Stock Market Analysis using Market and News Data

Sheryl Mathew Jaswant Naidu Tejas Ghanwat

EECS
Washington State University
Pullman, WA-99164

Abstract

Stock Market prediction is one of the major applications of Data mining and Machine Learning in today's market. Various applications and websites make use of machine learning algorithms to predict stock prices. The predictions made by these applications depends on both the algorithm used to develop the model as well as the dataset used to train the model. The range of datasets includes data extracted from stock market, news, history, web data, social media, etc. We have used a combination of financial data from stock market and news articles on companies such as Apple Inc. Adobe Systems, Johnson and Johnson etc. to predict the stock market after 10 days. Exploratory Data Analysis (EDA), Principal Component Analysis, Feature Engineering are performed on the news and the stock market data. Then predictions are made using linear regression, random forest classifier and support vector regression.

1 Introduction

Traders and investors previously used only the stock market data and tried to predict the behavior of the stocks. Though this was efficient, now with the availability of information from different sources like news articles, reports, editorials, twitter and other social media it has become easier to predict the trend and even the crash of a particular stock. But it is not possible for humans to analyze the millions of information that they have at their disposal so instead they use machines to analyze this information which enables them to take a well-informed decision.

The news articles that we consume may not explicitly mention market trends but may give a sense of it on some way. The data miners who deal with extracting this data have to deal with two challenges, first one is related to the form and function of the news which pertain to the different type of news reporting and intended news readership, the changing patterns of news consumption and distribution, the growing use of social media and the use of visual and audio news form. The second challenge is related to assessing and identifying appropriate news sources for text mining research, which pertains to gaining access to news text data in compliance with copyright and licensing law, and ensuring the needed news text is complete, fully searchable and available for the years or decades required of the research question.

Data collection is often regarded as the easiest step of the entire analytical process, yet it is actually one of the most complex, with the quality of collection and preparation process impacting every other stage of the analysis process. To get an insight and what features to use while modelling we have chosen some approaches such as Exploratory Data Analysis (EDA), Principal Component Analysis and Feature Engineering. We have visualized some possible combinations of features and tried concluding relationships among the features to use them while modelling. As they say, well begun is half done, getting the right data for building the model can prevent lot of upcoming complications and assure efficient model performance. Making actual predictions from news data will include machine learning algorithms and lot of other things. We have tried to show that using news and market data in combination has shown to improve the prediction process as compared to only using the market data.

2 Problem Definition

2.1 Questions Explored

1. **What features to use from the news and market datasets individually?**

   There are 17 features in Market dataset and 36 features in News dataset. It is important to find out which features are most suitable to predict the stock market after 10 days. This is achieved through Feature engineering.

2. **Can we gather more information from the available dataset?**

   From the given data we were able to extract other information like volume_to_mean which is each market volume divided by the mean of all the market values, close_to_open which is market close value divided by their corresponding market open value, asset_sentiment_count was found from news data among others.

3. **Which regression to use Random Forest, Linear Regression or Support Vector Regression?**

   We implemented all the above methods and compared their RMSE values to see which was the better model..

4. **Which technique to use in Linear regression – Linear model, Generalized linear model or Generalized Least Squares?**

   We tried all the techniques for the linear regression model and compared the R-squared values to see which was the better technique.

5. **What kind of interactions between the individual variables will best suit the linear regression model and the technique selected?**

   We used a library called glmulti which searches over all the possible interactions between the variables and gives the top n models.

2.2 Difficulties faced

1. **Computational difficulties**

   The size of the dataset is around 8 gigabytes which delays the approach towards the goal and also to visualize the relationship among the features. The hardware available to overcome the computational difficulty is not feasible.

   ```
   [1] "Dimensions of Market"
   [1] 4072956     17
   [1] "Dimensions of News"
   [1] 9328750     36
   ```

   As mentioned above these were the dimensions of the dataset that are used in the project. Working with such a dataset with a lower level hardware was a problem that triggered many other obstacles. So, we have chosen a subset of the data to overcome this problem.

2. **Data Sparsity**

   On a further note the datasets contain a lot of null values but omitting these null values is an ethical problem while training the model. On observing the data there are many features in the dataset, but we can only relate a few features with the target variable. This makes it difficult to build a model and gain a better result from that trained model

3 Solution Approach

3.1 Exploratory Data Analysis

Visualizing the data considering few features at each instance to figure out their role in the dataset. Plotting the different variables from the dataset also considering their frequency to check their relationship with the target variable. Some forms of EDA include histograms, barplots, ggplots, qplots. All the graphs that were generated while performing EDA have been included in the below section.

3.2 Feature Engineering

The process of extracting useful features is done through feature engineering. This is achieved by plotting histograms, frequency and density graphs after which they are analyzed and based on their interpretations features selected.

3.3 Principal Component Analysis

Principal Component Analysis is a dimensionality reduction method which is used for feature extraction. This is done by combining the input variables in a specific manner and finding the most important variables so that we can drop the unimportant ones. All the variables after PCA are independent of each other. Example for PCA in deep learning is document image analysis [12].

3.4 Linear Regression [4]

Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data. One variable is considered to be an explanatory variable, and the other is considered to be a dependent variable.

A linear regression line has an equation of the form $\boldsymbol{Y = a + bX}$, where $\boldsymbol{X}$ is the explanatory variable and $\boldsymbol{Y}$ is the dependent variable. The slope of the line is $\boldsymbol{b}$, and $\boldsymbol{a}$ is the intercept (the value of $\boldsymbol{y}$ when $\boldsymbol{x} = 0$).

3.4.1 Linear Model [3]

Linear models describe a continuous response variable as a function of one or more predictor variables. They can help you understand and predict the behavior of complex systems or analyze experimental, financial, and biological data.

Linear regression is a statistical method used to create a linear model. The model describes the relationship between a dependent variable y (also called the response) as a function of one or more independent variables X_i (called the predictors). The general equation for a linear model is:

$$y = \ \beta_0 + \sum \ \beta_i X_i + \epsilon_0$$

3.4.2 Generalized linear model [2]

The term *general linear model* (GLM) usually refers to conventional linear regression models for a continuous response variable given continuous and/or categorical predictors. It includes multiple linear regression, as well as ANOVA and ANCOVA (with fixed effects only). The form is $y_i \sim N(x, \ T_i, \ \beta, \sigma_2)$, where x_i contains known covariates and β contains the coefficients to be estimated. These models are fit by least squares and weighted least squares using, for example: SAS Proc GLM or R functions lsfit() (older, uses matrices) and lm() (newer, uses data frames).

3.4.3 Generalized Least Squares [4]

The most common method for fitting a regression line is the method of least-squares. This method calculates the best-fitting line for the observed data by minimizing the sum of the squares of the vertical deviations from each data point to the line (if a point lies on the fitted line exactly, then its vertical deviation is 0). Because the deviations are first squared, then summed, there are no cancellations between positive and negative values.

3.5 Random Forest Classifier [7]

We assume that the user knows about the construction of single classification trees. Random Forests grows many classification trees. To classify a new object from an input vector, put the input vector down each of the trees in the forest. Each tree gives a classification, and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest).

Each tree is grown as follows:

1. If the number of cases in the training set is N, sample N cases at random - but *with replacement*, from the original data. This sample will be the training set for growing the tree.
2. If there are M input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing.
3. Each tree is grown to the largest extent possible. There is no pruning.

3.6 Support Vector Regression [9]

Support Vector Machine can also be used as a regression method, maintaining all the main features that characterize the algorithm (maximal margin). The Support Vector Regression (SVR) uses the same principles as the SVM for classification, with only a few minor differences. First of all, because output is a real number it becomes very difficult to predict the information at hand, which has infinite possibilities. In the case of regression, a margin of tolerance (epsilon) is set in approximation to the SVM which would have already requested from the problem. But besides this fact, there is also a more complicated reason, the algorithm is more complicated therefore to be taken in consideration. However, the main idea is always the same: to minimize error, individualizing the hyperplane which maximizes the margin, keeping in mind that part of the error is tolerated.

4 Implementation

4.1 Dataset

The Market data was provided by Intrinio and the News data was provided by Thomson Reuters for the Kaggle Challenge – Two Sigma: Using News to Predict Stock Movements. This challenge has been hosted by

4.1.1 Market Data

- time(datetime64[ns, UTC]) - the current time (in marketdata, all rows are taken at 22:00 UTC)
- assetCode(object) - a unique id of an asset
- assetName(category) - the name that corresponds to a group of assetCodes. These may be "Unknown" if the corresponding assetCode does not have any rows in the news data
- universe(float64) - a boolean indicating whether or not the instrument on that day will be included in scoring. This value is not provided outside of the training data time period. The trading universe on a given date is the set of instruments that are available for trading (the scoring function will not consider instruments that are not in the trading universe). The trading universe changes daily

- volume(float64) - trading volume in shares for the day
- dayclose(float64) - the close price for the day (not adjusted for splits or dividends)open(float64) - the open price for the day (not adjusted for splits or dividends)
- returnsClosePrevRaw1(float64) - see returns explanation above
- returnsOpenPrevRaw1(float64) - see returns explanation above
- returnsClosePrevMktres1(float64) - see returns explanation above
- returnsOpenPrevMktres1(float64) - see returns explanation above
- returnsClosePrevRaw10(float64) - see returns explanation above
- returnsOpenPrevRaw10(float64) - see returns explanation above
- returnsClosePrevMktres10(float64) - see returns explanation above
- returnsOpenPrevMktres10(float64) - see returns explanation above
- returnsOpenNextMktres10(float64) - 10 day, market-residualized return. This is the target variable used in competition scoring. The market data has been filtered such that returnsOpenNextMktres10 is always not null.

4.1.2 News Data

- time(datetime64[ns, UTC]) - UTC timestamp showing when the data was available on the feed (second precision)
- sourceTimestamp(datetime64[ns, UTC]) - UTC timestamp of this news item when it was created
- firstCreated(datetime64[ns, UTC]) - UTC timestamp for the first version of the item
- sourceId(object) - an Id for each news item headline(object) - the item's headline
- urgency(int8) - differentiates story types (1: alert, 3: article)
- takeSequence(int16) - the take sequence number of the news item, starting at 1. For a given story, alerts and articles have separate sequences
- provider(category) - identifier for the organization which provided the news item (e.g. RTRS for Reuters News, BSW for Business Wire)
- subjects(category) - topic codes and company identifiers that relate to this news item. Topic codes describe the news item's subject matter. These can cover asset classes, geographies, events, industries/sectors, and other types
- audiences(category) - identifies which desktop news product(s) the news item belongs to. They are typically tailored to specific audiences. (e.g. "M" for Money International News Service and "FB" for French General News Service)
- bodySize(int32) - the size of the current version of the story body in characters
- companyCount(int8) - the number of companies explicitly listed in the news item in the subjects field
- headlineTag(object) - the Thomson Reuters headline tag for the news item
- marketCommentary(bool) - boolean indicator that the item is discussing general market conditions, such as "After the Bell" summaries

- sentenceCount(int16) - the total number of sentences in the news item. Can be used in conjunction with firstMentionSentence to determine the relative position of the first mention in the item
- wordCount(int32) - the total number of lexical tokens (words and punctuation) in the news item
- assetCodes(category) - list of assets mentioned in the item
- assetName(category) - name of the asset
- firstMentionSentence(int16) - the first sentence, starting with the headline, in which the scored asset is mentioned
- relevance(float32) - a decimal number indicating the relevance of the news item to the asset. It ranges from 0 to 1. If the asset is mentioned in the headline, the relevance is set to 1. When the item is an alert (urgency == 1), relevance should be gauged by firstMentionSentence instead
- sentimentClass(int8) - indicates the predominant sentiment class for this news item with respect to the asset. The indicated class is the one with the highest probability
- sentimentNegative(float32) - probability that the sentiment of the news item was negative for the asset
- sentimentNeutral(float32) - probability that the sentiment of the news item was neutral for the asset
- sentimentPositive(float32) - probability that the sentiment of the news item was positive for the asset
- sentimentWordCount(int32) - the number of lexical tokens in the sections of the item text that are deemed relevant to the asset. This can be used in conjunction with wordCount to determine the proportion of the news item discussing the asset
- noveltyCount12H(int16) - The 12 hour novelty of the content within a news item on a particular asset. It is calculated by comparing it with the asset-specific text over a cache of previous news items that contain the asset
- noveltyCount24H(int16) - same as above, but for 24 hours
- noveltyCount3D(int16) - same as above, but for 3 days
- noveltyCount5D(int16) - same as above, but for 5 daysnoveltyCount7D(int16) - same as above, but for 7 days
- volumeCounts12H(int16) - the 12 hour volume of news for each asset. A cache of previous news items is maintained and the number of news items that mention the asset within each of five historical periods is calculated
- volumeCounts24H(int16) - same as above, but for 24 hours
- volumeCounts3D(int16) - same as above, but for 3 days
- volumeCounts5D(int16) - same as above, but for 5 days
- volumeCounts7D(int16) - same as above, but for 7 days

4.2 Features Selected

After feature engineering we got a set of features. So considering only the market data we took a combination of features and then combining both market and data we selected a combination of features.

