Adam Thelen and An Nguyen

EE 425X

Homework 2B

In this experiment, handwritten images of digit images of 0's and 9's were fed into both GDA and Logistic Regression models for classification. The digits were obtained from the MNIST data sets. Each image is 28x28 pixels which take on a value between 0 and 255. The goal was to classify the images with their corresponding label. In this experiment, 0 and 9 were chosen as the two classes. After filtering out all other numbers except the 0's and 9's, the training set consisted of 11,872 data points and the test set consisted of 1989 points.

First, the images were flattened into single row vectors so that they could be more easily dealt with. Each row in the "X" matrix now corresponds to a single label in the "Y" column vector. This is the proper setup for logistic regression and GDA.

The files are put in a subfolder named **mnist**, so the file paths in our code are *"./mnis/file_name.gz"*

Logistic Regression

The logistic regression model was first trained on synthetic data generated via random numbers. The logistic regression parameters for both synthetic data and real data were: 1) 1000 iterations, and 2) learning rate of (1/m)*(1e-2). The stochastic nature of the synthetic data lead the logistic regression model to perform with an accuracy of 91.4% after being run for 100 iterations. This is good considering each feature only had 20 associated datapoints. As for the real data, the model performed much better, with 99.3% accuracy. This is likely due to the more predictable and consistent nature of the data.

NOTE:

When the logistic regression is performed, Python experiences an overflow of memory in the

hx = (1 / (1 + np.exp(-np.dot(x, theta_hat)))) term.

This is because the dot product of x and theta_hat is very large, and as a result, the power of e tends towards negative infinity. When this happens, Python warns the user, but continues running without error because the entire hx term becomes 1 when this happens.

Gaussian Discriminative Analysis (GDA)

In order to apply the GDA model, we had to remove all-zeros columns (otherwise the covariance matrix is singular). Our implementation of the GDA model only allows binary values of 0 and 1 for the labels, so we had to swap the labels (9 => 1) before estimating, then swap back again in the prediction.

We got a pretty good result of 97% accuracy on the MNIST dataset.