Assignment 2:

Logistic Regression, LDA, QDA, KNN

Loading all packages that will be used to develop models

```
library(quantmod)
library(xts)
library(dplyr)
library(MASS)
library(class)
library(ggplot2)
```

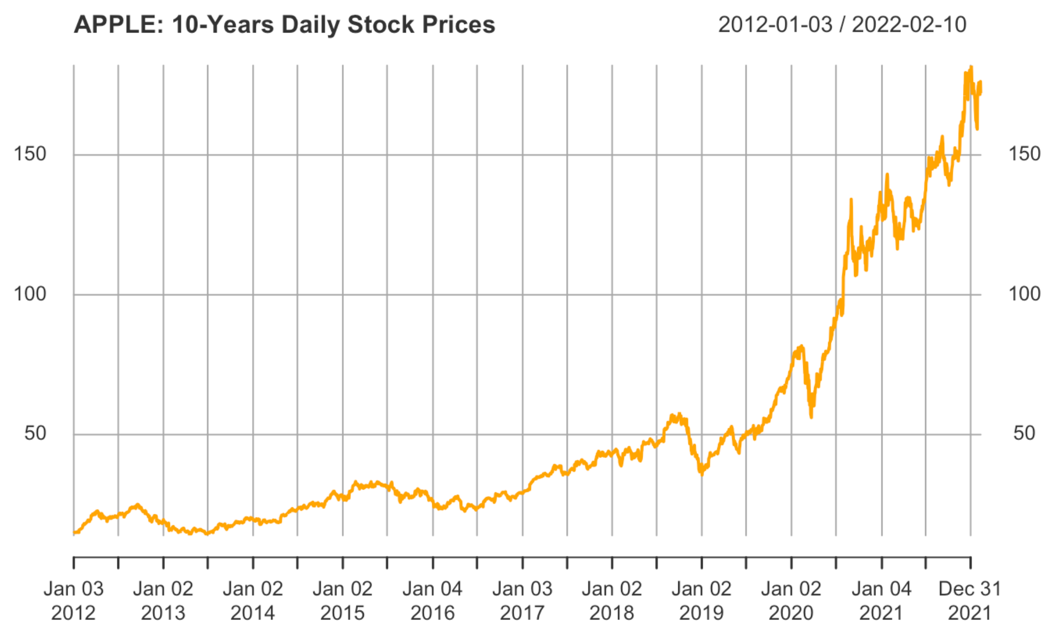
Loading data

We used R package's quantmod to download the recent ten years' daily stock prices from January 1, 2012, to February 11, 2022, for Apple Inc. from Yahoo Finance.

```
# Downloading Apple stock price using quantmod
AAPL <- getSymbols("AAPL",src = 'yahoo', from = '2012-01-01',to = "2022-02-11",warnings = FALSE, auto.assign = FALSE)
head(AAPL)

##            AAPL.Open AAPL.High AAPL.Low AAPL.Close AAPL.Volume
AAPL.Adjusted
## 2012-01-03  14.62143  14.73214 14.60714   14.68679   302220800
12.57592
## 2012-01-04  14.64286  14.81000 14.61714   14.76571   260022000
12.64350
## 2012-01-05  14.81964  14.94821 14.73821   14.92964   271269600
12.78387
## 2012-01-06  14.99179  15.09821 14.97214   15.08571   318292800
12.91751
## 2012-01-09  15.19643  15.27679 15.04821   15.06179   394024400
12.89702
## 2012-01-10  15.21107  15.21429 15.05357   15.11571   258196400
12.94320
```

The recent 10 years Daily price of Apple

```
plot(AAPL$AAPL.Close, main = "APPLE: 10-Years Daily Stock Prices", type =
"l", lwd = 2, col = 'orange')
```



