Data Mining for Business (BUDT758T)

**Predicting Stock Winners & Losers from Twitter Data**

Team Members: Anthony Zdrojewski (116320580)

Sadia Parvez (116092890)

Harsh Doshi (116191111)

Bingxian Li (115460164)

Luming Yang (115565197)

## ORIGINAL WORK STATEMENT

We the undersigned certify that the actual composition of this proposal was done by us and is original work.

|  |  |  |
| --- | --- | --- |
|  | **Typed Name** | **Signature** |
| Contact Author | Harsh Doshi |  |
| Contact Author | Anthony Zdrojewski |  |
|  | Sadia Parvez |  |
|  | Bingxian Li |  |
|  | Luming Yang |  |

**Executive Summary**

The motivation of this project is best summarized by the following hypothetical:

Imagine a firm that seeks to use quantitative models to develop short-term investment strategies, powered by data models. To extract all predictive power possible, the firm is turning to data from twitter to attempt to inform its intraday trades. Can we predict short-term stock price movements by observing the live data streaming in from twitter *right now*?

If such a model could be developed, it would obviously be very valuable to a trading firm, or even a smaller investor, if commissions do not present too great a hinderance. However, under the semi-strong version of the Efficient Markets Hypothesis, we would expect that these models would overwhelmingly fail. Twitter data is publicly available (to anyone with an account and a sufficiently powerful computer), and therefore all its information content is expected to be *built in*to the prices of these stocks. In other words, if the semi-strong version of market efficiency holds, we would expect to be unable to glean any predictive power from twitter data. So, this project is of probable interest to all practitioners of finance, both in academia and industry.

It is difficult to say just how useful our results are, but they are certainly intriguing. It seems that we are able to predict with well above 50% accuracy rates whether a stock “wins” or “loses” compared to the S&P 500 over fifteen-minute intervals. The portfolio rule implemented based of the model also performed very well, albeit over a short sample (three days). Further examination of the strategy is required, but we have already built the bulk of the infrastructure to implement it or a similar strategy in a fund-like business setting.

1. **Data Description**

At the outset, we intended to compile and use data for 30 companies. In selecting companies to include, we tried to reach across industries with the hope of finding differentiated predictions between companies. At the time of model development, 15 of the companies were removed from the analyses due to severe data limitations. The full list of (30) companies is shown below.

For each of the included companies, there are two main sources of data: The Twitter-R API, and (stock data source). This data was collected every fifteen minutes each day from (begin date) to (end date). The data was partitioned into training, validation, and test datasets according to the following schedule:

|  |  |  |
| --- | --- | --- |
| **Dataset** | **Beginning Date** | **Ending Date** |
| Training | 9 March 2019 | 7 April 2019 |
| Validation | 8 April 2019 | 29 April 2019 |
| Test | 30 April 2019 | 3 May 2019 |

The data were all managed and stored on a company-specific basis. It is worth noting that there was a need to backfill stock data for some of the company/dates due to a failure of the API. The data was backfilled by pulling the same variables from the Bloomberg excel API for the necessary days/times/companies. Further, the authors deviate from their initial intent to include news data as well, due to time constraints and the sizable ambition of the project as-is.

The query to pull Twitter data had to be done on an ongoing basis due to the limitations twitter imposes on accessing historical data. Continuously compiling the Twitter data over a series of weeks enabled us to work around this limitation at no cost. The individual tweets were queried and stored based on a search of the company names in the tweet text. The list below indicates each of the variables that were queried from Twitter (<https://twitter.com/>).

**Twitter Data:**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| User\_ID | The numerical identifier of the User |
| Status\_ID | The numerical identifier of the desired Tweet. |
| Created\_at | The UTC datetime that the user account was created on Twitter. |
| Text | The text of the tweet. Newline characters are stripped out. |
| Source | The application from which the tweet was posted. |
| Display\_text\_width | Indicates the value of the text parameter, for example, the Tweet text length has a length of 140 characters. |
| is\_retweet | Yes, if tweet is a retweet of another tweet, according to the tweet’s metadata; otherwise No. |
| favorite \_count | Number of times this tweet had been favorited/liked by other users at the time the tweet |
| retweet\_count | Number of times this tweet had been retweeted by other users at the time the tweet |
| hashtags | Represents hashtags which have been parsed out of the Tweet text. |
| symbols | Represents symbols, i.e. $tags, included in the text of the Tweet. |
| urls\_url | Represents URLs included in the text of a Tweet. |
| media\_url | Represents media URLs uploaded with the Tweet. |
| media\_type | Represents media elements uploaded with the Tweet. |
| mentions\_user\_id | Represents other Twitter users ID mentioned in the text of the Tweet. |
| quoted\_status\_id | This field only surfaces when the Tweet is a quote Tweet. This field contains the integer value Tweet ID of the quoted Tweet. |
| quoted\_favourite\_count | Indicates approximately how many times this Tweet has been liked by Twitter users |
| quoted\_retweet\_count | Indicates approximately how many times this Tweet has been retweeted by Twitter users |
| quoted\_user\_id | If tweet is a quote tweet, the Twitter identifier of the author or the source tweet. |
| quoted\_followers\_count | If tweet is a quote tweet, the number of followers who followed that quote tweet . |
| quoted\_friends\_count | If tweet is a quote tweet, the number of friends the account holder has, at the time of collection of the tweet. |
| quoted\_verified | Indicates that the quoted user’s account is verified. |
| retweet\_status\_id | This field only surfaces when the Tweet is a retweet. This field contains the integer value Tweet ID of the retweet. |
| retweet\_favourite\_count | Indicates approximately how many times this retweet has been liked by Twitter users |
| retweet\_count | Number of times this Tweet has been retweeted. |
| retweet\_user\_id | If tweet is a retweet, the Twitter identifier of the author or the source tweet. |
| retweet\_followers\_count | Number of times this Tweet has been retweeted by the followers. |
| retweet\_friends\_count | Number of times this Tweet has been retweeted by the friends. |
| retweet\_verified | When true, indicates that the retweeted user has a verified account. |
| location | The user-defined location for this account’s profile |
| followers\_count | The number of followers this account currently has. Under certain conditions, this field will temporarily indicate “0”. |
| friends\_count | The number of users this account is following (AKA “followings”). Under certain conditions, this field will temporarily indicate “0”. |
| statuses\_count | The number of Tweets (including retweets) issued by the user. |
| favourites\_count | Number of times this tweet had been favorited/liked by other users at the time the tweet |
| account\_created\_at | Date and time the account was created, in ISO 8601 format and UTC time zone. |
| Verified | When true, indicates that the user has a verified account. |
|  |  |

