15D013 Topics in Big Data Analytics II - Problem Set 1 - Computational Finance

Felix Gutmann

May 20, 2016

Prerequisites

In the following we will apply the following notation. Let R_t be the returns of a stock series defined as follows:

$$R_t := \frac{P_t}{P_{t-1}} - 1,$$

where P_t is the price of the stock at time t. In comparrison to that let r_t be the log returns of a series P defined as:

$$r_t := \log R_t = \log\left(\frac{P_t}{P_{t-1}} - 1\right) = \log(P_t) - \log(P_{t-1})$$

Exercise 1

Figure (1) depicts the the different acf values for the powers of the absolute returns series of the adjusted closing price of GOOGLE displaying the *Taylor-Effect*.

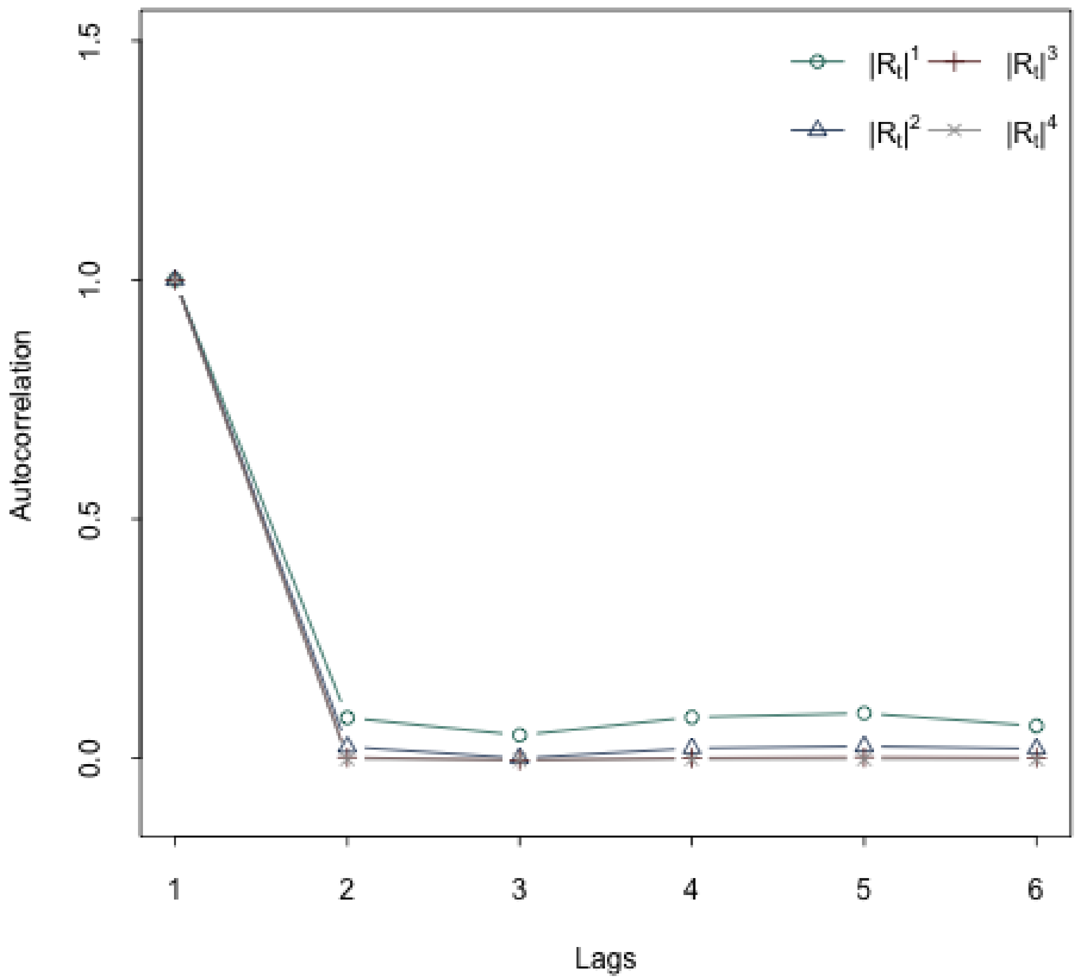


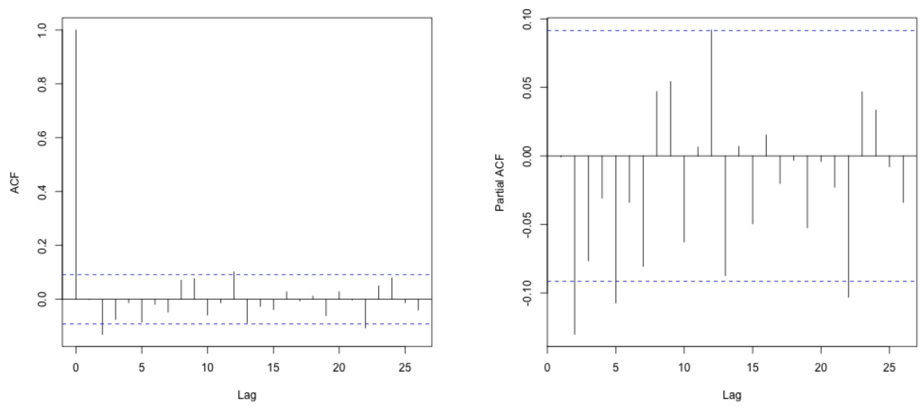
Figure 1: Autorrelations for $|R_t|^k$ for adjusted GOOGLE closing price (k=(1,...,4))

Exercise 2

Figure (2) depicts the monthly *Michigan Consumer Sentiment Index* (MCSI) from 01.01.1978 - 01.04.2016.

Figure 2: Michigan Consumer Sentiment Index (01.01.1978 - 01.04.2016)

Figure (3) displays the ACF (sub-figure (a)) and the PACF (sub-figure(b)) of the returns of the MCSI. We can observe that we may have a weak autoregressive component of order one (at most) and a moving average component. Conducting an *Augmented-Dicky-Fuller* test (ADF-test) ensures that the returns are stationary (see listing (3) in the appendix). The ARMA fitted by auto.arima() is an ARMA(1,2) (see listing (4) in the appendix) .



(a) ACF

(b) PACF

Figure 3: ACF and PACF of MCSI

