R Notebook

Code

Hide

```
#Loading the data

library(quantmod)
getSymbols("EA",src="yahoo",from=as.Date("2018-02-06"),to=as.Date("2023-02-06"))
```

```
[1] "EA"
```

Hide

```
head(EA)
```

```
           EA.Open EA.High EA.Low EA.Close EA.Volume EA.Adjusted
2018-02-06  118.86  123.35 117.76   123.13   4652300    121.4598
2018-02-07  122.86  125.00 122.18   123.05   4066900    121.3809
2018-02-08  123.00  123.00 116.52   116.54   5478900    114.9592
2018-02-09  117.96  122.14 114.67   120.64   5945100    119.0036
2018-02-12  121.78  124.16 121.53   122.22   3695100    120.5622
2018-02-13  120.85  123.13 120.58   122.28   2388700    120.6214
```

Hide

```
tail(EA)
```

```
           EA.Open EA.High EA.Low EA.Close EA.Volume EA.Adjusted
2023-01-27  129.14  130.57 128.79   128.87   1786200    128.6496
2023-01-30  128.92  129.47 128.11   128.99   2446900    128.7694
2023-01-31  129.19  129.99 128.38   128.68   3067700    128.4599
2023-02-01  116.78  117.22 112.58   116.76  14492300    116.5603
2023-02-02  117.50  117.52 114.10   115.99   6355600    115.7916
2023-02-03  115.15  115.54 113.78   113.92   4393500    113.7252
```

Hide

```
getSymbols("ATVI",src="yahoo",from=as.Date("2018-02-06"),to=as.Date("2023-02-06"))
```

```
[1] "ATVI"
```

Hide

```
head(ATVI)
```

```
           ATVI.Open ATVI.High ATVI.Low ATVI.Close ATVI.Volume ATVI.Adjusted
2018-02-06     66.00     69.84    65.72      69.70    10524300      67.60927
2018-02-07     69.62     70.86    69.43      69.46     6255200      67.37649
2018-02-08     69.63     69.79    65.76      65.83    11179300      63.85536
2018-02-09     66.99     67.78    63.32      67.08    18582300      65.06788
2018-02-12     67.16     69.20    67.16      68.32     8315700      66.27068
2018-02-13     67.96     68.21    67.18      68.03     5373600      65.98937
```

Hide

```
tail(ATVI)
```

```
           ATVI.Open ATVI.High ATVI.Low ATVI.Close ATVI.Volume ATVI.Adjusted
2023-01-27     75.50     76.76    75.22      76.61     4381700         76.61
2023-01-30     76.63     77.08    75.84      75.96     4247400         75.96
2023-01-31     76.13     77.00    75.85      76.57     4118000         76.57
2023-02-01     76.00     76.82    75.58      76.70     4575400         76.70
2023-02-02     76.50     77.39    76.07      77.11     4696100         77.11
2023-02-03     76.64     76.78    75.03      75.24     5779400         75.24
```

Hide

```
#EA - Electronic Arts
#ATVI - Activision Blizzard

#EA Log Return Calculation
EA_log_return_1<-diff(log(EA[,6]))
EA_log_return<-as.numeric(EA_log_return_1[-1])
head(EA_log_return)
```

```
[1] -0.0006498992 -0.0543562344  0.0345762877  0.0130119413  0.0004906305
[6]  0.0121116211
```

Hide

```
#Mean and sd EA Log Return Calculation
mean_EA<-mean(EA_log_return)
mean_EA
```

```
[1] -5.234602e-05
```

Hide

```
sd_EA<- sd(EA_log_return)
sd_EA
```

```
[1] 0.01994566
```

Hide

```
#EA Log Return Calculation
ATVI_log_return_1<-diff(log(ATVI[,6]))
ATVI_log_return<-as.numeric(ATVI_log_return_1[-1])
head(ATVI_log_return)
```

```
[1] -0.003448960 -0.053675557  0.018810460  0.018316489 -0.004253808  0.023534101
```

Hide

```
#Mean and sd EA Log Return Calculation
mean_ATVI<-mean(ATVI_log_return)
mean_ATVI
```

```
[1] 8.507391e-05
```

