Data Science Lab: Process and methods

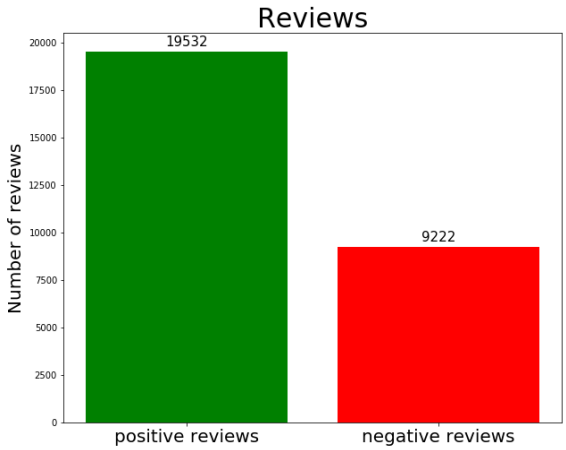
Politecnico di Torino

Project report
Student ID: s269748

Exam session: Winter 2020

1. Data exploration

The goal of this assignment is a sentiment analysis of hotel reviews.
An efficient structure to load data into memory is pandas DataFrames that allows an easy manipulation and exploration of the data.
The dataframe containing the developments data has two columns, one with the reviews, the second with the corresponding labels (just positive or negative); instead the other dataframe, containing the evaluation set, has only the column with the reviews.
Thanks to the functions of pandas dataframe, it is possible to easily do an overview of all the data: the development set has 28754 reviews, instead the evaluation one has 12323 reviews; there are no missing or empty values.
After that, it is necessary to check if the distribution of the classes is proportional, but it is not because the positive reviews are about the 70% of the total (there are 19532 positive reviews and 9222 negative reviews); this aspect must be considered when the set will be split in train and test in order to maintain the initial distribution of the data[1].
This must be done because otherwise there is a risk that the class of negative reviews will be underrepresented during the training part.



[1] In "train_test_split" function is possible to specify the parameter "stratify" which indicates the fashion to stratify data, and to do that it must be set to True also "shuffle" parameter.

A further observation that could be done at this step is that the raw data given are textual data that don't have a suitable format for the machine learning model.
For this reason the reviews will be processed by transforming each document into a feature vector (through two different techniques: "TF-IDF" and "Token counts" according to the classification approach used).
This process will lead to have an huge amount of features and for this reason a feature selection must be used in order to avoid the problem of "Curse of dimensionality".
Also the labels have to be processed because the classification models need of integer target values.

Furthermore, by analysing a random review it is possible to see how there are many grammatical errors and also some sentences don't make any sense: a possible improvement that could be done is try to correct these mistakes.

2. Preprocessing

Each dataframe must be pre-processed in order to have an homogeneous dataset.
Also here it is possible to take advantage of the dataframe structure used: its function "apply" allows to process fast each row of the dataframe.

The executed pre-processing consists in many steps:

- Case normalization: each word is converted to its lower case.
- Punctuation elimination: all punctuation marks are removed except the exclamation that it's not eliminated because in the context of a review it could be useful to better understand the meaning of a sentence.
  Another approach adopted because reviews are analyzed is to eliminate the emoticons that some users could have written.
- Number conversion: digit numbers are rewritten in words.
- Stopwords elimination: words which can be considered useless in text analysis are removed, "non" is not removed because it could be useful when n-grams will be studied.
- Short words elimination.
- Lemmatization: it consists on grouping together different inflected forms of a word. The lemmatization is done here, and not in the Tfidf Vectorizer or Count Vectorizer in order to be able to control better the process.[2]
- Stemming: it's useful because the used lemmatizer can't handle the gender of superlative adjectives and so this is done by the stemmer[3].
- Stopwords elimination: it's repeated because after lemmatization and stemming there might be new words that have to be removed.

Then reviews must be converted into feature vectors through the computation of the "TF-IDF" or "Token counts". Two different approaches are used because different classifiers work better with different type of feature vectors.
The obtained result is a sparse matrix which must be subjected to a dimensionality reduction in order to avoid the curse of dimensionality, eliminate irrelevant features and reduce noise. For this reason the Truncated SVD is used and to decide the dimensionality of output data it's studied the percentage of variance explained by each of the selected components.

[2] It's not use NLTK lemmatizer but spaCy lemmatizer which works quite well, but it's considerably slower than NLTK, so it's necessary to disable some of its functions that are not needed.
[3] Snowball Stemmer is used.

3. Algorithm choice

For classification task many algorithms could be used:

- Random Forest
- K-Nearest Neighbor
- Support Vector Classification
- Multinomial Naive Bayes
- Logistic Regression

Random Forest at first glance could be considered the best estimator because it is an ensemble learning technique which allows to avoid overfitting and improve stability. The problem of this estimator in this case is that using the SVD reduction previously done there is an higher risk that some reviews aren't correctly classified because they don't have a significant value for the feature used in a split (each review has a lot of features).

