Exercise 5 R Markdown

Santanu Mukherjee

07/26/2022

R Markdown

Question

This question should be answered using the *"Weekly"* data set, which is part of the **"ISLR"** package. This data is similar in nature to the *"Smarket"* data from this chapter's lab, except that it contains 1089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010.

Question a

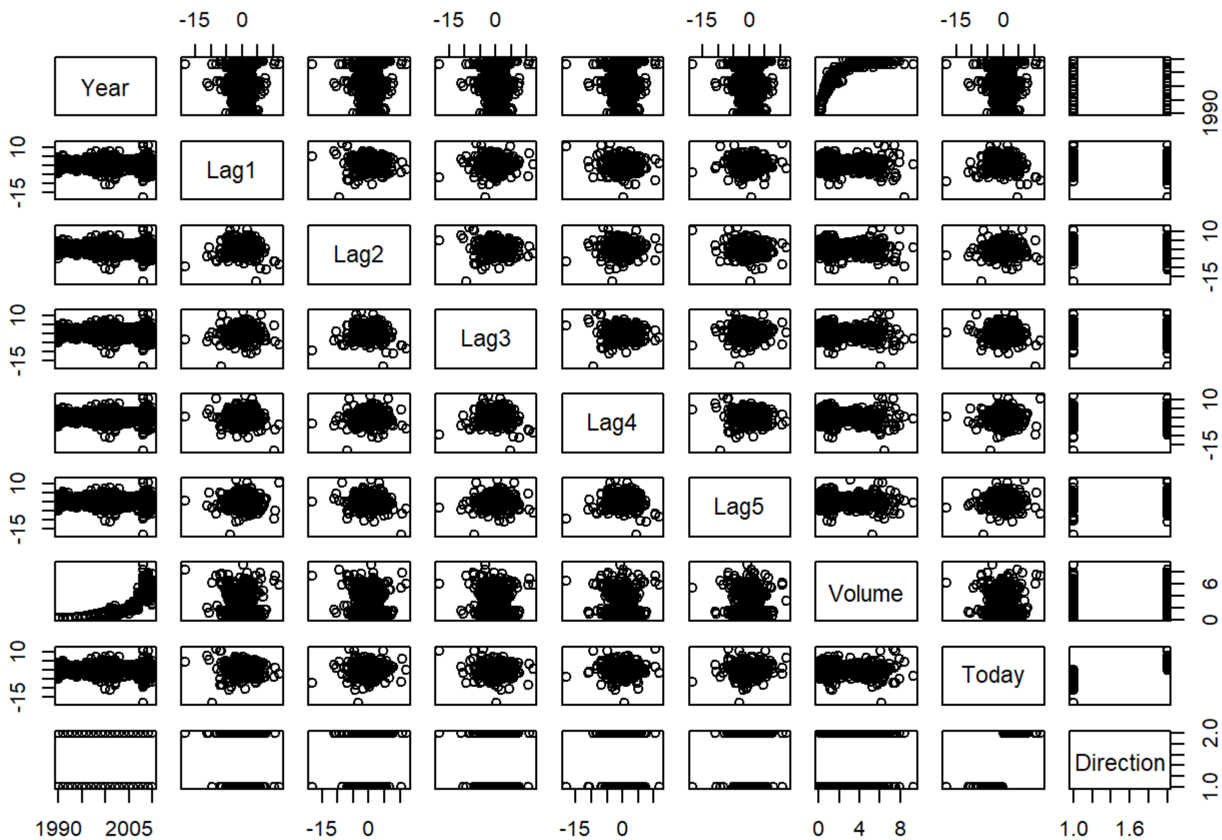
a. Produce some numerical and graphical summaries of the *"Weekly"* data. Do there appear to be any patterns ?

