Logistic Regression, LDA, QDA, kNN Analyses

Maureen Renaud

7/3/2020

INTRODUCTION

There are entire industries devoted to predicting the stock market, scouring statistical models for the slightest gain in prediction capability. This study aims to test several models and assess their ability to predict the stock market. These models are Logistic Regression, Linear Discriminant Analysis, Quadratic Discriminant Analysis, and K-Nearest Neighbors.

For this data set, the Logistic Regression and Linear Discriminant Analysis models using only Lag2 as a predictor were the most accurate, each with a 62.5% accuracy rate.

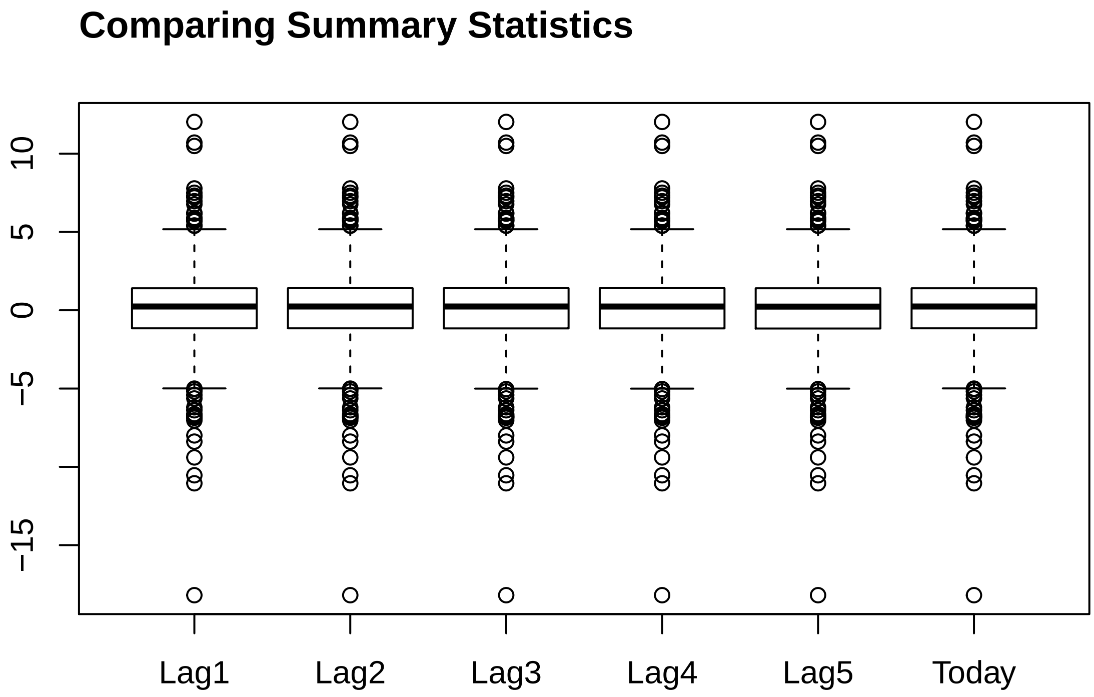
While this does not sound overwhelmingly high, when dealing with the stock market, an accuracy rate of 62.5% is fairly significant and could have significant financial implications for someone making trades based on these models.

DATA

The data for this study came from the the Weekly dataset in the ISLR library. It contains 1089 weekly returns for 21 years, from 1990 to 2010. According to write-up on the ISLR package[1], these values are raw values of the S&P 500 that were obtained from Yahoo Finance and then converted to percentages and lagged. The data set is complete and there is no missing data.

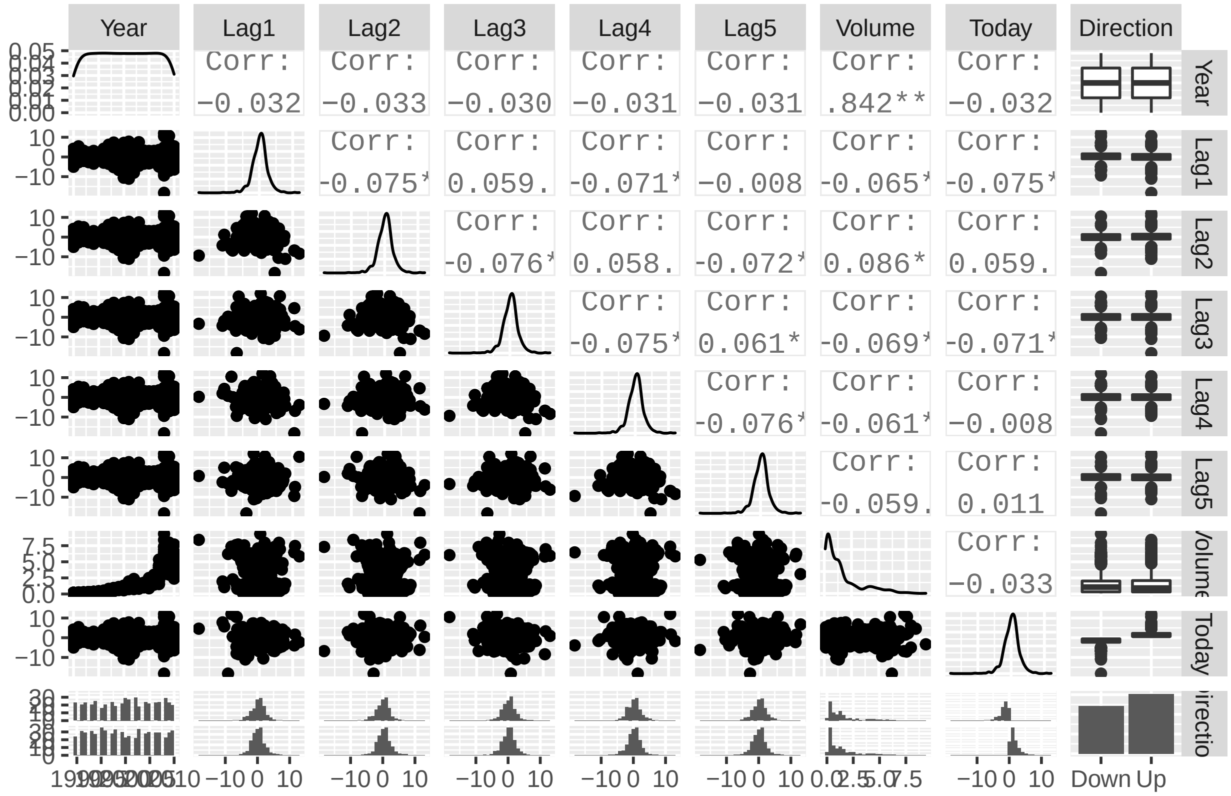
In order to get a better feel for this data, the summary is provided in Figure 1:

The boxplots below focus on the summary statistics of the lag and today predictors:



As expected these are nearly identifical, since the majority of the data points are the same. This is because the return in Lag1 will be the Lag2 return for the next week and the Lag3 return the week after that.

