Project Report
Group 3

Arjun Acharya
Data Analytics Engineering, M.S.
acharya.ar@northeastern.edu

Rohan Arora
Data Analytics Engineering, M.S.
arora.roha@northeastern.edu

Sukanya Aswini Dutta
Data Analytics Engineering, M.S.
dutta.su@northeastern.edu

Online Popularity of news articles

A. Acharya, R. Arora, S. Dutta
Northeastern University, IE 7374 - Machine Learning In Engineering, April 2021

Abstract
With the rapid growth of the online news services and social media, more and more people enjoy reading and sharing online news articles. The number of shares under a news article indicates how popular the news is. It is very beneficial to shed light on readers' intentions and to predict the popularity of the online news, which implies, whether, the online news will receive a significant amount of reader's attention. In this project, we aimed to find the best model and set of features to predict the popularity of online news, using machine learning techniques. Our datasource was Mashable, a well-known online news website which has a large and recently collected dataset with over 39000 articles. We implemented 5 different learning algorithms on the dataset, ranging from various regressions to neural networks. Their performances are recorded and compared. Feature selection methods are used to improve performance and reduce features. Logistic Regression turns out to be the best model for prediction, and it can achieve an accuracy of 0.64 with optimal parameters. Our work can help online news companies to predict news popularity before publication.

Machine learning algorithms can play an important role to process, analyze, understand and predict the best choice for the task depending on some or all features. Various methods are available for feature selection. The most famous machine learning algorithms are the regression algorithms, Decision Tree, Random Forest Classification, Naive Bayes, Support Vector Machine (SVM), Artificial Neural Network (ANN), etc. This project uses most of these machine learning algorithms utilizes Precision, Recall and F-measure as an evaluation measurements for results in order to find the best one. The results are compared with the previous work on the same dataset.

I. Introduction

In this information era, reading and sharing news have become the center of people's entertainment lives. Therefore, it would be greatly helpful if we could accurately predict the popularity of news prior to its publication, for social media workers (authors, advertisers, etc). For the purpose of this paper, we intend to make use of a largely and recently collected dataset with over 39000 articles from Mashable website, to first select informative features and then analyze and compare the performance of several machine learning algorithms. Some prediction approaches are based on analyzing early users' comments, or features about post contents and domains. Another proposed method predicted the article's popularity not only based on its own

appeal, but also other articles that it is competing with. Prediction models with SVMs are investigated, and more advanced algorithms such as Random Forest could increase the precision. This project however, incorporates a broader and more abstract set of features, and starts with basic regression and classification models to advanced ones, with elaborations about effective feature selection. This paper has the following structure. Section II introduces our dataset and feature selection. Section III gives our implementation of various learning algorithms. We analyze the result and compare the performances in Section IV. In Section V, we discuss possible future work.

II. Dataset Description

Data collection

This project aims to find a method to predict the popularity of an online article before it is published by using several statistical characteristics summarized from it. We use a dataset from UCI Machine Learning Repository. This dataset summarizes a heterogeneous set of features about articles published by Mashable in two years. The goal is to predict the number of shares in social networks (popularity). It contains 39767 observations with 61 variables, including 2 variables that are not used in other studies involving this dataset. We use one of these additional variables to build our predictive model.

The input of the algorithm is several features of Mashable articles: Words (e.g. number of words in the title), Links (e.g. number of Mashable article links), Digital Media (e.g. number of images), Time (e.g. day of the week), Keywords (e.g. number of keywords) and Natural Language Processing (e.g. closeness to top 5 LDA topics).

Number of Attributes: 61 (58 predictive attributes, 2 non-predictive, 1 goal field)

The following summarizes the attribute information:

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

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

53. max_positive_polarity: Max. polarity of positive words
54. avg_negative_polarity: Avg. polarity of negative words
55. min_negative_polarity: Min. polarity of negative words
56. max_negative_polarity: Max. polarity of negative words
57. title_subjectivity: Title subjectivity
58. title_sentiment_polarity: Title polarity
59. abs_title_subjectivity: Absolute subjectivity level
60. abs_title_sentiment_polarity: Absolute polarity level
61. shares: Number of shares (target)

