Predicting Up and Down Movement of the S&P 500 Stock Index Between 1990 and 2010

Presented to

CSUF

COLLEGE OF

Natural Sciences and Mathematics

California State University, Fullerton

Math 533 Fall 2023

Prepared by

Emilio VASQUEZ

September 23, 2023

Contents

1 Overview

The data provided, 'Weekly' from the ISLR2 package, contains weekly percentage returns for the S&P 500 stock index between 1990 and 2010. The task at hand requires building predictive models to predict a binary categorical response as (Up and Down) indicating whether the market had a positive or negative return for a given week. The report will analyze the performance in classification when Logistic Regression, Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), K-Nearest Neighbors (KNN), and Naive Bayes are applied. Potential predictors within the provided data are:

- Year - The year that the observation was recorded
- Lag1 - Percentage return for previous week
- Lag2 - Percentage return for 2 weeks previous
- Lag3 - Percentage return for 3 weeks previous
- Lag4 - Percentage return for 4 weeks previous
- Lag5 - Percentage return for 5 weeks previous
- Volume - Volume of shares traded (average number of daily shares traded in billions)
- Today - Percentage return for this week

2 Exploratory Data Analysis

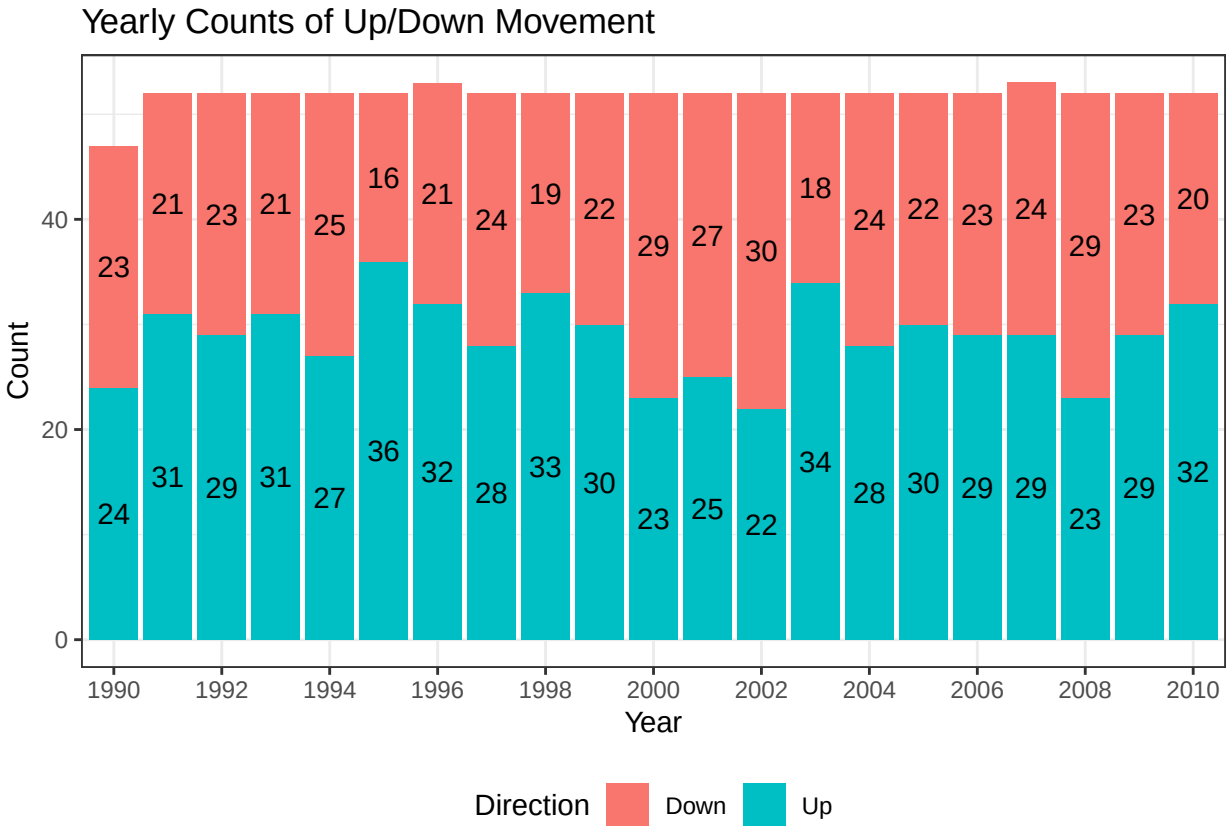

Figure 1. Yearly counts of weekly close (up/down)

Provided that the task at hand requires predicting a binary response, we start by looking at how often the market closed up or down since 1990. For the most part the market has largely gone up in a given year. There are a few notable highlights where in 2008 there were 29 weeks where the market closed down which coincides with the financial crisis of that time. In the following years, the S&P bounced back in 2009 and 2010 with 29 weeks and 32 weeks closing upwards. In 2002, there were only 22 weeks with an upward direction and 1990 had 24 weeks up. The years with the highest count of upward market direction were in 1995 and 2003.