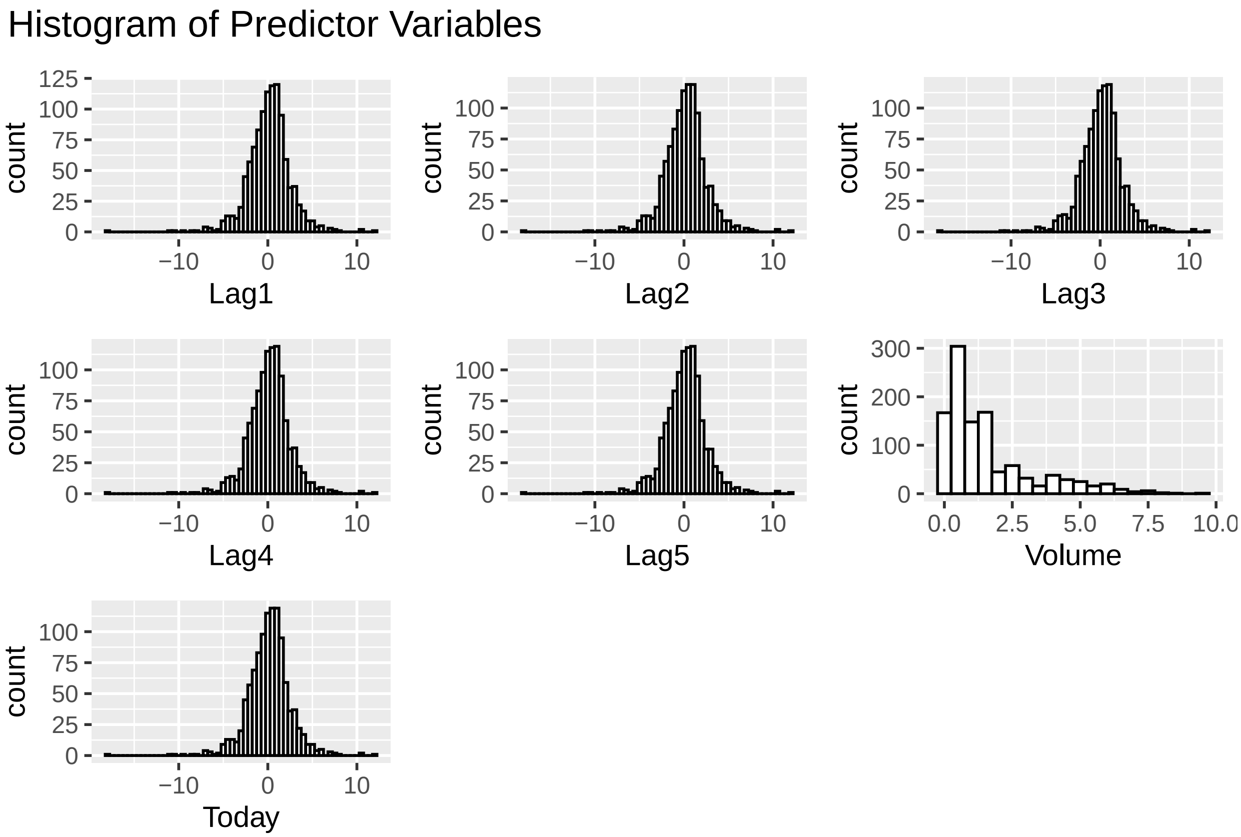
The next figure focuses on the correlations between the predictors:



Correlation Between Predictors in the Weekly Data Set

There does not appear to be strong correlations between any of the predictors, with the exception of Volume. Over time, volume has increased in what appears to be an exponential manner. The correlation between Year and Volume is 0.842.

The below figure focuses on the distribution of each of these variables. This is important to determine if any transformations are appropriate.



It appears that each of the Lag predictors are relatively normally distributed. However, Volume is right-skewed. Later on, this may be an appropriate variable to transform. Some of the models require a normal distribution, but some do not. This will be important to focus on when we reach those models.

ANALYSES

Models based on Logistic Regression, Linear Discriminant Analysis, Quadratic Discriminant Analysis, and k Nearest Neighbors will now be analyzed and compared.

LOGISTIC REGRESSION

According to *An Introduction to Statistical Learning*[2], a logistic regression model is appropriate when the response variable falls into one of two categories. For this data, the response is either "Up" or "Down" based on the direction of the market.
Additionally, logistic regression results in a linear decision, so it is ideal for models where the predictors have weak or no correlations. We saw in our data exploration, that there are only very weak correlations between most of the predictors in this data set. Finally, a logistic regression model does not assume a normal disribution like Linear Discriminant Analysis does. This means there is no need to be concerned about the skewness detected in the Volume predictor.

Below is a logistic regression with Direction as the response and the five lag variables plus Volume as predictors.

