Assignment Report

Konstantina Antonopoulou

April 5, 2022

Assignment Overview

The current project is concerned with a text classification task which employs machine learning algorithms. Text classification is the task of assigning an appropriate category/topic to a document. It is a supervised machine learning problem which means that first, an algorithm is trained and then is tested on new, unseen data. Below, there are given the steps I had to follow:

1. Collect 100 Greek news articles texts as text files which are saved into three different folders depending on their category (financial, sports, political). This is going to be the dataset.
2. Preprocess the text corpus so that there are no capital letters, numericals, punctuation.
3. Transform each text into a numerical representation and extract important features using TF-IDF.
4. Split into train and test sets (80% : 20%).
5. Evaluate the model by using evaluation metrics.

Tasks description

The project was conducted with the use of Python and Jupyter Notebook. First, I import all the necessary libraries in the Jupyter Notebook.

1. Text collection

I collect 100 Greek news articles texts from news websites. I manually import each text to a txt file. In our system, there is a folder named "raw data" which contains three folders. In the first folder, there are 30 text files of financial content. In the second folder, there 40 text files about sports. In the third file, there are 30 text files about politics. The reason that there are 40 sports texts is because they were smaller in size compared to the political and financial texts and I wanted to ensure that the data are as balanced as possible.

2. Text preprocessing

Text preprocessing is a task which is necessary when we want to transform our data into an analyzable and consistent form in order to be able to conduct machine learning tasks and applications. I have created a function which includes all of the preprocessing tasks:

1. Tokenize the texts into words using the Natural Language Toolkit library.
2. Convert the letters from uppercase to lowercase.
3. Remove numericals.
4. Remove punctuation symbols using again the Natural Language Toolkit.
5. Remove any English characters using Regex.

I import the 30 financial texts and I create a data frame. The rows contain the texts and there is one column with the label of the texts. In this way, I am able to apply the function to all texts at once. I repeat this three times as there are three different folders with raw texts that I have to import and preprocess. There is another folder in our system named "preprocessed texts" which contains three folders each corresponding to the three categories of texts. I create text files with the preprocessed texts and I import them to the folders. The first one contains the preprocessed financial texts, the second one the preprocessed sports texts and the third one the preprocessed political texts.

In the Jupyter Notebook, there are three data frames I created before in order to preprocess the texts. I concatenate all three data frames into one. Now, there is a data frame containing all of the 100 preprocessed texts. This data frame is imported into an excel file which contains information about the texts such as index, id and the topic of each text.

3. Vectorization

The next task is to convert the texts into numerical representations, so that the algorithm can understand them. This is called vectorization. I use TF-IDF from the scikit-learn library. The TF-IDF assigns the weights of importance of each word in the corpus. It measures the term frequency in each document and then, the inverse document frequency, which is how common/rare a term is in the entire corpus. At the same time, I can get the names of the features, that is the words that are used as the features for text classification. I also make sure that the labels of the texts are converted into integers as well.

4. Split into train and test set

I split the data into a train and a test set. The train set is the 80% and the test set is the remaining 20%. The 80% of the dataset is used to train the algorithm and the 20% will be used in the end to test the algorithm. I am going to use Support Vector Machines which is capable of performing classification.

5. Evaluation metrics

In this step, I use the 20% of the data to test the model, the SVM classifier. The score shows the model's accuracy. Except for that, I have to check how well our model performed in terms of precision, recall and f-score. Precision shows how well the algorithm did at labelling the texts correctly. Recall checks how many of the texts predicted as having a specific label, had actually that label/topic. F-score considers both precision and recall. I also create a confusion matrix, so as to inspect the actual true positives and true negatives and gain a better view of the results.