Changing daily prices to daily log returns

```
AAPL$Log.return <- diff(log(AAPL$AAPL.Close))
head(AAPL[-1])
```

```
##            AAPL.Open AAPL.High AAPL.Low AAPL.Close AAPL.Volume AAPL.Adjust
ed
## 2012-01-04  14.64286  14.81000 14.61714   14.76571   260022000      12.643
50
## 2012-01-05  14.81964  14.94821 14.73821   14.92964   271269600      12.783
87
## 2012-01-06  14.99179  15.09821 14.97214   15.08571   318292800      12.917
51
## 2012-01-09  15.19643  15.27679 15.04821   15.06179   394024400      12.897
02
## 2012-01-10  15.21107  15.21429 15.05357   15.11571   258196400      12.943
19
## 2012-01-11  15.09571  15.10179 14.97536   15.09107   215084800      12.922
10
##              Log.return
## 2012-01-04  0.005359694
## 2012-01-05  0.011040828
## 2012-01-06  0.010399504
## 2012-01-09 -0.001587396
## 2012-01-10  0.003574057
## 2012-01-11 -0.001631621
```

Formulating data frame with Direction and Lag 1

```
# Formulate Lag1
AAPL$Lag1 <- Lag(AAPL$Log.return, k=1)
#Formulate Direction
AAPL<- AAPL[-c(1,2),]
AAPL$Direction <- AAPL$Log.return
AAPL$Direction <- ifelse(AAPL$Direction < 0, "Down", "Up")
head(AAPL)
```

```
##            AAPL.Open   AAPL.High   AAPL.Low    AAPL.Close  AAPL.Volume
## 2012-01-05 "14.819643" "14.948214" "14.738214" "14.929643" "271269600"
## 2012-01-06 "14.991786" "15.098214" "14.972143" "15.085714" "318292800"
## 2012-01-09 "15.196429" "15.276786" "15.048214" "15.061786" "394024400"
## 2012-01-10 "15.211071" "15.214286" "15.053571" "15.115714" "258196400"
## 2012-01-11 "15.095714" "15.101786" "14.975357" "15.091071" "215084800"
## 2012-01-12 "15.081429" "15.103571" "14.955357" "15.049643" "212587200"
##            AAPL.Adjusted Log.return             Lag1
## 2012-01-05 "12.783867"   "0.0110408280486718"   "0.00535969367235145"
## 2012-01-06 "12.917508"   "0.0103995035996491"   "0.0110408280486718"
## 2012-01-09 "12.897018"   "-0.00158739563974031" "0.0103995035996491"
## 2012-01-10 "12.943195"   "0.00357405732123839"  "-0.00158739563974031"
## 2012-01-11 "12.922095"   "-0.00163162054267119" "0.00357405732123839"
## 2012-01-12 "12.886618"   "-0.00274897443297428" "-0.00163162054267119"
##            Direction
## 2012-01-05 "Up"
## 2012-01-06 "Up"
## 2012-01-09 "Down"
## 2012-01-10 "Up"
## 2012-01-11 "Down"
## 2012-01-12 "Down"
```

Finalize data frame used to build models

```
AAPL <- as.xts(AAPL, dateFormat = "Date")
AAPL <- fortify.zoo(AAPL)
AAPL$Log.return <- as.double(AAPL$Log.return)
AAPL$Year <- as.numeric(format(AAPL$Index,'%Y'))
AAPL$Lag1 <- as.double(AAPL$Lag1)
AAPL$Direction <- as.factor(AAPL$Direction)
AAPL <- AAPL %>% dplyr::select(Index,Log.return,Direction,Lag1,Year)
head(AAPL)
```

```
##        Index   Log.return Direction         Lag1 Year
## 1 2012-01-05  0.011040828        Up  0.005359694 2012
## 2 2012-01-06  0.010399504        Up  0.011040828 2012
## 3 2012-01-09 -0.001587396      Down  0.010399504 2012
## 4 2012-01-10  0.003574057        Up -0.001587396 2012
```

```
## 5 2012-01-11 -0.001631621      Down  0.003574057 2012
## 6 2012-01-12 -0.002748974      Down -0.001631621 2012
```

The data frame with 2543 observations on the following 5 variables.