Hide

```
sd_ATVI<- sd(ATVI_log_return)
sd_ATVI
```

```
[1] 0.02164872
```

Hide

```
#1-Kernel Density estimate, Two Parametric Density estimates, Student-t (with your chosen par
ameter) and normal.
library(fGarch)

x<-seq(-0.1,0.1,by=0.001)

#Chosen Parameter df
df<-4
df
```

```
[1] 4
```

Hide

```
#median absolute deviation estimator for EA
mad_EA<-mad(EA_log_return,constant = sqrt(df/(df-2))/qt(0.75,df))
mad_EA
```

```
[1] 0.02049552
```

Hide

```
#median absolute deviation estimator for ATVI
mad_ATVI<-mad(ATVI_log_return,constant = sqrt(df/(df-2))/qt(0.75,df))
mad_ATVI
```

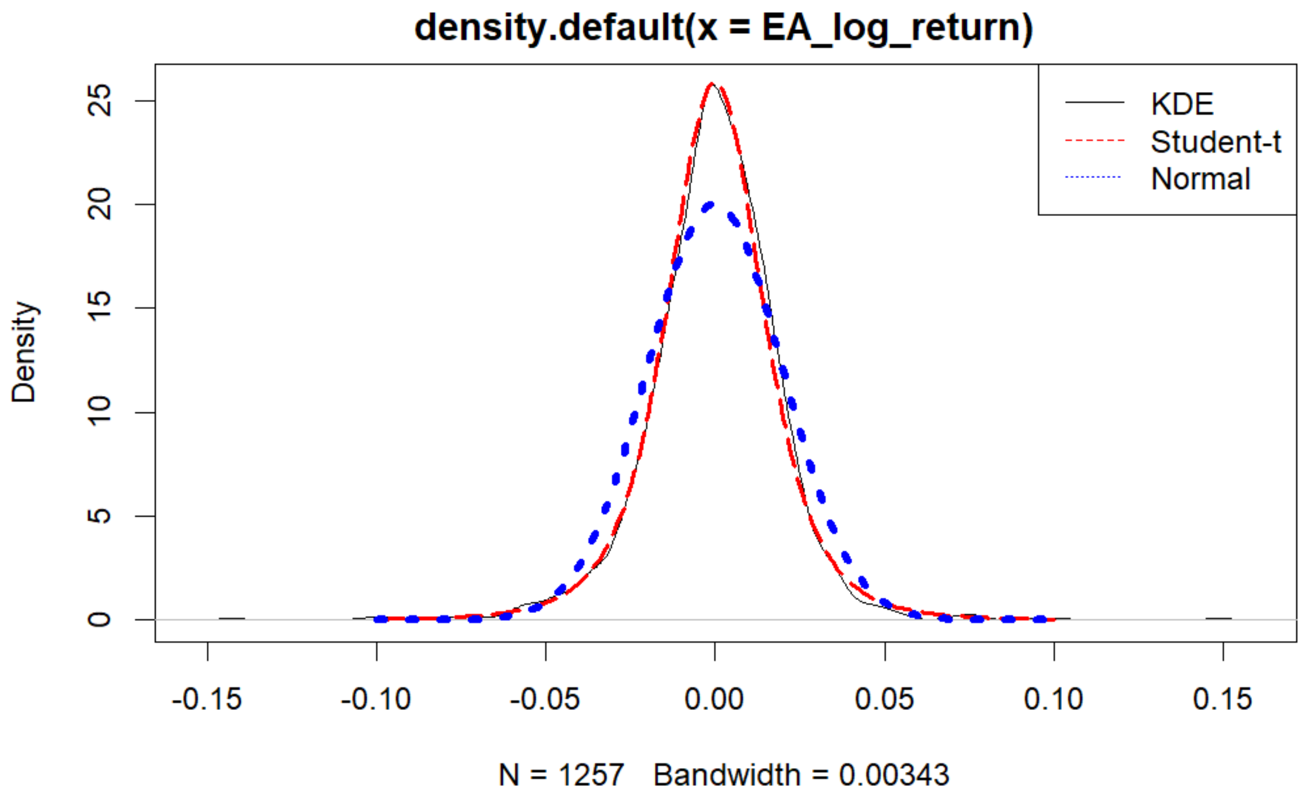
```
[1] 0.01904869
```

Hide

```
#Density plot for EA
plot(density(EA_log_return),col="black")
#EA Student t
lines(x,dstd(x, mean=mean_EA, sd=mad_EA, nu=df),lty=5, lwd=2, col="red")
```

