Leiden University

FACULTY OF SCIENCE

Text Mining

Assignment 1

Student Names

1. Giovanni Lunardi - s3923207
2. Karthik Narayanan Ravindran - s3934594

1 Introduction

In this assignment we will compare three different classifiers by performing a text categorization task with benchmark data using scikit-learn. The bench marking will be firstly done using the same pipeline, then multiple vectorizer, transformer and classifiers will be applied to the pipeline. The dataset is a collection of news splitted in 20 categories called Twenty Newsgroup.

2 Choosing the best classifier with the best features

The classifiers chosen for this task are:

- MultinomialNB: Naive Bayes Classifier for multimonomial models.
- SGDClassifier: this estimator implement the Stochastic Gradient Descent learning algorithm.
- RidgeClassifier: classifier implemention using ridge regression learning algorithm.

Initially we compared these 3 classifiers using the same pipeline: a CountVectorizer, tf-idf transformer and classifier. It comes out that the SGD and the Ridge classifier were able to achieve higher values than the Naive Bayes one. In fact, the metrics of the two classifier showed in the table are almost identical and above the 80%.

Classifier	Precision	Recall	F1-score	Accuracy
MultinomialNB	0.82	0.77	0.77	0.77
SGDClassifier	0.85	0.85	0.85	0.853
RidgeClassifier	0.86	0.85	0.85	0.852

Table 1: Average evaluation of precision, recall, f1-score, and accuracy for each classifier.

Classifier	Vectorizer	Precision	Recall	F1-score	Accuracy
MultinomialNB	**CountVectorizer**	0.83	0.76	0.76	0.77
	TfidfVectorizer	0.83	0.81	0.81	0.81
	TfVectorizer	0.79	0.71	0.69	0.70
SGDClassifier	**CountVectorizer**	0.85	0.85	0.85	0.853
	TfidfVectorizer	0.86	0.86	0.86	0.857
	TfVectorizer	0.86	0.85	0.85	0.855
RidgeClassifier	**CountVectorizer**	0.83	0.81	0.81	0.81
	TfidfVectorizer	0.86	0.85	0.85	0.855
	TfVectorizer	0.83	0.83	0.83	0.82

Table 2: Average evaluation of precision, recall, and f1-score for each classifier using different vectorization and transformation techniques. The results display that the best metrics are reached by the SGD classifier using tf-idf vectorization and tf-idf transformation techniques.

After that, we built different pipelines to compare the difference between each combination of classifier-feature using first different vectorizers to tokenize our data, then we used a tf-idf transformer to transform the latter data to a more useful representation and to scale down the impact of the tokens that appear very frequently and may are less informative. We also tried to use just the term-frequency and not the inverse document frequency to vectorize and transform our data.

3 Comparing parameters

In this section, we experiment with different parameters by making use of the GridSearchCV function and try to find the best tuned values. We chose randomizedsearchcv instead of gridsearchcv because it works best on large datasets as they tend to yield the best hyperparameters by only analyzing a few random parameter combinations. This also tends to decrease computational cost.

Table 3: Best Model Parameters and Accuracy

Table 4: When maxfeatures=10000,20000

Hyperparameter	**Value**
clf__alpha	0.001
clf__penalty	l2
vect__analyzer	word
vect__lowercase	True
vect__max_features	20,000
vect__ngram_range	(1, 2)
vect__stop_words	English
Accuracy (Validation)	0.887
Accuracy (Test)	0.823

Table 5: When maxfeatures=10,20,100

Hyperparameter	**Value**
clf__alpha	0.1
clf__penalty	l2
vect__analyzer	word
vect__lowercase	False
vect__max_features	50
vect__ngram_range	(1, 1)
vect__stop_words	None
Accuracy (Validation)	0.200
Accuracy (Test)	0.180

As we can observe from the results, the parameters which give the highest accuracy rate are found and displayed. Analyzing the result tends to reveal plenty of information about the nature of the given dataset. For instance, the best value for the 'clf-penalty' parameter is found to be l2 regularization, which mostly implies that the given dataset is linear. Furthermore, the optimal weight of the regularization (clf-alpha) is outputted as 0.001, the lowest value amongst the three, indicating that the data is more susceptible to underfitting than overfitting. In accordance to this, the optimal 'max-features' value is displayed as 20000, the largest of the meaning, implying the vast number of words present in the dataset. Excess verbiage is removed from the dataset if the vect-stopwords parameter is defined as 'english'. The results show that the word-type tokenization is optimal for this assignment because we're focused on classifying each and every individual word in accordance with their semantic meaning. Finally the optimal vect-lowercase parameter being 'True' implies that accuracy tends to increase when all letters are converted to lowercase, thereby avoiding the risk of case sensitivity.

4 Results

In order to classify the classifiers we need to take into account the precision and the recall because they allow us to understand how many relevant words were classified correctly among those classified and how many relevant words were classified correctly among all relevant words.

We can see that using the same pipeline we obtain similar behaviours. Is different, however, if we use different vectorizer and transformer. In fact, the Naive Bayes classifier has lower precision and recall than the other classifiers and this is true using all 3 different vectorizers. The SGD classifier and the Ridge classifier combined with the tf-idf vectorizer achieved the best results, however we chose the SGD classifier because is best suitable for this task thanks to its efficiency and speed.

Using the SGD classifier in combination with tf-idf vectorizer, a randomized search is performed to find the best parameters that yield the highest accuracy. As the values in the maxfeatures parameter increases, the accuracy also tends to increase, implying the vastness of the dataset and the high risk of underfitting. The resulting accuracy for the validation set using optimal parameters is found to be 0.887. Similarly, the recall and precision values which are used to identify positive instances and false positives of the declared categories respectively are also found to be sufficiently high (both at 0.86) when utilizing the optimal parameters.