Data Science Lab: Process and methods
Politecnico di Torino

Project report
Student ID: s273479

Exam session: Winter 2020

1 Data exploration (max. 400 words)

This report addresses a sentiment analysis task on Italian *TripAdvisor* reviews, labeled as positive or negative. The development set, to be used both as training and validation set, presents an unbalanced class distribution, as positive reviews are almost twice the negative ones.
At first, to get an idea of the reviews' content, a WordCloud is generated removing basic stopwords. From that, it is clear that the dataset contains reviews of Italian hotels.

Figure 1: Dataset WordCloud

Analysing the most frequent characters a few interesting things emerged. First, there is a correlation between labels and some punctuation (ellipsis, exclamation and question marks), as shown in figure 2. Next, several reviews contain emojis. They have a strong meaning and should be taken into account building the model. At last, some oriental characters are present both in the development and evaluation set and must obviously be processed to have a coherent dataset. Therefore, the dataset could contain non-Italian reviews. This is checked using *spaCy*[4], a natural language processing library which provides advanced features for the Italian language and language detection

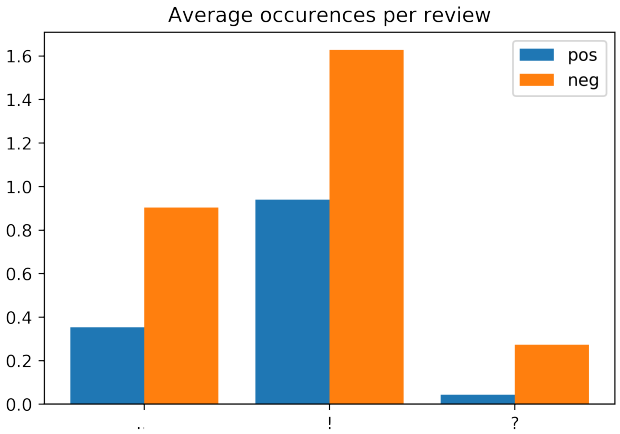

Figure 2: Punctuation occurrences

function, and the almost 10 reviews found are in different languages, hence can be considered outliers, or the reviewer provided also the Italian translation.

Then other possible features are searched. The length of the reviews follows a similar distribution for both classes (figure 3) so is not considered, while the presence of a person name in a review strongly suggests positive evaluation. This is inferred using a dataset containing some of the most popular names in Italy[2]. In addition, the shortest and the longest reviews also contain useful information, so they cannot be considered outliers.

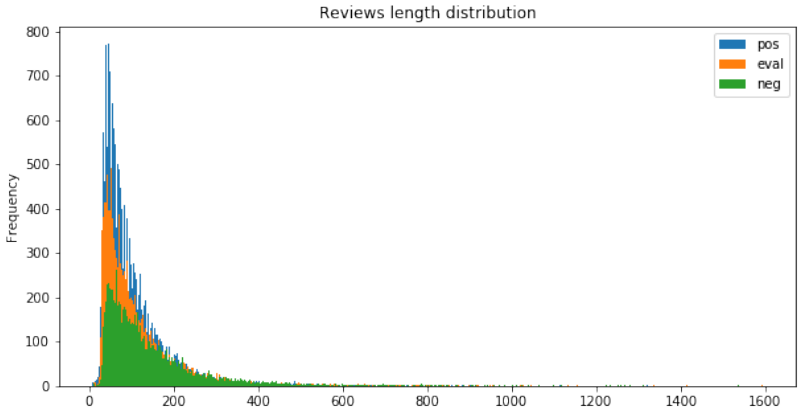

Figure 3: Reviews length distribution

Finally, with the aim of improving model stability and reduce overfitting probability more reviews can be added to the development set. To do that, a Python TripAdvisor scraper have been developed[5] and more than 6000 reviews have been downloaded. However, this is not enough to balance the two classes.

2 Preprocessing (max. 400 words)

Before the application of a classification algorithm textual data must be cleaned and processed.
First of all, non-Italian reviews described in the previous section are removed from the development set keeping only the Italian translation when available, while the reviews in the evaluation set are translated to Italian, as they cannot be removed to preserve result integrity. The translation process is done with *googletrans*[1], a Python library which implements Google Translate APIs and provides automatic translations. Text documents are then normalized to lower-case, non-informative characters (such as apostrophes and other punctuation) are removed and remaining punctuation marks are surrounded with blank spaces to treat them as separate tokens.
At this point, numerical features are extracted from the documents according to a bag-of-words representation with tf-idf weighting scheme. While processing the documents with scikit-learn's TfidfVectorizer, basic Italian stopwords provided by the Natural Language ToolKit are removed (with the exception of "non", which could appear in interesting bigrams), words are tokenized keeping punctuation and emojis and the Italian *SnowBall* stemmer[3] included in NLTK is used to reduce each word to its root form, as stemming proved to work better than lemmatization on Italian language after some tests. In addition, bigrams and trigrams are also considered to find frequent and meaningful word combinations.
The result of the transformation is a sparse matrix, whose dimensionality can be reduced through Singular Value Decomposition. However, that significantly decreases the classifiers' performances and therefore is not used in the final model.

