CMPE462 - Project Step3

Team BigBrains
Mahir Efe Kaya - Yahya Bedirhan Pak

Previous Work

In the step 2 of the project, we used Naive Bayes method with different vectorization method. We hadn't used an external library while getting these results and it was a good opportunity for us to understand the fundementals of text processing and implementinga learning algorithm. These were our results for the validation data:

```
Macro avg accuracy:      0.624
Macro avg precision:     0.6149441617189709
Macro avg recall:        0.624
```

New Methods

In this step, we focused on trying different machine learning algorithms by using the help of external libraries, mainly **scikit-learn**.

We tried using different classification algorithms such as:

- Support Vector Machine
- Logistic Regression
- Extra Trees Classifier
- Random Forest Classifier
- Gradient Boosting Classifier

Document Vectorization

In the previos step, we implemented document vectorization by hand with experimenting different techniques by taking word count and frequency into account. In this step we tried to use TFIDF vectorizer from the scikit-learn library. In addition to that we used pre-trained models to vectorize our documents such as:

- Universal Sentence Encoder
- BERT
- Glove

Results

Universal Sentence Encoder

We used universal sentence encoder to vectorize our documents and then tried support vector machine, logistic regression, gradient boost, random forest, extra trees learning methods. These are the results:

Support Vector Machine (Linear Kernel)

```
Macro avg accuracy:     0.6546666666666666
Macro avg precision:    0.6543454588456558
Macro avg recall:       0.6546666666666666
```

Logistic Regression

```
Macro avg accuracy:     0.628
Macro avg precision:    0.6285524111436726
Macro avg recall:       0.628
```

Gradient Boost

```
Macro avg accuracy:     0.5973333333333334
Macro avg precision:    0.59597640825711
Macro avg recall:       0.5973333333333333
```

Random Forest

```
Macro avg accuracy:     0.584
Macro avg precision:    0.5755902040022776
Macro avg recall:       0.584
```

Extra Trees

```
Macro avg accuracy:     0.596
Macro avg precision:    0.587654716175843
Macro avg recall:       0.596
```

BERT & SVM

In this method, we use BERT model to vectorize our documents. We then used SVM with linear kernel to classify the documents. These are the results:

```
Macro avg accuracy:     0.6146666666666667
Macro avg precision:    0.6136211818002154
Macro avg recall:       0.6146666666666666
```

Glove & SVM

In this method, we used Glove for word to vector transform. Then we calculated the mean of these vectors to calculate vector of a document. Then we used SVM to classify the documents. These are the results:

```
Macro avg accuracy:      0.5546666666666666
Macro avg precision:     0.5561333286782891
Macro avg recall:        0.5546666666666668
```

SVM & TfIdf

In this method, documents are vectorized according to tf-idf values for each word inside of them. Then we use these vectors to classify the documents with support vector machine algorithm with linear kernel. These are the results:

```
Macro avg accuracy:      0.668
Macro avg precision:     0.6681291383193938
Macro avg recall:        0.668
```

Work Distribution

Mahir has worked on

- Training the model with:
  - Universal sentence encoder with Support Vector Machine, Random Forest Classifier, Logistic Regression and Extra Trees Classifier methods.
- Comparing probable stop-words to use, for instance spacy's and nltk's stop words.

Yahya has worked on

- Training the model with:
  - Universal sentence encoder with Gradient Boost method
  - Support Vector Machine method with TfIdf vectorizer, Glove word embedding, BERT embedding.

Conclusion

We got the best results by using tf-idf for vectorization and support vector machine learning for classification. We haven't tried fine tuning methods for pre-trained models, by doing that results would have gotten better.

You can run our final model with this command:

```
python 462project_step3_BigBrains.py <path-to-pkl> <path-to-test-data>
```