# Financial Big Data Project

Georgios Kyritsis, Hua Zhong

December 12, 2016

**Abstract**

In this report, we summarize our findings for the Financial Big Data course. The purpose of the project is to perform portfolio optimization in order to give advice in which securities to invest. We explore statistically the returns of stocks and try to find typical characteristics of financial market data. We experiment using mean-variance optimization, minimum-variance optimization, and also machine learning techniques such as support vector machines and neural networks to forecast financial time series.

## 1 Getting and cleaning Data

Before doing any analysis we have to collect raw data and transform them into consistent data that can be analyzed. The R statistical environment provides a good environment for both downloading and cleaning data. Using the R package tseries we are able to download historical stock prices from Yahoo! Finance from the year 1950 till now. So we create a data frame of 16992 dates and 6132 ticker symbols. The next step is to count the number of missing values in this large data frame. We calculate that approximately 79% of the values are missing. From figure 1 it is clear that the older the date the larger the value of the missing values. Also from figure 1 it is apparent that number of missing values fluctuates a lot especially after the year 2010. The next step is to subset our data set and take stock prices after year 1990. So we end up with a matrix of size 6936 x 6132. In this situation 51.48% of values are missing. The first column in our data set is the value of S&P500 index. We use this column to filter out the rows in which the value of S&P500 is missing. Next, we count that 689 ticker symbols have no missing data. So we could end up with a matrix of size 6936 x 689, but we think that it is better for our analysis to have more tickers. So after trying different years, we subset our initial data set after year 1992. Our final data set that we use in our analysis contains 1018 ticker symbols and 6272 dates. When we perform optimization with rolling covariance matrices we use dates after 2000 to reduce somehow the computing time. The dataset in this case reduces to 4250 dates and 1018 ticker symbols. When we use machine learning, we use the dataset with dates starting from 1992.

## 2 Statistical exploration of assets returns

This section is trying to unveil some typical characteristics of financial market data. These are summarized in the literature as "stylized facts" [1]. Almost all return time series of stocks and indexes exhibit the same statistical properties. A risk model that does not capture the characteristics of the financial market data will not be useful for deriving risk measures. The main stylized facts are:

- the absence of simple arbitrage
- the power law decay of the tails of the return distribution

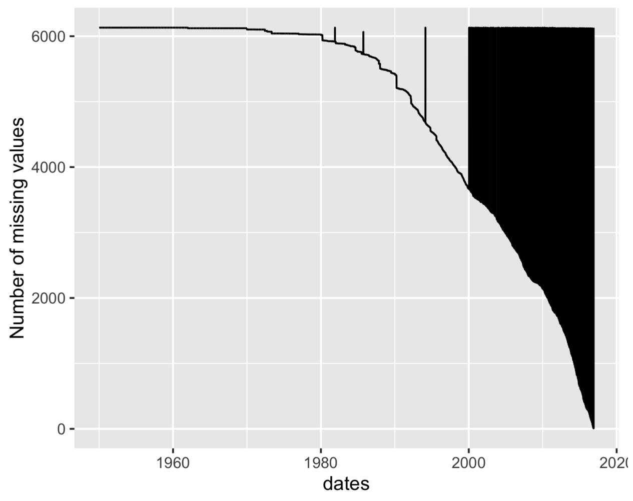

Figure 1: Number of missing values as a function of date.

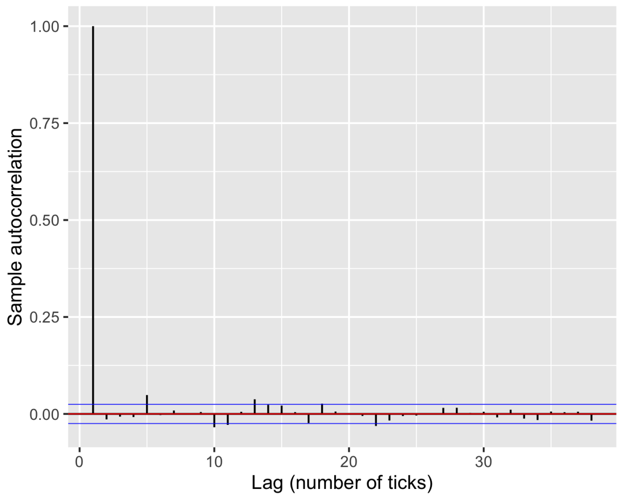

Figure 2: Autocorrelation function of tick by tick returns on Apple shares traded on the NYSE. Time scale: ticks.

