
Bachelor Thesis

July 17, 2007

Roman Bauer
bauerr@student.ethz.ch

Supervisor: Prof. Ivo Sbalzarini
Abstract

In this thesis state-of-the-art machine learning tools are applied on stock market data for predicting future share prices. The main problem of this task is the optimization respective to the input data, the indicators, and the parameters of the learner. Several methods using a Support Vector Machine learner are presented, and finally they are compared to each other. It is shown that a generic optimization method, composed of several subprocedures, performs well on new datasets.
Table of contents

1. Introduction 5
   a. Motivation 5
   b. Goal 5
   c. Scope of Work 6

2. Overview 7
   a. Tools 7
   b. Data Sets 7
   c. Indicators 9
   d. Fitness functions 10
     - Hit rate 10
     - MSE 10

3. Optimization 10
   a. Determining Optimal Training/Test Window size 10
   b. Feature Subset Selection 16
      i. Cross-Correlation Analysis 16
      ii. Fisher Information Content Analysis 17
      iii. Feature Selection Results 18
   c. Optimizing Support Vector Machine Parameters 25
      i. Parameters 25
      ii. Grid Based Optimization 26
      iii. Experimental Plots 26
         - Classification 26
         - Regression 28
   d. Feature Weight Optimization 30
      i. Overview 30
      ii. Algorithm 30
      iii. Parameters 30
      iv. Validation on Optimized Feature Sets 31
         - Classification 31
         - Regression 32
      v. Actual Hit Rates 34
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Validation of Predictions using Trading Strategies</td>
<td>36</td>
</tr>
<tr>
<td>a. Trading Strategies</td>
<td>36</td>
</tr>
<tr>
<td>b. Assessing the Strategies</td>
<td>37</td>
</tr>
<tr>
<td>i. Trading Strategies using Classification</td>
<td>39</td>
</tr>
<tr>
<td>ii. Trading Strategies using Regression</td>
<td>42</td>
</tr>
<tr>
<td>5. Conclusions and Outlook</td>
<td>43</td>
</tr>
<tr>
<td>a. Experiments and Results</td>
<td>43</td>
</tr>
<tr>
<td>b. Outlook</td>
<td>43</td>
</tr>
<tr>
<td>c. Personal Experience</td>
<td>44</td>
</tr>
<tr>
<td>5. Technical Documentation</td>
<td>45</td>
</tr>
<tr>
<td>6. Bibliography</td>
<td>47</td>
</tr>
</tbody>
</table>
1) Introduction

a. Motivation

Definition of “Machine Learning” :

The ability of a machine to improve its performance based on previous results. [1]

The fact that a machine can “learn” is very fascinating to me, showing that this is not a skill reserved to biological beings. So programming a computer to learn from real-world data, for example the stock market, caught immediately my interest.

Also the the mathematical structure of the stock market is a very interesting field to me. The efficient market hypothesis (EMH) claims that the stock market reflects the common knowledge in the current price, hence can not be outperformed systematically, knowing as much as the market knows. Considering the success of stock brokers like e.g. Warren Buffett, I must doubt that. The idea of applying computational mathematics to get a better insight into the stock market seems very promising to me, as the amount of data is huge.

b. Goal

It is commonly known that the European market follows the American market. In this work the optimal use of this dependency is focused to get a better performance. Different approaches are assessed and compared with each other by looking at the performance of classification and regression separately. To do this, methods for the optimization process had to be developed and implemented.

The main tasks were:

- Read the raw data from the Internet
- Extract Indicators (features) from the raw data
- Bring the data to an appropriate form to train and predict it with the Support Vector Machine
- Optimize the feature set using different methods

The Support Vector Machine is easily replaceable by any other learner which can return some kind of cost function.
This Bachelor thesis is considered to be the second part of Georg Schneiders semester thesis, which uses neural networks for the prediction process. The main additional work I do here is therefore:

