Online News Popularity

Anindita Mitra,
Anisha Vijayan,
Murtaza Agha,
Yu Ting Sun

Abstract

The web news is frequently distributed throughout the world with the aid of the internet. Many users today regularly read and share news online on social media sites like Facebook and Twitter. The number of readers, likes, or shares is typically a good indicator of how popular a piece of news is. It is particularly beneficial for web news stakeholders if the popularity of the news pieces can frequently be correctly predicted before the release. Therefore, using machine learning techniques to forecast the recognition of online news stories is intriguing and significant. In this paper, using a dataset of 39643 news stories from the website Mashable, we try to identify the most straightforward categorization learning method to determine whether a news story will become popular or not. Our study shows that using different classification algorithms—such as Naive Bayes, Logistic Regression, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), other unsupervised learning techniques (PCA), and Cost-Matrix —give different results when predicting news popularity using tweets on social media platforms based on their shares and hashtags used during different days of the week. The most accurate results obtained by our study come from the SVM model with an accuracy of 67% and AUC of 0.66, which are the highest among all.

1 Introduction

1.1 Motivation

With the expansion of rapid development of technology, the internet has replaced traditional paper information channels such as newspapers and magazines, and it has become more important in people's lives. There has been a growing interest in online news, which allows an easy and fast spread of information around the globe. Thus, predicting the popularity of online news is becoming a recent research trend [1]. This not only keeps society informed of events that are happening but also reveals the type of events that capture the public's interest. These predicted values can greatly benefit authors of articles, content providers, and also advertisers [2]. Utilizing Machine Learning algorithms helps to effectively identify what kind of articles will be more popular, which can further construct an article that is able to encourage people to spread the information they want to spread.

1.2 Problem Statement

The data comes from the website mashable.com from the beginning of 2015, and the dataset can be found at the UCI repository - Online News popularity Dataset. It summarizes a heterogeneous set of features about articles published by Mashable in a period of two years, and the goal is to predict if the article is among the most popular ones based on sharing in social networks [3].

In this paper, we utilized the online news popularity dataset to analyze and explore data to build Machine Learning predictive models. By exploratory data analysis and feature engineering in the dataset, we found the important features and transformed the number of shares into a classification problem of popularity or unpopularity.

The contribution of this paper is to find the best predictive model in the dataset by engineering imbalanced data and utilizing Machine Learning algorithms to achieve the goal. The paper will introduce methods such as Naive Bayes, Logistic Regression, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and other unsupervised learning techniques to experiment with a better model. Then, the results are evaluated by Accuracy, F-score, and AUC value to conclude the assessment of the models and further solve the problem.

2 Related Work

2.1 Background Research

Originally, the articles in the dataset were published by Mashable (mashable.com), and they have the rights to reproduce their contents. Hence, this dataset does not share the original contents but some statistics associated with it. In the dataset, the original contents are publicly accessed and retrieved using the provided URLs [3]. In the paper “A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News”, the estimated relative performance values were estimated by the authors using a Random Forest classifier and rolling windows as assessment methods [1][3]. They proposed an Intelligent Decision Support System (IDSS) that could analyze articles prior to their publication. By using a set of extracted and optimized features, the IDSS would search for an enhancement of the predicted popularity probability [1].

2.2 Relevant Projects and Work

A study developed by A. Tatar et al., showed that predicting the popularity of web content is useful in many areas such as network dimensioning, online marketing, or real-world outcome prediction like economic trends. In this survey, the web content reviewed the popularity prediction in current findings and further experimented with different popularity prediction models. The study also included showing good predictive features and finding out the factors that influence web content popularity. From presenting the different prediction methods, and reporting the performance, to suggesting several applications, we can benefit from these findings to build richer models and get a better understanding of popularity predictions [4].

Furthermore, J. Maguire and S. Michelson performed the prediction of a post's popularity by using basic Machine Learning techniques such as Naive Bayes, Perceptron, Least Squares, and SVM in the paper. After analyzing trends in the data and refining the learning processes, the performance of the model prediction was measured in terms of accuracy which was achieved at 85%. The results of the study demonstrated that basic Machine Learning models could be used to predict the popularity of social media posts [5]. Therefore, it leads us to build up models based on Machine Learning algorithms to predict online news popularity.

Implementing different approaches, H. Ren and Q. Yang, compared ten Machine Learning methods on the dataset ranging from various regressions to the SVM approach. As a classification model, Logistic Regression achieved a decent accuracy, better than most of the models in the paper. Thus, we selected the Logistic Regression to be one of the main models for predicting online news popularity. Also, feature selection methods were used to improve performance and reduce features in the study, and we will further apply those methods before building models in the exploratory data analysis part [6].

Lastly, M. Sokolova et al. discussed different evaluation measures to assess different characteristics of Machine Learning algorithms. It mentioned that measures of the quality of classification are built from a confusion matrix that records correctly and incorrectly recognized examples for each class. In the study, the accuracy is able to assess the overall effectiveness of the algorithm, and F-score benefits algorithms with higher sensitivity and challenges algorithms with higher specificity. For our evaluation methods selection, we also utilized ROC and AUC values to present a relation between the sensitivity and the specificity of the algorithm [7]. Based on our dataset, the values of precision and recall are both important to us. Therefore, we still intended to use standard accuracy, confusion matrix, and AUC value first, and further got the F-score which is the combination of accuracy and AUC value for the performance evaluation.

3 Model Design

3.1 Baseline Model - Naive Bayes

3.1.1 Naive Bayes Model

Naive Bayes is a probabilistic Machine Learning algorithm based on the Bayes Theorem, used in a wide variety of classification tasks. The classifier calculates the probabilities for every feature X. Thereafter it opts for the highest probability value of instance as a resultant class [8]. To calculate the probability, the classifier can apply Bayes Theorem as follows:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

P(A): Probability of A occurring
P(B): Probability of B occurring
P(A|B): Probability of A occurring given evidence B has already occurred
P(B|A): Probability of B occurring given evidence A has already occurred

Based on Bayes Theorem, the relationship between feature X and class can be calculated as below. Considering P(X1, X2, …, Xn) is constantly giving the input, the classification rule is given by the class with the maximum probability value.

$$P(C_i|\, x_1, x_2 \dots, x_n) = \frac{P(x_1, x_2 \dots, x_n|C_i).P(C_i)}{P(x_1, x_2 \dots, x_n)} \; for\; 1 < i < k$$

3.1.2 Naive Bayes Algorithm [8]

Input: Training data and Test data

---

1. Build the table of frequency using feature X vector (X1, X2, …, Xn) against every class Ci, where i = 1 to k
2. Calculate the likelihood for each class
3. Compute the conditional probability of all Test data class:

$$P(C_i|\, x_1, x_2 \dots, x_n) = \frac{P(x_1, x_2 \dots, x_n|C_i).P(C_i)}{P(x_1, x_2 \dots, x_n)} \; for\; 1 < i < k$$

4. Calculate the maximum value of P(Ci|X1, X2, …, Xn)

Output: Class with maximum probability value

3.2 Logistic Regression

3.2.1 Logistic Regression Model

Logistic Regression is a useful analytical technique for classification problems, in which predictions are discrete values after applying a transformation function. It is suited for binary classification, and the outcome will be two values such as true/false, yes/no, and so on: data sets where y = 0 or 1, where 1 denotes the default class.

