Task 1: Simple Classifier

Part 1: Manual Classification

In ‘Task 1 Part I: Resources’ you will find four film reviews. Please read each of them and record your own opinion about whether the sentiment of the review is positive or negative in the database.

We expect there to be some disagreements in these opinions. We will use the database with the judgements later in the course.

Part 2: Sentiment Lexicon Database

One very simple way of assigning sentiment automatically is to use a **lexicon** with words that we expect to be associated with particular sentiments. For example, we might expect the words *excellent* and *boring* to provide good clues.

Please open the sentiment lexicon (in ‘Task 1 Part 2: Resources’) and look at the entries for *excellent* and *boring*. What do you think the entries mean?

Based on the reviews that you looked at in part 1, please write down in your lab notebook 10 words which you think would be useful for the task of classifying the sentiment of film reviews and say whether you expect each to be positive or negative.

Now check whether your words occur in the sentiment lexicon. Do you agree with the lexicon as to whether they are positive or negative?

Part 3: Simple Classifier Virtual programming lab

The main task in this session is to use the sentiment lexicon to automatically classify a large number of film reviews as positive or negative. You will create some code and upload it to the automatic tester.

DON’T edit any files you download from Moodle (other than the tester). This includes the lexicon.

We became aware after the VPL was set up that the lexicon has a few spurious characters in a number of entries. The lexicon can still be read in by using the field names.

The database of reviews for this task is in ‘Task 1 Part 3: Resources’. The actual sentiment (**ground truth**) is encoded in the file `sentiment dataset/review_sentiment`.

Please follow the general instructions for programming labs.

1. Classification. Write a program that reads in the sentiment lexicon, and then reads in the review files (sentiment dataset/reviews). For each review file, it records how many positive lexicon words it contains, and how many negative ones, and then arrives at a decision about whether the entirety of the text is positive or negative (as described in the slides). For this experiment, in the case of a tie between positive and negative, resolve in favour of positive.

   Use the Stanford tokenizer accessible via `Tokenizer.java` to tokenize the review texts. Make sure to add `stanford-postagger.jar` to your build path during your experiments (you don't need to submit it).

2. Evaluation. Your program will not be right in all cases. In order to find out how close to the truth it gets, you will write a short routine that **evaluates** the algorithm, i.e. gives it a score expressing how close to the truth it gets. Write the evaluation program that reads in your output and the truth and tells you the proportion of times your program made the right decision. This number is called **accuracy**.

3. Improve your classifier. After recording the accuracy of your first attempt, you should try some simple ways to raise the accuracy of the system. For instance, can you improve the way that ties are handled? You have a choice as to exactly how to try and improve the system, but try the simplest things first, and record everything you try in your lab notebook.

At this point, you may submit your code to the automatic tester. Once your code has passed, please follow the procedure for obtaining the tick explained in the general instructions for programming labs.