Text Mining – Project: Opinion Mining Evaluation Forum

Team Name: Datalin

Members: *Alice Vale* **|** r20181074 **//** *Eva Ferrer* **|** r20181110

Rafael Sequeira **|** r20181128 **//** *Raquel Sousa* **|** r20181102

Description of the best approach

To better understand the data that was available, a series of plots were created to verify the distribution of the target in the *train* and *test* dataset. Both datasets had an identical number of instances for each target value, so no adjustment steps were applied in this stage. The number of observations per target value was also plotted, so to check for possible imbalances in the data.

Next, the absolute frequency of words was examined to grasp what type of expressions were most common in the dataset and simple text mining pre-processing steps were applied, namely lower casing the text, removing meaningless words, namely stop words, and *Lemmatisation*, the process of transforming different variations of a word into a single one to facilitate analysis. Punctuation was also removed except for exclamation and question marks since they are a useful indicator of the phrases' meaning and were considered valuable to ease the process of opinion mining.

For feature engineering both the *bag of words* and the *TF-IDF* methods were employed, as seen during the lectures. It is relevant to mention that only the *train* set was fitted to the functions in order to avoid data leakage, which would negatively impact the final results. Finally, various models were tested using cross validation such as *K-Nearest Neighbours* and *Support Vector Machine*, although the final model implemented was a simple *Logistic Regression.*

Experimental Setup

Several Python libraries were used in the development of this project, highlighting *Beautiful Soup* and the *Natural Language Toolkit*, which were imperative for the pre-processing of the sentences received. For the modeling and other relevant steps, the library used was scikit-learn.

The first stage was to perform a brief exploration of the dataset received, in regards of the target distribution. It was clear that in both training and test the predominant sentiment was Anger, whereas Surprise and Disgust where the least observed. A counter for the words was also applied, which retrieved punctuation marks as the most common words to appear, that lead us to believe that some of these should be included in the analysis, since they are a significant way to express sentiment.

The pre-processing applied is crucial to the success of the model deployed, and our approach was to lowercase, remove punctuation except for the exclamation and interrogation points, removing stop words and lemmatize, in this respective order. The choice in terms of punctuation firstly also included the keeping of ellipsis, since in our perspective they are discriminative of specific sentiments like hesitation or disappointment. However, their inclusion in the analysis resulted in the depreciation of the performance of the models tested, so this approach was disregarded.

The choice of lemmatization, instead of stemming, is based on the formers' advantages of considering the surrounding context and producing an actual word.

Bag of words and *TF_IDF* were used for feature engineering, so to represent the space in a sparse binary matrix, with the use of *CounterVectorizer*, which records if the word appeared or not in the sentence, and guarantee that more discriminant words for the labeling were given higher importance.

The evaluation of the model created is based on a *StratifiedKFold* with 10 folds which returns the Macro F1-score, Accuracy, Recall and Precision for both test and dev sets. Results can be seen per fold or as an average.

The two benchmarks used were a K-nearest neighbors' classifier and C-Support Vector Classification. Both were used in their default versions, meaning no fine-tuning was applied, in order to resemble the ones provided in the project guidelines for the weaker and stronger baselines respectively.

The code created is easily reproducible, especially with the creation of the *"Full Pipeline"* cell in the *Jupyter Notebook* delivered, which, when ran, deploys the entire process from the importing of the training dataset until the final prediction generation.

Evaluation of the Best Approach

The best approach found was the *Logistic Regression*, in the table below there is a summary of the metrics' results for each of the labels and as macro avg.

Label/Metric	Precision	Recall	F1-Score
1	0.34	0.58	0.43
2	0.40	0.42	0.41
3	0.24	0.10	0.15
4	0.38	0.23	0.29
5	0.50	0.39	0.44
6	0.30	0.25	0.27
7	0.34	0.24	0.28
8	0.41	0.40	0.41
Macro Average	***0.36***	***0.32***	***0.33***

	Logistic Regression	*KNN*	*SVM*
Accuracy (dev set)	0.371	0.267	0.361

Our best solution is clearly predicting the label 5 - joy - the best, and the label 3 – disgust - the least. Additionally, the values for the precision are higher than the other metrics. Our decision was based on the Accuracy results, since the *LR* model had the best values when comparing to the other baseline models.

Regarding the error analysis, we have created confusion matrixes that account for the data imbalance, for each one of the labels. We are showing the matrixes for the 3rd and the 5th labels, respectively.

898	25
69	8

865	38
59	38

Future Work & Limitations

Since the chosen model was an exceptionally straightforward but an effective approach to the goal of opinion mining, the following step would be to test other, more complex algorithms and evaluate the results.

Upon investigation of several scientific articles, some reports suggested that algorithms similar to *Neural Networks* could be insightful for opinion mining, yet it is known that this type of model usually operates as a *black-box* and is challenging to control and comprehend the classification process. Others mentioned the *Logistic Regression* as a potentially effective algorithm for this task, which validates our decision to employ such a model as our final solution. Nevertheless, we are aware that this choice comes with several limitations, the main one being that it performs poorly when attempting to detect more complex relationships, when compared to more robust procedures. Finally, *Naïve Bayes* could, likewise, be a potentially useful algorithm to test for opinion mining since it was cited in some articles and presented during theoretical lectures as a commonly chosen model for text mining challenges.

Bibliography

1. V. Dhanalakshmi, D. Bino and A. M. Saravanan, "*Opinion mining from student feedback data using supervised learning algorithms*," 2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC), 2016, pp. 1-5, doi: 10.1109/ICBDSC.2016.7460390.

2. K. L. S. Kumar, J. Desai and J. Majumdar, "*Opinion mining and sentiment analysis on online customer review*," 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 2016, pp. 1-4, doi: 10.1109/ICCIC.2016.7919584.