The sigmoid function is the link function that transforms our unbounded continuous prediction and maps [−∞, ∞] to [0, 1] by the below formula. Logistic Regression uses the hypothesis $h_\theta(x) = g(\theta^T x)$ to predict a continuous value (confidence) in [0, 1].

$$g(z) = \frac{1}{1+\exp(-z)}$$

$$p(y = 1|x; \theta) = h_\theta(x) = g(\theta^T x)$$
$$p(y = 0|x; \theta) = 1 - h_\theta(x) = 1 - g(\theta^T x)$$

Basically, it states that a solution to a Logistic Regression problem is the set of parameters that maximizes the likelihood which is defined via the maximum likelihood estimator. Instead of maximizing the likelihood, it is easier to maximize the natural log of the equation, which is called the log-likelihood function. By taking the negative of the log-likelihood function, it can be minimized using gradient descent for the loss function [9].

$$-\frac{\partial \ell(\theta; x, y)}{\partial \theta_j} = -\big(y - h_\theta(x)\big)x_j = (h_\theta(x) - y)x_j$$

3.2.2 Logistic Regression Algorithm [10]

Input: Training data and Test data

---

1. Initialize all parameters (z)
2. Calculate the dependent variable (hypothesis)
3. Calculate the loss function and the gradient for the loss function
4. Update all parameters and repeat the calculation steps
5. Assign the class label based on the probability of the hypothesis

Output: Class of Test dataset

3.3 K-Nearest Neighbor (KNN)

3.3.1 KNN Model

K-Nearest Neighbor (KNN) is a non-parametric supervised learning method, which means it does not make any assumption on underlying data. The KNN algorithm is also called a lazy learner algorithm because it does not learn from the training set immediately. Instead, it stores the data during the training time and does not perform any calculations. A query performs an action in the dataset and builds the model when it gets new data, then it classifies the data into a class that is much similar to the new data.

It can be used for regression problems as well as classification problems, but it is mostly used as a classification algorithm, working off the assumption that similar points can be found near one another. For

classification problems, a class label is assigned on the basis of a majority vote. That means the class with the majority vote will become the class of the data point in question.

To determine which data points are closest to a given query point, the distance between the query point and the other data points will need to be calculated. The distance metrics will help to form decision boundaries, which district query points into different areas. The most commonly used distance measure is Euclidean distance, which is the length of a line segment between the two points [11]. The formula is shown below:

$$distance = \sqrt{\sum_{i=0}^{n}(x_i - y_i)^2}$$

3.3.2 KNN Algorithm [12]

Input: Training set, Test object, and Class label set

---

1. Initialize the estimator
2. Calculate the distance in the training set between each training data and test object
3. Select the subset from the training set, and the subset contains the k training samples which are the k nearest neighbors of the test object
4. Compute the class of the test object:

$$c_x = \arg\max_{c \in C} \sum_{y \in N} I\left(c = class\left(y\right)\right)$$

Output: Class of Test object

3.4 Support Vector Machine (SVM)

3.4.1 SVM Model

Support vector machines (SVMs) are a set of related supervised learning methods used for classification and regression. They belong to a family of generalized linear classifiers. The idea of the SVM classifier is to come up with a hyperplane in an N-dimensional space that divides the data points into different classes. This hyperplane is chosen based on the margins as the hyperplane providing the maximum margin between the two classes is considered. These margins are further dependent on the data points that are near the hyperplane called the support vectors.

When the data is linearly separable, it is a hard-margin classifier problem. But in the real world, it is not likely to get an exactly separate line dividing the data within the space and we might have a curved decision boundary. To handle this problem, slack variables are introduced which give rise to the soft margin classifiers. This allows the point to be a small distance (slack) on the wrong side of the hyperplane without violating the constraints. There is a Lagrangian variable also present to penalize the large slack values.

The hard margin classifier is a quadratic optimization problem with linear inequality as shown below:[14]

$$\begin{aligned} \underset{\mathbf{w},b}{\text{minimize}} \quad & \tfrac{1}{2}||\mathbf{w}||^2 \\ \text{subject to:} \quad & y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) \geq 1 \quad i = 1, \ldots, n\,. \end{aligned}$$

whereas, a soft margin is computed as shown below:

$$\begin{aligned} \underset{\mathbf{w},b}{\text{minimize}} \quad & \tfrac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to:} \quad & y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0. \end{aligned}$$

To map the input data into a high dimensional space non-linearly, kernels are used. The Kernel trick allows the SVMs to form nonlinear boundaries. The idea of kernel functions is to enable operations to be performed in the input space rather than the potentially high dimensional feature space. The kernel function plays a critical role in SVM and its performance [13].

SVMs can also be applied to regression problems with the help of another loss function that involves a distance measure. The regression can also be linear or non-linear. The kernel function is again used in the non-linear regression approach to address the curse of dimensionality.

3.4.2 SVM Algorithm[15]

Input: Training set, Test object, and Class label set

---

1. Initialize the learning rate, lambda, number of iterations, weight vectors w, and bias b.
2. For every iteration:
   Compute the value of y*(dot product of x data and w - b)
   If value >=1:
   Decrement w by learning rate*(2* lambda * w)
   Else:
   Decrement w by learning rate*(2* lambda*w - (dot product of x & y))
   Decrement bias b by-product of learning rate and y
3. Get the best values of weight vectors w and b for making predictions
4. Prediction is given as the sign of dot product of test data x and its weight vector w subtracted by the bias term

Output: Class of Test object

3.5 Unsupervised Learning Model - PCA

3.5.1 PCA Model

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.

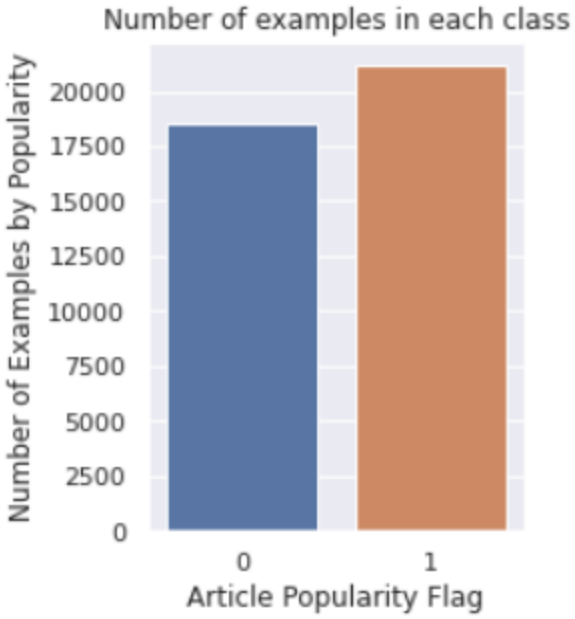
3.5.2 PCA Algorithm

- The covariance matrix comes first. Estimates of the relationships between each variable in Z and each other are contained in the matrix ZTZ. It is very useful to know how one variable is related to another.
- Eigenvalues and eigenvectors are crucial, too. Directions are portrayed by eigenvectors. A unique eigenvector can thus be viewed as a specific "direction" in your scatterplot of data. Eigenvalues stand for magnitude or significance. More significant directions are correlated with larger eigenvalues.
- Finally, we assume that the ability to describe the behavior of the dependent variable is correlated with the amount of variability in a specific direction. Little variability typically denotes noise, while a lot of fluctuation typically denotes signal. So, theoretically, the greater the variability in a given direction, the more likely it is that we are looking for anything significant.