Answer a

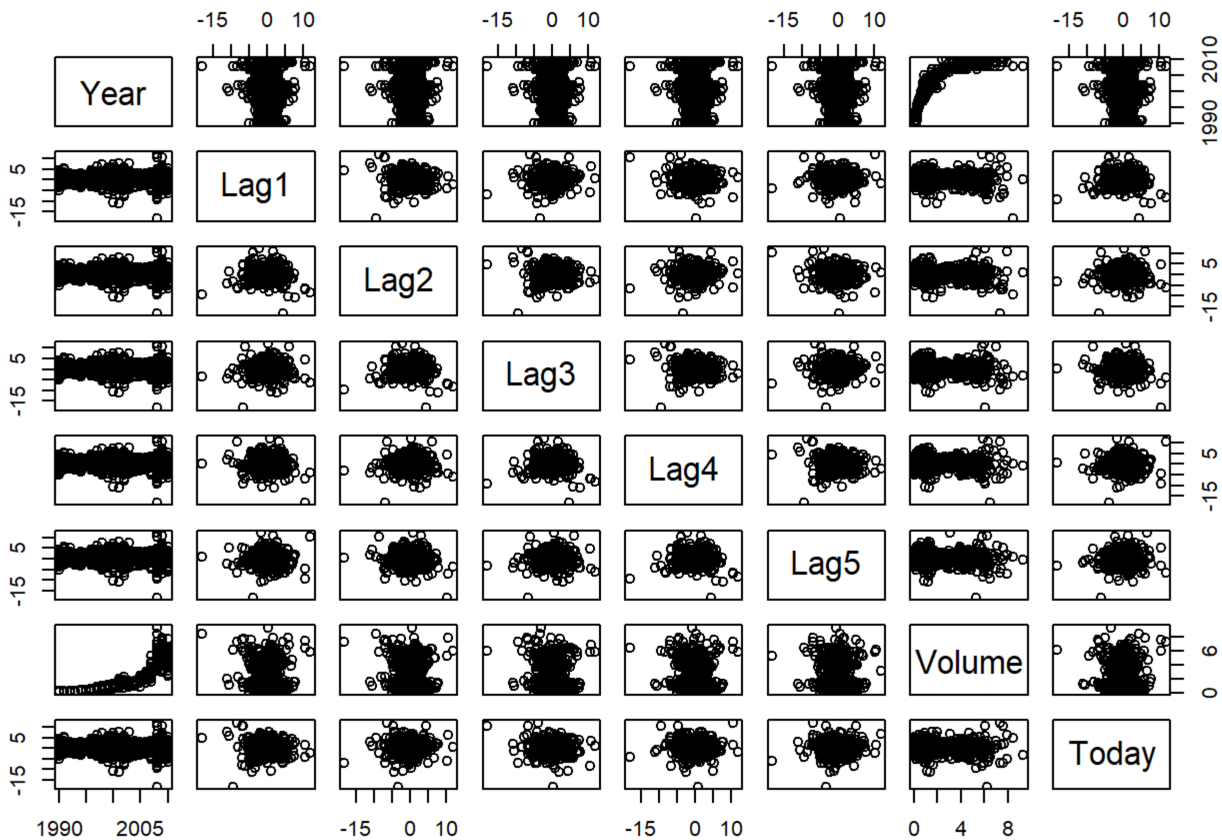
```
##Part (a) Weekly Data Summary
library(ISLR)
summary(Weekly)
```

```
##       Year           Lag1               Lag2               Lag3         
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950  
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580  
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410  
##  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472  
##  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090  
##  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260  
##       Lag4               Lag5              Volume            Today         
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747   Min.   :-18.1950  
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202   1st Qu.: -1.1540  
##  Median :  0.2380   Median :  0.2340   Median :1.00268   Median :  0.2410  
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462   Mean   :  0.1499  
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373   3rd Qu.:  1.4050  
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821   Max.   : 12.0260  
##  Direction 
##  Down:484  
##  Up  :605  
##            
##            
##            
##            
```

```
pairs(Weekly)
```



