Machine Learning and Text Sentiment Analysis to Predict Microsoft Stock Prices

IST707 Applied Machine Learning

INTRODUCTION

Stock price prediction has attracted many researchers in multiple disciplines including computer science, statistics, economics, finance, and operations research. In recent years, machine learning algorithms have enabled financial institutions to process copious amounts of data, identify complex patterns and trends, and make better informed decisions quickly and accurately. This has allowed them to manage risk and optimize investment performance more effectively.

Stock market prices are fluctuating and social media platforms like Twitter seem to have a significant impact on the market movements. This has been an intriguing field of study for many researchers. Throughout the course of this project, we will be using these machine-learning techniques and text sentiment analysis to forecast the stock prices for Microsoft, an American multinational technology corporation.

Using text sentiment analysis, we will identify the sentiment around the company based on the tweets made from 1st January 2023 to 14th April 2023 using a particular set of hashtags. We will also be using historical data and other metrics to train our machine-learning models to successfully predict future stock prices.

This can help investors make informed decisions about when to enter or exit a position, identify potential opportunities to make a profit or mitigate potential losses and help to better understand market trends, sentiment and economic factors that may impact the stock price, which can be useful for long-term investment strategies.

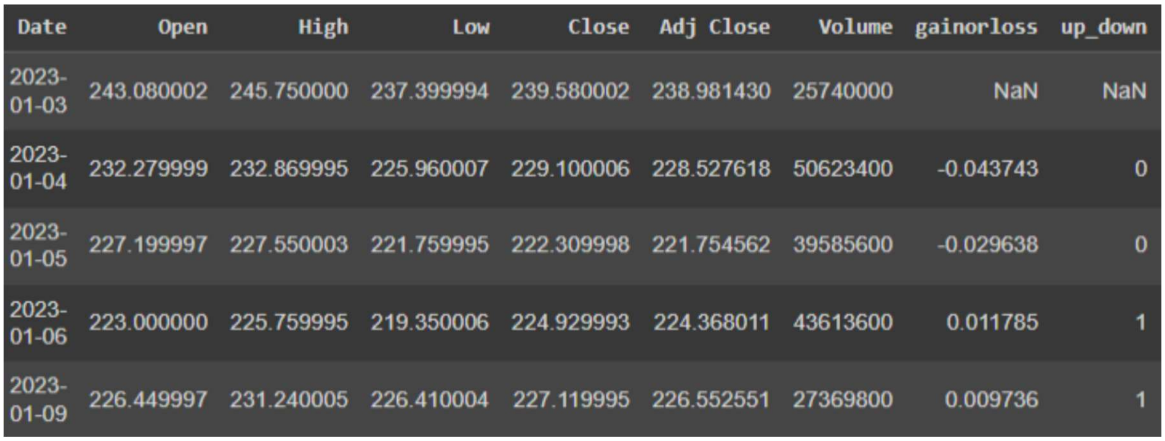
DATA ACQUISITION

For the project, the financial data was acquired from Yahoo Finance which is a popular financial website providing real-time stock market prices. The data that was downloaded as a CSV file format contain some important metrics like the Date, Open: The opening price of the stock for a given day, High: The highest price of the stock for a given day, Close: The closing price of the stock for a given day, Adj Close: The adjusted closing price for any dividends or other corporate actions and Volume: The total number of Microsoft's shares traded on that day.

For the tweets required for sentiment analysis, as the Twitter api had certain restrictions we decided to use Apify which is a platform providing web scraping tools necessary to collect large volumes of data. We manage to collect around 78,000 tweets from 1st January 2023 to 14th April 2023 related to two hashtags #MSFT and #Microsoft. We extracted important metrics like the Date of the tweet, Full text of the tweet, Total retweet count for that particular tweet, Total view count for that particular tweet, Total number of likes for the tweet, Total number of replies for the tweet

