Capstone Project
Predicting Popularity of Online News Articles

Swapnil Deshpande

1. Definition

1.1 Project Overview

With the advent of the Internet news articles are shared multiple times within moments they published. Also, they often go beyond the published location and being shared worldwide. It is important for the company to see which articles are popular because that can influence their editorial decisions and shows what readers like to read from their publication. For ad supported news model, this provides insights into which articles will fetch more ad revenue.

Lot of methods are used for promoting articles including but not limited to sharing on social media. To make news popular, catchy headlines are often used in predatory way but, there is no reliable way to understand the future popularity of the article. With more focus on removing fake news or misleading article titles and increasing journalistic standards, identifying subject areas and articles which lead to popularity of news article and media house is financially important for publishers.

Academicians have done research on prediction of online news popularity. They have used multiple different criteria to decide if article is popular or not. Tatar Alexandru et al. [1] used comments on article as measure of popularity, (Hensinger, Elena, Ilias Flaounas, and Nello Cristianini) [2] defines popularity in terms of competition where popular articles were the most appealing on that day. Support Vector Machine (SVM) is used to classify most appealing / non-appealing articles of day. In this project we will use dataset of around 40,000 articles from *Mashable*. We will try to find best classification algorithm to predict if news article will become popular or not, before its publication.

1.2 Problem Statement

In this project, our aim is to use machine learning algorithms to predict article will be popular or not, prior to its publication. This is binary classification problem. The popularity is measured in number of shares here, if number of shares are higher than threshold then article is popular otherwise not. We will implement and compare three classification algorithms including gradient boosting, random forest and logistic regression. The best model will be selected based on metrics defined in next part.

1.3 Metrics

We will use Accuracy, F1 score and Area Under Curve to measure the performance of our model against the benchmark model. Since we have balanced dataset for popular and non-popular news articles, we can use all three evaluation criteria described below.

a. Accuracy: Accuracy is the ratio of correct prediction to total predictions

$$Accuracy = \frac{true\ positives + true\ negatives}{total\ predictions}$$

b. F1 Score: F1 score is harmonic mean of precision and recall. We consider it robust measurement since it is not dependent on data distribution.

$$F1\ Score = \frac{2 * precision * recall}{precison + recall}$$

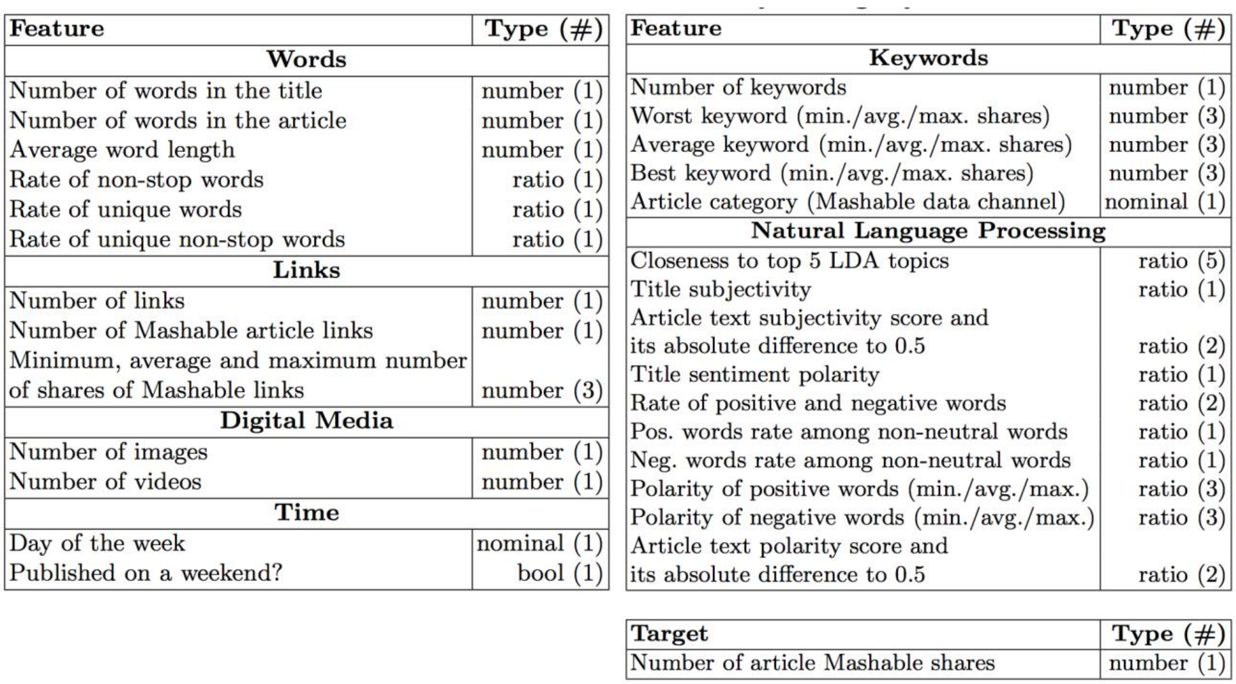
c. AUC is Area Under ROC Curve which is plot of true positive rate vs false positive rate.

2. Analysis

2.1 Data Exploration

This dataset consists of 39643 news articles from an online news website Mashable. These articles are collected over period of 2 years from Jan. 2013 to Jan. 2015. I obtained this dataset from UCI Machine Learning Repository(https://archive.ics.uci.edu/ml/datasets/online+news+popularity).

