Natural Language Processing

María Blanco González-Mohíno, José Alberto Seco Sánchez Camacho, Pablo Velasco Crespo and Adrián Ruíz Esteban

Contributing authors: Maria.Blanco4@alu.uclm.es; JoseAlberto.Seco@alu.uclm.es; Pablo.Velasco2@alu.uclm.es; Adrian.Ruiz6@alu.uclm.es;

Abstract

Working with Natural Language faces a lot of problems. To work with it, it is necessary to preprocess the data, which is not as hard as it used to because of the amount of libraries. The result of the essay will provide us a helpful way to work with Natural Language.

1 Problem description

Our main problem is how to get information from Natural Language, to do so it is necessary some preprocessing and vectorization. All the methods and results are in this GitHub and the dataframes used for the creation of the model are in the following drive.

1.1 Preprocessing

On this phase we clean the data from different things like useless symbols, capital letters, emojis and repeated words. We also replace contractions with its equivalent. We also lemmatized the data and implemented a spelling corrector. Finally we have stored all this clean data on a dataframe.

1.2 Vectorization

To make the vectorization we implemented the three configurations. Once we have all the methods applied we count some data like number of words, number of sentences, number of verbs and number of nouns.

2 Methods and materials

2.1 Classification algorithms and datasets

Two classification algorithms have been developed, SMV and Random Forest algorithm to compute a binary classification. These two algorithms have been developed for each of the vectorization types used, **TFIDF**, **TFIDF + N-grams** and **TFIDF + N-grams + Pos tagging** in order to compare them.

For the realization of the models, 70% of the best characteristics were selected using the *selectKBest*.

As we indicated, we used cross-validation with (cv = 3) in each of our models.

Testing dataset shape for each model:

	TFIDF	TFIDF-NGRAMS	TFIDF-NGRAMS-POSTAG
X_train	(420, 3520)	(420, 109767)	(420, 72689)
y_train	420	420	420

3 Experiments and results

Metrics for SVM model:

```
Precission:  1.0
Recall:  1.0
Accuracy:  1.0
False positive rate:  0.0
F-measure:  1.0
```

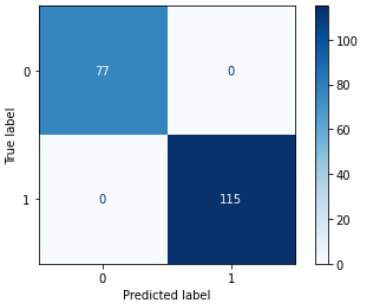
Metrics for Random Forest model:

```
Random Forest:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        71
           1       1.00      1.00      1.00       109

    accuracy                           1.00       180
   macro avg       1.00      1.00      1.00       180
weighted avg       1.00      1.00      1.00       180
```

We obtain in both models a precision of 100% and an accuracy of 100% too.

4 Conclusions

We obtain the same results for all our models. Confusion matrix:



As we can see there is no false negative or positive this is the optimal situation. Maybe, this is happening due to the k-best-selection, we are taking the best features to make our models, so we are trying to optimize them.