Assignment3

Yue Zhang

2025-09-27

```
setwd("/Users/yuezhang/Documents/Biostat/PH1976")
getwd()
library(ISLR2)
library(ggplot2)
library(tidyr)
library(dtplyr)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.1.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(ggcorrplot)
library(MASS)
```

```
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
##
## The following object is masked from 'package:ISLR2':
##
##     Boston
```

```
library(class)
library(e1071)
```

```
##
## Attaching package: 'e1071'
##
## The following object is masked from 'package:ggplot2':
##
##     element
```

5.

(a). For the training set, QDA will perform better as it is a more flexible model. However, for the test set, LDA will perform better. Because the Bayes boundary is linear, LDA has lower variance, QDA might encounter overfitting.

(b). For the training set, QDA will perform better as it has smaller training error. If the test set has large n, QDA will perform better as its lower bias will outweights the higher variance. If the test set has small n, LDA might be better due to QDA's overfitting.

(c). As n increases, we would expect the prediction accuracy of QDA relative to LDA to improve as there is more data to fit to subtle effects in the data as QDA's variance shrinks faster than its bias.

(d). False. Even though QDA is a more flexible mode, the flexibility lowers training error, not test error. For a linear Bayes boundary, LDA has lower variance and the correct pattern. QDA might lead to overfitting.

8.

We would prefer the logistic regression for its lower test error. The 1-nearest neighbors classification error on the training set is 0% as each point is its own nearest neighbor. Therefore, the test error should be 18% x 2 = 36% which is larger than 30%.

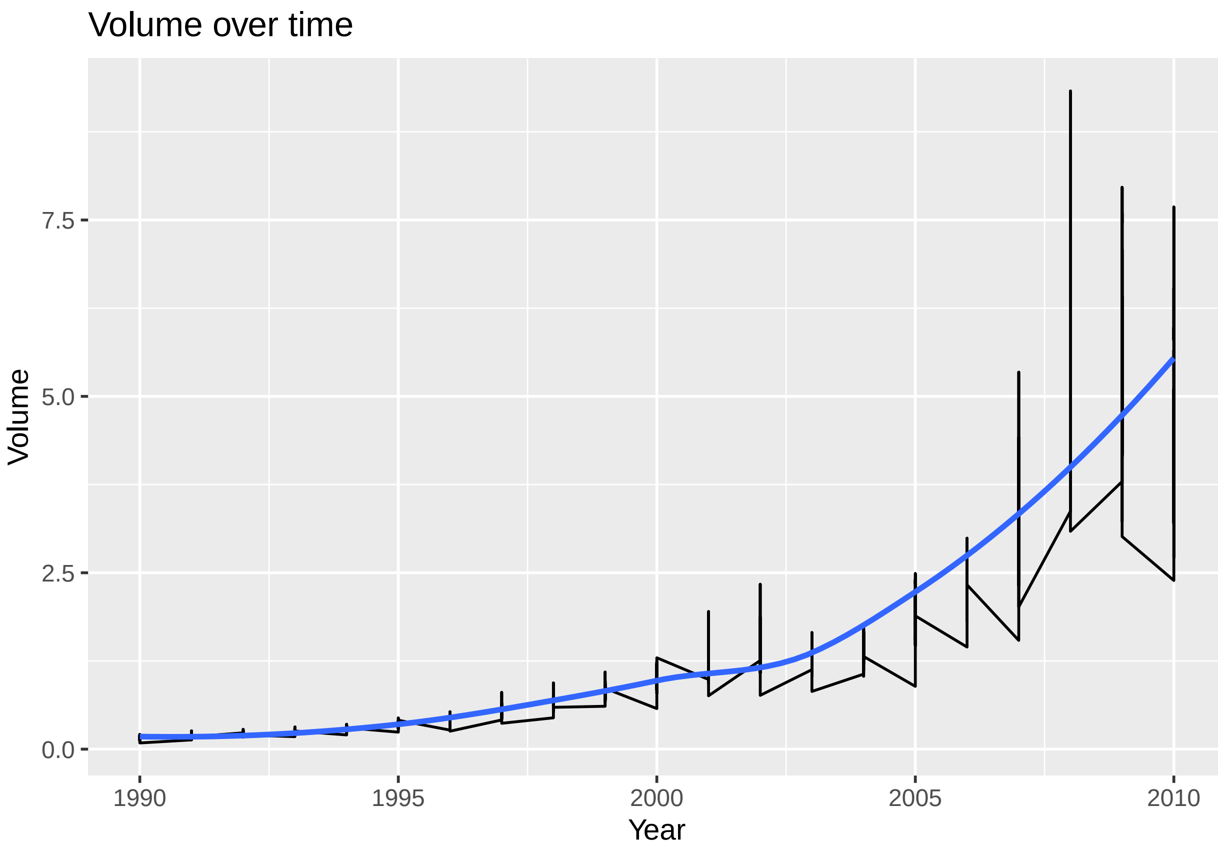
13.