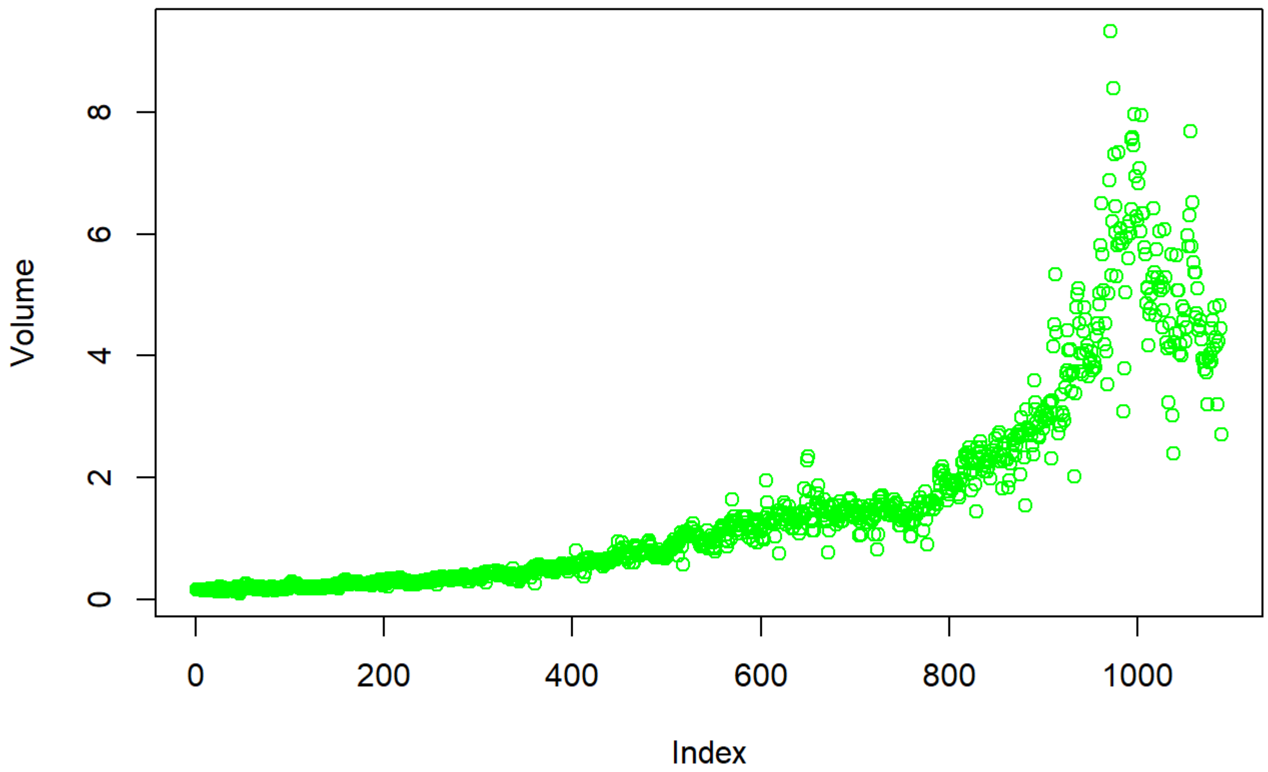

```
pairs(Weekly[,-9])
```




```
cor(Weekly[, -9])
```

```
##               Year        Lag1        Lag2        Lag3         Lag4
## Year    1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923
## Lag1   -0.03228927  1.000000000 -0.07485305  0.05863568 -0.071273876
## Lag2   -0.03339001 -0.074853051  1.00000000 -0.07572091  0.058381535
## Lag3   -0.03000649  0.058635682 -0.07572091  1.00000000 -0.075395865
## Lag4   -0.03112792 -0.071273876  0.05838153 -0.07539587  1.000000000
## Lag5   -0.03051910 -0.008183096 -0.07249948  0.06065717 -0.075675027
## Volume  0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617
## Today  -0.03245989 -0.075031842  0.05916672 -0.07124364 -0.007825873
##                Lag5      Volume        Today
## Year   -0.030519101  0.84194162 -0.032459894
## Lag1   -0.008183096 -0.06495131 -0.075031842
## Lag2   -0.072499482 -0.08551314  0.059166717
## Lag3    0.060657175 -0.06928771 -0.071243639
## Lag4   -0.075675027 -0.06107462 -0.007825873
## Lag5    1.000000000 -0.05851741  0.011012698
## Volume -0.058517414  1.00000000 -0.033077783
## Today   0.011012698 -0.03307778  1.000000000
```

```
attach(Weekly)
plot(Volume, col="green")
```



Step by Step Observations:

1. The *Summary* and subsequently the *pairs* showed that the variable “Direction” was insignificant.

2. So, then I got the correlation matrix with all variables except **Direction**.

3. The correlations between the **“lag”** variables and **Today* variable are close to zero.**

4.** The correlation between variables **“Year”** and **“Volume”** is the only significant one.

5. So, I have done plot “Volume”, and I see that is increasing over time.

Question b

b. Use the full data set to perform a logistic regression with *“Direction”* as the response and the five lag variables plus *“Volume”* as predictors. Use the summary function to print the results. Do any of the predictors appear to be statistically significant ? If so, which ones ?

Answer b

```
##Part (b) Logistic Regression

set.seed(1)
log.reg <-glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume, data=Weekly,family=binomial)
summary(log.reg)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial, data = Weekly)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6949  -1.2565   0.9913   1.0849   1.4579
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
## Lag1        -0.04127    0.02641  -1.563   0.1181
## Lag2         0.05844    0.02686   2.175   0.0296 *
## Lag3        -0.01606    0.02666  -0.602   0.5469
## Lag4        -0.02779    0.02646  -1.050   0.2937
## Lag5        -0.01447    0.02638  -0.549   0.5833
## Volume      -0.02274    0.03690  -0.616   0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

It seems that “Lag2” is the only predictor which is statistically significant at $\alpha = 0.05$ as its p-value is less than 0.05.

Question c

c. Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.

Answer c

```
##Part (c) Confusion Matrix

prob.log.reg <- predict(log.reg, type = "response")
pred.log.reg <- rep("Down", length(prob.log.reg))
pred.log.reg[prob.log.reg > 0.5] <- "Up"
table(pred.log.reg, Direction)
```

