uOttawa

Faculté de génie
Faculty of Engineering

Applied Machine Learning - Summer 2021

Assignment 3 - Clustering

Submitted by:

Name	Email
Youssef Metwally	ymetw027@uOttawa.ca
Dina Ibrahim Morsy	dabde007@uOttawa.ca
Adel El Nabarawy	aelna025@uOttawa.ca

Submitted to:

Dr. Murat SIMSEK, SMIEEE

Numerical Questions

Q1 (K-means):

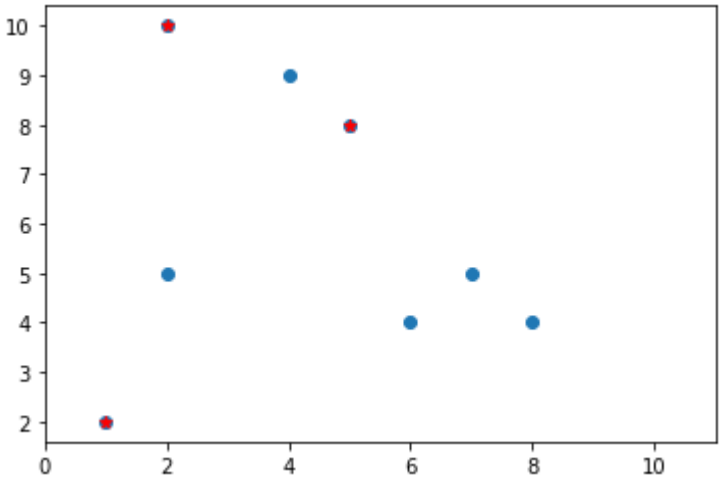
A1 = (2, 10), A2 = (2, 5), A3 = (8, 4), A4 = (5, 8), A5 = (7, 5), A6 = (6, 4), A7 = (1, 2), A8 = (4, 9)

Euclidean distance : $\sqrt{(x_{cluster} - x_{point})^2 + (y_{cluster} - y_{point})^2}$

ITERATION 1:

Cluster 1 : (2,10)	Cluster 2 : (5,8)	Cluster 3 : (1,2)
$\sqrt{(2-2)^2 + (10-10)^2}$	$\sqrt{(5-2)^2 + (8-10)^2}$	$\sqrt{(1-2)^2 + (2-10)^2}$
$\sqrt{(2-2)^2 + (10-5)^2}$	$\sqrt{(5-2)^2 + (8-5)^2}$	$\sqrt{(1-2)^2 + (2-5)^2}$
$\sqrt{(2-8)^2 + (10-4)^2}$	$\sqrt{(5-8)^2 + (8-4)^2}$	$\sqrt{(1-8)^2 + (2-4)^2}$
$\sqrt{(2-5)^2 + (10-8)^2}$	$\sqrt{(5-5)^2 + (8-8)^2}$	$\sqrt{(1-5)^2 + (2-8)^2}$
$\sqrt{(2-7)^2 + (10-5)^2}$	$\sqrt{(5-7)^2 + (8-5)^2}$	$\sqrt{(1-7)^2 + (2-5)^2}$
$\sqrt{(2-6)^2 + (10-4)^2}$	$\sqrt{(5-6)^2 + (8-4)^2}$	$\sqrt{(1-6)^2 + (2-4)^2}$
$\sqrt{(2-1)^2 + (10-2)^2}$	$\sqrt{(5-1)^2 + (10-2)^2}$	$\sqrt{(1-1)^2 + (2-2)^2}$
$\sqrt{(2-4)^2 + (10-9)^2}$	$\sqrt{(5-4)^2 + (10-9)^2}$	$\sqrt{(1-4)^2 + (2-9)^2}$

	Cluster 1 : (2, 10)	Cluster 2 : (5, 8)	Cluster 3 : (1, 2)
A1	0	$\sqrt{13}$	$\sqrt{65}$
A2	$\sqrt{25}$	$\sqrt{18}$	$\sqrt{10}$
A3	$\sqrt{36}$	$\sqrt{25}$	$\sqrt{53}$
A4	$\sqrt{13}$	0	$\sqrt{52}$
A5	$\sqrt{50}$	$\sqrt{13}$	$\sqrt{45}$
A6	$\sqrt{52}$	$\sqrt{17}$	$\sqrt{29}$
A7	$\sqrt{65}$	$\sqrt{52}$	0
A8	$\sqrt{5}$	$\sqrt{2}$	$\sqrt{58}$



New labeled data points:

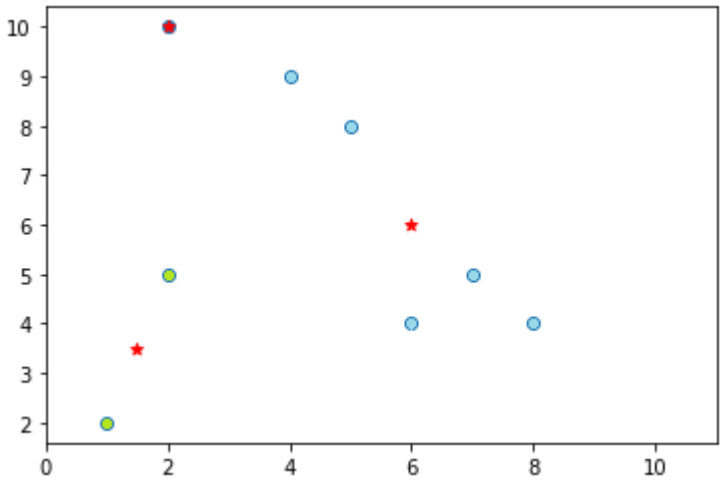
1: A1 , **2:** A3, A4, A5, A6, A8 , **3:** A2, A7

New clusters' centroids:

1: (2, 10)

2: ((8+5+7+6+4)/5,(4+8+5+4+9)/5) = (6, 6)

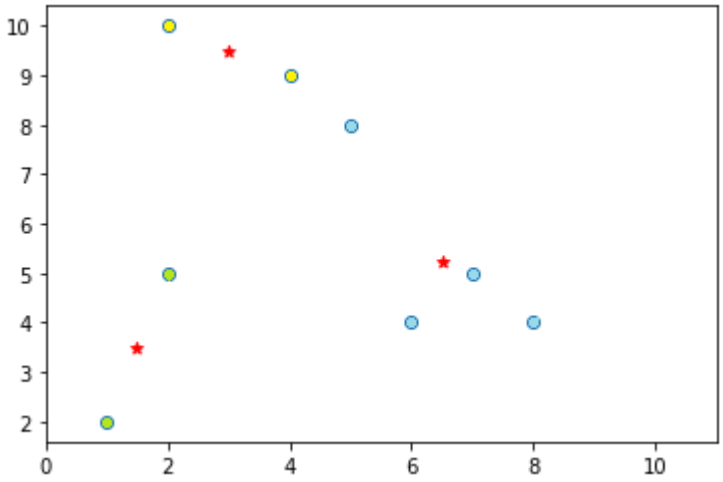
3: ((2+1)/2,(5+2)/2) = (1.5, 3.5)



ITERATION 2:

Cluster 2 : (6, 6)	Cluster 3 : (1.5, 3.5)
D(A1)=$\sqrt{(6-2)^2+(6-10)^2}$	D(A1)=$\sqrt{(1.5-2)^2+(3.5-10)^2}$
D(A2)= $\sqrt{(6-2)^2+(6-5)^2}$	D(A2)= $\sqrt{(1.5-2)^2+(3.5-5)^2}$
D(A3)= $\sqrt{(6-8)^2+(6-4)^2}$	D(A3)= $\sqrt{(1.5-8)^2+(3.5-4)^2}$
D(A4)= $\sqrt{(6-5)^2+(6-8)^2}$	D(A4)= $\sqrt{(1.5-5)^2+(3.5-8)^2}$
D(A5)= $\sqrt{(6-7)^2+(6-5)^2}$	D(A5)= $\sqrt{(1.5-7)^2+(3.5-5)^2}$
D(A6)= $\sqrt{(6-6)^2+(6-4)^2}$	D(A6)= $\sqrt{(1.5-6)^2+(3.5-4)^2}$
D(A7)= $\sqrt{(6-1)^2+(6-2)^2}$	D(A7)= $\sqrt{(1.5-1)^2+(3.5-2)^2}$
D(A8)= $\sqrt{(6-4)^2+(6-9)^2}$	D(A8)= $\sqrt{(1.5-4)^2+(3.5-9)^2}$

	Cluster 1 : (2, 10)	Cluster 2 : (6, 6)	Cluster 3 : (1.5, 3.5)
A1	0	5.65	6.52
A2	$\sqrt{25}$	$\sqrt{17}$	1.58
A3	$\sqrt{36}$	2.83	6.52
A4	$\sqrt{13}$	$\sqrt{5}$	5.7
A5	$\sqrt{50}$	$\sqrt{2}$	5.7
A6	$\sqrt{52}$	2	4.53
A7	$\sqrt{65}$	$\sqrt{41}$	1.58
A8	$\sqrt{5}$	$\sqrt{13}$	6.04



New labeled data points:

1: A1, A8, **2:** A3, A4, A5, A6, **3:** A2, A7

New clusters' centroids:

1: ((2+4)/2,(10+9)/2) = (3, 9.5)
2: ((8+5+7+6)/4,(4+8+5+4)/4) = (6.5, 5.25)
3: ((2+1)/2,(5+2)/2) = (1.5, 3.5)

ITERATION 3:

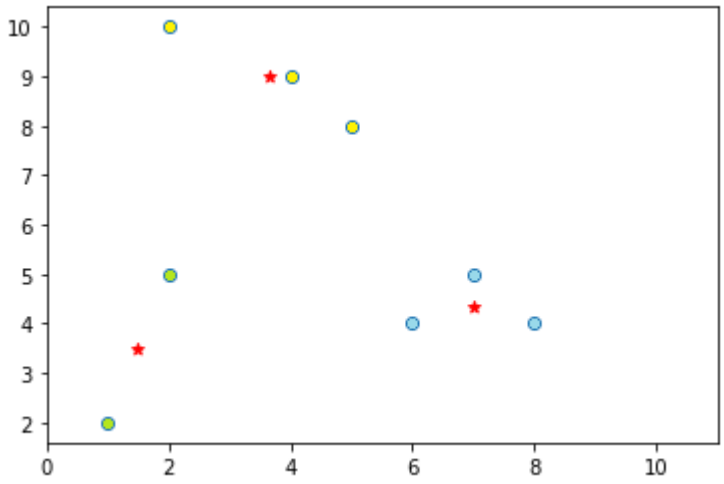
Cluster 1 : (3, 9.5)	Cluster 2 : (6.5, 5.25)
D(A1)=$\sqrt{(3-2)^2+(9.5-10)^2}$	D(A1)=$\sqrt{(6.5-2)^2+(5.25-10)^2}$
D(A2)= $\sqrt{(3-2)^2+(9.5-5)^2}$	D(A2)= $\sqrt{(6.5-2)^2+(5.25-5)^2}$
D(A3)= $\sqrt{(3-8)^2+(9.5-4)^2}$	D(A3)= $\sqrt{(6.5-8)^2+(5.25-4)^2}$
D(A4)= $\sqrt{(3-5)^2+(9.5-8)^2}$	D(A4)= $\sqrt{(6.5-5)^2+(5.25-8)^2}$
D(A5)= $\sqrt{(3-7)^2+(9.5-5)^2}$	D(A5)= $\sqrt{(6.5-7)^2+(5.25-5)^2}$
D(A6)= $\sqrt{(3-6)^2+(9.5-4)^2}$	D(A6)= $\sqrt{(6.5-6)^2+(5.25-4)^2}$
D(A7)= $\sqrt{(3-1)^2+(9.5-2)^2}$	D(A7)= $\sqrt{(6.5-1)^2+(5.25-2)^2}$
D(A8)= $\sqrt{(3-4)^2+(9.5-9)^2}$	D(A8)= $\sqrt{(6.5-4)^2+(5.25-9)^2}$

	Cluster 1 : (2, 10)	Cluster 2 : (6, 6)	Cluster 3 : (1.5, 3.5)
A1	1.11	6.54	6.52
A2	4.6	4.5	1.58
A3	7.4	1.95	6.52
A4	2.5	3.13	5.7
A5	6.2	0.56	5.7
A6	6.26	1.35	4.53
A7	7.76	6.38	1.58
A8	1.118	4.5	6.04

New labeled data points:
1: A1, A4, A8 , **2:** A3, A5, A6, **3:** A2, A7

New clusters' centroids:

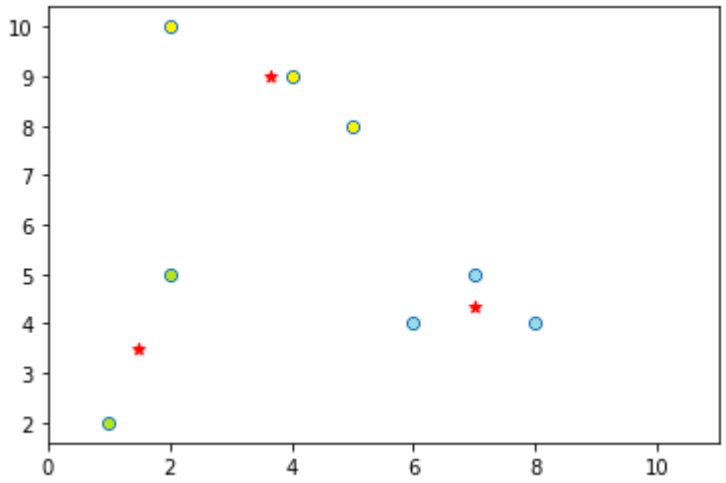
1: ((2+5+4)/3,(10+8+9)/3)=(3.66, 9)
2: ((8+7+6)/3,(4+5+4)/3) = (7, 4.33)
3: ((2+1)/2,(5+2)/2)=(1.5, 3.5)



ITERATION 4:

Cluster 1 : (3.66, 9)	Cluster 2 : (7, 4.33)
D(A1)=$\sqrt{(3.66-2)^2+(9-10)^2}$	D(A1)=$\sqrt{(7-2)^2+(4.33-10)^2}$
D(A2)= $\sqrt{(3.66-2)^2+(9-5)^2}$	D(A2)= $\sqrt{(7-2)^2+(4.33-5)^2}$
D(A3)= $\sqrt{(3.66-8)^2+(9-4)^2}$	D(A3)= $\sqrt{(7-8)^2+(4.33-4)^2}$
D(A4)= $\sqrt{(3.66-5)^2+(9-8)^2}$	D(A4)= $\sqrt{(7-5)^2+(4.33-8)^2}$
D(A5)= $\sqrt{(3.66-7)^2+(9-5)^2}$	D(A5)= $\sqrt{(7-7)^2+(4.33-5)^2}$
D(A6)= $\sqrt{(3.66-6)^2+(9-4)^2}$	D(A6)= $\sqrt{(7-6)^2+(4.33-4)^2}$
D(A7)= $\sqrt{(3.66-1)^2+(9-2)^2}$	D(A7)= $\sqrt{(7-1)^2+(4.33-2)^2}$
D(A8)= $\sqrt{(3.66-4)^2+(9-9)^2}$	D(A8)= $\sqrt{(7-4)^2+(4.33-9)^2}$

