Adam Rosenberg
CS5350
Perceptron Report
25 September, 2017

1 Perceptron

1.1 Summary

My program is written in C++ 11.

The first step of my Perceptron Program is to define four structs:

1. point(double x, double y)
2. meanR(double r1, double r2, double r3)
3. perceptronReturn(double accuracy, vector¡double¿ weights, meanR rates)
4. crossValidateReturn(meanR rates, double accuracy)

After that, my program reads in a file from a vector of file names. From this file, it created a vector of vectors of type point: vector<<vector<point>>matrix. Each point in the matrix is the corresponding element number and feature value. While doing this I create a vector of type int to store the labels per row. After reading in the file, I fill up the rest of matrix with the corresponding element number and a feature value of 0. Now that we have the sparse matrix and label vector established, I begin cross validation by calling cross validate and passing in the needed information such as the number of epochs, u, and t. Cross validate creates a vector<double>weights and fill it with random values as assigned in the assignment: -0.1 to 0.1. I seeded the random generator against the current OS time. Due to this, the output of my program might have different results than what is shown below. In general, the hyperparameters performed pretty equally. The standout for each hyperparamater was the 0.1 value. The middle value often performed the best while debugging. Cross validate then calls the perceptron method driver which organizes some information before calling basic perceptron. Please note, I had a higher accuracy when I incremented t for each mistake. However, the Canvas discussion states that we should update t per example. My accuracy for the dynamic would have been a couple percent higher otherwise.

Basic Perceptron decides what to do based on a couple flags passed in such as: int sectionNum. sectionNum is an indicator for which variant of perceptron to execute. I wanted to keep all of the perceptron code in one function; however, if I were to redo this assignment I would have split the variants of perceptron into separate functions. With my last assignment in ML, I didn't write enough functions and this time I honestly wrote too many. The function chain that happens is too complex and made the magrin perceptron hard to write. My code started getting messier throughout the assignment, but I kept my output organized. Below I have pasted the desired output from each variant of perceptron.

1.2 Simple Perceptron

1. **Means:**
   Average learning rate (r) accuracy from every iteration of 5 fold validation in basic perceptron after 10 epochs:
   Mean accuracy with a learning rate of 1.0: = 90.7245
   Mean accuracy with a learning rate of 0.1: = 91.0897
   Mean accuracy with a learning rate of 0.01: = 90.6821

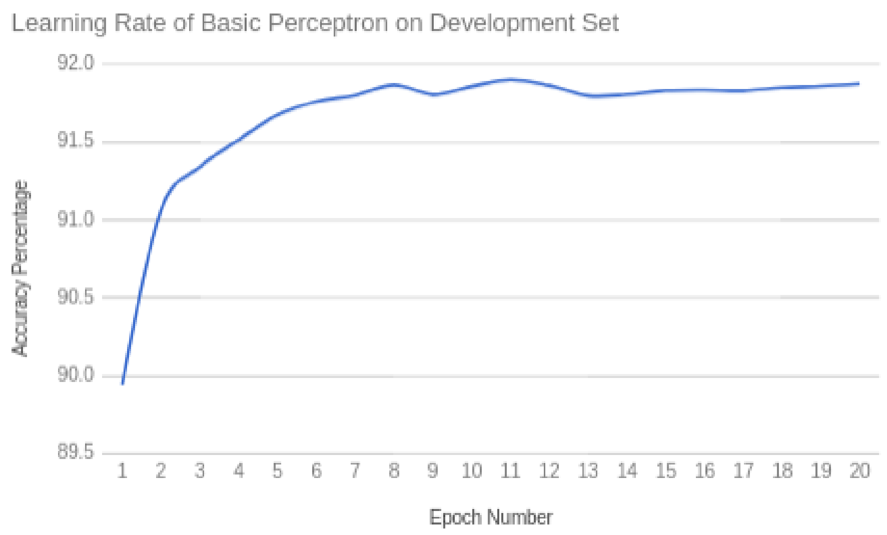
   The average accuracy while training against training file 0 = 90.7731
   The average accuracy while training against training file 1 = 91.7903
   The average accuracy while training against training file 2 = 91.8104
   The average accuracy while training against training file 3 = 92.2064
   The average accuracy while training against training file 4 = 92.3537

2. **Optimized Hyperparameter**
   The optimal hyperparamter for the learning rate is: 0.1 with a mean accuracy of: 91.0897

3. **Training Output**
   Using the optimal value of r on the training file for 20 epochs Testing classifier against dev set with an accuracy of: (per epoch)
   89.9421, 91.0637, 91.3411, 91.5159, 91.6787, 91.7631, 91.8028, 91.8687, 91.8074, 91.8596, 91.9024, 91.8657, 91.8012, 91.8079, 91.8331, 91.8370, 91.8320, 91.8516, 91.8615, 91.8777
   The average of these 20 accuracies = 91.6557
   The number of updates during the 10 epochs of the training session = 13063