Finally, the *Ljung-Box* on the squared returns on the MCSI-series (R_t^2) reveals ARCH effects in the series (see listing (5) in the appendix).. Hence, we fit a ARMA(1,2) + GARCH(1,1) model to the sentiment data (see listing (6) in the appendix for details). Since, a GARCH(1,1) has and ARCH(∞) representation this seems sufficient.

Exercise 3

In this exercise we perform support vector regression and a neural network regression to predict the S&P500 log returns.
herefore, let r_t^{SP} be the log returns of the S&P500 series. As predictor for the models we use textit"Divident to Price ratio" (denoted in the following by Γ_t) and some lagged values of both series . Furthermore, in the following let $p =$ number of lags.
For both models the training set is 80% of the full data and the test set 20% respectivaly.

Fitting SVM

Some notes on the support vector regression. For the model I solely consider *Radial-Basis-Functions* (rbf) also called *Gaussian - Kernels*.
Therefore, the tuning procedure is a grid search over three parameters: p for both series r_{t-p}^{SP} and Γ_{t-p}, C the cost parameter of the SVM controlling the cost of constraint violation and γ the parameter of the rbf-kernel.
SVM's are in general "costly" to tune. One can use the caret package. Nevertheless I wrote my own tuning functions (**smv.lag.grid.search()**). This function applies the tuning procedure just on training and test set for each parameter setting (Caret is doing stepwise tuning over multiple timeperiods with time-slice option, which increases time of grid search for each parameter setting). Table (1) gives an overview of the function arguments. Then function could be found in the attached R-script "auxilliary_functions.R".

