QF301. Homework #3.

2022-10-19

I pledge on my honor that I have not given or received any unauthorized assistance on this assignment/examination. I further pledge that I have not copied any material from a book, article, the Internet or any other source except where I have expressly cited the source.

By filling out the following fields, you are signing this pledge. No assignment will get credit without being pledged.

Name: Arjun Koshal

CWID: 10459064

Date: 10/21/2022

Instructions

In this assignment, you should use R markdown and Jupyter notebook to answer the questions below. Simply type your R and Python code into embedded chunks as shown above. When you have completed the assignment, knit both .Rmd and .ipynb files into a PDFs, and upload both .pdf files as well as the source code in .Rmd and .ipynb formats to Canvas.

```
CWID = 10459064 #Place here your Campus wide ID number, this will personalize
#your results, but still maintain the reproducible nature of using seeds.
#If you ever need to reset the seed in this assignment, use this as your seed
#Papers that use -1 as this CWID variable will earn 0's so make sure you change
#this value before you submit your work.
personal = CWID %% 10000
set.seed(personal)#You can reset the seed at any time in your code,
#but please always set it to this seed.
```

Question 1 (20pt)

Question 1.1

Use the quantmod package to obtain the daily adjusted close prices 2 different stocks. You should have at least two years of data for both assets. You should inspect the dates for your data to make sure you are including everything appropriately. Create a data frame of the daily log returns both both stocks along with the lagged returns (2 lags). You may wish to remove the date from your data frame for later analysis. Print the first 6 lines of your data frame. (You may use the same two stocks as in Homework 2.)

Solution:

```
library(quantmod)

getSymbols(c("TSLA","AMZN"),from="2018-01-01",to="2020-12-31")
```

```
## [1] "TSLA" "AMZN"
```

```
rTSLA = as.numeric(dailyReturn(TSLA$TSLA.Adjusted,type="log"))
rAMZN = as.numeric(dailyReturn(AMZN$AMZN.Adjusted,type="log"))

rTSLA1 = as.numeric(lag(rTSLA,k=1))[-(1:2)]
rTSLA2 = as.numeric(lag(rTSLA,k=2))[-(1:2)]
rTSLA = as.numeric(rTSLA)[-(1:2)]
rAMZN1 = as.numeric(lag(rAMZN,k=1))[-(1:2)]
rAMZN2 = as.numeric(lag(rAMZN,k=2))[-(1:2)]
rAMZN = as.numeric(rAMZN)[-(1:2)]

df = data.frame(rTSLA,rAMZN,rTSLA1,rAMZN1,rTSLA2,rAMZN2)
head(df)
```

Question 1.2

Split your data into training and testing sets (80% training and 20% test).

Linearly regress one of your stock returns as a function of the lagged returns (2 lags) for both stocks. This should be of the form

$$r_{1,t} = \beta_0 + \beta_{1,1} r_{1,t-1} + \beta_{1,2} r_{1,t-2} + \beta_{2,1} r_{2,t-1} + \beta_{2,2} r_{2,t-2}$$

. Evaluate the performance of this model with the mean squared error on the test data.

Solution:

```
train = sample(length(rTSLA),0.8*length(rTSLA),replace=FALSE)

lin.reg = glm(rTSLA ~ rTSLA1 + rTSLA2 + rAMZN1 + rAMZN2 , data=df,
              subset=train)
summary(lin.reg)
```

```
##
## Call:
## glm(formula = rTSLA ~ rTSLA1 + rTSLA2 + rAMZN1 + rAMZN2, data = df,
##     subset = train)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
##      0       0       0       0       0
##
## Coefficients: (2 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.000e+00  0.000e+00     NaN      NaN
## rTSLA1      1.000e+00  0.000e+00     Inf   <2e-16 ***
## rTSLA2             NA         NA      NA       NA
## rAMZN1      4.614e-19  0.000e+00     Inf   <2e-16 ***
## rAMZN2             NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for gaussian family taken to be 0)
##
##     Null deviance: 1.1565  on 601  degrees of freedom
## Residual deviance: 0.0000  on 599  degrees of freedom
## AIC: -Inf
##
## Number of Fisher Scoring iterations: 1
```

```
pred=predict(lin.reg,df[-train,])
lin.MSE = mean((pred-df$rTSLA[-train])^2) #Test MSE
lin.MSE
```

```
## [1] 0
```

Question 2 (35pt)

Question 2.1

Using the same data, train/test split ratio, and consider the same regression problem as in Question 1.2. Create a feed-forward neural network with a single hidden layer (2 hidden nodes) densely connected to the inputs. You may choose any activation functions you wish.

Question 2.1.1

Write the mathematical structure for this neural network.