	Cluster 1 : (2,10)	Cluster 2 : (6,6)	Cluster 3 : (1.5,3.5)
A1	1.93	7.56	6.52
A2	4.33	5.04	1.58
A3	6.62	1.05	6.52
A4	1.67	4.18	5.7
A5	5.2	0.67	5.7
A6	5.5	1.05	4.53
A7	7.49	6.44	1.58
A8	0.34	5.55	6.04



New labeled data points:

1: A1, A4, A8 , **2:** A3, A5, A6, **3:** A2, A7

New clusters' centroids:

1: ((2+5+4)/3,(10+8+9)/3)=(3.66, 9)
2: ((8+7+6)/3,(4+5+4)/3) = (7, 4.33)
3: ((2+1)/2,(5+2)/2)=(1.5, 3.5)

THE SAME RESULTS AND CLUSTERS SO WE WILL STOP AFTER 3 ITERATIONS

Q2 (DBSCAN):

1):

Parameters:

epsilon	6.5
minpoints	4

Distance matrix:

A1	A2	A3	A4	A5	A6	A7	A8
0	5	6	3.60555	7.07107	7.2111	8.06226	2.23607
	0	6.08276	4.24264	5	4.12311	3.16228	4.47214
		0	5	1.41421	1.41421	7.28011	6.40312
			0	3.60555	4.12311	7.2111	1.41421
				0	1.41421	6.7082	5
					0	5.38516	5.38516
						0	7.61577
							0

Data points and their neighbors according to the epsilon value:

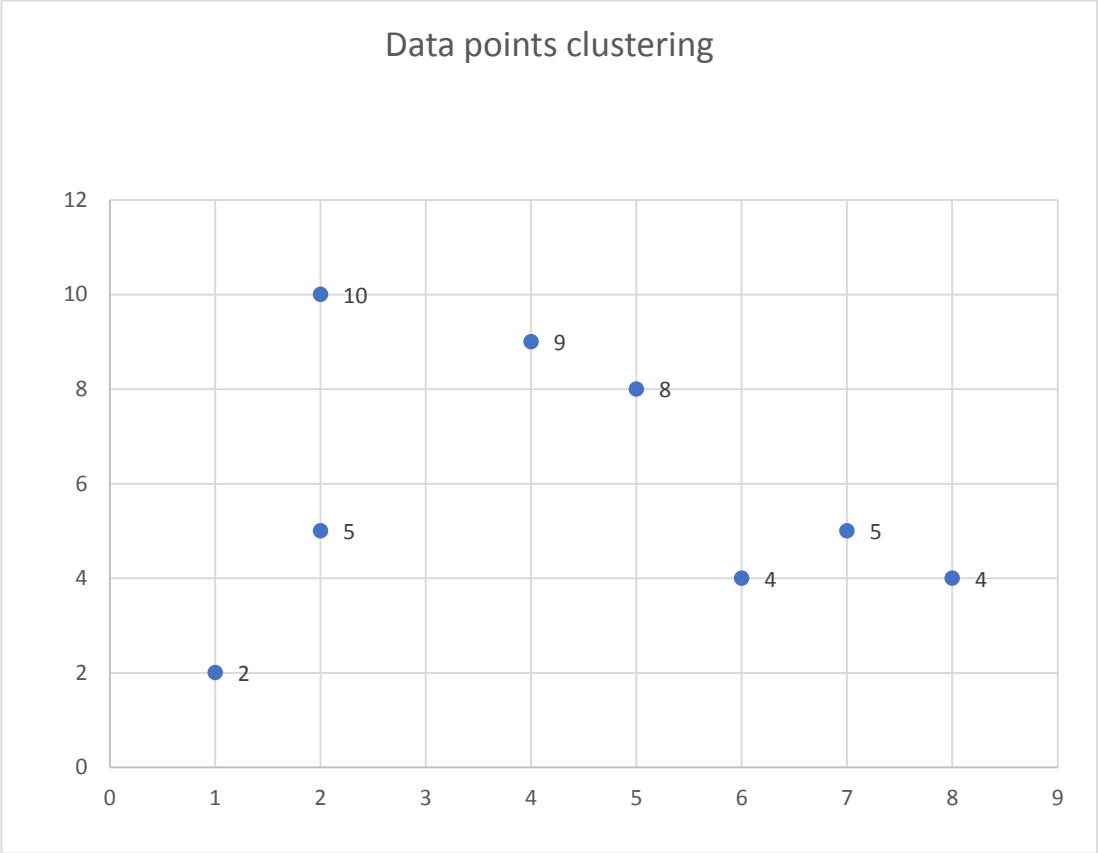
	A1	A2	A3	A4	A5	A6	A7	A8
A1	A1A1	**A2A1**	**A3A1**	**A4A1**	FALSE	FALSE	FALSE	**A8A1**
A2	**A2A1**	A2A2	**A3A2**	**A4A2**	**A5A2**	**A6A2**	**A7A2**	**A8A2**
A3	**A3A1**	**A3A2**	A3A3	**A4A3**	**A5A3**	**A6A3**	FALSE	**A8A3**
A4	**A4A1**	**A4A2**	**A3A4**	A4A4	**A5A4**	**A6A4**	FALSE	**A8A4**
A5	FALSE	**A5A2**	**A3A5**	**A4A5**	A5A5	**A6A5**	FALSE	**A8A5**
A6	FALSE	**A6A2**	**A3A6**	**A4A6**	**A5A6**	A6A6	**A7A6**	**A8A6**
A7	FALSE	**A7A2**	FALSE	FALSE	FALSE	**A6A7**	A7A7	FALSE
A8	**A8A1**	**A8A2**	**A3A8**	**A4A8**	**A5A8**	**A6A8**	FALSE	A8A8

