Stock Prediction Report

Mike MacGregor

2024-04-19

Introduction

The CSV file included on the GitHub repository comes directly from the following kaggle data set: https://www.kaggle.com/datasets/kalilurrahman/mastercard-stock-data-latest-and-updated

The data set consists of 3,872 entries of Mastercard stock information from consecutive days, recording the stock's opening and closing prices on that day, maximum and minimum prices, and more information including the volume of trades on the given day. While this particular data set only handles Mastercard stocks, it is reasonable to expect that stocks trend in similar ways and a large enough data set of this particular stock would be able to generalize well to patterns for any other stock.

My goal for this project was to build a R Shiny application that accurately predicts the stock's closing price from the other parameters. This was a particularly interesting goal because the ability to predict stock behavior by the end of a given day can provide valuable insight or inform trading decisions. For example, if a stock is predicted to have extreme positive change from the opening price to the closing price, it would make sense to wait until the end of the day to trade it given it would increase in value. On the other hand, a stock that is predicted to really drop in price by the end of the day might not want to be kept due to its depreciating value.

The associated R Shiny application can be viewed at https://mjmacgregor.shinyapps.io/Stock_Prediction/

Data Preparation and Exploration

First I will do some very brief exploratory data analysis to understand the data set.

```
stocks = read.csv('stocks.csv')
head(stocks)
```

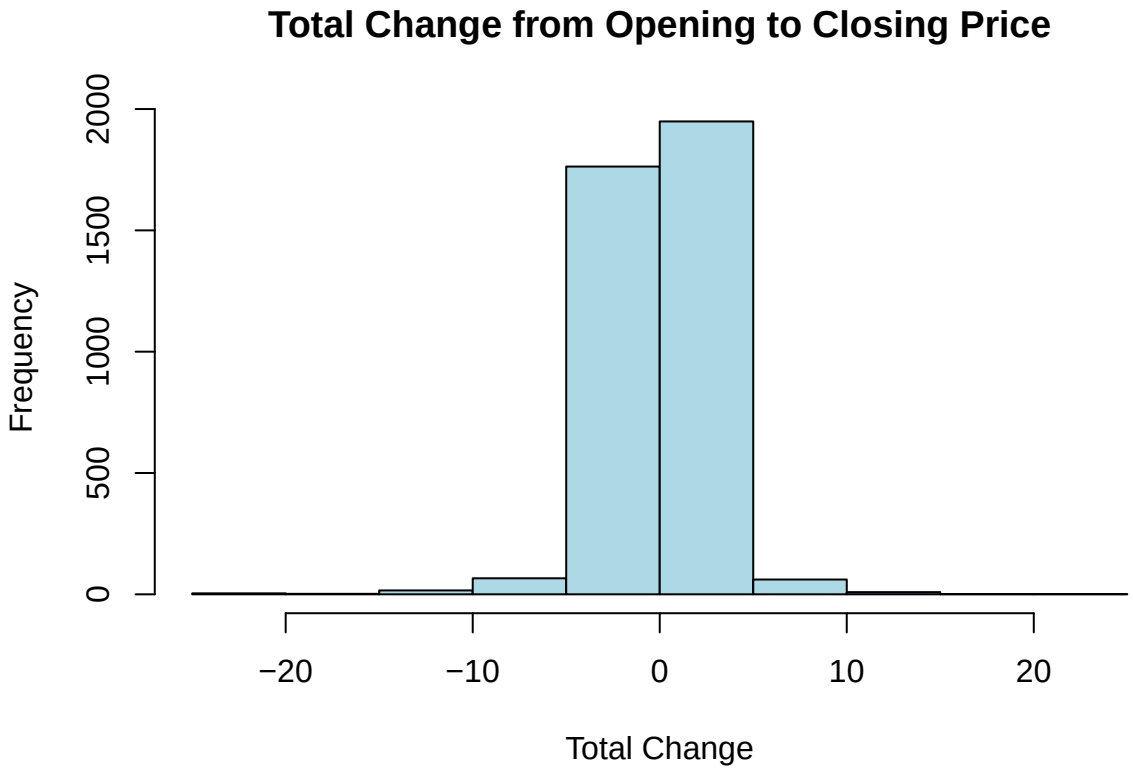
```
##         Date     Open     High      Low    Close    Volume Dividends
## 1 2006-05-25 3.748967 4.283869 3.739664 4.279217 395343000         0
## 2 2006-05-26 4.307126 4.348058 4.103398 4.179680 103044000         0
## 3 2006-05-30 4.183400 4.184330 3.986184 4.093164  49898000         0
## 4 2006-05-31 4.125723 4.219679 4.125723 4.180608  30002000         0
## 5 2006-06-01 4.179678 4.474572 4.176887 4.419686  62344000         0
## 6 2006-06-02 4.511782 4.530387 4.352707 4.371312  37253000         0
##   Stock.Splits
## 1            0
## 2            0
## 3            0
## 4            0
## 5            0
## 6            0
```

