Part 2: Unsupervised Learning (K-means)

Project Overview:

In this part, you are required to implement the k-means algorithm and apply your implementation on the given dataset, which contains a set of 2-D points. You are required to implement two different strategies for choosing the initial cluster centers.

> Strategy 1: randomly pick the initial centers from the given samples.
>
> Strategy 2: pick the first center randomly; for the i-th center (i>1), choose a sample (among all possible samples) such that the average distance of this chosen one to all previous (i-1) centers is maximal.

You need to test your implementation on the given data, with the number k of clusters ranging from 2-10. Plot the objective function value vs. the number of clusters k. Under each strategy, plot the objective function twice, each start from a different initialization.

(Referring to the course notes: When clustering the samples into k clusters/sets D_i, with respective center/mean vectors $\mu_1, \mu_2, \ldots \mu_k$, the objective function is defined as $\sum_{i=1}^{k} \sum_{\mathbf{x} \in D_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2$)

Algorithms:

k-Means Clustering

Resources:

A 2-D dataset to be provided.

Workspace:

Any Python programming environment.

Software:

Python environment.

Language(s):

Python. (MATLAB is equally fine, if you have access to it.)

Required Tasks:

1. Write code to implement the k-means algorithm with Strategy 1.
2. Use your code to do clustering on the given data; compute the objective function as a function of k (k = 2, 3, …, 10).
3. Repeat the above step with another initialization.
4. Write code to implement the k-means algorithm with Strategy 2.
5. Use your code to do clustering on the given data; compute the objective function as a function of k (k = 2, 3, …, 10).
6. Repeat the above step with another initialization.
7. Submit a short report summarizing the results, including the plots for the objective function values under different settings described above.

Optional Tasks:

1. Based on the code you developed above, design a way to choose a k that is optimal. Explain in what sense you view your k as optimal.

Optional tasks are to be explored on your own if you are interested and have extra time for them. No submission is required on the optional tasks. No grading will be done even if you submit any work on the optional tasks. No credit will be assigned to them even if you submit them. (So, please do not submit any work on optional tasks.)

What to Submit:

1. Code file with comments explaining what you do for each part as directed
2. A report that summarizes the results and includes the plots for each of the objective function values.

The code and reports are due at the end of Week 5.

Evaluation criteria:

Working code and correct final results:

Code:

- 4 points - Correctly implements the k-Means algorithm with Strategy 1
- 1 point - Code is used to correctly do clustering on the given data; computes the objective function as a function of k (k - 2, 3, …, 10).
- 4 points - Correctly implements the k-Means algorithm with Strategy 2
- 1 point - Code is used to correctly do clustering on the given data; computes the objective function as a function of k (k - 2, 3, …, 10).

- 4 points - Each of these code parts is correctly documented in comments

Report:

- 2 points - Report summarizes results
- 4 points - The plots for the objective function values under each of the settings described in the code criteria

Algorithm:

Convolutional Neural Network

Resources:

MNIST dataset, Google CoLab

Workspace:

Google CoLab (see file intro_to_colab.docx for more details)

Software:

Google CoLab

Language(s):

Python

Getting Started:

Read this document carefully, as well as additional files included (intro_to_colab.docx and baseline.docx). For more details about Colab, please go to https://colab.research.google.com/notebooks/welcome.ipynb

Required Tasks:

1. Read intro_to_colab.docx to get familiar with the platform.
2. Run the baseline code (as provided) and report the accuracy.
3. Change the kernel size to 5*5, redo the experiment, plot the learning errors along with the epoch, and report the testing error and accuracy on the test set.
4. Change the number of the feature maps in the first and second convolutional layers, redo the experiment, plot the learning errors along with the epoch, and report the testing error and accuracy on the test set.
5. Submit a brief report summarizing the above results, along with your code.

Optional Tasks:

1. Change the kernel size to 9*9 and redo the experiment and report your results.