeds for the Friedman dataset as an example



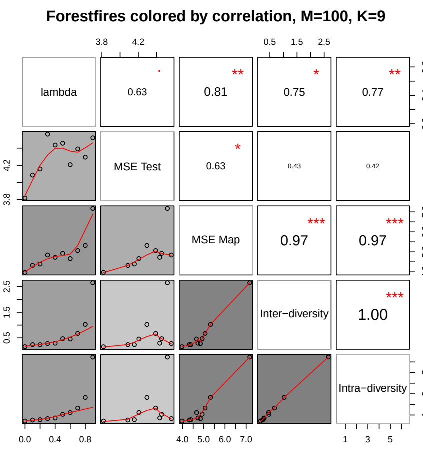
(a) Friedman, $K = 5$



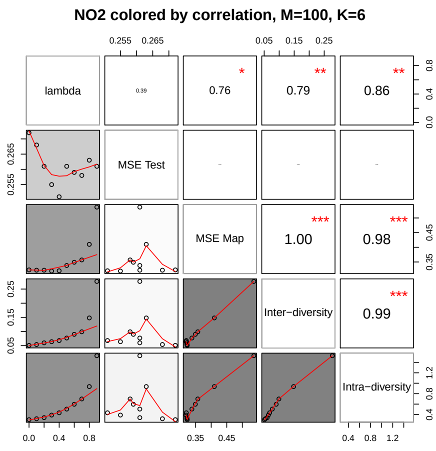
(b) Friedman, $K = 9$



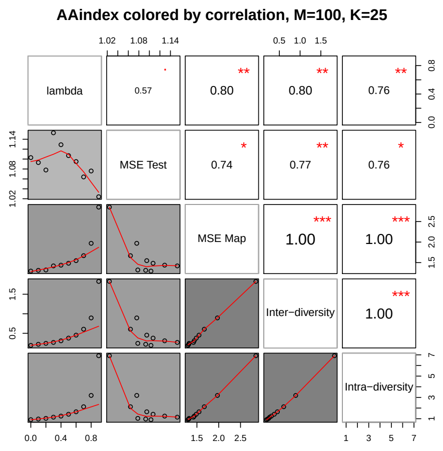
(c) Boston



(d) Forestfires



(e) NO2



(f) AAindex

Figure 6.4 *(previous page)*: The terms MSE_{Test}, MSE_{Map}, $\text{DIV}_{\text{inter}}$ and $\text{DIV}_{\text{intra}}$ are recorded regarding NCL-parameter $\lambda \in [0,1]$ for all datasets. The lower panels of the matrix are colored with gray values, reflecting the correlation between the according measures or λ. Correlation values $\text{r} \in [0,1]$ are mapped to gray values (white (no correlation) to dark gray($|\text{r}| = 1$)). The upper panels show the exact r. The higher λ, the higher is the reduction of MSE_{Test} due to an increasing amount of diversity inside the SOMs.

in Figure 6.5. The internal measure $\text{MSE}^m_{\text{Map}}$ is given by

$$\text{MSE}_{\text{Map}} = \frac{1}{M}\sum_{m=1}^{M} \text{MSE}^m_{\text{Map}} = \frac{1}{M}\sum_{m=1}^{M} (y - f_m)^2 \tag{6.1}$$

The change in $\text{MSE}^m_{\text{Map}}$ is recorded for three major situations of interest depicted: To which extend the predictors are internally affected by NCL, when analyzing the state among the ensemble predictors being fully connected, i. e. $\lambda = 1$, is compared to the situation, where the predictors are acting independently ($\lambda = 0$). In order to monitor the third situation of interest, the predictors are analyzed, if the cooperation among them is stressed to a higher level than $\lambda = 1$, s. t. the upper bound of λ is exceeded. From the total amount of $M = 25$ predictors, ten of those are randomly selected and their learning progress is recorded over the total number of $t_{max} = 40$ epochs. The results are captured as shown in Figure 6.5, for the situations when $\lambda = 0$, i. e. no NCL is applied in (a) and (d), second, if $\lambda = 1$ (b),(e), as well as $\lambda = 1.01$ ($\lambda = 1.02$ respectively) in (c),(f). Note that only the first 20 training epochs are outlined to focus on the beginning of training. The figures in the upper row (a)-(c) correspond to $K = 10$, while the Figures in the lower row (d)-(f) correspond to $K = 5$. If the total

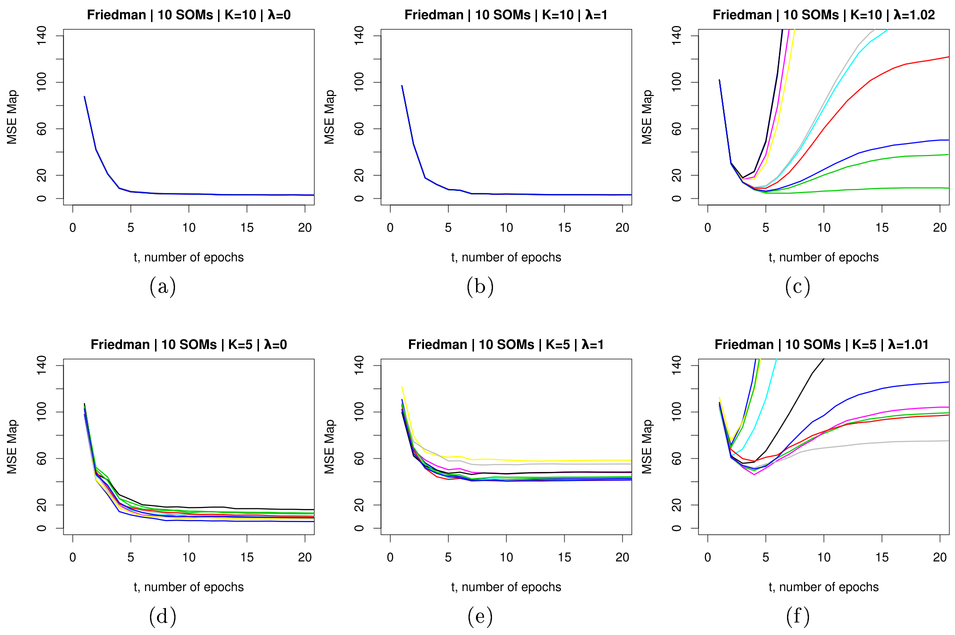


Figure 6.5: $\mathrm{MSE_{Map}}$ for Friedman data changing over training time for ten randomly sampled SOMs. Evaluation of the mean squared error per SOM ($\mathrm{MSE_{Map}}$) for Friedman data changing over training time t. Varying $\lambda \in \{0, 1, 1.01\}$, shows a deviant behavior for the error rates per SOM over number of training epochs.

set of features ($K = d_{\text{in}} = 10$) is used and $\lambda = 0$ (a), the high error gradients lead to a fast convergence in learning for all individual predictors in the same manner. They are not forced to follow different trajectories and hence they do not differ significantly in their errors. All the single predictors are trained on the same subtask, while the only statistical elements are introduced due to the random initialization of the weights and the positions as well as due to the random permutation of the training samples per epoch. Forcing negatively correlated errors of the predictors with $\lambda = 1$ in (b), the ten randomly selected LLMs in the ensemble show similar characteristic error rates as for no NCL in (a).

In contrast to that, the application of RSM with sub-sampling the features for each predictor, s. t. $K = 5$, leads to a rise in diversity and a deviation in MSE_{Map} along the predictors (d). The effect is amplified by NCL and an increasing λ up to the upper bound of $\lambda = 1$ in (e). This way, the MSE_{Map} is increased globally for all SOMs and a higher variation in errors is observed.

When λ exceeds its upper bound in the last column of the Figure (f), (c), however, the amount of variance introduced by NCL is above the critical value and the predictors and therefore their MSE_{Map} are disrupted after some few epochs.

6.5.2 Dynamics in λ

The situation concerning the dynamics in MSE_{Map} over training has been considered for the accounted states of NCL-strength with respect to $\lambda = 0$ and $\lambda = 1$. To provide insights, how the measures of $\text{MSE}_{\text{Map}}, \text{DIV}_{\text{inter}}$ and $\text{DIV}_{\text{intra}}$ are distributed internally along the ensemble predictors, the cooperation between the individual predictors is varied over the NCL strength parameter $\lambda \in [0, 1]$. In Figure 6.6, violin plots are shown, which combine boxplots and a kernel density estimation [Adler, 2005]. The stated violinplots are given for the datasets Friedman, Boston and AAindex for ensembles with $M = 25$ predictors. Every single boxplot corresponds to the distribution of a certain vector of values, one of $(\text{MSE}_{\text{Map}}, \text{DIV}_{\text{inter}}, \text{DIV}_{\text{intra}})$, recorded for all M predictors inside an ensemble as a function of the actual chosen of λ. Hence, a violinplot consists of ten single boxplots. Each violinplot accounts for one of the occurring rates $(\text{MSE}_{\text{Map}}, \text{DIV}_{\text{inter}}, \text{DIV}_{\text{intra}})$. Note that the y-axis is logarithmized resulting in non-equidistant breaks. As expected, the error and diversity measures vary fundamentally over λ: When boosting NCL with $\lambda = 0$ up to $\lambda = 1$, the critical measures increase nearby exponentially and the distributions are shifted towards the maximum. Ensemble predictors with only low diversity are found in ensembles, which are trained with a small λ, and vice versa. The ensembles with strongly forced NCL yield maximum MSE_{Map}, $\text{DIV}_{\text{inter}}$ and $\text{DIV}_{\text{intra}}$ along all the predictors.

6.6 Supporting Altered Penalty Functions

In Section 3.3.3 on page 53, a family of parameterized penalty functions is introduced. Up to now, the LERRANCO ensembles are evaluated based on the $\bar{f}$-penalty according to $\Upsilon = 1$ (see Equation (3.15) on page 54). To evaluate other than the previously applied penalty function, four penalty functions are explored in this Section. The first one is referred to as $\bar{f}$- penalty given in Equation (3.9) on page 53 corresponding to $\Upsilon = 1$ and the second induced by $\Upsilon = 1/M$ (referred to as reciprocal-penalty). The remaining two functions are given by the corresponding *local* versions of the former. The gradient of the regular functions is extended by a *local* diversity propagation factor

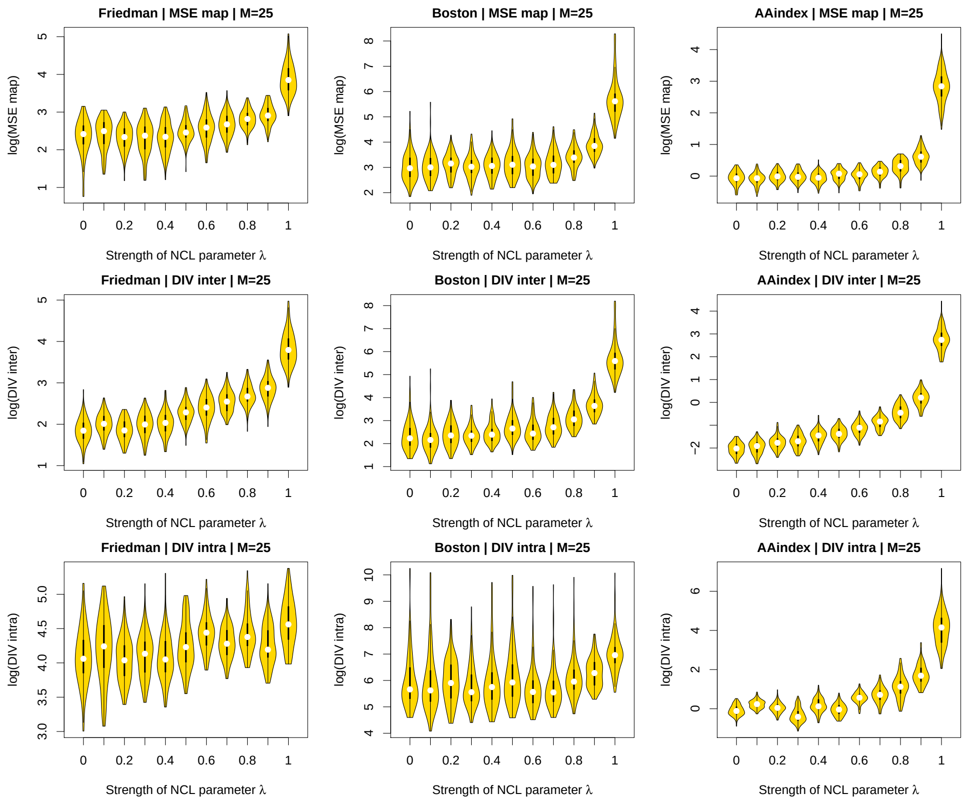


Figure 6.6: Violinplots showing dynamics in λ, the derived errors and the quantified diversity when applying various states of NCL-strength for the Friedman, Boston and AAindex data and $M = 25$. Note that the y-axis is logarithmized resulting in non-equidistant breaks.

h_c according to:

$$\frac{\partial e_m}{\partial f_m}(\mathbf{x}_n) = (1-\lambda)(y_n - f_m(\mathbf{x}_n)) + h_c \cdot \lambda \Upsilon (y_n - \bar{f}(\mathbf{x}_n)) \quad (6.2)$$
$$\text{with } h_c = exp\left(-\frac{|M-s|}{c}\right).$$

The diversity is encouraged depending on the relative distance s of samples to each of the M SOMs and propagated according to the Gaussian neighborhood function with width of $c = M/3$. This extension is based on the idea that samples leading to high diversity of ensemble predictors are far away from samples with non-diverse ensemble predictions. This way, cooperation between the SOMs is either allowed or suppressed.

From the results analyzed for the other remaining three penalty variants, as listed in Table 6.3, no improvement in prediction performance was achieved. In neither case, a sufficient justification is found to supply other than the $\bar{f}$-penalty.

