**Machine Learning PS1: Perceptron**

Template Write-Up

**NOTES:**

**X** = E-mail threshold for a word to be considered a feature (word has to show up in X e-mails)

**N** = The size of your training set (it’s how many e-mails in *spam\_train.txt* you’ll use to train)

You should start with **X = 25** and **N = 4000**. Some questions ask you to vary these numbers and then explain/show how your program’s results change.

Q1. Explain why measuring the performance of your final classifier would be problematic had you not created a validation set (size = **5000 – N**) from your training data.

|  |
| --- |
| Because the validation set is necessary in order to measure the efficiency of the chosen training method before applying it to a real world scenario. It is also useful in assessing the properties of our modelling, such giving an estimate on what the error rate on the real test data should be like. |

Q4. How many passes did your implementation of the perceptron algorithm make before it converged? How many total mistakes did the algorithm make before convergence?

|  |
| --- |
| 10 passes and 627 errors. |

Q4. What is your error rate with the validation set? (The answer is a percentage; if it is above 10% of the size of your validation set, check your implementation)

|  |
| --- |
| 2.4% |

Q5. Which 15 words are most correlated with spam? That is, which 15 “features” have the most *positive* weights? Provide a screenshot or copy/paste of your program output.

|  |
| --- |
| ('yourself', 20), ('free', 21), ('name', 21), ('guarante', 22), ('cb', 22), ('spamd', 22), ('deathtospamdeathtospamdeathtospam', 23), ('pai', 25), ('death', 25), ('nb', 25), ('pleas', 27), ('sight', 29), ('remov', 45), ('click', 53) |

Q5. Which 15 words are least correlated with spam? That is, which 15 “features” have the most *negative* weights? Provide a screenshot or copy/paste of your program output.

|  |
| --- |
| ('wrote', -39), ('but', -27), ('prefer', -24), ('rob', -23), ('not', -23), ('reserv', -23), ('author', -22), ('head', -21), ('http', -20), ('date', -19), ('would', -19), ('there', -19), ('dn', -19), ('log', -19), ('httpaddr', -19) |

Q6. Vary N = 100, 200, 400, 800, 2000, 4000. Create a plot of the validation error percentage as a function of N.

|  |
| --- |
| ../Screen%20Shot%202016-02-02%20at%209.57.47%20PM.png |

Q7. Vary N = 100, 200, 400, 800, 2000, 4000. Create a plot of the number of perceptron algorithm passes as a function of N.

|  |
| --- |
| ../Screen%20Shot%202016-02-02%20at%209.40.50%20PM.png |

Q9. Keep N=4000 constant, and vary the value of X. Try X = 30, X = 35, and any other values of X you’d like. What do you think is the optimal value of X for your perceptron configuration? (Optimal here means lowest validation set error percentage)

|  |
| --- |
| X = 15 and X = 10 and X = 35= > 2.1%  X = 5 => 2.6%  X = 1 program would never finish ☹  X = 50 and higher => >2.6%  Best result I got for X = 30, with 1.9% but overall the difference feels minimal across the board |

Q9. Now use your best configuration on all of *spam\_train.txt* (N=5000, X=whatever you found to work well). Train with it, and report your error percentage on *spam\_test.txt*

|  |
| --- |
| 2.0% |

Q10. Try setting X=1000. How many features do you have in your model? What happens when the perceptron runs? Can you explain what’s going on, and why it happens?

|  |
| --- |
| There are 90 features.  The data is not linearly separable anymore, as the words that appear in a 1000 emails is a very small sample. More precisely, because there are fewer features in the vector, the number of linearly separable functions is exponentially lower. If the data is not linearly separable the algorithm most likely won’t converge. |