**Stock Market Forecasting using Machine Learning & Sentiment Analysis**

**Subham Bansal**

[**s03bans@gmail.com**](mailto:s03bans@gmail.com)

**B Tech, Electronics and Electrical Communication Engg.**

**Minor, Computer Science and Engineering**

**IIT Kharagpur**

1. **Introduction**

Stock price forecasting is a popular and important topic in financial and academic studies. Time series analysis is the most common and fundamental method used to perform this task. In the long run it might not be possible to outplay the market using a simple backward looking statistical model, but in the short run intelligent estimates based on model and subject matter expertise could prove to be helpful.

There are two main schools of thought in the financial markets, technical analysis and fundamental analysis. Fundamental analysis attempts to determine a stock’s value by focusing on underlying factors that affect a company’s actual business and its future prospects. Fundamental analysis can be performed on industries or the economy as a whole. Technical analysis, on the other hand, looks at the price movement of a stock and uses this data to predict its future price movements.

In this study, we first fit an autoregressive moving average (ARMA) model to better understand the data and predict the future variation in it. We then look at technical indicators of stock market’s past data, to identify patterns using Machine Learning models like **Random Forests, Decision Trees and SVM**. This is then extended to plugging **a Sentiment analysis** system, which analyses the sentiment of tweets about a firm on a specific day, which improves the prediction accuracy of the system.

1. **Efficient Market Hypothesis**

The basic theory regarding stock price forecasting is the **Efficient Market Hypothesis** (EMH)[1], which asserts that the price of a stock reflects all information available and everyone has some degree of access to the information. There are three variants of the hypothesis: "weak", "semi-strong", and "strong" form:

* **Weak form efficient**: The weak-form efficiency or random walk would be displayed by a market when the consecutive price changes (returns) are uncorrelated. This implies that any past pattern of price changes are unlikely to repeat by itself in the market. Hence, technical analysis that uses past price or volume trends do not to help achieve superior returns in the market [2].
* **Semi-Strong efficient**: The semi-strong form efficiency implies that all the publicly available information gets reflected in the prices instantaneously. The hypothesis suggests that only information that is not publicly available can benefit investors seeking to earn abnormal returns on investments [2].
* **Strong efficient**: Such efficiency would imply that both publicly available information and privately (non-public) available information are fully reflected in the prices instantaneously and no one can earn excess returns. A test of strong form efficiency would be to ascertain whether insiders of a firm are able to make superior returns compared to the market [2].

The random walk hypothesis is a financial theory stating that stock market prices evolve according to a random walk and thus cannot be predicted. It is consistent with the efficient-market hypothesis.

The implication of EMH is that the market reacts instantaneously to news and no one can outperform the market in the long run. However the degree of market efficiency is controversial and many believe that one can beat the market in a short period of time.  Some of the common criticisms of the theory are:

* **Behavioural Imperfections:** Behavioural economists attribute the imperfections in financial markets to a combination of cognitive biases such as overconfidence, overreaction, representative bias, information bias, and various other predictable human errors in reasoning and information processing.
* **Reversal of Returns**: Over a specific time, stocks which were ‘losers’ (giving poor returns) until now, have overtaken ‘winners’ (giving higher returns). This tendency of returns to reverse over long horizons indicates that Losers would have to have much higher betas than winners in order to justify the return difference. The study showed that the beta difference required to save the EMH is just not there [3].
* **Beating the Market**: Many of the successful portfolio managers and market investors, like Peter Lynch and Warren Buffet, have beaten the market consistently. If EMH were true, nobody would be able to outperform the market.

1. **Approach**

The stock considered is of **Infosys** (NSE: INFY). Infosys is an Indian multinational corporation that provides business consulting, information technology and outsourcing services. On 15 February 2015, its [market capitalisation](https://en.wikipedia.org/wiki/Market_capitalisation) was $42.51 billion, making it India's sixth largest publicly traded company.

The data is fetched from Yahoo Finance [4], and the window for the same is from **1st Jan, 2010 to 1st Jan, 2016 (6 year period).**

In this paper, we first apply the conventional ARMA time series analysis on the historical weekly stock prices and obtain forecasting results. This is followed by feature engineering to calculate and identify certain technical indicators like Relative Strength Index(RSI), Moving Average Convergence Divergence (MACD) etc, which is input to a Machine Learning system, testing with various algorithms like Naïve Bayes, SVM and Random Forests.

