Group Project
COMP9417 Machine Learning and Data Mining
T1, 2020

Group Name: GLaDOS
Group Members: Jessica Boyle z5163641, Amy Chov z3461086, Nathan Lam z5113345, Jack Murrie z5207632, Jingtao Zhou z5188310

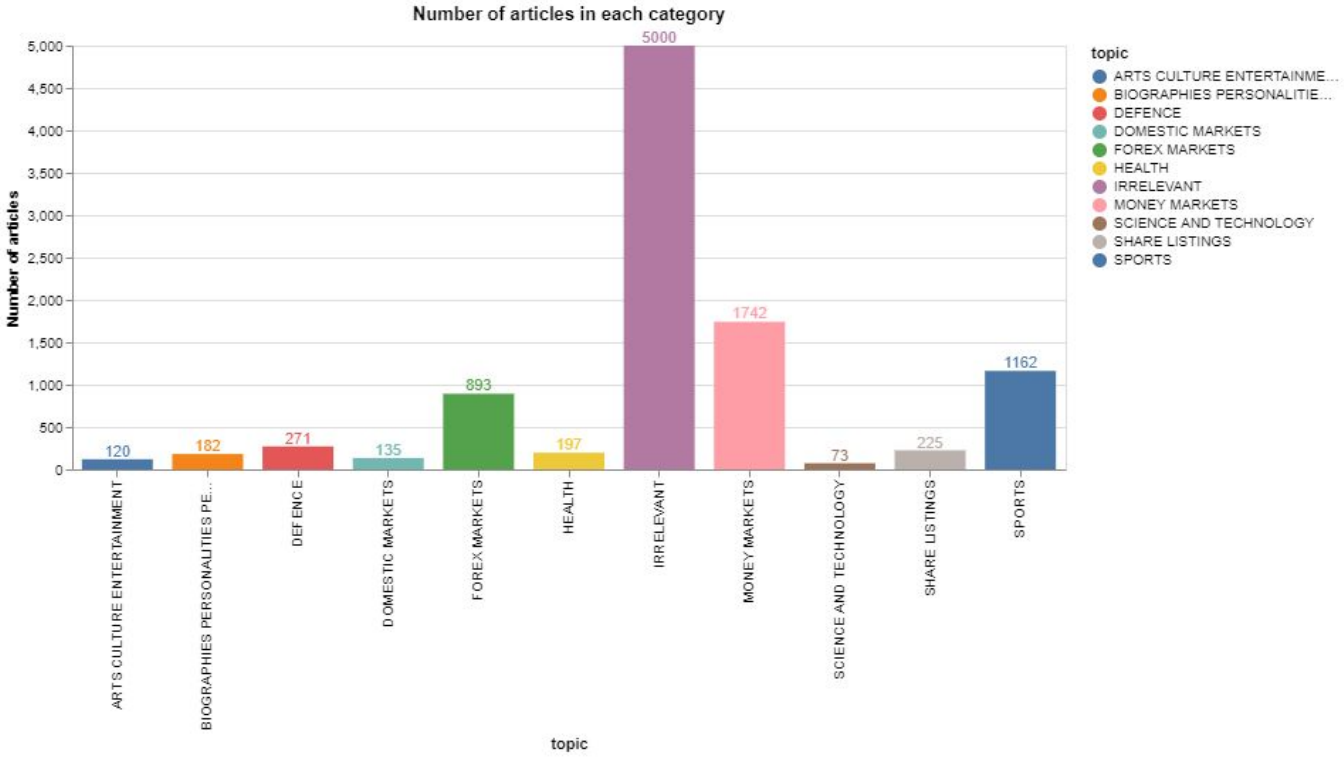
Introduction

The problem presented in this assignment is how to help users find the most interesting articles to read according to their preferred interest topics. The aim of this project is to conduct text classification on the article data provided in order to predict the most relevant news articles for each user by classifying the articles into ten distinct topics. In addition, we will present the user with up to ten of the most relevant articles per category. We will be developing two methods and then selecting the best model and parameters to predict the most relevant articles.

