Adam Thelen and An Nguyen

EE 425X

Homework 5

4/1/2020

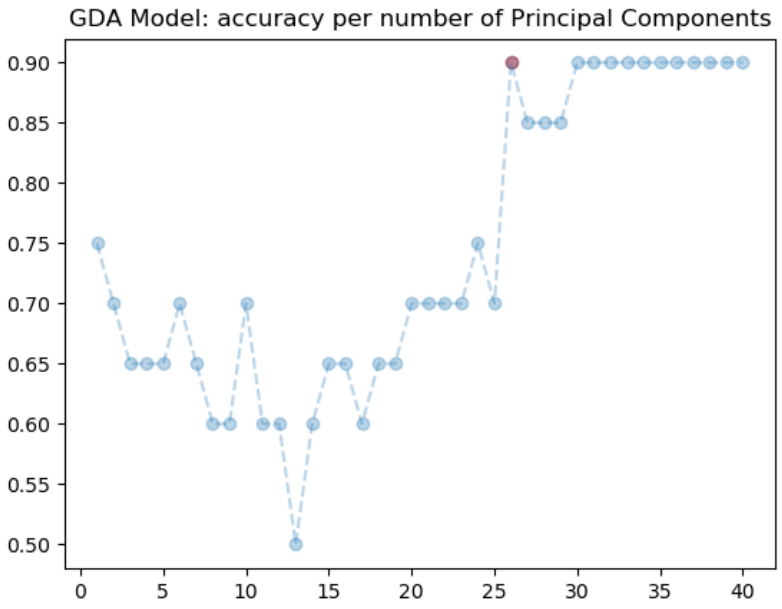
PCA for GDA and Logistic Regression

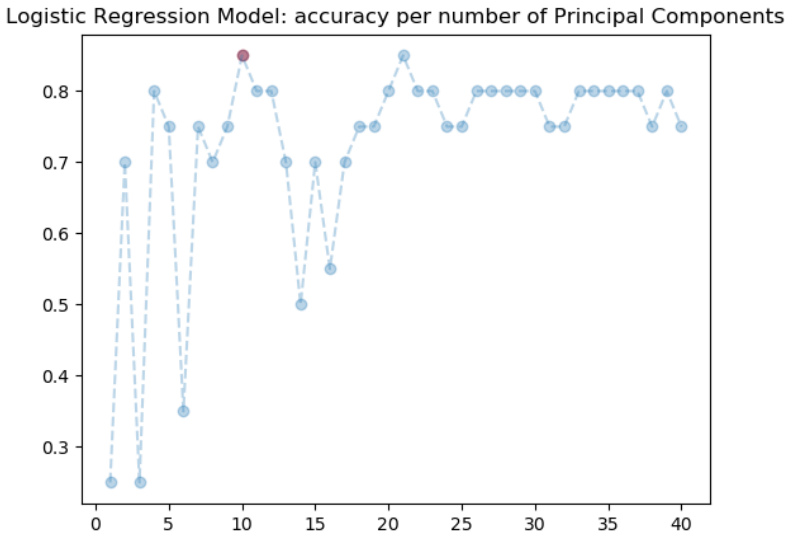
In this investigation, the PCA preprocessing method was applied to both a simulated dataset and the MNIST dataset from previous homework. The MNIST set is first broken down into only the images and labels for zeros and nines. To make the MNIST data smaller from the start, any column with only zeros was removed. After the removal of columns, the MNIST data has 625 features. The PCA preprocessing method is then applied to both the training and testing data. In this experiment, the preprocessed data is run through both a GDA model and a logistic regression model.

1. Simulated Data:

Estimating the GDA model and Logistic Regression Model gives very poor result (75% and 0.55% testing accuracy, respectively). This is expected because the simulated data has multicollinearity (the covariance matrix is not diagonal). This is why we use PCA to project the data into a new subspace where components (features) are orthogonal.

Figure 1 below show the results of the GDA model on simulated data. The simulated data has 100 features and m = 40 data points.





The best numbers of principal components to be included are:

- GDA Model: 26 (90% testing accuracy)
- Logistic Regression: 10 (85% testing accuracy)

We can see that increasing the number of PC can result in worse accuracy. This can be explained by the fact that our models will become more and more biased toward the training data as the number of PCs is increased.

2. Real Data:

   a. Small set of training data (m = 200, n = 446)

      We had to remove all columns whose variances are 0 (otherwise the covariance matrix in GDA model will not be invertible, and those columns are not meaningful for predicting the labels anyway). Therefore, the number of features is less than 784. Nevertheless, it does not affect our analysis for (m < n)

      Estimating parameters of the GDA & Logistic Regression model on the reduced training data set gives us the following accuracy:

      - GDA: 99.8 %
      - Logistic Regression: 99.1 %

      We can see that the accuracy is already very good.

      In the following steps, we use Single Value Decomposition to obtain the PC space V, then increasing the **number of first principal components** used to project the training dataset and record the testing accuracy for each value.