- Replacing the neural network by a SVM
- Exploiting the dependency of European and American markets
- Cross-Correlation analysis, Fisher Feature selection and closed-loop optimization of feature weights with CMA

c. Scope of Work

This chapter gave a brief overview on my bachelor thesis. The next chapter describes the tools and data used for the prediction process. In the third chapter the different aspects of the optimization procedures are treated, and compared to each other using various computer experiments and visualizations. In the last chapter the results are described in order to obtain a optimization procedure which can generically be used.
2) **Overview**

a. **Tools**

I have looked at several machine learning environments, including YALE [2], which has been used by my predecessor Georg Schneider who has done his semester thesis on the same topic using Neural Networks as learners [3]. As a consequence I focused on Support Vector Machines (SVM), Unfortunately the Support Vector Machine learner in YALE turned out to be too slow, and also the coupling with an optimizer would have been very complicated. As the focus should lie on programming optimization procedures with different methods, I decided to use libSVM [4], a very fast Support Vector Machine library, which could easily be adapted in C++ for my own purposes.

I used MATLAB® to extract the indicators and calculate other mathematical functions (e.g. Cross-Correlation, Fisher Information).

The final optimization process using the Covariance Matrix Adaption (CMA) [5] algorithm was done in C++.

b. **Data Sets**

As already mentioned, the dependency of the American and European market should be exploited, so the input data sets were mostly American assets and indices. Also the branch has a big influence on the price values. I chose three different European companies, which were in the focus of the prediction:

<table>
<thead>
<tr>
<th>Company</th>
<th>Country</th>
<th>Sector</th>
<th>Industry</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAP AG</td>
<td>Germany</td>
<td>Technology</td>
<td>Application Software</td>
</tr>
<tr>
<td>UBS AG</td>
<td>Switzerland</td>
<td>Financial Money Center</td>
<td>Foreign Money Center Banks</td>
</tr>
<tr>
<td>E.ON AG</td>
<td>Germany</td>
<td>Utilities</td>
<td>Electric Utilities</td>
</tr>
</tbody>
</table>
The following input share were chosen:

- To predict SAP AG:
  1. DJ EURO STOXX 50
  2. International Business Machines Corp.
  3. S&P 500 INDEX
  4. Microsoft Corp.
  5. NASDAQ Composite Index
  6. Oracle Corp.

- To predict UBS AG:
  1. Bank of America Corp.
  2. Citigroup Inc.
  3. Credit Suisse Group
  4. Deutsche Bank AG
  5. DJ EURO STOXX 50
  6. HSBC HLDGS PLC ADS
  7. JP Morgan Chase & Co.
  9. Wachovia Corp.
  10. NASDAQ COMPOSITE
  11. S&P 500 INDEX

- To predict E.ON AG:
  1. BP plc
  2. Chevron Corp.
  3. Dow Jones Index
  4. Exxon Mobil Corp.
  5. S&P 500
  6. Suez
  7. Total SA
  8. Veolia Environnement SA

The time ranges were:

- for the SAP data set: 17. 05. 2002 – 22. 03. 2007
  (total size: 1217 trading days)
- for the UBS data set: 06. 06. 2002 – 25. 05. 2007
  (total size: 1252 trading days)
- for the E.ON data set: 08. 10. 2001 – 29. 05. 2007
  (total size: 1411 trading days)
c. Indicators

The data sets were downloaded from Yahoo© [9] and contain daily opening, closing, lowest and highest prices. Also an adjusted closing price is included, which is adapted to dividends and splits (see http://help.yahoo.com/l/us/yahoo/finance/quotes/quote-12.html for further information).

Many indicators are used by chart analysts, but their usefulness is unclear and unproven. Therefore I restricted myself to the most simple and commonly most used indicators:

Simple Moving Average: This is the average of the closing prices over certain periods. I used the following periods: 5, 10, 12, 15 and 26 days

Exponential Moving Average: This is an exponentially weighted average, making the closing prices at the beginning of the period contribute less to the result. I used the following periods: 5, 10, 12, 15 and 26 days

Moving Variance: Analog to the Simple Moving Average, the average is calculated for the variance over a certain time period. I used the following periods: 5, 10, 12, 15 and 26 days

Moving Average Convergence Divergence (MACD): This is one of the most used indicators in technical analysis. It is obtained by calculating the difference of two Exponential Moving Averages. I used the difference of the 12 and 26 days Exponential Moving Average

Relative Strength Index (RSI): This is a technical momentum indicator that compares recent gains to recent losses. I calculated it over 14 and 26 days

Also the changes of the closing prices for all included shares from the day before were added to the input features. As the calculations took very long for relative differences, I only calculated total differences, which surprisingly was better for this task. The change of closing price from the share, which had to be predicted was calculated over the last three days.