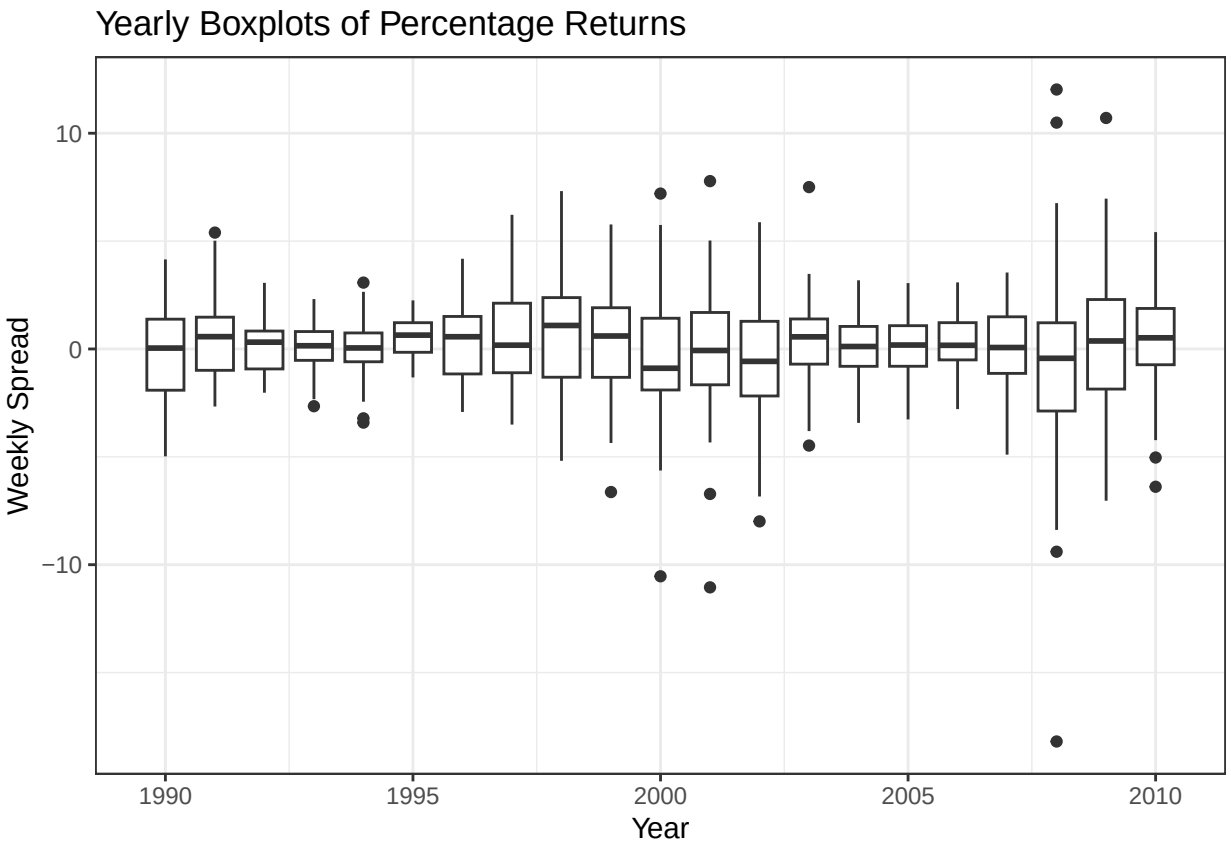

Figure 2. Boxplot of percentage returns

In figure 2 above, one can see the weekly distribution of weekly returns for a given year. 2008 visually has the widest spread of data in the entire data set. There were weeks where the weekly spread was above 10% (twice!) and a week where the weekly spread was below 15%. The interquartile range (top of the box to the bottom of the box) which encapsulates the 25 and 75 percentile of data was the largest of all years as well. In the years leading up to 2008, there was not much difference in the weekly spread as the boxplots are similar in length.

