Predicting S&P 500 Market Return Direction: Classification

By Vishwa Pardeshi

In this notebook, we will predict the market return direction for S&P 500 index using the Weekly S&P Stock Market Data. This is a classification problem where, we aim to predict the direction - Down or Up.

Learning Outcome: By following the notebook you will be able to

1. Perform context inspired EDA to understand relationship between predictor variables and Direction (whether the market had a positive or negative return on a given week)
2. Implement & infer Linear Discriminant Analysis
3. Implement & infer Quadratic Discriminant Analysis

Setup

```r
knitr::opts_chunk$set(echo = TRUE)
library(ISLR)
library(corrplot)
library(ggplot2)
library(MASS)
```

Exploratory Data Analysis

Glimpse of the data

Let us first look at the head of the dataset.

```
##   Year   Lag1   Lag2   Lag3   Lag4   Lag5    Volume  Today Direction
## 1 1990  0.816  1.572 -3.936 -0.229 -3.484 0.1549760 -0.270      Down
## 2 1990 -0.270  0.816  1.572 -3.936 -0.229 0.1485740 -2.576      Down
## 3 1990 -2.576 -0.270  0.816  1.572 -3.936 0.1598375  3.514        Up
## 4 1990  3.514 -2.576 -0.270  0.816  1.572 0.1616300  0.712        Up
## 5 1990  0.712  3.514 -2.576 -0.270  0.816 0.1537280  1.178        Up
## 6 1990  1.178  0.712  3.514 -2.576 -0.270 0.1544440 -1.372      Down
```

```r
cat("The Weekly dataset shape is ", dim(Weekly)[1], "x", dim(Weekly)[2])
```

```
## The Weekly dataset shape is  1089 x 9
```

The 9 columns capture the year, weekly lag, volumne, % return for today and direction (a factor).

```r
summary(Weekly)
```