Index: Dates
Log.return: Daily log returns
Direction: A factor with levels Down and Up indicating whether the company had a positive or negative return on a given day
Lag1: Yesterday's log return
Year: The year that the observation was recorded

Splitting Train-Test data

```
# Create train data set with the first 7 years from January 2012 to December
2018
train <- AAPL %>% filter(Year < 2019)

# Create test data set with the last 3 years from January 2019 to February 20
22
test <- AAPL %>% filter(Year >= 2019)

# Extract Direction from test set to evaluate the model
Direction.3yrs <- test$Direction

sprintf("%s are %i observations and %i variables", "Training set dimensions",
dim(train)[1],dim(train)[2])

## [1] "Training set dimensions are 1758 observations and 5 variables"

sprintf("%s are %i observations and %i variables", "Test set dimensions", dim
(test)[1],dim(test)[2])

## [1] "Test set dimensions are 785 observations and 5 variables"
```

Logistic Regression

Model Fitting

```
glm.fit = glm(Direction ~ Lag1 , data = train, family = binomial)
summary(glm.fit)

##
## Call:
## glm(formula = Direction ~ Lag1, family = binomial, data = train)
##
## Deviance Residuals:
```

```
##    Min      1Q  Median      3Q     Max
## -1.378  -1.207   1.112   1.148   1.246
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.07224    0.04778   1.512    0.130
## Lag1        -2.94174    2.97618  -0.988    0.323
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2434.9  on 1757  degrees of freedom
## Residual deviance: 2433.9  on 1756  degrees of freedom
## AIC: 2437.9
##
## Number of Fisher Scoring iterations: 3
```

Logistic regression model was implemented to classify the direction of stock price of Apple company. We have trained and tested our model on two separate data sets: training was performed using the first seven years' data, and testing was performed using the last three years' data.

Model Prediction

```
glm.probs = predict(glm.fit, newdata = test, type = "response")
glm.probs[1:5]

##         1         2         3         4         5
## 0.5109849 0.5172152 0.5940909 0.4873198 0.5196893
```

Model Evaluation

```
glm.pred = rep("Down", nrow(test))
glm.pred[glm.probs > 0.5] = "Up"
table(glm.pred, Direction.3yrs)

##         Direction.3yrs
## glm.pred Down  Up
##     Down   44  32
##     Up    312 397

glm.accuracy <- mean(glm.pred == Direction.3yrs)
accuracy.inc <- 397/(397+312)
sprintf("%s is %0.3f", "The accuracy", glm.accuracy)

## [1] "The accuracy is 0.562"

sprintf("%s is %0.3f", "The accuracy rate of increasing", accuracy.inc)

## [1] "The accuracy rate of increasing is 0.560"
```

The result appears 56.2% of the daily movements have been correctly predicted. The confusion matrix shows that on days when logistic regression predicts an increase in the company, it has a 56% accuracy rate.

Linear Discriminant Analysis (LDA)

Model Fitting

```
lda.fit = lda(Direction ~ Lag1 , data = train)
lda.fit

## Call:
## lda(Direction ~ Lag1, data = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4823663 0.5176337
##
## Group means:
##              Lag1
## Down 0.0009493160
## Up   0.0001901832
##
## Coefficients of linear discriminants:
##            LD1
## Lag1 62.17604
```

The LDA output indicates that $\hat{\pi}_1 = 0.48$ and $\hat{\pi}_2 = 0.52$; in other words, 48% of the training observations correspond to days during which the return went down.

