Predicting popularity of a news article based on general sentiment

Silvija Radzevičiūtė (14002833), ZhaoYu Tu (13987496), Gergana Ivanova (14029154)

I. Introduction

The development of modern technology has altered the traditional way of living in a number of ways. Just a few decades ago, the most common source of news were the newspapers, being printed out every day. Nowadays, substantial numbers of news articles are being published on the internet every second of every day all over the world. These articles belong to certain categories like entertainment, technology, sports etc., covering stories with mixed emotions, including happy, terrible, and neutral. Sentiment analysis, often known as opinion mining, is a technique used in natural language processing (NLP) to evaluate the emotional undertone of a document. Our goal is to predict whether general sentiment of an article will make it popular.

II. Dataset explanation

K. Fernandes, P. Vinagre and P. Cortez were the ones who initially obtained and pre-processed the dataset. There are 61 attributes in the dataset (58 predictive attributes, 2 non-predictive, 1 goal field), describing different aspects of 39,644 news articles.

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. Features description

III. Feature selection

Our goal of this project is to predict popularity of an article using sentiment analysis and see in which fields it is the most important, therefore we use all the Natural Language Processing features, listed above in Table 1.

```
data.columns
```

```
Index(['LDA_00', 'LDA_01', 'LDA_02', 'LDA_03', 'LDA_04', 'global_subjectivity',
       'global_sentiment_polarity', 'global_rate_positive_words',
       'global_rate_negative_words', 'rate_positive_words',
       'rate_negative_words', 'avg_positive_polarity', 'min_positive_polarity',
       'max_positive_polarity', 'avg_negative_polarity',
       'min_negative_polarity', 'max_negative_polarity', 'shares',
       'shares_cat'],
      dtype='object')
```

IV. Data pre-processing

One of the first things we did was *clean* the data. The data is mostly clean and processed already, however, we found that the column names have an empty space at the beginning, so we deleted it.

Also, we deleted the outliers of our target variable, that we detected using boxplots and the ‘describe’ function. As we can see, before deleting the outliers maximum of shares was 843,300, after it is 5,500.

```
[55] data['shares'].describe()
```

```
count     39644.000000
mean       3395.380184
std       11626.950749
min           1.000000
25%         946.000000
50%        1400.000000
75%        2800.000000
max      843300.000000
Name: shares, dtype: float64
```

Before deleting the outliers

```
data['shares'].describe()
```

```
count    35103.000000
mean      1671.972652
std       1103.199563
min          1.000000
25%        903.000000
50%       1300.000000
75%       2100.000000
max       5500.000000
Name: shares, dtype: float64
```

After deleting the outliers

Some machine learning algorithms' objective functions won't function effectively without normalization since raw data's value range varies greatly and it may affect the prediction of some features more than the others. Therefore, we *scaled* the data using MinMaxScaler.

The last step of the data pre-processing was converting our target variable (shares) into binary numbers. This process is called *binary classification*. In our experiment, we chose 0 as deeming an article to be unpopular (with less than 1400 shares) and 1 as it being popular (with over 1400 shares). We chose 1400 as the threshold, because it is the median of shares without the outliers and there is almost no class imbalance (60/40). After testing with multi-class classification (5 classes), we determined that for our data, it yields accuracy two times lower (~0.30) compared to binary (~0.60 accuracy). For that reason, we decided to use binary classification.

V. Classification Formulations

We demonstrated classifiers including LinearSVC, Logistic Regression, KNeighbors Classifiers and RandomForest to our filtered dataset. By utilizing GridSearchCV and doing 5-fold cross validation, we found the best parameter, training and test set scores(rounded up to 2 decimals) for each classifier. After experimenting with different models, the classification reports were used for evaluating model performances.

- **LinearSVC**

  LinearSVC model was tuned with 4 parameters, [0.01, 0.1, 1, 10]. With 17 features used, it showed that C=10 is the optimal parameter with the training score 0.60, and test score is 0.60. Since LinearSVC model uses all the features and is highly possible to have underfitting problem, this is not the best model.

```
LinearSVC(C=10)
Training set score: 0.60
Test set score: 0.60
Number of features used: 17
```

- **KNeighbors Claasifier**

  KNeighbors classifier was tuned with parameters n_neighbors ranging from 0 to 30. It is shown that the optimal n_neighbors is 24 with the training set score 0.64 and 0.60 for the test set score. Comparing with LinearSVC, KNeighbors classifier performed slightly better with lower possibility of underfitting issue.

```
KNeighborsClassifier(n_neighbors=24)
Training set score: 0.64
Test set score: 0.60
```

- **Logistic Regression**

We run Logistic Regression with a 5-fold GridSearchCV to find the best value for parameter C = 0.1. Fitting the model yields 0.60 for both the training and test sets while using all of the features in our dataset.

```
LogisticRegression(C=0.1, max_iter=1000)
Training test score: 0.60
Test set score: 0.60
Number of features used: 17
```