Data points classification according to minpoints value:

(2,10)	4	**core point**
(2,5)	7	**core point**
(8,4)	6	**core point**
(5,8)	6	**core point**
(7,5)	5	**core point**
(6,4)	6	**core point**
(1,2)	2	border
(4,9)	6	**core point**

The whole data points classification:

(2,10)	(2,5)	(8,4)	(5,8)	(4,9)			Cluster 1
(2,5)	(8,4)	(5,8)	(7,5)	(6,4)	(1,2)	(4,9)	Cluster 1
(8,4)	(5,8)	(7,5)	(6,4)	(4,9)			Cluster 1
(5,8)	(7,5)	(6,4)	(4,9)				Cluster 1
(7,5)	(6,4)	(4,9)					Cluster 1
(6,4)	(1,2)	(4,9)					Cluster 1
(1,2)							Cluster 1
(4,9)							Cluster 1



2):

Parameters:

epsilon	3.16228
minpoints	2

Distance matrix:

A1	A2	A3	A4	A5	A6	A7	A8
0	5	6	3.60555	7.07107	7.2111	8.06226	2.23607
	0	6.08276	4.24264	5	4.12311	3.16228	4.47214
		0	5	1.41421	1.41421	7.28011	6.40312
			0	3.60555	4.12311	7.2111	1.41421
				0	1.41421	6.7082	5
					0	5.38516	5.38516
						0	7.61577
							0

Data points and their neighbors according to the epsilon value:

	A1	A2	A3	A4	A5	A6	A7	A8
A1	A1A1	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	**A8A1**
A2	FALSE	A2A2	FALSE	FALSE	FALSE	FALSE	**A7A2**	FALSE
A3	FALSE	FALSE	A3A3	FALSE	**A5A3**	**A6A3**	FALSE	FALSE
A4	FALSE	FALSE	FALSE	A4A4	FALSE	FALSE	FALSE	**A8A4**
A5	FALSE	FALSE	**A3A5**	FALSE	A5A5	**A6A5**	FALSE	FALSE
A6	FALSE	FALSE	**A3A6**	FALSE	**A5A6**	A6A6	FALSE	FALSE
A7	FALSE	**A2A7**	FALSE	FALSE	FALSE	FALSE	A7A7	FALSE
A8	**A1A8**	FALSE	FALSE	**A4A8**	FALSE	FALSE	FALSE	A8A8

Data points classification according to minpoints value:

(2,10)	1	border
(2,5)	1	border
(8,4)	2	**Core**
(5,8)	1	border
(7,5)	2	**Core**
(6,4)	2	**Core**
(1,2)	1	border
(4,9)	2	**Core**

The whole data points classification:

(2,10)	(4,9)		
(2,5)	(1,2)		
(8,4)	(7,5)	(6,4)	Cluster 1
(5,8)	(4,9)		
(7,5)	(8,4)	(6,4)	Cluster 1
(6,4)	(8,4)	(7,5)	Cluster 1
(1,2)	(2,5)		Cluster 3
(4,9)	(2,10)	(5,8)	Cluster 2