Argument	**Meaning**	Default
predictor	Vector of external predictors	-
response	Vector of the response variable	-
response.name	String indicating the name of the response varialbe	"SP500"
lag.array	Vector containing a sequence of lags that should be tested	1:10
include.self	Logical indicating whether autoregressive data should be used	TRUE
gamma.array	Vector with values for γ	0.5:4
cost.array	Vector with C values	1:10
data.split	percentage defining the size of test set ($\in (0, 1)$	0.99
inner.trace	Logical prints the number of each sub iteration	FALSE
outer.trace	Logical prints the current result after each value of c	TRUE
final.trace	Logical prints the the best model	TRUE

Table 1: SVM tuning function arguments

The following listing shows the output of the final.trace and the best corresponding parameter setting.

Listing 1: SVM Tuning Result

```
***********************************
---------------------------------------------
FULL GRID SEARCH FINISHED
---------------------------------------------
 Number of Lags in Data: 7
 Training Observations: 1361
 Test Observations: 340
BEST PARAMETER SPECIFICATION:
GAMMA: 0.01
COST: 0.11
MSE: 0.001478756
---------------------------------------------

---------------------------------------------
***********************************
```

Neural Network

For the neural network I used the H20-package, which allows realtivaly easy to fit a deep neural network. This package is highly optimized and performs quite fast. However, we can tune a lot of different things in the neural networks. On page 19 et. seqq., [Candel et.al., 2004] gives an example on doing an automized grid search. a possble resulting model reveals the following results.

Listing 2: Deep Learning Result

```
H2ORegressionMetrics: deeplearning
** Reported on validation data. **
Description: Metrics reported on temporary (load-balanced) validation frame

MSE:  0.001525455
R2 :  -0.0969398
Mean Residual Deviance :  0.001525455
```

Exercise 4

As in the first exercise I used the adjusted closing price of GOOGLE from 01.01.2011 - 31.12.2015 as data input. Let r_t^G be the log-return of this series.
For the Genetic Programming the code got extended with a small chunk taking into account the squared series (as suggested in the exercise). The following table depicts the $h-$step ahead forecast result under the *Mean-Absolute-Error* (MAE) for the ARMA+GARCH and the Genetic Programming. The model was run for $h = 32$.
From the table we can see that both models perform mostly the same.

Model	**MAE**
ARMA(1,1) + GARCH(1,1)	0.01059
Genetic Programming	0.01057

Table 2: MAE for both models with $h = 32$

Exercise 5

For the first part of the exercise we show the general theorem that the sum of two random variables can be expressed as follows (see **Theorem 3.20** [Wassermann, 2004]):

$$Var(aX + bY) = a^2 Var(X) + b^2 Var(Y) + 2Cov(X,Y) \tag{1}$$

A proof follows by simple straight forward calculation. The variance of a random variable, say X, is defined as:

$$\begin{aligned} Var(X) &= \mathbb{E}\left[(X - \mathbb{E}[X])^2\right] \\ &= \mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2 \end{aligned} \tag{2}$$

Let X_1 and X_2 be two random variables. Furthermore, let Z be another random variable, such that $Z = a_1 X_1 + a_2 X_2$, where a_1 and a_2 are constants. Proceed by applying definition (2) to Z, which leads to:

$$\begin{aligned} Var(Z) &= Var(a_1 X_1 + a_2 X_2) \\ &= \mathbb{E}\left[(a_1 X_1 + a_2 X_2)^2\right] - \mathbb{E}\left[(a_1 X_1 + a_2 X_2)\right]^2 \\ &= \mathbb{E}\left[\sum_{i=1}^{2}\sum_{j=1}^{2} a_i a_j X_i X_j\right] - \sum_{i=1}^{2}\sum_{j=1}^{2} a_i a_j \mathbb{E}[X_i]\,\mathbb{E}[X_j] \end{aligned}$$

By linearity of the expexted value (first term) we proceed as follows:

$$\begin{aligned} &= \sum_{i=1}^{2}\sum_{j=1}^{2} a_i a_j \mathbb{E}[X_i X_j] - \sum_{i=1}^{2}\sum_{j=1}^{2} a_i a_j \mathbb{E}[X_i]\,\mathbb{E}[X_j] \\ &= \sum_{i=1}^{2}\sum_{j=1}^{2} \left(a_i a_j \mathbb{E}[X_i X_j] - a_i a_j \mathbb{E}[X_i]\,\mathbb{E}[X_j]\right) \\ &= \sum_{i=1}^{2}\sum_{j=1}^{2} Cov(a_i X_i, a_j X_j) \end{aligned}$$

