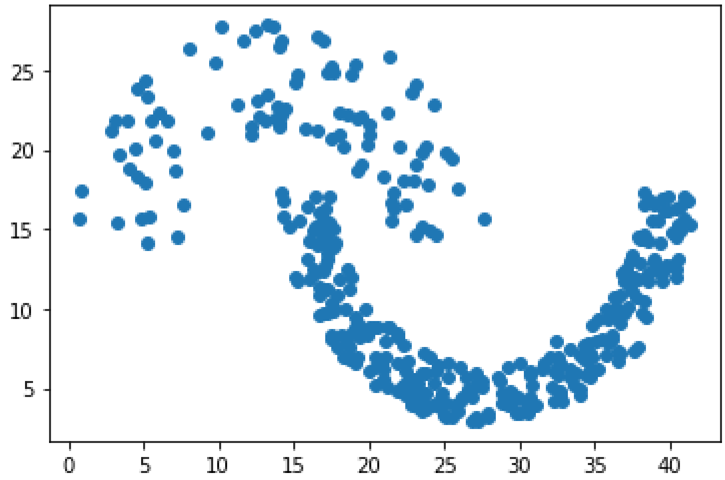
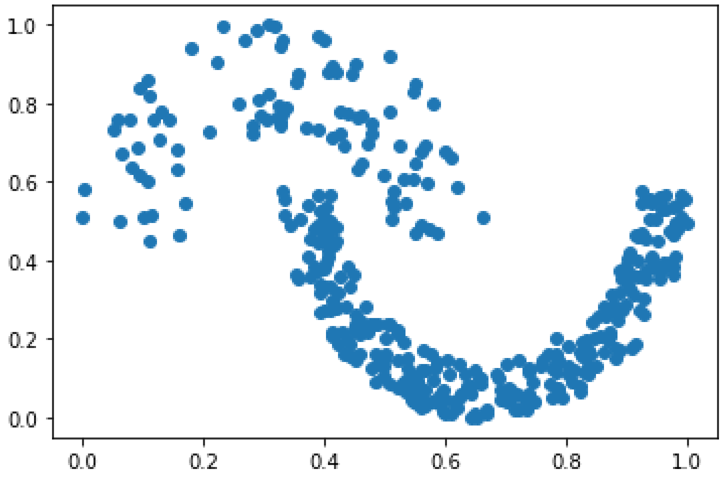
Question 1

I imported the dataset from the given link http://cs.joensuu.fi/sipu/datasets/jain.txt using NumPy. This dataset contains features as well as labels. As K-means clustering only required features. So, I split the dataset in label and features

After plotting the features using scatter plot from Matplotlib I got the below graph



Range of the data is too high, so I used min-max scaler to scale the data points. And got the below graph after scaling the feature points using min-max scaler and plotting them using matplotlib



To perform the k-means clustering I created the class k-means which contains the below methods

__init__ :

This constructor initializes the class variables

distance(self, p1, p2):

This method takes class object and 2 feature vectors as points as arguments and returns Euclidian distance between them.

create_cluster(self,centroid):

This method takes class object centroids of the feature vectors and feature vectors as class variable and returns the vectors separated as cluster as tuples

nearest_centroid(self,centroid):

This method takes class object, centroids, and feature vectors as class variables and return nearest centroid for each point as a NumPy array

update_centroid(self,cluster_0,cluster_1):

This method takes class object, clusters, and features as class variable. Update the clusters as mean of the features for each cluster and returns the clusters as a list

check_covergence(self,old_cent,centroid):

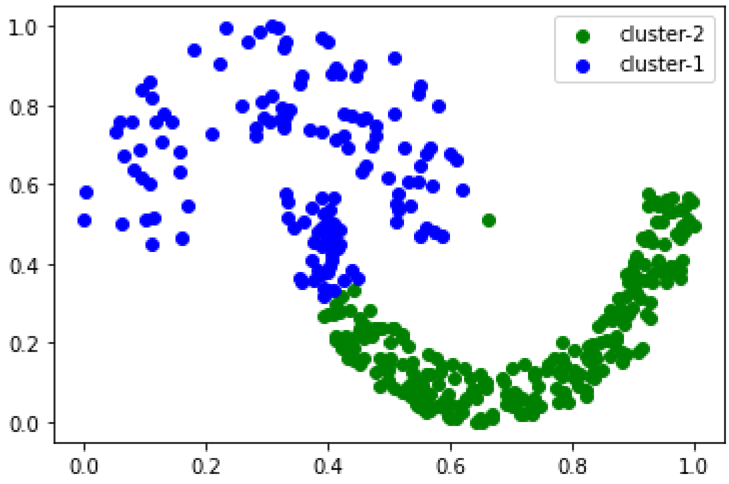
This method takes class object old centroids and new centroid as arguments.