Some of the main issues that need to be addressed are the following. The first issue is that the original dataset is mixed up with some irrelevant articles which do not belong to any of the topics. Therefore, information filtering is a crucial process. We need to extract useful information then create features based on that. During the data exploration phase we will explore this issue further and come up with a method to address this. Another issue is that the dataset does not contain an approximately equal portion of each topic. Some topics may have less than ten relevant articles to recommend and irrelevant articles should not be suggested to the users. To solve this we will investigate using a cut-off threshold value on our chosen performance metric. This way the suggestions will not just be the top articles recommended by our method but they will have to also be over this benchmark value. An additional point is that not all features in the data set are as relevant as others, we will aim to find the relatively best combination of features from the original dataset.

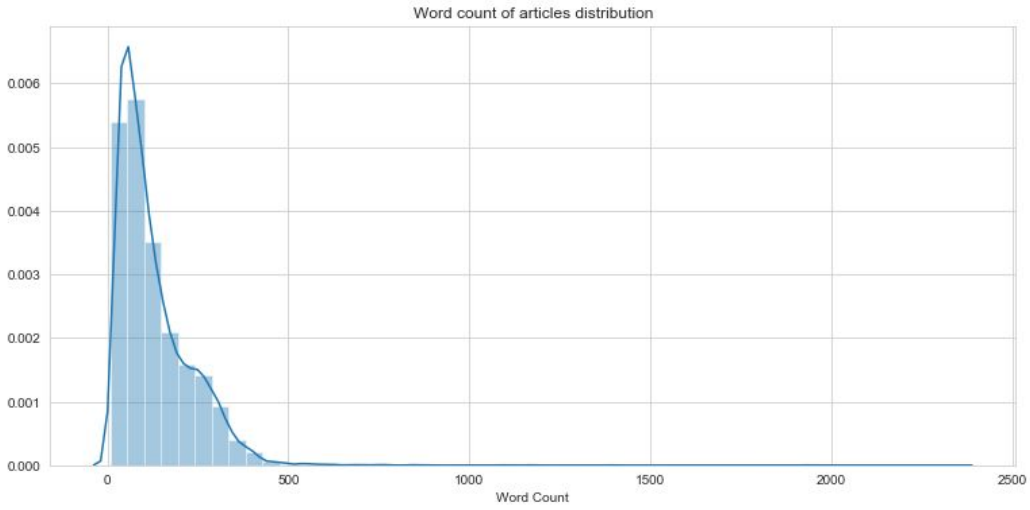
Exploratory data analysis

We performed some exploratory data analysis to get an understanding of the distribution of classes and the features. The first thing we investigated was the distribution of the classes, i.e the number of articles in the data for each topic. This is important because it helps us identify if the data is balanced. If the data is imbalanced we might need to consider oversampling from minority classes and undersampling the majority classes. Looking at the graph below we can see that there is a very large number of irrelevant articles. Aside from the irrelevant articles that other majority classes are the topics ‘Money Markets’, ‘Sports’ and ‘Forex Markets’.

