ONLINE NEWS POPULARITY PREDICTION

Introduction:

The consumption of online news accelerates day by day due to the widespread adoption of smartphones and the rise of social networks. More interestingly, some particular types of content can capture the attention of a significant amount of Internet users within a short period of time. As a consequence, researchers focus on the analysis of online news content such as predicting the popularity of news articles.

The prediction of the popularity of online news content has remarkable practical values in many fields. For example, by utilizing the advantages of popularity prediction, news organization can gain a better understanding of different types of online news consumption of users. As a result, the news organization can deliver more relevant and engaging content in a proactive manner as well as the organization can allocate resources more wisely to develop stories .

Furthermore, prediction of news content is also beneficial for trend forecasting, online marketing, media advertising,for authors, advertisers, activists, content providers and others.

Objective :

The objective of this project is to predict whether an article will be popular or not prior to its publication by estimating the number of shares, using machine learning algorithms.The motivation behind tackling this problem is that before investing resources and time into the publication of a new article, we can predict whether or not it is worthy of being published in the first place.

Data Collection :

The data was collected from the UCI Machine Learning repository. It contains the content of all the articles published during a two year period, from 2013 to 2015on Mashable (mashable.com).
(https://archive.ics.uci.edu/ml/datasets/Online+News+Popularity)
The dataset consists of 39644 records and 61 attributes.

Data Description

Attributes are as follows:

0. url: URL of the article (non-predictive)
1. timedelta: Days between the article publication and the dataset acquisition (non-predictive)
2. n_tokens_title: Number of words in the title
3. n_tokens_content: Number of words in the content
4. n_unique_tokens: Rate of unique words in the content
5. n_non_stop_words: Rate of non-stop words in the content
6. n_non_stop_unique_tokens: Rate of unique non-stop words in the content
7. num_hrefs: Number of links
8. num_self_hrefs: Number of links to other articles published by Mashable
9. num_imgs: Number of images
10. num_videos: Number of videos

11. average_token_length: Average length of the words in the content
12. num_keywords: Number of keywords in the metadata
13. data_channel_is_lifestyle: Is data channel 'Lifestyle'?
14. data_channel_is_entertainment: Is data channel 'Entertainment'?
15. data_channel_is_bus: Is data channel 'Business'?
16. data_channel_is_socmed: Is data channel 'Social Media'?
17. data_channel_is_tech: Is data channel 'Tech'?
18. data_channel_is_world: Is data channel 'World'?
19. kw_min_min: Worst keyword (min. shares)
20. kw_max_min: Worst keyword (max. shares)
21. kw_avg_min: Worst keyword (avg. shares)
22. kw_min_max: Best keyword (min. shares)
23. kw_max_max: Best keyword (max. shares)
24. kw_avg_max: Best keyword (avg. shares)
25. kw_min_avg: Avg. keyword (min. shares)
26. kw_max_avg: Avg. keyword (max. shares)
27. kw_avg_avg: Avg. keyword (avg. shares)
28. self_reference_min_shares: Min. shares of referenced articles in Mashable
29. self_reference_max_shares: Max. shares of referenced articles in Mashable
30. self_reference_avg_sharess: Avg. shares of referenced articles in Mashable
31. weekday_is_monday: Was the article published on a Monday?
32. weekday_is_tuesday: Was the article published on a Tuesday?
33. weekday_is_wednesday: Was the article published on a Wednesday?
34. weekday_is_thursday: Was the article published on a Thursday?
35. weekday_is_friday: Was the article published on a Friday?
36. weekday_is_saturday: Was the article published on a Saturday?
37. weekday_is_sunday: Was the article published on a Sunday?
38. is_weekend: Was the article published on the weekend?
39. LDA_00: Closeness to LDA topic 0
40. LDA_01: Closeness to LDA topic 1
41. LDA_02: Closeness to LDA topic 2
42. LDA_03: Closeness to LDA topic 3
43. LDA_04: Closeness to LDA topic 4
44. global_subjectivity: Text subjectivity
45. global_sentiment_polarity: Text sentiment polarity
46. global_rate_positive_words: Rate of positive words in the content
47. global_rate_negative_words: Rate of negative words in the content
48. rate_positive_words: Rate of positive words among non-neutral tokens
49. rate_negative_words: Rate of negative words among non-neutral tokens
50. avg_positive_polarity: Avg. polarity of positive words
51. min_positive_polarity: Min. polarity of positive words
52. max_positive_polarity: Max. polarity of positive words
53. avg_negative_polarity: Avg. polarity of negative words
54. min_negative_polarity: Min. polarity of negative words
55. max_negative_polarity: Max. polarity of negative words
56. title_subjectivity: Title subjectivity
57. title_sentiment_polarity: Title polarity
58. abs_title_subjectivity: Absolute subjectivity level

59. abs_title_sentiment_polarity: Absolute polarity level
60. shares: Number of shares (target)

Data Exploration :

Data Analysis :

The Online News Popularity Data Set contains 58 predictors, 2 non-predictors and a response variable, which is a number of shares of an online news article. The 2 non-predictors are the unique URL and number of days between the article publication and the dataset acquisition. The predictors contain information extracted from a news articles, such as number of images, number of videos, title polarity, number of words in the content, rate of positive/negative words in the content etc. Among them, some predictors are categorical: type of article (data channel), day of week when the article was published and whether the article was published on the weekend.

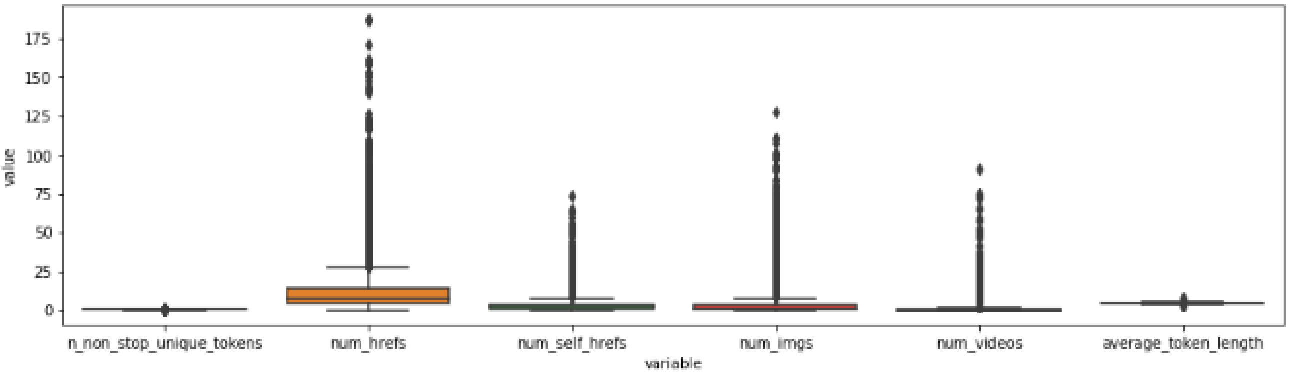
Data Cleaning :

There are a total of 39797 instances in the dataset, which is uniquely identified with a URL. All the 58 predictors and the response variable were analyzed for the detection of missing values, outliers and collinearity between predictors.