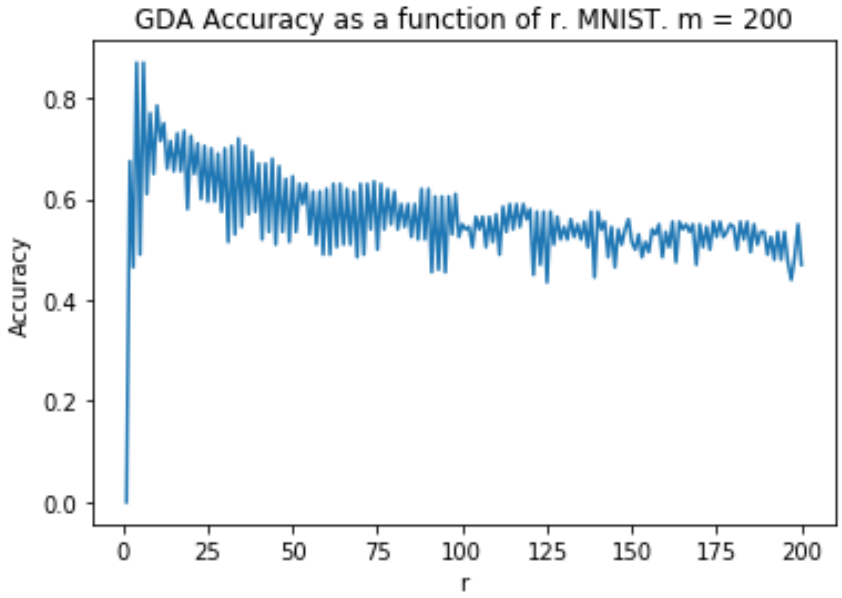


Figure 2 – GDA model testing accuracy per number of Principal Components

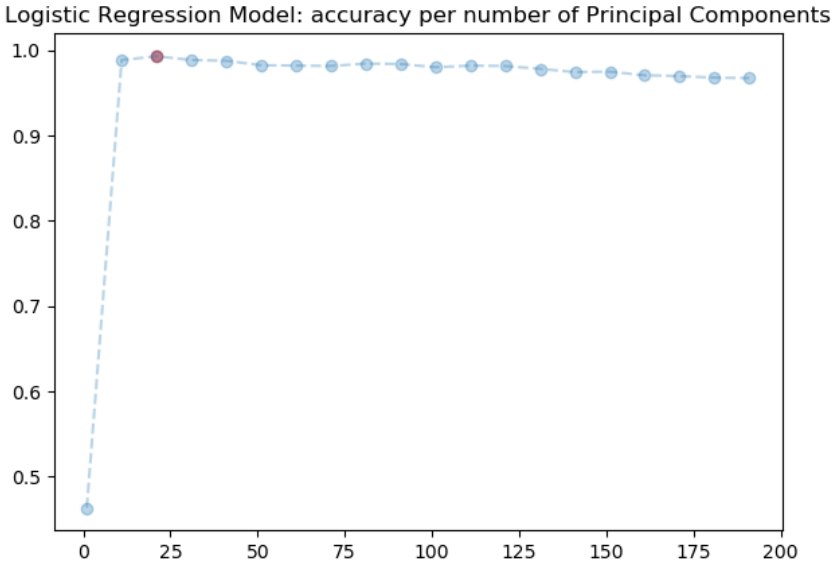


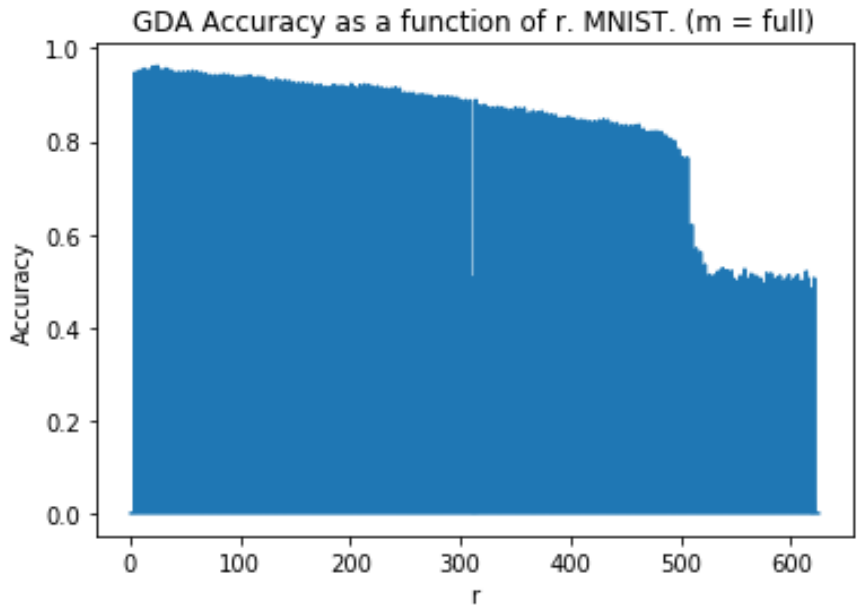
Figure 3 – Logistic Regression model testing accuracy per number of Principal Components

The best numbers of principal components to be included are:

- GDA Model: 31 => accuracy: 0.9967
- Logistic Regression: 21 => accuracy: 0.9929

The result is marginally worse than using the original dataset.

b. Full set of training data:



Looking closer at the GDA model's performance on the MNIST data, it becomes clear that the PCA preprocessing is detrimental to the accuracy of the model. Without PCA preprocessing, the GDA model obtained a maximum accuracy of 0.97. With the PCA preprocessing, the model struggled to obtain this same accuracy. However, the GDA model did come close, peaking at 0.94 at an r value of 30. This goes to show the power that the PCA preprocessing has in helping users to reduce the number of features necessary to get accurate predictions.

Conclusion:

We can see that in the case of our simulated data (where dependent variables are highly correlated), using PCA before training makes a whole lot of difference. But in the case where there are significant more observations than the number of features, and / or multicollinearity is not assumed, PCA doesn't necessarily improves the result.

However, one noticeable advantage is that we no longer have to delete columns whose variances are 0. (since the components are orthogonal and are arranged in descending order w.r.t their corresponding singular values).