A tremendous amount of effort was then put into processing this data. The SentimentR package was used to score the (cleaned) texts from all of the tweets, generating the additional variables “sentiment” and “st\_dev” for each tweet. Additionally, a number of the columns were transformed into dummy variables to indicate aspects of a tweet: urls\_urls indicates the presence of URLs, mentions\_user\_id indicates whether another user is mentioned in the tweet, etc. For a full list of these transformations, one might refer to the script (insert appropriate script name).

The list below indicates each of the variables that were queried from (stock data source and link).

**Stock Data:**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| cpl.companyName | Company Name |
| cpl.open | Last stock open price |
| cpl.close | Last Stock close price |
| cpl.latestPrice | Latest stock price |
| cpl.latestVolume | No. of stocks traded in the day |
| cpl.marketCap | Current Company market cap |
| cpl.peRatio | Current P/E ratio for company |
| cpl.change | Percentage change in stock price from last close. |
| date | Date and time for the record |

From the prices above, another column was created to indicate the return over the previous fifteen minutes at each time frame (15-minute intervals). In addition, the return of the S&P 500 for the same fifteen-minute interval was merged into the dataset.

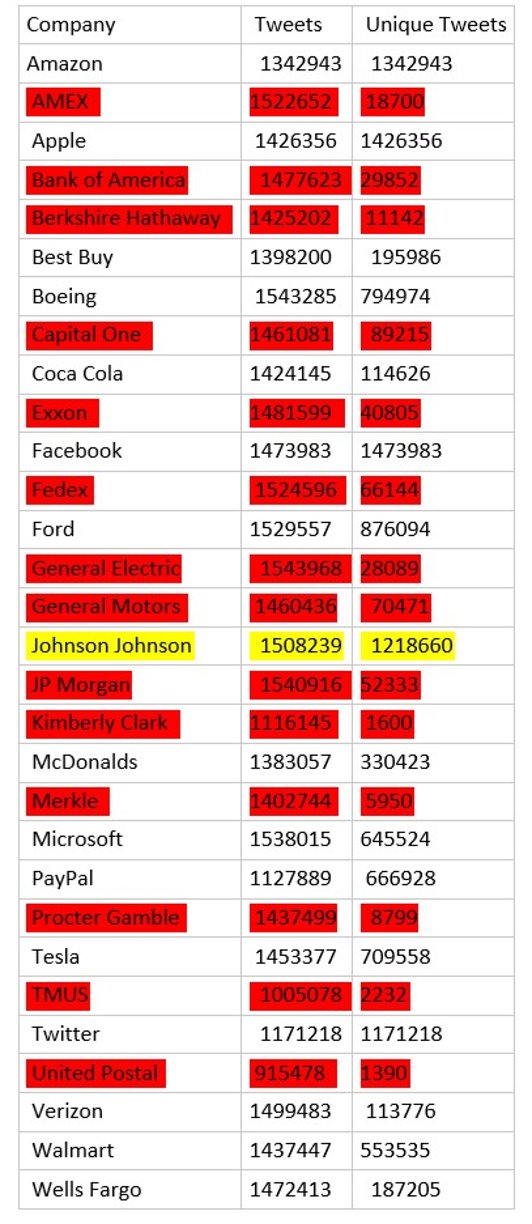
Within each of the stock data files, we created our dependent variable: win\_lose. For each of the fifteen-minute intervals, win\_lose is a dummy variable which is 1 if the stock outperforms the S&P 500 over the following fifteen minutes, and a 0 otherwise. The underline in the previous sentence is meant to emphasize that the models are indeed predictive.

For each company, the stock price data was merged into the twitter data on the basis of matching the fifteen-minute intervals. This dataset is hereafter referred to as the “unaggregated” dataset, as its records are all tweets (usually 1.3 million +).

On the other hand, each company also has an “aggregated” dataset, whose records are unique fifteen-minute intervals. The aggregated dataset is a manipulation of the unaggregated data set; roughly speaking it is an average of the predictors in each time period, paired with the win\_lose outcome of the following period. For greater detail see the script (insert).

The two datasets described above are the beginning points of the author’s models for each of the companies.

During model development, the authors became aware that for several companies, there were many duplicated tweets in their dataset. In fact, for some of the companies, there were ~1000 unique tweets over the period identified as the training time. Due to the scarcity of unique observations for some companies, the authors decided to remove from analysis companies with less than 100,000 unique tweets in the training set. The logic in doing so was that this equated to ~20 tweets per 15 minutes, which is the baseline of what felt comfortable as inputs into the predictive models. Below is a full list of the companies, number of tweets, and number of unique tweets. The highlighted companies were those that were excluded from the modeling analysis.