It returns true if there are no changes in centroid points otherwise returns false

predict_cluster(self,x):

This method takes class object and feature vectors as arguments. Initially takes arbitrary points as centroids and create cluster using create_cluster method, updates its centroid using update_centroid methods, check if convergence condition has met. Otherwise continues the process iteratively till 300 iterations. And finally returns feature points as cluster.

I got the cluster points with the above class methods with the features and plotting those points as scatter plot, I got the below graph



On validating the cluster points with labelled data, I found the accuracy as below

```
Accuracy for clustering is:88.20 %
```

For spectral clustering I 1st did the scaling of the features

And implemented spectral clustering with the Spectral class which contains the below classes

distance(self, p1, p2):

This method takes class object and 2 feature vectors as points as arguments and returns Euclidian distance between them.

laplacian(self,x):

This method takes class object and feature vectors and returns the Laplacian matrix as distance between each point considering each row of feature vectors as a points

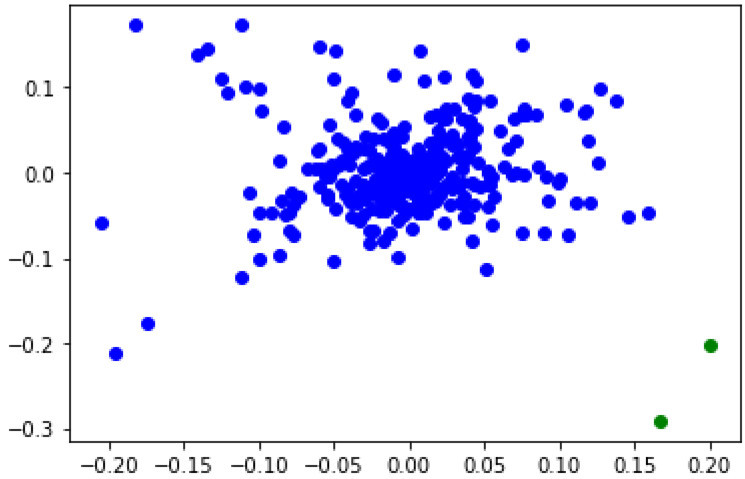
adjacency(self,lap):

This method takes class object and laplacian matrix as arguments and returns adjacency matrix considering a value greater than a threshold as adjacency.

Got the laplacian and adjacency matrix using the above class methods and we found the eigen vectors and eigen values using eig method and argsorted them using argsort method from numpy

Taken least two rows as our feature vectors and applied k-means on those feature vectors.

I found the cluster points from the above operation and on plotting them I got the below graph.



<u>Question 2</u>

I imported the data from the given link https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data . Which contains features and labels.

From which I extracted features and labels and replaced unique classes as unique integers

Performed PCA transformation using PCA class which contains the below methods

fit(self,features):

This method takes arguments as class object and features vectors

Calculate covariance matrix from the feature vectors,

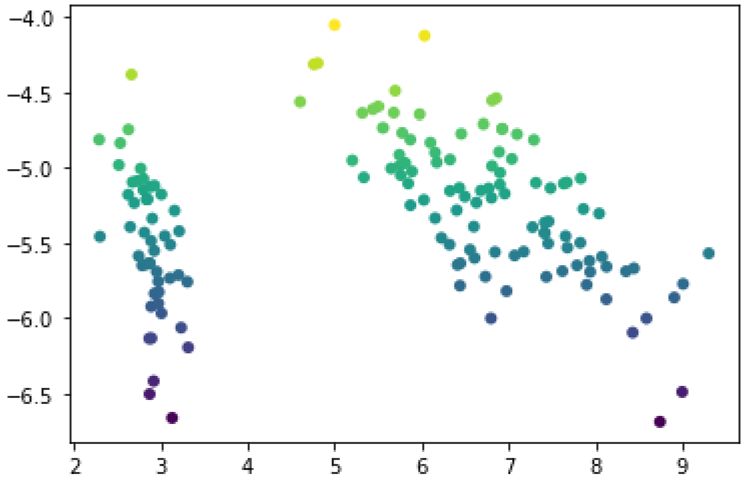
Eign values ans eigen vectros using eig method from numpy from the covariance matrix.

Set class variables as eign components as 1st 2 columns of transpose matrix of the sorted eign vectors as reverse order

transform(self,features):

This method takes class object, features as argument, and returns dot product of feature vector and eign vector components of class variable

I got the below graph on plotting the points as scattered plot after dimensionality reduction


−4.0
−4.5
−5.0
−5.5
−6.0
−6.5
2
3
4
5
6
7
8
9