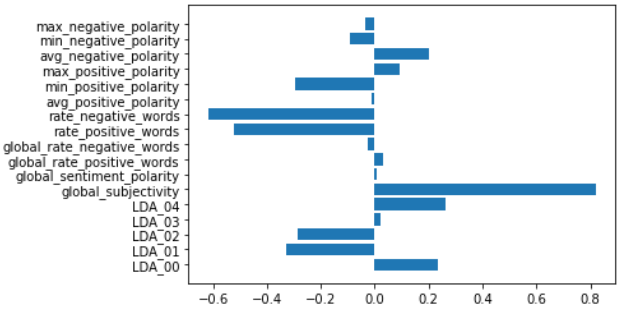
- **Random Forest Classifier**

With random forest it is crucial to choose the optimal parameters first for the model to work well. Therefore, we chose 3 parameters to tune: max_depth, n_estimators and max_features. The optimal parameters were max_depth=10, max_features=10 and n_estimators=1000. Fitting the model yields 0.75 on training set and 0.61 on test set. The test set score is the best of all models, however we can see the case of overfitting, because the training set score is higher.

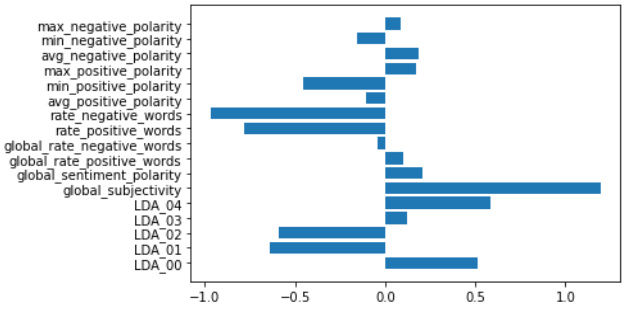
```
Accuracy on training set: 0.748
Accuracy on test set: 0.612
```

Coefficients:

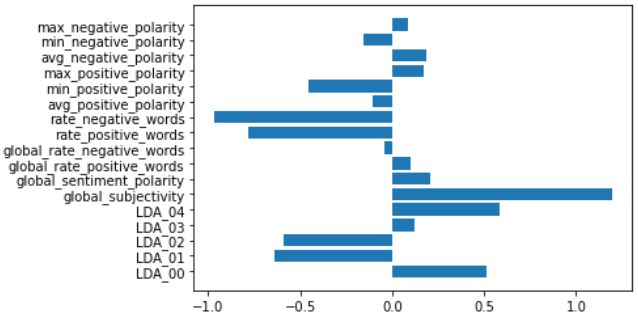
After fitting all models on the dataset, we can retrieve the *coef_* (execpt for KNeighbors) property that contains the coefficients found for each input variable:



Graph 1. feature importance coeff. for LinearSVC.



Graph 2. feature importance for Logistic Regression.



Graph 3. feature importance coeff. for Random Forest.

We can notice from the box plots above (Graph 1, 2, 3) that the coefficients are both positive and negative. As our classification problem is binary, the positive scores indicate a feature that predicts class 1 (a popular article), whereas the negative scores indicate a feature that predicts class 0 (an unpopular article). We can gather that an important factor within popular articles is that they have a high score for global subjectivity, LDA Topic 4 and LDA Topic 0. Global subjectivity has the highest value ~1.96 for both Logistic Regression and Random Forest, making it the most important coefficient feature in determining popularity. Where as unpopular articles tend to have a high rate for strongly negative and positive words as well as for LDA Topic 1 and 2.

VI. Model Comparison

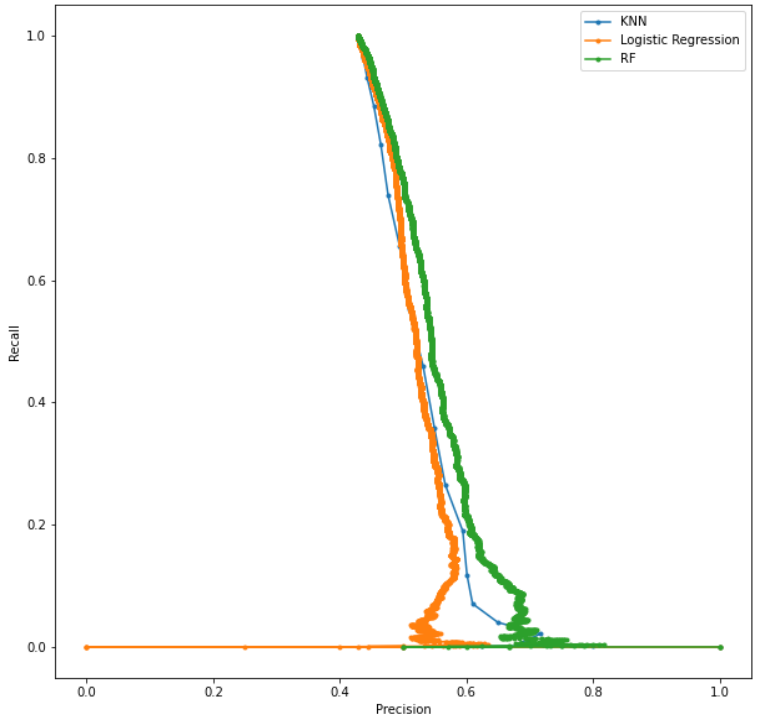
As we can see, Random Forest has the best result for both f1-score and average precision, meaning that the quality of a positive prediction and harmonic mean of precision and recall are better than other models.