4.2.1 Market Data

Combination 1: volume, returnsClosePrevRaw1, returnsOpenPrevRaw1, returnsClosePrevRaw10, open, close

Combination 2: volume_to_mean, returnsOpenPrevRaw1_to_volume, close_to_open

Combination 3: volume_to_mean, returnsOpenPrevRaw1_to_volume, close_to_open, volume, returnsClosePrevRaw1, returnsOpenPrevRaw1, returnsClosePrevRaw10, open, close

4.2.2 Market and News Data

Combination 1: volume_to_mean, returnsOpenPrevRaw1_to_volume, close_to_open, asset_codes_len, sentence_word_count, headlines_len, asset_sentiment_count, headline_tag_t

Combination 2: volume_to_mean, returnsOpenPrevRaw1_to_volume, close_to_open, sentence_word_count, headlines_len, asset_sentiment_count

4.3 Regression Methods

The different combination of data shown in the above section are trained over Linear Regression models, Random Forest Classifiers and Support Vector Machines to see their performance. Different parameters are calculated for each type of model and corresponding graphs are plotted to better understand the output of the model.

4.4 Regression Tools

4.4.1 Library: glmulti [11]

It is a R package for automated model selection and multi-model inference with glm, lm and gls. From a list of explanatory variables, the provided function glmulti builds all possible unique models involving these variables and, optionally, their pairwise interactions. Models are fitted with standard R functions like glm. The n best models and their support are returned, allowing model selection and multi-model inference through standard R functions. The package is optimized for large candidate sets by avoiding memory limitation, facilitating parallelization and providing, in addition to exhaustive screening, a compiled genetic algorithm method.

4.4.2 Library: randomforest [10]

It is a R package which implements Breiman's random forest algorithm (based on Breiman and Cutler's original Fortran code) for classification and regression.

4.4.3 Library: e1071 [1]

This R package contains the interface to perform support vector regression as well as tune the model to get better results.

4.5 Evaluation Criteria

4.5.1 R-squared [6]

R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression.

4.5.2 MAE

MAE measures the average magnitude of the errors in a set of predictions, without considering their direction. It's the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight.

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n} |y_j - \hat{y}_j|$$

4.5.3 MSE [5]

The mean squared error tells you how close a regression line is to a set of points. It does this by taking the distances from the points to the regression line (these distances are the "errors") and squaring them. The squaring is necessary to remove any negative signs. It also gives more weight to larger differences. It's called the mean squared error as you're finding the average of a set of errors.

4.5.4 RMSE [8]

The regression line predicts the average y value associated with a given x value. Note that is also necessary to get a measure of the spread of the y values around that average. To do this, we use the root-mean-square error (r.m.s. error). To construct the r.m.s. error, you first need to determine the residuals. Residuals are the difference

between the actual values and the predicted values. I denoted them by $y_i - \widehat{y_i}$, where y_i is the observed value for the ith observation and $\widehat{y_i}$ is the predicted value.

4.5.5 MAPE [?]

It is a simple average of absolute percentage errors. The MAPE calculation is as follows:

$$\mathrm{MAPE} = \frac{\Sigma \frac{|A-F|}{A} \times 100}{N}$$

4.5.6 AIC

The model fit (Akaike's Information Criterion value) is measured ask likelihood of the parameters being correct for the population based on the observed sample. The number of parameters is derived from the degrees of freedom that are left. AIC value roughly equals the number of parameters minus the likelihood of the overall model – Therefore the smaller the AIC value the better the model

4.6 Graphs

4.6.1 IC Profile

Information Criteria profile plot shows how the IC changes from the best model (the one with the highest IC) to the worse model (the one with the lowest IC).

4.6.2 Model Averaged Importance of Terms

This gives the importance of each interaction of variables, the individual variables with respect to each other and how much these variables contributed in making the prediction.

4.6.3 ROC for best model

ROC curve is used to display how a model distinguishes between the true positive and the true negative values. This is done by plotting sensitivity which is the probability of predicting that a positive value is predicted as positive against 1-specificity which is the probability of predicting that a negative value is predicted as positive

4.6.4 Performance of 'svm'

This graph is used to plot the grid search result that was performed while tuning the SVM. The RMSE values will be closer to zero in the regions where the region is darker which implies that the model is better in those regions.

4.7 Method implemented

Initially the data was analyzed through Exploratory Data Analysis. After which feature engineering was performed to extract the important features. Using the features extracted regression is done.

Principal Component Analysis is performed over the numerical columns of market and news data alone since PCA can be done only over numerical data. It is done using both prcomp and princomp methods. In prcomp, the calculation is done by a singular value decomposition of the data matrix. In princomp, the calculation is done using eigen on the correlation or covariance matrix. The summary of the data after applying prcomp and princomp is displayed. The scree plot and the component loadings which are the correlation coefficients between the variables and factors are used to perform PCA.

Word Cloud is generated based on the news headlines to see the most commonly occurring words in all the news headlines. This could be done only after cleaning the headlines by removing the punctuation, spaces and converting all the data to lowercase. After this stemming and tokenization is performed over the news headlines. This is done to create the corpus of words which is then used to generate the term document and document term matrix. Term document matrix is the crucial information needed to generate the word cloud.

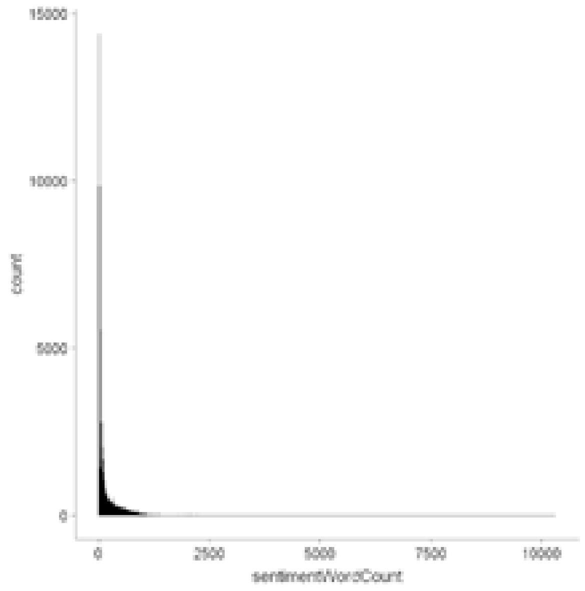
Using glmulti, linear regression is done with linear model lm, generalized linear model glm and generalized least squares gls techniques over each combination of data. After which the best model is selected and their different evaluation parameters like R-squared, MAE, MSE, RMSE are calculated. Then the top 5 models are displayed with all their evaluation parameters. Also, the IC profile was generated along with the Model-averaged importance of terms graph as well as ROC curve for the best model.

Then Random Forest is used for regression using the same combination of variables. Then Mean of squared residuals is evaluated along with the summary of the model displayed. IncNodePurity is used to give the estimation of variable importance.

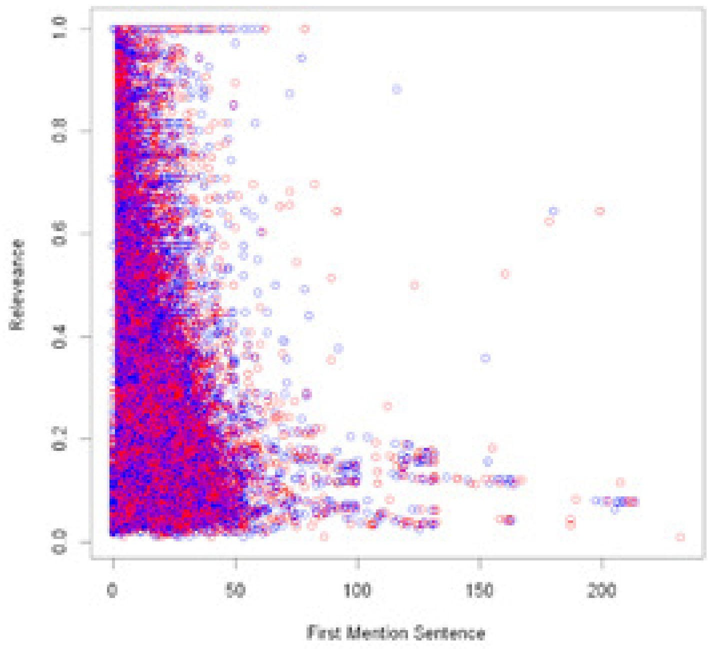
Support Vector Machine is used to perform regression over the same combination of variables. The RMSE values are calculated for svm model without tuning the parameters and for a svm model over which hyperparameter tuning has been performed. Their corresponding RMSE values are shown. For the tuned model the best hyper parameters are selected after performing 10-fold cross validation. Using the Performance of svm graph we can visualize where the model predicts better based on epsilon and cost.

5 Results

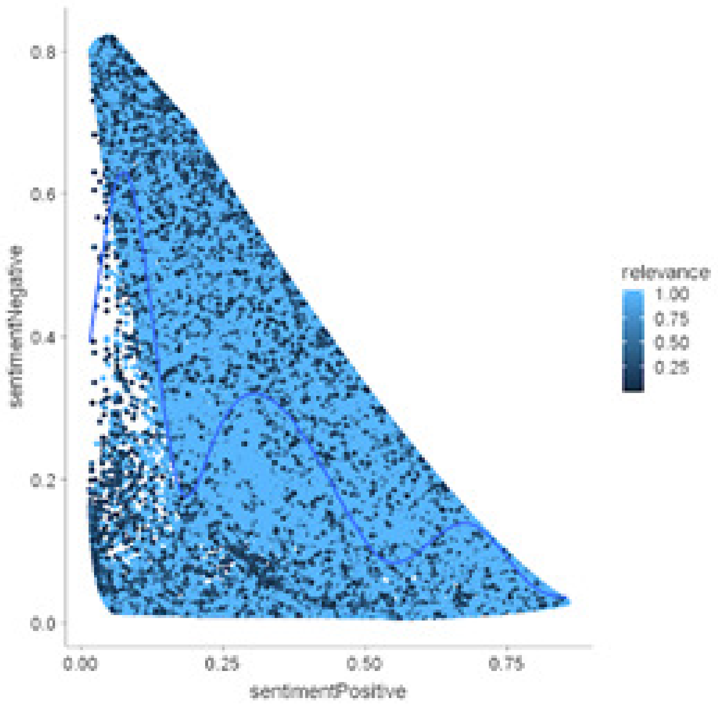
5.1 Exploratory Data Analysis and Feature Engineering



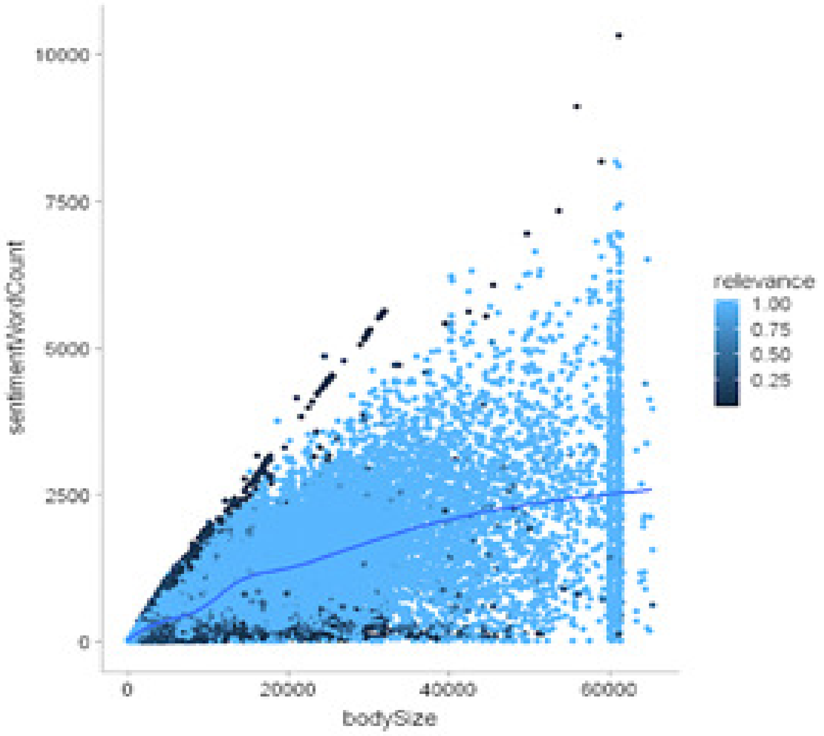
The above plot clearly shows that the sentiment word count falls mostly between 0-5.



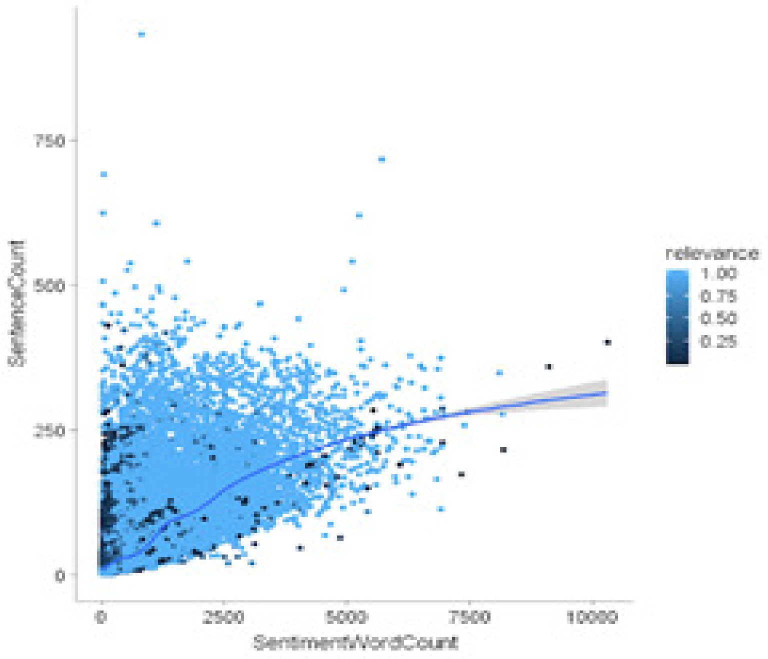
We can see that if A first mention sentence mentions assets scored in headlines, first sentence of body and so on, the news_all has high relevance. We can see highest relevance at 1,2... first mention sentence.