4 Experimental Results

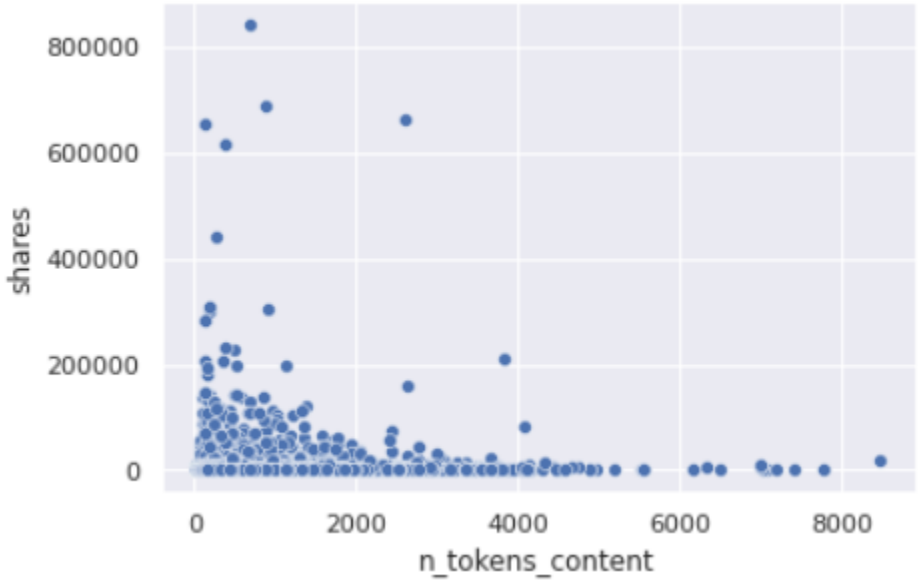
4.1 Exploratory Data Analysis

In the dataset collected from the UCI repository, we have 39,644 records and 61 features. We first checked for missing values in the dataset and found no missing ones. The column names had spaces so we cleared that up. Then we checked for the statistics of the target variable shares, and found the 50% quantile value as 1400 and used that to convert this problem into a binary classification one by splitting the dataset into two classes (0,1) for a new target variable is_popular. If the shares were greater than 1400, then it belonged to class 1 otherwise 0. We checked for the distribution of the class labels in the whole dataset and found the below findings.

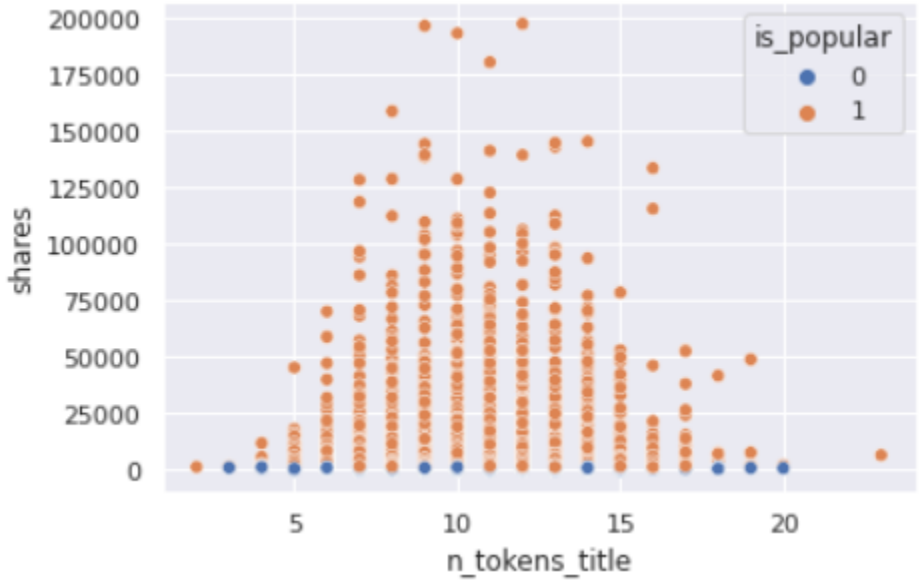


The class distribution seems to be balanced and this type of distribution imposes fewer challenges when building Machine Learning models.

We checked for the statistics of the different variables like n_tokens_content, n_tokens_title, n_non_stop_words, n_non_stop_unique_tokens, and found that n_non_stop_words and n_non_stop_unique_tokens didn't make any impact on the target variable, so we removed them from the dataset. In the case of n_tokens_content, we found that good articles generally had a value between 100 and 2000 words which is shown in the below scatterplot. Also, we found that there were 1181 examples with no words in the content, so we removed those as well.

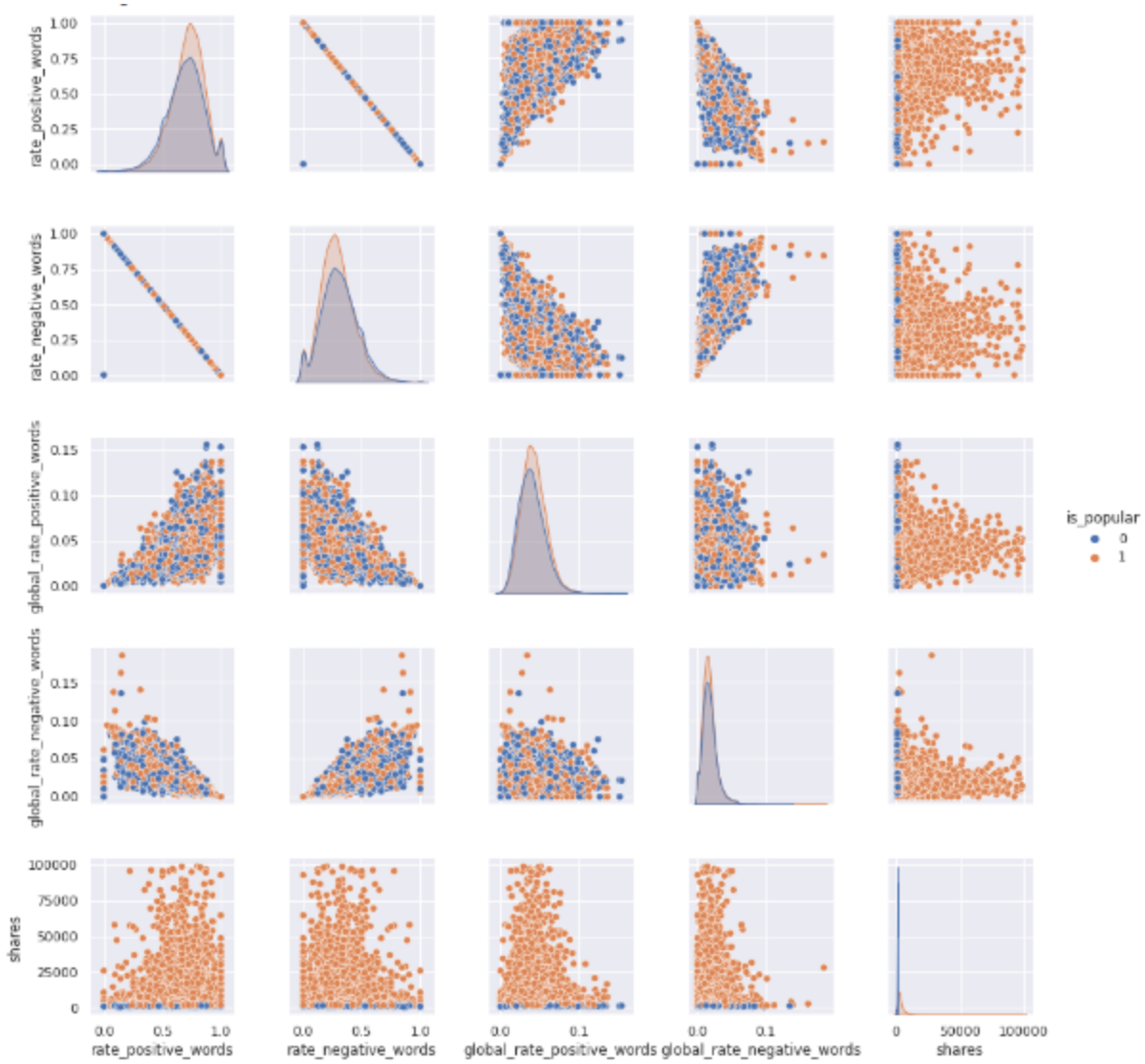


In the case of n_tokens_title, we found that good articles generally had a title word count between 6 and 17.

