

## 1 Methods

In this section we describe which methods and classifiers we used. Including features and different possible parameters.

### 1.1 Classifiers

We have chosen 3 different classifiers and those are:

1. Naïve Bayes
2. SGD (Stochastic Gradient Descent)
3. SVM (Support Vector Machine)

The Naïve Bayes Classifier looks at the characteristics of the object/topic and tries to classify it calculating the probability of it belonging to each category and then choosing the category with the highest probability.

The SGD Classifier looks only at small random subsets of the data instead of looking at the whole data set and then it tries to iterate with small steps towards a solution by updating the parameters for the classification.

The SVM classifier in simple terms looks for best boundary to separate different categories and applies the resultant classification on the data set.

### 1.2 Features

The investigated features are:

1. counts
2.  $tf$  (term frequency)
3.  $tf-idf$  (term frequency - inverse document frequency)

Feature counts is counting how many times each term appears in the document.

Feature  $tf$  works as relative occurrence of the given word in the document.

Feature  $tf-idf$  adds on  $tf$  considering also in how many documents of the collection the given term appears.

## 2 Results

In this section we describe the results obtained through the code implemented in the attached Jupyter notebook.

### 2.1 Classifiers and classifiers with features

Below table (Table 1) shows individual accuracies for each method. We can see that the SVM model with tf-idf feature has achieved the best performance.

Table 1: Classifier performance with different feature types

Classifier	Feature Type	Accuracy	Precision	Recall	F1 Score
Naive Bayes	Counts	0.77283	0.76166	0.77283	0.75111
	TF	0.70525	0.78548	0.70525	0.69205
	TF-IDF	0.77389	0.82187	0.77389	0.76844
SGD	Counts	0.75212	0.763514	0.75212	0.75239
	TF	0.76978	0.77852	0.76978	0.76299
	TF-IDF	0.82488	0.82803	0.82488	0.81907
SVM	Counts	0.73366	0.73823	0.73366	0.73359
	TF	0.75916	0.76476	0.75916	0.75800
	TF-IDF	0.83470	0.83960	0.83470	0.83451

### 2.2 CountVectorizer and different parameters

Parameter	Accuracy	Precision	Recall	F1 Score
Lowercasing (True)	0.83470	0.83470	0.83960	0.83451
Lowercasing (False)	0.82912	0.82912	0.83550	0.82946
Stop Words (None)	0.83470	0.83470	0.83960	0.83451
Stop Words (English)	0.83497	0.83497	0.84049	0.83498
Analyzer (Word, 1-1)	0.83470	0.83470	0.83960	0.83451
Analyzer (Word, 1-2)	0.83855	0.83855	0.84529	0.83878
Analyzer (Word, 2-2)	0.77190	0.77190	0.78960	0.77506
Analyzer (Word, 2-3)	0.75916	0.75916	0.78059	0.76270
Analyzer (Word, 3-3)	0.67883	0.67883	0.72925	0.69005
Analyzer (Char, 1-1)	0.24575	0.24575	0.25456	0.22320
Analyzer (Char, 1-2)	0.48393	0.48393	0.51886	0.47751
Analyzer (Char, 2-2)	0.61776	0.61776	0.63033	0.61749
Analyzer (Char, 2-3)	0.72384	0.72384	0.74147	0.72648
Analyzer (Char, 3-3)	0.73924	0.73924	0.75759	0.757595
Analyzer (Char_wb, 1-1)	0.19636	0.19636	0.20399	0.16119
Analyzer (Char_wb, 1-2)	0.37413	0.37413	0.42468	0.36485
Analyzer (Char_wb, 2-2)	0.61311	0.61311	0.62524	0.61264
Analyzer (Char_wb, 2-3)	0.72039	0.72039	0.73812	0.72313
Analyzer (Char_wb, 3-3)	0.73685	0.73685	0.75612	0.74042
Max Features (10,000)	0.81226	0.81226	0.81790	0.81261
Max Features (200,000)	0.83470	0.83470	0.83960	0.83451
Max Features (5,000,000)	0.83470	0.83470	0.83960	0.83451

Table 2: ContVectorizer with different parameters

## 3 Discussion

In this section we will discuss the result from Table 2.

### 3.1 Lowercasing

When enabling `lowercasing=True` we can notice a small difference across all metrics. The results suggest that using this parameter the model can achieve increased performance.

### 3.2 Stopwords

Performance increases when using Stop Words (English) perhaps indicating that in this context it is useful to use stop words.

### 3.3 Analyzer with ngrams range

When trying out different different combinations we can see a few interesting facts. When using ngram range of (1,2) we can see slight improvement across all metrics indicating that this is one of the most beneficial parameter. When ngram range increases (2,2) to (3,3) we can observe significant dip across metrics.

First few analyzer=`char` perform significantly worse indicating it's not worth using them. We see much better performance at higher ngram range indicating possible better performance as ngram increases. Same thing occurs with `char` `wb` across whole range. As range increases the metrics improve as well.

### 3.4 Conclusion

As curious as we are we also tried to combine all the best performing parameters together and we did not see significant improvement but neither significant drop in performance either (visible looking at the last cell of the attached Jupyter Notebook).

In first iteration of task 4 we also tried to combine all parameters to make a table of all possible combinations but the highest performing one achieves around 0.6 for the accuracy level.

In conclusion the best performing classifier is SVM with `tf-idf` feature.

Slightly above in accuracy and precision metric is SVM with `tf-idf` and stop word parameter set to `'english'`.

For Recall and F1 score the best performing classifier was Analyzer with word and ngram range of 1-2.