Finally, we use Stanford NLP’s state-of-the-art Sentiment Analysis package to judge the sentiments of tweets about the firm. The tweets are gathered through the ‘twitteR’ package of R.

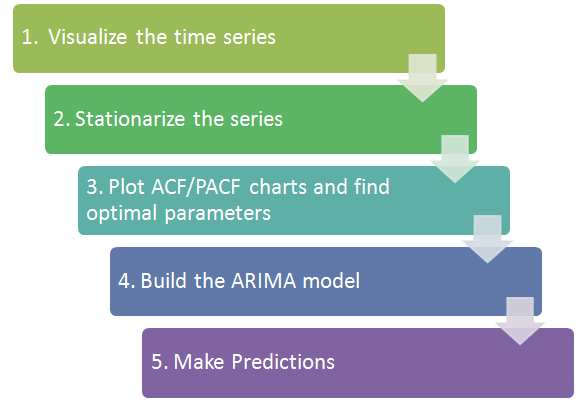
1. **ARMA forecasting**
2. **Introduction**

ARMA models are commonly used in time series modelling. The AR part of ARMA indicates that the evolving variable of interest is regressed on its own lagged (i.e., prior) values. The MA part indicates that the regression error is actually a linear combination of error terms whose values occurred contemporaneously and at various times in the past. ARMA has been explored in literature quite exhaustively [5],[6],[7].

**AR(1):** x(t) = alpha \*  x(t – 1) + error (t)

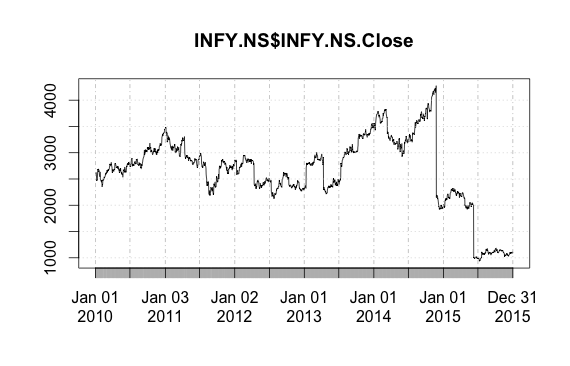
**MA**: x(t) = beta \*  error(t-1) + error (t)

1. **Steps**

****

**Exhibit 1:** ARMA Steps[8]

1. **Visualisation**

****

**Exhibit 2:** INFY Stock price (1st Jan, 10 – 1st Jan, 16)

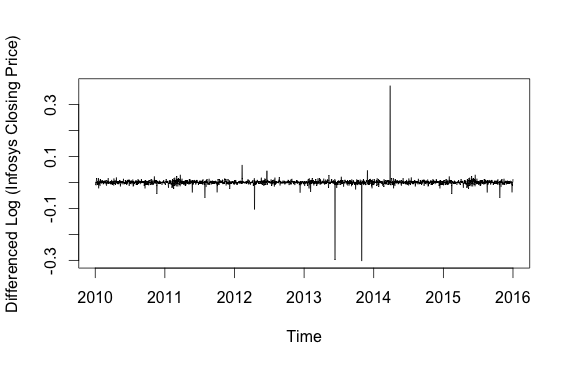
1. **Stationarizing the series**

ARMA is not applicable on non-stationary series. Since from the plot above, we see that the stock price is a also a non-stationary series (mean varies with time), we first need to stationarize it by differencing. We perform 1st order differencing to make the data **stationary on mean** using the following formula:

**Y’t = Yt – Yt-1**

To make the data **stationary on variance**, one of the best ways is through transforming the original series through **log transform** [9].

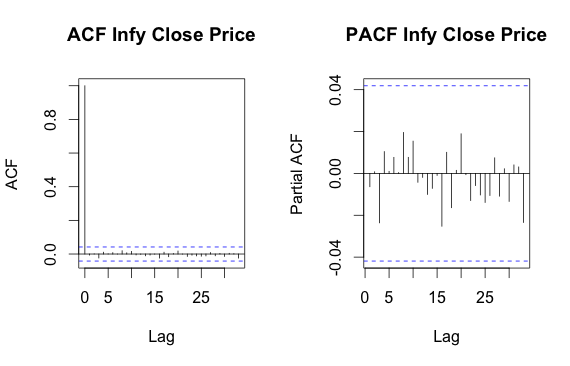
**Ynewt = log10(Yt)**

****

**Exhibit 3:** Stationarizing the series by differencing and log transform.

1. **ACF and PACF**

To identify the presence of AR and MA components in the stationary time series and to find out their orders, Autocorrelation and Partial Autocorrelation functions are used.