	$\bar{f}$-penalty	**reciprocal**	
MSE$_{\text{Test}}$	**local,$\lambda \to 1$**	**$\lambda \to 1$**	**local, $\lambda \to 1$**
Friedman	4.28	8.739	9.121
Boston	10.777	19.419	16.02
Forestfires	4.667	4.549	4.353
NO2	0.249	0.276	0.279
AAindex	1.082	1.29	1.259

Table 6.3: Mean Squared Test error (MSE), averaged over 50 iterations. Results are given for ensembles of size $M = 100$ and $K = 5$ in case of Friedman, $K = 8$ for Boston, $K = 7$ for Forestfires based on SOMs of grid size 1×2, $K = 5$ in case of NO2 dataset and $K = 25$ for AAindex.

6.7 Discussion

The LERRANCO architecture provides a comfortable way to successful ensemble learning with ensemble predictors sharing uniform internal parameters like network size, learning rate and neighborhood width. As reported, powerful ensembles are generated with very small nets (i. e. with small number of nodes). This may be observed due to the fact that with larger nets the categorization as weak learners is no longer valid, since every net gets too specialized to the presented subtask. As a rule of thumb, setting the neighborhood connectivity parameter σ uniformly to the range of approximately 2.0 up to 3.0 is a good choice for every of the evaluated datasets.

With randomly chosen values of σ for every single SOM independently out of $[0.4, 5]$, the $\mathrm{MSE}_{\mathrm{Test}}$ rises above (4.149 in case of Friedman and 1.138 for AAindex data) the bulk of uniformly set values in the same range. The actual chosen value of a learning parameter is a subordinate factor - the prediction performance rather depends on raw guessed but uniform learning parameters superseding separately fine tuned ones.

One interesting point to mention is the deviant role of the Gaussian neighborhood width defined by σ at the extremes of λ: If $\lambda = 0$, the minimum $\mathrm{MSE}_{\mathrm{Test}}$ is achieved with a low connectivity of nodes according to $\sigma < 1$ for the Friedman data as an example, see Figure 5.4a on page 74. On the other hand, if $\lambda = 1$, the lowest $\mathrm{MSE}_{\mathrm{Test}}$ is achieved approximately with $\sigma = 2$, see Figure 6.3a on page 86. It seems that a higher intra-connectivity among the nodes is needed to compensate for the boosted inter-cooperation among the predictors.

The results show that the variation in feature sets facilitates the ensemble predictors to cover diverse aspects of the learning task at hand. The rise in the diversity even surmounts the loss of individual accuracy ($\mathrm{MSE}_{\mathrm{Map}}$) among the predictors. To this end, the RSM has shown to be a substantial instrument for enhancing the ensemble prediction as compared to using the entire feature set.

Friedman data takes on a special position: It is the only synthetic dataset and it provides to be one of the showpiece where the considered effects are most likely to occur and emerge. This may be observed due to its clear artificial origin and the common error distribution. In contrast to the Friedman data, the error of the real-world datasets is not under control.

The best generalization performance is achieved if the positive error correlations between the single networks are penalized and λ is next to 1. However, if supplying SOMs with a minimum number of nodes, the intrinsic (i. e. $\lambda = 0$) diversity necessitates no further boosting of diversity by NCL, see Figure 6.4d on page 88. This applies in the case of Forestfires data performing best with SOMs of minimum grid size (1×2).

The deviant behavior of Boston and Forestfires data, may be explained by the underlying statistical properties of the input space (two clusters in the latter case). With an increasing λ in the former case, see Figure 6.4c, the $\mathrm{DIV}_{\mathrm{intra}}$ decreases. This suggests a reduction in the complexity of the input-output mapping the SOMs approximate.

With a large number of networks the theoretical upper bound of λ (Equation (3.17) on page 55) is conform with the previous statement by Liu and Yao [1999] that the domain of λ should be in $[0, 1]$. However, this upper bound of λ is not tight for all datasets. I found narrowing the domain of $\lambda \in [0.9, 1.0]$ optimal in that sense that, for grid sizes greater than 1×2, the generalization error is minimized, when the SOMs are fully interlinked. Beyond this point, stability is no longer guaranteed due to the enormous rise in variance, see Figure 6.5 on page 90.

The analysis of supporting other penalty functions shows a clear advantage from the application of the $\bar{f}$-penalty induced by $\Upsilon = 1$. Supplying LERRANCO with the y-penalty function given by $\Upsilon = M$, leads to a non-feasible learning process with too steep gradients and therefore is inconvenient. This is somewhat surprising, because the

y-penalty is found to be more successful on classification problems in some experiments [Brown et al., 2005b].

6.8 Complexity and Computation Time

One drawback with ensembles results from the increased capacities needed to store the models, which are linear in the ensemble size M. Basically, the amount of storage in the LERRANCO architecture is made up of the capacities needed for the storage of the $n_l \cdot M$ prototype vectors ($\mathbf{w}_{mi}^{\text{in}}$, where $m = 1, \dots, M, i = 1 \dots, n_l$), the matrices of the linear mappings ($\mathbf{A}_{mi}$) and the intercept vectors ($\mathbf{w}_{mi}^{\text{out}}$). Therewith, the vector of predicted values $f_m(\mathbf{x})$ and the matrix of membership degree $\mathbf{U}$ are implicitly given with the models. For each ensemble, the following capacities C are required:

$$\begin{aligned} \mathbf{w}^{\text{in}}\text{:} \quad & (n_l \cdot M \cdot K) \\ \mathbf{A}\text{:} \quad & (n_l \cdot M \cdot K) \\ \mathbf{w}^{\text{out}}\text{:} \quad & (n_l \cdot M) \\ \text{which sums up to:} \quad C \; = \; & M \cdot n_l(2K + 1) \end{aligned}$$

Under the assumption of an uniform description length for the numbers, the space needed to store an ensemble model with M predictors depends on the number of nodes n_l for each predictor, the size K of the subspaces and the number of training samples N. The only essential information needed to store are the prototype vectors, whose sizes are independent of the number of samples N. By contrast, the application of a SVM requires a representation of the kernel matrix containing $C_{\text{SVM}} = N \times N$ entries. For the Friedman data, this becomes to $C_{\text{SVM}} = 40,000$, and $C_{\text{SVM}} = 111,556$ for the AAindex data.

Nonetheless, the question arises, how many nodes n_l^* in a single predictor setup are equivalent to n_l nodes in an ensemble setup with M predictors with regard to C. In order to compare these capacities C to a single predictor case, the upper sum C is rearranged to solve for n_l and substituting n_l^* into, we get:

$$n_l^* = \frac{C}{M \cdot (2K + 1)} \tag{6.3}$$

To determine the exact capacities needed for LERRANCO ensemble models, let us assume the following two situations:

1. Friedman dataset: With ensembles of size $M = 100$, and $n_l = 12$ nodes per predictor, training sample size of $N = 200$ and subsets of variables of size $K = 5$, the space required would be $C = 13,200$. In contrast to that, in a single predictor setup, M equals 1 and furthermore no feature selection process is performed, hence K amounts to $K = 10$. If $M = 1$, a single SOM with $n_l^* = 628$ nodes

requires the equivalent storage capacities C. Hence, with a single SOM of grid size for example 25×25, the space under consideration is the same as for an ensemble with $M = 100$ predictors with $M \cdot n_l = 1200$ nodes in total. For analysis or visualization purposes, this order of magnitude is not uncommon.

2. AAindex dataset: The second example demonstrates that the ensembles' capacities mainly depend on K: With $M = 100, n_l = 4, N = 334$ and $K = 25$, we get $C = 20,400$. Substituting this into Equation (6.3) on page 95, for $M = 1$ and the entire input space of $K = 531$ dimensions, the same amount of space is required as for a single SOM of size $n_l^* = 19$, approximately a grid of 4×5 nodes. Here the major improvement follows from the relation of the space required per node by the ensemble feature subsets of size $K < d_{\text{in}}$. The relation of capacities per every ensemble node $\frac{C}{M \cdot n_l} = (2K + 1) = 51$ offers an advantage when compared to the relation of capacities per single predictor node $\frac{C}{n_l^*} = 1063$.

The SOM ensembles can be easily scheduled in parallel and hence allow for fast training.

6.9 Conclusion

For several benchmark datasets the LERRANCO ensemble architecture outperforms other reference learners like RandomForests or even ν-Support Vector Regression. Ensemble learning benefits from SOMs in several ways: The preconditions for successful ensembles are fulfilled with nets of low complexity, i. e. with a low number of nodes. Hence, supplying ensemble component learners with SOMs eliminates the constraints or disadvantages of many parameters to tune, rather offers good generalization with parameters set according to rules of thumb.

NCL fully exploits the potential of SOM ensemble learning when the neural networks cooperate at highest level and stability is satisfied. I demonstrate the important and correlative role of diversity among the ensemble predictors as well as inside the predictors conforming the requirement of weak learners.

Due to the transparency of SOMs, I am able to shed light on the diversity emerging over a combination of several factors:

The first is the *explicit* diversity forcing impact by NCL on *inter*-SOM level. The more emphasis on covariance or diversity is *explicitly* placed by NCL, the more the ensemble test error rate is reduced. All individual single networks are connected to each other and interlinked as one single, complex learning unit. By the combination of the two re-sampling methods Bagging and RSM the diversity *between* SOMs is enforced *implicitly*. Improved accuracy, speed and interpretability are achieved by the application of RSM in either case. Third factor is the *implicit* diversity *inside* the SOMs controlled by the width σ of the Gaussian neighborhood function corresponding to the connectivity of nodes. A single SOM can be viewed as an ensemble itself.

As every SOM has a diversity feature representing the variances between the nodes, a quantification of this *intra-SOM diversity* is defined in Equation (4.3) on page 63. The *inter-SOM diversity* exhibits a highly positive correlation to the *intra-SOM diversity*. The increase in error per every single SOM is ascribed to the nodes, which are topped out at diversity and hence are *specialized locally*. In other words, the inter-SOM diversity enforcing connection surpasses the intra-SOM connection linked via the Gaussian neighborhood between the nodes. The diversity breaks up between these with greater emphasis on the inter-SOM diversity.

The dynamics on the variances and the effect on the stability of learning of the ensemble predictors are exploited over the progress of training. The total $\mathrm{MSE}_{\mathrm{Map}}$ and the diversities as well as the according distributions along the ensemble predictors are identified to be positively correlated to the strength of NCL.

The capacities required to store the M models of the ensemble predictors are independent of the number of samples N. The numbers of nodes in a single predictor setup are related to those in an ensemble setup with M predictors according to the equivalent demand of storage capacities. This shows great advantage for the proposed SOM ensembles in terms of memory-usage. With regard to the computation time, ensembles provide an attractive option, since they can be easily scheduled in parallel.

7 Aggregation of Hypotheses to Ensemble Prediction

The challenging problem of aggregation is how to combine predictors and on which to rely on. With Bagging, as the short form of bootstrap aggregating, two commonly used aspects are covered in Ensemble Learning:

1. M bootstrap training sets are the training bases for the constructed regressors.
2. These regressors are used to form the bagged predictor, i. e. by simple averaging.

The predictors built on bootstrapped training sets have been shown to perform superior (see 5.4 on page 69) compared to those built without bootstrapped training data. Aggregation by averaging is probably the most simple and widely used method to form an ensemble predictor. The reduction in error can be viewed as arising from a reduction in variance due to the averaging over many hypotheses [Bishop, 1995].

In this chapter, the issue is addressed, if the assumption of the model averaging is beneficial. One would expect to improve the ensemble accuracy if some predictors are given greater weights than others. The question arises likewise from the perspective of a single SOM, where the WTA-rule induces an aggregation performed along the local experts.

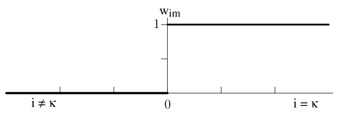
7.1 Aggregation of Local Experts

The aggregated predictor $\bar{f}$ is formed by the weighted combination of the ensemble predictors f_m. The output of every predictor m is the weighted sum of its local experts' output $\mathcal{C}_{im}, i = 1, \dots n_l$:

$$\bar{f} = \sum_m w_m \sum_i w_{im}\phi_{im} = \sum_m w_m \sum_i h_{i\sigma*}\mathcal{C}_{im}$$

where $\sigma*$ denotes the Gaussian neighborhood width. Samples are mapped to an output value by their corresponding local expert κ according to the winner-takes-all (WTA) rule, see Equation (2.2) on page 16. In the notation mentioned above, this can be written as:

$$w_{im} = h_{i\sigma*} = \theta(i) = \begin{cases} 0 & : i \neq \kappa \\ 1 & : i = \kappa \end{cases}$$



Thus, the WTA rule induces a very rough bound at the tessellations of the local experts. Therefore the locality of every single node is enhanced. In the following Figure 7.1, the influence of the neighbors according to the strength-defining parameter $\sigma* = \{0.4, 1.0, 2.0\}$ with increasing distance to node $\mathbf{v}_\kappa$ along the grid is demonstrated. With a small $\sigma* = 0.4$, only the winner node with index κ and its direct topological neighbor are affected. Thereby, the weights w_{im} of the combination rule as induced by $\sigma* = 0.4$ are close to those induced by WTA-rule. The more $\sigma*$ is increased, the more are the topological neighbors i of $\mathbf{v}_\kappa$ accounted for.

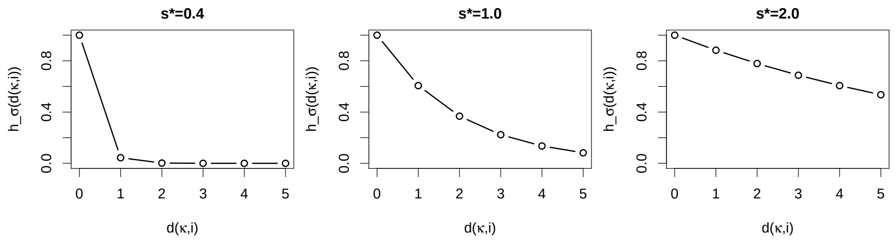


Figure 7.1: Strength of Gaussian Neighborhood depending on chosen σ.