Hide

```
#EA Normal
lines(x,dnorm(x, mean=mean_EA, sd=sd_EA),lty=3, lwd=4, col="blue")
legend("topright", legend = c("KDE", "Student-t", "Normal"),
       col = c("black", "red", "blue"), lty = 1:3)
```

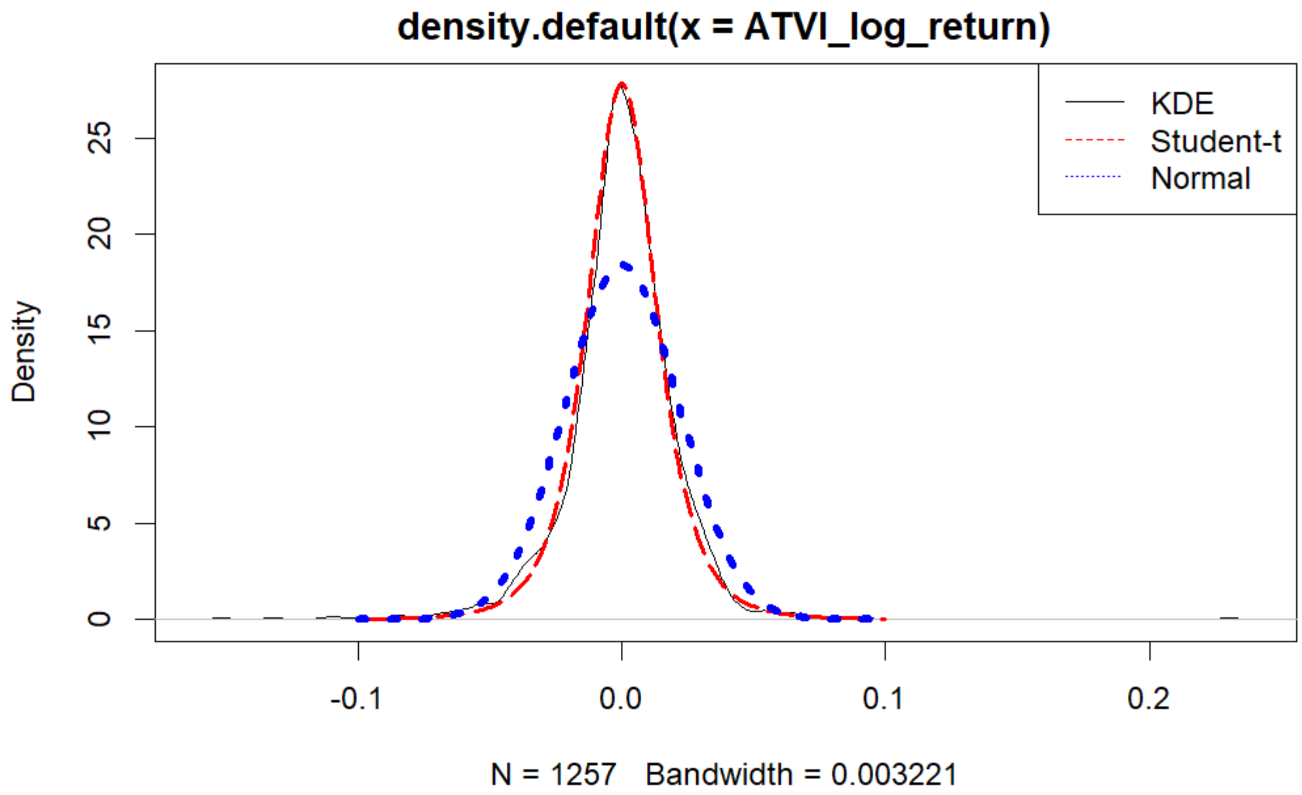


Hide

```
#Density plot for ATVI
plot(density(ATVI_log_return),col="black")
#ATVI Student t
lines(x,dstd(x, mean=mean_ATVI, sd=mad_ATVI, nu=df),lty=5, lwd=2, col="red")
```

Hide

```
#ATVI Normal
lines(x,dnorm(x, mean=mean_ATVI, sd=sd_ATVI),lty=3, lwd=4, col="blue")
legend("topright", legend = c("KDE", "Student-t", "Normal"),
       col = c("black", "red", "blue"), lty = 1:3)
```