# III. Research Questions

The question we wish to investigate is whether we can use twitter data to predict whether a stock is likely to win or lose (relative to the S&P 500) over a 15-minute trading-day interval. The application we have in mind is simple:

1. In real-time, generate a probability that each stock will win/lose over the next fifteen minutes.
2. Implement a portfolio rule in real-time that takes an equal-weighted long position in the highest-probability stocks, and a short position in the lowest-probability stocks. The number of stocks in each position is flexible, but the authors’ original intent was 5 on both the long and short side, leaving the other 20 companies uninvested. The authors chose to keep the portfolio rule the same, however, despite only modeling for 15 companies (ie: 5 would be uninvested every fifteen-minute interval).
3. Evaluate the profits and Sharpe Ratio[[1]](#footnote-2) of this strategy on unseen (test) data, ignoring transactions costs.[[2]](#footnote-3)

In terms of the underlying models, we seek to run classification models, since the win\_lose variable we developed is binary. A further analysis might compute a predicted return, but in the interest of developing clear portfolio rules without conflating the complicated risk-expected return dynamic, we have left the models as classification efforts.

1. **Methodology**

It is worth noting here that an extraordinary effort was made in compiling the data for use in this study, and that the compilation of data is a valuable data mining technique. Unlikely sources of data were brought together in an attempt to leverage predictive ability.

As noted previously, another technique used was text-mining, as we used SentimentR to attempt to quantify the way people are talking about the companies on Twitter. From there, we could use this quantified sentiment to test whether it is useful in determining the short-term direction of the stock.

In terms of models, several different approaches were applied, usually to both the aggregated and unaggregated datasets. We have run logistic regression, K-nearest neighbors, random forests, and XGBoost algorithms. By leveraging all these different models, we hoped to identify which performed the best for each company in terms of achieving the highest validation set accuracy rate. In a way, this is a poor-man’s approach to an ensemble method; it applies the algorithm that best suits each company in determining the necessary probabilities to power the portfolio rule described in (III).

Each of the algorithms we ran were classification algorithms, with the reasoning outlined in (III).

1. **Results and Finding**

First, it should be noted that several attempts were made at developing a useful model from the unaggregated datasets, but to no avail. In some cases (KNN, Random Forest, XGBoost), the computation time was prohibitively extensive due to the size of our data. As a result, we ran each of these 4 models with aggregated data, and chose to run the best model for each company to generate its probability of winning.

The random forest method was taken into consideration though, larger number of predictor variables, gave an efficient accuracy of 65% approximately. This method was considered for the aggregated validation set with the training test. The best predictor variables for the same are baseline and the latest stock price. The baseline value indicates the S&P 500 return from last period which certainly makes sense.

The XGBoost model performed best on 7 companies. For all of them, baseline is also the most important variable, which is consistent with the economic intuition that the individual stock performance is correlated with the market performance. Some tweet related variables are also important such as retweet count, followers count, and retweet friend count. This suggests that the model captures “noise” in the trading. “Noise” is very important because our prediction time horizon is 15 minutes. Surprisingly, sentiment score itself is not so important in most cases, but it is worth paying attention that the standard deviation of sentiment score plays an important role in some cases.

In addition, when we try to implement K-nearest neighbors and logistic regression to predict whether the stock will beat the S&P 500 or not. Those models don’t have the performance comparable to Random Forest or XGBoost though, so they don’t show up as a selected model for any company for us.

The table below shows the validation accuracy for all the models.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Company** | **Logistic** | **Random Forest** | **KNN** | **XGBoost** |
| Amazon | 0.4945 | 0.6198 | 0.4917 | 0.5825 |
| Apple | 0.5402 | 0.6188 | 0.5097 | 0.6170 |
| Best Buy | 0.5525 | 0.6171 | 0.5028 | 0.5974 |
| Boeing | 0.5083 | 0.5124 | 0.5000 | 0.5949 |
| Coca Cola | 0.5552 | 0.5635 | 0.5470 | 0.5872 |
| Facebook | 0.5718 | 0.6171 | 0.5000 | 0.6103 |
| Ford | 0.5138 | 0.5647 | 0.5055 | 0.5974 |
| McDonalds | 0.5125 | 0.5207 | 0.4709 | 0.6103 |
| Microsoft | 0.5304 | 0.5785 | 0.4613 | 0.5692 |
| PayPal | 0.4583 | 0.6243 | 0.5389 | 0.5923 |
| Tesla | 0.5167 | 0.5346 | 0.5111 | 0.6240 |
| Twitter | 0.5430 | 0.6036 | 0.5134 | 0.5594 |
| Verizon | 0.5222 | 0.5592 | 0.5311 | 0.5911 |
| Walmart | 0.5263 | 0.5675 | 0.5194 | 0.6077 |
| Wells Fargo | 0.5429 | 0.6088 | 0.5313 | 0.5849 |

The table below shows the best model we selected for each company and their performance on test set. In the appendix, there is a considerably more expansive look at how the models faired.

|  |  |  |  |
| --- | --- | --- | --- |
| **Company** | **MODEL** | **AUC** | **ACCURACY** |
| Amazon​ | Random Forest​ | 0.718924​ | 0.651 |
| Apple​ | Random Forest​ | 0.659032​ | 0.639 |
| Best Buy​ | Random Forest​ | 0.608164​ | 0.59 |
| Facebook​ | Random Forest​ | 0.618087​ | 0.578 |
| Microsoft​ | Random Forest​ | 0.641421​ | 0.627 |
| PayPal​ | Random Forest​ | 0.667941​ | 0.639 |
| Twitter​ | Random Forest​ | 0.659522​ | 0.593 |
| Wells Fargo​ | Random Forest​ | 0.67056​ | 0.639 |
| Boeing​ | XGBoost​ | 0.539286​ | 0.537 |
| Coca Cola​ | XGBoost​ | 0.487132​ | 0.5 |
| Ford​ | XGBoost​ | 0.468098​ | 0.463 |
| McDonalds​ | XGBoost​ | 0.487805​ | 0.488 |
| Tesla​ | XGBoost​ | 0.507599​ | 0.561 |
| Verizon​ | XGBoost​ | 0.547024​ | 0.549 |
| Walmart​ | XGBoost​ | 0.630383​ | 0.634 |

In most settings, these test accuracy rates would not be considered impressive. However, the super competitive nature of financial markets makes anything notably better than 50% a serious win. Only two of the fifteen companies had models that performed below 50%. In a real-life setting, companies could remove those companies from their strategy, and so only make decisions based on models they are comfortable with.

