Text Processing final project

Assigned: 11/6/2023
Due to: 11/7/2023 23.59
The assignment is to be done in groups of two!

Aspect Based Sentiment Analysis (ABSA) is the task of mining and summarizing opinions from text about specific entities (e.g. a restaurant). Typically, an ABSA system attempts to detect the main aspects that are discussed and estimate the polarity for each of them. In this setting one approach is to define an inventory (a closed set) of aspects for a domain and then build a system that can identify these aspects in text and decide about polarity. For the restaurant domain an aspect inventory has been defined in the context of SemEval conferences; e.g., the inventory contains FOOD#QUALITY, RESTAURANT#GENERAL, RESTAURANT#PRICES etc. For more info, see Appendix A in http://galanisd.github.io/Papers/2016SemEvalABSATaskOverview.pdf. T**he goal of this assignment is to build a system that decides the polarity of a given aspect**. For this you have to register and login to http://metashare.ilsp.gr/ . Then download this dataset http://metashare.ilsp.gr:8080/repository/browse/semeval-2016-absa-restaurant-reviews-english-train-data-subtask-1/cd28e738562f11e59e2c842b2b6a04d703f9dae461bb4816a5d4320019407d23/ . It is an XML file that contains 350 restaurant reviews split into **2000 sentences**. Each sentence has been manually annotated **with *aspects ("category" attribute)*** and for e**ach aspect a "polarity" label has been assigned *(positive, neutral, negative)*.**

~~1.Write a python program (split.py) that reads/parses the XML file and splits it to 10 parts (35 reviews per part). Each part should be stored in a separate file (part1.xml, part2.xml, … part10.xml)~~

2. Write a **python function** (in a train.py) that trains a model and saves it to disk **(save to disk=pickle!).** The function should take as parameter an **array (to list_)** that indicates which parts will be used for training.

a. As a ML model do *not* use k-NN but select another one, e.g. ***Support Vector Machines, Logistic Regression.***

b. Use a variety of features, e.g. based on **unigrams, bigrams, trigrams**, POS tags, bigrams of POS, **number of named entities**, **TF-IDF scores,** *sentiment lexica*, **word2vec**, etc. Whatever you think is more appropriate.

3.Write a python function (in a test.py) that loads a saved model and uses it for *predicting the polarities for the sentence aspects of a specific part*. A parameter of the function should specify which part will be used.

4.Write an experiments.py that uses the functions of train.py and test.py. The experiments.py should do 10-fold cross validation. At each iteration, *9 parts will be used for*

training and 1 for testing/evaluation. Calculate accuracy for each of the 10 folds/iterations and an average of them.

5.Use a **feature selection technique** and assess if all the features that you have used are important. For example, if you have in total n features you can keep the k (k<n) most important (based on feature selection) and re-run experiments.py. Report results with at least 2 different k values.

```
def show_most_informative_features(vectorizer, clf, n=20):
    feature_names = vectorizer.get_feature_names()
    coefs_with_fns = sorted(zip(clf.coef_[0], feature_names))
    top = zip(coefs_with_fns[:n], coefs_with_fns[:-(n + 1):-1])
    for (coef_1, fn_1), (coef_2, fn_2) in top:
        print "\t%.4f\t%-15s\t\t%.4f\t%-15s" % (coef_1, fn_1, coef_2, fn_2)
```

The **results (output logs)** of your experiments should be described in a report.docx; i.e., which **features** are used, which **ML algorithm**, which feature selection method was applied, which features are more important, your results. Your code should contain comments that explain what you are doing.