Models	Training score	Test score	Accuracy	F1-score	Average Precision
LinearSVC	0.60	0.60	0.596	0.442	0.470
KNeighbors	0.64	0.60	0.599	0.433	0.472
Logistic Regression	0.60	0.60	0.595	0.430	0.516
Random Forest	0.75	0.61	0.608	0.456	0.553

Table 2. training scores, test scores, classification accuracy, f1-scores and average-precision (AP) from different Machine Learning Algorithms.

- **Precision-Recall curve**

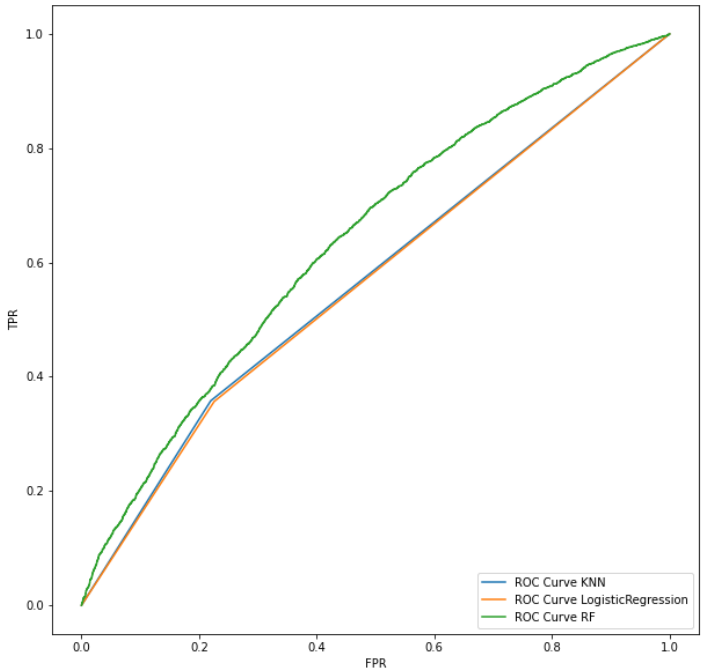
  We take the best three models, KNeighbors, Logistic Regression and Random Forest, to make the Recall-Precision Curve. As the graph shown, Random Forest outperformed the other two models, given that the best outcome will be higher precision and higher recall.



Graph 4. Precision-Recall curve

- **ROC Curve**

  Using the smodel selection from the Recall-Precision Curve, it demonstrates that Random Forest outperforms the other two models, given that our goal with the ROC curve is to get as “top-left” as possible. Since the ideal situation is to get lower False Positive Rate(FPR) and higher True Positive Rate(TPR).



Graph 5. ROC Curve

In general, Precision-Recall Curve is well-suited for an experiment that has significant class imbalance while for the ROC curve the opposite applies. Overall, ROC curve is better presented in our scenario since we barely have the class imbalance problem (see classification reports from the notebook).

VII. Conclusion

To conclude, out of the four models we tested, Random Forest yields the best results. This is supported by the F1 and precision scores (see Table 2.), as well as by the ROC curve (Graph 5.). We get an accuracy score of ~0.60.

From our analysis on general sentiment, we can confirm that it does play a role in determining the popularity of an article. The most important parameter coefficient for predicting a popular article is the global subjectivity. This also confirms, as researchers have found that people give more attention to the information supporting their beliefs (Frost et al., 2015). Thus, a subjectively written article has a higher chance of going popular. The usage of strongly negative and positive words within the article will influence the prediction to an unpopular article.

VIII. References

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.

Modeling confirmation bias and polarization. *Scientific reports*, *7*(1), 1-9.

Frost, P., Casey, B., Griffin, K., Raymundo, L., Farrell, C., & Carrigan, R. (2015). The influence of confirmation bias on memory and source monitoring. *The Journal of general psychology*, *142*(4), 238-252.