Data Science Lab: Process and methods

Politecnico di Torino

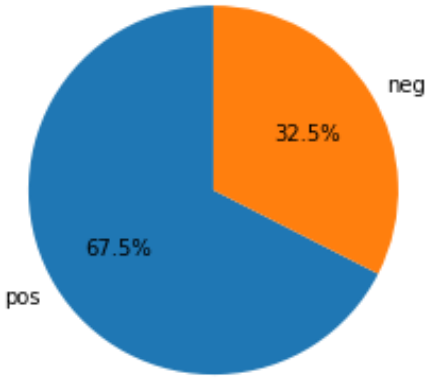
Project report

Student ID: s278312

Exam session: Winter 2020

1. Data exploration

Sentiment Analysis is the automated process of analyzing text data and sorting it into positive or negative sentiments.
The given dataset has been specifically scraped from the tripadvisor.it Italian web site. It contains 41077 textual reviews written in the Italian language, divided into 2 set: development one and evaluation one.
The development set contains 28754 reviews with their correspondent label, that could be positive or negative.
At first, it has been verified if the development set contains some missing values in the reviews or in the labels, but it correctly presented all the values.
Then, the dataset is explored internally to see if there are some special characters, or inconsistencies that could be given from the fact that our reviews are traduced from other languages. Some incongruities, that will be eliminated in data preprocessing phase, are found.
Another step is to see if our dataset is balanced or not.
The development dataset, is imbalanced, indeed it presents a dominant class which is the positive one.



The ratio of positive to negative class is approximately of 70/30, so it has to be balanced.
This because, after the generation of the model, despite the accuracy could probably be high, this is not the insurance of the generation of a good model.
The accuracy is not the better metric to be evaluated, because it is not always reliable. Indeed, a model trained on imbalanced data tends to predict more often the dominant class in our dataset, and so, it has difficulty to predict the less dominant one.
So, the best metric to evaluate our classifier is the f1_score, the harmonic mean of the precision and the recall.

2. Preprocessing

Text preprocessing is one of the most important phase of a data science project, because the choice made for the representation of the data input, affects the quality of the result.
On the dataset, are performed a series of operations.
All the punctuation is removed, for the excessive presence, that not help to distinguish documents, and because it doesn't matter sentiment analysis, as well as all the numbers and alone text characters.
Another important step of this phase is removal of the stop-words, commonly used words such as (in Italian language) 'a, gli, le..' that are repeated many times in both kind of reviews, positive and negative, that are very not interesting for sentiment analysis task.
Furthermore, all the extra tabs are removed, in fact they only represent additive noise of the data.
In the data exploration phase, we have found some inconsistencies, like emoticons, that could express a sentiment, but not in case of text processing, indeed because of their nature they couldn't be considered as text values.
Other important steps are the conversion to lower case of all the characters of the reviews, because upper case characters doesn't help in the specific task, and the normalization of all the accents.
Another important step of text preprocessing is the tokenization. This technique helps to transform data into a more digestible way for the model. Each review is transformed in a vector composed by all his words or some combination of them.
So, it is used the Bag-of-Words model, text representation that disregards grammar and words order but keep multiplicity.
Some mathematical computations have to be performed before fitting the classifier.
What is computed on the Bag of Words is the weight function TF-IDF (Term Frequency-Inverse Document Frequency), that measures the importance of a term with respect of a document or of a collection of documents.
Each token is so related to its TF-IDF, that is the principal value taken into account for the analysis.
The lower is this value, more frequent the token associated is, in all the documents, making so, hard the capacity of discriminating on it.
The higher is, more the token is helpful to distinguish a positive review from a negative one.

3. Algorithm choice

It has been used a Pipeline of evaluators, which performs a series of operations useful to approaches this specific sentiment analysis task.
The preprocessing phase is computed in part by a customized function and in part by a precomputed function of sci-kit learn, that does the major part of the work.
The used function is called TfidfVectorizer.
It is preserved its part of preprocessing, that consists of converting to lower case all the words, and normalizing the word with some accents.
The tokenization phase, so the creation of the Bags-of-Words, and the TF-IDF computation are also performed by the previously mentioned function.
For a more complete analysis, not only single words are considered, but also bigrams, which are sequences of two adjacent elements from a string of tokens.
The bigrams are very useful for the task, because it is more probable that a combination of two words characterizes a specific label, instead of only one.
It has been seen that the development set is imbalanced, so to prevent the domination of the positive label on the negative one, it has been used a function of imbalanced-learn API which allows to do oversampling of the minority class.
RandomOverSampler function over-samples the minority class by picking samples at random, copying it and reputting it in the population.
Seen the high number of features, and because not all of these are useful for the specific task, despite the selection done by the TfidfVectorizer, it is also performed a Feature Selection operation with the selectPercentile method of sci-kit learn.
Given a specific percentile and specified a scoring function, it performs a selection of the best features basing on univariate statistical sets.
This task could be approached with different methods. The one which is preferred is the Multinomial Naïve Based classifier.
This model recursively takes a significative test document that belongs to the class that has higher probability, computed with the Bayes' rule, and compute the probability of obtaining a document like it by computing the product between the count of words in the test document and the probability that those words appears in one review that belongs to a specific class.
It is chosen this model because is pretty fast in model generation and in prediction, and performs very well on text classification, if it works with a weighting function, and on simple tasks like this, that expects a classification on a single attribute: the sentiment.

4. Tuning and validation

The 75% of development set is used for training and validation, while 20% for the test set.
For the tuning of the hyperparameters, it is used GridSearchCV method of sci-kit learn, which allows to pass the Pipeline, and perform a cross-validation to find the best hyperparameters on the specific scoring function passed.
The hyperparameters to tune are different:

- min_df (minimum document frequency)$\in [3,5]$: all the features that doesn't appear in less than 3 documents are not considered useful for the task.
- max_df (maximum document frequency)$\in [0.03,0.3]$: the aim is to find the features which better characterize a determinate label. So, seen the higher number of features, only the ones which are present in less than the 30% of the documents have to be considered, indeed too frequent are, worse is.
- norm$\in$ ['l1', 'l2']: the unit norm of each output of the rows. It could be the sum of squares(l2) or the sum of absolute values of the vector(l1).
- Score function of the feature selector could be 'f_classif' based on ANOVA statistical test or 'chi2' which use chi-square statistic for categorical targets.
- Percentile of features$\in [20,23]$ with the highest values of the score function to be considered.
- Smoothing parameter alpha$\in [0.4, 1]$ is a way of regularizing Naïve Bayes, in order to avoid frequency-based probability equal to zero that will wipe out all the information in the other probabilities.

The best solution found has the average weighted f1_score computed on 5 fold of validation is 0.9598, obtained with the hyperparameters:

- min_df=2, max_df=0.06, norm='l2' for TfidfVectorizer;
- Feature selection performed with ANOVA statistical test, by taking the 25 percentile of features.
  So, the features passed to the fit method, after these operations are 40338.

It is showed an extraction of the best 1000 features:

- The smoothing parameter is 0.7.

At the end, it is computed the f1 weighted measure on the test set that is 0.9617.
So, it could be presumed that the model hasn't overfitted because it has performed well on different data, so it has generalized good.
From the confusion matrix, it could be seen that is more frequent that the model fails in classifying a negative review than a positive one. This is obvious, because with the oversampling the problem of imbalanced dataset is partially solved, because the negative samples are however generated by an artificial function, by seeing the pre-existent ones. A further improvement could certainly be the import of more negative reviews.