Aalto University
School of Science
Degree Programme in Computer Science and Engineering

Dominik Mäkinen

Comparison of machine learning methods for social media and open data to predict sales

Master's Thesis
Espoo, November 27, 2015

Supervisor: Professor Petri Vuorimaa
Advisor: Niko Myller PhD. (Computer Science)

Aalto University
School of Science
Degree Programme in Computer Science and Engineering

ABSTRACT OF MASTER'S THESIS

Author:	Dominik Mäkinen		
Title: Comparison of machine learning methods for social media and open data to predict sales			
Date:	November 27, 2015	**Pages:**	58
Major:	Software Technology	**Code:**	T3001
Supervisor:	Professor Petri Vuorimaa		
Advisor:	Niko Myller PhD. (Computer Science)		

Machine learning is a part of artificial intelligence research. It is a combination of mathematics, statistics, and computer science, and its aim is to teach machines with a data to conduct accurate predictions. Machine learning can be divided into three categories: supervised learning, unsupervised learning, and reinforced learning.

The prediction problems consist of labeling a correct class (classification) or estimating a numerical value (regression) to an unseen data object. Machine learning is executed with a choice of different methods or a combination of them (ensemble). This thesis looks more closely on the four different machine learning methods: support vector machines/regressors, multilayer perceptron, random forest, and linear regression.

One of the uses of machine learning lies in sentiment analysis. It means retrieving opinions and categorizing them to numerical values from a textual data. The values are called sentiments and can range from 0 to 10. The lowest sentiment is totally negative opinion, five is a neutral opinion, and 10 is totally positive opinion.

The methods in this thesis are evaluated via receiver operating characteristics curves and different mean error values. The four methods were evaluated in the context of predicting car sales amounts with previous sales amounts and social media sentiment analysis data. It was noticed, that random forest produced the best results. The experimentation was conducted by a program written for the thesis called Sales Predictor.

Keywords:	machine learning, regression, support vector machines, multilayer peceptrons, random forests, linear regression, supervised learning, sentiment analysis
Language:	English

Aalto-yliopisto
Perustieteiden korkeakoulu
Tietotekniikan koulutusohjelma

DIPLOMITYÖN TIIVISTELMÄ

Tekijä:	Dominik Mäkinen		
Työn nimi: Koneoppimisen menetelmien vertailu myynnin ennustamisessa sosiaalista mediaa ja avointa dataa hyödyntäen			
Päiväys:	27. marraskuuta 2015	**Sivumäärä:**	58
Pääaine:	Ohjelmistotekniikka	**Koodi:**	T3001
Valvoja:	Professori Petri Vuorimaa		
Ohjaaja:	Filosofian tohtori Niko Myller		

Koneoppiminen on osa tekoälytutkimusta. Se on yhdistelmä matematiikkaa, tilastotiedettä, ja tietojenkäsittelytiedettä ja sen tavoite on opettaa koneita muodostamaan tarkkoja ennusteita datan avulla. Koneoppiminen voidaan jakaa kolmeen kategoriaan: ohjattu oppiminen, epäohjattu oppiminen ja vahvistettu oppiminen.

Ennusteongelmat sisältävät oikeaan luokkaan merkitsemistä (luokittelu) tai numeerisen arvon estimointia (regressio) tuntemattomalle dataobjektille. Koneoppiminen suoritetaan valitsemalla tietty menetelmä tarpeen mukaan tai yhdistelemällä eri menetelmiä. Tämä työ käy tarkemmin läpi neljä eri menetelmää: tukivektorikoneet/-regressorit, monikerrosperseptronit, satunnaismetsät ja lineaariregression.

Eräs koneoppimisen käyttökohde on sentimenttianalyysi. Se tarkoittaa mielipiteiden keräämistä teksteistä ja niiden luokittelua numeerisiin arvoihin. Näitä arvoja kutsutaan sentimenteiksi ja ne voivat olla välillä 0:sta 10:een. Alin sentimentti on täysin kielteinen mielipide, arvo viisi on neutraali mielipide ja kymmenen on täysin positiivinen mielipide.

Tämän työn metodit arvioidaan hyödyntämällä ns. receiver operating characteristics -käyriä sekä eri keskivirhelukuja. Neljän metodin arvioinnit tehtiin automerkkien myyntilukujen kontekstissa, jossa yhdistettiin aikaisemmat myyntiluvut sosiaalisen median sentimenttianalyysidataan. Havaittiin, että satunnaismetsä tuotti parhaimmat tulokset. Koe suoritettiin työtä varten kirjoitetulla Sales Predictor -ohjelmalla.

Asiasanat:	koneoppiminen, regressio, tukivektorikoneet, monikerrosperseptronit, satunnaismetsät, lineaariregressio, ohjattu oppiminen, sentimenttianalyysi
Kieli:	Englanti

Acknowledgements

I would like to express my gratitude to my advisor PhD Niko Myller for giving an opportunity for the thesis, his sharing of knowledge of machine learning and data mining, and helping me overall throughout the thesis. I would also like to thank my supervisor Prof. Petri Vuorimaa for giving advice to the structure of the thesis.

Espoo, November 27, 2015

Dominik Mäkinen

Abbreviations and Acronyms

ANN	Artificial neural network
AIC	Akaike information criterion
AOC	Area over curve
ARFF	Attribute-relation file format
AUC	Area under curve
BIC	Bayesian information criterion
CART	Classification and regression trees
DOA	Direction of accuracy
EM	Expectation-maximization
GCV	Generalized cross-validation
IDF	Inverse document frequency
LR	Linear regression
MAE	Mean absolute error
MEB	Mean error bias
MSE	Mean squared error
MLP	Multilayer perceptron
NLP	Natural language processing
REC	Receiver error characteristic
RF	Random forest
RMSE	Root mean squared error
ROC	Receiver operating characteristic
RROC	Regression receiver operating characteristic
SMSA	Social media sentiment analysis
SQL	Structured query language
SSE	Sum of square errors
SST	Sum of square total
SVM	Support vector machine
SVR	Support vector regression/regressor
TF	Term frequency

Contents

Abbreviations and Acronyms **v**

1 Introduction **1**
1.1 Problem statement . . . 2
1.2 Structure of the Thesis . . . 2
1.3 Related work . . . 3

2 Machine learning **4**
2.1 Machine learning in general . . . 4
2.2 Different categories of machine learning . . . 7
2.2.1 Supervised learning . . . 7
2.2.1.1 Classification . . . 7
2.2.1.2 Regression . . . 8
2.2.1.3 Ensemble . . . 9
2.2.2 Unsupervised learning . . . 9
2.2.3 Reinforced learning . . . 10
2.3 The performance evaluation of machine learning methods . . . 11

3 Machine learning methods **17**
3.1 Support vector machines . . . 18
3.2 Multilayer perceptrons . . . 23
3.3 Random forests . . . 26
3.4 Linear regression . . . 30

4 Text mining **35**
4.1 Sentiment analysis . . . 37

5 Sales Predictor **40**
5.1 The data . . . 43
5.2 Experimentation setup . . . 46
5.3 Results . . . 47

6 Conclusion **55**

Chapter 1

Introduction

In general, there has always been a need to predict future events or states. The question is, whether the object to be predicted is the weather of tomorrow, the sales figures of the company in the next year, or which team wins the Stanley Cup in the National Hockey League in the next season, how the prediction can be done. One way could be to look the previous results and hope that the history repeats itself. However, there could be more robust and accurate methods to conduct the predictions and there is. The answer for that lies in the artificial intelligence. It is a field of study that explores the capabilities of machines to perform intelligent actions and behaviours. One area in the artificial intelligence, that really aims to excel in the prediction of the future, is called machine learning. It really means what its name suggests: to teach a machine with examples to learn or discover specific patterns.

Machine learning is a combination of mathematics, statistics, and computer science to create models that transfers input to output from which the output is a predicted value. The prediction process has at least two phases. The first is the learning phase and the second is the actual prediction phase. In the learning phase, the input is data, that has examples of the values of the features (or attributes or columns) and what should be the outcome (a certain class or a numeric value). In the prediction phase, the input data is similar to the data in the learning phase but without the supposed outcome. The aim is to predict the outcome value and that is the output of the model. Machine learning can also be used to discover patterns in the data, where the input is given only once and the output can be clusters of data, which should have similar properties. The next sections state the main objectives as well as the structure of the thesis.

1.1 Problem statement

The aim of the thesis is to find out which of the selected four machine learning methods produce the most accurate predictions in the context of car sales figures. Another sub-objective is to see, does social media sentiment analysis increase the prediction accuracy.

The machine learning methods used in this thesis are: support vector machines/regressors, linear regression, random forest, and multilayer perceptron. The prediction problem is in the genre of regression problems, as the output is numerical instead of classes. The main evaluation metrics used to see which method produces most accurate results, are root mean squared error and regression receiver operating curves.

The experimentation, to provide the results for the objectives of this thesis, is executed by the Java program, mainly written for this thesis and later to be used in other purposes, called Sales Predictor. The input data for the program is previous car sales amounts and social media sentiment analysis data. The output of the program is evaluation metrics for different specified machine learning methods.

The thesis uses mainly books for sources of the background information, because of the level of generality. The results chapter uses also some code related sources.

1.2 Structure of the Thesis

Chapter 2 is about different aspects of machine learning. It starts with stating different definitions of machine learning and what is machine learning in general. Then it elaborates on different categories of teaching a machine such as supervised learning, unsupervised learning, and reinforced learning. The chapter ends with looking on evaluation metrics and methods to determine the quality of different machine learning methods and models. Chapter 3 elaborates on the chosen four machine learning methods: support vector machine/regressor, multilayer perceptron, random forest, and linear regression. These sections take into account both classification and regression problems. Chapter 4 looks more closely in mining text and sentiment analysis. This is background information to understand the data that Sales Predictor uses. The final chapter (5) before the conclusion is the experimentation part of the thesis. The chapter starts by examining the program called Sales Predictor, which evaluates different machine learning methods. The chapters looks more in detail the data the program uses, the experimentation setup, and the results for the problem statements mentioned in the previous section.

1.3 Related work

There have been several studies, where different machine learning methods are compared in both classification and regression problems. They usually follow the following procedure: pick several methods, apply them to a specific domain in question, and evaluate results with some statistics, usually different mean errors. Garcia-Gutierrez et al. compared seventeen machine learning regression methods in a light detection and ranging regression domain [13]. They used R-value with root mean squared error to determine the best method. Support vector regression with Gaussian kernels produced the best results. Caruana et al. [9] used eight different evaluation metrics to ten different machine learning methods in eleven classification problems to determine the best method overall. Boosted trees produced the best results in that case. Abu-Nimeh et al. [1] studied phishing detection (classification) by comparing six different methods. They used false positive and false negative rate along with precision, recall, F-value, and regression operating characteristics in evaluating the methods. There was not one best method and the study points out that it is difficult to find a single metric to find the best one. There was not found a study that compared different machine learning methods exactly in the domain of car sales amount with social media sentiment analysis data, thus the above studies are examples of different works related to the actual comparison of different methods independent of the domain.

Chapter 2

Machine learning

This chapter provides general information about the subject of machine learning. First, I will go through how the machine learning is defined and how it is related to other close fields, such as data mining and statistics. I will also explain a few core concepts related to learning with machines. Second, I will look closer on different categories of machine learning, for example, supervised learning, unsupervised learning, and reinforced learning. Finally, I will explain how different machine learning methods are evaluated by their performance.

2.1 Machine learning in general

There is an increasing need to process huge amounts of data available to companies. Data is gathered from nearly everywhere and everything. The data is also not always completely random. For example, people buy certain things during certain seasons forming a pattern. Machine learning is about gaining insight about the pattern or patterns of the data. Machine learning can also be said to be about optimizing a performance criterion with example data or past experience. [3] In other words, machine learning is the development of computer algorithms for transforming data into intelligent action. The action can be a classification, a numerical prediction, or an action sequence (reinforcement learning). Rather than training the machines as general learners, machine learning aims to teach a machine how to find solutions to specific problems. It can be said, that machine learning is analogically closer to training an employee than raising a child.

How is the role of teaching a machine then related to other similar fields? Machine learning is closely tied to data mining and statistics. Machine learning is about teaching a machine, while data mining aims to find novel insight from large databases. [19] The machine learning methods are mainly based on statistics, and they both have many concepts, that have same meaning but use different words. For example, inference in statistics means going from particular observations to general descriptions in machine learning, estimation is called learning, and discriminant analysis goes by the name of classification. [12]

In the past few decades, it was believed that what we need is a new type of thinking or a new model of computations for the machine learning to be highly effective. Because of the recent successes of machine learning, it seems that what we actually need is a huge amount of learning data. [3] Machine learning has been used in, for example, detecting spam messages from e-mail, predicting the outcomes of elections, and predicting stock prices [19].

Formally, the point of machine learning is to find a function $f : X \Rightarrow Y$, which maps data (X) to predictions (Y). The function belongs to a certain function class, which consists of different learning algorithms. The elements of X and Y are application-specific. [6]

A learning process usually looks as follows:

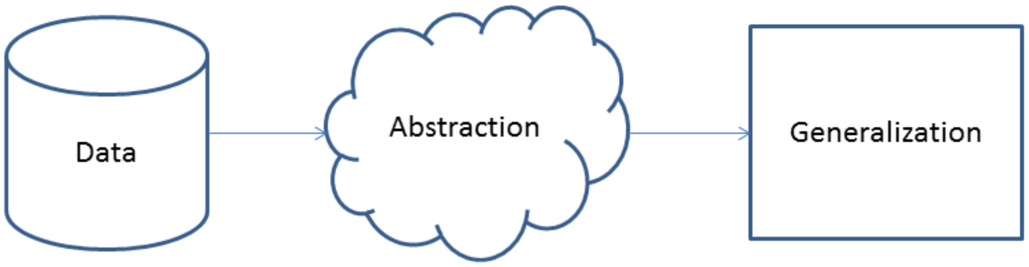


Figure 2.1: Learning process.