To analyze the effect of the WTA-rule to the ensemble accuracy, a smoothing of the output function is forced by the smoothing parameter $\sigma*$. In Figure 7.2, the evaluation of $\mathrm{MSE}_{\mathrm{Test}}$ for different combinations of λ in steps of 0.1 and the smoothing parameter $\sigma* \in \{0.2, 0.4, 0.6, 0.8, 1.0, 1.5, 2.0, 3.0, 4.0\}$ is shown for Friedman, Boston and AAindex data. Enlarging the influence between the topological neighbors inside each SOM by $\sigma*$ greater than 0.4, leads to a significant loss of prediction performance for all of the evaluated datasets, as compared to the WTA-rule. Especially for highly cooperating SOMs due to $\lambda \to 1$, a slightly increased width of the neighborhood leads to a drastic deterioration in terms of $\mathrm{MSE}_{\mathrm{Test}}$.

7.2 Aggregation of Ensemble Predictors

The ensemble predictors are forced to cover contrary or diverse regions of the hypothesis space. It turned out that averaging of "maximum" diverse hypotheses is a succeeding combination strategy. Nonetheless, it is expected that there is a benefit from an adjustment on which of predictors to rely on. An adequate combination of the diverse hypotheses might figure out some "redundancy" in the hypothesis space. To

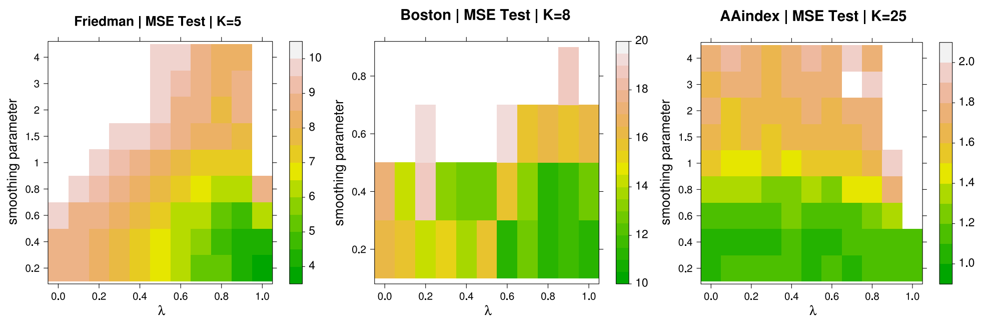


Figure 7.2: MSE_{Test} when varying smoothness of output function by parameter $\sigma*$ inside each SOM and strength parameter λ for NCL. Evaluation is done for $M = 100$ ensemble predictors for Friedman, Boston and AAindex data.

this end, the issue of reweighting the ensemble predictors by an adjustment proportional to their error and diversity criteria is addressed here. The quantitative criteria in terms of $\text{DIV}_{\text{inter}}$, $\text{DIV}_{\text{intra}}$ and MSE_{Map} allow us to adjust the weightings w_m of the individual predictors according to:

(1) Weighting of ensemble predictor m by the individual mean error $\text{MSE}^m_{\text{Map}}$:

$$w_m(\text{MSE}_{\text{Map}}) = \frac{{\text{MSE}^m_{\text{Map}}}^{\beta}}{\sum_{j=1}^{M} {\text{MSE}^j_{\text{Map}}}^{\beta}}, \quad \text{where } \text{MSE}^m_{\text{Map}} = (y - f_m)^2$$

The weights are set proportional to the individual $\text{MSE}^m_{\text{Map}}$, see Equation (6.1) on page 89. This way, the ensemble prediction is severely formed by combining those predictors, which exhibit the highest mean prediction error.

(2) Weighting of ensemble predictor m is set proportional to the individual inter-diversity:

$$w_m(\text{DIV}_{\text{inter}}) = \frac{{\text{DIV}^m_{\text{inter}}}^{\beta}}{\sum_{j=1}^{M} {\text{DIV}^j_{\text{inter}}}^{\beta}}, \quad \text{where } \text{DIV}^m_{\text{inter}} = (f_m - \bar{f})^2$$

(3) Weighting of ensemble predictor m is set proportional to the intra-diversity:

$$w_m(\text{DIV}_{\text{intra}}) = \frac{{\text{DIV}^m_{\text{intra}}}^{\beta}}{\sum_{j=1}^{M} {\text{DIV}^j_{\text{intra}}}^{\beta}}, \quad \text{where } \text{DIV}^m_{\text{intra}} = \frac{1}{n_l}\sum_{i=1}^{n_l}(\mathcal{C}_{im} - f_m)^2$$

where β is the exponent defining the strength of adjustment.

Evaluation is performed over 20 repetitions with the same setup of LERRANCO as described in Section 6.1. The decision, which predictors are used to form the aggregated predictor $\bar{f}$, relies on their individual weighting according to the criteria (1) $\mathrm{MSE}_{\mathrm{Map}}$, (2) $\mathrm{DIV}_{\mathrm{inter}}$ and (3) $\mathrm{DIV}_{\mathrm{intra}}$. The weightings w_m are scaled up to fulfill the convex combination, s. t. the weights sum up to one and $w_m \geq 0$, $\forall m$. The $\mathrm{MSE}_{\mathrm{Test}}$ is analyzed with respect to the different states of NCL, from $\lambda = 0$ to $\lambda = 1.0$ in steps of 0.1, and the strength of the adjusted weightings by $\mathrm{MSE}_{\mathrm{Map}}$, $\mathrm{DIV}_{\mathrm{inter}}$ of the ensemble predictors with respect to $\beta \in \{0.1, 0.2, 0.5, 1.0, 2.0, 5.0\}$. The results are summarized in Figure 7.3 for the Friedman, Boston and AAindex data. $\beta = 0$ corresponds to the traditional aggregation by averaging, while $\beta = 5$ emphasizes strongly on those predictors, which exhibit high individual errors or diversity measures.

If the individual weights of the ensemble predictors are adjusted according to $\mathrm{DIV}_{\mathrm{inter}}$ and $\mathrm{DIV}_{\mathrm{intra}}$ no precise statement can be given. In contrast to that, aggregation and ranking of the predictors performed according to their individual $\mathrm{MSE}_{\mathrm{Map}}$, seems to be an appropriate strategy especially in case of Friedman data. With an increasing β, a minimum $\mathrm{MSE}_{\mathrm{Test}}$ is reached even faster, i. e. with a smaller λ. If the parameters are set to $\beta = 1$ and $\lambda = 0.8$ or $\beta = 2$ as $\lambda = 0.6$, the ensembles used for the prediction of the Boston as well as the AAindex data outperform those ensembles, which are combined by averaging. In Table 7.1 on page 102 the exact resulting performances of ensembles aggregated according to $\mathrm{MSE}_{\mathrm{Map}}$ are compared to those, which are combined by averaging. Actually, the ensembles aggregated according to $\mathrm{MSE}_{\mathrm{Map}}$, have an advantage over the latter for any evaluated datasets except for the Friedman data.

	LERRANCO		**RandomForest**		**ν-SVR**
$\mathrm{MSE}_{\mathrm{Test}}$	$w_m = \frac{1}{M}$	$w_m(\mathrm{MSE}_{\mathrm{Map}})$			
Friedman	**3.9**	4.0	7.1	(5.7^{+})	5.1
Boston	10.1	**9.5**	10.7	(10.2^{+})	10.5
Forestfires	4.0	**3.9**	4.6		4.2
NO2	0.24	**0.22**	0.22		0.25
AAindex	1.02	**0.98**	1.16		1.03

Table 7.1: Mean Squared Test error ($\mathrm{MSE}_{\mathrm{Test}}$), averaged over 50 iterations for LERRANCO, RF and ν-SVR. Results marked with ($^{+}$) are taken from Breiman [2001] with randomly linear combined feature spaces of higher dimension ($d_{\mathrm{in}} = 25$). The corresponding best results are listed for LERRANCO ensembles of size $M = 100$, $K < d_{\mathrm{in}}$, either combined by simple averaging ($w_m = 1/M$) or by aggregating according to the ranking of predictors individually by $\mathrm{MSE}_{\mathrm{Map}}$.

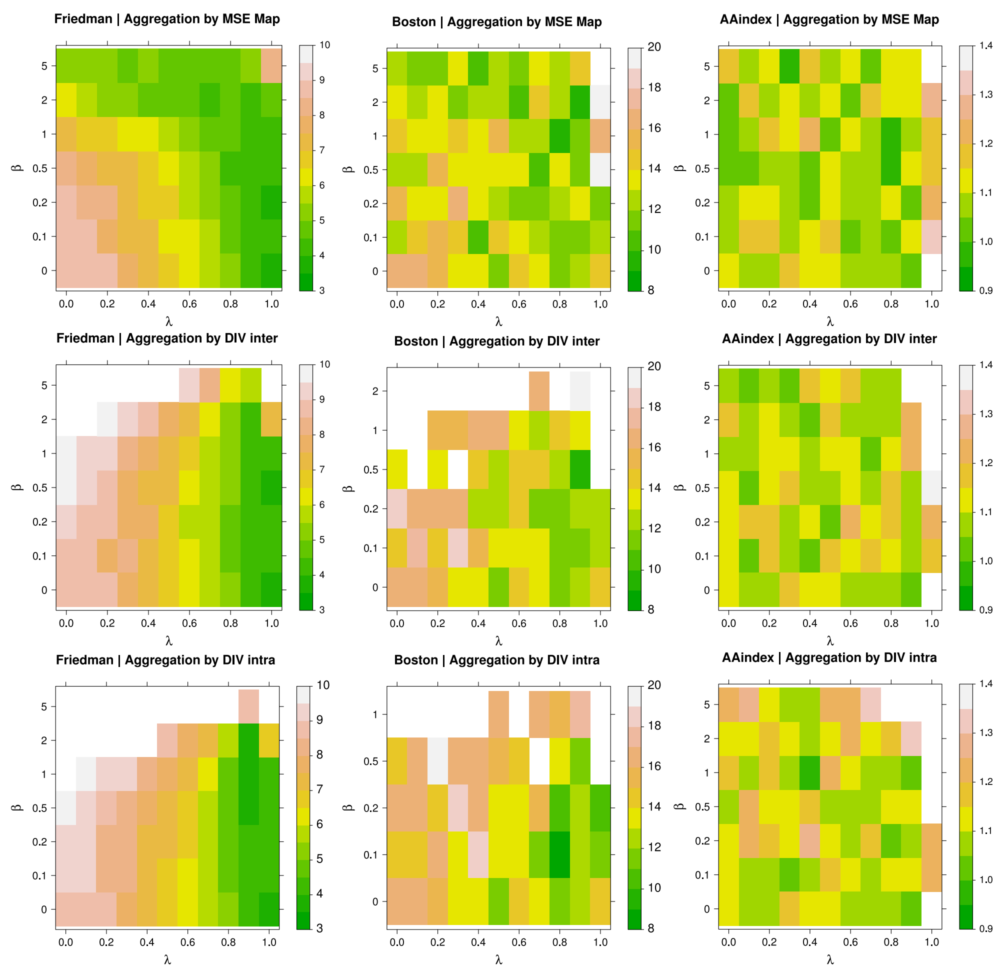


Figure 7.3: Levelplot of MSE_{Test} when changing the aggregation to form the ensemble predictors when varying the strength parameter λ for NCL and the strength of adjusted weightings by MSE_{Map}, $\text{DIV}_{\text{inter}}$ as well as $\text{DIV}_{\text{intra}}$ by β. Evaluation is done for $M = 100$ ensemble predictors for the Friedman ($K = 5$), Boston ($K = 8$) and AAindex ($K = 25$) data.

7.3 Discussion

NCL provides a way of boosting the cooperation between the predictors explicitly and therewith inside each SOM implicitly. The strength of connection between the SOMs surpasses that of the intra-connections as discussed in Section 6.7 on page 93. The nodes are forced to be local specialists. Hence, if the cooperation inside each SOM is enforced by a higher $\sigma*$, a loss in generalization accuracy is determined. In contrast to the WTA-rule, forcing a more smooth output function neglects the diversity along the topological neighbors of the small nets. Thus, the prediction of a certain sample should be performed by its corresponding local expert according to the WTA-rule.

The choice of aggregation strategy for ensemble predictors is a more crucial one. Next to Bagging, three complementary strategies are employed to adjust the aggregation of the predictors. The two strategies employing diversity criteria both fail with regard to improving the ensembles' performances as compared to simple averaging. This is in direct contrast to the approach of ranking predictors by their individual MSE_{Map}. For the Friedman data in Figure 7.3 on page 103, the evaluation of the aggregation according to the MSE_{Map}-criterion exemplarily indicates, that the impact by NCL is emphasized by the effect of β. The optimal trade-off between MSE_{Map} and the diversity is reached with smaller values of λ.

Since the strong relationship between the terms MSE_{Map} and $\text{DIV}_{\text{inter}}$ is known, it is somewhat surprising that actually the ensemble accuracies can be improved by the adjustment of the predictor weights by the MSE_{Map}-criterion for almost all datasets. This superiority in prediction performance by adjusting the ensemble predictor weights by means of the MSE_{Map}-criterion if compared to a fully-connected ensemble by means of $\lambda \to 1$, is traced back to the stronger emphasis on the local specialized learners. This indicates that the objective of diversity between the predictors is subordinate to the local specialization of the ensemble predictors. But if the gain in accuracy justifies the additional computational effort required, depends on the specific concern and problem at hand.

The issue of aggregation of ensemble members is one of the main topics of recent research activities. Many approaches can be found in order to select an appropriate subset of predictors from the initial set and further to reduce the capacities needed to store the models.

7.4 Conclusion

The WTA-rule imposes less constraints to form the prediction inside a SOM-based ensemble predictor and hence should be the method of choice. The more the output function is smoothed, the more are the variances and the local specializations between the topological neighbors deregulated.

The adjustment of the ensemble predictor weights by means of MSE_{Map} used to the aggregate the ensemble output provides a way towards an improvement of the generalization performance of ensembles. This superiority approves the strong emphasis put on locally specialized predictors.

8 Feature Relevance

In many application areas, datasets contain high numbers of dimensions with redundant features. This redundancy is often a consequence of the lack of information about which type of features should be used [Verleysen, 2003].