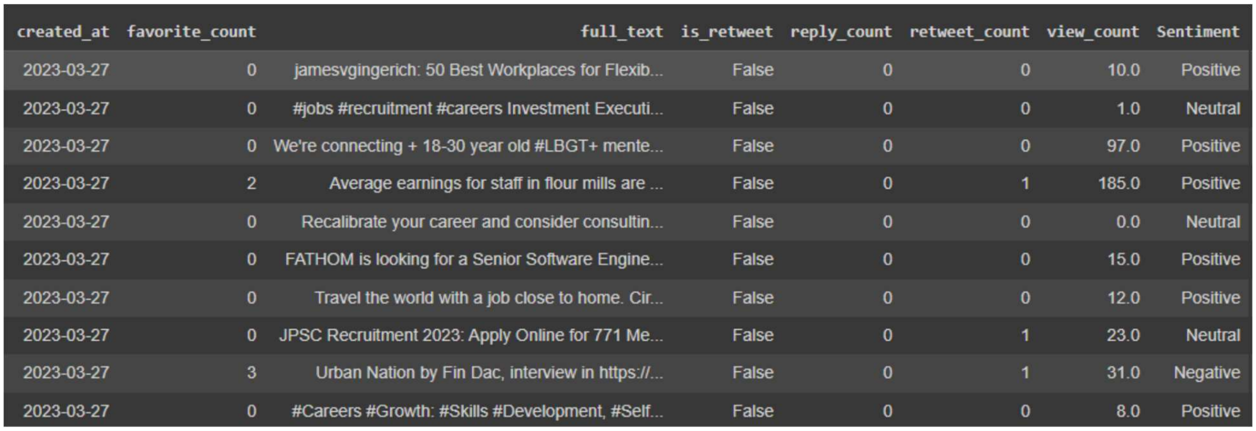
DATA PRE-PROCESSING

Before feeding the data into a machine learning model for analysis, it is important to preprocess and clean the data to ensure that it is accurate, consistent, and ready for analysis. We begin by looking for any null, or incomplete data values after loading in the financial data. As we discover that we do not have any of the above issues we proceed to calculate the percentage change in the adjusted closing price for each given day to determine the gain or loss and assign a new column with all the values. We then define a function called up_down that takes in the gainloss values and returns a 1 or 0 based on the percentage change in the adjusted closing price with 1 being assigned when the value is a positive value and 0 being assigned when the value is negative. In cases where the percentage change is exactly 0, the function returns 'NaN', indicating that there was no change in the stock price on that day. The results are stored in a new column called up_down.



Date	Open	High	Low	Close	Adj Close	Volume	gainorloss	up_down
2023-01-03	243.080002	245.750000	237.399994	239.580002	238.981430	25740000	NaN	NaN
2023-01-04	232.279999	232.869995	225.960007	229.100006	228.527618	50623400	-0.043743	0
2023-01-05	227.199997	227.550003	221.759995	222.309998	221.754562	39585600	-0.029638	0
2023-01-06	223.000000	225.759995	219.350006	224.929993	224.368011	43613600	0.011785	1
2023-01-09	226.449997	231.240005	226.410004	227.119995	226.552551	27369800	0.009736	1

For the tweets data, we again look for any null or incomplete values and remove them in case any are present in the data after reading the data file we got from Apify. We have around 78469 rows and 8 columns present. We then remove all the non-alphabetic characters from the tweet texts and put them in a list. For assigning the sentiment score to the tweet we decided on using a pre-trained Sentiment Analyzer tool called Vader Sentiment which has a lexicon of words and phrases with a score assigned to them and based on that score we get the overall calculated score for any text. We created a list called sent_list which will contain the sentiment score i.e positive, negative, or neutral for each tweet in the tweet list created above. This list is then assigned to a new column called Sentiment in the data frame.



created_at	favorite_count	full_text	is_retweet	reply_count	retweet_count	view_count	Sentiment
2023-03-27	0	jamesvgingerich: 50 Best Workplaces for Flexib...	False	0	0	10.0	Positive
2023-03-27	0	#jobs #recruitment #careers Investment Executi...	False	0	0	1.0	Neutral
2023-03-27	0	We're connecting + 18-30 year old #LBGT+ mente...	False	0	0	97.0	Positive
2023-03-27	2	Average earnings for staff in flour mills are ...	False	0	1	185.0	Positive
2023-03-27	0	Recalibrate your career and consider consultin...	False	0	0	0.0	Neutral
2023-03-27	0	FATHOM is looking for a Senior Software Engine...	False	0	0	15.0	Positive
2023-03-27	0	Travel the world with a job close to home. Cir...	False	0	0	12.0	Positive
2023-03-27	0	JPSC Recruitment 2023: Apply Online for 771 Me...	False	0	1	23.0	Neutral
2023-03-27	3	Urban Nation by Fin Dac, interview in https://...	False	0	1	31.0	Negative
2023-03-27	0	#Careers #Growth: #Skills #Development, #Self...	False	0	0	8.0	Positive

We then group the data frame by date and sentiment to get a total count of the number of positive, negative, and neutral tweets for each given day. The grouped data frame is then pivoted to create new columns and display the total sum of the tweets for the sentiment for each day. It is then merged with another data frame that is created by grouping the original data frame by date to get the sum of likes, replies, and views for every day in the data. We then calculate the adjusted sentiment score for each day by subtracting the number of negative tweets from the number of positive tweets and dividing the entire value by the total number of tweets and stored in a column called adjusted_score. Finally, both the financial and tweets data are merged on the date after the pre-processing step to get the final data frame which will be used for the machine learning analysis.

Basic setting before building up models

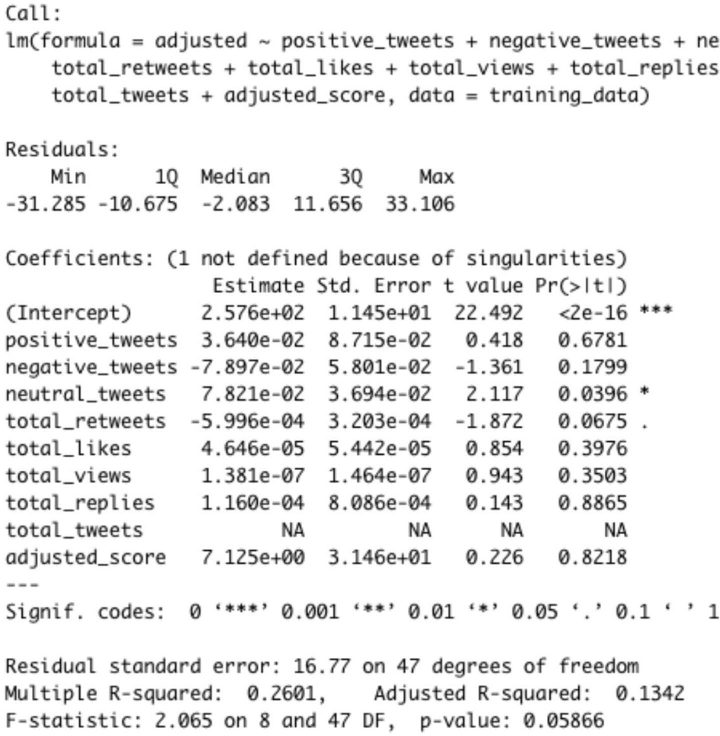
(1) Factors:

- Outcome variables: Adj Close (regression), Up/Down (classification)
- Input variables: positive_tweets, negative_tweets, neutral_tweets, total_retweets, total_likes, total_views, total_replies, total_tweets, adjusted_score

(2) Splitting Training and Testing datasets: 80% as training dataset and 20% as testing dataset

Linear regression

The model involves the values of the coefficient that are used in the representation of the data. It includes the statistical properties that are used to estimate those coefficients. We use the adjusted close stock price as the outcome variable.



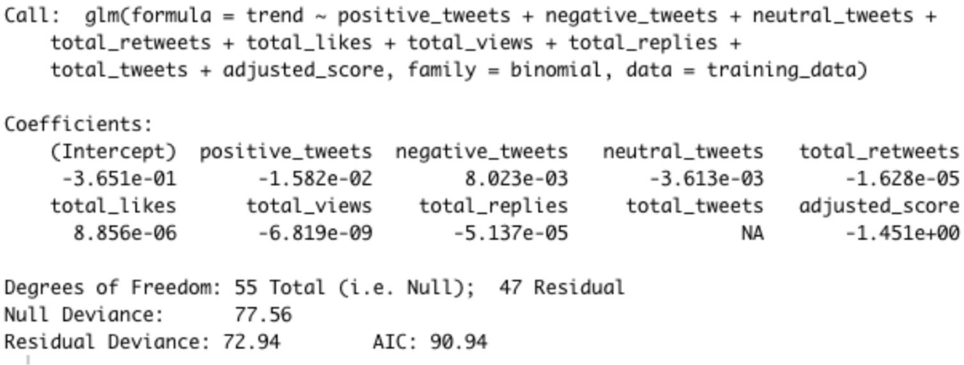
```
Call:
lm(formula = adjusted ~ positive_tweets + negative_tweets + neutral_tweets +
    total_retweets + total_likes + total_views + total_replies +
    total_tweets + adjusted_score, data = training_data)

Residuals:
    Min      1Q  Median      3Q     Max
-31.285 -10.675  -2.083  11.656  33.106

Coefficients: (1 not defined because of singularities)
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)      2.576e+02  1.145e+01  22.492   <2e-16 ***
positive_tweets  3.640e-02  8.715e-02   0.418   0.6781
negative_tweets -7.897e-02  5.801e-02  -1.361   0.1799
neutral_tweets   7.821e-02  3.694e-02   2.117   0.0396 *
total_retweets  -5.996e-04  3.203e-04  -1.872   0.0675 .
total_likes      4.646e-05  5.442e-05   0.854   0.3976
total_views      1.381e-07  1.464e-07   0.943   0.3503
total_replies    1.160e-04  8.086e-04   0.143   0.8865
total_tweets            NA         NA      NA       NA
adjusted_score   7.125e+00  3.146e+01   0.226   0.8218
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 16.77 on 47 degrees of freedom
Multiple R-squared:  0.2601,    Adjusted R-squared:  0.1342
F-statistic: 2.065 on 8 and 47 DF,  p-value: 0.05866
```