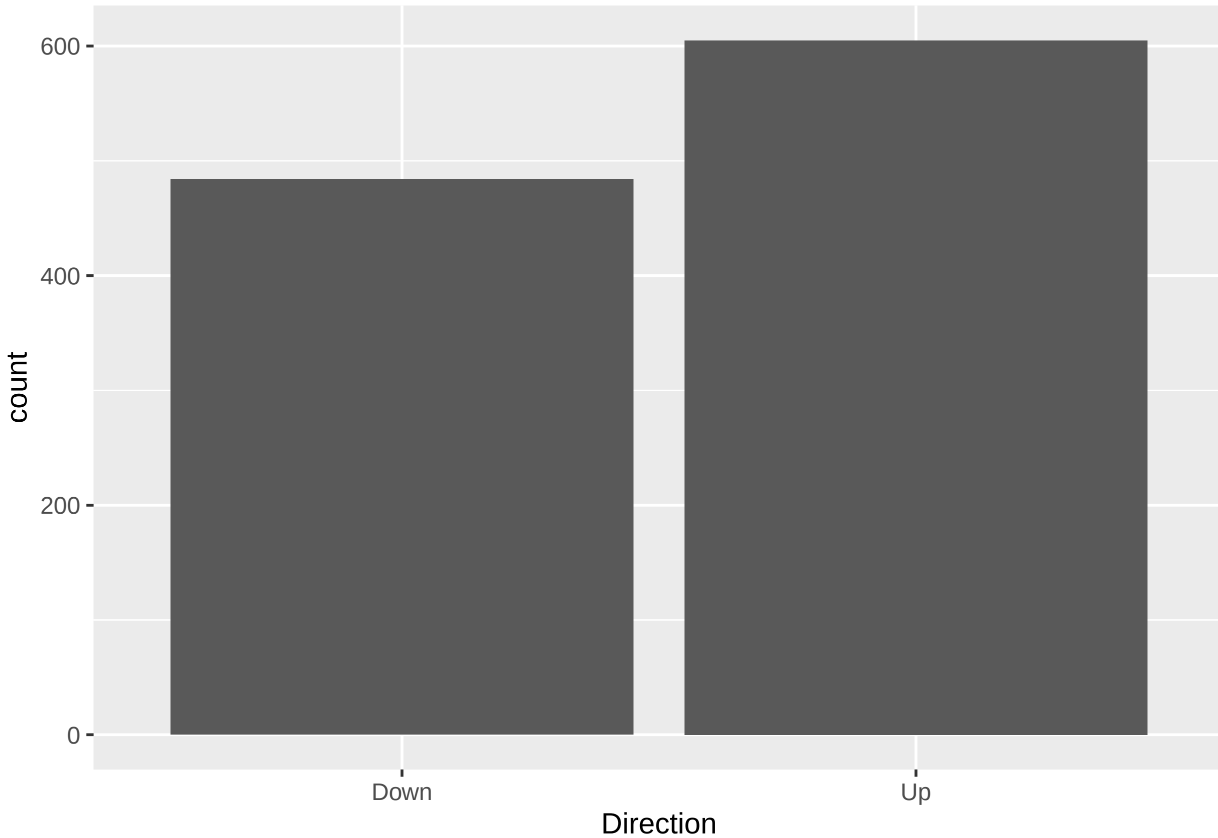
```
##       Year           Lag1               Lag2               Lag3         
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950  
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580  
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410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

We can observe from the summary, that the distribution of values in the Direction class is comparable and not highly skewed.

Target Class Distribution

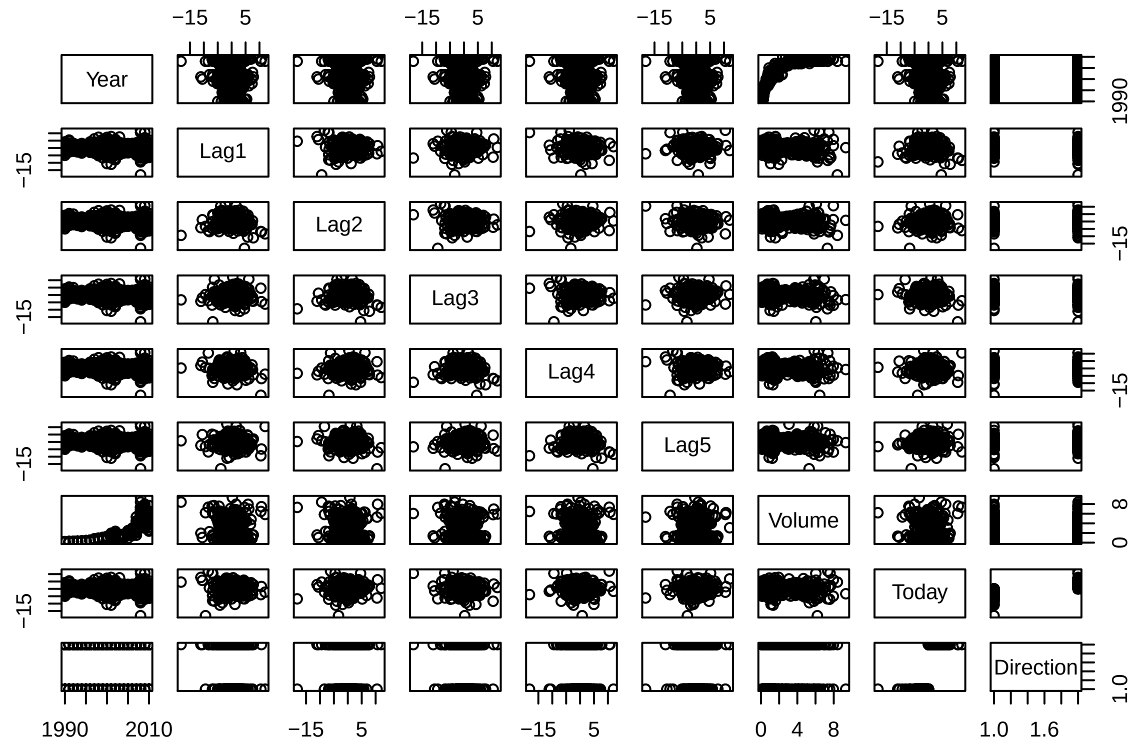
We observe that the distribution of target values is not highly skewed. Thus, class imbalance is absent.