We notice that we can express the sum as the sum over the covariances. Finally, let $Var(X_i) = Cov(X_i, X_i)$ and notice that $Cov(X_i, X_j) = Cov(X_j, X_i)$. We can rexpress the variace of sum of random variables as proposed in the begining by splitting up the covariance terms in the following way:

$$\begin{aligned} Var(a_i X_i + a_j X_j) &= \sum_{i=1}^{2} Cov(X_i, X_i) + 2 \sum_{i=1, i\neq j}^{2} \sum_{j=1,\neq j}^{2} Cov(X_i, X_j) \\ Var(aX_i + a_j X_j) &= \sum_{i=1}^{2} Var(X_i) + 2 \sum_{i=1, i\neq j}^{2} \sum_{j=1,\neq j}^{2} Cov(X_i, X_j) \end{aligned}$$

For the second part of the exercise, let $\sigma_i^2 = Var(X_i)$, $\sigma_j^2 = Var(X_j)$ and $\rho_{i,j} = Cov(X_i, X_j)$. We want to show the proposed statement:

$$Var\left(\sum_{i=1}^{n} a_i X_i\right) = \sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j \sigma_i \sigma_j \rho_{i,j}$$

We show the former expression within in an iterative process. **For** $n = 2$ **:**

$$\begin{aligned}
Var\left(\sum_{i=1}^{2} a_i X_i\right) &= Var(a_1X_1 + a_2X_2) \\
&= a_1^2 Var(X_1) + a_2^2 Var(X_2) + 2a_1a_2Cov(X_1, X_2) \\
&= a_1^2\sigma_1^2 + a_2^2\sigma_2^2 + 2a_1a_2\rho_{1,2} \\
&= a_1a_1\sigma_1\sigma_1 + a_2a_2\sigma_2\sigma_2 + a_1a_2\rho_{1,2}
\end{aligned}$$

Note, since $\rho_{i,i}$ is equal to one, we can add it to the first two terms. The expression becomes:

$$\begin{aligned}
&= a_1a_1\sigma_1\sigma_1\rho_{1,1} + a_2a_2\sigma_2\sigma_2\rho_{2,2} + a_1a_2\rho_{1,2} \qquad (3) \\
&= \sum_{i=1}^{2}\sum_{j=1}^{2} a_i a_j \sigma_i \sigma_j \rho_{i,j}
\end{aligned}$$

Finally, we generalize it to n:

$$Var\left(\sum_{i=1}^{n} a_i X_i\right) = Var(a_1X_1 + a_2X_2 + \ldots, a_nX_n)$$

Using the representation of (3) we get:

$$\begin{aligned}
&= a_1a_1\sigma_1\sigma_1\rho_{1,1} + \cdots + a_na_n\sigma_n\sigma_n\rho_{n,n} \\
&+\ 2a_1a_2\rho_{1,2} + \ldots + 2a_{(n-1)}a_n\sigma_n\sigma_{(n-1)}\rho_{(n-1),n} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j \sigma_i \sigma_j \rho_{i,j}
\end{aligned}$$

Exercise 6

Usually in regression with linear basis functions we want to find a linear function, such that the error of modeling $\mathbb{E}[X|Z]$ is minimized. Let ℓ be a loss function and $\hat{X}$ be the forecast of the model for X at time $t+h$. In particular let ℓ be squared loss function (for convenience):

$$\mathbb{E}\left[\ell(X,\hat{X})\middle|T\right] = \mathbb{E}\left[\left(X-\hat{X}\right)^2\middle|T\right]$$

Let $\mathbb{E}[X|T] = \mu_{t+h|T}$ denote the mean of the forecast. By adding and substracting in the last equation we obtain:

$$\begin{aligned}
&= \mathbb{E}\left[\left(X-\mu_{t+h|T}+\mu_{t+h|T}-\hat{X}\right)^2\middle|T\right] \\
&= \mathbb{E}\left[\left(X-\mu_{t+h|T}+\mu_{t+h|T}-\hat{X}\right)^2\middle|T\right] \\
&= \mathbb{E}\left[\left(X-\mu_{t+h|T}\right)^2+\left(\mu_{t+h|T}-\hat{X}\right)+2\left(X-\mu_{t+h|T}\right)\left(\mu_{t+h|T}-\hat{X}\right)\middle|T\right]
\end{aligned}$$