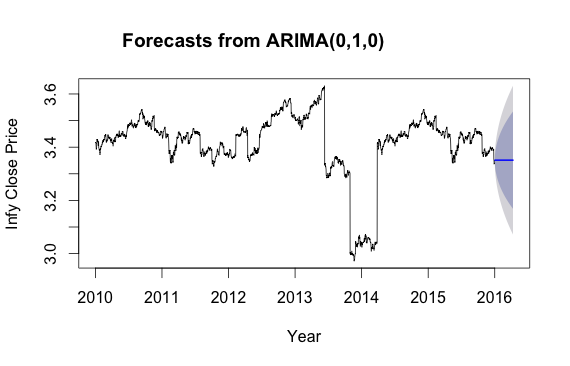


**Exhibit 4:** ACF & PACF

The single large spike outside the insignificant zone in ACF shows that this is mostly an AR(1) process.

1. **Building the model/Predictions**

With the parameters in hand, we build the ARMA model using the ‘auto.arima’ R function. The best fit model is selected based on Akaike Information Criterion (AIC) , and Bayesian Information Criterion (BIC) values. The model is used to predict stock price range for the rest of 2016.



**Exhibit 4:** ARIMA Forecasts

1. **Machine Learning Approaches**

The stock price prediction problem is a widely studied problem using machine learning based solutions. In our case, we model it as a **classification** problem, with the classes being

* **UP**: if next day’s closing price is more than that day’s opening price.
* **DOWN**: if next day’s closing price is less than that day’s opening price.

This is modelled using the alpha: **open(stock) – close(stock),** where, *open* and *close* are the open and close prices respectively.

The entire dataset consists of **1480** observations, divided into:

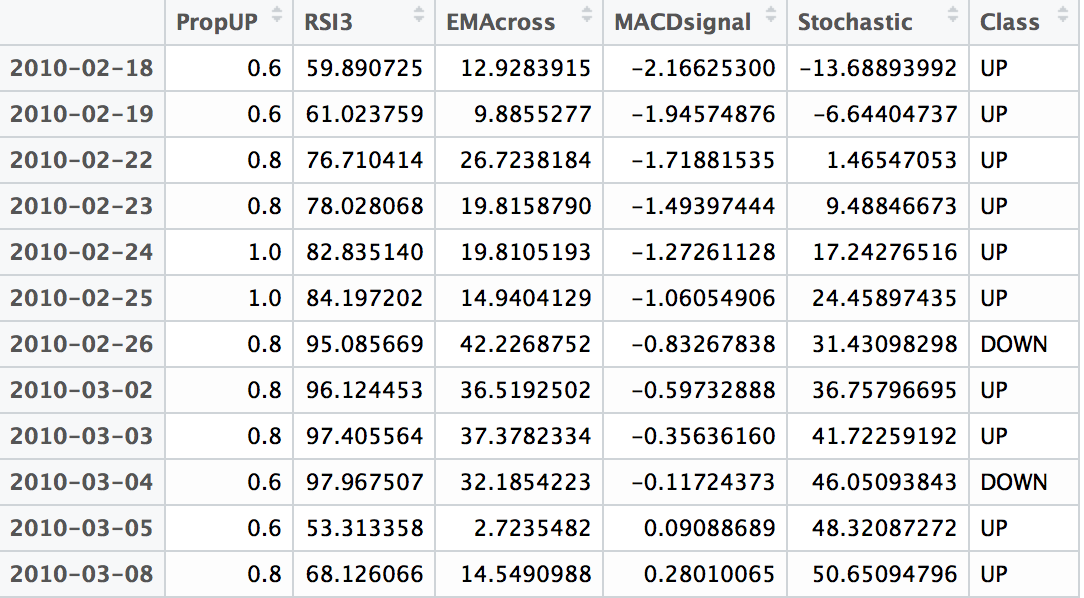
* **Training Set**, consisting of 1000 observations and “known” to the model.
* **Test Set**, consisting of 480 observations, unseen by the model, and on which the model will be tested for prediction correctness.