(a).

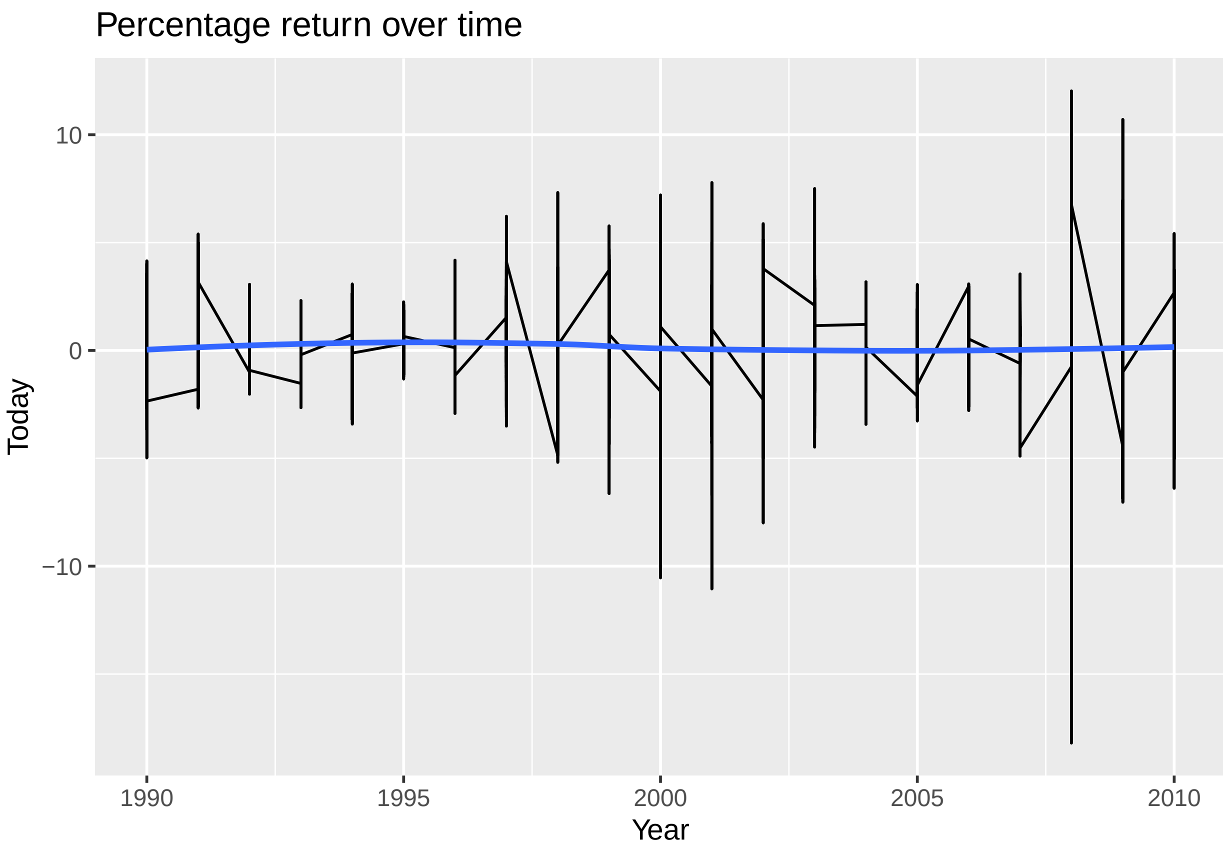
```r
attach(Weekly)
summary(Weekly)
```

```
##       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
##
##
##
##
```