In the figure 2.1, data or data input includes observations, memory storage, and factual basis for further reasoning. Abstraction translates the data into broader representations and gives the data a meaning. Abstracted connections are a basis of knowledge representation. In knowledge representation, the data inputs are summarized into a model, that is explicit description of the structured patterns among data. Equations, trees, and logical if-else rules are examples of a model. The process, where a suitable model is searched, is called training. [19] The point of machine learning is rarely to replicate the training data but to predict new, unseen cases. This procedure of predicting new instances is called a generalization ability of a method. [3] Generalization uses abstracted data to form action and makes the learner useful for applications. While every possible model could be examined, it is not feasible. Thus, learners use heuristics to reduce the possible models. With heuristics, there comes also a bias towards certain solutions. [19]

There are a couple of concepts related to learning itself. First, the problems are called ill-posed in learning. If a problem is ill-posed, it means that there is no unique solution present and the solutions are not stable. There might be different solutions with a different training data in a same learning problem. The learning becomes less ill-posed as more training data becomes available. [3] [17]

Second, to have a unique solution for a learning problem, additional assumptions (heuristics) have to be made. These assumptions are called inductive bias. For example, choosing an optimizing criterion (an error to be minimized) is an inductive bias. An inductive bias for the data, which the learning algorithms often use, is independent and identically distributed samples. These kind of samples are all drawn from the same joint distribution. [3]

The independent and identically distributed assumption might not hold, as the distributions might not be identical. Covariate shift is an assumption in machine learning, that the training input items and the test input items follow different probability distributions but the distributions of output values with input items are unchanged. [24]

Third, if we have a model, whose complexity is less than the complexity of the function that generates the data, the model underfits. Fitting a linear model to the data that is generated by a higher degree polynomial, is an example of underfitting. On the other hand, if the model fits the training data almost perfectly, we might have an overfitting model. Overfitting is not good as any noise in the training data decreases the prediction capability of the model.

Finally, the data might have a huge amount of dimensions, in which case, dimensionality reduction might be needed. The dimensionality of the data affects the time and space complexities of certain learning algorithms, so by reducing the dimensions of the input data, the algorithms become faster. There are also practical reasons why using many dimensions might not be feasible, as the computing and the memory requirements increases substantially. The high dimensional data can not be visualized, unless the dimensions are reduced.

The main methods for dimensionality reduction are feature selection and feature extraction. In feature selection, the k most informative dimensions are found while the other dimensions are discarded. Subset selection is one method for feature selection. In feature extraction, a new set of k dimensions are found, that are combinations of the original dimensions. A widely used method for feature extraction is called principal component analysis. [3]

To use machine learning in applications, a following process can be used. First, the data needs to be collected. Second, the data should be prepared for the learners. This step takes usually a lot of effort (80% of total effort). Third, a model is trained with the data. Fourth, the performance is evaluated, and finally, the model is improved. [19]

2.2 Different categories of machine learning

The following categorization is quite general in my opinion. The categories could also be divided by, for example, how the classifying method works. Then, the categories would be functions, trees, rules, etc.

2.2.1 Supervised learning

Supervised learning aims to find a mapping from the input to an output. The correct output values are known, hence the word supervised. [3] Supervised learning algorithms construct a prediction function f from training data. The function is then applied to test instances. The training data consists of labeled examples of items to be learned. Supervised learning algorithms aim to produce generalizing functions, that attempt to label unseen data as correctly as possible. [6]

2.2.1.1 Classification

Classification is about deciding which class a certain data item belongs to. A machine learning system that does the classification is called a classifier. The classifier learns a discriminant, which is a function that separates the data items to different classes. Thus, the classifier can predict which class a new data item belongs to using the discriminant. [3]

As an example of classification, let us consider a case of classifying family cars. We have data, for example, engine power, price, seating capacity, brand, and color, about different cars. The next step is to label the data to whether a car is a family car (positive example) or not (negative example). The labeling is done by, for example, group of experts. The labels are names for the classes. The class is a description that is shared by all the positive examples and not by the negative examples. Then, we decide which parts of data, that is attributes, are relevant for separating family cars from the rest. Let us say, that the price and engine power of a car are the relevant attributes.

For example, with the analysis of data, we could see that the price and the engine power fall into a certain range of values for the positive examples (family cars), thus generating a rectangle. The set of every area inside these ranges is called a hypothesis class. An area is called a hypothesis. The aim is to find a hypothesis that best describes which cars are family cars and which are not. The hypothesis class includes the most specific hypothesis and the most general hypothesis. The most specific hypothesis is the smallest area where the positive examples fall. The most general hypothesis is the largest area where no negative example falls. The goal hypothesis is somewhere between the most specific and the most general hypothesis. Also, every hypothesis that falls inside the most specific and the most general ones make up the version space. The goal hypothesis is also the classifier. So, each new case (car) to be predicted will be classified as a family car, if its price and engine power falls in the goal hypothesis. Of course, the assumption that the classifier is a rectangle, might be not perfect. The perfect classifier might also be a much more complex shape than a rectangle. Thus, the rectangle classifier might contain errors.

The error, or empirical error, is the proportion of misclassified instances. The reason why the rectangle classifier is more feasible than the perfect one is the simplicity. First, it is easy to check whether a point is inside a rectangle. Second, it is easier to train and has fewer parameters. Finally, it is more easily explained than a complex one. [3]

2.2.1.2 Regression

Regression is a prediction problem in which the output value is a number. For example, a price of a car could be an output in a regression system, where, for example, mileage, brand, and year, could be the input. [3] So, the difference between regression and classification is that regression aims to predict a real number, while classification aims to predict an item from a finite set of classes. [6]

Regression can be expressed as $r^t = f(x^t) + \epsilon$, where r is the numeric value to be predicted, $f(x^t)$ is an unknown regression function and epsilon is random noise. The noise can be interpreted as unobservable extra hidden variables. The loss function on empirical error used in the regression is squared difference of the predicted and the actual value, that is $E(g|X) = \frac{1}{N}\sum_{t=1}^{N}(r^t - g(x^t))^2$. The goal is to find a model g(x) that minimizes the loss function in the learning process. Linear regression is expressed as $g(x) = w_1 x + w_0$, where w_1 and w_0 are learned from the data. [3]

2.2.1.3 Ensemble

The methods can be combined to achieve better accuracy than using a single method, because no single learning algorithm is overall the best. For example, a parametric classifier and a nonparametric classifier can be combined. These sub classifiers are called base-learners. These base-learners form a new type of learner called combined learner. The tradeoff for using a combined learner instead of individual learners is increased time and space complexity. Also, if the base learners are not combined sufficiently, the accuracy might even be worse than using a single learner. The point is to combine a set of diverse learners, so that the learners complement each other. Also, one method is using one learner but with different hyperparameter combinations, such as different number of hidden units in a multilayer perceptron. The base learners can also be taught by using different parts of data for different base learners. This helps reducing the effect of the curse of dimensionality. Overall, the accuracy of the combined learner is more important than the accuracies of single base learners. [3]

2.2.2 Unsupervised learning

In unsupervised learning the correct output values are not known, so the only data is the input data. The unsupervised learning aims to find regularities from the input data. [3] The specific goal of unsupervised learning depends heavily on the situation and the applications and it is sometimes not mathematically well-defined. [24]

Closely related to unsupervised learning is semi-supervised learning. Semi-supervised learning has properties in both supervised and unsupervised learning. In semi-supervised learning the data consists of two parts. The first part is a data which has labels in it and the second part is a data which has no labels in it. The aim is then to predict only the second part of the data using the first part of the data. This is also called transductive learning. [10]

One of the popular unsupervised learning method is clustering, where the aim is to find clusters or groupings of the data. In business, finding customer segments from the data, that contains demographic information and past transactions with the company, is an example of clustering. Clustering can also be used as a preprocessing technique for supervised learning. For example, after a clustering algorithm is executed on the data, similarities inside data might be found. These similarities form groups or clusters and then, for example, experts can name these groups as labels for the supervised learning. [3]

A distance measure has to be defined before a clustering algorithm can be ran. The distance measure represents how far away each data point is from each other. [24]

k-Means is an algorithm to perform clustering. Before any calculations, the value of k is determined, for example, by simply choosing one. The value specifies, how many clusters are selected to represent the data. Then, k means are generated to serve as identifiers for the clusters. The following two step cycle is then repeated, until the error is insignificant. First, every data point is assigned to its nearest mean. Second, the means are recalculated to represent better the clusters. k-Means is a special case of an expectation-maximization algorithm (EM). EM-algorithm also consists of two steps. First, it calculates an expected value with a parameter for the likelihood of observed and hidden values. Second, it maximizes the parameter [3]

2.2.3 Reinforced learning

There are systems, for example, turn based games or a robot navigating in an environment, where the actions are points of interest. More specifically, the actions themselves are not important but the correct sequence of actions. Learning of these kinds of action sequences is called reinforcement learning.

The components of a reinforcement learning problem include: agents, an environment, a set of states, a set of actions, and rewards. The agents interact with the environment and make decisions to achieve a certain goal. The environment is a set of rules, where the agents act. The environment is always in a certain state, that influences the next decisions of the agents. A reward is a feedback for the agents to tell, whether a sequence of actions was towards the goal. The agents take an action based on the state of the environment and the reward received. The best sequences of actions are those that generate the maximum reward. The reinforcement learning can be implemented using Markov decision process. Alongside the aforementioned components, there is a policy, which defines the behaviour of an agent. Thus, the policy is a mapping from the states to actions: $\pi : S- > A$. The value of a policy is either $V^{\pi}(s_t) = E[\sum_{i=1}^{T} r_{\mathrm{t+i}}]$ (finite-horizon, step limit present), or $V^{\pi}(s_t) = E[\sum_{i=1}^{\infty} \gamma^{\mathrm{i\text{-}1}} r_{\mathrm{t+i}}]$ (infinite-horizon, no step limit). The gamma is a discount rate $(0 <= \gamma < 1)$ to keep the return finite. The point in the reinforcement learning is to find the optimal policy for an agent. More formally, the task is to find a policy π^*, such that $V^*(s_t) = \max_{\pi} V^{\pi}(s_t) \forall s_t$. One famous solution for that is Bellman's equation: $Q^*(s_t, a_t) = E[r_{t+1}] + \gamma \sum_{s_{t+1}} P(s_{t+1}|s_t, a_t) \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1})$, where Q means a state-action pair. [3]

2.3 The performance evaluation of machine learning methods

The point of the evaluation of different machine learning methods is to calculate an error for a method and compare it to the error of other methods in a given application. The general principle of the evaluation is to split the data into training, validation, and test sets. The method is taught using the training set, validated against a validation set to get the expected error, and finally use test set to see how the method would perform in a real situation. The comparison of different methods in general is useless. This is, because the performance of the methods rely heavily on the data used. One method, that performs well with one data set, might be totally worthless in another. Thus, there exist no universally best method for everything, there is no free lunch. There are different indicators of performance other than error rates. For example, training time, testing time, space complexity, and the simplicity of the method could be other indicators. Usually, these are not treated equal because of the different needs of different applications.

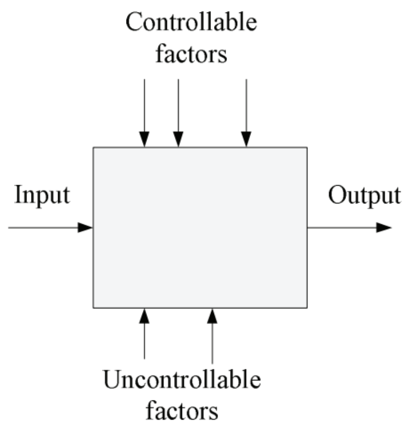
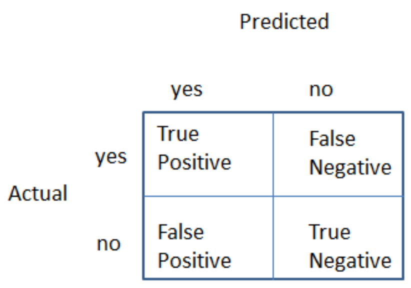


Figure 2.2: Setup for machine learning experiminent.

Figure 2.2 shows a machine learning experimentation setup. The heart of the experiment are the controllable factors, whose values the evaluator aims to optimize for the application. These include the method in use or the hyperparameters of the method, such as, the number of hidden units for a multilayer perceptron. The experimenter tries to minimize the amount and the effects of the uncontrollable factors. These factors include the noise in the input data and randomness in the optimization process.

Cross-validation is used to evaluate, how erroneous the methods are in a classification problem. The training data is split into K parts, which again is split into a training set and a validation set. Then, in each part, the method is trained and validated. This is then repeated by using the same amount of but different splits. K-fold cross-validation splits the training data into K equally sized sets. One set is used as a validation set, while others are used as a training set (size K-1). This is then repeated so that each split is being a validation set once, thus giving K amount of repetitions. The errors are gathered and averaged. The errors in cross-validation techniques are slightly dependent as the data will overlap with itself. [3]



		Predicted	
		yes	no
Actual	yes	True Positive	False Negative
	no	False Positive	True Negative

Figure 2.3: Confusion matrix

A useful tool for measuring classification performance is confusion matrix, as shown in the figure 2.3. The confusion matrix is a table, which indicates the amounts of correctly and incorrectly labeled examples. The size of the matrix is the amount of the classes times the amount of the classes. However, the usual case is to discern one class versus all others, so the size of the matrix will become two times two.

The parts of the confusion matrix are as follows:

- True Positive (TP): correctly predicted as a positive example
- True Negative (TN): correctly predicted as a negative example
- False positive (FP): incorrectly predicted as a positive example
- False negative (FN): incorrectly predicted as a negative example

There are a total of five useful metrics derived from the confusion matrix. The first is the accuracy. The accuracy tells the proportion of correctly predicted examples. The formula for accuracy is $accuracy = \frac{TP+TN}{TP+FP+TN+FN}$. Usually, only accuracy is used to measure performance. However, a case, where 99990 instances of 100000 are negative, a classifier that outputs every time a negative prediction, would have an accuracy of 99,99%. This tells nothing of the usefulness of the classifier, thus, more performance measurements are needed. The second and third measures are the sensitivity and the specificity. The sensitivity indicates the proportion of correctly classified positive examples. It is defined as: $sensitivity = \frac{TP}{TP+FN}$. The specificity tells the proportion of correctly predicted negative examples. It is specified as $specificity = \frac{TN}{TN+FP}$. The fourth and the fifth performance measures are used mainly in the context of information retrieval and they are precision and recall. The precision means, how well the model is to be trusted. The precision tells the proportion of the positive examples that are truly positive. It is defined as $precision = \frac{TP}{TP+FP}$. The recall indicates the breadth of the model, that is, how large the proportion of the correctly labeled examples is from the total positive examples. The equation of the recall is exactly same as the equation of the sensitivity, $recall = \frac{TP}{TP+FN}$.