The number of total features for the three data sets were 98 for SAP AG, 119 for E.ON AG and 152 for UBS.

Of course not the number, but the quality of the features is important. As later will be shown, a too big number of input features selected for the SVM learner is even bad for the performance and adaptivity of the learner, therefore the feature selection methods will be assessed.

d. Fitness functions

I used two different fitness functions for the optimization process, one for classification and one for regression. It turns out that the results for optimizing the regression process
are quite different from the results for optimizing the classification process, which is probably a consequence of the chosen fitness function.

Hit rate

This is the number of correctly predicted directions of the closing price (this means, wheter the share goes up or down), divided by the totally predicted directions. As we have only two directions (the probability of a share to stay exactly the same is very low, so I only looked at upwards und downward directions), we can see this prediction process as a classification.

Mean Squared Error

This negative of the mean squared error can be seen as a fitness function. It is calculated by summing up the squares of the deviations of predicted to true changes:

$$\sum_{testset} \left( \frac{\text{price}_{today,predicted} - \text{price}_{today,true}}{\text{price}_{yesterday}} \right)^2$$

It turned out that predicting the absolut changes was very difficult for the regression process. However, the input differences of closing prices were all absolute, as the optimization process did much better than using relative features.
3) Optimization

Various parameters can be adjusted to obtain better results, the following ones are inspected in this chapter:

- The time over which the SVM is trained and tested.
- Methods to decrease the number of features
- The parameters $C$, $\gamma$ (Gamma) and $\nu$ (Nu) for the SVM learner
- The weights of the input features

a. Determining Optimal Training/Test Window size

The total time span of the data sets is about 6 years (see 2.b)). The stock market is a very dynamic system, especially the ones with high volatility. Therefore, it is better to choose the training time window small. On the other hand, a small training time window means less generalization and more adaption on special situations.

Also $C$, the penalty parameter for misclassification of the SVM, is shown to have a big influence on the optimal training set size. The following diagrams have been plotted to show this dependance on the SAP data set. The SVM is trained on a training set with changing size, and applied on a test set of 100 days:

Figure 1: Plot of hit rates for different training sizes, done with $C=10$
Figure 2: Plot of hit rates for different training sizes, done with C=100
It becomes clear that a higher $C$ should go hand in hand with a bigger size of the training set. This is not astonishing, as a higher $C$ means more specialization and therefore must be balanced by more data in order not to overfit.

There seem to be two regions, separated by a strongly rising curve at about 550 days. The same calculations on other data sets confirmed this.

The correlation of the share price which has to be predicted with the other input prices is mostly decreasing with the time lag, like the autocorrelation function for the E.ON AG share:
Figure 4: Autocorrelation for E.ON AG share price

Sometimes the correlation between the share prices was increasing, like e.g. the correlation of the E.ON AG share with the Chevron Corp. share:
So at least 600 days of training size should be appropriate, which is a little bit more than suggested in [7]. I used this length also for the validation later on.

The test window size was chosen (as already mentioned above) to be 100 trading days. This proportion of 6 to 1 for training size to test size is also commonly nod unusual. The following plot for the number of badly classified direction movements shows that there is no systematic error of the hit rate with respect to the time in the test sample. This confirms that 100 trading days are appropriate for the test window size, as the error scales linearly all the time:
b. Feature Subset Selection

i) Cross-Correlation Analysis

As the amount of data grows with the number of input features, it is favourable to get rid of redundant indicators. As later on will be shown, a large number of features decreases the performance, therefore features which don’t contribute much new information should be deselected.

The most simple method to do this is calculating the cross-correlation coefficient matrix, and neglecting the most correlated features. Also the variance of the prediction can be decreased using this method. A disadvantage here is, that only the linear correlation is calculated, so it could be that features which depend e.g. exponentially on each other are both chosen, although the SVM is already using a radial basis function and therefore calculating the exponential of the features.

The threshold for the correlation coefficient was chosen to be 0.7 for all the data sets. The selected number of features scales linearly with the threshold, as can be seen in the following plot for the E.ON AG data set:

![Plot showing the scaling of the total number of badly classified directions for UBS data set](image)
After having applied this method on a feature set containing all the features, the subset is called corr_set.