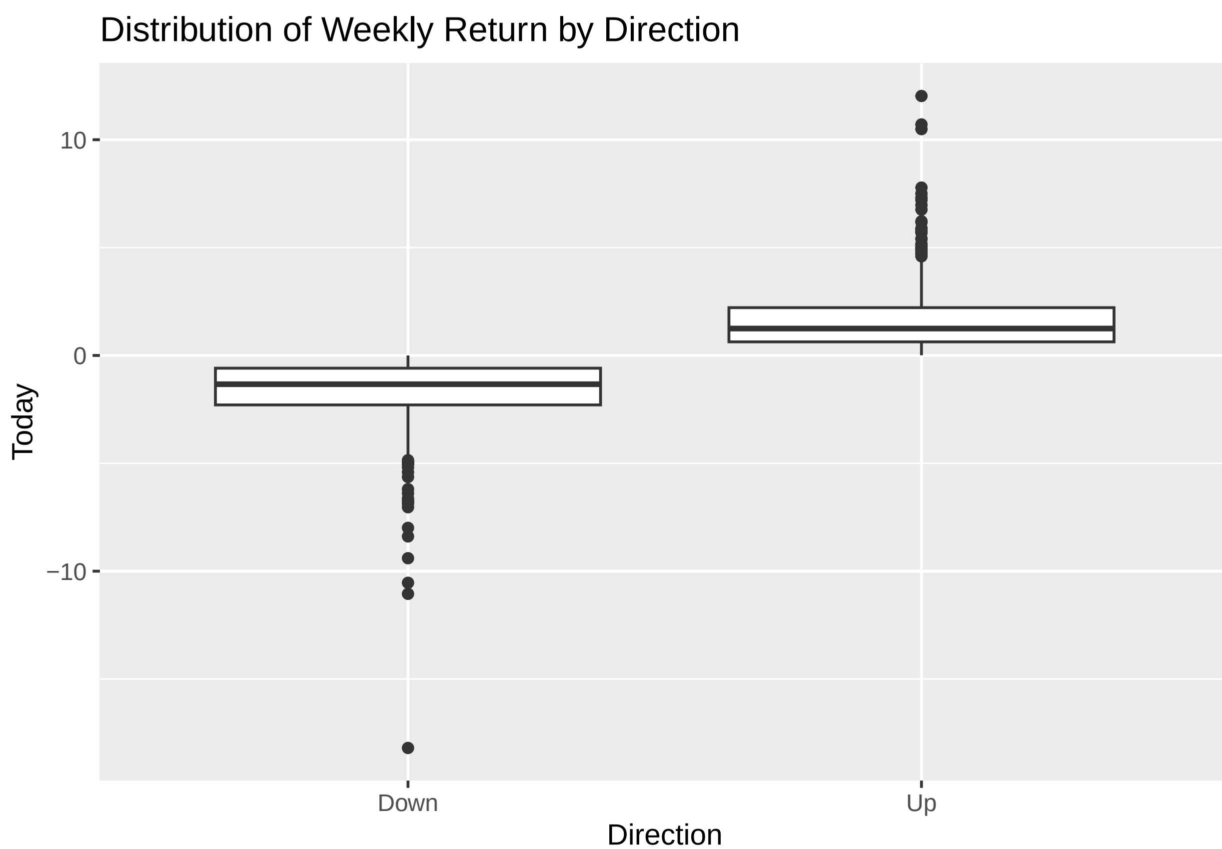
```r
#Plot
ggplot(Weekly, aes(x = Year, y = Volume)) +
  geom_line() +
  geom_smooth(se = FALSE, method = "loess") +
  labs(title = "Volume over time")
```