KNN isn't used since it's very slow when the number of observations is high because it doesn't construct an internal model but simply stores instances of the training data.

SVC offers instead different kernels, each of them applies a particular transformation to the data. Since the problem is linear (just positive or negative), it's useless to transform data with different type of SVC kernels, a linear one is enough.
Precisely Linear SVC is preferred to SVC with linear kernel because it's faster.

Naive Bayes works very well, most of all with the feature vectors found through Count Vectorizer. It might be though that using this method could be risky because of the assumption of conditional independence between every pair of features: but it is not true because the dependence of two words is already represented by the relative n-grams.

Logistic Regression instead implements a regularized logistic regression and so it's suitable for the problem since the data has to be classified in just positive or negative.

In the choice of the algorithm used there is the only difference between the two projects that have produced the two results chosen in the leaderboard.
The best one was reached by combining 3 different algorithms, a sort of random tree implementation, where the final prediction was done by a majority voting between Linear SVC, Logistic Regression and Multinomial Naive Bayes. This implementation has the purpose of discovering a result as robust as possible that it's not influenced too much by outliers that could affected a classifier.
The second result instead was given by a Linear SVC since it was the best in the training part of the model, this to ensure that it is not affected by wrong predictions made by other, less accurate, classifiers.

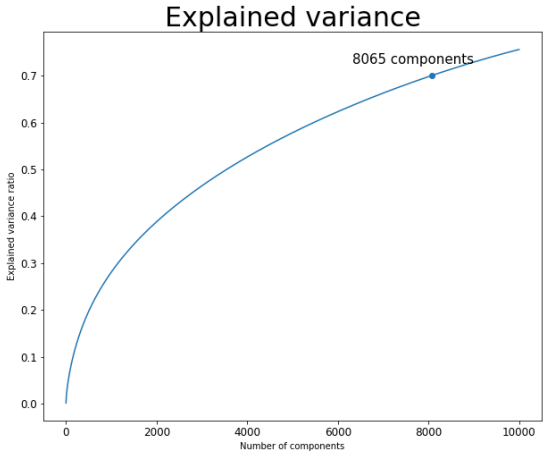
4. Tuning and validation

The final model used to classify the evaluation dataset is obtained through the same process made to study the development dataset which gives the best result.

There are many values that have to be set in the whole model:

- Number of components in the Truncated SVD
- Document frequency in Tfidf Vectorizer and in Count Vectorizer
- N-gram range in Tfidf Vectorizer and in Count Vectorizer
- Hyperparameters selection for each classifiers

The number of components in the Truncated SVD is chosen according the value of total variance explained: 0.7 is chosen as threshold for explained variance.



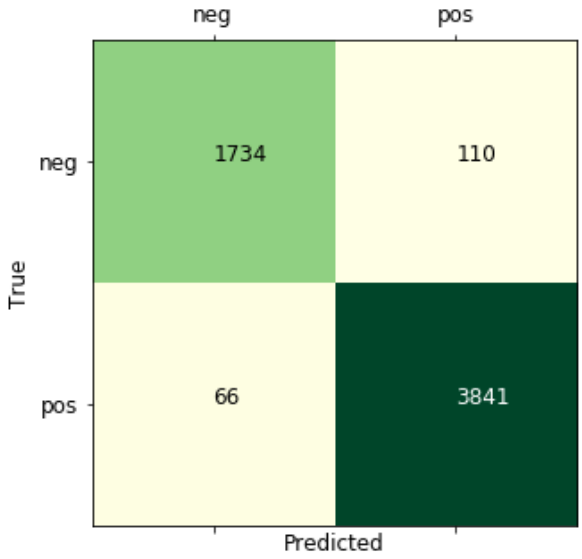
Then it's important to select an appropriate value for the minimum and maximum document frequency that a n-gram must have in order to be considered. 5 is chosen as minimum document frequency, in order to leave out words that are in less than 5 documents and so could be misspelled words or in general words too particular to be useful. Instead as maximum document frequency 0.7 is selected (which means that only n-grams that appear in at most the 70% of the reviews are considered), this because the number of positive reviews is just under 70% and so in this way all that words that appear indistinctly in all the categories are not considered.

Another main step is the selection of a correct value for the n-gram range which indicates the lower and upper boundary of the range of n-values that have to be extracted. In fact in addition to unigrams, also bigrams and trigrams are considered because in this way it's possible to capture the meaning of some words which taken in isolation could have a completely different interpretation.

Finally to perform the hyperparameters selection each of the classifiers has to be tuned: this is performed by a cross-validation with a grid search that thanks to a dictionary tries all the possible combinations of the parameter values passed; furthermore this

approach ensures that there won't be overfitting on the chosen values because at each iteration a partition of the data it's used as validation set.

This is the confusion matrix obtained in the training part by the model where three different classifiers are used, it's possible to conclude that it works well, especially for the positive reviews, it has more problems for the negative reviews and this could be due to fact that they have different proportions in the development dataset.



Sofia Perosin