There are also two more measures, which are a combination of the confusion matrix measures. The first is the kappa statistic. The kappa statistic is an adjustment for the accuracy, that takes account the possibility of a correct prediction by chance alone. The kappa statistic takes values between zero and one and is defined as $kappa = \frac{P(a)-P(e)}{1-P(e)}$, where $P(a)$ is the proportion of actual agreement between the classifier and the true values and $P(e)$ is the expected agreement between the classifier and the true values. $P(a)$ is same as the accuracy, while $P(e)$ is calculated as the sum of the proportion of predicted positive examples times the proportion of predicted negative examples and the proportion of actual positive examples times the proportion of actual negative examples. The second measure, the F-measure, is harmonic mean of the recall and the precision. It is defined as $F-measure = \frac{2*precision*recall}{recall+precision}$, thus it combines the precision and the recall into a one number.

A common visualization tool for the performance evaluation is the receiver operating characteristic curve, or the ROC curve. The ROC curve indicates the tradeoff between the detection of true positives and the false positives.

Most classifiers fall between the perfect classifier and the classifier with no predictive value as the test classifier in the figure 2.4. Usually, a measure called area under the ROC, or AUC, is calculated to see how close a classifier is to a perfect classifier. [19]

In regression problems, a tool for evaluating performance is regression

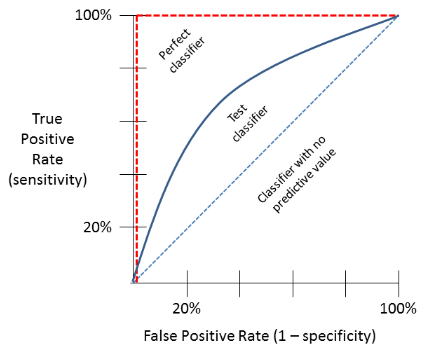

100%
20%
True Positive Rate (sensitivity)
Perfect classifier
Test classifier
Classifier with no predictive value
20%
100%
False Positive Rate (1 – specificity)

Figure 2.4: ROC curve visualization

error characteristic (REC). They are a generalization for ROC curves to regression. REC curves have an error tolerance on x-axis and percentage of points predicted within the tolerance on y-axis. The result is an estimate of cumulative distribution function of the error. The metric of the error can be simply the difference between the actual value and the predicted value, a squared difference, or some other error metric. Analysis of REC curve provides area over the REC curve as an estimate of an expected error. A visualization of REC curve can be seen in figure 2.5. [7]

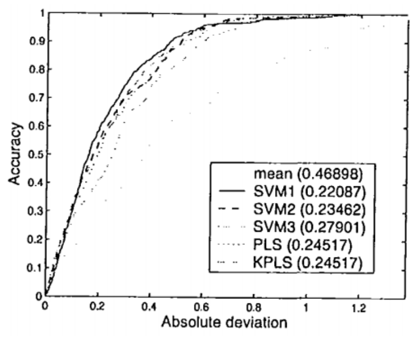


Figure 2.5: REC curve visualisation [7]

However, REC curves do not take account the possible asymmetricity of a loss function. A loss function is any function $l : Y \times Y \to R$, which compares the elements in the output domain. The commonly used loss functions are the absolute error and the squared error. The absolute error is difference between a predicted value and the actual value, while the squared error is the same difference squared. If the errors are calculated between every predicted-actual value pairs and the result is divided by the number of elements, we get mean absolute error (MAE) and mean squared error (MSE). There is also root mean squared error (RMSE), if we take a square root from MSE. No superior loss function exists and many different loss functions should be used, when evaluating the performance of a model. These loss functions are symmetric, because there is no weight difference between over-estimating ($predicted - actual > 0$) or under-estimating ($predicted - actual < 0$) the prediction. However, an asymmetric loss means, that over-estimation and under-estimation are not equal. For, example compared to symmetric MAE, an asymmetric linear-linear error function is defined as:

$$l_\alpha(\widehat{y}, y) = \begin{cases} 2\alpha(y - \widehat{y}) & \text{if } \widehat{y} < y \\ 2(1-\alpha)(\widehat{y} - y) & \text{otherwise} \end{cases}$$

where α is the cost proportion (between 0 and 1) with increasing values meaning higher cost for low predictions.

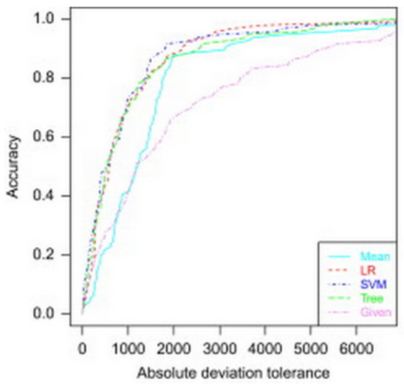


Figure 2.6: ROCC curve visualization [16]

An evaluation tool that takes account asymmetric loss is called regression ROC (RROC) curves. The RROC curves are defined in the RROC space, which is a two dimensional space with the sum of over-estimations in the horizontal dimension, and the sum of under-estimations in the vertical dimension. A location $(0, 0)$ is called RROC heaven and it is situated in the upper left corner. The closer model's predictions are from the RROC heaven (in Manhattan distance), the better. RROC curves are created by using the concept of shifts. It is a constant value, that is added to each $predicted - actual$ pair. The value of shift is defined as:

$$s^*(\alpha) = arg \min_s \{2(1-\alpha) \cdot OVER\langle s \rangle - 2\alpha \cdot UNDER\langle s \rangle\}$$

As α has only values between 0 and 1, there is a certain range for the shift values. The number of shifts needed to make a RROC curve is equal to unique $prediction - actual$ pairs. The curve is drawn through each $predicted - actual + shift$ values. Thus, the result looks similar, as ROC curve for classifier. As the point of interest in ROC curves for classifiers is the area under curve, in ROCC the interest is in the area over the curve (AOC). Lower values for AOC are better than higher. A visualization of ROCC curve can be seen in figure 2.6. [16]

Chapter 3

Machine learning methods

The main factor for choosing the machine learning methods for comparison in this thesis is how different they are. The methods are support vector machine, multilayer perceptron, random forest, and linear regression. Each have their own unique way to represent the data and make predictions. Here I will give a short summary of each method. The following chapters explain the methods in more detail. The equations used in the chapters are from the sources that the chapter references.

Support vector machine (chapter 3.1) is a method that aims to find a plane, or a hyperplane, that differentiates the example data, by the class, as well as possible. Thus, if there are two classes, the hyperplane lies between them and its distance to the classes is as maximal as possible. In numerical predictions (regression problems), the goal is to find a function that deviates from the example data only a certain amount and that the deviation is as small as possible.

Multilayer perceptron (chapter 3.2) is a method that mimics biological neural systems. They consist of neurons, or perceptrons, that take input and translate it to an output through a certain function. The input consist of the features of a data point. The input is then weighed and given as a parameter to a function. The output of the function is then given to another perceptron as an input. This propagation continues, until there are no layers to give the output and the final solution is then the output of the multilayer perceptron. A layer is just a set of perceptrons in the output-to-input chain. After the final output is calculated, the backpropagation begins, where the output is transferred from the final layer to the first layer to optimize the weights. This, the propagation and the backpropagation, continues, until a certain threshold for accuracy is reached.

The method random forest (chapter 3.3) is based on the tree data structure. The random forest consists of multiple trees, which have a root, middle nodes, and leaves. The trees are decision trees, which are a standalone learning methods on their own. The decision trees are a series of tests (the nodes in the tree) about the values of the features of the data. The tests are basically whether a value of a feature is larger or smaller than a certain threshold. The result of a single test determines the next test to be executed. Thus, the random forest is an ensemble method, that make predictions through the individual learners and outputs the best result or an averaged one.

Linear regression (chapter 3.4) is probably one of the most common prediction methods in statistics. The point is to create a linear model $y = \beta_0 + \beta_1 x + \epsilon$ from the data and make predictions based on that model.

3.1 Support vector machines

A support vector machine (SVM) can be defined as an abstract learning machine, which will learn from a training data and attempt to generalize on novel data. In a simplest case of the problems, a binary class problem, the task of an SVM is to find a directed hyperplane, which is maximally distant from the two classes. The closest points from the both sides of the hyperplane have the most influence and they are called support vectors. This can be seen in figure 3.1.

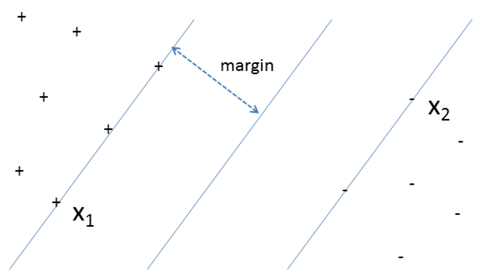


Figure 3.1: SVM visualization. x_1 and x_2 represent support vectors. The lines, where the support vectors lie, are called canonical hyperplanes and the line in the middle is called the hyperplane.

The resulting hyperplane fulfills the equation $w \cdot x + b = 0$, where w or weights are the normal to the hyperplane, x are the points within the hyperplane, and b is the bias or offset of the hyperplane from the origin in the input space. This SVM requires that the data clusters are linearly separable.

Generalization error is a theoretical prediction error, when applying a classifier to an unseen data. In order to achieve best possible predictions, this error needs to be minimized. Additionally, to obtain a more detailed definition of an SVM, the upper bound of the generalization error for the SVM needs to be considered. The upper bound has two features. First, the bound is minimized by maximizing the margin, γ, that is the minimal distance between the hyperplane and the closest data points. Second, the bound is independent of the dimensionality of the space.

Let us consider a problem, where $x_i (i = 1, ..., m)$ are the data points and $y_i = \pm 1$ are the corresponding class labels. The decision function is

$$f(x) = sign(w \cdot x + b) \tag{3.1}$$

The data is correctly classified, if $y_i(w \cdot x_i + b) < 0 \forall i$, since $(w \cdot x_i + b)$ should be positive, when $y_i = +1$ and vice versa in the negative case. Because the function 3.1 is invariant under a positive rescaling of $(w \cdot x_i + b)$, we define a scale for (w, b) by setting $w \cdot x + b = 1$ for the closest points on one side and $w \cdot x + b = -1$, for the closest points on the other side. These two define canonical hyperplanes and the region between these hyperplanes defines the margin band. For retrieving the margin, let x_1 and x_2 be two points inside the canonical hyperplanes, one on the positive side and one on the negative side. Thus, $w \cdot x_1 + b = 1$ and $w \cdot x_2 + b = -1$ and they are deduced to $w \cdot (x_1 - x_2) = 2$. The normal vector of the hyperplane is $\frac{w}{||w||}$. Thus, the distance between the canonical hyperplanes is equal to the projection of $(x_1 - x_2)$ onto the normal vector of the hyperplane. This gives us $(x_1 - x_2) \cdot \frac{w}{||w||} = \frac{2}{||w||}$. The margin is half of that so $\gamma = \frac{1}{||w||}$. The first feature of the upper bound of the generalization error is obtained by maximizing the margin and thus minimizing $\frac{1}{2}||w||$ with constraints

$$y_i(w \cdot x_i + b) \geq 1, \forall i \tag{3.2}$$

This can be solved, for example, with Lagrange multipliers. The result is:

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \tag{3.3}$$

which must be maximized with respect to the α_i subject to the constraints $\alpha_i \geq 0$ and $\sum_{i=1}^{m} \alpha_i y_i = 0$ Equation 3.3 is an example of duality, where α_i and α_j are the Lagrange multipliers and the data points are in inner product.

The second feature of the upper bound of the generalization error (the bound is independent of the dimensionality of the space) can be used in a following way. The data points can be alternatively represented by mapping them to a different dimension: $x_i \cdot x_j \to \phi(x_i) \cdot \phi(x_j)$ where ϕ is the mapping function. The mapping is done, because the data might not be linearly separable in the input space. [8]

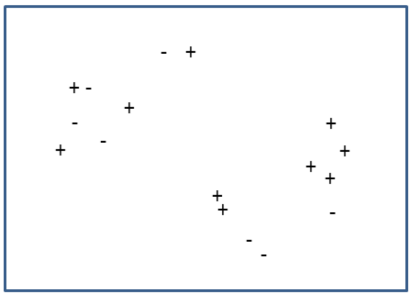


Figure 3.2: A dataset, which can not be separated linearly. However, by adding additional dimension, for example, by moving the plus-sign data forward, towards the viewer, the dataset is linearly separable.

