Masterarbeit

Towards Optimally Diverse Randomized Ensembles of Artificial Neural Networks

zur Erlangung des akademischen Grades eines Master of Science (M.Sc.)

Anna I. Martin
December 21, 2016

Gutachter:
Prof. Dr. Michael R. Berthold
Prof. Dr. Bastian Goldlücke

Universität Konstanz
Fachbereich für Informatik und Informationswissenschaft
D–78457 Konstanz
Germany
Martin, Anna:
Towards Optimally Diverse Randomized Ensembles of Artificial Neural Networks
Masterarbeit, Universität Konstanz, 2016.
Abstract

The concept of ensemble learning has become exceptionally popular over the last couple decades due to the ability of a group of base classifiers trained for the same problem to often demonstrate higher accuracy than that of a single model. The main idea behind such an ensemble of models, which outperforms a single model, is to combine a set of diverse classifiers. This thesis concentrates on neural networks as base classifiers and explores the influence of the parameters of neural networks, whose randomization leads to generating diverse ensembles with better generalisation ability compared to a single model. With the aim of stimulating disagreement among the members of an ensemble of neural networks, to the data we apply the sampling strategy similar to one implemented by the RANDOM FORESTS algorithm and combine the sampling strategy with the variation of the network parameters. In our experiments we use the entropy measure as the main measure of diversity. Experimental results demonstrate that by random varying different network parameters it is possible to induce diversity to an ensemble of neural networks, but it is not necessarily associated with an improvement in ensemble accuracy.
Acknowledgements

I would like to thank the entire chair of Bioinformatic and Information Mining, including the employees and the fellow students for making the time of writing this thesis valuable and memorable for me. In particular, my special thanks to my two advisers - Prof. Dr. Michael R. Berthold and Prof. Dr. Bastian Goldlücke for their patience, to Oliver Sampson for all the help, support and believing in me even when I did not, to Alexander Fillbrunn for his ideas, help and unfailing positive attitude, to Peter Burger for setting up and administering servers to run the experiments on, to Heather Fyson for all administrative work, to Tim-Oliver Buchholz for his help and setting deadlines for me, to Gabriel Einsdorf for his positive support, to Yelena Kuprienrko, Ferry Abt and Silvio Widmer for their great help with proofreading. My sincere thanks to my family, whose endless love and support is difficult to overestimate.
Contents

List of Figures ix
List of Tables xii
List of Algorithms xiii

1 Introduction 1
   1.1 Supervised Learning and Decision Combination 1
   1.2 Ensemble Methods Application 2
   1.3 Ensemble Diversity 3
   1.4 Neural Networks 3
   1.5 Ensemble of Artificial Neural Networks 4
   1.6 Motivation 5
   1.7 Overview 5

2 Ensemble of Artificial Neural Networks 7
   2.1 Artificial Neural Networks 7
      2.1.1 Perceptron 8
      2.1.2 Decision Boundaries and Multilayer Perceptron 10
      2.1.3 Activation Function 13
   2.2 Backpropagation Algorithm 14
      2.2.1 Learning Rate 16
      2.2.2 RPROP Algorithm 17
   2.3 Ensemble Learning 18
2.3.1 The Ensemble Framework ........................................ 18
2.3.2 Majority Voting ................................................. 20
2.3.3 Why Ensembles Work ........................................... 21
2.4 Constructing Ensembles ............................................. 22
  2.4.1 Data Manipulation ............................................. 23
  2.4.2 Feature Manipulation ......................................... 24
  2.4.3 Injection of Randomness .................................... 25
2.5 Diversity ............................................................... 26
  2.5.1 Diversity and Overfitting .................................... 27
  2.5.2 Overfitting ..................................................... 27
  2.5.3 Measuring Diversity ......................................... 29
  2.5.4 Q-statistics ................................................... 30
  2.5.5 Entropy Measure ............................................. 31
  2.5.6 Kohavi-Wolpert Variance .................................. 32
2.6 Random Forests ...................................................... 32
2.7 Summary ............................................................... 34

3 Randomized Ensemble of Artificial Neural Networks 35
  3.1 Randomized Ensemble Framework ............................... 35
    3.1.1 Data Manipulation and Sampling Strategy Influence .... 37
  3.2 Model Manipulation ............................................. 38
    3.2.1 Topology Regulating Parameters .......................... 38
    3.2.2 Learning Process Regulating Parameters .................. 39
  3.3 Random Connections ............................................ 40
  3.4 More Randomness ................................................ 41
  3.5 Diversity and Accuracy ......................................... 41
  3.6 Summary ........................................................... 42

4 Experimental Results ................................................ 43
  4.1 Experimental Setup ............................................. 43
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2 Benchmark Accuracy</td>
<td>44</td>
</tr>
<tr>
<td>4.3 Best Sampling Parameters</td>
<td>44</td>
</tr>
<tr>
<td>4.4 Choice of Random Parameters</td>
<td>46</td>
</tr>
<tr>
<td>4.5 Reading the Graphs</td>
<td>46</td>
</tr>
<tr>
<td>4.6 The Experimental Parameters</td>
<td>48</td>
</tr>
<tr>
<td>4.7 Experiments</td>
<td>49</td>
</tr>
<tr>
<td>4.7.1 Influence of Sampling</td>
<td>49</td>
</tr>
<tr>
<td>4.7.2 Randomized Ensemble Framework Parameters, Diversity and Accuracy</td>
<td>51</td>
</tr>
<tr>
<td>4.7.3 Random Number of Hidden Neurons, Diversity and Accuracy</td>
<td>55</td>
</tr>
<tr>
<td>4.7.4 Random Number of Hidden Layers, Diversity and Accuracy</td>
<td>57</td>
</tr>
<tr>
<td>4.7.5 Large Number of Hidden Neurons and Hidden Layers, Diversity and</td>
<td>59</td>
</tr>
<tr>
<td>Accuracy</td>
<td></td>
</tr>
<tr>
<td>4.7.6 Random Learning Rate, Diversity and Accuracy</td>
<td>61</td>
</tr>
<tr>
<td>4.7.7 Random Number of Epochs, Diversity and Accuracy</td>
<td>63</td>
</tr>
<tr>
<td>4.7.8 Random Connections, Diversity and Accuracy</td>
<td>65</td>
</tr>
<tr>
<td>4.7.9 Combination of Randomized Parameters, Diversity and Ensemble</td>
<td>67</td>
</tr>
<tr>
<td>Accuracy</td>
<td></td>
</tr>
<tr>
<td>4.8 Additional Experiments</td>
<td>72</td>
</tr>
<tr>
<td>4.9 Comparison of Approaches</td>
<td>72</td>
</tr>
<tr>
<td>4.10 Summary</td>
<td>79</td>
</tr>
<tr>
<td>5 Conclusion and Future Work</td>
<td>81</td>
</tr>
<tr>
<td>5.1 Future Work</td>
<td>82</td>
</tr>
<tr>
<td>Appendices</td>
<td>83</td>
</tr>
<tr>
<td>A Additional Graphical Results</td>
<td>84</td>
</tr>
<tr>
<td>B Numerical Experimental Results</td>
<td>90</td>
</tr>
<tr>
<td>B.1 Sampling Influence Strategy</td>
<td>90</td>
</tr>
<tr>
<td>B.2 Random Proportion of Training Patterns</td>
<td>92</td>
</tr>
</tbody>
</table>
B.3 Random Proportion of Features ............................................. 93
B.4 Random Number of Hidden Neurons ................................. 94
B.5 Random Number of Hidden Layers ..................................... 95
B.6 High Number of Hidden Neurons and Hidden Layers .......... 96
B.7 Random Learning Rate ...................................................... 97
B.8 Random Connections ......................................................... 98
B.9 Random Number of Epochs ............................................... 99
B.10 Combination of Randomized Parameters .......................... 100

References .............................................................................. 103
List of Figures

2.1 Artificial neuron ................................................. 7
2.2 Perceptron example and hyperplane with solution vector and solution region ........................................... 8
2.3 Possible decision boundaries for ANN output ........................................... 10
2.4 Artificial Neural Network classifier topology with one hidden layer ........................................... 11
2.5 Multilayer perceptron with one hidden layer ........................................... 12
2.6 Sigmoid functions ................................................. 14
2.7 Ensemble framework scheme ........................................... 19
2.8 Ensemble accuracy with majority voting on a binary classification problem ........................................... 20
2.9 Graphical explanation why an ensemble may outperform a single classifier ........................................... 22
2.10 Example of the ensemble of three neural networks and the result of their voting ........................................... 23
2.11 Example of overfitting ................................................. 28
2.12 Example of overfitting with prediction on test and training data ........................................... 29
3.1 Space of possible training sets for an ensemble ........................................... 36
3.2 Neural network with randomly deleted connections ........................................... 41
4.1 Influence of the sampling strategy on ensemble accuracy ........................................... 49
4.2 Ensemble accuracy and diversity distributions in the experiments on sampling strategy influence ........................................... 50
4.3 Influence of sampling strategy on ensemble diversity and accuracy ........................................... 51
4.4 Influence of the random training patterns proportion range on ensemble accuracy and diversity ........................................... 52
4.5 Ensemble accuracy and diversity distributions in the experiments on the training patterns proportion $D_p$ variation .......................... 53
4.6 Influence of random feature proportion on ensemble accuracy and diversity ................................................................. 54
4.7 Ensemble accuracy and diversity distributions in the experiments on random feature proportion range variation ......................... 54
4.8 Influence of the range of random number of hidden neurons on ensemble diversity and accuracy ............................................. 55
4.9 Aggregated results: influence of the range of random number of hidden neurons with and without sampling ................................. 56
4.10 Influence of the range of the random number of the hidden layers on ensemble diversity and accuracy ....................................... 57
4.11 Ensemble accuracy and diversity distribution in the experiments on the random number of hidden layers range variation .............. 58
4.12 Influence of the high number of the hidden neurons and high number of the hidden layers on ensemble diversity and accuracy ........ 59
4.13 Aggregated results: influence of the high number of the hidden neurons and high number of the hidden layers ...................... 60
4.14 Influence of the range of the random learning rate on ensemble diversity and accuracy ......................................................... 61
4.15 Ensemble accuracy and diversity distributions in the experiments on the random learning rate range variation ......................... 62
4.16 Influence of the range of the random number of epochs on ensemble diversity and accuracy .................................................. 63
4.17 Ensemble accuracy and diversity distributions in the experiments on the random number of epochs range variation .................. 64
4.18 Influence of the range of the proportion of randomly deleted connections on ensemble diversity and accuracy ......................... 65
4.19 Ensemble accuracy and diversity distributions in the experiments on the proportion of randomly deleted connections $C_n$ variation . . . . . . . 66
4.20 Combination of randomized parameters . . . . . . . . . . . . . . . . . 67
4.21 Ensemble accuracy and diversity distributions in the experiments on the combination of randomized parameters . . . . . . . . . . . . . . 68
4.22 Combination of randomized parameters with proportion of random connections and number of the hidden layers . . . . . . . . . . . . 69
4.23 Ensemble accuracy and diversity distributions in the experiments on combination of randomized parameters with the proportion of random connections and the number of the hidden layers . . . . . . . 70
4.24 Aggregated diversity distribution on the approaches without sampling 74
4.25 Aggregated diversity distribution on the approaches with sampling . . 75
4.26 Aggregated accuracy distribution on the approaches without sampling 76
4.27 Aggregated accuracy distribution on the approaches with sampling . . 77
4.28 Ensemble accuracy and diversity distributions in the experiments on the approaches with and without sampling . . . . . . . . . . . . . . 77
A.1 Influence of number of hidden neurons on single network accuracy . . 84
A.2 Aggregated results: number of neurons influence on single network . . 85
A.3 Comparison between performance on test and training data with different number of the hidden neurons . . . . . . . . . . . . . . . . . . . 85
A.4 Comparison of influence of the number of epochs on network performance with different number of the hidden neurons . . . . . . . . . 86
A.5 Influence of the number of hidden layers and hidden neurons on single network performance . . . . . . . . . . . . . . . . . . . . . . . . . . . . 87
A.6 Comparison between performance on test and training data with different number of the hidden layers . . . . . . . . . . . . . . . . . . . . . . . . . 88
A.7 Comparison of influence of the number of epochs on network performance with different number of layers . . . . . . . . . . . . . . . . . . 89
List of Tables

2.1 Q-statistics: Relationship between two classifiers .......................... 31

4.1 List of used data sets ....................................................... 43

4.2 Parameters and accuracy from the experiments with the best performance for a single network and within the RANDOMISED ENSEMBLE Framework .................................................. 44

4.3 Parameters used for the experiments ...................................... 48

4.4 Comparison of the accuracy and diversity figures aggregated for all data sets for the approaches with and without sampling ............... 78

4.5 List of hypotheses with results ........................................... 80

B.1 Sampling strategy influence: Comparison between features and training patterns sampling ...................................................... 91

B.2 Random training patterns proportion: Comparison between sampling and nosampling approaches ................................. 92

B.3 Random feature proportion: Comparison between sampling and nosampling approaches ................................................. 93

B.4 Random number of hidden neurons: Comparison between sampling and nosampling approaches ....................................... 94

B.5 Random number of hidden layers: Comparison between sampling and nosampling approaches .................................... 95

B.6 High number of hidden neurons and hidden layers: Comparison between two approaches ............................................... 96
B.7 Random learning rate value: Comparison between sampling and nosampling approaches ........................................ 97
B.8 Random Connections: Comparison between sampling and nosampling approaches ........................................ 98
B.9 Random number of epochs: Comparison between sampling and nosampling approaches ........................................ 99
B.10 Combination of randomized parameters with proportion of random connections and number of the hidden layers ................. 100
B.11 Combination of randomised parameters with proportion of random connections and number of the hidden layers ................. 101
List of Algorithms

2.1 The Backpropagation Algorithm ........................................... 16
2.2 The Bagging Algorithm ..................................................... 24
2.3 The Random Subspace Method Algorithm ............................. 25
2.4 The Random Forests Algorithm ......................................... 33
3.1 The Randomized Ensemble Framework Algorithm ................. 36
Chapter 1

Introduction

The idea of mixing several opinions is intrinsic to the nature of human beings. We tend to seek several points of view when it comes to making an important decision like going to several doctors before having some medical procedure or reading user reviews before ordering an item [Pol12]. We also trust a group of people to whom we delegate decision making on our behalf like parliaments, juries or committees.

The theoretical basis for democracy is built upon majority decisions. Expressed by Marquis de Condorcet in 1785 in *Essay on the Application of Analysis to the Probability of Majority Decisions* [Con85], the Condorcet’s Jury theorem speaks about a group of people who need to make a decision with two possible outcomes. The theorem states that if the probability of a voter voting correctly is $p > 0.5$, then the probability of the majority vote to be correct $P$ approaches 1 as the number of voters increases.

In 1907 Sir Francis Galton published in *Nature* an article called “Vox Populi” on a weight-judging competition in a livestock fair at Plymouth, Cornwall. The participants of the competition were asked to estimate dressed weight of an ox. Some of the competitors were experts, like butchers and farmers, in judging cattle, while some were just ordinary fair visitors. All in all there were 787 ballots that Galton was able to study afterwards. Surprisingly, the calculated median on all votes was within 0.8 percent of the measured ox weight while the mean had zero error [W+14].

The concept of combination of the opinions is successfully employed in supervised machine learning.

1.1 Supervised Learning and Decision Combination

The purpose of supervised learning, a machine learning technique, is to produce a function, describing the underlying relationships for labelled training data in order to be able to correctly determine and predict unlabelled instances.

A training set for supervised learning usually consists of instances comprising two elements: an input feature vector, which describes each training instance by the set of
features, and a desired output or class label. The number of features of a training data set is called the dimensionality of this data set. A model which is built using a machine learning algorithm and able to map unseen examples is called classifier.

The principles of the majority rule, described in the previous section, can be applied for supervised learning in the ensemble methods. In contrast to the ordinary approaches that train a single model on training data, the main idea behind ensemble methods is a combination of a set of models (classifiers) trained for the same problem on the same data. However, the set of classifiers in an ensemble should have their own strengths and weaknesses, that is, to be experts on different part of training data and to make errors on different training instances.

The concept of supervised learning with ensembles has been explored since the late Seventies when in 1977 Tukey [Tuk77] proposed to combine two linear regression predictors: one for the initial training data and another one for the residuals. The idea was further developed in 1979 by Dasarathy and Sheela [DS79] who suggested partitioning an input feature space with an ensemble consisting of the linear and nearest neighbour classifiers.

Ensemble methods have become a subject of intensive research since the 1990s when Hansen and Salmon [HS90] showed that an ensemble of neural networks is more likely to outperform a single network. Another significant contribution to the research of ensemble methods was made by Schapire [Sch90] who demonstrated that several weak classifiers can be combined into a strong classifier, which is difficult to achieve for real-world problems in case of a single model. Weak classifier is defined as a classifier which predicts slightly better than random guessing; strong classifier refers to a predictor, well-correlated with the true classification.

1.2 Ensemble Methods Application

Today’s popularity of the concept of ensemble learning can be explained by the superior prediction ability of the ensembles. On the most famous data mining competition KDD-Cup, which includes a range of diverse tasks such as network intrusion detection, protein locale prediction, customer relationship management, music recommendation etc., ensemble methods won more competitions than any other approach [Zho12].

Ensemble methods were intensively and successfully used for various real-world problems such as object detection [VJ04], face recognition [HZZC00, LFG+01], gender and ethnic origin classification [GHJW00], object tracking [Avi07], and intrusion detection [GRD03]. They have been broadly applied to medical diagnosis tasks as lung cancer cell identification [ZJYC02], Alzheimer’s disease diagnosis [PTP+08], and other daily tasks such as weather forecasting [MKA04, GR05] and electric load prediction [TB02].
1.3 Ensemble Diversity

When constructing an ensemble of classifiers it is important to create an ensemble of diverse classifiers. Although there is no universal definition of this term in the machine learning community, diversity refers to the difference between individual classifiers. It is intuitively clear that similar models will produce similar results and, when combined, will not yield any improvement in performance. Therefore diverse classifiers is the key issue in the ensemble approach.

Tumer and Ghosh [TG96] showed that the average expected added error of an ensemble can be described as follows:

\[ E_{\text{ave}}(H) = \frac{1 + \sigma(N - 1)}{N} E_{\text{add}}(h) \]  

(1.1)

where \( E_{\text{add}}(h) \) is the expected added error of the individual classifier \( h \), \( N \) is the number of the classifiers in the ensemble and \( \sigma \) is the total correlation between classifiers.

Equation 1.1 reveals that if individual classifiers are independent, i.e., \( \sigma = 0 \), then the second part of the numerator will disappear and the ensemble error will be reduced with the coefficient \( N \). In the contrary, if the classifiers are highly correlated, i.e., \( \sigma = 1 \), then the error of the ensemble equals to the initial error and no improvement will be accomplished. This equation demonstrates how important the term of diversity is for ensembles.

Nevertheless, creating a set of diverse classifiers is not a trivial problem since both the task and the data, the classifiers are trained for, are identical, and therefore the models in the ensemble are highly correlated. However, it is possible to create an ensemble of diverse classifiers with the approaches introduced in Chapter 2.

The problem is even more complicated considering that the individual classifiers should not be too erroneous, since a combination of a set of extremely badly performing classifiers could yield an even worse performance. At the same time the classifiers should not be too accurate, as one could expect highly accurate models, trained for the same problem, to be very similar. Therefore, the classifiers should be both diverse and accurate [Zho12].

1.4 Neural Networks

The average human brain comprises about \( 1.5 \cdot 10^{10} \) neurons of different types. Each neuron receives signals through over \( 10^4 \) synapses. This type of complexity is the reason why the human brain is agreed to be the most complex biological structure on earth [Lau91].

Neurons exchange electric signals in the form of impulses in the voltage of the cell membranes. The connections between the neurons are carried out by synapses located in dendrites - cell’s branches. Considering the number of connections to each
neuron it is apparent that a neuron constantly receives a great number of signals, sums it up and transmits through its axon, as a voltage change along the membrane, in case the sum of the signals exceeds a certain threshold. In this case the neuron is said to fire. Whether a neuron fires or not is determined by the type of incoming signal. Inhibitory signals prevent the neuron from firing, while excitatory signals encourage it to emit an impulse. [Gur97]. The mathematical model of the neuron was inspired by the ability of a human brain to learn and adapt.

Artificial neural networks (ANNs) have been a subject of research for several decades now, aimed to imitate human behaviour in real-world tasks. ANNs will be thoroughly explored in this thesis as a major component of the ensembles of artificial neural networks.

1.5 Ensemble of Artificial Neural Networks

Ensembles of ANN have already been a subject of some extensive research. Usually each individual network is trained separately from the others, then the predictions are merged by majority vote for a classification or by averaging for a regression problem.

Perron and Cooper in [PC91] created an ensemble by generating a population of neural networks for solving real-world regression problems. They demonstrated that the ensemble is producing better result than any of its members.

Cunningham et al. [CCJ00] explored the stability problem for feed-forward neural networks trained with the BACKPROPAGATION algorithm. The problem occurred as a consequence of high variance intrinsic to a single network, especially to the networks trained on different data subsets. The solution suggested by the authors is to build an ensemble of neural networks which produces a single prediction by averaging. This sustains ensemble stability eliminating high variance and disagreement among ensemble members and higher accuracy of its predictions.

Hansen and Salmon [HS90] also showed an improvement of the generalization error by exploiting an ensemble of similar neural networks. The authors claimed that it is profitable to use all networks in the ensemble, even those performing significantly worse than the most accurate model in the ensemble. They also suggested that better classification accuracy can be achieved by tuning available parameters of the models and using different subsets of the training data.

Hashem in [Has97] used the outputs of a set of neural networks, referred as optimal linear combinations (OLCs), to remarkably improve the resulting model accuracy. He also named OLCs as a facility, which is able to substitute excessive training and exhaustive network topology exploration in order to find the best topology, suitable for a particular problem.

Krogh and Vedelsby in [KV95] employed an ensemble of neural networks for continuous valued functions to improve prediction accuracy. They explored the ambiguity notion as a measure of disagreement among ensemble members and used it in com-
bination with cross-validation to get a reliable estimation of the generalization error. Ambiguity is defined as the variation of the ensemble members’ output, averaged over unlabelled data; it quantifies the disagreement among the classifiers [KV95]. The authors demonstrated how to make identical networks to disagree by holding out some training examples for each model during the training process. Additionally, they proposed a method for active learning when the most “difficult” examples from the test data, which produce strong disagreement among ensemble members, are marked and included to the training set afterwards.

1.6 Motivation

The application of the ensemble methods and neural networks is extremely broad. Predictive analytics with its ability to learn patterns from data, to predict and clarify previously unseen instances is not only beneficial for various areas such as transportation, biology, sociology, business and politics, it saves lives, improves health care and makes it possible to cut operating expenses. Therefore, more accurate prediction means not only saved money but considerably improved well-being.

1.7 Overview

In this thesis we explore how the utilisation of different approaches, influencing both data and the parameters of neural networks, affects diversity in the ensemble and how diversity in the ensemble correlates with accuracy.