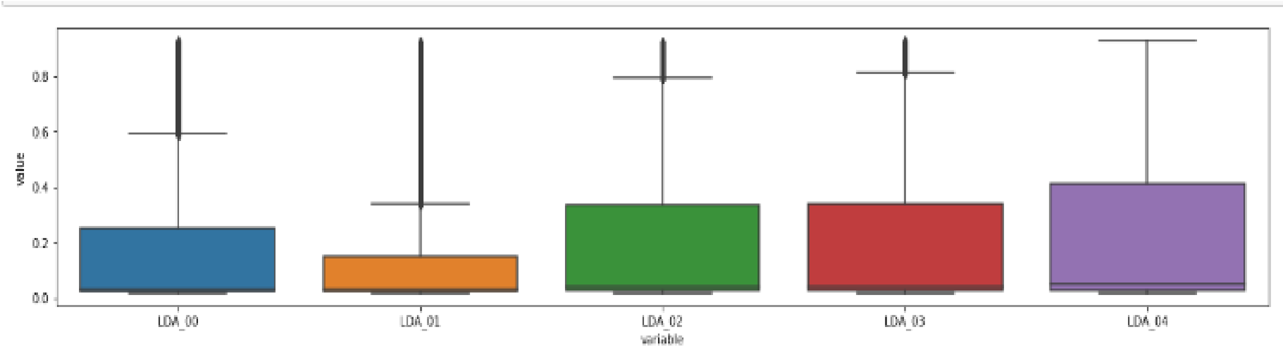
Missing values: The UCI machine learning repository which hosts the current data set has indicated that the data set has no missing values.

Further, while checking the dataset for 0 values, I found 0 values in rate_positive_words, rate_negative _words, and average_token_length. Removed all these

Outliers: From the boxplots and the pair plots, it is clear many attributes contain outliers.

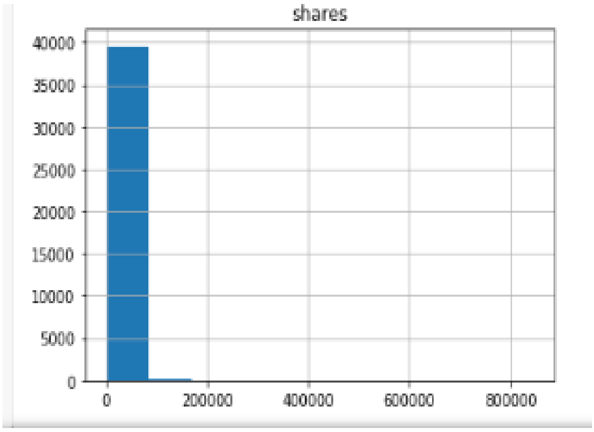


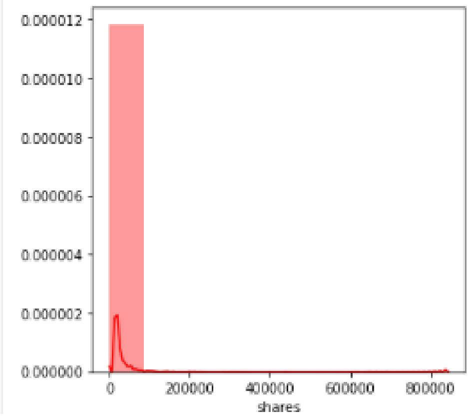
1. All the records in n_non_stop_words lie in the range from 0 to 1, except 1 record which is 1041 and it has been removed as an outlier.
2. 2. The five predictors which represent the closeness of a news article to five topics using Latent Dirichlet Allocation (LDA), have a high number of outliers on the positive side.
3. 2. The five predictors which represent the closeness of a news article to five topics using Latent Dirichlet Allocation (LDA), have a high number of outliers on the positive side.



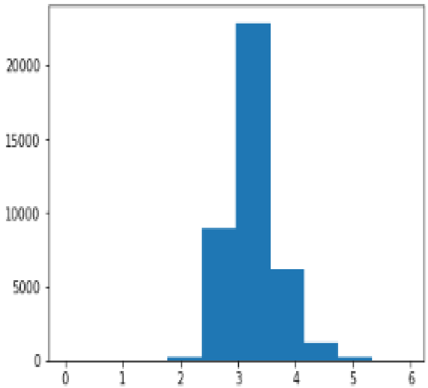
Target Variable :

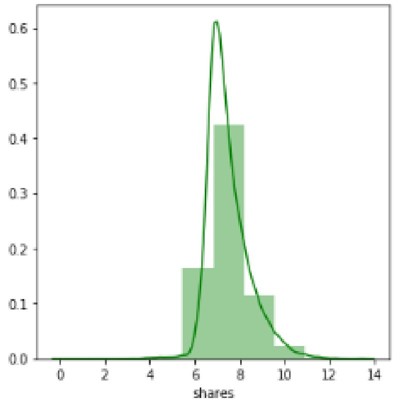
In Online news popularity prediction, the target variable is the number of shares the article has got.





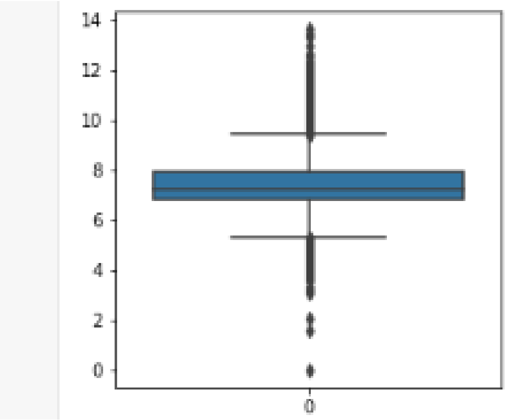
From the above box plot the shares are heavily skewed, and far from being normally distributed. Target variable is normalized using log transformation.

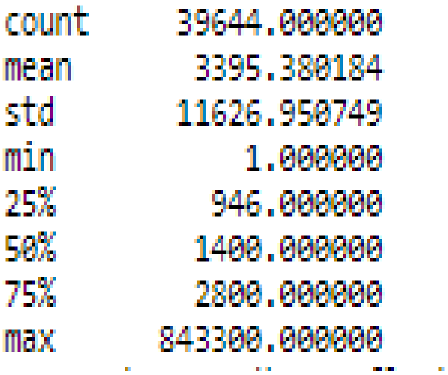




Outliers in the target variable :

One news post has gone viral and the number of shares is more than 8,00,000 contrastingly for another post where the number of shares is only 1. The average number of shares is 3355 and the median lies at 1400.





```
count     39644.000000
mean       3395.380184
std       11626.950749
min           1.000000
25%         946.000000
50%        1400.000000
75%        2800.000000
max      843300.000000
```

Removing outliers in the target:

To remove the outliers I am using the standard deviation method. Three standard deviations from the mean is a common cut-off in practice for identifying outliers in a Gaussian or Gaussian-like distribution.3 Standard Deviations from the Mean will cover 99.7% of the data.

