ONLINE NEWS POPULARITY PREDICTION

INTRODUCTION

- ➔ The dataset summarizes a set of features about articles published by Mashable , a news website over a period of two years.
- ➔ Data Source : UCI ML Repository (https://archive.ics.uci.edu/ml/datasets/Online+News+Popularity)
- ➔ Number of attributes : 61
- ➔ Number of records : 39645
- ➔ Target variable : Number of shares
- ➔ No missing values
- ➔ The objective is to predict if an article will be popular or not

Data Description

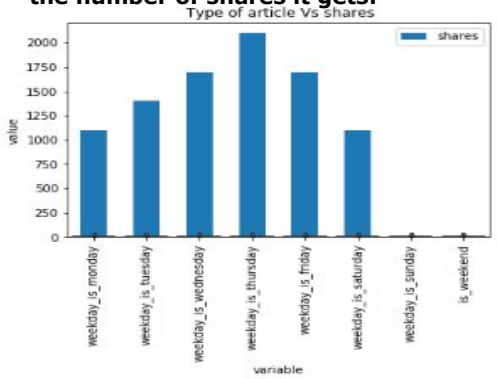
0. url: URL of the article (non-predictive)
1. timedelta: Days between the article publication and the dataset acquisition (non-predictive)
2. n_tokens_title: Number of words in the title
3. n_tokens_content: Number of words in the content
4. n_unique_tokens: Rate of unique words in the content
5. n_non_stop_words: 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

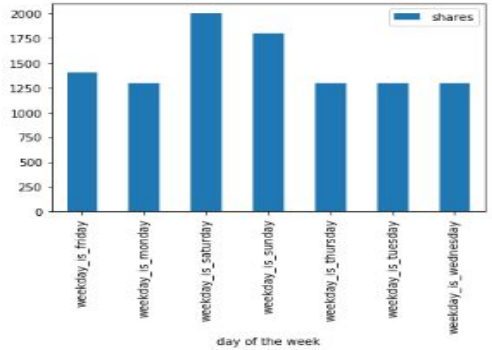
Data cleaning : There are no missing values. Outliers are present in many attributes including the target variable, I used the standard deviation method to remove the outliers

Plotted the graphs to visualize the relation between the variables

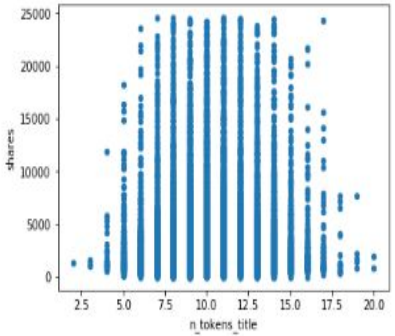
Comparing the Type of the article and the number of shares it gets:



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



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



Target Variable

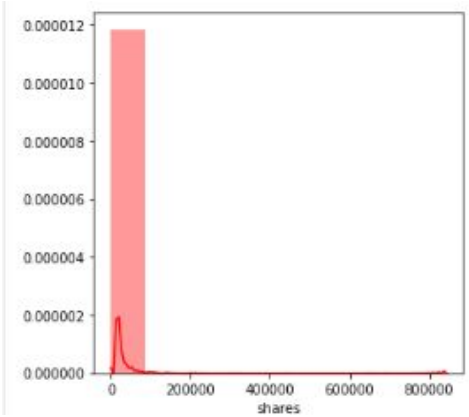
Used the log transformation on the target variable to make it normally distributed.This works better for classification model as classification involves the log transform.