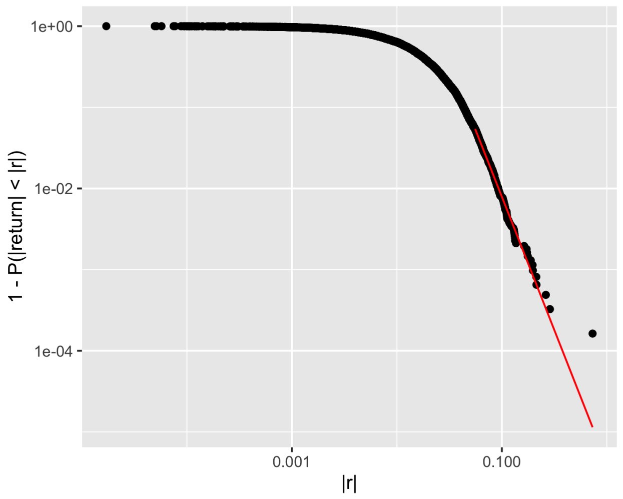

Figure 3: CDF of Apple returns with line of best fit.

- the volatility clustering

In the following sections we check whether these stylized facts are applicable to the daily log returns of Apple stock and S$P500 index. Before proceeding further, log return for a particular day is given by $ret_t = log(p_t/p_{t-1})$ where $p_t$ denotes price on date $t$. Campbell et al.(1997) [2] give reasons why returns are preferred instead of prices.

## 2.1 Absence of simple arbitrage

The absence of arbitrage implies that it is impossible to make profits in financial markets without dealing with a risky investment. In other words the market is efficient in the sense that it quickly tries to eliminate arbitrage opportunities. The price series up to now gives no information about the next asset price. The autocorrelation function of returns is assumed as a good measure of the market efficiency. We report the autocorrelation of returns for the Apple stock in figure 2. It is seen in figure 2 how the autocorrelation function for the Apple returns series decays to zero after one lag.

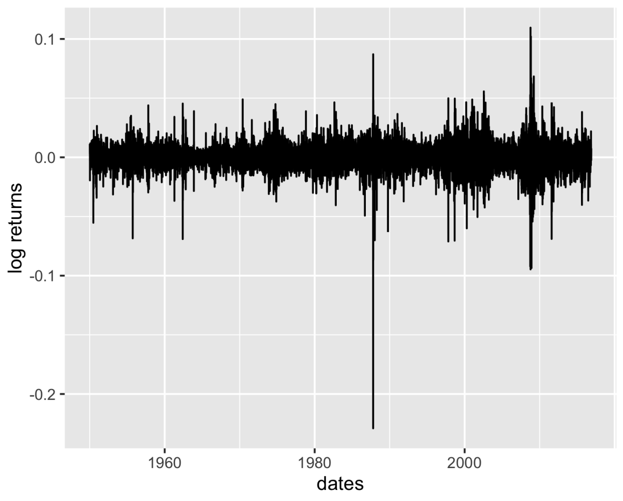

Figure 4: Log returns of S&P500 for period 1950-2016.

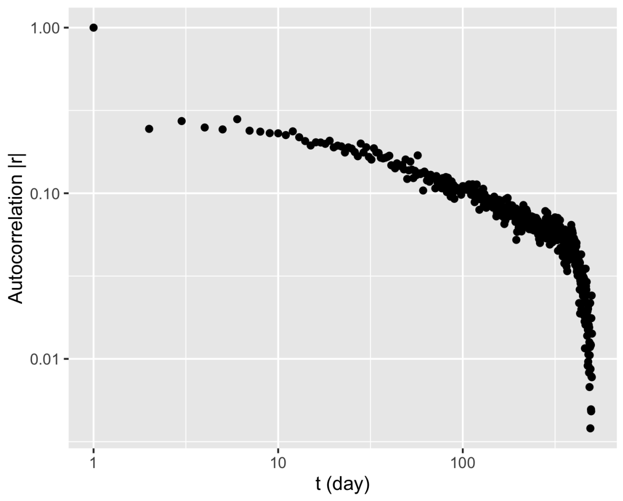

Figure 5: Autocorrelation function of volatility measured as the absolute value of returns for returns of S&P500 for period 1950-2016.

## 2.2 Fat-Tailed distribution of returns

The distribution of returns is not a Gaussian and prices do not follow a simple random walk. In details very large fluctuations are much more likely in stock market with respect to random walk and dramatic crashes are approximately observed every 5-10 years on average. These large events cannot be explained by gaussian returns. Therefore to characterize the probability of these events we introduce the complementary cumulative distribution function $F(x)$:

$$F(x) = 1 - Prob(X < x)$$

which describes the tail behavior of the distribution P(x) of returns. The complementary cumulative distribution function $F(x)$ of real returns is found to be approximately a power law $F(x) \sim \; x^{-a}$ with exponent in the range 2-5 [1], i.e. the tails of the probability density function (pdf) decay with an exponent $a + 1$. Since the decay is much slower than a gaussian this evidence is called Fat or Heavy Tails. Moreover the return pdf is characterized by large kurtosis, whereas a gaussian is characterized by kurtosis equals to three. In our case (Apple returns) $a = 4.28$ and $kurtosis = 6.52$. In figure 3 we report the complementary distribution function $F(x)$ of Apple returns with the best fit.

