CS 334 - Homework 4

Alex Welsh

October 21st 2020

1 Question 1 - Feature Extraction + Model Selection

a. I chose to do holdout with a train, validation, and test data set. In the model_assessment method, I create the train/validation combined set and the test data set of emails. The code for this can be found in **q1.py**. My method for holdout, which returns the train and validation AUC can be found in **perceptron.py**. I chose holdout because I got perfect train AUC's and very high (96% and 95% for binary and count data, respectively) validation AUC's. My **y_Train.csv** and **y_Test.csv** are created in this method and saved in the main method of **q1.py**.

b. I built my vocabulary map with the train/validation data set and got around $\tilde{2}$426 words that occur in 30+ emails. The code can be found in **q1.py**.

c. I built my binary data set and saved them to **binary_Train.csv** and **binary_Test.csv**. The files are saved in the main method of **q1.py**.

d. I built my count data set and saved them to **count_Train.csv** and **count_Test.csv**. The files are saved in the main method of **q1.py**.

2 Question 2 - Spam Detection via Perceptron

a. The code for the perceptron can be found in **perceptron.py**. There is also a holdout method for model selection

b. According to Figure 1, the binary data set converges at about 7 to 10 epochs, whereas the count data set converges around 50 to 60 epochs. For the binary data set, I got 48 mistakes or 97% accuracy on the test data. For the the count data set, I got 85 mistakes or 95% accuracy on the test data. The code to run the models and generate the graph is in **perceptron.py**.

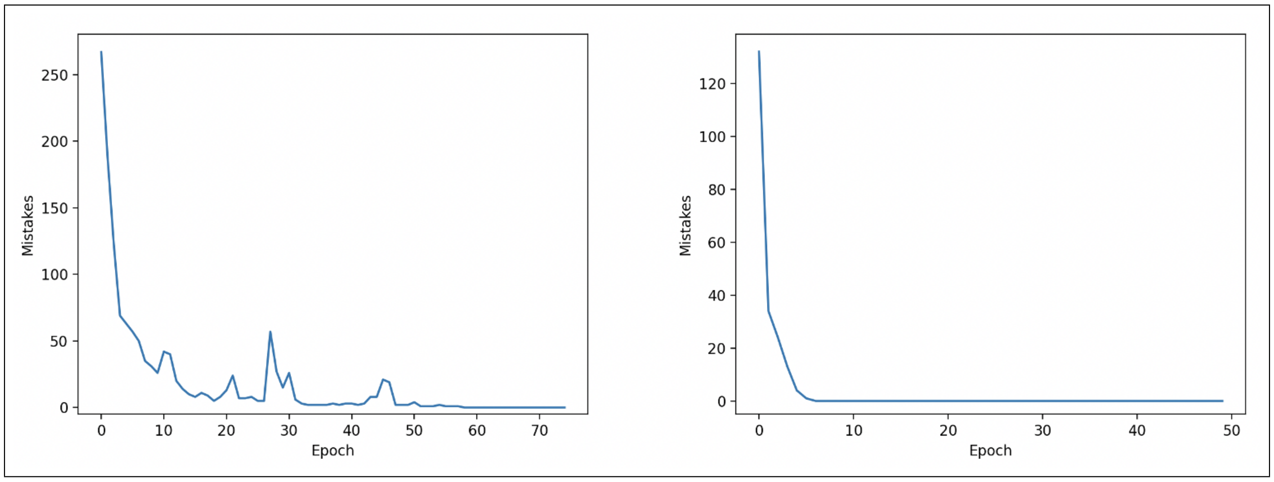
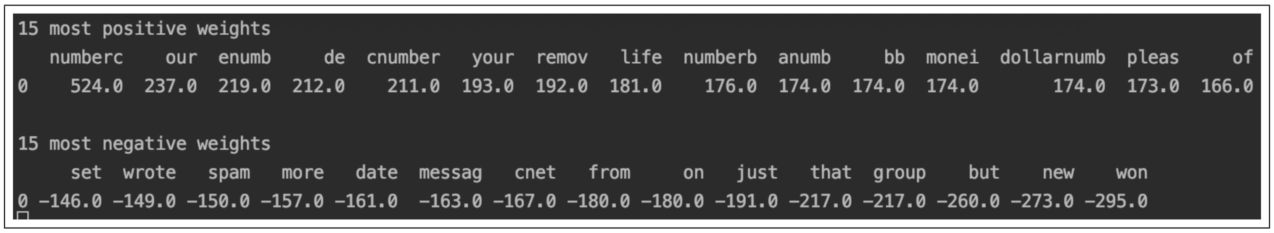