A machine learning system has 3 stages:

* Feature Engineering
* Training using ML algorithms
* Validating on test data.

1. **Feature Engineering: Stock Market Technical Indicators**

Certain indicators of the stock prices encapsulate information about its trend & momentum, and also the fundamentals (P/E ratio) The following indicators are used as features:

* **Proportion of past n-days up-closing**: In the past **n** days, what proportion of days saw an “UP” trend in the stock. We use n = 5.
* **Relative Strength Index (RSI)**: Helps traders determine if a stock or market is overbought or oversold.
* **Exponential Moving Average (EMA):** Average value of prices over a n-day period, with more weight given to recent prices.
* **Moving Average Convergence/Divergence (MACD):** Trend-following momentum indicator based on the difference between two EMAs
* **Stochastic Momentum Index (SMI)**: Shows how a stock's price is doing relative to past movements, specifically, where the current close has taken place relative to the midpoint of the recent high to low range****

**Exhibit 5:** Data Snapshot showing 5 types of features

1. **Models and Algorithms**

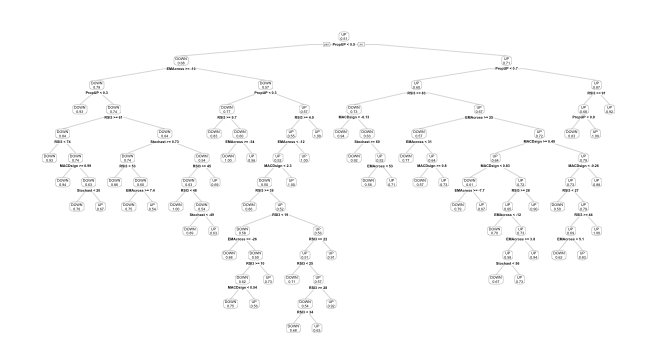
We use the following ML training algorithms:

1. **Decision Trees**

Decision trees divide input data according to features that maximise information gain (that is, make it easiest to classify when looking at it alone). The **rpart** package provides us with the functionality.

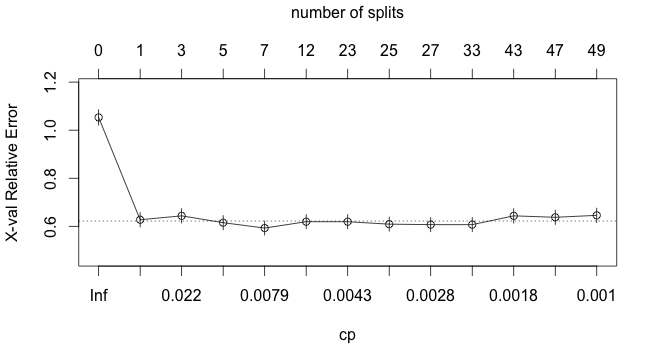
Decision trees can also be used to help select and evaluate indicators. Indicators that are closer to the top of the tree lead to more pure splits, and contain more information, than those towards the bottom of the tree. In our case, the **Stochastic Oscillator didn’t even make it onto the tree!**

An Unpruned tree consists of many nodes as shown below (Here, complexity parameter(cp) = 0.0079)



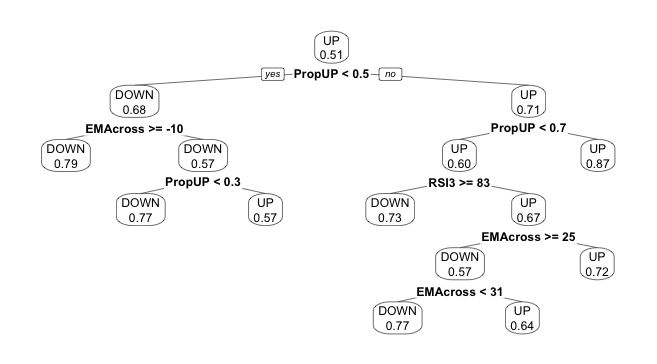
**Exhibit 6:** The Decision Tree; The most useful feature turns out to be proportion of UP closings.

The Complexity parameter is chosen so as to minimise the cross validation error.



