CSCI4390-6390 Assign7

Assign6: Clustering: EM and Kernel KMeans

Due Date: Dec 2, before midnight (11:59:59PM, Alofi Time; GMT-11)

You will use the Appliances energy prediction data set. You should ignore the first attribute, which is a date-time variable, and you should also remove the last attribute, which is a duplicate of the previous one. Also, the first attribute (after removing the date-time variable), which denotes the **Appliances Energy Use**, will NOT be used for clustering; instead we will use it as the cluster label to assess the performance of the clustering methods. Thus, ignoring "date", "Appliances", and "rv2", the remaining attributes will be used as the data for clustering.

Note that the **Appliances Energy Use** attribute takes values in the range $[10, 1080]$. However, for clustering, we will group these values into 6 "true" clusters: c_1 is for values in the range $[10, 40]$, c_2 for values $[50, 50]$, c_3 for values $[60, 60]$, c_4 for values $[70, 90]$, c_5 for values $[100, 160]$, and c_6 for values $[170, 1080]$. This will result in cluster sizes $|c_1| = 3094$, $|c_2| = 4368$, $|c_3| = 3282$, $|c_4| = 3780$, $|c_5| = 3003$ and $|c_6| = 2208$.

CSCI4390/6390: Expectation Maximization Clustering

Implement the Expectation-Maximization (EM) algorithm for clustering (see Algorithm 13.3 in Chapter 13). Use the 'Appliances' attribute as the true cluster label as described above, and use it for the purity-based clustering evaluation (see below). Run with $k = 6$ clusters.

For initializing the clusters, you can either choose random means as described in the algorithm. Or you can shuffle the data, and pick the first k points as the mean. This has the advantage that the means are always points in the dataset.

For practical purposes, you may want to use the logexpsum trick for expectation step, where you compute the log probabilities so that you can deal with very small probability values, otherwise, you may find that weights of a point for the clusters are zero.

As another practical point, you can get an error when inverting the covariance matrix, so you should add a small ridge value λ value along the diagonal entries to make the matrix invertible. This can be considered as a regularized estimate of the covariance matrix, i.e.,

$$\Sigma + \lambda \mathbf{I}$$

.

Your program output should consist of the following information:

- The final mean for each cluster
- The final covariance matrix for each cluster
- Size of each cluster, after assigning each point to the cluster with highest posterior probability $P(c_i|x_j)$.
- The 'purity score' for your clustering, computed as follows: Assume that c_i denotes the set of points assigned to cluster i by the EM algorithm, and let T_i denote the true cluster id. Purity score is defined as:

  $$\frac{1}{n}\sum_{i=1}^{k} max_{j=1}^{K}\{c_i \cap T_j\}$$

  where K is the true number of clusters, and k is the input number of clusters to EM. See Eq(17.1) on pg 427 for more details on the purity measure.

CSCI6390: Kernel K-Means

Also implement the Kernel K-Means algorithm 13.2 on pg 341. Use only the Gaussian kernel, but you'll have to choose the value of spread.

Your code must output the following information:

- Size of each cluster
- The Purity for your clustering

What to submit

- For EM, write a script named as **Assign7.py**, which will be run as

Assign6.py FILENAME k EPS RIDGE MAXITER

FILENAME is the datafile name, k is the number of clusters to find, and EPS is the convergence threshold, RIDGE the λ value for the ridge, and MAXITER is the maximum number of iterations to run (we need this since it may take a long time to converge for low EPS). Note that you should report the output for $k = 6$, but your code should run for any input value of k. And you must output the purity for the given value of k (which may not correspond to the true value $K = 6$).

- For CSCI6390, for Kernel Kmeans, write a script **Assign7-KK.py**, which will be run as

Assign7-KK.py FILENAME k EPS SPREAD

The parameters have the same meaning as given above, but SPREAD is the spread parameter for the Gaussian kernel.

Note that computing the full kernel matrix for 19K+ points will be memory intensive, so if you do not have enough memory, one option is for you to repeatedly compute the required kernel values. Alternatively, you can show results on at least 5000 points. However, you have to select these points using **stratified sampling**, so that you choose a proportional number of points from each cluster label. You can use StratifiedShuffleSplit from scikit-learn for stratified sampling if you wish.

- Submit a PDF file named Assign7.pdf that should include your output
- (just cut and paste the output from python). **Failure the submit the PDF will result in lost points.**

- Submit the scripts and pdf file via submitty

Policy on Academic Honesty

You are free to discuss how to tackle the assignment, but all coding must be your own. Please do not copy or modify code from anyone else, including code on the web. Any students caught violating the academic honesty principle will get an automatic F grade on the course and will be referred to the dean of students for disciplinary action.