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

Looking at the output of the logistic regression, the only predictor that appears to be statistically significant at a 0.05 significance level, is Lag 2. Lag1 is close with a p-value of 0.1181. I find it interesting that percentage return from two weeks previous was a better predictor than percentage return from the previous week.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down  Up
##       Down   54  48
##       Up    430 557
##
##                Accuracy : 0.5611
##                  95% CI : (0.531, 0.5908)
##     No Information Rate : 0.5556
##     P-Value [Acc > NIR] : 0.369
##
##                   Kappa : 0.035
##
##  Mcnemar’s Test P-Value : <2e-16
##
##             Sensitivity : 0.9207
##             Specificity : 0.1116
##          Pos Pred Value : 0.5643
##          Neg Pred Value : 0.5294
##              Prevalence : 0.5556
##          Detection Rate : 0.5115
##    Detection Prevalence : 0.9063
##       Balanced Accuracy : 0.5161
##
##        ’Positive’ Class : Up
##
```

The confusion matrix, with a sensitivity of 92% indicates that this model does a good job of predicting when the market will go up. However, with a specificity of 11%, it does a poor job of predicting when the market will go down.

On average 55.55% of the returns go up. This model has a 56% accuracy rate, meaning it is only slightly better than just guessing that the market will always go up.

To truly assess the prediction capability of this model, I will fit the logistic regression model using a training data period from 1990 to 2008. Since Lag2 was the only statistically significant predictor, that is the only variable used in this model.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down    9  5
##       Up     34 56
##
##                Accuracy : 0.625
##                  95% CI : (0.5247, 0.718)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.2439
##
##                   Kappa : 0.1414
##
##  Mcnemar’s Test P-Value : 7.34e-06
##
##             Sensitivity : 0.9180
##             Specificity : 0.2093
##          Pos Pred Value : 0.6222
##          Neg Pred Value : 0.6429
##              Prevalence : 0.5865
##          Detection Rate : 0.5385
##    Detection Prevalence : 0.8654
##       Balanced Accuracy : 0.5637
##
##        ’Positive’ Class : Up
##
```

It is interesting to note that the sensitivity (ability of the model to accurately predict when the market goes up) only decreased slightly to 91.8%. However, the specificity (ability of the market to accurately predict when the market goes down), increased to 20.9% Overall, the fraction of correct predictions for the held out data, given in the output, is 62.5%. In the test set, 58.7% of the returns go up. This model is an improvement on guessing the market will always go up, so there may be some value in it.

LINEAR DISCRIMINANT ANALYSIS

Linear Discriminant Analysis, according to *An Introduction to Statistical Learning*[2], is an improvement upon Logistic Regression when the classes are well separated and if the distribution of predictors is approximately normal when n is small.

In this data set, Volume is not normally distributed. However, n is not particularly small and we are only using Lag2 as a predictor, so it is still an appropriate model for the data.

Below is the linear discriminant analysis model using the training data period from 1990 to 2008, with Lag2 as the only predictor:

```
## 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
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down    9  5
##       Up     34 56
##
##                Accuracy : 0.625
##                  95% CI : (0.5247, 0.718)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.2439
##
##                   Kappa : 0.1414
##
##  Mcnemar’s Test P-Value : 7.34e-06
##
##             Sensitivity : 0.9180
##             Specificity : 0.2093
##          Pos Pred Value : 0.6222
##          Neg Pred Value : 0.6429
##              Prevalence : 0.5865
##          Detection Rate : 0.5385
##    Detection Prevalence : 0.8654
##       Balanced Accuracy : 0.5637
##
##        ’Positive’ Class : Up
##
```

Interestingly enough, these are the exact same values obtained by running the Logistic Regression. This makes sense because they are similar models. As mentioned above, the main reasons we would choose a Linear Discriminant Analysis model over Logistic Regression are not issues for our data and the predictors chosen. So, it is logical that the output be similar.

QUADRATIC DISCRIMINANT ANALYSIS

Now, we can examine Quadratic Disciminant Analysis (QDA). According to *An Introduction to Statistical Learning*^{2}, QDA assumes that the observations from each class are normally distributed. The result comes from plugging estimates for the parameters into Bayes' theorem.
Additionaly, QDA assumes that each class has its own covariance matrix. This is in opposition to LDA, which assumes that all classes have the same covariance matrix. This means that LDA is much less flexible with lower variance. If the assumption that the classes share a common covariance matrix is correct, LDA may be a better predictor. However, if the assumption is not met, then LDA will suffer from bias and will no longer be an ideal predictor.

In general, QDA is the recommended model if the data set is very large or if it is clear all classes do not share a covariance matrix. When summarizing the data, we noted that the Lag variables and Today variable contained nearly the exact same data. Given that, it is likely safe to assume a common covariance matrix. This means that we will likely see that the LDA model is a better predictor than QDA.

```
## 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
```

The confusion matrix for the test set:

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down    0  0
##       Up     43 61
##
##                Accuracy : 0.5865
##                  95% CI : (0.4858, 0.6823)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.5419
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 1.504e-10
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.5865
##          Neg Pred Value :    NaN
##              Prevalence : 0.5865
##          Detection Rate : 0.5865
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
```