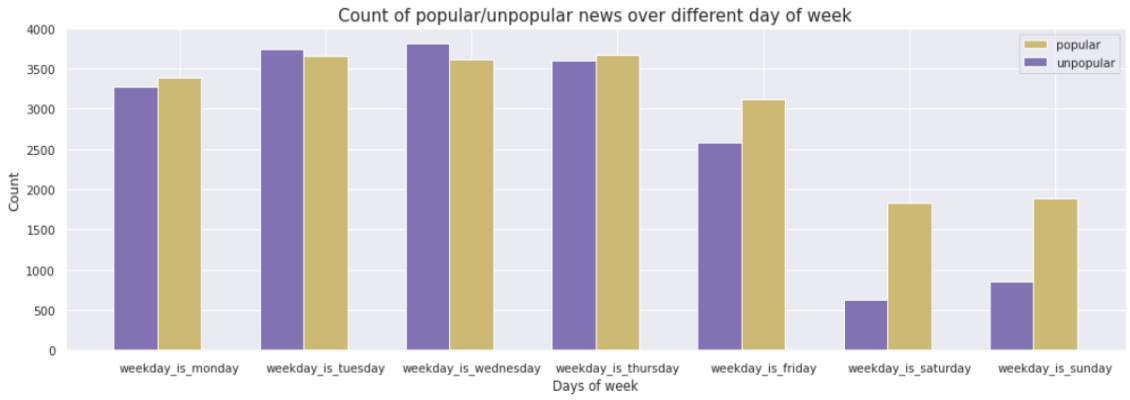


4.2 Relationships between variables

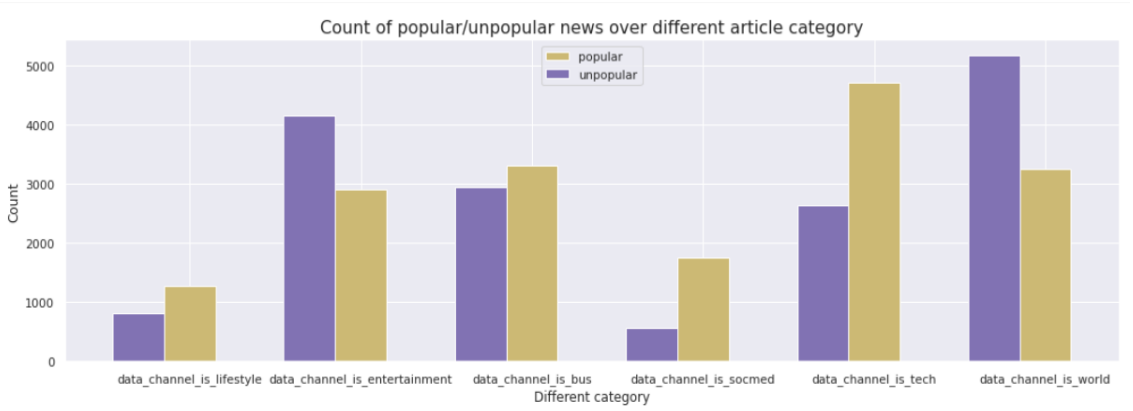
We tried to find the relationship between rate_positive_words, rate_negative_words, global_rate_positive_words, global_rate_negative_words, and target variable shares by plotting a heatmap between them (shown below). We found that there was a linear relationship between rate_positive_words and rate_negative_words but no special relationship with the target variable. Also, global_rate_positive_words and global_rate_negative_words showed a slight relationship with shares.



We also plotted a graph between is_popular and the different days of the week and found that popular articles were posted usually on the weekdays rather than the weekends.



The below graph shows the count of popular and unpopular articles based on the different data channels. It is clear that social media and technology are more popular than others.

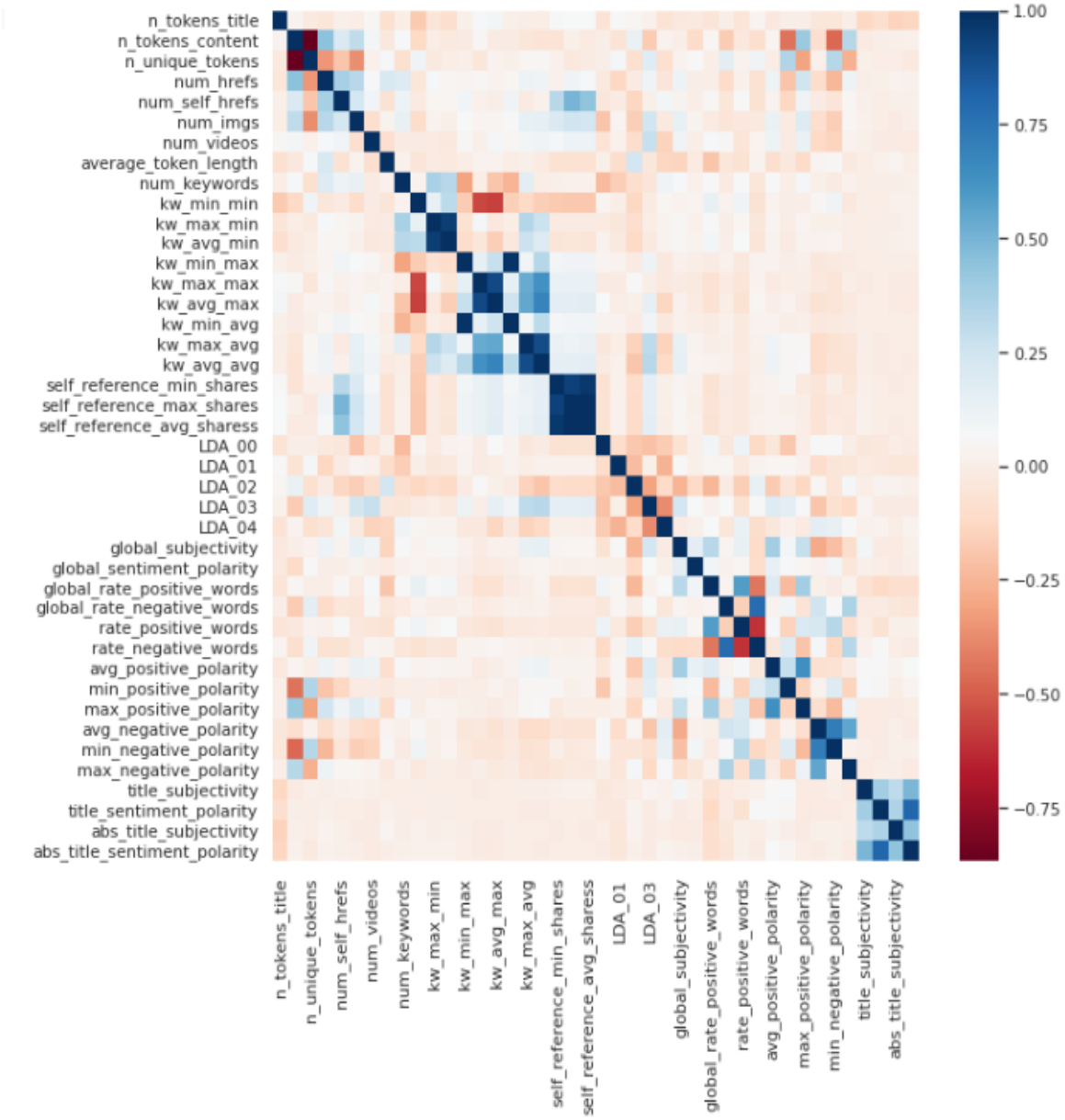


4.3 Feature Engineering

We have dropped URL and time delta features for the modeling part as these features are non-predictor features. From our previous analysis, we found that n_non_stop_unique_tokens and n_non_stop_words didn't really make any contribution so we removed them.