The samples of a dataset are situated in a d_{in}-dimensional space, where the points are located in a d^*_{in}-dimensional surface or sub-manifold with $d^*_{\text{in}} < d_{\text{in}}$, corresponding to the true dimensionality of the problem at hand. The so called *intrinsic* dimensionality is a common property of the problem, even if these real d^*_{in}-dimensional surface is embedded in a higher-dimensional space with $d_{\text{in}} > d_{\text{in}}$. The characteristic of the intrinsic dimensionality d^*_{in} cannot be assessed directly, but has to be estimated. Several approaches exist to determine the number of dimensions necessary to describe the data. The probably most widely used method is the Principal Component Analysis, also known as Karhunen-Loewe-Transformation, as a linear projection method. In physics, one tries to assess the *fractal dimension* by statistical methods and determine the number of coordinates required to specify a unique point in the data space.

The objective of higher accuracy, speed and more cost-effective learners is often achieved with a reduction in the number of features by feature selection. Feature selection thereby defies the curse of dimensionality. The computational as well as statistical problems are limited if a viable access to the most relevant or useful features is provided. Hence, an estimation on the number of features essentially needed to describe the underlying process, that generated the data, would be helpful and provide one step towards the selection of an appropriate subset of features and improved prediction performance.

To let the abstract notion of feature relevance, importance or usefulness become more concrete, a brief introduction is given to them: Various notions of feature *relevance* exist in the literature, depending on the context and reference. Following Blum and Langley [1997], one discriminates between:

Relevance to the target i. e. if there exists a pair of examples $\mathbf{x}_a$ and $\mathbf{x}_b$, such that they only differ in their assignment to feature k and $\mathbf{y}_a \neq \mathbf{y}_b$

Strong relevance to the sample/distribution similar to the previous notion, but $\mathbf{x}_a$ and $\mathbf{x}_b$ are elements of the training set $\mathbf{T}$.

Weak relevance to the sample/distribution is given, if it is possible to remove a subset of features so that feature k becomes strongly relevant.

Relevance as a complexity measure Given different models, not necessarily of the same type, trained on $\mathbf{T}$ with d relevant features according to definition 1, this model has the fewest relevant features out of all those whose error over $\mathbf{T}$ is least.

Incremental usefulness Given a sample of $\mathbf{T}$, a learning algorithm $\mathcal{L}$, and a feature set $\mathbf{K}$, feature k is incrementally useful to $\mathcal{L}$ with respect to $\mathbf{K}$ if the accuracy of the hypothesis that $\mathcal{L}$ produces using the feature set $\{k \cup \mathbf{K}\}$ is better than the accuracy achieved using just $\mathbf{K}$.

8.1 Feature Relevance in the Context of Ensemble Learning

When applying RSM, as a method for randomly selecting the subsets of features, the curse of dimensionality is circumvented for ensembles. The randomization in feature selection for individual predictors assures somehow to reduce the effects of redundancy. This probability increases with a sufficient number of predictors. A completely useless variable by itself can provide significant performance improvement when taken with others, while two variables that are useless by themselves can be useful together [Guyon and Elisseeff, 2003]. The chances of accomplishing successful predictor models are thereby enhanced with a growing number of ensemble predictors built on subsampled feature subsets.

The benefit from a viable access to the most relevant features is two-fold and provides reduced computational effort and higher generalization as well as a basis for data mining and visualization purposes. This challenging issue will be addressed in this Chapter. In the context of Ensemble Learning and feature relevance, two approaches are covered:

1. Given an estimation of feature relevance FR *a priori*, subsample features with respect to FR used to build the predictor model.
2. Given a built predictor model, estimate the relevance of features FR *a posteriori*.

8.2 Random Weighted Subspace Method

Similar to the approach proposed in Section 2.4 on page 35, the LERRANCO architecture can be extended to account for a variation of the underlying feature sample distribution.

Up to now, each ensemble predictor is trained on an uniformly sampled subset of features. The features are assumed to be independent and identically distributed (i.i.d), so that the probability of choosing one feature is unique and constant.

To focus on the relevant features that contribute to the preceding regression model, the estimated feature relevances $\mathbf{z}$ are used to define the underlying probability distribution of feature subsampling. This way, features that are presumably highly relevant are more likely to be selected for the individual ensemble predictors. I name this technique *Random Weighted Subspace Method* (RWSM). A schematic representation of proposed RWSM is shown in Figure 8.1. A preceded estimation of feature

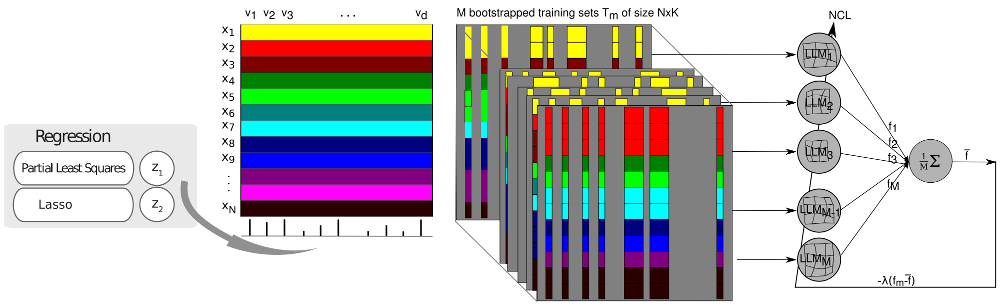


Figure 8.1: Schematic representation of proposed Random Weighting Subspace Method

relevance by model specific metrics given a priori from Least-Angle- and Partial-Least-Squares (PLS)-Regression [Wehrens and Mevik, 2007] is applied and interpreted here as feature selection probability $\mathbf{z}$ scaled to $[0.1, 1.0]$. The least absolute shrinkage and selection operator (Lasso) [Tibshirani, 1996] is a shrinkage and selection method for linear regression using L_1 constraints. It minimizes the residual sum of squared errors, s. t. the sum of the absolute values of the coefficients $\mathbf{b} = (b_0, b_1, \ldots, b_{d_{\text{in}}})$ is less than a constant s:

$$\arg\min_b RSS = \arg\min_b \sum_{n=1}^{N} (y_n - f(\mathbf{x}_n))^2 = \arg\min_b \sum_{n=1}^{N} \left(y_n - b_0 - \sum_{d=1}^{d_{\text{in}}} x_{nd} b_d \right)^2$$

$$\text{subject to} \sum_{d=1}^{d_{\text{in}}} |b_d| \leq s$$

With s, the solutions are constrained versions of the least squares estimates (Section 2.4.1.1 on page 38). An optimal solution to the Lasso method is tackled efficiently with the Least-Angle-Regression procedure provided by the R package `lars` [Hastie and Efron, 2007].

The corresponding estimated feature relevances are abbreviated and indicated by $\mathbf{z}_{\text{Lasso}}$ and $\mathbf{z}_{\text{PLS}}$, where $\mathbf{z}$ denotes the a posteriori estimation of feature relevance in general. The input variables are subsequently scaled to $\tilde{\mathbf{x}} = \mathbf{z} \cdot \mathbf{x}$.

In Table 8.1 the results are listed for the evaluated datasets Friedman, Boston, Forestfires, NO2 and AAindex. LERRANCO models are built with $M = 100$ predictors over 50 repetitions, while the randomly selected subsets of features are sampled according to the probability $\mathbf{z}$ as decided by PLS or Lasso models.

Changing the underlying probability distribution of features randomly sampled by RWSM with respect to $\mathbf{z}_{\text{PLS}}$, takes the ensembles used to predict the the high-dimensional AAindex data to a more successful generalization.

MSE_{Test}		**LERRANCO**		**RandomForest**	**ν-SVR**
	RWSM	$\lambda = 0$	$\lambda \to 1$		
Friedman	Uniform	5.2	**3.9**	7.1	5.1
	PLS	9.1	7.4	7.2	4.7
	Lasso	4.6	4.6	7.2	4.7
Boston	Uniform	11.7	**10.1**	10.7	10.5
	PLS	14.3	11.4	11.5	11.3
	Lasso	17.2	12.6	11.5	12.5
Forestfires	Uniform	**4.0**	4.1	4.6	4.2
	PLS	4.2	4.4	4.5	4.3
	Lasso	4.6	4.2	4.7	**3.9**
NO2	Uniform	0.26	0.24	**0.22**	0.25
	PLS	0.27	0.25	0.23	0.27
	Lasso	0.26	0.24	0.22	0.26
AAindex	Uniform	1.04	1.02	1.16	1.03
	PLS	1.04	**0.97**	1.17	1.13
	Lasso	1.06	1.06	1.15	1.05

Table 8.1: Mean Squared Test error (MSE_{Test}) with respect to RWSM, averaged over 50 iterations for LERRANCO, RF and ν-SVR. The corresponding best results are listed for LERRANCO ensembles of size $M = 100$, $K < d_{\text{in}}$, $\lambda = 0$ (no NCL) or $\lambda \to 1$. With PLS and Lasso, variation of feature selection probability for RSM as well as a scaling of the Euclidean space is performed.

8.3 Assessing the Feature Relevance based on OOB a posteriori

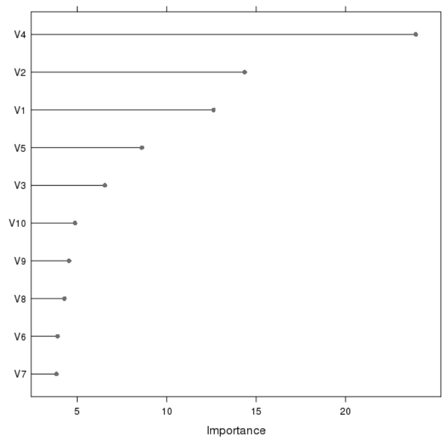
LERRANCO ensembles offer to assess the feature relevance a posteriori in two ways: Similar to the corresponding estimation of variable relevance based on RF models, its contribution to the built ensemble model is either estimated by a permutation on the features of the unseen input OOB-data ($\mathbf{z}_{\text{OOB}}$). Or a permutation is performed on the linear mappings ($\mathbf{z}_{\text{LM}}$) as approximated by each SOM predictor. This variation is covered in the next Section 8.4 on page 112. Both approaches assess the feature relevance $\mathbf{z}$ based on the change in error rate when permuting features. The analysis of feature relevance is grounded on the assumption that permuted features with no or little effect on the error rate can be regarded as useless and are less contributing to the ensemble model. Permuted features with enormous effect on the error rate are of major relevance.

A sketch of the proceeding is given in Algorithm 2. The required arguments are the trained models of predictors $\{f_1, \ldots, f_M\}$ and their corresponding training sets $\{(\mathbf{T}_1, \mathbf{K}_1), \ldots (\mathbf{T}_M, \mathbf{K}_M)\}$ based on $\mathbf{T}$. For the number np of iterations, the MSE on out-of-bag data, here denoted as MSE_{OOB}, is computed for each individual predictor f_m. The permutation $\mathbf{x}_{\text{perm}} \leftarrow \mathbf{p}_{mk}(\mathbf{x}_{\text{OOB}})$ is performed on the out-of-bag data $\mathbf{x}_{\text{OOB}}$ for each predictor f_m. For a certain k, $(\mathbf{p}(\mathbf{x}_{1k}), \ldots, \mathbf{p}(\mathbf{x}_{N_{\text{OOB}}k}))$ means a permutation in feature with index $k \in \mathbf{K}_m$, while the others $1, \ldots, k-1, k+1, \ldots, K$ are kept unchanged. After calculating the MSE_{perm} on OOB data with permuted feature k, the change in error is determined and normalized by the number of OOB data samples N_{OOB}.

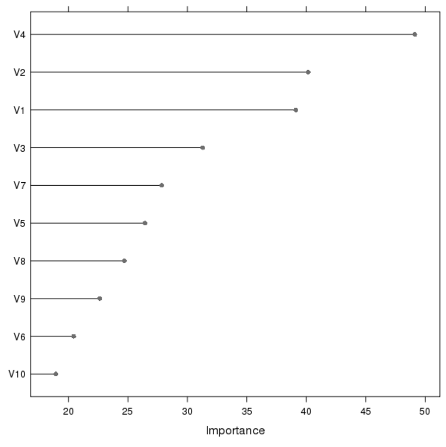
The evaluations $\mathbf{z}_{\text{OOB}}$ of the most relevant features contributing to the LERRANCO models are given in Figure 8.2 for the datasets Friedman, Boston, Forestfires and AAindex. Evaluation is shown for the 20 most relevant features regarding Friedman data with $K = 10$ as well as $K = 5$, Boston ($K = 8$), Forestfires ($K = 9$) and AAindex data ($K = 25$). For the Friedman data it is known by construction that only the first five features contribute to the final output $\mathbf{y}$. In case of the other datasets, as real-world ones, this knowledge is missing for assessment and interpretation purposes. Having LERRANCO trained on Friedman data with $K = d_{\text{in}} = 10$, i. e. without any sub-sampling of features, the five most relevant features contributing to the model are V_4, V_2, V_1, V_5, V_3, exactly corresponding to those features, that make up the final output. On the other hand, the irrelevant features ($V_{10}, V_9, V_8, V_6, V_7$) are "noise" added and actually correctly "classified". If K is set to $K = 5$ on the Friedman data, the resulting ranking of features is no longer that unique as before, but still the five features V_4, V_2, V_1, V_3, V_5 are found on top - only V_5 is ranked slightly lower and positioned at place number six.