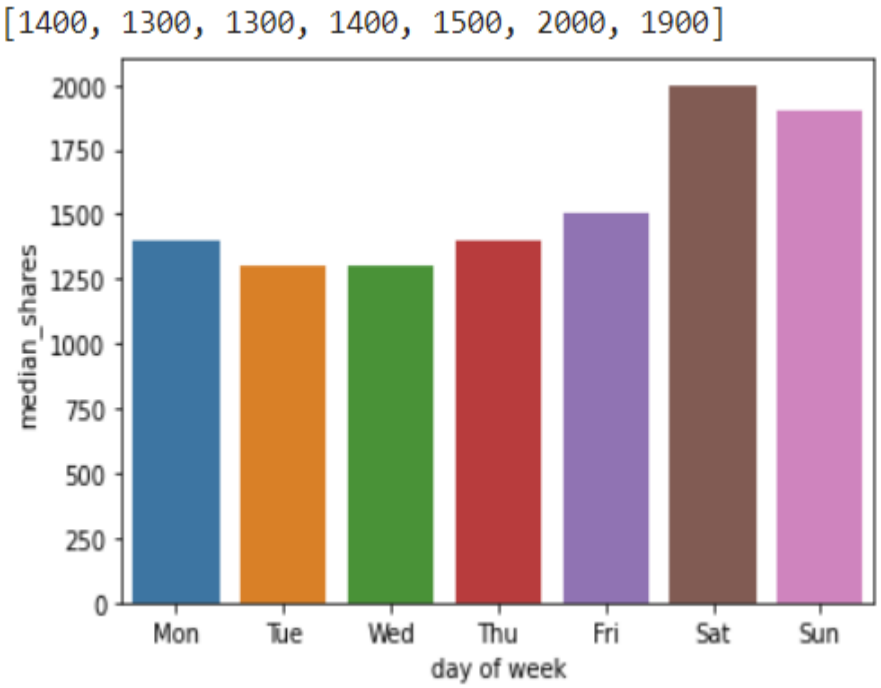
The entire data has been randomly divided into training and testing set. 70% of the data has been used for training the remaining data has been used for finding generalization error of the models.

Exploratory Data Analysis

The UCI machine learning repository hosts the current data set which has indicated that the dataset has no null values. Taking this into account, we haven't performed any missing value analysis.

Below, we have few of the data analysis graphs and our findings:

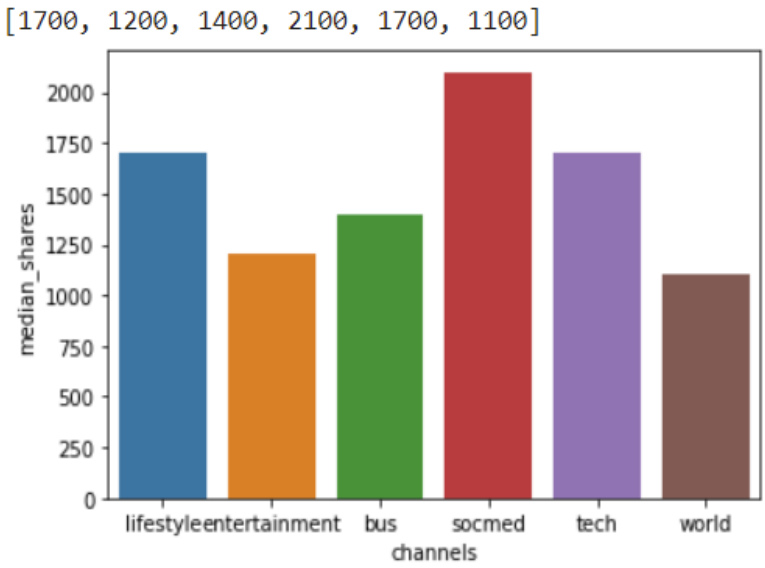
The following graph gives information about the number of shares received by articles published on each week day.



From the graph above, the articles published in the middle of the week received fewer shares than the articles published during the week-end. So, this important insight can be used by the publishers to publish new articles during the weekend so that they become more popular.

We can see articles published on Saturday and Sunday have the highest number of shares.