Hide

```
#2- Logistic Regression (family= binomial) EA

volume_data <- Ad(EA) * as.numeric(Vo(EA))

# Preprocessing - Create a data frame with the variables of interest for EA Stock
EA_stock <- data.frame(
  direction = ifelse(diff(log(Cl(EA))) > 0, 1, 0),
  lag1 = lag(diff(log(Cl(EA)))),
  lag2 = lag(diff(log(Cl(EA))), 2),
  vol = diff(log(volume_data))

)
# Remove missing values
EA_stock <- na.omit(EA_stock)

# View the first few rows of the data frame
head(EA_stock)
```

	EA.Close <dbl>	EA.Close.1 <dbl>	EA.Close.2 <dbl>	EA.Adjusted <dbl>
2018-02-09	1	-0.0543562309	-0.0006498822	0.116239288
2018-02-12	1	0.0345763257	-0.0543562309	-0.462547792
2018-02-13	1	0.0130118117	0.0345763257	-0.435767702
2018-02-14	1	0.0004907812	0.0130118117	0.598441296
2018-02-15	1	0.0121114915	0.0004907812	-0.270052733
2018-02-16	0	0.0216592363	0.0121114915	-0.001338891

6 rows

Hide

```
attach(EA_stock)
```

```
The following objects are masked from EA_stock_new (pos = 5):

    EA.Close, EA.Close.1, EA.Close.2

The following objects are masked from EA_stock (pos = 7):

    EA.Adjusted, EA.Close, EA.Close.1, EA.Close.2

The following objects are masked from EA_stock_new (pos = 10):

    EA.Close, EA.Close.1, EA.Close.2

The following objects are masked from EA_stock (pos = 12):

    EA.Adjusted, EA.Close, EA.Close.1, EA.Close.2

The following objects are masked from EA_stock (pos = 17):

    EA.Adjusted, EA.Close, EA.Close.1, EA.Close.2
```

Hide

```
library(caret)
set.seed(123) # for reproducibility

# Create a vector of row indices
rows <- 1:nrow(EA_stock)


# Randomly sample 80% of the row indices for the training set
training_rows <- sample(rows, floor(0.7 * length(rows)))

# The remaining rows are for the testing set
testing_rows <- setdiff(rows, training_rows)

training_data <- EA_stock[training_rows, ]
testing_data <- EA_stock[-training_rows, ]

#Division Verification in number of Examples
cat("Number of examples in training data:", nrow(training_data), "\n")
```

```
Number of examples in training data: 878
```

Hide

```
cat("Number of examples in testing data:", nrow(testing_data), "\n")
```

```
Number of examples in testing data: 377
```

Hide

```
#Fitting the logistic regression (Binomial Family)-EA
glm.EA.fit<- glm(EA.Close~EA.Close.1+EA.Close.2+EA.Adjusted,training_data, family = binomial)

#Model Summary
summary(glm.EA.fit)
```

```
Call:
glm(formula = EA.Close ~ EA.Close.1 + EA.Close.2 + EA.Adjusted,
    family = binomial, data = training_data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.4162  -1.1775   0.9714   1.1726   1.3709

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.008805   0.067690   0.130   0.8965
EA.Close.1  -5.862084   3.439721  -1.704   0.0883 .
EA.Close.2  -3.378574   3.384507  -0.998   0.3182
EA.Adjusted  0.065342   0.175497   0.372   0.7096
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1217.1  on 877  degrees of freedom
Residual deviance: 1213.5  on 874  degrees of freedom
AIC: 1221.5

Number of Fisher Scoring iterations: 3
```

Hide

```
#Predictions EA
glm.EA.probs <- predict (glm.EA.fit , newdata=testing_data)
glm.EA.pred <- rep ("0", 377)
glm.EA.pred[glm.EA.probs > 0.5] <- "1"
table(glm.EA.pred, testing_data$EA.Close)
```

```
glm.EA.pred   0   1
          0 186 191
```

Hide

```
#Accuracy EA
mean(glm.EA.pred==testing_data$EA.Close)
```

```
[1] 0.4933687
```

