CS918: 2018-19

Exercise two: Sentiment classification for social media.

Submission: 12 pm (midday) Monday 10th December 2018

Notes:

a) This exercise will contribute towards 20% of your overall mark.
b) Submission should be made on Tabula and should include a .zip file with **Python code** and a **report** of 3-5 pages summarising the techniques and features you've used for the classification, as well as the performance results.
c) You can use any Python libraries you like. Re-use of existing sentiment classifiers will receive fewer marks. The idea is for you to build your own sentiment classifier.

Topic: Building a sentiment classifier for Twitter

Semeval competitions involve addressing different challenges pertaining to the extraction of meaning from text (semantics). Organisers of those competitions provide a dataset and a task, so that different participants can develop their system. In this exercise, we will focus on the task 4 of Semeval 2017 (http://alt.qcri.org/semeval2017/task4/). We will focus particularly on Subtask A, i.e. classifying the overall sentiment of a tweet as positive, negative or neutral.

As part of the classification task, you will need to preprocess the tweets. You are allowed (and in fact encouraged) to reuse and adapt the preprocessing code you developed for Exercise 1. You may want to tweak your preprocessing code to deal with particularities of tweets, e.g. #hashtags or @user mentions.

You are requested to produce a standalone Python script that somebody else could run on their computer, with the only requirement of having the Semeval data downloaded. Don't produce a Python script that runs on some preprocessed files that only you have, as we will not be able to run that. Exercise guidelines:

- **Data:** The training, development and test sets can be downloaded from the module website (semeval-tweets.tar.bz2). This compressed archive includes 5 files, one that is used for training (twitter-training-data.txt) another one for development (twitter-dev-data.txt) and another 3 that are used as different subsets for testing (twitter-test[1-3].txt). You may use the development set as the test set while you are developing your classifier, so that you tweak your classifiers and features; the development set can also be useful to compute hyperparameters, where needed. The files are formatted as TSV (tab-separated-values), with one tweet per row that includes the following values:

  `tweet-id<tab>sentiment<tab>tweet-text`

  where sentiment is one of {positive, negative, neutral}. The tweet IDs will

be used as unique identifiers to build a Python dictionary with the predictions of your classifiers, e.g.:

```
predictions = {'163361196206957578': 'positive',
               '768006053969268950': 'negative',
               …}
```

- **Classifier:** You are requested to develop classifiers that learn from the training data and test on each of the 3 test sets separately (i.e. evaluating on 3 different sets). You are given the skeleton of the code (sentiment-classifier.tar.bz2), with evaluation script included, which will help you develop your system in a way that we will then be able to run on our computers. Evaluation on different tests allows you to generalise your results. You may achieve an improvement over a particular test set just by chance (e.g. overfitting), but improvement over multiple test sets makes it more likely to be a significant improvement.

  You should develop 3 different classifiers (or 3 different sets of features, or a classifier with 3 different settings), which you will then present and compare in your report. You can develop 2 baseline classifiers, and a third, further developed classifier. You should briefly compare the performance of the classifiers.

- **Evaluation:** You will compute and output the macroaveraged F1 score of your classifier for the positive and negative classes over the 3 test sets.

  An evaluation script is provided, evaluation.py, which has to be imported and used, as in the skeleton code provided. This evaluation script produces the macroaveraged F1 score you will need to use. You can also compute a confusion matrix, which will help you identify where your classifier can be improved as part of the error analysis.

  If you perform error analysis that has led to improvements in the classifier, this should be described in the report.

To read more about the task and how others tackled it, see the task paper:
http://alt.qcri.org/semeval2017/task4/data/uploads/semeval2017-task4.pdf

Marking will be based on:

a. Your performance on the task: good and consistent performance across the test sets. While you are given 3 test sets, we will be running your code in 5 test sets to assess its generalisability. Therefore, making sure that your code runs is very important. [20 marks]
b. Clarity of the report. [20 marks]
c. Producing runnable, standalone code. [20 marks]
d. Innovation in the use of features. [15 marks]
e. Methodological innovation. [10 marks]

Total: 85 marks