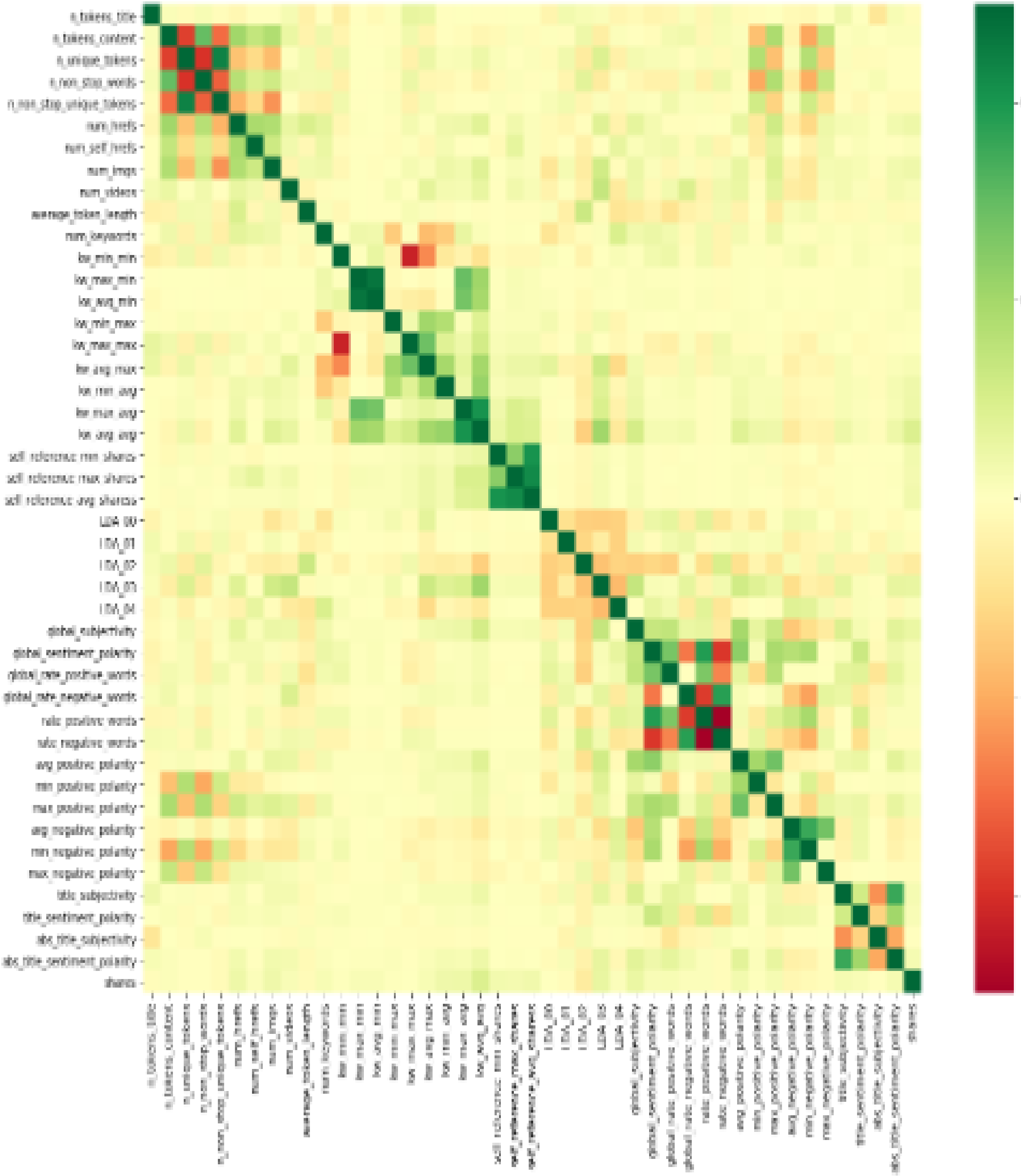
After removing all the outlier the dataset contains 37895 observation, i.e we lost almost 4.5% of the data.

The non-predictive attributes :

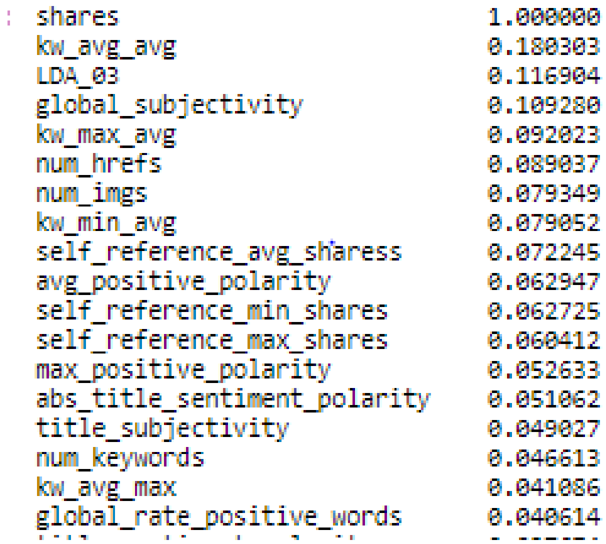
1. URL: Since every URL is unique for each column
2. time delta: Days between the article publication and the dataset acquisition

Correlation between the data :

I have analyzed correlation between the predictors using seaboen heatmaps.


n_tokens_title
n_tokens_content
n_unique_tokens
n_non_stop_words
n_non_stop_unique_tokens
num_hrefs
num_self_hrefs
num_imgs
num_videos
average_token_length
num_keywords
kw_min_min
kw_max_min
kw_avg_min
kw_min_max
kw_max_max
kw_avg_max
kw_min_avg
kw_max_avg
kw_avg_avg
self_reference_min_shares
self_reference_max_shares
self_reference_avg_sharess
LDA_00
LDA_01
LDA_02
LDA_03
LDA_04
global_subjectivity
global_sentiment_polarity
global_rate_positive_words
global_rate_negative_words
rate_positive_words
rate_negative_words
avg_positive_polarity
min_positive_polarity
max_positive_polarity
avg_negative_polarity
min_negative_polarity
max_negative_polarity
title_subjectivity
title_sentiment_polarity
abs_title_subjectivity
abs_title_sentiment_polarity
shares
0.8
0.4
0.0
-0.4
-0.8

The top order of the correlation values with the shares.



```
: shares                          1.000000
  kw_avg_avg                      0.180303
  LDA_03                          0.116904
  global_subjectivity             0.109280
  kw_max_avg                      0.092023
  num_hrefs                       0.089037
  num_imgs                        0.079349
  kw_min_avg                      0.079052
  self_reference_avg_sharess      0.072245
  avg_positive_polarity           0.062947
  self_reference_min_shares       0.062725
  self_reference_max_shares       0.060412
  max_positive_polarity           0.052633
  abs_title_sentiment_polarity    0.051062
  title_subjectivity              0.049027
  num_keywords                    0.046613
  kw_avg_max                      0.041086
  global_rate_positive_words      0.040614
```

There are few attributes are having negative correlation with the target variable means that one variable increases whenever the other decreases.(LDA_02, avg_negative popularity)

Multi-collinearity

Multi-collinearity is a serious problem as it can increase the variance in the coefficient estimation of a predictor and can cause the coefficient to change sign.

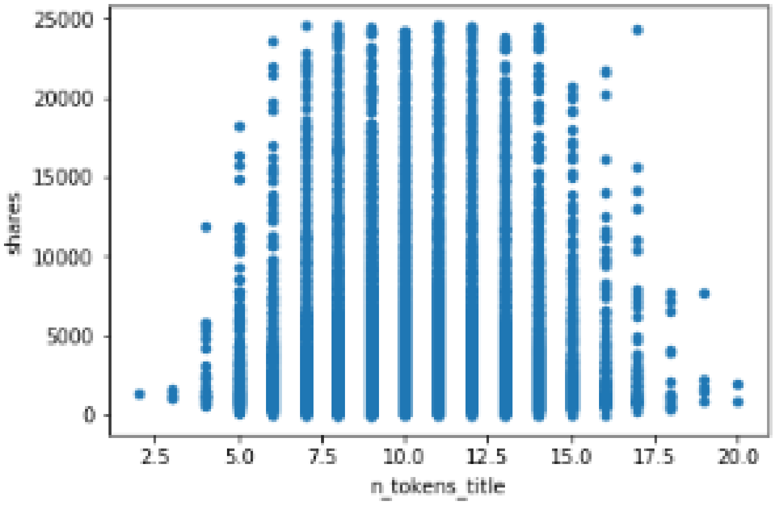
1. n_non_stop_unique_tokens and n_unique_tokens have correlation Those two predictors represent similar information, so n_non_stop_unique_tokens can be removed from the analysis as both the predictors are similar.

2. self_reference_avg_sharess has high correlation with self_reference_min_shares and with self_reference_max_shares. Predictors self_reference_min_shares and self_reference_max_shares are removed can be removed

3. kw_min_avg and kw_min_max have high correlation of 0.986.with each other and hence one of them can be removed. Predictor variable kw_min_max was removed from the analysis.