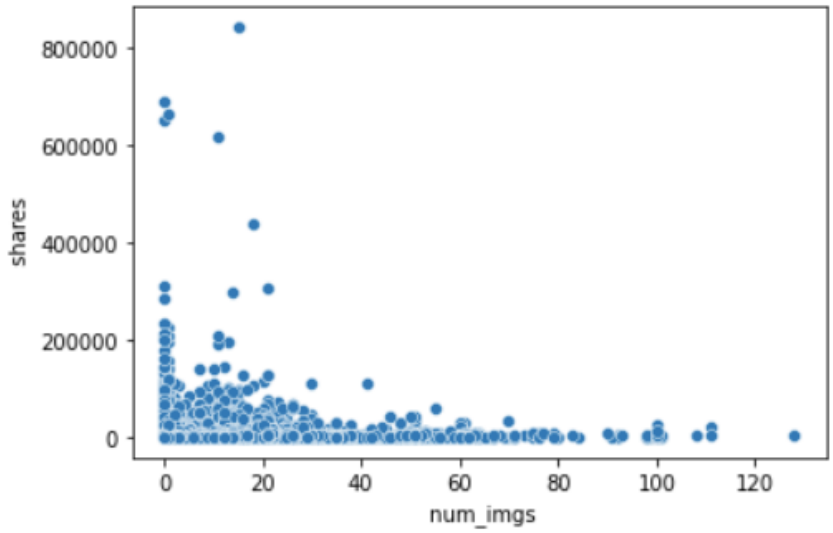
The news articles in the dataset belong to one of the six predefined categories i.e. (Lifestyle, Entertainment, Bus, Social Media, Technology and World). For each of this category, the median shares received in the social media has been calculated and the results are displayed below:



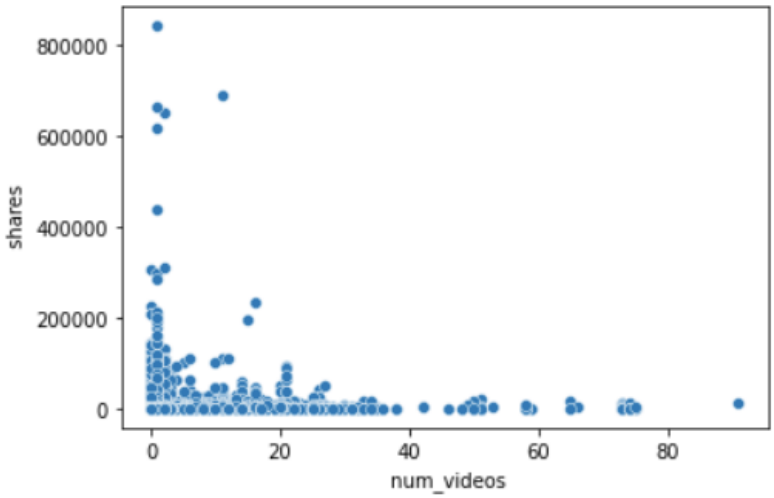
We can see that amongst the 6 categories news articles belonging to the ‘social media’ category have received the highest popularity in digital media.

We can see that articles belonging to social media received the highest popularity.

The following graph shows the distribution of target variable ‘shares’.



From the graph we can see that the distribution of the target variable is right skewed. So, the mean of shares may not be a good measure of the center. This suggests an important action (log transformation) that can be taken during preprocessing.



Feature Selection

1. Dropping URL and Time Delta as it has no predictive power
2. Dropping ' weekday_is_saturday', ' weekday_is_sunday' as we have 'is_weekend' column which denotes the same thing.
3. Removing 'kw_min_min', 'kw_max_min', 'kw_min_max', 'kw_max_max','kw_min_avg', 'kw_max_avg' , 'self_reference_min_shares', 'self_reference_max_shares',

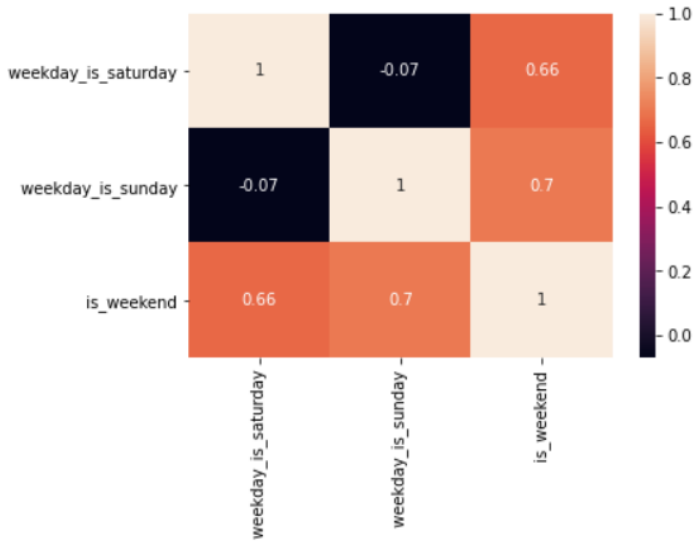
'min_positive_polarity', 'max_positive_polarity','min_negative_polarity', 'max_negative_polarity' as we are interested in only average values.