It should be noted as well that neither logistic regression nor K-nearest neighbors performed well for any of the companies. This was identified on the validation set, and so when the best model for each company was run on the test data set, they were excluded. We will make little attempt to discuss these models, since they were in the end fruitless. More emphasis will be placed on the dominating models, and their role in the associated portfolio.

In addition to computing the accuracy rates above, we implemented the previously described portfolio rule on a string of 3 days (out of sample, and in fact, post-sample). We rank the companies based on the output probabilities of winning the S&P 500 and form a long-short portfolio in which we long the top 5 companies and short the bottom 5 companies. Assuming that we were to begin trading at 9:30 am at open prices, liquidate portfolio at end-of-day price, pay no transaction fees, and invest an equally-weighted portfolio, we would have realized a 3-day return of 3.73%, which is much higher than the S&P 500 return of 0.35% in test period. This is a very small sample size to make conclusions about our models’ usefulness or true profitability, but it is at least suggestive that further research may yield something of value.

The portfolio calculations were done manually at this point in excel, but it is something that could be easily implemented within a script, which would be possibly the next steps in the project.

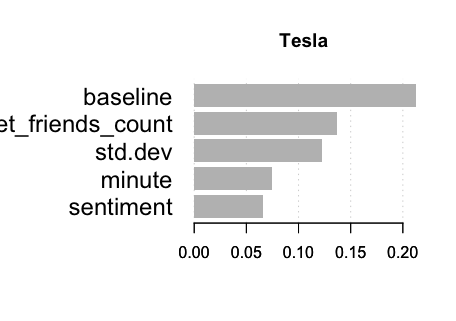
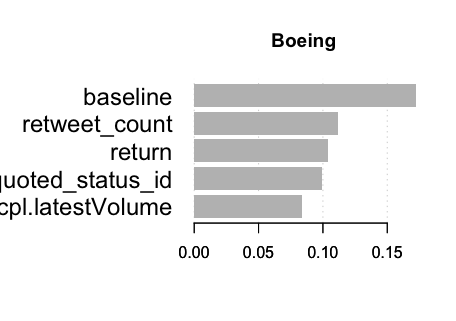
1. **Conclusion**

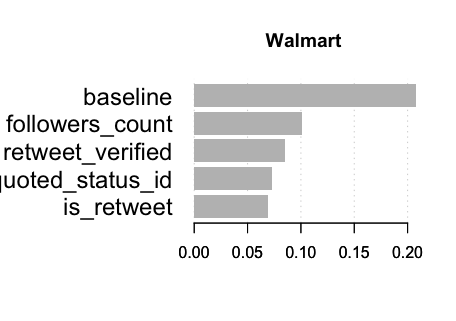
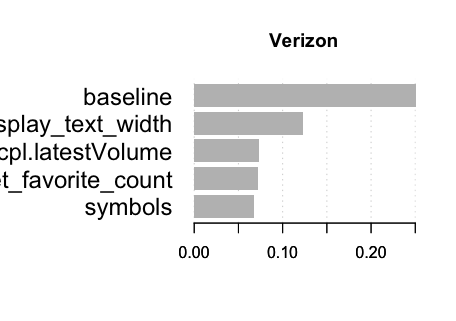
At the outset of the project, the researchers had come to understand that the odds were stacked severely against them in their attempt to generate a true predictive model. While the classification accuracy rates here would not be considered successful in some contexts, in this particular case anything above 50% is noteworthy. While from this project alone one may not draw conclusions about market efficiency or about the long-term efficacy of this type of strategy, we are encouraged by the results and believe that we have identified an unlikely source of predictability in equities markets - and indeed one that might be profitable to the right firm.

1. **Appendix:**

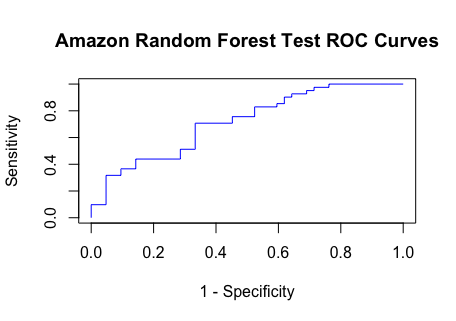
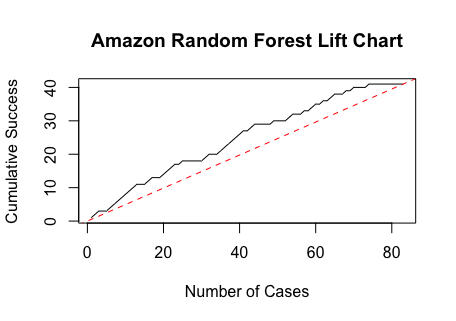
The files including the scripts are available at this [link](https://1drv.ms/f/s!AsOHjTbGh_wfssM6XHnC8rvF9P7myw). The output and the scripts are available under the folder “Scripts and Output”. The models used to run are saved in the “Models Folder”. The twitter and stock data are contained in the folder “Final” under “Modelling Data”. They are separated into training, validation and test data sets. The intermediate files are stored in the other folders.