Figure 1. Number of mistakes for each epoch of both the count (left) and the binary (right) data sets.

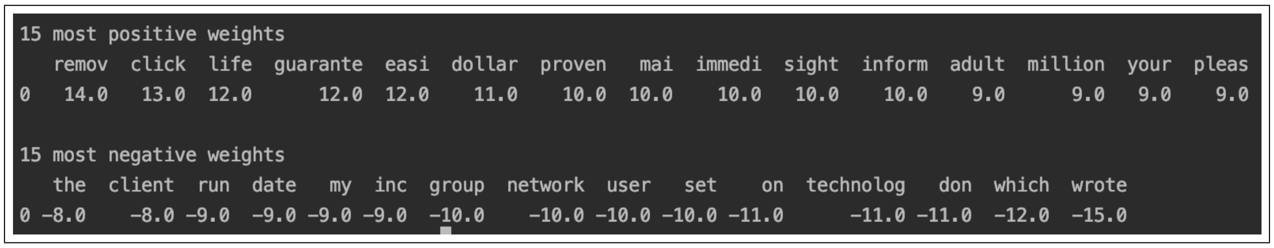
c. In Figure 2, the 15 most positively and negatively weighted words are shown for the count data set. In Figure 3, the 15 most positively and negatively weighted words are shown for the binary data set. The code to find the words is in **perceptron.py**.



```
15 most positive weights
   numberc    our  enumb     de  cnumber   your  remov   life  numberb  anumb     bb  monei  dollarnumb  pleas     of
0    524.0  237.0  219.0  212.0    211.0  193.0  192.0  181.0    176.0  174.0  174.0  174.0       174.0  173.0  166.0

15 most negative weights
     set  wrote   spam   more   date  messag   cnet   from     on   just   that  group    but    new    won
0 -146.0 -149.0 -150.0 -157.0 -161.0  -163.0 -167.0 -180.0 -180.0 -191.0 -217.0 -217.0 -260.0 -273.0 -295.0
```

Figure 2. 15 most positively and negatively weighted words for the count data set.



```
15 most positive weights
   remov  click  life  guarante  easi  dollar  proven   mai  immedi  sight  inform  adult  million  your  pleas
0   14.0   13.0  12.0      12.0  12.0    11.0    10.0  10.0    10.0   10.0    10.0    9.0      9.0   9.0    9.0

15 most negative weights
   the  client  run  date   my  inc  group  network  user   set    on  technolog   don  which  wrote
0 -8.0    -8.0 -9.0  -9.0 -9.0 -9.0  -10.0    -10.0 -10.0 -10.0 -11.0      -11.0 -11.0  -12.0  -15.0
```

Figure 3. 15 most positively and negatively weighted words for the binary data set.

3 Question 3 - Spam Detection using Naive Bayes and Logistic Regression

a. To run Naive Bayes with the binary data, the program will use BernoulliNB by running the following command: ”python q3.py binary_Train.csv yTrain.csv binary_Test.csv yTest.csv”. To run Naive Bayes with the count data, the program will run MultinomialNB using the following command: ”python q3.py count_Train.csv yTrain.csv count_Test.csv yTest.csv”. The model makes 84 mistakes on the binary data set and 52 mistakes on the count data set. The code can be found in **q3.py**.

b. To run Logistic regression with the binary data, use the same command for binary data as in question 3a. To run Logistic regression with the count data, use the same command for count data as in question 3a.

The model makes 29 mistakes on the binary data set and 36 mistakes on the count data set. The code can be found in **q3.py**.