3 Algorithm choice (max. 400 words)

To perform a sentiment analysis task several classification techniques can be used. In particular, this report compares the performances obtained by Multinomial Naive Bayes classifier, Support Vector Machines with linear kernel and Random Forest, as they deal with a large dataset with a high number of features. Every classifier is first evaluated according to weighted F1 score and trained with its default configuration on the development set, cross-validating on five folds. The results are reported in the following table:

Classifier	Accuracy	Label	Precision	Recall	F-measure	Weighted F-measure
Random Forest	0.933	neg	0.970	0.818	0.887	0.932
		pos	0.929	0.988	0.953	
Multinomial NB	0.929	neg	0.989	0.787	0.877	0.927
		pos	0.908	0.996	0.950	
Linear SVC	0.968	neg	0.959	0.940	0.949	0.968
		pos	0.972	0.981	0.976	

Table 1: Classification algorithm comparison

Support is always 9220 for negative labeled samples and 19532 for the positive ones.
As we can see, Random Forest and Multinomial Naive Bayes classifiers have similar performances with respect to all the metrics, while the Linear Support Vector Classifier generally performs better. Indeed it gets high values when the others are poor, getting good precision in classifying positive reviews and good recall for the negative ones, therefore the F1 score is better. That means that the linear Support Vector Machine is able to find a linear hyperplane which separates well the two classes.
Interestingly, the Multinomial Naive Bayes classifier obtains a very high recall for the positive class,

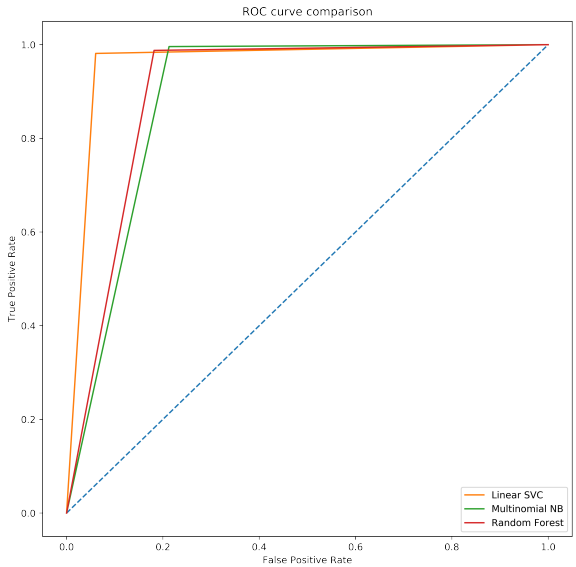

Figure 4: Receiver Operating Characteristic curves

so it can almost perfectly detect positive reviews. However, precision is much lower and negative recall is quite bad, so it does probably tend to label the majority of the reviews as positive.
In addition, to graphically compare the classification models, the Receiver Operating Characteristic curves have been plotted for the three classifiers (figure 4), showing that the linear SVC is quite near to the ideal model, while Random Forest and Multinomial NB still perform well but have an higher false positive rate. As a consequence, linear Support Vector Machines is chosen as final classification model and tuned on the training data.

4 Tuning and validation (max. 400 words)

To build a good final model for sentiment analysis it is necessary to find the best hyperparameters for both the vectorizer and the classifier. To train the final model the additional samples and features mentioned in section 1 have not been used, as they increase the complexity without bringing significant improvements.
To extract numerical features from the documents several combinations of scikit-learn's TfidfVectorizer's hyperparameters have been tried and the best results have been obtained filtering too frequent and too rare combinations of words, setting relative maximum document frequency threshold to 0.6 and absolute minimum document frequency to 4.
Next the classifier must be tuned. This is done through an exhaustive search over the possible scikit-learn's LinearSVC hyperparameters, by means of a grid search with cross validation to find their best combination. The best resulting classifier is then trained and validated on the development set, getting weighted F-measure 0.969, and its confusion matrix is plotted. Since the classes are unbalanced, the matrix has also been normalized with respect to the number of negative and positive labeled samples. In this way the recall scores are displayed on the main diagonal and the actual results can be better visualised.

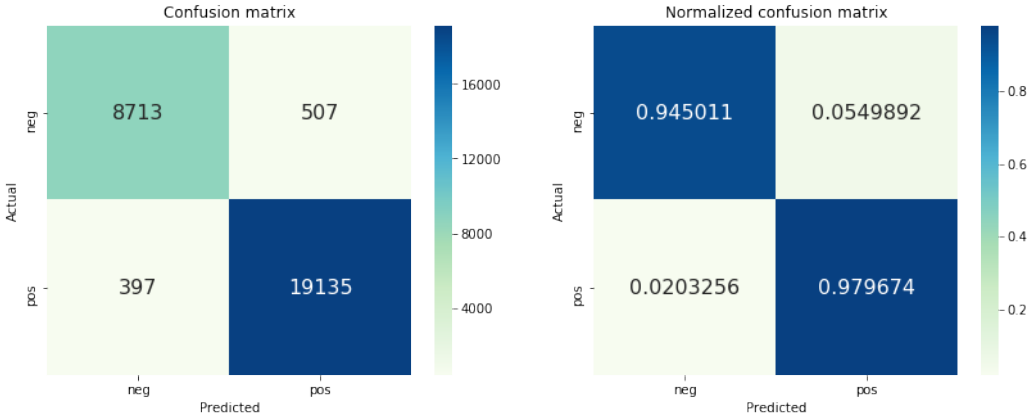

Figure 5: Confusion matrices

In conclusion, the final model is able to provide good sentiment predictions and achieves good scores, so it is used to compute predictions on the evaluation set. The predicted labels and the development set labels show a similar distribution, which is desirable as the two dataset have been probably sampled from the same initial one, therefore the results are exported to a .csv file and uploaded on the submission platform.

References

[1] *googletrans · PyPI*. URL: https://pypi.org/project/googletrans.

[2] *Italian newborn names*. URL: https://gist.github.com/gabridego/a7a19daab9a5de763c5c28f783e5dee5.

[3] *nltk.stem package — NLTK 3.4.5 documentation*. URL: https://www.nltk.org/api/nltk.stem.html.

[4] *spaCy · Industrial-strength Natural Language Processing in Python*. URL: https://spacy.io.

[5] *TripAdvisor scraper for Italian hotels*. URL: https://github.com/gabridego/tripadvisor-scraper.