Most real-life problems are linearly non-separable. This means that there are no linear equation that separates the classes perfectly. This is illustrated in 3.2. If a linear model is used, then one must tolerate misclassifications. To separate the classes perfectly, the input space must be mapped to a higher dimension. The dimension can also be infinite. The mapping is done by using a kernel substitution. It enables us to work in large dimensional feature spaces without having to do explicit computations in this space. The computations are done in another space after the use of kernel substitution. [25]

Also, the second feature of the upper bound of the generalization error indicates that there is no loss of performance when the mapping to the higher dimension is executed. How is the ϕ then found? It does not need to be as it is implicitly defined by the choice of kernel: $K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$. If the dataset is not separable with the choice of a linear kernel $K(x_i, x_j) = x_i \cdot x_j$, we could choose, for example, a Gaussian kernel:

$$K(x_i, x_j) = e^{-(x_i - x_j)^2/2\sigma^2}$$

Many kernel choices are available, such as a polynomial kernel: $K(x_i, x_j) = (x_i \cdot x_j + 1)^d$ or a feedforward neural network: $K(x_i, x_j) = \tanh(\beta x_i \cdot x_j + b)$. With the kernel substitution, the equation 3.3 becomes:

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{3.4}$$

with the same constraints as 3.3. The bias can be found by noting that:

$$\min_{i|y_i=+1}(w \cdot x_i + b) = \min_{i|y_i=+1}(\sum_{j=1}^{m} \alpha_j y_j K(x_i, x_j)) + b = 1$$

for $y_i = 1$ and similarly for $y_i = -1$. The bias then becomes:

$$b = -\frac{1}{2}(\max_{i|y_i=-1}(\sum_{j=1}^{m} \alpha_j y_j K(x_i, x_j)) + \min_{i|y_i=+1}(\sum_{j=1}^{m} \alpha_j y_j K(x_i, x_j)))$$

A binary SVM can then be constructed by placing the data into 3.4, maximizing $W(\alpha)$, and calculating bias from the optimal values of α_i. Thus, the class of an unseen value is then based on the sign of: $\phi(z) = \sum_{i=1}^{m} \alpha_i^* y_i K(x_i, z) + b^*$ [8]

The linear SVM equation is an example of quadratic programming. A useful fact is that the solution is unique and globally optimal. The reason for this is, because the matrix related to the quadratic term in α is positive definite or positive semidefinite. However, the solution might be unique, but α not, thus there may exist equivalent expressions of w, which require fewer support vectors. [25]

The above strategy works for binary class problems. For multiclass classification, the main idea is to reduce the problem to multiple binary class problems. There are three main strategies. First, if the number of classes is small, a directed acyclic graph will be constructed with a pair of classes in each node. For example, if there are three classes, first the prediction is done by the decision of labeling the novel data point into the class one or the class three. Then the result is compared to the class two, and the novel data point is then classified. Second strategy is called a series of one-against-all classifications, where each class is used one by one as a positive example, while the rest of the classes are negative examples. The third strategy is to construct a boundary around the normal data so that a novel data point will fall outside the boundary.

SVM can also be used in prediction of real valued numbers, where the method becomes support vector regression (SVR). We will start with the linear function of the form:

$$g(x_i) = w^T x_i \tag{3.5}$$

where w are the weights and x_i are the data points. To obtain the values for w, a loss function need to be minimized. Here, the loss function is a quadratic error term:

$$L(w) = \frac{1}{2}\sum_{i=1}^{m}(y_i - g(x_i))^2 \tag{3.6}$$

which is not to be confused with a Lagrangian. One must be aware that an overfit might occur. The overfitting can be handled by penalizing $w^T w$ with a regularization term. Thus the equation 3.6 becomes:

$$L(w, \lambda) = \frac{1}{2}\sum_{i=1}^{m}(y_i - w^T x_i)^2 + \frac{1}{2}\lambda w^T w \tag{3.7}$$

The solution of the minimization of 3.7 is:

$$w = (\sum_{i=1}^{m}(x_i x_i^T) + \lambda I)^{-1}(\sum_{j=1}^{m}(x_j y_j))$$

Typically λ is found by a cross-validation. With kernel substitution, the weight becomes:

$$\max_{\alpha_i,\lambda}(W = -\lambda^2\sum_{i=1}^{m}(\alpha_i^2) + 2\lambda\sum_{i=1}^{m}(\alpha_i y_i) - \lambda\sum_{i=1}^{m}(\alpha_i\alpha_j K(x_i, x_j)) - \lambda B^2) \tag{3.8}$$

where $\alpha_i = \frac{1}{2\lambda}\beta_i$ and β_i is a Lagrange multiplier. The minimization of 3.8 with the respect of α_i produces: $\alpha = (K + \lambda I)^{-1}y$

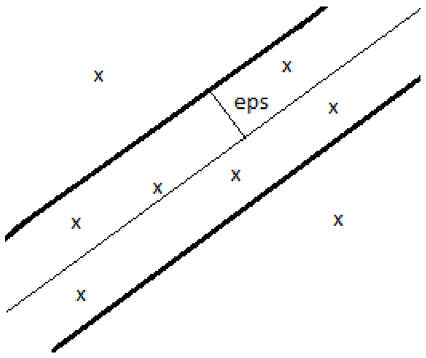


Figure 3.3: A visualization of support vector regression. eps means epsilon.

However, the above formulation is not sparse, as it uses implicitly every data point via the kernel matrix K. Sample sparsity is desired since it makes the model simpler, because we leave out data points that are not support vectors. An alternative approach is to use an $\epsilon - SV$ regression method. This method is based on statistical learning theory. Instead of the constraints 3.2, we will use $y_i - w^T x_i - b \leq \epsilon$ and $w^T x_i + b - y_i \leq \epsilon$. The ϵ is for allowing some deviation between the targets for the outputs and the function $g(x)$. This can be visualized as a tube around the hypothesis function $g(x)$, as seen in figure 3.3, and any points that will not fall inside the tube are called training errors. The function modelling the data in $\epsilon - SV$ regression is defined as:

$$g(z) = \sum_{i=1}^{m} (\alpha_i^* - \widehat{\alpha_j^*}) K(x_i, z) + b^*$$

Thus, support vector machines are capable to both classify and perform regression. [8]

3.2 Multilayer perceptrons

Multilayer perceptron is a special case of artificial neural networks (ANN). I will have a few words about artificial neural networks overall, before going into the details of multilayer perceptrons. ANNs are a computational model that are analogous to the central nervous system of a biological entity (for example a human). A unit in biological nervous system is called a neuron 3.4. The neuron consists of dendrites, which are the input for the neuron, axon, which transfers information, and pre-synaptic region, which connects to other neurons to form synapses. In ANN, the units are also neurons 3.5. The dendrite analogy for ANN neurons is called weights or input and the pre-synaptic region is simply called output.

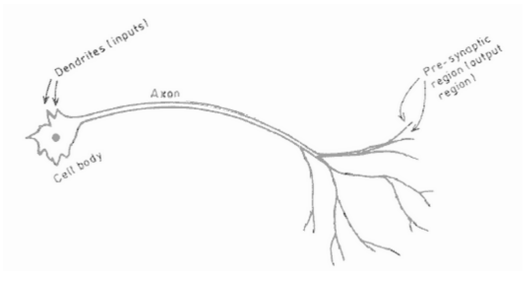


Figure 3.4: Illustration of a biological neuron [14]

The ANNs allow using very simple computational operations to solve complex problems. Also, ANNs have a self-organizing feature, that adapts ANNs to a wide range of problems. High computational parallelity is also a feature of ANNs. The ANNs are not limited to one interpretation, as there

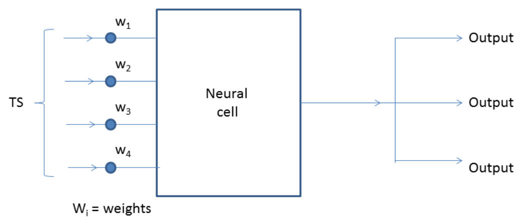


Figure 3.5: Illustration of an artificial neuron

exists different structures of ANNs. The perceptron, the artron, the adaline, and the madaline are some of the popular structures. Here, we will focus on the perceptron and its back propagation utilizing version, the multilayer perceptron.

The perceptron is a neuron in an ANN, that has

$$z = \sum_i w_i x_i$$

as an input for the neuron and

$$y = f_N(z)$$

as output from the neuron. This is illustrated in the figure 3.6.

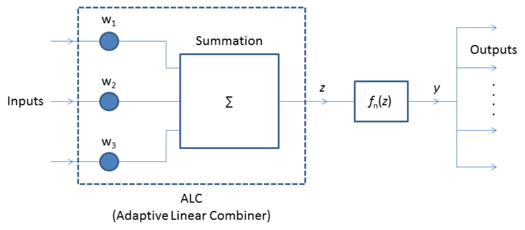


Figure 3.6: Perceptron

First, the features of the data is multiplied by certain weights. Second, the multiplications are summed, and the result is given to the neuron. Finally, the neuron executes a function, called an activation function, and the result of the execution is the output of the perceptron. Two of the most common functions used in the perceptron are:

$$f(z_i) = \frac{1}{1 + e^{-z_i}}$$

and

$$f(z_i) = \frac{1+\tanh(z_i)}{2} = \frac{1}{1-e^{-2z_i}}$$

However the perceptron, or the single-layer perceptron has limitations. For example, it can not solve a two-state exclusive-or (XOR) problem, where two inputs of the same values output to zero and two inputs of different values output to one. This is, because that type of problem is not linearly separable. There is an extension to the perceptron, however, that can solve the above problem, for example, and that is the multilayer perceptron.

The multilayer perceptron differs from the perceptron, or single-layer perceptron, in that it has one or more hidden layers of neurons (see figure 3.7). These neurons are like single-layer perceptrons themselves. The multilayer perceptron has two phases of execution, which are performed pairwise in iterations, until a certain convergence has been reached. First, the procedure is similar to the single-layer perceptron. The inputs are weighed, summed, ran through a function, and given to the first hidden layer. This procedure is continued between the hidden layers towards the output of the neuron. Finally, the output is calculated by the last hidden layer. The second phase of the execution of the multilayer perceptron is called backpropagation. The point is to calculate the difference of the output and the desired value and propagate the value back to the input layer, where new weights are calculated based on the difference. The formula that the multilayer perceptrons base the calculations of the difference is:

$$\epsilon = \frac{1}{2}\sum_k (d_k - y_k)^2$$

where d_k is the desired value and y_k is the output. The procedure of the backpropagation goes as follows (the derivations of some of the equations are left out). First, the difference of weights of the output layer is calculated as:

$$\triangle w_{kj}(p) = \eta \Phi_k(p) y_j(p-1)$$

where η is learning rate (a number between zero and one) and

$$\Phi_k = y_k(1-y_k)(d_k - y_k)$$

Second,

$$\triangle w_{ji}(r) = \eta \Phi_j(r) y_i(r-1)$$

is computed for hidden layers, where r denotes the hidden layer and

$$\Phi_j(r) = y_j(r)[1-y_j(r)] \sum_k \Phi_k(r+1) w_{kj}(r+1)$$

Third,

$$w_{kj}(m+1) = w_{kj}(m) + \triangle w_{kj}(m)$$

is computed to update the weights for the iteration m.

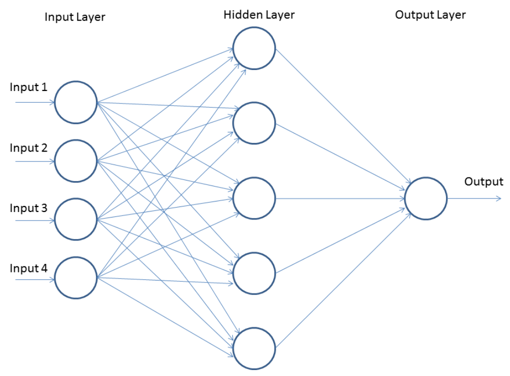


Figure 3.7: Multilayer perceptron

In classification, the multilayer perceptron uses the aforementioned activation functions (the sigmoid) to determine the class. As the function results in two options: a value over or under 0.5, the classifier is binary. Same techniques, as described in the chapter 3.1, can be used in multiclass problems. In regression, the activation functions of the hidden layers are usually same as in the classification, but the output layer uses a linear function. [14]

3.3 Random forests

Before going into the details of random forests, let us first look classification and regression trees (CARTs) and decision trees. CART is an umbrella term to include every treelike model for classification and regression problems. A tree is a data structure and a special type of graph. It is made of a collection of nodes connected with edges in a hierarchical way. It is an acyclic graph whose nodes have only one incoming edge. In classification and regression, a tree is used to divide a complex problem into many smaller ones. The division is done by making series of tests using the input data, until a class or a real number is determined. In the series, the next test to be performed is determined by the result of the previous tests. This kind of structure is called a decision tree. The figure 3.8 is an illustration of a decision tree, where it is determined, whether certain picture is taken from indoors or outdoors.

Thus, a decision tree has a data as an input, tests about the data features (selected by hand or automatically) as internal nodes, and a class (classifi-

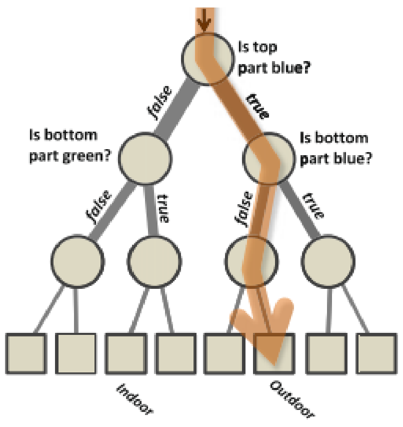


Figure 3.8: A decision tree [11]

cation) or an equation (regression) as leaf nodes. A split function is another name for a single test. Each internal node has an associated split function. The function is defined as:

$$h(v, \theta_j) : \mathbb{R}^d \times T \rightarrow \{0, 1\},$$

where $\theta_j \in T$ are the split parameters and T is the space of the split parameters. This is in the case the decision tree is binary, that is, every internal node and the root has a maximum of two child nodes. The training is then done by finding optimal values for θ_j, that is:

$$\theta_j = \max_{\theta_j \in T} I(S_j, \theta),$$

where S_j means the subset of training points (items in the training set) reaching the node j and I is an objective function and its definition depends on, for example, whether the problem is a classification or a regression. Further, the I can be divided as

$$S_j^L(S_j, \theta) = \{(v, \cdot) \in S_j | h(v, \theta) = 0\}$$
$$S_j^R(S_j, \theta) = \{(v, \cdot) \in S_j | h(v, \theta) = 1\},$$

where the superscripts L and R means the subset of training points going to the left and to the right respectively. The $\cdot$ is the classes or real values. One common way to define I is of the concepts of information gain and entropy. Thus, in that way, I becomes information gain or:

$$I = H(S) - \sum_{i \in [L,R]} \frac{|S^i|}{|S|} H(S^i) \tag{3.9}$$

Here $H(S)$ is entropy. Information gain means, in the context of decision trees, the reduction in uncertainty after splitting the training data in a node. $H(S)$, or entropy, is a measure for the uncertainty. It is defined as (Shannon entropy):

$$H(S) = -\sum_{c \in C}(p(c)log(p(c))),$$

where c is class label and $p(c)$ is the empirical distribution extracted from the training points.

