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

EE379K: Data Science Lab — Fall 2016

Lab Three

Caramanis/Dimakis Due: Tuesday, September 20, 5:00pm 2016.

Comments/Remarks: This problem set is due on Tuesday, September 20, by 5:00pm. As with the previous labs, we will continue to accept submissions up to 2 hours past this time. For the programming exercises, submit one report per group of two people. The report should include a pdf of your code, your results (plots, or output, as applicable), and also any discussion, again as applicable. Also submit all your code. 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 Algebra in Python. You can use all Python functions to solve this problem.

1. Consider the linear subspace $S = span\{v_1, v_2, v_3, v_4\}$ where $v1 = [1, 2, 3, 4], v2 = [0, 1, 0, 1], v3 = [1, 4, 3, 6], v4 = [2, 11, 6, 15]$. Create a vector inside S different from v_1, v_2, v_3, v_4. Create a vector not in S. How would you check if a new vector is in S?
2. Find the dimension of the subspace S.
3. Find an orthonormal basis for the subspace S.
4. Solve the optimization problem $min_{x \in S}||x - z^*||_2$ where $z^* = [1, 0, 0, 0]$.

Problem 2: PCA.

1. Generate 20 random points in $d = 3$, from a Gaussian multivariate distribution with mean $[0, 0, 0]$ and covariance matrix $\begin{pmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.7 \end{pmatrix}$. Let's call this data with label 1. Also generate 20 random points in $d = 3$ from another Gaussian with mean $[1, 1, 1]$ and covariance $\begin{pmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.01 \end{pmatrix}$. 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. What do the points look like ?
3. Concatenate all the points and ignore the labels for now. You have created an X matrix with 40 data points in $d = 3$ dimensions. Find the covariance matrix of this dataset. **Do not simply use np.cov, build the covariance matrix from the definition using linear algebra operations in python.**

4. Let's do PCA on this dataset using $k = 2$ dimensions: Find the two eigenvectors of the covariance matrix with the largest eigenvalues. Project the data points on these two vectors and show the two dimensional plot with the clouds of points. Also show the labels of the points. Did PCA make it easier to distinguish the two labels in two dimensions ? **Again, do not simply use sklearn PCA. You are only allowed to use matrix operations and np.linalg.eig to find eigenvectors of a matrix.**

Problem 3: Low rank approximation of Mona Lisa.

1. Load the Mona Lisa image (in grayscale) and treat it as a matrix M. Perform a singular value decomposition on this matrix using **linalg.svd**. You can perform a low-rank approximation by zeroing out singular values and keeping only the top k. Show the best rank $k = 2$, $k = 5$ and $k = 10$ approximation to Mona Lisa.

2. If each pixel is represented by two bytes, how many bits is your compressed Mona Lisa for each of those k rank approximations?

Problem 4: Starting in Kaggle.

1. Lets start with our first Kaggle submission in a playground regression competition. Make an account to Kaggle and find https://www.kaggle.com/c/house-prices-advanced-regression-techniques/

2. Follow the data preprocessing steps from https://www.kaggle.com/apapiu/house-prices-advanced-regression-techniques/regularized-linear-models. Then run a ridge regression using $\alpha = 0.1$. Make a submission of this prediction, what is the RMSE you get?
(Hint: remember to exponentiate np.expm1(ypred) your predictions).

3. Try to get to build the best model you can. Report the best RMSE you got on the Kaggle wall and how you got it.