Note that the last term is going to zero and hence we end up with:

$$= \mathbb{E}\left[\left(X-\mu_{t+h|T}\right)^2+\left(\mu_{t+h|T}-\hat{X}\right)\middle|T\right]$$

The first expression is the variance. Furthermore, the second expression is minmized when the forecast $\hat{X}$ is equal to the the expected value of $\mathbb{E}[X|T]$, which is in fact a the linear regression on the information Z, since Z is know at time T:

$$\begin{aligned}
&= \mathbb{E}\left[\left(X-\mu_{t+h|T}\right)^2+\left(\mu_{t+h|T}-\hat{X}\right)\middle|T\right] \\
&= \mathbb{E}\left[\left(X-\mu_{t+h|T}\right)^2\right]+\mathbb{E}\left[\left(\mu_{t+h|T}-\hat{X}\right)\middle|T\right] \\
&= \sigma^2_{t+h|T}+\mathbb{E}\left[\left(\mu_{t+h|T}-\hat{X}\right)\middle|T\right]
\end{aligned}$$

Sources

[**Wasserman, 2004**] Wasserman, Larry (2004): All of Statistics: A Concise Course in Statistical Inference, Springer Publishing Company, Incorporated

[**Candel et.al., 2004**] Candel, Arno;Lanford, Jessica; LeDell ,Erin; Parmar, Viraj; Arora, Anisha (2015): Deep Learning with H2O

Appendix

Listing 3: Augmented-Dicky-Fuller test on MCSI returns data

```
Augmented Dickey-Fuller Test

data:  sentiment.index[, 1]
Dickey-Fuller = -8.7801, Lag order = 7, p-value = 0.01
alternative hypothesis: stationary
```

Listing 4: ARMA-Model

```
Series: sentiment.index
ARIMA(1,0,2) with zero mean

Coefficients:
         ar1      ma1      ma2
      0.5580  -0.5796  -0.1332
s.e.  0.1663   0.1669   0.0533

sigma^2 estimated as 0.002439:  log likelihood=729.33
AIC=-1450.67   AICc=-1450.58   BIC=-1434.15
```

Listing 5: Ljung-Box test on squared MCSI returns

```
Box-Ljung test

data:  sentiment.squared
X-squared = 4.0884, df = 1, p-value = 0.04318
```

Listing 6: ARMA Garch on the MCSI returns

```
Title:
 GARCH Modelling

Call:
 garchFit(formula = UMCSENT ~ arma(1, 2) + garch(1, 1), data = sentiment.index,
    trace = FALSE)

Mean and Variance Equation:
 data ~ arma(1, 2) + garch(1, 1)
<environment: 0x116c156d0>
 [data = sentiment.index]

Conditional Distribution:
 norm

Coefficient(s):
         mu          ar1          ma1          ma2        omega
alpha1       beta1
 9.5295e-04   5.5250e-01  -6.3146e-01  -9.2143e-02   5.8968e-05
9.2567e-02   8.8507e-01

Std. Errors:
 based on Hessian

Error Analysis:
         Estimate  Std. Error  t value Pr(>|t|)
mu      9.530e-04   6.532e-04    1.459  0.14459
ar1     5.525e-01   2.029e-01    2.723  0.00647 **
ma1    -6.315e-01   2.039e-01   -3.097  0.00196 **
ma2    -9.214e-02   6.255e-02   -1.473  0.14075
omega   5.897e-05   2.886e-05    2.043  0.04102 *
alpha1  9.257e-02   2.319e-02    3.992 6.57e-05 ***
beta1   8.851e-01   2.468e-02   35.857  < 2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Log Likelihood:
 755.2853    normalized:  1.645502

Description:
 Fri May 20 20:43:25 2016 by user:
```