**Exhibit 7:** The Decision Tree cross validation error; minimum at cp = 0.0079

Its necessary to prune the tree to avoid overfitting the training data.



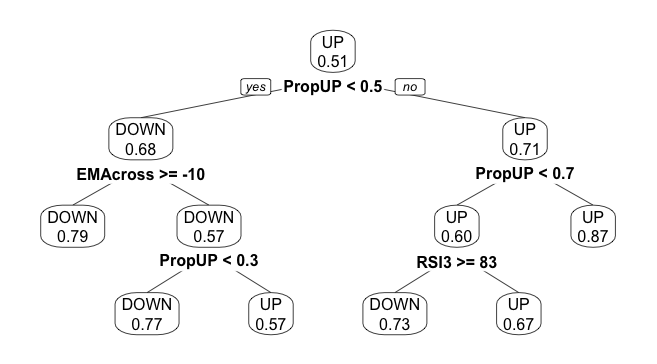
**Exhibit 8:** The Pruned Decision Tree; cp = 0.0079; MACD is eliminated

We see that MACD & SMI are not deemed important and thus are “pruned” out as filtering parameters.

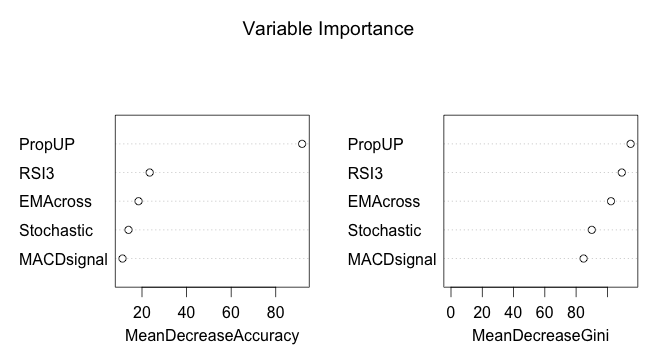
1. **Random Forests**

**Random forests** are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees [10]. This tends to reduce overfitting.

The R package used is ‘randomForest’. The number of trees in the forest was set at 500.



**Exhibit 9:** A Sample random forest tree, pruned.

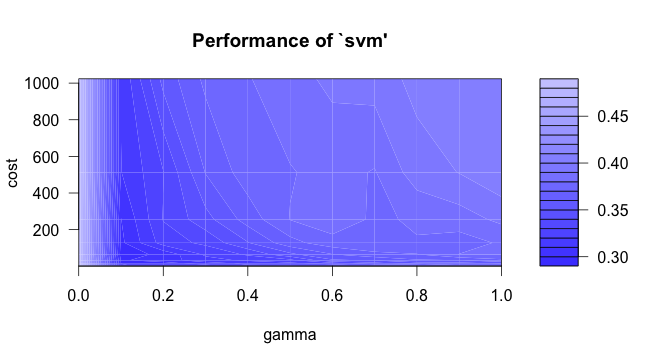


**Exhibit 10:** Variable Importance Plot in RandomForest; PropUp and RSI are the most informational indicators

1. **Support Vector Machines**

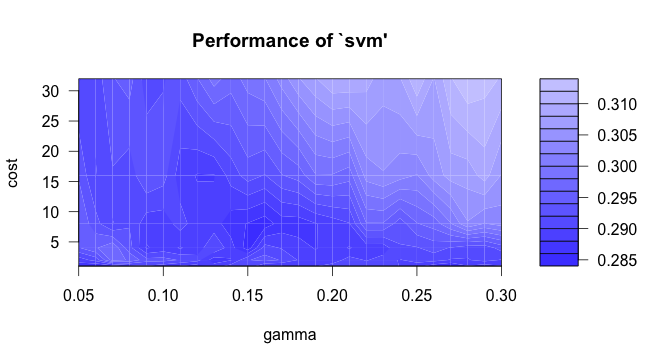
An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on [11].

The fitting model is soft-margin SVM with an **RBF kernel**. We thus have 2 parameters to optimise: **regularisation cost parameter ‘C’ and RBF parameter ‘γ’.** In R, we can tune the algorithm over multiple values of both to find out the optima. This is done by plotting the svm cross-validation performance against these values.