### ii) Fisher Information Analysis

Here we calculate the Fisher Information of each feature, represented as a matrix. The features with highest Fisher Information are selected for the input data set. A feature a has more “information” than a feature b if the difference of the two fisher matrices, (A-B) is positive definite. A disadvantage of this method is the neglection of correlations between features, as we calculate the matrices seperately, and therefore information which can only be obtained using several features is lost.

The selected number of features were chosen equal to the number of features selected in the cross-correlation selection process. This was done in order to provide a comparable performance statement of the two selection methods.

After having applied this method on a feature set containing all the features, the subset is called fisher_set.
iii) Feature Selection Results

In order to provide a comparable visualization, I also selected several features by hand for each of the data sets. This was done so that the hit rate was ok and over 50 %. This subset is called rand_set.

The obtained hit rates (classification), mean squared error (regression) and standard deviations (classification and regression), using 5-fold validation are plotted in the following diagrams:

Figure 8: Hit rates without and with the two feature selection methods, using SVM standard parameter C=1
Figure 9: Standard deviation for classification without and with the two feature selection methods, using SVM standard parameter C=1

Figure 10: Hit rates without and with the two feature selection methods, using SVM parameter C=0.4
The feature selection methods perform both comparable to each other on the new data with respect to the standard deviation, which was often decreased. The cross-correlation analysis method often decreased the hit rate for SVM parameter C=1, the Fisher information selection procedure performs mostly good for the parameters obtained later on in the next chapter, and the variance is mostly reduced. All in all the results indicate that this methods can increase the performance, but there is no insurance that you obtain better results using them. However, as the number of features decreases significantly, the calculation time is reduced and generally the overview is enhanced. Also it is shown that the SVM parameters have an influence on the success of the feature selection method, as for C=0.4 the performance is better, compared to the results obtained with C=1.

As a main advantage, the following plots show that trivial predictions can be turned into non-trivial predictions using feature selection:
Figure 12: Plot of real direction and classifier prediction on total SAP data set using SVM parameter $C=1$.

Figure 13: Plot of real direction and classifier prediction on the same part of the data set after Fisher feature selection using the same parameters.
In the first plot the prediction was always -1, but after the selection process the prediction became non-trivial, which is a consequence of the reduced total number of input features.

Regression

As already mentioned in [2], a high performance of the classification procedure does not necessarily mean low mean squared error. Therefore classification and regression have to be considered separately. The following calculations were obtained using the SVM parameters $C$ and $\nu$ from the next chapter.

![Bar chart showing MSE values for SAP data set with and without feature selection.](image)

**Figure 14**: MSE (Mean squared error) of SAP data set for regression without and with the feature selection methods, using SVM parameters $C=0.2$ and $\nu =0.1$
Figure 15: Standard Deviation of MSE of SAP data set for regression without and with the feature selection methods, using SVM parameters C=0.3 and $\nu=0.1$

Figure 16: MSE (Mean squared error) of UBS data set for regression without and with the feature selection methods, using SVM parameters C=0.2 and $\nu=0.1$
Figure 17: Standard Deviation of MSE of UBS data set for regression without and with the feature selection methods, using SVM parameters $C=0.2$ and $\nu=0.1$

Figure 18: MSE (Mean squared error) of E.ON AG data set for regression without and with the feature selection methods, using SVM parameters $C=0.2$ and $\nu=0.1$
Figure 19: Standard Deviation of MSE on E.ON AG data set for regression without and with the feature selection methods, using standard SVM parameters

The selection procedures affected the performance comparable to the classification hit rate, which is not surprising. Notice that the MSE is calculated with respect to the residuals of relative changes of closing prices (multiplied by 10), so the mean squared errors are not comparable in the diagramms with each other, as some shares are more volatile than others and therefore fluctuating more strongly. A bigger MSE for the SAP data set does not mean that the regression performed worse here than on the other data sets.

Also here, the feature subset selection using cross-correlation analysis also performs mostly worse than Fisher information subset selection here, but it is compatible with it. The learner applied on the features randomly chosen by me performs not bad, but mostly worse than the two selection procedures.