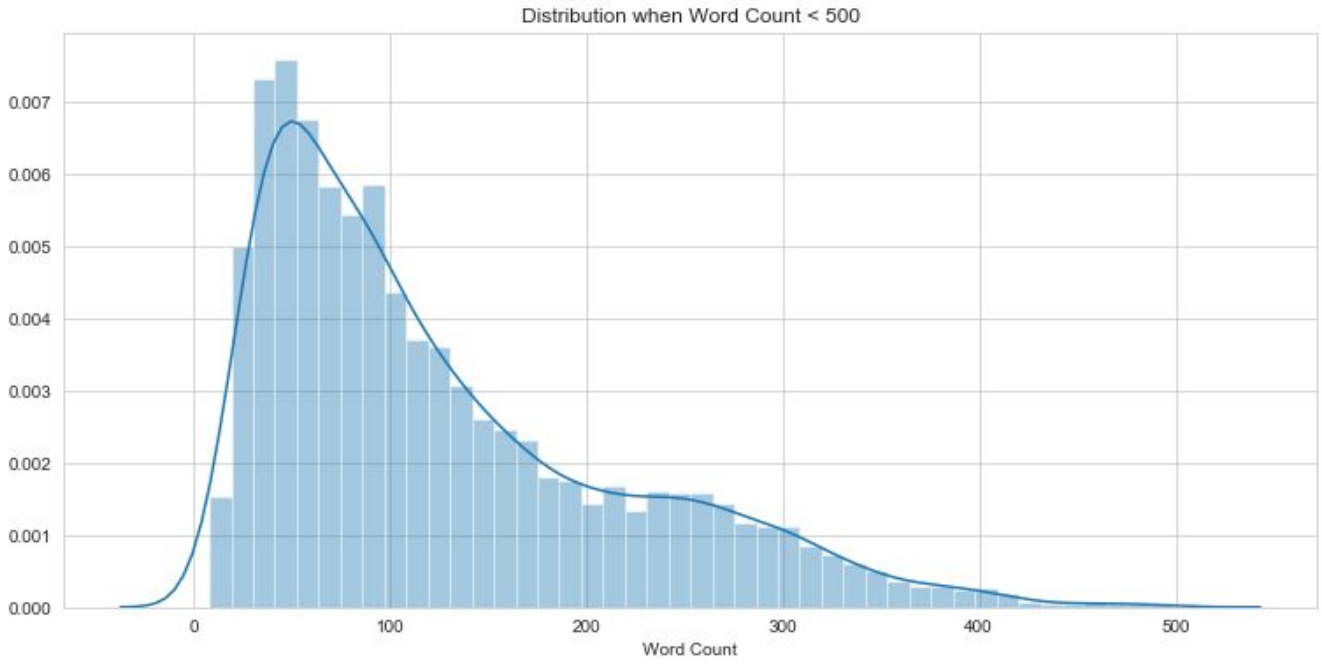


The following data analysis (see below) has been done to investigate the nature of the feature data provided, specifically looking at the length of the articles. This is important because if some topics consistently have much longer articles than the other topics they will be more feature data for those topics and this will need to be accounted for.

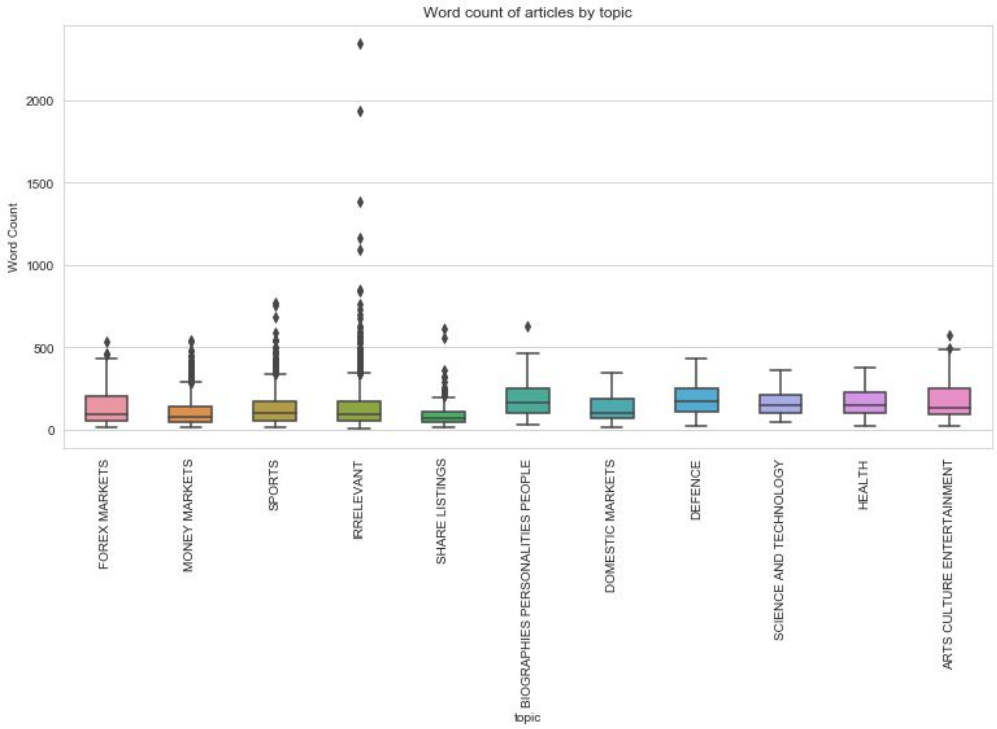
The graphs below look at the distribution of the word counts in the articles. There are some clear outliers for articles with more than 500 words.