## 2.3 Volatility Clustering

We report the return time series of the S&P500 index for period 1950-2016. It is clear from figure 4 that periods of large fluctuations are followed by large fluctuations regardless of the sign and the same behavior happens for small ones. As a consequence the autocorrelation function of absolute returns is non zero. This is presented in figure 5. The very slow decay means that volatility is correlated on very long time scales from minutes to several months/years. The autocorrelation function of absolute returns decays slowly as a function of the time lag, roughly as a power law with an exponent $\beta \in [0.2, 0.4]$. The volatility clustering was observed the first time by Mandelbrot [3].

## 2.4 Other Stylized Facts

We can present other effects that are widespread in financial markets such as:

Table 1: Summary statistics for S&P500 returns from 1950 to 2016

| Time Scale | Mean | Median | Std | Skewness | Kurtosis | JB | JB p-value |
|---|---|---|---|---|---|---|---|
| Daily | 0.00029 | 0.00046 | 0.00970 | -1.01057 | 27.09344 | 518160 | <2.2e-16 |
| Weekly | 0.00139 | 0.00307 | 0.02126 | -0.63072 | 5.524142 | 4691.2 | <2.2e-16 |
| Monthly | 0.00608 | 0.00906 | 0.04155 | -0.65472 | 2.452925 | 264.16 | <2.2e-16 |
| Quarterly | 0.01821 | 0.02723 | 0.07728 | -0.96021 | 2.012611 | 91.008 | <2.2e-16 |
| Annually | 0.07287 | 0.10787 | 0.16253 | -0.93114 | 1.139258 | 15.832 | 0.013 |

- Gain loss asymmetry: one observes large drawdowns in stock prices and stock index values but not equally large upward movements. This is linked to the asymmetry of the return pdf. To prove this we calculate the skewness and the kurtosis of daily log returns of S&P 500 for the period 1950-2016. $Kurtosis = 27.09$ indicating fat tails, and $skewness = -1.01$ indicating the lack of symmetry. Concretely our skewness is negative, i.e. we have frequent small gains and a few extreme losses.

- Aggregational Gaussianity: as ones increases the time scale over which returns are calculated, their distribution looks more and more like a normal distribution. Table 1 shows how kurtosis and the value of Jarque-Bera normality test decreases as time scale increases. Jarque-Bera normality test determines if assets historical returns' probability distribution skewness and kurtosis match a standard normal probability distribution.

## 3 Analysis based on Covariance Matrix

There exists many long-term quantitative strategies such that **Modern portfolio theory or mean-variance analysis** [4] introduced by Harry Markowitz and **asset allocation** [5]. All these methods of building your portfolio are proved to be wrong at many cases. Moreover the conjecture that the risk premium is non negative and increasing is at most times not true. This conjecture justifies the mean-variance portfolio optimization. So in a world with zero risk premium the optimal portfolio is not the mean-variance portfolio but the minimum-variance portfolio. In this optimization method security weights are independent of the expected security returns [6]. In our analysis we try both mean-variance and minimum-variance portfolio optimization.
The first step in every kind of analysis is to compute a rolling window covariance matrix. We choose the size of the window to be 2000 ($\approx 2 * number\ of\ stocks$). More concretely we calculate the covariance matrices for all the dates that Stock Exchange was open between 18.12.2007 - 21.11.2016 using the 2000 previous dates. So we end up with 2249 covariance matrices. The size of every matrix is 1018 x 1018, as we have 1018 ticker symbols in our dataset. We use the R package WGNA [7], a package that calculates correlation matrix in a very efficient way. After that step we can optimize out portfolio. But the problem is that the calculated covariance matrix is not **positive definite** to ensure the optimization problem is convex. So we could not solve the optimization problem with this covariance matrix. By using the R package Matrix and its function nearPD, we are able to calculate the nearest positive definite matrix to our initial covariance matrix and thus to overcome our problem.
As mentioned in class nobody should be using the sample covariance matrix for the purpose of portfolio optimization. It contains estimation error of the kind most likely to perturb a mean-variance optimizer. There is a method called shrinkage [8] that transforms the sample covariance matrix into a matrix that can be used to optimize our portfolio. One advantage of this method is that the matrix is always positive definite. Both in minimum-variance portfolio and in mean-variance portfolio we compare the expected return and the volatility when using the sample covariance matrix and the matrix after shrinkage.

