Online News Popularity

Abstract

Almost all the news channels nowadays use machine learning models to figure out if a particular new article is going to be a hit or a miss. Linear regression is used for the prediction on number of shares and Logistic regression, SVM, Naive Bayes and Neural Networks are few of the models that could be used for classifying the articles popularity. We intend to find the best model that could predict the number of reshares a particular article is going to get and if it would be popular or not from the features extracted from different data mining techniques and feature engineering. The dataset was taken from UCI repository and for the regression model, RMSE and MAE are used for performance evaluation. For the classification models on popularity prediction, accuracy is used for performance evaluation. Naive Bayes have given the best accuracy of 90% amongst all models for classification. The can be used by any social media platform before publication in predicting the popularity.

Introduction

In the modern age, the age of science and technology, the traditional newspapers and magazines have been taken over by the internet thereby making it crucial for the channels to predict the popularity of an article on the web. Since a major portion of a company's budget is allocated to its marketing, an efficient way to get the popularity is by building a model that can reveal the best available options to the user in understanding the features that affect the output and predict the score of a given article/blog. The principle of this project is to use data mining techniques to analyze the data and build multiple machine learning models to predict the popularity of a news article

Data Source

The data was taken from the UCI Machine learning repository.
https://archive.ics.uci.edu/ml/datasets/online+news+popularity
The data consists of statistics of articles that were published by Mashable.

Problem Definition

Number of views/clicks, number of likes/dislikes, number of re-shares are few of the many ways to estimate the popularity of a particular paper. In this project we will be considering shares to estimate the popularity of an online new article.
The assumptions we will be making as part of this project are,
● There is no relation between popularity of the article with the article itself and the current events that are happening in the world.
● The quality of the article and the affinity between the reader and the article is ignored.

The following items will be inspected and addressed as part of this project:

● What are the contributing factors/attributes that make an article popular?
● What are the predictors that affect the shares of an article?
● What are the best models to predict and classify a particular article as popular?

Data Description and Cleaning

The dataset is from the UCI Machine Learning Repository. This data summarizes a heterogeneous set of features about articles published by Mashable. There are 39797 records and 61 attributes. In the 61 attributes, there are 58 predictive, 2 non-predictive (URL and time), and 1 target variable - number of shares. The 58 predictive variables constitute both numerical and categorical fields - length of the article, length of the title, count of unique words, stop words, average token length, type of channel the article was published in, day of the week the article was published on.

Since the articles had the textual data, the sentiment analysis was already performed on the data. The sentiment polarity scores for the articles and titles along with positive and negative word count data was generated and is already part of the predictors.

Here are sample records from the online news popularity data set.

	url	timedelta	n_tokens_title	n_tokens_content	n_unique_tokens	n_non_stop_words	n_non_stop_unique_tokens	num_hre
0	http://mashable.com/2013/01/07/amazon-instant-...	731.0	12.0	219.0	0.663594	1.0	0.815385	4
1	http://mashable.com/2013/01/07/ap-samsung-spon...	731.0	9.0	255.0	0.604743	1.0	0.791946	3
2	http://mashable.com/2013/01/07/apple-40-billio...	731.0	9.0	211.0	0.575130	1.0	0.663866	3
3	http://mashable.com/2013/01/07/astronaut-notre...	731.0	9.0	531.0	0.503788	1.0	0.665635	9
4	http://mashable.com/2013/01/07/att-u-verse-apps/	731.0	13.0	1072.0	0.415646	1.0	0.540890	19

5 rows × 61 columns

negative_polarity	min_negative_polarity	max_negative_polarity	title_subjectivity	title_sentiment_polarity	abs_title_subjectivity	abs_title_sentiment_polarity	shares
-0.350000	-0.600	-0.200000	0.500000	-0.187500	0.000000	0.187500	593
-0.118750	-0.125	-0.100000	0.000000	0.000000	0.500000	0.000000	711
-0.466667	-0.800	-0.133333	0.000000	0.000000	0.500000	0.000000	1500
-0.369697	-0.600	-0.166667	0.000000	0.000000	0.500000	0.000000	1200
-0.220192	-0.500	-0.050000	0.454545	0.136364	0.045455	0.136364	505

The dataset was clean, it mostly did not have any missing or null values.

Model Description

Various models will be fitted on the training data set and tested on validation data for the regression and classification task.

Linear Regression:

Linear Regression is a linear approach to predict the dependent variable with relation to one or more independent variables. Linear regression can be used to determine the degree of relationship between a dependent variable and a number of different independent variables. It is used to make predictions.

- Ridge Regression:
  Ridge regression uses L2 regularization technique. It is a method used to prevent high correlation between independent variables

- Lasso Regression:
  Lasso regression uses L1 regularization technique and could also be used for feature selection.

Logistic Regression:

Logistic Regression is a classification method that aims to find the propensity or probability of a new record to belong to one class. Just like linear regression, it assumes that the predictors are not related and there is no multicollinearity among the predicting variables.

Naive Bayes:

Naive Bayes is one of the strongest probabilistic classifiers in Machine Learning. It uses Bayesian probability to classify and it assumes independence of all the features.

SVM:

Support Vector Machine is a supervised machine learning algorithm that uses support vectors and builds a maximum margin hyperplane to classify the data points to the target variable, the objective of the support vector machine algorithm is to find an optimal separating hyperplane in an N-dimensional space.

Neural Networks:

Neural Networks is a series algorithm that mimics the way the human brain operates. The model can have any number of layers and nodes. Neural network models can learn and predict for datasets that are complex and could also perform better in cases where there is nonlinear relationship between predictors and target variables.

Exploratory Data Analysis (EDA)

Exploratory Data Analysis helps us in understanding the data and the relationship among the variables. We started the exploratory data analysis by checking for the null values. The dataset has no null values.