Relation Between (n_tokens_title)Number of words in the title and number of shares:

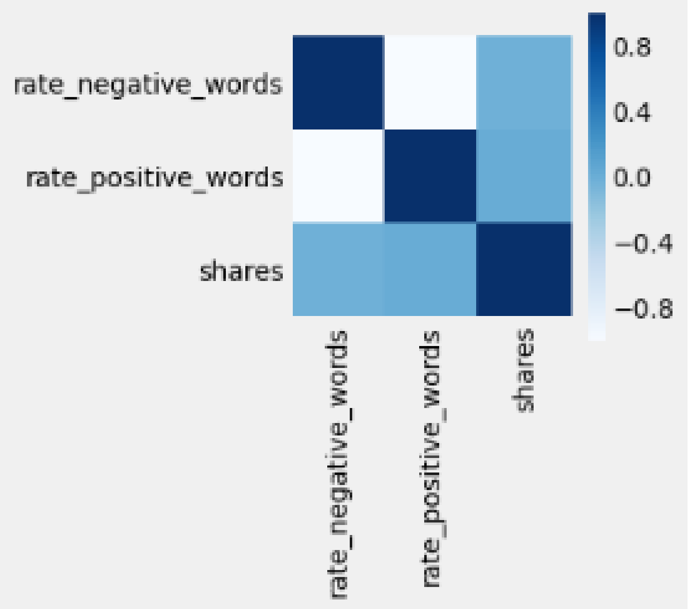
This seems to indicate that , the articles with the most number of shares should have titles that are around 6 to 14 words.



Relation between rate negative words/positive words and number of shares:

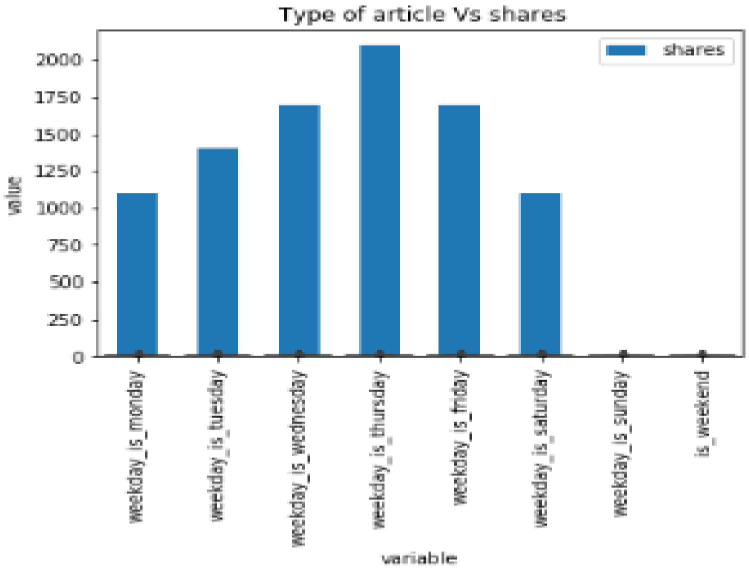
From the scatter plot and the correlation matrix, we can see that a strong positive relation exists between number of shares and positive words.when the rate of positive is between 0.6 to 0.9 the article is getting popular with high number of shares ,

There is a negative relation between negative words and the number of shares. When the rate of negative words is between 0.2 to 0.4 , the article is getting popular , but gradually number of shares are decreasing.



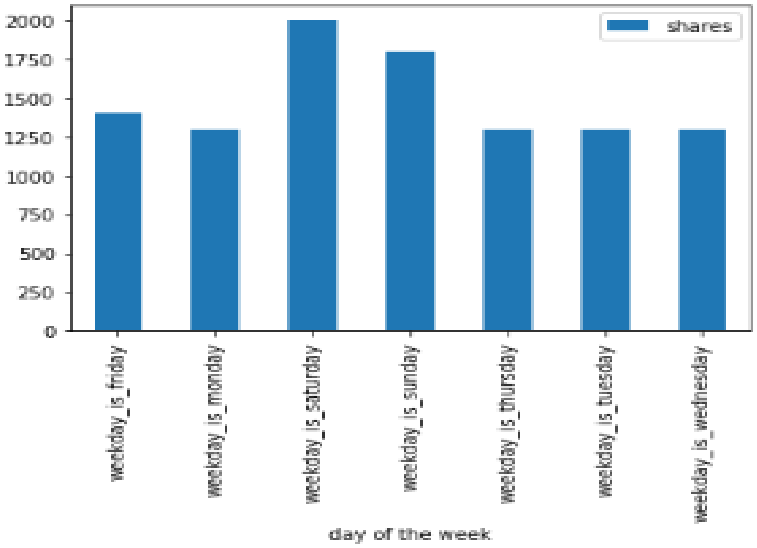
Comparing the Type of the article and the number of shares it gets:

From the bar graph it is evident that articles are getting more number of shares when the data channel is either lifestyle or social media



Comparing the number of shares depending on the day of the week

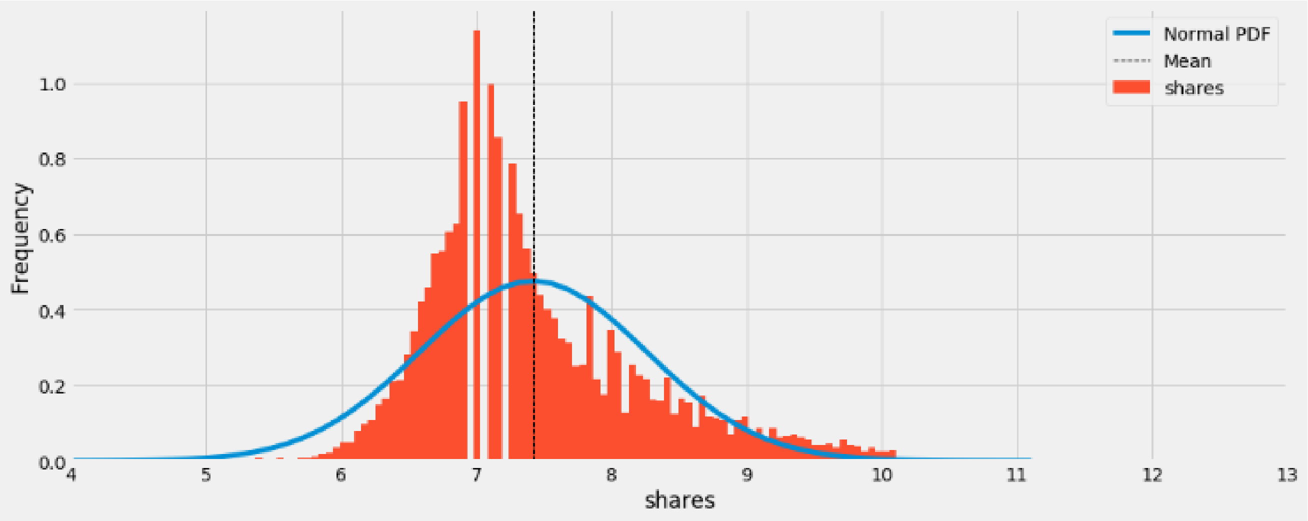
Box plot for the day article was published and the number of shares is plotted, which showed articles posted on weekend has slightly more shares compared to other news posted on weekdays.