## 3.1 Minimum variance Portfolio

As already mentioned, the minimum variance portfolio is the optimal portfolio if we assume that the risk premium is equal to zero. The classic Markowitz mean-variance optimal portfolio is the solution to:

$$\min_w w^T \Omega w - \frac{1}{\gamma} \mu^T w$$

$$Subject\ to : \sum_i w_i$$

where $w \in R^n$ are vectors of portfolio weights, $\mu^T w$ is sample mean portfolio return, and $w^T \Omega w$ is sample covariance of the portfolio returns. The constraint (summation equals to one) ensure that the investor invests all his money meaning full investment. Also we don't allow short selling, which adds the following non-negative optimization constraint:

$$\forall i : w_i \geq 0$$

Also by assuming that the risk premium is zero, is equivalent that the aversion to risk is infinite, thus $\gamma \to \infty$. With all those assumptions, the above optimization problem reduces to the following problem:

$$\min_w w^T \Omega w$$

$$Subject\ to : \sum_i w_i$$

We construct the portfolio only with stocks. In order to minimize the idiosyncratic risk we set an upper bound $\epsilon$ to every weight $w_i$, and to avoid having only a few stocks in our portfolio.
R package quadprog and its function solve.QP is used for portfolio optimization. So we get weights for every date between 18.12.2007 - 21.11.2016. At an average our model chooses approximately 80 ticker symbols in order to minimize the variance, therefore the risk. In figure 6 we report the daily in-sample expected return, and in figure 7 the corresponding standard deviation. We perform backtesting to evaluate the performance of the optimized portfolio. To do this we choose a second part of stock returns from our dataset and we call it out of sample evaluation sample. By separating the estimation sample and the evaluation sample we give a realistic assessment of the performance of the portfolio. It is crucial that there is no overlap between the estimation sample and the evaluation sample. Otherwise there is a look ahead bias. This means that we are optimizing the portfolio using more information than we actually have at the time of the investment decision. Such a look ahead bias leads to a too optimistic view on how good the optimized portfolio performs in real life. We choose a window of 200 for our evaluation sample. For example suppose we are in date 18.12.2007, we evaluate our portfolio in the returns that happened between the dates 19.12.2007 - 03.10.2008 (200 days). The results for out-of-sample expected returns and standard deviations are presented in figures 8, 9 Also the out-of-sample cumulative expected return is presented in figure 10.

## 3.2 Mean variance Portfolio

In this optimization problem, not only do we give as input to the function solve.QP the in-sample covariance matrix but also the in-sample expected returns of the ticker symbols, along with the constraints that we used when constructing the min-variance portfolio (no short sales, sum of weights equal

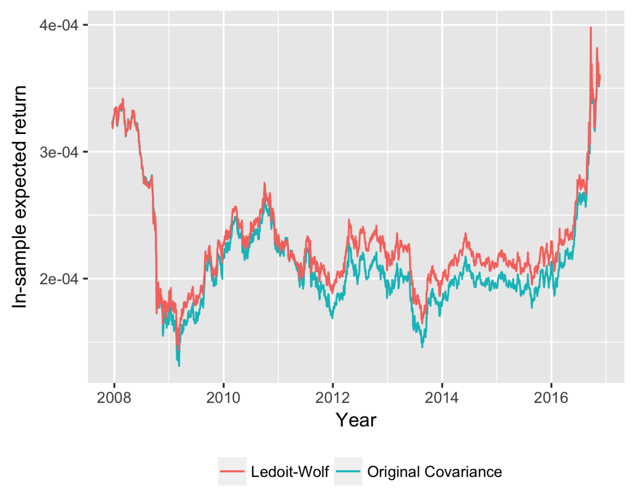

Figure 6: In sample expected returns (minimum-variance portfolio).

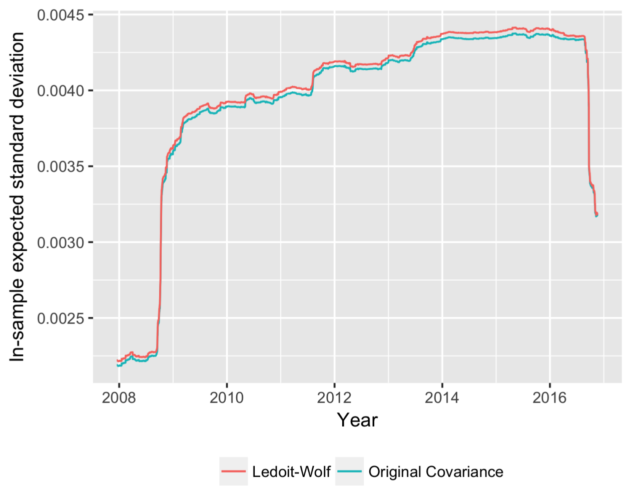

Figure 7: In sample expected standard deviations (minimum-variance portfolio).