Hide

```
#Logistic Regression (family= binomial) ATVI

volume_data2 <- Ad(ATVI) * as.numeric(Vo(ATVI))

# Preprocessing - Create a data frame with the variables of interest for EA Stock
ATVI_stock <- data.frame(
  direction = ifelse(diff(log(Cl(ATVI))) > 0, 1, 0),
  lag1 = lag(diff(log(Cl(ATVI)))),
  lag2 = lag(diff(log(Cl(ATVI))), 2),
  vol = diff(log(volume_data2))

)
# Remove missing values
ATVI_stock <- na.omit(ATVI_stock)

# View the first few rows of the data frame
head(ATVI_stock)
```

	ATVI.Close <dbl>	ATVI.Close.1 <dbl>	ATVI.Close.2 <dbl>	ATVI.Adjusted <dbl>
2018-02-09	1	-0.05367534	-0.003449242	0.52695612
2018-02-12	1	0.01881027	-0.053675341	-0.78574773
2018-02-13	0	0.01831658	0.018810275	-0.44090103
2018-02-14	1	-0.00425378	0.018316583	0.31578366
2018-02-15	1	0.02353396	-0.004253780	-0.11898067
2018-02-16	0	0.03304446	0.023533959	0.02440424

6 rows

Hide

```
attach(ATVI_stock)
```

```
The following objects are masked from ATVI_stock_new (pos = 5):

    ATVI.Close, ATVI.Close.1, ATVI.Close.2

The following objects are masked from ATVI_stock (pos = 7):

    ATVI.Adjusted, ATVI.Close, ATVI.Close.1, ATVI.Close.2

The following objects are masked from ATVI_stock_new (pos = 9):

    ATVI.Close, ATVI.Close.1, ATVI.Close.2

The following objects are masked from ATVI_stock (pos = 12):

    ATVI.Adjusted, ATVI.Close, ATVI.Close.1, ATVI.Close.2

The following objects are masked from ATVI_stock (pos = 14):

    ATVI.Adjusted, ATVI.Close, ATVI.Close.1, ATVI.Close.2
```

Hide

```
library(caret)
set.seed(123) # for reproducibility

# Create a vector of row indices
rows2 <- 1:nrow(ATVI_stock)

# Randomly sample 80% of the row indices for the training set
training_rows2 <- sample(rows2, floor(0.7 * length(rows2)))

# The remaining rows are for the testing set
testing_rows2 <- setdiff(rows2, training_rows2)

training_data2 <- ATVI_stock[training_rows2, ]
testing_data2 <- ATVI_stock[-training_rows2, ]

#Division Verification in number of Examples
cat("Number of examples in training data:", nrow(training_data2), "\n")
```

```
Number of examples in training data: 878
```

Hide

```
cat("Number of examples in testing data:", nrow(testing_data2), "\n")
```

```
Number of examples in testing data: 377
```

Hide

```
#Fitting the logistic regression (Binomial Family)-EA
glm.ATVI.fit<- glm(ATVI.Close~ATVI.Close.1+ATVI.Close.2+ATVI.Adjusted,training_data2, family
= binomial)

#Model Summary
summary(glm.ATVI.fit)
```

```
Call:
glm(formula = ATVI.Close ~ ATVI.Close.1 + ATVI.Close.2 + ATVI.Adjusted,
    family = binomial, data = training_data2)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6299  -1.1801   0.8826   1.1604   1.5433

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)     0.04122    0.06809   0.605 0.544984
ATVI.Close.1  -12.86079    3.50522  -3.669 0.000243 ***
ATVI.Close.2   -1.58234    3.31932  -0.477 0.633570
ATVI.Adjusted  -0.06879    0.17822  -0.386 0.699496
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1216.8  on 877  degrees of freedom
Residual deviance: 1201.7  on 874  degrees of freedom
AIC: 1209.7

Number of Fisher Scoring iterations: 4
```

Hide

```
#Predictions ATVI
glm.ATVI.probs <- predict (glm.ATVI.fit , newdata=testing_data2)
glm.ATVI.pred <- rep ("0", 377)
glm.ATVI.pred[glm.ATVI.probs > 0.5] <- "1"
table(glm.ATVI.pred, testing_data2$ATVI.Close)
```

```
glm.ATVI.pred   0   1
            0 180 180
            1  10   7
```