Because I aim to model the closing price of the stock in order to better understand the difference between its opening and closing price, I add features for the total change in price for the stock from open to close, as well as the percent change.

```
stocks$Change = stocks$Close - stocks$Open
stocks$Percent_Change = (stocks$Change/stocks$Open)*100
```

I can create histograms for both the total and percent change in order to better understand the distribution of the data:

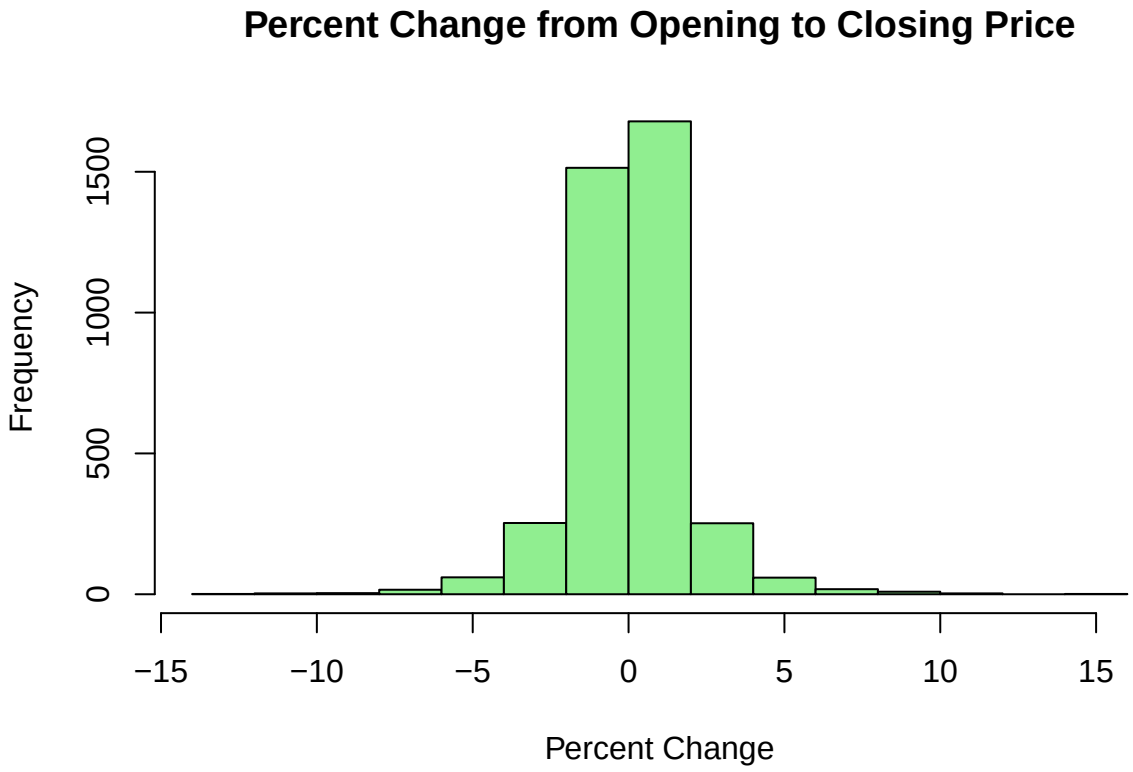
```
hist(stocks$Change, xlab = 'Total Change', ylab = 'Frequency', main = 'Total Change from Opening to Clos
```



As can be seen, the data is relatively evenly distributed for positive and negative change. As can be expected, none of the stocks experience that drastic of a change, but even small change *is significant* considering is it usually a very high volume of stocks that are acquired and traded.

