Lab 10. Clustering using K-means and Gaussian Mixture Model (GMM)

You are given with "`2D_points2.txt`" file containing 2D points belonging to 4 classes. Write a program to do the following.

1. Apply K-means (K=4) clustering on the data. Plot the data points in these clusters (use different colors for each cluster). Obtain the sum of squared distances of samples to their closest cluster center.
(Use **`model.fit`** to train the model and **`model.labels_`** to obtain the cluster labels. Use **`model.inertia_`** to get sum of squared distances).

2. Build a GMM with 4 components (use **`GMM.fit`**) on the original data. Use this GMM to cluster the data points (Use **`GMM.predict`**). Plot the points in these clusters using different colors.

3. Find the optimum number of clusters in both the methods using Elbow method.

4. Obtain the homogeneity score for both type of clustering.

5. Repeat part 1 and 2 with different number of clusters. (K= 2,3,5,10)

6. Obtain the purity score for both K-means and GMM (For K=4 only) using the code snippet given below.

7. Perform the ***K*-medoids** clustering with K=4 on the original data and compare the results.

##

```
# Purity score calculation:
# For obtaining the true labels, assume that the first 500 points
belong to class 0, 501-1000 belongs to class 1, 1001-1500 belong
to class 2 and last 500 point belong to class 3 )
from sklearn import metrics
def purity_score(y_true, y_pred):
    # compute contingency matrix (also called confusion matrix)
    contingency_matrix =
metrics.cluster.contingency_matrix(y_true, y_pred)
    # return purity
    return np.sum(np.amax(contingency_matrix, axis=0)) /
np.sum(contingency_matrix)
```