Hide

```
#Accuracy ATVI
mean(glm.ATVI.pred==testing_data2$ATVI.Close)
```

```
[1] 0.4960212
```

Hide

```
#Comments on generated logistic regression models for EA and ATVI
# For this application the logistic regression misclassified a lot of the data when taking in
consideration both stocks. it resulted in giving more true negatives and false positives. For
EA resulted in a accuracy of 0.4933687 equivalent to 49.34% and ATVI = 0.4960212 equivalent t
o 49.6%
```

Hide

```
# Preprocessing - Create a data frame with the variables of interest for EA Stock For LDA
EA_stock_new <- data.frame(
  direction = ifelse(diff(log(Cl(EA))) > 0, 1, 0),
  lag1 = lag(diff(log(Cl(EA)))),
  lag2 = lag(diff(log(Cl(EA))), 2)
)
# Remove missing values
EA_stock_new <- na.omit(EA_stock_new)

# View the first few rows of the data frame
head(EA_stock_new)
```

	EA.Close <dbl>	EA.Close.1 <dbl>	EA.Close.2 <dbl>
2018-02-09	1	-0.0543562309	-0.0006498822
2018-02-12	1	0.0345763257	-0.0543562309
2018-02-13	1	0.0130118117	0.0345763257
2018-02-14	1	0.0004907812	0.0130118117
2018-02-15	1	0.0121114915	0.0004907812
2018-02-16	0	0.0216592363	0.0121114915

6 rows

Hide

```
attach(EA_stock_new)
```

```
The following objects are masked from EA_stock (pos = 4):

    EA.Close, EA.Close.1, EA.Close.2

The following objects are masked from EA_stock_new (pos = 7):

    EA.Close, EA.Close.1, EA.Close.2

The following objects are masked from EA_stock (pos = 9):

    EA.Close, EA.Close.1, EA.Close.2

The following objects are masked from EA_stock_new (pos = 12):

    EA.Close, EA.Close.1, EA.Close.2

The following objects are masked from EA_stock (pos = 14):

    EA.Close, EA.Close.1, EA.Close.2

The following objects are masked from EA_stock (pos = 19):

    EA.Close, EA.Close.1, EA.Close.2
```

Hide

```
library(caret)
set.seed(123) # for reproducibility

# Create a vector of row indices
rows3 <- 1:nrow(EA_stock_new)

# Randomly sample 80% of the row indices for the training set
training_rows3 <- sample(rows3, floor(0.7 * length(rows3)))

# The remaining rows are for the testing set
testing_rows3 <- setdiff(rows3, training_rows3)

training_data3 <- EA_stock_new[training_rows3, ]
testing_data3 <- EA_stock_new[-training_rows3, ]

#Division Verification in number of Examples
cat("Number of examples in training data:", nrow(training_data3), "\n")
```

```
Number of examples in training data: 878
```

Hide

```
cat("Number of examples in testing data:", nrow(testing_data3), "\n")
```

```
Number of examples in testing data: 377
```

Hide

```
#3- LDA (Linear Discriminant Analysis) EA

library (MASS)

lda.EA.fit <- lda (EA.Close ~ EA.Close.1 + EA.Close.2 , data = training_data3)
lda.EA.fit
```

```
Call:
lda(EA.Close ~ EA.Close.1 + EA.Close.2, data = training_data3)

Prior probabilities of groups:
        0         1 
0.4965831 0.5034169 

Group means:
     EA.Close.1    EA.Close.2
0  0.0004158849  0.0001190794
1 -0.0017165436 -0.0009265810

Coefficients of linear discriminants:
                 LD1
EA.Close.1 -45.94584
EA.Close.2 -26.81085
```

Hide

```
# Prediction LDA EA
lda.EA.pred <- predict (lda.EA.fit , newdata=testing_data3)
lda.EA.class <- lda.EA.pred$class

#Confusion Matrix EA
table(lda.EA.class, testing_data3$EA.Close)
```

```
            
lda.EA.class   0   1
           0  94 101
           1  92  90
```

Hide

```
#Accuracy EA
mean(lda.EA.class==testing_data3$EA.Close)
```

```
[1] 0.4880637
```