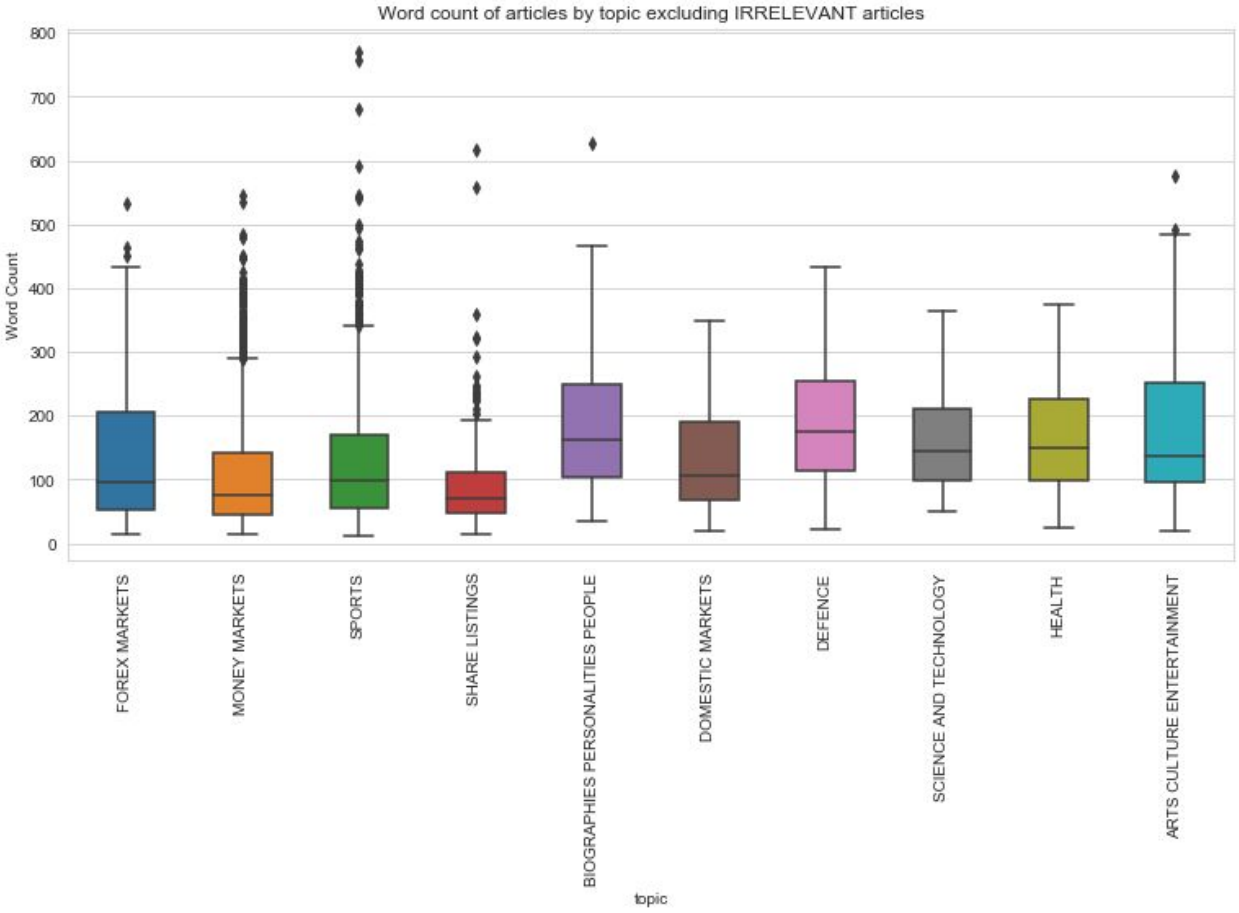
We also looked just at the distribution of word counts excluding these outliers to get a closer look at the majority of the data.



The next visualisations investigate the distribution of article length by topic. The irrelevant category clearly contained all of the outliers in terms of article length. Meaning there was an even more significant portion of the feature data that was irrelevant than we previously thought when we only knew the article count.



Again we removed the data from the irrelevant topic, and with it the outliers, so as to get a closer look at the distribution of article length for the relevant topics. The topics with the largest median article length were, ‘Defence’ and ‘Biographies, Personalities, People’. The ‘Sports’ topic had the largest overall length with several large outliers.



Methodology

The methods we decided to use to classify the articles were the Multinomial Naive Bayes method and the Support Vector Machine (SVM) method. The Naive Bayes method is a common approach used in text classification and is one of the simplest and most practical learning methods to implement. SVMs are well suited for text classification as they handle the high dimensional feature space which is important when working with text and SVMs also avoid overfitting.

