COMS W4721: Machine Learning for Data Science

Columbia University, Spring 2019

Homework 3: Due April 19, 2019 by 11:59pm

Please read these instructions to ensure you receive full credit on your homework. Submit the written portion of your homework as a *single* PDF file through Courseworks (less than 5MB). In addition to your PDF write-up, submit all code written by you in their original extensions through Courseworks (e.g., .m, .r, .py, .c). Any coding language is acceptable, but do not submit notebooks, do not wrap your files in .rar, .zip, .tar and do not submit your write-up in .doc or other file type. Your grade will be based on the contents of one PDF file and the original source code. Additional files will be ignored. We will not run your code, so everything you are asked to show should be put in the PDF file. Show all work for full credit.

Late submission policy: Late homeworks will have 0.1% deducted from the final grade for each minute late. *Your homework submission time will be based on the time of your **last** submission to Courseworks. I will not revert to an earlier submission!* Therefore, do not re-submit after midnight on the due date unless you are confident the new submission is significantly better to overcompensate for the points lost. Submission time is non-negotiable and will be based on the time you submitted your last file to Courseworks. The number of points deducted will be rounded to the nearest integer.

Problem 1 (K-means) – 15 points

Implement the K-means algorithm discussed in class. Generate 500 observations from a mixture of three Gaussians on $\mathbb{R}^2$ with mixing weights $\pi = [0.2, 0.5, 0.3]$ and means μ and covariances Σ,

$$\mu_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \mu_2 = \begin{bmatrix} 3 \\ 0 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \mu_3 = \begin{bmatrix} 0 \\ 3 \end{bmatrix}, \Sigma_3 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

a) For $K = 2, 3, 4, 5$, plot the value of the K-means objective function per iteration for 20 iterations (the algorithm may converge before that).

b) For $K = 3, 5$, plot the 500 data points and indicate the cluster of each for the final iteration by marking it with a color or a symbol.

Problem 2 (Bayes classifier revisited) – 30 points

In this problem, you will implement the EM algorithm for the Gaussian mixture model, with the purpose of using it in a Bayes classifier. The data is a processed version of the spam email data you looked at in Homework 2. Now, each labeled pair (x, y) has $x \in \mathbb{R}^{10}$. We discussed how the Bayes classifier learns class-conditional densities, and unsupervised learning algorithms can be useful here. In this problem, the class conditional density will be the Gaussian mixture model (GMM). In these experiments, please initialize all covariance matrices to the empirical covariance of the data being modeled. Randomly initialize the means by sampling from a single multivariate Gaussian where the parameters are the mean and covariance of the data being modeled. Initialize the mixing weights to be uniform.

a) Implement the EM algorithm for the GMM described in class. Using the training data provided, for each class separately, plot the log marginal objective function for a 3-Gaussian mixture model over 10 different runs and for iterations 5 to 30. There should be two plots, each with 10 curves.

b) Using the best run for each class after 30 iterations, predict the testing data using a Bayes classifier and show the result in a 2×2 confusion matrix, along with the accuracy percentage. Repeat this process for a 1-, 2-, 3- and 4-Gaussian mixture model. *Show all results nearby each other, and don't repeat Part (a) for these other cases.* Note that a 1-Gaussian GMM doesn't require an algorithm, although your implementation will likely still work in this case.

Problem 3 (Matrix factorization) – 30 points

In this problem, you will implement the MAP inference algorithm for the matrix completion problem discussed in class. As a reminder, for users $u \in \mathbb{R}^d$ and movies $v \in \mathbb{R}^d$, we have

$$u_i \sim N(0, \lambda^{-1} I), \quad i = 1, \ldots, N_1, \qquad v_j \sim N(0, \lambda^{-1} I), \quad j = 1, \ldots, N_2.$$

We are given an $N_1 \times N_2$ matrix M with missing values. Given the set $\Omega = \{(i, j) : M_{ij} \text{ is measured}\}$, for each $(i, j) \in \Omega$ we model $M_{ij} \sim N(u_i^T v_j, \sigma^2)$.

You should run your code on the user-movie ratings dataset provided on Courseworks and the course website. For your algorithm, set $\sigma^2 = 0.25$, $d = 10$ and $\lambda = 1$. Train the model on the larger training set for 100 iterations. For each user-movie pair in the test set, predict the rating using the relevant dot product. Note that the mean rating has been subtracted from the data and you do not need to round your prediction. Since the equations are in the slides, there's no need to re-derive it.

a) Run your code 10 times. For each run, initialize your u_i and v_j vectors as $N(0, I)$ random vectors. On a *single* plot, show the the log joint likelihood for iterations 2 to 100 for each run. In a table, show in each row the final value of the training objective function next to the RMSE on the testing set. Sort these rows according to decreasing value of the objective function.

b) For the run with the highest objective value, pick the movies "Star Wars" "My Fair Lady" and "Goodfellas" and for each movie find the 10 closest movies according to Euclidean distance using their respective locations v_j. List the query movie, the ten nearest movies and their distances. A mapping from index to movie is provided with the data.