Algorithm 2 FeatureRelevanceOOB(M, $\{f_1, \dots, f_M\}$, $\{(\mathbf{T}_1, \mathbf{K}_1), \dots, (\mathbf{T}_M, \mathbf{K}_M)\}$, np)

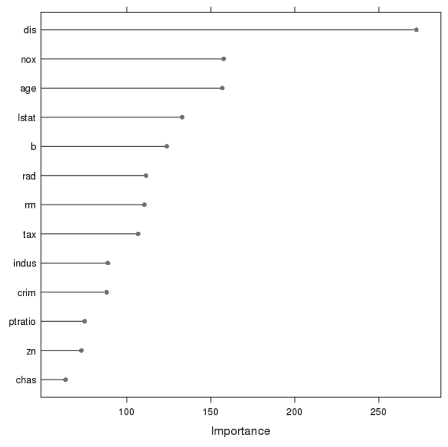
```
Require: M, the number of models
Require: Trained LLM models $\{f_1, \dots, f_M\}$
Require: Training samples $\{(\mathbf{T}_1, \mathbf{K}_1), \dots, (\mathbf{T}_M, \mathbf{K}_M)\}$
Require: np, the number of permutations
 1: Init vector $\mathbf{z}$ for resulting relevances of length $d_{\text{in}}$
 2: for $m = 1$ to $M$ for each network do
 3:     Get out-of-bag data for network $m$: $\mathbf{x}_{\text{OOB}} \leftarrow \{\mathbf{x}_n \in \mathbf{T} : \mathbf{x}_n \notin \mathbf{T}_m, \forall n = 1, \dots, N\}$, where $N_{\text{OOB}}$ denotes number of items $|\mathbf{x}_{\text{OOB}}|$
 4:     Get corresponding out-of-bag targets: $\mathbf{y}_{\text{OOB}} \leftarrow \{y_n : \mathbf{x}_n \notin \mathbf{T}_m, \forall n = 1, \dots, N\}$
 5:     Calculate $\text{MSE}_{\text{OOB}}$, the error per predictor on out-of-bag data:
            $\text{MSE}_{\text{OOB}} = (\mathbf{y}_{\text{OOB}} - f_m(\mathbf{x}_{\text{OOB}}))^2$
 6:     for $k = 1$ to $d_{\text{in}}$ for each feature do
 7:         if $k \in \mathbf{K}_m$ then
 8:             $\text{MSE}_{\text{perm}} \leftarrow 0.0$
 9:             for $t = 1$ to $np$ for each permutation do
10:                 Permute out-of-bag data $\mathbf{x}_{\text{OOB}}$ for $f_m$, s.t.
                        $\mathbf{p}_{mk} = (\mathbf{p}(\mathbf{x}_{1k}), \dots, \mathbf{p}(\mathbf{x}_{N_{\text{OOB}}k}))$
11:                 Perform prediction for model $f_m$ with $\mathbf{x}_{\text{perm}} \leftarrow \mathbf{p}_{mk}(\mathbf{x}_{\text{OOB}})$ on OOB data: $f_m(\mathbf{x}_{\text{perm}})$
12:                 $\Delta\text{MSE}_{\text{perm}} = (\mathbf{y}_{\text{OOB}} - f_m(\mathbf{x}_{\text{perm}}))^2$
13:             end for
14:             Update $\Delta\mathbf{z}[k] = (\text{MSE}_{\text{perm}}/np - \text{MSE}_{\text{OOB}})/N_{\text{OOB}}$
15:             Undo permutation: $\mathbf{x}_{\text{OOB}} \leftarrow \mathbf{p}_{mk}^{-1}$
16:         end if
17:     end for
18: end for
19: return $\mathbf{z}/M$
```



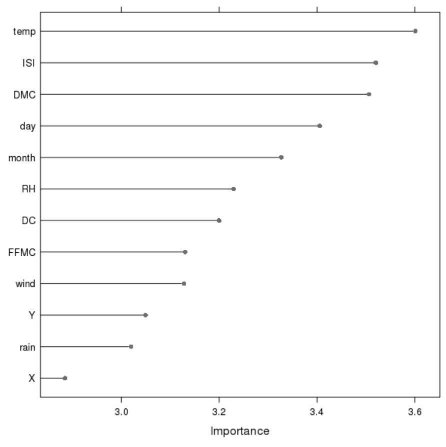
(a) Friedman, K=10, $\lambda = 1$



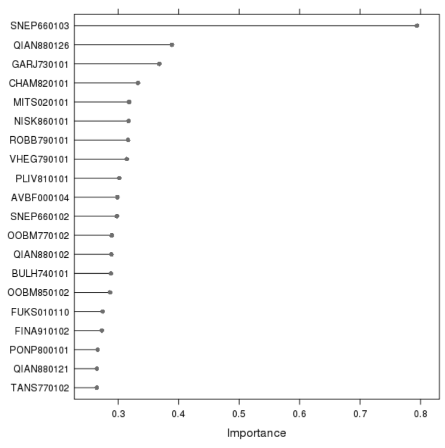
(b) Friedman, K=5, $\lambda = 1$



(c) Boston, K=8, $\lambda = 0.96$



(d) Forestfires, K=9, $\lambda = 0$



(e) AAindex, K=25, $\lambda = 0.9$

Figure 8.2 *(previous page)*: Feature Relevance $\mathbf{z}_{\text{OOB}}$ as estimated by LERRANCO ensembles with $M = 100$ based on the OOB data and change in error rates when permuting features. The 20 most relevant features are shown regarding (a) Friedman data with $K = 10$ as well as (b) $K = 5$, (c) Boston ($K = 8$), (d) Forestfires ($K = 9$) and (e) AAindex data ($K = 25$).

8.4 Assessing the Feature Relevance based on Linear Mappings a posteriori

Every SOM ensemble predictor provides an additional way of assessing the feature relevance. The second model specific metric in an approach to assess the feature relevance a posteriori is based on the corresponding linear mappings learned ($\mathbf{z}_{\text{LM}}$). The locally trained linear maps from the input to the output space are the matrices $\mathbf{A}_i \in \mathbb{R}^{d_{\text{in}} \times d_{\text{out}}}$, where the matrices are vectors of length d_{in} since $d_{\text{out}} = 1$.

In order to estimate the features' contribution based on the linear mappings, the previously presented lines 9 to 15 in Algorithm 2 on page 110 have to be modified. The permutation performed previously is replaced by a permutation over the entries of matrix $\mathbf{A}_m$, with corresponding n_l rows and K columns, where all individual vectors $\mathbf{A}_i, i = 1 \ldots, n_l$ of predictor m are merged into the matrix $\mathbf{A}_m$. One permutation $\mathbf{p}_{mk} = (\mathbf{p}(\mathbf{A}_{m1k}), \ldots, \mathbf{p}(\mathbf{A}_{mn_lk}))$ randomly shuffles the entries of one column k of each matrix $\mathbf{A}_m$, if and only if $k \in \mathbf{K}_m$. Again, the MSE_{OOB} is computed on the out-of-bag data for each predictor, and then the same computed (MSE_{perm}) after permuting column k in $\mathbf{A}_m$. The differences are averaged and normalized.

Algorithm 3 FeatureRelevanceLM($M, \{f_1, \dots, f_M\}$, np, $\{(\mathbf{T}_1, \mathbf{K}_1), \dots, (\mathbf{T}_M, \mathbf{K}_M)\}$)

```
 9:         for $t = 1$ to $np$ for each permutation do
10:            Permute linear mapping of $f_m$, s. t.
                   $\mathbf{p}_{mk} = (\mathbf{p}(\mathbf{A}_{m1k}), \dots, \mathbf{p}(\mathbf{A}_{mn_lk}))$
11:            Perform prediction for model $f_m(\mathbf{x}_{\mathrm{OOB}})$ where $\mathbf{A}_{mk} \leftarrow \mathbf{p}_{mk}$ on $\mathbf{x}_{\mathrm{OOB}}$
12:            $\Delta\mathrm{MSE}_{\mathrm{perm}} = (\mathbf{y}_{\mathrm{OOB}} - f_m(\mathbf{x}_{\mathrm{OOB}}))^2$
13:         end for
14:         Update $\Delta\mathbf{z}[k] = (\mathrm{MSE}_{\mathrm{perm}}/np - \mathrm{MSE}_{\mathrm{OOB}})/N_o ob$
15:         Undo permutation: $\mathbf{A}_{m.k} \leftarrow \mathbf{p}_{mk}^{-1}$
```

The estimations of features contributing to the LERRANCO models according to the specific metric $\mathbf{z}_{\mathrm{LM}}$ are given in Figure 8.3 for Friedman, Boston, Forestfires and AAindex data.

When comparing this ranking of features $\mathbf{z}_{\mathrm{LM}}$ to the previous ranking according to $\mathbf{z}_{\mathrm{OOB}}$ as shown in Figure 8.2, there are some significant differences in the assessment of feature contribution. For Friedman data, all five features V_1 to V_5 are positioned on top by $\mathbf{z}_{\mathrm{LM}}$, while V_4 and V_5 are erroneously displaced downwards by V_8. In case of Boston data, the $\mathbf{z}_{\mathrm{LM}}$ estimations with regard to features “nox”, “dist”, “age”, “lstat” and “b” are complementary to those by $\mathbf{z}_{\mathrm{OOB}}$, while for the Forestfires data the assessed feature relevances $\mathbf{z}_{\mathrm{OOB}}$ are close to accordance with $\mathbf{z}_{\mathrm{LM}}$. In case of the AAindex data, unanimously, great consensus exists for a number of eleven out of 20 features (“SNEP660103”, “MITS020101”, “NISK860101”, “ROBB790101”, “AVBF000104”, “SNEP660102”, “QIAN880102”, “FUKS010110”, “FINA910102”, “PONP800101”, “QIAN880121”).

8.5 Robustness of Feature Relevance

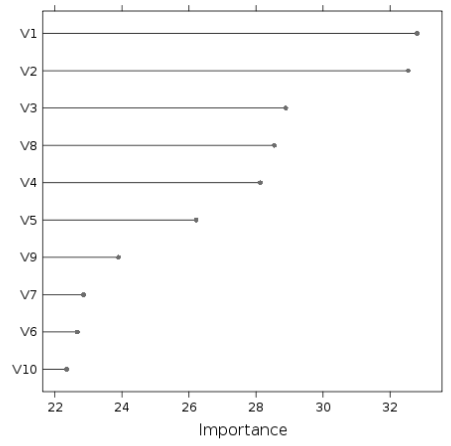
To find out about the impact on the resulting LERRANCO regression model if an a priori feature relevance estimation is accounted for, two situations are considered for Figure 8.4. For the Friedman data as well as for AAindex, a parallel coordinates plot is given, where for every situation of interest the range of estimated feature relevances is covered. The situations analyzed are the a posteriori assessed feature relevances

(a, first and second coordinate) $\mathbf{z}_{\mathrm{OOB}}$, $\mathbf{z}_{\mathrm{LM}}$ assessed by LERRANCO models with *uniformly* sampled features

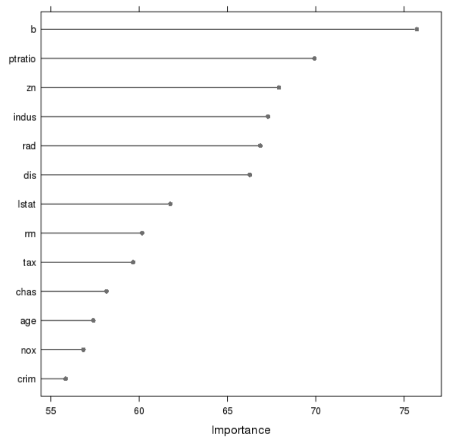
(b, third coordinate) $\mathbf{z}_{\mathrm{PLS}}$ assessed by a PLS model

(c, fourth and fifth coordinate) $\mathbf{z}_{\mathrm{WOOB}}$, $\mathbf{z}_{\mathrm{WLM}}$ assessed by LERRANCO models accounting for the features being subsampled by the *RWSM* according to $\mathbf{z}_{\mathrm{PLS}}$.

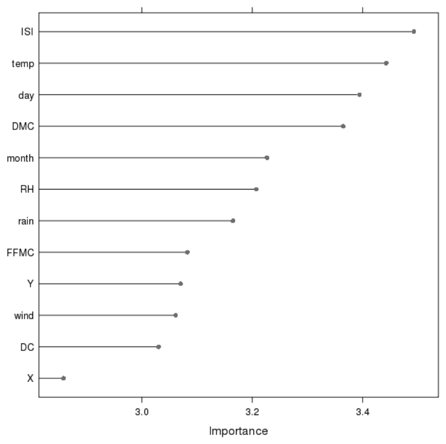
This provides a way of assessing the impact of the RWSM according to $\mathbf{z}_{\mathrm{PLS}}$ to the features contributing to the LERRANCO models.



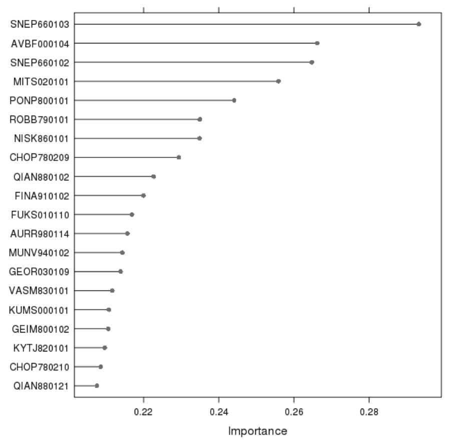
(a) Friedman, K=5, $\lambda = 1$



(b) Boston, K=8, $\lambda = 0.96$



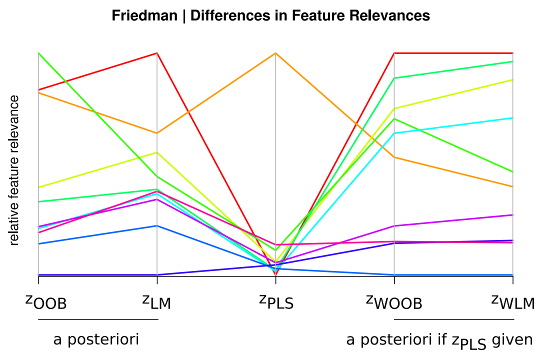
(c) Forestfires, K=9, $\lambda = 0$



(d) AAindex, K=25, $\lambda = 0.9$

Figure 8.3: Feature Relevance $\mathbf{z}_{\text{LM}}$ as estimated by LERRANCO ensembles with $M = 100$ based on the linear mappings and change in error rates when permuting features. The 20 most relevant features are shown regarding (a) Friedman data with $K = 5$, (b) Boston ($K = 8$), (c) Forestfires ($K = 9$) and (d) AAindex data ($K = 25$).

The a posteriori estimated feature relevances, $\mathbf{z}_{\mathrm{OOB}}$ and $\mathbf{z}_{\mathrm{LM}}$, are apparently strongly connected for both datasets. Moreover, $\mathbf{z}_{\mathrm{OOB}}$, $\mathbf{z}_{\mathrm{LM}}$ as well as $\mathbf{z}_{\mathrm{WOOB}}$ and $\mathbf{z}_{\mathrm{WLM}}$ yield high correlations for each of two considered situations as marked by the bold lines. The $\mathbf{z}_{\mathrm{PLS}}$ induces a contrary estimation of feature relevance if compared to the estimations based on LERRANCO. Even with the disruptive factor introduced by $\mathbf{z}_{\mathrm{PLS}}$, the LERRANCO models are capable of extracting the "right" features ($\mathbf{z}_{\mathrm{WOOB}}$, $\mathbf{z}_{\mathrm{WLM}}$) in case of the Friedman data. In case of the AAindex data, the models, which account for the $\mathbf{z}_{\mathrm{PLS}}$ a priori, stronger concentrate on a lesser number of features.