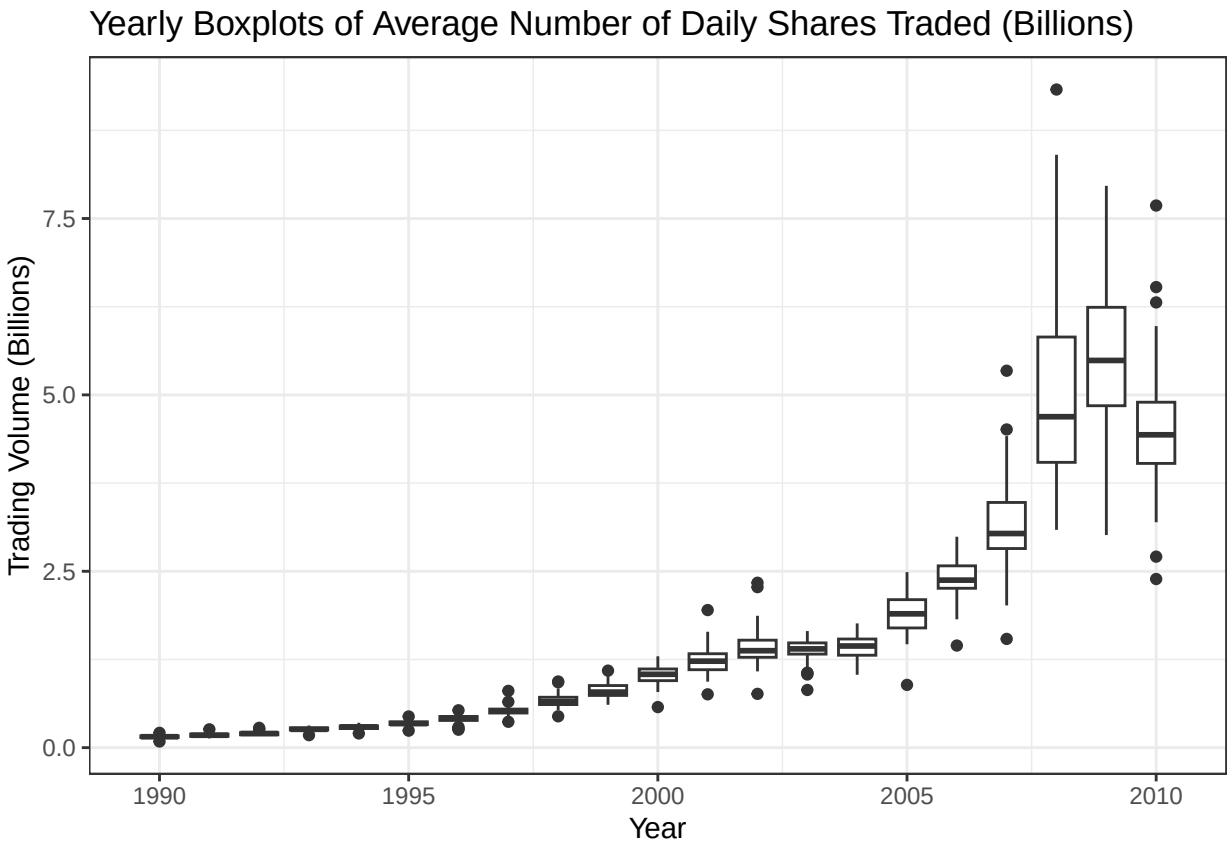

Figure 3. Boxplot of trading volume

With the boxplot above, one can see the mass difference in trading volume since 1990. This might be attributed to the fact that around 2005 the world was becoming more connected and it was becoming easier than ever for the average person to participate in the stock market with internet brokerages becoming more of the norm. The need for calling a broker for every transaction was slowly going away to where we are now where trading can be done from an app on a phone.

The increase in market volume was also driven by a combination of factors that created a seemingly robust and flourishing economy. The housing market was a key driver with soaring home prices enticing both homeowners and investors to participate. The housing boom led to an increase in mortgage lending, as financial institutions developed increasingly complex and risky financial products, such as mortgage-backed securities and collateralized debt obligations, which were in high demand among investors searching for higher returns.

Simultaneously, historically low interest rates set by the Federal Reserve encouraged borrowing and investment, while regulatory changes and a general belief in the invincibility of financial innovation led to a introduction of complex derivatives and trading strategies. This thrilling period in the financial markets contributed to the overall increase in market volume as investors sought to capitalize on the seemingly limitless opportunities for profit.

However, beneath the surface and unbeknown to the average person at large, the increasing market volume masked the growing fragility of the financial system, as excessive risk-taking, poor underwriting standards, and a lack of transparency in these complex financial instruments set the stage for the devastating financial crisis of 2008.

