# CMSC5741 Big Data Tech. & Apps.

# Assignment 2

***Due Date: 23:59 Dec.8, 2018***

**Submission Instruction:**

For this assignment, please submit electronic version only. We don't accept hard copy. For the programming questions, you need to submit BOTH your codes and your results. Submit codes as zipped tar file and the output of your program in plain-text file. You should place the relevant files (if any) in their separate directory (preferable 1, 2, 3… for each question). For other questions, answer them in **one word** document.

Please compress all your files as one zip file named by your student id, e.g., 10xxxxxxxx.zip, and submit to cmsc5741@cse.cuhk.edu.hk with email title "CMSC5741 Asg#1, Your name, Your student ID".

Policy on Late Submission:

1). Late within 2 days - 30% deduction.

2). Late after 2 days - not allowed.



## Part A. PCA Programming (20 points)



*The following problem is based on lecture 6.*

Principal component analysis provides a way of creating an optimal low-dimensional representation of a dataset. In this exercise, we will do such a PCA analysis on handwritten digits.



About the dataset (mnist-subset.zip): This is a subset of the MNIST dataset containing just the digits 0,1 and 2. Next, begin with the homework starter source file (named with example.py), which contains basic utilities for reading and displaying MNIST dataset.

1. Write a function to perform PCA on a group of images.
2. Use the PCA function from question 1 to compute the Digit-0-Space. Plot the mean image and then the first 20 eigenvectors (as images). Plot the eigenvalues (in decreasing order) as a function of dimension (for the first 100 dimensions). Describe what you find in both plots.
3. Use the PCA function from question 1 to compute the Digit-2-Space. Plot the mean image and then the first 20 eigenvectors (as images). Plot the eigenvalues (in decreasing order) as a function of dimension (for the first 100 dimensions). Describe what you find in both plots.
4. Use the PCA function from question 1 to compute the Digit-Space. Plot the mean image and then the first 20 eigenvectors (as images). Plot the eigenvalues (in decreasing order) as a function of dimension (for the first 100 dimensions). Compare and contrast what you find in this plots to the ones you created in questions 2 and 3.

For this question, submit your code as zipped tar file. For question 2-3, submit your image in jpg,or png file, submit the description in a plain text.

## Part B. Matrix Factorization (30 points)

*The following problem is based on lecture 7.*

In this question, you will implement a Probabilistic Matrix Factorization algorithm to predict the potential ratings that users would assign to items. Distributed with this file is "pmf_data.zip" which contains two files. The file named "train.dat" contains all the training data. Each line of this file is a tab-separated triplet, of which the first column is the user id, the second column is the movie id and the last column is the rating the user assigned to the movie, e.g. "`AH3QC2PC1VTGP  0000143561 2.0`". The other file named "test.dat" contains all the challenges. Each line of this file is a tab-separated numbers, of which the first is the user id and the second is the movie id, e.g. "`A3R5OBKS7OM2IR0000143502`". Based on the training data given in "train.dat", you are to implement a Probabilistic Matrix Factorization method. Train your model on the training data, and output your predicted ratings for all the challenges, in the exactly same order as the "test.dat". Only output the predicted number (Don't print the user id and movie id; you can output integers or floats), one number each line.

Your score for this question will be based on your RMSE value you can achieve. The definition of RMSE you can find from here: http://en.wikipedia.org/wiki/Root-mean-square_deviation



Your prediction will be compared with the ground truth. So try to be as accurate as possible. **You can modify the PMF algorithm where you see as appropriate**. You will submit two files for this assignment: one file containing the source code of your program and another txt file containing your response to all the challenges.



In this question, you need to concern the efficiency of your code. For naïve implementation, C/C++ and Java are fast enough to finish the training within one day, Python will take several days to finish the training.



## Part C. SVM Programming (25 points)

*The following problem is based on lecture 10.*

You are required to use SVM with stochastic gradient decent method introduced in the lecture to a Google search snippets classification task. Distributed with this file is "svm_data.zip" which contains three files. You will run your code on the given dataset (train.dat) and then conduct prediction on the test dataset (test.dat). You can find more details about the dataset in README. The assessment of your score is your accuracy on the test dataset. (Tips: since there is no label given in the test dataset, you need to create a validation dataset from the training set in order to tune parameters).

Description for the dataset:

There are 8 classes (e.g. business, computers) in the dataset. The number of instances in the training are 10,060 and 2,280 respectively.

The format of the dataset is as follows:

Each line represents an instance, where the last word is the class label, previous words are the search snippets.

Training and testing datasets have the same format except that in the testing dataset the last word (i.e. class label) is missing, which you need to predict.

You will need to preprocess the input data by yourself, and are encouraged to use LIBSVM or other SVM libraries you can get to accomplish the task.

You need to submit your code and a plain text of the results. The format of your result is: each line only contains the class label, e.g. business, sports. And we will mark according to your accuracy on testing dataset.

## Part D. Online Learning Programming (25 points)

*The following problem is based on lecture 11.*

You are required to implement a version of Perceptron algorithm introduced in the lecture. You will run your code on the given dataset (online_data.zip) and then conduct prediction on the test dataset. All the training data can be only processed once, just like a stream input. You need to output:

1. Final precision, recall and F1 score on the testing dataset.
2. ROC curve (as image) on the testing dataset. The definition of ROC curve can be found in https://en.wikipedia.org/wiki/Receiver_operating_characteristic.

Description for the dataset:
There are two classes (e.g. +1 and -1) in the dataset. The number of instances in the training and testing datasets are 29,298 and 3,263 respectively.
The format of the dataset is as follows:
Each line represents an instance, where the first filed is the class label (e.g. +1, -1), followed by non-zero features, each of which is denoted as "feature:value", for example "39:1" represents the value of feature 39 is 1. Training and testing datasets have the same format.
You need to submit your code and a word file containing the results and ROC curve image.