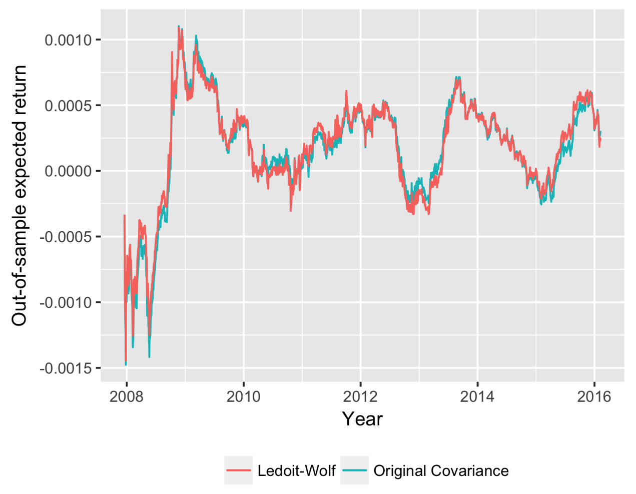

Figure 8: Out of sample expected returns (minimum-variance portfolio).

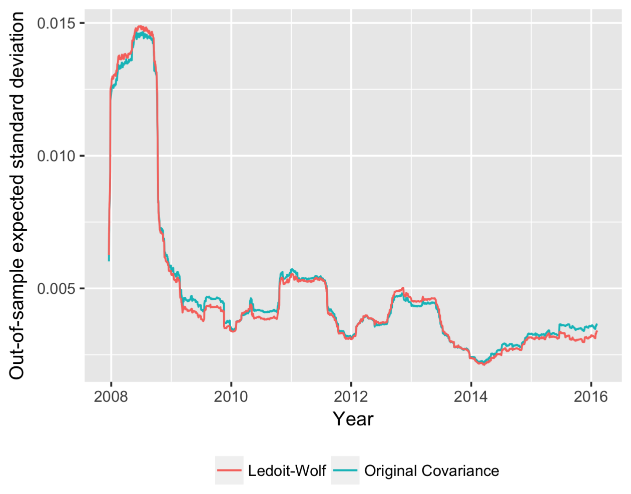

Figure 9: Out of sample expected standard deviations (minimum-variance portfolio).

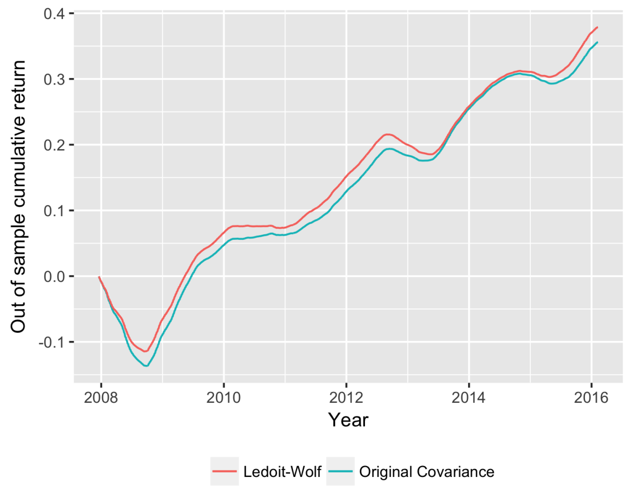

Figure 10: Out of sample cumulative expected return (minimum-variance portfolio).

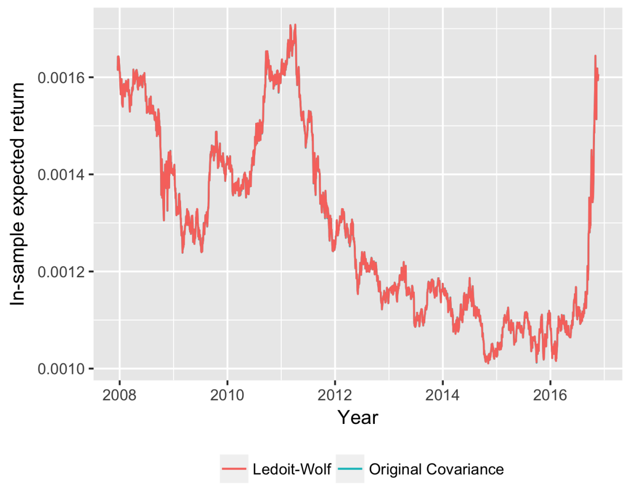

Figure 11: In sample expected returns (mean-variance portfolio).