The thesis is structured as follows: Chapter 2 reviews the literature on neural networks and ensemble learning. We start with the description of the properties of artificial neurons and continue with the most popular network types, their parameters, characteristics, geometric perspectives of the function they produce, and the learning mechanisms. The neural network part is concluded with the main algorithm employed for their training and the RPROP (resilient backpropagation) algorithm description applied for the training in our experiments.

The ensemble learning part gives an overview of the concept of ensemble learning, its main components and the most popular mechanism on prediction aggregation. We describe the theoretical reasons why an ensemble performs better than a single predictor and look at the ways of constructing an ensemble of diverse classifiers.

The following sections are devoted to the term of diversity and to understanding how overfitting can be connected to this term and its link to ensemble performance. The chapter is wrapped up with the narrative on the mechanism of work of the RANDOM FORESTS algorithm; the data sampling principles of this algorithm are used for our Randomized Ensemble Framework.
Chapter 3 gives an idea of the RANDOMIZED ENSEMBLE Framework and its strategies of data sampling. It outlines the approaches chosen in order to create an ensemble of diverse neural networks, concentrating on the two main techniques - data manipulation and model manipulation. Together with the description of the motivation behind each of the approaches we propose the respective hypotheses.

Chapter 4 details the experimental setup and data sets used for our experiments as well as parameters controlled during the tests. Later in the chapter we introduce the results obtained during the experiments.

In Chapter 5 we discuss the results from the previous chapter and suggest further steps which can be made in the direction towards diverse and accurate ensembles of artificial neural networks.
Chapter 2

Ensemble of Artificial Neural Networks

2.1 Artificial Neural Networks

ARTIFICIAL NEURAL NETWORKS (ANNs) is a family of statistical learning models inspired by biological neural networks. They are known since the introduction of the first artificial neuron by McCulloch and Pitts in 1943 [MP43].

![Artificial neuron diagram](image)

Figure 2.1: Artificial neuron, applying weighted sum of inputs.

An ANN can be viewed as a mapping of \( m \) inputs to \( c \) outputs. It consists of mutually connected artificial neurons, each of which is a mathematical prototype of a neuron (Figure 2.1), where connections act as synapses and weights represent the intensity of input signals. Each neuron performs a weighted sum of the input signals and outputs the signal after applying an activation function on it. This means that a neuron fires if the weighted sum is greater than the threshold specified by the activation function. A neuron is said to fire when it sends a signal to other neurons along its connections [Abr05, Fyf05].
The first artificial network, introduced in 1957 by Rosenblatt for modelling pattern recognition abilities, was the PERCEPTRON algorithm [Ros57] (Figure 2.2a). However, Rosenblatt also proved, that the PERCEPTRON is not capable of recognizing many classes and can be trained to learn only linearly separable patterns [MP90], as it is not able to learn the EXCLUSIVE DISJUNCTION FUNCTION (XOR). The incorrect belief that the same holds for multilayer networks and miss-citation of his work is the main reason that ANN fell out of favour in the machine learning community for almost 20 years.

Figure 2.2: (a) PERCEPTRON producing weighted sum $z$ of input vector $\vec{x}$ and bias $b$, and applying activation function on it; (b) hyperplane (solid yellow stroke) in the input space with two possible hyperplanes (dotted yellow strokes), solution vector $w$ perpendicular to the hyperplane and the solution region (light red triangle).

Nevertheless, in 1973 Grossberg introduced the network capable of modelling the XOR function [Gro82] and in 1974 Paul Werbos described the process of training neural networks with the BACKPROPAGATION algorithm [Wer94]. This idea was further developed in 1986 by David Rumelhart [RHW88]. The BACKPROPAGATION algorithm makes the process of training faster than before and allowed the use of neural networks to solve problems that had been previously unsolvable. By this ANNs have been gradually resurgent and today are known as a powerful class of machine learning algorithms.

### 2.1.1 Perceptron

PERCEPTRON is a supervised learning algorithm for learning a binary classifier. It models a function mapping an input value $x$ to an output value $f(x)$ using a threshold activation function as follows:
$$f(x) = \begin{cases} 1, & \text{if } z \geq 0 \\ 0, & \text{otherwise} \end{cases} \quad (2.1)$$

where scalar value $z$ is defined as

$$z = \vec{w}^T \vec{x} + b \quad (2.2)$$

where $\vec{x}$ is a real-valued vector of features, $\vec{w}$ is a real-valued vector of weights that describes the direction of the decision hyperplane (Figure 2.2b)

$$\vec{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix} \quad (2.3)$$

and $\vec{w}^T \cdot \vec{x}$ is the dot product of two vectors

$$\vec{w}^T \cdot \vec{x} = \sum_{i=1}^{m} w_i x_i \quad (2.4)$$

where $m$ is the number of features in the input vector $\vec{x}$. The parameter $b$ in Equation 2.2 is the bias. It is required for shifting the decision hyperplane away from the origin and independent from the input vector $\vec{x}$. It is usually combined with weight vector $\vec{w}$ and input vector $\vec{x}$ into the compound vectors $\hat{\vec{w}}$ and $\hat{\vec{x}}$:

$$\hat{\vec{w}} = \begin{bmatrix} w_1 \\ \vdots \\ w_m \\ b \end{bmatrix} \quad (2.5)$$

$$\hat{\vec{x}} = \begin{bmatrix} x_1 \\ \vdots \\ x_m \\ 1 \end{bmatrix} \quad (2.6)$$

Considering Equations 2.5 and 2.6, Equation 2.2 can be written as

$$z = \hat{\vec{w}}^T \hat{\vec{x}} \quad (2.7)$$

The task is to find the solution vector $\hat{\vec{w}}$ which characterises the direction of the hyperplane separating one class from another (Figure 2.2b) such that the number of misclassified training instances is minimal. If the training data is linearly separable there exists such a hyperplane. However the solution vector is not unique (as well as the hyperplane) and lies within the solution region [Gal15b] (the light red region in figure 2.2b).
Definition 2.1.1. Two sets of points $A$ and $B$ in an $m$-dimensional space are called **linearly separable** if $m + 1$ real numbers $w_1, \ldots, w_{m+1}$ exist, such that every point $(x_1, x_2, \ldots, x_m) \in A$ satisfies $\sum_{i=1}^{m} w_i x_i \geq w_{m+1}$ and every point $(x_1, x_2, \ldots, x_m) \in B$ satisfies $\sum_{i=1}^{m} w_i x_i < w_{m+1}$ [Roj13].

If the training samples are not linearly separable, PERCEPTRON will not converge [Ros57]. This problem can be solved by adding a second level of neurons to the network.

### 2.1.2 Decision Boundaries and Multilayer Perceptron

The decision boundaries developed by an ANN depend on the network topology. Let us consider three main cases:

1. **Single layer.** The architecture of a PERCEPTRON described above: the algorithm, which is capable to place a hyperplane in the input space, i.e., suitable for **linearly separable** problems (Figure 2.3a).

![Figure 2.3: Possible decision boundaries for ANN output: (a) a hyperplane in the input space placed by PERCEPTRON, (b) a convex region produced by MULTILAYER PERCEPTRON (MLP) with one hidden layer, (c) arbitrary decision boundaries output by MLP with two hidden layers. The figures are taken from [Gal15a].](image)

2. **Two layers.** An example of the topology of MULTILAYER PERCEPTRON (MLP) with one hidden layer: the algorithm is able to produce a decision boundary in...
the form of a convex region in the input space (Figure 2.3b), i.e., it can realize a non-linearly separable function. In contrast to the Perceptron, an MLP, is a universal approximator [HSW89].

3. **Three layers.** MLP with two hidden layers, which is capable to create arbitrary decision boundaries (Figure 2.3c).

MLP is a feed-forward network that maps a set of input data onto a set of appropriate outputs. In feed-forward networks the signal goes from input to output neurons. As demonstrated in Figure 2.4, an MLP consists of multiple layers of neurons with each layer being fully connected to the next one. Except for the input nodes, each node is a neuron with a non-linear activation function.

![Artificial Neural Network classifier topology](image)

Figure 2.4: Artificial Neural Network classifier topology with one hidden layer, an input layer with m neurons (dimensionality of the input vector $\vec{x}$), the output vector $\vec{y}(x)$ and c output neurons (the number of class labels).

An MLP consists of three types of layers: input, hidden and output.

1. **Input layer.** Receives the data vector $\vec{x}$. The number of neurons on this layer is equal to the number of features $m$ in the data set.

2. **Hidden layer.** Receives the outputs from the previous layer, weights them and passes through non-linear activation function.

   The number of hidden layers in MLP may be more than one. However, in many applications one hidden layer is sufficient for the majority of problems, since, given a non-linear activation function (see section 2.1.3) and enough neurons on hidden layer, the network can closely approximate any decision boundaries with one hidden layer of neurons [Cyb89, Bau88].

3. **Output layer.** Accepts the weighted output from the final hidden layer, passes it through an output activation function and generates the final output. The number of neurons on the output layer depends on the type of problem to be
2.1. Artificial Neural Networks

Figure 2.5: Multilayer Perceptron with one hidden layer and \( h \) neurons on it, input vector \( \vec{x} \) with \( m \) features and output layer with \( c \) neurons equivalent to the number of classes. Weight vectors \( \vec{w}_{ij} \) and \( \vec{w}_{jk} \) regulate respectively the strength of the signal on the hidden and output layers.

solved. Given a regression problem, the output layer comprises only one neuron which outputs the resulting value. For a classification problem, the output layer has one neuron for each possible class label in the data set.

Let us consider an MLP (Figure 2.5) with one hidden layer consisting of \( h \) neurons. The number of output neurons is equal to the number of classes \( c \); the input signal is described by an input vector \( \vec{x}_n = \{x_1, \ldots, x_m\} \) of dimensionality \( m \), synapses’ strength from the input neuron \( i \) to the hidden neuron \( j \) by the weight \( w_{ij} \), synapses’ strength from the hidden neuron \( j \) to the output neuron \( k \) by the weight \( w_{jk} \); the bias terms are represented by the vectors \( \vec{b}_j \) and \( \vec{b}_k \) for the hidden layer neurons and the output layer neurons respectively.

The output signal of the neuron \( o_{j}^{hid} \) on the hidden layer is a weighted sum of its inputs connected to the neurons on the input layer:

\[
o_{j}^{hid} = \phi(z_i) = \phi(\vec{w}_{ij}^T \cdot \vec{x}_n + b_j) = \phi(\sum_{i=1}^{m} w_{ij} x_i + b_j)
\]  
(2.8)

Then the output signal of the neuron \( o_{k}^{out} \) on the output layer is a weighted sum of the inputs from the neurons on the hidden layer:

\[
o_{k}^{out} = \phi(z_k) = \phi(\vec{w}_{jk}^T \cdot \vec{o}_{j}^{hid} + b_k) = \phi(\sum_{j=1}^{h} w_{jk} o_{j}^{hid} + b_k)
\]  
(2.9)

Combining of Equations 2.8 and 2.9 gives the expression for the network output:
In the Equations 2.8, 2.9 and 2.10 φ stands for the activation function which transforms the input signal and regulates the neuron output. It is described in more detail below.

### 2.1.3 Activation Function

The typical activation function used by Perceptron is the threshold (linear) step activation function with binary output signal \( f(x) \), produced as follows:

\[
 f(x) = \begin{cases} 
 1, & \text{if } \vec{w}^T \cdot \vec{x} \geq \theta \\
 -1, & \text{otherwise} 
\end{cases} \quad (2.11)
\]

where \( \theta \) is the threshold value.

This activation function realizes the threshold logic: if the weighted sum exceeds the threshold \( \theta \) then it outputs 1, otherwise it outputs \(-1\) (Figure 2.2).

An essential condition for gradient-based optimization methods used for minimising the error function and calculating weight updates is a continuously differentiable activation function. The threshold activation function is not differentiable at 0, and it differentiates to 0 for all other values, therefore gradient-based methods are not suitable for it [Sny05]. In contrast to Perceptron, in MLP each neuron uses a non-linear activation function, for which the derivative can be easily calculated.

The more commonly used non-linear activation function is a special case of the logistic function: the sigmoid function. It is applied by each neuron on the input signal:

\[
 s_{\log}(z) = \frac{1}{1 + e^{-\lambda z}} \quad (2.12)
\]

The constant \( \lambda \) can be arbitrarily selected and regulates the steepness of the function. Higher values of \( \lambda \) transform the sigmoid shape such that it is closer to the one corresponding to the step function, with increasing \( \lambda \to \infty \) the sigmoid function converges to the step function [Roj13]. The shape of sigmoid changes accordingly with \( \lambda \) and can be seen in Figure 2.6a where the blue line corresponds to \( \lambda = 1 \) while green represents a sigmoid function with \( \lambda = 3 \).

The output of the neuron \( o \) is calculated by applying the sigmoid function \( s_{\log} \) to the weighted sum of neuron inputs \( z \) for forward computation. The error on each neuron for backward computation is calculated by calculating the derivative \( y' \) for Equation 2.12 with respect to \( z \):

\[
 y' = \frac{d}{dz} s_{\log}(z) = s_{\log}(z)(1 - s_{\log}(z)) \quad (2.13)
\]
2.2. Backpropagation Algorithm

The process of training of neural networks requires an algorithm which minimizes the differences between the desired and actual outputs by adjusting the network
weights. The most common algorithm used with ANNs is the BACKPROPAGATION algorithm [RHWW88].

The algorithm was first described by Paul Werbos in 1974 [Wer74]. It is used in conjunction with gradient descent and the chain rule to iteratively compute gradients for each layer while looking for the minimum of the error function in the weight space. Consequently, the solution of the learning problem is nothing else than the combination of weights minimizing the error function $E$, or in other words, throughout the learning process weights between each two connected neurons of the network are changed such that the difference between the network outputs and the true function is minimized [RG00].

First, the chain rule is recurrently applied in order to calculate the error function $E$ with respect to all the weights in the network [RB93]:

\[
\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial o_j} \frac{\partial o_j}{\partial z_j} \frac{\partial z_j}{\partial w_{ij}}
\] (2.16)

where $w_{ij}$ is the weight from neuron $i$ to neuron $j$, $z_j$ is the weighted sum of the inputs for neuron $j$ and $o_j^{hid}$ is the output of the activation function on the neuron $j$.

The common choice for the error function is the sum of squared errors (SSE) which is the sum over squared differences between the target and the desired output for all output neurons $o_{k^{out}}$ and all available training examples $x \in \mathcal{X}$:

\[
E_{SSE} = \frac{1}{2} \sum_{x \in \mathcal{X}} \sum_{k=1}^{c} (y_k(x) - o_{k^{out}}(x))^2
\] (2.17)

where $c$ is the number of output neurons, $y_k(x)$ and $o_{k^{out}}(x)$ are the target and the actual outputs on the training instance $x$ respectively.

Then, knowing partial derivative for each weight, gradient descent is applied to calculate weight updates:

\[
w_{ij}(t+1) = w_{ij}(t) - \epsilon \frac{\partial E}{\partial w_{ij}}(t)
\] (2.18)

where $\epsilon$ is the learning rate (described in the the next section).

The BACKPROPAGATION algorithm (see Algorithm 2.1) can be divided into several steps:

1. **Feed-forward computation**: propagate a training pattern through the neural network and generate the outputs on the output neurons.
2. **Calculate errors of output neurons**: consider the difference between target and computed output.
3. **Update output layer weights**: calculate weight updates of all output neurons by multiplying output error and activation output; update the weights.
4. **Calculate (back-propagate) hidden layer errors**: take the errors from the output neurons and run them back through the weights to get the hidden layer errors.

5. **Update hidden layer weights**: calculate weight updates of all hidden neurons by multiplying hidden layer error and activation output; update the weights.

**Algorithm 2.1 The Backpropagation Algorithm [HPK11]**

**Data**: Training data $\mathcal{X} = \{X_1, X_2, ..., X_N\}$ with $X_n = (x_n, y_n)$, where $y_n$ is the correct class label.

**Double $\epsilon$ - learning rate.**

**A multilayer feed-forward network.**

**Return**: trained neural network.

1: Initialize all weights $w_{ij}$ in network
2: while terminating condition is not satisfied do
3:   for each training example $X_n$ do
4:     // Propagate the inputs forward
5:       for each input layer neuron $j$ do
6:         $O_j = x_{ij}$ // output of an input neuron is its actual input value
7:     end for
8:     for each hidden or output layer neuron $j$ do
9:       $z_j = \sum_i w_{ij}O_i + b_j$ // compute the net input of neuron $j$ with respect to the previous layer $i$ and bias $b_j$
10:      $O_j = \frac{1}{1+e^{-z_j}}$ // compute the output of each neuron $j$
11:     end for
12:     // Backpropagate the errors
13:     for each neuron $j$ in the output layer do
14:       $Err_j = O_j(1-O_j) (T_j-O_j)$ // compute the error considering target output $T_j$
15:     end for
16:     for each neuron $j$ in the hidden layers, from the last to the first hidden layer do
17:       $Err_j = O_j(1-O_j) \sum_k Err_k w_{jk}$ // compute the error with respect to the next higher layer $k$
18:     end for
19:     for each weight $w_{ij}$ in network do
20:       $\Delta w_{ij} = \epsilon Err_j O_i$ // calculate weight update
21:       $w_{ij} = w_{ij} + \Delta w_{ij}$ // update weight
22:     end for
23:     for each bias $b_j$ in network do
24:       $\Delta b_j = \epsilon Err_j$ // calculate bias update
25:       $b_j = b_j + \Delta b_j$ // update bias
26:     end for
27:   end for
28: end while

**2.2.1 Learning Rate**

In the Equation 2.18 $\epsilon$ is the learning rate parameter. The choice of this parameter is important since it scales the derivative and defines how soon the minima of the error function is reached. If it is too small, the time needed for the algorithm to converge...
is too high. On the other hand, if $\epsilon$ is too large, it might lead to oscillations and the algorithm will fail to converge [RB93].

With an eye to solving the learning rate problem, a momentum-term $\mu$ was introduced [RB93]:

$$\Delta w_{ij}(t) = -\epsilon \frac{\partial E}{\partial w_{ij}}(t) + \mu \Delta w_{ij}(t - 1)$$

(2.19)

Here the parameter $\mu$ adjusts the effect of the previous weight update on the current one. When the gradient keeps pointing to the same direction, the step size will be increased. And the other way around, in case of oscillations, when the gradient keeps pointing to different directions, the momentum will decrease and thereby ease fluctuation.

However, when the direction of the gradient is the same from step to step, using both large learning rate $\epsilon$ and momentum $\mu$ will result in huge steps to the local error minima. Therefore choosing the optimum value of the momentum parameter $\mu$ might be another important problem.

### 2.2.2 RPROP Algorithm

With the aim of addressing the optimum momentum problem, several adaptive learning algorithms were proposed in the literature. The RPROP (resilient backpropagation) algorithm described in [RB93] deals with this problem by updating the learning rate considering the behaviour of the error function. It directly adapts weights based on local gradient information with the following steps. First, for each weight its individual update value $\Delta_{ij}$ is introduced and set to an initial value $\Delta_{ij}^{0}$, which regulates the size of the first weight update. On each update step $\Delta_{ij}$ independently determines the size of the weight update.

The value of this parameter changes during the learning process in accordance with the rule:

$$\Delta_{ij}^{(t)} = \begin{cases} 
\eta^{+}\Delta_{ij}^{(t-1)}, & \text{if } \frac{\partial E}{\partial w_{ij}}^{(t-1)} \cdot \frac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\
\eta^{-}\Delta_{ij}^{(t-1)}, & \text{if } \frac{\partial E}{\partial w_{ij}}^{(t-1)} \cdot \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\
\Delta_{ij}^{(t-1)}, & \text{otherwise}
\end{cases}$$

(2.20)

where $0 < \eta^{-} < 1 < \eta^{+}$.

The expression stated above means that when the partial derivative $\frac{\partial E}{\partial w_{ij}}$ changes the sign, which indicates that the last update was bigger than needed and the algorithm jumped over the local minimum, the update value $\Delta_{ij}$ is reduced on the value of $\eta^{-}$. In case the sign remains the same, $\Delta_{ij}$ grows a little on the value of $\eta^{+}$ aimed to stimulate convergence.
After update values $\Delta_{ij}$ are computed, the final values for the weight updates are calculated considering the following rule:

$$
\Delta w^{(t)}_{ij} = \begin{cases} 
-\Delta^{(t)}_{ij}, & \text{if } \frac{\partial E^{(t)}}{\partial w_{ij}} \geq 0 \\
+\Delta^{(t)}_{ij}, & \text{if } \frac{\partial E^{(t)}}{\partial w_{ij}} \leq 0 \\
0, & \text{otherwise}
\end{cases}
$$

(2.21)

i.e., if the error grows and the derivative is positive, the weight update is reduced by its update value, on the contrary, if the error decreased (derivative is negative), the weight update is increased on its $\Delta_{ij}$.

The optimal choice of the increase parameter $\eta^+$ and the decrease parameter $\eta^-$ suggested by the authors are $\eta^+ = 1.2$ and $\eta^- = 0.5$.

2.3 Ensemble Learning

ANNs have been shown as the third best classifier after RANDOM FORESTS and SUPPORT VECTOR MACHINES when tested among 17 families of machine learning classifiers [FDCBA14]. Nevertheless, neural networks are known to be unstable learning algorithms [Die00]. An algorithm is termed unstable if its output is highly dependent on even small changes in the training data. Consequently, different subsets of training data may result in the models with highly varying generalization ability.

It was shown by Cunningham et al. [CCJ00] that the stability problem of neural networks can be solved by building an ensemble instead of using a single ANN. Generally, it is well-known that ensemble methods can improve the accuracy and error generalization [Sch90, HS90].

Ensemble Learning is the concept of combining and leveraging the power of multiple individual classifiers (students) to form a classifier capable to outperform any individual classifier alone and to provide accurate predictions. In an ensemble of classifiers students are trained on the same problem.

In order to train a model for an unknown target function $y(x)$, $N$ training examples $X_n = \{(x_1,y_1), \ldots, (x_N,y_N)\}$ are shown to a classifier. Let $\bar{x}_i$ be a vector $(x_{i,1}, x_{i,2}, \ldots, x_{i,m})$ where $x_{ij}$ is the $m$-th feature of example $x_i$. The class values $y$ belongs to the discrete set $\{1, \ldots, C\}$.

Then an ensemble $E$ of $K$ classifiers $h_t(x)$ on the target function $f(x)$ is built. Each student is a hypothesis on the target function $f(x)$ and is denoted here as $\{h_1, \ldots, h_K\}$.

2.3.1 The Ensemble Framework

An ordinary ensemble framework for a classification problem can be divided into several parts [Rok10]:
1. A labelled training set \( \{(x_1; y_1), ..., (x_m; y_m)\} \in \mathcal{X} \) with the class values \( y \) belonging to the discrete set \( \{1, ..., C\} \in \mathcal{Y} \).

2. Base Inducer \( I \) - the induction algorithm that forms a classifier \( h_k \) trained on a training set and representing the underlying relationship between input features and the target class.

3. Diversity Generator - the part responsible for generating diverse classifiers, erroneous on different training examples.

4. Combiner - this part is in charge of combining the predictions of the base classifiers in the ensemble and outputting the final classification for an unseen example.

The schematic representation of the ensemble framework is depicted in Figure 2.7.