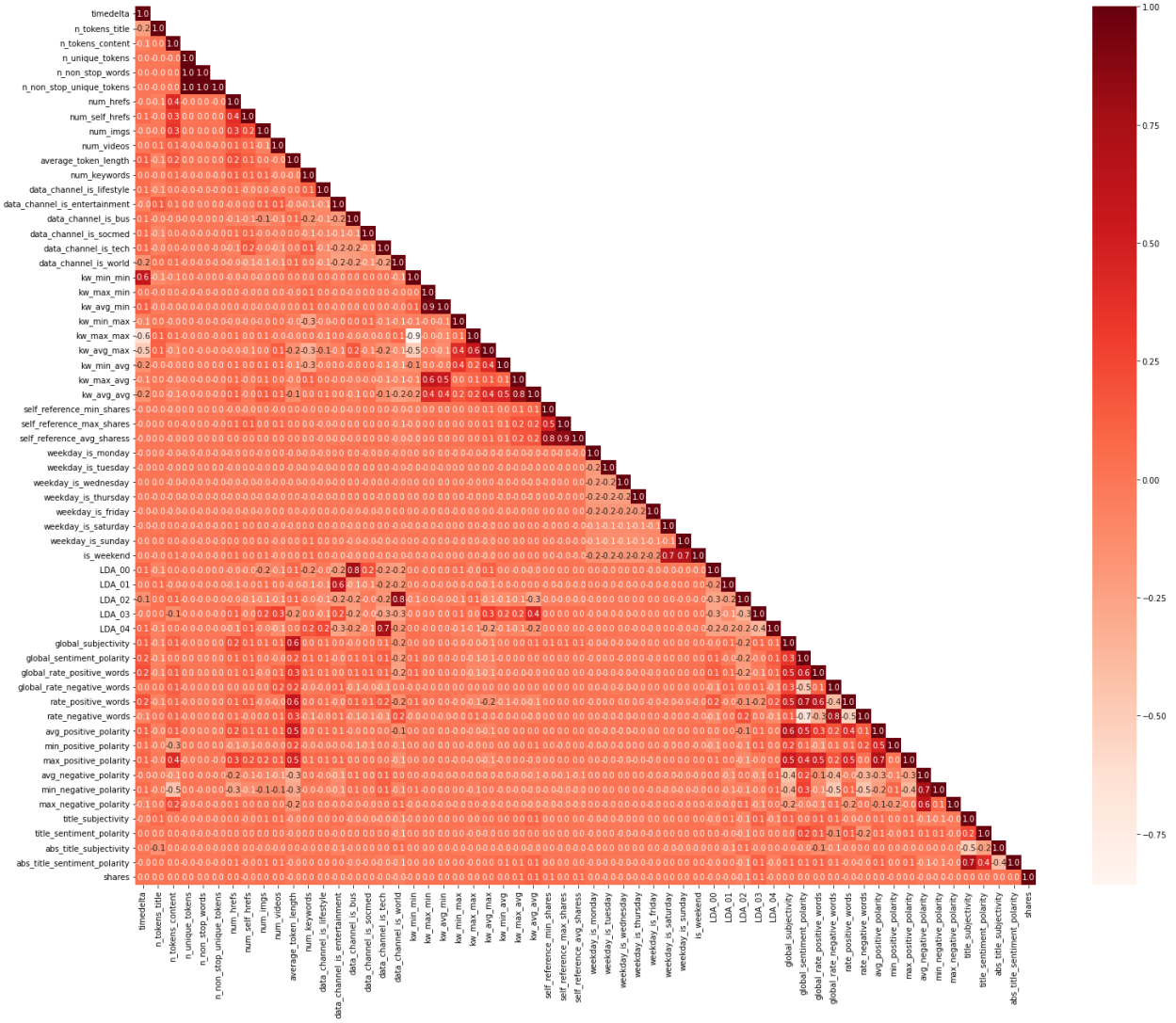

Fig.1

Fig.1 shows a correlation map. We considered features to be highly correlated for correlation greater than or equal to 0.9 and have dropped one of the variables amongst the highly correlated pairs.