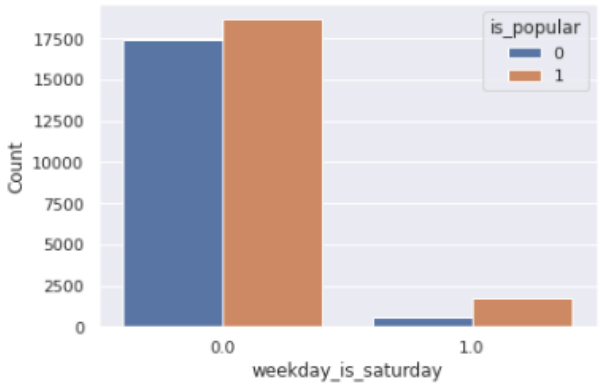
4.3.1 Correlation Matrix

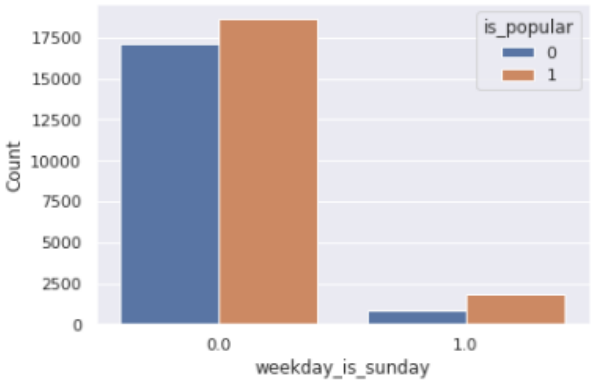
Based on the correlation matrix we dropped self_reference_avg_sharess, avg_negative_polarity, avg_positive_polarity, and kw_avg_min.

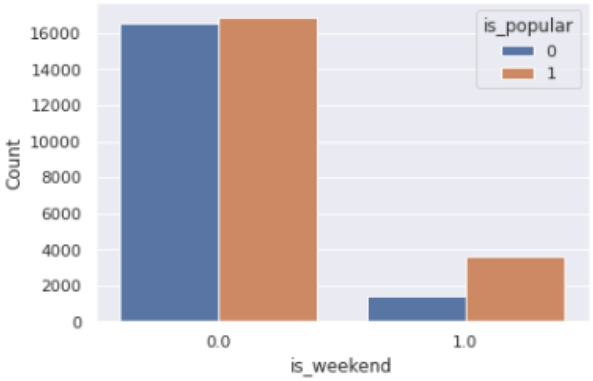


4.3.2 Chi-square Test

Based on the cross plots and chi-square tests, we have dropped unimportant features such as weekday_is_saturday and weekday_is_sunday as these features are already covered in the is_weekend feature.








```
observed data:
 is_popular      0      1
is_weekend
0.0         16571  16865
1.0          1428   3598
expected data:
 [[15646.99090011 17789.00909989]
 [ 2352.00909989  2673.99090011]]
chi-squared value: 784.8625 for 1 dof; p-value = 0.0000
```

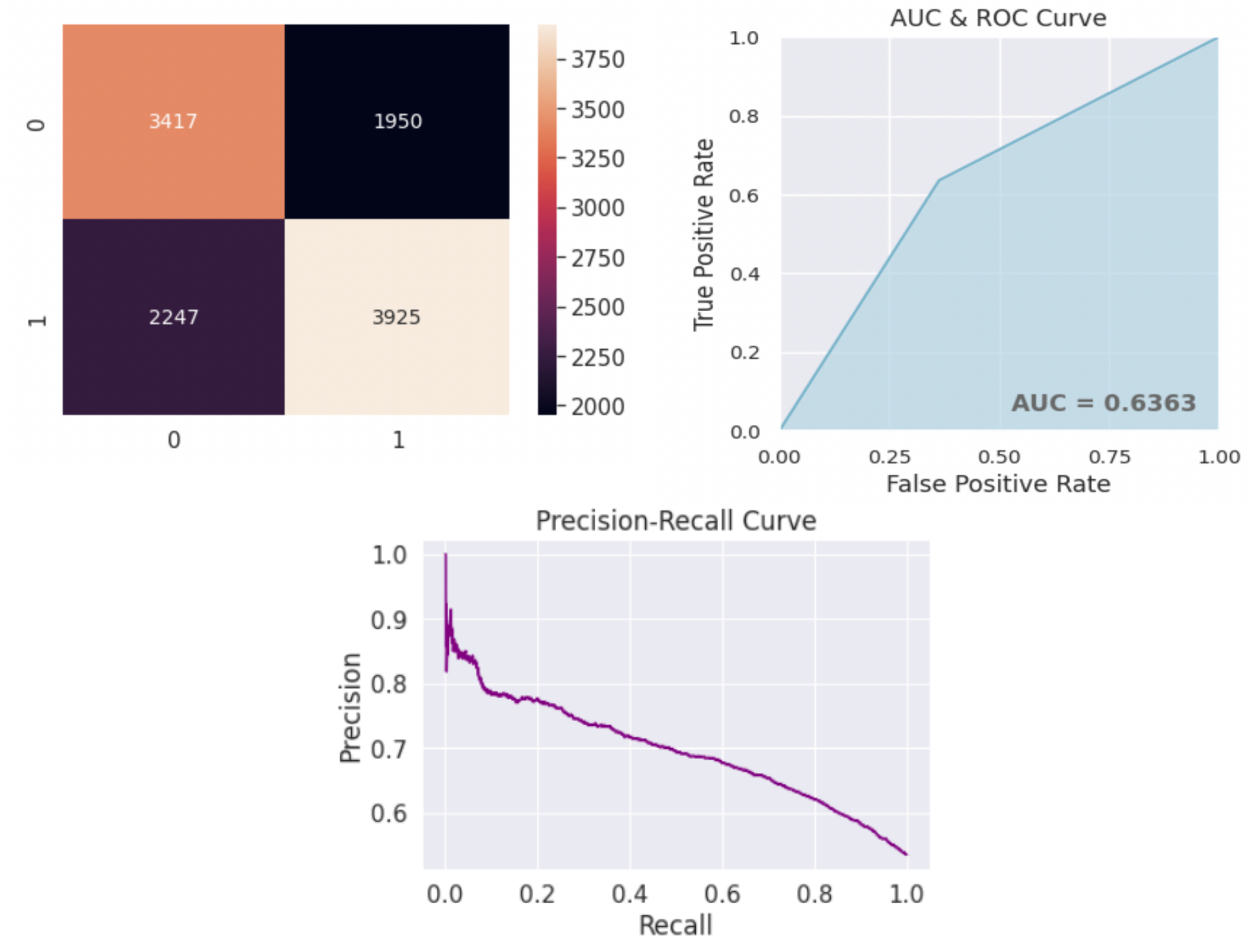
We also found that a lot of independent variables didn't have a normal distribution and showed peakedness or skewness. To handle this we did a log transformation on the numerical features.

4.4 Models and Evaluation

4.4.1 Baseline Model - Naive Bayes

In this project, Naive Bayes is set to be the baseline model. With the balanced data in the target variable, the model should be evaluated by the accuracy, AUC value, and F-score for the results. Showing in the below graph, the AUC value is around 0.64 based on the Naive Bayes model. Also, there are listed accuracy of around 0.64 and an F1-score of around 0.65.

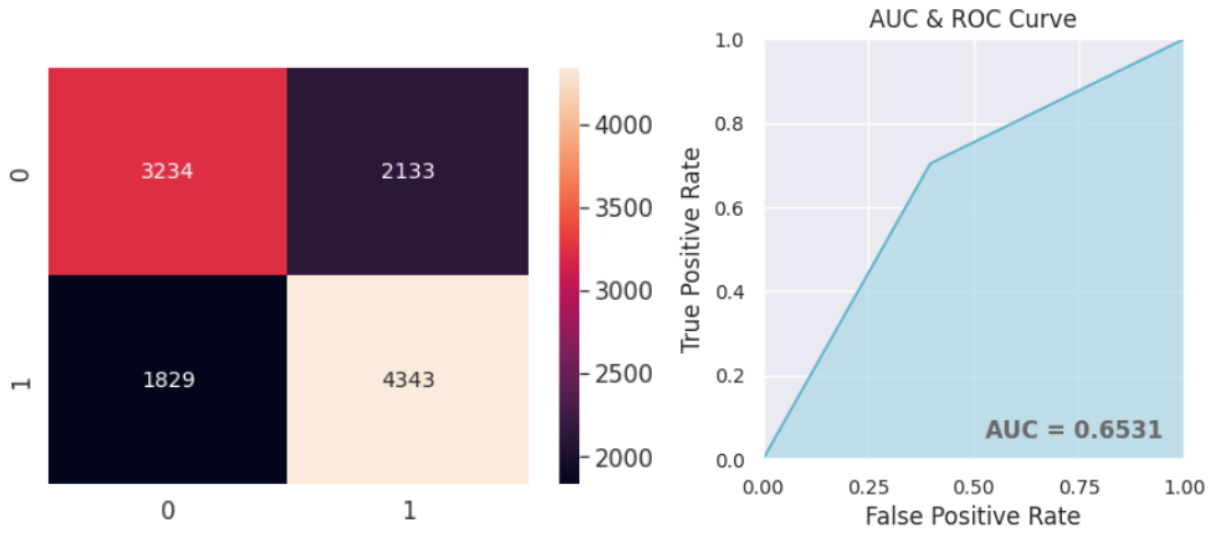
- Accuracy = 0.6363
- AUC = 0.6363
- F-score = 0.6516

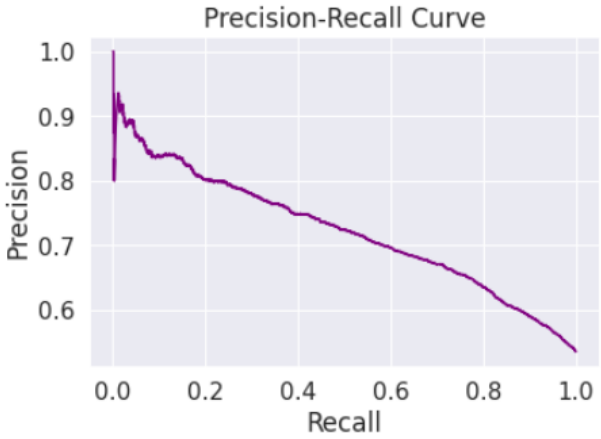


4.4.2 Logistic Regression

As a classification model, Logistic Regression achieves better accuracy than the baseline model. The AUC value is around 0.65, and F-score is around 0.68, which means Logistic Regression gives a fairly good result compared to the Naive Bayes model.

- Accuracy = 0.6554
- AUC = 0.6522
- F-score = 0.6845

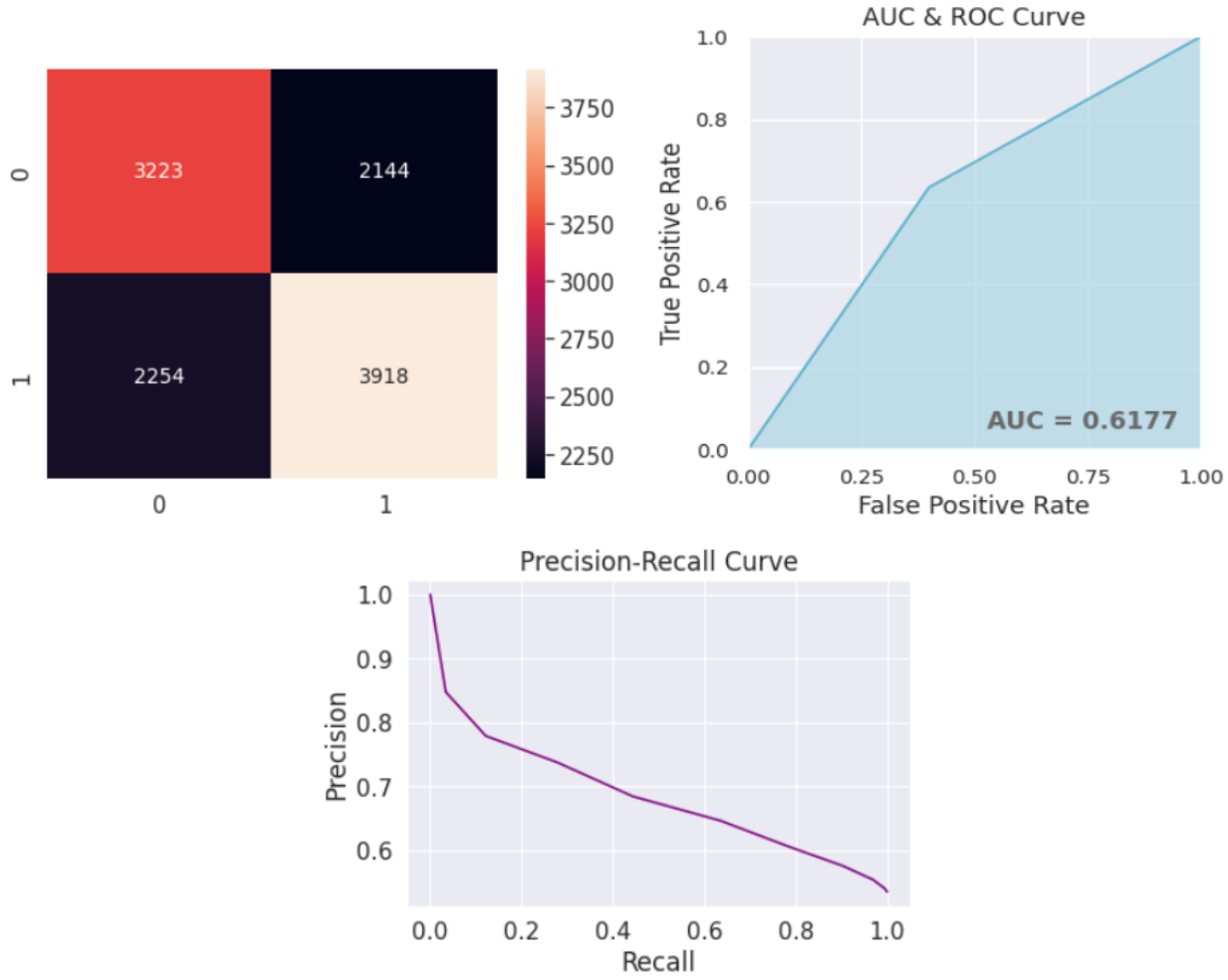




4.4.3 KNN

Based on the KNN model, we first found the best k value by calculating the distances, and the parameter comes to 9 with a higher score of around 0.62. With the k value = 9, the prediction results are not better than the baseline model. We got an F-1 score of around 0.64, and an AUC value of around 0.62.