This dataset has 61 variables including target variable number of shares and 2 non-predictive features (URL and Days between the article publication and dataset acquisition). Removing these variables, we have 58 predictive variable available in the dataset. This dataset is already preprocessed, categorical features like published day of week have been transformed by one-hot encoding scheme. Skewed features like number of words in article has been log-transformed.



Feature	Type (#)
Words	
Number of words in the title	number (1)
Number of words in the article	number (1)
Average word length	number (1)
Rate of non-stop words	ratio (1)
Rate of unique words	ratio (1)
Rate of unique non-stop words	ratio (1)
Links	
Number of links	number (1)
Number of Mashable article links	number (1)
Minimum, average and maximum number of shares of Mashable links	number (3)
Digital Media	
Number of images	number (1)
Number of videos	number (1)
Time	
Day of the week	nominal (1)
Published on a weekend?	bool (1)

Feature	Type (#)
Keywords	
Number of keywords	number (1)
Worst keyword (min./avg./max. shares)	number (3)
Average keyword (min./avg./max. shares)	number (3)
Best keyword (min./avg./max. shares)	number (3)
Article category (Mashable data channel)	nominal (1)
Natural Language Processing	
Closeness to top 5 LDA topics	ratio (5)
Title subjectivity	ratio (1)
Article text subjectivity score and its absolute difference to 0.5	ratio (2)
Title sentiment polarity	ratio (1)
Rate of positive and negative words	ratio (2)
Pos. words rate among non-neutral words	ratio (1)
Neg. words rate among non-neutral words	ratio (1)
Polarity of positive words (min./avg./max.)	ratio (3)
Polarity of negative words (min./avg./max.)	ratio (3)
Article text polarity score and its absolute difference to 0.5	ratio (2)

Target	Type (#)
Number of article Mashable shares	number (1)

Table 1. List of predictive attributes [3]

Before starting to analyze data, we need to decide threshold for number of shares above which article would be considered popular. If we look at the statistics for shares variable, we have median at 1400 and mean at 3395. It makes sense to use 1400 as threshold as it will create balanced group of popular and unpopular articles.

In the next few figures I will try to see if there are any features which closely associate with the number of shares. In Fig.1 I plotted count of popular/ unpopular news against day of the week. We can clearly see that on weekends there is better chance for article to get more shares and it is also obvious since people will have more time over weekends to read and share more. In Fig.2 I have plotted popular articles against category. Tech and Social Media particularly stand out as they have higher popular articles than unpopular news articles. In the Fig. 3 I tried to double check the distribution of the numerical variables. As mentioned earlier since they are log-transformed already distribution looks well behaved and we do not need to preprocess it more.

In fig.4 I plotted mean of number of words in the popular and unpopular news articles. Mean is very close for both categories that means number of words in the article does not really matter for the shares of the article. This is counterintuitive for me since I was assuming less words in article will lead to more shares since it will be quick to read and share.