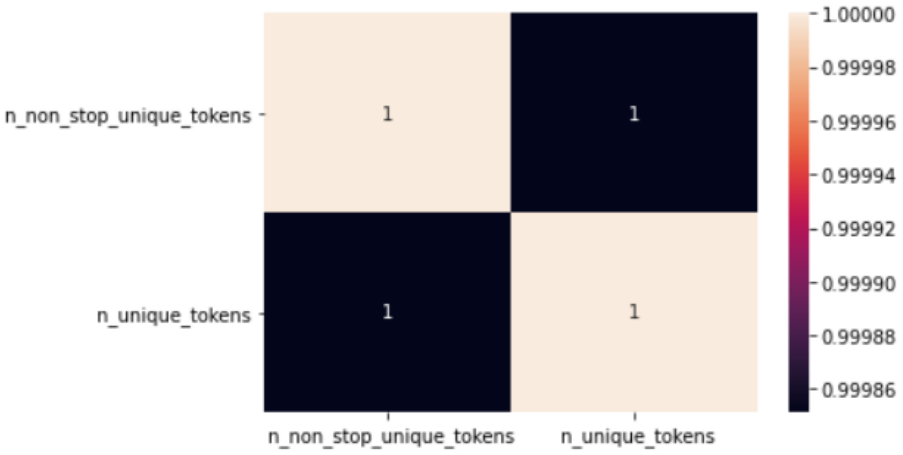
Statistical Analysis

The effort here was to remove missing values and redundant data as well as outliers to make sure they had no influence on the statistical analysis and that the data was ready for both visualization purposes and modeling.

A Correlation plot was created to investigate any dependencies among variables and to identify the most correlated numeric variables in a dataset. The data set has 58 predictive features and some of them might be correlated to each other. We have not included the entire correlation matrix because of its size (58 x 58). The table below shows some pairs of attributes which are highly correlated i.e whose Pearson Coefficient is greater than 0.85. In the following pairs of highly correlated attributes, one in each pair is a potential candidate for removal and this will be handled in the preprocessing section. From the correlation plot, it was noted that:

```
max corr: 0.999851516017834 , min corr:  -0.6507174017206482
 n_non_stop_unique_tokens   n_unique_tokens                0.999852
 n_unique_tokens            n_non_stop_words               0.999572
 n_non_stop_words           n_non_stop_unique_tokens       0.999532
 LDA_02                     data_channel_is_world          0.836618
 rate_negative_words        global_rate_negative_words     0.779556
 LDA_00                     data_channel_is_bus            0.774651
 LDA_04                     data_channel_is_tech           0.749737
 rate_positive_words        global_sentiment_polarity      0.727827
 title_subjectivity         abs_title_sentiment_polarity   0.714528
 global_subjectivity        avg_positive_polarity          0.631749
 rate_positive_words        global_rate_positive_words     0.628626
dtype: float64
```





Confusion matrix

Confusion matrix is a matrix which shows predicted and actual classifications. The size of a confusion matrix is N × N, where N is the number of different classes. [1,7] We can see how many observations are correctly or incorrectly classified. The elements in the diagonal of the confusion matrix indicate correct predictions, while the off-diagonal elements are incorrect predictions. So we can get the accuracy which is the percentage of correct predictions from all predictions. One way to get the better model is choosing the one which has higher accuracy.

III. Model Description

Linear Regression

First we used linear regression to get a quick start. Linear regression represents a least-square fit of the response to the data. It chooses the hypothesis:

$$h_\theta(x) = \sum_{i=0}^{n} \theta_i x_i$$

By minimizing the cost function

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Due to the high variance of the target variable (number of shares), direct application of linear regression was not acceptable, specifically, on testing samples, only 20% prediction values (number of shares) are within 1000 of the actual results.

Logistic Regression

We then use a classification model trying to improve our accuracy further. For logistic regression, the hypothesis is:

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta T x}}$$

and parameters are chosen as to maximize their likelihood

$$\prod_{i=1}^{m} p(y^{(i)}|x^{(i)}; \theta)$$

We classified the data and used stochastic gradient ascent rules to implement it, and we got a similar result as the linear regression model. For multinomial classifications (say k classes), the model uses logarithmic function to estimate the relative probability of each category with reference to the kth category. For example, for k = 3 (i. e. we classify the data into 3 categories, “unpopular", “popular", “very popular"), we comparing the follows:

$$log\left(\frac{P(y^{(i)}=1)}{P(y^{(i)}=3)}\right) = \sum_{k=1}^{n} \beta_{k,1} x_{k-1}^{(i)}$$

$$log\left(\frac{P(y^{(i)}=2)}{P(y^{(i)}=3)}\right) = \sum_{k=1}^{n} \beta_{k,2} x_{k-1}^{(i)}$$

and the prediction result is the class with the largest probability. Both generalization and training error increases with the increasing k, (e.g., for k = 3, logistic regression gives X accuracy). Since we mainly focused on predicting whether a news would be popular or not, we did not go further in multiclass problems.