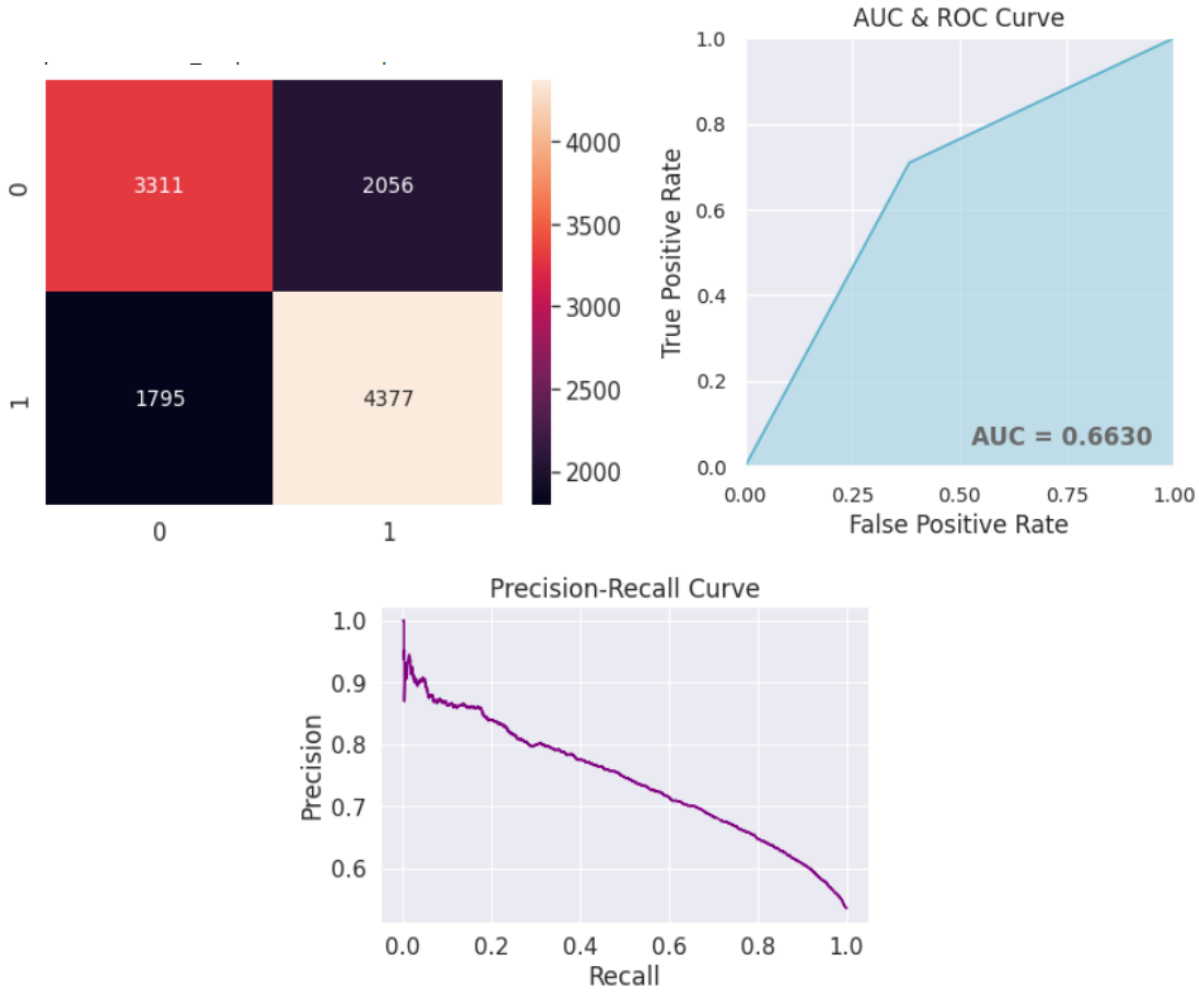
- Accuracy = 0.6188
- AUC = 0.6176
- F1-score = 0.6405



4.4.4 SVM

For the SVM model, we tried the radial basis function, rbf kernel as well as polynomial kernel methods for the soft margin SVM classifier. We experimented with multiple values of the parameter C and got the least test error with the value 6.4. So we trained the model with that value and got an accuracy of 66% for the SVM model with the rbf kernel and 53% for the SVM model with the poly kernel. The SVM classifier with RBF kernel gives the highest accuracy compared to all the other models.

- Accuracy = 0.6662
- AUC = 0.6630
- F1-score = 0.6944



SVM has also classified the most shared or popular as well as unpopular article (shown below) correctly.

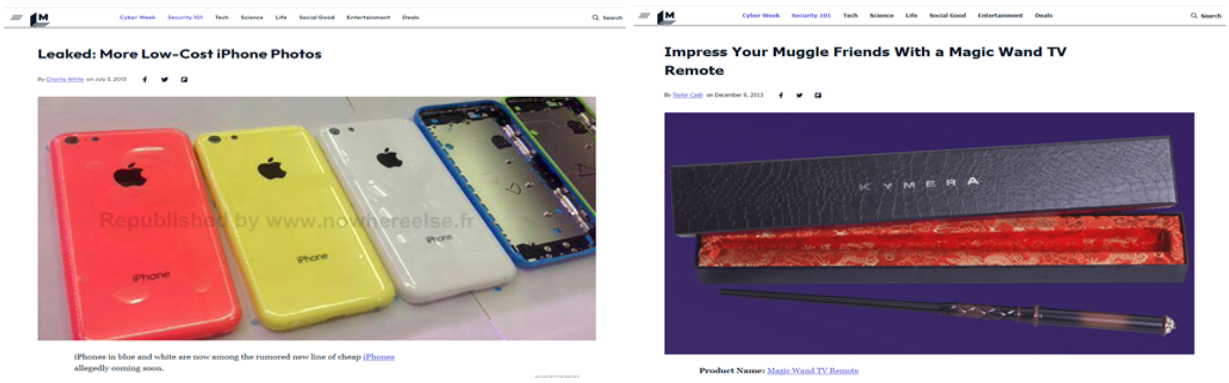


Fig: Left: Most popular or most shared article that was posted by Mashable. Right: Unpopular or least shared article that was posted by Mashable

4.4.5 PCA

After performing Exploratory Data Analysis, we were left with 50 features in our dataset.
We considered two cases for our dataset:
Case 1: when all components are kept.
Case 2: To find desired components to explain 85% of the variance.

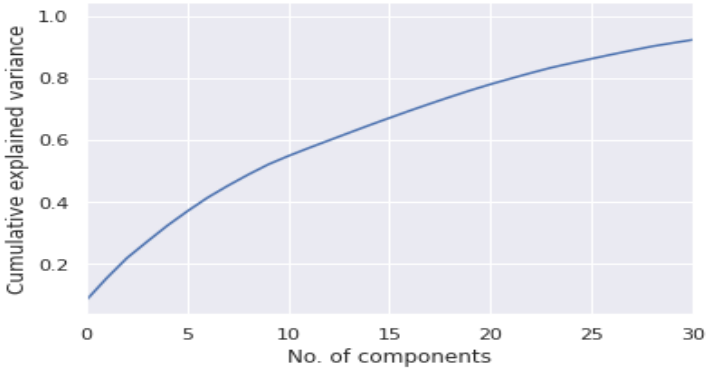


Fig. - Case 1: when all components are kept

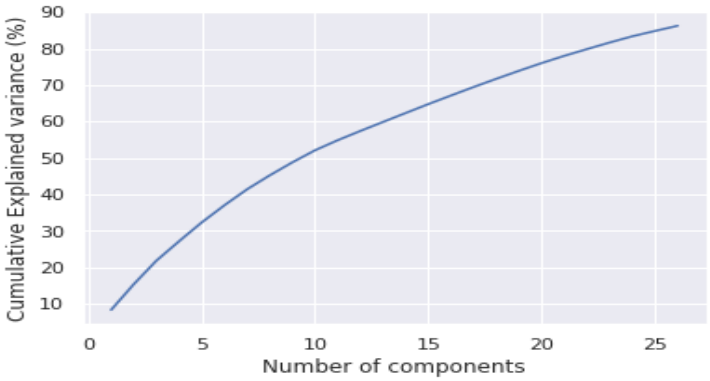


Fig. Case 2: number of components for 85% explained variance

Using case 1, the no. of components required to explain 95% of the variance is 50, which is still high. In case 2, the number of components turned out to be 26 to explain 85% of the variance. We trained the logistic regression model on the dimensionally reduced dataset containing 26 components, but accuracy dipped to 64%.
Since there are still a lot of components needed to explain the variation in this dataset, using PCA would not be appropriate. Our model's interpretability would be lost if we used PCA. Consequently, we did not use PCA in our dataset.

