K-MEANS CHEAT SHEET

Loading Data from sklearn

Import sklearn datasets and the specific data you want to use	`from sklearn.datasets import load_data` `data_name = load_data()`

Turn that data into a pandas DataFrame	`import pandas as pd`
	`df = pd.DataFrame(data_name.data,`
	`columns=data_name.feature_names)`

Create X

You choose what features of the data frame to include for your clusters.	`X = df[["column_name1", "column_name2"]]`

Creating your model

Import your model and then initialize it. In this case, `name` is the name of the model. This is helpful for using the model to predict.	`from sklearn.cluster import KMeans` `name = KMeans(n_clusters = K)`
Use your defined X to create the prediction. y will store the data for the clusters.	`y = name.fit_predict(X)`

Evaluating with Silhouette Score

Silhouette score evaluates on a range of -1 to 1, where 1 is optimal. 0 represents overlapping clusters, and negative values indicate a wrong assignment for data in clusters. Remember, `name` represents the name of your model.	`from sklearn.metrics import silhoutte_score`
	`score = silhoutte_score(X, name.labels_,`
	`metric='euclidean'))`
	`print(score)`

Predicting

Given a data point, this will tell you which cluster said point would be assigned to. Remember, `name` represents the name of your model.	`x_to_predict = [[.],[.]] # 2D array`
	`prediction = name.predict(x_to_predict)`
	`print(prediction)`

Example for homework 3.1	`x_to_predict = [[6,2], [6,6], [6,5], [7,6],`
	`[5,3]] # must be a 2D array`
	`prediction = kmeans1.predict(x_to_predict)`
	`print(prediction)`

(Difficult) Visualizing for two-dimensional clusters–More features requires advanced visualizations

```
plt.scatter(X[y == 0, 0], X[y == 0, 1], s = 50, c = 'red', label = '0')
plt.scatter(X[y == 1, 0], X[y == 1, 1], s = 50, c = 'blue', label = '1')
plt.scatter(X[y == 2, 0], X[y == 2, 1], s = 50, c = 'green', label = '2')
# include more or less of these lines depending on number of clusters

plt.scatter(name.cluster_centers_[:, 0],   # name is the name of the model
            name.cluster_centers_[:, 1],   # name is the name of the model
            s = 100, c = 'yellow', label = 'Centroids', marker = "*")

plt.title('title')
plt.xlabel('x label')
plt.ylabel('y label')
plt.legend()
plt.show()
```

* Remember, `name` represents the name of your model.