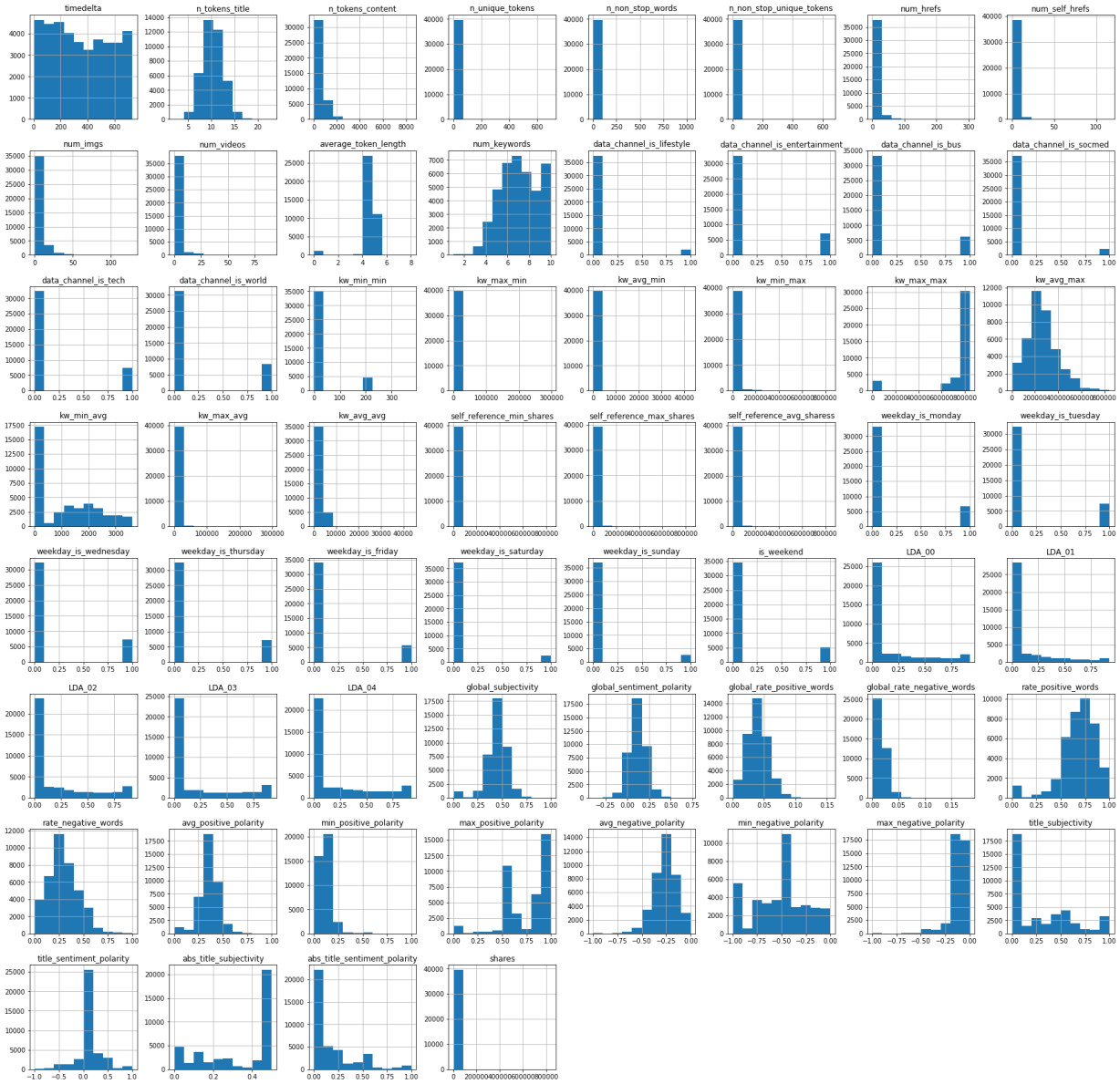

Fig.2

In Fig.2, we can see histograms of all the features. We plotted histograms to check the distribution of the dataset and observed that the data is not normally distributed and not single-mode for most of the features and some features are binary, that is., they have two discrete values - 0 and 1.

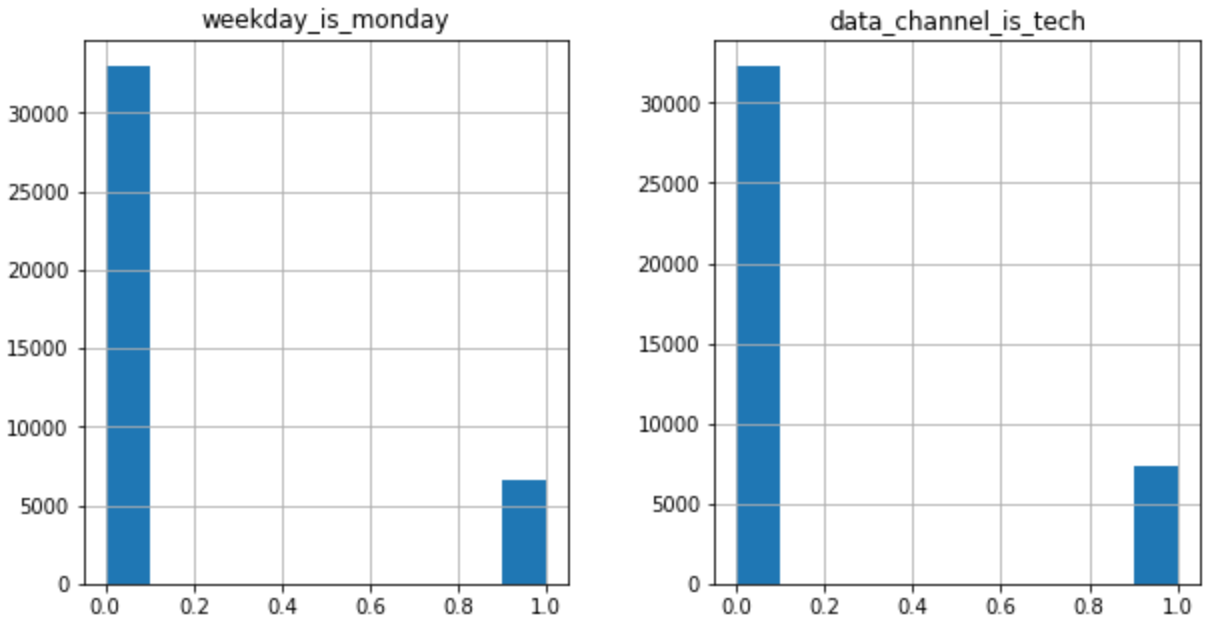

Fig.3

Fig.3 shows two of the binary features. In total, there are 14 binary features. They are: weekday_is_monday, weekday_is_tuesday, weekday_is_wednesday, weekday_is_thursday, weekday_is_friday, weekday_is_saturday, weekday_is_sunday, is_weekend, data_channel_is_lifestyle, data_channel_is_entertainment, data_channel_is_bus, data_channel_is_tech, data_channel_is_world, data_channel_is_socmed.

We later separated the continuous features from the above mentioned binary features and performed feature engineering on continuous features. As part of feature engineering we used lasso regression for feature selection and PCA (details are mentioned in the PCA section) for feature extraction. Two different datasets were extracted from the mentioned techniques and were used for training the models.

Next, we are looking at categorical features with the target feature (shares). The target variable has been used to create a popularity feature with “shares” greater than median as popular article and less than median as not popular article.