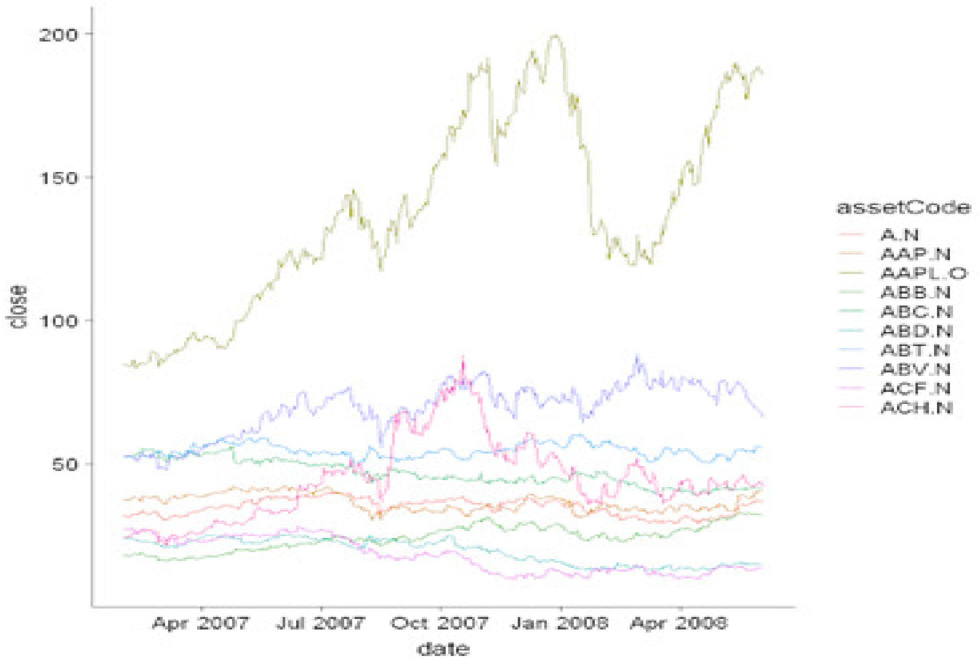
The above plot could easily pass for a beautiful picture of a snow covered mountain. When there is higher probability of the sentiment being positive, there is higher relevance and vice versa, when there is, probability of being a negative sentiment, the relevance is low.



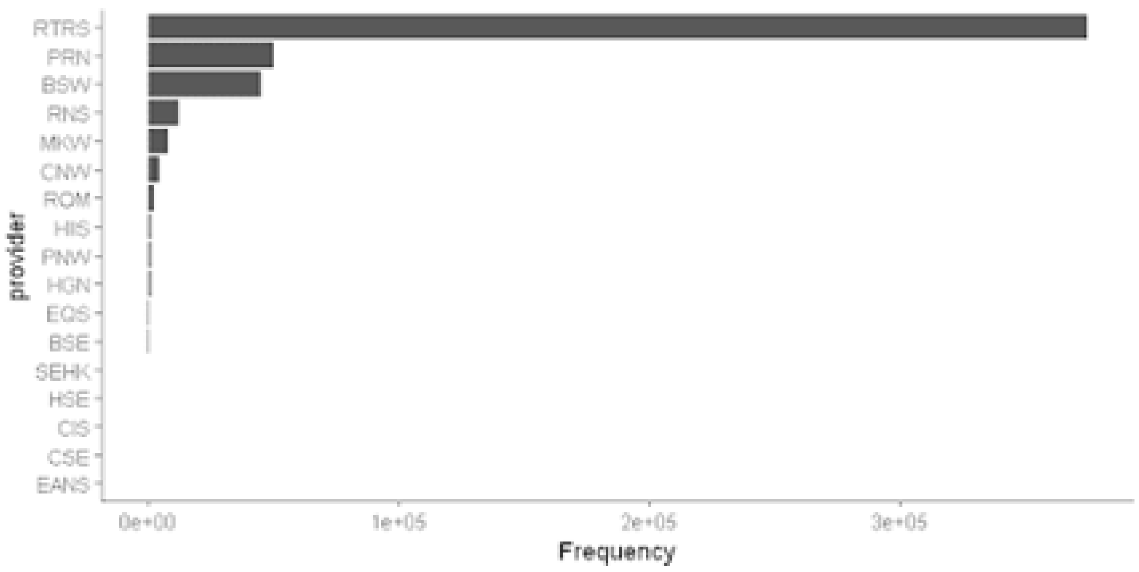
We can observe from the above graph that as the body size and the sentiment word count increases, there is drop in relevance.

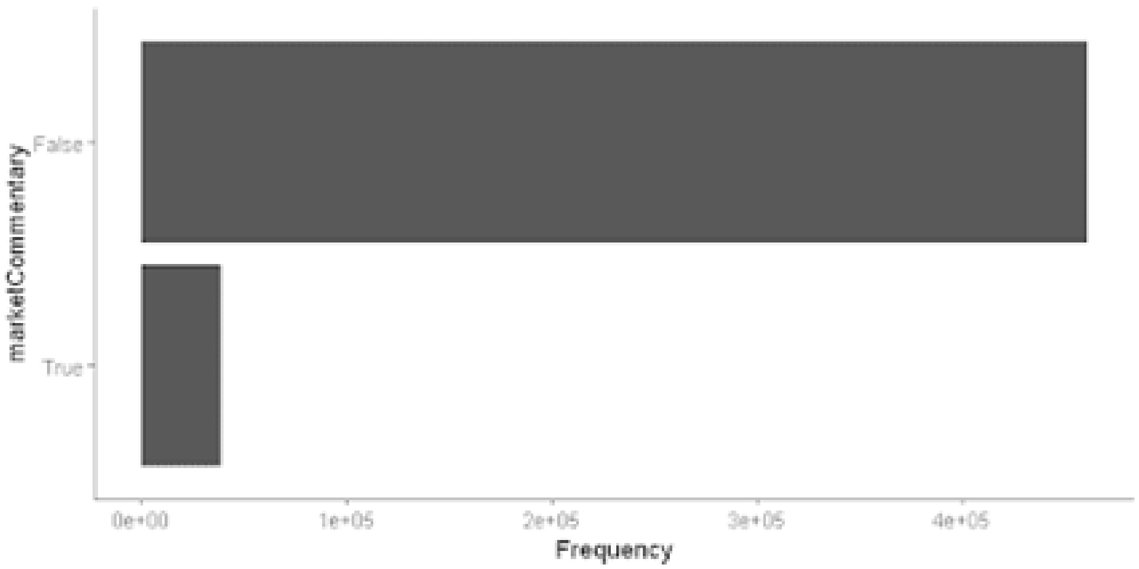


The sentence count from 0-300 has higher relevance when sentimentWordCount in between 0-5000.

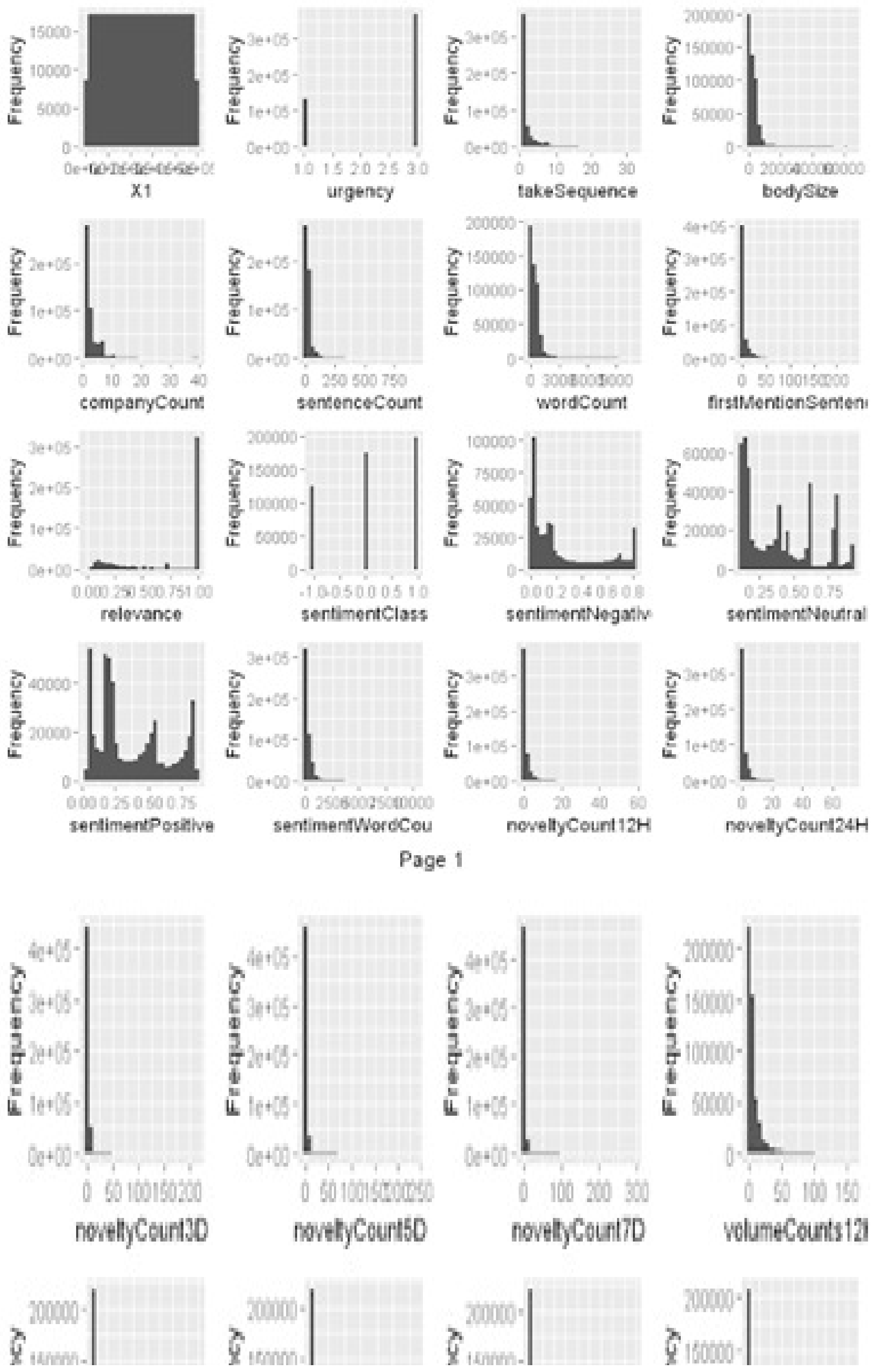


The above graph is actually a dynamic plot which represents the closing price of each asset represented with different colors. The closing prices vary based on the respective years.

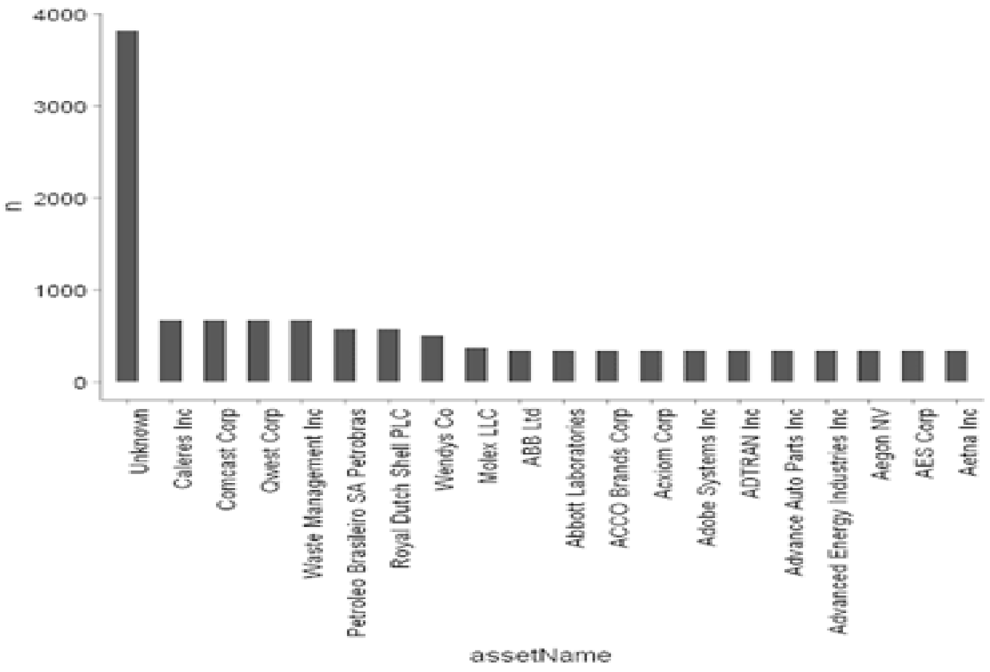




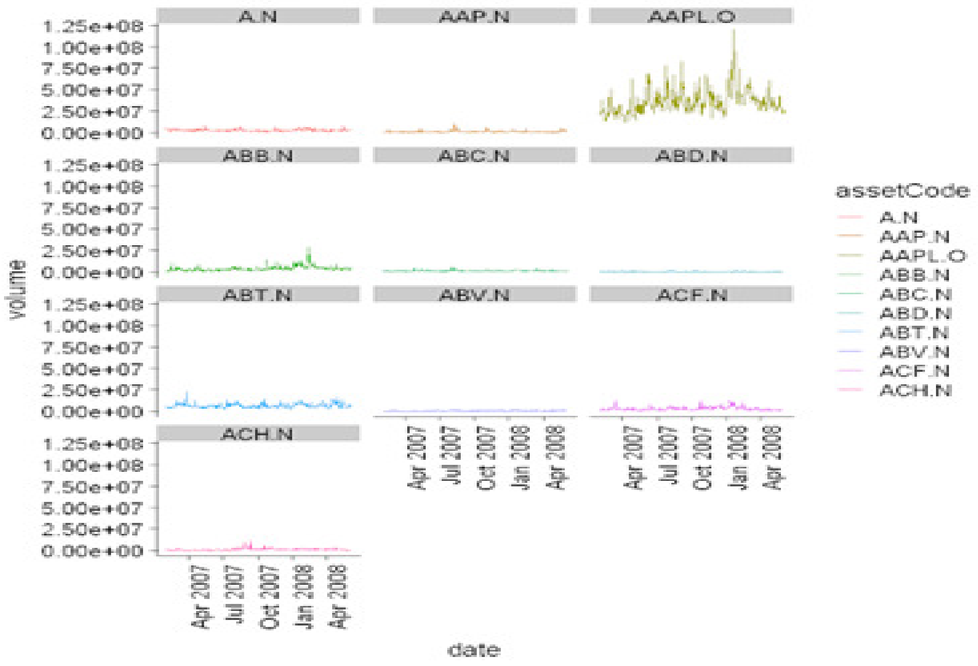
1. The above plot is the frequency of no. of times the assets have been mentioned in the dataset.
2. Next is the frequency of news articles which make marketCommentary or not. The frequency of 'False' or 'True' values is shown.



