**LAB - 4**

**KMEANS CLUSTERING**

**PRIYANSHU SHARMA**

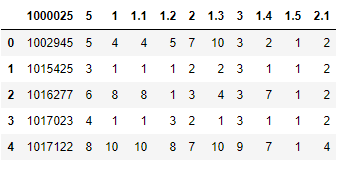
**15BCE1282**

**CODE**

import pandas as pd

data = pd.read\_csv('C:/Users/PRIYANSHU SHARMA/Desktop/PRIYANSHU/6 STUDY/6 SEMSTER/MACHINE LEARNING/LAB/breast.csv')

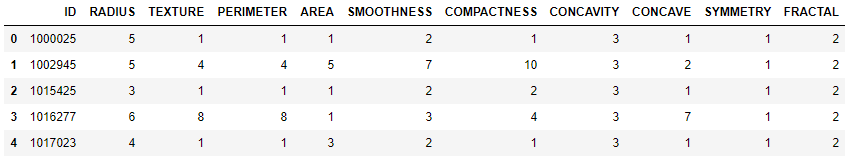
data.head()



colnames=['ID', 'RADIUS', 'TEXTURE', 'PERIMETER', 'AREA', 'SMOOTHNESS', 'COMPACTNESS', 'CONCAVITY', 'CONCAVE', 'SYMMETRY', 'FRACTAL']

data = pd.read\_csv('C:/Users/PRIYANSHU SHARMA/Desktop/PRIYANSHU/6 STUDY/6 SEMSTER/MACHINE LEARNING/LAB/breast.csv', names=colnames, header=None)

data.head()



import matplotlib.pyplot as plt

import seaborn as sb

sb.pairplot(data,vars=colnames,hue='FRACTAL')

X, y = data.iloc[:, [2, 6]].values, data['FRACTAL']

from sklearn.preprocessing import StandardScaler

sc=StandardScaler()

X\_std=sc.fit\_transform(X)

X\_std

plt.scatter(X\_std[:,0],X\_std[:,1],c='blue',marker='o')

plt.grid()

plt.show()



from sklearn.cluster import KMeans

km = KMeans(n\_clusters=2,init='random',n\_init=100,max\_iter=30,tol=1e-04,random\_state=0)

y\_km = km.fit\_predict(X\_std)

plt.scatter(X\_std[y\_km==0,0],X\_std[y\_km ==0,1],c='lightgreen',marker='s',label='cluster 1')

plt.scatter(X\_std[y\_km ==1,0],X\_std[y\_km ==1,1],c='orange',marker='o',label='cluster 2')

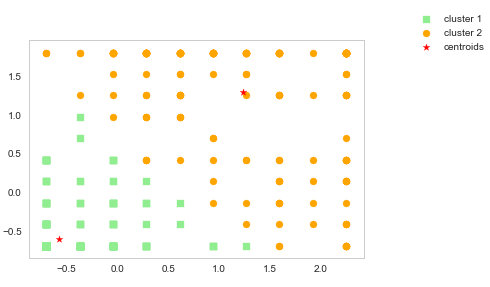
plt.scatter(km.cluster\_centers\_[:,0],km.cluster\_centers\_[:,1],marker='\*',c='red',label='centroids')

plt.legend(bbox\_to\_anchor=(1, 1),

bbox\_transform=plt.gcf().transFigure)

plt.grid()

plt.show()



print('Distortion: %.2f' % km.inertia\_)



distortions = []

for i in range(1, 11):

km = KMeans(n\_clusters=i,init='k-means++',n\_init=10,max\_iter=300,random\_state=0)

km.fit(X)

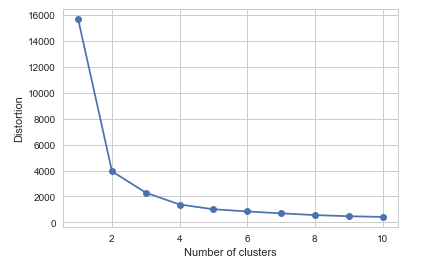
distortions.append(km.inertia\_)

plt.plot(range(1,11), distortions, marker='o')

plt.xlabel('Number of clusters')

plt.ylabel('Distortion')

plt.show()



from matplotlib import cm

from sklearn.metrics import silhouette\_samples

import numpy as np

cluster\_labels = np.unique(y\_km)

n\_clusters = cluster\_labels.shape[0]

silhouette\_vals = silhouette\_samples(X\_std,y\_km,metric='euclidean')

y\_ax\_lower, y\_ax\_upper = 0, 0

yticks = []

for i, c in enumerate(cluster\_labels):

c\_silhouette\_vals = silhouette\_vals[y\_km == c]

c\_silhouette\_vals.sort()

y\_ax\_upper += len(c\_silhouette\_vals)

color = cm.jet(i / n\_clusters)

plt.barh(range(y\_ax\_lower, y\_ax\_upper),

c\_silhouette\_vals,

height=1.0,

edgecolor='none',

color=color)

yticks.append((y\_ax\_lower + y\_ax\_upper) / 2)

y\_ax\_lower += len(c\_silhouette\_vals)

silhouette\_avg = np.mean(silhouette\_vals)

plt.axvline(silhouette\_avg,color="red",linestyle="--")

plt.yticks(yticks, cluster\_labels + 1)

plt.ylabel('Cluster')

plt.xlabel('Silhouette coefficient')

plt.show()