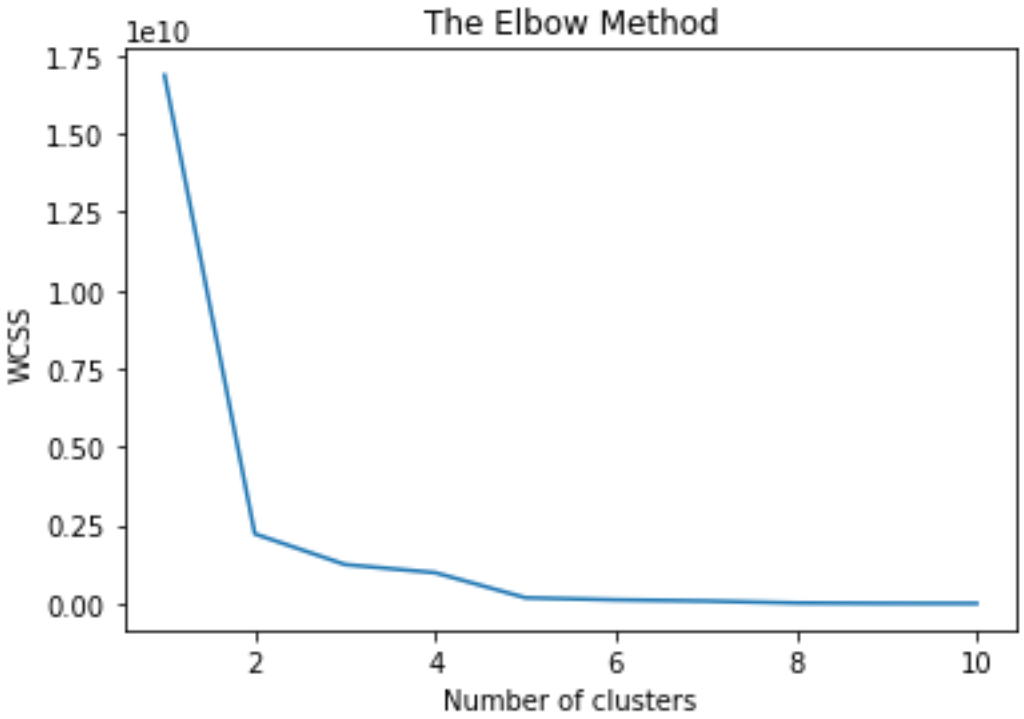
As we can see, it's the same result obtained using K-Means clustering algorithm.

Programming Questions:

Q4 (choose the best number of k for k-means algorithm):

1. Using the elbow rule, plot the distortion score (a.k.a inertia) vs the number of clusters.

```
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-
means++', max_iter=100, n_init=1, random_state=69)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
```



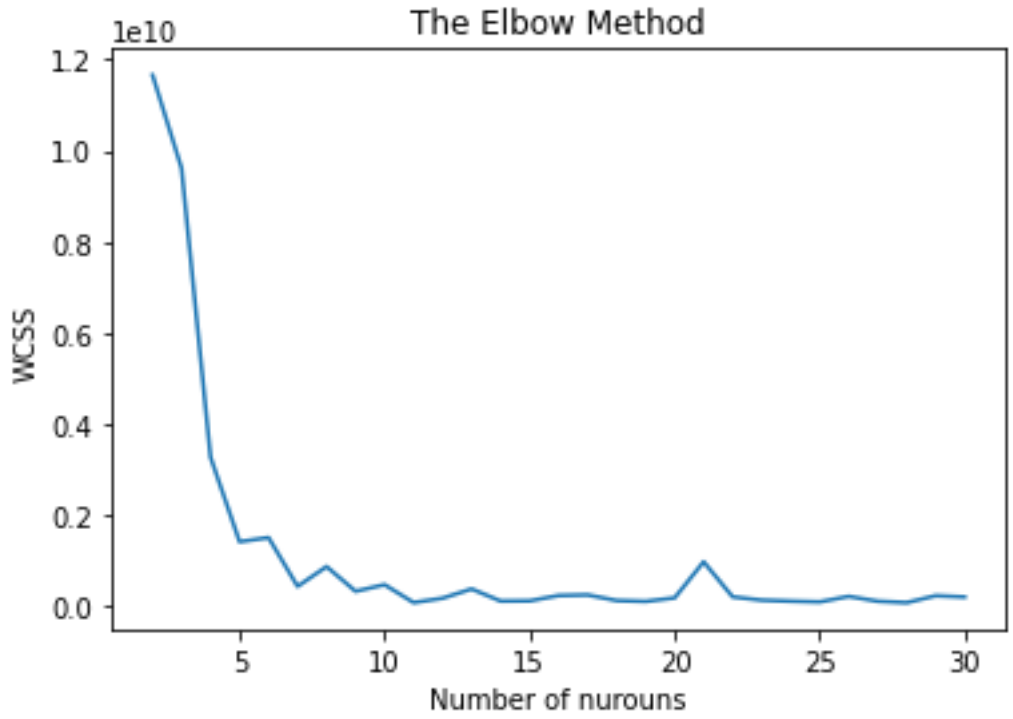
2. Determine the optimal number of clusters for k-means.

The optimal number of clusters for k-means is **5** according to the elbow plot.

Q5 (choose the best number of neurons for SOM algorithm):

1. Using the elbow rule, plot the distortion score (a.k.a inertia) vs the number of neurons (max 30 neurons).

```
wcss = []
dim = X.shape[1]
for i in range(2, 31):
    som = SOM(m=i, n=1, dim=dim)
    som.fit(X)
    wcss.append(som.inertia_)
plt.plot(range(2, 31), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of nurouns')
plt.ylabel('WCSS')
```



2. Determine the optimal number of neurons for SOM.

The optimal number of neurons for SOM is **14** according to the elbow plot.