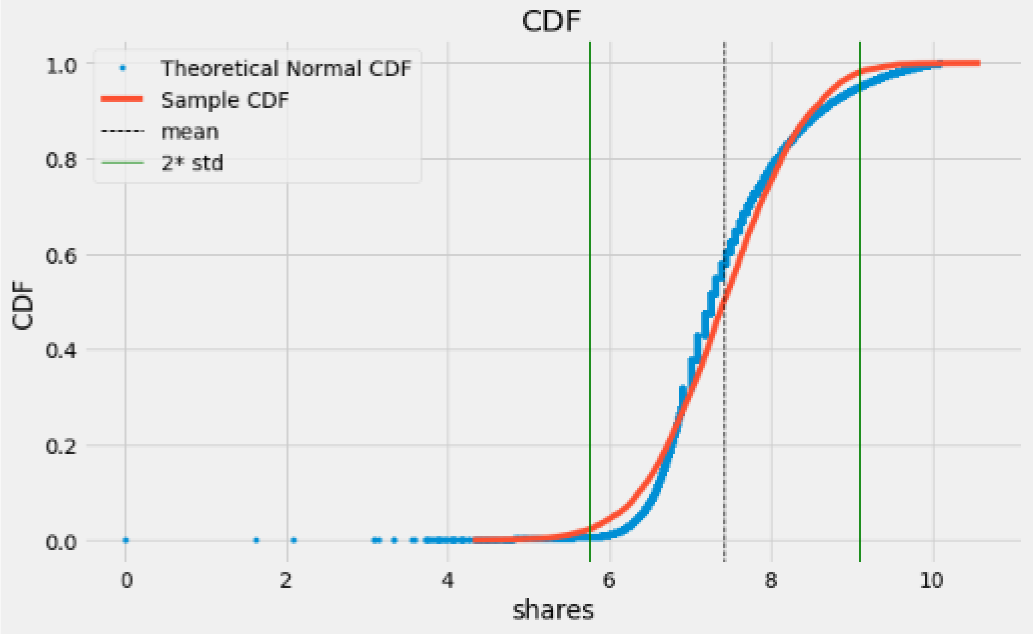


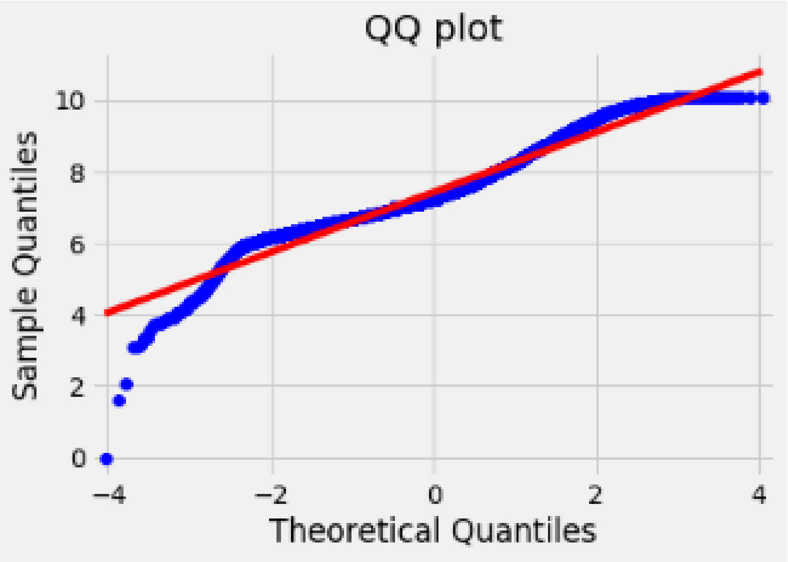
Inferential Statistics :

Normality of target variable :

After doing the log transformation , the target variable show the folliowing distribution.







1 . Characteristic of the normal distribution is that it is symmetrical. This means that if the distribution is cut in half, each side would be the mirror of the other. It also must form a bell-shaped curve to be normal. A bimodal or uniform distribution may be symmetrical; From the Histogram and the PDF plot we can see that the distribution is almost symmetrical and bell-shaped. Mean almost cuts the distribution into half

2 . For Normal distribution the mean , median are equal. Mean = 7.47 Median =7.244

3 . The CDF of the data overlaps with the CDF of a sample set that is modeled as a normal distribution and 68% of observations are within 1 standard deviation and that 95% of observations are within 2 standard deviations.

4 . The QQ plot and the CDF plot , show that the distribution of the data is very close to normal, other than few extreme values.

5 . From the central limit theorem, when sample size is greater than or equal to, 30, we can treat the sampling distribution of as approximately normal regardless of the shape of the parent population.

6 . For normal distribution we need individual observations to be independent.(our observations are independent)

Is there a significant difference mean number of shares on a weekday and a weekend:

I am using “ **A two -sample bootstrap hypothesis test”**

Null Hypothesis: There is no difference in mean shares between the weekday and weekend

Alternate Hypothesis: Mean of shares weekday and weekend are different

alpha = 0.05

P value obtained = 0, we reject the null Hypothesis, i.e there is a difference between the number of shares on weekday and weekend

Test whether the number of images significantly affects “number of shares”

Hypothesis test on the Pearson coefficient:

H0: number of images does not affect the number of shares

H1: Number of images affect the number of shares

The final p-value =0, we reject the null hypothesis, i.e the number of images do affect the number of shares

Metrics

As a classification task, we will adopt the following three evaluation metrics: accuracy, F1-score and AUC. For all three metrics, the higher value of the metric means the better performance of the model.

(a) Accuracy: Accuracy is a direct indication of the proportion of correct classification. It considers both true positives and true negatives with equal weight and it can be computed as

accuracy =(true positives + true negatives) / dataset size

Although the measure of accuracy might be naive when the data class distribution is highly skewed, but it is still an intuitive indication of the model's performance.

(b) F1-score: F1-score is an unweighted measure for accuracy by taking harmonic mean of precision and recall, which can be computed as

F1 = (2 * precision * recall) / (precision + recall)

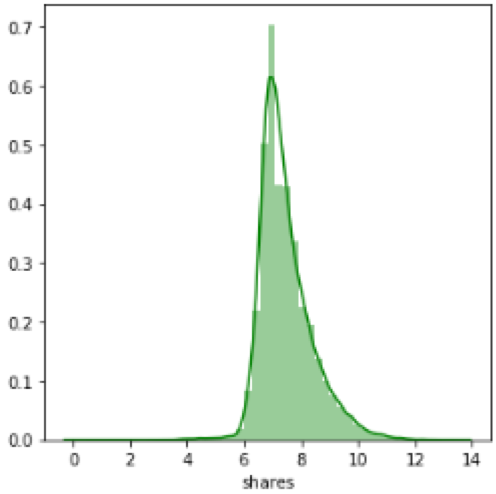
It is a robust measurement since it is independent of data class distribution.

(c) AUC: The AUC is the area under the ROC (Receiver Operating Characteristics) curve, which is a plot of the True Positive Rate versus the False Positive Rate. AUC value is a good measure of classier's discrimination power and it is a more robust measure for model performance. The larger value of the performance metric area under the ROC curve (AUC) indicates the higher popularity prediction accuracy.

Target Label :

This target column(shares) will be the column your classifier will predict, using a combination of the other columns. The objective of this project is to predict whether an article will be popular or non-popular, so I see this as a classification problem.

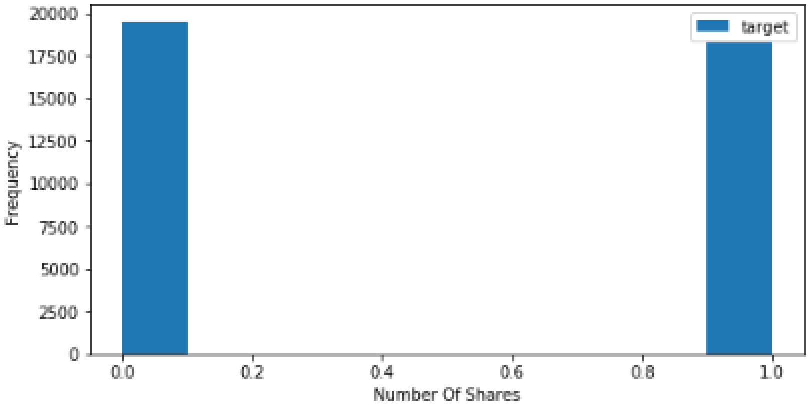
Now we have to create the target label.



As one can see that the distribution is heavily skewed towards the right and hence picking the mean to label the target would not be correct.