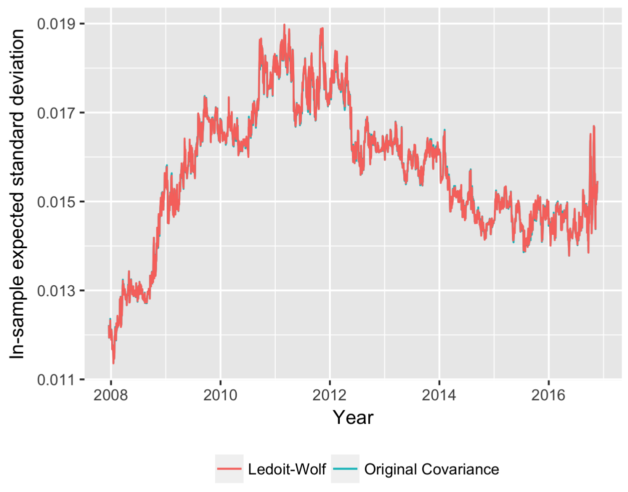

Figure 12: In sample expected standard deviations (mean-variance portfolio).

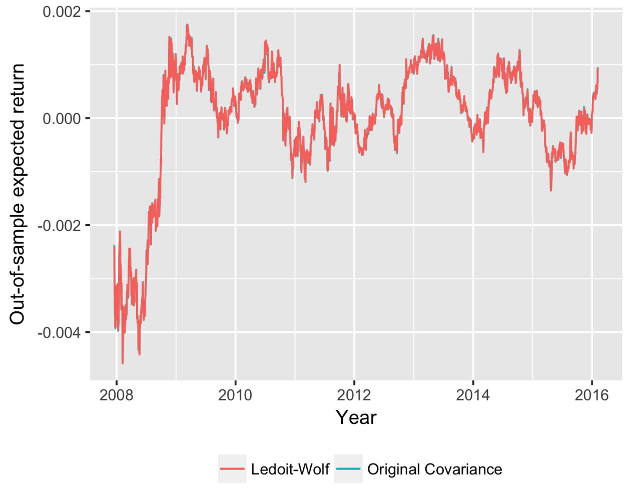

Figure 13: Out of sample expected returns (mean-variance portfolio).

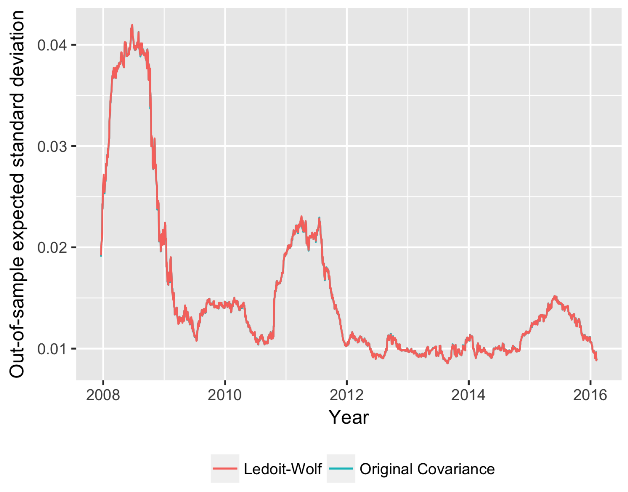

Figure 14: Out of sample expected standard deviations (mean-variance portfolio).

to one, and upper bounded weights to avoid idiosyncratic risk). As in the previous portfolio optimization we calculate weights for both matrices and compare the results. In figure 11 we report the daily in-sample expected return, and in figure 12 the corresponding standard deviation. As usual we perform backtesting to evaluate the portfolio performance. The results for out-of-sample expected returns and standard deviations are presented in figures 13, 14. Also the out-of-sample cumulative expected return is presented in figure 15.

## 3.3 Comparison

In all cases when we use the transformed covariance matrix through shrinkage [8], that gives better results compared to the results when we use the original covariance matrix. The minimum-variance portfolio which only goal was to minimize the variance outperforms the mean-variance portfolio. In the case of mean-variance portfolio, the results are almost equal for both covariance matrices. Also the fact that it tries to reach a previously specified expected return has a negative impact on the results because

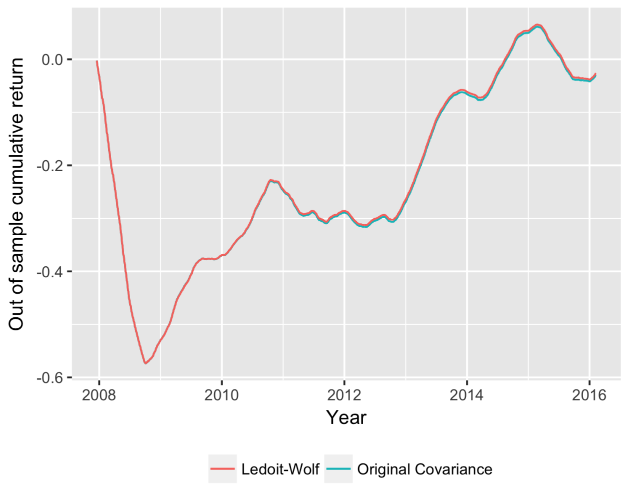

Figure 15: Out of sample cumulative expected return (mean-variance portfolio).

it does not diversify enough and as a result it performs poorly in the evaluation sample.