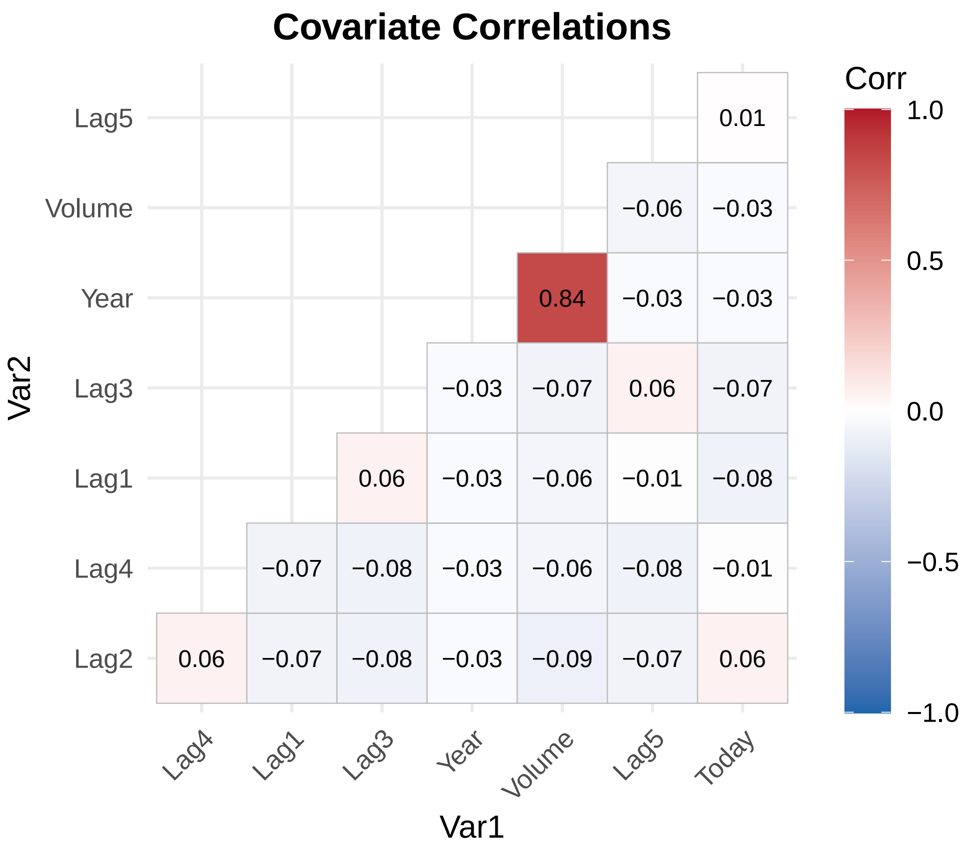

```r
ggplot(Weekly, aes(x = Year, y = Today)) +
  geom_line() +
  geom_smooth(se = FALSE, method = "loess") +
  labs(title = "Percentage return over time")
```




```r
ggplot(Weekly, aes(x = Direction, y = Today)) +
  geom_boxplot() +
  labs(title = "Distribution of Weekly Return by Direction")
```




```
cont_vars = Weekly[, -9]
cor_mat = cor(cont_vars, use = "pairwise.complete.obs")
ggcorrplot(
  cor_mat,
  method = "square",
  type = "lower",
  hc.order = TRUE,
  lab = TRUE,
  lab_size = 3,
  colors = c("#2166ac", "white", "#b2182b"),
  hc.method = "complete",
  tl.srt = 45,
  show.diag = NULL
) +
  ggplot2::labs(title = "Covariate Correlations") +
  ggplot2::theme_minimal(base_size = 12) +
  ggplot2::theme(
    plot.title = element_text(face = "bold", hjust = 0.5),
    axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1)
  ) +
  guides(fill = guide_colorbar(barheight = unit(8, "cm")))
```



The correlation matrix shows that volumn and year is highly positive correlated, other variables have relatively low correlation. Percentage return for weeks fluctuate around zero, the trend is quite stable except for 2008 which suffered from financial crisis.When direction is up, the median return is positive, when direction is down, the median return is negative. There are some outliers in both directions. The volume over time shows a upward trend which also reflects the correlation matrix.