Also here, a too large number of features clearly has an influence on the performance.

c. Optimizing Support Vector Machine Parameters

i) Parameters

The SVM learner can adapt the classification and regression process with two parameters. The penalty parameter C and the parameter $\gamma$ for the RBF (radial basis function) kernel function, which can be adjusted in order to generalize better and therefore obtain better performance results, and they must be obtained experimentally.
For the regression task the type “nu-SVR” was tested with different parameters C and \( \nu \), which defines the number of used support vectors.

ii) Grid Based Optimization

LibSVM includes a Python script which searches for optimal parameters using a grid based optimization procedure. This is done by finding the best parameters for a certain grid spacing and then by “zooming” into the interesting area. Unfortunately the validation is done using cross-validation, and the time direction is only preserved for two-fold cross-validation, which is not sufficient to ensure the generalization of the performance on the data set. Therefore I implemented a parameter search program using 5-fold validation.

iii) Experimental Plots

Classification

The following 3d plots of the parameter space have been made on the SAP data set, using 5 fold validation:

![Figure 20: Plot of hit rate vs. SVM parameters C and \( \gamma \) for UBS data set](image)
The standard parameters $C=1$ and $\gamma = 1/k$ (k is number of input records) for the SVM seem both to be appropriate for the classification problem, as the hit rate is constant. This is due to trivial solutions which didn’t change with the chosen parameters.

Now this not true anymore for smaller feature sets, as the following plot on the UBS data set shows after having reduced the number of features using Fisher information selection:

![Plot of hit rate vs. SVM parameters C and $\gamma$ for UBS data set (after Fisher information selection)](image_url)

**Figure 21:** Plot of hit rate vs. SVM parameters C and $\gamma$ for UBS data set (after Fisher information selection)

The mean hit rate is much higher in the region where C and $\gamma$ are both small. As this is normally the case for $\gamma$, I only adapted C for the classification process.

C = 0.4 was used for the optimization procedure later on, done with CMA.
For regression, both the C and \( \nu \) have a big influence on the MSE: The smallest errors are obtained for small C and extremal \( \nu \), which means near 0 and 1. Simulations on other total data sets and on data sets after having performed feature selection confirmed this, and usually the best results were obtained for small C and \( \nu \). The standard value for \( \gamma \) also turned out to be appropriate. Therefore I used \( C=0.2 \) and \( \nu =0.1 \) for the weight optimization procedure later on.

The following two plots show the strong influence of the parameter C on the predicted deviations:
Figure 23: Plot of predicted (red) and true relative deviations (multiplied by 10), done on the full E.ON data set with parameters $C=0.2$ and $\nu=0.1$.

Figure 24: Plot of predicted (red) and true relative deviations (multiplied by 10), done on the full E.ON data set with parameters $C=100$ and standard parameter $\nu$. 
For small C the regression has much smaller deviations and is normally in the nearer regions of the mean. For C=100 the deviations are much more radical, as the learner is more specialized.

e. Feature Weight Optimization

i. Overview

As a final method of optimization I implemented adaptive feature weights, found by a closed-loop coupling of the CMA (Covariance Matrix Adaption) optimizer and the SVM learner. The features were each scaled by a certain factor to maximize the performance. This way of optimizing the feature set is much more effective as it chooses the features respective to a specific model, which was not done in the two feature selection methods above and in the thesis of Georg Schneider. Both classification and regression have been tested this way.

ii. Algorithm

The CMA algorithm adapts the covariance matrix of the normal distribution in an evolutionary way, in order to minimize a cost function. The algorithm performs very good also on unsmooth and difficult spaces. The correlations between the variables are defined in this algorithm, which is very good respective to this work, as the dependency with the American assets can be better exploited this way. For more information on the CMA algorithm, see [5] and [6].

iii. Parameters

The number of iterations was chosen to be 10 times the number of features, and this was done 5 times for each data set in order to enhance the reliability. The parameters Sigma and Minsigma were chosen 0.07 and 0.03. As the classification process is not smooth, the parameter for the offspring (Mu) was set to 1, which means that only one point is chosen for the mutation process. Otherwise, if the cost function is smooth, the parameter Mu is normally set to half of the number of offspring, which I set to 10 in this case.