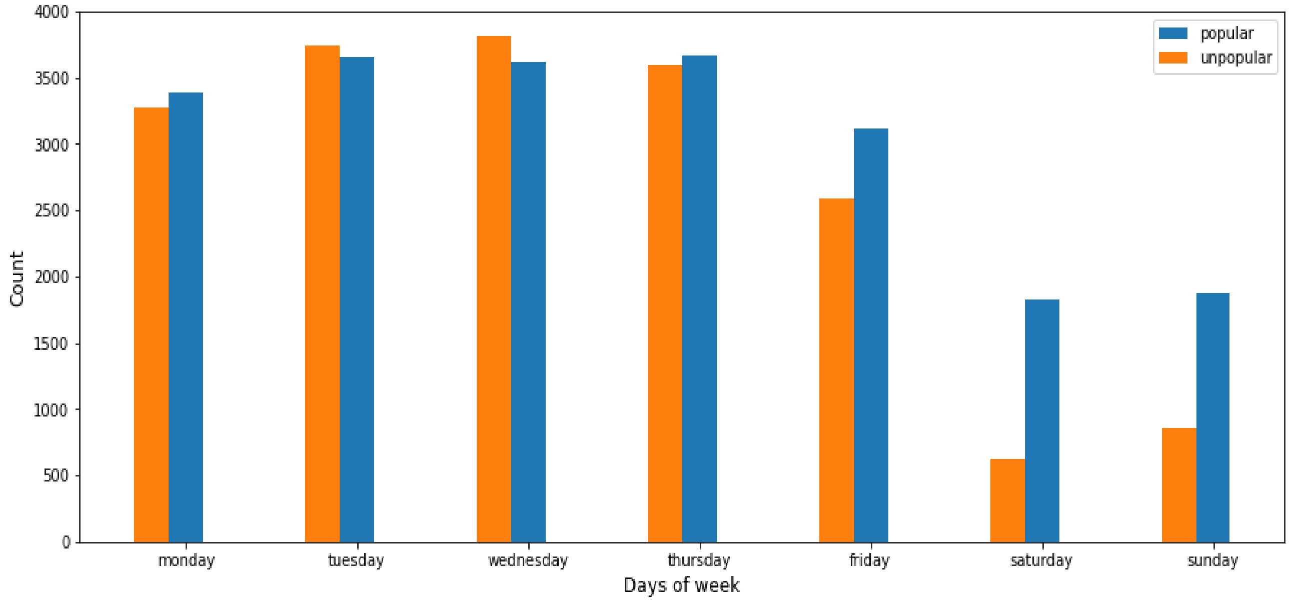


Fig.1 Count of Popular/Unpopular news articles over day of the week

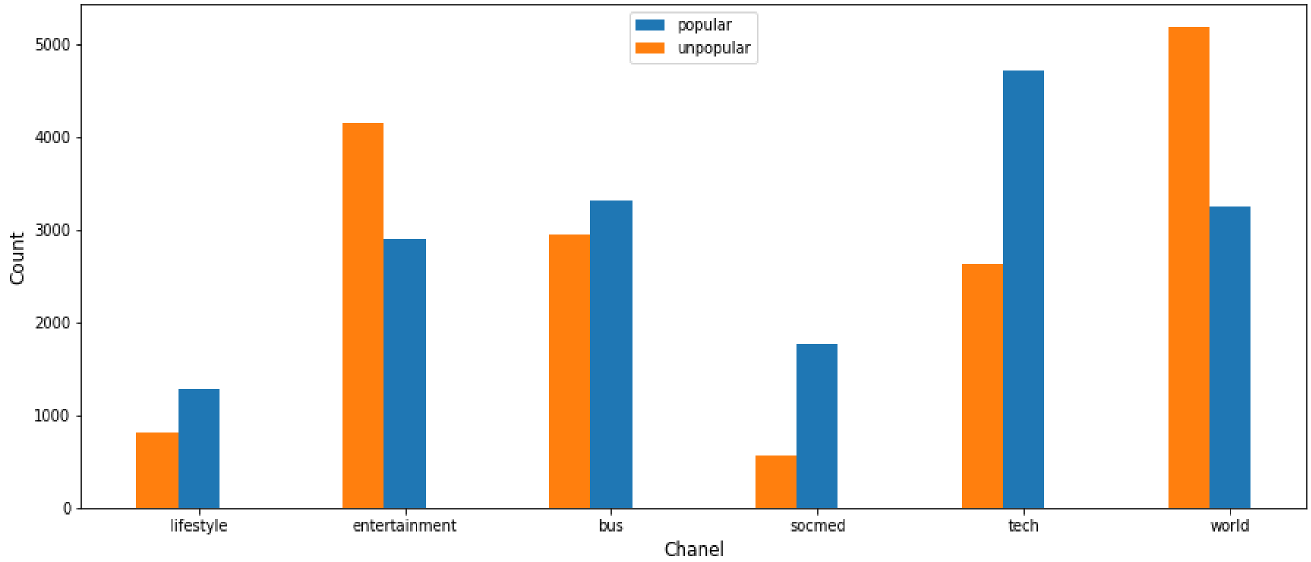


Fig.2 Count of Popular/Unpopular news articles by article category

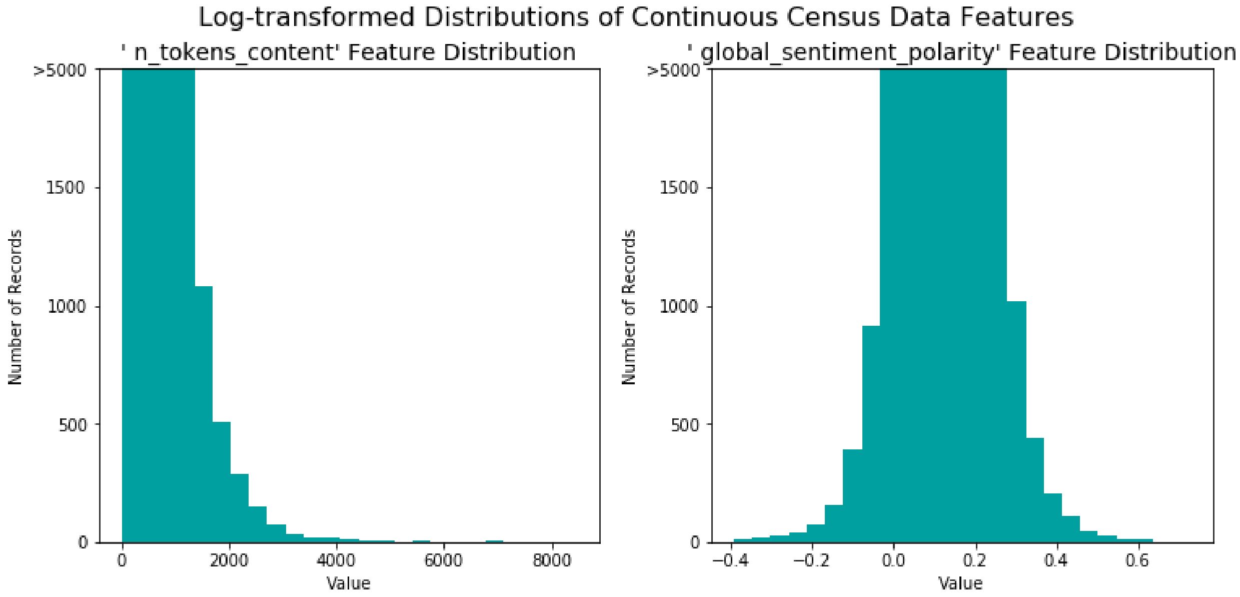


Fig.3 Distribution of few numerical variables in the dataset

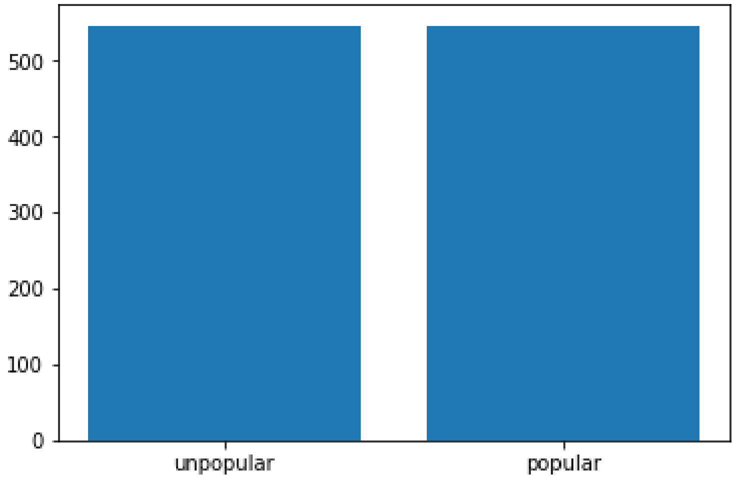


Fig. 4 Mean count of words in article for popular/unpopular news articles

After finishing exploratory data analysis, I normalized numerical features in the dataset. I used MinMaxScalar transformation to perform scaling on numerical variables. After this preprocessing and data exploration we are ready to use supervised learning algorithms on this dataset.

2.2 Algorithms and techniques

We have converted this task in to binary classification problem by converting shares in to binary variable. We will implement supervised learning algorithm which work well with binary classification such as Logistic Regression, Random Forest and Gradient Boosting method. We will evaluate the model based on Accuracy, F-1 score and Area under ROC curve.

1. Logistic Regression
   Logistic regression is a linear model for classification. In this model, the output of a single trial can be interpreted as class probability, which is generated by a logistic function or sigmoid function. One hyperparameter to tune is "C", which is the inverse of regularization strength. The advantage of logistic regression is that the training and predicting speed is very fast and it works particularly well with binary classification.

2. Random Forest
   RF is an ensemble classifier, it fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. We can control hyperparameter called “n_estimators” which is number of trees in the forest.

3. Gradient Boosting
   Gradient Boosting classifier works by combining multiple weak learners to create strong predictor. During the model training, algorithm looks for instances where it had predicted badly, we call it weak learners and increases the weight of correct predictions for those instances in the next round. In each round algorithm finds the strong / best learner and adds that to ensemble. This process continues till the specified number of rounds or until we cannot improve model predictions any further. In the end all the strong learners from each round are combined to make ensemble model. We can tune hyperparameters like learning rate and n_estimators in this model.