**Figure 2.7:** Ensemble framework scheme. Training set is manipulated by the Dataset Manipulator, created Subset is then passed to the Inducer, which trains the model and hands it to the Classifier. Predictions for the unlabelled examples are then combined by the Classifiers Composer. The figure is based on [Rok10, p. 4].

For the independent ensemble framework, used in the scope of this work, each classifier is built independently one from the others, and the outputs of the classifiers are combined later. Another option is the dependent framework where prediction of a classifier is used to create and improve the output of the next one.
2.3.2 Majority Voting

Figure 2.8: Ensemble accuracy $P_{\text{ens}}$ with majority voting of $K$ classifiers on a binary classification problem. Accuracy increases with the number of classifiers if the probability of the individual classifier to output the correct label is $p > 0.5$, it decreases if the opposite is true, and it is constant on 0.5 if $p = 0.5$. The figure is taken from [Zho12, p. 73].

The most common combination method used to aggregate the predictions of the base classifiers for a classification problem is majority voting. The final decision on class $y \in \mathcal{Y}$ for each unlabelled example $x$ is performed by voting. Each classifier “votes” (predicts) the class for an unseen instance $x \in X$ and the class with the highest number of votes (the most frequent one) is then assigned to the instance $x$:

$$class(x) = \arg \max_{y_n \in \mathcal{Y}} \left( \sum_k g(h_k(x), y_n) \right)$$ \hspace{1cm} (2.22)

where $h_k(x)$ is the prediction of the $k$-th classifier and $g(h, y)$ is an indicator function defined as [Rok10]:

$$g(h, y) = \begin{cases} 1, & \text{if } y = h_k(x) \\ 0, & \text{if } y \neq h_k(x). \end{cases} \hspace{1cm} (2.23)$$

Let us consider $K$ classifiers for a binary output. The ensemble predicts correctly if at least $\lfloor K/2 + 1 \rfloor$ classifiers output the correct class. Considering that all classifiers are independent and choose the correct class with the probability $p$, the probability of the ensemble to predict correctly, which implies the probability of getting at
least \( \lfloor K/2 + 1 \rfloor \) correct outputs, can be calculated as follows using binomial distribution \([HS90]\):

\[
P_{\text{ens}}(K) = \sum_{i = \lfloor K/2 + 1 \rfloor}^{K} \binom{K}{i} p^i (1 - p)^{K - i}
\]

(2.24)

The influence of the variation of the number of classifiers \( K \) and the probability of the correct output \( P \) is demonstrated in Figure 2.8.

It was proven in \([LS97]\) that:

1. If \( p > 0.5 \), then \( P_{\text{ens}}(K) \) is monotonically increasing in \( K \) and \( P_{\text{ens}}(K) \to 1 \) as \( n \to \infty \).
2. If \( p < 0.5 \), then \( P_{\text{ens}}(K) \) is monotonically decreasing in \( K \) and \( P_{\text{ens}}(K) \to 0 \) as \( n \to \infty \).
3. If \( p = 0.5 \), then \( P_{\text{ens}}(K) = 0.5 \) for all \( K \).

However, it is important to notice, that the previous only holds with the assumption about mutual independence of the classifiers, which is usually not true, since trained on the same data, classifiers are highly correlated. Thus, it is extremely unlikely that increasing the number of classifiers \( K \) towards infinity will lead to the probability \( p \) of the ensemble to predict the correct label to be equal to 1.

### 2.3.3 Why Ensembles Work

Dietrich in \([Die00]\) described three fundamental reasons why an ensemble of classifiers in most cases outperforms a single predictor. The reasons are listed below:

1. **Statistical** (Figure 2.9a). A learning algorithm can be considered as inspecting the hypothesis space \( \mathcal{H} \) trying to find the best available hypothesis. If the given training data does not contain a sufficient amount of training examples, and the number of possible hypotheses is large compared to the available data, it leads to the statistical problem when the learning algorithm can find several hypotheses in \( \mathcal{H} \) yielding the same accuracy. However, with ensemble learning over this hypothesis space \( \mathcal{H} \) the result of the predictions by the several classifiers will be aggregated and averaged thereby minimizing the possibility of an unfortunate selection of a mistaken hypothesis.

In reference to the ensemble of neural networks, it was claimed by \([HS90]\) that when it comes to ANNs, networks will vary in the values of their weights \( w \) due to a number of reasons: different starting points (initial weights), different order of the training examples or different data set partitions used for training different networks. Various weights \( w \) lead to different generalization errors on
2.4 Constructing Ensembles

The main idea behind ensemble learning is, by creating a set of diverse base classifiers, to make them capable of correcting the errors of each other. Apparently, if classifiers generate the same result for a new data point, this mutual correction is not possible.
Figure 2.10: Example of the ensemble of three neural networks trained to identify inputs \((x, y)\) with the result of classification for each network (a-c) and the result of their voting (d) [HS90].

Conversely, if the classifiers make different errors on the data points, the aggregation of their predictions might reduce the overall error. There are several methods to construct diverse ensembles which will be discussed below.

### 2.4.1 Data Manipulation

This technique is suitable for unstable learning algorithms such as neural networks which are very sensitive to changes in the training data. The most common technique for data manipulation is BAGGING [Bre96a] or bootstrap aggregating. It is based on BOOTSTRAPPING [ET94] and the notion of aggregation.

This technique creates \(K\) training sets of the size \(|D|\) by sampling randomly with replacement in order to train \(K\) predictors with the generated subsets. Sampling with replacement refers to the possibility of an element to appear multiple times in the same sample. For instance, a fisherman catching fish, measuring it, returning it to the water and continuing catching is an example for sampling with replacement since he can measure the same fish several times.

By sampling with replacement, approximately 63.2\% of the records in a subset are expected to be unique, whereas the rest of them are repeated due to the replacement sampling strategy [Bre96a].

The bootstrapping step on the training set \(X = \{X_1, X_2, \ldots, X_N\}\) is realized as follows [SD02]: a bootstrap replicate \(X^k = \{X^k_1, X^k_2, \ldots, X^k_N\}\) is generated by taking replicate \(X^k\) of the training data set \(X\) \(K\) times. Then \(K\) classifiers \(h^k(x)\) are built on the \(X^k\) sample (see Algorithm 2.2).
Algorithm 2.2 Bagging [Pol12]

**Data:** Training data $X = \{ X_1, X_2, ..., X_N \}$ with $X_n = (x_n, y_n)$, where $y_n$ being a correct class.

**Integer K** - number of sampled versions of the training set.

**Double F** - fraction of available training patterns to create bootstrapped training sample.

**Return:** classified instances.

1: for each $k = 1, ..., K$ do
2: Take a bootstrapped replicate $X^k$ by randomly drawing $F$ of $X$
3: Train classifier $h_k$ with the sample $X^k$
4: Add $h_k$ to the ensemble $E$
5: end for
6: for each unlabelled instance $X$ do
7: Evaluate the ensemble $E = \{ h_1, ..., h_K \}$ on $X$
8: Let $v_{k,j} = \begin{cases} 1, & \text{if } h_k \text{ picks class } y_j \text{ by classifier } h_k \\ 0, & \text{otherwise} \end{cases}$
9: for each class $j = 1, ..., Y$ do
10: Obtain total vote received $V_j = \sum_{k=1}^{K} v_{k,j}$.
11: end for
12: Choose the class that receives the highest total vote as the final classification.
13: end for

On the aggregating step $K$ classifiers $h^k(x)$ are combined by majority voting (Equation 2.22).

**Bagging** amplifies overfitting of a base classifier, yet improves the stability and accuracy of the ensemble, reduces variance, and helps avoiding overfitting in the ensemble [Bre96a].

### 2.4.2 Feature Manipulation

Another approach for generating diverse classifiers is to manipulate data in the feature space. With the **RANDOM SUBSPACE METHOD** (RSM) proposed by Ho [Ho98], multiple classifiers are learned on the same data set but with different independently selected subsets of the available attributes. Since the training data used to train a classifier is incomplete, it becomes a “weaker” classifier and has its own bias.

Given training data $X = \{ X_1, X_2, ..., X_N \}$ where each data point $X_n$ is an $m$-dimensional vector $X_n = \{ x_{n1}, x_{n2}, ..., x_{nm} \}$, the RSM algorithm randomly chooses $s \leq m$ features from the data set $X$ thereby generating the random $s$-dimensional subspace from the original $m$-dimensional feature space. Consequently, the new generated training set $X^k = \{ X^k_1, X^k_2, ..., X^k_N \}$ comprises training examples with $s$-dimensions $X^k_n = \{ x^k_{n1}, x^k_{n2}, ..., x^k_{ns} \}$. Then, analogously to BAGGING, $K$ classifiers are trained on the $X^k$ samples (see Algorithm 2.3) and the final prediction is made by majority voting [SD02].
Algorithm 2.3 The Random Subspace Method [Ho98]

Data: Training data $X = \{X_1, X_2, \ldots, X_N\}$ with $X_n = (x_n, y_n)$, where $y_n$ being a correct class.

Integer $K$ - number of sampled versions of the training set.

Double $F$ - fraction of available features to create training sample.

Integer $m$ - number of features in $X$.

Return: classified instances.

1: for each $k = 1, \ldots, K$ do
2: Create a training set $X^k$ by choosing randomly $F \cdot m$ features from $m$ without replacement
3: Train classifier $h_k$ with the sample $X^k$
4: Add $h_k$ to the ensemble $\mathcal{E}$
5: end for
6: for each unlabelled instance $X$ do
7: Evaluate the ensemble $\mathcal{E} = \{h_1, \ldots, h_K\}$ on $X$
8: Let $v_{k,j} = \begin{cases} 1, & \text{if } h_k \text{ picks class } y_i \text{ by classifier } h_k \\ 0, & \text{otherwise} \end{cases}$
9: for each class $j = 1, \ldots, Y$ do
10: Obtain total vote received $V_j = \sum_{k=1}^{K} v_{k,j}$.
11: end for
12: Choose the class that receives the highest total vote as the final classification.
13: end for

2.4.3 Injection of Randomness

The last method for constructing an ensemble of classifiers described in this chapter is introducing to the learning algorithm the sources of randomness. Random assignment of the initial weights by the BACKPROPAGATION algorithm when training neural networks is one of the possible examples of this technique [Die00]. Random assignment of the initial weights basically means beginning the search through the hypothesis space $\mathcal{H}$ from various starting points and can serve as a simple tool for providing diversity among base classifiers, since it increases the probability of different trajectories followed by the learning algorithm in the search space. It is worth mentioning though, that in the case of neural networks, the search space and hypothesis space are not the same. The learning algorithm is searching in the possible weight values space whereas the weight values define the behaviour of the network, which is a part of the hypothesis space.

It was shown by Sharkey et al. [SNS95] that initial random weights affect the number of iterations a network needs to converge and defines if the algorithm converges at all. However initialising random weights is not enough to exhibit sufficient diversity [BWHY05] and neural networks trained this way generate similar patterns of generalization. Partridge and Yates in [YP96] also pointed out that compared to varying number of hidden neurons, network type and training data, initialisation of random weights is the least efficient way to create a set of diverse neural networks.

Another popular approach aimed to increase randomization in an ensemble of ANNs is building neural networks with randomly chosen activation functions, num-
ber of hidden neurons, and learning rate parameters [MNM08]. Sharkey et al. in [SNS95] also suggested that inducing diversity into an ensemble of neural networks can be achieved by affecting one of the four factors: the initial weights, the training data, the architecture of the network and the training algorithm.

2.5 Diversity

In his book The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations? [Sur05] Surowiecki claims that under certain circumstances the aggregated opinion of a group of people provides excellent results and is superior to any single opinion even of an expert, and even considering that a member of this group is not especially well-informed or rational. However, in order to be “wise,” a crowd should satisfy the following conditions:

- Diversity of opinions: each member of the crowd has some private information or has a bias.
- Independence: an opinion of a member is formed independently from the other members of the crowd.
- Decentralization: the crowd members are able to determine their opinion based on their specialised knowledge.
- Aggregation: there is a mechanism which assembles and combines the private judgements of the crowd members into a collective decision.

Those principles are highly applicable to supervised learning in an ensemble: machine learning models should be diverse; should be based on the different sources of information and have local knowledge (diverse subsets of the training data); they should be able to produce hypotheses based on their knowledge (previously seen examples); their predictions should be aggregated by some mechanism (for instance, majority voting) in order to output the final prediction.

Diversity in a classification problem stimulates different classifiers to develop various decision boundaries and is obtained when the classifiers are erroneous on different previously unseen instances.

In the literature there is a proven connection between good ensemble performance and diversity among base classifiers [KW03]. In order to form a good ensemble their members should disagree in their predictions. Krogh et al. [KV95] showed that increasing ambiguity enhances the overall generalization. Dietrich showed in [Die00] that a larger diversity property leads to more accurate ensembles. Hansen and Salamon [HS90] contributed, that in order for the error rate to go to zero as the number
of the base classifiers goes to infinity, the base classifiers should be differently erroneous and mutually independent. Breiman [Bre01] also claimed that lower correlation between classifiers, i.e., increased diversity, leads to a better performance of the RANDOM FORESTS algorithm, which will be discussed later in this chapter.

However, increasing diversity is not infinitely beneficial for an ensemble. At some point increasing the degree of diversity increases the error [MNM08, CC00]. Therefore it is important to not only increase diversity but to find out the point when it will yield an accuracy reduction.

2.5.1 Diversity and Overfitting

Some classification algorithms are unstable, i.e., the trained model will have high variance [Cun00, Bre96b]. Variance is defined by Kohavi et al. in [KW+96] as sensitivity of the learning algorithm to changes in the training set which are independent of the underlying target. Breiman showed in [Bre96a] that neural networks are unstable predictors. Minor fluctuations in the training data may result in significantly different models, which means their different behaviour on the test data.

A possible reason for such different models might be that algorithm is hindered by various local minima of the error surface [Bre96a], the similar explanation was suggested by Cunningham et al. [CCJ00] and Perron and Cooper [PC91].

Hansen and Salmon [HS90] also proposed that since selection of the weights \(w\) is an optimisation problem with many local minima, therefore the “optimal” parameters vary considerably for different algorithm runs. The same holds for all global optimisation methods as a result of randomly chosen initial weights \(w_0\) and the order of the training examples.

Nevertheless, within the concept of ensemble learning instability of the neural networks for ensemble methods is not a disadvantage. Firstly, because sub-optimal performance of neural networks due to existence of many sub-optimal local minima can be overcome with their efficient utilization with averaging methods (ensembles) [PC91]. Secondly, variance is an important condition for constructing a good ensemble. Namely, as discussed above, variations in prediction of the individual base classifier make the ensemble diverse and increase its ambiguity, what is crucial for ensemble learning.

One of the ways to construct an ensemble of diverse classifiers is to train a set of predictors which overfit.

2.5.2 Overfitting

The 14th century English philosopher William of Ocham stated that “plurality must never be posited without necessity.” Named after him, Ocham’s Razor principle is the concept of parsimony, commonly used in the scientific world as a guide in developing
2.5. Diversity

Figure 2.11: Example of overfitting. The green curve represents the true functional relationship, while the blue curve shows the learned function, which overfits the training data (red dots). The figure is taken from [Pog12].

Theoretical models. It calls for choosing the simplest hypothesis containing all necessary over a complex one. Applied to Machine Learning, this principle means that among many models it is beneficial to select the one which is simplest, but complex enough to describe the underlying relationship of the problem.

As an example, if the target function can be explained by the linear function, which is described by two parameters, the intercept and slope values, then trying to explain the relationship by quadratic function interferes parsimony and Ocham’s Razor principle.

Considering Occam’s Razor, overfitting is the usage of a model which conflicts with this principle, i.e., requires many different parameters and uses more sophisticated approach than it is necessary.

It is possible to differentiate two types of overfitting. The first occurs when model is more flexible than it is required. For instance, using neural networks, which are very flexible when underlying relationship can be described with linear regression will result in unnecessary complexity of the model and consequently poor performance on unseen data [Haw04].

The second type of overfitting appears when the model comprises unrelated compartments such as using polynomial of higher degree than it is needed (Figure 2.11).

Overfitting can be easily recognised by poor performance on the test data. An example is represented in Figure 2.12 where model predicts well on the training data but do not generalize well on unseen (test) data.

Apart from stimulating poor prediction, overfitting is very undesirable for several other reasons such as wasted resources due to unnecessary model complexity and possible difficulties with reproducibility of scientific results gained with overfitting.
2.5.3 Measuring Diversity

Some empirical results have shown that there is a connection between diversity among classifiers and accuracy of the ensemble [Die00, KW03]. Although several measures of diversity have been proposed in the literature, the notion of diversity has not been strictly and explicitly defined in machine learning, pattern recognition and computer science. Furthermore, none of the proposed measures has been confirmed to be better than the others [Kun03].

Krogh and Vedelsby [KV95] have proven the relationship between ambiguity (diversity) and the ensemble generalization error with ambiguity decomposition:

$$E = \bar{E} - \bar{A}$$

(2.25)

where $E$ is the overall ensemble error over input distribution, $\bar{E}$ is the error of the individual classifiers averaged over test instances, $\bar{A}$ is aggregation of individual ambiguities. Individual ambiguities represent the variances of the output of individual predictors over the ensemble. More detailed, Equation 2.25 can be written as
(f_{\text{ens}}(x) - y)^2 = \frac{1}{K} \sum_{k=1}^{K} (f_k(x) - y)^2 - \frac{1}{K} \sum_{k=1}^{K} (f_k(x) - f_{\text{ens}}(x))^2 \quad (2.26)

where $y$ is the target output of the test example $x$, $K$ is the number of predictors and $f_{\text{ens}}$ is the combination of individual predictors:

$$f_{\text{ens}}(x) = \sum_{k=1}^{K} f_k(x)$$ \quad (2.27)

The importance of Equation 2.25 is that it demonstrates that the error of the ensemble will be less than or equal to the average error over individual predictors.

The decomposition consist of two terms - the average error term $\bar{E}$ and the average ambiguity $\bar{A}$. Since the the second term is always positive, it is guaranteed that ensemble error will be lower than average individual error. The larger the value of the ambiguity term, the larger the decrease of the overall ensemble error. Nevertheless, as the variance of the predictors increases, the value of $\bar{E}$ also grows. This demonstrates that the only diversity increase is not sufficient for improvement of the ensemble performance, and the equilibrium between diversity and individual accuracy should be considered [BWHY05].

Ambiguity decomposition demonstrated in Equation 2.25 is only true for a regression ensemble. There is no ambiguity decomposition for classifier ensembles and an appropriate measure for them is still an open question. Also, some experimental results on various diversity measures were confusing and demonstrated their inefficiency [TSY06].

Kuncheva et al. [KW03] proposed ten statistics to measure diversity in an ensemble of classifiers: four averaged pairwise measures (the $Q$-statistics, the correlation, the disagreement and the double fault) and six non-pairwise measures (the entropy of the votes, the difficulty index, the Kohavi-Wolpert variance, the interrater agreement, the generalized diversity, and the coincident failure diversity), some of them will be described below.

Let $\mathcal{E} = \{h_1, h_2, ..., h_K\}$ be a set of classifiers, $\mathcal{Y} = \{Y_1, Y_2, ..., Y_c\}$ be a set of class labels, $x \in \mathcal{R}^n$ be an $n$-dimensional vector with labels in $\mathcal{Y}$, $K$ be the number of classifiers in the ensemble $\mathcal{E}$ and $Z = \{z_1, ..., z_N\}$ be a labelled data set with $N$ instances.

Here we consider the output of a binary classifier, i.e., the output indicates correctness of the predicted class label. Therefore the output $h_k(x)$ equals to 1 if class is estimated correctly by $h_k$ and 0 if not. This approach is referred in the literature as oracle output. In that case output of classifier $h_k$ is $N$-dimensional vector with binary values.

### 2.5.4 $Q$-statistics

The $Q$-statistics is a representative of the group of pairwise diversity measures.
Each output of the classifier $h_k$ can be represented as an $N$-dimensional binary vector $V_k = [v_{1,k}, ..., v_{N,k}]^T$ indicating whether $h_k$ predicts training instance $z_j$ correctly ($v_{jk} = 1$) or not ($v_{jk} = 0$), $k = 1, ..., K$.

Yule’s Q-statistics [Yul00] for two classifiers $h_i$ and $h_k$ is the following

$$Q_{i,k} = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}}$$

(2.28)

where $N^{ab}$ is the amount of elements $z_j$ with $y_{i,j} = a$ and $y_{j,k} = b$ in accordance with Table 2.1.

<table>
<thead>
<tr>
<th></th>
<th>$h_k$ correct (1)</th>
<th>$h_k$ incorrect (0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_i$ correct (1)</td>
<td>$N^{11}$</td>
<td>$N^{10}$</td>
</tr>
<tr>
<td>$h_i$ incorrect (0)</td>
<td>$N^{01}$</td>
<td>$N^{00}$</td>
</tr>
</tbody>
</table>

Table 2.1: Relationship between two classifiers.

If two classifiers are independent, the expected value of $Q$ is 0, being in the range $[1, -1]$ otherwise. Classifiers predicting the same instances correctly will have positive values of $Q$, those classifiers which are erroneous on the same instances will have negative $Q$ value.

The average $Q$ statistics for an ensemble $\mathcal{E}$ of $K$ classifiers is

$$\bar{Q} = \frac{2}{K(K - 1)} \sum_{k=1}^{K-1} \sum_{i=k+1}^{K} Q_{k,i}$$

(2.29)

### 2.5.5 Entropy Measure

Unlike $Q$-statistics, the entropy measure belongs to the non-pairwise diversity measures. Let the number of base classifiers from $\mathcal{E}$ that estimates correct class label of $z_j$ be

$$l(z_j) = \sum_{k=1}^{K} y_{jk}.$$  

(2.30)

The greatest diversity for an ensemble of classifiers for a certain $h_k \in \mathcal{E}$ can be achieved when a half of all classifiers $K/2$ outputs one value (either 0 or 1) while the second half $K/2$ outputs another possible value. There is no diversity in the ensemble in the case when all the classifiers output correct (1) or incorrect (0) class [KW03].

Then the entropy measure of diversity can be described as
2.6. Random Forests

\[ E = \frac{1}{N} \sum_{j=1}^{N} \frac{1}{K - \lceil K/2 \rceil} \min\{l(z_j), K - l(z_j)\} \]  \hspace{1cm} (2.31)

### 2.5.6 Kohavi-Wolpert Variance

Kohavi and Wolpert in [KW+96] proposed the decomposition formula for the classification error. One of the four components is variance which shows how predicted class \( y \) varies for a data point \( x \) for different training data sets for a particular classifier:

\[ \text{variance}_x = \frac{1}{2} \left( 1 - \sum_{i=1}^{C} P(y = y_i | x)^2 \right) \]  \hspace{1cm} (2.32)

where \( P(y = y_i | x) \) is calculated by taking average over the data sets.

