11-411/611 HW5

February 2019

1 Introduction

In this assignment, you will create a Naive Bayes text classifier. Your classifier will distinguish speeches given by Republican and Democrats running for president. The training dataset includes 46 speeches distributed among 4 candidates, 2 from each party. The test dataset includes 18 different speeches from 3 of the 4 candidates. You will need to write a program to estimate the multinomial distributions P(term—class) and P(class) from the training dataset. Split on whitespace to tokenize. You do not need to deal with the problem of new words in the test data since some of the singletons (words appearing only once) in the training data have been replaced by the token UNKNOWNWORD. The multinomial conditional probability distributions of terms given a class should incorporate a single feature: the term itself. When estimating them, use add-one smoothing. Make sure to add the size of the vocabulary to the denominator when normalizing. If the previous two sentences are not clear, read the textbook and/or supplementary reading. The training and testing files have one document (speech) per line. The class of the document is given first, followed by a tab character, followed by the speech. Refer to the submission guidelines on details of input/output.

2 Task1 [50 points]

Write a python or java program to train a Naive Bayes classifier using the training data (train.txt). Use the test file (test.txt) to evaluate your classifier. Report overall accuracy (proportion correct) across categories, precision for each category, and recall for each category. Now repeat your tests using a different pair of training and test data, train2.txt and test2.txt. Report values using the same evaluation metrics. (Note: test.txt and test2.txt are more or less the same except that test2.txt has more unknown words since fewer words appear in train2.txt.)

In your program, please output all your result to task1.txt in the same format as example.txt in handout.

3 Task2 [25 points]

What are some problems with only testing on a test set of 18 speeches? What changes can you think of to make for a better evaluation? Are there drawbacks to your suggested changes?

4 Task3 [25 points]

Examine the probabilities of the predicted classes for the test set speeches. Are some predictions more "certain" than others? If any of the classifications are wrong, do the probabilities for these indicate relatively low confidence?

5 Bonus Task [50 points]

Naive Bayes models can be feature-based; that is, you can then think of each token in the document as consisting of a vector of features. The simplest case is what you have implemented so far: a token's sole feature is the entire word. Additional features might capture aspects of the word's internal structure or context. Extend your model to incorporate multiple types of features. Did classification accuracy improve? Why do you think you saw the effect you did? Be sure to explain the features you tried and measure their impact.

6 Submission Guidelines

Please submit a zip archive named handin.tar containing the following items to AutoLab. **Please do not put them in a folder inside the archive.**
(You can use command: tar cvf handin.tar YOURFILES)

1. A file called naivebayes.py or naivebayes.java which generates task1.txt (refer format below and the example.txt in handout). Your python file should take 4 parameters train1, test1, train2, test2, in that order. The code will be run using command "python3 naivebayes.py train.txt test.txt train2.txt test2.txt"

2. A file called README. Whether or not you collaborated with other individuals or employed outside resources, you must include a section in your README documenting your consultation (or non-consultation) of other persons and resources. If you have any additional information to give us, you should also put it there.

Please submit a single pdf containing the following items to Canvas.

1. Brief answers and comments for task2

2. Brief answers and comments for task3

3. Otionally, brief answers and comments for bonus task

task1.txt should read as follow:

overall accuracy
0.0
precision for red
0.0
recall for red
0.0
precision for blue
0.0
recall for blue
0.0

overall accuracy
0.0
precision for red
0.0
recall for red
0.0
precision for blue
0.0
recall for blue
0.0