The lm model's coefficient and standard error for the variable "total_tweets" are listed as NA (not available) because the variable has perfect multicollinearity with the intercept, meaning it does not provide any additional information beyond what is already captured by the intercept. Therefore, it is dropped from the model.

The adjusted R-squared value of 0.1342 indicates that the model does not explain a substantial portion of the variance in the response variable. Therefore, we have decided against using this model to directly predict the price.

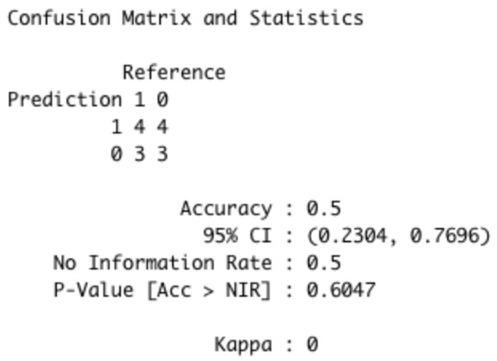
Logit regression

The goal of logistic regression is to find the best set of coefficients that minimize the difference between the predicted probabilities and the observed binary outcomes in the training data.

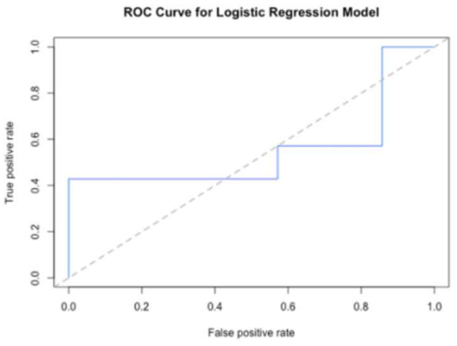


```
Call:  glm(formula = trend ~ positive_tweets + negative_tweets + neutral_tweets +
    total_retweets + total_likes + total_views + total_replies +
    total_tweets + adjusted_score, family = binomial, data = training_data)

Coefficients:
    (Intercept)  positive_tweets  negative_tweets   neutral_tweets   total_retweets
     -3.651e-01       -1.582e-02        8.023e-03       -3.613e-03       -1.628e-05
    total_likes      total_views    total_replies     total_tweets   adjusted_score
      8.856e-06       -6.819e-09       -5.137e-05               NA       -1.451e+00

Degrees of Freedom: 55 Total (i.e. Null);  47 Residual
Null Deviance:      77.56
Residual Deviance: 72.94        AIC: 90.94
```

The coefficients are like the linear regression, but the accuracy is higher as follows:



```
Confusion Matrix and Statistics

          Reference
Prediction 1 0
         1 4 4
         0 3 3

               Accuracy : 0.5
                 95% CI : (0.2304, 0.7696)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : 0.6047

                  Kappa : 0
```



From the test we can get the accuracy is 0.50, and which makes sense.

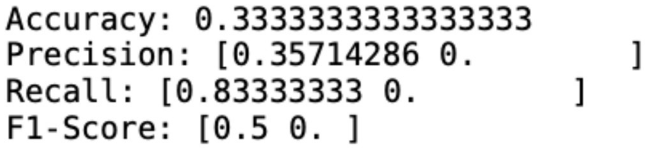
Machine Learning techniques

Supervised learning is a type of machine learning where an algorithm is trained using input/output pairs. The algorithm learns from these pairs to create a mapping between the input and output variables and can then use this mapping to predict the output for new input data.

