Lab 1 Report

Sumon Biswas

Class Priors:

```
src — -bash — 134×40
[Sumons-MacBook-Pro:src sumon$ clear
[Sumons-MacBook-Pro:src sumon$ javac *.java
Sumons-MacBook-Pro:src sumon$ java NaiveBayes vocabulary.txt map.csv train_label.csv train_data.csv test_label.csv test_data.csv
Class Priors:
P(Omega =
             1) = 0.0426
P(Omega =
             2) = 0.0516
3) = 0.0508
P(Omega =
P(Omega =
             4) = 0.0521
             5) = 0.0510
6) = 0.0525
P(Omega =
P(Omega =
P(Omega =
             8) = 0.0525
9) = 0.0529
P(Omega =
P(Omega =
P(Omega =
           10) = 0.0527
P(Omega = 11) = 0.0531
P(Omega = 12) = 0.0527
P(Omega =
            13) = 0.0524
P(Omega = 14) = 0.0527
P(Omega = 15) = 0.0526
P(Omega = 16) = 0.0532
P(Omega = 17) = 0.0484
P(Omega = 18) = 0.0500
P(Omega = 19) = 0.0412
P(Omega = 20) = 0.0334
```

In our result P_{BE} is always greater than P_{MLE} . Because when we calculate P_{MLE} , only one word with zero occurrence can make the whole posterior to zero. Therefore, often it can not provide the correct result. In case of P_{BE} , we counter this by adding 1 with the number of occurrences.

Performance on Training Data:

In our result, we the overall accuracy on training data is more than accuracy on the test data. For training data it is about 94% and in case of test data is about 78%.

First, we use Bayesian Estimator to measure the performance. Later we also used Maximum Likelihood to measure. The performance using BE on training data is shown below:

src — -bash — 134×40

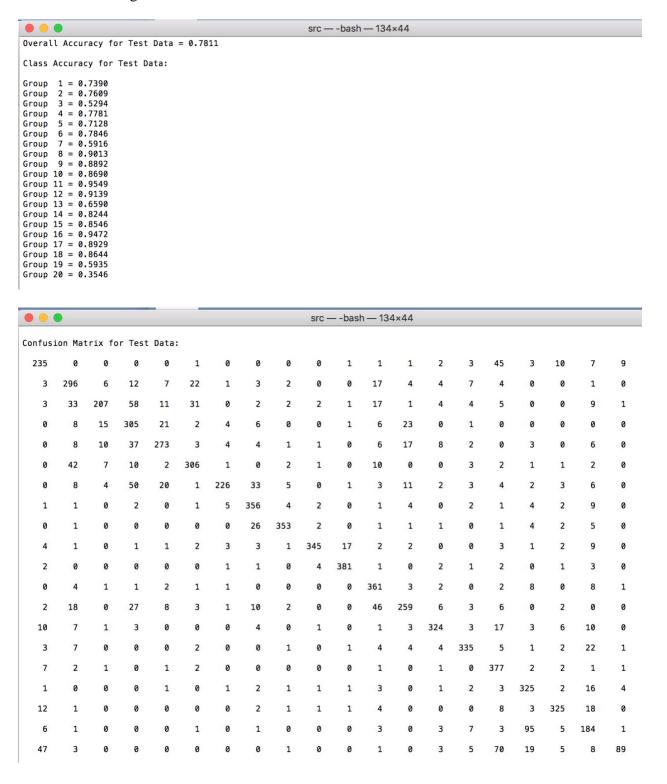
Overall Accuracy for Training Data = 0.9411

Class Accuracy for Training Data:

Group 1 = 0.9667 Group 2 = 0.9191Group 3 = 0.8794Group 4 = 0.9302Group 5 = 0.9409Group 6 = 0.9493 7 = 0.7749 Group Group 8 = 0.9662Group 9 = 0.9631Group 10 = 0.9714Group 11 = 0.9783Group 12 = 0.9798Group 13 = 0.9239Group 14 = 0.9764Group 15 = 0.9781 Group 16 = 0.9833Group 17 = 0.9853Group 18 = 0.9681Group 19 = 0.9698Group 20 = 0.7606

src — -bash — 134×44 Confusion Matrix for Training Data:

Performance using BE on test data is shown below:



When we use MLE the performance dropped significantly. The overall accuracy it gives is only 12%. The reason is that for a given class only one word with zero count can make the whole posterior to zero. The performance when we use MLE is given by:

