Text Categorization
Group 5

Text Mining - Assignment 1

Ivan Horokhovskyi (s3069176), Andreas Savva (s3316491)

1 Introduction

In this assignment, we were asked to follow the tutorial[1] from scikit-learn about working with text data. In the tutorial, a subset of the dataset Twenty Newsgroups[2] is used and the procedure of extracting features, training and optimizing a classifier is explained. In our assignment, we used the entire dataset to compare three classifiers with three types of features: word counts, term frequencies and term-frequency-inverse-document-frequency. We compared the Naive Bayes classifier, SGDC and K-NN. Furthermore, we had to test different parameters affecting the vectorizer.

2 Features

In order to be able to categorize text, our classifiers need features they can work with. For this reason, we compared each classifier with 3 distinct features.

2.1 Counts

Counts are the simplest of the three features,counting the observations of each word in a document. Counts are not a reliable method since a document containing 5 times more a word than a different one, doesn't make it more relevant to the category.

2.2 Term Frequencies (TF)

With this method, we do not use raw term counts in a document but rather weight them using log scale since relevance doesn't increase proportionally.

$$tf_{t,d} = \begin{cases} 1 + \log_{10} tc_{td}, & \text{if } tc_{td} > 0 \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

2.3 Term Frequency - Inverse Document Frequency (TF-IDF)

Because rare items are usually more informative than more frequent ones, we want to reward them. This is the reason we combine Term frequency with Inverse document frequency.

$$idf_t = \log_{10} \frac{N}{df_t} \quad (2) \quad tf\_idf = tf_{t,d} \times idf_t \quad (3)$$

3 Classifiers

3.1 K-NN

The k-nearest neighbour algorithm is based on the fact that similar items will be closer in space and it is a non-parametric algorithm. The k variable sets how many neighbours it uses to classify an item. As the heuristics for K-NN we are using Minkowski distance. Higher K value is computationally more expensive and results in more smoothing while small K is impacted by noise more.

3.2 Naive Bayes

This classifier is based on probability theory and uses the prior probabilty for every category. In the end, a posterior probability distribution is created over the possible categories for a given item. $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$

3.3 SGDC

SGDC uses a linear classifier in combination with stochastic gradient decent for optimization of a convex function. In our case, it uses a Support Vector Machine network.

4 Experiments

4.1 Experiment 1 - Default Parameters

In the first experiment, we test all the classifiers in their default setting combined with all three types of features.

As we can see from the Table [1], the K-NN classifier performs poorly when compared to Naive

Classifier	Features	Mean Accuracy	Precision	Recall	F1
K-NN	Counts	0.352	0.421	0.352	0.357
	TF	0.408	0.489	0.408	0.417
	TF-IDF	0.659	0.674	0.659	0.66
Naive Bayes	Counts	0.773	0.762	0.773	0.751
	TF	0.705	0.785	0.705	0.692
	TF-IDF	0.774	0.822	0.774	0.768
SGDC	Counts	0.742	0.767	0.742	0.748
	TF	0.811	0.811	0.811	0.807
	TF-IDF	0.853	0.854	0.853	0.852

Table 1: Test results for each classifier with 3 different feature types and default settings.

Bayes and SGDC classifiers as expected. In addition we can see that in all cases the TF-IDF features give better results in all metrics.

4.2 Experiment 2 - Hyperparameter Optimization

In this experiment, we set out to optimize each of the classifiers by tuning their hyperparameters. This was done on all three available features in order to be able to determine which features perform best in the best case scenario where the classifier performs well. Below are the parameters we tested for each classifier:

- K-NN number of neighbours: 5[default], 11, 21, $\sqrt{doc\_size}$, $\sqrt{doc\_size}/2$.
- Naive Bayes & SGDC alpha: 0.0001[default, 0.001, 0.01, 0.1, 1.

From Table[2] we can see that for K-NN, 5 neighbours give the best performance even though we tried using some of the heuristics for setting the neighbourhood size. We see a small improvement in mean accuracy for Naive Bayes when using higher smoothing parameter alpha but this gives a big improvement in all other metrics. Lastly, using higher alpha value for SGDC only helps when the used features is counts whereas all other metrics perform better with a lower smoothing value. This is likely because the other features are much more informative.

