The University of Texas at Austin
Department of Electrical and Computer Engineering

EE379K: Data Science Lab — Spring 2017

Lab Four

Caramanis/Dimakis — Due: Tuesday, February 14, 5:00pm 2017.

Submit a Lab report pdf file that shows your name, your lab partner's name, your code, your results and a discussion of your results. Include your Kaggle names, the Team Kaggle name and a screenshot of your public submission score in the pdf of the lab report.

Also submit all your full code in separate files. For this lab, you can submit in either .ipynb format or .py format. If you choose to submit .py files, submit them in the format problemX.py or if you need, problemXa.py, problemXb.py, and so on.

Problem 1: Linear Discriminant Analysis.

1. Generate 20 random points in $d = 3$, from a Gaussian multivariate distribution with mean $[0, 0, 0]$ and covariance matrix $\left(\begin{smallmatrix}1 & 0.9 & 0.9\\ 0.9 & 1 & 0.9\\ 0.9 & 0.9 & 1\end{smallmatrix}\right)$. Let's call this data with label 1. Also generate 20 random points in $d = 3$ from another Gaussian with mean $[0, 0, 1]$ and covariance $\left(\begin{smallmatrix}1 & 0.8 & 0.8\\ 0.8 & 1 & 0.8\\ 0.8 & 0.8 & 1\end{smallmatrix}\right)$. Let's call that data with label 2. Create a three dimensional plot of the clouds of data points, labeled with the two labels.

2. Perform a projection of the data on one dimension using Fischer's Linear Discriminant as explained in class (see also `http://research.cs.tamu.edu/prism/lectures/pr/pr_l10.pdf`). (no sklearn Linear Discriminant Analysis functions here, just friendly linear algebra.)

3. Use sklearn to perform Linear Discriminant Analysis. Compare the results.

Problem 2: Using Low Rank Structure for Corrupted Entries.

Download files `CorrMat1.csv` and `CorrMat3.csv` from Canvas. These are each 100 by 100 matrices. Look at the data and find which entries are corrupted. Then try to correct these corrupted entries. Explain your approach.

Problem 3: Getting more into Kaggle.

1. Goal: Get the best score you can in the Housing prices competition. https://www.kaggle.com/c/house-prices-advanced-regression-techniques/

2. Train a ridge regression and a lasso regression model. Optimize the alphas using cross validation. What is the best scrore you can get from a single ridge regression model and from a single lasso model ?

3. Plot the l_0 norm (number of nonzeros) of the coefficients that lasso produces as you vary alpha.

4. Add the outputs of your models as features and train a ridge regression on all the features plus the model outputs (This is called Ensembling and Stacking). Be careful not to overfit. What score can you get?

5. Install XGBoost (Gradient Boosting) and train a gradient boosting regression. What score can you get just from a single XGB? (you will need to optimize over its parameters).

6. Do your best to win. Try feature engineering and stacking many models. You are allowed to use any public tool in python. No nonpython tools allowed.

7. Read (and post) in the Kaggle forums. Include in your report if you find something in the forums you like, or if you made a post or code, especially if other Kagglers used it afterwards.

8. Be sure you do not violate the rules of Kaggle! No sharing of code or data outside the Kaggle forums. Every student should have their own individual Kaggle account and teams can be formed in the Kaggle submissions with your Lab partner.

9. You will be graded based on your public score (include that in your report) and also on the creativity of your solution. In your report (**that you will submit as a pdf file**), explain what worked and what did not work. Many creative things will not work, but you will get partial credit for developing them. We will invite teams with interesting solutions to present them in class.