The authors in [KW03] examined variability of the classes predicted by classifiers in \( \mathcal{E} \) for \( x \) on given training set, considering only two (correct or incorrect) possible outputs; \( P(y = 1|x) \) and \( P(y = 0|x) \) were derived as an average over predictions of different classifiers in \( \mathcal{E} \):

\[ \bar{P}(y = 1|x) = \frac{l(x)}{K} \quad \text{and} \quad \bar{P}(y = 0|x) = \frac{K - l(x)}{K} \]  \hspace{1cm} (2.33)

Replacing 2.32 with 2.33 gives

\[ \text{variance}_x = \frac{1}{2} (\bar{P}(y = 1|x)^2 - (\bar{P}(y = 1|x))^2) \]  \hspace{1cm} (2.34)

Then the Kohavi-Wolpert variance measure can be calculated as average over all classifiers in \( \mathcal{E} \):

\[ KW = \frac{1}{NK^2} \sum_{n=1}^{N} l(h_n)(K - l(h_n)) \]  \hspace{1cm} (2.35)

During our experiments the three measures described above exhibited high mutual correlation and therefore for the representation of the experimental results only the entropy measure \( E \) will be used.

### 2.6 Random Forests

In order to create an ensemble of diverse artificial neural networks we used the sampling strategy implemented by RANDOM FORESTS, therefore we will consider the main characteristics of the algorithm below.
The **Random Forests** algorithm is an example of the concept of ensemble learning. The main idea of the Random Forests is to grow an ensemble of Decision Trees and consider the majority class for the final decision by voting.

Random Forests has been shown to be one of the most robust and effective machine learning algorithms [FDCBA14]. This stems from the elimination of the very high variance intrinsic to Decision Trees and is achieved by averaging predictions of several trees trained on independently sampled, randomized subsets of data. Randomization increases bias which is initially low for decision trees while the variance among the ensemble members is reduced by averaging. The algorithm is able to deal with a large amount of data due to its sampling strategy.

The two main techniques exploited by the algorithm have been already mentioned above: Bagging (see Section 2.4.1) and the Random Subspace Method (see Section 2.4.2).

The Random Forests algorithm creates an ensemble of a large number of Decision Trees (Algorithm 2.4). The input parameter $f$ represents the number of features used to find the best split for each node. This number should be considerably smaller than the total number of the available features in the original data set.

---

**Algorithm 2.4 Random Forests** [Rok10]

Data: Training data $X = \{X_1, X_2, \ldots, X_N\}$ with $X_n = (x_n, y_n)$, where $y_n$ being a correct class.

DT Decision Tree classifier

Integer $K$ - number of classifiers in the ensemble.

Integer $N$ - number of instances for each training subsample.

Integer $f$ - number of features used in each node.

Return: classified instances.

1: for each $k = 1, \ldots, K$ do
2: Create a training set $X^k$ by sampling $N'$ instances from $X$ with replacement
3: Build the classifier $h_k$ using DT and $f$ features for each split on the sample $X^k$
4: Add $h_k$ to the ensemble $E$
5: end for
6: for each unlabelled instance $x$ do
7: Evaluate the ensemble $E = \{h_1, \ldots, h_k\}$ on $x$
8: Let $v_{k,j} = \begin{cases} 1, & \text{if } h_k \text{ pics class } y_i \text{ by classifier } h_k \\ 0, & \text{otherwise} \end{cases}$
9: for each class $j = 1, \ldots, Y$ do
10: Obtain total vote received $V_j = \sum_{k=1}^{K} v_{k,j}$.
11: end for
12: Choose the class that receives the highest total vote as the final classification.
13: end for

The algorithm uses Random Subspace Method for each split, i.e., at each node $f$ features are randomly chosen from $m$ available in the training data $X$, and the best split is chosen based on this feature subspace [Bre01]. Additionally, unlike classification and regression trees (CART) [BFSO84], trees, grown by the algorithm, are not pruned [Bre01], so they have their maximal possible depth and therefore prone to
overfit the data.

2.7 Summary

Neural networks, whose development started more than half of a century ago, have grown from a single neuron in 1943 to a excessively complex networks today. A MULTILAYER PERCEPTRON with one hidden layer of neurons is one of the most popular representatives of neural networks which is usually trained using the BACKPROPAGATION algorithm. The most common network topology with one hidden layer has proven to be successful in approximating the decision boundaries for the most of the real-world problems.

Ensemble learning has become a very popular concept in the machine learning community in the last several decades due to ability of a set of models to be a better approximator than that of a single model and for solving a problem of model instability. An ensemble ability to generalise well is shown to correlate with the amount of disagreement among its members. The classifiers should be diverse, i.e., develop different decision boundaries and be erroneous on different training examples. Overfitting might be a way of creating of a set of diverse predictors, it maintains variety among ensemble members and provides disagreement.

Among various techniques used to induce diversity in an ensemble, the most common and effective ones are data manipulation (BAGGING) and feature manipulation (the RANDOM SUBSPACE METHOD). Both techniques are used by the RANDOM FORESTS algorithm. A similar sampling approach is used when creating a randomized ensemble of artificial neural networks.
Chapter 3

Randomized Ensemble of Artificial Neural Networks

As described in Chapter 2, there is a positive correlation between diversity in an ensemble and its generalisation ability.

The range of approaches aimed at creating a set of diverse classifiers can be divided into two main groups: data manipulation (see Sections 2.4.1 and 2.4.2) and model manipulation (see Section 2.4.3). The former refers to the operations on the data set, more precisely, to applying various sampling strategies to the data, and the latter denotes the manipulation with the trained model, in our case with neural networks.

We use both groups of the approaches in order to create an ensemble of diverse neural networks by conducting a set of experiments intended to prove or disprove a series of hypotheses which will be discussed later in this chapter.

3.1 Randomized Ensemble Framework

In order to apply data manipulation techniques and to provide each network with a different training set, we implemented the RANDOMIZED ENSEMBLE framework (REF). It generates a predefined amount of subsets, one for each ensemble member, with a predefined proportion of features and a predefined proportion of the training instances.

REF uses two techniques to manipulate training data: BAGGING (See section 2.4.1) and the RANDOM SUBSPACE METHOD (RSM) (See Section 2.4.2). The principle of REF operation is inspired by the RANDOM FORESTS algorithm (see Section 2.6), which trains each Decision Tree on a training subset generated using BAGGING. Unlike RANDOM FORESTS, which applies RSM for each tree split, REF applies RSM and BAGGING simultaneously, generating a training sample to learn a model.

A schematic explanation of the REF work is illustrated in Figure 3.1 where the front square symbolizes the original training set, on which the ensemble is trained.
3.1. RANDOMIZED ENSEMBLE FRAMEWORK

Figure 3.1: Space of possible training sets for an ensemble. Each rectangle represents the initial training set, dashed horizontal and vertical lines denote respectively training patterns and features taken for a subset, dotted bold horizontal lines stand for the training patterns taken more than once.

Algorithm 3.1 Randomized Ensemble Framework

Data: Training data $\mathcal{X} = \{X_1, X_2, \ldots, X_N\}$ with $X_n = (x_n, y_n)$, where $y_n$ being a correct class.

Integer $K$ - number of classifiers in the ensemble.

Integer $N$ - number of training instances in $\mathcal{X}$.

Integer $m$ - number of features in $\mathcal{X}$.

Integer $D_p$ - proportion of train instances for each training subsample.

Integer $F_p$ - proportion of features used for each training subset.

Boolean $s_r$ indicates weather data should be sampled with replacement or not

Return: set of training subsets $S$.

1: for each $k = 1, \ldots, K$ do
2: if $s_r$ is true then
3: Create a training set $X^k$ by sampling $D_p \cdot N$ instances from $X$ with replacement using $F_p$ randomly chosen features from $m$
4: else
5: Create a training set $X^k$ by sampling $D_p \cdot N$ instances from $X$ without replacement using $F_p$ randomly chosen features form $m$
6: end if
7: Add $X^k$ to the set of subsets $S$
8: end for

The RANDOMIZED ENSEMBLE Framework algorithm (Algorithm 3.1) generates a set of overlapping subsets comprising different parts of the whole data set: it partitions randomly both training patterns (BAGGING) and features (RSM) creating a training subspace with randomly sampled $D_p \cdot N$ training instances from $\mathcal{X}$ (dashed horizontal bars in Figure 3.1) using $F_p \cdot m$ features (dashed vertical bars). The set of all squares represents the space of possible training sets generated by the sampling
algorithm to train the ensemble. The training data is sampled with replacement, i.e.,
around 33% of the training patterns will appear in the subset more than once (are
depicted with the bold dotted horizontal bars in Figure 3.1).

As seen in Chapter 2 more randomness in an ensemble usually means better en-
semble generalisation ability. The main idea behind the set of the approaches de-
scribed below is to create the additional sources of randomness by varying different
parameters regulating both model (neural network) and the data subset, the model is
trained on.

In addition to the experiments with varying a tested parameter we look at the com-
bination of the tested parameters with the sampling provided by REF and compare
the results of both sampling and nosampling approaches for each of the tested param-
eters. Later we refer to the experiments conducted on the data subsets generated by
REF as sampling, and those conducted on the original data without REF are referred
as nosampling.

**Hypothesis 1.** Randomizing the data and model parameters along with applying REF sam-
pling on the data increases a. diversity and b. accuracy in the ensemble.

Aside from the general hypothesis above, we included this assumption to our hy-
potheses where applicable, testing this assumption separately for each examined pa-
rameter.

### 3.1.1 Data Manipulation and Sampling Strategy Influence

The proportion of training patterns used by REF for each subset is referred as $D_p$. The
proportion of features employed for each sample generation is referred referred as $F_p$.

In the data sets which are free from the class imbalance problem (class labels are
more or less evenly distributed) the influence of the randomly chosen proportion of
training instances $D_p$ should not be that significant when compared to the influence
of $F_p$, regulating the size of the randomly chosen part of the feature space, which
is introduced to the learning algorithm. In the former case, some representatives of
each class are most likely present in a training subset, while in the latter case no single
representative of some features will be shown to the network.

**Hypothesis 2.** The influence on a. diversity and b. accuracy of the proportion of features
$F_p$ is more significant compared to the influence of the proportion of training patterns $D_p$
introduced to each model in an ensemble of neural networks.

Generating training subsets using REF should help to increase diversity among
the members of an ensemble. Furthermore, combining the data manipulation ap-
proach with another source of randomness by randomly generating values for $D_p$
and $F_p$ in a predefined range may be beneficial for the ensemble in the sense of diver-
sity stimulating.
Hypothesis 3. Randomizing the training patterns proportion $D_p$ parameter of the Randomized Ensemble Framework a. increases diversity among the members of an ensemble of neural networks and b. improves the ensemble performance; feature sampling (RSM), applied additionally, improves both c. diversity and d. accuracy.

Hypothesis 4. Randomizing the feature proportion $F_p$ parameter of the Randomized Ensemble Framework a. increases diversity among the members of an ensemble of neural networks and b. improves the ensemble performance; BAGGING, applied additionally, improves both c. diversity and d. accuracy.

3.2 Model Manipulation

Unlike data manipulation approaches, model manipulation techniques operate on different parameters regulating a model’s behaviour. One of the methods is to randomize the parameters.

In the case of neural networks there is a set of the parameters varying of which can be advantageous for injecting randomness into an ensemble.

In our experiments on model manipulation we divide the parameters into two groups: topology regulating parameters and learning process regulating parameters.

3.2.1 Topology Regulating Parameters

One of the popular approaches used to bring diversity to an ensemble of neural networks is generating neural networks with different topologies. We addressed this approach by randomizing two parameters defining the architecture of the network: the number of hidden layers $H_{lay}$ and the number of neurons $H_{neu}$ on hidden layer.

We hypothesize that assigning random values within a certain range to the number of hidden layers $H_{lay}$ and the number of hidden neurons $H_{neu}$ for each neural network in an ensemble allows to generate the ensemble with diverse members.

Hypothesis 5. By randomizing the number of hidden layers $H_{lay}$ it is possible to a. stimulate diversity among the members of an ensemble of neural networks and b. by this to improve the ensemble performance; REF, applied additionally, improves both c. diversity and d. accuracy.

Hypothesis 6. By randomizing the number of hidden neurons $H_{neu}$ it is possible to a. stimulate diversity among the members of an ensemble of neural networks and b. by this to improve the ensemble performance; REF, applied additionally, improves both c. diversity and d. accuracy.

As seen in Section 2.6 the RANDOM FORESTS algorithm uses unpruned DECISION TREES to create an ensemble. Unpruned trees have a tendency to overfit the training data. Analogously we would like the networks in an ensemble to be prone to overfitting. Overfitting might be beneficial in an ensemble since overfitting models trained...
on different data subsets are proved to be more diverse (See Section 2.5.2). At the same time the final model is safe from the overfitting problem and stable due to averaging. Redundant neurons on the hidden layer should lead the network to overfit the training data and be very sensitive to the noise.

Additionally, the training of neural networks can lead to more accurate predictions in larger networks, and therefore, it is might be possible for such networks to perform with better generalization ability [LGT98].

As discussed in Section 2.1.2, one hidden layer is enough for a close approximation of any decision boundaries. However, the networks with multiple hidden layers are able to build a complex hierarchy of concepts, which when applied to complex data, may be beneficial.

To summarize, it seems to be promising to explore the influence of large values of both number of hidden neurons $H_{neu}$ and number of hidden layers $H_{lay}$ on diversity and accuracy in an ensemble applied together with REF.

**Hypothesis 7.** Increasing the amount of hidden neurons in an ensemble of neural networks positively influences a. the ensemble diversity and b. the ensemble performance.

**Hypothesis 8.** Increasing the amount of hidden layers in an ensemble of neural networks positively influences a. the ensemble diversity and b. the ensemble performance.

### 3.2.2 Learning Process Regulating Parameters

Two other network parameters, favourable as a potential source of randomness, influence learning process.

The first one is learning rate $\epsilon$. It defines how fast the algorithm converges when finding the local minima of the error function (see Section 2.2.1). Working with the implementation of MLP with the RPROP algorithm, which performs a local adaptation of the weight-updates according to the behaviour of the error function (see Section 2.2.2), we vary the learning rate parameter $\epsilon$ while keeping the increase parameter $\eta^+$ and the decrease parameter $\eta^-$ as it was suggested by the authors of the algorithm ($\eta^+ = 1.2$ and $\eta^- = 0.5$).

**Hypothesis 9.** By randomizing learning rate $\epsilon$ while fixing the increase parameter $\eta^+ = 1.2$ and the decrease parameter $\eta^- = 0.5$ for each member of the ensemble it is possible to a. promote diversity among the members of an ensemble of neural networks and b. to improve the ensemble generalisation ability; REF, applied additionally, improves both c. diversity and d. accuracy.

The second parameter which regulates learning process is the number of epochs $E_p$. This parameter defines the number of times the training data is shown to the network. For the feed-forward networks like MLP one epoch corresponds to one forward- and one backward pass of all the instances in the training subset.
Hypothesis 10. It is possible to promote diversity among the members of an ensemble of neural networks by randomizing the number of epochs $E_p$ parameter for each member of the ensemble; REF, applied additionally, improves both c. diversity and d. accuracy.

Another parameter in our experiments, which influences behaviour of the network, indicates whether initial weights are assigned randomly for each network or not. The assignment of random weights leads the learning process to start from different locations in the search space. It should encourage the learning algorithm to move along various trajectories for different networks. And although this approach was named as the least efficient for stimulating disagreement among the members of an ensemble of neural networks (see Section 2.4.3), we use it in our experiments as an additional source of randomness.

This parameter is denoted as $w_{seed}$ with possible values 0 (weights are randomly assigned without a random seed) and 1 (weights are set randomly with a constant random seed). In the experiments on the data and model parameters varying, where we needed to exclude all the other sources of randomness, so that it is possible to test a particular parameter and clearly see if it causes any changes in the ensemble behaviour, we set $w_{seed}$ to 1.

3.3 Random Connections

This approach differs from the others and can be seen as an unconventional one. We suggested that deletion of some amount of randomly chosen connections between neurons, made during the network training, can be beneficial for boosting diversity in an ensemble of neural networks. Theoretically, this approach might have an interesting effect on the trajectories in the search space.

Deletion of a connection between two neurons implies that the weight of this connection is equal to 0 and it is never considered during the weight update phase. The fraction of the network connections to be deleted is regulated by the parameter $t \in \{0.1; 0.8\}$.

Hypothesis 11. By randomizing the proportion of deleted connections, $t$, for each member of the ensemble it is possible a. to increase diversity among the members of an ensemble of neural networks and b. to increase the ensemble accuracy; REF, applied additionally, improves both c. diversity and d. accuracy.

The random connections deletion algorithm is implemented as follows: the weights are first initialised in accordance with the parameter $w_{seed}$. Then the fraction $t$ of the connections’ weights is set to 0. During the training phase, for both forward and backward propagations, when gradients are computed, the connections which have been randomly chosen to be “removed” are not taken into account when calculating the weighted sum of a neuron inputs.
Analogously, for prediction of unlabelled training instances by the trained model, the weights of the “removed” connections are set to 0, so the connections behave as if they do not exist. The approach is illustrated in Figure 3.2 where the dotted lines depict the “removed” connections.

### 3.4 More Randomness

In the literature more randomness in an ensemble means better ensemble generalisation ability. Following this logic, concurrent randomizing of several data sampling and model regulating parameters might be even more advantageous regarding disagreement in the ensemble.

**Hypothesis 12.** Simultaneous randomizing the various ANN and REF parameters increases a. diversity in the ensemble of neural networks and b. improves the ensemble performance.

### 3.5 Diversity and Accuracy

The main purpose of the approaches described above is to improve ensembles generalisation ability. The main objective of all the experiments described in the next chapter is to find out how increasing diversity correlates with the ensemble performance.
As mentioned before, an uncontrolled increase of diversity may not always be favourable. In order to increase accuracy of an ensemble of binary classifiers, the probability of a single model $p$ to predict a correct class should be $> 0.5$ (see Section 2.3.2). Clearly, it is not true for too erroneous networks.

Most of the data sets, we are dealing with in our experiments, are not binary. Therefore, it is not a trivial task to estimate the threshold of reliability, which indicates a model’s utility for the ensemble. Nevertheless, it is obvious that an ensemble of too erroneous classifiers cannot yield a good result.

**Hypothesis 13.** Increasing diversity among the members of an ensemble of neural networks improves the ensemble accuracy.

### 3.6 Summary

The approaches aimed at generating a set of diverse classifiers can be divided into the data manipulation and model manipulation approaches.

The **RANDOMIZED ENSEMBLE** Framework represents the former approaches and implements a sampling strategy similar to that implemented by **RANDOM FORESTS**: it simultaneously and randomly samples training instances and manipulates data in the feature space.

The model manipulation approaches can be addressed by randomizing various neural network parameters, which can be divided into the topology regulating and learning process regulating parameters. The former are represented by the number of hidden neurons $H_{\text{neu}}$ and the number of hidden layers $H_{\text{lay}}$; the latter include the learning rate $L_r$ and the number of epochs $E_p$.

A combination of both data manipulation and model manipulation techniques as well as introducing more randomness while simultaneously randomizing several parameters can be considered promising in the sense of generating an ensemble of more diverse neural networks, which makes the ensemble more accurate.
Chapter 4

Experimental Results

In this chapter we describe the methods employed for conducting our experiments and the benchmark parameters we found in order to compare the experimental results. We also specify how we chose the parameters for the different experiments, and discuss the obtained results considering the hypotheses we proposed in Chapter 3.

4.1 Experimental Setup

In order to evaluate the influence of randomization of the parameters of an ensemble of neural networks on the ensemble diversity and accuracy we conducted a set of experiments using KNIME [BCD+07] on a server under Linux.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Num. of features</th>
<th>Num. of instances</th>
<th>Num. of classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>chess</td>
<td>37</td>
<td>3197</td>
<td>2</td>
</tr>
<tr>
<td>faults</td>
<td>61</td>
<td>208</td>
<td>2</td>
</tr>
<tr>
<td>page blocks</td>
<td>11</td>
<td>5473</td>
<td>5</td>
</tr>
<tr>
<td>seeds</td>
<td>7</td>
<td>210</td>
<td>3</td>
</tr>
<tr>
<td>sonar</td>
<td>60</td>
<td>208</td>
<td>2</td>
</tr>
<tr>
<td>spambase</td>
<td>57</td>
<td>4601</td>
<td>2</td>
</tr>
<tr>
<td>statlog</td>
<td>36</td>
<td>4435</td>
<td>5</td>
</tr>
<tr>
<td>wave form</td>
<td>57</td>
<td>3059</td>
<td>2</td>
</tr>
<tr>
<td>wine quality (red)</td>
<td>12</td>
<td>1599</td>
<td>5</td>
</tr>
<tr>
<td>wine quality (white)</td>
<td>12</td>
<td>4898</td>
<td>5</td>
</tr>
<tr>
<td>yeast</td>
<td>22</td>
<td>5000</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 4.1: Collection of 11 data sets from the UCI database used for experiments and their main characteristics.
The experiments were conducted on 11 datasets from the UCI Machine Learning Repository [Lic13] and are listed in Table 4.1.

For creating an ensemble of neural networks we employed the KNIME implementation of the RPROP algorithm for multilayer feedforward networks described in Section 2.2.2.

For the sake of more accurate estimations of the prediction performance and delivering the statistically reliable results, the experiments were performed with 5-fold cross-validation. We used the results of the cross-validation to compute both diversity and accuracy.

### 4.2 Benchmark Accuracy

In order to evaluate and compare the outcomes of our experiments we calculated the benchmark accuracy taking the best result of the performance of a single network with different values for the number of hidden neurons $H_{\text{neu}}$ for each tested data set. The results are shown in Table 4.2 under the “Single network” header.

<table>
<thead>
<tr>
<th>Data set</th>
<th>$D_p$</th>
<th>$F_p$</th>
<th>$H_{\text{neu}}$</th>
<th>Accuracy, %</th>
<th>$H_{\text{neu}}$</th>
<th>Accuracy, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>chess</td>
<td>0.8</td>
<td>0.8</td>
<td>34</td>
<td>99.41 (0.49)</td>
<td>15</td>
<td>99.09 (0.45)</td>
</tr>
<tr>
<td>faults</td>
<td>0.4</td>
<td>0.8</td>
<td>7</td>
<td>100.00 (0.00)</td>
<td>10</td>
<td>100.00 (0.00)</td>
</tr>
<tr>
<td>page blocks</td>
<td>0.8</td>
<td>0.8</td>
<td>11</td>
<td>97.44 (0.34)</td>
<td>6</td>
<td>97.30 (0.54)</td>
</tr>
<tr>
<td>seeds</td>
<td>0.9</td>
<td>0.9</td>
<td>8</td>
<td>95.71 (2.61)</td>
<td>8</td>
<td>95.71 (2.61)</td>
</tr>
<tr>
<td>sonar</td>
<td>0.8</td>
<td>0.4</td>
<td>7</td>
<td>88.47 (5.95)</td>
<td>32</td>
<td>83.66 (5.69)</td>
</tr>
<tr>
<td>spambase</td>
<td>0.9</td>
<td>0.6</td>
<td>56</td>
<td>95.59 (0.75)</td>
<td>39</td>
<td>94.50 (1.10)</td>
</tr>
<tr>
<td>statlog</td>
<td>0.7</td>
<td>0.9</td>
<td>36</td>
<td>86.90 (1.44)</td>
<td>29</td>
<td>86.88 (0.95)</td>
</tr>
<tr>
<td>waveform</td>
<td>0.8</td>
<td>0.7</td>
<td>11</td>
<td>87.40 (1.49)</td>
<td>14</td>
<td>86.66 (0.62)</td>
</tr>
<tr>
<td>wine quality (red)</td>
<td>0.5</td>
<td>0.9</td>
<td>8</td>
<td>62.04 (2.96)</td>
<td>10</td>
<td>60.91 (2.45)</td>
</tr>
<tr>
<td>wine quality (white)</td>
<td>0.9</td>
<td>0.7</td>
<td>11</td>
<td>55.02 (1.22)</td>
<td>10</td>
<td>54.80 (2.00)</td>
</tr>
<tr>
<td>yeast</td>
<td>0.4</td>
<td>0.9</td>
<td>9</td>
<td>60.65 (2.64)</td>
<td>9</td>
<td>60.24 (1.84)</td>
</tr>
</tbody>
</table>

Table 4.2: Parameters and accuracy values from the experiments with the best performance for a single network and within the RANDOMIZED ENSEMBLE Framework.