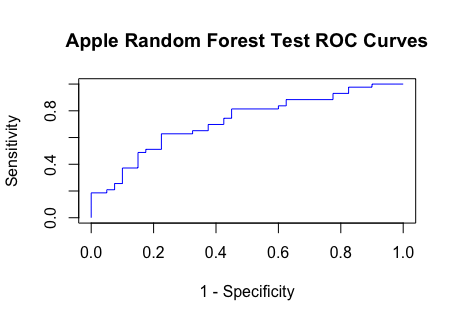
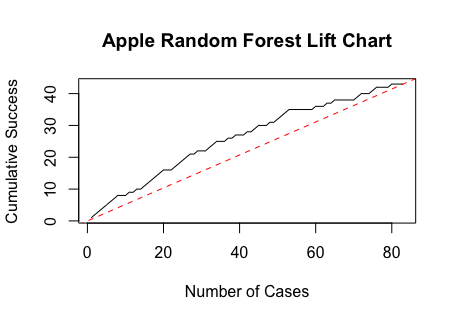
**XGBoost feature importance**

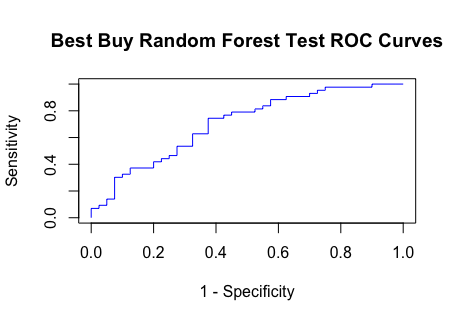
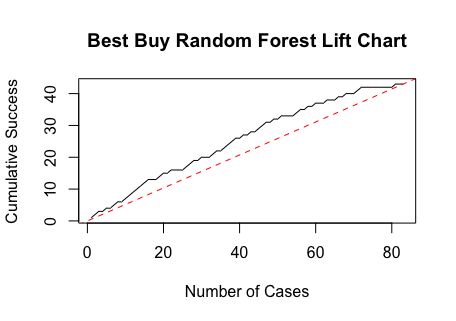


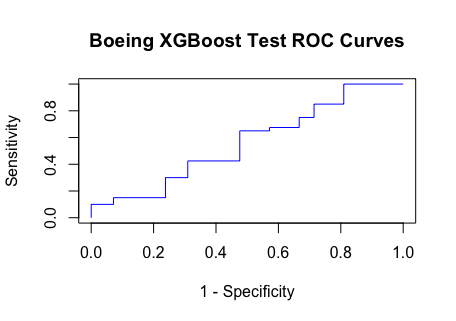
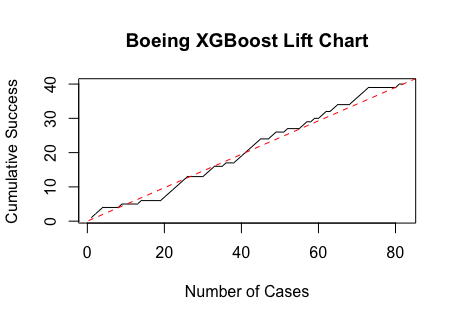


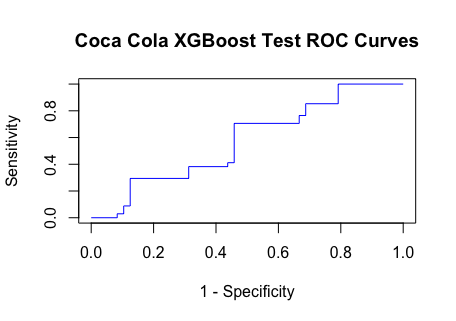
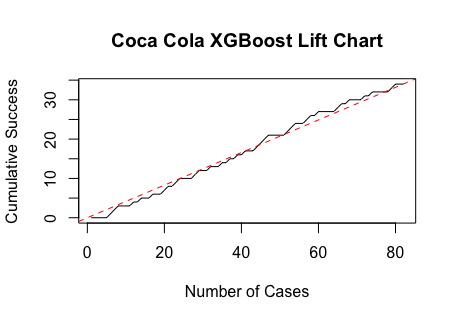
**ROC curves and lift charts for all the companies on test set**

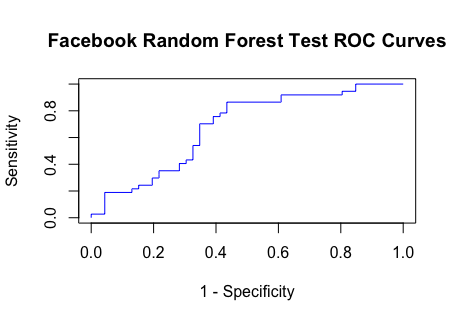
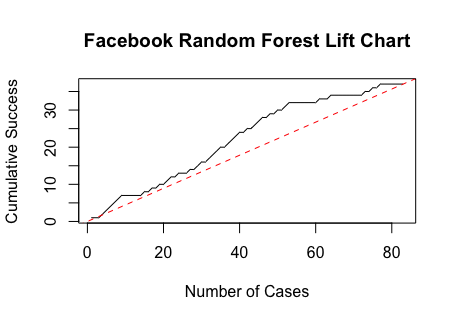


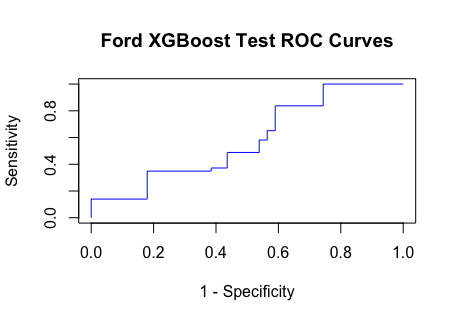
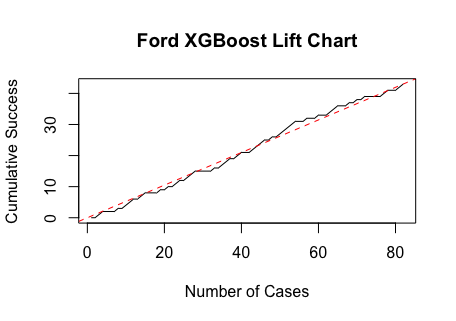