For the classification, the cost function was the negative hit rate. For the regression, it was the MSE.
iv. Validation on Optimized Feature Sets

These are the results after having optimized the weights on the different data sets:

Classification:

Figure 25: Mean hit rate of classification after weight optimization on the three data sets
Regression:

Figure 26: Standard deviation of classification after weight optimization on the three data sets
Figure 27: MSE for regression after weight optimization on the SAP data set, using SVM parameters $C=0.2$ and $\nu=0.1$

Figure 28: Standard deviation of MSE for regression after weight optimization on the SAP data set, using SVM parameters $C=0.2$ and $\nu=0.1$

The MSE is smaller and has smaller variance for the bigger feature sets, which is the contrary of what happens for the classification process, where the performance gets better. A reason for this fact could be the different measure of performance: The mean
squared error is probably smaller if the predicted deviation is near the average deviation and if it is small. A bigger feature set and therefore a more general, unspecific model probably fulfills this better than a widely fluctuating line, which is predicted for smaller feature sets.

The standard deviation even grows now, after having optimized the feature subsets. This occurs probably because the learner now is more specific and adapted on the smaller subsets, as it is optimized exactly for this subsets. On the total data set, it is very difficult to do this, and therefore it is more general and unadapted.

v. Actual Hit Rates

Of course one can not conclude from the obtained results in these diagramms that the learner performs so good on previously unseen data. The prediction was just adapted to the used validation data, as the optimizer iteratively used this data to obtain a model. In order to look at the actual performance I used the obtained optimized data and the optimized SVM parameters to predict the next 100 unseen changes of closing prices:

![Figure 29: Hit rate of classification after weight optimization on previously unseen data](image)

The hit rates are very good, compared to the hit rates reached in the thesis of Georg Schneider. This can have many reasons, as for instance the American assets or the SVM parameters.
learner, but all in all the optimization process helped very much. Also the time over which the predictions are made is rather short, so the dynamics of the system can’t change very much.

Like already mentioned, the predictions using the total data set (and some reduced feature sets, as for SAP AG) were trivial, so the high hit rates obtained for this data are probably coincidence, as the share was following a trend (mostly increasing). However, if the training data itself had to be predicted, the solution was not trivial anymore. Therefore, it was not a trivial model, but depended very much on the input.
4) Validation of Predictions using Trading Strategies

This chapter deals with assessing the performance in form of effectively gaining money at the stock market, and comparing this with the performance reached by Georg Schneider. It is shown that this composed method of using the dependency of different shares, feature subset selection, parameter optimization and closed-loop feature scaling performs quite good.

i) Trading Strategies

The validation was done on the E.ON data set, as only here enough data, which has not already been used in the optimization process, was available. Fortunately, the results obtained here for the classification were also the best among the three data sets. In order to use the information of the prediction in different ways, I looked at different trading strategies. They are very similar to the ones described in the thesis of Georg Schneider (and performing best), so I explain them only in a short way:

Rand (random trading): This strategy randomly makes buying or selling decisions. The final performance is obtained by averaging over 100 runs.

Trad: The MACD indicator and the signal line were calculated. If the MACD crosses the signal line from below, a buying decision is made, and vice versa.

Null (no trading): This strategy buys as many shares as possible at the beginning, and does not do anything at all after that. Therefore, this strategy can only work during increasing periods.

Rare (no strategy): This is no strategy. I just exactly does what the predicted direction indicates.

Trend: If three succeeding predicted price changes are positive, this strategy assumes an upwards trend and therefore buys shares. The selling decision is made for three decreasing prices.

Inv (inverse trend): This is just the opposite of the Trend strategy: If the predicted direction is three times the same, this strategy assumes the next direction to be in the opposite direction, and accordingly the buying or selling signal is made.

Tp (turning point): This strategy looks for three movements in the same direction, followed by a predicted movement in the opposite direction. This is then interpreted as a turning point, which can be used as a buying or selling signal.

Mix_tp (mixed turning point): This is also the same as the Tp strategy. It uses both real and predicted movements. If two real movements in the same direction are succeeded by two predicted movements in the other direction, this is interpreted as a turning point, and therefore a selling signal.