Naive Bayes Methodology

Method Selection

Multinomial Naive Bayes is a common, simple and practical method for text classification. An important characteristic of MNB is that it assumes attributes to be statistically independent, this allows posteriors to be easily calculated.

Feature Selection

The first step on development of the MNB model is feature extraction. The first method of feature extraction used was Count Vectorizer. This tokenizes and counts the frequency of word occurrence for each document and builds a vocabulary of known words, thus providing features of length equal to the length of the vocabulary used. A MNB model is then constructed on the training data after it has been vectorized.

Tf-idf Vectorization is another method of feature extraction tested. Some words will be very present (the, as, is etc.) hence carrying very little meaningful information about the actual contents of the document, the tf–idf vectorizer gives weight to words depending on their frequency, thus words very common among all categories of article will be given less influence over the model.

Both vectorizer's have several parameters that may increase the generalisation accuracy of the model. Stop words are words like "and", "the", "him", which are presumed to be uninformative in representing the content of a text. These can be omitted. Unigrams as well as bigrams (groups of 2 words) can also be used.

The method of feature selection chosen through cross validation results for the MNB model used was a Count Vectorizer with stop words omitted, unigrams and bigrams, and a min df - ignore terms that have a document frequency strictly lower than 2.

Hyper-parameter Tuning

There is only one hyperparameter in the MNB model, alpha which is a smoothing parameter. An alpha of 2 was chosen through grid search.

Evaluation Metrics

Metrics of skelarn's accuracy score as well as precision and f1 score through Classification Report. These metrics were performed on cross validated training and development data to

determine the optimal model, before then being used to assess the accuracy of the model on test data.

Design Choices

The training set was divided into training and development sets, 9000 instances in training and 500 instances in the development set, in order to test model parameters. A problem that was encountered in the training data was heavily unbalanced classes. This was addressed by oversampling the classes with fewer instances using imblearn's SMOTE.

Support Vector Machines Methodology

Method Selection

Support Vector Machines are a promising method when it comes to text classification. They are able to handle high dimensional feature spaces given they only utilise a subset of the feature space, the support vectors. Additionally, this will prevent our model from overfitting and allows it to generalise well on unseen data.

The SVM implementation chosen was Sklearn's SGDClassifier which implements linear SVM with stochastic gradient descent. The advantage of using stochastic gradient descent would be an improvement in training efficiency.

Feature Extraction

For feature extraction tuning, several methods were tested:

- CountVectorizer with default parameters
- CountVectorizer with an ngram range between 1 and 3
- CountVectorizer with minimum document frequency of 2
- CountVectorizer with a combination of English stopwords dropped, ngram range between 1 and 3 and minimum document frequency of 2
- TfidfVectorizer with English stopwords dropped

After cross-validation, the scores for the above methods were compared and the Tf-idf vectoriser was chosen. So, the text documents were represented as vectors using Tf-idf vectorisation with English stopwords being dropped due to their lack of predictive capabilities.

Feature Selection

Feature selection was then performed to select the k best performing features according to the Chi-squared metric. This was performed with the aim of improving overall efficiency of the algorithm. After performing cross-validation with the value of k equal to 100, 1000, 2500, 5000, 7500, 10000 and 30000, with the optimal value of k = 5000, it was possible to reduce the number of features from 34724 down to 5000.

Hyper-parameter Tuning
The value of the class weight hyperparameter and the alpha value was tuned using GridSearchCV. The optimum combination found was a balanced class weight selected, and an alpha value of 0.0001

Evaluation Metrics
The best model was selected through GridSearchCV with 10-fold cross validation based on the accuracy score. In training the model, modified Huber was selected as the loss function to determine how well the model performs at each iteration as it searches the parameter space for the point with minimum loss.

Design Choices
To combat imbalanced classes, imblearn's SMOTE sampler was used to artificially inflate the number of samples in the minority classes. Modified Huber was then selected as the loss function as it smooths the loss and reduces sensitivity to outliers.