The decision trees, as predictor, are prone to overfit. This issue can be minimized, however, by generating multiple different decision trees on the input data and averaging their prediction results. This combination is called a random forest (synonyms: a random decision forest, a decision forest). It is called random, because randomness is involved in creating the individual trees. Two popular methods to apply randomness are called bagging and randomized node optimization. The former means to give different trees different random samples from the input data as training data. The latter method gives each node different random sample from the parameter space in the node optimization. The θ thus becomes:

$$\theta_j = \max_{\theta_j \in T_j} I(S_j, \theta)$$

As the parameter space might be infinite, we introduce new parameter: $\rho \in \{1, ..., T\}$. The new parameter indicates the degree of randomness in the forest. The closer ρ is to one, the more random the forest is. Also, the more random the forest is, the more uncorrelated the individual trees are to each other. This is shown in the figure 3.9.

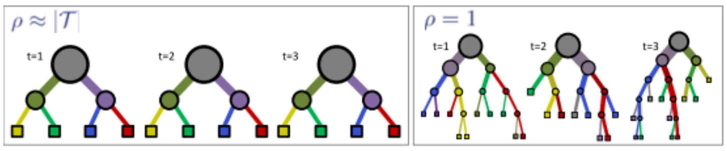


Figure 3.9: Impact of forest randomness [11]

The main parameters for a forest are:

- the maximum depth D
- the amount of randomness ρ
- the forest size T

- the choice of the model for the split functions
- the training objective function I
- the choice of features in a practical application

As regression is more relevant than classification in this thesis, we will look more closely on the regression forest. The input data used here is one dimensional and one regression tree in the forest looks like in the figure 3.10.

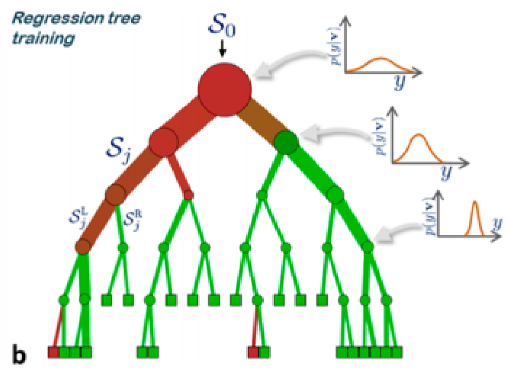


Figure 3.10: A regression tree [11]

After each split, the entropy decreases, which is indicated by thinner edge and when the entropy decreases, the confidence level increases, which is portrayed by the sharper form of the distribution of the possible values. The actual prediction is done in the leaf nodes and each leaf node has a certain prediction model. The model can be, for example, a polynomial or a linear function. This can be seen in figure 3.11

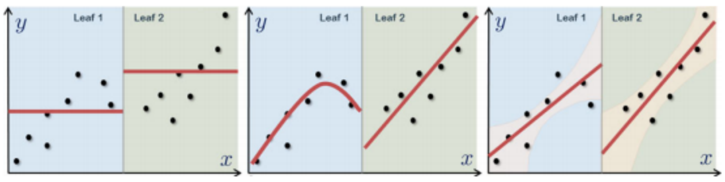


Figure 3.11: Different regression tree prediction models [11]

Each prediction has also a certain confidence level. The prediction output of a regression forest is an average of the prediction output of the individual trees and is defined as:

$$p(y|v) = \frac{1}{T}\sum_t (p_t(y|v)$$

The training objective function is derived from 3.9. The entropy of a regression forest is defined as:

$$H(S) = -\frac{1}{|S|}\sum_{x\in S}\int p(y|x)log(p(y|x))\,\mathrm{d}y$$

Here, the conditional probability is modeled as a normal distribution:

$$p(y|x) = N(y; \bar{y}(x), \sigma_y^2(x))$$

Finally, with the use of linear function $\bar{y}(x)$ and because we use one dimensional training data, the training objective function becomes:

$$I(S_j, \theta) \propto \sum_{x\in S_j}(log(\sigma_y(x)) - \sum_{i\in\{L,R\}}(\sum_{x\in S_j^i}(log(\sigma_y(x))))$$

[11]

3.4 Linear regression

Regression analysis can be viewed as a method for finding relationships, or more specifically causal relationships, from a variable or variables to other variables. There are response variables or dependent variables whose values are to be predicted (analogical to classes in classification problems) and there are predictors or individual variables whose relationship to the predicted variables are to be discovered. The regression analysis can be executed by using either the simple linear regression, the multiple linear regression, or the nonlinear regression. The first one, as its name suggests, is the simplest one. It is used to find the relationships between two variables. The simple linear regression is modeled by:

$$y = \beta_0 + \beta_1 x + \epsilon$$

where y is the dependent variable, x is the independent variable, β_0 is the y intercept, β_1 is the slope of the regression line, and ϵ is a random error. Usually, it is assumed that the ϵ is normally distributed. As an simple example, let us look the table 3.1 and its the simple linear regression equation:

Table 3.1: Example data for linear regression

Parent	64.5	65.5	66.5	67.5	68.5	69.5	70.5	71.5	72.5
Children	65.8	66.7	67.2	67.6	68.2	68.9	69.5	69.9	72.2

$$child\, height = 21.52 + 0.69 * parent\, height$$

which mimics the simple linear regression model. The simple linear regression is a special case of the multiple linear regression. The latter has one dependent variable and more than one independent variables. The multiple linear regression model is defined as:

$$y = \beta_0 + \beta_1 x + \cdots + \beta_p x_p + \epsilon$$

where the betas are the regression coefficients. Usually, the multiple linear regression is modeled in a matrix form, that is:

$$y = X\beta + \epsilon$$

Here, y, β, and ϵ are vectors, while X is a matrix of the form:

$$M = \begin{pmatrix} x_{1,1} & \cdots & x_{1,p} \\ x_{2,1} & \cdots & x_{2,p} \\ \vdots & \ddots & \vdots \\ x_{n,1} & \cdots & x_{n,p} \end{pmatrix}$$

The independent variables can also be polynomial, or a parameter to a certain function. Thus, these are valid multiple linear regression models:

$$y = \beta_0 + \beta_1 x + \beta_2 x_1^2 + \beta_3 x_2 + \epsilon$$

and

$$y = \beta_0 + \beta_1 x + \beta_2 \ln x_1 + \beta_3 \ln x_2 + \epsilon$$

The third way of executing the regression analysis is by using the nonlinear regression. It is defined as:

$$y = \frac{\alpha}{1 + e^{\beta t}} + \epsilon$$

where y is called as the growth of a particular organism as a function of time t, α and β are the model parameters, and ϵ is the random error. The nonlinear regression is not discussed further in this chapter, as the focus is on the linear ones. All of the regression execution methods can be concluded in the following form:

$$y = E(y) + \epsilon$$

$E(y)$ here means the mathematical expectation of the dependent variable. If $E(y)$ is a linear combination of $x_1, x_2, \cdots, x_k; k \geq 1$, the regression is linear regression. If $k = 1$ the regression is simple linear regression and multiple linear regression otherwise. If $E(y)$ is nonlinear, the regression is nonlinear. Let us now look more closely on the simple linear regression. To find β_0 and β_1, a method called least squares estimation can be used. The principle of the method is to find the estimates b_0 and b_1 such that the sum of the squared distance from the actual dependent variable and the predicted dependent variable $\hat{y} = \beta_0 + \beta_1 x_i$ reaches the minimum among all possible choices of regression coefficients β_0 and β_1. This can be formulized as:

$$(b_0, b_1) = \arg\min_{\beta_0,\beta_1} \sum_{i=1}^{n} [y_i - (\beta_0 + \beta_1 x_i)]^2$$

The least squares estimates are given by solving:

$$\frac{\partial}{\partial \beta_0} \sum_{i=1}^{n} [y_i - (\beta_0 + \beta_1 x_i)]^2 = 0$$

and

$$\frac{\partial}{\partial \beta_1} \sum_{i=1}^{n} [y_i - (\beta_0 + \beta_1 x_i)]^2 = 0$$

The solution for these are:

$$b_1 = \frac{\sum_{i=1}^{n} (y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

and

$$b_0 = \bar{y} - b_1 \bar{x}$$

In linear regression, the models are not known beforehand and they have to be found. Proposed models for solving the linear regression problems may underfit or overfit (chapter 2.1) and the aim is to find a model that is as accurate as possible but not too complex. The model is constructed by including or excluding certain independent variables. More formally, let us consider a linear model $y = X\beta + \epsilon$, where X is $n(k + 1)$ of full column rank (k + 1) and ϵ $MVN\{0, \sigma^2 * I\}$ (multivariate normal distribution). This can be written in partitioned form:

$$y = X_1\beta_1 + X_2\beta_2 + \epsilon$$

where X_1 is $n(p+1)$ of full rank and X_2 is $n(k-p)$. Here, if $X_2\beta_2$ is excluded when it should be included, underfitting occurs and vice versa for overfitting. There are two ways to find $X_1\beta_1$: all possible regressions and a stepwise selection. In the first one the point is to try every possible regression model that can be generated from the data and select the one that optimizes a model selection criterion. Possibly the simplest criterion is the sum of square errors:

$$SSE = \sum (y_i - \hat{y}_i)^2$$

which is just the sum of the squared difference of actual and predicted values. However, this criterion does not take account the complexity of the model, thus it is not much of use. Another criterion is the coefficient of determination:

$$R^2 = 1 - \frac{SSE}{SST}$$

where SST is the sum of square total $SST = \sum (y_i - \bar{y}_i)^2$. More sophisticated criteria are PRESS and GCV. The former is defined as:

$$PRESS = \sum_{i=1}^{n} \{y_i - \hat{y}_{i(-i)}\}^2$$

where $\hat{y}_{i(-i)}$ is the predicted value for the ith case based on the regression model that is fitted by excluding the ith case. GCV or generalized cross-validation criterion is defined as:

$$GCV = \frac{n}{n - (p+1)} \cdot MSE$$

and is to be used if PRESS is too expensive for calculation. All of the above are examples of asymptotically efficient criteria, where the goal is to select the model with minimum mean squared error. There are also a group called consistent criteria, where a true model is selected with a probability of one. These are Akaike information criterion (AIC) and Bayesian information criterion (BIC). The first has a general form of:

$$AIC = n * log - likelihood + 2 * number\, of\, parameters$$

or

$$AIC \simeq n \cdot \log(SSE) + 2p$$

AIC penalizes the complexity of the regression model with the number of parameters. BIC is a slightly modified version of AIC:

$$BIC \simeq n \cdot \log(SSE) + \log(n)p$$

If one criterion is to be chosen, it would probably be BIC as it is effective and suitable for large samples. These were one way to find the aforementioned $X_1\beta_1$. The other way is with a stepwise selection. It is basically just a choice between feature selection and feature extraction, which are explained in 2.1. [31]

Chapter 4

Text mining

Textual data is one of the most common form of data produced by humans. Text is often considered to be unstructured data as in the context of how data is usually represented: rows of feature vectors. However, text has linguistic structure intended for humans. Text as data is often “dirty” or inconsistent. The words might be misspelled, the punctuation might be random, or the words might be ran together. Even flawless text might contain synonyms and it is tied to a certain context. Thus, understanding the context is crucial for understanding the text, or use it as a data. A lot of preprocessing has to be done to the text to use it as an input for machine learning or data mining methods. The process of gaining and using information from textual data is called text mining. [22]

The primary goal of text mining is to discover patterns, trends, and outliers by analyzing textual information. A close relative to text mining is information retrieval that focuses on accessing information. [2]

The concepts used both in text mining and information retrieval are document, tokens and terms, and corpus. A document is a string of text that can only be one sentence or a hundred page long report. The document consists of tokens or terms, which are basically the words in the document. A collection of documents forms a corpus. [22]

The main feature of textual data is that it is sparse and high dimensional. For example, a corpus could contain thousands of documents with a total of 100 000 different words, but one document might contain only a fraction of words from that 100 000 word pool. That corpus can be represented as a term-document matrix of the size $n * d$, where n is the number of documents and d is the size of the lexicon. The (i, j)th entry would then be the normalized frequency of jth word in the lexicon in document i. [2]

There are multiple approach for representing data in text mining of which bag of words is one. The approach treats each word in a document as an important keyword, collects them, while ignoring grammar, word order, and punctuation. After collecting the words, each document is represented by either one or zero depending on does a document contain a certain word. An improved approach is to count how many times a word or term is used in a document.

Usually the case of the terms is normalized by converting each term to a lowercase. Second, the terms are stemmed, meaning that the suffixes are removed and the terms are in singular form. Finally, stop words, or words that are very common and have little information value, such as the, or, of, etc. are removed.

In addition to the term frequency, or TF, another measurement is inverse document frequency (IDF), or how common a term is in the entire corpus. IDF of a term is calculated as:

$$IDF(t) = 1 + \log \frac{\text{Total number of documents}}{\text{Number of documents containing t}}$$

Term frequency and inverse document frequency can also be combined to form TFIDF, which is calculated by:

$$TFIDF(t, d) = TF(t, d) * IDF(t)$$

Thus, each document becomes a feature vector that can be given as an input for machine learning or data mining algorithms. [22]

In the next few paragraphs the key problems of text mining are explained briefly.

Information, such as entities and their relations, extraction from textual data can reveal more information about the text than a simple bag-of-words model. It is usually needed to draw conclusions about the knowledge of the text.

It could be beneficial to abbreviate large texts or set of texts to get the essential information. This text mining problem is called text summarization.

Unsupervised machine learning methods are discussed briefly in 2.2.2. In text mining they are used mainly in topic modeling in a way that each cluster represents a topic. Usually soft clustering is used in order to give documents probabilities to belong to a certain cluster.

In textual data the dimensionality might be too large for learning methods, thus dimensionality reduction is then needed. A common method used for textual data is called latent semantic indexing. The method reveals the key semantic aspects of the text, making the data more suitable for mining applications.