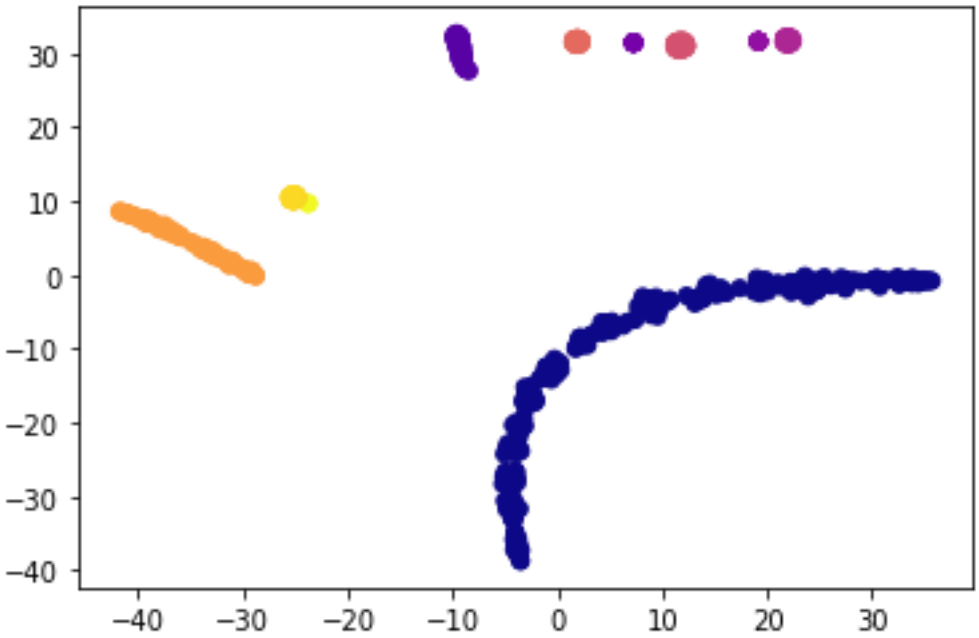
Bonus:

Q5 (choose the best number of k for k-means algorithm):

3. Use T-SNE method to visualize the SOM clusters, obtained from the previous step, on a 2D figure. Use different color code for each cluster.

```
X_reduced = TSNE(n_components=2, random_state=123).fit_transform(X)
dim = X.shape[1]
poki_som = SOM(m=14, n=1, dim=dim)
poki_som.fit(X)
y_predict = poki_som.predict(X)
y_clusters = getTrueLabel(y_predict, y)
plt.scatter(X_reduced[:, 0], X_reduced[:, 1], c=y_predict, cmap="plasma")
print("Silhouette Coefficient:", metrics.silhouette_score(X, y_predict))
print("Accuracy:", metrics.accuracy_score(y_clusters, y))
```

```
Silhouette Coefficient: 0.9744251560548564
Accuracy: 0.27166666666666667
```