Results

Results for Method 1- Multinomial Naive Bayes

Best cross-validation score for hyper parameter: 0.74 (Alpha = 2)

Vectorizer	Accuracy Score
TfidfVectorizer()	0.676
CountVectorizer(stop_words="english")	0.736
CountVectorizer(ngram_range=(1,2)	0.754
CountVectorizer(min_df = 2)	0.73
CountVectorizer(stop_words="english", ngram_range=(1,2), min_df = 2)	0.76

Cross Validation Results for Final Model on Training Data (cv = 5) : [0.74105263 0.74157895 0.74052632 0.74368421 0.74894737]
The mean score and the 95% confidence interval of the scores is: 0.74 (+/- 0.01)

Results for Method 2 - SVM

Cross Validation Results on Training Data (cv = 5) : [0.77222222 0.77722222 0.78166667 0.76833333 0.76944444]
The mean score and the 95% confidence interval of the scores is: 0.77 (+/- 0.01)

Selected Method - SVM

The final results for each class calculated on the whole test set with the final select method (SVM) are detailed in the following table.

Topic name	Precision	Recall	F1
ARTS CULTURE ENTERTAINMENT	0.43	0.75	0.55
BIOGRAPHIES PERSONALITIES PEOPLE	0.44	0.5	0.47
DEFENCE	0.64	0.5	0.56
DOMESTIC MARKETS	0.67	0.57	0.62
FOREX MARKETS	0.58	0.43	0.5
HEALTH	0.7	0.78	0.74
IRRELEVANT	0.85	0.94	0.89
MONEY MARKETS	0.66	0.62	0.64
SCIENCE AND TECHNOLOGY	1	0.67	0.8
SHARE LISTINGS	0.86	0.67	0.75
SPORTS	0.98	0.95	0.97

Our final article recommendations using our final selected method of SVM are detailed in the following table.

Topic name	Suggested Articles	Precision	Recall	F1
ARTS CULTURE ENTERTAINMENT	9526, 9604, 9703, 9789, 9830, 9834, 9952	0.43	1.00	0.60
BIOGRAPHIES PERSONALITIES PEOPLE	9695, 9758, 9797, 9854, 9878, 9896, 9933, 9940, 9956, 9988	0.70	0.47	0.56
DEFENCE	9559, 9576, 9607, 9616, 9670, 9713, 9759, 9770, 9773, 9842	0.80	0.62	0.70
DOMESTIC MARKETS	9640, 9750, 9796, 9910, 9989, 9994	0.33	1.00	0.50
FOREX MARKETS	9551, 9588, 9625, 9632, 9671, 9682, 9693, 9772, 9875, 9986	0.30	0.06	0.10
HEALTH	9609, 9661, 9807, 9833, 9873, 9887, 9911, 9926, 9937, 9978	0.70	0.50	0.58
IRRELEVANT	9642, 9651, 9674, 9690, 9716, 9719, 9785, 9913, 9914, 9957	1.00	0.04	0.07

MONEY MARKETS	9516, 9618, 9672, 9755, 9761, 9765, 9766, 9769, 9871, 9998	0.80	0.12	0.20
SCIENCE AND TECHNOLOGY	9617, 9735, 9982	0.00	0.00	0.00
SHARE LISTINGS	9518, 9601, 9666, 9667, 9668, 9715, 9864, 9972, 9999	0.56	0.71	0.63
SPORTS	9574, 9656, 9663, 9752, 9754, 9774, 9858, 9886, 9942, 9997	0.80	0.13	0.23

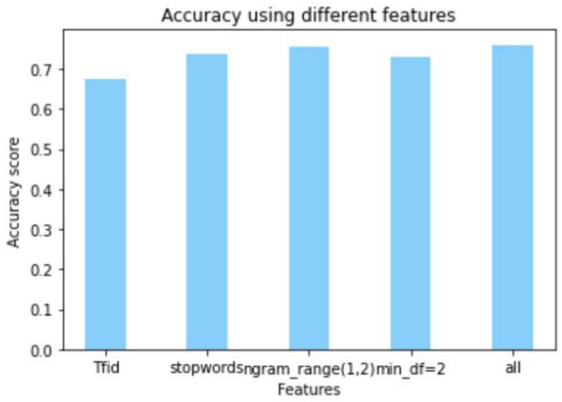
Discussion

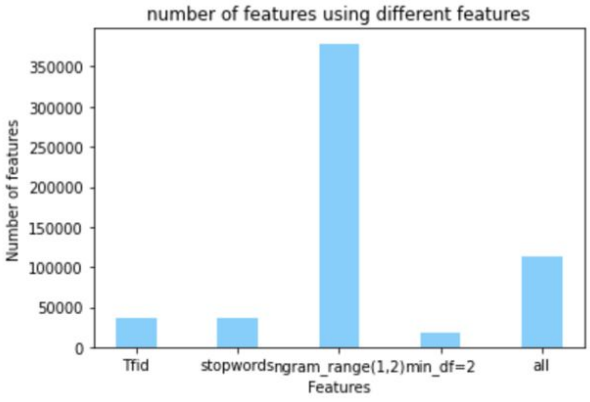
In comparing the two models we found that the Support Vector Machine model had better performance than the Multinomial Naive Bayes so we chose this for our final prediction. We go on to give a more thorough comparison in this discussion.