```
##             Direction
## pred.log.reg Down  Up
##         Down   54  48
##         Up    430 557
```

Based on the results of the table above, We may conclude that the percentage of correct predictions (Down * Down & Up *Up) on the training data is $(54 + 557)/1089$ which is equal to 56.11%. So, we can say that 43.89% is the training error rate.

If we look at the data from another angle , meaning we could also conclude that for the *weeks* when the market goes **Up**, the model is right 92.07% of the time $(557/(48 + 557))$.
Similarly, for the *weeks* when the market goes **Down**, the model is right only 11.16% of the time $(54/(54 + 430))$.

Question d

d. Now fit the logistic regression model using a training data period from 1990 to 2008. Compute the confusion matrix and the overall fraction of correct predictions for the held out data (that is, the data from $2009 to 2010$).

Answer d

```
##Part (d) Logistic regression with data from 2009-2010 and the only predictor being "Lag2"

train.data <- (Year < 2009)
Weekly.2009.2010 <- Weekly[!train.data, ]
Direction.2009.2010 <- Direction[!train.data]
log.reg.lag2 <- glm(Direction ~., data = Weekly, family = binomial, subset = train.data)
summary(log.reg.lag2)
```

```
##
## Call:
## glm(formula = Direction ~ ., family = binomial, data = Weekly,
##     subset = train.data)
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -1.883e-03  -2.000e-08   2.000e-08   2.000e-08   1.570e-03
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.258e+03  1.437e+06  -0.002    0.998
## Year         1.632e+00  7.213e+02   0.002    0.998
## Lag1        -4.830e+00  1.233e+03  -0.004    0.997
## Lag2         7.448e+00  1.083e+03   0.007    0.995
## Lag3         1.445e+00  9.872e+02   0.001    0.999
## Lag4         7.540e-01  6.473e+02   0.001    0.999
## Lag5         1.185e+01  1.320e+03   0.009    0.993
## Volume      -8.664e+00  4.189e+03  -0.002    0.998
## Today        8.160e+02  1.686e+04   0.048    0.961
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1.3547e+03  on 984  degrees of freedom
## Residual deviance: 9.2831e-06  on 976  degrees of freedom
## AIC: 18
##
## Number of Fisher Scoring iterations: 25
```

```
##Part (d) Confusion Matrix

prob2.log.reg <- predict(log.reg.lag2, Weekly.2009.2010, type = "response")
pred2.log.reg <- rep("Down", length(prob2.log.reg))
pred2.log.reg[prob2.log.reg > 0.5] <- "Up"
table(pred2.log.reg, Direction.2009.2010)
```

```
##                Direction.2009.2010
## pred2.log.reg Down Up
##          Down   43  0
##          Up      0 61
```

Based on the results of the table above, we can conclude that the percentage of correct predictions on the test data is $(43 + 61)/104$ (Down * Down & Up * Up) which is equal to 100%. So, we can say that the test error rate is 0%.

If we look at the data from another angle , meaning we could also conclude that for the *weeks* when the market goes **Up** or **Down**, the model is correct 100% of the time.

Question e

e. Repeat (d) using LDA.

Answer e

```
##Part (e) 1st part - Repeating part (d) using LDA

library(MASS)
fit.lda <- lda(Direction ~., data = Weekly, subset = train.data)
fit.lda
```

```
## Call:
## lda(Direction ~ ., data = Weekly, subset = train.data)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##          Year         Lag1        Lag2       Lag3       Lag4       Lag5
## Down 1999.295  0.289444444 -0.03568254 0.17080045 0.15925624 0.21409297
## Up   1998.853 -0.009213235  0.26036581 0.08404044 0.09220956 0.04548897
##        Volume     Today
## Down 1.266966 -1.687018
## Up   1.156529  1.603956
##
## Coefficients of linear discriminants:
##                  LD1
## Year   -0.0106936942
## Lag1    0.0003606345
## Lag2    0.0169738374
## Lag3    0.0295058746
## Lag4   -0.0155046298
## Lag5   -0.0279798179
## Volume  0.0587137582
## Today   0.6322256639
```

```
##Part (e) 2nd part - Repeating part (d) using LDA

pred.e.lda <- predict(fit.lda, Weekly.2009.2010)
table(pred.e.lda$class, Direction.2009.2010)
```