We use the median of the distribution to divide the shares into two classes: popular, unpopular. The median of shares is 1400.

shares > 1400 = 1 , popular
shares <= 1400 = 0 , unpouplar



Models:

Since the problem has been reduced to a classification problem, we intend to use them best-performing out of the following classification algorithms:

1) Logistic Regression
2) Support Vector Machines (SVM)
3) k – Nearest Neighbor (kNN)
4) Random Forests
5) Decision Trees
6) Gradient Boost

Parameters:

We will use cross-validation in order to optimize the parameters for the above algorithms. For example, for SVM, we will compare different kernels, and for K-Nearest Neighbors, we will compare different values of k. We will then run our model using the parameter value that gives us the best accuracy.

Evaluation:

We will create a confusion matrix based on our model. Based on this matrix, we can calculate the accuracy, precision, recall, and F-score. Using these metrics, we can then plot the ROC and AUC curves.

Normalizing the data:

Normalization is a technique often applied as part of data preparation for machine learning. The goal of normalization is to change the values of numeric columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information.

The great difference in the *scale* of the numbers could cause problems when you attempt to combine the values as features during modeling.

Normalization avoids these problems by creating new values that maintain the general distribution and ratios in the source data while keeping values within a scale applied across all numeric columns used in the model.

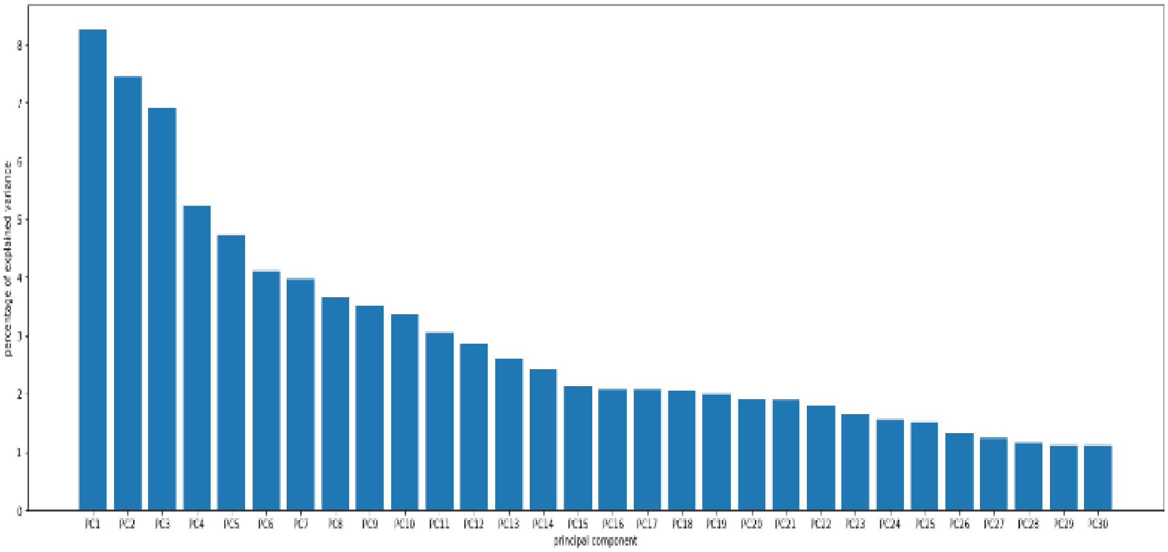
Dimensionality Reduction

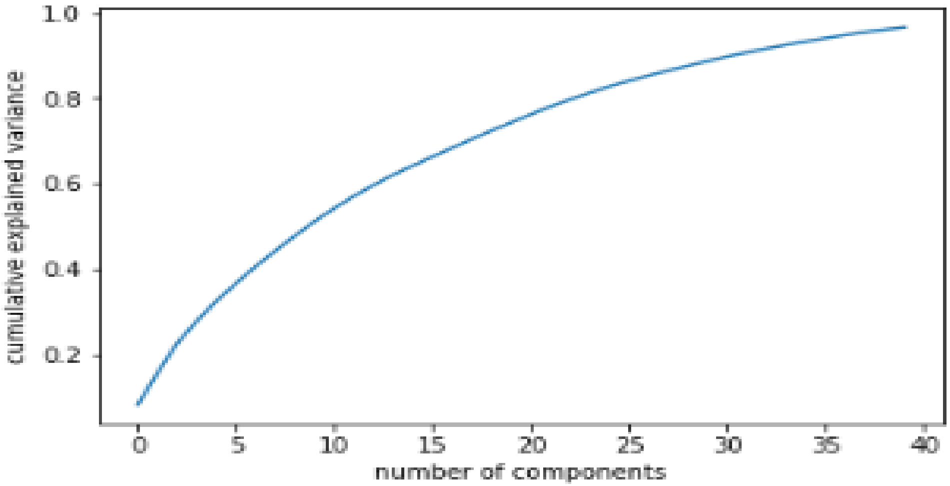
In machine learning classification problems, there are often too many factors on the basis of which the final classification is done. These factors are basically variables called features. The higher the number of features, the harder it gets to visualize the training set and then work on it. Sometimes, most of these features are correlated, and hence redundant. This is where dimensionality reduction algorithms come into play. Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables. The method I am using for dimensionality reduction is Principal Component Analysis (PCA)

Principal Component Analysis (PCA) :

This technique extracts some amount of the features from the original dataset. These extracted features/variables are called Principal Components. The main motive behind this technique is to extract a low dimensional set of features from a high dimensional dataset. And it is more useful when we are dealing with 3 or high dimensional data.

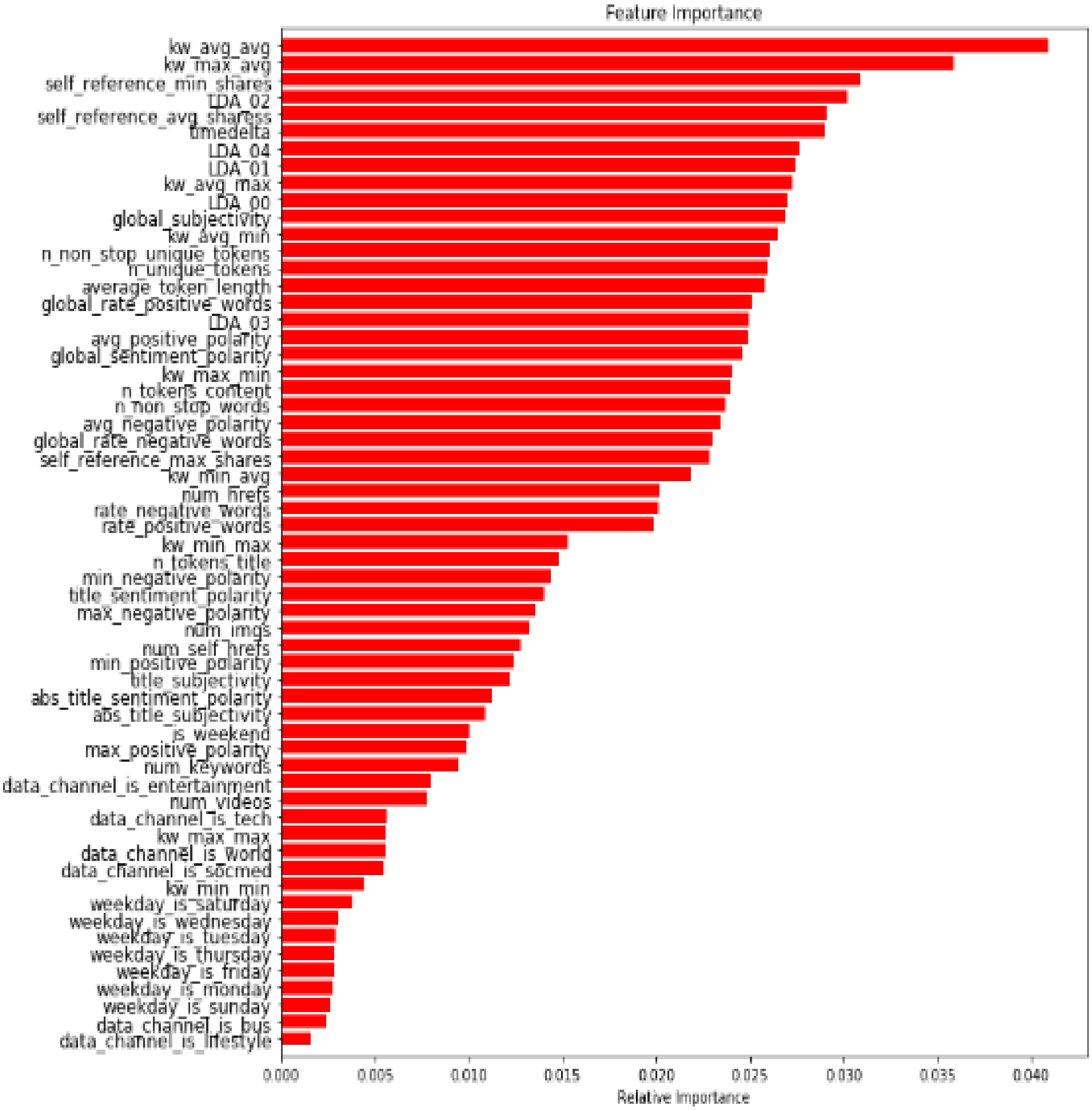
- A principal component is a linear combination of the original variables
- Principal components are extracted in such a way that the first principal component explains maximum variance in the dataset
- Second principal component tries to explain the remaining variance in the dataset and is uncorrelated to the first principal component
- Third principal component tries to explain the variance which is not explained by the first two principal components and so on