P_b_s (proportional): This strategy make a buying decision if the relative predicted movement is above an adaptable threshold b. Straightforwardly, it makes a selling decision for the movement above the adaptable threshold s.
**Pa_b_s (proportional amount):** This strategy is similar to the P_b_s strategy. The only difference is that the number of shares being bought is:

\[
\text{Buy\_number} = \frac{\text{liquid\_assets} \times \text{rel\_pred\_movement} \times 10}{\text{price}_{\text{today}}}
\]

The number of shares being sold is:

\[
\text{Sell\_number} = \text{available\_shares} \times \text{rel\_pred\_movement} \times 10;
\]

ii) Assessing the Strategies

As a measure of performance I used, adapting to Georg Schneider, the TVA (total value of assets). It is calculated by the value of liquid assets and currently possessed shares:

\[
\text{TVA} = \text{liquid\_assets} + \text{available\_shares} \times \text{price}_{\text{today}}
\]

The E.ON AG share value is mostly increasing in the focused time span of 385 trading days, which is more than one and a half year. The following plot visualizes an example of the buying and selling decisions made:
The predicted directions are obtained using the optimized fisher_set and the rare strategy. The red circles indicate selling decisions, the green crosses indicate buying decisions.

In order to compare the strategies, the TVA at the end of the time span is focused. At the beginning, all strategies start with a TVA consisting of only liquid assets, which I chose to be 100000 $. The transaction fee was chosen to be 50 $ for each transaction.

The following diagramms visualize the performance of the strategies on the optimized data sets.
Figure 31: Diagramm, showing the reached amount of money in percent of the initial TVA. The results are very similar to the ones in Georg Schneiders thesis.

The Trad strategy reached only 109.8 % of the TVA at the beginning, which consisted of liquid assets. As the price was rising very fast during the focused period, the Null strategy performed very well, reaching 152.3 %. It was therefore not surprising that the other strategies didn’t to better.

The following plots show the TVA, the percentage of liquid assets and the accumulated fees over time, using the Mix_tp strategy and the predictions on the optimized corr_set. The share price was scaled and included to compare the behaviour.
Figure 32: TVA (blue) and share price *1100 (red) during time

Figure 33: Percentage of liquid assets during time
Figure 34: Percentage of accumulated fees during time

The TVA behaves very similar to the share price. The percentage of liquid assets is shrinking, which is good, as the price is in an upwards trend.
Trading Strategies using Regression

Figure 35: Diagram, showing the reached amount of money in percent of the TVA at the beginning. Also here, the results are very similar to the ones in Georg Schneiders thesis.

This is very similar to the results obtained by Georg Schneider. Also here the performance reached using the information of the predicted size of deviation, the regression, is higher than only using the predicted direction. Additionally, also here the Pa_b_s strategy performed best. The parameters b and s didn’t affect the performance very much in the chosen range, so 0.05 is appropriate for both b and s.

The percentage of liquid assets decreased very fast, and the accumulated amount of fees was constant over a short time, as all the money was already invested then.

It was not possible for the strategies to outperform the Null strategy, but the Pa_0.005_0.005 on the optimized total and correlation coefficient selected set performed good. In reality it is also possible to buy Put options, so the implementation of strategies using this additional tool could also gain money on constantly decreasing share prices. However, the calculation of the option pricing would be too difficult to do in this thesis.
5) Conclusions and Outlook

a. Experiments and Results

In this work several methods have been investigated in order to optimize the performance of the learner.

The training time window size was chosen by looking at the continuous performance of classification. The obtained “optimal” size is slightly more than found in other works (see [7]).

Feature Subset selection is done in order to reduce the training time and the variance. Fisher Information analysis and Cross-Correlation analysis turn out to be compatible with each other, but the performance the regression is mostly better when the total data set is used. The reason is probably that the classification process gets stuck in trivial solutions more easily than the regression process, which can handle large amounts of input data better in terms of minimizing the MSE.

Feature weight optimization clearly outperforms the other optimization procedures, although it takes much longer to do all the iterations. The clue is that combining the weight optimization and the feature selection method leads to much better results, especially in classification. A big advantage of the optimizer is that the correlations between the variables are not neglected, which is the case for the Fischer information analysis, and to some degree also in the Cross-correlation analysis done with the correlation coefficients. However, the idea of coupling the model and the features makes a big difference on the performance, as not only the number of informative features has an influence on the whole prediction performance, but also the adaptivity of them to a certain model.