```
##       Direction.2009.2010
##        Down Up
##   Down   36  0
##   Up      7 61
```

Based on the results, we conclude that the output is similar but not exactly the same as part (d), which means in this case, the **Logistic Regression** and **LDA** has different results.

From the results of the table above, we can conclude that the percentage of correct predictions on the test data is $(36 + 61)/104$ (Down * Down & Up * Up) which is equal to 93.2%. So, we can say that the test error rate for LDA is 6.8%.

Question f

f. Repeat (d) using Partial least squares discriminant analysis.

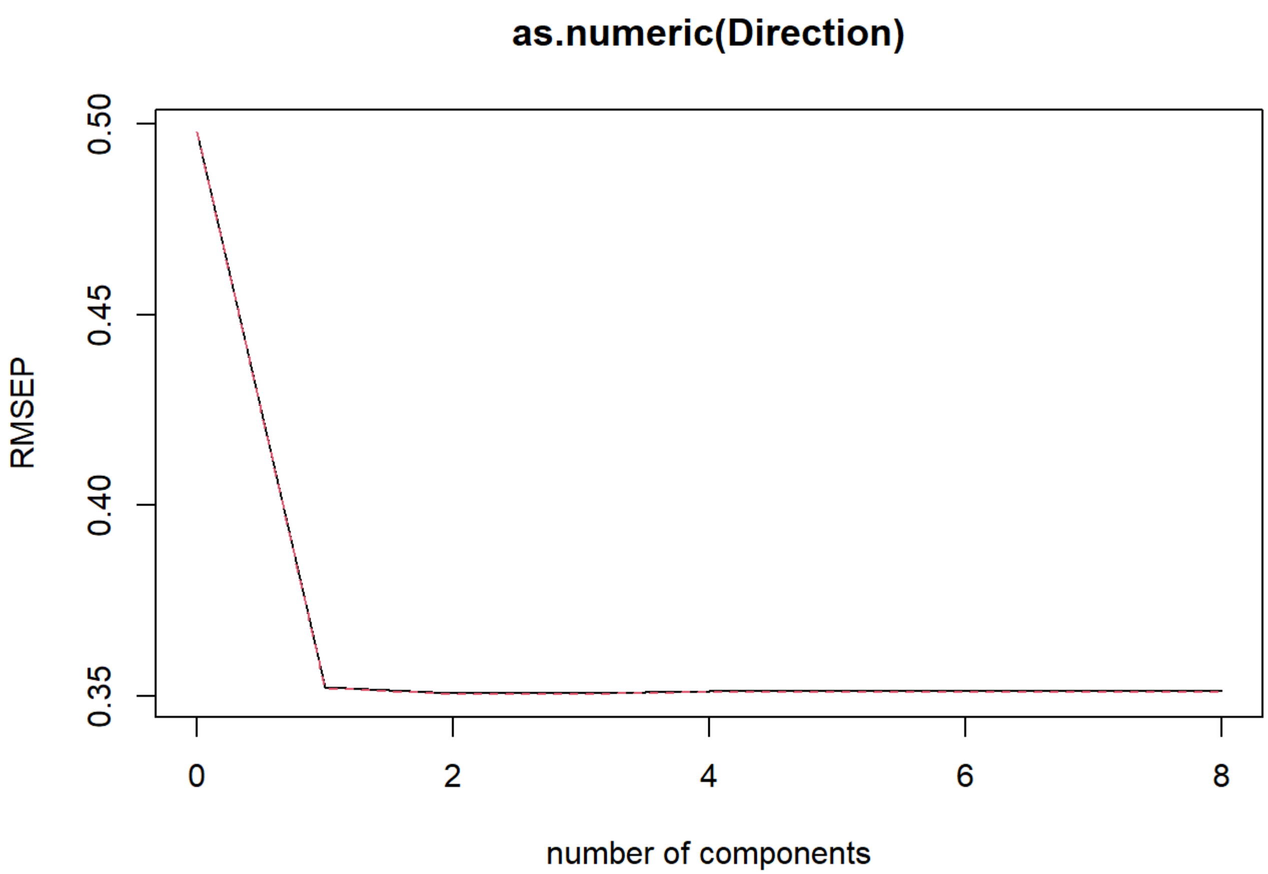
Answer f

```
# Implementing PLS using plsr() function

set.seed(1)
fit.pls <- plsr(as.numeric(Direction) ~ ., data = Weekly, subset = train.data, scale=TRUE, validation="CV")
summary(fit.pls)
```

```
## Data:    X dimension: 985 8
##  Y dimension: 985 1
## Fit method: kernelpls
## Number of components considered: 8
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV          0.4978   0.3524   0.3507   0.3509   0.3514   0.3513   0.3513
## adjCV       0.4978   0.3522   0.3505   0.3506   0.3510   0.3510   0.3510
##        7 comps  8 comps
## CV      0.3513   0.3513
## adjCV   0.3510   0.3510
##
## TRAINING: % variance explained
##                        1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## X                        13.68    32.67    50.47    62.33    66.89    77.18
## as.numeric(Direction)    50.31    51.76    51.87    51.89    51.90    51.90
##                        7 comps  8 comps
## X                        88.64    100.0
## as.numeric(Direction)    51.90     51.9
```

```
validationplot(fit.pls ,val.type="RMSEP")
```




```
pred.e.pls <- predict(fit.pls, Weekly.2009.2010, ncomp=2)
table(pred.e.pls, Direction.2009.2010)
```