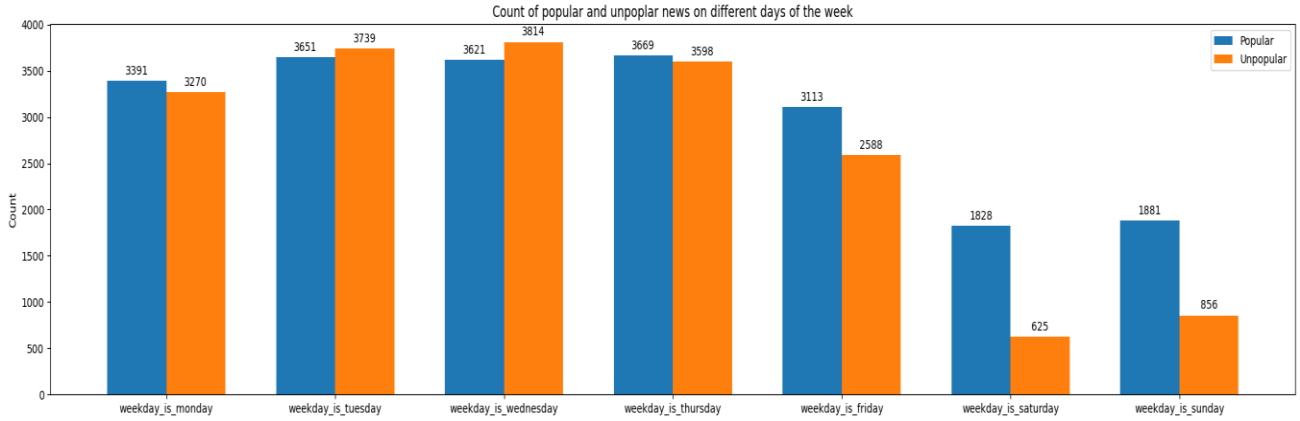

Fig.4

In Fig.4, we can observe that there are more articles published over the weekdays as compared to weekends, however, during the weekends with less count, amongst the released the articles, the popular ones seem to be in great number.

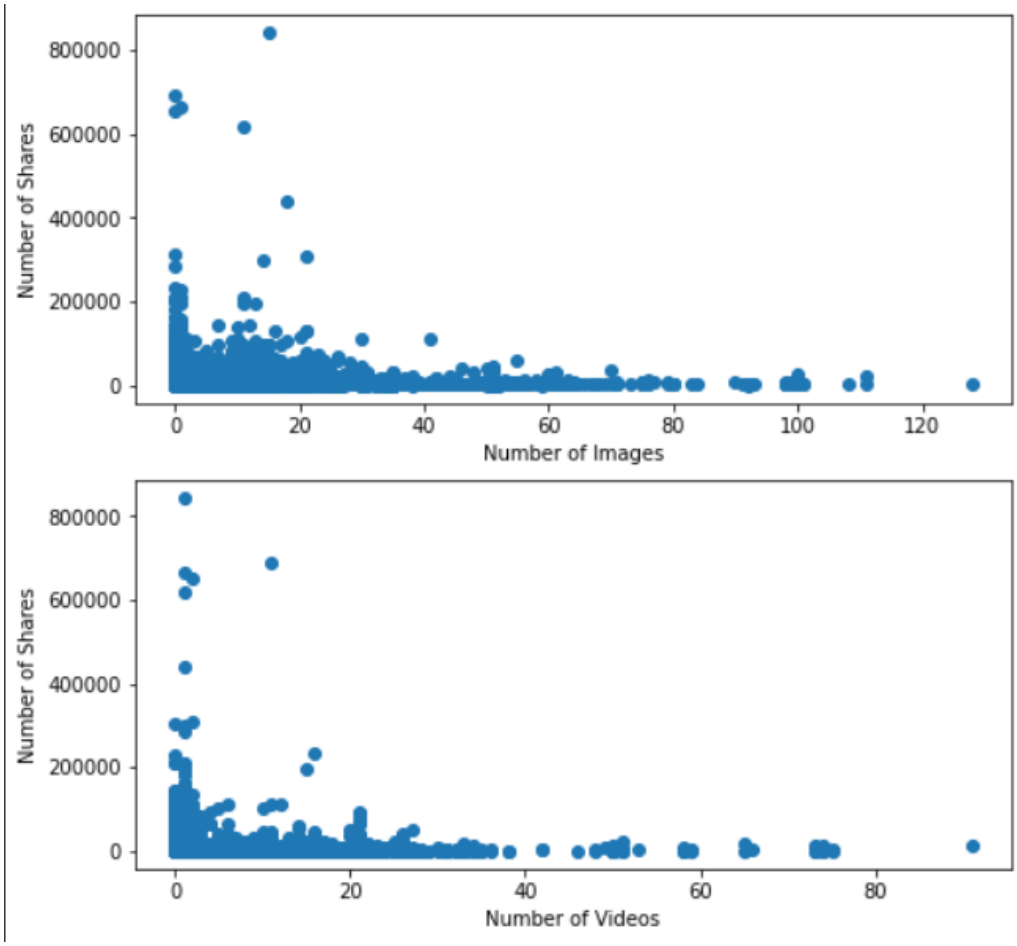

Fig.5

In Fig 5, we tried to check the relation of attributes of a news article like number of videos and images to the number of shares. We observed that as the number of images and videos increased in an article the number of shares reduced.
We further want to check whether the number of images and videos have an effect on the shares when categorized into popular and unpopular based on median value.