Hide

```
# Preprocessing - Create a data frame with the variables of interest for ATVI Stock LDA
ATVI_stock_new <- data.frame(
  direction = ifelse(diff(log(Cl(ATVI))) > 0, 1, 0),
  lag1 = lag(diff(log(Cl(ATVI)))),
  lag2 = lag(diff(log(Cl(ATVI))), 2)
)
# Remove missing values
ATVI_stock_new <- na.omit(ATVI_stock_new)

# View the first few rows of the data frame
head(ATVI_stock_new)
```

	ATVI.Close <dbl>	ATVI.Close.1 <dbl>	ATVI.Close.2 <dbl>
2018-02-09	1	-0.05367534	-0.003449242
2018-02-12	1	0.01881027	-0.053675341
2018-02-13	0	0.01831658	0.018810275
2018-02-14	1	-0.00425378	0.018316583
2018-02-15	1	0.02353396	-0.004253780
2018-02-16	0	0.03304446	0.023533959

6 rows

Hide

```
attach(ATVI_stock_new)
```

```
The following objects are masked from ATVI_stock (pos = 4):

    ATVI.Close, ATVI.Close.1, ATVI.Close.2

The following objects are masked from ATVI_stock_new (pos = 7):

    ATVI.Close, ATVI.Close.1, ATVI.Close.2

The following objects are masked from ATVI_stock (pos = 9):

    ATVI.Close, ATVI.Close.1, ATVI.Close.2

The following objects are masked from ATVI_stock_new (pos = 11):

    ATVI.Close, ATVI.Close.1, ATVI.Close.2

The following objects are masked from ATVI_stock (pos = 14):

    ATVI.Close, ATVI.Close.1, ATVI.Close.2

The following objects are masked from ATVI_stock (pos = 16):

    ATVI.Close, ATVI.Close.1, ATVI.Close.2
```

Hide

```
library(caret)
set.seed(123) # for reproducibility

# Create a vector of row indices
rows4 <- 1:nrow(ATVI_stock_new)

# Randomly sample 70% of the row indices for the training set
training_rows4 <- sample(rows4, floor(0.7 * length(rows4)))

# The remaining rows are for the testing set
testing_rows4 <- setdiff(rows4, training_rows4)

training_data4 <- ATVI_stock_new[training_rows4, ]
testing_data4 <- ATVI_stock_new[-training_rows4, ]

#Division Verification in number of Examples
cat("Number of examples in training data:", nrow(training_data4), "\n")
```

```
Number of examples in training data: 878
```

Hide

```
cat("Number of examples in testing data:", nrow(testing_data4), "\n")
```

```
Number of examples in testing data: 377
```

Hide

```
#LDA (Linear Discriminant Analysis) ATVI

lda.ATVI.fit <- lda (ATVI.Close ~ ATVI.Close.1 + ATVI.Close.2 , data = training_data4)

lda.ATVI.fit
```

```
Call:
lda(ATVI.Close ~ ATVI.Close.1 + ATVI.Close.2, data = training_data4)

Prior probabilities of groups:
        0         1 
0.4897494 0.5102506 

Group means:
  ATVI.Close.1  ATVI.Close.2
0  0.002833969 -0.0002650264
1 -0.002700683 -0.0003368555

Coefficients of linear discriminants:
                    LD1
ATVI.Close.1 -46.274852
ATVI.Close.2  -5.746408
```

Hide

```
# Prediction LDA ATVI

lda.ATVI.pred <- predict (lda.ATVI.fit , newdata=testing_data4)
lda.ATVI.class <- lda.ATVI.pred$class

#Confusion Matrix ATVI
table(lda.ATVI.class, testing_data4$ATVI.Close)
```

```
lda.ATVI.class   0   1
             0  73  84
             1 117 103
```

Hide

```
#Accuracy ATVI
mean(lda.ATVI.class==testing_data4$ATVI.Close)
```

```
[1] 0.4668435
```

Hide

```
#The LDA uses prior probabilities to determine up and down and this time it predicted more tr
ue positives and true negatives compared to the logistic regression. For EA had a up probabil
ity of 50.3% and down probability of 49.7 and resulting in a overall accuracy of 48.8%.

#For ATVI, the up probability was 51.03% and down probability of 48.98 and resulting in a ove
rall accuracy of 46.6%.

#The amount of data being classified as false positives and negatives are affecting the overa
ll performance of the classifier.
```