There might be a case, when the training data in one domain might not be enough. For example, on some problem there might be an abundance of training examples in English but nearly zero examples in Chinese. A method to mitigate this is called transfer learning. The goal is to transfer the knowledge from one domain to another.

Many recent web applications, such as social networks and news streams, create a huge amount of textual data or continuous textual streams. This kind of data need to be processed in the context of a one-pass constraint. This means that it is difficult to process and store the data for offline processing and it is necessary to perform online mining. [2]

One of the key problems of text mining is extracting opinions and sentiments from text. This is discussed further in the next section.

4.1 Sentiment analysis

While the textual mining and information retrieval focuses mainly on objective facts, sentiment analysis or opinion mining focuses more on the subjective side such as opinions, emotions, and sentiments. [23]

The purpose of sentiment analysis is to identify the valuation that an opinion holder puts to a certain entity such as an object, a state, an event, or a topic. This information can be used to analyze different emotional reactions that, for example, certain product or service of a company creates. In addition for example in finance, sentimental analysis is essential with the company of fundamental and technical analysis. The emotional reactions of people and their rumors about certain entities (through textual communication) might predict or reveal new events or behaviours. [18]

A crucial component in sentiment analysis, an opinion, can be defined as either positive or negative attitude, view, or emotion about certain entity (for example, product or person) from a user. Apart from the textual representation, an opinion can also be defined more formally as a five-tuple:

$$\text{Opinion} = (e_j, a_{jk}, so_{ijkl}, h_i, t_l),$$

where e is entity (the target of an opinion), a is a feature of the entity, so is the numerical sentiment value, h is the opinion holder (the user), and t is the time of the emergence of the opinion.

Opinions can be grouped into regular and comparative opinions. The regular opinions can further be divided into direct and indirect opinions. Direct opinions are, as its name suggests, direct opinions about an entity, whereas indirect opinions express an opinion on an entity based on the effects on other entities.

The following tasks are directly linked to the sentiment analysis. The tasks might contain overlapping elements but they are: sentiment classification, subjectivity classification, opinion summarization, opinion retrieval, and detecting sarcasm and irony.

Sentiments of opinions are classified into at least three groups: positive, neutral, or negative. The division is not scale dependent and it does not need to be discreet, as it can be a range, for example, from -1 to 1, or absolute negative to absolute positive. This task might be complex, when multiple domains or languages are present.

Subjectivity classification determines, whether a textual piece (for example a sentence) is subjective or objective. Factual texts should be objective and for example blogs or other opinionated texts should be subjective. This task might precede the sentiment classification.

Opinion summarization focuses on extracting the main features of an entity shared in one or multiple documents. The features can be, for example, changes in the sentiment orientation or links between different entities. The sentences are then grouped into the features.

Opinion retrieval is similar to information retrieval in that it tries to return relevant opinions to a given query. There are two scores to be measured in each document: the relevance, or how close the document is the query in the respect of, for example, cosine similarity, and the opinion, or how close the opinion measure of the query is to the document.

The sarcasm and irony detection is currently under development and in the absence of common agreement between researchers.

Sentiment analysis techniques can be grouped as in the figure 4.1.

Major machine learning techniques (except probabilistic methods) themselves are discussed in more detail in section 3. Some of the most important features for the input to the machine learning methods are: terms and their frequency (TF), part of speech information (are the terms adjectives or nouns), how negations change meanings, and syntactic dependencies.

Lexicon-based approaches use sentiment lexicons, that are compiled lists of terms, idioms, and phrases. They also use more complex structures like ontologies or dictionaries. Lexicon-based approaches are further divided by two groups: dictionary-based and corpus-based. The former uses a set of terms without a context and the latter uses dictionaries with specific context.

Natural language processing or NLP is a field in artificial intelligence research, where the long term aim is to teach computers to understand human languages. [15]

Sentiment analysis is a restricted NLP problem, as it is only necessary to understand the positive and the negative sentiments on each sentence. [23]

For being an NLP problem, there are different levels of analysis. First, there is document level, which considers a whole document as an opinion of an entity. Second, there is sentence level, that is used if a single document

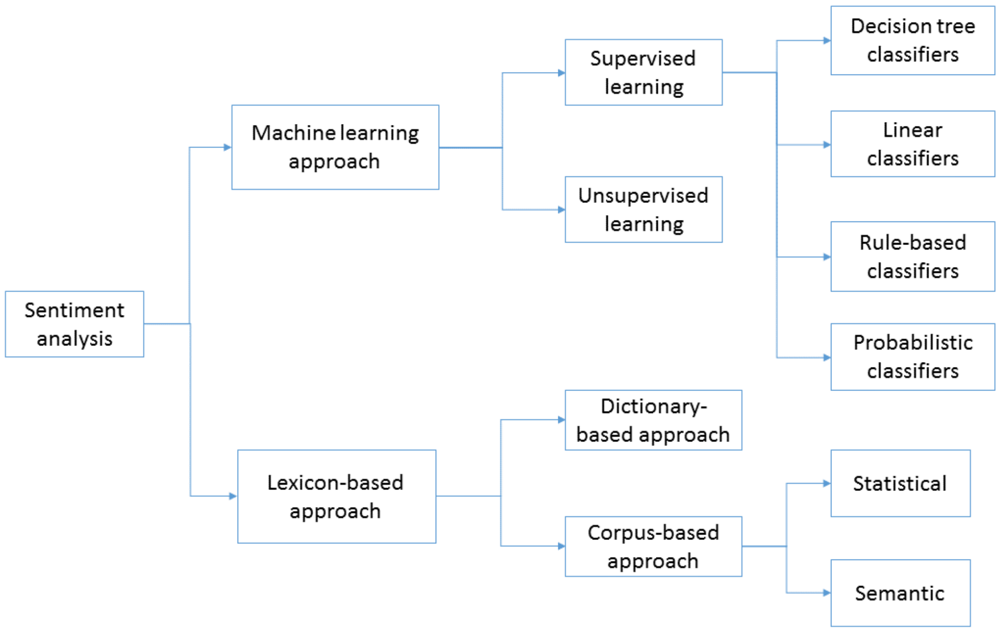


Figure 4.1: Classification techniques in sentiment analysis

contains multiple entities with different opinions. Aforementioned subjectivity classification is a closely related task to be done in sentence level. Finally, there is entity/aspect level, which considers a target where an opinion holder expresses a positive or a negative opinion. [23]

Chapter 5

Sales Predictor

The point of the program, named Sales Predictor, is to see which of the four machine learning methods (SVM, MLP, RF, or LR) gives the most accurate numerical prediction results in the context of car sales. The point is also to see, whether a sentiment analysis of social media makes the predictions more accurate. The machine learning methods are not manually programmed, as Sales Predictor uses Waikato Environment for Knowledge Analysis, or Weka, 3.7.12 data mining software for Java for the use of the methods [30]. Thus, Sales Predictor is also written in Java.

The development of the program started with learning Weka and its architecture. In this process, a simple classification exercises were made and with it the use of Attribute-Relation File Format (ARFF) files. Sales Predictor modeled the regression problem as a classification problem prior to the knowledge, that a time series package existed. First, the training data was rows of three consecutive months of sales data (monthly sales of Toyota car manufacturer) followed by a class, which was a range of the sales amount in which the sales of the fourth month fall (0 to 249, 250 to 500, etc.). For example, two rows of that data looked like (in textual representation):

- January sales, February sales, March sales, A range for April sales
- February sales, March sales, April sales, A range for May sales

This with 10-fold cross-validation gave about 70 percent accuracy. This procedure seemed close to an artificial one and there had to be an alternative way. A plugin package called WekaForecaster was an answer to make the procedure more natural. With WekaForecaster came the concept of time series and with that, the prediction problem could be modeled as a true regression problem. Thus, the aforementioned range classes were dropped. The ARFF file format was also dropped, as the data was to be stored in an SQL database. For more information about the data itself, see the next section [5.1].

Next, let us look what parts the program does have. Here is a list of the main classes and their purposes of the program:
Main:

- ties parts together
- retrieves data
- orders predictions
- saves results

Sales Predictor:

- executes actual predictions through WekaForecaster

SalesPredictorDAO:

- data access object between the program and the database
- creates database connection
- executes SQL queries

The next three classes are used to represent the data the program uses.
DataLabel:

- a label for each data item
- holds the product id, the name, and the lag value (specified in the data chapter)

DataForPrediction:

- a class to convey data for Sales Predictor
- ties a data label with the training and the overlay data (specified in the data chapter)

PredictionResult:

- a class to hold the results of a prediction
- holds evaluation statistics such as root mean square error etc.
- holds RROC curve

Finally, there is a Utitilies class, which contains mainly functions to calculate evaluation statistics for predictions.
A basic run of the program goes as follows:

1. Open up the database connections
2. Retrieve data for predictions
3. Execute the predictions
4. Retrieve actual sales
5. Create prediction result groups (more on this later)
6. Calculate evaluation statistics
7. Print results
8. Close database connections

Next we will look a bit of how linear regression, multilayer perceptron, random forest, and support vector regressor are used in Weka and their major option selections. Sales Predictor uses the default values of the options in the methods.

Linear regression in Weka uses basic multiple variable linear regression models. The model is selected using Akaike information criterion (see chapter 3.4).
Here are several important options for linear regression in Weka [26] [5]:

- selection method, default uses ML5, which builds the LR model initially with every attribute and then removes the attributes with the smallest standardized regression coefficients, until no improvement is made for the estimated average prediction error by Akaike information criterion.
- whether or not collinear attributes are tried to be eliminated, default is try to eliminate
- ridge parameter, or how the model handles collinearity, default is 10^{-8}

Multilayer perceptron in Weka is a backpropagation artificial neural network. The output of the nodes are unthresholded linear units.
Here are several important options for multilayer perceptron in Weka [27]:

- learning rate between 0 and 1, default is 0.3

- momentum rate between 0 and 1, the higher the value, the lower the probability to stay in a local minimum, default is 0.2
- number of epochs, or how many iterations are made before the training stops, default is 500

Random forest in Weka is similar to the one explained in the chapter 2. The method uses bagging in the sample selection. The trees in the forest are of the class RandomTree of Weka.
Here are several important options for random forest in Weka [28]:

- number of trees, default is 100
- number of features, if lower than 1, $\log M + 1$ is used, default is 0
- maximum depth of the trees, default is 0, which means unlimited depth

Support vector regressor in Weka uses sequential minimal optimization for training the regressor (more information in [21]). Every attribute is normalized, and multi-class problems are handled with pairwise 1 versus 1.
Here are several important options for support vector regression in Weka [29]:

- complexity, default is 1
- tolerance, default is 10^-3

Thus, the program evolved from solving a classification problem into a one that predicts time series with the use of Weka machine learning methods. In the next section we will look the data that the program uses.

5.1 The data

This section goes through in more detail the data that Sales Predictor uses. In the program, three database tables are used, which are defined as follows:
products:

- id: the primary key
- name: the name of the product
- type: a type or a class of the product, for example, 0 means car manufacturer

This table holds data for each of the products. In this thesis, there are only one type of products and that is the car manufacturers, for example, Toyota, Volkswagen, and Ford.

An example row in products could look like:

id	name	type
1	Toyoyta	0

sales amounts:

- id: the primary key
- fk products id: a reference to a product
- begin timestamp: the start time of a sales time window, usually a month
- end timestamp: the end time of a sales time window, usually a month
- sold quantity: how many cars were sold in the time window

This table holds data for the sales amounts of each of the products in a certain time window. The time window is one month and it started in January 2013. Thus, sales amounts is basically the monthly sales of each product.

An example row in sales amounts could look like:

id	fk products id	begin timestamp	end timestamp	sold quantity
1	1	2013-01-01 00:00:00	2013-02-01 00:00:00	1234

some data (social media data):

- id: the primary key
- fk products id: a reference to a product
- mean: mean value of the sentiments
- median: median value of the sentiments
- total count: how many times a car manufacturer is mentioned in social media
- negative count: how many of the total count is of negative sentiment

- neutral count: how many of the total count is of neutral sentiment
- positive count: how many of the total count is of positive sentiment
- begin timestamp (bts): the start time of a social media discussion time window, usually a month
- end timestamp (ets): the end time of a social media discussion time window, usually a month
- type: same as in mlsp products
- purchase intent count: the amount of an actual intent to buy a car of a certain manufacturer

This table holds the sentimental analysis data about the discussions of a certain product in the social media.

An example row in some data could look like (tmstmp is a timestamp value, e.g. 2013-01-01 00:00:00):

id	fk pid	mean	med	total	neg	neut	pos	beg	end	type	pic
1	1	6.2	5.0	11	3	3	5	bts	ets	0	2

The data for products table and sales amounts table are extracted from the registration data of Autoalan tiedotuskeskus [4]. The data is in Excel format and, thus, must be preprocessed. It was decided, that the car manufacturers were sufficient rather than elaborate them with the different car models. After the preprocessing, the data is converted into SQL statements and fed to the database tables.

One of the special features of the relation between car sales data and the social media sentiment analysis (SMSA) data is that there is always some amount of lag. This means that there is a certain amount of time between a discussion of a certain car model in the social media and the actual purchasing moment. The lag in Sales Predictor is an integer representing a month. This value is used as an offset to match the social media data with a later sales data.

Sales Predictor uses also overlay data in predictions. The overlay data is data that is used in the predictions but is not itself predicted. In this program the overlay data consists of:

- mean: mean value of the sentiments
- median: median value of the sentiments

- total count: how many times a car manufacturer is mentioned in social media
- negative count: how many of the total count is of negative sentiment
- neutral count: how many of the total count is of neutral sentiment
- positive count: how many of the total count is of positive sentiment
- purchase intent count: how many times a car manufacturer is mentioned in an intent to purchase

The sentiments are categorized in eleven different values. Sentiments 0 to 4 are called negative sentiments, 5 is called a neutral sentiment, and 6 to 10 are called positive sentiments.

The data, thus, consists of previous car sales amounts and social media sentiment analysis data. The section goes into more detail about the experimentation itself.