4. **Test Output**
   Accuracy of the trained classifier against the test set = 90.448

5. **Graph**



1.3 Perceptron with dynamic learning Rate

1. **Means:**

   Average learning rate (r) accuracy from every iteration of 5 fold validation in dynamic perceptron after 10 epochs:
   Mean accuracy with a learning rate of 1.0: = 90.7011
   Mean accuracy with a learning rate of 0.1: = 90.8443
   Mean accuracy with a learning rate of 0.01: = 90.0763

   The average accuracy while training against training file 0: 90.6947
   The average accuracy while training against training file 1: 91.7802
   The average accuracy while training against training file 2: 92.0054
   The average accuracy while training against training file 3: 91.9873
   The average accuracy while training against training file 4: 92.2593

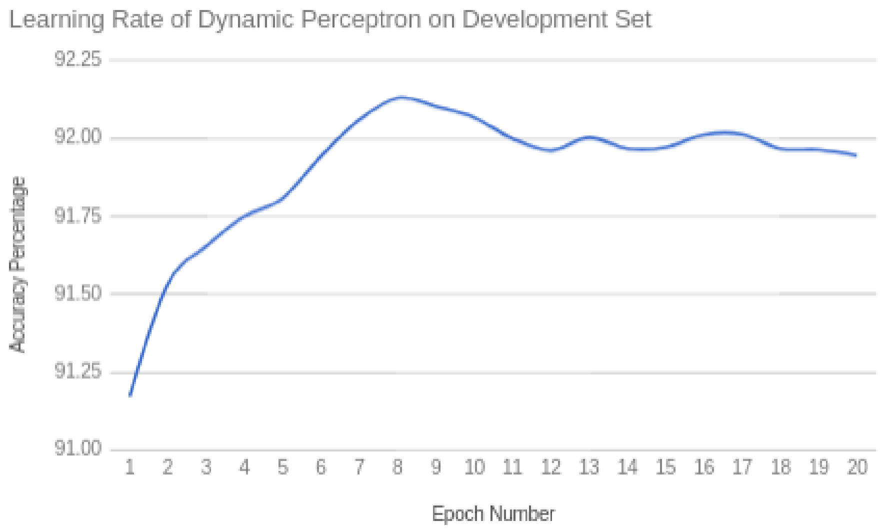
2. **Optimized Hyperparameter**
   The optimal hyperparamter for the learning rate is: 0.1 with a mean accuracy of: 90.8443

3. **Training Output**
   Using the optimal value of r on the training file for 20 epochs Testing classifier against dev set with an accuracy of: (per epoch)
   91.1722, 91.5340, 91.6546, 91.7511, 91.8090, 91.9440, 92.0612, 92.1310, 92.1048, 92.0695, 92.0011, 91.9626, 92.0043, 91.9682, 91.9730, 92.0134, 92.0150, 91.9682, 91.9644, 91.9465
   The average of these 20 accuracies = 91.9024

The number of updates during the 10 epochs of the training session = 1412

4. **Test Output**
   Accuracy of the trained classifier against the test set = 90.8104

5. **Graph**



1.4 Margin Perceptron

1. **Means:**

   Average learning rate (r) accuracy from every iteration of 5 fold validation in margin perceptron after 10 epochs:
   Mean accuracy with a learning rate of 1.0: = 69.169
   Mean accuracy with a learning rate of 0.1: = 86.7579
   Mean accuracy with a learning rate of 0.01: = 80.0635

   The average accuracy while training against training file 0: 72.3957
   The average accuracy while training against training file 1: 79.0459
   The average accuracy while training against training file 2: 76.2577
   The average accuracy while training against training file 3: 86.6646
   The average accuracy while training against training file 4: 76.4316

2. **Optimized Hyperparameter Combination**
   The optimal hyperparamter pair for the margin perceptron has a learning rate of: 0.1 with a mean accuracy of: 86.7579.
   and a optimal hyperparameter u = 0.01

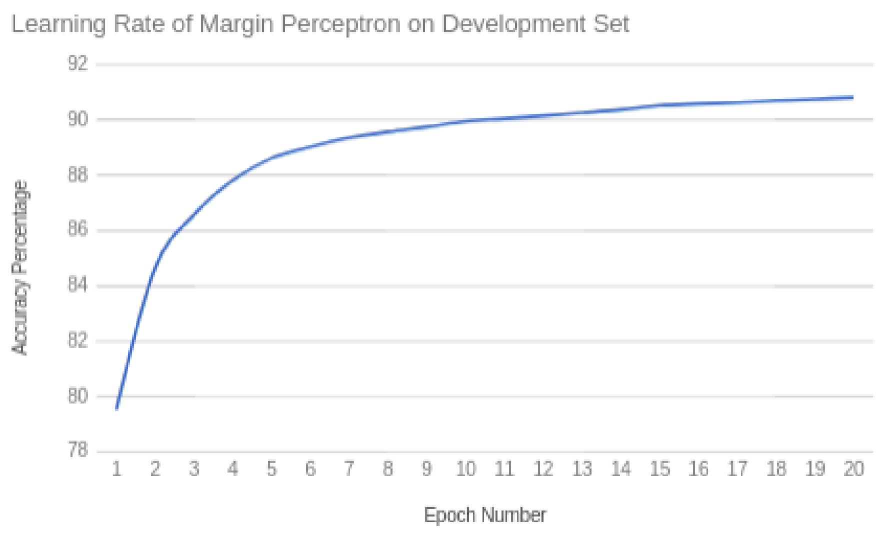
r = 0.1
u = 0.01

3. **Training Output**
   Using the optimal value of r on the training file for 20 epochs Testing classifier against dev set with an accuracy of: (per epoch)
   79.5224, 84.6599, 86.5895, 87.8437, 88.6397, 89.0497, 89.3736, 89.5984, 89.7733, 89.9711, 90.0671, 90.1712, 90.2816, 90.3969, 90.5451, 90.6024, 90.6529, 90.7139, 90.7685, 90.8321

   The average of these 20 accuracies = 89.0026
   The number of updates during the 10 epochs of the training session = 107884

4. **Test Output**
   Accuracy of the trained classifier against the test set = 85.0941

5. **Graph**



1.5 Averaged Perceptron

1. **Means:**

   Average learning rate (r) accuracy from every iteration of 5 fold validation in averaged perceptron after 10 epochs:
   Mean accuracy with a learning rate of 1.0: = 0.907922
   Mean accuracy with a learning rate of 0.1: = 0.909604
   Mean accuracy with a learning rate of 0.01: = 0.902902

The average accuracy while training against training file 0: 90.6947
The average accuracy while training against training file 1: 91.9511
The average accuracy while training against training file 2: 91.935
The average accuracy while training against training file 3: 92.0938
The average accuracy while training against training file 4: 92.2212

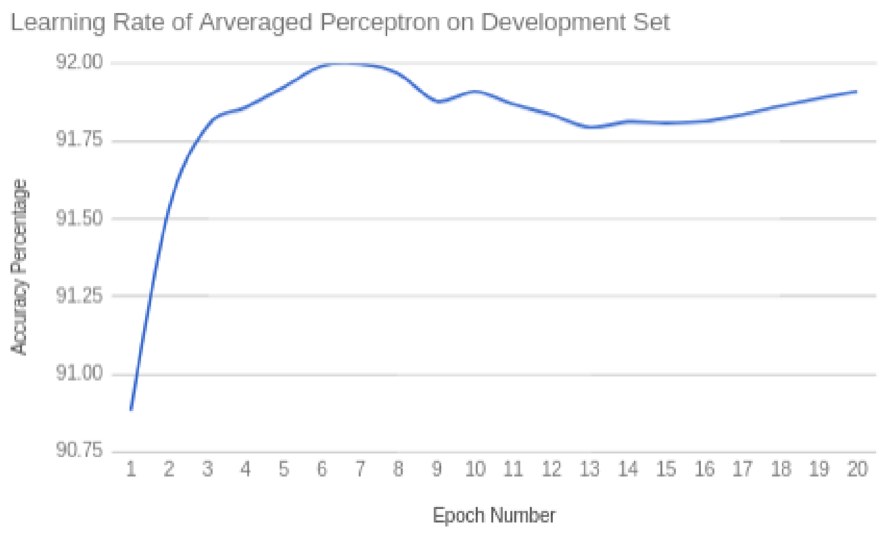
2. **Optimized Hyperparameter** The optimal hyperparamter for the learning rate is: 0.1 with a mean accuracy of: 0.909604

3. **Training Output**
   Using the optimal value of r on the training file for 20 epochs Testing classifier against dev set with an accuracy of: (per epoch)
   90.8828, 91.534, 91.7993, 91.8596, 91.9247, 91.9923, 91.9992, 91.9682, 91.8797, 91.9103, 91.8695, 91.8355, 91.7956, 91.8131, 91.809, 91.8144, 91.8362, 91.8636, 91.8882, 91.9103

   The average of these 20 accuracies = 90.8038
   The number of updates during the 10 epochs of the training session = 13159

4. **Test Output**
   Accuracy of the trained classifier against the test set = 90.3039

5. **Graph**


Learning Rate of Arveraged Perceptron on Development Set
Accuracy Percentage
Epoch Number