```
ggplot(Weekly) +
    geom_bar(aes(x = Direction))
```



Scatterplot Matrix

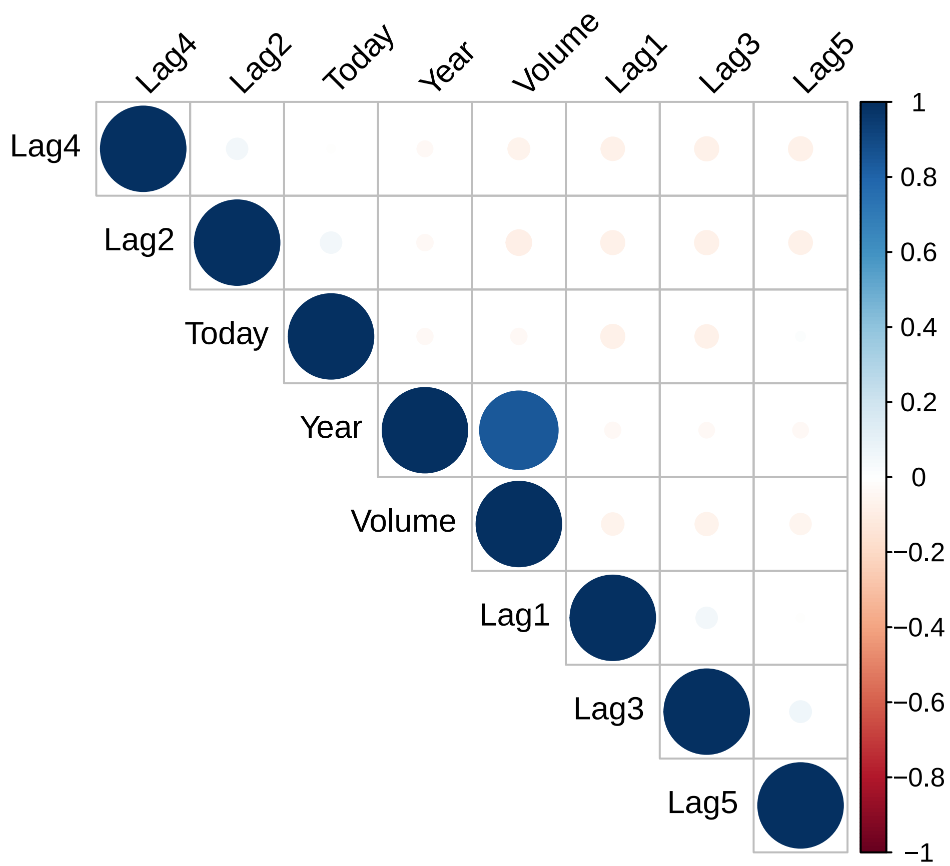
```
pairs(Weekly)
```



From the scatterplot matrix, we notice that the scatterplot from Year and Volumn reveals an almost logarithmic relationship

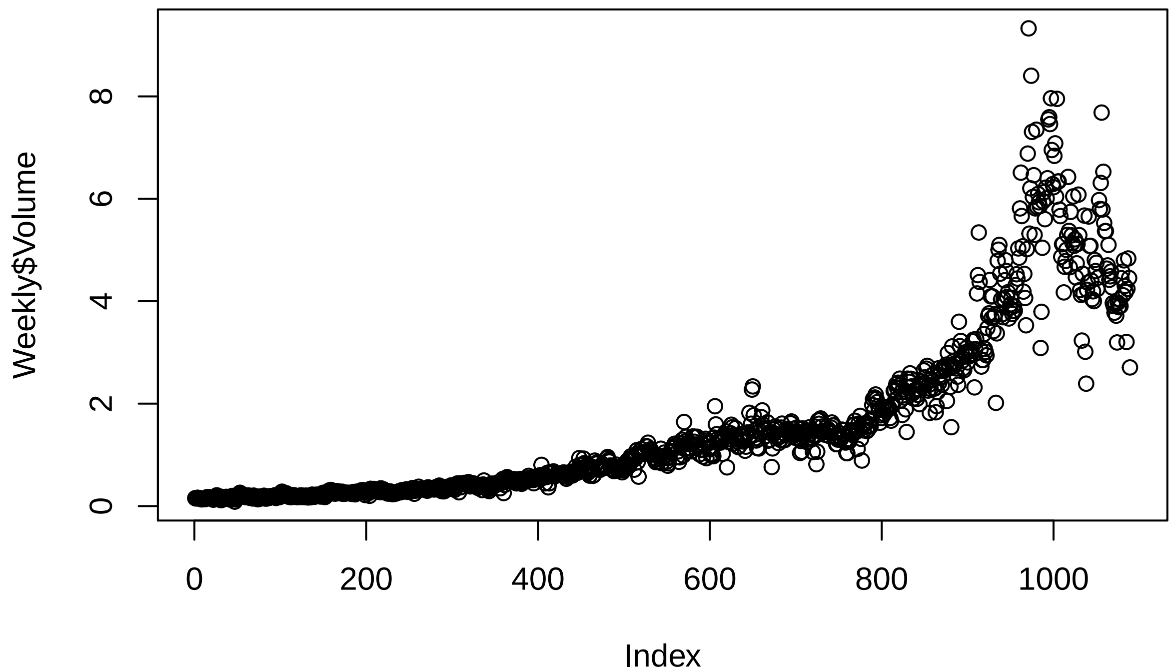
Heatmap of Correlation