```
##
##        ’Positive’ Class : Up
##
```

This model actually has perfect specificity, since it predicted the market will always go up. This means the sensitivity, at 0%, is very poor. Because the model never predicted the market could go down, it has zero accurate predictions. Looking at the overall fraction of correct predictions, we see that it is only 58.7%, which as mentioned earlier, is also the accuracy rate for always predicting the market will go up.

In addition to have a lower accuracy rate than the Logistic Regression and LDA models, it is clear this is an inappropriate model for this particular data set.

K-NEAREST NEIGHBORS

Finally, we examine K-Nearest Neighbors. According to *An Introduction to Statistical Learning*[2], K-Nearest Neighbors is non-parametric and is the best choice when the decision boundry is non-linear.

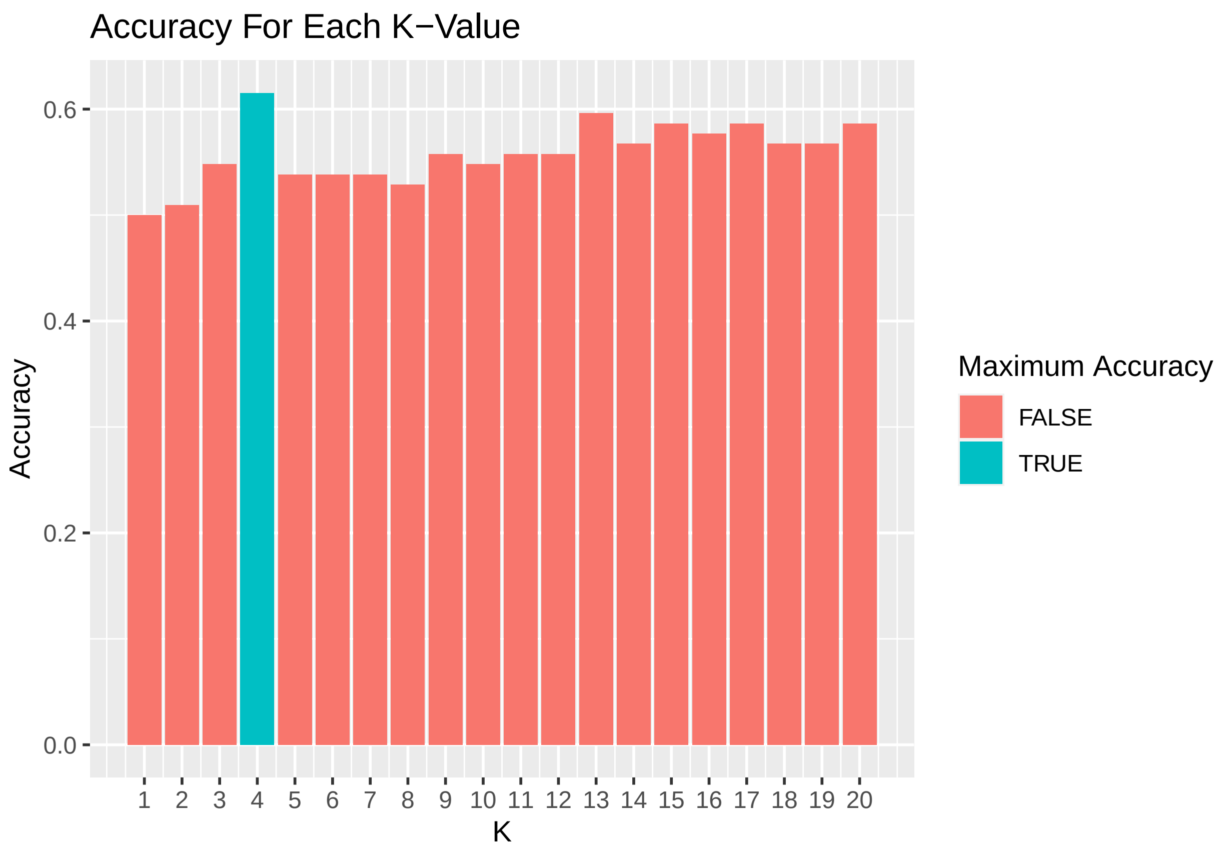
To assess this model, we will first examine the confusion matrix when k = 1

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down   21 30
##       Up     22 31
##
##                Accuracy : 0.5
##                  95% CI : (0.4003, 0.5997)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.9700
##
##                   Kappa : -0.0033
##
##  Mcnemar’s Test P-Value : 0.3317
##
##             Sensitivity : 0.5082
##             Specificity : 0.4884
##          Pos Pred Value : 0.5849
##          Neg Pred Value : 0.4118
##              Prevalence : 0.5865
##          Detection Rate : 0.2981
##    Detection Prevalence : 0.5096
##       Balanced Accuracy : 0.4983
##
##        ’Positive’ Class : Up
##
```

We see that the sensitivity is 50.8% and the specificity is 48.8% with an overall accuracy of 50%.

The predictions for this model are about as good as guessing and worse than just guessing the market will always go down. However, k = 1 is typically not the best value for k-nearest neighbors, so it is necessary to compare other k-values to determine which leads to the highest accuracy.

Below, is a table with the accuracy reading for each value of k:



Maximum accuracy occurs when k = 4. Below is the confusion matrix for K-Nearest Neighbors when k = 4:

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down   20 17
##       Up     23 44
##
##                Accuracy : 0.6154
##                  95% CI : (0.5149, 0.7091)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.3110
##
##                   Kappa : 0.1903
##
##  Mcnemar’s Test P-Value : 0.4292
##
##             Sensitivity : 0.7213
##             Specificity : 0.4651
##          Pos Pred Value : 0.6567
##          Neg Pred Value : 0.5405
##              Prevalence : 0.5865
##          Detection Rate : 0.4231
##    Detection Prevalence : 0.6442
##       Balanced Accuracy : 0.5932
```