The performance of the trading strategies showed about the same behaviour as in the thesis of Georg Schneider. Also here, the regression reached the best TVA at the end. The use of the correlation analysis or Fisher information selection method, together with the optimizer lead to good results.

b. Outlook

There is still plenty of work to do. One could apply other learners, see [9]. Also the use of other feature selection tools, as for instance genetic algorithms, which also couple the selection process with the learner, could be helpful for reaching a better performance. Especially for very big feature numbers, this could be the appropriate method (See e.g. [8] for more information).

Respective to the quality of the input data set, the influence of other factors like the season of the year, various Asian assets or indicators could be investigated. This is mainly an experimental task.

Another very interesting method for feature selection would be the maximization of the performance of the trading algorithm instead of the hit rate or the mean squared error. This would probably get a better TVA at the end, as (similarly to the feature set...
weighting with CMA) the feature set is chosen optimally according to a certain trading strategy.

Also using a full rotation matrix as optimizer weights instead of a diagonal matrix could be an alternative way of increasing the performance.

Of course the nature of this tasks is mainly experimental, so alternative theories as for instance fractal theory could provide a new fundament for this topic [10].

c. Personal Experience

I learned very much about state-of-the-art machine learning tools, because at the beginning of this work I was searching for an appropriate learner and therefore had to look at several tools, including YALE and the Stuttgarter Neuronale Netzwerke Simulator. The mathematical clearness of the Support Vector Machine is a big advantage, as Neural Networks are often criticized as being a “black box”.

Also my programming skills got much better, I learned writing shell scripts for the smaller tasks of this work. Furthermore, the parallel optimization of the feature weights on several Computers was a big challenge for me.

And at last but not least, I genererally learned much about the topic “financial mathematics” and the current main topics in this field.

I want to thank my supervisor Prof. Ivo Sbalzarini for his helpful advices and comments during this work.
5) Technical Documentation

The following chapter gives a brief overview with respect to the used codes, scripts and C++ programs.

Preparing the data and extracting the indicators

After having downloaded the share values, the data had to be brought together by looking at the date. This was a very time consuming process, as it often happened that in some shares daily closing prices were included which weren't included in other shares. Therefore I had to erase the closing prices at the right places by hand. As sometimes the prices were splitted, they had to be adjusted at the right positions. The prices then were loaded from Excel files into Matlab© and converted into a appropriate form including the indicators using the data_conv.m function. This function uses various other M-files for calculating the indicators, and then brings them in matrix form. Notice that the variables to be predicted must be in the first row for the Support Vector Machine later. The matrix was then copied into a .txt or .dat file, from which it was then converted with the LightDataAgentv21 binary into a usable form for the Support Vector Machine. Especially for the regression process, the scaling of the input data set is very important, otherwise nonsense results are obtained. I therefore used svm-scale.exe to scale both the input data, as well as the relative differences of the closing prices. The information on the scaling process can also be saved, so that it can be reused to obtain the predicted differences of relative closing prices.

Subset selection

Applying the Fisher information selection algorithm was done with the get_fisher_set.m script. This script uses the ecmnfish.m script from the Matlab Finance Toolbox© for calculating the Fisher information matrix.

The cross-correlation analysis was done with the corr_coef_sparser.m script, where a threshold for the correlation coefficient can be chosen.

Grid based SVM parameter optimization

The optimization procedures were done in C++, using the parameter_search.c code for classification and parameter_reg_search.c code for regression. I used 5-fold validation on succeeding test sets to preserve the time direction. This means that the validation was done over a time window of 5 times 100 trading days, which is about two years.

Feature weight optimization with CMA

The optimization process is done using the CMA algorithm. There are two functions, given to this algorithm as the evaluator function: evaluator_function.c or reg_evaluator_function.c, depending on whether the classification or regression process is being optimized. They both use 5-fold valuation in order to preserve the time
direction. The time for an optimization process is very long (up to several days), so it is appropriate to simultaneously use several computers.

Assessing the trading strategies

The prediction on the new datasets was done using the shell script *my_param_script.sh*, which can be used to select features from a data file containing the total feature set. The scaling process, which uses the weights obtained from the optimizer, can be done with the C program *scale_function.c*. The hit rate was calculated using the *get_hit_rate.c* code.

The trading algorithms were all done in MATLAB© using various M-files.
6) Bibliography


[7] Steven Walczak: An Empirical Analysis of Data Requirements for Financial Forecasting with Neural Networks