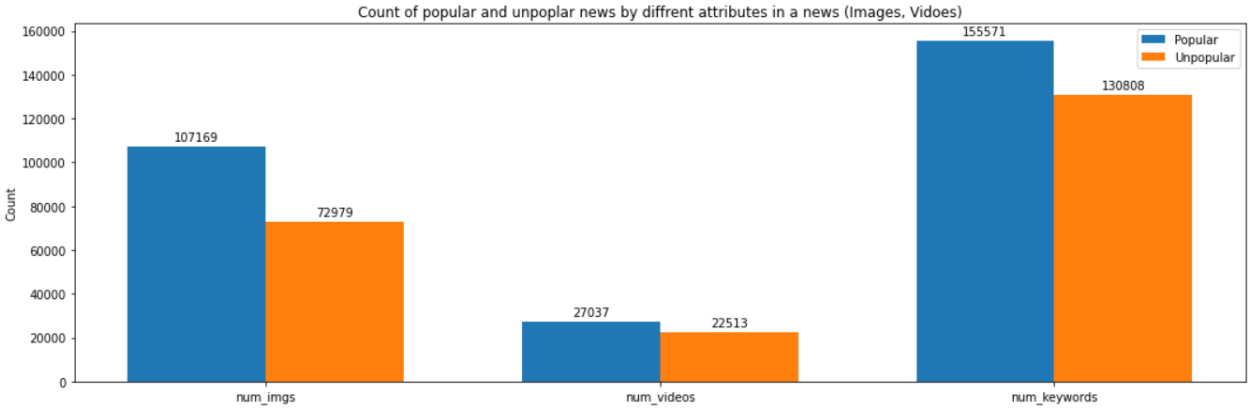

Fig.6

In Fig 6, we observe that we see more articles with keywords and the number of articles with videos is far lesser than that with images. When the number of images and videos are more in an article they are usually popular. We can't explicitly say that articles with videos tend to be more popular as the numbers are close to each other.

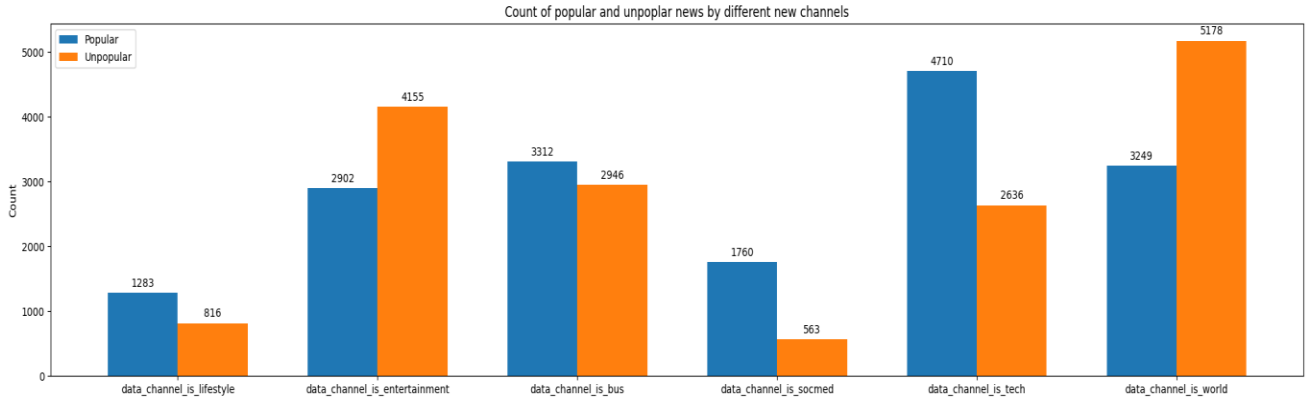

Fig.7

Further, we deep dive into the type of channel of an article with respect to its popularity. The most popularity is observed when an article is of the type, Technology and the most unpopular ones are from channel type entertainment and world.

Feature Engineering

Based on the correlation plot, we dropped 'URL' as it is a non-numeric attribute and is not valuable to our analysis, 'timedelta' as it is not a feature, but a non-predictive attribute and 'n_non_stop_words', 'n_non_stop_unique_tokens', 'kw_avg_min' , 'kw_max_max', 'self_reference_avg_sharess' as they are highly correlated attributes.

Models usually converge faster or perform better when the feature scales are similar and when there is normality of the features in the dataset. So, a parametric, monotonic transformation called power transform was used that helped in mapping data to a gaussian distribution in order to minimize skewness and stabilize variance. The power transformation used was Box-Cox transform. However, Box-Cox can be applied on only strictly positive data. So, all the negative values were converted into positive and then the Box-Cox transformation was applied. After transformation, we performed scaling on the features. We used standard scaler as it assumes features to be normally distributed and scales to have a mean of value 0 and standard deviation of value 1.

