Purdue University Duc Viet Le
CS 59000SA

Assignment 2

Task 3.1

Note: in both tasks, I removed last 1327 records in the training folder because it's duplicate with testing folder.

- **Naive Bayes:** In naive Bayes, I have implemented the same algorithm covered in the lecture. The only modification from the lectures is that I removed all stopwords (e.g "the", "an", "a", ...). The following data is my algorithm's performance on test data:

```
False Positive Rate:  28
False Negative Rate:  8
Recall:               0.9911699779249448
Precision:            0.9697624190064795
F-score beta = 1:     0.980349344978166
```

- **SVM:** In SVM, I used the default SVM modeled provided by the library. I used 300 attributes for training and testing. The following data is the algorithm's performance on testing data:

```
[+] SVM on testing data
False Positive Rate:     37
False Negative Rate:     16
Recall:                  0.9823399558498896
Precision:               0.9600862998921251
F-score beta = 1:        0.9710856519367157
```

Task 3.2

- Evaluate the results of both algorithm, which one is better from your point of view and why?
  Ans. I think both algorithms perform well on testing data. From my point of views, I think naive Bayes is simpler, and the performance is somewhat better than SVM. Moreover, naive Bayes's algorithm performs better as the training data increases.

  However, In the case of spam filter, SVM is also good in the sense that it only needs around 300 attributes to give a good classifiers, so I think SVM is more efficient in term of computation compared to naive Bayes. However, it may not be true for other classifiers.

- Is your model overfiting/underfiting? How do you know whether your model is overfiting/underfiting or not?
  Ans. For Naive bayes, I think the model is neither overfitting or underfitting because there is no other "tweaks" other than removing stopwords, and the performance of the classifier on testing data

is good.

For SVM, I think my model is overfitting a little bit because I choose the number of attribute that yields a good F score on testing data.

Task 3.3 Cross Validation: Make a 5-fold cross validation for your SVM. Briefly explain how you prepare the 5-fold dataset and the result

Ans. In cross validation, I partition the training data in two 5 subsets:

```
partitionMatrices = np.split(processedTrainMatrix, 5)
```

Each time I used the other 4 subsets as training data, and evaluate the classifier against the last subset. The following data is the result for each fold:

```
[+] 5-fold Cross Validation
[First fold]
False Positive Rate:     35
False Negative Rate:  9
Recall:                  0.9772151898734177
Precision:               0.9168646080760094
F-score beta = 1:        0.946078431372549
[Second fold]
False Positive Rate:     21
False Negative Rate:     9
Recall:                  0.9783132530120482
Precision:               0.9508196721311475
F-score beta = 1:        0.9643705463182898
[Third fold]
False Positive Rate:     27
False Negative Rate:     8
Recall:                  0.9809069212410502
Precision:               0.9383561643835616
F-score beta = 1:        0.9591598599766628
[Fourth fold]
False Positive Rate:     19
False Negative Rate:     8
Recall:                  0.9801980198019802
Precision:               0.9542168674698795
F-score beta = 1:        0.967032967032967
[Fifth fold]
False Positive Rate:     22
False Negative Rate:     6
Recall:                  0.9853658536585366
Precision:               0.9483568075117371
```

```
F-score beta = 1:        0.9665071770334929
```

Task 3.4 For each of the algorithm, try to come up with an spam instance with word get, free, and iphone which can circumvent the detection and describe how you construct such instance. **Ans.** the instance of the spam email is in `duc-spam` folder. For naive Bayes, if we keep the body short with limited instance of spam word such as `free, iphone` and try not to use characteristics of spam email such as all upper case character, mispell, etc.

For SVM i think, it should be easier because we only need to create an instance that either close to a ham email or contains words that are not in attributes used for computation.