5.2 Experimentation setup

The experimentation was conducted in two parts. First, it was seen that including SMSA data into the prediction procedure actually made the predictions more accurate. Second, it was seen which machine learning method produced the most accurate predictions. For each prediction, or sales amount case, nine next values are predicted, meaning the prediction time window is nine months. The value nine is arbitrary.

In the second part, a concept of shift was used (definition in chapter 2). The process goes in following way. First, an error array is constructed in each case from the predictions (simply: error = predicted − actual). Second, the opposite number of each element in the error array serves as a shift to be added to the predictions. Finally, evaluation metrics are evaluated for each prediction with a shift applied. The shift with the lowest root mean squared error is kept as the final result for a prediction, thus generating an optimum shift.

The predictions would normally be made to each car manufacturer with every method and every lag value. There are about thirty car manufacturers, four methods, and four lag values (3, 6, 9, and 12 months). Totally, this would mean $30 * 4 * 4 = 480$ different prediction scenarios. To make the scenario pool more manageable for the purpose of this thesis, the car manufacturers are divided into two groups: high sales and low sales. The dividing point is the average of the overall sales. Thus, there is then $2 * 4 * 4 = 32$ different prediction scenarios.

Here are the definitions of the evaluation metrics used for evaluating the results [20]. N means the amount of predicted values.
Mean average error:

$$MAE = \frac{\sum(|\text{predicted} - \text{actual}|)}{N}$$

Mean squared error:

$$MSE = \frac{\sum(|\text{predicted} - \text{actual}|^2)}{N}$$

Root mean squared error:

$$RMSE = \sqrt{\frac{\sum(|\text{predicted} - \text{actual}|^2)}{N}}$$

Mean error bias:

$$MEB = \frac{\sum(\text{predicted} - \text{actual})}{N}$$

Direction of accuracy (ac, ap: actual current/previous; pc, pp: predicted current/previous):

$$DoA = \frac{count(sign(\text{av} - \text{ap}) == sign(\text{pc} - \text{pp}))}{N}$$

Area over (RROC) curve:

$$AOC = MSE - MEB^2$$

5.3 Results

The results consist of the values of the evaluation metrics defined in the previous section and RROC curves for the different machine learning methods. First, there are the results for the question, does social media sentiment analysis data make the predictions more accurate than without it. Table 5.1 shows how random forest performs with and without the sentiment analysis data. The reason for the choice of random forest is explained in a while. Unknown abbreviations in the table 5.1:

- Maj: major sales

Table 5.1: Evaluation metrics of random forest to see if SMSA data increases prediction accuracy.

Case	Lag	Method	RMSE S	DoA S	AOC S	RMSE	DoA	AOC
Maj	3	RF	121,33	0,64	14655	127,82	0,67	16337
Maj	6	RF	121,36	0,64	14670	131,52	0,62	17297
Maj	9	RF	121,19	0,65	14625	129,39	0,62	16735
Maj	12	RF	127,39	0,64	16174	133,31	0,63	17771
Min	3	RF	40,8	0,61	1650	41,75	0,54	1742
Min	6	RF	40,95	0,61	1661	41,54	0,62	1725
Min	9	RF	40,84	0,61	1654	42,28	0,58	1786
Min	12	RF	39,32	0,62	1546	40,54	0,58	1643

- Min: minor sales
- Lag: lag value in months
- RF: random forest
- evaluation metrics with S in the header: SMSA data used
- evaluation metrics without S in the header: SMSA data not used

In every case (major or minor sales with the lag value) the root mean squared error and the area over curve, when the SMSA data is included, is smaller than when the data is not included. This means better results. For example, RMSE of major sales of lag value 6 with SMSA data is nearly 10 percent lower than without SMSA data. Direction of accuracy is overall higher (better) with SMSA data included, although major sales of lag value 3 without SMSA data has the highest DoA.

Next, we will look the actual results of the experimentation about the comparison of the different methods. Table 5.2 shows the evaluation metrics of the major sales group and table 5.3 shows the evaluation metrics of the minor sales group. In both tables the lowest RMSE is bolded in each lag value.

Unknown abbreviations in both tables 5.2 and 5.3:

- SVR: support vector regressor
- LR: linear regression
- MLP: multilayer perceptron
- RF: random forest

Table 5.2: Evaluation metrics of different methods of major sales.

Lag	Method	MAE	MSE	RMSE	MEB	DoA	AOC
3	SVR	138,71	37894,75	194,67	31,3	0,66	36915,02
3	LR	92,92	18961,75	137,7	-2,35	0,66	18956,24
3	MLP	225,53	87133,08	295,18	-111,24	0,57	74759,81
3	RF	81,94	14720,79	**121,33**	-8,09	0,64	14655,32
6	SVR	147,21	41924,77	204,76	44,5	0,66	39944,39
6	LR	93,19	19374,96	139,19	-2,02	0,62	19370,87
6	MLP	211,75	75827,74	275,37	-76,84	0,52	69923,67
6	RF	81,95	14727,71	**121,36**	-7,54	0,64	14670,91
9	SVR	161,03	51032,76	225,9	41,46	0,62	49313,76
9	LR	96,57	19174,97	138,47	2,93	0,66	19166,38
9	MLP	183,34	62123,06	249,24	-74,76	0,54	56534,21
9	RF	81,55	14686,51	**121,19**	-7,81	0,65	14625,56
12	SVR	173,31	56264,15	237,2	-38,4	0,56	54789,61
12	LR	119,6	27082,66	164,57	11,84	0,66	26942,48
12	MLP	191,74	66803,73	258,46	-83,5	0,44	59831,49
12	RF	88,9	16227,71	**127,39**	-7,27	0,64	16174,92

Table 5.3: Evaluation metrics of different methods of minor sales.

Lag	Method	MAE	MSE	RMSE	MEB	DoA	AOC
3	SVR	47,16	4007,7	63,31	-1,69	0,51	4004,84
3	LR	31,9	2301,64	47,98	-3,08	0,53	2292,18
3	MLP	56,58	5545,87	74,47	-2,16	0,53	5541,22
3	RF	29,03	1664,52	**40,8**	-3,76	0,61	1650,37
6	SVR	46,69	4173,6	64,6	10,07	0,52	4072,29
6	LR	35,78	2682,92	51,8	-6,58	0,54	2639,62
6	MLP	46,96	3998,51	63,23	-10	0,52	3898,58
6	RF	29,12	1676,53	**40,95**	-3,87	0,61	1661,57
9	SVR	42,25	3508,05	59,23	2,1	0,55	3503,66
9	LR	35,3	2621,59	51,2	-11,87	0,54	2480,78
9	MLP	52,66	5091,75	71,36	-16,62	0,58	4815,59
9	RF	29,08	1668,14	**40,84**	-3,7	0,61	1654,44
12	SVR	55,53	5783,76	76,05	24,27	0,65	5194,78
12	LR	54,44	5932,09	77,02	2,68	0,64	5924,92
12	MLP	36,85	2580,83	50,8	15,49	0,58	2340,83
12	RF	29,28	1546,44	**39,32**	-0,56	0,62	1546,13

In addition to RMSE, DoA, and AOC, there are mean average error, mean squared error, and mean error bias. The former two are just additional information for RMSE. MEB tells whether the method overestimates or underestimates. It can be seen, that in both of the tables, the methods generally underestimate.

Let us elaborate both AOC and MEB. Figures 5.1 to 5.8 show the regression receiver operating curves for each of the cases. These show how much the methods over or underestimate. The area over the curve, which can be calculated by the equation in the previous section, tells how much the error varies. By having a lower AOC, the errors are more consistent, or have a low variance. Thus, by applying a certain shift, the errors will be minimal. If the errors would not be consistent, there would not be a certain shifts but many different shifts to achieve lower errors.

The curves accompanied with the tables 5.2 and 5.3 show that random forest seems produce the most accurate results. This is also the reason for choosing random forest in the experiment to see does SMSA data increase prediction accuracy. Linear regression is not far behind but support vector regressor and especially multilayer perceptron produces the worst predictions.