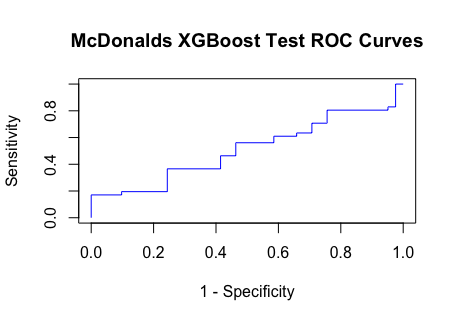
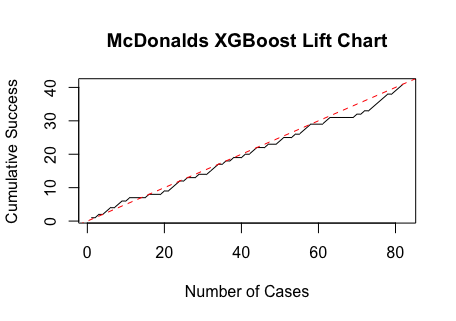


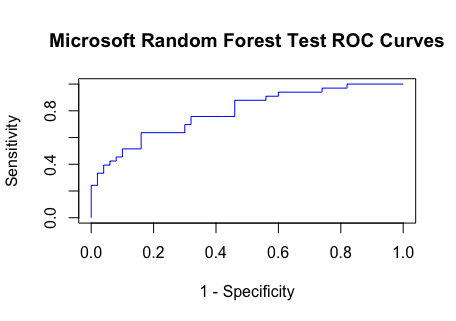
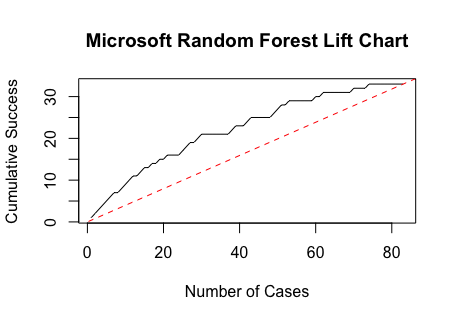


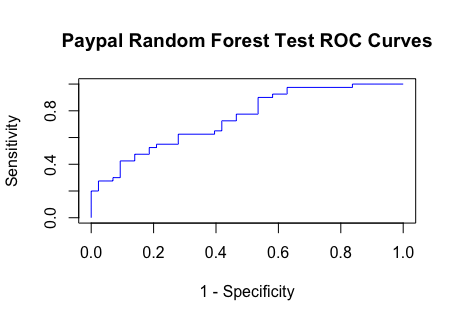
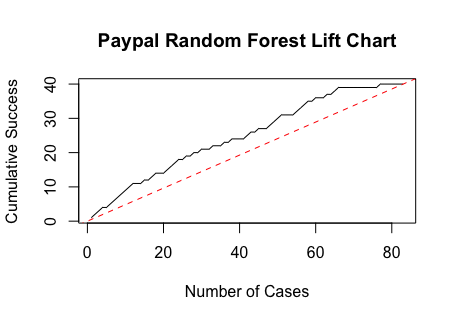


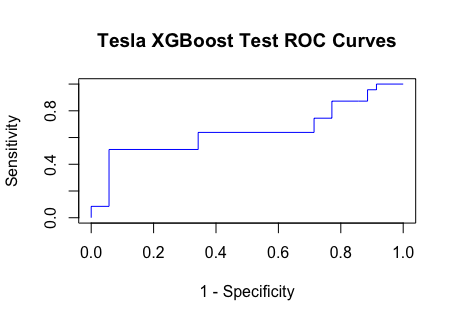
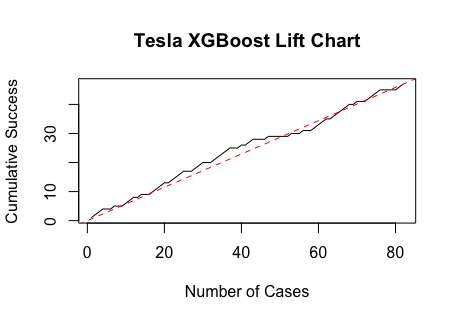


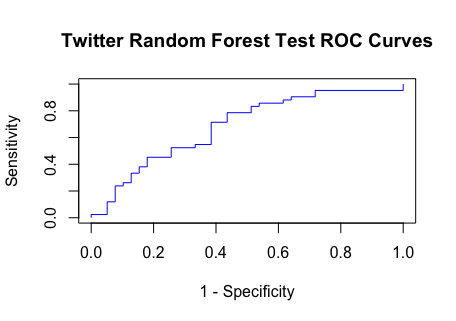
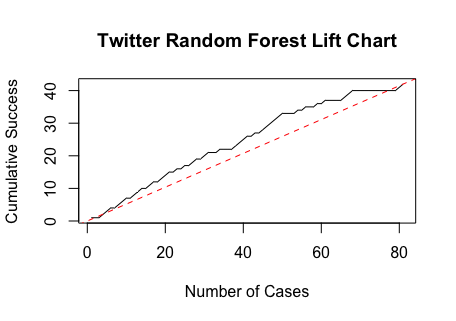