```r
correlation_matrix <- cor(Weekly[, -which(names(Weekly) == "Direction")])
corrplot(correlation_matrix, type = "upper", order = "hclust", tl.col = "black", tl.srt = 45)
```



Additionally, Year & Volume have a high positive correlation By plotting the data we see that Volume is increasing over time. In other words, **the average number of shares traded daily increased from 1990 to 2010.**

```
plot(Weekly$Volume)
```



Baseline Logistic Regression

A logistic regression model is trained on the full training dataset to predict 'Direction' using the five lag variables and Volume as predictor.

```
logit.fit <- glm(Direction~., data=Weekly[,c(2:7,9)], family=binomial)
summary(logit.fit)
```

```
##
## Call:
## glm(formula = Direction ~ ., family = binomial, data = Weekly[,
##     c(2:7, 9)])
##
## 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
```

From the summary of the logistic linear model, we notice that only Lag2 variable has a statistical significant predictive value for alpha level 0.01.

Confusion matrix for logistic regression model

Confusion matrix helps capture the performance of classification model. Here, we create the confusion matrix by using the predicted class.

The class is predicted by using the class probability predicted using the above logistic regression model. A prediction cut off of 0.5 is used.

```
logit_prob <- predict(logit.fit, Weekly, type="response")
logit_pred <- ifelse(logit_prob > 0.5, "Up", "Down")
conf_matrix <- table(logit_pred, Weekly$Direction)
conf_matrix
```

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

From the confusion matrix, we notice that a lot of data points belonging to Down class are misclassified as Up. This indicates that the logistic regression model performs well for Up class.

Let's look at the various performance metric by writing a function to calculate the values