### 4.3 Best Sampling Parameters

In order to find out if there is a correlation between randomizing the tested parameters and the corresponding diversity/accuracy figures, we conducted the tests in accordance with two strategies:
1. **Nosampling** strategy: all the sources of randomness are excluded, i.e., the data is not sampled and the same training data is shown to each network in the ensemble (the proportion of training patterns $D_p = 1$, no sampling with replacement; the proportion of features $F_p = 1$); the initial weights are randomly assigned with the same seed for the random initialisation ($w_{seed} = 1$), such that the initial weights are the same for all the networks in the ensemble; all the other parameters, apart from the one being tested, are fixed on the same values for all the networks.

2. **Sampling** strategy: in this case we sample the training data with the **RANDOMIZED ENSEMBLE** Framework using constant $D_p$ and $F_p$; the initial weights are randomly set without a constant seed ($w_{seed} = 0$), i.e., the initial weights are different for each network in the ensemble; all the other parameters, apart from the one being tested, are constant.

For the first approach we found the number of hidden neurons per layer $H_{neu}$ by exhaustive search of the parameters which yield the best accuracy for a single classifier (Table 4.2). The lower $b_{min}$ and upper $b_{max}$ bounds values were calculated as follows:

$$b_{min, max} = F_{num} \pm (C_{num} + 1)$$

where $F_{num}$ is the number of features and $C_{num}$ is the number of classes in the original data set.

For the second testing strategy the following parameters needed to be determined: the proportion of training patterns $D_p$, the proportion of features $F_p$ and the number of hidden neurons on hidden layer $H_{neu}$. The values for these parameters for each data set were found through a brute force search with the range of possible parameters:

- $D_p \in \{0.2 \ldots 0.9\}$
- $F_p \in \{0.2 \ldots 0.9\}$
- $H_{neu} \in b_{min}, b_{max}$ where the upper $b_{min}$ and lower $b_{max}$ bounds values were calculated as follows:

$$b_{min, max} = \left\lfloor \frac{F_{num} + C_{num}}{2} \right\rfloor \pm \left\lfloor \frac{F_{num} + C_{num}}{4} \right\rfloor$$

The number of epochs $E_p$ was set to 150 for almost all the experiments. For the experiments with the larger number of hidden neurons and/or hidden layers, we doubled this amount ($E_p = 300$).

Later, the parameters which produced the best accuracy were used as the constant parameters in the experiments where they had to be fixed. The parameters and the accuracy values can be seen in Table 4.2 under the “REF” heading.
4.4 Choice of Random Parameters

With the aim of creating an ensemble of the networks with randomly chosen parameters, we randomly assigned values for each tested parameter $P$ guided by the following conventions:

1. Several tests are conducted.

2. The boundaries of the range $R$, within which the values are varying, are widened for each following test with a certain step size $v_s$.

3. The range for the first test has equal lower and upper bounds, by this the parameter is not varied since its values are held constant for all the ensemble members and is the mean value $\bar{v}$ for the following tests.

4. For each following test $i$, the range $R$ is widened from the mean value $\bar{v}$ by the step size $v_s$ to both directions, such that by each subsequent step the range is extended by $2v_s$:

   \[
   R_{\text{min, max}} = \bar{v} \pm i \cdot v_s
   \]  

5. The mean values (the first test values) for the tested parameters are chosen as follows:

   - The proportion of training patterns $D_p$ and the proportion of features $F_p$: the mean value among possible proportion values, i.e., 0.5.
   - The number of hidden neurons $H_{\text{neu}}$: the values obtained from the experiments with the best performing single network for the nosampling experiments (Table 4.2), and the values obtained from the experiments within the REF for the sampling tests.
   - The number of epochs $E_p$: 150.

In more detail, all of the parameters used for the experiments described in this chapter are represented in Table 4.3, where $bp$ refers to the best parameters values, $st.$ denotes the standard learning rate $L_r = 0.1$. Regarding the proportion of deleted connections $C_p$, it was only applied in the respective experiments and therefore is marked with a dash for the rest of the tests.

4.5 Reading the Graphs

In the next sections we introduce the results of the experiments represented in the form of the scatter plots, density plots and histograms. With the following we propose some important conventions for their reading.
The figures with multiple scatter plots represent one plot for each tested data set. Each point corresponds to an ensemble of neural networks. For all the graphs showing relationship between accuracy and diversity, the \( y \) axis depicts increase or decrease of accuracy compared to the single neural network performance on the respective data set (indicated by red dashed lines).

With few exceptions, the graphs compare ensemble performance for two approaches: with and without sampling. The resulting graphs for the experiments with sampling (\textit{sampling}) are depicted in blue and without sampling (\textit{nosampling}) are shown in red.

The whiskers show standard deviation for resulting ensemble accuracy as the mean value for the outcomes obtained with 5-fold cross-validation.

In the figures for the experiments on the randomizing parameters solid connections between points reflect the order of the experiments, such that in each next experiment the range of the possible value variations is larger than in the previous one, starting from no variation for the tested parameter (indicated with a bigger point of the corresponding color).

Further in this chapter, mentioning range in the graph discussion we refer to the range of variation for the respective parameter.

Similarly, in the figures for the experiments on the large number of hidden neurons and hidden layers, the amount of the tested units is larger with each next experiment. The reader is advised though that connections made with the dashed lines only show the trends revealed by the points.

Please note that vertical and horizontal comparisons between the graphs are not intended in order to distinctly indicate the changes in numbers, i.e., \( x \)- and \( y \)-axes for adjacent plots are not aligned and should be considered independently from each other.

The histograms and the density plots show respectively the distribution of accuracy changes and diversity aggregated for 11 tested data sets; the vertical dashed lines demonstrate mean values.
## 4.6 The Experimental Parameters

<table>
<thead>
<tr>
<th>Experiment</th>
<th>$D_p$ influence</th>
<th>$F_p$ influence</th>
<th>Random $D_p$</th>
<th>Random $F_p$</th>
<th>Random $H_{neu}$</th>
<th>Random $H_{lay}$</th>
<th>High $H_{neu}$ and $H_{lay}$</th>
<th>Random $D_p$, $F_p$, $H_{neu}$, $H_{lay}$</th>
<th>Random $D_p$ and $F_p$, $H_{neu}$, $C_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>step</td>
<td>range</td>
<td>start</td>
<td>step</td>
<td>range</td>
<td>start</td>
<td>step</td>
<td>range</td>
<td>range</td>
</tr>
<tr>
<td>$D_p$</td>
<td>0.05</td>
<td>0.95</td>
<td>(0.05;0.95)</td>
<td>1.0</td>
<td>bp</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$F_p$</td>
<td>1.0</td>
<td></td>
<td>0.05</td>
<td>0.95</td>
<td>(0.05;0.95)</td>
<td>bp</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$H_{neu}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$H_{lay}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.3: Parameters used for the experiments. $bp$ refers to the best parameters from the experiments within REF introduced in the Table 4.2.
4.7 Experiments

4.7.1 Influence of Sampling

Figure 4.1: Influence of the sampling strategy on ensemble accuracy. The features’ proportion, $F_p$, dramatically affects the accuracy figures, specifically for $F_p < 0.5$, whereas the training patterns proportion, $D_p$, barely influences accuracy numbers. The maximum accuracy is achieved with the maximum $F_p$ and $D_p$.

As described in Section 3.1.1 the influence of the magnitude of the proportion of features $F_p$ and the proportion of training instances $D_p$ is compared here to see which of those two parameters influence diversity and accuracy in the ensemble more significantly.

Figure 4.1 shows changes of ensemble accuracy compared to the single model performance (dashed red line) with increasing values of both variables $F_p$ and $D_p$. Accuracy values drops dramatically for very small $F_p$ values for the most of the data sets,
continually improving with increasing $F_p$. The $F_p$ curves reach their saturation point around 0.3 for 3 data sets, while for the other 8 data sets this point is attained after 0.5.

Regarding the $D_p$ curves, the accuracy values hold on the same level for every data set, except sonar, where very small $D_p$ values negatively influence accuracy, but trend to the accuracy of the single model with higher values.

All things considered, it seems reasonable to say that the influence of feature proportion $F_p$ is more significant than that of the proportion of training instances $D_p$. This allows us to verify Hypothesis 2b. It is also confirmed by the results, aggregated for all the data sets, which demonstrate a statistically significant difference ($p < 0.05$) in accuracy distribution for the two tested parameters and in Figure 4.2a where in the case of $D_p$, most of the accuracy change values are concentrated in the area with the values slightly higher than 0, being slightly lower than 0 for the $F_p$ values distribution. The reason of that is the poor ensemble performance with the low $F_p$ magnitudes.

Figure 4.3 introduces the changes in both the diversity and accuracy with the accuracy figures showing how accuracy changed compared to the benchmark accuracy.

Interestingly, in the plots for the blocks, chess, faults, whitewine and yeast data sets the feature proportion $F_p$ curves make the bends from the minimal proportion marks (big blue dots). It reveals that the minimal $F_p$ does not yield the maximum diversity numbers, as it might be expected, but trend towards the benchmark accuracy with the higher $F_p$ values.

Similarly to accuracy, diversity seems to be notably higher for 5 data sets (chess, faults, spambase, statlog, waveform) for the feature proportion experiments. The results, aggregated for all the data sets, together with the diversity density plot in Figure 4.2b manifest our suggestion, that $F_p$ affects diversity more significantly compared to $D_p$, which supports Hypothesis 2a.
4.7.2 Randomized Ensemble Framework Parameters, Diversity and Accuracy

In the experiments on randomizing the parameters of the RANDOMISED ENSEMBLE Framework we separately varied the proportion of training patterns \( D_p \) and the proportion of features \( F_p \) parameters within two groups of experiments: sampling and nosampling. The former refers to the variation of the tested parameter with the second one (either \( D_p \) or \( F_p \)) held constant. In other words, in the case of varying \( D_p \), we additionally use RSM, while when varying \( F_p \), BAGGING is additionally applied to the data.

Figure 4.4 shows that for the nosampling experiments on the waveform, spambase,
yeast and whitewine data sets there is a steady but insignificant growth of diversity when the range increases. For the other six data sets there is no clear trend to observe. Accuracy slightly increases only in the case of the spambase and waveform data sets, being worse than the benchmark accuracy for all the other data sets. Consequently, we have to reject both Hypothesis 3a and Hypothesis 3b.

The results of the experiments with sampling demonstrate a considerable diversity increase compared to nosampling approach, but again without any clear trend with regards to the range increase. Figures 4.5a and 4.5b and unpaired t-tests reveal significant difference in the diversity figures but insignificant changes in accuracy distribution for sampling and nosampling experiments, respectively validating Hypothesis 3d, but negating Hypothesis 3c.

Despite the lack of clear trends regarding diversity changes with the range growth for the nosampling experiments, the diversity numbers increase compared to the initial
CHAPTER 4. EXPERIMENTAL RESULTS

Figure 4.5: Ensemble (a) accuracy and (b) diversity distributions in the experiments on the training patterns proportion $D_p$ range variation with and without sampling. Insignificant difference in accuracy distribution, significant difference in diversity distribution.

point, depicting the ensemble with static $F_p$ (Hypothesis 4a is verified) as shown in Figure 4.6. Once again, we have to reject Hypothesis 4b, since for the majority of the data sets accuracy seems to stay on the same level.

Similarly, the sampling curves do not reveal a clear tendency of diversity growth (except on the spambase and statlog data sets), but substantiate a diversity boost compared to the nosampling approach.

The results, aggregated for all the data sets, in Figure 4.7 demonstrate statistically significant variations in diversity figures for both groups (Hypothesis 4c is supported) and insignificant difference in accuracy changes distribution (Hypothesis 4d has to be rejected).
Figure 4.6: Influence of the random feature proportion $F_p$ range on diversity and ensemble accuracy with and without sampling. Higher diversity for the experiments with sampling. The numerical results can be seen in Table B.3 of Appendix B.

Figure 4.7: Ensemble (a) accuracy and (b) diversity distributions in the experiments on random feature proportion $F_p$ range variation with and without sampling. Insignificant difference in the accuracy distribution, significant difference in the diversity distribution.
4.7.3 Random Number of Hidden Neurons, Diversity and Accuracy

Figure 4.8: Influence of the range of the random number of hidden neurons $H_{neu}$ on ensemble diversity and accuracy with and without sampling. Higher diversity for the experiments with REF; almost no difference in both accuracy and diversity numbers for the sampling tests; growing diversity but quite similar accuracy for the nosampling tests. The numerical results can be seen in Table B.4 of Appendix B.

As described in Section 3.2.1 we randomly vary the number of hidden neurons $H_{neu}$, having excluded all the other sources of randomness for the nosampling approach and adding REF for the sampling approach.

The figure 4.8 gives a demonstration of the benefits of the varying $H_{neu}$ for the ensemble diversity. Interesting behaviour is observed for the redwine, sonar, spambase, statlog, waveform and whitewine data sets, where the second ensembles with a small range of the parameter variation (after the first initial ensembles with no variation) have the remarkably increased diversity figures. On the contrary, the following range
Ensemble (a) accuracy and (b) diversity distributions in the experiments on the random number of hidden neurons $H_{\text{neu}}$ range variation with and without sampling. Insignificant difference in accuracy and significant in diversity.

Figure 4.9: Ensemble (a) accuracy and (b) diversity distributions in the experiments on the random number of hidden neurons $H_{\text{neu}}$ range variation with and without sampling. Insignificant difference in accuracy and significant in diversity.

increase yield either tiny or insignificant augmentation in the diversity figures. However, it is clear that Hypothesis 6a is validated.

The most clear trend for the nosampling tests is revealed on the block data set where diversity growths steadily even reaching the level of diversity for the ensembles with REF. This is not applicable to any other data set in the experiments on this approach.

Notably, by contrast to the nosampling approach, adding REF makes both ensemble diversity and accuracy quite similar from experiment to experiment. Nevertheless, comparing the sampling and nosampling approaches, diversity considerably increases in the case of the former. This, unpaired t-test, remarkable difference of mean values and the areas of density for the sampling and nosampling experiments (Figure 4.8b) allow us to verify Hypothesis 6c.

On the other hand, accuracy changes do not look somewhat significant for both types of the experiments and for the most of the data sets accuracy figures are below the benchmark lines. The mean values for the aggregated data are almost identical (Figure 4.9a) and the difference for the accuracy changes distribution is not statistically significant. Therefore we have to reject both Hypothesis 6b and 6d.
4.7.4 Random Number of Hidden Layers, Diversity and Accuracy

Figure 4.10: Influence of the range of the random number of hidden layers $H_{\text{lay}}$ on ensemble diversity and accuracy. Higher diversity for the experiments with sampling; a tendency to increase diversity but decrease accuracy for both approaches with the range increase. The numerical results can be seen in Table B.5 of Appendix B.

Continuing with the idea mentioned in Section 3.2.1, in the following experiments we vary the number of hidden layers $H_{\text{lay}}$. The results of the experiments are represented in Figure 4.10. Interestingly, the behaviour of the curves for nosampling and sampling approaches for some data sets (blocks, chess, and especially obviously for spambase and statlog) repeat each other but in the area of higher diversity for the sampling experiments.

Importantly, in both types of the experiments, varying $H_{\text{lay}}$ does increase diversity in the ensemble (Hypothesis 5a and Hypothesis 5c are verified) with a fairly clear trend for nosampling ensembles, but to some extent chaotic curves for the redwine, sonar, chess data sets. When it comes to accuracy, the figures even tend to worsen in
4.7. Experiments

![Graphs showing accuracy and diversity distribution.](image)

(a) Accuracy changes distribution. Unpaired t-test, \( p < 0.05 \).

(b) Diversity density plot. Unpaired t-test, \( p < 0.05 \).

Figure 4.11: Ensemble (a) accuracy and (b) diversity distribution in the experiments on the random number of hidden layers \( H_{\text{lay}} \) range variation with and without sampling. Significant difference in both accuracy and diversity figures.

the experiments with the REF (Hypothesis 5b and Hypothesis 5d are disproved). It is especially true for the block data set.

Confirmed by the t-test with \( p < 0.05 \) and by notable differences in the mean values on the aggregated data, both accuracy and diversity significantly differ for the nosampling and sampling approaches (Figure 4.11). However, for the sampling experiments the diversity figures are higher, while more significant accuracy numbers are obtained in the nosampling experiments.
4.7.5 Large Number of Hidden Neurons and Hidden Layers, Diversity and Accuracy

Figure 4.12: Influence of the high number of hidden neurons $H_{\text{neu}}$ and high number of hidden layers $H_{\text{lay}}$ on ensemble diversity and accuracy. Increased number of neurons much less influential than growing number of hidden layers. For the former, increasing diversity values but decreasing accuracy figures for the most of the data sets. The numerical results can be seen in Table B.6 of Appendix B.

To a certain extent, staying apart from the set of the experiments on randomization of a particular parameter, but continuing the exploration of the influence of number of hidden layers $H_{\text{lay}}$ and number of hidden neurons $H_{\text{neu}}$, are the experiments with constant but large number of $H_{\text{lay}}$ or $H_{\text{neu}}$.

In this experiments we do not operate on the range and do not vary the parameters among the members of an ensemble, keeping them constant. Also in this case we do not conduct the experiments without REF since all the members of an ensemble are
4.7. Experiments

(a) Accuracy changes distribution.
Unpaired t-test, $p < 0.05$

(b) Diversity density plot.
Unpaired t-test, $p < 0.05$

Figure 4.13: Ensemble (a) accuracy and (b) diversity distribution in the experiments on the high number of the hidden neurons $H_{neu}$ and high number of the hidden layers $H_{lay}$. Significant difference for both diversity and accuracy values.

The same and there is no any other source for diversity in the ensemble. Instead, we only use the approach with the REF gradually increasing the number of the tested units (layers or neurons).

The most obvious feature seen in Figure 4.12 is that the results on the growth of $H_{lay}$ reveal no clear correlation neither with accuracy nor with diversity. It is slightly better in the case of diversity figures, but a significant amount of oscillations makes it hard to interpret those curves. Therefore we have to reject both Hypothesis 7a and Hypothesis 7b.

In contrast to the experiments with large number of neurons, the experiments with large number of hidden layers seem to yield more clear results. For most of the data sets (blocks, faults, seeds, statlog, waveform), diversity distinctly increases. Interestingly, for some data sets (redwine, sonar, whitewine, yeast) there is a “loop-back” effect. Apparently, diversity increases until a certain amount and the following increase of this amount decreases diversity in the ensemble (Hypothesis 8a is partly validated).

With concern to accuracy it is noticeable that accuracy gradually declines when $H_{lay}$ gets bigger for the data sets where diversity strictly increases. However, for the “loop-back” data sets accuracy seems to be decreasing together with diversity on the backward curve. Thereby Hypothesis 8b is proven to be false.

The aggregated results also demonstrate a significant difference in both accuracy (Figure 4.13a) and diversity (Figure 4.12b).
4.7.6 Random Learning Rate, Diversity and Accuracy

Figure 4.14: Influence of the range of the random learning rate $L_r$ on ensemble diversity and accuracy with and without sampling. Higher diversity for the sampling approach, but similar for both approaches accuracy figures. The numerical results can be seen in Table B.7 of Appendix B.

In the experiments described in this section we vary learning rate $L_r$ among the members of an ensemble. The results are represented in Figure 4.14. The plots on most of the data sets demonstrate insignificant steady growth of diversity in the nosampling experiments. Interestingly, accuracy figures slightly grow and diminish after (blocks, redwine, sonar, waveform), being higher than the benchmark lines for two of the mentioned data sets.

The main difference, when compared to the results on the other parameters, for the blocks, chess, seeds data sets is that the diversity changes are notably more remarkable on the experiments with the REF. Accuracy for these experiments has either a trend to a slight decrease or stays on the level of the benchmark lines.
4.7. Experiments

(a) Accuracy changes distribution. Unpaired t-test, $p > 0.05$.

(b) Diversity density plot. Unpaired t-test, $p < 0.05$.

Figure 4.15: Ensemble (a) accuracy and (b) diversity distributions in the experiments on the random learning rate $L_r$ range variation with and without sampling. Significant difference in diversity but insignificant difference in accuracy distribution.

The observations from the aggregated data indicate significant difference in the diversity figures when sampling and nosampling experiments are compared. However, when it comes to accuracy the difference is statistically insignificant (Figure 4.15).

Taking the findings above into account, we can validate Hypothesis 9a and Hypothesis 9c, but have to reject Hypothesis 9b and Hypothesis 9d.
4.7.7 Random Number of Epochs, Diversity and Accuracy

Figure 4.16: Influence of the range of the random number of epochs $E_p$ on ensemble diversity and accuracy. Diversity tends to increase with the growth of the variation range for nosampling approach. The numerical results can be seen in Table B.9 of Appendix B.

As already mentioned in Section 3.2.2, the number of epochs $E_p$ defines the length of the trajectories followed by the learning algorithm in the search space. Shorter the trajectory, less likely it reaches a local minima.

The experiments described in this section examine the effect of the random variation of $E_p$ among the members of an ensemble of neural networks.

The results in Figure 4.16 shows that regarding the nosampling test for the spambase, whitewine, yeast datasets diversity tends to get bigger with the larger range, especially distinctly diversity increases for the statlog dataset. On the other hand, for the rest of the data sets the results seem to be chaotic and therefore hardly
interpretible. Yet for the most of the data sets a tendency of the diversity figures to grow is clear, therefore we consider Hypothesis 10a to be proved.

Regarding accuracy figures, for both types of the experiments we have to admit that the accuracy figures barely change with either the range or diversity increase (Hypothesis 10b is rejected).

Concerning the sampling experiments, the results show that the combination of the REF and the variation of the learning rate parameter leads to a significant increase of diversity, but since diversity does not increase with the range increment for the experiments with sampling, we can conclude that the REF itself notably increased diversity in the ensemble (Hypothesis 10c is validated).

The Sampling approach also causes a slight increase of accuracy, but again since accuracy does not get larger with a bigger range we have to claim that accuracy growth occurs due to the REF. Interestingly, for the whitewine and yeast data sets the approach even yields a decrease for both the accuracy and diversity figures, since the initial (bigger) points are in the most right upper part of the graphs.

