Week 9 - k-Means Clustering

MACHINE INTELLIGENCE LABORATORY

Teaching Assistants
Sarasharish
Abaksy
Vighnesh kamath

kMeans Clustering is one of the popular unsupervised learning algorithm
In week 9 you are required to code for the class KMeansClustering which will implement kMeans Algorithm.

Your task is to complete the code for the class KMeansClustering and its methods

You are provided with the following files:

1. **Week9.py**
2. **SampleTest.py**

Note: These sample test cases are just for your reference.

SAMPLE TEST CASE Data

This is a randomly generated test case and can be analyzed by opening sampleTest.py

Important Points:

1. Please do not make changes to the function definitions that are provided to you. Use the skeleton as it has been given. Also do not make changes to the sample test file provided to you. Run it as it is.
2. You are free to write any helper functions that can be called in any of these predefined functions given to you. Helper functions must be only in the file named 'YOUR_SRN.py'.
3. Your code will be auto evaluated by our testing script and our dataset and test cases will not be revealed. Please ensure you take care of all edge cases!
4. **The experiment is subject to zero tolerance for plagiarism. Your code will be tested for plagiarism against every code of all the sections and if found plagiarized both the receiver and provider will get zero marks without any scope for explanation.**
5. **Kindly do not change variable names or use any other techniques to escape from plagiarism, as the plagiarism checker is able to catch such plagiarism**
6. Hidden test cases will not be revealed post evaluation.

week4 .py

- √ You are provided with structure of class KMeansClustering
- √ The class AdaBoost contains one constructor and 5 methods out of which two is already written.
- √ Your task is to write code for the three of these 5 methods.
- √ Do not make any changes to the already written methods

def__init__(self, n_clusters, n_init=10, max_iter=1000, delta=0.001):

- ■ Initializes max_iteration, number of clusters , the delta value and n_init number of times to initilize centroid and start over def init_centroids(self, data):

- Already implemented
- Initialises the centroid

def fit(self, data):

- Fits the model to the training dataset.
- Already implemented

def e_step(self, data):

- Expectation Step.
- Finding the cluster assignments of all the points in the data passed based on the current centroids.

def m_step(self, data, cluster_assgn):

- Maximization Step.
- Compute the centroids

def evaluate(self, data):

- K-Means Objective
- Returns metric value

1. **You may write your own helper functions if needed**
2. **You can import libraries that come built-in with python 3.7**
3. **You cannot change the skeleton of the code**
4. **Note that the target value is an int**

SampleTest.py

1. This will help you check your code.
2. Make sure you have installed numpy and sklearn
3. Passing the cases in this does not ensure full marks, you will need to take care of edge cases
4. Name your code file as YOUR_SRN.py
5. Run the command

   python3 SampleTest.py --SRN YOUR_SRN

   if import error occurs due to any libraries that is mentioned in the skeleton code try:

   python3.7 SampleTest.py --SRN YOUR_SRN