Model Prediction

```
lda.pred = predict(lda.fit, newdata = test)
lda.class <- lda.pred$class
lda.class[1:5]

## [1] Up   Up   Up   Down Up
## Levels: Down Up
```

Model Evaluation

```
table(lda.class, Direction.3yrs)

##          Direction.3yrs
## lda.class Down  Up
```

```
##       Down    44   32
##       Up     312  397

lda.accuracy<- mean(lda.class == Direction.3yrs)
accuracy.inc <- 397/(397+312)
sprintf("%s is %0.3f", "The accuracy", lda.accuracy)

## [1] "The accuracy is 0.562"

sprintf("%s is %0.3f", "The accuracy rate of increasing",accuracy.inc)

## [1] "The accuracy rate of increasing is 0.560"
```

As we observed, the LDA and logistic regression predictions are identical.

Quadratic Discriminant Analysis (QDA)

Model Fitting

```
qda.fit = qda(Direction ~ Lag1 , data = train)
qda.fit

## Call:
## qda(Direction ~ Lag1, data = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4823663 0.5176337
##
## Group means:
##              Lag1
## Down 0.0009493160
## Up   0.0001901832
```

The QDA output contains the group means, but it does not contain the coefficients of the linear discriminant, because the QDA classifier involves a quadratic, rather than a linear, function of the predictors.

Model Prediction

```
qda.pred = predict(qda.fit, newdata = test)
qda.class <- qda.pred$class
qda.class[1:10]

##  [1] Up   Up   Down Down Up   Down Up   Up   Up   Up
## Levels: Down Up
```

Model Evaluation

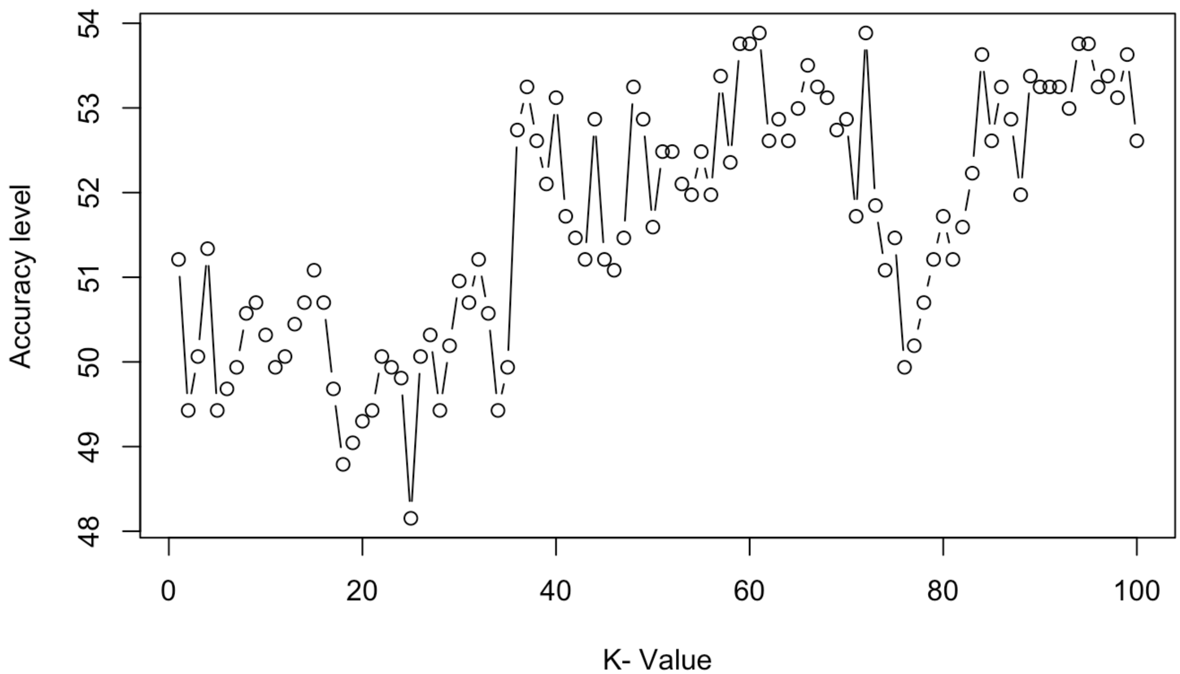
```
table(qda.class, Direction.3yrs)

##          Direction.3yrs
## qda.class Down  Up
##      Down   97 108
##      Up    259 321

qda.accuracy<- mean(qda.class == Direction.3yrs)
accuracy.inc <- 321/(321+259)
sprintf("%s is %0.3f", "The accuracy rate", qda.accuracy)

## [1] "The accuracy rate is 0.532"

sprintf("%s is %0.3f", "The accuracy rate of increasing",accuracy.inc)

## [1] "The accuracy rate of increasing is 0.553"
```