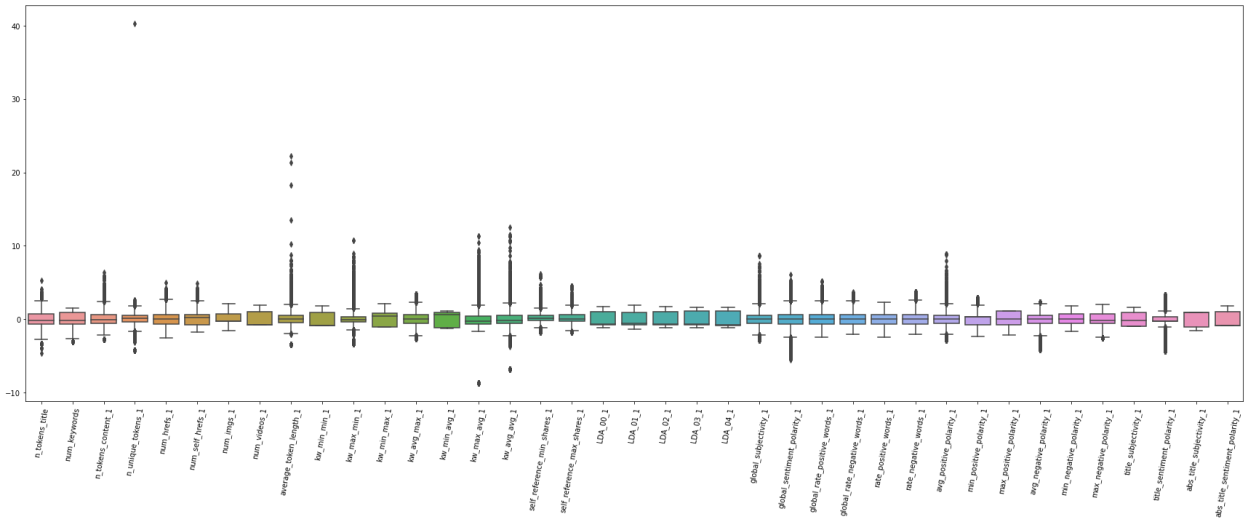

Fig.8

Fig.8 shows the box plots of features after data transformation and scaling. We can clearly see that there are outliers present in the data. So, we removed outliers using standard deviation. We calculated the upper and lower boundary by taking two standard deviations from the mean of the values. The data is approximately in normal distribution as we have performed the Box-Cox transformation. So, in that case, normal distribution has a property that 95% of data falls with two standard deviations of the mean.

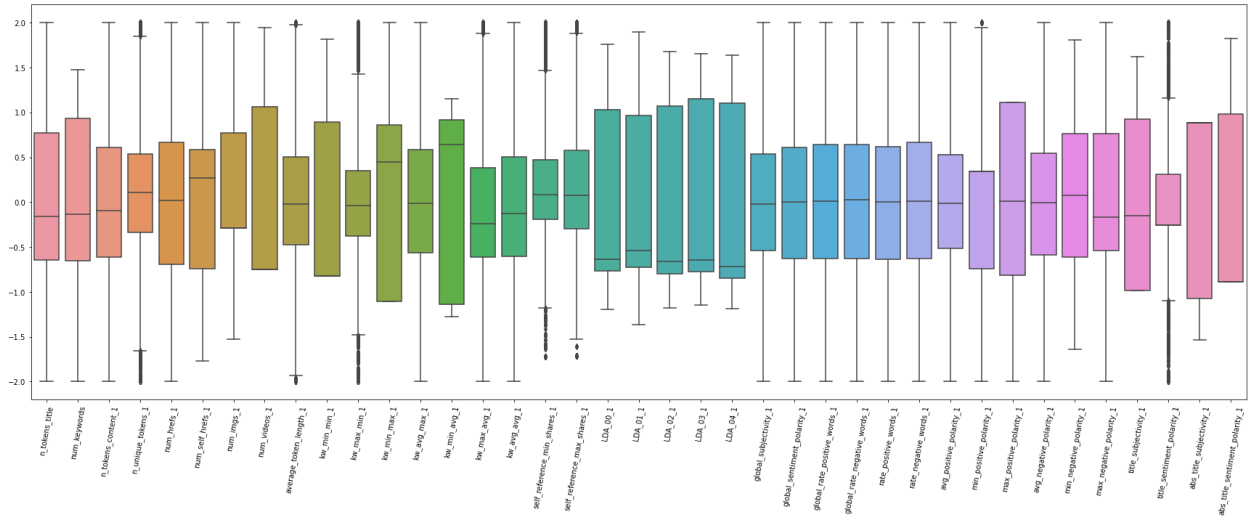

Fig.9

Fig.9 shows the box plots of features after outlier treatment using standard deviation.

Feature Selection:

Lasso Regression

Lasso (Least Absolute Shrinkage and Selection Operator) regularization is used under regression methods to perform feature selection. The method used shrinkage, which is shrinking the data values towards the mean. It uses L1 regularization technique, i.e it adds a penalty.

$$LossFunction = \frac{1}{N}\sum_{i=1}^{N}(\hat{Y} - Y)^2 + \lambda \sum_{i=1}^{N} \mid \theta_i \mid$$

For performing lasso regression, the parameters given were: learningRate=0.00001, tolerance=0.00005, regularizationParam = 0.001 and maxIteration = 1000.

Once lasso regression was performed on the data, the highest top 10 and 15 columns were considered that were highly linearly correlated with the target variable.

The top 10 features obtained from lasso regression :

- Self_reference_max_shares: Max. shares of referenced articles in Mashable
- Kw_avg_avg: Avg. keyword (avg. shares)
- Kw_min_max: Best keyword (min. shares)
- Num_self_hrefs: Number of links to other articles published by Mashable
- Kw_min_avg: Avg. keyword (min. shares)
- Data_channel_is_entertainment: Is data channel 'Entertainment'?
- Num_videos: Number of videos
- N_tokens_content: Number of words in the content
- Num_hrefs: Number of links

The 5 features which showed the least correlation with target variable were:

- Average_token_length: Average length of the words in the content
- Weekday_is_wednesday: Was the article published on a Wednesday?
- Weekday_is_sunday: Was the article published on a Sunday?

- Rate_positive_words: Rate of positive words among non-neutral tokens
- LDA_01: Closeness to LDA topic 1

The top 10 and 15 columns data obtained were considered as separate datasets respectively to further run through the models.

Principal Component Analysis

PCA is an unsupervised dimension reduction technique that is used on top of datasets with high variables to convert them to a data of smaller dimension that still contains the information of the original dataset. Although PCA comes with its own drawbacks like loss of information on dimensionality reduction and less interpretability of the components, it is a small price to pay for handling bias-variance tradeoff.