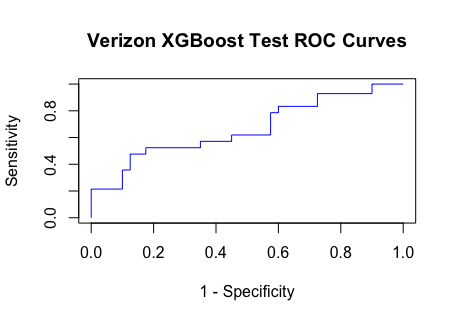
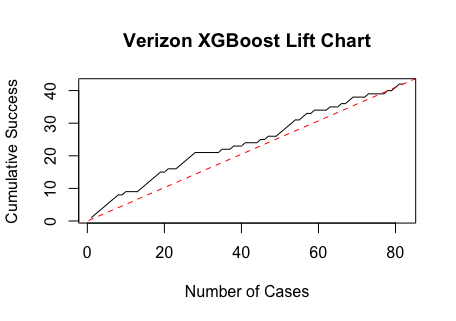


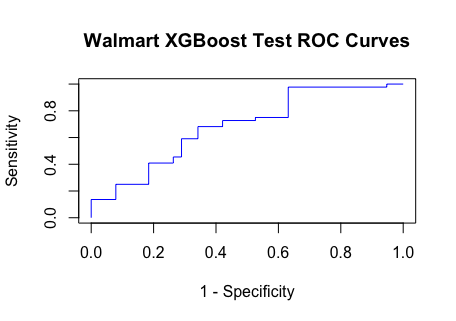
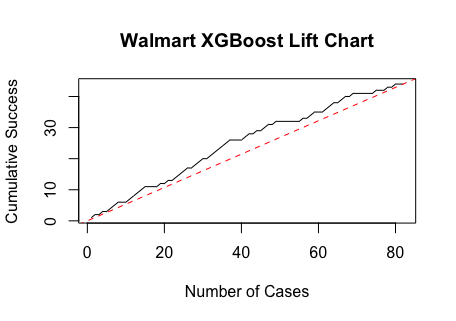


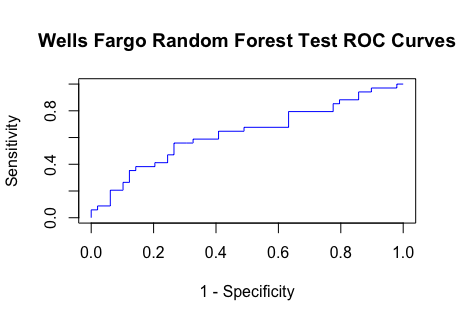
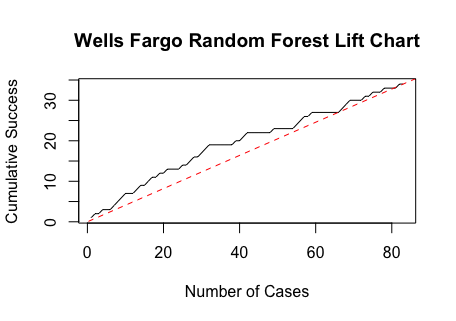