The result appears to be a little worse: 53.2% of the daily movements have been correctly predicted. However, the confusion matrix shows that on days when QDA predicts an increase in the company, it has a 55.3% accuracy rate.

K-Nearest Neighbors (KNN)

Model Fitting

```
# Select X from train set
train.X <- cbind(train$Lag1)

# Select X from test set
test.X <- cbind(test$Lag1)

# Select y from train set
train.Direction <- train$Direction

# Compute optimal value of k
i=1
k.optm=1
for (i in 1:100){
    knn.mod <-  knn(train.X,test.X,train.Direction, k = i)
    k.optm[i] <- 100 * sum(Direction.3yrs == knn.mod)/NROW(Direction.3yrs)
    k=i
    #cat(k,'=',k.optm[i],'\n')       # to print % accuracy
}

plot(k.optm, type="b", xlab="K- Value",ylab="Accuracy level")
```




```
sprintf("%s is %i", "The optimal k",which.max(k.optm))

## [1] "The optimal k is 61"
```

To identify optimal value of K, we selected K produces the highest accuracy rate which is 61. We may improve model performance by setting larger length for K; however, this can create overfitting model. Therefore, we should use cross-validation to test model while improving the model accuracy.

```
## Fitting KNN model
set.seed(1)
knn.pred <- knn(train.X,test.X,train.Direction, k = 61)
```

We fitted model using K = 61.

Model Evaluation

```
table(knn.pred, Direction.3yrs)

##          Direction.3yrs
## knn.pred Down  Up
##     Down  151 157
##     Up    205 272
```

```
knn.accuracy<- mean(knn.pred == Direction.3yrs)
accuracy.inc <- 272/(272+205)
sprintf("%s is %0.3f", "The accuracy rate", knn.accuracy)

## [1] "The accuracy rate is 0.539"

sprintf("%s is %0.3f", "The accuracy rate of increasing",accuracy.inc)

## [1] "The accuracy rate of increasing is 0.570"
```

The results using K = 61 appears that 53.9% of the observations are correctly predicted. The confusion matrix shows that on days when KNN predicts an increase in the company, it has a 57% accuracy rate.