Nevertheless, comparing two approaches with the aggregated results, the difference in the accuracy distribution, shown by unpaired t-test, turned out to be significant (Figure 4.17a), therefore validating Hypothesis 10d. And, as expected, difference in the diversity distribution demonstrated in Figure 4.17b is demonstrated to be significant.
4.7.8 Random Connections, Diversity and Accuracy

Figure 4.18: Influence of the range of proportion of randomly deleted connections $C_n$ on ensemble diversity and accuracy with and without sampling. Diversity increases with increase of $C_n$, but accuracy tends to decrease. The numerical results can be seen in Table B.8 of Appendix B.

As described in Section 3.3, in these experiments a random proportion of the connections between neurons, regulated by the parameter $C_n \in \{0.05; 0.8\}$, is removed for each network in the ensemble.

Figure 4.18 shows the experimental results. It is interesting to note that for six data sets the curves of the nosampling experiments overlap with the sampling experiments curves. It speaks in favor of the fact that the random removing of the connections between neurons is quite beneficial in terms of inducing disagreement among the networks. For the chess data set the blue curve (nosampling experiments) even goes further than the red one (sampling experiments).

Clearly, both approaches stimulate diversity in the ensemble (Hypothesis 11a and Hypothesis 11c are shown to be correct). The density plot (Figure 4.19b) and unpaired
(a) Accuracy changes distribution. Unpaired t-test, $p > 0.05$.

(b) Diversity density plot. Unpaired t-test, $p < 0.05$.

Figure 4.19: Ensemble (a) accuracy and (b) diversity distributions in the experiments on the proportion of randomly deleted connections $C_n$ variation with and without sampling.

t-test allows us to claim that the diversity distribution for the sampling and nosampling experiments is significantly different. However, the approach does not seem to be convincing in respect to increase of the accuracy figures. Certainly, when the proportion of the deleted connections is too large, the curves for the both sampling and nosampling experiments indicate poor ensemble performance (therefore we reject Hypothesis 11b). Similarly, applying the REF does not improve the ensemble performance, since the difference between accuracy numbers is not significant (Figure 4.19a). For that reason Hypothesis 11d has to be rejected.
4.7.9 Combination of Randomized Parameters, Diversity and Ensemble Accuracy

Figures 4.20: Combination of randomized parameters: comparison between the feature proportion \( F_p \) and the training pattern proportion \( D_p \), and number of hidden neurons \( H_{neu} \) and number of hidden layers \( H_{lay} \). The numerical results can be seen in Table B.11 of Appendix B.

The experiments conducted in the scope of this section comprise a combination of the various approaches described above. They are implemented by simultaneously randomizing several parameters. Analogously to the experiments described in the previous sections, the ranges of the parameters are concurrently increased with each step.

Figure 4.20 displays and compare the results of two approaches. The approaches imply generation of an ensemble of neural networks with the following parameters chosen randomly:
4.7. Experiments

(a) Accuracy changes distribution. Unpaired t-test, \( p > 0.05 \).

(b) Diversity density plot. Unpaired t-test, \( p < 0.05 \).

Figure 4.21: Ensemble (a) accuracy and (b) diversity distributions in the experiments on the combination of randomized \( F_p \) and \( D_p \), and \( H_{neu} \) and \( H_{lay} \). Significant difference in the diversity distribution but insignificant difference in the accuracy distribution.

1. The approach referred as \( RFR \):
   
   (a) the proportion of training patterns \( D_p \)
   
   (b) the proportion of features \( F_p \).

2. The approach referred as \( RNL \):
   
   (a) the number of hidden neurons \( H_{neu} \)
   
   (b) the number of hidden layers \( H_{lay} \).

The \( RFR \) approach seem to be working in the matter of diversity increase for six data sets outputting quite clear trend. However, on the whitewine, chess and yeast data sets the curves have the backward loops, demonstrating that at some point the range increase yields diversity decrease.

When compared to each other, \( RNL \) approach shows more significant and explicit diversity increase than \( RFR \). Here for all the data sets except sonar and redwine diversity enlarges with the range augmentation.

With regard to accuracy, only for a couple of data sets (whitewine and statlog) \( RHL \) approach outperforms the benchmark accuracy. In the case of \( RFR \) the same holds only for the spambase data set. In general, the accuracy figures are inclined to shrink for the \( RNL \) and to stay relatively the same for \( RFR \).

The exceptions for \( RNL \) are the statlog and faults data sets. For the former accuracy first rise dramatically, but gradually diminishes with the bigger range of the parameters variations. Regarding the latter, accuracy have a tendency to grow throughout the entire range increase with a small decline in the middle.

Comparing aggregated results for the two approaches reveal significant difference in the diversity figures and insignificant difference in the accuracy distribution (Figure 4.21).
Figure 4.22: Combination of randomized parameters: the feature proportion $F_P$, the training patterns proportion $D_P$ and the number of hidden neurons $H_{neu}$: comparison between additionally randomized the proportion of random connections $C_n$ and the number of hidden layers $H_{lay}$. The latter stimulates generation of much more diverse ensembles. The numerical results can be seen in Table B.10 of Appendix B.

The next step in our experiments is to randomize even more parameters. The results demonstrated in Figure 4.22 compare two approaches where the following parameters are randomized:

1. The approach referred as connections:
   (a) the proportion of training patterns $D_P$
   (b) the proportion of features $F_P$
   (c) the number of hidden neurons $H_{neu}$
   (d) the proportion of randomly deleted connections between neurons $C_n$
4.7. Experiments

Figure 4.23: Ensemble (a) accuracy and (b) diversity distributions in the experiments on combination of randomized $F_p, D_p, H_{neu}$ and additionally randomized $C_n$ and $H_{lay}$. Significant difference in the diversity numbers; insignificant difference in the accuracy distribution.

2. The approach referred as layers:

(a) the proportion of training patterns $D_p$
(b) the proportion features $F_p$
(c) the number of hidden neurons $H_{neu}$
(d) the number of hidden layers $H_{lay}$

The experiments are referred here as connections and layers due to the fact that the three former varying parameters stay the same for the both approaches, the latter ones distinguishes one approach from the other one.

The most eye-catching observation is that compared to the connections experiments, the layers experiments generate more diverse ensembles.

More importantly, a very clear trend in diversity improvement is revealed with each step for the layers experiments. Also the difference in diversity figures for each two adjacent steps is quite large when compared to the connections experiments curves. Neither former nor latter have been observed in our experiments before.

On the contrary, the connections experiments do not demonstrate convincing results. Although diversity distinctly increases with each step for the redwine, seeds, spambase, waveform and whitewine data sets, for the other data sets the behaviour of the curves is rather confusing.

In spite of the diversity growth, accuracy improves only for few data sets (statlog for both approaches and whitewine) for the layers approach. The experiments on the other data sets in general revealed a tendency of a minor accuracy drop.

The t-tests on the aggregated data and the respective plots (Figure 4.23) show significant difference between the approaches in terms of diversity and insignificant difference regarding accuracy.

Summarising the results of the experiment described in this section with regard to our hypotheses we can conclude that:
• Randomizing several parameters when creating neural networks for an ensemble encourages the networks to be more diverse (Hypothesis 12a is verified). Nonetheless, increased diversity numbers do not yield accuracy improvement (Hypothesis 12b is rejected).

• Randomizing the data and model parameters together with applying the RANDOM ENSEMBLE Framework do increase diversity in the ensemble (Hypothesis 1a is supported), but do not stimulate the ensemble accuracy augmentation (Hypothesis 1a is not supported).
4.8 Additional Experiments

Separately from the main experiments in favor of being able to establish evidence or to refute the proposed earlier hypotheses, we conducted a group of experiments for the sake of clarifying the influence of the number of hidden layers $H_{lay}$ and the number of hidden neurons $H_{neu}$ in the ensemble.

For that we explore the influence of the two parameters on a single network. The graphical results are placed to Appendix A.

The results shown in Figure A.1 explores effect of the number of hidden neurons on a single network performance. The plots reveal that for most of the data sets accuracy remains stable up to 500 hidden neurons. For sonar, seeds, faults and statlog data sets the number of the hidden neurons when accuracy starts to decrease is close to 250. Together with the accuracy decrease (Figure A.2a), a clear significant increase in standard deviation figures is observed. The box plot diagram in Figure A.2b demonstrates the aggregated for 11 data sets distribution of the standard deviation figures with increase of mean values from value closed to zero for 10 neurons to over 20 for 1010 neurons.

However, the Figure A.3 shows that the reason for accuracy decrease is not overfitting since the accuracy numbers on the test and train data are very similar. The reason of the poor performance seems also not to be a lack of the number of iteration for larger networks (Figure A.4) since the the networks perform mostly identically with 100 and 700 epochs.

The same holds for the number of hidden layers $H_{lay}$ for a single network: accuracy decreases with the growth of $H_{lay}$ (Figure A.5). Interestingly, the networks do not overfit the train data, demonstrating similar performance on both training and test data (Figure A.6). Nevertheless, for some data sets (sonar, yeast, statlog, waveform) accuracy with 700 iterations superior to the accuracy figures with 100 iterations especially the difference is apparent for the networks with the higher amount of hidden layers.

4.9 Comparison of Approaches

In this section we compare the results of all the conducted experiments considering different approaches. Below we provide the sets of the box plots for comparison of diversity and accuracy distribution for each approach. The box plots are ordered respectively to their median values of the inspected measure (either accuracy or diversity).

In the graphs below the approaches are coded as follows:

- **FRNC** - randomizing proportion of features $F_p$, proportion of training instances $D_p$, number of hidden neurons $H_{neu}$, proportion of deleted connections between neurons $C_n$
Chapter 4. Experimental Results

- **FRNL** - randomizing proportion of features \( F_p \), proportion of training instances \( D_p \), number of hidden neurons \( H_{neu} \), number of hidden layers
- **HL** - randomizing the large number of hidden layers
- **HN** - randomizing the large number of hidden neurons
- **RC** - randomizing proportion of deleted connections between the neurons \( C_n \)
- **RCR** - randomizing proportion of features \( F_p \) and proportion of training instances \( D_p \)
- **RE** - randomizing number of epochs \( E_p \)
- **RF** - randomizing proportion of features \( F_p \)
- **RL** - randomizing number of hidden layers \( H_{lay} \)
- **RLR** - randomizing learning rate \( L_r \)
- **RN** - randomizing number of hidden neurons \( H_{neu} \)
- **RNL** - randomizing number of hidden neurons \( H_{neu} \) and number of hidden layers \( H_{lay} \)
- **RR** - randomizing proportion of training instances \( D_p \)

We divided all the experiments into two groups: the experiments conducted within the REF and the experiments conducted without the REF. The reason is that the ensembles generated within the REF are, as expected, more diverse than the ones generated without.

The dashed green line in the graphs examining accuracy refers to the benchmark accuracy used in our experiments, i.e., the best accuracy outcome by a single neural network on a respective data set.

The dashed red line in the plots investigating diversity, points on the benchmark diversity. This diversity was extracted from the experiments with REF resulted in the best accuracy respectively for each data set.

Among the experiments without REF (Figure 4.24) the approach providing the most diverse ensembles seem to be RF (randomized feature proportion \( F_p \)) yielding the highest diversity figures for 8 data sets and being 2nd or 3rd best in the other 3 data sets. More importantly, it is the only approach able to exceed the benchmark diversity for all the datasets except the sonar data set.

Another approach showing the good results in terms of ability to induce diversity is RR (randomizing training patterns proportion \( D_p \)).

Interestingly, both RF and RR (especially the former one) approaches demonstrate very low diversity variability, which means that the variation range increase does not significantly influence diversity.
4.9. COMPARISON OF APPROACHES

Figure 4.24: Comparison of diversity distribution for the approaches without sampling.

The opposite holds for the RLR (randomized learning rate $L_r$) experiments which disclose a high variation of the diversity figures for statlog, sonar and spambase data sets.

It is interesting to note that RC (randomized deleted connections proportion $C_n$) has outliers for most of the data sets which characterises the approach as unstable.

The worst performing approach, concerning diversity stimulation in an ensemble, is the RE (randomizing epochs $E_p$) approach: its median values are the lowest for most of the datasets.

Most importantly, in the case of the experiments without REF diversity figures exceed the benchmark diversity for few approaches and few data sets. The opposite is true for the experiments with REF (Figure 4.25) where almost all the results are better than the benchmark diversity.

Not surprisingly, the most diversity-productive approaches are those where several parameters were randomized simultaneously and the one with the highest diversity results is FRNL (randomized proportion of features $F_p$, proportion of training
Notably, the experiments with the randomized learning rate (RLR) for both the experiments with and without REF are among the most diversity-productive ones for some of the data sets, but at the same time they are the approaches with a very high variance.

The RR (randomizing the proportion of training instances $D_p$), RE (randomizing the number of epochs), RC (randomizing the proportion of deleted connections $C_n$) approaches seem to be the less effective approaches when it comes to generating diverse neural networks.

However, the most intriguing part is to see how the diversity figures correlate with the accuracy figures. Surprisingly the “winner” of both groups of the approaches in terms of diversity generating, RF (randomizing proportion of features $F_p$), turned to be among the least accurate approaches for the nosampling group, being the best one only for the sonar data set (Figure 4.26). This does not change much for the sampling group either (Figure 4.27).

The RL (randomizing the number of hidden layers $H_{lay}$) which demonstrate an average diversity-inducing ability compared to the other approaches in the experiments without sampling exhibits the best accuracy distribution for most of the data sets. It is also proved by the aggregated results in Table 4.4 where the results are ordered by the mean accuracy numbers. Among the experiments with sampling RF (randomizing feature proportion $F_p$) is the 5th best out of 13 approaches. However, the mean
accuracy figures for all the experiments are fairly close to each other.

By contrast to the diversity results, the experiments with simultaneous variation of several parameters in general yield worse accuracy compared to those where only one parameter is randomized.

Comparing the results with two benchmark accuracy values - the best one from a single model performance and the best one from the ensemble with REF, we can observe that, first, the two benchmarks are very similar for most of the data sets and if the difference exists it is not significant; second, the accuracy results in the experiments with REF are only slightly better than the accuracy figures in the nosampling experiments. This claim is proved by the accuracy distribution for the aggregated results depicted in Figure 4.28a where the distributions seem to be very similar and the mean values coincide. The frequency of the values for sampling is higher due to the higher number of the experiments with sampling and the difference between two groups is not statistically significant.

In contrast, the aggregated results for diversity differ significantly (Figure 4.28b). It is clearly demonstrated by the different areas on the density plot occupied by the
### Chapter 4. Experimental Results

Figure 4.27: Comparison of accuracy distribution for the approaches with sampling.

(a) Aggregated accuracy distribution. Unpaired t-test, $p < 0.05$.

(b) Diversity density plot. Unpaired t-test, $p < 0.05$.

Figure 4.28: Ensemble (a) accuracy and (b) diversity distributions in the experiments on the approaches with and without *sampling*. The experiments with *sampling* demonstrate significantly higher diversity figures; accuracy distribution is quite similar for the experiments with and without REF

* sampling and *nosampling* approaches. The area of the former situates notably to the right, while the most of the values of the latter one are concentrated quite close to 0.
Table 4.4: Comparison of the accuracy and diversity figures aggregated for all data sets for the approaches with and without sampling.
4.10 Summary

The experiments described throughout this chapter makes it possible to conclude about the last and the main hypothesis we proposed. Table 4.5 aggregates the results of all the conducted experiments with regard to the hypotheses proposed in Chapter 3.

The experimental results proved that the randomizing different parameters of an ensemble of neural networks certainly leads to an increase of disagreement among the members of an ensemble of neural networks, however the increased diversity do not necessarily leads to any improvement in the ensemble generalisation ability. In other words, diversity increase by randomizing data and model manipulation parameters in an ensemble of neural networks does not result in accuracy improvement. Therefore we claim that Hypothesis 13 has to be rejected.

<table>
<thead>
<tr>
<th>#</th>
<th>Hypothesis</th>
<th>Accepted/ rejected</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Randomizing the data and model parameters along with applying REF sampling on the data increases a. diversity and b. accuracy in the ensemble.</td>
<td>a. accepted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>b. rejected</td>
</tr>
<tr>
<td>2</td>
<td>The influence on a. diversity and b. accuracy of the proportion of features $F_p$ is more significant compared to the influence of the proportion of training patterns $D_p$ introduced to each model in an ensemble of neural networks.</td>
<td>a. rejected</td>
</tr>
<tr>
<td></td>
<td></td>
<td>b. accepted</td>
</tr>
<tr>
<td>3</td>
<td>Randomizing the training patterns proportion $D_p$ parameter of REF a. increases diversity among the members of an ensemble of neural networks and b. improves the ensemble performance; feature sampling (RSM), applied additionally, improves both c. diversity and d. accuracy.</td>
<td>a. rejected</td>
</tr>
<tr>
<td></td>
<td></td>
<td>b. rejected</td>
</tr>
<tr>
<td></td>
<td></td>
<td>c. rejected</td>
</tr>
<tr>
<td></td>
<td></td>
<td>d. accepted</td>
</tr>
<tr>
<td>4</td>
<td>Randomizing the features proportion $F_p$ parameter of REF a. increases diversity among the members of an ensemble of neural networks and b. improves the ensemble performance; Bagging, applied additionally, improves both c. diversity and d. accuracy.</td>
<td>a. accepted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>b. rejected</td>
</tr>
<tr>
<td></td>
<td></td>
<td>c. accepted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>d. rejected</td>
</tr>
<tr>
<td>5</td>
<td>By randomizing the number of hidden layers $H_{lay}$ it is possible to a. stimulate diversity among the members of an ensemble of neural networks and b. by this to improve the ensemble performance; REF, applied additionally, improves both c. diversity and d. accuracy.</td>
<td>a. accepted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>b. rejected</td>
</tr>
<tr>
<td></td>
<td></td>
<td>c. accepted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>d. rejected</td>
</tr>
<tr>
<td>6</td>
<td>By randomizing the number of hidden neurons $H_{neu}$ it is possible to a. stimulate diversity among the members of an ensemble of neural networks and b. by this to improve the ensemble performance; REF, applied additionally, improves both c. diversity and d. accuracy.</td>
<td>a. accepted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>b. rejected</td>
</tr>
<tr>
<td></td>
<td></td>
<td>c. accepted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>d. rejected</td>
</tr>
<tr>
<td>7</td>
<td>Increasing the amount of hidden neurons in an ensemble of neural networks positively influences a. the ensemble diversity and b. the ensemble performance.</td>
<td>a. rejected</td>
</tr>
<tr>
<td></td>
<td></td>
<td>b. rejected</td>
</tr>
<tr>
<td>Hypothesis</td>
<td>Outcome</td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>-----------</td>
<td></td>
</tr>
</tbody>
</table>
| Increasing the amount of hidden layers in an ensemble of neural networks positively influences a. the ensemble diversity and b. the ensemble performance. | a. accepted  
| b. rejected                                                               |           |
| By randomizing the learning rate $\epsilon$ parameter while fixing the increase parameter $\eta^+ = 1.2$ and the decrease parameter $\eta^- = 0.5$ for each member of the ensemble it is possible to a. promote diversity among the members of an ensemble of neural networks and b. to improve the ensemble generalisation ability; REF, applied additionally, improves both c. diversity and d. accuracy. | a. accepted  
| b. rejected                                                               |           |
| a. accepted                                                               |           |
| Table 4.5: List of hypotheses with results.                               |           |
| It is possible to promote a. diversity among the members of an ensemble of neural networks by randomizing the number of epochs $E_p$ parameter for each member of the ensemble and b. by this to increase accuracy; REF, applied additionally, improves both c. diversity and d. accuracy. | a. accepted  
| b. rejected                                                               |           |
| Simultaneous randomizing of the various ANN and REF parameters increases a. diversity in the ensemble of neural networks and b. improves the ensemble performance. | a. accepted  
| b. rejected                                                               |           |
| Increasing diversity among the members of an ensemble of neural networks improves the ensemble performance. | rejected   |
| Increasing diversity among the members of an ensemble of neural networks improves the ensemble accuracy. | rejected   |
Chapter 5

Conclusion and Future Work

The experimental results presented in this work show that randomizing various neural network parameters applied together with the RANDOMIZED ENSEMBLE framework and randomizing the REF parameters is an efficient way of inducing diversity in an ensemble of neural networks. Some parameters demonstrate a stronger ability to influence disagreement among the members of an ensemble. When it comes to stimulating disagreement the REF parameters, the proportion of features $F_p$ and the proportion of training patterns $D_p$ seem to be the most efficient ones. However, since the variation of the diversity values for all the data sets is very low, the range of the parameter variation does not seem to be influential. This implies that not the random parameter assigning makes ensemble more diverse, but sampling itself. Concerning the model manipulation parameters, learning rate $L_r$ and number of hidden layers $H_{lay}$ appear to be the most diversity-productive among the other parameters considered.

As expected, the approaches that include the elements of both model manipulation (randomizing the neural network parameters) and data manipulation (REF) approaches as well as the experiments where several parameters were simultaneously randomized yield more diverse ensembles. A certain winner among these approaches is the one where we vary four parameters: the proportion of training instances $D_p$, the proportion of features $F_p$, the number of hidden layers $H_{lay}$ and the number of hidden neurons $H_{neu}$.

However, we did not find a prove of consistent relationships between diversity and accuracy in an ensemble of neural networks: in spite of increase of diversity, generalisation ability of the ensemble is not necessarily improved. Barely any of the investigated approaches in the experiments without REF outperforms benchmark (single model) accuracy. The same holds for the experiments with REF where some approaches surpass single model approach but do not exceed best REF benchmark accuracy.
5.1 Future Work

Considering and summarising the results of the experiments, it is reasonable to suggest that diversity is not the only condition ensuring a good generalisation of an ensemble of neural networks. Possibly, ANNs are not suitable for ensemble learning or an ensemble of neural networks is not sensitive enough to diversity changes with regard to accuracy improvement. This question needs a further and deeper investigation with possible applying of the other neural networks algorithms.

Another question for the investigation could be a mechanism of overfitting of neural networks and how overfitting of a single neural network correlates with diversity in an ensemble of such overfitting networks and the ability of the ensemble to generalise. One of the interesting problems to look into is the way neural networks curve the solution space with each additional layer of hidden neurons and how exactly the number of hidden neurons influences the decision boundaries.
Appendices
Appendix A

Additional Graphical Results

Figure A.1: Influence of number of hidden neurons on single network accuracy. Accuracy decrease and standard deviation increase with higher number of the hidden neurons.
APPENDIX A. ADDITIONAL GRAPHICAL RESULTS

Figure A.2: Aggregated results: number of neurons influence on single network.