We chose Support Vector Machines (SVM), decision trees, and random forests for both regression and classification in this case.

1. SVM model

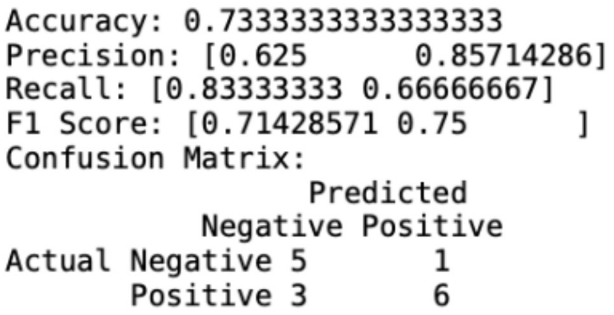
A support vector machine (SVM) is a technique that constructs a hyperplane or set of hyperplanes in a high-dimensional space, which can then be used for classification or regression tasks. The test models we get the accuracy as 0.33, which is not well, so we decided to move on to the other models.



```
Accuracy: 0.3333333333333333
Precision: [0.35714286 0.        ]
Recall: [0.83333333 0.        ]
F1-Score: [0.5 0. ]
```

2. Decision Tree

Decision trees are another technique that works by recursively splitting the data into smaller subsets based on the values of the input variables, and then predicting the output based on the majority class or average value within each subset.



```
Accuracy: 0.7333333333333333
Precision: [0.625      0.85714286]
Recall: [0.83333333 0.66666667]
F1 Score: [0.71428571 0.75      ]
Confusion Matrix:
                   Predicted
           Negative Positive
Actual Negative 5      1
       Positive 3      6
```

As the conclusion that the above shows, we built Classification Decision Tree:

The test data yielded an impressive accuracy of 73% in the decision tree model. To further enhance the results, we believed that implementing a random forest could prove beneficial.

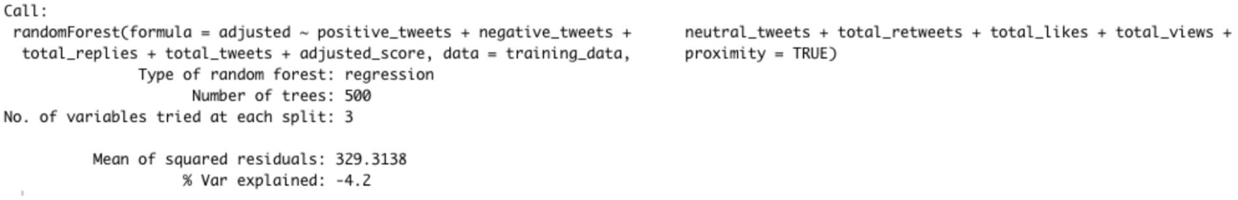
3. Random Forest:

Random forests are an ensemble method that combines multiple decision trees to improve the accuracy and robustness of the predictions. In a random forest, each decision tree is trained on a random subset of the training data and a random subset of the input variables, which helps to reduce overfitting and increase the diversity of the trees in the ensemble.

We tried to use the 'randomForest' package to predict the RF models; however, this package is limited due to cross-validation method is not a built-in function, and we must manually set the repeat cv in this package.

(1) Regression Random Forest – Daily base forecasting

We use the regression RF to predict the next day stock price

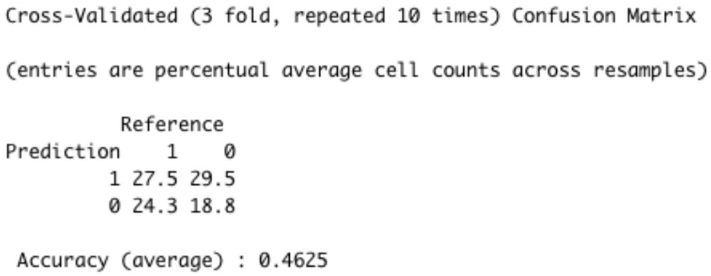


```
Call:
 randomForest(formula = adjusted ~ positive_tweets + negative_tweets +      neutral_tweets + total_retweets + total_likes + total_views +
  total_replies + total_tweets + adjusted_score, data = training_data,      proximity = TRUE)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 3

          Mean of squared residuals: 329.3138
                    % Var explained: -4.2
```