```
##
##        ’Positive’ Class : Up
##
```

Using an appropriate value for K gives us an accuracy of 61.5%, which is nearly as good as the accuracy for Logistic Regression, and LDA. The sensitivity is slightly lower (than Logistic Regression and LDA) but the specificity is higher.

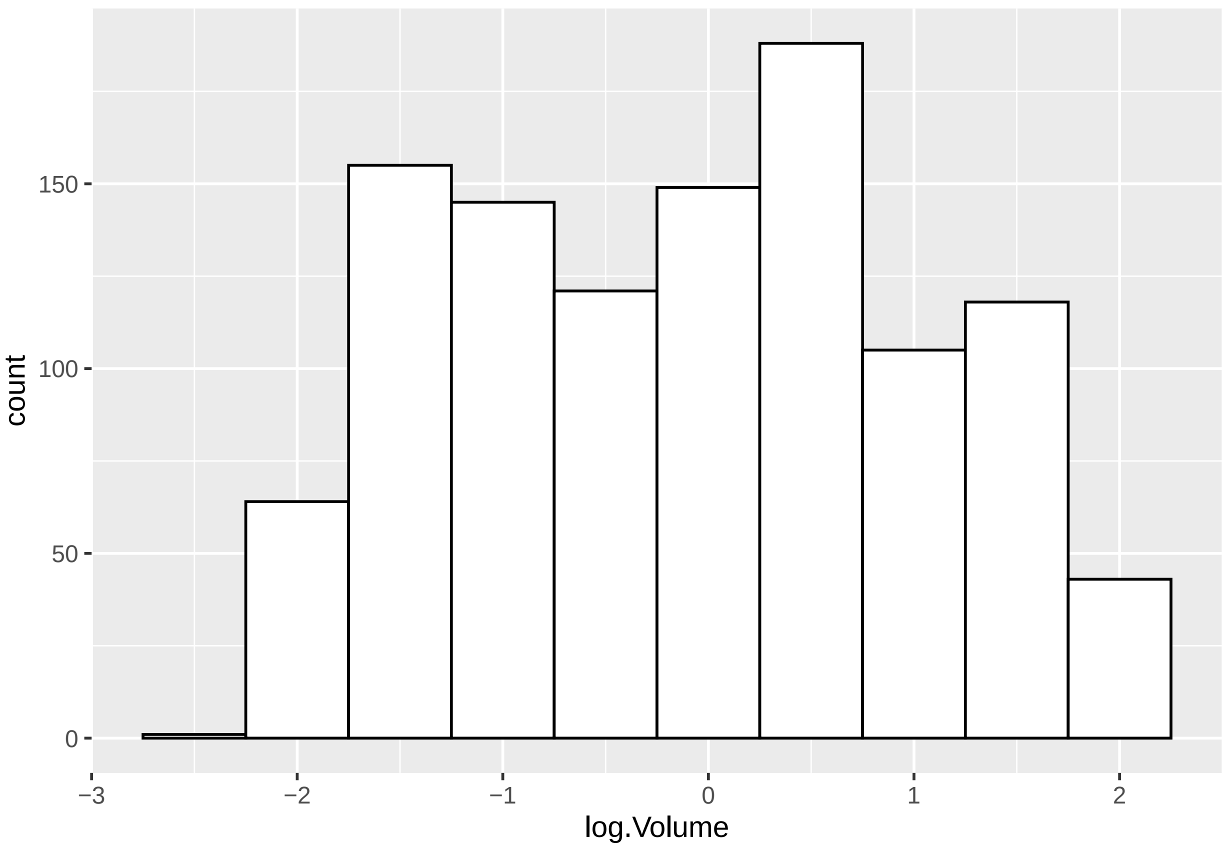
BEST RESULTS

Based on the model outputs, we see that Logistic Regression and LDA have the best predictive capability. QDA and K-Nearest Neighbors did not perform as well.
It is likely that there is a common covariance matrix rather than each class having its own. Because of this QDA has a higher variance without a low enough bias to offset it. This makes QDA an inappropriate model for this data set.

Regarding K-Nearest Neighbors, we are not violating assumptions by choosing the model, it is just not as accurate as the logistic regression and LDA models. While this model performed very poorly on the data using k = 1, we saw a solid improvement when using k = 4, making it nearly as accurate as logistic regression and LDA. If the data were in stronger need of a non-parametric model, k-Nearest Neighbors may have outperformed Logistic Regression and LDA. However, because Logistic Regression and LDA were well-suited to this data, they performed the best.

EXPERIMENTATION

It was noted in the summary section that the predictor Volume was not normally disributed. This did not play a role in the results above, because Volume was not a significant predictor of Direction. However, I will now attempt to transform Volume to see if that impacts the significance of the predictor.



Taking the log of Volume eliminates the skewness and gives us a more normally distributed predictor. Rerunning the logistic regression with all of the predictors give us:

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     log.Volume, family = binomial, data = Weekly)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6922  -1.2600   0.9928   1.0847   1.4665
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.22562    0.06224   3.625 0.000289 ***
## Lag1        -0.04127    0.02637  -1.565 0.117578
## Lag2         0.05834    0.02679   2.178 0.029433 *
## Lag3        -0.01607    0.02663  -0.603 0.546213
## Lag4        -0.02790    0.02643  -1.055 0.291218
## Lag5        -0.01457    0.02636  -0.553 0.580433
## log.Volume  -0.05133    0.05607  -0.915 0.359988
## ---
## 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: 1485.9  on 1082  degrees of freedom
## AIC: 1499.9
##
## Number of Fisher Scoring iterations: 4
```

While the p-value for the log of Volume has decreased, it is still not a statistically significant predictor. Thus, there is no need to rerun the other models with the log of Volume as a predictor.

Now, I will explore if there are interactions between any of the variables:

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     log.Volume + Lag1 * Lag2 + Lag1 * Lag3 + Lag1 * Lag4 + Lag1 *
##     Lag5 + Lag1 * log.Volume + Lag2 * Lag3 + Lag2 * Lag4 + Lag2 *
##     Lag5 + Lag2 * log.Volume + Lag3 * Lag4 + Lag3 * Lag5 + Lag3 *
##     log.Volume + Lag4 * Lag5 + Lag4 * log.Volume, family = binomial,
##     data = Weekly)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6024  -1.2537   0.9713   1.0768   1.8268
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.239605   0.064377   3.722 0.000198 ***
## Lag1            -0.041028   0.030059  -1.365 0.172283
## Lag2             0.052345   0.030302   1.727 0.084087 .
## Lag3            -0.013426   0.030270  -0.444 0.657369
## Lag4            -0.036354   0.030393  -1.196 0.231642
## Lag5            -0.006021   0.029160  -0.206 0.836417
## log.Volume      -0.057567   0.057165  -1.007 0.313914
## Lag1:Lag2        0.006467   0.008366   0.773 0.439532
## Lag1:Lag3        0.009133   0.009579   0.953 0.340360
## Lag1:Lag4       -0.007981   0.008582  -0.930 0.352413
## Lag1:Lag5       -0.013529   0.010168  -1.331 0.183348
## Lag1:log.Volume  0.016962   0.026148   0.649 0.516541
## Lag2:Lag3        0.006275   0.008175   0.768 0.442717
## Lag2:Lag4       -0.001931   0.009384  -0.206 0.836990
## Lag2:Lag5       -0.003171   0.008673  -0.366 0.714654
## Lag2:log.Volume  0.026042   0.026494   0.983 0.325641
## Lag3:Lag4        0.017339   0.008320   2.084 0.037156 *
## Lag3:Lag5       -0.007047   0.009105  -0.774 0.438931
## Lag3:log.Volume  0.014307   0.026431   0.541 0.588297
## Lag4:Lag5       -0.009689   0.008475  -1.143 0.252933
## Lag4:log.Volume -0.001229   0.026230  -0.047 0.962623
## ---
## 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: 1474.3  on 1068  degrees of freedom
## AIC: 1516.3
##
## Number of Fisher Scoring iterations: 4
```

In this model, Lag2 is no longer statistically significant. Instead, the Interaction of Lag3 and Lag4 appears to be significant, with a p-value of .037. So, I will run a model with each of these variables and their interaction.

