Stats 141C High Performance Statistical Computing Spring 2017

Homework 1

Lecturer: Cho-Jui Hsieh Date Due: April 21, 11:59pm, 2017

Keywords: *Python Programming, Quora Question Pair Competition*

Let's try to come up with a simple solution for the Kaggle Quora-question-pairs competition: `https://www.kaggle.com/c/quora-question-pairs`. Given a pair of questions (two strings), the goal is to predict whether they are the same question or not.

We will use the file "training.csv" and "validation.csv" to test our algorithm.[1] This file is in the csv format, and each line of the training data contains the following information about a pair of sentences:

```
id, qid1, qid2, question1, question2, is_duplicate
```

where

- id: the id of a training set question pair
- qid1, qid2: unique ids of each question (only available in train.csv)
- question1, question2: the full text of each question
- is_duplicate - the target variable, set to 1 if question1 and question2 have essentially the same meaning, and 0 otherwise.

Problem 1. Preprocessing [20pt]

Write a function in python to pre-process a sentence (string). The function will look like:

```
def preprocess( str_in ):
    ...
    ...
    return str1_out
```

We require "str_out" to be the output of the following pre-processing steps:

1. Change all the uppercase letters in "str_in" to lowercase letters.
2. Replace all the following parameters by " " (white space):

   ```
   ? , ! . ( ) ' " :
   ```

3. Remove all the "-" character in "str_in".

[1] Since quora didn't provide the labels for their test data, we randomly split "train.csv" (the full training data on their website) into "training.csv" and "validation.csv", for evaluating our algorithms.

Problem 2. Compute the "overlapping score" for each question pairs [30pt]

Write a program, given "training.csv" as input, output the overlapping score for each row. Assume the question pair q_1 and q_2 are the preprocessed strings after Problem 1, and $q_1 = [w_{11} \quad w_{12} \cdots \quad w_{1n}]$, $q_2 = [w_{21} \quad w_{22} \cdots \quad w_{2m}]$ (the strings are split into words using white space), then we define the overlapping scores as the number of overlapping words divided by $m + n$ (sum of number of words). More specifically,

$$\text{overlapping score}(q_1, q_2) := \left(\sum_{i=1}^{n} I(w_{1i} \in q_2) + \sum_{i=1}^{m} I(w_{2i} \in q_1) \right) / (m + n)$$

where $I(w_{1i} \in q_2) = 1$ if $w_{1i} \in q_2$, otherwise $I(w_{1i} \in q_2) = 0$.

For each line of "train.csv" (excluding the first line), output the overlapping score. The input/output format of your code should be:

```
$ python ComputeScore.py training.csv
score_1
score_2
...
score_n
```

Report the following:

1. Overlapping scores for the first 10 lines
2. Maximum overlapping score, minimum overlapping score, medium overlapping score
3. Briefly discuss your findings.

Problem 3. Compute the training accuracy [25pt]

After having the overlapping scores, we can predict whether two questions are the same by thresholding. More specifically, we can predict the label for a question pairs (q_1, q_2) by

$$\text{sign}(\text{overlapping score}(q_1, q_2) - \text{thr}),$$

where thr is a positive real number for thresholding. The testing accuracy is defined as

$$(\text{number of correct predicted pairs})/(\text{total number of pairs})$$

Write a program to compute the testing accuracy for a given dataset and threshold:

```
$ python ComputeAccuracy.py training.csv thr
0.6001
```

("training.csv" is the filename of input dataset, "thr" is a real number (e.g., 0.25), and the output is accuracy.) Also, answer the following questions:

1. Report the accuracy on training.csv with thr $= 0.1, 0.2, 0.3, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.8, 1.0$. Which threshold gives you the best result?
2. Use the best threshold, and run the code on validation.csv. What is the validation accuracy?
3. Briefly discuss your findings.

Problem 4. Remove Stop Words [25 pt]

One problem of our approach is that the overlapping score might be dominated by the "stop words", such as "is", "a", "the", "what". Now we try to improve our model by removing the stop words. Note that if all the words in both questions are stop words, just report overlapping score to be 0.

Now we define the stop words by *all the words that appeared more than 10000 times in "training.csv"*. Now, try to modify ComputeAccuracy.py by removing the stop-words before computing overlapping scores. The input/output format should be:

```
$ python AccuracyRemoveStops.py training.csv thr
0.6051
```

(thr is the threshold and the output is accuracy). Also, report:

1. Report the accuracy on training.csv with thr $= 0.1, 0.2, 0.3, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.8, 1.0$. Which threshold gives you the best result?
2. Use the best threshold, and run the code on validation.csv. What is the validation accuracy?
3. Briefly discuss your findings.

Problem 5. Select the best weight? (Bonus) [10 pt]

In problem 3 and 4, we try all the threshold manually. Is there a better way to automatically select the best threshold? Try implement your method and report your findings.