CS 334 Machine Learning HW #4

1(a) Justification for size and split method:

Training size was determined by testing accuracy on splits that are commonly used. A split of 0.8 to 0.2 was chosen because the perceptron trained on that split ratio seemed to have the lowest mistakes between the two methods.

2(c) Optimal Epochs: (Using 3-fold cross validation, higher values of k converges too early as dataset is too sparse)

For the Binary datasets,

Optimal Epoch: 50

#mistakes for training: 0, 0 (flat, proportion)

#mistakes for test: 24, 0.02 (flat, proportion)

For the Count datasets,

Optimal Epoch: 50

#mistakes for training: 23, 0.00479167 (flat, proportion)

#mistakes for test: 32, 0.02666667 (flat, proportion)

2(d) Positive & Negative Weights of Vocabulary List:

For the Binary datasets, (with corresponding weights round to three decimals)

15 Most Negative (most negative to least):

['basenumb', 'out', 'iirc', 'mai', 'balanc', 'style', 'greg', 'perform', 'kernel', 'few', 'accord', 'advanc', 'mark', 'voic', 'becaus']

['-14.506', '-14.428', '-14.423', '-13.451', '-13.077', '-12.421', '-12.362', '-12.340', '-12.297', '-12.244', '-12.132', '-12.014', '-11.805', '-11.767', '-11.722']

15 Most Positive (least positive to most):

'recommend', 'gpg', 'might', 'explor', 'from', 'unless', 'shape', 'correspond', 'aol', 'think', 'action', 'french', 'teledynam', 'fulli', 'encrypt']

['11.618', '11.639', '11.671', '11.701', '12.089', '12.119', '12.184', '12.423', '14.590', '16.461', '16.838', '17.147', '18.161', '18.196', '18.848']

For the Count datasets, (with corresponding weights round to three decimals)

15 Most Negative (most negative to least):

['paul', 'mason', 'out', 'basenumb', 'exmh', 'few', 'face', 'technic', 'argu', 'fit', 'truli', 'restor', 'album', 'through', 'surpris']

'-750.250', '-683.734', '-567.428', '-562.506', '-554.613', '-537.244', '-526.578', '-524.873', '-508.708', '-500.802', '-499.016', '-495.783', '-493.166', '-487.816', '-484.224']

15 Most Positive (least positive to most):

['senior', 'medic', 'life', 'diseas', 'let', 'think', 'useless', 'neither', 'thursdai', 'director', 'upon', 'campaign', 'encrypt', 'addit', 'recommend']

['553.926', '557.938', '570.227', '576.534', '576.716', '586.461', '588.640', '598.482', '648.682', '650.693', '662.481', '703.597', '719.848', '721.803', '733.618']

3(a) Naive Bayes Method:

For the Binary datasets,

Test Accuracy Naive Bayes: 0.9775, 0.0225 (acc, err)

Training Accuracy Naive Bayes: 0.97542, 0.0246 (acc, err)

Comparing the % errors for both training and test with the respective values for our perceptron, it seems that a simple perceptron performs significantly better for the training data (with no mistakes) but only slightly (but not significantly) better for the test data than Naive Bayes for our very simple 2 class classification problem. The error values (mistakes) seem to be less than 5% (hovering around 2%) for all the predictions in the dataset, so both models should perform very similarly - although we must note that our dataset is very small, so we do not know if this would hold in practice.

For the Count datasets,

Test Accuracy Naive Bayes: 0.9675, 0.0325 (acc, err)

Train Accuracy Naive Bayes: 0.97125, 0.0288 (acc, err)

Comparing the % errors for both training and test with the respective values for our perceptron, it seems that a simple perceptron performs slightly (but not significantly) better than Naive Bayes for our very simple 2 class classification problem. The error values (mistakes) seem to be less than 5% (hovering around 2-3%) for all the predictions in the dataset, so both models should perform very similarly - although we must note that our dataset is very small, so we do not know if this would hold in practice.

3(b) Logistic Regression

For the Binary datasets,

Test Accuracy Logistic Regression: 0.98 0.02 (acc, err)

Training Accuracy Logistic Regression: 0.9996, 0.00042 (acc, err)

The % error for both the logistic regression model and the perceptron model looks identical for both the training and test, both models will likely perform similarly - although we must note that our dataset is very small, so we do not know if this would hold in practice.

For the Count datasets,

Test Accuracy Logistic Regression: 0.975, 0.025 (acc, err)

Training Accuracy Logistic Regression: 0.999, 0.001 (acc, err)

We observe the same observations in this dataset as the previous one. The % error for both the logistic regression model and the perceptron model looks identical for both the training and test, both models will likely perform similarly - although we must note that our dataset is very small, so we do not know if this would hold in practice.