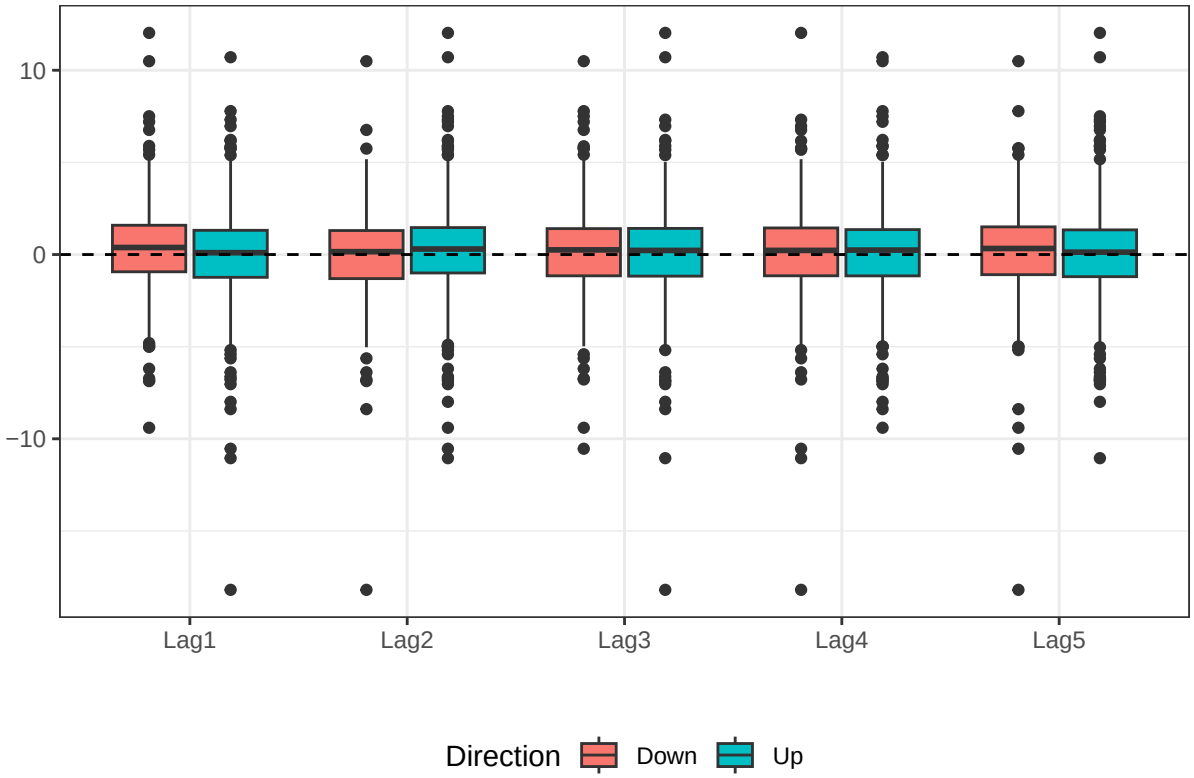

Figure 4. Boxplot of trading volume

All this brings us to the final piece of EDA in figure 4, where one can see various predictor variables and how many of the time lags correspond with upward and downward movement. The Lag2 variable is the only variable where the market closed on a weekly basis in an upward direction more than it went down. In terms of predicting whether or not the price will go up, this variable seems to be the most promising. Lag3 and lag4 do not have much of a difference between the directions. Lag1 and Lag5 look like they had a downward direction more often than having an upward direction.

3 Model Building

3.1 Initial Logistic Regression Model

As requested, an initial logistic regression model was created with direction as the response variable and the five lag variables plus the volume were used as predictors. This model was trained using the entire dataset.