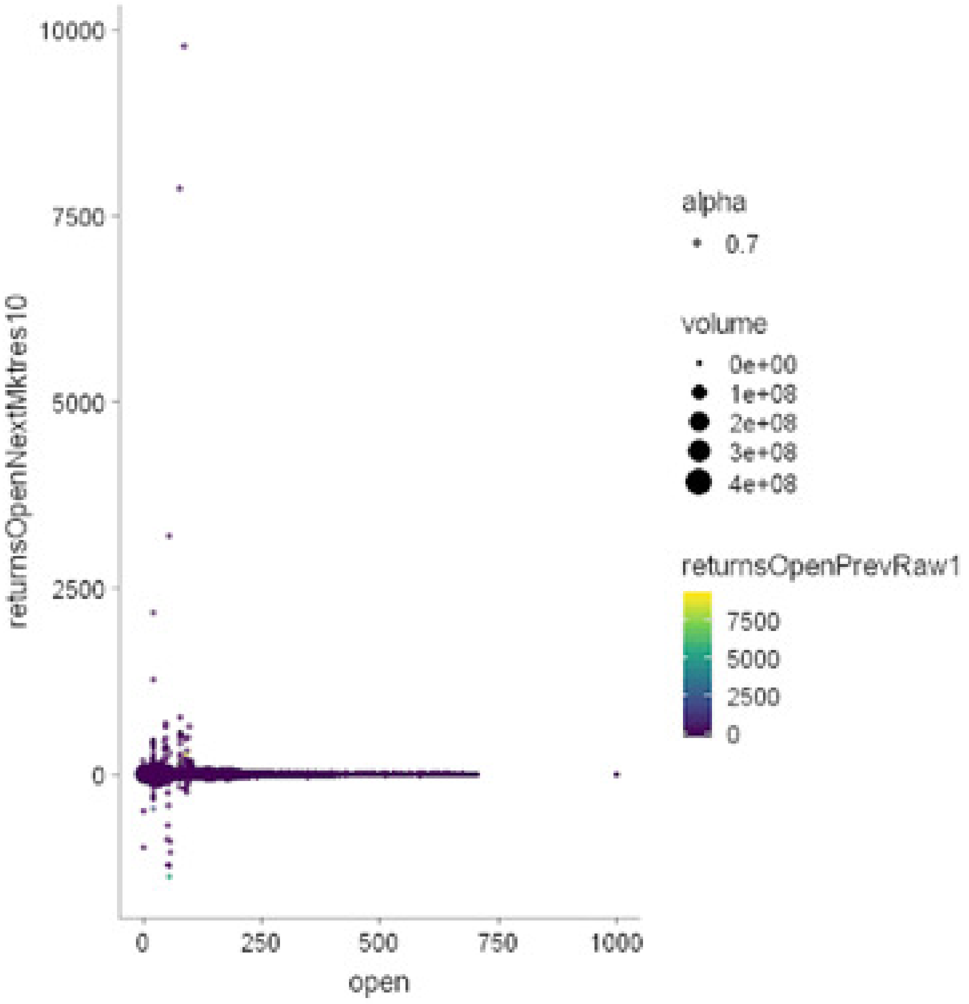
Frequency plot of each feature in the news dataset is shown in the above plot.



Assets and the no. of times they appeared in the data is represented by above graph.



Trading volume in shares for the day is the volume variable and here we consider the plot for the top 10 most used asset codes in the data.

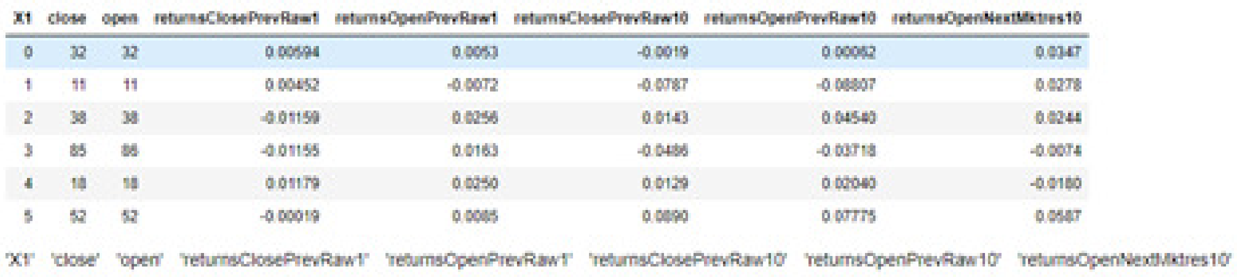


The above graph represents the relationship between open stock prices for the next 10 days and open stock price on the current day for the top 10 used assetcodes.

5.2 Principal Component Analysis

5.2.1 Market Data

1. **Data Analyzed**

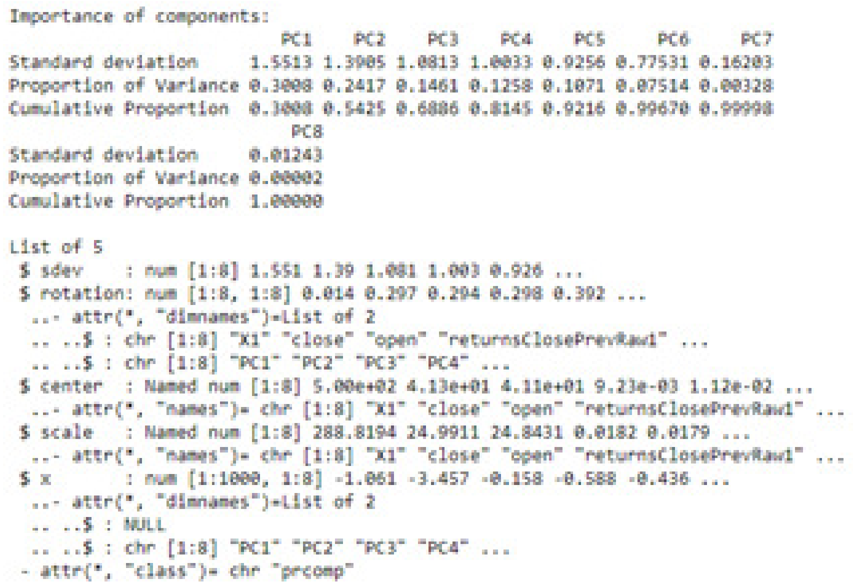


X1	close	open	returnsClosePrevRaw1	returnsOpenPrevRaw1	returnsClosePrevRaw10	returnsOpenPrevRaw10	returnsOpenNextMktres10
0	32	32	0.00594	0.0053	-0.0019	0.00062	0.0347
1	11	11	0.00452	-0.0072	-0.0787	-0.08807	0.0278
2	38	38	-0.01159	0.0256	0.0143	0.04540	0.0244
3	85	86	-0.01155	0.0163	-0.0486	-0.03718	-0.0074
4	18	18	0.01179	0.0250	0.0129	0.02040	-0.0180
5	52	52	-0.00019	0.0085	0.0890	0.07775	0.0587

'X1' 'close' 'open' 'returnsClosePrevRaw1' 'returnsOpenPrevRaw1' 'returnsClosePrevRaw10' 'returnsOpenPrevRaw10' 'returnsOpenNextMktres10'

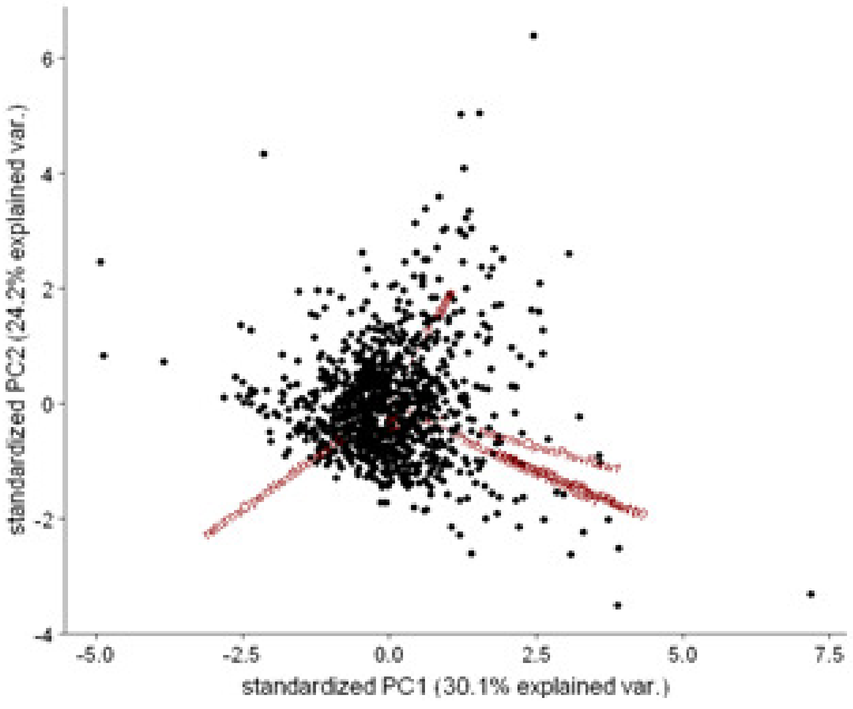
2. **prcomp**

   (a) **Summary**



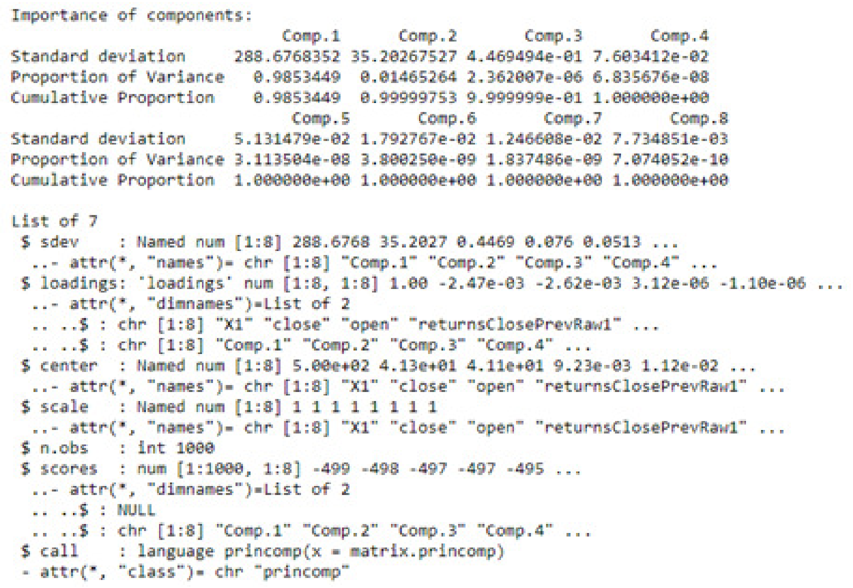
```
Importance of components:
                          PC1    PC2    PC3    PC4    PC5     PC6     PC7
Standard deviation     1.5513 1.3905 1.0813 1.0033 0.9256 0.77531 0.16203
Proportion of Variance 0.3008 0.2417 0.1461 0.1258 0.1071 0.07514 0.00328
Cumulative Proportion  0.3008 0.5425 0.6886 0.8145 0.9216 0.99670 0.99998
                           PC8
Standard deviation     0.01243
Proportion of Variance 0.00002
Cumulative Proportion  1.00000

List of 5
 $ sdev    : num [1:8] 1.551 1.39 1.081 1.003 0.926 ...
 $ rotation: num [1:8, 1:8] 0.014 0.297 0.294 0.298 0.392 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:8] "X1" "close" "open" "returnsClosePrevRaw1" ...
  .. ..$ : chr [1:8] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:8] 5.00e+02 4.13e+01 4.11e+01 9.23e-03 1.12e-02 ...
  ..- attr(*, "names")= chr [1:8] "X1" "close" "open" "returnsClosePrevRaw1" ...
 $ scale   : Named num [1:8] 288.8194 24.9911 24.8431 0.0182 0.0179 ...
  ..- attr(*, "names")= chr [1:8] "X1" "close" "open" "returnsClosePrevRaw1" ...
 $ x       : num [1:1000, 1:8] -1.061 -3.457 -0.158 -0.588 -0.436 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
  .. ..$ : chr [1:8] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
```