## 4 Machine Learning for forecasting financial time series

In this section we present two popular machine learning techniques, namely SVM (support vector machines) [9] and NN (neural networks) [10]. We use them both of them in order to predict the annual returns of S&P500 and APPLE and compare them to conclude which method gives more accurate predictions.

### 4.1 SVM

SVMs are among the best "off-the-shelf" supervised learning algorithm that can be used for classification and regression analysis. Since we want to predict the returns of a time series we use SVMs for regression analysis. To use this method we need to create features. One good way to do this is by lagging the time series of returns. So our 4 features are lags 1, 2, 3, 5 of the time series. SVMs are characterized by parameters that we need to tune. The most 2 important parameters are the kernel function and the cost. By testing and using all available kernel functions we ended up to the linear kernel. The optimal value for C (cost) is equal to one. One important thing is that SVM gives very good predictions when we use as input the rolling **annual returns** and not the daily or the weekly returns. So in our analysis we use the annual returns as input in order to create the features and then to predict annual returns. We use root mean square error (RMSE) to evaluate our model. The results are presented in Table 2. RMSE for both S&P500 and APPLE are low, that is our model predicts the annual returns accurately. In figures 16, 17 we can see an almost straight line between the values of actual returns and the corresponding predicted values. The low values of RMSE for both S&P500 and APPLE and the two figures are a good sign that our model can predict the future annual returns with high accuracy.

### 4.2 NN

Neural networks are organized in layers. Layers are made up of a number of interconnected nodes which contain an activation function. Patterns are presented to the network via the input layer, which communicates to one or more hidden layers where the actual processing is done via a system of weighted connections. The hidden layers then link to an output layer where the output is the answer. As before,

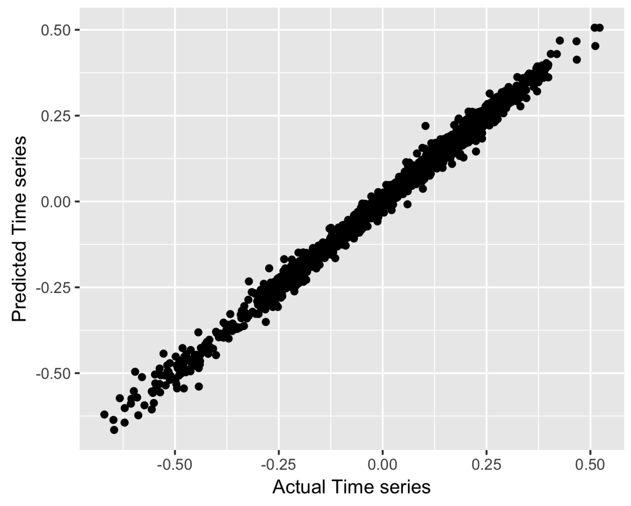

Figure 16: Predicted value with the corresponding real value of returns for S&P500.

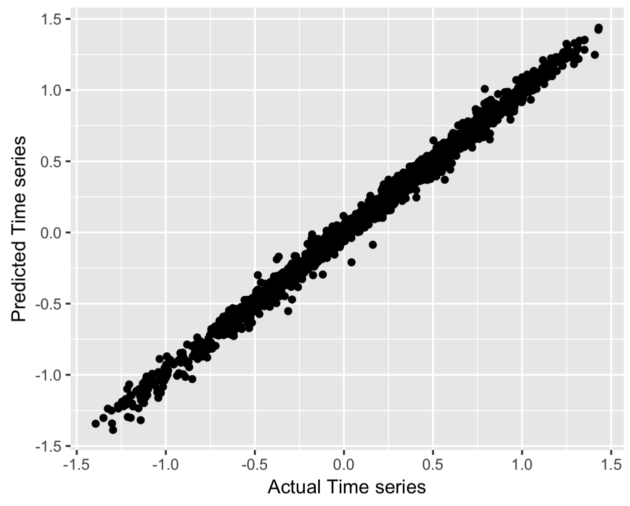

Figure 17: Predicted value with the corresponding real value of returns for APPLE.