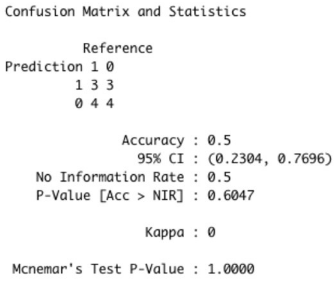
The mean of squared residuals is 329.3138, which indicates that the model does not fit the data very well, and there is still a significant amount of variation that is not explained by the model. The percentage of variance explained negative value, indicates that the model does not explain any variance in the response variable and is performing worse than a model that predicts the mean value of the response variable.

(2) Classification Random Forest -- Daily base forecasting

We train another RF model to classification the daily base trend



```
Cross-Validated (3 fold, repeated 10 times) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction    1    0
         1 27.5 29.5
         0 24.3 18.8

 Accuracy (average) : 0.4625
```

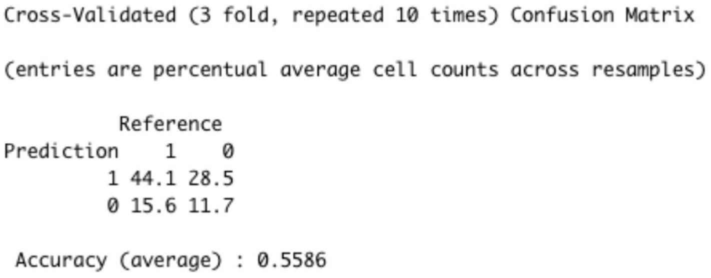


```
Confusion Matrix and Statistics

          Reference
Prediction 1 0
         1 3 3
         0 4 4

               Accuracy : 0.5
                 95% CI : (0.2304, 0.7696)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : 0.6047

                  Kappa : 0

 Mcnemar's Test P-Value : 1.0000
```

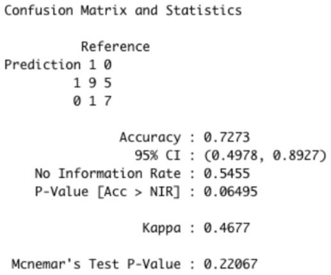
The average accuracy is 0.4625, and test data has 0.5 accuracy just like prior results.

(3) Classification Random Forest -- Daily base forecasting (all days contain weekends)

We believe that incorporating additional observed data to build the model is a viable approach, especially since Twitter activity is consistent throughout the week and is not disrupted by weekends or holidays. Therefore, we intend to include all available tweet data for training and use the latest stock price to fill in the missing values during weekends and holidays.



```
Cross-Validated (3 fold, repeated 10 times) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction    1    0
         1 44.1 28.5
         0 15.6 11.7

 Accuracy (average) : 0.5586
```

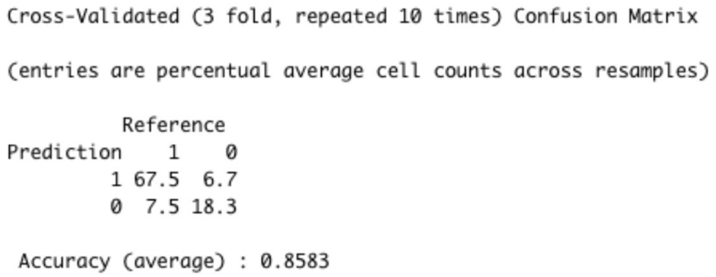


```
Confusion Matrix and Statistics

          Reference
Prediction 1 0
         1 9 5
         0 1 7

               Accuracy : 0.7273
                 95% CI : (0.4978, 0.8927)
    No Information Rate : 0.5455
    P-Value [Acc > NIR] : 0.06495

                  Kappa : 0.4677

 Mcnemar's Test P-Value : 0.22067
```

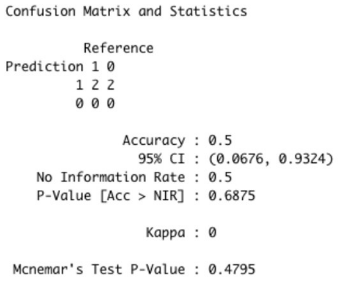
The average accuracy is 0.5586 and the test accuracy is 0.7273 seems to improve.