Firstly, we made some rough predictions to evaluate the models. The MNB and SVM had similar base model accuracy scores on the test data.

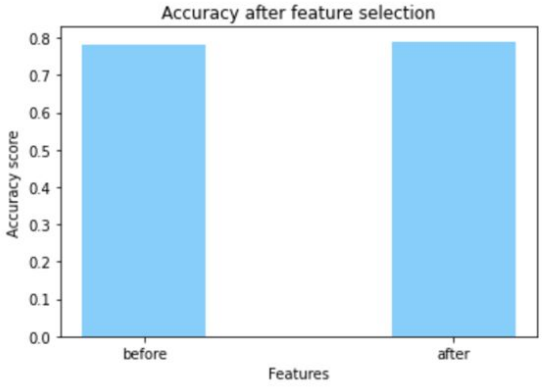
```
MultinomialNB: Training set Accuracy score for base model: 0.8287777777777777
MultinomialNB: Test set Accuracy score for base model: 0.732
SVM: Training Accuracy score for base model: 0.9697777777777777
SVM: Test set Accuracy score for base model: 0.728
```

Furthemore, we compared the accuracy of the models using different features and the number of these features. See below the graphs of the scores and the number of features for the MNB model. This is the MNB using CountVectorizer and set stop_words="english", ngram_range=(1,2) , min_df = 2.

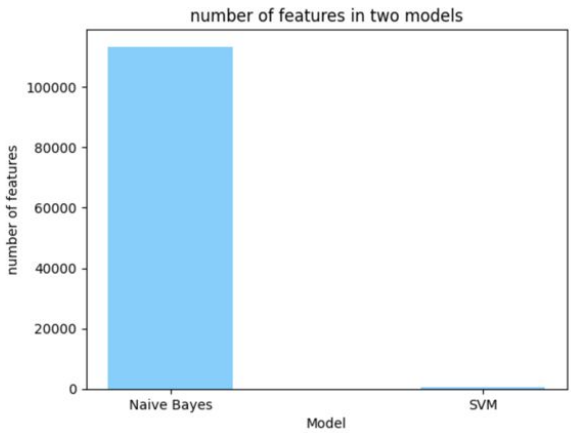


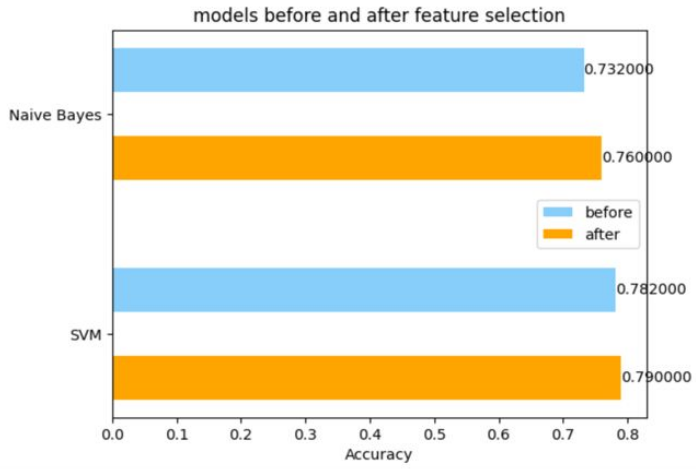


In the following graphs we visualise the accuracy scores and number of features for the SVM model. We choose TfidfVectorizer(set stop_words = 'english') and use SelectKBest(set k to 512), the figure shows the accuracy after feature selection and Gridseatch. The graphs below show how the accuracy is maintained before and after feature selection and the number of features is greatly reduced.