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial, data = Weekly)
##
## 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
```

Figure 5. Initial logistic regression model readout

The initial model leads one to believe that lag2 is the most statistically significant predictor which coincides with the EDA shown in figure 1. One can deduce this because lag2 has the lowest p-value in the table (denoted by the column on the right under "Pr(>|z|)"). Although the p-value

for lag1 is not statistically significant (because the p-value is over 0.05, the classical threshold), it is comparably lower than the other predictors used in the model. The initial model has an AIC value of 1500.4, which means little on it's own but we can use it as a baseline for future models that will be created.

3.2 Initial Model Confusion Matrix

The confusion matrix gives a summary of the actual outcomes vs. the predicted outcomes from the model. Here's what the confusion matrix tells you:

1. **True Positive (TP):** The number of times the model correctly predicted Up when the actual direction was Up.
2. **True Negative (TN):** The number of times the model correctly predicted Down when the actual direction was Down.
3. **False Positive (FP):** The number of times the model incorrectly predicted Up when the actual direction was Down.
4. **False Negative (FN):** The number of times the model incorrectly predicted Down when the actual direction was Up.

The types of mistakes made by logistic regression can be understood from the false positives and false negatives. If FP is high, it means the model is predicting Up too often when it's actually Down. Similarly, a high FN means the model predicts Down too often when it's actually Up. Another important point is to look at the balance between FP and FN. If one is significantly higher than the other, it may indicate a bias in the model's predictions.

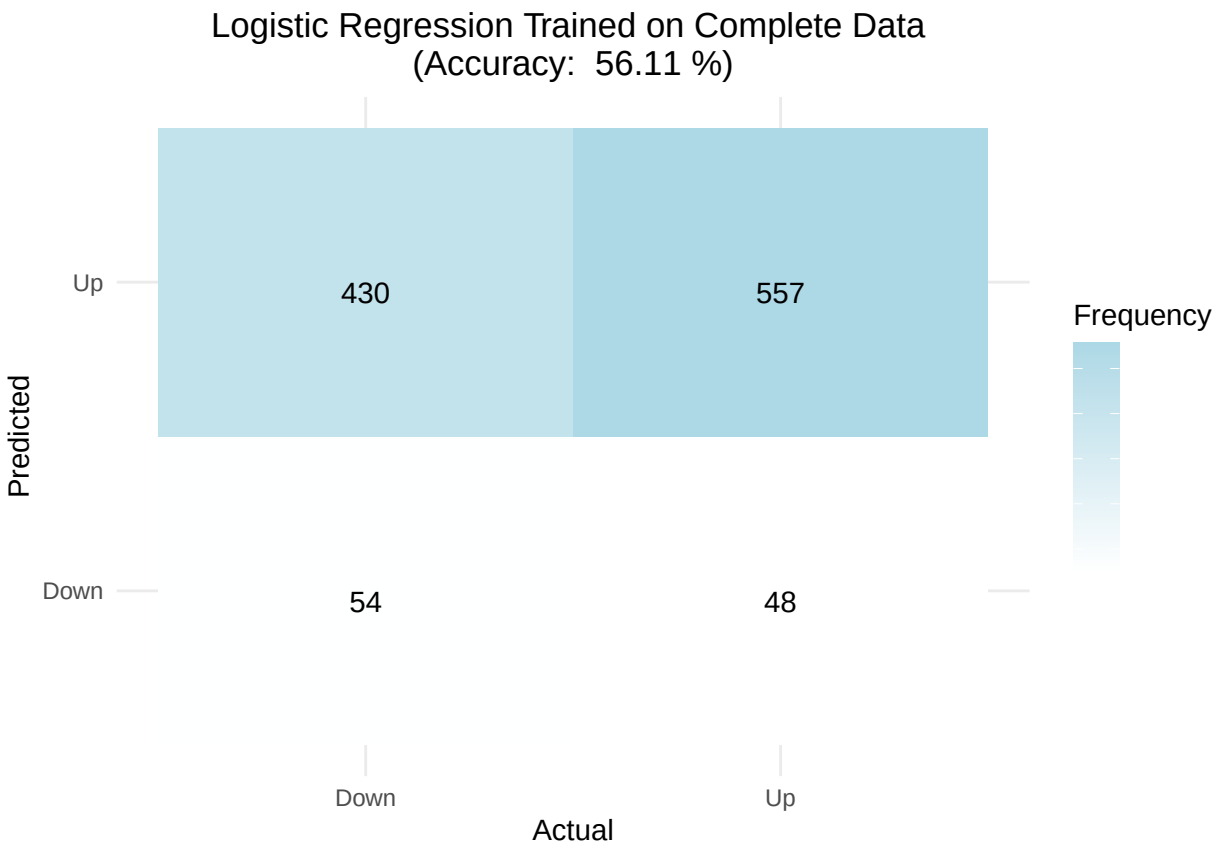

Figure 6. Initial logistic regression model readout

In the figure above, one can see the confusion matrix for the initial model created. If the predicted probability of the model was above 50%, it would be classified as up. The initial model had an accuracy of 56.11% which is not much better than flipping a coin to determine if the week will close up or down. The times the model correctly guessed up and down would be the numbers in the 1,2 and 2,1 quadrants.

4 Spliting the Data into Train and Test