   (b) **Plot**



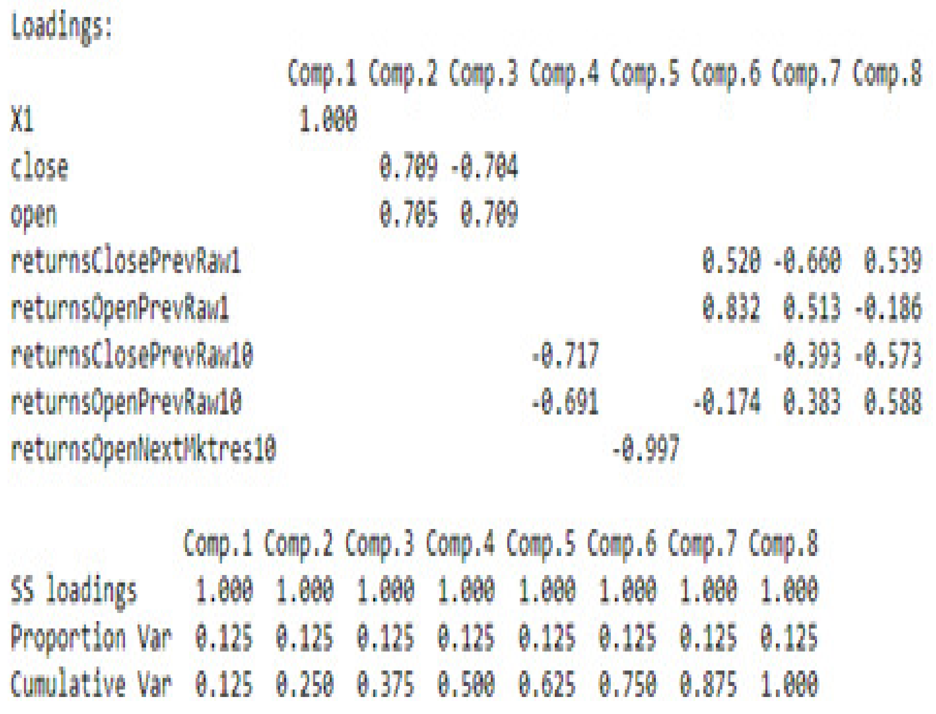
3. **princomp**

   (a) **Summary**



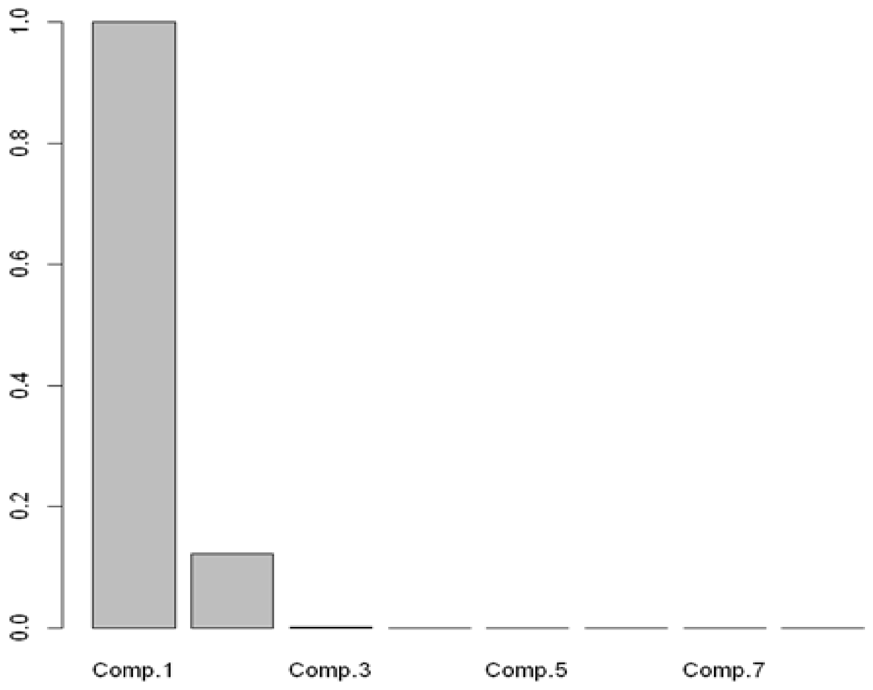
```
Importance of components:
                            Comp.1      Comp.2       Comp.3       Comp.4
Standard deviation     288.6768352 35.20267527 4.469494e-01 7.603412e-02
Proportion of Variance   0.9853449  0.01465264 2.362007e-06 6.835676e-08
Cumulative Proportion    0.9853449  0.99999753 9.999999e-01 1.000000e+00
                             Comp.5       Comp.6       Comp.7       Comp.8
Standard deviation     5.131479e-02 1.792767e-02 1.246608e-02 7.734851e-03
Proportion of Variance 3.113504e-08 3.800250e-09 1.837486e-09 7.074052e-10
Cumulative Proportion  1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00

List of 7
 $ sdev    : Named num [1:8] 288.6768 35.2027 0.4469 0.076 0.0513 ...
  ..- attr(*, "names")= chr [1:8] "Comp.1" "Comp.2" "Comp.3" "Comp.4" ...
 $ loadings: 'loadings' num [1:8, 1:8] 1.00 -2.47e-03 -2.62e-03 3.12e-06 -1.10e-06 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:8] "X1" "close" "open" "returnsClosePrevRaw1" ...
  .. ..$ : chr [1:8] "Comp.1" "Comp.2" "Comp.3" "Comp.4" ...
 $ center  : Named num [1:8] 5.00e+02 4.13e+01 4.11e+01 9.23e-03 1.12e-02 ...
  ..- attr(*, "names")= chr [1:8] "X1" "close" "open" "returnsClosePrevRaw1" ...
 $ scale   : Named num [1:8] 1 1 1 1 1 1 1 1
  ..- attr(*, "names")= chr [1:8] "X1" "close" "open" "returnsClosePrevRaw1" ...
 $ n.obs   : int 1000
 $ scores  : num [1:1000, 1:8] -499 -498 -497 -497 -495 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
  .. ..$ : chr [1:8] "Comp.1" "Comp.2" "Comp.3" "Comp.4" ...
 $ call    : language princomp(x = matrix.princomp)
 - attr(*, "class")= chr "princomp"
```

   (b) **Loadings**



```
Loadings:
                        Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
X1                       1.000                                                 
close                           0.709 -0.704                                   
open                            0.705  0.709                                   
returnsClosePrevRaw1                                         0.520 -0.660  0.539
returnsOpenPrevRaw1                                          0.832  0.513 -0.186
returnsClosePrevRaw10                        -0.717               -0.393 -0.573
returnsOpenPrevRaw10                         -0.691        -0.174  0.383  0.588
returnsOpenNextMktres10                             -0.997                     

               Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
Proportion Var  0.125  0.125  0.125  0.125  0.125  0.125  0.125  0.125
Cumulative Var  0.125  0.250  0.375  0.500  0.625  0.750  0.875  1.000
```

(c) **Plot**

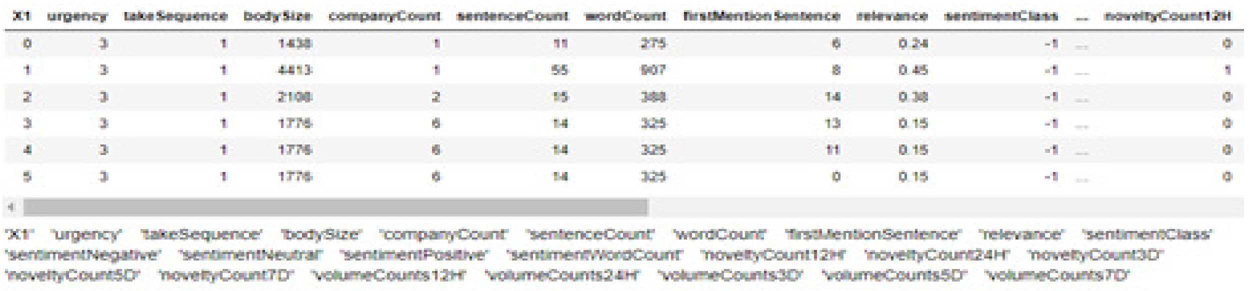


4. **Analysis**

   From here it is clear that when we get a barplot by plotting the standard deviation of all the PC's divided by standard deviation of PC 1 we can observe that the only relevant component is Comp 1 which is "close"

5.2.2 News Data

1. **Data Analyzed**

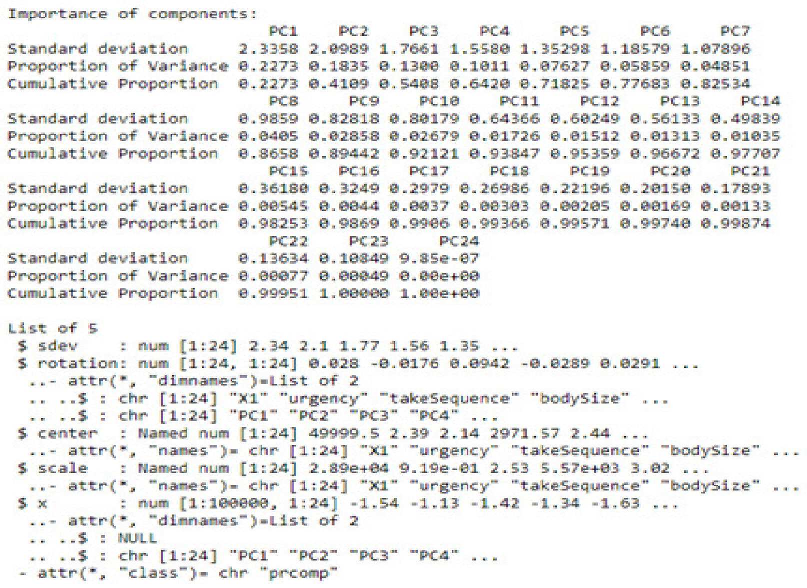


X1	urgency	takeSequence	bodySize	companyCount	sentenceCount	wordCount	firstMentionSentence	relevance	sentimentClass	...	noveltyCount12H
0	3	1	1438	1	11	275	6	0.24	-1	...	0
1	3	1	4413	1	55	907	8	0.45	-1	...	1
2	3	1	2108	2	15	388	14	0.38	-1	...	0
3	3	1	1776	6	14	325	13	0.15	-1	...	0
4	3	1	1776	6	14	325	11	0.15	-1	...	0
5	3	1	1776	6	14	325	0	0.15	-1	...	0

```
'X1' 'urgency' 'takeSequence' 'bodySize' 'companyCount' 'sentenceCount' 'wordCount' 'firstMentionSentence' 'relevance' 'sentimentClass'
'sentimentNegative' 'sentimentNeutral' 'sentimentPositive' 'sentimentWordCount' 'noveltyCount12H' 'noveltyCount24H' 'noveltyCount3D'
'noveltyCount5D' 'noveltyCount7D' 'volumeCounts12H' 'volumeCounts24H' 'volumeCounts3D' 'volumeCounts5D' 'volumeCounts7D'
```