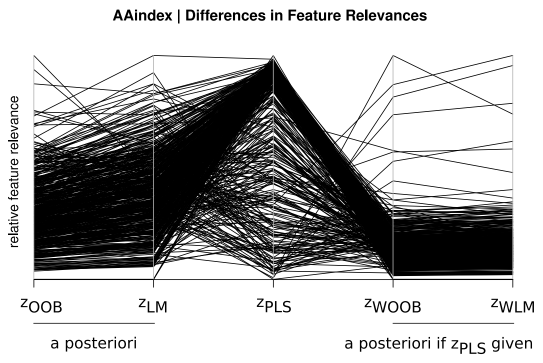


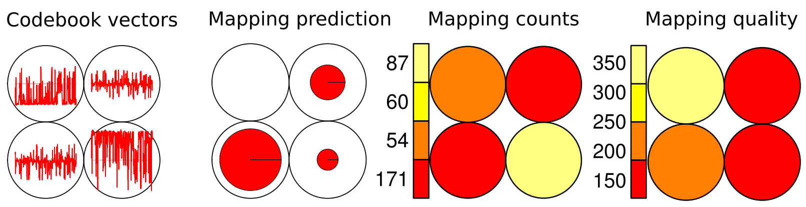
Figure 8.4: Parallel Coordinates plots for the differences in the assessed feature relevance a priori vs. a posteriori. in case of the Friedman and the AAindex data. The situations analyzed are the a posteriori estimation of feature relevance (a, first and second coordinate) $\mathbf{z}_{\mathrm{OOB}}$, $\mathbf{z}_{\mathrm{LM}}$ assessed by LERRANCO models with *uniformly* sampled features (b, third coordinate) $\mathbf{z}_{\mathrm{PLS}}$ assessed by a PLS model (c, fourth and fifth coordinate) $\mathbf{z}_{\mathrm{WOOB}}$, $\mathbf{z}_{\mathrm{WLM}}$ assessed by LERRANCO models accounting for the features being subsampled by the *RWSM* according to $\mathbf{z}_{\mathrm{PLS}}$.

8.6 Contribution of Features to Visualization

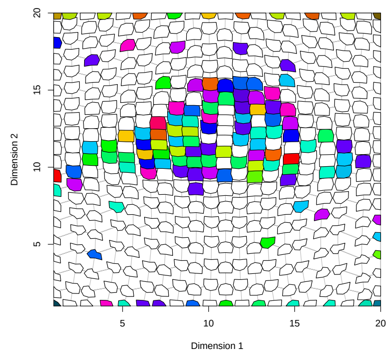
Ensembles bare some shortcomings as the aggregation over individual predictors causes in general the loss of simple and comprehensible structures. To this end, a single SOM predictor is preferable, since it offers transparency and enhanced visualizations. These advantages seem to get lost in the context of EL, since every individual SOM predictor on its own contains not as much information to derive a valuable picture. Each SOM predictor represents only a small detail of the whole picture. With the small nets or small number of nodes per each SOM adapting to a certain sub-task, enhanced visualizations as U-matrix [Ultsch, 1993] are not appropriate choices. A direct representation of the components, the mapping and the quality is the only we can rely on. Having trained for an example a 2×2 SOM on the AAindex data, a representation of each ensemble predictor can be derived as in Figure 8.5a. In this Figure the codebook vectors, the corresponding predicted mappings of each node, as given by $\mathbf{w}^{\text{out}}$, the counts of samples, this node is coding vector for, and the quality of mapping, as defined by the mean quantization error, are represented. The left upper prototype vector can be identified to cover lower ranges over the majority of features, while the codebook in the right lower corner represents few samples with high feature values. The according node mappings are well distinguished.

The number of ensemble predictors as well as the feature subsampling are counterparts to the comprehensibility. As discussed in Section 4.1 on page 57, forcing of commonality along the trained nets, can help to accomplish a consistent common clustering over the independently trained nets.

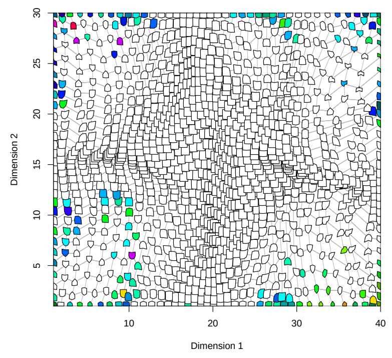
Assembling the vast amount of SOM prototypes to an overall picture is one of the major issues for visualization. One strategy to tackle the obstacles is to map the total set of resulting prototypes onto one subsequently applied SOM approximating as a secondary clustering. An example of a secondary clustering by a SOM of grid size 30×40 is given in Figure 8.5c for the Friedman data. The number of nodes is equal to the number of nodes $M \cdot 2 \cdot 6$ in the ensemble setup corresponding to 1200 prototype vectors. The entries of the prototype vectors $\mathbf{w}^{\text{in}}_{im} \in \mathbb{R}^{d_{\text{in}}}$ corresponding to node index i, where $i = 1, \ldots, n_l$ for each predictor $m = 1, \ldots, M$, are set to a value of 0, if the feature with index k is not contained in the randomly sampled subset of features $\mathbf{K}_m$ for predictor m. Another example of a secondary clustering in presented in Figure 8.5b. This SOM of size 20×20 approximates the resulting prototype vectors of a LERRANCO model trained on the AAindex data. The plots correspond to the representation suggested by Cottrell and de Bodt [1996] for SOMs and provided by the R packages `klaR` [Weihs et al., 2005] and `som` [Yan, 2004]. In these representations (b),(c), the shape of each of the subsequent approximating prototypes depends on the relative distance to its eight neighbors. The color code is modified in that way that the approximated distribution of the target values ($\mathbf{w}^{\text{out}}_i$) is represented in a binned ($b = 20$) version. But this process increases computational costs and we cannot rely on a comprehensible map or visualization.



(a) AAindex, single 2×2 SOM



(b) AAindex, secondary clustering by a 20×20 SOM



(c) Friedman, secondary clustering by a 30×40 SOM

Figure 8.5: Visualization of ensemble predictors. (a) shows the representations of a single SOM (2×2) adapting to the AAindex data. In (b) an example of a secondary clustering is given by a 20×20 SOM approximating the prototypes of an LERRANCO ensemble model trained on AAindex. In (c) a secondary clustering by a 30×40 SOM for the Friedman dataset is presented.

A second effort to compensate the loss of comprehensibility is to benefit from the ranking of features $\mathbf{z}$ in order to facilitate a visualization of the prototypes as in Figure 8.6. This figure shows the prototype vectors for the five most relevant features and the vectors approximating the distribution of target values $\mathbf{w}^{\text{out}}$ with regard to the benchmark datasets Friedman, Boston, Forestfires and AAindex.

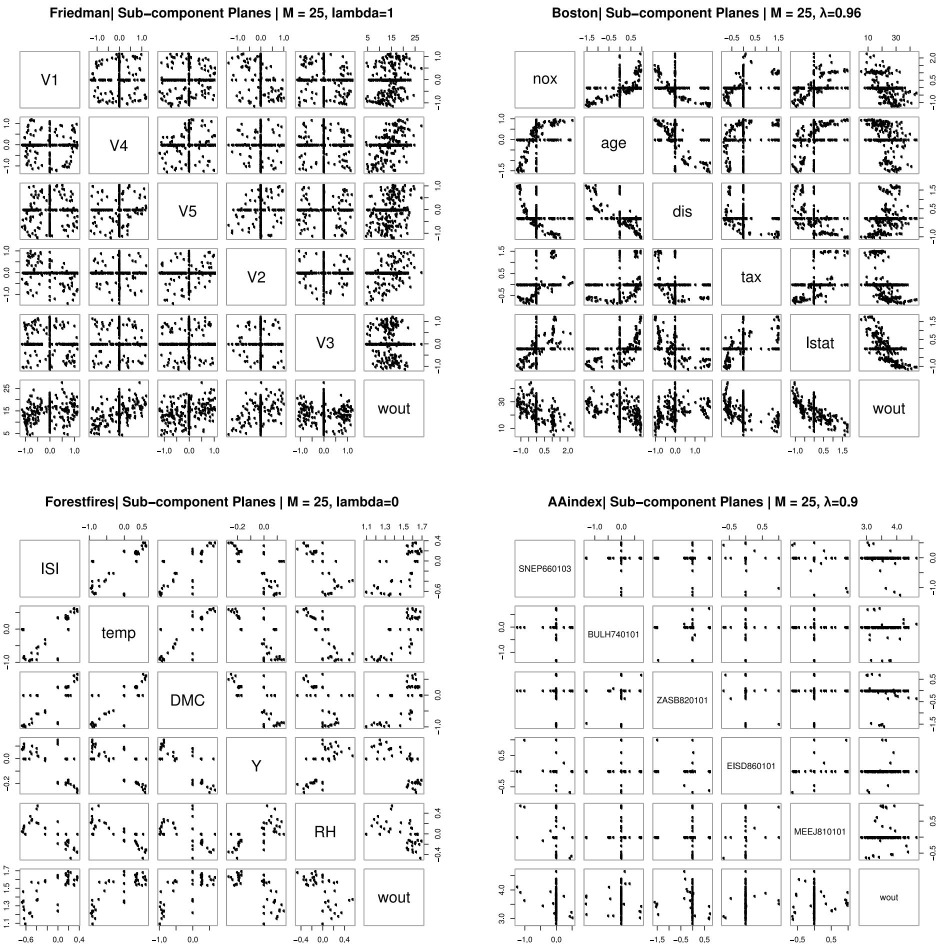


Figure 8.6: Main Components of ensembles prototypes for benchmark datasets Friedman, Boston, Forestfires and AAindex arranged in a scatterplot-matrix. This figure shows the prototype vectors for the five most relevant features according to $\mathbf{z}_{\mathrm{OOB}}$. The sixth component is given by the vectors approximating the distribution of target values $\mathbf{w}^{\mathrm{out}}$.

8.7 Discussion

With ensembles, completely new challenges arise for feature selection and the task is even harder than the classical feature selection but instead one needs to find a subset of features promoting diversity along the ensemble predictors [Opitz, 1999].

The RWSM supplies a sampling of features according to a different probability distribution than i.i.d.. By adjusting the feature sample distribution according to $\mathbf{z}_{\text{PLS}}$, one gets an advantage by RWSM in terms of MSE_{Test} for the AAindex data. This way, one achieves the highest precision of 0.97 for this dataset over all tested methodologies.

For the other datasets, no improvement in accuracy can be observed. In case of the small dimensional datasets, the RWSM imposes a skew in the information that is presented to each predictor. As a result, the diversity between the subsets of features and thus between the sub-tasks is reduced along the ensemble predictors. This may be due to a lower fraction of redundant features, in contrast to the high-dimensional AAindex data, which is supposed to contain many redundant and useless features.

The analysis of feature relevance with regard to the both methods $\mathbf{z}_{\text{OOB}}$ and $\mathbf{z}_{\text{LM}}$ applicable to LERRANCO models, has shown a strong indication of the robustness of estimation. Both $\mathbf{z}_{\text{OOB}}$ and $\mathbf{z}_{\text{LM}}$ are in agreement about the features' contribution to the model and thus none can be favorably pointed out. Only regarding the computational effort, $\mathbf{z}_{\text{LM}}$ has an advantage, since $M \cdot K \cdot n_l$ permutations are required to shuffle the entries of the linear maps, while $\mathbf{z}_{\text{OOB}}$ requires $M \cdot K \cdot N$ permutations. Interesting insights are expected from a *qualitative* investigation and analysis of the proposed assessment of feature relevances. Though, further high-dimensional datasets and more knowledge about them are needed for the semantic level of analysis.

Ensembles still demand for enhanced visualizations, as to compensate for the loss of comprehensibility. With the application of re-sampling methods like Bagging and RSM, the performance improvement comes at the cost of commonality. The difficulty arises even for transparent ensemble predictors as proposed with LLMs based on SOMs. The main issue comes with the problem of re-ordering the high amount of prototypes as long as a topology is non existent between the predictors. To circumvent this lack of commonality, some of the strategies discussed in Section 4.1 on page 57 might be helpful.

8.8 Conclusion

Ensembles provide a way of globally handling the information presented and circumvent the curse of dimensionality. When applying the RSM, in context of EL two major goals are accomplished: First, the effect of noise in data as well as the probability of redundancy in the subsets of features is reduced. Second, a diverse set of accurate predictors is created. Moreover, the computational effort is reduced for high-dimensional data.

The RSM is amended to the *Random Weighted Subspace Method* (RWSM) in order to benefit from an a priori given assessment of feature relevance. The estimated feature relevances are used to define the underlying probability distribution of features being randomly sampled. Especially for the high-dimensional AAindex data, the RWSM facilitates an improved prediction performance.

In an effort to give a better understanding of the process that generated the data, the proposed LERRANCO architecture provides two ways to assess the feature relevance a posteriori. The features' contribution to the built ensemble model is estimated by a permutation performed either on OOB-data or on the local linear mappings approximated by each SOM predictor. The analysis of feature relevance with regard to both metrics has strongly indicated the robustness of estimation.

Assembling the vast amount of SOM prototypes in ensembles to an overall picture is one of the major issues for visualization. One strategy for an enhanced visualization is analyzed by mapping the total set of resulting prototypes onto one subsequently applied SOM, which approximates as a secondary clustering. The visualization in the form of a scatterplot-matrix benefits from the derived feature relevances in terms of comprehensibility.

9 Conclusions

In this thesis, the four main challenges arising exemplarily in bioinformatics applications are covered and coped with an ensemble architecture method offering a promising way to the key issues stated in the introductory Chapter: (a) finding an adequate and suitable predictor, (b) improving the performance of prediction, (c) reducing the number of features and (d) providing a basis for data mining purposes.