First, we will implement all three algorithms with default parameter values, then we will use grid search to optimize values of hyperparameters. For logistic regression we will tune hyperparameter “C”, which is inverse of regularization strength. Smaller the value of C means less chances of over-fitting a model. For random forest we will tune n_estimators which represent number of trees in the forest. Higher number of trees leads to better estimation, but it also increases processing time. In Gradient Boosting method also, we can tune number of trees and trade off with speed is similar. Higher learning rate means small n_estimators are needed but it might lead to over-fitting. On the other hand, smaller learning rate means higher number of n_estimators which slows down the model speed.

2.3 Benchmark

Fernandes, Vinagre and Cortez [3] donated this dataset and published their research. They used multiple models and received best results using Random Forest Classifier with accuracy of 0.67, F1 score 0.69 and Area Under Curve (AUC) 0.73. We will build the model which will either improve or closely match the performance of this model.

3 Methodology

3.1 Data Preprocessing

We have been provided with some sort of pre-processed data from the dataset owner. Hot-encoding for day of the week is already done and also number of words in news article is log-transformed. I have normalized and scaled numerical variables before importing those variables in the model. After doing all this preprocessing I split the dataset into testing and training dataset. Testing data is 20% of the total data size. We will run the model on 1%, 10% and 100% of the training data so have created two additional datasets with 1% and 10% data of the original training dataset. We have included all 58 predictive variables in the dataset.

3.2 Implementation

I used sklearn function library and used LogisticRegression(), GradientBoostingClassifier() and RandomForestClassifier() function the implement three model. I run the model first with default hyperparameter values, which means for logistic regression “C” is 1.0 for RF “n_estimators” is 10 and for GBM learning rate is 0.1 and “n_estimators” is 100. The result comparison graph for all three methods across accuracy score, F1 score, AUC score and time to run is given below.