```
##                    Direction.2009.2010
## pred.e.pls          Down Up
##   0.416864761405766    1  0
##   0.568827594842175    1  0
##   0.582271494839188    1  0
##   0.739425503630556    1  0
##   0.821620099806649    1  0
##   0.831697873207287    1  0
##   0.839805488936676    1  0
##   0.840489933167065    1  0
##   0.887289870391909    1  0
##   0.907048147711969    1  0
##   0.930789176064328    1  0
##   0.945331025137073    1  0
##   0.983787304279044    1  0
##   1.00864454452195     1  0
##   1.12318347594407     1  0
##   1.14920098387987     1  0
##   1.18067226652948     1  0
##   1.1863541979638      1  0
##   1.19655938436577     1  0
##   1.23277276087046     1  0
##   1.25800281085742     1  0
##   1.26067394745872     1  0
##   1.26816783050745     1  0
##   1.3067579725455      1  0
##   1.31983876173129     1  0
##   1.35216167640392     1  0
##   1.36829546833293     1  0
##   1.39057881671399     1  0
##   1.41547156327689     1  0
##   1.41686643434701     1  0
##   1.42279675023241     1  0
##   1.42448310380898     1  0
##   1.43409433932698     1  0
##   1.44131727135222     1  0
##   1.45474035742736     1  0
##   1.45726489400392     1  0
##   1.48661542814847     1  0
```

```
##   1.494985923391       1  0
##   1.50550722147485     1  0
##   1.51065189513061     1  0
##   1.51209376287901     1  0
##   1.51250935956369     0  1
##   1.51400734316725     1  0
##   1.5440980027812      0  1
##   1.54964442417551     0  1
##   1.55322120103711     0  1
##   1.57603053362911     0  1
##   1.59784838913319     1  0
##   1.60056921924452     0  1
##   1.60918073276943     0  1
##   1.61109474454666     0  1
##   1.62021461031825     0  1
##   1.62212442905777     0  1
##   1.63399155924961     0  1
##   1.65530256822212     0  1
##   1.66324114548606     0  1
##   1.66869175355046     0  1
##   1.68084536990454     0  1
##   1.68409870375885     0  1
##   1.69555854289287     0  1
##   1.69932369701547     0  1
##   1.70817415850165     0  1
##   1.71179460330613     0  1
##   1.71454943060704     0  1
##   1.72269039364966     0  1
##   1.73008506898438     0  1
##   1.73559084886382     0  1
##   1.73887200963136     0  1
##   1.77151490885743     0  1
##   1.79308529877918     0  1
##   1.79987961114314     0  1
##   1.81536913403994     0  1
##   1.82093336190006     0  1
##   1.82940463501608     0  1
##   1.83119892081229     0  1
##   1.8336647256317      0  1
```

```
##   1.85051947708256     0  1
##   1.8896139081596      0  1
##   1.89003772653626     0  1
##   1.90715430847037     0  1
##   1.91170908820663     0  1
##   1.92005972537925     0  1
##   1.9371947496612      0  1
##   1.96336243568322     0  1
##   1.96772198088562     0  1
##   1.98315072394469     0  1
##   1.98575608964938     0  1
##   1.9892487916503      0  1
##   2.06433026573221     0  1
##   2.06510948331842     0  1
##   2.07344131915426     0  1
##   2.11113356938556     0  1
##   2.11495067723367     0  1
##   2.11729273554395     0  1
##   2.12953756539186     0  1
##   2.16641249040232     0  1
##   2.17121276887075     0  1
##   2.26163807798678     0  1
##   2.32647499717323     0  1
##   2.42191350282431     0  1
##   2.44082687820175     0  1
##   2.63133869858083     0  1
##   2.65594045639992     0  1
##   3.22925164028569     0  1
```

```
mean((pred.e.pls - as.numeric(Weekly.2009.2010$Direction))^2)
```

```
## [1] 0.114054
```

The test error rate for PLS for ncomp = 2 is 11%

Question g

g. Repeat (d) using Nearest Shrunken Centroids.

Answer g

```
# Implementing Nearest Shrunken Centroids using pamr function

library(pamr)

ctrl <- trainControl(summaryFunction = twoClassSummary, classProbs = TRUE, savePredictions = TRUE)