4.4.6 Cost Matrix

A cost matrix (error matrix) is useful when specific classification errors are more severe than others. The trade-off of avoiding 'expensive' classification errors is an increased number of 'cheap' classification

errors. Thus, the number of errors increases while the cost of the errors decreases in comparison with the same classification without a cost matrix [16].
It influences the decision-making of a model. It helps to minimize costly misclassifications and maximize beneficial accurate classifications.
Since we are not getting significantly high accuracy for any of the models, we tried to set the misclassification rates for each case. The cost matrix is similar to the confusion matrix except for the fact that we are calculating the cost of wrong prediction or right prediction.

- Correctly classifying popular articles will help Mashable to increase its revenue by setting the appropriate price at which the advertisements are sold to other social media platforms. This is the True Positive case.
- By correctly classifying unpopular articles, Mashable can reduce the number of such advertisements altogether and set lower costs for them. This is the True negative case.
- When a popular article is misclassified as an unpopular one, then it could incur a loss as the ads will be sold at a lower cost. This is the False negative case.
- When an unpopular article is misclassified as a popular one, then it would not be sold much and might even lose out on potential future ads. This is the False positive case.

The False negative case is costlier than the false positive case as selling popular articles at a lower cost would result in a higher loss than selling unpopular articles at a higher price. Therefore, we have set the below costs for each scenario in terms of the cost matrix.

Cost Matrix	0	1
0	$100	($50)
1	($200)	$300

Naïve Bayes	0	1
0	$341,700	($97,500)
1	($449,400)	$1,177,500

Logistic Regression	0	1
0	$323,400	($106,650)
1	($365,800)	$1,293,900

KNN	0	1
0	$322,300	($107,200)
1	($450,800)	$1,175,400

SVM	0	1
0	$331,100	($102,800)
1	($359,000)	$1,313,100

Fig. Costs for different models based on the confusion matrix(test data) and the const matrix

Model	Profitability
Naïve Bayes	$972,300
Logistic Regression	$1,144,850
KNN	$939,700
SVM	**$1,182,400**

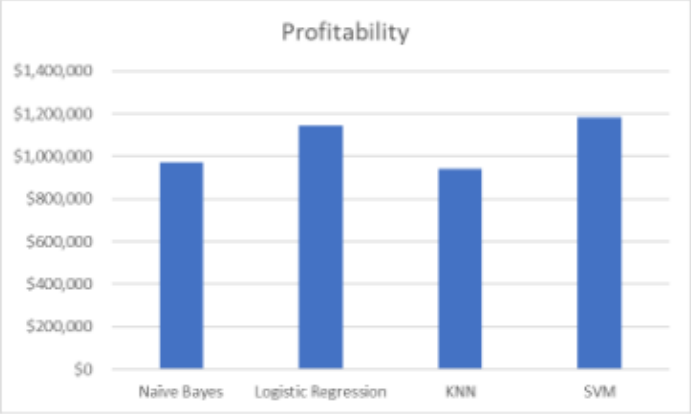


Fig. (Left) Total Profitability by each model. (Right) Graph of Profitability for each model

As we can see even with lower accuracy (66%) SVM provides profitability of $1,182,400 with regards to the costs we had set. This indicates that accuracy is not the only measure when evaluating a model's performance.

4.4.7 Evaluation Results

We have trained the Naive Bayes (Baseline), Logistic regression, KNN, SVM, and PCA (unsupervised) models on the data. Below is the list of their performances based on accuracy, AUC value, and F1-score in different Machine Learning algorithms in the below table:

Algorithms	Accuracy	AUC	F-score
Naive Bayes	0.6363	0.6363	0.6516
Logistic Regression	0.6554	0.6522	0.6845
KNN	0.6188	0.6176	0.6405
SVM	0.6662	0.6630	0.6947

5 Conclusion

In conclusion, the paper discusses numerous algorithms that include four supervised model classifiers Naïve Bayes (Baseline), Logistic Regression, KNN, and SVM, one unsupervised model classifier PCA, and one Cost Matrix used in the method of popularity prediction of news articles. Various news channels have been taken into consideration. We have studied how their features' values are calculated, categorized into categories, and optimized them. The classification models are compared for parameters like accuracy, precision, recall, F1 score, and AUC. After comparing model outcomes, SVM proves to be the highest accurate among all models giving us an accuracy rate of 67%. The training and test of SVM are significantly accelerated and it facilitates SVM in attaining the finest overall performance in terms of accuracy, AUC (66%), and F1-score (69%) along with significant profitability in terms of the cost matrix.

There is scope to improve the classification accuracy and optimization results. Firstly, other variables like the number of news shares, number of likes, number of comments or tweets, number of views, etc can be used as target variables. Secondly, new features like medium can be created because the accuracy results and score depend on the medium through which the article is shared. For example, if the article is shared by a celebrity on Instagram or Twitter, the article would gain more popularity. Thirdly, as far as models are concerned, regression or Ensemble models can be used, the SVM model can be tried with different kernels, or Naive-Bayes and SVM can be combined based on some characteristics of examples. All such measures can improve accuracy and optimization results. However, it is believed that accuracy alone is not always the best measure of model evaluation and other options can be used to determine the overall efficiency of the model.

References

[1] K. Fernandes, P. Vinagre, and P. Cortez. A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News.
https://core.ac.uk/download/pdf/55638607.pdf

[2] M. Tsai and Y. Wu. Predicting online news popularity based on machine learning.
https://doi.org/10.1016/j.compeleceng.2022.108198

[3] K. Fernandes, P. Vinagre, P. Sernadela, and P. Cortez. Online News Popularity Data Set.
https://archive.ics.uci.edu/ml/datasets/Online+News+Popularity

[4] A. Tatar, M. Amorim, S. Fdida, and P. Antoniadis. A survey on predicting the popularity of web content.
https://doi.org/10.1186/s13174-014-0008-y

[5] J. Maguire and S. Michelson. Predicting the Popularity of Social News Posts.
https://cs229.stanford.edu/proj2012/MaguireMichelson-PredictingThePopularityOfSocialNewsPosts.pdf

[6] H. Ren and Q. Yang. Predicting and Evaluating the Popularity of Online News.
https://cs229.stanford.edu/proj2015/328_report.pdf

[7] M. Sokolova, N. Japkowicz, and S. Szpakowicz. Beyond Accuracy, F-score and ROC: a Family of Discriminant Measures for Performance Evaluation.
https://www.aaai.org/Papers/Workshops/2006/WS-06-06/WS06-06-006.pdf

[8] V. Kumar. Evaluation of computationally intelligent techniques for breast cancer diagnosis.
https://doi.org/10.1007/s00521-020-05204-y

[9] N. Zumel. The Simpler Derivation of Logistic Regression.
https://win-vector.com/2011/09/14/the-simpler-derivation-of-logistic-regression/

[10] S. Gupta. Scikit-Learn: A Complete Guide With a Logistic Regression Example.
https://www.springboard.com/blog/data-science/scikit-learn/

[11] IBM. What is the k-nearest neighbors algorithm?
https://www.ibm.com/topics/knn

[12] Cao, Q., La, L., Liu, H., & Han, S. Mixed Weighted KNN for Imbalanced Datasets. International journal of performability engineering, 14, 1391.
http://www.ijpe-online.com/EN/Y2018/V14/I7/1391

[13] V. Jakkula. Tutorial on Support Vector Machine (SVM)
https://course.ccs.neu.edu/cs5100f11/resources/jakkula.pdf

[14] A. Ben-Hur, J.Weston. A User's Guide to Support Vector Machines.
https://pyml.sourceforge.net/doc/howto.pdf

[15] P. Loeber. (2019) SVM (Support Vector Machine) in Python - ML From Scratch 07.
https://www.python-engineer.com/courses/mlfromscratch/07_svm/

[16] IBM. Data Warehousing and analytics. 03, 2021.
https://www.ibm.com/docs/en/db2/9.7?topic=classification-cost-matrix