Now I will plot a histogram for the percent change.

```
hist(stocks$Percent_Change, xlab = 'Percent Change', ylab = 'Frequency', main = 'Percent Change from Op
```



The histogram displays a similar shape to the previous one, but the values tend to be even smaller which can be explained since a higher total change might be a smaller percent change on a stock that has a larger overall value.

```
summary(stocks$Change)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -24.38114  -0.39175   0.02958  -0.01410   0.48469  20.96184
```

```
summary(stocks$Percent_Change)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -12.22571  -0.78540   0.06590   0.05824   0.87477  14.14391
```

However, there are some data points with a lot larger changes. The greatest negative change on the data set is *-24.38*, with the greatest positive being *20.96*.

Similarly, the percent changes range from *-12.23%* to *14.14*.

Now that I have a better understanding of the data trends between opening and closing prices, I can proceed to creating the model used in the application.

Model Selection and Linear Regression

To recap, my application aims to predict the stock's closing price from several quantitative predictors. This is a *regression* problem and I decided to use a **linear regression model** for this application.

The model's predictors will be *opening price*, *maximum price*, *minimum price*, and *volume of trades*.

Linear regression models take the form

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + \epsilon_i$$

where the response variable Y is fit to the predictors X_i. The regression coefficients β_i are estimated using ordinary least squares to minimize mean square error, or essentially, the residuals within the model. The *lm* function within R is implemented to build the linear regression model. For my application the model will take the form

$$Closing = \beta_0 + \beta_1 * (Opening) + \beta_2 * (Maximum) + \beta_3 * (Minimum) + \beta_4 * (Volume) + \epsilon_i$$

Now that I have an understanding of what my model will look like, I can move on to actually building and testing my model.

Model Building and Testing

Because I want to be able to test my model accuracy, I first split my initial data set into a training and testing set. Here I am using about 20% of the data for testing and 80% for training. I do this by randomly sampling observations from the data set as follows:

```
set.seed(1)
test = sample(1:3872, 774)
stocks_test = stocks[test, ]
stocks_train = stocks[-test, ]
```

Now that I have partitioned my data set, I fit the aforementioned linear regression model to the training data. The summary can be seen below:

```
lm_closing_price = lm(Close ~ Open + High + Low + Volume, data = stocks_train)
summary(lm_closing_price)
```

```
##
## Call:
## lm(formula = Close ~ Open + High + Low + Volume, data = stocks_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.1846  -0.2424  -0.0101   0.2059  13.3843
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.100e-03  3.508e-02   0.088    0.930
## Open        -5.163e-01  1.501e-02 -34.396   <2e-16 ***
## High         7.169e-01  1.351e-02  53.053   <2e-16 ***
## Low          8.005e-01  1.144e-02  69.974   <2e-16 ***
## Volume       7.788e-10  1.171e-09   0.665    0.506
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.068 on 3093 degrees of freedom
## Multiple R-squared:  0.9999, Adjusted R-squared:  0.9999
## F-statistic: 7.709e+06 on 4 and 3093 DF,  p-value: < 2.2e-16
```

The R^2 value for the linear regression model is *0.9999*, which is very close to 1, indicating that essentially all variability is explained in the model, and that the model is a good fit for the data.

Additionally, the p-values for the coefficients of the opening price, high price, and low price are all very statistically significant. The trade volume coefficient is not statistically significant, but it has such a low value anyways that it is kept in the model.

In order to test its accuracy, I can predict the values using the model on the test data.

```
lm_predictions = predict(lm_closing_price, stocks_test)
```

Now I can compute the associated root mean squared error of prediction.

```
test_rmse = sqrt(mean((lm_predictions - stocks_test$Close)^2))
test_rmse
```

```
## [1] 0.8600761
```

The root mean squared error of prediction is **0.860**, which is very low. This further indicates that the model is very effective at predicting the closing price for the data set.

Now that I have validated my model on my data set, I am confident in my model's ability to generate accurate predictions. This is the model that is used in the associated R Shiny application in the GitHub repository and linked at the top of this report.