Mini Project 2 – Question Classification

Catarina Sousa 93695, Nelson Trindade 93743

November 10, 2021

1 Models

Our first model is a machine learning algorithm (Naïve Bayes). We start by parsing the input files to have labels and questions separated. We then pre-process the questions by *tokenizing*, *lowing* the words, *stemming*, and *lemmatizing*. At the end of our pre-process, we have a list with all the sentences. After processing, we use *Count Vectorizer* to *fit_transform* our train set so that our model learns the mean and variance of the features of our set. We then transform our test set so that the model uses the same mean and variance that was calculated from the train set to transform the test set. We then use a *Multinomial* Naïve Bayes distribution since it is suitable for the classification of discrete features.

Our second model is a SVM[1], with TF-IDF[2]. We parse the train and dev files just as we do in our first model, but then our pre-process is just a *tokenizer*. We use the *Count Vectorizer* to *fit_transform*[3] our train set and *transform*[4] our test set. After this, we use *tfidfTransformer*[5] to *fit_transform* the training matrix and transform the testing matrix obtained by the *Count Vectorizer*. These functions transform the count matrix into a normalized *tf-idf* representation. We then use a Support Vector Machine to predict the test labels.

We started by doing small steps each time to be sure that what was done was correct. When we created our parser function, we tested it. When we created our processing function, we tested it. We did this technique to catch the errors as soon as possible. After having these functions functional, we started testing different approaches. In the Naïve Bayes approach, we tested different distributions until we concluded that the *Multinomial* was the best. We tested with different pre-processing techniques until we reached the best one with this model (86.8%).

In the Support Vector Machine approach, we also tested with different pre-processing techniques until we realized that using only the *tokenizer* was the best approach. We first didn't use the *tfidf-Transformer* functions and obtained an accuracy of 84%. We started thinking about how we could improve our model and we decided that we wanted to use *tf-idf*, since it reflects how important a word is in a corpus. We concluded that with *tf-idf* we could get better results and that was what happened, we got 88.4%.

The *tokenizer*, the *stemmer* and the *lemmatizer* that we use are from the *nltk* library, while the *MultinomialNB*, *Count Vectorizer* and *tfidfTransformer* are from the sklearn library.

To get the accuracy from our models, we compared the predicted labels with the correct labels using the *sklearn.metrics.accuracy_score()*.

[1] Support Vector Machine approach

[2] Term Frequency–Inverse Document Frequency

[3] Learn the vocabulary dictionary and return document-term matrix

[4] Transform the document to document-term matrix

[5] Transform a count matrix to a normalized tf-idf representation

2 Results

MODEL/LABEL	GEOGRAPHY	MUSIC	LITERATURE	HISTORY	SCIENCE	GENERAL
Naïve Bayes	42/50	42/50	42/50	42/50	42/50	86.8%
SVM + TF-IDF	42/50	42/50	42/50	42/50	42/50	88.4%

Table 1: Results obtained in the tests performed.

3 Error analysis

In the first approach of SVM we had more pre-process, as we mentioned before. This led to a lower accuracy because some words were wrongly stemmed and lemmatized. For example, the stemmer would convert the word "upcoming" to the word "upcom", which is incorrect. An example of a wrong lemmatization is the word "done". If we don't specify the part of speech, we get the word "done" instead of the word "do".

We continued to analyze the errors and realized that our model didn't consider the importance of a word, as we mentioned before. For example, our model failed to predict the label for the sentence "The Hubble Space Telescope was deployed by this space shuttle; 3 years later the Endeavour fixed its mirror — Discovery". Our model predicted the label "History" instead of the label "Science". This occurred because our model didn't realize that the words "Hubble", "space" and "telescope" were the most important ones to predict the correct label. Using the *tfidfTransformer*, our model started to predict it correctly.

By analyzing the errors, we realized that our model failed some sentences because some words in it appeared more than once in the predicted label (incorrect one). For example, in the sentence "The Babylonians kept abreast of the times using a form of this instrument seen here: Sundial" our model predicted the label History due to the word "Babylonians". The word "Babylonian" appears in 6 sentences in the train set and 5 of them correspond to the label "History". The answer to this sentence is very important to predict the right label, "Science". The problem here is that the word "Sundial" never appears in the train set, so our model cannot predict that this sentence is related to "Science".

In our Naïve Bayes approach, we started with the *Gaussian* technique, but we didn't get the best accuracy. This result is understandable, since this distribution is appropriate for continuous data. By analyzing the available distributions, we realized that the *Multinomial* one was the best, since it is appropriate for discrete features.

4 Future Work

If we had more time to conclude our project, we would like to improve the pre-processing by implementing part-of-speech tagging, which would improve lemmatization's results.

Although we aren't allowed to use neural networks, if we had more time, we would like to test it to see which accuracy we would obtain.

5 Bibliography

- Fenix
- and like this.