Now instead of using the entire dataset to train and test, the data will now be split into a proper train/test split. The predictive models were trained on data from 1990 to 2008 and evaluated on data from 2009 and 2010. Given that the EDA and initial model showed that Lag2 was the most statistically significant, we will solely use that as a predictor because it was the only one that truly had any statistical merit in the context of the initial model.

4.1 Predictive Model Building and Comparison

The following models were created and evaluated using the same test/train split:

- Logistic regression
- LDA
- QDA
- KNN with K = 1
- Naive Bayes

The code that was used to create these models is in the appendix. Below are all the confusion matrices for each model with their prediction accuracy.

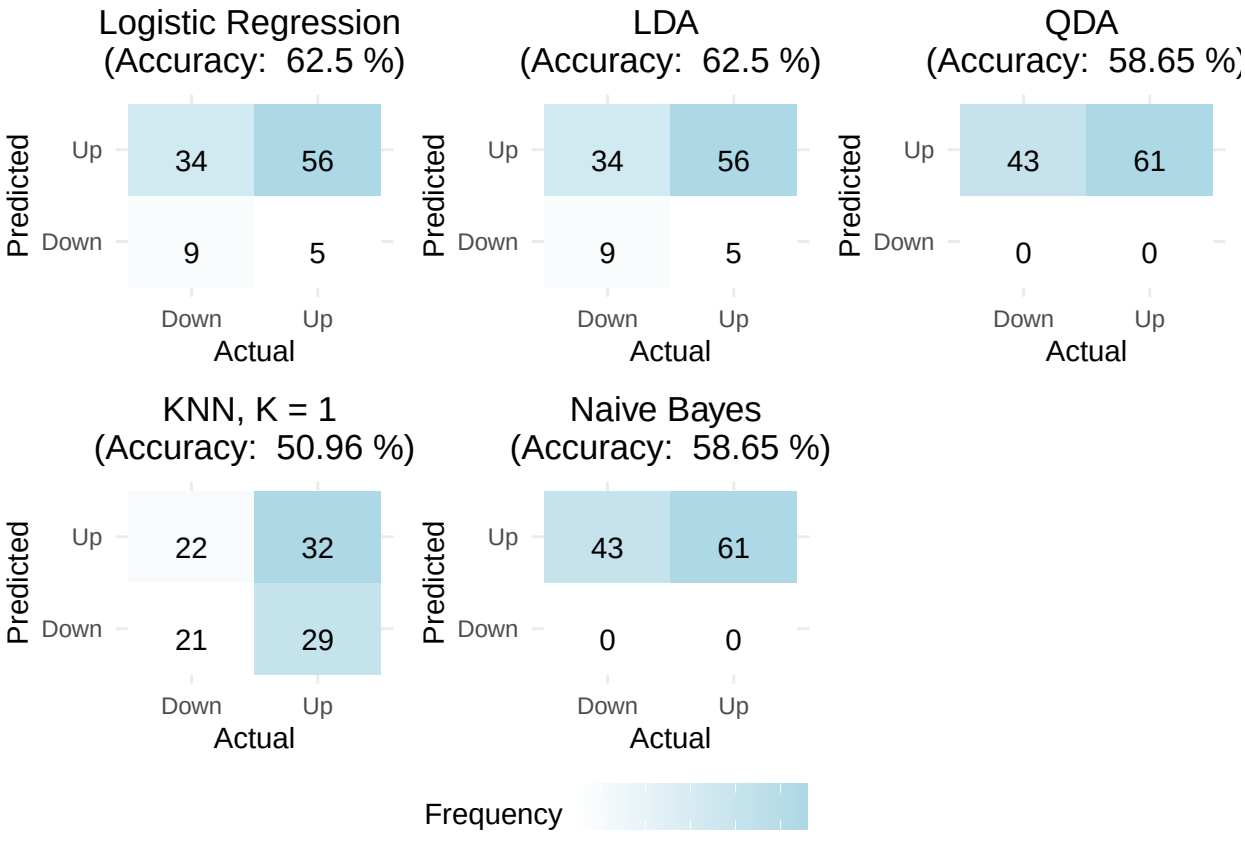

Figure 7. Confusion matrix for models

From an accuracy perspective, the worst performing model was KNN while the top performing model was a tie between logistic regression and LDA with an accuracy of 62.5%. The two best

models performed identically in performance and in the way they correctly/incorrectly classified. In addition, the logistic regression model had an AIC of 1354.543, lower than the baseline model. This would be another hint that this model will be better than the baseline.

The worst model, KNN, performed worse than the baseline model presented earlier where all the predictors were used in logistic regression. Furthermore, the Naive Bayes and QDA were unable to predict down direction in their current configuration. If this issue could be fixed, there is a chance that these models could beat out LDA and logistic regression because their accuracy is not far behind at 58.65% each when compared to LDA/logistic regression's 62.5%.

5 Additional Model Exploration

For further investigation, six additional models were created using the same train/test split.

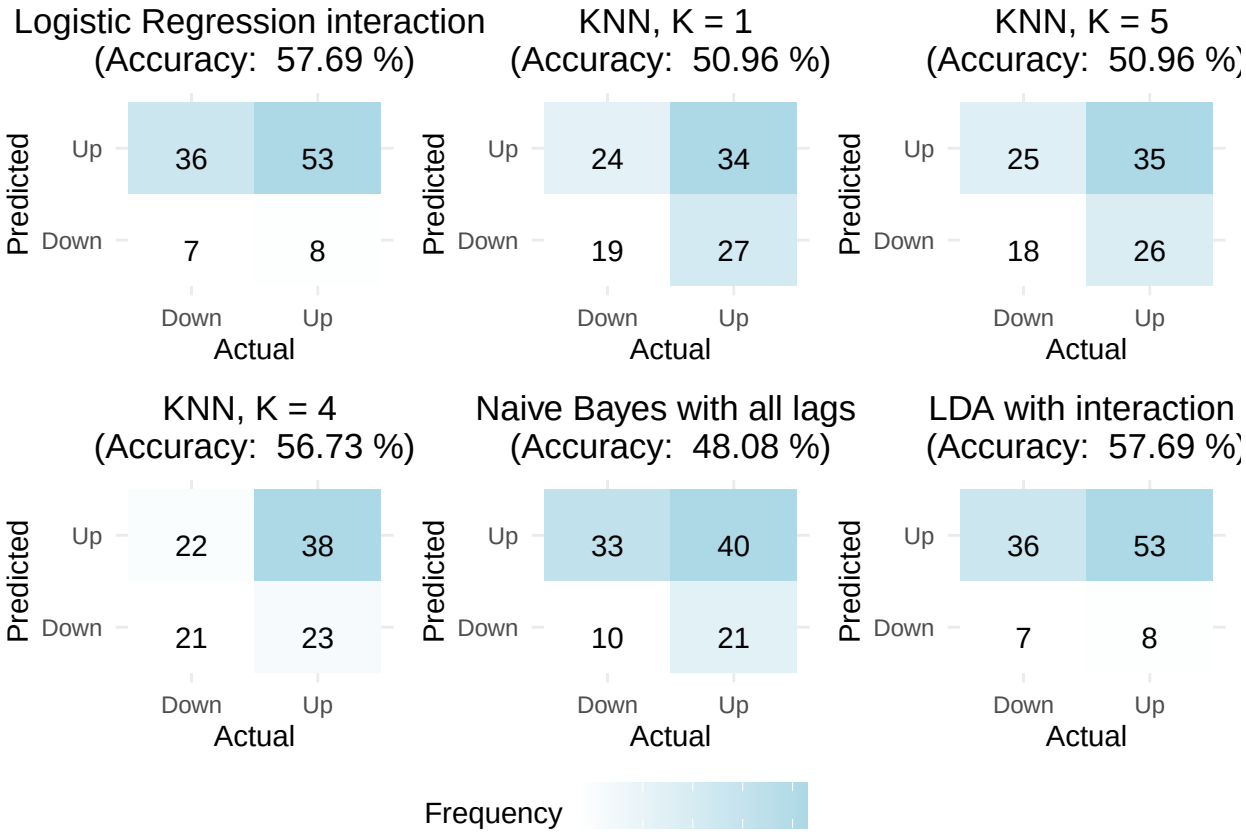

Figure 8. Confusion matrix for six additional models

The logistic regression model and LDA were taken further by having an interaction between Lag1 and L2. These models compared to the logistic regression above performed worse overall as it had about a 5% drop in accuracy. For KNN, adjusting K to 4 resulted in a 5% increase compared to

using KNN = 1. When using K = 5, the accuracy dropped back down to the same as K=1. When including all lags with naive bayes, the accuracy dropped by 10%. So overall, these models are worse then just using a predictor lag2.

6 Appendix

6.1 Part B - Initial Model Code