Within this general context, the primary objective of this work is to address *how* and *why* SOMs introduced into the field of Ensemble Learning succeed. To this end, I propose the LERRANCO ensemble architecture combining the negatively correlated SOM ensembles as part-adapting structures promoted by the resampling methods Bagging and RSM.

9.1 How SOM ensembles succeed

The SOM demands for a compromise between the limiting factors accuracy, the number of free parameters, the tolerance to redundant variables, the computational effort, the storage capacities and the transparency.

In this thesis, I address the issue how and why SOMs introduced into the field of Ensemble Learning succeed and if the weaknesses of SOM learning can be turned into strengths. For regression task, the SOM is extended by an additional layer of nodes to a Local Linear Map (LLM) by fitting a set of local linear functions to the training data. The LERRANCO ensemble architecture is proposed to combine the negatively correlated LLM ensembles based on SOMs as part-adapting structures with resampling methods.

Ensembles voting from a set of hypotheses do not demand for highly accurate predictors, but for *diverse* hypotheses aggregated to form the ensemble predictor. A growing number of predictors is not the only universal mean of improving the ensemble performance, but the *diversity* among the ensemble predictors.

The LERRANCO architecture forces the ensemble predictors based on SOMs to cover different sub-tasks of the entire learning task at hand, in order to achieve a sufficient level of diversity among the individual ensemble predictors. The results of this thesis show a benefit from SOM ensembles and their topology preserving characteristic when compared to other VQ-based approaches like (Growing) Neural Gas or Fuzzy C-means clustering. SOMs are suited well to fulfill the theoretically posed requirements of weak and unstable algorithms as basis for successful ensembles. The preconditions for successful ensembles are fulfilled with nets of low complexity, i. e. low number of nodes. Ensemble predictors sharing the **internal parameters uniformly** like network size, and the learning rates as well as neighborhood width **set according to rules of thumb**, are found to yield the best performances. The critical parameters left are the grid size and the size of randomly sampled subspaces used for training. The potential of SOMs to Ensemble Learning is mainly exploited by an effective **reduction in the models' complexity** according to the number of free parameters, the size of each SOM and the dimensionality of input space of each SOM. Using LERRANCO I am able to achieve results **outperforming** other reference regression architectures (Random Forests and SVR) when the individual predictors are fully-connected to a single, complex learning unit.

9.2 Why SOM ensembles succeed

We have to distinguish the diversity emerging over a combination of four factors: *implicit* and *explicit*, as well as *between* and *within* the ensemble predictors.

First factor is the *explicit* diversity forcing impact by NCL on *inter*-SOM level. By a combination of the two resampling methods Bagging and RSM the diversity *between* SOMs is enforced *implicitly*. The third factor is the *implicit* diversity *inside* the SOMs controlled by σ corresponding to the connectivity of nodes. The *intra-SOM diversity* is affected implicitly by the inter-SOM diversity due to NCL.

Boosting the diversity explicitly by employing NCL, a shift is performed towards interlinking individual learning stages including a term penalizing correlation in errors. NCL allows to balance between single predictor accuracy and diversity, controlled by the cooperation among the neural networks, thereby dispensing with a sub-local accuracy for a higher overall generalization ability. The relation between two commonly used penalty functions is identified leading to a family of parameterized functions. The diversity forcing impact by NCL on *inter*-SOM level forces stronger connections along the ensemble predictors. Enforcing the diversity explicitly succeeds until the ensemble predictors are interlinked as one **single, complex learning unit**. The effective dynamics on interplay among the ensemble predictors are exploited. To this end, the positive relationship between NCL strength and ensemble diversity in general, i. e. for $\mathrm{DIV}_{\mathrm{inter}}$ as well as $\mathrm{DIV}_{\mathrm{intra}}$, exerts its fundamentally equally positive influence within each ensemble.

By combining the two re-sampling methods Bagging and RSM the diversity *between* SOMs is enforced *implicitly*. The predictors adapt to different sub-tasks and are capable to handle the information globally. **Improved accuracy, speed and interpretability** are achieved by the application of RSM in either case. Therewith, the predictors are forced to locally specialize or to discriminate in these local regions of the input space. Predictors strongly benefit from a reduction of features thereby defying the curse of dimensionality, offering **reduced time and space complexity**.

The generalization accuracy may be further improved by including a variation in the feature sample distribution.

A single SOM can be seen as an ensemble itself. It enforces intrinsic *intra-SOM* diversity by the width of the Gaussian neighborhood function. As every SOM has a diversity feature representing the variances between the nodes, the quantification of the *intra-SOM diversity* is introduced. The *inter-SOM diversity* exhibits a highly positive correlation to the quantified *intra-SOM diversity.* The increase in error per every single SOM is ascribed to the nodes, which are topped out at diversity. In other words, the inter-SOM diversity enforcing connection surpasses the intra-SOM connection linked via the Gaussian neighborhood between the nodes. The diversity breaks up between these with greater emphasis on the inter-SOM diversity. A higher intra-connectivity among the nodes by $\sigma = 2.0$ is needed so as to compensate the boosted inter-cooperation among the predictors. These characterizations of **locally specialized learners** help to understand *why* the SOM ensembles succeed.

SOM *ensembles* lack of the comprehensible structures and the topology along the ensemble predictors, facilitating analysis and enhanced visualizations, as it is for any other learning architecture applied. For visualization purposes, the vast amount of SOM prototypes in ensembles has to be assembled to an overall picture. In order to get further insights into the data generating process, a ranking of features according to their estimated relevance is a beneficial hint. For this purpose, the feature relevance is assessed a posteriori by two approaches with regard to the features' contribution to LERRANCO models. The first approach is based on a permutation of the OOB samples, while the second uses a permutation of the locally linear mappings as approximated by the SOM predictors. The analysis of feature relevance with regard to both metrics has strongly indicated the robustness of estimation.

With regard to the computation time, ensembles provide an attractive option, since they can be easily scheduled in parallel. Moreover, conform to the small number of nodes as well as the reduced size of ensemble feature subsets, ensembles offer the great advantage of a fast and effective parallel computation. The capacities required to store the M models of the ensemble predictors are independent of the number of samples N. The reduction in the size of the ensemble feature subsets offers **great advantage for high-dimensional** datasets. Next to the inherent parallelism, SOMs have a great attraction for their capability of **incremental learning**.

9.3 Outlook

Ensemble Learning has shown great success and a vast amount of methodologies exist in efforts to boost diversity. As it is underlined by Zhou [2009] in his invited plenary talk at MCS'09, recently, only a few papers are published in the top machine learning conferences with the topic of Ensemble Learning. He ascribes that to the easier tasks being finished. In the forthcoming chapter of Brown [2010], he concludes that the generation and investigation of diverse models “is nowadays rather oversubscribed". The focus of the recent challenges and activities is now on methods for non-standard data. Examples of these are non-stationary data or challenges arising in semi-supervised learning or incremental learning. Within this fields of applications, the proposed SOM ensembles may show great promise.

9.4 Summary

In this thesis, I address the issue how and why Self-Organizing Maps (SOMs) introduced into the field of Ensemble Learning succeed. SOM Ensemble Learning benefits from non-independent learning when individual learning stages are interlinked by a term penalizing positive error correlations along the ensemble predictors. For regression task, an extension of the SOM is used (Local Linear Map) by fitting a set of local linear functions to the training data. The proposed LERRANCO ensemble architecture combines the negatively correlated LLM ensembles based on SOMs as part-adapting structures with the resampling methods Bagging and RSM. Using LERRANCO I am able to achieve results outperforming other reference regression architectures. With SOMs applied as a classifier and reconsidered as an ensemble itself, we gain completely new perspectives in the context of diversity and Ensemble Learning: The total diversity balances over a dynamic interplay of factors arising implicitly and explicitly, as well as on inter- and intra-SOM level. The potential of SOMs to Ensemble Learning is mainly exploited by an effective reduction in the models' complexity according to the number of free parameters, the size of each SOM and the dimension of input space. The SOM predictors succeed with uniformly (internal) parameters set according to rules of thumb. LERRANCO ensembles succeed with the highest generalization performance when the SOM predictors cooperate at highest level and stability is satisfied. By putting a strong emphasis on locally specialized predictors, improved accuracy, reduced time and space complexity are accomplished.

Acknowledgements

Thanks goes to those who supported me during the development of this thesis: Firstly, I would like to gratefully acknowledge to my supervisor, JunProf. Dr. Tim W. Nattkemper, providing continued encouragement and invaluable suggestions during this PhD project. This work would not have been possible without his constructive comments and the motivating and helpful discussions.

I also appreciate the help of my workmate Wiebke Timm, who accompanied this work in its earlier phase. Without her general advices, providing the MS data, generating the features, doing the forward feature selection, and at last the understanding gained about Mass spectrometry, peaks, tryptic digestion, ionizability and intensities, the second Chapter would not have been set up. Thanks to Prof. Sebastian Böcker for inviting me to Jena and for the scientific exchange.

I am also grateful to all my colleagues Julia Herold, Christian Loyek and Dr. Jörg Ontrup for all of their support, proof-reading my thesis and the friendly atmosphere. Thanks to all the others of the "Biodata Mining & Applied Neuroinformatics Group".

Thanks to Marc Sturm in Tübingen, who has assisted to complete the integration of the OpenMS module. I would like to acknowledge support for this project from the Deutsche Forschungsgemeinschaft (DFG).

Finally, I'd like to thank all of my friends, for their patience, understanding and encouraging words. Very special thanks go to my family and to Markus for their tons of moral support and helping me to accomplish this mission.

Bibliography

Hervé Abdi. *Partial Least Squares (PLS) Regression.*, pages 740–744. Thousand Oaks, 2007.

Daniel Adler. *vioplot: Violin plot*, 2005. URL `http://wsopuppenkiste.wiso.uni-goettingen.de/~dadler`. R package version 0.2.

M. A. Aizerman, E. M. Braverman, and L. I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

A. Asuncion and D. J. Newman. UCI Machine Learning Repository, 2007. URL `http://www.ics.uci.edu/~mlearn/{MLR}epository.html`.

Ran Avnimelech and Nathan Intrator. Boosting regression estimators. *Neural Computation*, 11:491–513, 1999.

Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants, 1998.

James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Kluwer Academic Publishers, Norwell, MA, USA, 1981. ISBN 0306406713.

Christopher M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, November 1995. ISBN 0198538642.

Avrim Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2):245–271, 1997.

Leo Breiman. Random forests. In *Machine Learning*, pages 5–32, 2001.

Leo Breiman. Bagging predictors. In *Machine Learning*, pages 123–140, 1996a.

Leo Breiman. Out-of-bag estimation. Technical report, Statistics Department, University of California, 1996b.

Leo Breiman. Arcing classifiers, 1998.

Gavin Brown. *Diversity in Neural Network Ensembles.* PhD thesis, University of Birmingham, 2004.

Gavin Brown. Ensemble learning. In C. Sammut and G. I. Webb, editors, *Encyclopedia of Machine Learning*. Springer, In Press, expected publication, 2010. ISBN 0-387-307-680.

Gavin Brown, Jeremy Wyatt, Rachel Harris, and Xin Yao. Diversity creation methods: A survey and categorisation. *Journal of Information Fusion*, 6:5–20, 2005a.

Gavin Brown, Jeremy Wyatt, and Ping Sun. Between two extremes: Examining decompositions of the ensemble objective function. In LNCS, editor, *International Workshop on Multiple Classifier Systems*, volume 3541, 2005b.

Gavin Brown, Jeremy L. Wyatt, and Pack Kaelbling. Managing diversity in regression ensembles. *Journ. of Machine Learning Research*, 6:2005, 2005c.

Christopher J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

John G. Carney and Padraig Cunningham. The NeuralBAG algorithm: optimizing generalization performance in bagged neural networks. In *Proc. of the 7th European Symposium on Artificial Neural Networks*, pages 35–40, 1999.

John M. Chambers and Trevor J. Hastie. *Statistical Models in S*. Chapman & Hall, London, 1992. ISBN 9780412830402.

Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

W. S. Cleveland and S. J. Devlin. Locally-weighted regression: An approach to regression analysis by local fitting. *J. of the American Stat. Assoc.* , 83: 596–610, 1988.

P. Cortez and A. Morais. A data mining approach to predict forest fires using meteorological data. In M. F. Santos J. Neves and J. Machado Eds., editors, *New Trends in Artificial Intelligence, Proceedings of the 13th EPIA 2007 - Portuguese Conference on Artificial Intelligence*, pages 512–523, 2007.

M. Cottrell and E. de Bodt. A kohonen map representation to avoid misleading interpretations. In D-Facto, editor, *Proceedings of the European Symposium on Atrificial Neural Networks*, pages 103–110, 1996.

T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967.

B. V. Dasarathy and B. V. Sheela. Composite classifier system design: concepts and methodology. *Proc. of the IEEE*, 67(5):708–713, 1979.

Thomas G. Dietterich. *The Handbook of Brain Theory and Neural Networks, Second Edition*, chapter Ensemble Learning, pages 405–408. (M.A. Arbib, Ed.), Cambridge, MA: The MIT Press, 2002.

Thomas G. Dietterich. Machine-learning research – four current directions. *AI MAGAZINE*, 18:97–136, 1997.

Thomas G. Dietterich and Doug Fisher. An experimental comparison of three methods for constructing ensembles of decision trees. In *Bagging, boosting, and randomization. Machine Learning*, pages 139–157, 2000.

Evgenia Dimitriadou, Kurt Hornik, Friedrich Leisch, David Meyer, and Andreas Weingessel. *e1071: Misc Functions of the Department of Statistics (e1071), TU Wien*, 2009. R package version 1.5-19.

Michael Dittenbach, Dieter Merkl, and Andreas Rauber. The growing hierarchical self-organizing map. In *IJCNN (6)*, pages 15–19. IEEE Computer Society, 2000.

Harris Drucker. Improving regressors using boosting techniques. In Douglas H. Fisher, editor, *ICML*, pages 107–115. Morgan Kaufmann, 1997. ISBN 1-55860-486-3.