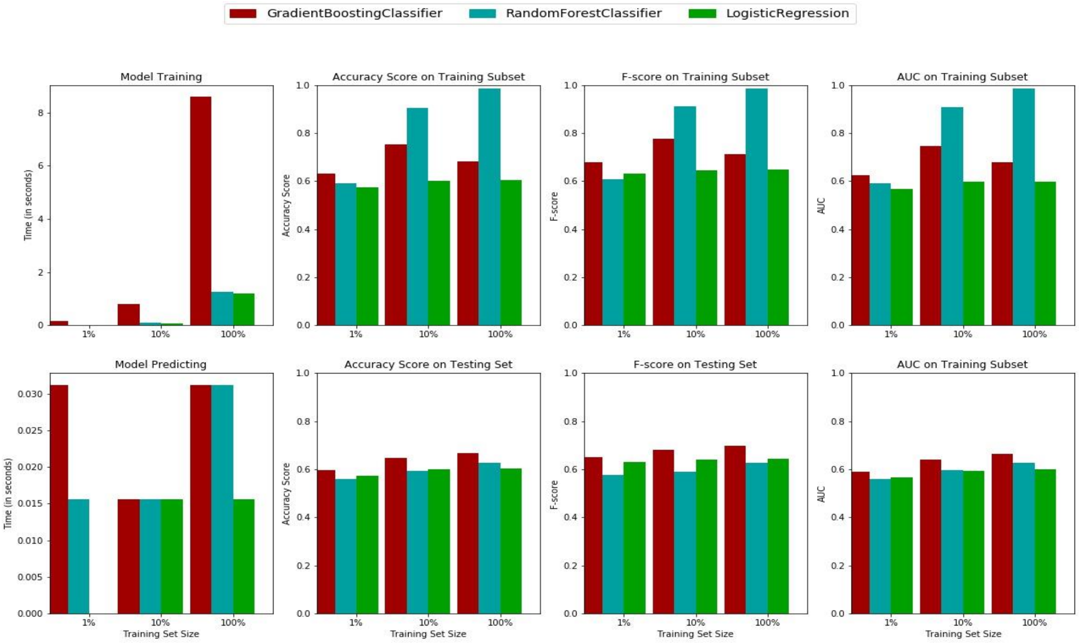
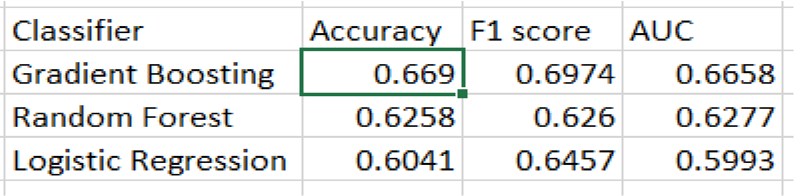


Fig. 5 Performance of classifiers under default parameter settings



Classifier	Accuracy	F1 score	AUC
Gradient Boosting	0.669	0.6974	0.6658
Random Forest	0.6258	0.626	0.6277
Logistic Regression	0.6041	0.6457	0.5993

Table 2. Non optimized score for 3 classifiers

As you can see in the above summary Gradient Boosting provides best score in all categories by some margin. Random forest is second best in all except F1 score. Logistic Regression has better F1 score compared to Random Forest and LR is fastest of the three in terms model training time. Gradient Boosting is slowest amongst these three.

3.3 Refinement

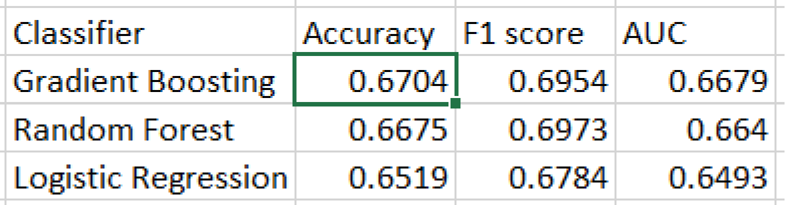
I will try to optimize model performance in this section by tuning hyperparameters. I will use Grid Search to test model against multiple hyperparameter values. Grid Search method search through all possible combinations of model performance, does the cross validation and then decide which set of parameter values gives best performance. I used following parameter and values to tune the model for each classifier.

Gradient Boosting: {"n_estimators": [100,300,500,700], "learning_rate": [0.1,0.5,1]}

Random Forest: {"n_estimators": [50,100,200,250,300,500]}

Logistic Regression: {"penalty": ['l1','l2'], "C": [0.1,0.5,1.,2.,2.5,5]}

After running the grid search the refined parameters are Gradient boosting [“n_estimators” = 300, learning rate = 0.1], Random forest [“n_estimators” = 200] and Logistic Regression [“C” = 2.5, “penalty” = l1]. Now we will run these optimized models again, the scores are summarized below. You can see Random Forest and Logistic Regression has jumped its performance compared to other Gradient Boosting model.



Classifier	Accuracy	F1 score	AUC
Gradient Boosting	0.6704	0.6954	0.6679
Random Forest	0.6675	0.6973	0.664
Logistic Regression	0.6519	0.6784	0.6493

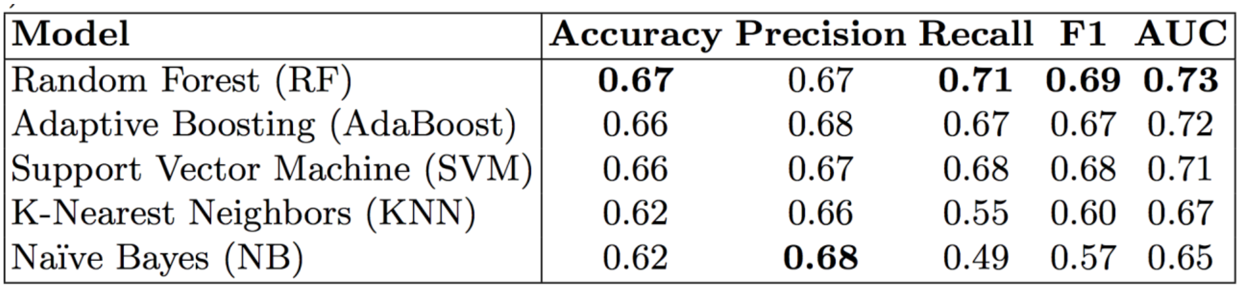
Table 3. Optimized score for three classifiers

4 Results

4.1 Model Evaluation and Validation

After optimizing model parameters, we see that Gradient Boosting is best performer, but Random Forest is very close second. Final accuracy for GBM is 0.6704 and F1 score 0.6954 and AUC 0.6679. Comparing it RF accuracy 0.6675, F1 score 0.6973 and AUC 0.664 we can see they are very different. In fact, by F1 score RF performs better than Gradient boosting. Results are not exceptional, but they match with the benchmark model performance.