4.3 Experiment 3 - Vectorizer Parameters

In our last experiment, we tested different parameters that affect the vectorizer and therefore the features. Below is the list with the parameters we have tried.

- lowercase: True[def.], False
- stop_words: None[def.], 'english'
- Word n-gram ranges: (1, 1)[def.], (1, 2), (2, 2)
- Char. n-gram ranges: (2, 2), (3, 3), (4, 4)
- max_features: 1000, 5000, 10000

The lowercase setting affects whether all words will be lowercased. Using stop words, means that we have a set of words that are very frequent and give no information and exclude them from our features. The analyzer in word setting, uses words in range as features where as char uses characters in range. Lastly, the max features limits the amount of features. For this experiment we use the best hyperparameters found with TF-IDF.

Tables [3] [4] [5] give the results for K-NN, Naive Bayes and SGDC. The first two classifiers prefer lowercasing but not achieve better testing results. Stopwords are prefered for K-NN and SGDC but only the first achieves marginally better test results. SGDC prefers word unigrams and bigrams giving better test results. All three classifiers for character ngrams choose range 4 but results are worse. Lastly limiting the features to 10000 gives worse results on all 3 classifiers

Classifier	Features	Best Parameters	Mean Accuracy	Precision	Recall	F1
K-NN	Counts	Number of Neighbors: 5	0.352	0.421	0.352	0.357
	TF		0.408	0.489	0.408	0.417
	TF-IDF		0.659	0.674	0.659	0.66
Naive Bayes	Counts	Alpha: 0.001	0.797	0.804	0.797	0.785
	TF		0.833	0.834	0.833	0.831
	TF-IDF		0.835	0.836	0.835	0.834
SGDC	Counts	Alpha: 0.01	0.816	0.817	0.816	0.814
	TF	Alpha: 0.0001	0.81	0.812	0.81	0.807
	TF-IDF		0.853	0.854	0.853	0.852

Table 2: The results when evaluating on the test set of the Twenty Newsgroups dataset, of the classifiers using the best hyperparameters found by grid search.

Vect. Params	Mean Acc.	Precision	Recall	F1
Lower.: False	0.657	0.671	0.657	0.657
Stop Words:'english'	0.676	0.685	0.676	0.676
Analyzer:'word', Ngrams:(1,1)	0.659	0.674	0.659	0.66
Analyzer':'char', Ngrams:(4,4)	0.559	0.59	0.559	0.565
Max Feat:10000	0.638	0.653	0.638	0.64

Table 3: K-NN classifier best test results for vectorizer parameters. Uses the best found hyperparameters with TF-IDF.

Vect. Params	Mean Acc.	Precision	Recall	F1
Lower.: False	0.837	0.838	0.837	0.835
Stop Words: None	0.835	0.836	0.835	0.834
Analyzer: 'word', Ngrams:(1,1)	0.834	0.837	0.834	0.833
Analyzer: 'char', Ngrams:(4,4)	0.805	0.809	0.805	0.802
Max Feat.: 10000	0.811	0.812	0.811	0.81

Table 4: Naive Bayes classifier best results on the test set of the Twenty Newsgroups with multiple vectorizer parameters when using the best found hyperparameters with TF-IDF.

5 Conclusion

We experimented with three different classifiers and three different types of features and as expected TF-IDF is the best of three. K-NN performs poorly for this type of classification probably due to the "curse of dimensionality". Naive Bayes has very good results but SGDC gives the best results. Hyperparameter optimization makes a big difference for the classifiers. Laslty, limiting the available features gives worse results as expected.

Vect. Params	Mean Acc.	Precision	Recall	F1
Lowercase: True	0.854	0.854	0.854	0.852
Stop Words: 'english'	0.851	0.851	0.851	0.849
Analyzer: 'word', Ngrams': (1, 2)	0.859	0.86	0.859	0.858
Analyzer: 'char', Ngrams: (4, 4)	0.845	0.845	0.845	0.843
Max Feat.: 10000	0.828	0.828	0.828	0.826

Table 5: SGDC classifier best results on the test set of the Twenty Newsgroups with multiple vectorizer parameters when using the best found hyperparameters with TF-IDF.

References

[1] Scikit-Learn Working with Text Data. https://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html.

[2] Twenty Newsgroups. http://qwone.com/~jason/20Newsgroups/.