**Exhibit 11:** SVM performance for different values of gamma and cost(C); Evidently, region of interest is **γ**<0.3

The **darker** the region, the **better** the performance is. Thus, we find our region of interest to be **0.05 < γ < 0.3**, and cost, **C < 32 (**Optimal C comes out to be 8, so a lower range is being preferred**).** We can then fine tune it further using these ranges.



**Exhibit 12:** SVM performance for a more precise region of gamma & cost (**0.05** < γ < **0.3**, and C < **32**)

Optimal Parameters found are:

|  |  |
| --- | --- |
| **Parameters** | **Value** |
| **Gamma, γ** | 0.15 |
| **Cost, C** | 4 |
| **Mean Square error(MSE)** | 0.2848812 |

1. **Results: Validating on test data**

The algorithms are used for predictions on a test data. The **confusion matrix** is plotted and the evaluation metrics used are **Precision, Recall And Accuracy.**

A confusion matrix is of the form:

|  |  |  |
| --- | --- | --- |
|  | ACTUAL positive | ACTUAL negative |
| Predicted positive | True Positive(TP) | False Positive (FP) |
| Predicted negative | False Negative (FN) | True Negative (TN) |

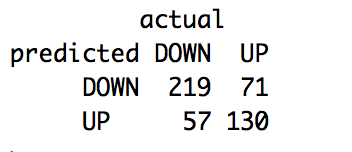
**Exhibit 13:** Confusion Matrix

**Precision:** TP/(TP+FP)

**Recall:** TP/(TP+FN)

**Accuracy:** (TP+TN)/(TP+TN+FP+FN)

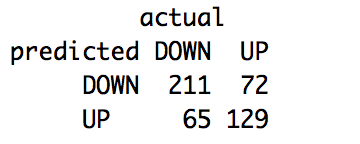
1. **Decision Trees**

****

**Exhibit 14:** Confusion Matrix (Decision Tree)

|  |  |
| --- | --- |
| **Metrics** | **Value** |
| **Precision** | **75.52%** |
| **Recall** | **79.35%** |
| **Accuracy** | **73.17%** |

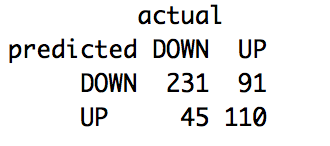
1. **Random Forests**

****

**Exhibit 15:** Confusion Matrix (Random forest)

|  |  |
| --- | --- |
| **Metrics** | **Value** |
| **Precision** | **74.83%** |
| **Recall** | **77.54%** |
| **Accuracy** | **71.91%** |

1. **SVM**

****

**Exhibit 16:** Confusion Matrix (SVM)

|  |  |
| --- | --- |
| **Metrics** | **Value** |
| **Precision** | **71.74%** |
| **Recall** | **83.70%** |
| **Accuracy** | **71.49%** |

Thus, whereas a **simple decision tree gives the best accuracy and precision**, **SVM gives the highest recall**, among the 3.

1. **Augmenting the above by Sentiment Analysis of News**

Millions of tweets[12] are generated everyday, about various topics, news & events. News and announcements have a profound impact on the stock prices in a day, and almost all major news sources are hugely proactive via twitter. All kinds of news have 2 parts:

**News = Expected + Surprise elements**

The “surprise elements” are the ones which are responsible for volatility in a stock’s price.

Thus, if the news and events about a firm are tracked in real time, and their effects on the stock price are predicted (“good” or “bad” for it), a significant headstart can be obtained in the market by the procurer of this information. The idea is that prices don’t instantaneously reflect the new “surprise” information, but show some delay in doing so. Thus, **technological advantage**, which can process new info in milliseconds, can have a profound impact on the trading strategies of firms as well as individuals.

We extend our Machine Learning system above to use an additional feature: **Sentiment score of news-based tweets about a firm**. The steps in computing this are as follows:

1. All the news tweets about a firm posted on a specific day are mined, based on news agencies like Reuters, PTI, Bloomberg, IIFL, Economic Times etc.
2. A new incoming tweet is checked for redundancy – whether it has already been tracked.
3. Stanford Natural Language Processing group’s (**Stanford NLP**) **Sentiment Analysis** tool is used to classify each new incoming tweet as **positive** or **negative**.
4. A weight of **alpha** is assigned to each positive tweet.
5. Sentiment score for each day is computed as

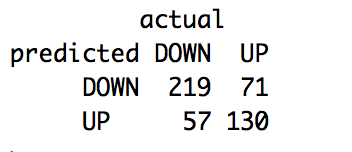
**Sentiment Score (κ) =**

According to **Prospect Theory** [13], negative information has a much greater impact on individuals’ attitudes than does positive information . Hence, it is more reasonable to assign a slightly larger initial absolute value to negative news than to positive news, which leads to our **choice of alpha as below 0.5** (we take alpha as 0.4). The study can be extended to tuning the value of alpha according to one’s attitude towards risk, or one which maximises any of the evaluation criteria.