Figure A.3: Mostly identical performance on test and training data with different number of the hidden neurons.
Figure A.4: Relatively similar performance of the networks with the different number of epochs while changing the number of the hidden neurons.
Figure A.5: Influence of the number of hidden layers and hidden neurons on single network performance. The networks with higher number of hidden neurons perform worse. Greater amount of hidden layers reduces accuracy.
Figure A.6: Similar performance of single network with different number of the hidden layers on test and training data.
Figure A.7: Comparison of influence of the number of epochs on network performance with different number of layers. Increasing number of epochs increase accuracy for few data sets.
Appendix B

Numerical Experimental Results

B.1 Sampling Influence Strategy
## Appendix B. Numerical Experimental Results

### Table B.1: Sampling strategy influence: Comparison between features and training patterns sampling.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Diversity</th>
<th>Feature sampling</th>
<th>Accuracy</th>
<th>Acc.change</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min (0)</td>
<td>max (0)</td>
<td>mean (0)</td>
<td>min (0)</td>
</tr>
<tr>
<td>blocks</td>
<td>0.03</td>
<td>0.01 (0.01)</td>
<td>89.84</td>
<td>97.55</td>
</tr>
<tr>
<td>chess</td>
<td>0.02</td>
<td>0.12 (0.06)</td>
<td>56.76</td>
<td>99.28</td>
</tr>
<tr>
<td>faults</td>
<td>0.01</td>
<td>0.08 (0.04)</td>
<td>65.32</td>
<td>100</td>
</tr>
<tr>
<td>redwine</td>
<td>0.05</td>
<td>0.11 (0.04)</td>
<td>49.84</td>
<td>61.54</td>
</tr>
<tr>
<td>seeds</td>
<td>0.01</td>
<td>0.05 (0.04)</td>
<td>84.76</td>
<td>95.24</td>
</tr>
<tr>
<td>sonar</td>
<td>0.06</td>
<td>0.11 (0.04)</td>
<td>78.33</td>
<td>86.5</td>
</tr>
<tr>
<td>spambase</td>
<td>0.02</td>
<td>0.05 (0.03)</td>
<td>78.68</td>
<td>95.7</td>
</tr>
<tr>
<td>statlog</td>
<td>0.02</td>
<td>0.04 (0.03)</td>
<td>82.98</td>
<td>86.54</td>
</tr>
<tr>
<td>waveform</td>
<td>0.02</td>
<td>0.11 (0.06)</td>
<td>62.1</td>
<td>87.26</td>
</tr>
<tr>
<td>whitewine</td>
<td>0.05</td>
<td>0.09 (0.03)</td>
<td>44.92</td>
<td>54.9</td>
</tr>
<tr>
<td>yeast</td>
<td>0.03</td>
<td>0.12 (0.04)</td>
<td>31</td>
<td>60.04</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data set</th>
<th>Diversity</th>
<th>Training patterns sampling</th>
<th>Accuracy</th>
<th>Acc.change</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min (0)</td>
<td>max (0)</td>
<td>mean (0)</td>
<td>min (0)</td>
</tr>
<tr>
<td>blocks</td>
<td>0.02</td>
<td>0.01 (0)</td>
<td>96.42</td>
<td>97.5</td>
</tr>
<tr>
<td>chess</td>
<td>0.05</td>
<td>0.01 (0.01)</td>
<td>95.71</td>
<td>99.41</td>
</tr>
<tr>
<td>faults</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>redwine</td>
<td>0.04</td>
<td>0.09 (0.04)</td>
<td>60.04</td>
<td>61.54</td>
</tr>
<tr>
<td>seeds</td>
<td>0.01</td>
<td>0.03 (0.03)</td>
<td>91.43</td>
<td>95.71</td>
</tr>
<tr>
<td>sonar</td>
<td>0.02</td>
<td>0.03 (0.03)</td>
<td>93.35</td>
<td>95.24</td>
</tr>
<tr>
<td>spambase</td>
<td>0.02</td>
<td>0.03 (0.01)</td>
<td>85.84</td>
<td>86.74</td>
</tr>
<tr>
<td>statlog</td>
<td>0.01</td>
<td>0.03 (0.02)</td>
<td>86.58</td>
<td>87.16</td>
</tr>
<tr>
<td>waveform</td>
<td>0.04</td>
<td>0.07 (0.04)</td>
<td>54.02</td>
<td>55.27</td>
</tr>
<tr>
<td>yeast</td>
<td>0.03</td>
<td>0.07 (0.03)</td>
<td>58.69</td>
<td>60.58</td>
</tr>
</tbody>
</table>
## B.2 Random Proportion of Training Patterns

<table>
<thead>
<tr>
<th>Data set</th>
<th>nosampling Diversity</th>
<th>nosampling Accuracy</th>
<th>nosampling Acc.change</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
</tr>
<tr>
<td>blocks</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01 (0)</td>
</tr>
<tr>
<td>chess</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01 (0)</td>
</tr>
<tr>
<td>faults</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00 (0)</td>
</tr>
<tr>
<td>redwine</td>
<td>0.09</td>
<td>0.10</td>
<td>0.09 (0)</td>
</tr>
<tr>
<td>seeds</td>
<td>0.03</td>
<td>0.04</td>
<td>0.03 (0)</td>
</tr>
<tr>
<td>sonar</td>
<td>0.11</td>
<td>0.12</td>
<td>0.12 (0)</td>
</tr>
<tr>
<td>spambase</td>
<td>0.03</td>
<td>0.03</td>
<td>0.03 (0)</td>
</tr>
<tr>
<td>statlog</td>
<td>0.03</td>
<td>0.03</td>
<td>0.03 (0)</td>
</tr>
<tr>
<td>waveform</td>
<td>0.03</td>
<td>0.03</td>
<td>0.03 (0)</td>
</tr>
<tr>
<td>whitewine</td>
<td>0.07</td>
<td>0.07</td>
<td>0.07 (0)</td>
</tr>
<tr>
<td>yeast</td>
<td>0.06</td>
<td>0.07</td>
<td>0.07 (0)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data set</th>
<th>sampling Diversity</th>
<th>sampling Accuracy</th>
<th>sampling Acc.change</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
</tr>
<tr>
<td>blocks</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01 (0)</td>
</tr>
<tr>
<td>chess</td>
<td>0.06</td>
<td>0.07</td>
<td>0.06 (0)</td>
</tr>
<tr>
<td>faults</td>
<td>0.04</td>
<td>0.04</td>
<td>0.04 (0)</td>
</tr>
<tr>
<td>redwine</td>
<td>0.10</td>
<td>0.11</td>
<td>0.10 (0)</td>
</tr>
<tr>
<td>seeds</td>
<td>0.03</td>
<td>0.04</td>
<td>0.04 (0)</td>
</tr>
<tr>
<td>sonar</td>
<td>0.15</td>
<td>0.16</td>
<td>0.15 (0.01)</td>
</tr>
<tr>
<td>spambase</td>
<td>0.04</td>
<td>0.05</td>
<td>0.05 (0)</td>
</tr>
<tr>
<td>statlog</td>
<td>0.03</td>
<td>0.03</td>
<td>0.03 (0)</td>
</tr>
<tr>
<td>waveform</td>
<td>0.06</td>
<td>0.07</td>
<td>0.07 (0)</td>
</tr>
<tr>
<td>whitewine</td>
<td>0.09</td>
<td>0.10</td>
<td>0.10 (0)</td>
</tr>
<tr>
<td>yeast</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10 (0)</td>
</tr>
</tbody>
</table>

Table B.2: Random training patterns proportion: Comparison between sampling and nosampling approaches.
### B.3 Random Proportion of Features

<table>
<thead>
<tr>
<th>Data set</th>
<th>Diversity</th>
<th>nosampling</th>
<th></th>
<th></th>
<th>Accuracy</th>
<th>Acc.change</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td></td>
</tr>
<tr>
<td>blocks</td>
<td>0.01</td>
<td>0.02</td>
<td>0.01 (0)</td>
<td>96.99</td>
<td>97.13</td>
<td>97.09 (0.06)</td>
<td>-0.31</td>
<td>-0.17</td>
<td>-0.21 (0.06)</td>
<td></td>
</tr>
<tr>
<td>chess</td>
<td>0.12</td>
<td>0.14</td>
<td>0.13 (0.01)</td>
<td>95.18</td>
<td>96.34</td>
<td>95.59 (0.45)</td>
<td>-3.91</td>
<td>-2.75</td>
<td>-3.5 (0.45)</td>
<td></td>
</tr>
<tr>
<td>faults</td>
<td>0.09</td>
<td>0.11</td>
<td>0.1 (0.01)</td>
<td>93.61</td>
<td>96.7</td>
<td>95.12 (1.1)</td>
<td>-6.39</td>
<td>-3.3</td>
<td>-4.88 (1.1)</td>
<td></td>
</tr>
<tr>
<td>redwine</td>
<td>0.11</td>
<td>0.12</td>
<td>0.12 (0.01)</td>
<td>59.1</td>
<td>60.16</td>
<td>59.72 (0.39)</td>
<td>-1.81</td>
<td>-0.75</td>
<td>-1.19 (0.39)</td>
<td></td>
</tr>
<tr>
<td>seeds</td>
<td>0.03</td>
<td>0.06</td>
<td>0.04 (0.01)</td>
<td>92.38</td>
<td>94.76</td>
<td>94 (0.99)</td>
<td>-3.33</td>
<td>-0.95</td>
<td>-1.71 (0.99)</td>
<td></td>
</tr>
<tr>
<td>sonar</td>
<td>0.1</td>
<td>0.12</td>
<td>0.1 (0.01)</td>
<td>82.23</td>
<td>85.13</td>
<td>83.18 (1.13)</td>
<td>-1.43</td>
<td>1.47</td>
<td>-0.48 (1.13)</td>
<td></td>
</tr>
<tr>
<td>spambase</td>
<td>0.04</td>
<td>0.05</td>
<td>0.04 (0)</td>
<td>95.18</td>
<td>95.5</td>
<td>95.4 (0.13)</td>
<td>0.68</td>
<td>1</td>
<td>0.9 (0.13)</td>
<td></td>
</tr>
<tr>
<td>statlog</td>
<td>0.03</td>
<td>0.04</td>
<td>0.03 (0)</td>
<td>86.07</td>
<td>86.47</td>
<td>86.3 (0.19)</td>
<td>-0.81</td>
<td>-0.41</td>
<td>-0.58 (0.19)</td>
<td></td>
</tr>
<tr>
<td>waveform</td>
<td>0.09</td>
<td>0.12</td>
<td>0.1 (0.01)</td>
<td>86.46</td>
<td>86.8</td>
<td>86.61 (0.15)</td>
<td>-0.2</td>
<td>0.14</td>
<td>-0.05 (0.15)</td>
<td></td>
</tr>
<tr>
<td>whitewine</td>
<td>0.1</td>
<td>0.11</td>
<td>0.11 (0)</td>
<td>53.04</td>
<td>53.72</td>
<td>53.44 (0.34)</td>
<td>-1.76</td>
<td>-1.08</td>
<td>-1.36 (0.34)</td>
<td></td>
</tr>
<tr>
<td>yeast</td>
<td>0.17</td>
<td>0.17</td>
<td>0.17 (0)</td>
<td>53.37</td>
<td>54.85</td>
<td>54.23 (0.55)</td>
<td>-6.87</td>
<td>-5.39</td>
<td>-6.01 (0.55)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data set</th>
<th>Diversity</th>
<th>sampling</th>
<th></th>
<th></th>
<th>Accuracy</th>
<th>Acc.change</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td></td>
</tr>
<tr>
<td>blocks</td>
<td>0.01</td>
<td>0.02</td>
<td>0.02 (0)</td>
<td>96.91</td>
<td>97.11</td>
<td>97.03 (0.08)</td>
<td>-0.39</td>
<td>-0.19</td>
<td>-0.27 (0.08)</td>
<td></td>
</tr>
<tr>
<td>chess</td>
<td>0.13</td>
<td>0.14</td>
<td>0.13 (0)</td>
<td>94.24</td>
<td>97.68</td>
<td>95.85 (1.31)</td>
<td>-4.85</td>
<td>-1.41</td>
<td>-3.24 (1.31)</td>
<td></td>
</tr>
<tr>
<td>faults</td>
<td>0.1</td>
<td>0.11</td>
<td>0.1 (0)</td>
<td>95.72</td>
<td>97.99</td>
<td>96.63 (0.84)</td>
<td>-4.28</td>
<td>-2.01</td>
<td>-3.37 (0.84)</td>
<td></td>
</tr>
<tr>
<td>redwine</td>
<td>0.14</td>
<td>0.14</td>
<td>0.14 (0)</td>
<td>58.48</td>
<td>60.35</td>
<td>59.67 (0.72)</td>
<td>-2.43</td>
<td>-0.56</td>
<td>-1.24 (0.72)</td>
<td></td>
</tr>
<tr>
<td>seeds</td>
<td>0.04</td>
<td>0.06</td>
<td>0.05 (0.01)</td>
<td>91.9</td>
<td>95.24</td>
<td>93.71 (1.24)</td>
<td>-3.81</td>
<td>-0.47</td>
<td>-2 (1.24)</td>
<td></td>
</tr>
<tr>
<td>sonar</td>
<td>0.12</td>
<td>0.14</td>
<td>0.13 (0)</td>
<td>79.81</td>
<td>84.12</td>
<td>82.29 (1.9)</td>
<td>-3.85</td>
<td>0.46</td>
<td>-1.37 (1.9)</td>
<td></td>
</tr>
<tr>
<td>spambase</td>
<td>0.04</td>
<td>0.06</td>
<td>0.05 (0)</td>
<td>94.87</td>
<td>95.44</td>
<td>95.11 (0.21)</td>
<td>0.37</td>
<td>0.94</td>
<td>0.61 (0.21)</td>
<td></td>
</tr>
<tr>
<td>statlog</td>
<td>0.03</td>
<td>0.04</td>
<td>0.03 (0)</td>
<td>85.95</td>
<td>86.54</td>
<td>86.29 (0.22)</td>
<td>-0.93</td>
<td>-0.34</td>
<td>-0.59 (0.22)</td>
<td></td>
</tr>
<tr>
<td>waveform</td>
<td>0.09</td>
<td>0.11</td>
<td>0.11 (0.01)</td>
<td>86.42</td>
<td>86.8</td>
<td>86.6 (0.17)</td>
<td>-0.24</td>
<td>0.14</td>
<td>-0.06 (0.17)</td>
<td></td>
</tr>
<tr>
<td>whitewine</td>
<td>0.11</td>
<td>0.12</td>
<td>0.11 (0)</td>
<td>53.1</td>
<td>53.9</td>
<td>53.51 (0.3)</td>
<td>-1.7</td>
<td>-0.9</td>
<td>-1.29 (0.3)</td>
<td></td>
</tr>
<tr>
<td>yeast</td>
<td>0.18</td>
<td>0.18</td>
<td>0.18 (0)</td>
<td>53.37</td>
<td>56.4</td>
<td>55.3 (1.15)</td>
<td>-6.87</td>
<td>-3.84</td>
<td>-4.94 (1.15)</td>
<td></td>
</tr>
</tbody>
</table>

Table B.3: Random feature proportion: Comparison between sampling and nosampling approaches.
## B.4 Random Number of Hidden Neurons

<table>
<thead>
<tr>
<th>Data set</th>
<th>nosampling</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Diversity</td>
<td>Accuracy</td>
<td>Acc.change</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td></td>
</tr>
<tr>
<td>blocks</td>
<td>0</td>
<td>0.01</td>
<td>0.01 (0)</td>
<td>96.89</td>
<td>97.15</td>
<td>97.09 (0.1)</td>
<td>-0.41</td>
<td>-0.15</td>
<td>-0.21 (0.1)</td>
<td></td>
</tr>
<tr>
<td>chess</td>
<td>0</td>
<td>0</td>
<td>0 (0)</td>
<td>99.03</td>
<td>99.5</td>
<td>99.24 (0.15)</td>
<td>-0.06</td>
<td>0.41</td>
<td>0.15 (0.15)</td>
<td></td>
</tr>
<tr>
<td>faults</td>
<td>100</td>
<td>100</td>
<td>100 (0)</td>
<td>0</td>
<td>0</td>
<td>0 (0)</td>
<td>0</td>
<td>0</td>
<td>0 (0)</td>
<td></td>
</tr>
<tr>
<td>redwine</td>
<td>0.04</td>
<td>0.03 (0.01)</td>
<td>60.16</td>
<td>61.16</td>
<td>60.51 (0.3)</td>
<td>-0.75</td>
<td>0.25</td>
<td>-0.4 (0.3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>seeds</td>
<td>0.02</td>
<td>0.01 (0.01)</td>
<td>94.29</td>
<td>96.19</td>
<td>95.36 (0.55)</td>
<td>-1.42</td>
<td>0.48</td>
<td>-0.35 (0.55)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sonar</td>
<td>0.06</td>
<td>0.05 (0.02)</td>
<td>77.89</td>
<td>85.06</td>
<td>81.97 (2.11)</td>
<td>-5.77</td>
<td>1.4</td>
<td>-1.69 (2.11)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>spambase</td>
<td>0.02</td>
<td>0.01 (0)</td>
<td>94.11</td>
<td>95.48</td>
<td>95.03 (0.39)</td>
<td>-0.39</td>
<td>0.98</td>
<td>0.53 (0.39)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>statlog</td>
<td>0.02</td>
<td>0.02 (0.01)</td>
<td>85.84</td>
<td>86.79</td>
<td>86.34 (0.27)</td>
<td>-1.04</td>
<td>-0.09</td>
<td>-0.54 (0.27)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>waveform</td>
<td>0.01</td>
<td>0.01 (0)</td>
<td>86.6</td>
<td>87.02</td>
<td>86.89 (0.13)</td>
<td>-0.06</td>
<td>0.36</td>
<td>0.23 (0.13)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>whitewine</td>
<td>0.03</td>
<td>0.03 (0.01)</td>
<td>54.21</td>
<td>54.96</td>
<td>54.63 (0.24)</td>
<td>-0.59</td>
<td>0.16</td>
<td>-0.17 (0.24)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>yeast</td>
<td>0.05</td>
<td>0.03 (0.02)</td>
<td>58.7</td>
<td>60.11</td>
<td>59.31 (0.47)</td>
<td>-1.54</td>
<td>-0.13</td>
<td>-0.93 (0.47)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data set</th>
<th>sampling</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Diversity</td>
<td>Accuracy</td>
<td>Acc.change</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
</tr>
<tr>
<td>blocks</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01 (0)</td>
<td>97.08</td>
<td>97.33</td>
<td>97.22 (0.08)</td>
<td>-0.22</td>
<td>0.03</td>
<td>-0.07 (0.08)</td>
</tr>
<tr>
<td>chess</td>
<td>0.06</td>
<td>0.07</td>
<td>0.06 (0)</td>
<td>98.97</td>
<td>99.22</td>
<td>99.08 (0.1)</td>
<td>-0.12</td>
<td>0.13</td>
<td>-0.01 (0.1)</td>
</tr>
<tr>
<td>faults</td>
<td>0.04</td>
<td>0.05</td>
<td>0.05 (0)</td>
<td>99.95</td>
<td>100</td>
<td>99.98 (0.03)</td>
<td>-0.05</td>
<td>0</td>
<td>-0.02 (0.03)</td>
</tr>
<tr>
<td>redwine</td>
<td>0.09</td>
<td>0.1</td>
<td>0.1 (0)</td>
<td>60.29</td>
<td>61.6</td>
<td>60.95 (0.38)</td>
<td>-0.62</td>
<td>0.69</td>
<td>0.04 (0.38)</td>
</tr>
<tr>
<td>seeds</td>
<td>0.03</td>
<td>0.03</td>
<td>0.03 (0)</td>
<td>93.81</td>
<td>95.24</td>
<td>94.35 (0.58)</td>
<td>-1.9</td>
<td>-0.47</td>
<td>-1.36 (0.58)</td>
</tr>
<tr>
<td>sonar</td>
<td>0.13</td>
<td>0.14</td>
<td>0.14 (0)</td>
<td>81.25</td>
<td>86.6</td>
<td>82.76 (1.52)</td>
<td>-2.41</td>
<td>2.94</td>
<td>-0.9 (1.52)</td>
</tr>
<tr>
<td>spambase</td>
<td>0.04</td>
<td>0.04</td>
<td>0.04 (0)</td>
<td>95.09</td>
<td>95.57</td>
<td>95.36 (0.15)</td>
<td>0.59</td>
<td>1.07</td>
<td>0.86 (0.15)</td>
</tr>
<tr>
<td>statlog</td>
<td>0.03</td>
<td>0.03</td>
<td>0.03 (0)</td>
<td>86.22</td>
<td>86.9</td>
<td>86.6 (0.21)</td>
<td>-0.66</td>
<td>0.02</td>
<td>-0.28 (0.21)</td>
</tr>
<tr>
<td>waveform</td>
<td>0.06</td>
<td>0.06</td>
<td>0.06 (0)</td>
<td>86.48</td>
<td>87.22</td>
<td>86.9 (0.22)</td>
<td>-0.18</td>
<td>0.56</td>
<td>0.24 (0.22)</td>
</tr>
<tr>
<td>whitewine</td>
<td>0.08</td>
<td>0.09</td>
<td>0.08 (0)</td>
<td>54.06</td>
<td>54.82</td>
<td>54.5 (0.21)</td>
<td>-0.74</td>
<td>0.02</td>
<td>-0.3 (0.21)</td>
</tr>
<tr>
<td>yeast</td>
<td>0.09</td>
<td>0.1</td>
<td>0.09 (0)</td>
<td>59.16</td>
<td>60.11</td>
<td>59.63 (0.29)</td>
<td>-1.08</td>
<td>-0.13</td>
<td>-0.61 (0.29)</td>
</tr>
</tbody>
</table>

Table B.4: Random number of hidden neurons: Comparison between sampling and nosampling approaches.
### B.5 Random Number of Hidden Layers

| Data set     | nosampling |                   |                   |                   |                   |                   |                   |                   |
|--------------|------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|
|              | Diversity  | Accuracy          | Acc.change        |                   |                   |                   |                   |
|              | min        | max               | mean              | min               | max               | mean              | min               |
|              |            |                   |                   |                   |                   |                   |                   |                   |
| blocks       | 0          | 0.02              | 0.01 (0.01)       | 96.38             | 97.66             | 97.21 (0.41)     | -0.92             |
| chess        | 0          | 0.02              | 0.01 (0.01)       | 98.62             | 99.47             | 99.2 (0.26)      | -0.47             |
| faults       | 0          | 0.02              | 0.01 (0.01)       | 100               | 100               | 100 (0)          | 0                 |
| redwine      | 0.04       | 0.07              | 0.06 (0.01)       | 60.41             | 61.16             | 60.7 (0.22)      | -0.5              |
| seeds        | 0.01       | 0.1               | 0.04 (0.03)       | 92.86             | 96.19             | 93.99 (1.22)     | -2.85             |
| sonar        | 0.06       | 0.09              | 0.08 (0.02)       | 80.74             | 85.61             | 83.54 (1.59)     | -2.92             |
| spambase     | 0.02       | 0.02              | 0.02 (0)          | 94.41             | 95.41             | 95.11 (0.32)     | -0.09             |
| statlog      | 0.02       | 0.05              | 0.03 (0.01)       | 86.65             | 88.86             | 88.18 (0.69)     | -0.23             |
| waveform     | 0.01       | 0.07              | 0.03 (0.02)       | 86.9              | 87.18             | 87.06 (0.11)     | 0.24              |
| whitewine    | 0.03       | 0.07              | 0.05 (0.01)       | 54.78             | 55.31             | 55.09 (0.17)     | -0.02             |
| yeast        | 0.03       | 0.14              | 0.08 (0.04)       | 57.01             | 59.84             | 58.48 (0.99)     | -3.23             |

<table>
<thead>
<tr>
<th>Data set</th>
<th>sampling</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Diversity</td>
<td>Accuracy</td>
<td>Acc.change</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>blocks</td>
<td>0.01</td>
<td>0.03</td>
<td>0.02 (0.01)</td>
<td>95.38</td>
<td>97.35</td>
<td>96.53 (0.73)</td>
<td>-1.92</td>
</tr>
<tr>
<td>chess</td>
<td>0.06</td>
<td>0.07</td>
<td>0.06 (0)</td>
<td>98.56</td>
<td>99.22</td>
<td>98.91 (0.27)</td>
<td>-0.53</td>
</tr>
<tr>
<td>faults</td>
<td>0.05</td>
<td>0.07</td>
<td>0.06 (0.01)</td>
<td>99.95</td>
<td>100</td>
<td>99.99 (0.02)</td>
<td>-0.05</td>
</tr>
<tr>
<td>redwine</td>
<td>0.1</td>
<td>0.11</td>
<td>0.1 (0)</td>
<td>58.6</td>
<td>61.29</td>
<td>60.27 (0.89)</td>
<td>-2.31</td>
</tr>
<tr>
<td>seeds</td>
<td>0.03</td>
<td>0.13</td>
<td>0.06 (0.04)</td>
<td>91.43</td>
<td>93.33</td>
<td>92.2 (0.72)</td>
<td>-4.28</td>
</tr>
<tr>
<td>sonar</td>
<td>0.13</td>
<td>0.15</td>
<td>0.14 (0.01)</td>
<td>79.9</td>
<td>85.12</td>
<td>82.59 (1.62)</td>
<td>-3.76</td>
</tr>
<tr>
<td>spambase</td>
<td>0.04</td>
<td>0.05</td>
<td>0.04 (0)</td>
<td>95.07</td>
<td>95.52</td>
<td>95.31 (0.17)</td>
<td>0.57</td>
</tr>
<tr>
<td>statlog</td>
<td>0.03</td>
<td>0.06</td>
<td>0.04 (0.01)</td>
<td>86.4</td>
<td>88.7</td>
<td>87.95 (0.77)</td>
<td>-0.48</td>
</tr>
<tr>
<td>waveform</td>
<td>0.06</td>
<td>0.12</td>
<td>0.08 (0.02)</td>
<td>86.54</td>
<td>87.22</td>
<td>86.86 (0.24)</td>
<td>-0.12</td>
</tr>
<tr>
<td>whitewine</td>
<td>0.09</td>
<td>0.11</td>
<td>0.1 (0.01)</td>
<td>54.23</td>
<td>54.8</td>
<td>54.56 (0.22)</td>
<td>-0.57</td>
</tr>
<tr>
<td>yeast</td>
<td>0.1</td>
<td>0.17</td>
<td>0.13 (0.03)</td>
<td>56</td>
<td>59.91</td>
<td>58.34 (1.39)</td>
<td>-4.24</td>
</tr>
</tbody>
</table>

Table B.5: Random number of hidden layers: Comparison between sampling and nosampling approaches.
### Table B.6: High number of hidden neurons and hidden layers: Comparison between two approaches.