(b).

```
weekly_logistic = glm(
  data = ISLR2::Weekly,
  Direction ~ Volume + Lag1 + Lag2 + Lag3 + Lag4 + Lag5,
  family = binomial
)
summary(weekly_logistic)
```

```
##
## Call:
## glm(formula = Direction ~ Volume + Lag1 + Lag2 + Lag3 + Lag4 +
##     Lag5, family = binomial, data = ISLR2::Weekly)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
## Volume      -0.02274    0.03690  -0.616   0.5377
## 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
```

```
## ---
## 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
```

Based on the summary, only Lag2 appears to be statistically significant with a p-value less than 0.05.

(c).

```r
contrasts(Weekly$Direction)
```

```
##      Up
## Down  0
## Up    1
```

```r
prob = predict(weekly_logistic, type = "response")
pred_class = factor(ifelse(prob > 0.5, "Up", "Down"),
                     levels = levels(Weekly$Direction))

confusion_matrix = table(pred_class, Weekly$Direction)
print(confusion_matrix)
```

```
##
## pred_class Down  Up
##       Down   54  48
##       Up    430 557
```

```r
accuracy_logistic = mean(pred_class == Weekly$Direction)
cat("Accuracy:", round(accuracy_logistic, 4), "\n")
```

```
## Accuracy: 0.5611
```

```r
sensitivity = 557/(557 + 48)
specificity = 54/(54 + 430)
cat("Sensitivity:", round(sensitivity, 4))
```

```
## Sensitivity: 0.9207
```

```r
cat("Specificity:", round(specificity, 4))
```

```
## Specificity: 0.1116
```

The overall fraction of correct predictions is 56.11%, the sensitivity is 92.07%, and the specificity is 11.16%. This demonstrates that this regression can predict the direction of up well but mis-predicts the direction of down.

(d).

```r
training = Weekly %>% filter(Year <= 2008)
test = Weekly %>% filter(Year > 2008)

weekly_logistic2 = glm(data = training, Direction ~ Lag2, family = "binomial")
weekly_predict = predict(newdata = test, weekly_logistic2, type = "response")
```

```r
pred2 = ifelse(weekly_predict > 0.5, "Up", "Down")

confusion_matrix2 = table(pred2, test$Direction)
print(confusion_matrix2)
```

```
##
## pred2  Down Up
##   Down    9  5
##   Up     34 56
```

```r
accuracy_logistic2 = mean(pred2 == test$Direction)
cat("Accuracy:", round(accuracy_logistic2, 4), "\n")
```

```
## Accuracy: 0.625
```

```r
sensitivity2 = 56/(56 + 5)
specificity2 = 9/(9 + 34)
cat("Sensitivity:", round(sensitivity2, 4))
```

```
## Sensitivity: 0.918
```

```r
cat("Specificity:", round(specificity2, 4))
```

```
## Specificity: 0.2093
```

The overall fraction of correct predictions is 62.5%, the sensitivity is 91.8%, and the specificity is 20.93%. This demonstrates that this regression can predict the direction of up well but mis-predicts the direction of down. The accuracy is better than the previous model, but still it has many false positives.

(e).

```r
weekly_lda = lda(data = training, Direction ~ Lag2)
weekly_predict_lda = predict(newdata = test, weekly_lda, type = "response")$class

confusion_matrix_lda = table(weekly_predict_lda, test$Direction)
print(confusion_matrix_lda)
```

```
##
## weekly_predict_lda Down Up
##               Down    9  5
##               Up     34 56
```

```r
accuracy_lda = mean(weekly_predict_lda == test$Direction)
cat("Accuracy:", round(accuracy_lda, 4), "\n")
```

```
## Accuracy: 0.625
```

The overall fraction of correct predictions is 62.5%, the sensitivity is 91.8%, and the specificity is 20.93%.