```
##############################################################################
# Part b
##############################################################################


logistic_model <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
                      data=Weekly, family=binomial)

summary(logistic_model)
```

6.2 Part C - Initial Confusion Matrix Code

```
##############################################################################
# Part c
##############################################################################

# Predicted probabilities
predicted_probabilities <- predict(logistic_model, type="response")

# Convert predicted probabilities to class labels
predicted_direction <- ifelse(predicted_probabilities > 0.5, "Up", "Down")

confusion_matrix <- table(Actual = Weekly$Direction,
                          Predicted = predicted_direction)
print(confusion_matrix)



accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
print(accuracy)

confusion_matrix_df <- as.data.frame(confusion_matrix)
```

```r
# Create a bar plot of the confusion matrix
log_reg_CM <- cm_ggplot(confusion_matrix_df, plt_title = 'Logistic Regression Trained on C
                        acc_perc = accuracy )
```

6.3 Part D

```r
##############################################################################
# Part d
##############################################################################

# Define training and test datasets
train_data <- subset(Weekly, Year < 2009)
test_data <- subset(Weekly, Year >= 2009)

logistic_model_train <- glm(Direction ~ Lag2, data=train_data, family=binomial)

# Getting the predicted probabilities on test data
predicted_probabilities_test <- predict(logistic_model_train,
                                        newdata=test_data, type="response")

# Convert predicted probabilities to class labels
predicted_direction_test <- ifelse(predicted_probabilities_test > 0.5, "Up", "Down")

confusion_matrix_test <- table(Actual = test_data$Direction, Predicted = predicted_directi

accuracy_test <- sum(diag(confusion_matrix_test)) / sum(confusion_matrix_test)
print(accuracy_test)

confusion_matrix_df <- as.data.frame(confusion_matrix_test)


# Create a bar plot of the confusion matrix
log_reg_CM <- cm_ggplot(confusion_matrix_df, plt_title = 'Logistic Regression', acc_perc =
```

```r
print(confusion_matrix_test)
```

6.4 Part E

```r
################################################################################
# Part e
################################################################################

# Now we apply with LDA

lda_model <- lda(Direction ~ Lag2, data=train_data)

lda_predictions <- predict(lda_model, test_data)

confusion_matrix_lda <- table(Actual = test_data$Direction,
                              Predicted = lda_predictions$class)
print(confusion_matrix_lda)

accuracy_lda <- sum(diag(confusion_matrix_lda)) / sum(confusion_matrix_lda)

print(accuracy_lda)


confusion_matrix_df <- as.data.frame(confusion_matrix_lda)

# Create a bar plot of the confusion matrix
LDA_CM <- cm_ggplot(confusion_matrix_df, plt_title = 'LDA', acc_perc = accuracy_lda )
```

6.5 Part F

```r
################################################################################
# Part f
################################################################################
```

```r
qda_model <- qda(Direction ~ Lag2, data=train_data)

qda_predictions <- predict(qda_model, test_data)

confusion_matrix_qda <- table(Actual = test_data$Direction,
                              Predicted = qda_predictions$class)
print(confusion_matrix_qda)

accuracy_qda <- sum(diag(confusion_matrix_qda)) / sum(confusion_matrix_qda)

print(accuracy_qda)



confusion_matrix_df <- as.data.frame(confusion_matrix_qda)

# Create a bar plot of the confusion matrix
QDA_CM <-  cm_ggplot(confusion_matrix_df, plt_title = 'QDA', acc_perc = accuracy_qda )
```

6.6 Part G

```r
##############################################################################
# Part g
##############################################################################

train_x <- as.matrix(train_data$Lag2)
test_x <- as.matrix(test_data$Lag2)
train_y <- train_data$Direction

set.seed(555)  # For reproducibility
knn_predictions <- knn(train_x, test_x, train_y, k=1)

confusion_matrix_knn <- table(Actual = test_data$Direction,
                              Predicted = knn_predictions)
print(confusion_matrix_knn)
```

```r
accuracy_knn <- sum(diag(confusion_matrix_knn)) / sum(confusion_matrix_knn)

print(accuracy_knn)


confusion_matrix_df <- as.data.frame(confusion_matrix_knn)

# Create a bar plot of the confusion matrix
KNN_CM <- cm_ggplot(confusion_matrix_df, plt_title = 'KNN, K = 1', acc_perc = accuracy_knn
```

6.7 Part H

```r
################################################################################
# Part h
################################################################################

nb_model <- naiveBayes(Direction ~ Lag2, data=train_data)

nb_predictions <- predict(nb_model, test_data[, "Lag2", drop=FALSE])

confusion_matrix_nb <- table(Actual = test_data$Direction,
                             Predicted = nb_predictions)
print(confusion_matrix_nb)

accuracy_nb <- sum(diag(confusion_matrix_nb)) / sum(confusion_matrix_nb)

print(accuracy_nb)

confusion_matrix_df <- as.data.frame(confusion_matrix_nb)

# Create a bar plot of the confusion matrix
NB_CM <- cm_ggplot(confusion_matrix_df, plt_title = 'Naive Bayes', acc_perc = accuracy_nb)
```

6.8 Part J

```
# Model 1
# Implementing logistic regression with an interaction between Lag1 and Lag2
# Lag1:Lag2 is the same as Lag1 * Lag2 multiplication of the predictors
log_model_interaction_train <- glm(Direction ~ Lag1 + Lag2 + Lag1:Lag2,
                                   data=train_data, family=binomial)

# Predicting on the test set
logistic_preds_interaction <- predict(log_model_interaction_train,
                                      newdata=test_data, type="response")

