# Sorry it is in this format, Math Wiki won’t accept uploading python files.

**import** numpy **as** np  
**import** matplotlib.pyplot **as** plt  
**from** scipy.spatial.distance **import** cdist  
  
**from** sklearn.cluster **import** KMeans, AgglomerativeClustering, SpectralClustering  
**from** sklearn.mixture **import** GaussianMixture  
**import** skfuzzy **as** fuzz  
  
**from** sklearn **import** datasets  
**import** os  
  
  
**def** make\_data(n\_samples, dataset\_type):  
 **if** dataset\_type == **"circles"**:  
 datset = datasets.make\_circles(n\_samples, noise=0.05, factor=0.5)  
 **elif** dataset\_type == **"moons"**:  
 datset = datasets.make\_moons(n\_samples, noise=0.05)  
 **else**:  
 datset = datasets.make\_blobs(n\_samples)  
 **return** datset  
  
  
**def** plot\_raw\_data(ds):  
 x, y = ds  
 plt.plot(x[:,0], x[:, 1], **'bo'**)  
 plt.grid(**True**)  
 plt.show()  
  
  
**def** cluster\_plot(ds, method):  
 x, y = ds  
 p = x.shape[0]  
 colors = [np.random.rand(3) **for** g **in** range(10)]  
 **if** method == **"Kmeans"**:  
 plt.figure()  
 kmeans = KMeans(n\_clusters=3, n\_init=10, tol=0.001).fit(x)  
 **for** lab **in** range(kmeans.n\_clusters):  
 plt.plot([x[i][0] **for** i **in** range(p) **if** kmeans.labels\_[i] == lab],  
 [x[i][1] **for** i **in** range(p) **if** kmeans.labels\_[i] == lab], **"."**, color=colors[lab])  
 plt.title(**'K-Means Clustering'**)  
 plt.show()  
 **elif** method == **"Hierarchic"**:  
 plt.figure()  
 h\_clus = AgglomerativeClustering(n\_clusters=4, linkage=**'single'**).fit(x)  
 **for** lab **in** range(h\_clus.n\_clusters):  
 plt.plot([x[i][0] **for** i **in** range(p) **if** h\_clus.labels\_[i] == lab],  
 [x[i][1] **for** i **in** range(p) **if** h\_clus.labels\_[i] == lab], **"."**, color=colors[lab])  
 plt.title(**'Hierarchical Clustering'**)  
 plt.show()  
 **elif** method == **"Spectral"**:  
 plt.figure()  
 h\_clus = SpectralClustering(n\_clusters=2).fit(x)  
 **for** lab **in** range(h\_clus.n\_clusters):  
 plt.plot([x[i][0] **for** i **in** range(p) **if** h\_clus.labels\_[i] == lab],  
 [x[i][1] **for** i **in** range(p) **if** h\_clus.labels\_[i] == lab], **"."**, color=colors[lab])  
 plt.title(**'Spectral Clustering'**)  
 plt.show()  
 **elif** method == **"Fuzzy"**:  
 plt.figure()  
 cntr, u, u0, d, jm, p, fpc = fuzz.cluster.cmeans(np.transpose(x), c=2, m=0.1,  
 error=0.001, maxiter=100)  
 fcm\_labels = np.argmax(u, axis=0)  
 **for** lab **in** range(max(fcm\_labels) + 1):  
 plt.plot([x[i][0] **for** i **in** range(p) **if** fcm\_labels[i] == lab],  
 [x[i][1] **for** i **in** range(p) **if** fcm\_labels[i] == lab], **"."**, color=colors[lab])  
 plt.title(**'Fuzzy c-Means clustering'**)  
 plt.show()  
 **else**:  
 plt.figure()  
 gmm = GaussianMixture(n\_components=2, n\_init=10, covariance\_type=**'diag'**, init\_params=**'random'**).fit(x)  
 gmm\_labels = gmm.predict(x)  
 **for** lab **in** range(max(gmm\_labels) + 1):  
 plt.plot([x[i][0] **for** i **in** range(p) **if** gmm\_labels[i] == lab],  
 [x[i][1] **for** i **in** range(p) **if** gmm\_labels[i] == lab], **"."**, color=colors[lab])  
 plt.title(**'GMM clustering'**)  
 plt.show()  
  
  
**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 data = make\_data(10000, **'blobs'**)  
 plot\_raw\_data(data)  
 cluster\_plot(data, **'Kmeans'**)