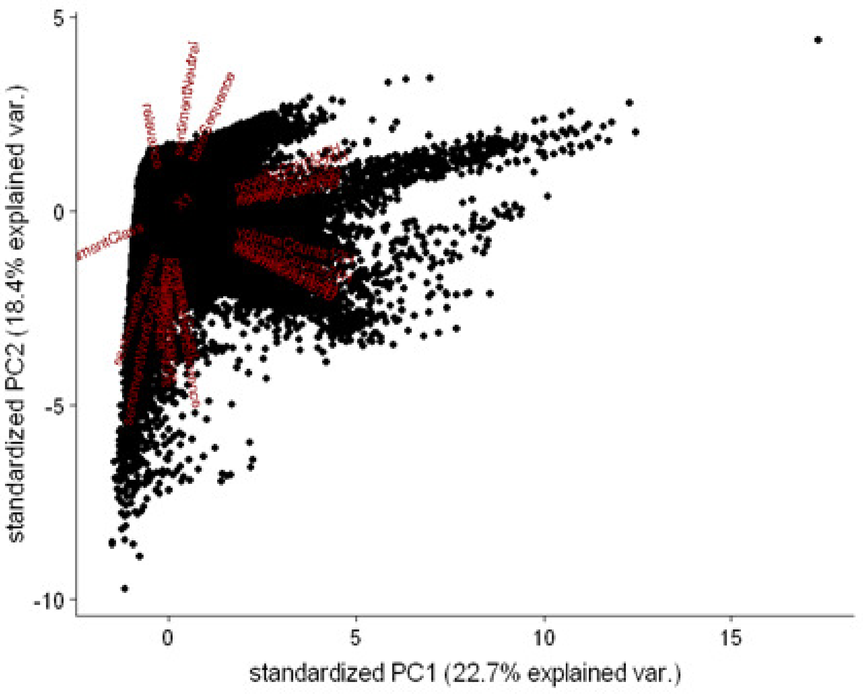
2. **prcomp**

(a) **Summary**



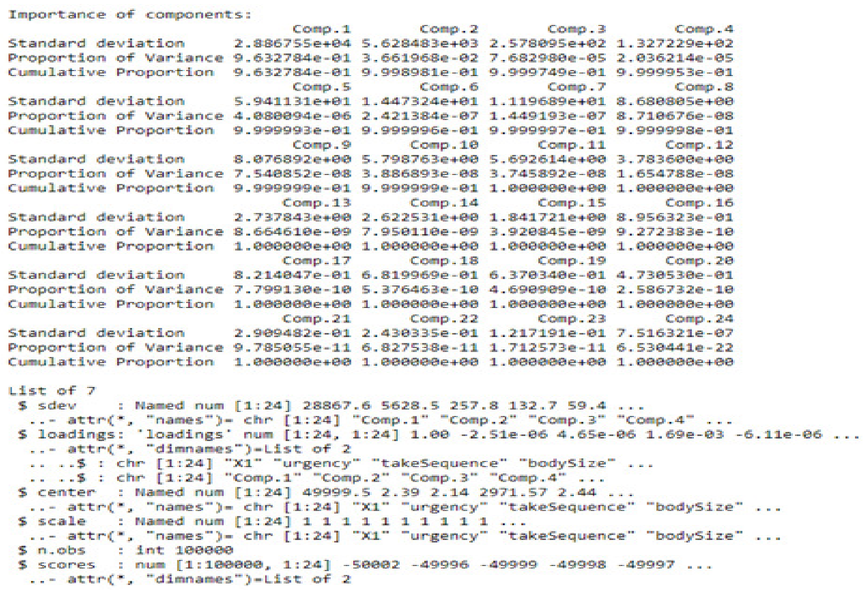
```
Importance of components:
                          PC1    PC2    PC3    PC4     PC5     PC6     PC7
Standard deviation     2.3358 2.0989 1.7661 1.5580 1.35298 1.18579 1.07896
Proportion of Variance 0.2273 0.1835 0.1300 0.1011 0.07627 0.05859 0.04851
Cumulative Proportion  0.2273 0.4109 0.5408 0.6420 0.71825 0.77683 0.82534
                          PC8     PC9    PC10    PC11    PC12    PC13    PC14
Standard deviation     0.9859 0.82818 0.80179 0.64366 0.60249 0.56133 0.49839
Proportion of Variance 0.0405 0.02858 0.02679 0.01726 0.01512 0.01313 0.01035
Cumulative Proportion  0.8658 0.89442 0.92121 0.93847 0.95359 0.96672 0.97707
                          PC15   PC16   PC17    PC18    PC19    PC20    PC21
Standard deviation     0.36180 0.3249 0.2979 0.26986 0.22196 0.20150 0.17893
Proportion of Variance 0.00545 0.0044 0.0037 0.00303 0.00205 0.00169 0.00133
Cumulative Proportion  0.98253 0.9869 0.9906 0.99366 0.99571 0.99740 0.99874
                          PC22    PC23     PC24
Standard deviation     0.13634 0.10849 9.85e-07
Proportion of Variance 0.00077 0.00049 0.00e+00
Cumulative Proportion  0.99951 1.00000 1.00e+00

List of 5
 $ sdev    : num [1:24] 2.34 2.1 1.77 1.56 1.35 ...
 $ rotation: num [1:24, 1:24] 0.028 -0.0176 0.0942 -0.0289 0.0291 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:24] "X1" "urgency" "takeSequence" "bodySize" ...
  .. ..$ : chr [1:24] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:24] 49999.5 2.39 2.14 2971.57 2.44 ...
  ..- attr(*, "names")= chr [1:24] "X1" "urgency" "takeSequence" "bodySize" ...
 $ scale   : Named num [1:24] 2.89e+04 9.19e-01 2.53 5.57e+03 3.02 ...
  ..- attr(*, "names")= chr [1:24] "X1" "urgency" "takeSequence" "bodySize" ...
 $ x       : num [1:100000, 1:24] -1.54 -1.13 -1.42 -1.34 -1.63 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
  .. ..$ : chr [1:24] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
```

(b) **Plot**



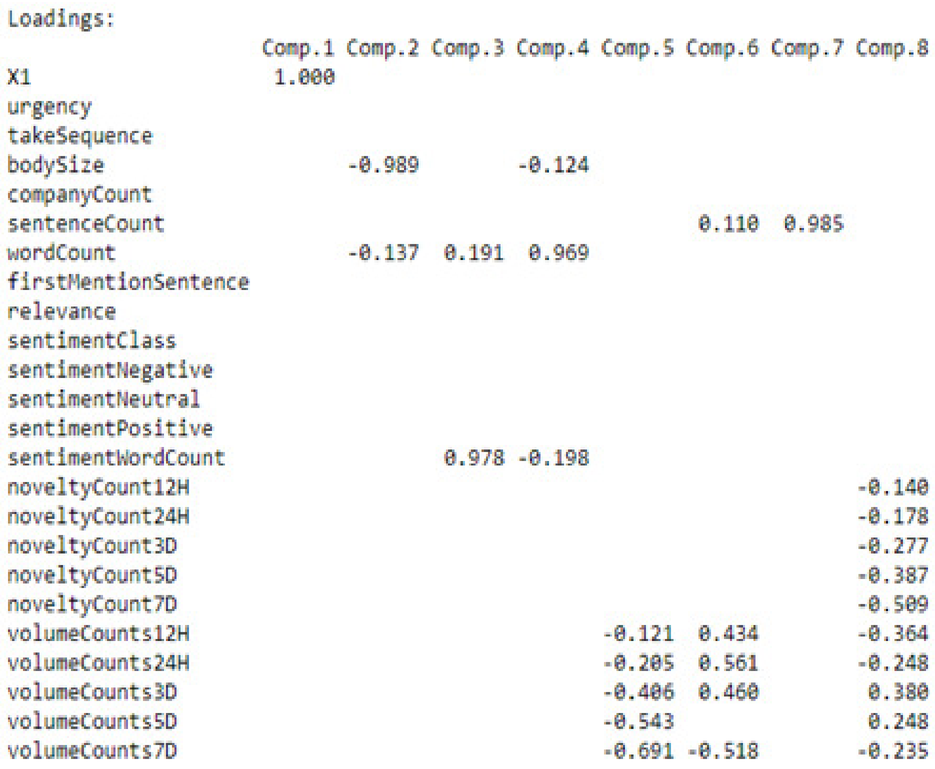
3. **princomp**

   (a) **Summary**



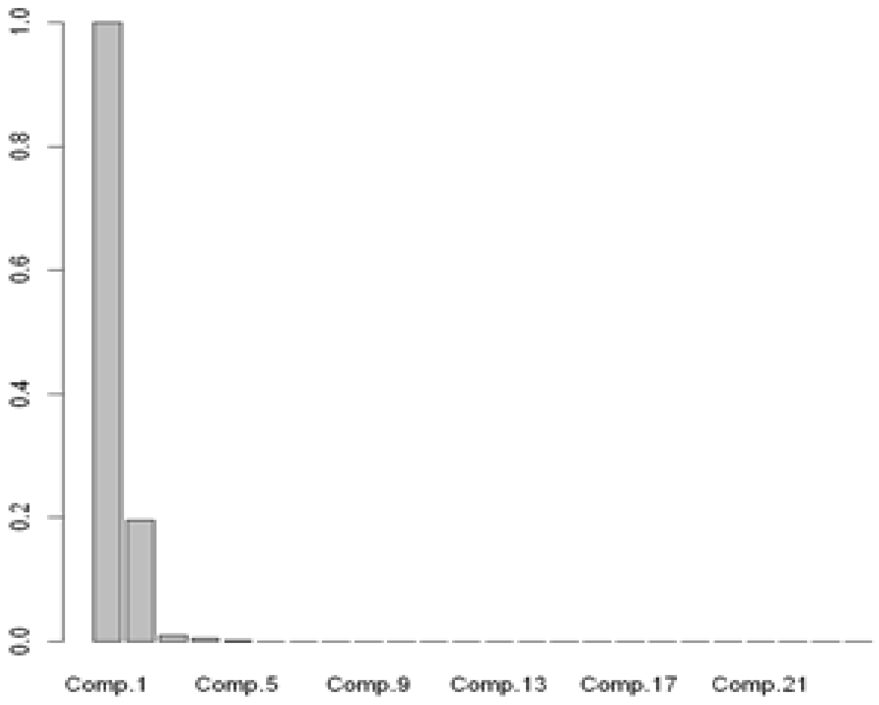
```
Importance of components:
                          Comp.1       Comp.2       Comp.3       Comp.4
Standard deviation     2.886755e+04 5.628483e+03 2.578095e+02 1.327229e+02
Proportion of Variance 9.632784e-01 3.661968e-02 7.682980e-05 2.036214e-05
Cumulative Proportion  9.632784e-01 9.998981e-01 9.999749e-01 9.999953e-01
                          Comp.5       Comp.6       Comp.7       Comp.8
Standard deviation     5.941131e+01 1.447324e+01 1.119689e+01 8.680805e+00
Proportion of Variance 4.080094e-06 2.421384e-07 1.449193e-07 8.710676e-08
Cumulative Proportion  9.999993e-01 9.999996e-01 9.999997e-01 9.999998e-01
                          Comp.9      Comp.10      Comp.11      Comp.12
Standard deviation     8.076892e+00 5.798763e+00 5.692614e+00 3.783600e+00
Proportion of Variance 7.540852e-08 3.886893e-08 3.745892e-08 1.654788e-08
Cumulative Proportion  9.999999e-01 9.999999e-01 1.000000e+00 1.000000e+00
                         Comp.13      Comp.14      Comp.15      Comp.16
Standard deviation     2.737843e+00 2.622531e+00 1.841721e+00 8.956323e-01
Proportion of Variance 8.664610e-09 7.950110e-09 3.920845e-09 9.272383e-10
Cumulative Proportion  1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
                         Comp.17      Comp.18      Comp.19      Comp.20
Standard deviation     8.214047e-01 6.819969e-01 6.370340e-01 4.730530e-01
Proportion of Variance 7.799130e-10 5.376463e-10 4.690909e-10 2.586732e-10
Cumulative Proportion  1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
                         Comp.21      Comp.22      Comp.23      Comp.24
Standard deviation     2.909482e-01 2.430335e-01 1.217191e-01 7.516321e-07
Proportion of Variance 9.785055e-11 6.827538e-11 1.712573e-11 6.530441e-22
Cumulative Proportion  1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00

List of 7
 $ sdev    : Named num [1:24] 28867.6 5628.5 257.8 132.7 59.4 ...
  ..- attr(*, "names")= chr [1:24] "Comp.1" "Comp.2" "Comp.3" "Comp.4" ...
 $ loadings: 'loadings' num [1:24, 1:24] 1.00 -2.51e-06 4.65e-06 1.69e-03 -6.11e-06 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:24] "X1" "urgency" "takeSequence" "bodySize" ...
  .. ..$ : chr [1:24] "Comp.1" "Comp.2" "Comp.3" "Comp.4" ...
 $ center  : Named num [1:24] 49999.5 2.39 2.14 2971.57 2.44 ...
  ..- attr(*, "names")= chr [1:24] "X1" "urgency" "takeSequence" "bodySize" ...
 $ scale   : Named num [1:24] 1 1 1 1 1 1 1 1 1 1 ...
  ..- attr(*, "names")= chr [1:24] "X1" "urgency" "takeSequence" "bodySize" ...
 $ n.obs   : int 100000
 $ scores  : num [1:100000, 1:24] -50002 -49996 -49999 -49998 -49997 ...
  ..- attr(*, "dimnames")=List of 2
```

   (b) **Loadings**



```
Loadings:
                     Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
X1                    1.000                                                 
urgency                                                                     
takeSequence                                                                
bodySize                    -0.989        -0.124                            
companyCount                                                                
sentenceCount                                           0.110  0.985        
wordCount                   -0.137  0.191  0.969                            
firstMentionSentence                                                        
relevance                                                                   
sentimentClass                                                              
sentimentNegative                                                           
sentimentNeutral                                                            
sentimentPositive                                                           
sentimentWordCount                  0.978 -0.198                            
noveltyCount12H                                                       -0.140
noveltyCount24H                                                       -0.178
noveltyCount3D                                                        -0.277
noveltyCount5D                                                        -0.387
noveltyCount7D                                                        -0.509
volumeCounts12H                                  -0.121  0.434        -0.364
volumeCounts24H                                  -0.205  0.561        -0.248
volumeCounts3D                                   -0.406  0.460         0.380
volumeCounts5D                                   -0.543                0.248
volumeCounts7D                                   -0.691 -0.518        -0.235
```

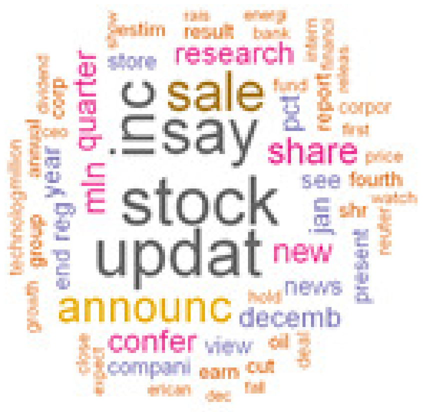
(c) **Plot**



4. **Analysis**

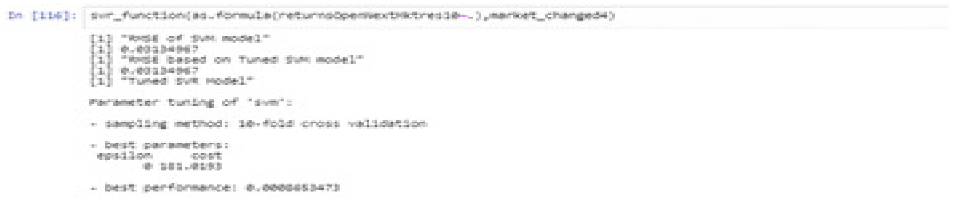
   From here it is clear that when we get a barplot by plotting the standard deviation of all the PC's divided by standard deviation of PC 1 we can observe that the only relevant component is Comp 1 which is "urgency"