nscGrid <- data.frame(.threshold = 0:25)
nscTune <- train(x = as.matrix(Weekly[,1:8]),
 y = Weekly$Direction,
method = "pam",
 preProc = c("center", "scale"),
 tuneGrid = nscGrid,
 metric = "ROC",
trControl = ctrl)
```

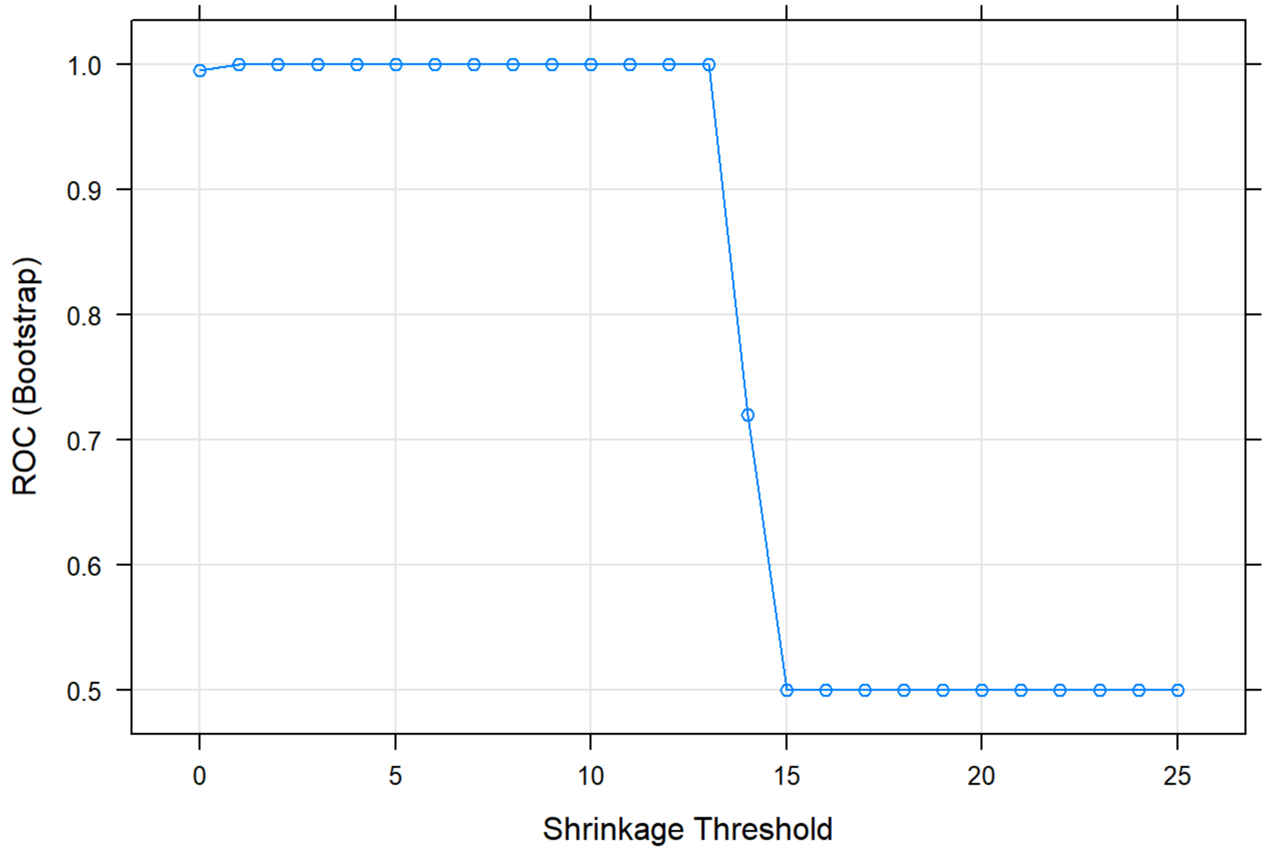
```
## 11111111111111111111111111
```