Feature Selection Using Random Forest

Random Forests are often used for feature selection in a data science workflow. The reason is that the tree-based strategies used by random forests naturally rank by how well they improve the purity of the node. This means a decrease in impurity over all trees (called Gini impurity). Nodes with the greatest decrease in impurity happen at the start of the trees, while notes with the least decrease in impurity occur at the end of trees. Thus, by pruning trees below a particular node, we can create a subset of the most important features.

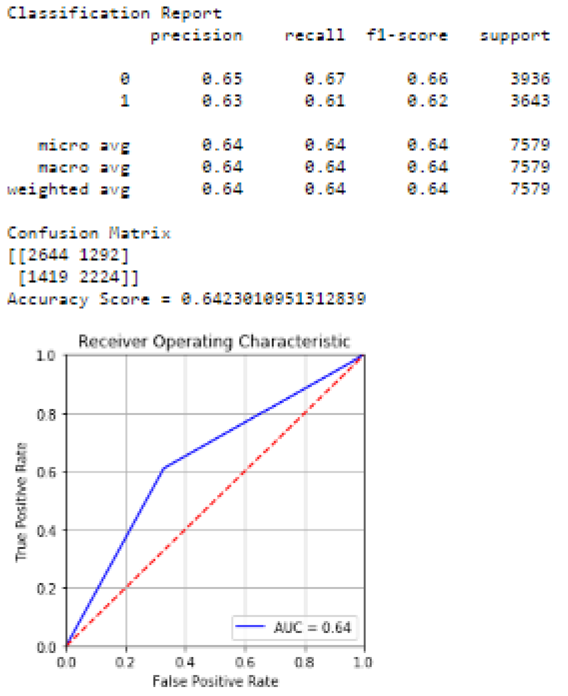


I divided the dataset into a training set which contains 70% of the dataset and testing set which contains the remaining 30% of the dataset. The training set is used for applying algorithms whereas the testing set is used to predict the accuracy of the model. In the end, I will compare the accuracy of all the models and will find the best-predicted model.

Logistic Regression

Logistic regression is a linear model for classification. In this model, the output of a single trial can be interpreted as a class probability, which is generated by a logistic function or sigmoid function. Sklearn provides a function called LogisticRegression(). One typical tunable hyperparameter is "C", which is the inverse of regularization strength. The advantage of logistic regression n is that the training and predicting speed is very fast.

The accuracy achieved by the logistic regressor model is 60%. Then I used Grid Search and tuned the hyperparameters and cross-validated the model and achieved an accuracy of 0.642 and AUC 0.64

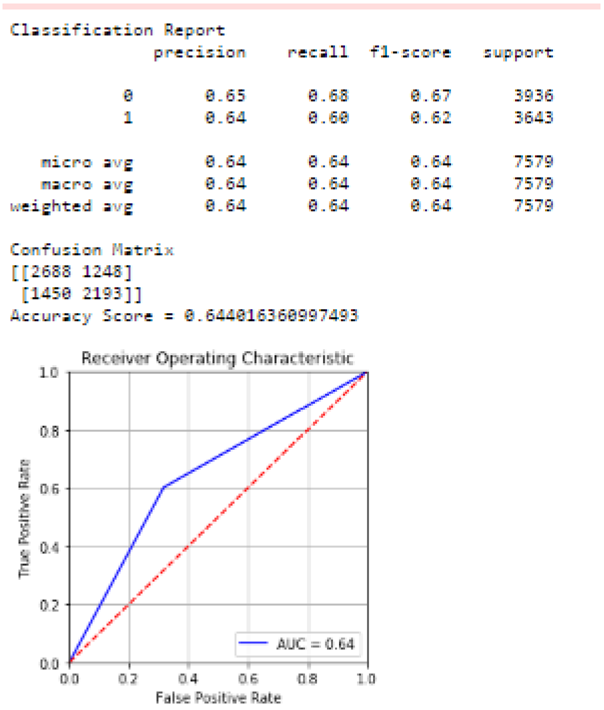


```
Classification Report
              precision    recall  f1-score   support

           0       0.65      0.67      0.66      3936
           1       0.63      0.61      0.62      3643

   micro avg       0.64      0.64      0.64      7579
   macro avg       0.64      0.64      0.64      7579
weighted avg       0.64      0.64      0.64      7579

Confusion Matrix
[[2644 1292]
 [1419 2224]]
Accuracy Score = 0.6423010951312839
```

Support Vector Machine

The next supervised machine learning model which I have applied to my dataset is Support Vector Machine. This algorithm outputs an optimal hyperplane dividing plane into two parts wherein each class can lay on the other side. In my dataset, Support Vector Machine divides hyperplane into

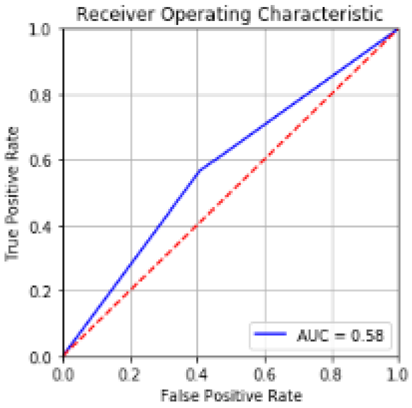
two parts i.e. popular and not popular and provide optimal hyperplane. The accuracy achieved by using Support Vector Machine is 64%.



```
Classification Report
              precision    recall  f1-score   support

           0       0.65      0.68      0.67      3936
           1       0.64      0.60      0.62      3643

   micro avg       0.64      0.64      0.64      7579
   macro avg       0.64      0.64      0.64      7579
weighted avg       0.64      0.64      0.64      7579

Confusion Matrix
[[2688 1248]
 [1450 2193]]
Accuracy Score = 0.644016360997493
```

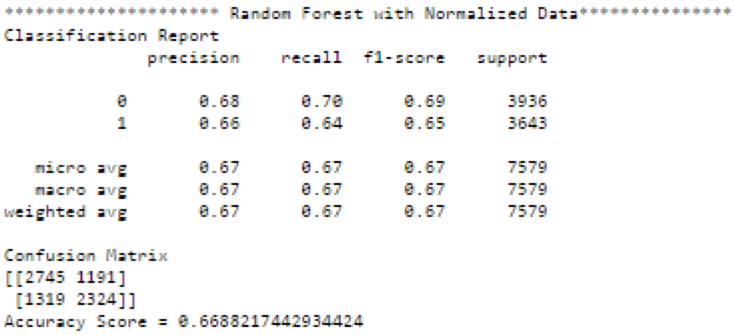
Decision Tree Classifier

Decision Tree, as its name says, makes a decision with a tree-like model. It splits the sample into two or more homogeneous sets (leaves) based on the most significant differentiators in your input variables. To choose a differentiator (predictor), the algorithm considers all features and does a binary split on them (for categorical data, split by cat; for continuous, pick a cut-off threshold). It will then choose the one with the least cost (i.e. highest accuracy), and repeats recursively until it successfully splits the data in all leaves (or reaches the maximum depth).