Support Vector Machine

Support Vector Machines provide us with some major advantages while dealing with classification problems where the data set has high dimensions. Hence, we have decided to try it for our data set. The objective of SVM is to find a hyperplane (could be a line or a 2d plane too) to effectively separate our data while also allowing for some error to ensure a balanced bias and variance. The key idea behind SVM is to use gradient descent to adjust weights and biases used

in the Kernel function which defines the shape of the separating hyperplane. It also works on a loss function known as Hinge Loss. For our problem, SVM gave us an Accuracy of 0.6165 and a Recall of 0.7213. This performed fairly well but only a linear separation was attempted. Using other Kenrels such as the BiSquare kernel which can generate more flexible hyperplanes would be a good future step to take.

$$w.x_i - b \geq 1 \quad for \quad y_i = 1$$
$$w.x_i - b \leq -1 \quad for \quad y_i = -1$$

Overall,

$$y_i(w.x_i - b) \geq 1$$

Hinge Loss Function -

$$l = max(0, 1 - y_i(w.x_i - b))$$

Neural Network

Neural Network being one of the most powerful algorithms currently in use would be a good option to boost accuracy, provided the Neural Network is constructed efficiently. In our case, the Neural Network architecture consisted of 1 input layer with as many nodes as relevant features, i.e. 58 in this case, 2 hidden layers with 36 and 16 nodes respectively and an output layer with a single node because our problem was simply a binary classification. The objective of the neural network is to adjust the weights and biases, similar to most other ML models, using an optimizer and a learning rate. Here we used the Relu activation at each layer except the final layer where we applied the sigmoid binary cross entropy loss function to compute the cost at each iteration and backpropagate to update the weights and biases.

The Neural Network we constructed gave us an accuracy of 0.5674 and a recall of 0.2585. More improvements to the model structure and loss and optimization functions need to be made in order to achieve a better score.

IV. Results

We train several regression and classification models to predict the popularity of some online articles before them are published. We use a number of shares to judge the popularity of the articles. In this project, we implemented 5 different machine learning models. Their accuracy and recall (sensitivity) are listed in Table 1. We went deeper into how parameter affects performance for SVM and Random Forest, since they have more parameters to consider.

As a classification model, logistic regression achieves a decent accuracy, better than most of the models.

For our SVM, a linear separator was used and using a polynomial kernel function or a higher degree kernel function will help boost performance

Using a different set of optimizer and loss functions, and improving on the architecture of the neural network can improve its performance greatly.

Algorithms	Accuracy	Recall
Logistic Regression	0.64	0.67
SVM	0.6165	0.7213
Neural Network	0.5674	0.2585

Table 1. Performance Comparison

V. Future Work

As is seen from the result, no algorithm can reach 70% accuracy given the data set we have, even though they are state-of-the-art. To improve accuracy, there is little room in model selection but much room in feature selection. In the preprocessing round, 59 features were extracted from news articles, and our later work is based on these features. However, the content of news articles hasn't been fully explored. Some features are related to the content, such as LDA topics (feature #39 - #43), which are convenient to use for learning, but reflect only a small portion of information about the content. In the future, we could directly treat all the words in an article as additional features, and then apply machine learning algorithms like Naive Bayes and SVM. In this way, what the article really talks about is taken into account, and this approach should improve the accuracy of prediction if combined with our current work.

References

[1] Douglas G Altman and J Martin Bland. Diagnostic tests. 1: Sensitivity and specificity. *BMJ: British Medical Journal*, 308(6943):1552, 1994.

[2] Tatar, Alexandru, et al. "Predicting the popularity of online articles based on user comments." *Proceedings of the International Conference on Web Intelligence, Mining and Semantics*. ACM, 2011.

[3] Hensinger, Elena, Ilias Flaounas, and Nello Cristianini. "*Modelling and predicting news popularity.*" Pattern Analysis and Applications 16.4 (2013): 623-635.

Dataset link: https: //archive.ics.uci.edu/ml/datasets/online+news+popularity
GitHub Repository: https://github.com/suku111ad/Online-News-Popularity