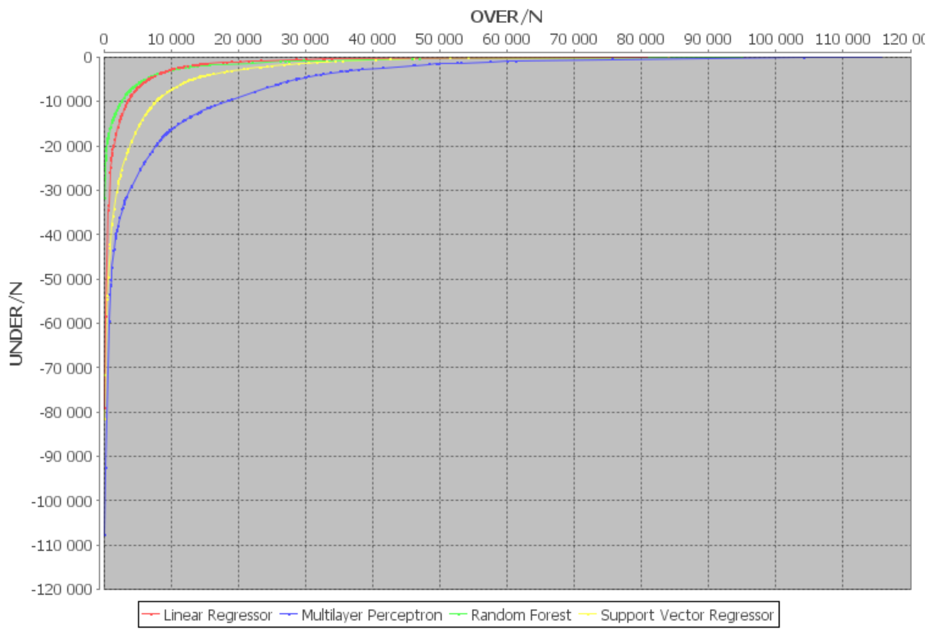


Figure 5.1: RROC curves for major sales with lag value 3

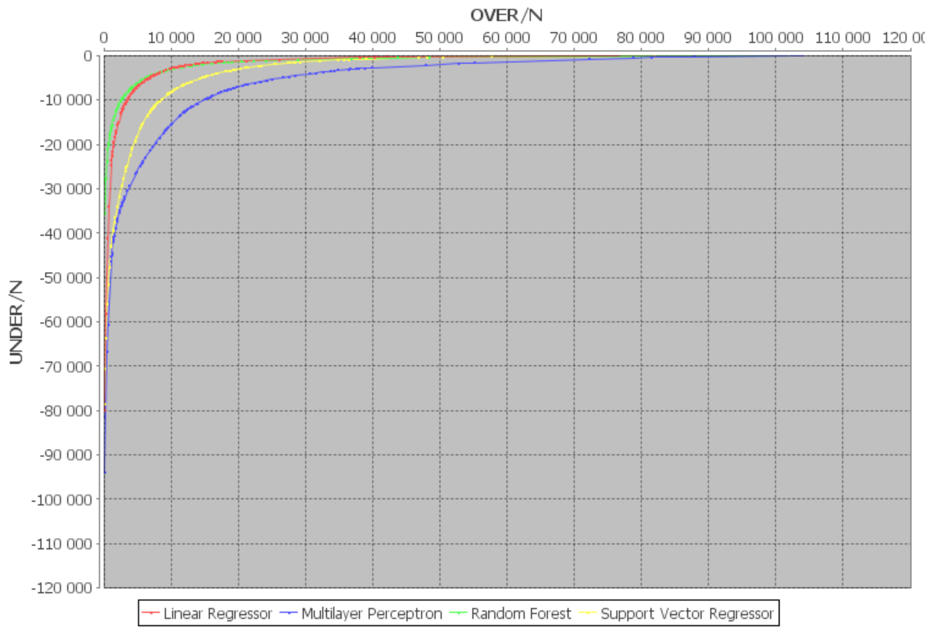


Figure 5.2: RROC curves for major sales with lag value 6

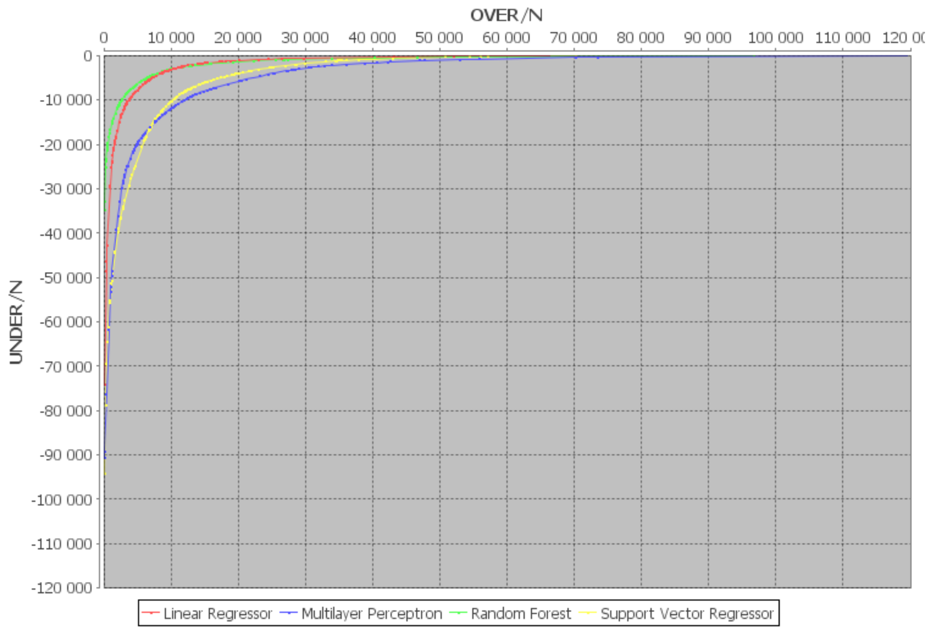


Figure 5.3: RROC curves for major sales with lag value 9

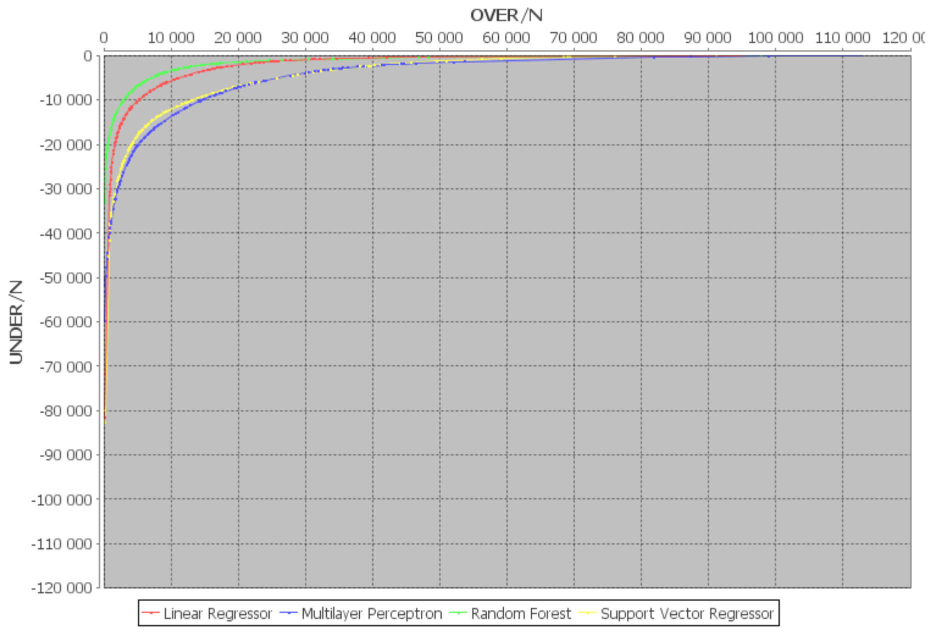


Figure 5.4: RROC curves for major sales with lag value 12

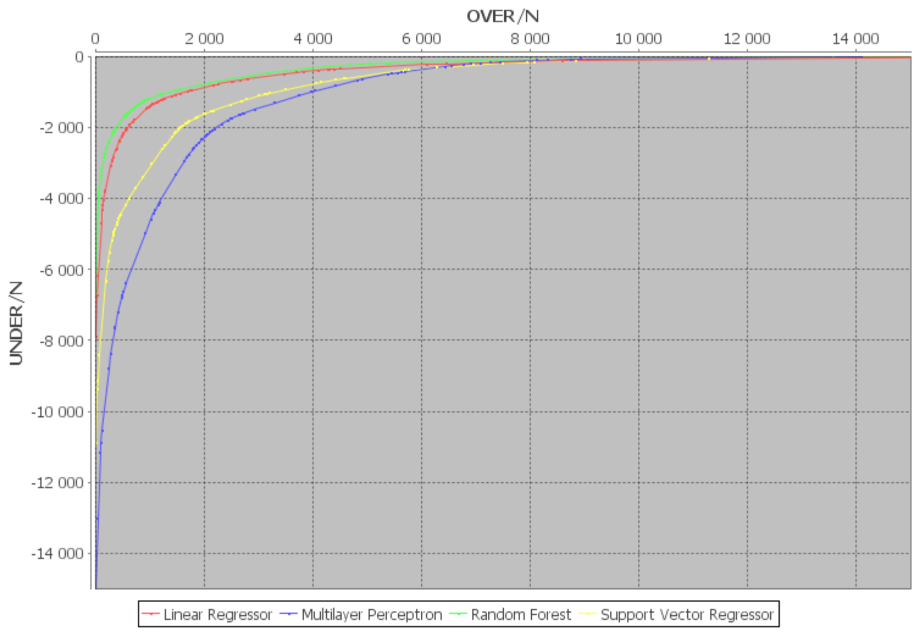


Figure 5.5: RROC curves for minor sales with lag value 3

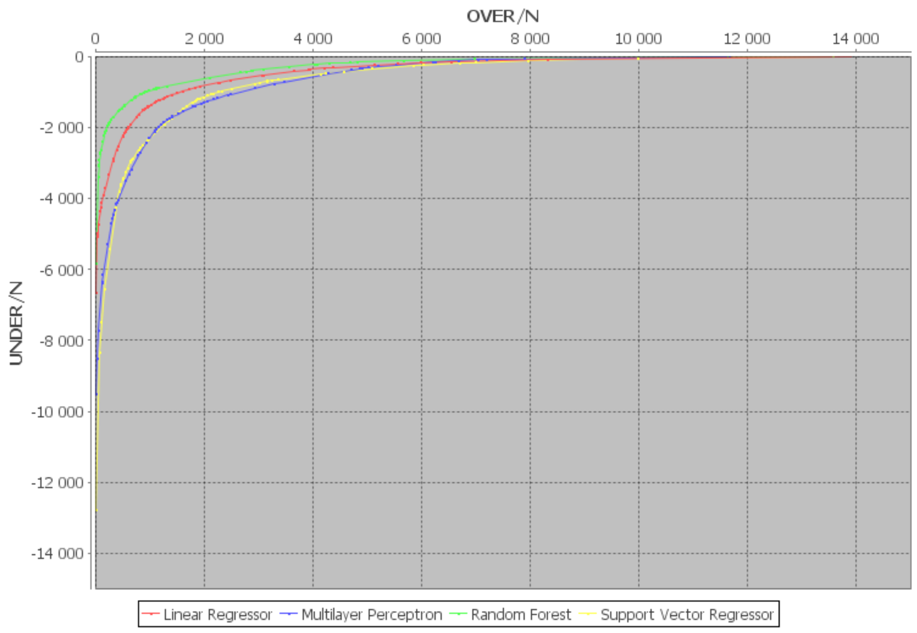


Figure 5.6: RROC curves for minor sales with lag value 6

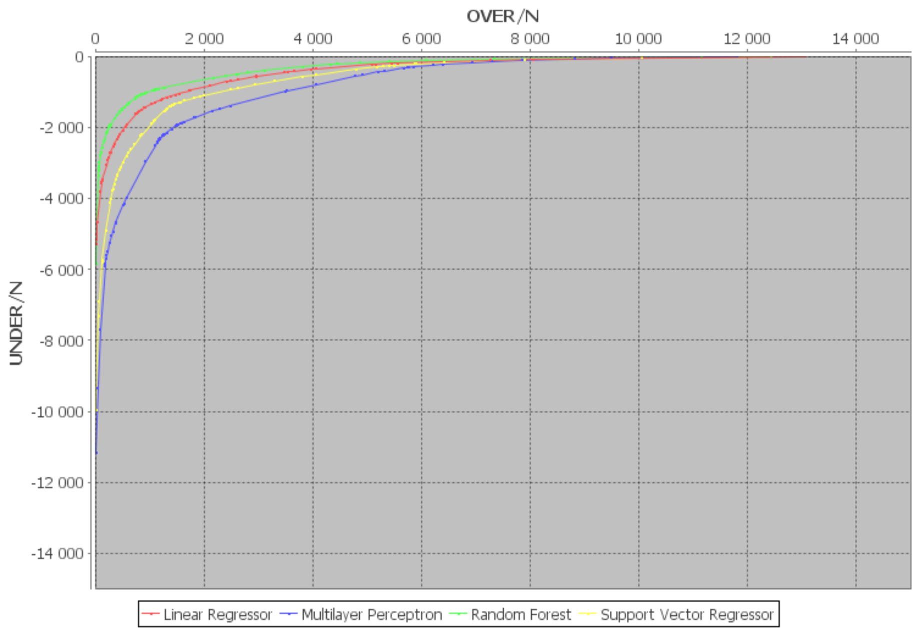


Figure 5.7: RROC curves for minor sales with lag value 9

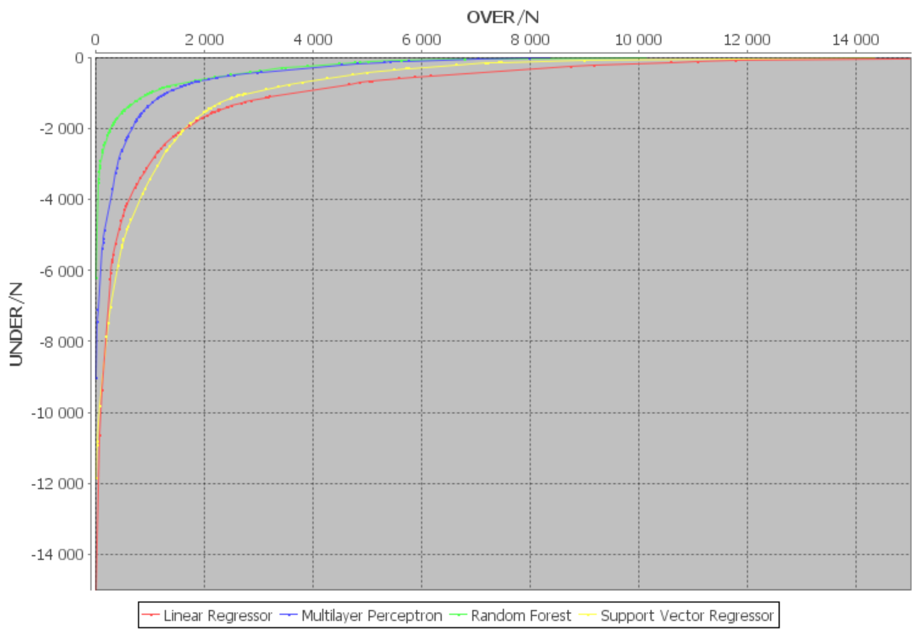


Figure 5.8: RROC curves for minor sales with lag value 12

Chapter 6

Conclusion

This thesis studied how four different machine learning methods (SVM, MLP, RF, and LR) compared to each other in prediction performance to forecast as accurate car sales figures as possible with the help of social media sentiment analysis data. Chapter 2 described the basics of machine learning, such as its definitions, its different categories, and evaluation methods. Chapter 3 explained how the chosen machine learning methods work in both classification and regression problems. Chapter 4 brought in a bit of context of text mining and sentiment analysis. Chapter 5 presented Sales Predictor, the program to conduct the evaluation of the different methods, as well as the results. Sales Predictor found that social media sentiment analysis data does increase the prediction accuracy or reduce the error in average by ten percent. It was also found that random forest does produce the most accurate results in the context of car sales. As an extension to other related work, this thesis used ROC for regression extensively (along with RMSE) to evaluate the methods. Also, there was not found any research concerning the prediction of car sales figures with previous sales figures and social media sentiment analysis data.

It is known that tree like structures, for example decision trees are prone to overfit. However, random forest mitigates this by taking random samples from the trees it has. Linear regression was also not far behind from random forest in prediction accuracy. Support vector machine or regressor and multilayer perceptron were far away, though, especially the latter.

It seemed sometimes that multilayer perceptron would oscillate in certain value or local minimum. This could be avoided by tweaking the value of momentum rate. However, it was decided that the program would use the default parameters for each method.

Further research could find optimum parameters for each method to produce the best results. One point of interest in further research could also be constructing an ensemble method from the different methods, or referring to the RROC curves, use different method in different cases, where more overestimation or more underestimation is required.

In practice, Sales Predictor will be used to predict the sales figures of individual car manufacturers, rather than in groups, such as manufacturers with large or small sales. The sales amounts of different car models of the manufacturers could also be predicted.

Bibliography

[1] Abu-Nimeh, S., et al. A comparison of machine learning techniques for phishing detection, 2007.

[2] Aggarwal, C., et al. *Mining Text Data.* Springer, 2012.

[3] Alpaydin, E. *Introduction to Machine Learning.* Adaptive Computation and Machine Learning. The MIT Press, 2014.

[4] Autoalan tiedotuskeskus. 2015 - autoalan tiedotuskeskus, 2015.

[5] Baskin, I., et al. Tutorial on machine learning, 2015.

[6] Bekkerman, R., et al. *Scaling up Machine Learning.* Cambridge University Press, 2012.

[7] Bi, J., et al. Regression error characteristic curves, 2003.

[8] Campbell, C., et al. *Learning with Support Vector Machines.* Morgan and Claypool, 2011.

[9] Caruana, R., et al. An empirical comparison of supervised learning algorithms, 2006.

[10] Chapelle, et al. *Semi-Supervised Learning.* MIT Press, 2006.

[11] Criminisi, A., et al. *Decision Forests for Computer Vision and Medical Image Analysis.* Springer Verlag, 2013.

[12] Dougherty, G. *Pattern Recognition and Classification.* Springer-Verlag New York, 2013.

[13] Garcia-Gutierrez, J., et al. A comparison of machine learning regression techniques for lidar-derived estimation of forest variables, 2014.

[14] Graupe, D. *Principles of Artificial Neural Networks (3rd Edition).* World Scientific Publishing Company, 2013.

[15] Handke, J. *Natural Language Processing : Structure of the Lexicon : Human versus Machine*. Walter de Gruyter, 2012.

[16] Hernandez-Orallo, J. Roc curves for regression, 2013.

[17] Kabanikhin, S. Definitions and examples of inverse and ill-posed problems, 2008.

[18] Khurshid, A. *Affective Computing and Sentiment Analysis*. Springer, 2011.

[19] Lantz, B. *Machine Learning with R*. Adaptive Computation and Machine Learning. Packt Publishing, 2013.

[20] Pentaho. Time series analysis and forecasting with weka, 2015.

[21] Platt, J. Fast training of support vector machines using sequential minimal optimization, 2000.

[22] Provost, F., et al. *Data Science for Business*. O'Reilly, 2013.

[23] Serrano-Guerrero, J., et al. Sentiment analysis: A review and comparative analysis of web services, 2015.

[24] Sugiyama, M., et al. *Machine Learning in Non-Stationary Environments : Introduction to Covariate Shift Adaptation*. Massachusetts Institute of Technology, 2012.

[25] Suykens, J., et al. *Least Squares Support Vector Machines*. World Scientific, 2002.

[26] The University of Waikato. Class linearregression, 2015.

[27] The University of Waikato. Class multilayerperceptron, 2015.

[28] The University of Waikato. Class randomforest, 2015.

[29] The University of Waikato. Class smo, 2015.

[30] The University of Waikato. Weka 3: Data mining software in java, 2015.

[31] Yan, X. *Linear Regression Analysis : Theory and Computing*. World Scientific, 2009.