Although, we have done an absolute categorisation of tweets, a lot of the time, the judgment of whether a piece of news is good or bad is subjective. People might have different, even opposite, interpretations of the same information, which poses a challenge to the sentiment analysis applications.

The Sentiment scores for each day for INFY are computed and a new column is added to the dataset. The aforementioned algorithms are again applied and evaluated on this modified dataset:

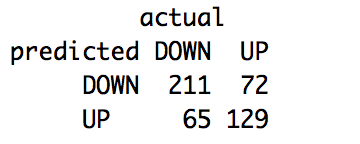
1. **Decision Trees**

****

**Exhibit 14:** Confusion Matrix (Decision Tree)

|  |  |
| --- | --- |
| **Metrics** | **Value** |
| **Precision** | **81.12%** |
| **Recall** | **84.68%** |
| **Accuracy** | **80.19%** |

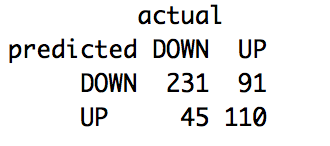
1. **Random Forests**

****

**Exhibit 15:** Confusion Matrix (Random forest)

|  |  |
| --- | --- |
| **Metrics** | **Value** |
| **Precision** | **78.11%** |
| **Recall** | **83.64%** |
| **Accuracy** | **76.47%** |

1. **SVM**

****

**Exhibit 16:** Confusion Matrix (SVM)

|  |  |
| --- | --- |
| **Metrics** | **Value** |
| **Precision** | **81.16%** |
| **Recall** | **88.29%** |
| **Accuracy** | **79.44%** |

Thus, clearly, **incorporating news sentiment information has boosted the accuracies** and other metrics of all the models, with **decision tree outperforming** the rest with an **80.19%** accuracy. This points to a strong correlation between stock price movement and News Sentiment Score.

1. **Future Work**

Simulation of the models with a starting sum of money for a specific time period is to be explored. Also, inclusion of fundamental indicators like EPS, cash position, profit growth and others may be studied. Besides, various other algorithms including **deep learning** algorithms is to be studied and applied to our situation.

1. **References**
2. *Efficient Capital Markets: A Review of Theory and Empirical Work*, Fama, 1969
3. *Fundamentals of Corporate Finance*, Ross, Westerfield, Jordan, 6th edition
4. *"Does the Stock Market Overreact"*. *Journal of Finance*. **40**: 793–805; DeBondt, Werner F.M., Thaler, Richard H., 1985
5. Yahoo Finance, <https://in.finance.yahoo.com/q/hp?s=INFY.BO>
6. *Hypothesis testing in time series analysis*, George E. P. Box and Gwilym Jenkins, 1971
7. Multiple time series. Wiley series in probability and mathematical statistics. New York: John Wiley and Sons; Hannan, Edward James*,* 1970
8. *Time Series: Theory and Methods* (2nd ed.). New York: Springer. p. 273. ISBN 9781441903198; Brockwell, P. J.; Davis, R. A., 2009
9. Analytics Vidhya, https://www.analyticsvidhya.com/wp-content/uploads/2015/02/flowchart.png
10. *Stationarity and Differencing*, <http://people.duke.edu/~rnau/411diff.htm>
11. [*Random Decision Forests*](http://ect.bell-labs.com/who/tkh/publications/papers/odt.pdf), Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282; Ho, Tin Kam, 1995
12. "Support-vector networks".Machine Learning.**20**(3): 273–297; Cortes, C.; Vapnik, V., 1995
13. Twitter Inc., www.twitter.com; <http://www.internetlivestats.com/twitter-statistics/>
14. *Prospect Theory: An Analysis of Decision under Risk*, Econometrica, Vol. 47, No. 2. (Mar., 1979), pp. 263-292; Daniel Kahneman, Amos Tversky, 1979