# Converting predictions into "Up" or "Down" based on a threshold of 0.5
log_preds_interaction <- ifelse(logistic_preds_interaction > 0.5, "Up", "Down")

confusion_matrix_test <- table(Actual = test_data$Direction,
                               Predicted = log_preds_interaction)

accuracy_test <- sum(diag(confusion_matrix_test)) / sum(confusion_matrix_test)

confusion_matrix_df <- as.data.frame(confusion_matrix_test)


# Create a bar plot of the confusion matrix
log_reg_interact_CM <- cm_ggplot(confusion_matrix_df,
                                   plt_title = 'Logistic Regression interaction',
                                   acc_perc = accuracy_test )


# Model 2
# Using the first three lags as predictors
train_x_multi <- as.matrix(train_data[,c("Lag1", "Lag2", "Lag3")])
test_x_multi <- as.matrix(test_data[,c("Lag1", "Lag2", "Lag3")])

# Predicting using KNN with k=3
knn_predictions_multi <- knn(train_x_multi, test_x_multi, train_y, k=3)
```

```r
confusion_matrix_knn <- table(Actual = test_data$Direction,
                              Predicted = knn_predictions_multi)

accuracy_knn <- sum(diag(confusion_matrix_knn)) / sum(confusion_matrix_knn)
confusion_matrix_df <- as.data.frame(confusion_matrix_knn)

# Create a bar plot of the confusion matrix
KNN_CM_multi <- cm_ggplot(confusion_matrix_df, plt_title = 'KNN, K = 1',
                          acc_perc = accuracy_knn)

# Model 3
# Using Lag4, Lag5, and Volume as predictors
train_x_knn2 <- as.matrix(train_data[,c("Lag4", "Lag5", "Volume")])
test_x_knn2 <- as.matrix(test_data[,c("Lag4", "Lag5", "Volume")])

# Predicting using KNN with k=5
knn_predictions_knn2 <- knn(train_x_knn2, test_x_knn2, train_y, k=5)

confusion_matrix_knn2 <- table(Actual = test_data$Direction,
                               Predicted = knn_predictions_knn2)

accuracy_knn2 <- sum(diag(confusion_matrix_knn2)) / sum(confusion_matrix_knn2)
confusion_matrix_df_knn2 <- as.data.frame(confusion_matrix_knn2)

# Create a bar plot of the confusion matrix
KNN2_CM_multi <- cm_ggplot(confusion_matrix_df_knn2, plt_title = 'KNN, K = 5',
                           acc_perc = accuracy_knn2)

# Using only Lag2 as the predictor
train_x_knn3 <- as.matrix(train_data[,"Lag2"])
test_x_knn3 <- as.matrix(test_data[,"Lag2"])

# Predicting using KNN with k=4
knn_predictions_knn3 <- knn(train_x_knn3, test_x_knn3, train_y, k=4)

confusion_matrix_knn3 <- table(Actual = test_data$Direction,
                               Predicted = knn_predictions_knn3)
```

```r
accuracy_knn3 <- sum(diag(confusion_matrix_knn3)) / sum(confusion_matrix_knn3)
confusion_matrix_df_knn3 <- as.data.frame(confusion_matrix_knn3)

# Create a bar plot of the confusion matrix
KNN3_CM_multi <- cm_ggplot(confusion_matrix_df_knn3, plt_title = 'KNN, K = 4',
                           acc_perc = accuracy_knn3)



# Model 5
# Implementing Naive Bayes using all the lag variables as predictors
naive_bayes_model <- naiveBayes(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5,
                                data=train_data)

# Predicting on the test set
naive_bayes_predictions <- predict(naive_bayes_model, test_data)

#nb_predictions <- predict(nb_model, test_data[, "Lag2", drop=FALSE])

confusion_matrix_nb <- table(Actual = test_data$Direction,
                             Predicted = naive_bayes_predictions)

accuracy_nb <- sum(diag(confusion_matrix_nb)) / sum(confusion_matrix_nb)

confusion_matrix_df <- as.data.frame(confusion_matrix_nb)

# Create a bar plot of the confusion matrix
NB_CM <- cm_ggplot(confusion_matrix_df, plt_title = 'Naive Bayes with all lags',
                   acc_perc = accuracy_nb)

# Model 6
# Implementing LDA using Lag1, Lag2 and their interaction
lda_model <- lda(Direction ~ Lag1 + Lag2 + Lag1:Lag2, data=train_data)

# Predicting on the test set

lda_pred_interaction <- predict(lda_model, test_data)
```

```r
confusion_matrix_lda <- table(Actual = test_data$Direction,
                              Predicted = lda_pred_interaction$class)

accuracy_lda <- sum(diag(confusion_matrix_lda)) / sum(confusion_matrix_lda)

confusion_matrix_df <- as.data.frame(confusion_matrix_lda)

# Create a bar plot of the confusion matrix
LDA_CM_interaction <- cm_ggplot(confusion_matrix_df,
                                plt_title = 'LDA with interaction',
                                acc_perc = accuracy_lda )



# combine all confusion matrices into a pretty ggplot
ggarrange(
  log_reg_interact_CM, KNN_CM_multi, KNN2_CM_multi,
  KNN3_CM_multi, NB_CM, LDA_CM_interaction,
  ncol = 3,
  nrow = 2,
  common.legend = TRUE,
  legend = 'bottom'
)
```