```
generate_metric <- function(data, confMatrix){
  TP = confMatrix[2,2]
  FP = confMatrix[1,2]
  FN = confMatrix[2,1]
  TN = confMatrix[1,1]
  total_accuracy <- (TP + TN)/nrow(data)
  class_a_accuracy <- TN/(TN + FN)
  class_b_accuracy <- TP/(FP + TP)
  precision <- TP/(TP+FP) # Calculate the Precision
  recall <- TP/(TP+FN) # calculate recall
  metrics <- data.frame("measurements"=c("Total Accuracy", "Class Down Accuracy", "Class Up Accuracy",
  return(metrics)

}
```

```
logit_metric <-generate_metric(Weekly, conf_matrix)
logit_metric
```

```
##          measurements      rate
## 1      Total Accuracy 0.5610652
## 2 Class Down Accuracy 0.1115702
## 3   Class Up Accuracy 0.9206612
## 4           Precision 0.9206612
## 5              Recall 0.5643364
```

Looking at the total accuracy which is just above 50% isnt any better than a random binary classifier. Additionally, the accuracy of our logistic model performs worse than a random model for data points belonging to the 'Down' class. This is reflected in the low recall value.

We recall that the logistic regression model had very underwhelming p- values associated with all of the predictors, and that the smallest p-value, though not very small, corresponded to Lag2. erhaps by removing the variables that appear not to be helpful in predicting Direction, we can obtain a more effective model.

Linear Discriminant Analysis using training data from 1990 to 2008

Using predictors that have no relationship with the response tends to cause a deterioration in the test error rate (since such predictors cause an increase in variance without a corresponding decrease in bias), and so removing such predictors may in turn yield an improvement.

Here,`Lag2` is used as the only predictor. Here, we will use LDA

Create training & test subset

```
train = (Weekly$Year<=2008)
test = Weekly[!train,]
```

Perform LDA on training data

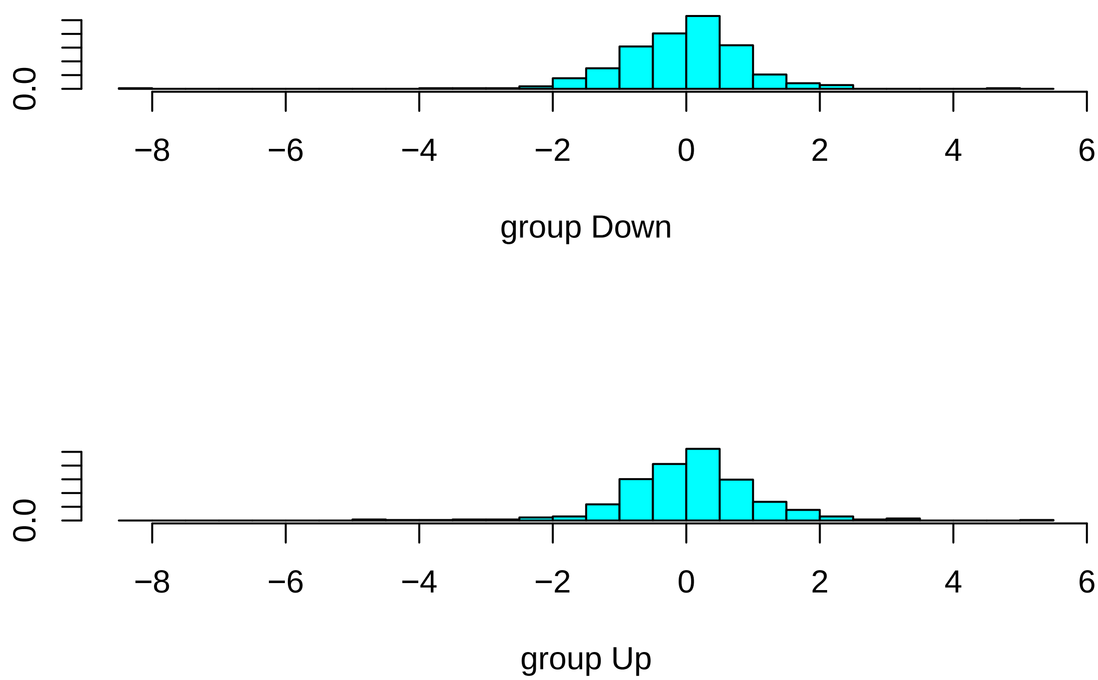
```
lda.fit <- lda(Direction ~ Lag2, data=Weekly, subset=train)
lda.fit
```