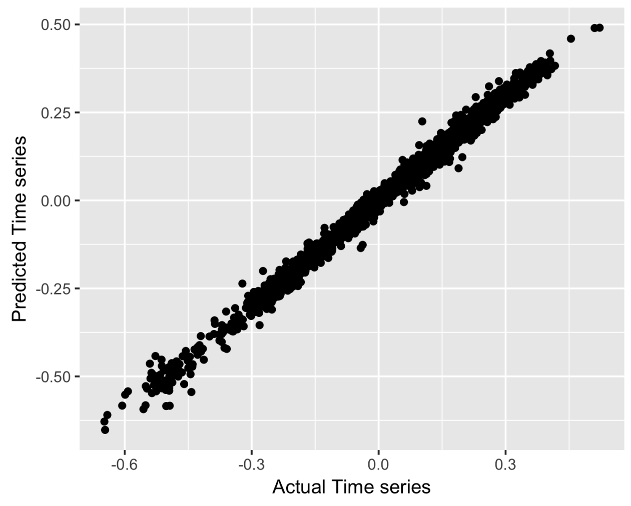

Figure 18: Predicted value with the corresponding real value of returns for S&P500.

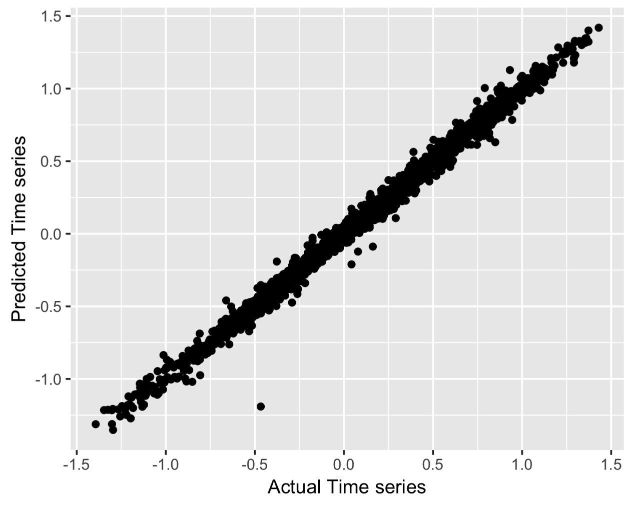

Figure 19: Predicted value with the corresponding real value of returns for APPLE.

we use the rolling **annual returns** because they give better predictions. We create from the annual returns, 4 features (lags 1, 2, 3, 5), so our neural network needs to have 4 neurons in the input layer. NN are characterized by parameters that we need to tune. The most 2 important parameters are the number of hidden layers and the weights decay. By testing we ended up to use 1 hidden layer of one neuron. Regarding weights decays, it is equal to 0.001 for S&P500 and 0 for APPLE. We use root mean square error (RMSE) to evaluate our model. The results are presented in Table 2. RMSE for both S&P500 and APPLE are low, that is our model predicts the annual returns accurately. In figures 18, 19 we can see an almost straight line between the values of actual returns and the corresponding predicted values. The low values of RMSE for both S&P500 and APPLE and the two figures are a good sign that our model can predict the future annual returns with high accuracy.

## 4.3 Comparison

From Table 2 it is obvious that both support vector machines and neural networks provide very good results. The fact that RMSE is a little smaller in SVM does not provide a certainty that SVM gives

Table 2: Comparing the accuracy of the 2 models

| **Machine Learning Technique** | **RMSE S&P500** | **RMSE APPLE** |
|---|---|---|
| SVM, linear kernel, C = 1 | 0.01635442 | 0.0401503 |
| NN, 1 hidden layer, decay = 0.001, (decay = 0 for APPLE) | 0.01640119 | 0.04210067 |

more accurate results. It is possible that this little difference in values is caused by luck.

## 5 Conclusions

Our analysis leads us to choose the minimum-variance portfolio and SVM or NN as the best ways to optimize our portfolio.

## References

[1] Cont, R. (2001). Empirical properties of asset returns: stylized facts and statistical issues. Quantitative Finance, 1, 223-236.

[2] Campbell, J., Lo, A., MacKinlay, A. (1997). The Econometrics of Financial Markets. Princeton: Princeton University Press.

[3] Mandelbrot, B. (1963). The variation of certain speculative prices. Journal of Business, 35, 394.

[4] Modern portfolio theory by Harry Markowitz

[5] Asset Allocation

[6] Roger G Clarke, Harindra de Silva, and Steven Thorley. Minimum-Variance Portfolios in the U.S. Equity Market. The Journal of Portfolio Management, Fall 2006, Vol. 33, No. 1: pp. 10-24

[7] Langfelder P, Horvath S (2008) WGCNA: an R package for weighted correlation network analysis. BMC Bioinformatics 2008, 9:559

[8] Olivier Ledoit, Michael Wolf. Honey, I Shrunk the Sample Covariance Matrix. The Journal of Portfolio Management, 2004, Vol 30, No. 4: pp. 110-119

[9] Support Vector Machines

[10] Neural Networks