(4) Classification Random Forest -- Weekly base forecasting (all days contain weekends)

We split the dataset into weekly base to build another forest.



```
Cross-Validated (3 fold, repeated 10 times) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction    1    0
         1 67.5  6.7
         0  7.5 18.3

 Accuracy (average) : 0.8583
```

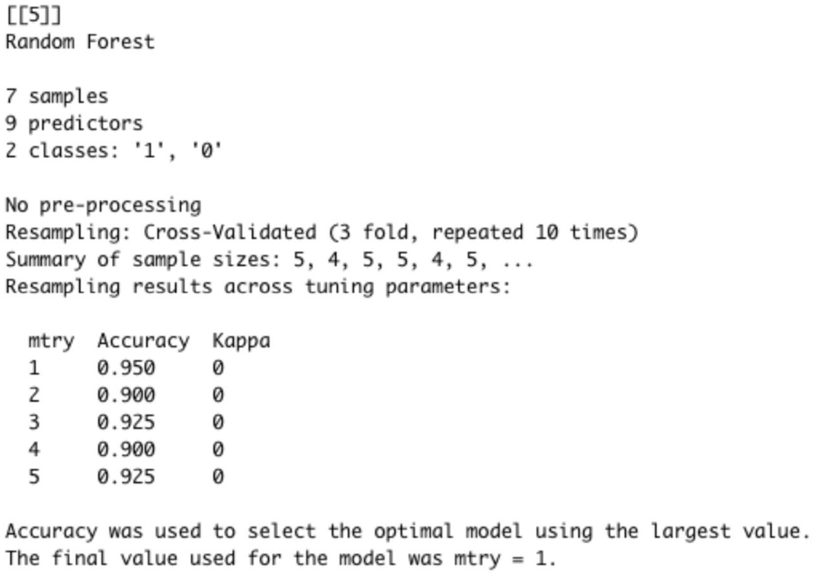


```
Confusion Matrix and Statistics

          Reference
Prediction 1 0
         1 2 2
         0 0 0

               Accuracy : 0.5
                 95% CI : (0.0676, 0.9324)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : 0.6875

                  Kappa : 0

 Mcnemar's Test P-Value : 0.4795
```

The average accuracy of 0.65 appears great, however, as the test dataset only contains four rows, it is not ideal for testing. As a solution to this issue, we decided to use the running window model.

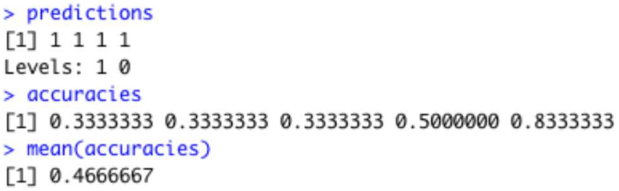
(5) Classification Random Forest -- Weekly base forecasting (running window)

Given that the data is a time series dataset, we opted to utilize a weekly running window to train the forest.



```
[[5]]
Random Forest

7 samples
9 predictors
2 classes: '1', '0'

No pre-processing
Resampling: Cross-Validated (3 fold, repeated 10 times)
Summary of sample sizes: 5, 4, 5, 5, 4, 5, ...
Resampling results across tuning parameters:

  mtry  Accuracy  Kappa
  1     0.950     0
  2     0.900     0
  3     0.925     0
  4     0.900     0
  5     0.925     0

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 1.
```

The 5th model from the running window models looks fine, since the accuracy was high

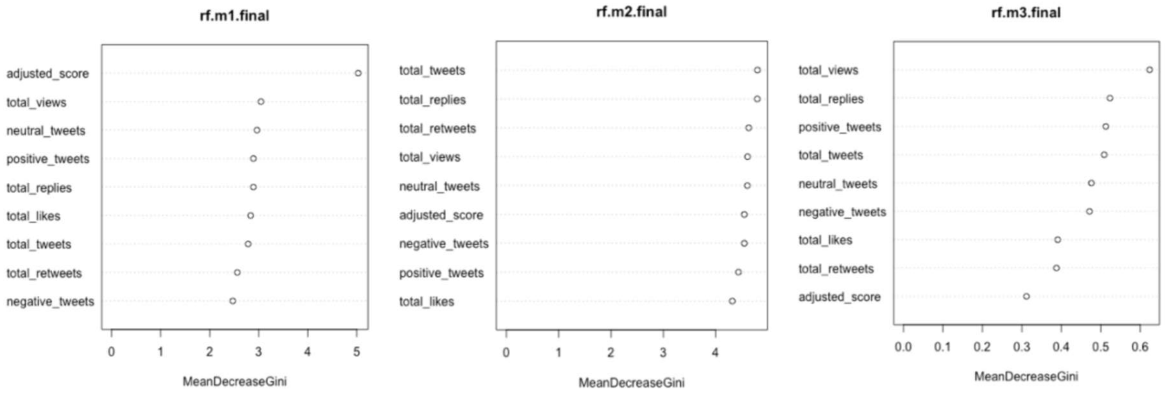


```
> predictions
[1] 1 1 1 1
Levels: 1 0
> accuracies
[1] 0.3333333 0.3333333 0.3333333 0.5000000 0.8333333
> mean(accuracies)
[1] 0.4666667
```