```
##
## Call:
## glm(formula = Direction ~ Lag3 + Lag4 + Lag3 * Lag4, family = binomial,
##     data = Weekly, subset = train)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.480  -1.267   1.063   1.089   1.559
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.219078   0.064604   3.391 0.000696 ***
## Lag3        -0.003595   0.030470  -0.118 0.906069
## Lag4        -0.019086   0.029080  -0.656 0.511597
## Lag3:Lag4    0.011892   0.007954   1.495 0.134900
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1351.6  on 981  degrees of freedom
## AIC: 1359.6
##
## Number of Fisher Scoring iterations: 3
```

Minimizing the data down to just Lag3 and Lag4 and thier interaction no longer shows any statistical significance. In the original model, Lag1 came the closest to being significant, so I will examine a model with Lag1, Lag2, and their interaction:

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag1 * Lag2, family = binomial,
##     data = Weekly, subset = train)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.573  -1.259   1.003   1.086   1.596
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.211419   0.064589   3.273  0.00106 **
## Lag1        -0.051505   0.030727  -1.676  0.09370 .
## Lag2         0.053471   0.029193   1.832  0.06700 .
## Lag1:Lag2    0.001921   0.007460   0.257  0.79680
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1346.9  on 981  degrees of freedom
## AIC: 1354.9
##
## Number of Fisher Scoring iterations: 4
```

This no longer show statistical significance in any of the predictors either. However, out of curiosity, I will run some of our models to see if there any major differences in accuracy.

LOGISTIC REGRESSION

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down    7  8
##       Up     36 53
##
##                Accuracy : 0.5769
##                  95% CI : (0.4761, 0.6732)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.6193
##
##                   Kappa : 0.035
##
##  Mcnemar's Test P-Value : 4.693e-05
##
##             Sensitivity : 0.8689
##             Specificity : 0.1628
##          Pos Pred Value : 0.5955
##          Neg Pred Value : 0.4667
##              Prevalence : 0.5865
##          Detection Rate : 0.5096
##    Detection Prevalence : 0.8558
##       Balanced Accuracy : 0.5158
##
##        'Positive' Class : Up
##
```

The accuracy of this model is only 57.7%, which is worse than using only Lag2 as a predictor and worse than just guessing the market will always go up.

LDA

```
## Call:
## lda(Direction ~ Lag1 + Lag2 + Lag1 * Lag2, data = Weekly, subset = train)
##
## Prior probabilities of groups:
```

```
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##               Lag1        Lag2  Lag1:Lag2
## Down  0.289444444 -0.03568254 -0.8014495
## Up   -0.009213235  0.26036581 -0.1393632
##
## Coefficients of linear discriminants:
##                    LD1
## Lag1      -0.285484602
## Lag2       0.295080109
## Lag1:Lag2  0.009629381
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down    7  8
##       Up     36 53
##
##                Accuracy : 0.5769
##                  95% CI : (0.4761, 0.6732)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.6193
##
##                   Kappa : 0.035
##
##  Mcnemar's Test P-Value : 4.693e-05
##
##             Sensitivity : 0.8689
##             Specificity : 0.1628
##          Pos Pred Value : 0.5955
##          Neg Pred Value : 0.4667
##              Prevalence : 0.5865
##          Detection Rate : 0.5096
##    Detection Prevalence : 0.8558
##       Balanced Accuracy : 0.5158
##
##        'Positive' Class : Up
##
```

Like before, when using Lag2 as a predictor, the performance of the LDA model was identical to the performance of the Logistic Regression model. And as we saw above, with an accuracy of 57.7%, this model performed worse than the models using only Lag2.

QDA

```
## Call:
## qda(Direction ~ Lag1 + Lag2 + Lag1 * Lag2, data = Weekly, subset = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##              Lag1        Lag2  Lag1:Lag2
## Down  0.289444444 -0.03568254 -0.8014495
## Up   -0.009213235  0.26036581 -0.1393632
```

Examining confusion matrix for the test set:

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down   23 36
##       Up     20 25
##
##                Accuracy : 0.4615
##                  95% CI : (0.3633, 0.562)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.99616
##
##                   Kappa : -0.0524
##
##  Mcnemar's Test P-Value : 0.04502
##
##             Sensitivity : 0.4098
##             Specificity : 0.5349
##          Pos Pred Value : 0.5556
##          Neg Pred Value : 0.3898
##              Prevalence : 0.5865
##          Detection Rate : 0.2404
##    Detection Prevalence : 0.4327
##       Balanced Accuracy : 0.4724
##
##        'Positive' Class : Up
##
```

As was clear in the first part of this study, Quadratic Discriminant Analysis continues to be a poor model choice for this data. Not only does this model do worse than the QDA model using Lag2, but it also does worse than just guessing the model will always go up or just guessing randomly.

K-Nearest Neighbors %nbsp;

Using k = 4 again gives the following output:

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down   26 27
##       Up     17 34
##
##                Accuracy : 0.5769
##                  95% CI : (0.4761, 0.6732)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.6193
##
##                   Kappa : 0.1567
##
##  Mcnemar’s Test P-Value : 0.1748
##
##             Sensitivity : 0.5574
##             Specificity : 0.6047
##          Pos Pred Value : 0.6667
##          Neg Pred Value : 0.4906
##              Prevalence : 0.5865
##          Detection Rate : 0.3269
##    Detection Prevalence : 0.4904
##       Balanced Accuracy : 0.5810
##
##        ’Positive’ Class : Up
##
```

Once again, this model does worse than the k-nearest neighbors model using Lag2. It is interesting that in this case, k-nearest neighbors performs equally to logistic regression and LDA.

Overall, using a model built with Lag2 is the best choice. None of the models with Lag1 and Lag2 as predictors had a better accuracy rate. By using the model with only Lag2, we will have higher accuracy and a simpler model.

One final aspect I want to analyze is using a cut-off other than 0.5. We see that the market is more likely to go up than down, so I will try using 0.47 as the predicted probability cut-off for the classifier for logistic regression.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Down Up
##       Down    2  1
##       Up     41 60
##
##                Accuracy : 0.5962
##                  95% CI : (0.4954, 0.6913)
##     No Information Rate : 0.5865
##     P-Value [Acc > NIR] : 0.4626
##
##                   Kappa : 0.0349
##
##  Mcnemar’s Test P-Value : 1.768e-09
##
##             Sensitivity : 0.98361
##             Specificity : 0.04651
##          Pos Pred Value : 0.59406
##          Neg Pred Value : 0.66667
##              Prevalence : 0.58654
##          Detection Rate : 0.57692
##    Detection Prevalence : 0.97115
##       Balanced Accuracy : 0.51506
```

```
## 
##        ’Positive’ Class : Up
## 
```