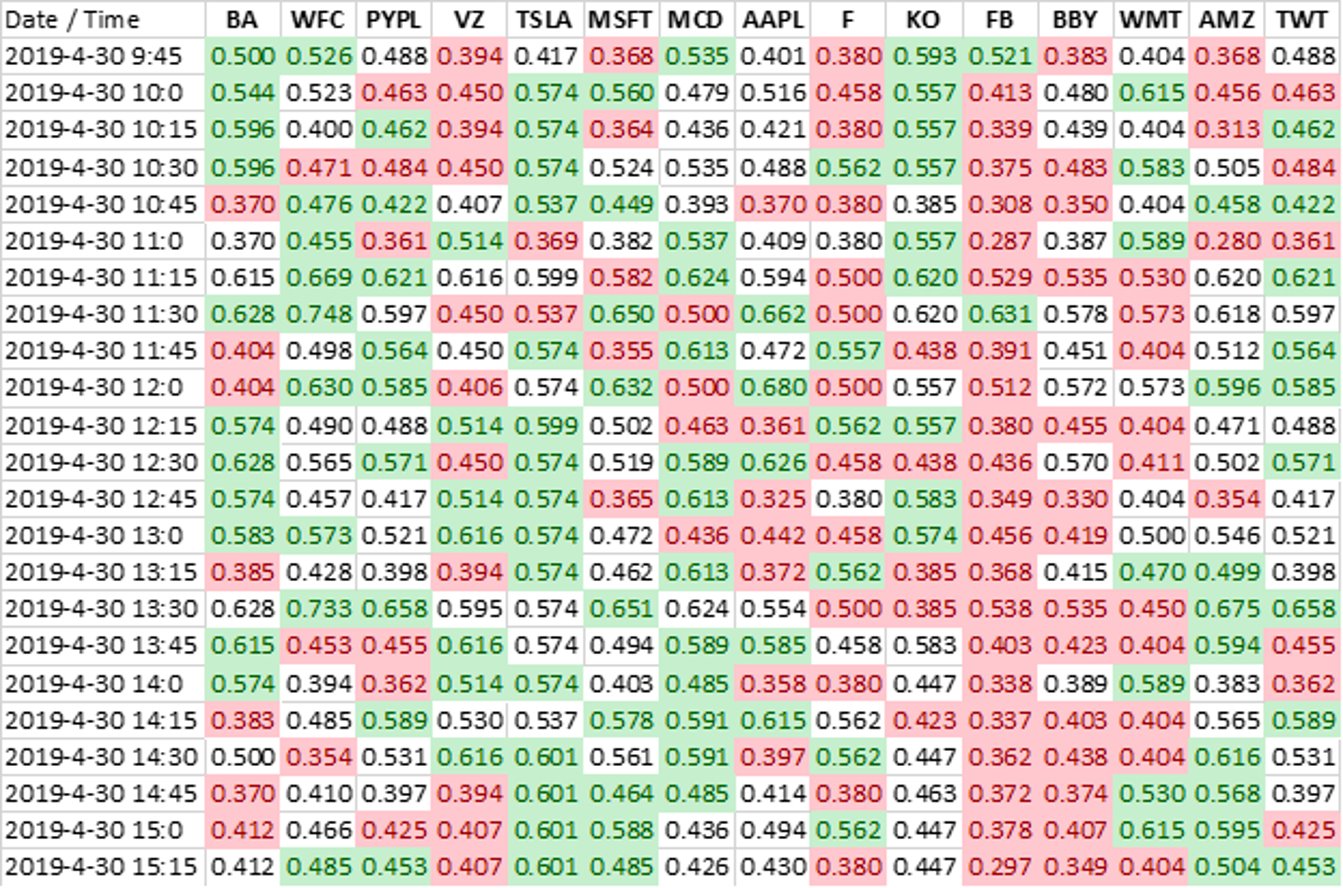


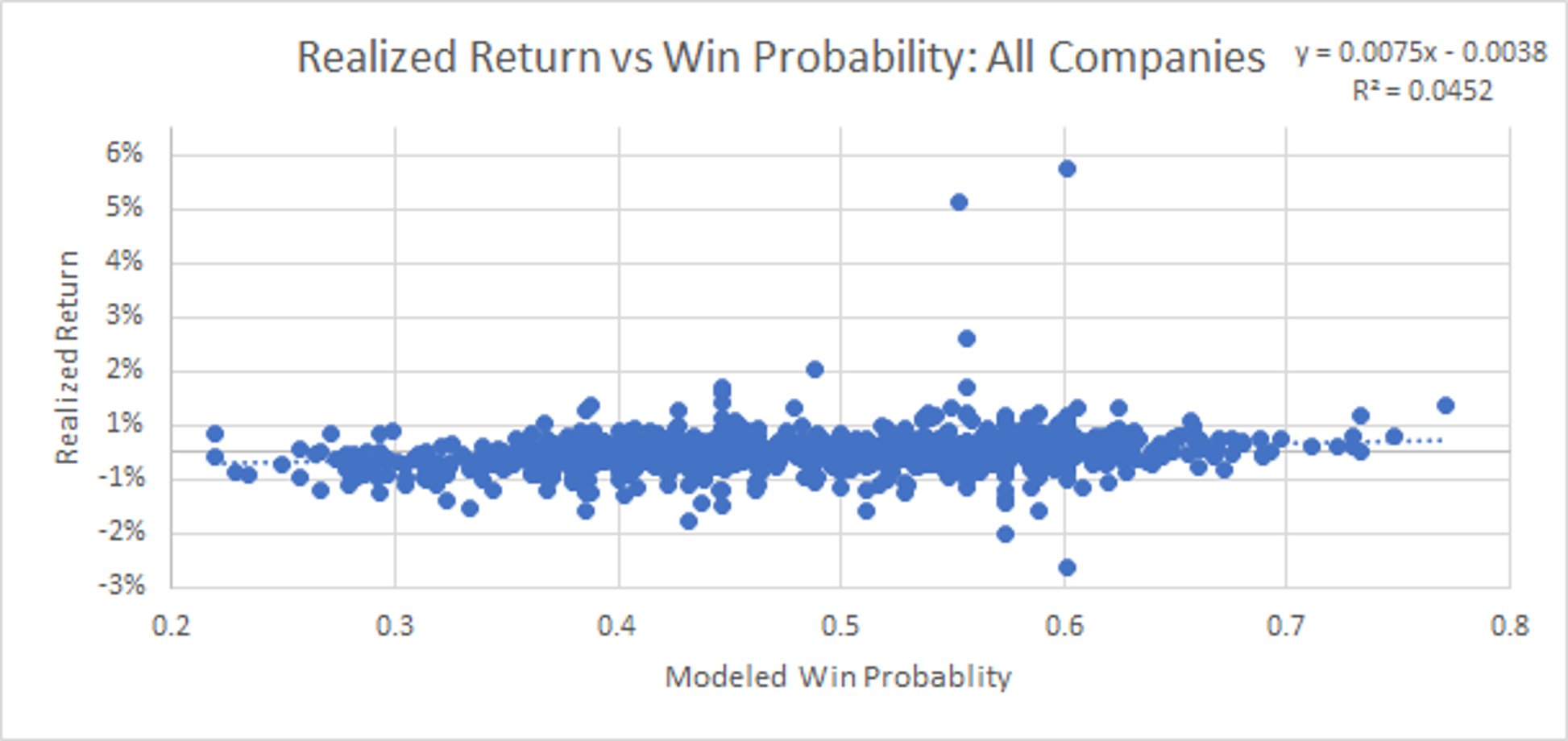




**Sample of Portfolio Rule Outcomes:**

*Note: the full list of outcomes, as well as the calculation for the portfolio return, is available in the submitted excel sheet “MasterSheet\_Updated.xlsx”.*

****

**Realized Next-Period Return vs. Model-Suggested Win Probability, All Companies: **

1. Due to the small (3-day) test window, the authors decided to save Sharpe Ratio calculations for a later time, when a more complete examination of the strategy is available. Therefore, it is not available in this report, but would be a key metric in practical implementation. [↑](#footnote-ref-2)
2. While most small investors could ill-afford to ignore transactions costs, we are comfortable doing so because of the business setting of working for a trading firm. Many of these firms specialize in short-term (intra-day) trading, and so have achieved commissions /value ratios that are quite favorable. Further, the advent of “commision less” trade platforms like RobinHood may not rule out intraday strategies even for small investors. [↑](#footnote-ref-3)