```
nscTune
```

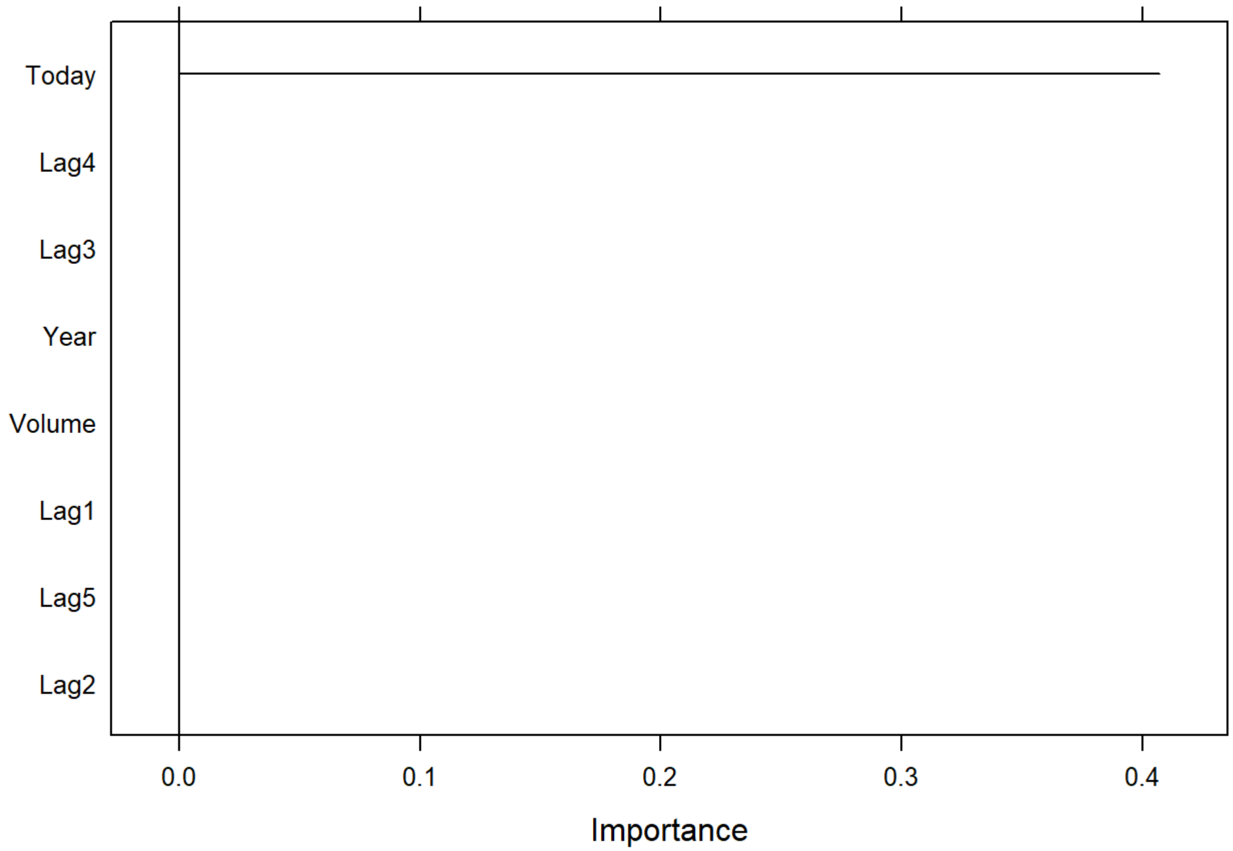
```
## Nearest Shrunken Centroids
##
## 1089 samples
##    8 predictor
##    2 classes: 'Down', 'Up'
##
## Pre-processing: centered (8), scaled (8)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 1089, 1089, 1089, 1089, 1089, 1089, ...
## Resampling results across tuning parameters:
##
##   threshold  ROC        Sens          Spec
##    0         0.9948327  0.6201356750  0.9998298
##    1         0.9997970  0.5898867640  1.0000000
##    2         1.0000000  0.5630311618  1.0000000
##    3         1.0000000  0.5357010979  1.0000000
##    4         1.0000000  0.5037749125  1.0000000
##    5         1.0000000  0.4651664733  1.0000000
##    6         1.0000000  0.4169004341  1.0000000
##    7         1.0000000  0.3595070091  1.0000000
##    8         1.0000000  0.3025287315  1.0000000
##    9         1.0000000  0.2366655196  1.0000000
##   10         1.0000000  0.1655621794  1.0000000
##   11         1.0000000  0.0943030044  1.0000000
##   12         1.0000000  0.0431982273  1.0000000
##   13         1.0000000  0.0085297598  1.0000000
##   14         0.7200000  0.0006864315  1.0000000
##   15         0.5000000  0.0000000000  1.0000000
##   16         0.5000000  0.0000000000  1.0000000
##   17         0.5000000  0.0000000000  1.0000000
##   18         0.5000000  0.0000000000  1.0000000
##   19         0.5000000  0.0000000000  1.0000000
##   20         0.5000000  0.0000000000  1.0000000
##   21         0.5000000  0.0000000000  1.0000000
##   22         0.5000000  0.0000000000  1.0000000
##   23         0.5000000  0.0000000000  1.0000000
##   24         0.5000000  0.0000000000  1.0000000
##   25         0.5000000  0.0000000000  1.0000000
##
```

```
## ROC was used to select the optimal model using the largest value.
## The final value used for the model was threshold = 2.
```

```
plot(nscTune)
```




```
#variale importance
plot(varImp(nscTune, scale =FALSE))
```




```
#Prediction and Error rates

pred.pamr <- predict(nscTune, Weekly.2009.2010)
table(pred.pamr, Direction.2009.2010)
```

```
##           Direction.2009.2010
## pred.pamr Down Up
##      Down   24  0
##      Up     19 61
```

Based on the results of the table above, We may conclude that the percentage of correct predictions (Down * Down & Up *Up) on the training data is $(24 + 61)/104$ which is equal to 81.73%. So, we can say that 18.27% is the training error rate.

Question h

h. Which of these methods appears to provide the best results on this data?

Answer h

Based on the models that we have run and looking at the test error rate data, we can say that **Logistic Regression** is the best model.