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

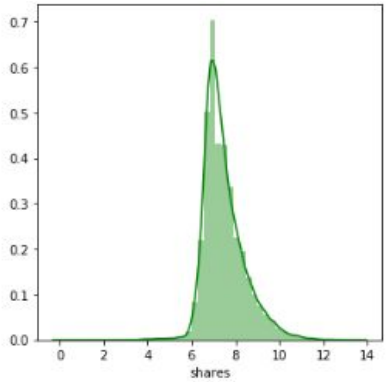
shares > 1400 = 1 , popular

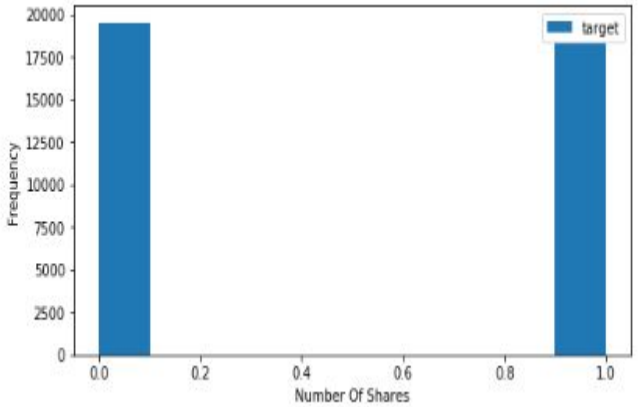
shares <= 1400 = 0 , unpouplar

Before Log transformation



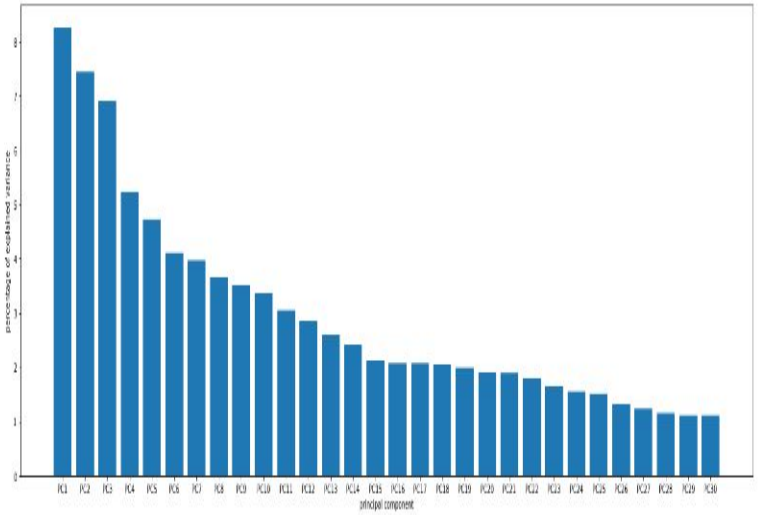
After log transformation

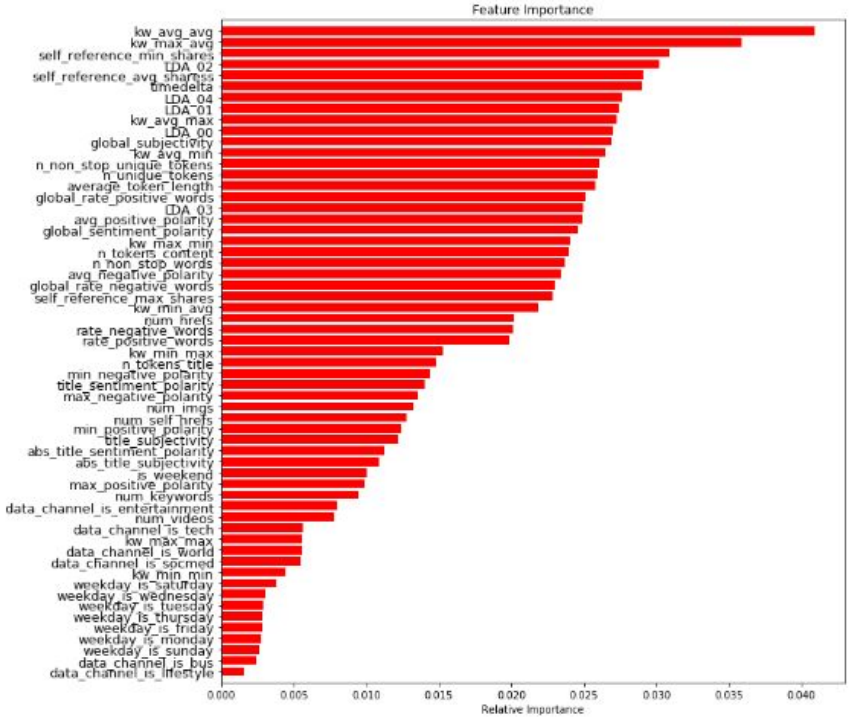




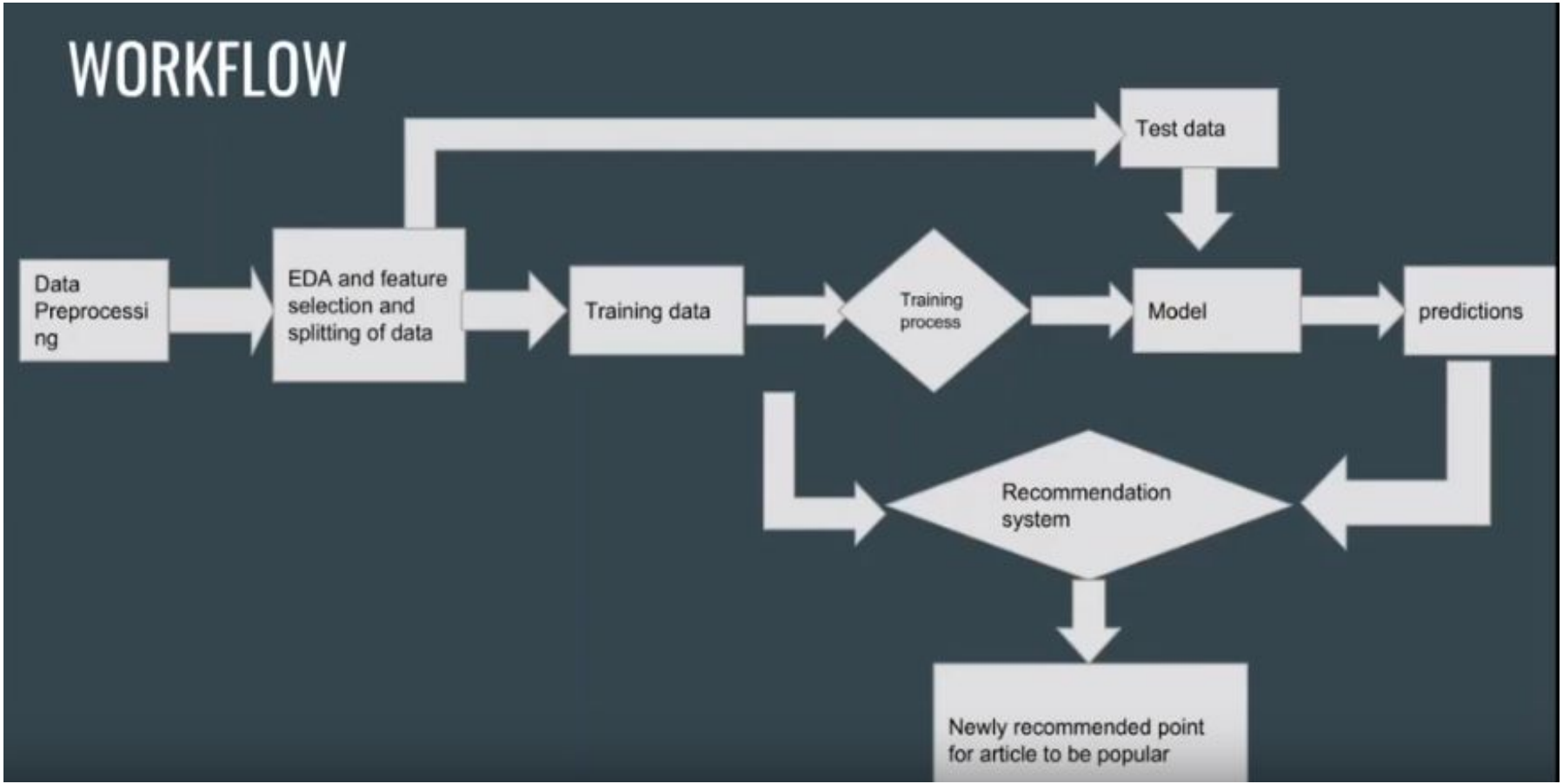
Data Preparation

- ➔ Normalization is applied as part of data preparation.
- ➔ Applied PCA to the dataset as a dimensionality reduction process , and used the top 30 features as the principal components. But the PCA did not give better results.
- ➔ Applied Random Forest Features selection method ,
  and use the top 30 important features ,
  which gave better results.
- ➔ I divided the dataset into a training set which contains 70% ,
  and testing set which contains the remaining 30% of the dataset.