This model does have improved sensitivity as expected. However, this comes at the expense of specificity and overall accuracy.

The best accuracy readings come from using 0.5 as a cut-off.

PLOTS AND TABLES

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

Figure 1

CONCLUSION

This study aimed to compare models for Logistic Regression, Linear Discriminant Analysis, Quadratic Discriminant Analysis, and K-Nearest Neighbors to assess their ability to predict the stock market.

After analyzing the data, it appeared that a model with Lag2 as the only predictor provided the best prediction. Models based on other predictors and interactions did not do as well.

For this data set and predictor, the highest accuracy came from using Logistic Regression and Linear Discriminant Analysis, each with an accuracy rate of 62.5%.

Quadratic Discriminant Analysis had a low accuracy rate because it was not an appropriate model. It was inaccurate to assume that there was not a common covariance matrix.

K-Nearest Neighbors did well after an appropriate k-value was selected, but it was not quite as accurate as Logistic Regression or LDA. It likely would have been a good choice had LDA and Logistic Regression not been a good fit for the data.

By using a Logistic Regression or Linear Discriminant Analysis model, a stock market analyst could find moderate success as they are guaranteed to at least perform better than going off pure chance or non-mathematical models.

SOURCES

1) James, G et al. (2017). *Package 'ISLR'*. https://cran.r-project.org/web/packages/ISLR/ISLR.pdf

2) James, G et al. (2017). *An Introduction To Statistical Learning.* New York, NY.