4.2 Justification



Model	Accuracy	Precision	Recall	F1	AUC
Random Forest (RF)	**0.67**	0.67	**0.71**	**0.69**	**0.73**
Adaptive Boosting (AdaBoost)	0.66	0.68	0.67	0.67	0.72
Support Vector Machine (SVM)	0.66	0.67	0.68	0.68	0.71
K-Nearest Neighbors (KNN)	0.62	0.66	0.55	0.60	0.67
Naïve Bayes (NB)	0.62	**0.68**	0.49	0.57	0.65

Table 4. Performance of the Benchmark model

Above is the performance metrics for the benchmark model, they received best performance by RF with accuracy 0.67, F1 score o.69 and AUC 0.73. Comparing it to our results we got best performance from GBM and Accuracy and F1 score are in same range but we have significantly less AUC than benchmark model. I would say both GBM and RF do better in terms of Accuracy and F1 score compared to benchmark model and do worse regarding AUC. Overall model achieves comparable performance as benchmark model and results are significant enough to use this model for popular news classification problem.

5 Conclusion

5.1 Free-Form Visualization

Below is the performance of the models after optimizing all parameters. You can see model training time increased a lot for GBM and RF due larger number of trees. Their performance is very close in all measurement metrics now.

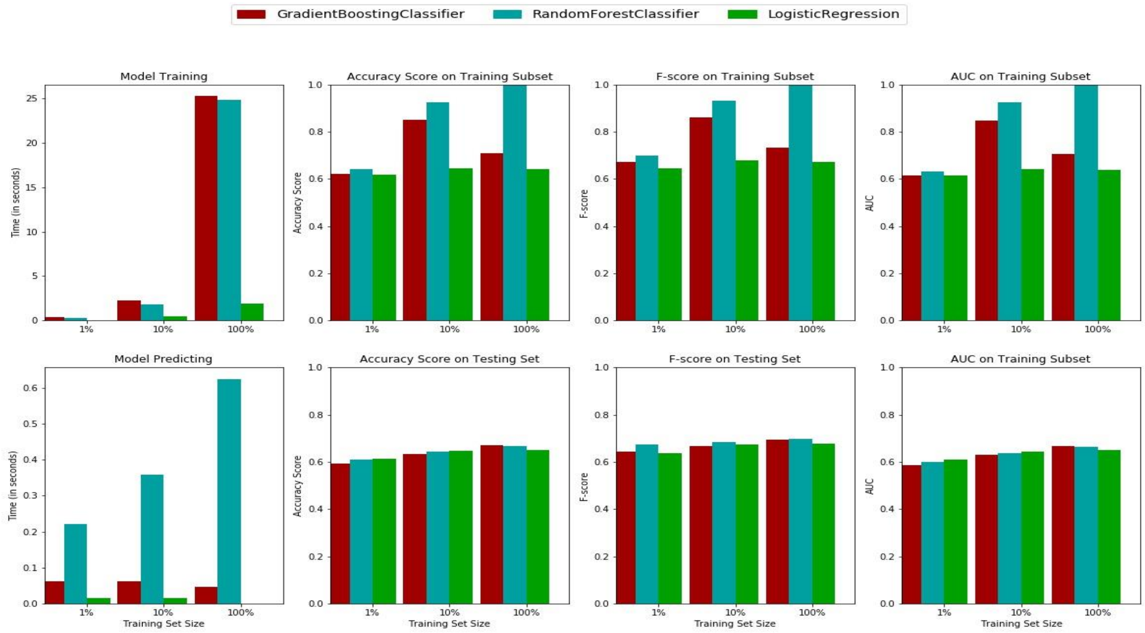
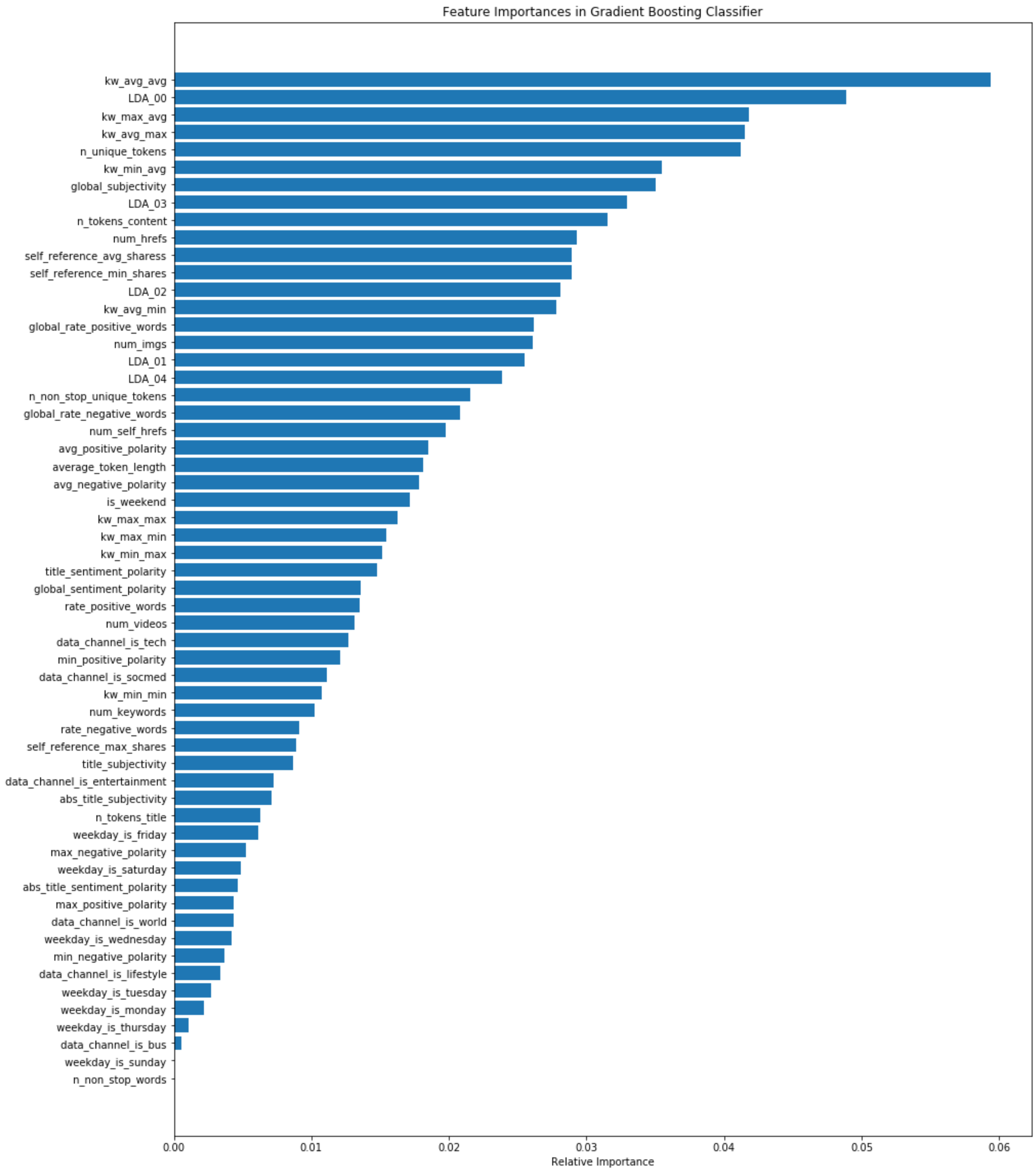
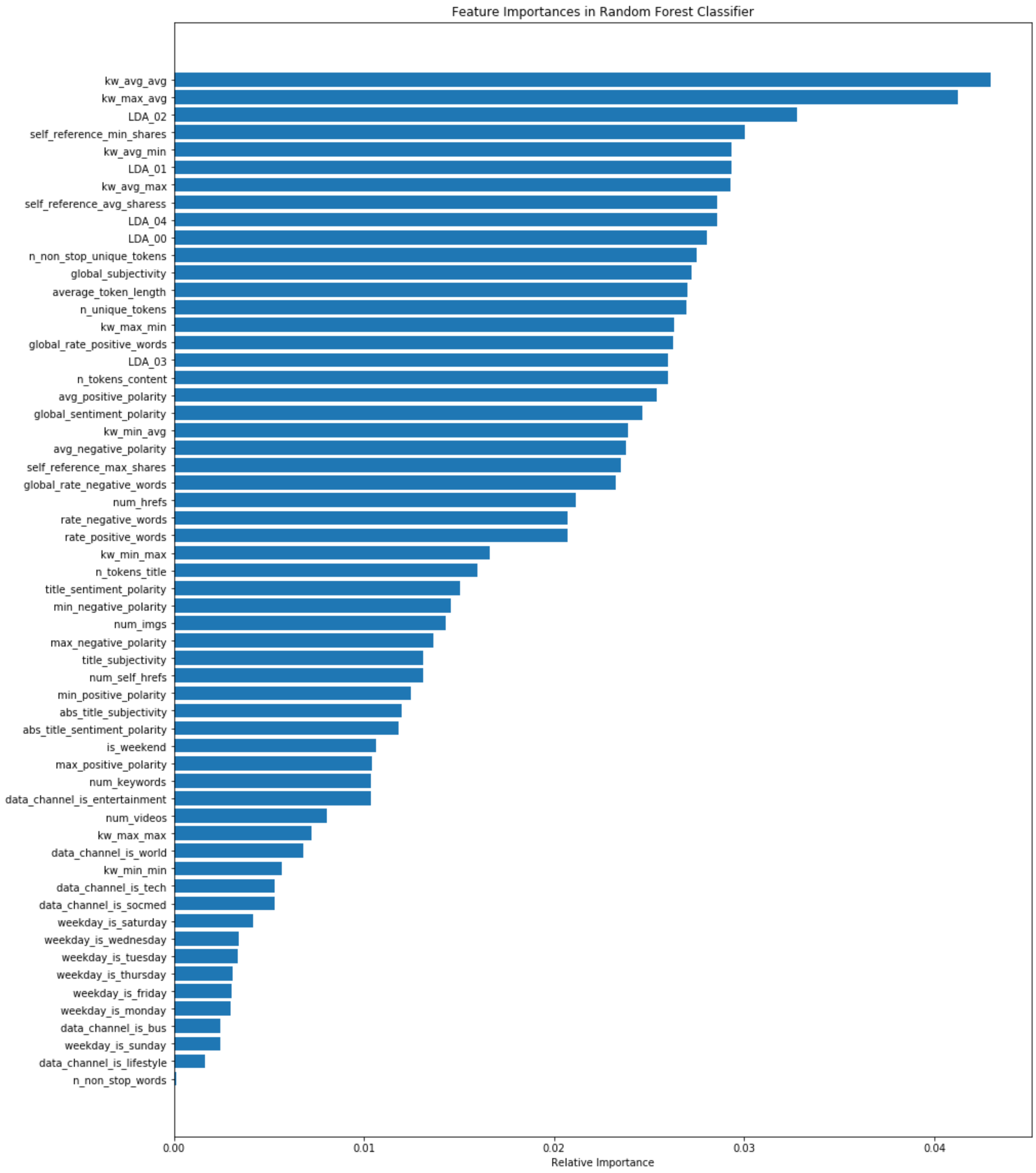