We observed a clear trend in the accuracy of the models, with an improvement from 0.333 to 0.500 and 0.833 over time. This led us to believe that the accuracy of the model increases as the date approaches.

Random Forest Factors' importance

Another interesting finding is that the order of the importance of the factors is different in each forest, seems that the tweet factors share the same importance on measuring the impact.



From the data getting more (the rf.m1 only has weekday daily data, rf.m2 has contain weekend daily data and rf.m3 is weekly data) the accuracy is getting higher but the factor of adjusted_score is lower.

COMPARE THE MODELS

The table of the accuracy rate for the weekday daily basic model

	Linear Regression	Logistic Regression	SVM (Class)	Decision Tree (Class)	Random Forest (Class)
Accuracy	Mean Square Error 18 R square 13.4%	50.0%	33.3%	73.3%	50.0%
Precision		50.0%	35.7%	16.7%	50.0%
Recall		57.1%	88.3%	50.0%	100.0%

So when we added stock price data factors to train the daily basis predicts, include transaction volume and prices. The result is as follows:

	Linear Regression	Logistic Regression	SVM (Class)	Decision Tree (Class)	Random Forest (Class)
Accuracy	Mean Square Error R square 59.8%	46.6%	46%	93.33%	72.73%
Precision		54%	57%	90%	64.23%
Recall		66.7%	83%	83.33%	90%

CONCLUSION

According to the efficient markets hypothesis (Fama E, 1970), it is impossible to predict stock prices because they respond to the arrival of new information, and the news cannot be anticipated. However, we know that somehow in this era, social media has been an important factor and impact on the real life and market. We believe that the point of view of the twitter user can make impact. Here are some interesting findings from the research and insights by utilizing machine learning techniques.

(1) Through our research, we discovered that classification models are more effective in predicting the stock price trend than regression models. Most of these models predicted an upward trend in the next period, which we believe is due to the short duration of observation resulting from the limitations of the Twitter API, and Microsoft is making significant strides in the AI space, intensifying the competition among industry players. If we could obtain more data and expand our observations to include other social media or platforms, we may be able to develop a more accurate and robust model.

(2) The importance of the factors of random forest shows a wired trend that the more change we made, the lower the importance of the sentiment score. We believe that it is because the sentiment score may be sensitive to noise and fluctuations, making it less reliable as a predictor of the stock price trend. Almost all the sentiment reflected negative might also create some unbalance problems.

(3) While tracking the data based on the hashtags of Microsoft, it's important to note that there may be other factors that could influence investors' decisions to trade Microsoft's stock. Observing the relationship between the algorithms of Twitter and other factors such as search trends can provide a more comprehensive understanding of the market.

(4) The stock price dataset twitter dataset are time series data and the methods we choose were all supervised models. Some articles indicate that deep learning can do a better job on time series data. The other way to modeling time series data with machine learning algorithms can be challenging due to several reasons such as high dimensionality, non-stationarity, nonlinearity, and dynamic dependencies.

(5) We found out that decision tree worked the best, and forest works by combining multiple decision trees, each trained on a different subset of the data, and then making predictions based on the majority vote of the trees. This ensemble approach can reduce overfitting and improve the generalization of the model. In this case, a single decision tree may be able to capture the underlying structure of the data well enough to achieve good performance.

To summarize, although we didn't create a flawless model for predicting the stock price trend, we did discover some correlation between the Twitter factor (with most models having an accuracy greater than 50%, surpassing the probability of a coin flip), and we also gained valuable insights on data generation and model building.