```
## Call:
## lda(Direction ~ Lag2, data = Weekly, subset = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##             Lag2
## Down -0.03568254
## Up    0.26036581
##
## Coefficients of linear discriminants:
```

```
##              LD1
## Lag2 0.4414162
```

1. The LDA output indicates the prior probabilities; in other words, 44.77% of the training observations correspond to days during which the market went down.
2. It also provides the group means; these are the average of each predictor within each class, and are used by LDA as estimates of mu. These suggest that there is a tendency for the previous 2 days' returns to be negative on days when the market decreases.
3. The coefficients of linear discriminants output provides the coefficient of Lag2 that are used to form the LDA decision rule.
4. The plots of the linear discriminants, obtained by computing $0.44 \times$ Lag2 for each of the training observations.

```r
plot(lda.fit)
```



Predict

```r
lda_preds <- predict(lda.fit, newdata=test)
conf_matrix_lda <- table(lda_preds$class, test$Direction)
conf_matrix_lda
```

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

```r
# compute overall of correct predictions
metric_lda <- generate_metric(test, conf_matrix_lda)
metric_lda
```

```
##          measurements      rate
## 1      Total Accuracy 0.6250000
## 2 Class Down Accuracy 0.2093023
## 3   Class Up Accuracy 0.9180328
## 4           Precision 0.9180328
## 5              Recall 0.6222222
```

The current linear discriminant analysis has an improved accuracy from the baseline model. There is also an observed improvement in classifying 'Down' values. But this improvement is not impressive nor would be acceptable for making good bets. In other words, when linear discriminant analysis predicts a decrease in the model, it has a 20.93% accuracy rate which is poorer than a naive approach. This suggests a possible trading strategy of buying on days when the model predicts an increasing market, and avoiding trades on days when a decrease is predicted. Ofcourse this is not a reliable strategy.

Quadratic Discriminant Analysis

When the true decision boundaries are linear, then the LDA and logistic regression approaches will tend to perform well. When the boundaries are moderately non-linear, QDA may give better results. Thus, we will explore QDA.

Perform QDA on training data

```r
quad.fit <- qda(Direction ~ Lag2, data=Weekly, subset=train)
quad.fit
```

```
## Call:
## qda(Direction ~ Lag2, data = Weekly, subset = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##             Lag2
## Down -0.03568254
## Up    0.26036581
```

1. The QDA output indicates the prior probabilities; in other words, 44.77% of the training observations correspond to days during which the market went down.

2. It also provides the group means; these are the average of each predictor within each class, and are used by QDA as estimates of mu. These suggest that there is a tendency for the previous 2 days' returns to be negative on days when the market decreases.

3. This has the same result as LDA except for coefficients which are not generated for QDA.

Predict

```
qda_preds <- predict(quad.fit, newdata=test)
conf_matrix_qda <- table(qda_preds$class, test$Direction)
conf_matrix_qda
```

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

```
# compute overall of correct predictions
metric_qda <- generate_metric(test, conf_matrix_qda)
metric_qda
```

```
##          measurements      rate
## 1      Total Accuracy 0.5865385
## 2 Class Down Accuracy 0.0000000
## 3   Class Up Accuracy 1.0000000
## 4           Precision 1.0000000
## 5              Recall 0.5865385
```

QDA accuracy is lower than LDA suggesting an absence of quadratic relation. On carefully observing the confusion matrix, we notice that the QDA classifies all the data point to UP which is as good as a randoml, naive approach and hence should not be picked.