Fig.6 Performance of classifiers under optimized parameter settings

I ran feature importance for the both RF and GBM to see if they see same parameter as important for classification. Turns out that they attach quite different weight to each parameter. Few of the parameters in top 10 are common but some are very different. For example, LDA02 is third important in RF but 13th important in GBM. So, we can say GBM and RF has same (or close to same) performance using different parameters in the data. The feature importance graph for both the methods given below


Feature Importances in Gradient Boosting Classifier
kw_avg_avg
LDA_00
kw_max_avg
kw_avg_max
n_unique_tokens
kw_min_avg
global_subjectivity
LDA_03
n_tokens_content
num_hrefs
self_reference_avg_sharess
self_reference_min_shares
LDA_02
kw_avg_min
global_rate_positive_words
num_imgs
LDA_01
LDA_04
n_non_stop_unique_tokens
global_rate_negative_words
num_self_hrefs
avg_positive_polarity
average_token_length
avg_negative_polarity
is_weekend
kw_max_max
kw_max_min
kw_min_max
title_sentiment_polarity
global_sentiment_polarity
rate_positive_words
num_videos
data_channel_is_tech
min_positive_polarity
data_channel_is_socmed
kw_min_min
num_keywords
rate_negative_words
self_reference_max_shares
title_subjectivity
data_channel_is_entertainment
abs_title_subjectivity
n_tokens_title
weekday_is_friday
max_negative_polarity
weekday_is_saturday
abs_title_sentiment_polarity
max_positive_polarity
data_channel_is_world
weekday_is_wednesday
min_negative_polarity
data_channel_is_lifestyle
weekday_is_tuesday
weekday_is_monday
weekday_is_thursday
data_channel_is_bus
weekday_is_sunday
n_non_stop_words
0.00
0.01
0.02
0.03
0.04
0.05
0.06
Relative Importance



Feature Importances in Random Forest Classifier

Fig 7. Feature importance in RF and GBM Models

5.2 Reflection

I started from collecting the data from UCI machine learning repository. Dataset contains 40000 online news articles published during Jan 2013 to Jan 2015. This dataset is donated by K. Fernandes, P. Vinagre and P. Cortez [3].

After acquiring I preprocessed data such normalized and scaled all numerical variables so that they would be treated equally in supervised learning model. I also selected threshold as median value of shares to convert shares in binary variable. After that I plotted different graphs to see any pattern in the data and to confirm distribution of variables.

I used three classification algorithms Logistic Regression, Random Forest and Gradient Boosting method. First, I used them with default values of hyperparameters and after that I tuned hyperparameter values using Grid Search method. I compared performance of tuned model against benchmark model using accuracy, F1 and AUC score. I plotted feature importance for best scoring two models to see which variables are important for each model.

The surprising part is I found both models don't have similar top 10 features and different is quite large for example, 3rd important variable in RF is 13th important in GBM. Hardest thing for me is how to define problem should shares variable be continuous for to make it binary? I also spent time of thinking which classification algorithms to choose. This problem more difficult if shares would have been continuous target variable but when I converted it to binary, choices for classification algorithm got narrowed down. In the end 2 of the 3 selected models achieve very good performance and match the performance of the benchmark model.

5.3 Improvement

I got impressive performance from two algorithms which is very close to benchmark model, but I think there is scope for improvement.

1. Increase the size of dataset – Both RF and GBM good with working larger data so increasing the data size of dataset and including more variables will be helpful to improve performance. Additional variables can be included in the dataset such as tagging parameters associated with article and method of share like weblink or Facebook /twitter share. This will help to understand how social media can be used to increase number of shares.
2. Try same model with advanced cross validation methods. This might increase training time especially with a greater number of trees but provide better performance.

References

1. 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.

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

3. K. Fernandes, P. Vinagre and P. Cortez. *“A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News*”. Proceedings of the 17th EPIA 2015 - Portuguese Conference on Artificial Intelligence, September, Coimbra, Portugal.