Mark Eastwood and Bogdan Gabrys. The dynamics of negative correlation learning. *The Journal of VLSI Signal Processing*, 49(2):251–263, November 2007. doi: 10.1007/s11265-007-0074-5.

J. E Elias, F. D Gibbons, O. D King, F. P Roth, and S. P Gygi. Intensity-based protein identification by machine learning from a library of tandem mass spectra. *Nat Biotechnol*, 22(2):214–219, Feb 2004. doi: 10.1038/nbt930.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting, 1995.

Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *In Proceedings of the Thirteenth International Conference on Machine Learning*, pages 148–156. Morgan Kaufmann, 1996.

Jerome H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–67, 1991. ISSN 00905364. doi: 10.2307/2241837.

Bernd Fritzke. Fast learning with incremental RBF networks. In *Neural Processing Letters*, pages 2–5, 1994.

Bernd Fritzke. Some competitive learning methods, 1997.

S. Gay, P.-A. Binz, D. F Hochstrasser, and R. D Appel. Peptide mass fingerprinting peak intensity prediction: extracting knowledge from spectra. *Proteomics*, 2(10): 1374–1391, Oct 2002. doi: 3.0.CO;2-D.

Stuart Geman, Elie Bienenstock, and Rene Doursat. Neural networks and the bias/variance dilemma. *Neural Comput.*, 4(1):1–58, 1992. ISSN 0899-7667. doi: http://dx.doi.org/10.1162/neco.1992.4.1.1.

Apostolos Georgakis and Haibo Li. An ensemble of som networks for document organization and retrieval. In Timo Honkela, Ville Könönen, Matti Pöllä, and Olli Simula, editors, *Proceedings of AKRR'05, International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning*, pages 141–147, Espoo, Finland, June 2005.

Nils Goerke, Florian Kintzler, and Rolf Eckmiller. Self organized partitioning of chaotic attractors for control. In *ICANN '01: Proceedings of the International Conference on Artificial Neural Networks*, pages 851–856, London, UK, 2001. Springer-Verlag.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003. ISSN 1533-7928.

L. K. Hansen and P. Salamon. Neural network ensembles. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(10):993–1001, 1990. ISSN 0162-8828. doi: http://dx.doi.org/10.1109/34.58871.

J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Applied Statistics*, 28(1):100–108, 1979. ISSN 00359254. doi: 10.2307/2346830.

Sherif Hashem. Optimal linear combinations of neural networks. *Neural Networks*, 10: 599–614, 1994.

Trevor Hastie and Brad Efron. *lars: Least Angle Regression, Lasso and Forward Stagewise*, 2007. URL `http://www-stat.stanford.edu/~hastie/Papers/#LARS`. R package version 0.9-7.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, 2001.

Johan Himberg. A som based cluster visualization and its application for false coloring, 2000.

Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20:832–844, 1998.

Hirotaka Inoue and Hiroyuki Narihisa. Effective pruning method for a multiple classifier system based on self-generating neural networks. In *Artificial Neural Networks and Neural Information Processing - ICANN/ICONIP 2003*, volume 2714/2003, pages 11–18. Springer Berlin / Heidelberg, 2003. ISBN 978-3-540-40408-8. doi: 10.1007/3-540-44989-2.

Yuan Jiang and Zhi hua Zhou. SOM ensemble-based image segmentation. *Neural Processing Letters*, 20:171–178, 2004.

Michael I. Jordan. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 6:181–214, 1994.

Michael I. Jordan and Robert A. Jacobs. *Modular and hierarchical learning systems*, pages 579–582. MIT Press, Cambridge, MA, USA, 1998. ISBN 0-262-51102-9.

L. Kaufman and P. Rousseeuw. *Finding Groups in Data. An Introduction to Cluster Analysis.* Wiley-Interscience, New York, 1990.

S. Kawashima, H. Ogata, and M. Kanehisa. AAindex: Amino Acid Index Database. *Nucleic Acids Res*, 27(1):368–369, Jan 1999.

T. Kohonen. Self-organized formation of topologically correct feature maps. In *Biological Cybernetics*, volume 43, pages 59–69, 1982.

Teuvo Kohonen. *Self-Organizing Maps*, volume 30. Springer Series in Information Sciences, 3rd edition, 2001. ISBN 978-3-540-67921-9.

S B Kotsiantis. Supervised machine learning: A review of classification techniques. *Informatica*, 31(3):249–268, 2007.

Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. In *Adv. in NIPS*, pages 231–238. MIT Press, 1995.

Max Kuhn. *caret: Classification and Regression Training*, 2008. R package v. 3.16.

Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002. URL `http://CRAN.R-project.org/doc/Rnews/`.

Y. Liu and X. Yao. Ensemble learning via negative correlation. *Neurocomputing*, 12 (10):1399–1404, 1999. doi: 10.1016/S0893-6080(99)00073-8.

Richard Maclin. An empirical evaluation of bagging and boosting. In *In Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 546–551. AAAI Press, 1997.

J. B. Macqueen. Some methods of classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

T. M. Martinetz, S. G. Berkovich, and K. J. Schulten. 'Neural Gas' network for vector quantization and its application to time-series prediction. In *IEEE Trans. Neural Networks*, volume 4, pages 558–569, 1993.

John Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philos. Trans. Roy. Soc. London*, 1909.

Risto Miikkulainen. Script recognition with hierarchical feature maps. *Connection Science*, 2:83–101, 1990.

P. J. Millington and W. L. Baker. Associative reinforcement learning for optimal control. In *Proc. Conf. on AIAA Guid. Nav. and Cont.*, volume 2, pages 1120–1128, 1990.

F. L. Minku, Hirotaka Inoue, and Xin Yao. Negative correlation in incremental learning. *Natural Computing*, 8(2):289–320, 2009.

David Opitz and Richard Maclin. Popular ensemble methods: an empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.

David W. Opitz. Feature selection for ensembles. In *In Proceedings of 16th National Conference on Artificial Intelligence (AAAI*, pages 379–384. Press, 1999.

Nikhil R. Pal, James C. Bezdek, and Richard J. Hathaway. Sequential competitive learning and the fuzzy c-means clustering algorithms. *Neural Netw.*, 9(5):787–796, 1996. ISSN 0893-6080.

M. P. Perrone and L. M. Cooper. When networks disagree: Ensemble methods for hybrid neural networks. In R. J. Mammone, editor, *Artificial Neural Networks for Speech and Vision*, pages 126–142, London, UK, 1993. Chapman and Hall.

Lina Petrakieva and Colin Fyfe. Bagging and bumping self organising maps. *Computing and Information Systems Journal*, 2003.

R. Polikar. Ensemble based systems in decision making. *Circuits and Systems Magazine, IEEE*, 6(3):21–45, 2006. ISSN 1531-636X. doi: 10.1109/MCAS.2006.1688199.

E. Prudhomme and S. Lallich. Optimization of self-organizing maps ensemble in prediction. In *International Conference on Data Mining (DMIN'08)*, 2008.

J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, March 1986. doi: 10.1007/BF00116251.

J. R. Quinlan. Bagging, boosting, and c4.5. In *In Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 725–730. AAAI Press, 1996.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Stat. Comp., Austria, 2008. URL `http://www.R-project.org`. ISBN 3-900051-07-0.

Helge Ritter. Learning with the self-organizing map. In T. Kohonen and et al., editors, *Artificial Neural Networks*, pages 379–384. Elsevier Science Publishers, 1991.

Frank Rosenblatt. *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms.* Spartan Books, 1962.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland, et al., editors, *Parallel Distributed Processing: Volume 1: Foundations*, pages 318–362. MIT Press, Cambridge, 1987.

Robert E. Schapire. The strength of weak learnability. *Mach. Learn.*, 5(2):197–227, 1990. ISSN 0885-6125. doi: http://dx.doi.org/10.1023/A:1022648800760.

Alexandra Scherbart. *Lerranco: SOM ensembles*, 2009. URL `http://ani.www.techfak.uni-bielefeld.de/`. R package version 0.1.

Alexandra Scherbart and Tim W. Nattkemper. The diversity of regression ensembles combining bagging and random subspace method. In Mario Köppen, Nikola K. Kasabov, and George G. Coghill, editors, *ICONIP (2)*, volume 5507 of *Lecture Notes in Computer Science*, pages 911–918. Springer, 2008. ISBN 978-3-642-03039-0.

Alexandra Scherbart, Wiebke Timm, Sebastian Böcker, and Tim W. Nattkemper. Neural network approach for mass spectrometry prediction by peptide prototyping. In Joaquim Marques de Sá, Luís A. Alexandre, Wlodzislaw Duch, and Danilo P. Mandic, editors, *ICANN (2)*, volume 4669 of *Lecture Notes in Computer Science*, pages 90–99. Springer, 2007a. ISBN 978-3-540-74693-5.

Alexandra Scherbart, Wiebke Timm, Sebastian Böcker, and Tim W. Nattkemper. Som-based peptide prototyping for mass spectrometry peak intensity prediction. In *WSOM'07*, 2007b.

Alexandra Scherbart, Wiebke Timm, Sebastian Böcker, and Tim W. Nattkemper. Improved mass spectrometry peak intensity prediction by adaptive feature weighting. In Mario Köppen, Nikola K. Kasabov, and George G. Coghill, editors, *ICONIP (1)*, volume 5506 of *Lecture Notes in Computer Science*, pages 513–520. Springer, 2008. ISBN 978-3-642-02489-4.

Bernhard Schölkopf, Peter Bartlett, Alex Smola, and Robert Williamson. Shrinking the Tube: A New Support Vector Regression Algorithm. In *Advances in Neural Information Processing Systems*, 1999.

Udo Seiffert, Barbara Hammer, Samuel Kaski, and Thomas Villmann. Neural networks and machine learning in bioinformatics – theory and applications. In Michel Verleysen, editor, *Proceedings of the 14. European Symposium on Artificial Neural*

Networks ESANN 2006, pages 521–532, Evere, Belgium, 2006. D-Side Publications. ISBN 2-930307-06-4.

I. Shadforth, D. Crowther, and C. Bessant. Protein and peptide identification algorithms using MS for use in high-throughput, automated pipelines. *Proteomics*, 5 (16):4082–4095, Nov 2005. doi: 10.1002/pmic.200402091.

Amanda J. C. Sharkey. On combining artificial neural nets. *Connection Science*, 8: 299–313, 1996.

Amanda J.C. Sharkey, A J. C. Sharkey, and Noel E. Sharkey. Combining diverse neural nets. *The Knowledge Engineering Review*, 12:231–247, 1997.

Alex J. Smola and Bernhard Schölkopf. A tutorial on support vector regression. Technical report, Statistics and Computing, 2003.

H. Tang and et al. A computational approach toward label-free protein quantification using predicted peptide detectability. *Bioinformatics*, 22(14):e481–e488, Jul 2006. doi: 10.1093/bioinformatics/btl237.

Terry M Therneau, Beth Atkinson, and R port by B. Ripley. *rpart: Recursive Partitioning*, 2009. URL `http://mayoresearch.mayo.edu/mayo/research/biostat/splusfunctions.cfm`. R package version 3.1-43.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996.

Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a dataset via the gap statistic. *Journal of the Royal Statistical Society, Series B*, 63:411–423, 2000.

W. Timm, S. Böcker, T. Twellmann, and T. W. Nattkemper. Peak intensity prediction for PMF mass spectra using support vector regression. In *Proc. of the 7th International FLINS Conference on Applied Artificial Intelligence*, 2006.

Wiebke Timm, Alexandra Scherbart, Sebastian Böcker, Oliver Kohlbacher, and Tim W. Nattkemper. Peak intensity prediction in MALDI-TOF mass spectrometry: a machine learning study to support quantitative proteomics. *BMC Bioinformatics*, 9:443, 2008. doi: 10.1186/1471-2105-9-443.

A. Ultsch. Maps for the visualization of high-dimensional data spaces. In *Proceedings Workshop on Self-Organizing Maps (WSOM 2003)*, pages 225–230, 2003a.

A. Ultsch. U*-matrix: a tool to visualize clusters in high dimensional data. Technical Report 36, Dept. of Mathematics and Computer Science, University of Marburg, Germany, 2003b.

A. Ultsch. Self-organizing neural networks for visualization and classification, 1993.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, 1995.

Michel Verleysen. Learning high-dimensional data. In *Limitations and Future Trends in Neural Computation 186*, pages 141–162, 2003.

Juha Vesanto. Som-based data visualization methods. *Intelligent Data Analysis*, 3: 111–126, 1999.

P. Vlachos. Statlib datasets archive, 2005. URL `http://lib.stat.cmu.edu/datasets/`.

Ron Wehrens and Björn-Helge Mevik. *pls: Partial Least Squares Regression (PLSR) and Principal Component Regression (PCR)*, 2007. URL `http://mevik.net/work/software/pls.html`. R package version 2.1-0.

Claus Weihs, Uwe Ligges, Karsten Luebke, and Nils Raabe. klar analyzing german business cycles. In D. Baier, R. Decker, and L. Schmidt-Thieme, editors, *Data Analysis and Decision Support*, pages 335–343, Berlin, 2005. Springer-Verlag.

G. N. Wilkinson and C. E. Rogers. Symbolic descriptions of factorial models for analysis of variance. *Applied Statistics*, pages 392–9, 1973.

H. Wold. Estimation of principal components and related models by iterative least squares. In P.R. Krishnaiah, editor, *Multivariate Analysis*, pages 391–420. Academic Press, New York, 1966.

David H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.

David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION*, 1(1):67–82, 1997.

Jun Yan. *som: Self-Organizing Map*, 2004. R package version 0.3-4.

Gabriele Zenobi and Padraig Cunningham. Using diversity in preparing ensembles of classifiers based on different feature seature subsets to minimize generalization error. In *Lecture Notes in Computer Science*, pages 576–587. Springer Verlag, 2001.

Zhi-Hua Zhou. When semi-supervised learning meets ensemble learning. In Jon Atli Benediktsson, Josef Kittler, and Fabio Roli, editors, *MCS*, volume 5519 of *Lecture Notes in Computer Science*, pages 529–538. Springer, 2009. ISBN 978-3-642-02325-5.

Zhi-Hua Zhou and Wei Tang. Clusterer ensemble. *Knowl.-Based Syst.*, 19(1):77–83, 2006.