5.3 Word Cloud

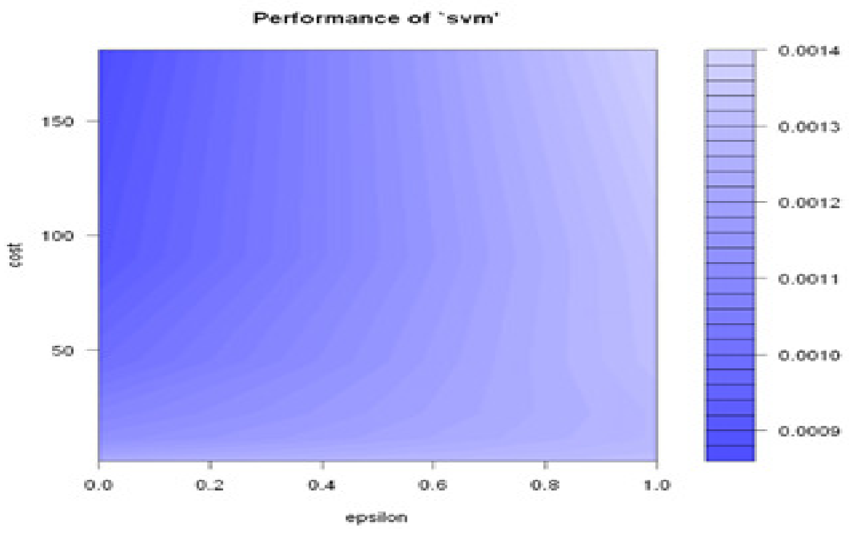


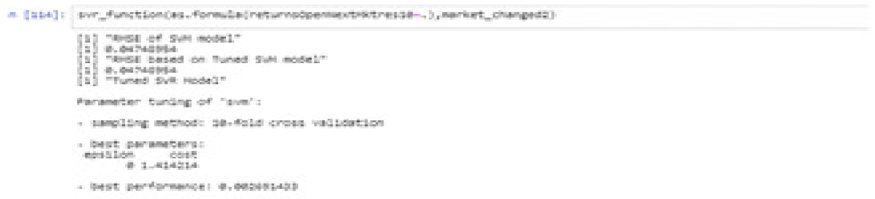
5.4 Regression

5.4.1 Support Vector Regression

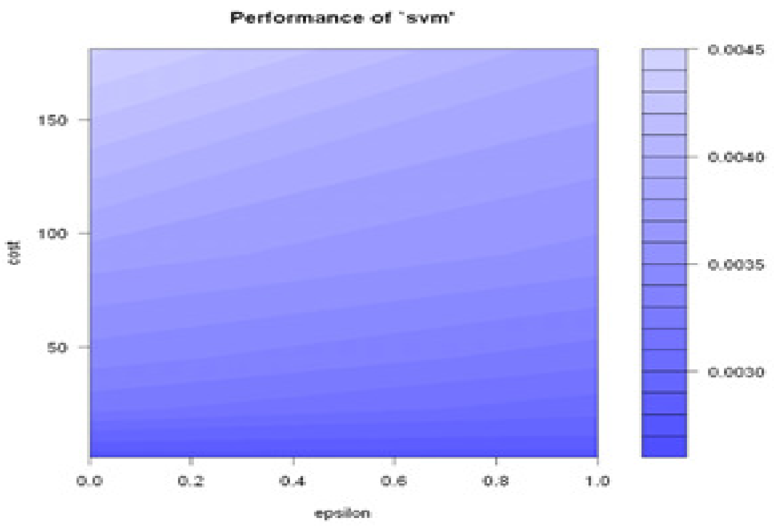


```
In [116]: svr_function(as.formula(returnsOpenNextMktres10~.),market_changed4)

[1] "RMSE of SVM model"
[1] 0.03134967
[1] "RMSE based on Tuned SVM model"
[1] 0.03134967
[1] "Tuned SVM Model"

Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 epsilon      cost
       0 181.0193

- best performance: 0.0008653473
```

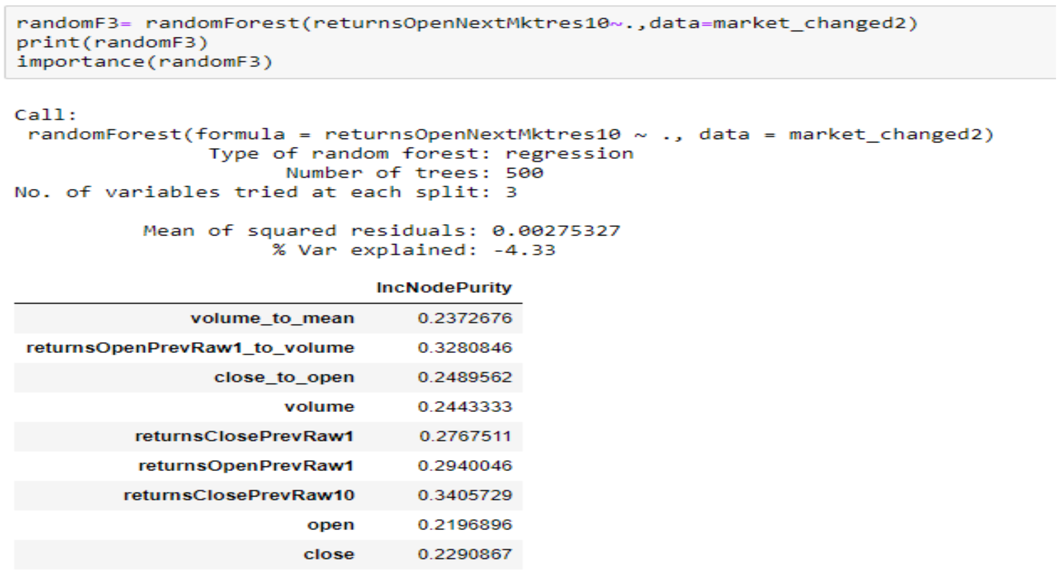




```
In [114]: svr_function(as.formula(returnsOpenNextMktres10~.),market_changed2)

[1] "RMSE of SVM model"
[1] 0.04742984
[1] "RMSE based on Tuned SVM model"
[1] 0.04742984
[1] "Tuned SVM Model"

Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 epsilon      cost
       0 1.414214

- best performance: 0.002691433
```



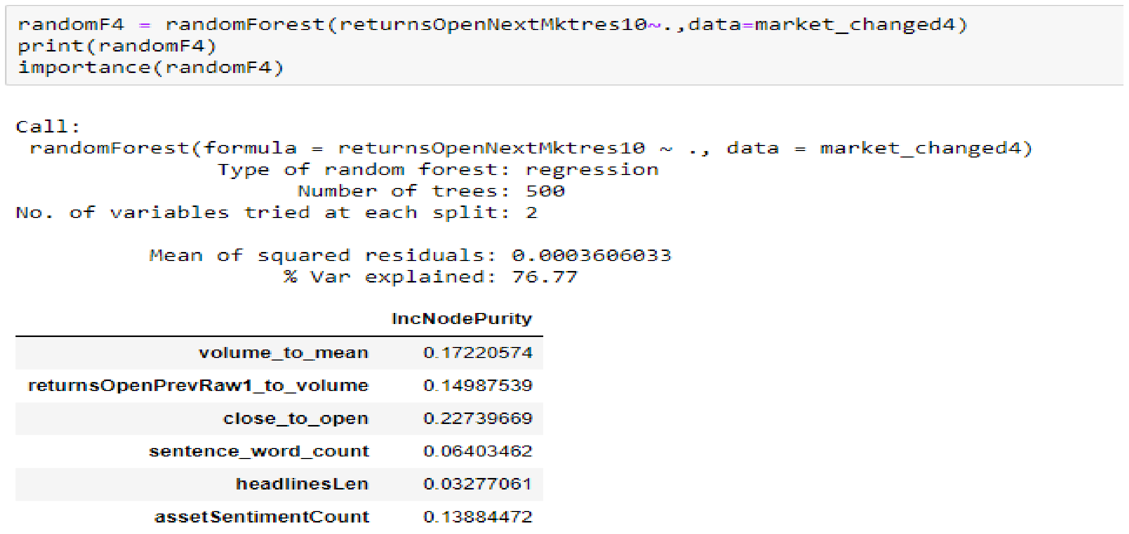
5.4.2 Random Forest Classifier



```
randomF3= randomForest(returnsOpenNextMktres10~.,data=market_changed2)
print(randomF3)
importance(randomF3)
```

```
Call:
 randomForest(formula = returnsOpenNextMktres10 ~ ., data = market_changed2)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 3

          Mean of squared residuals: 0.00275327
                    % Var explained: -4.33
```

	IncNodePurity
volume_to_mean	0.2372676
returnsOpenPrevRaw1_to_volume	0.3280846
close_to_open	0.2489562
volume	0.2443333
returnsClosePrevRaw1	0.2767511
returnsOpenPrevRaw1	0.2940046
returnsClosePrevRaw10	0.3405729
open	0.2196896
close	0.2290867



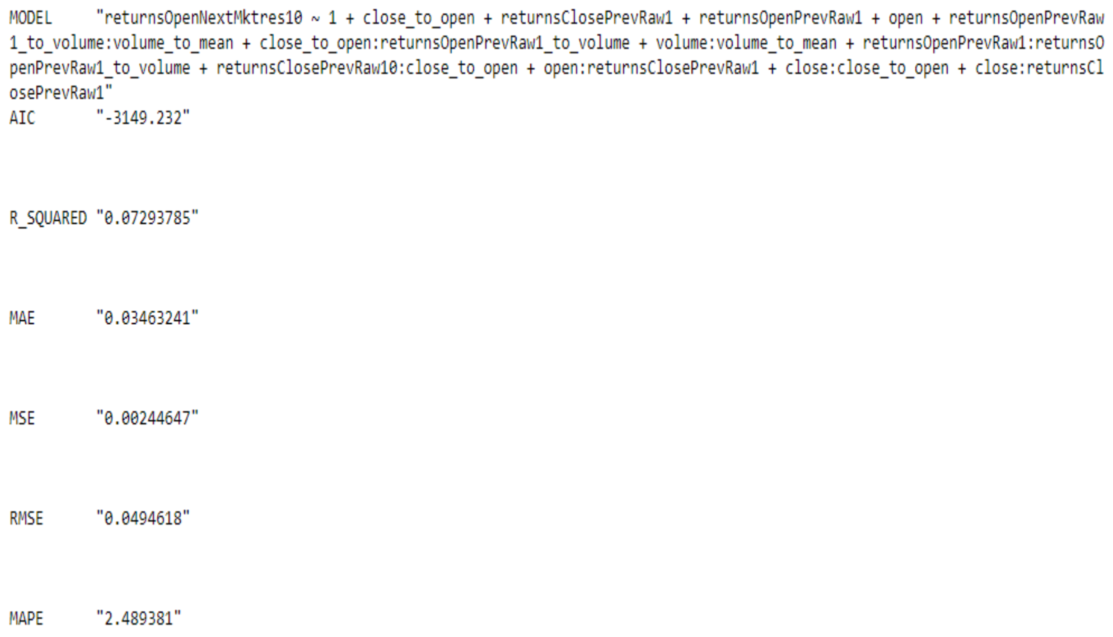
```
randomF4 = randomForest(returnsOpenNextMktres10~.,data=market_changed4)
print(randomF4)
importance(randomF4)
```

```
Call:
 randomForest(formula = returnsOpenNextMktres10 ~ ., data = market_changed4)
              Type of random forest: regression
                    Number of trees: 500
No. of variables tried at each split: 2

          Mean of squared residuals: 0.0003606033
                    % Var explained: 76.77
```

	IncNodePurity
volume_to_mean	0.17220574
returnsOpenPrevRaw1_to_volume	0.14987539
close_to_open	0.22739669
sentence_word_count	0.06403462
headlinesLen	0.03277061
assetSentimentCount	0.13884472