Hide

```
#Preprocess to perform KNN EA

# Create X.train and X.test data frames that exclude the EA.Close
X.train3 <- training_data3[, -which(names(training_data3) == "EA.Close")]
X.test3 <- testing_data3[, -which(names(testing_data3) == "EA.Close")]
Y.train3 <- training_data3$EA.Close
Y.test3 <- testing_data3$EA.Close
```

Hide

```
#4- KNN (K-Nearest Neighbors) EA . K=sqrt(n=1255)

library(class)

# Build the KNN Classifier model

knn.EA.pred = knn(X.train3, X.test3, Y.train3, k=sqrt(1255))

# KNN with K=13 (by trial 13 was found to be an optimal parameter)
knn.EA.pred1 = knn(X.train3, X.test3, Y.train3, k=13)

##Confusion matrices

#k=sqrt(1255)
table(knn.EA.pred, Y.test3)
```

```
           Y.test3
knn.EA.pred   0   1
          0  82  91
          1 104 100
```

Hide

```
#k=13
table(knn.EA.pred1, Y.test3)
```

```
            Y.test3
knn.EA.pred1   0   1
           0  90  89
           1  96 102
```

Hide

```
# Calculate the test errors
test_error1 <- sum(knn.EA.pred != Y.test3) / length(Y.test3)
test_error2 <- sum(knn.EA.pred1 != Y.test3) / length(Y.test3)

cat("Test error1:", test_error1, "\n")
```

```
Test error1: 0.5172414
```

Hide

```
cat("Test error2:", test_error2, "\n")
```

```
Test error2: 0.4907162
```

Hide

```
#Accuracy EA

Accuracy<-1-test_error1
Accuracy2<-1-test_error2

cat("Accuracy:", Accuracy, "\n")
```

```
Accuracy: 0.4827586
```

Hide

```
cat("Accuracy2:", Accuracy2, "\n")
```

```
Accuracy2: 0.5092838
```

Hide

```
#Preprocess to perform KNN ATVI

# Create X.train and X.test data frames that exclude the ATVI.Close
X.train4 <- training_data4[, -which(names(training_data4) == "ATVI.Close")]
X.test4 <- testing_data4[, -which(names(testing_data4) == "ATVI.Close")]
Y.train4 <- training_data4$ATVI.Close
Y.test4 <- testing_data4$ATVI.Close
```

Hide

```
#KNN (K-Nearest Neighbors) ATVI . K=sqrt(n=1255)

# Build the KNN Classifier model

knn.ATVI.pred = knn(X.train4, X.test4, Y.train4, k=sqrt(1255))

# KNN with K=13 (by trial 13 was found to be an optimal parameter)
knn.ATVI.pred1 = knn(X.train4, X.test4, Y.train4, k=13)

##Confusion matrices

#k=sqrt(1255)
table(knn.ATVI.pred, Y.test4)
```

```
             Y.test4
knn.ATVI.pred  0  1
            0 92 89
            1 98 98
```

Hide

```
#k=13
table(knn.ATVI.pred1, Y.test4)
```

```
             Y.test4
knn.ATVI.pred1   0   1
             0  87  84
             1 103 103
```

Hide

```
# Calculate the test errors
test_error3 <- sum(knn.ATVI.pred != Y.test4) / length(Y.test4)
test_error4 <- sum(knn.ATVI.pred1 != Y.test4) / length(Y.test4)

cat("Test error3:", test_error3, "\n")
```

```
Test error3: 0.4960212
```

Hide

```
cat("Test error4:", test_error4, "\n")
```

```
Test error4: 0.4960212
```

Hide

```
#Accuracy ATVI

Accuracy3<-1-test_error3
Accuracy4<-1-test_error4

cat("Accuracy3:", Accuracy3, "\n")
```

```
Accuracy3: 0.5039788
```

Hide

```
cat("Accuracy4:", Accuracy4, "\n")
```

```
Accuracy4: 0.5039788
```

Hide

```
#Finally, the KNN Classifier:

#EA first the sqrt of the sample size was used as it is known of sometimes be a good estimato
r for the n_neighbours and then by trial we found that k=13 was giving the best overall accur
acy. first k 48.3% of accuracy and second k about 51% of accuracy.

#ATVI first k 50.4% of accuracy and second k giving 50.4% of accuracy.
```