Workflow


WORKFLOW
Test data
Data Preprocessi ng
EDA and feature selection and splitting of data
Training data
Training process
Model
predictions
Recommendation system
Newly recommended point for article to be popular

Models Used

- I used different classification models to train our dataset and predict the results on the test data.
  - Logistic Regression
  - K Nearest Neighbors
  - Decision Trees
  - Random Forest
  - Support Vector machine
  - Gradient Boosting method

I am using the Grid Search method on these models tuning the models and selecting the best performing hyperparameters.

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.

Accuracy achieved with normalized data is 0.59, and after hyperparameters tuning using gridsearch the accuracy was 0.64.

Accuracy after tuning the hyperparameter

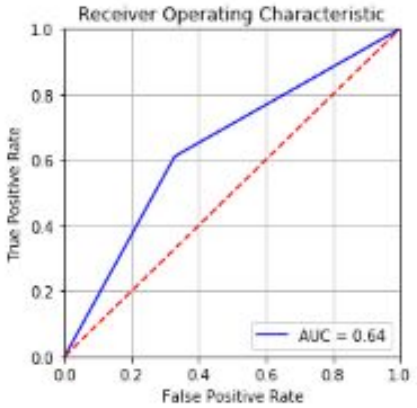
Classification Report

	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



K Nearest Neighbors

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 nearest neighbor.

Accuracy achieved with normalized data is 0.58, and after hyperparameters tuning using gridsearch the accuracy was 0.62.

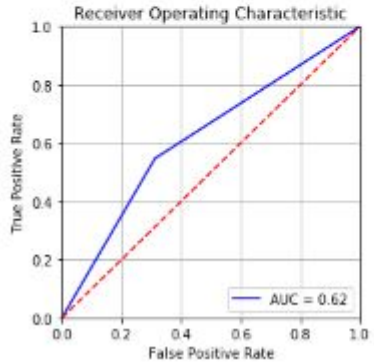
Classification Report

	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



Random Forest

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.

Accuracy achieved with Random Forest model with best parameters 0.67.

********** Random Forest with Normalized Data**********

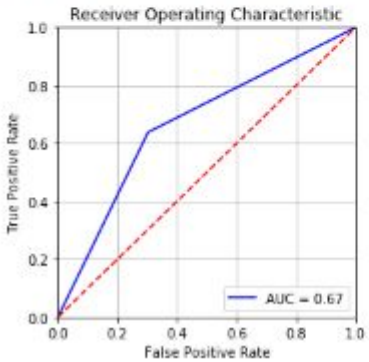
Classification Report

	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



Support Vector machine(SVM)

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 maximum accuracy achieved by using Support Vector Machine is 64%.

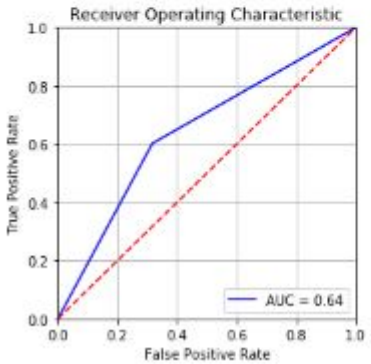
Classification Report

	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



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

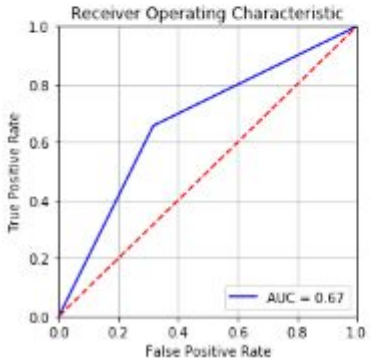
Classification Report

	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



Result

As we can see, over all the algorithms applied, Random forest and the Gradient boosting classifiers performed really well over such kind of data.

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 improvement in the feature selection as there is very little room in model selection. We could utilise other state-of-the-art feature selection methods to improve the selection of combination of features for better performance.

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