HELP MANUAL

DDCN-2019@CSED, MNNIT Allahabad

Clustering using k-Means and MeanShift Algorithm

k-Means Clustering on Lists

- **Required packages to implement k-Means Clustering algorithm on Lists**
  - import numpy as np
  - import matplotlib.pyplot as plt
  - from sklearn.cluster import KMeans

k-Means Clustering on Lists Contd...

- **Creating Lists and displaying them**
  - x=[1,5,1.5,8,1,9]
  - y=[2,8,1.8,8,0.6,11]
  - print (x)
  - print (y)
- **Plot and display scatter chart of x and y**
  - plt.scatter(x,y)
  - plt.show()
- **Creating an array X which stores pair (x, y)**
  - X=np.array([[1,2],[5,8],[1.5,1.8],[8,8],[1,0.6],[9,11]])

k-Means Clustering on Lists Contd…

- **Apply KMeans function with two number of clusters and store its output in variable kmeans, representing a clustering model**
  - kmeans=KMeans(n_clusters=2)
- **Fit kmeans clustering model on array X.**
  - kmeans.fit(X)
- **Extract centroids and labels from the model kmeans and print them on console.**
- centroids=kmeans.cluster_centers_
- print(centroids)
- labels=kmeans.labels_
- print (labels)

k-Means Clustering on Lists Contd…

- **Define color list having two different colors red and green to represent two clusters.**
  - colors=["r.","g."]
- **For each element of the array X**
  - **Print coordinates and labels along with the element of X**
  - **Plot each element of X using colors and labels**
- Solution:
  - for i in range(len(X)):
  - print ("coordinate:",X[i],"labels:",labels[i])
  - plt.plot(X[i][0],X[i][1],colors[labels[i]])

k-Means Clustering on Lists Contd…

- **Plot centroids of both clusters**
  - plt.scatter(centroids[:,0],centroids[:,1],marker="x",s=150)
- **Display scatter chart showing all elements of X with designed clusters in specified colors.**
- plt.show()
- **Import "pandas" package" and copy dataset "faithful.csv" to the Destop folder on your system**
  - import pandas as pd

k-Means Clustering on Lists Contd…

- **Open dataset file "faithful.csv" and store it a variable "d"**
  - d=pd.read_csv('c:/users/username/Desktop/DDCN-2019/faithful.csv')
- **Print first five records of the variable "d"**
  - print (d.head())
- **Plot scatter chart of columns "eruptions" and "waiting" of the variable "d"**
  - plt.scatter(d.eruptions,d.waiting)

k-Means Clustering on Lists Contd…

- **Show scatter chart with chart title as "Old Faithfull Data Scatter Plot", x axis as "Length of eruptions" , and y axis as "Time between eruptions" .**
  - plt.title('Old Faithfull Data Scatter Plot')
  - plt.xlabel('Length of eruptions')
  - plt.ylabel('Time between eruptions')
  - plt.show()

k-Means Clustering on Lists Contd…

- **Create an array "d1" which stores the elements of the variable "d".**
  - d1=np.array(d)
- **Apply KMeans function with two number of clusters and store its output in variable kmeans, representing a clustering model.**
  - k=2
  - kmeans=cluster.KMeans(n_clusters=k)
-

k-Means Clustering on Lists Contd…

- **Fit kmeans clustering model on array "d1".**
  - kmeans.fit(d1)
- **Extract centroids and labels from the model kmeans.**
  - labels=kmeans.labels_
  - centroids=kmeans.cluster_centers_

k-Means Clustering on Lists Contd…

- **For each element of the array "d1"**
  - **Extract observations for each level from the array "d1" and store it in variable "ds"**
  - **Plot both columns from the variable ds and centroids for each cluster**
  - **Increase the size of centroid points**
- **Solution:**
  - for i in range (k):
  - ds=d1[np.where(labels==i)]
  - plt.plot(ds[:,0],ds[:,1],'o',markersize=7)
  - lines=plt.plot(centroids[i,0],centroids[i,1],'kx')
  - plt.setp(lines,ms=15.0)
  - plt.setp(lines,mew=4.0)

k-Means Clustering on Lists Contd…

- **Display scatter chart showing all elements of the datasets with designated clusters and centroids.**
  - plt.show()

Hierarchical Clustering using MeanShift Algorithm

Clustering using MeanShift Algorithm

- **Required packages to implement Hierachichal Clustering using MeanShift algorithm**
  - import numpy as np
  - import matplotlib.pyplot as plt
  - from sklearn.cluster import MeanShift
- **Import packages to generate sample data**
  - from sklearn.datasets.samples_generator import make_blobs

Clustering using MeanShift Algorithm

- **Define center points as** [1,1],[5,5]
  - centers=[[1,1],[5,5]]
- **Generate sample of data sets and store it in X,Y**
  - X,Y=make_blobs(n_samples=200,centers=centers, cluster_std=1)
- **Display scatter chart of generated sample data (X)**
  - plt.scatter(X[:,0],X[:,1])
  - plt.show()

Clustering using MeanShift Algorithm

- **Apply MeanShift function and store its output in variable kmeans, representing a clustering model.**
  - ms=MeanShift()
- **Fit generated clustering model on the data X.**
  - ms.fit(X)
- **Extract centroids and labels from the cluster model.**
- # Extracting labels
- labels=ms.labels_
- # Extracting cluster centres
- clusters_centers=ms.cluster_centers_

Clustering using MeanShift Algorithm

- **Extract number of clusters from the cluster model and print the number of clusters on console.**
  - n_clusters=len(np.unique(labels))
  - print (" Number of Estimated Clusters",n_clusters)
- **Define list of colors**
  - colors=["g.","r.","c.","y.","b.","k","y.","m."]
- **Print colors and labels**
- print(colors)
- print(labels)

Clustering using MeanShift Algorithm

- **For each observation of data X**
  - **Plot observations with suitable colors according to the designated labels**
- Solution:
  - for i in range(len(X)):
  - #print ("coordinate:",X[i],"labels:",labels[i])
  - plt.plot(X[i][0],X[i][1],colors[labels[i]],markersize=10)

Clustering using MeanShift Algorithm

- **Plot scatter chart of cluster centers and mark them with "x"**
  - plt.scatter(clusters_centers[:,0],clusters_centers[:,1],marker="x",s=150,linewidth=5, zorder=10)
- **Display scatter chart**
  - plt.show()