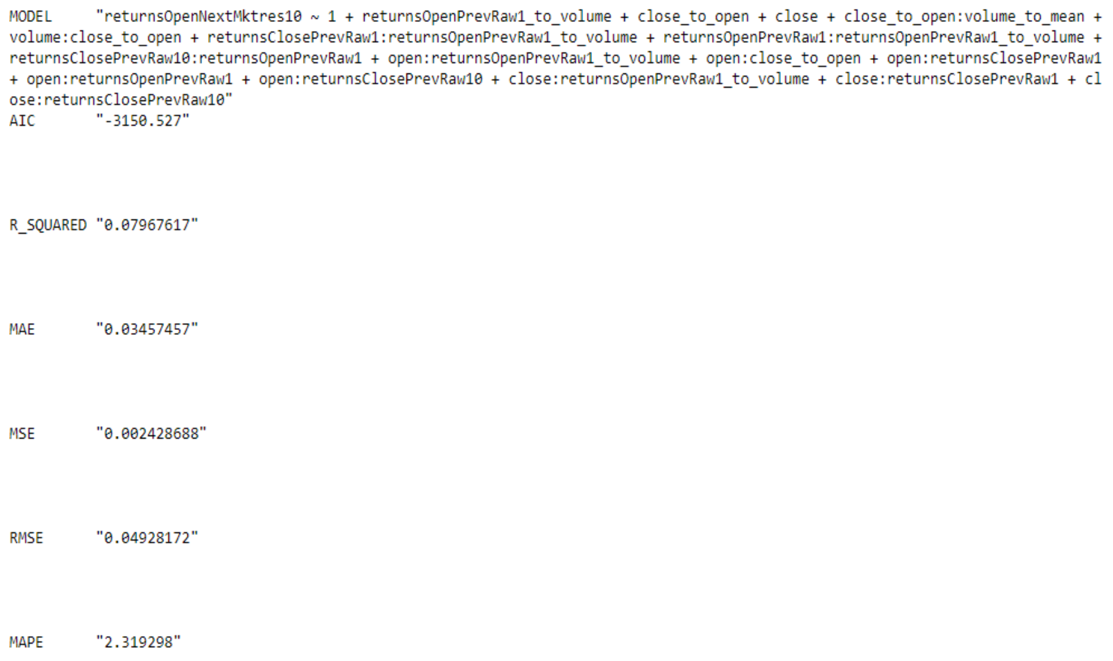
5.4.3 Linear Regression

1. **LM - Combination 2**



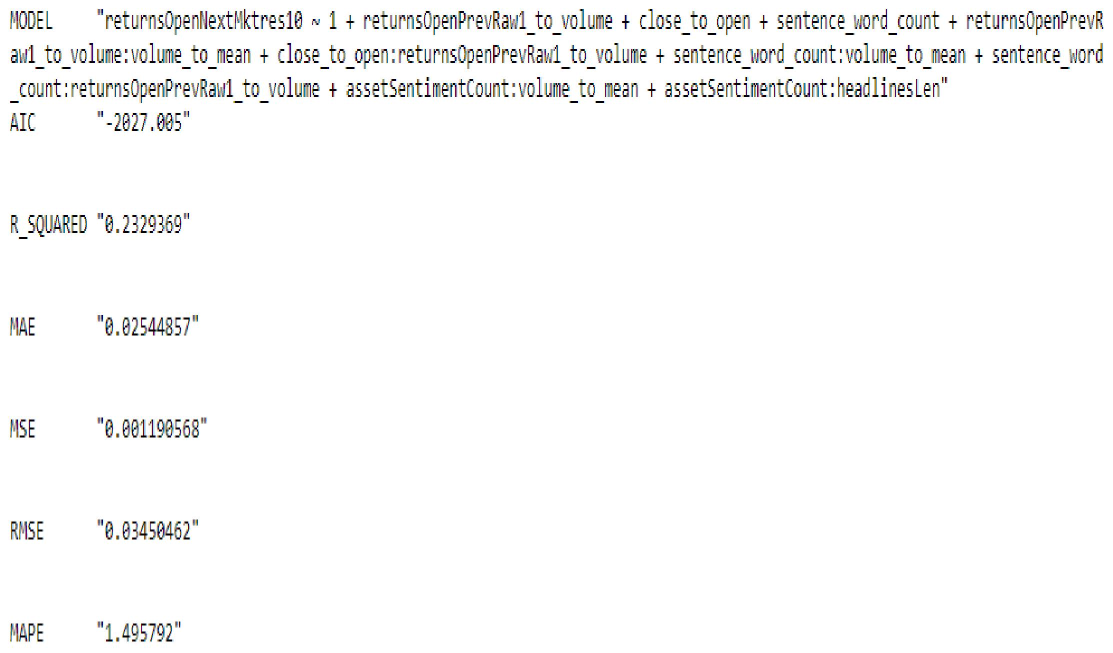
```
MODEL     "returnsOpenNextMktres10 ~ 1 + close_to_open + returnsClosePrevRaw1 + returnsOpenPrevRaw1 + open + returnsOpenPrevRaw
1_to_volume:volume_to_mean + close_to_open:returnsOpenPrevRaw1_to_volume + volume:volume_to_mean + returnsOpenPrevRaw1:returnsO
penPrevRaw1_to_volume + returnsClosePrevRaw10:close_to_open + open:returnsClosePrevRaw1 + close:close_to_open + close:returnsCl
osePrevRaw1"
AIC       "-3149.232"


R_SQUARED "0.07293785"


MAE       "0.03463241"


MSE       "0.00244647"


RMSE      "0.0494618"


MAPE      "2.489381"
```

2. **GLM - Combination 2**



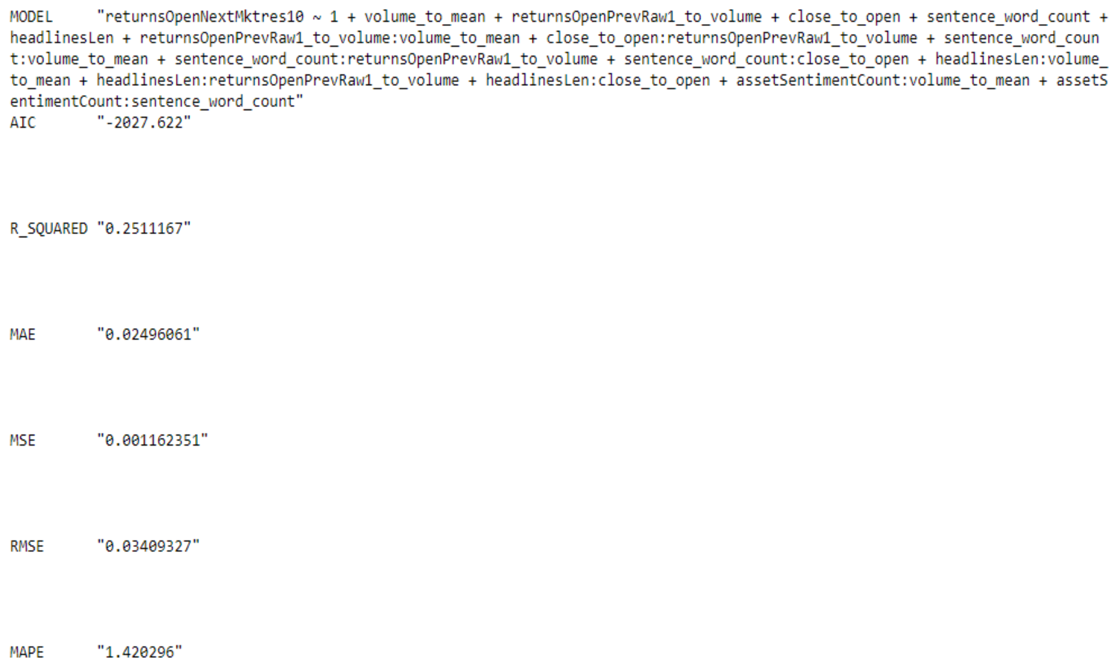
```
MODEL     "returnsOpenNextMktres10 ~ 1 + returnsOpenPrevRaw1_to_volume + close_to_open + close + close_to_open:volume_to_mean +
volume:close_to_open + returnsClosePrevRaw1:returnsOpenPrevRaw1_to_volume + returnsOpenPrevRaw1:returnsOpenPrevRaw1_to_volume +
returnsClosePrevRaw10:returnsOpenPrevRaw1 + open:returnsOpenPrevRaw1_to_volume + open:close_to_open + open:returnsClosePrevRaw1
+ open:returnsOpenPrevRaw1 + open:returnsClosePrevRaw10 + close:returnsOpenPrevRaw1_to_volume + close:returnsClosePrevRaw1 + cl
ose:returnsClosePrevRaw10"
AIC       "-3150.527"


R_SQUARED "0.07967617"


MAE       "0.03457457"


MSE       "0.002428688"


RMSE      "0.04928172"


MAPE      "2.319298"
```

3. **LM - Combination 4**



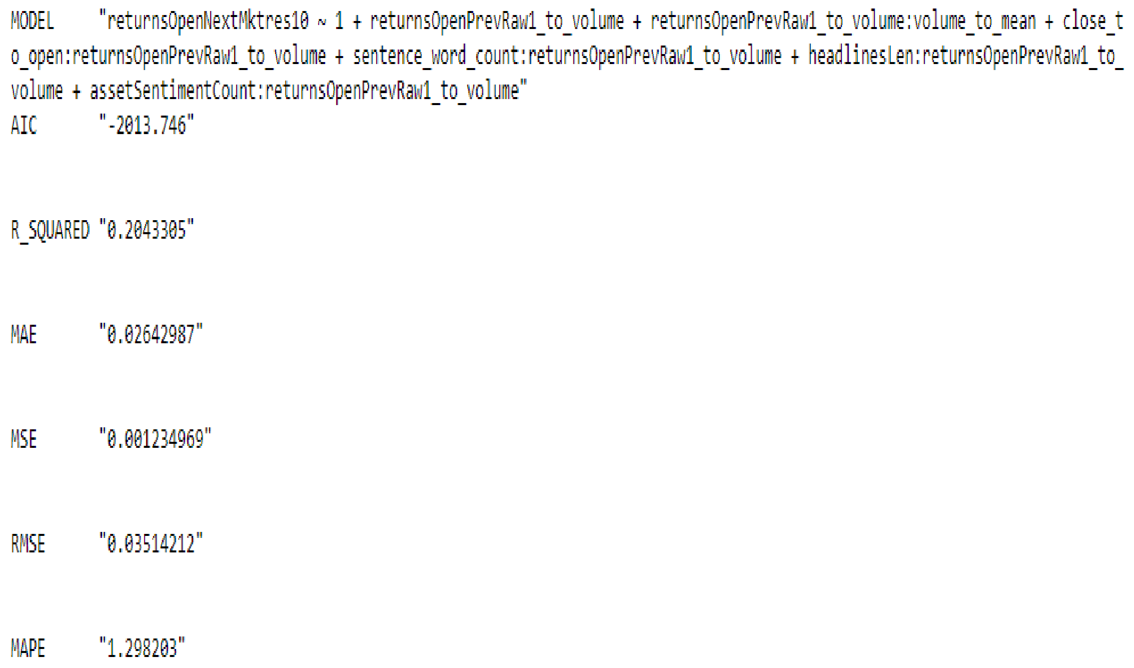
```
MODEL     "returnsOpenNextMktres10 ~ 1 + returnsOpenPrevRaw1_to_volume + close_to_open + sentence_word_count + returnsOpenPrevR
aw1_to_volume:volume_to_mean + close_to_open:returnsOpenPrevRaw1_to_volume + sentence_word_count:volume_to_mean + sentence_word
_count:returnsOpenPrevRaw1_to_volume + assetSentimentCount:volume_to_mean + assetSentimentCount:headlinesLen"
AIC       "-2027.005"


R_SQUARED "0.2329369"


MAE       "0.02544857"


MSE       "0.001190568"


RMSE      "0.03450462"


MAPE      "1.495792"
```

4. **GLM - Combination 4**



```
MODEL     "returnsOpenNextMktres10 ~ 1 + volume_to_mean + returnsOpenPrevRaw1_to_volume + close_to_open + sentence_word_count +
headlinesLen + returnsOpenPrevRaw1_to_volume:volume_to_mean + close_to_open:returnsOpenPrevRaw1_to_volume + sentence_word_coun
t:volume_to_mean + sentence_word_count:returnsOpenPrevRaw1_to_volume + sentence_word_count:close_to_open + headlinesLen:volume_
to_mean + headlinesLen:returnsOpenPrevRaw1_to_volume + headlinesLen:close_to_open + assetSentimentCount:volume_to_mean + assetS
entimentCount:sentence_word_count"
AIC       "-2027.622"


R_SQUARED "0.2511167"


MAE       "0.02496061"


MSE       "0.001162351"


RMSE      "0.03409327"


MAPE      "1.420296"
```

5. **GLS - Combination 4**



```
MODEL     "returnsOpenNextMktres10 ~ 1 + returnsOpenPrevRaw1_to_volume + returnsOpenPrevRaw1_to_volume:volume_to_mean + close_t
o_open:returnsOpenPrevRaw1_to_volume + sentence_word_count:returnsOpenPrevRaw1_to_volume + headlinesLen:returnsOpenPrevRaw1_to_
volume + assetSentimentCount:returnsOpenPrevRaw1_to_volume"
AIC       "-2013.746"


R_SQUARED "0.2043305"


MAE       "0.02642987"


MSE       "0.001234969"


RMSE      "0.03514212"


MAPE      "1.298203"
```

6 Conclusion

It has been observed that any model with the combination of news and market data has shown to perform better. The best model from Support Vector Regression, Random Forest Classifier and Linear Regression is Linear Regression with General Linearized Squares Technique.

But the R-squared value is low because we could not use the entire dataset due to computational complexity and the data that we used does not have enough data for each asset to fit the model correctly.

This is a Kaggle challenge and we have uploaded our code only 2 days back and there has not enough time passed for it to be ranked. We will keep you posted on the update. Our code can be seen on Kaggle for Exploratory Data Analysis and Feature Engineering and Regression Analysis

7 Related Work

There have been many papers that discussed exploiting web data for various applications and finding out useful patterns in making decisions. One of the most popular was sentiment analysis using Twitter data. Using the data, positive and negative sentiments are classified using machine learning algorithms which was very successful. Similarly, news data is also being hugely considered for making great predictions not only in the field of stock market but various other areas. According to the Efficient Market Hypothesis (EMH), in the financial market opportunities are exploited as soon as they arise. On the one hand, plenty sources of the information such as the stock prices, historical data and company's information make it extremely hard to predict accurately, it is possible to forecast the stock market. In fact, it takes time for the market to adjust itself to the incoming news. It will be more profitable to generate an action signal (buy, sell) in corresponding to the market news rather than accurately predict the future prices of the stock.

References

[1] e1071.

[2] Generalized linear model.

[3] Linear model.

[4] Linear regression.

[5] Mean squared error.

[6] R-squared.

[7] Random forests.

[8] Rms error.

[9] Support vector machine.

[10] Breiman, L. Random forests. *Mach. Learn. 45*, 1 (Oct. 2001), 5–32.

[11] Calcagno, V., and de Mazancourt, C. glmulti: An r package for easy automated model selection with (generalized) linear models. *Journal Of Statistical Software 34* (05 2010).

[12] Seuret, M., Alberti, M., Liwicki, M., and Ingold, R. Pca-initialized deep neural networks applied to document image analysis. pp. 877–882.