Programming Assignment

Word Sense Disambiguation (WSD) with Decision Lists

Write a Python program called `decision-list.py` that implements a decision list classifier to perform word sense disambiguation. This method is described on pages 641-644 of the JM text. You should also read the original paper about this method, which is available on BlackBoard (`ACL94_Yarowsky_Decision_List.pdf`). Note that even though that paper focuses on accent restoration, the method can be used with many classification problems, including word sense disambiguation.

Your program should use as many features from the original Yarowsky's method as you think will result in an accurate classifier. You are free to add other features if you think they will help. Please make sure you only identify features from the training data, and that you clearly explain what features you are using in your detailed comments. Your classifier should run from the command line as follows:

```
$ python decision-list.py line-train.xml line-test.xml my-decision-list.txt >
my-line-answers.txt
```

This command should learn a decision list from `line-train.xml` and apply that decision list to each of the sentences found in `line-test.xml` in order to assign a sense to the word **line**. Do not use `line-test.xml` in any other way (and only identify features from `line-train.xml`). Your program should output the decision list it learns to `my-decision-list.txt`. You may format your decision list as you wish, but please make sure to show each feature, the log-likelihood score associated with it, and the sense it predicts. The file `my-decision-list.txt` is intended to be used as a log file in debugging your program. Your program should output the answer tags it creates for each sentence to `STDOUT`. Your answer tags should be in the same format as found in `line-answers.txt`.

`line-train.xml` contains examples of the word line used in the sense of a phone line and a product line where the correct sense is marked in the text (to serve as an example from which to learn). `line-test.xml` contains sentences that use the word line without any sense being indicated, where the correct answer is found in the file `line-answers.txt`. You can find `line-train.xml` and `line-test.xml` in the files section of our site in a compressed directory called `line-data.zip`.

Your program `decision-list.py` should learn its decision list from `line-train.xml` and then apply that to `line-test.xml`.

You should also write a utility program called `scorer.py` which will take as input your sense tagged output and compare it with the **gold standard** "key" data in `line-answers.txt`. Your scorer program should report the overall accuracy of your tagging, and provide a confusion matrix similar to the one found on page 156 of JM. This program should write output to `STDOUT`.

The scorer program should be run as follows:

```
$ python scorer.py my-line-answers.txt line-answers.txt
```

You can certainly use your `scorer.py` program from the previous assignment as a foundation for this program.

Both `decision-list.py` and `scorer.py` should be documented according to the standards of the programming assignment rubric. In `decision-list.py` include your accuracy and confusion matrix in the comments. And compare your results to that of the most frequent sense baseline.

Please submit your program source code (`decision-list.py` and `scorer.py`) as well as a script file called `decision-list-log.txt` that you should create as follows:

```
$ script decision-list-log.txt
$ python decision-list.py line-train.xml line-test.xml my-decision-list.txt >
my-line-answers.txt
$ head -50 my-decision-list.txt
$ head -10 my-line-answers.txt
$ python scorer.py my-line-answers.txt line-answers.txt
$ exit
```

NOTE: if you use **jupyter notebook**, you can have all comments and required running outputs/logs in the notebook(s). And then save into HTML(s) and zip all notebook file(s), HTML files, and other files into ONE zip file. Please submit **only one zip file**.