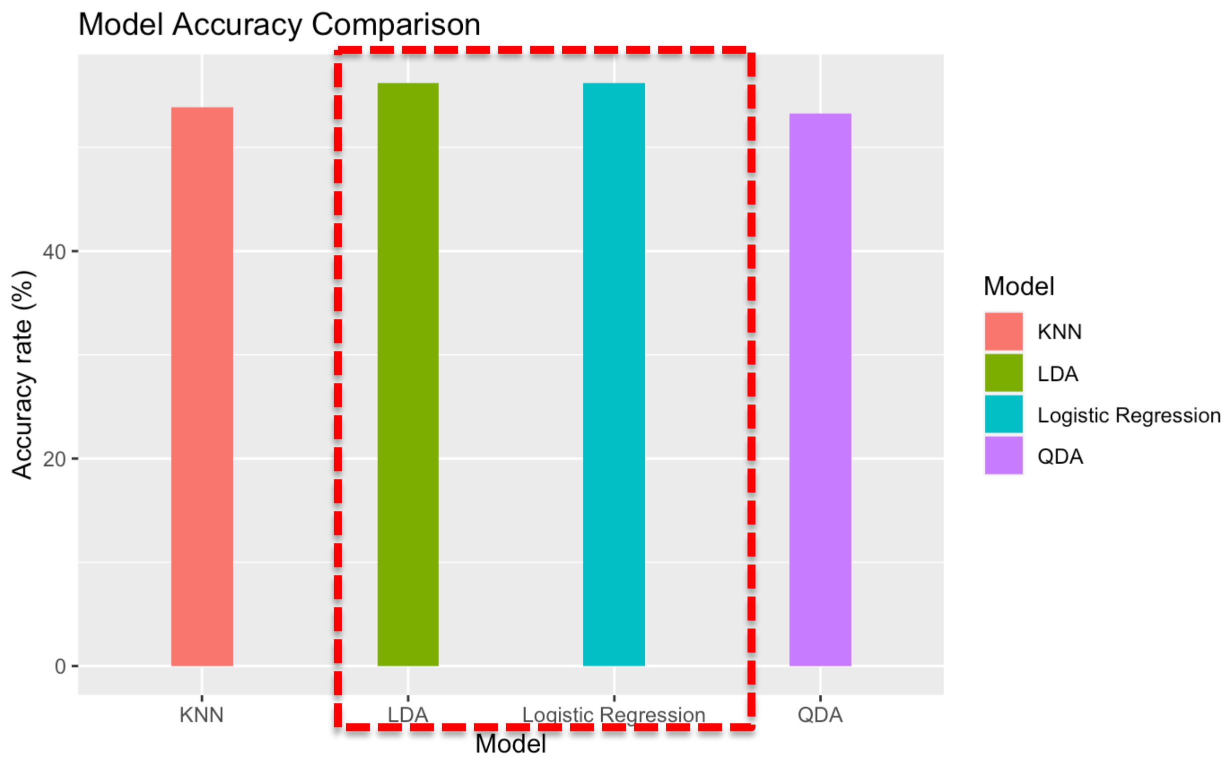
Models Comparison

```
accuracy <- data.frame(Model = (c("Logistic Regression","LDA","QDA","KNN")),A
ccuracy.Rate= (c(glm.accuracy,lda.accuracy,qda.accuracy,knn.accuracy)))

print(accuracy)

##                 Model Accuracy.Rate
## 1 Logistic Regression     0.5617834
## 2                 LDA     0.5617834
## 3                 QDA     0.5324841
## 4                 KNN     0.5388535
```

The results indicate as follows:

- The accuracy rates of Logistic regression and LDA are identical equals to 56.2%, which are the highest accuracy models.
- The accuracy of QDA model is 53%.
- The least accuracy model is KNN, 54%.

```
ggplot(accuracy, aes(x = Model, y = Accuracy.Rate*100, fill = Model)) +
  geom_col(width = 0.3) +
  labs(title = "Model Accuracy Comparison")+
  ylab("Accuracy rate (%)")
```



Conclusion

From the results, we observed that **logistic regression and LDA are the best models to classify an increase in the company and a decrease in the company since it appeared to be the model with the highest accuracy rate of 56%.** This suggests a possible trading strategy of buying on days when the model predicts an increasing return of the company and avoiding trades on days when a decrease is predicted. Also, the linear forms assumed by LDA and linear regression may capture the true relationship more accurately than the quadratic form assumed by QDA. For KNN model, the accuracy depends on the value of K we use to fit model; therefore, we need to find the optimal K that generates the higher accuracy to improve the model performance. However, the level of accuracy seems to be a little better than random guessing, because the stock price is quite hard to model accurately. In fact, the model may be required to add more features or use time-series analysis methods to access better prediction because only one feature may not be enough for predicting the direction of a company's returns and time series forecasting may work better with this type of data.