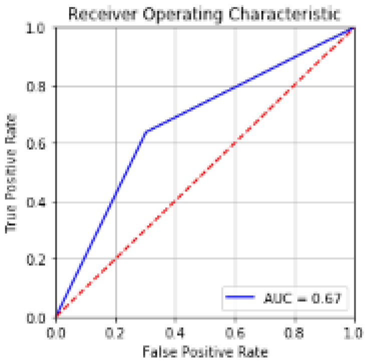


Random Forest Classifier

Random forest is an ensemble model that grows multiple trees and classify objects based on the “votes” of all the trees. i.e. An object is assigned to a class that has most votes from all the trees. Random forest classifier is a meta-estimator that fits a number of decision trees on various sub-samples of datasets and uses the average to improve the predictive accuracy of the model and controls over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement.



```
********************* Random Forest with Normalized Data***************
Classification Report
              precision    recall  f1-score   support

           0       0.68      0.70      0.69      3936
           1       0.66      0.64      0.65      3643

   micro avg       0.67      0.67      0.67      7579
   macro avg       0.67      0.67      0.67      7579
weighted avg       0.67      0.67      0.67      7579

Confusion Matrix
[[2745 1191]
 [1319 2324]]
Accuracy Score = 0.6688217442934424
```

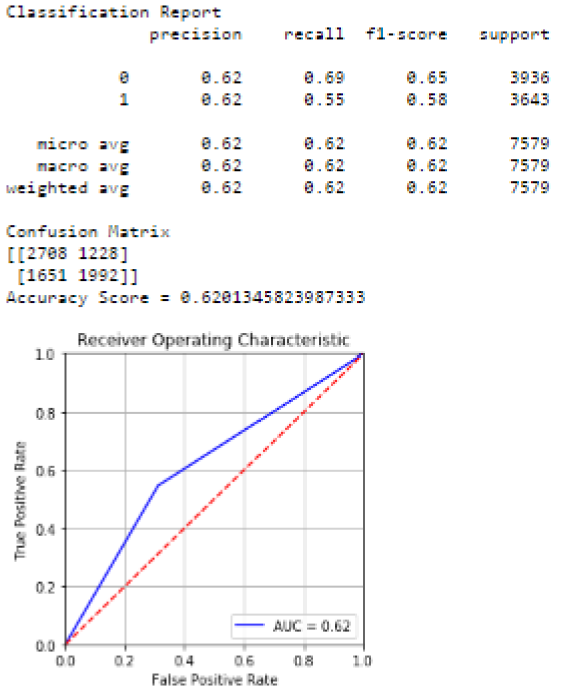


K-NEAREST NEIGHBOUR (KNN):

KNN classifies an object by a majority vote of the object’s neighbors, in the space of the input parameter. The object is assigned to the class which is most common among its k (an integer specified by a human) nearest neighbor.

It is a non-parametric, lazy algorithm. It’s non-parametric since it does not make any assumption on data distribution (the data does not have to be normally distributed). It is lazy since it does not really learn any model and make generalization of the data (It does not train some parameters of some function where input X gives output y). It simply classifies objects based on feature similarity (feature = input variables). Classification is computed from a simple majority vote of the k nearest neighbors of each point.

The accuracy achieved with normalized data is 0.58, and after hyperparameters tuning using grid search the accuracy was 0.62.

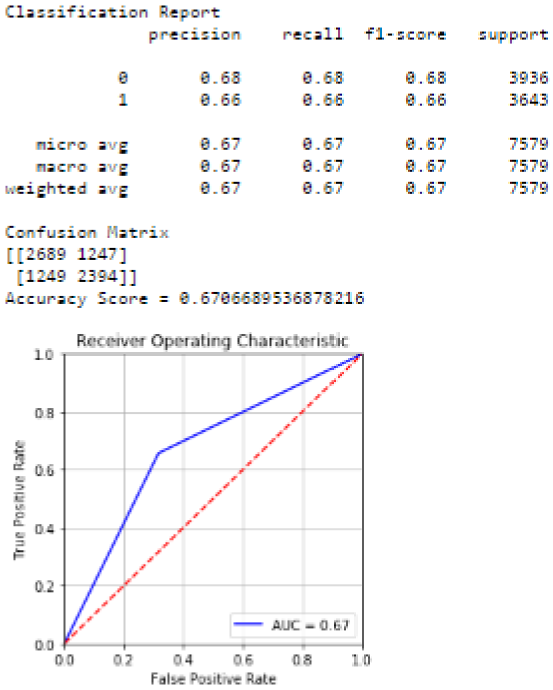


```
Classification Report
              precision    recall  f1-score   support

           0       0.62      0.69      0.65      3936
           1       0.62      0.55      0.58      3643

   micro avg       0.62      0.62      0.62      7579
   macro avg       0.62      0.62      0.62      7579
weighted avg       0.62      0.62      0.62      7579

Confusion Matrix
[[2708 1228]
 [1651 1992]]
Accuracy Score = 0.6201345823987333
```

Gradient Boost Classifier

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

The accuracy achieved with gradient boosting is 0.67



```
Classification Report
              precision    recall  f1-score   support

           0       0.68      0.68      0.68      3936
           1       0.66      0.66      0.66      3643

   micro avg       0.67      0.67      0.67      7579
   macro avg       0.67      0.67      0.67      7579
weighted avg       0.67      0.67      0.67      7579

Confusion Matrix
[[2689 1247]
 [1249 2394]]
Accuracy Score = 0.6706689536878216
```

Model tuning

Tuning is the process of maximizing a model’s performance without overfitting or creating too high of a variance. In machine learning, this is accomplished by selecting appropriate hyperparameters.”

Choosing an appropriate set of hyperparameters is crucial for model accuracy, but can be computationally challenging.

I am using the Grid Search method for selecting appropriate hyperparameters. This method involves manually defining a subset of the hyperparametric space and exhausting all combinations of the specified hyperparameter subsets. Each combination’s performance is then evaluated, typically using cross-validation, and the best performing hyperparametric combination is chosen.

Results

Model	Original Data	After PCA	After feature selection	Grid Search and Hyperparameters tuning
Logistic Regression	0.595	0.642	0.628	0.65
KNN	0.587	0.631	0.62	0.649
Decision Trees	0.584	0.569	0.567	0.580
Random Forest	0.667	0.638	0.64	0.68
SVM	0.498	0.638	0.64	
Gradient Boost	0.671			0.69

Recommendations :

One way of dealing with such type of data is an improvement in the feature selection as there is very little room in model selection. We could utilize other state-of-the-art feature selection methods to improve the selection of a combination of features for better performance.

As the number of samples is high enough for a neural network to learn. We could build a neural network that could learn the training data and can be done for the hyperparameters of the network for better performance. With advancements in neural networks and deep neural networks, we could utilize state-of-the-art methodologies for a better evaluation of the samples.