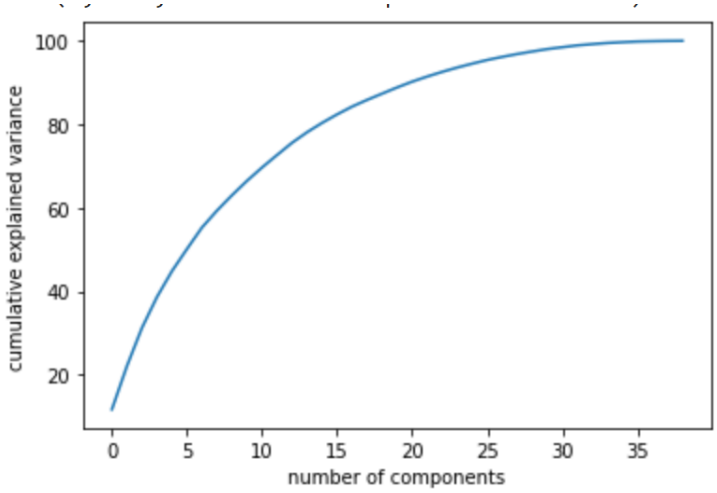

Fig.10

The cumulative sum of explained variance is plotted with the number of variables (Fig 10) for the outlier treatment dataset and variables were selected with 10 components, 15 components and 95% variance captured for analysis.

Implementation of models with evaluation metrics

This section we will have implementation of all models in python code with the help of NumPy and Pandas libraries without using the scikit-learn library. The models will be evaluated based on performance evaluation metrics like accuracy for classification models and RMSE and MAE for regression models.

The datasets were split into train and test with 70:30 ratio. The train data was again split into train and validation sets in the same ratio.

Linear Regression

Linear regression with gradient descent is implemented. Gradient descent is an iterative algorithm used in linear loss function which is the sum of squared error. We first randomly initialize weights and using gradient descent the final weights are obtained which give minimum loss. We use the final weights obtained to predict the output and performance metrics like RMSE(Root Mean Squared Error) and MAE(Mean Absolute Error) for testing data. The model performance is checked with and without ridge regularization. The parameters used for the model are: learningRate=0.00001, tolerance=0.00005, regularizationParam = 0.001 and maxIteration = 1000

$$y = \beta_0 + \beta_1 X_1 + \ldots + \beta_n X_n + \epsilon$$

Logistic Regression

Logistic regression with stochastic gradient descent is implemented and training, validation and test accuracies are obtained for the fixed parameters. The model is improved by changing the parameters iteratively and seeing what combination of parameters work the best . Our dataset involved predicting if a record belongs to class 0 or 1. The sigmoid function is used to calculate values for logistic regression involving two classes; sigmoid is used because it pushes the input values to fall within the range [0,1].

$$Fx_i = 1/(1 + \exp\text{-}x_i)$$

Gaussian Naive Bayes

Gaussian Naive Bayes assumes that each class follows a Gaussian distribution. We had our target feature as two classes (0 and 1), we did this by manipulating the original target feature (shares) 1 when shares greater than median of shares and 0 when shares less than median of shares. As mentioned earlier, normal distribution of data was done using Box Cox method, so having the distribution of values in th columns as gaussian, mean, ,standard deviations are calculated. For any new point, the probability distribution of the dimension values to find the probability of it belonging to the respective distribution is taken into account. The function is -

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Support Vector Machine (SVM)

Support Vector Machine is one of the powerful learning algorithms that captures the non-linearity relationships. We have made use of soft-margin SVM, in SVM the response variable should be 1 or -1, so we converted the target variable to suit SVM's needs. The objective and constraints for the soft-margin SVM optimization problem would be -

$$\begin{aligned} \min_{\mathbf{w},b,\xi_i} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.t.} \quad & y_i(\mathbf{w}^\top \mathbf{x}_i - b) \geq 1 - \xi_i \quad \forall i \\ & \xi_i \geq 0 \quad \forall i \end{aligned}$$

We have used Quadratic programming to solve alphas and this is a very costly operation and time consuming, that's why we sampled the data and have considered only 5000 samples while running the SVM model.

Neural Networks

Neural Network models were implemented with Keras. There were 2 models implemented.
Model 1 (simple): 2 Dense layers 128 and 1 nodes in each layer respectively. Relu activation function is used in the first layer and Sigmoid in the latter one.
Model 2 (complex): 4 dense layers with 128, 128, 64, 1 nodes in each layer respectively with 2 dropout layers of rate 0.5 in both. The activation functions used were Relu in the first 2 layers and Sigmoid in the last 2 layers..
Accuracy was used for performance evaluation in both the models.

Performance Comparison and Results

The model performance is measured by the metrics - RMSE, MAE scores for regression models and Accuracy for the classification models

Dataset 1 - Baseline model

Model	RMSE	MAE
Linear Regression without Regularization	8153.21	3066.08
Linear Regression with Ridge Regularization	8153.20	3066.07

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
Logistic Regression	66.01	64.93	64.95
Gaussian Naive Bayes	62.05	-	61.15

SVM	68.57	60.19	63.06
Neural Network(simple)	92.2	58.1	60
Neural Network (complex)	87	62	63.4

Dataset 2 - After Outiler treatment

Model	RMSE	MAE
Linear Regression without Regularization	8162.71	3076.21
Linear Regression with Ridge Regularization	8162.70	3076.20

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
Logistic Regression	66.20	65.09	65.19
Gaussian Naive Bayes	62.62	-	61.85
SVM	66.20	64.38	63.73
Neural Network (simple)	93.5	61.9	59
Neural Networks (complex)	89	63.6	63

Dataset 3 - After applying PCA
(PCA 95% variance)

Model	RMSE	MAE
Linear Regression without Regularization	8148.98	3048.30
Linear Regression with Ridge Regularization	8148.97	3048.29

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
Logistic Regression	46.21	47.50	47.0
Gaussian Naive Bayes	85.28	-	85.50
SVM	59.54	65.18	64.22
Neural Network(simple)	89.5	57.2	59.5
Neural Networks (complex)	85.8	61.8	61.9