(f).

```r
weekly_qda = qda(data = training, Direction ~ Lag2)
weekly_predict_qda = predict(newdata = test, weekly_qda, type = "response")$class

confusion_matrix_qda = table(weekly_predict_qda, test$Direction)
print(confusion_matrix_qda)
```

```
##
## weekly_predict_qda Down Up
##               Down    0  0
```

```
##                Up     43 61
```

```r
accuracy_qda = mean(weekly_predict_qda == test$Direction)
cat("Accuracy:", round(accuracy_qda, 4), "\n")
```

```
## Accuracy: 0.5865
```

The overall fraction of correct predictions is 58.65%.

(g).

```r
xtrain = scale(training[, "Lag2", drop = FALSE])
xtest = scale(test[, "Lag2", drop = FALSE],
              center = attr(xtrain, "scaled:center"),
              scale = attr(xtrain, "scaled:scale"))
ytrain = training$Direction
ytest = test$Direction

set.seed(123)
weekly_knn = knn(train = xtrain, test = xtest, cl = ytrain, k = 1)

confusion_matrix_knn = table(weekly_knn, ytest)
print(confusion_matrix_knn)
```

```
##           ytest
## weekly_knn Down Up
##       Down   21 29
##       Up     22 32
```

```r
accuracy_knn = mean(weekly_knn == ytest)
cat("Accuracy:", round(accuracy_knn, 4), "\n")
```

```
## Accuracy: 0.5096
```

The overall fraction of correct predictions is 50.96%.

(h).

```r
weekly_nb = naiveBayes(data = training, Direction ~ Lag2)
weekly_predict_nb = predict(newdata = test, weekly_nb, type = "class")

confusion_matrix_nb = table(weekly_predict_nb, test$Direction)
print(confusion_matrix_nb)
```

```
##
## weekly_predict_nb Down Up
##              Down    0  0
##              Up     43 61
```

```r
accuracy_nb = mean(weekly_predict_nb == test$Direction)
cat("Accuracy:", round(accuracy_qda, 4), "\n")
```

```
## Accuracy: 0.5865
```

The overall fraction of correct predictions is 58.65%.

(i).

LDA and logistic regression perform the best as they have higher accuracy.

(j).

```r
#Logistic regression
fit_logit = glm(data = training,Direction ~ Lag1 + Lag2, family = binomial)
pred_logit = predict(fit_logit, newdata = test, type = "response")
class_logit = ifelse(pred_logit > 0.5, "Up", "Down")
table(class_logit, test$Direction)
```

```
##
## class_logit Down Up
##        Down    7  8
##        Up     36 53
```

```r
mean(class_logit == test$Direction)
```

```
## [1] 0.5769231
```

```r
fit_logit2 = glm(data = training, Direction ~ Lag2 + poly(Lag2, 2), family = binomial)
pred_logit2 = predict(fit_logit2, newdata = test, type = "response")
class_logit2 = ifelse(pred_logit2 > 0.5, "Up", "Down")
table(class_logit2, test$Direction)
```

```
##
## class_logit2 Down Up
##         Down    8  4
##         Up     35 57
```

```r
mean(class_logit2 == test$Direction)
```

```
## [1] 0.625
```

```r
#LDA and QDA
fit_lda = lda(data = training, Direction ~ Lag1 + Lag2)
pred_lda = predict(fit_lda, newdata = test)$class
table(pred_lda, test$Direction)
```

```
##
## pred_lda Down Up
##     Down    7  8
##     Up     36 53
```

```r
mean(pred_lda == test$Direction)
```

```
## [1] 0.5769231
```

```r
fit_qda = qda(data = training, Direction ~ poly(Lag2, 2))
pred_qda = predict(fit_qda, newdata = test)$class
table(pred_qda, test$Direction)
```

```
##
## pred_qda Down Up
##     Down    7  3
##     Up     36 58
```

```r
mean(pred_qda == test$Direction)
```

```
## [1] 0.625
```

```r
#Naive Bayes
fit_nb = naiveBayes(data = training, Direction ~ Lag1 + Lag2)
pred_nb = predict(fit_nb, newdata = test)
```

```r
table(pred_nb, test$Direction)
```