APPENDIX

The R code is provided below:

```
library(ISLR)
summary(Weekly)
attach(Weekly)

#Boxplots of the summary statistics of the today and lag predictors

boxplot(Weekly[,c(2,3,4,5,6,8)])
title("Comparing Summary Statistics", adj = 0)

#Correlations between the predictors

library(GGally)
ggpairs(Weekly, title = "Correlation Between Predictors in the Weekly Data Set")

#Distribution plots

library(ggplot2)
Lag1.hist <- ggplot(Weekly, aes(x=Lag1)) +
    geom_histogram(binwidth=.5, colour="black", fill="white")
Lag2.hist <- ggplot(Weekly, aes(x=Lag2)) +
    geom_histogram(binwidth=.5, colour="black", fill="white")
Lag3.hist <- ggplot(Weekly, aes(x=Lag3)) +
    geom_histogram(binwidth=.5, colour="black", fill="white")
Lag4.hist <- ggplot(Weekly, aes(x=Lag4)) +
    geom_histogram(binwidth=.5, colour="black", fill="white")
Lag5.hist <- ggplot(Weekly, aes(x=Lag5)) +
    geom_histogram(binwidth=.5, colour="black", fill="white")
Volume.hist <- ggplot(Weekly, aes(x=Volume)) +
    geom_histogram(binwidth=.5, colour="black", fill="white")
Today.hist <- ggplot(Weekly, aes(x=Today)) +
    geom_histogram(binwidth=.5, colour="black", fill="white")

library(cowplot)
subset.plot.row <- plot_grid(
  Lag1.hist, Lag2.hist, Lag3.hist, Lag4.hist, Lag5.hist,
  Volume.hist, Today.hist, align = 'h'
  )

title <- ggdraw() +
  draw_label(
    "Histogram of Predictor Variables",
    x = 0,
    hjust = 0
  ) +
```

```r
  theme(
    plot.margin = margin(0, 0, 0, 7)
  )
plot_grid(
  title, subset.plot.row,
   ncol = 1,
  rel_heights = c(0.1, 1)
)

#Logistic Regression

glm.fits = glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
data = Weekly, family = binomial)
summary(glm.fits)

glm.probs = predict(glm.fits, type = "response")
glm.pred = rep("Down", 1089)
glm.pred[glm.probs >.5] = "Up"

#Creating the confusion matrix

library(caret)
confusionMatrix(as.factor(glm.pred), as.factor(Direction), positive = "Up")

#Creating Test and Train Sets

train = (Year < 2009)
Weekly.2009 = Weekly[!train, ]
Direction.2009 = Direction[!train]

#Fitting logistic regression on the training data

glm.fits.Lag2 = glm(Direction ~ Lag2, data = Weekly, family = binomial, subset = train)
glm.probs.Lag2 = predict(glm.fits.Lag2, Weekly.2009, type= "response")
glm.pred.Lag2 = rep("Down", 104)
glm.pred.Lag2[glm.probs.Lag2 > .5]= "Up"

#Confusion Matrix of Performance on Test Set

confusionMatrix(as.factor(glm.pred.Lag2), as.factor(Direction.2009), positive = "Up")

#LDA model using training data

library(MASS)
lda.fit = lda(Direction ~ Lag2, data = Weekly, subset = train)
lda.fit

lda.pred = predict(lda.fit, Weekly.2009)
lda.class = lda.pred$class

#Confusion Matrix for LDA model

confusionMatrix(as.factor(lda.class), as.factor(Direction.2009), positive = "Up")
```

```r
#QDA Model

qda.fit = qda(Direction ~ Lag2, data = Weekly, subset = train)
qda.fit

#Confusion Matrix for the test set

qda.class = predict(qda.fit, Weekly.2009)$class
confusionMatrix(as.factor(qda.class), as.factor(Direction.2009), positive = "Up")

#Setting up test and training sets for KNN

library(class)
train.X = as.matrix(as.matrix(Lag2)[train,])
test.X = as.matrix(as.matrix(Lag2)[!train,])
train.Direction = Direction[train]

#KNN with K=1

set.seed(1)
knn.pred = knn(train.X, test.X, train.Direction, k = 1)

#Confusion Matrix for KNN
knn.confusion <- confusionMatrix(as.factor(knn.pred), as.factor(Direction.2009),
                                 positive = "Up")
knn.confusion

#Creating a loop to pull the accuracy readings for k=1:20

accuracy <- vector()
for (i in 1:20) {
    set.seed(1)
    knn.pred.2 = knn(train.X, test.X, train.Direction, k = i)
    knn.confusion <- confusionMatrix(as.factor(knn.pred.2), as.factor(Direction.2009),
                                     positive = "Up")
    accuracies <- knn.confusion$overall[’Accuracy’]
    accuracy[i] <- accuracies

}

#Plotting accuracy rates for each k value

k <- c(1:20)
accuracy.data <- as.data.frame(cbind(k, accuracy))
accuracy.plot <- ggplot(data = accuracy.data, aes(x = k, y = accuracy)) +
   geom_bar(stat = "identity") +
   geom_col(aes(fill = k == which.max(accuracy))) +
   labs(x = ’K’, y = ’Accuracy’, title = ’Accuracy For Each K-Value’) +
   scale_x_continuous(breaks = 1:20) +
   scale_fill_discrete(name="Maximum Accuracy")

accuracy.plot
```

```r
#Confusion matrix for KNN when k = 4:

set.seed(1)
knn.pred.3 = knn(train.X, test.X, train.Direction, k = 4)

knn.confusion <- confusionMatrix(as.factor(knn.pred.3), as.factor(Direction.2009),
                                 positive = "Up")
knn.confusion

#Log Transforming Volume

log.Volume <- log(Volume)
Log.Volume.hist <- ggplot(Weekly, aes(x=log.Volume)) +
    geom_histogram(binwidth=.5, colour="black", fill="white")
Log.Volume.hist

#Running Logistic Regression with Transformed Volume

glm.fits.2 = glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + log.Volume,
data = Weekly, family = binomial)
summary(glm.fits.2)

#Running Logistic Regression With all Variables and Interactions

glm.fits.3 = glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + log.Volume +
                 Lag1*Lag2 + Lag1*Lag3 + Lag1*Lag4 + Lag1*Lag5 + Lag1*log.Volume +
                 Lag2*Lag3 + Lag2*Lag4 + Lag2*Lag5 + Lag2*log.Volume +
                 Lag3*Lag4 + Lag3*Lag5 + Lag3*log.Volume +
                 Lag4*Lag5 + Lag4*log.Volume,
data = Weekly, family = binomial)
summary(glm.fits.3)

#Running a logistic regression with Lag3 and Lag4 and Interaction

glm.fits.Lag34 = glm(Direction ~  Lag3 + Lag4 + Lag3*Lag4, data = Weekly,
                     family = binomial, subset = train)
summary(glm.fits.Lag34)
glm.probs.Lag34 = predict(glm.fits.Lag34, Weekly.2009, type= "response")
glm.pred.Lag34 = rep("Down", 104)
glm.pred.Lag34[glm.probs.Lag34 > .5]= "Up"

#Running logistic regression with Lag1 and Lag2 and Interaction

glm.fits.Lag12 = glm(Direction ~  Lag1 + Lag2 + Lag1*Lag2, data = Weekly,
                     family = binomial, subset = train)
summary(glm.fits.Lag12)
glm.probs.Lag12 = predict(glm.fits.Lag12, Weekly.2009, type= "response")
glm.pred.Lag12 = rep("Down", 104)
glm.pred.Lag12[glm.probs.Lag12 > .5]= "Up"

#Confusion Matrix for logistic regressoin
confusionMatrix(as.factor(glm.pred.Lag12), as.factor(Direction.2009),
                positive = "Up")
```

```r
#LDA for Lag1, Lag2, and Interaction

lda.fit.2 = lda(Direction ~ Lag1 + Lag2 + Lag1*Lag2, data = Weekly,
                subset = train)
lda.fit.2

lda.pred.2 = predict(lda.fit.2, Weekly.2009)
lda.class.2 = lda.pred.2$class

#Confusion Matrix for LDA
confusionMatrix(as.factor(lda.class.2), as.factor(Direction.2009),
                positive = "Up")

#QDA for Lag1, Lag2, and Interaction

qda.fit.2 = qda(Direction ~ Lag1 + Lag2 + Lag1*Lag2, data = Weekly,
                subset = train)
qda.fit.2

#Confusion Matrix for QDA

qda.class.2 = predict(qda.fit.2, Weekly.2009)$class
confusionMatrix(as.factor(qda.class.2), as.factor(Direction.2009),
                positive = "Up")

#KNN for Lag1, Lag2, and Interaction with k=4

train.X.2 = cbind(Lag1,Lag2,Lag1*Lag2)[train,]
test.X.2 = cbind(Lag1,Lag2,Lag1*Lag2)[!train,]
train.Direction = Direction[train]

set.seed(1)
knn.pred.2 = knn(train.X.2, test.X.2, train.Direction, k = 4)

#Confusion Matrix for KNN

knn.confusion.2 <- confusionMatrix(as.factor(knn.pred.2), as.factor(Direction.2009),
                                   positive = "Up")
knn.confusion.2

#Testing a cut-off of 0.47

glm.fits.Lag2 = glm(Direction ~ Lag2, data = Weekly, family = binomial,
                    subset = train)
glm.probs.Lag2 = predict(glm.fits.Lag2, Weekly.2009, type= "response")
glm.pred.Lag2 = rep("Down", 104)
glm.pred.Lag2[glm.probs.Lag2 > .47]= "Up"

#Logistic Regression Confusion Matrix for cut-off of 0.47

confusionMatrix(as.factor(glm.pred.Lag2), as.factor(Direction.2009),
                positive = "Up")
```

```r
#Summary Table

summary(Weekly)
```