(PCA 15 numerical)

Model	RMSE	MAE
Linear Regression without Regularization	8176.80	3049.13
Linear Regression with Ridge Regularization	8176.79	3049.12

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
Logistic Regression	46.21	47.52	47.0
Gaussian Naive Bayes	85.27	-	85.61
SVM	61.55	66.28	64.93
Neural Network(Simple)	83.3	59	58.1
Neural Networks (complex)	79.3	61	61

Dataset 4 - After Feature Selection using Lasso Regression
(Top 10)

Model	RMSE	MAE
Linear Regression without Regularization	8147.80	3052.05
Linear Regression with Ridge Regularization	8147.79	3052.04

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
Logistic Regression	46.22	47.53	47.00
Gaussian Naive Bayes	90.50	-	90.60
SVM	62.52	61.68	60.87
Neural Network (simple)	67.7	62.2	62.8
Neural Network (complex)	67.4	62.8	64.1

(Top 15)

Model	RMSE	MAE
Linear Regression without Regularization	8159.31	3057.24
Linear Regression with Ridge Regularization	8159.30	3057.23

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
Logistic Regression	46.21	47.52	46.90
Gaussian Naive Bayes	89.69	-	89.87
SVM	62.75	61.46	60.24
Neural Network (simple)	73	61.5	59.7
Neural Network (complex)	72.7	63.1	61.2

Bias - Variance Tradeoff

The property of Bias-variance tradeoff means that the bias of the model can be decreased by increasing the variance across the variables. Similarly the variance of the parameters estimated could be dressed by increasing the bias across the estimated parameters.

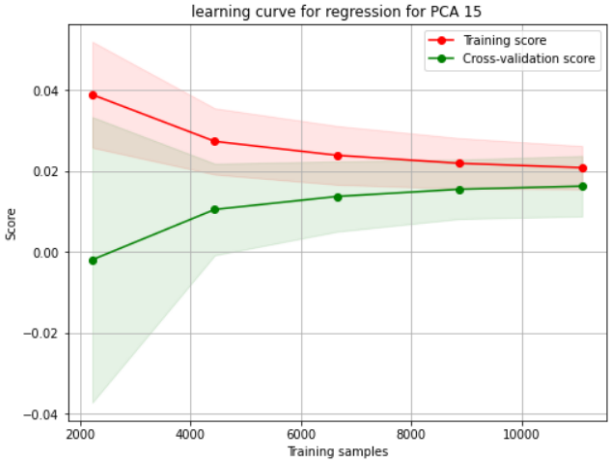

Fig 11

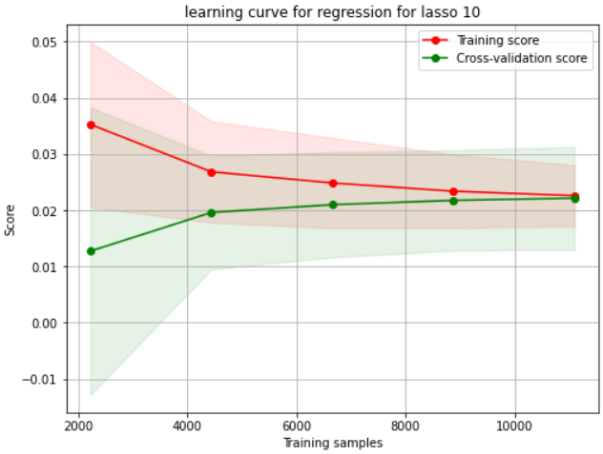

Fig 12

Fig 11,12 show the learning curves for linear regression models with PCA 15 and Lasso 10 datasets. The R squared metrics were used for scores. We see that the graphs are converging implying that the training score is very high at the beginning and decreases and the validation score is very low at the beginning and increases. It shows that the data has high bias.

Discussion and Conclusion

Based on the criteria for model selection, the optimal models would be -.

Model	Dataset	RMSE	MAE
Linear Regression without Regularization	Dataset 4 - After Feature Selection using Lasso Regression (Top 10)	8147.80	3052.05
Linear Regression with Ridge Regularization	Dataset 4 - After Feature Selection using Lasso Regression (Top 10)	8147.79	3052.04

Model	Dataset	Accuracy
Logistic Regression	Dataset 2- After Outlier Treatment	65.19
Gaussian Naive Bayes	Dataset 4 - After Feature Selection using Lasso Regression (Top 10)	90.60
SVM	Dataset 3 - After applying PCA (with 15 numericals)	64.93

Neural Network (simple)	Dataset 4 - After Feature Selection using Lasso Regression (Top 10)	62.8
Neural Network (complex)	Dataset 4 - After Feature Selection using Lasso Regression (Top 10)	64.1

When comparing the neural network models amongst each other, i.e., simple and complex, we see not much of an improvement in performance on tripling the layers in the model and hence we can choose the shallow network for our datasets if needed.

Dataset 4**:** The dataset extracted from lasso regularization with 10 variables performed better with almost all the models. The variables that were selected from this are:

- Self_reference_max_shares: Max. shares of referenced articles in Mashable
- Kw_avg_avg: Avg. keyword (avg. shares)
- Kw_min_max: Best keyword (min. shares)
- Num_self_hrefs: Number of links to other articles published by Mashable
- Kw_min_avg: Avg. keyword (min. shares)
- Data_channel_is_entertainment: Is data channel 'Entertainment'?
- Num_videos: Number of videos
- N_tokens_content: Number of words in the content
- Num_hrefs: Number of links

For classification, the Gaussian Naive Bayes classifier out performed every other model giving the best accuracy rate of 90.60%.

For regression, Dataset 4 performed better for regression as well giving the RMSE and MAE scores as 8147 and 3052 for test data. The regression with regularization and without regularization gave us almost the scores as we see them converge after some iterations.