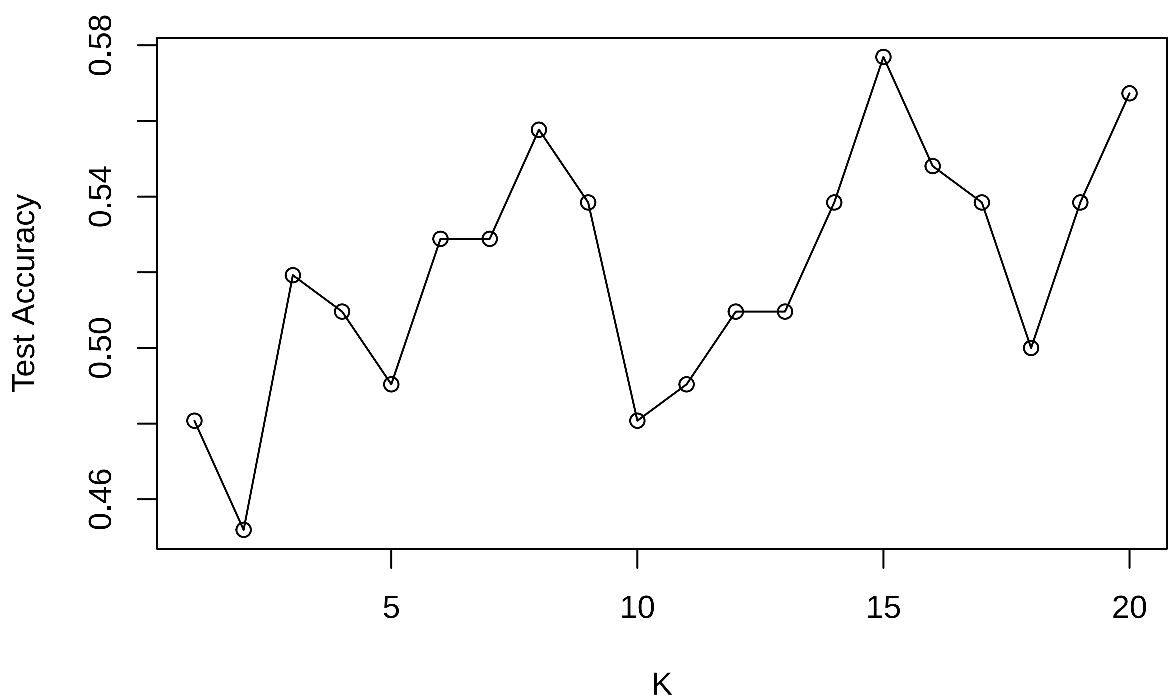
```
##
## pred_nb Down Up
##    Down    3  8
##    Up     40 53
```

```r
mean(pred_nb == test$Direction)
```

```
## [1] 0.5384615
```

```r
#KNN
xtrain2 = scale(training[, c("Lag1","Lag2")])
xtest2 = scale(test[, c("Lag1","Lag2")],
               center = attr(xtrain2, "scaled:center"),
               scale  = attr(xtrain2, "scaled:scale"))
ytrain2 = training$Direction
ytest2 = test$Direction

set.seed(123)
accs = sapply(1:20, function(k) {
  predk = knn(train = xtrain2, test = xtest2, cl = ytrain2, k = k)
  mean(predk == ytest2)
})
plot(1:20, accs, type="o", xlab="K", ylab="Test Accuracy")
```




```r
best_k = which.max(accs)
best_k
```

```
## [1] 15
```

```r
set.seed(123)
knn_best = knn(xtrain2, xtest2, ytrain2, k = best_k)
table(knn_best, ytest2)
```

```
##          ytest2
## knn_best Down Up
##     Down   21 22
##     Up     22 39
```

```r
mean(knn_best == ytest2)
```

```
## [1] 0.5769231
```

Based on all the tests, QAD and logistic regression that includes Lag2 and Lag2 squared perform best as their accuracy is 62.5%.