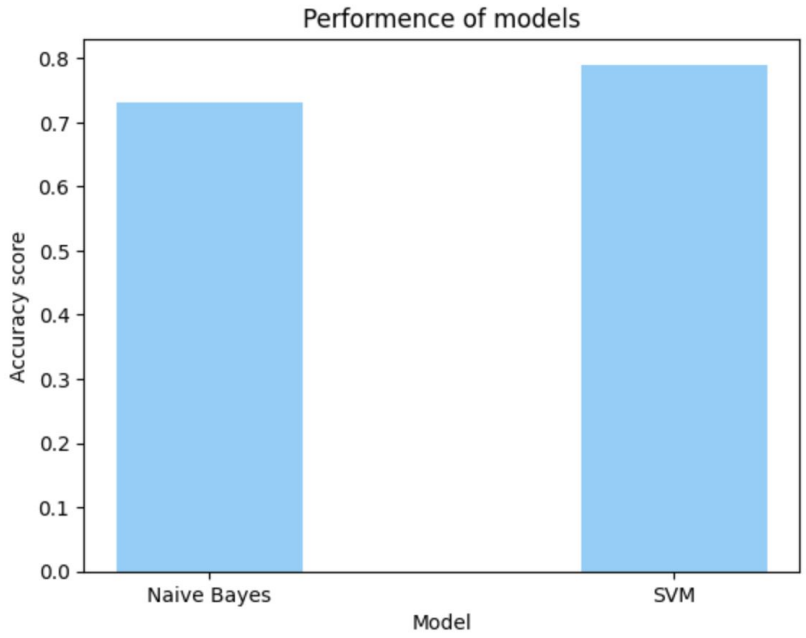


In comparing the two models, we can see from the figures that after feature extraction and selection, SVM has the better accuracy score. Although both accuracy scores increase slightly. Additionally the SVM model has much less features than the Naive Bayes model. These comparisons of the number of features and the accuracy of the models before and after feature selection are visualised in the following graphs.





After feature extraction, selection, hyperparameter tuning, data sampling, we obtain the final scores of these two models. The SVM has the better performance so we chose this model for our final prediction.



In regards to the metrics in the above two tables in the results section the F1 score is more appropriate because it uses harmonic average which tends to favor smaller values when increasing. It also penalizes extreme cases where there is a large difference between the precision rate and the recall rate, well balancing the precision rate and the recall rate.

If we were to continue this project there would be several ways we could improve. There are many models such as KNeightborsClassifier, RandomForestClassifier, GaussianNB, LogisticRegression, BernoulliNB DecisionTreeClassifier, AdaBosstClassifierand, etc. that we could investigate to see if they give better performance on this text classification problem. Specifically we briefly looked into the RandomForestClassifier (RF) and LogisticRegression (LF) and discovered they give higher accuracy for the base models (without feature selection and Grid Search).

RF: Training Accuracy score for base model: 0.9896666666666667
RF: Test set Accuracy score for base model: 0.766
LR: Training Accuracy score for base model: 0.978
LR: Test set Accuracy score for base model: 0.768

Conclusion

The aim of this project was to classify and recommend articles based on their content and how it was relevant to several topics. First of all we discovered that an important part of machine learning is to perform exploratory data analysis to gain an understanding of what the data looks like and specifically to see whether the data is balanced. Seeing whether the data is balanced allowed us to determine if we needed to undersample or oversample. We implemented two methods, Multinomial Naive Bayes and SVM. We tried many ways to improve the models, from feature selection and parameters tuning to evaluation. From this feature engineering, extracting features to represent the text we learnt that obtaining suitable features are crucial and directly related to the final result. When doing model and parameter selection we selected the model by testing the score performance of each model and employed the Grid Search and cross-validation to tune and select the best parameter combination. Finally, after completing all of the above steps, we needed to test again and again on each step to improve, and try more new methods on each step to find more useful functions or ideas to improve performance. This project enhanced our ability and understanding of machine learning techniques and how to develop them. There are still many different methods we could explore and implement to try and improve the article classification.

Reference

Susan Li. 'Multi-Class Text Classification Model Comparison and Selection'. **Towards Data Science.** Sept 25, 2018. 05/04/2020. <https://towardsdatascience.com/multi-class-text-classification-model-comparison-and-selection-5eb066197568 >

Miguel Fernandez Zafra. 'Text Classification in Python'. **Towards Data Science.** Jun 16, 2019. 05/04/2020. <https://towardsdatascience.com/text-classification-in-python-dd95d264c802>

Andreas C. Müller & Sarah Guido, "Introduction to Machine Learning with Python", Pages 323-356, 10/04/2020.

Code Submission

See submitted file.