Monsoon 2020 - CSE 642 Advanced Machine Learning
Mid-Sem - Oct. 16, 2020

Maximum score: 50 **Submit by: 11:59AM, Oct. 20, 2020**

Instructions:

- Attempt both questions.
- Keep your derivations clean and to the point. Do not skip any non-trivial steps.
- You may refer to material on the web, textbooks, notes, etc. before beginning your work on the solution. Your solution should be your own work, and your alone. Any evidence of plagiarism will right away fetch you a zero, and a grade reduction. Random viva exams may be conducted for evaluation.
- In the unusual case that a question is not clear, please state your assumptions *clearly* and solve the question. Reasonable assumptions will be accounted for while grading.
- Late submissions will be penalized at 50% of the grade every two hours of delay.

1. (25 points) Kernelization of Linear Methods: Given a set of vectors stacked in a matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$, where $\mathbf{x}_i \in \mathbb{R}^d$. The mean vector for this set of vectors is given as

$$\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$$

a) (5 points) Compute the distance between $\bar{\mathbf{x}}$ and an *in-sample* point $\mathbf{x}_i$ (i.e., for some $i \in [n]$, where $[n]$ is the set of all integers between 1 and n) using only inner products. {*Hint*: Work from first principles, by expanding the squared Euclidean distance between $\mathbf{x}_i$ and $\bar{\mathbf{x}}$. }

b) (5 points) Assume the linear kernel matrix $\mathbf{K}_{lin}$ is generated from the data matrix $\mathbf{X}$, i.e., $\mathbf{K}_{lin} = \mathbf{X}^\top\mathbf{X}$. Write the expression for the squared Euclidean distance between $\mathbf{x}_i$ and $\bar{\mathbf{x}}$, i.e., $d_E^2(\mathbf{x}_i, \bar{\mathbf{x}})$ in terms of $\mathbf{K}_{lin}$.

c) (5 points) The weighted mean of a set of vectors is given by

$$\widetilde{\mathbf{x}} = \frac{\sum_{i=1}^{n} w_i\mathbf{x}_i}{\sum_{i=1}^{n} w_i}, \qquad \text{where } w_i \geq 0 \ \ \forall\ i,\ \ \sum_{i=1}^{n} w_i = 1$$

Write the expression for the squared Euclidean distance between an in-sample point $\mathbf{x}_i$ and $\widetilde{\mathbf{x}}$, i.e., $d_E^2(\mathbf{x}_i, \bar{\mathbf{x}})$ in terms of $\mathbf{K}_{lin}$ and $\mathbf{w}$, the vector of weights w_i.

d) (10 points) Using the above derivations, write the algorithm for the kernelized k-means algorithm, i.e., the k-means algorithm working in the kernel space defined by $\mathbf{K}_{lin}$. The goal of this algorithm is to assign each data point to exactly one of the k clusters. However, all computations should be in terms of $\mathbf{K}_{lin}$ (i.e., inner products) and not directly in the space of $\mathbf{x} \in \mathbb{R}^d$.

e) (Extra credit - 10 points) Write the expression of $d_E^2(\mathbf{y}, \widetilde{\mathbf{x}})$ in part c) for an *out-of-sample* point $\mathbf{y}$, i.e., a point $\mathbf{y}$ is not one of the column vectors of $\mathbf{X}$. Write the kernelized k-means in d) for out-of-sample points.

2. (25 points) Programming: Download the MNIST dataset, with the first 200 samples from classes 0-4. This would give you a total of 1000 samples. Use PCA to perform dimensionality reduction by retaining upto 95% of the total energy (if you are not familiar with this terminology, please google it). We will now evaluate k-means clustering variants for $k = 5$ clusters. Implement the in-sample kernelized k-means algorithm that you have developed in the theory question and code it in python.

a) (10 points) Use the linear kernel $\mathbf{K}_{lin}$ and run your implementation.

b) (10 points) Use the Gaussian kernel $\mathbf{K}_g$ with a suitable scale parameter (picked by cross-validation) and run it through your implementation. If cross-validation is too computationally expensive, you may subsample the dataset.

c) (5 points) Use scikit-learn to run your usual k-means on the dimensionality reduced data you obtained after PCA.

d) Report the clustering metrics (from scikit learn) of Adjusted Rand Index (ARI), Normalized Mutual Information (NMI) and Adjusted Mutual Information (AMI), with the mean and stdev computed over ten runs of each of parts b)-d) above. Use one run to generate the t-SNE plot for each of the cases. You can choose to take the SVD of the kernel matrix to plot the t-SNE plots.

Note on Submission: For the first part, you may do it on paper, which should be scanned (using a flatbed scanner or an equivalent app) to create clear, legible photo(s). Include these in the PDF. For the programming part, put together a PDF report briefly explaining your approach, and the quantitative and qualitative results. In addition, the prepare your entire code as a jupyter notebook and submit it along with your report. Please make sure you document your code and include any additional instructions / dependencies in the report. The evaluation of the programming assignment will be based on the report and the submitted code.