<table>
<thead>
<tr>
<th>Data set</th>
<th>hidden neurons</th>
<th>Diversity</th>
<th>Accuracy</th>
<th>Acc.change</th>
<th>hidden layers</th>
<th>Diversity</th>
<th>Accuracy</th>
<th>Acc.change</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
</tr>
<tr>
<td>blocks</td>
<td></td>
<td>0.01</td>
<td>0.01</td>
<td>0.01 (0)</td>
<td>97.09</td>
<td>97.73</td>
<td>97.49 (0.17)</td>
<td>-0.21</td>
</tr>
<tr>
<td>chess</td>
<td></td>
<td>0.13</td>
<td>0.14</td>
<td>0.13 (0)</td>
<td>94.62</td>
<td>96.75</td>
<td>95.81 (0.57)</td>
<td>-4.47</td>
</tr>
<tr>
<td>faults</td>
<td></td>
<td>0.08</td>
<td>0.1</td>
<td>0.09 (0)</td>
<td>96.08</td>
<td>98.66</td>
<td>97.29 (0.61)</td>
<td>-3.92</td>
</tr>
<tr>
<td>redwine</td>
<td></td>
<td>0.13</td>
<td>0.15</td>
<td>0.14 (0)</td>
<td>59.79</td>
<td>62.54</td>
<td>61.24 (0.84)</td>
<td>-1.12</td>
</tr>
<tr>
<td>seeds</td>
<td></td>
<td>0.04</td>
<td>0.05</td>
<td>0.04 (0)</td>
<td>91.19</td>
<td>94.76</td>
<td>93.53 (0.86)</td>
<td>-3.81</td>
</tr>
<tr>
<td>sonar</td>
<td></td>
<td>0.12</td>
<td>0.13</td>
<td>0.12 (0)</td>
<td>80.26</td>
<td>85.64</td>
<td>83.21 (1.57)</td>
<td>-3.4</td>
</tr>
<tr>
<td>spambase</td>
<td></td>
<td>0.05</td>
<td>0.05</td>
<td>0.05 (0)</td>
<td>94.89</td>
<td>95.65</td>
<td>95.33 (0.18)</td>
<td>0.39</td>
</tr>
<tr>
<td>statlog</td>
<td></td>
<td>0.03</td>
<td>0.03</td>
<td>0.03 (0)</td>
<td>86.47</td>
<td>89.09</td>
<td>88.52 (0.67)</td>
<td>-0.41</td>
</tr>
<tr>
<td>waveform</td>
<td></td>
<td>0.09</td>
<td>0.09</td>
<td>0.09 (0)</td>
<td>86.26</td>
<td>87.16</td>
<td>86.6 (0.23)</td>
<td>-0.4</td>
</tr>
<tr>
<td>whitewine</td>
<td></td>
<td>0.11</td>
<td>0.12</td>
<td>0.12 (0)</td>
<td>54.51</td>
<td>55.7</td>
<td>55.2 (0.33)</td>
<td>-0.29</td>
</tr>
<tr>
<td>yeast</td>
<td></td>
<td>0.17</td>
<td>0.19</td>
<td>0.18 (0)</td>
<td>54.45</td>
<td>57.95</td>
<td>56.42 (1.06)</td>
<td>-5.79</td>
</tr>
</tbody>
</table>
### B.7 Random Learning Rate

<table>
<thead>
<tr>
<th>Data set</th>
<th>Diversity</th>
<th>nosampling</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>Acc.change</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
</tr>
<tr>
<td>blocks</td>
<td>0</td>
<td>0.01</td>
<td>0.01 (0)</td>
<td>97.06</td>
<td>97.24</td>
<td>97.14 (0.07)</td>
<td>-0.24</td>
<td>-0.06</td>
<td>-0.16 (0.07)</td>
</tr>
<tr>
<td>chess</td>
<td>0</td>
<td>0.01</td>
<td>0.01 (0)</td>
<td>99.03</td>
<td>99.25</td>
<td>99.17 (0.09)</td>
<td>-0.06</td>
<td>0.16</td>
<td>0.08 (0.09)</td>
</tr>
<tr>
<td>faults</td>
<td>0</td>
<td>0</td>
<td>0 (0)</td>
<td>100</td>
<td>100</td>
<td>100 (0)</td>
<td>0</td>
<td>0</td>
<td>0 (0)</td>
</tr>
<tr>
<td>redwine</td>
<td>0.03</td>
<td>0.05</td>
<td>0.04 (0.01)</td>
<td>60.79</td>
<td>61.66</td>
<td>61.11 (0.32)</td>
<td>-0.12</td>
<td>0.75</td>
<td>0.2 (0.32)</td>
</tr>
<tr>
<td>seeds</td>
<td>0.01</td>
<td>0.02</td>
<td>0.01 (0)</td>
<td>93.33</td>
<td>95.24</td>
<td>94.67 (0.78)</td>
<td>-2.38</td>
<td>-0.47</td>
<td>-1.04 (0.78)</td>
</tr>
<tr>
<td>sonar</td>
<td>0.05</td>
<td>0.14</td>
<td>0.1 (0.04)</td>
<td>69.67</td>
<td>83.16</td>
<td>78.16 (5.17)</td>
<td>-13.99</td>
<td>-0.5</td>
<td>-5.5 (5.17)</td>
</tr>
<tr>
<td>spambase</td>
<td>0.02</td>
<td>0.13</td>
<td>0.07 (0.05)</td>
<td>60.6</td>
<td>95.2</td>
<td>88.05 (15.35)</td>
<td>-33.9</td>
<td>0.7</td>
<td>-6.45 (15.35)</td>
</tr>
<tr>
<td>statlog</td>
<td>0.02</td>
<td>0.18</td>
<td>0.11 (0.06)</td>
<td>76.32</td>
<td>86.34</td>
<td>83.99 (4.3)</td>
<td>-10.56</td>
<td>-0.54</td>
<td>-2.89 (4.3)</td>
</tr>
<tr>
<td>waveform</td>
<td>0.01</td>
<td>0.04</td>
<td>0.02 (0.01)</td>
<td>86.56</td>
<td>87.18</td>
<td>86.85 (0.25)</td>
<td>-0.1</td>
<td>0.52</td>
<td>0.19 (0.25)</td>
</tr>
<tr>
<td>whitewine</td>
<td>0.04</td>
<td>0.05</td>
<td>0.04 (0.01)</td>
<td>54.72</td>
<td>54.98</td>
<td>54.85 (0.11)</td>
<td>-0.08</td>
<td>0.18</td>
<td>0.05 (0.11)</td>
</tr>
<tr>
<td>yeast</td>
<td>0.03</td>
<td>0.11</td>
<td>0.05 (0.03)</td>
<td>58.89</td>
<td>59.5</td>
<td>59.27 (0.29)</td>
<td>-1.35</td>
<td>-0.74</td>
<td>-0.97 (0.29)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data set</th>
<th>Diversity</th>
<th>sampling</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>Acc.change</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
<td>mean</td>
</tr>
<tr>
<td>blocks</td>
<td>0.01</td>
<td>0.02</td>
<td>0.01 (0)</td>
<td>97.09</td>
<td>97.22</td>
<td>97.17 (0.06)</td>
<td>-0.21</td>
<td>-0.08</td>
<td>-0.13 (0.06)</td>
</tr>
<tr>
<td>chess</td>
<td>0.06</td>
<td>0.19</td>
<td>0.11 (0.05)</td>
<td>98.15</td>
<td>98.94</td>
<td>98.64 (0.34)</td>
<td>-0.94</td>
<td>-0.15</td>
<td>-0.45 (0.34)</td>
</tr>
<tr>
<td>faults</td>
<td>0.04</td>
<td>0.06</td>
<td>0.05 (0)</td>
<td>99.95</td>
<td>100</td>
<td>99.99 (0.02)</td>
<td>-0.05</td>
<td>0</td>
<td>-0.01 (0.02)</td>
</tr>
<tr>
<td>redwine</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1 (0)</td>
<td>59.91</td>
<td>61.35</td>
<td>60.47 (0.54)</td>
<td>-1</td>
<td>0.44</td>
<td>-0.44 (0.54)</td>
</tr>
<tr>
<td>seeds</td>
<td>0.02</td>
<td>0.06</td>
<td>0.04 (0.01)</td>
<td>92.86</td>
<td>94.29</td>
<td>93.62 (0.54)</td>
<td>-2.85</td>
<td>-1.42</td>
<td>-2.09 (0.54)</td>
</tr>
<tr>
<td>sonar</td>
<td>0.16</td>
<td>0.26</td>
<td>0.21 (0.04)</td>
<td>81.2</td>
<td>84.65</td>
<td>82.2 (1.43)</td>
<td>-2.46</td>
<td>0.99</td>
<td>-1.46 (1.43)</td>
</tr>
<tr>
<td>spambase</td>
<td>0.04</td>
<td>0.26</td>
<td>0.17 (0.09)</td>
<td>95.07</td>
<td>95.59</td>
<td>95.33 (0.23)</td>
<td>0.57</td>
<td>1.09</td>
<td>0.83 (0.23)</td>
</tr>
<tr>
<td>statlog</td>
<td>0.08</td>
<td>0.27</td>
<td>0.17 (0.09)</td>
<td>86.49</td>
<td>87.51</td>
<td>86.93 (0.38)</td>
<td>-0.39</td>
<td>0.63</td>
<td>0.05 (0.38)</td>
</tr>
<tr>
<td>waveform</td>
<td>0.06</td>
<td>0.09</td>
<td>0.07 (0.01)</td>
<td>86.7</td>
<td>86.94</td>
<td>86.87 (0.1)</td>
<td>0.04</td>
<td>0.28</td>
<td>0.21 (0.1)</td>
</tr>
<tr>
<td>whitewine</td>
<td>0.08</td>
<td>0.09</td>
<td>0.09 (0)</td>
<td>54.33</td>
<td>54.68</td>
<td>54.48 (0.13)</td>
<td>-0.47</td>
<td>-0.12</td>
<td>-0.32 (0.13)</td>
</tr>
<tr>
<td>yeast</td>
<td>0.09</td>
<td>0.11</td>
<td>0.1 (0.01)</td>
<td>59.03</td>
<td>60.44</td>
<td>59.61 (0.61)</td>
<td>-1.21</td>
<td>0.2</td>
<td>-0.63 (0.61)</td>
</tr>
</tbody>
</table>

Table B.7: Random learning rate value: Comparison between sampling and nosampling approaches.
## B.8 Random Connections

<table>
<thead>
<tr>
<th>Data set</th>
<th>nosampling</th>
<th>sampling</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Diversity</td>
<td>Accuracy</td>
</tr>
<tr>
<td></td>
<td>min</td>
<td>max</td>
</tr>
<tr>
<td>blocks</td>
<td>0</td>
<td>0.02</td>
</tr>
<tr>
<td>chess</td>
<td>0</td>
<td>0.1</td>
</tr>
<tr>
<td>faults</td>
<td>0</td>
<td>0.11</td>
</tr>
<tr>
<td>redwine</td>
<td>0</td>
<td>0.12</td>
</tr>
<tr>
<td>seeds</td>
<td>0</td>
<td>0.11</td>
</tr>
<tr>
<td>sonar</td>
<td>0</td>
<td>0.08</td>
</tr>
<tr>
<td>spambase</td>
<td>0</td>
<td>0.02</td>
</tr>
<tr>
<td>statlog</td>
<td>0</td>
<td>0.03</td>
</tr>
<tr>
<td>waveform</td>
<td>0</td>
<td>0.11</td>
</tr>
<tr>
<td>whitewine</td>
<td>0</td>
<td>0.08</td>
</tr>
<tr>
<td>yeast</td>
<td>0</td>
<td>0.17</td>
</tr>
</tbody>
</table>

Table B.8: Random Connections: Comparison between sampling and nosampling approaches
## B.9 Random Number of Epochs

<table>
<thead>
<tr>
<th>Dataset</th>
<th>nosampling</th>
<th></th>
<th>sampling</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Diversity</td>
<td>Accuracy</td>
<td>Diversity</td>
<td>Accuracy</td>
</tr>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
</tr>
<tr>
<td>blocks</td>
<td>0</td>
<td>0</td>
<td>0.00 (0)</td>
<td>97.02</td>
</tr>
<tr>
<td>chess</td>
<td>0</td>
<td>0</td>
<td>0.00 (0)</td>
<td>99.09</td>
</tr>
<tr>
<td>faults</td>
<td>0</td>
<td>0</td>
<td>0.00 (0)</td>
<td>99.95</td>
</tr>
<tr>
<td>redwine</td>
<td>0</td>
<td>0.02</td>
<td>0.01 (0.01)</td>
<td>59.35</td>
</tr>
<tr>
<td>seeds</td>
<td>0</td>
<td>0.01</td>
<td>0.00 (0)</td>
<td>92.86</td>
</tr>
<tr>
<td>sonar</td>
<td>0</td>
<td>0.02</td>
<td>0.01 (0.01)</td>
<td>76.47</td>
</tr>
<tr>
<td>spambase</td>
<td>0</td>
<td>0.01</td>
<td>0.01 (0)</td>
<td>94.28</td>
</tr>
<tr>
<td>statlog</td>
<td>0</td>
<td>0.02</td>
<td>0.01 (0.01)</td>
<td>85.23</td>
</tr>
<tr>
<td>waveform</td>
<td>0</td>
<td>0.01</td>
<td>0.01 (0)</td>
<td>86.64</td>
</tr>
<tr>
<td>whitewine</td>
<td>0</td>
<td>0.02</td>
<td>0.01 (0.01)</td>
<td>54.10</td>
</tr>
<tr>
<td>yeast</td>
<td>0</td>
<td>0.02</td>
<td>0.01 (0.01)</td>
<td>57.88</td>
</tr>
</tbody>
</table>

Table B.9: Random number of epochs: Comparison between sampling and nosampling approaches.
### B.10 Combination of Randomized Parameters

<table>
<thead>
<tr>
<th>Data set</th>
<th>Diversity</th>
<th>layers</th>
<th>Accuracy</th>
<th>Acc.change</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
</tr>
<tr>
<td>blocks</td>
<td>0.01</td>
<td>0.03</td>
<td>0.02 (0)</td>
<td>96.51</td>
</tr>
<tr>
<td>chess</td>
<td>0.14</td>
<td>0.15</td>
<td>0.14 (0.01)</td>
<td>95.34</td>
</tr>
<tr>
<td>faults</td>
<td>0.1</td>
<td>0.16</td>
<td>0.12 (0.02)</td>
<td>91.81</td>
</tr>
<tr>
<td>redwine</td>
<td>0.14</td>
<td>0.19</td>
<td>0.18 (0.02)</td>
<td>59.04</td>
</tr>
<tr>
<td>seeds</td>
<td>0.05</td>
<td>0.17</td>
<td>0.1 (0.05)</td>
<td>89.05</td>
</tr>
<tr>
<td>sonar</td>
<td>0.14</td>
<td>0.2</td>
<td>0.16 (0.02)</td>
<td>77.39</td>
</tr>
<tr>
<td>spambase</td>
<td>0.05</td>
<td>0.08</td>
<td>0.06 (0.01)</td>
<td>94.39</td>
</tr>
<tr>
<td>statlog</td>
<td>0.04</td>
<td>0.08</td>
<td>0.06 (0.01)</td>
<td>86.38</td>
</tr>
<tr>
<td>waveform</td>
<td>0.09</td>
<td>0.15</td>
<td>0.12 (0.02)</td>
<td>86.22</td>
</tr>
<tr>
<td>whitewine</td>
<td>0.12</td>
<td>0.17</td>
<td>0.15 (0.02)</td>
<td>54.33</td>
</tr>
<tr>
<td>yeast</td>
<td>0.18</td>
<td>0.21</td>
<td>0.2 (0.01)</td>
<td>52.09</td>
</tr>
</tbody>
</table>

### Additional Table: Combination of randomized parameters: feature proportion $F_p$, training patterns proportion $D_p$ and number of hidden neurons $H_n$, comparison between additionally randomized proportion of random connections $C_n$ and number of the hidden layers $H_l$.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Diversity</th>
<th>connections</th>
<th>Diversity</th>
<th>Accuracy</th>
<th>Acc.change</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>mean</td>
<td>min</td>
<td>max</td>
</tr>
<tr>
<td>blocks</td>
<td>0.01</td>
<td>0.02</td>
<td>0.02 (0)</td>
<td>97.02</td>
<td>97.39</td>
</tr>
<tr>
<td>chess</td>
<td>0.14</td>
<td>0.14</td>
<td>0.14 (0)</td>
<td>95.4</td>
<td>96.84</td>
</tr>
<tr>
<td>faults</td>
<td>0.1</td>
<td>0.12</td>
<td>0.1 (0.01)</td>
<td>96.24</td>
<td>98.3</td>
</tr>
<tr>
<td>redwine</td>
<td>0.15</td>
<td>0.17</td>
<td>0.16 (0.01)</td>
<td>59.85</td>
<td>61.91</td>
</tr>
<tr>
<td>seeds</td>
<td>0.05</td>
<td>0.07</td>
<td>0.06 (0.01)</td>
<td>92.86</td>
<td>94.76</td>
</tr>
<tr>
<td>sonar</td>
<td>0.14</td>
<td>0.16</td>
<td>0.15 (0.01)</td>
<td>75.03</td>
<td>80.79</td>
</tr>
<tr>
<td>spambase</td>
<td>0.05</td>
<td>0.06</td>
<td>0.06 (0)</td>
<td>94.78</td>
<td>94.94</td>
</tr>
<tr>
<td>statlog</td>
<td>0.04</td>
<td>0.06</td>
<td>0.05 (0.01)</td>
<td>86.4</td>
<td>88.77</td>
</tr>
<tr>
<td>waveform</td>
<td>0.09</td>
<td>0.12</td>
<td>0.11 (0.01)</td>
<td>86.54</td>
<td>86.92</td>
</tr>
<tr>
<td>whitewine</td>
<td>0.12</td>
<td>0.14</td>
<td>0.13 (0.01)</td>
<td>53.21</td>
<td>54.76</td>
</tr>
<tr>
<td>yeast</td>
<td>0.18</td>
<td>0.19</td>
<td>0.18 (0)</td>
<td>53.1</td>
<td>56.67</td>
</tr>
</tbody>
</table>
## Table B.11: Combination of randomised parameters: feature proportion $F_p$, training patterns proportion $D_p$ and number of hidden neurons $H_n$, comparison between additionally randomised proportion of random connections $C_n$ and number of the hidden layers $H_l$.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>RNL</th>
<th>RFR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Diversity</td>
<td>Accuracy</td>
</tr>
<tr>
<td>blocks</td>
<td>0.01</td>
<td>0.02</td>
</tr>
<tr>
<td>chess</td>
<td>0.13</td>
<td>0.15</td>
</tr>
<tr>
<td>faults</td>
<td>0.1</td>
<td>0.12</td>
</tr>
<tr>
<td>redwine</td>
<td>0.12</td>
<td>0.15</td>
</tr>
<tr>
<td>seeds</td>
<td>0.04</td>
<td>0.12</td>
</tr>
<tr>
<td>sonar</td>
<td>0.12</td>
<td>0.13</td>
</tr>
<tr>
<td>spambase</td>
<td>0.04</td>
<td>0.06</td>
</tr>
<tr>
<td>statlog</td>
<td>0.03</td>
<td>0.14</td>
</tr>
<tr>
<td>waveform</td>
<td>0.09</td>
<td>0.14</td>
</tr>
<tr>
<td>whitewine</td>
<td>0.1</td>
<td>0.13</td>
</tr>
<tr>
<td>yeast</td>
<td>0.17</td>
<td>0.19</td>
</tr>
</tbody>
</table>

Table B.11: Combination of randomised parameters: feature proportion $F_p$, training patterns proportion $D_p$ and number of hidden neurons $H_n$, comparison between additionally randomised proportion of random connections $C_n$ and number of the hidden layers $H_l$. 
Bibliography


Selbständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig und nur mit erlaubten Hilfsmitteln angefertigt habe.

Konstanz, den December 21, 2016

Anna Martin