Solution:

$$r_{1,t} = b^{(2)} + w_1^{(2)} h_1 + w_2^{(2)} h_2$$

$$h_1 = g(b_0^{1,(1)} + w_{1,1}^{1,(1)} r_{1,t-1} + w_{1,2}^{1,(1)} r_{1,t-2} + w_{2,1}^{1,(1)} r_{2,t-1} + w_{2,2}^{1,(1)} r_{2,t-2})$$

$$h_2 = g(b_0^{2,(1)} + w_{1,1}^{2,(1)} r_{1,t-1} + w_{1,2}^{2,(1)} r_{1,t-2} + w_{2,1}^{2,(1)} r_{2,t-1} + w_{2,2}^{2,(1)} r_{2,t-2})$$

Question 2.1.2

Train this neural network on the training data.
Evaluate the performance of this model with the mean squared error on the test data.

Solution:

See python code

Question 2.2

Using the same train/test split and consider the same regression problem as in Question 1.2. Train and test another neural network of your own design.

Solution:

See python code

Question 2.3

How would you determine if your models are overfitting the data? Do you suspect this is happening in your models?

Solution:

Overfitting is when the test error is significantly larger than the training error. When testing my code, I did not observe overfitting in my models.

Question 3 (35pt)

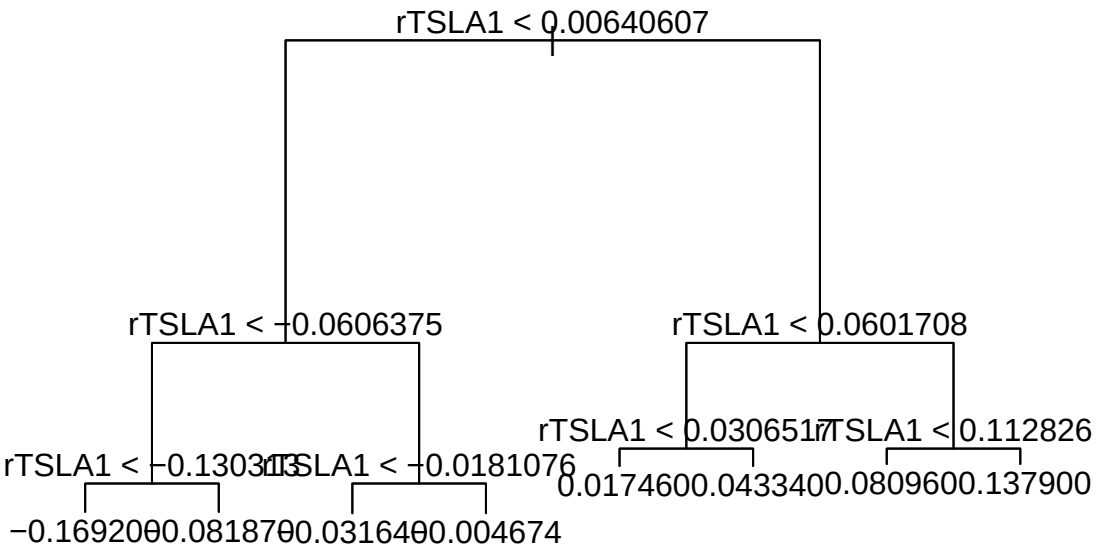
Question 3.1

Using the same data, train/test split, and consider the same regression problem as in Question 1.2. Train a decision tree on the training data. Evaluate the performance of this model with the mean squared error on the test data.

Solution:

```
library("tree")

tree.reg = tree(rTSLA ~ rTSLA1 + rTSLA2 + rAMZN1 + rAMZN2 , data=df,
                subset=train)
plot(tree.reg)
text(tree.reg)
```



```
mean((df$rTSLA[train] - predict(tree.reg, df[train,]))^2)
```

```
## [1] 9.444458e-05
```

```
pred=predict(tree.reg,df[-train,])
tree.MSE = mean((pred-df$rTSLA[-train])^2)
tree.MSE
```

```
## [1] 9.870546e-05
```

Question 3.2

Using the same train/test split and consider the same regression problem as in Question 1.2. Train and test a random forest with 250 trees and 2 predictors.

Solution:

```
library("randomForest")

rf.reg = randomForest(rTSLA ~ rTSLA1 + rTSLA2 + rAMZN1 + rAMZN2 , data=df,
                      subset=train , ntree=250 , mtry=2 , importance = TRUE)

mean((df$rTSLA[train] - predict(rf.reg, df[train,]))^2)
```

```
## [1] 3.901435e-06
```

```
pred=predict(rf.reg,df[-train,])
rf.MSE = mean((pred-df$rTSLA[-train])^2)
rf.MSE
```

```
## [1] 2.069322e-06
```

Question 3.3

How would you determine if your models are overfitting the data? Do you suspect this is happening in either the decision tree or random forest?

Solution:

Overfitting is when the test error is significantly larger than the training error. When testing my code, the random forest gives slight indications of overfitting.

Question 4 (10pt)

Question 4.1

Consider the same regression problem as in Question 1.2. Of the methods considered in this assignment, which would you recommend in practice? Explain briefly (1 paragraph) why you choose this fit.

Solution:

I would choose the linear regression as the test MSE is only slightly worse. Since the model is easy to implement, I would choose this fit over the others.