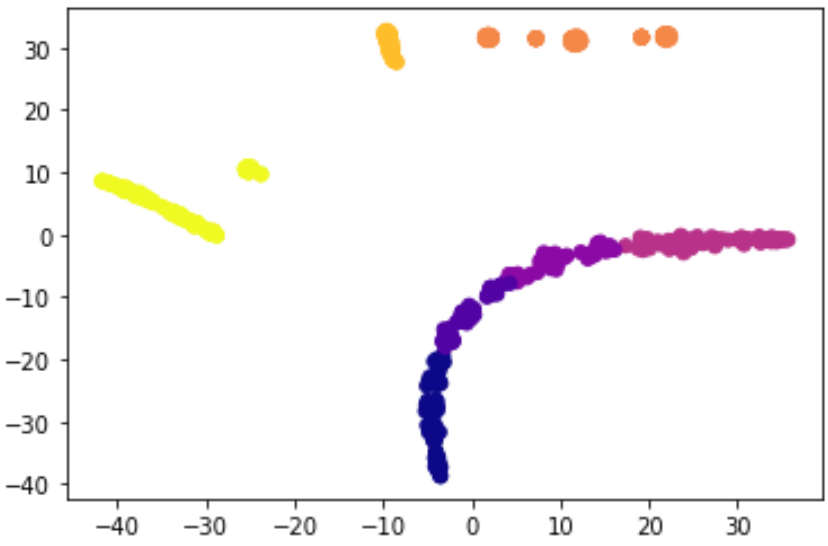


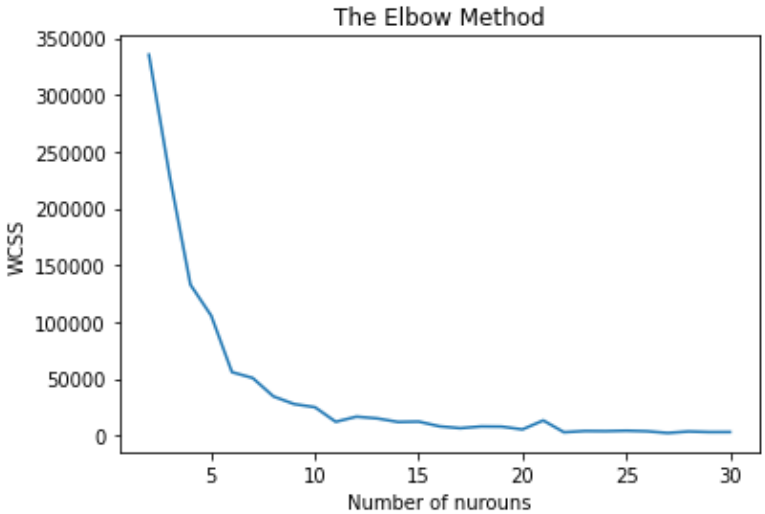
4. Compare the results of the previous step with T-SNE before applying SOM on the Pokémon dataset.

```
# Elbow Plot
wcss = []
dim = X_reduced.shape[1]
for i in range(2, 31):
    som = SOM(m=i, n=1, dim=dim)
    som.fit(X_reduced)
    wcss.append(som.inertia_)
plt.plot(range(2, 31), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of nurouns')
plt.ylabel('WCSS')

# Clusters Visualization
dim = X_reduced.shape[1]
poki_som = SOM(m=8, n=1, dim=dim)
poki_som.fit(X_reduced)
y_predict = poki_som.predict(X_reduced)
y_clusters = getTrueLabel(y_predict, y)
plt.scatter(X_reduced[:, 0], X_reduced[:, 1], c=y_predict, cmap="plasma")
print("Silhouette Coefficient:", metrics.silhouette_score(X_reduced, y_pre
dict))
print("Accuracy:", metrics.accuracy_score(y_clusters, y))
```

It yields worse result according to the silhouette coefficient and accuracy.

```
Silhouette Coefficient: 0.63209724
Accuracy: 0.235
```





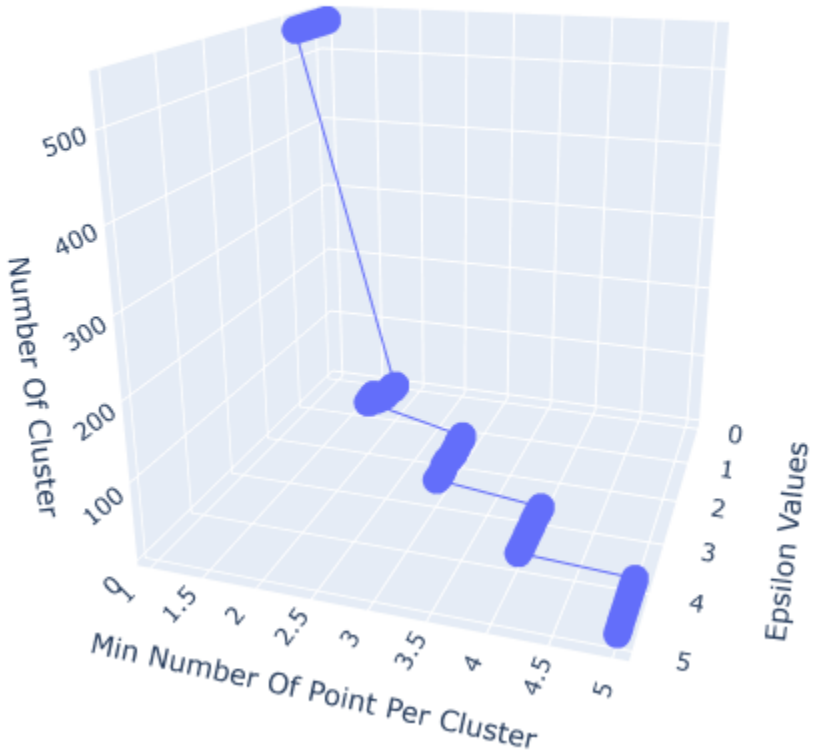
Q6 (DBSCAN tuning):

- **Plot the epsilon and minpoints values as x-y axes using a 3D figure to show the 10 combinations of epsilon and minpoints that brings you closer to the 10 clusters.**

```
def tuneDBSCAN(tune_rate = 0.1):
    eps_vals = []
    min_points_vals = []
    num_clusters = []
    eps = tune_rate
    for min_p in range(1, 6):
        while(eps < min_p):
            dbscan = DBSCAN(eps=eps, min_samples = min_p)
            y_predict = dbscan.fit_predict(X)
            eps_vals.append(eps)
            min_points_vals.append(min_p)
            num_clusters.append(len(np.unique(y_predict)) - 1)
            eps += tune_rate
    re-
turn np.array(eps_vals), np.array(min_points_vals), np.array(num_clusters)

e, p, c = tuneDBSCAN(tune_rate=0.01)

c_9 = np.where(c == 9)[0]
c_10 = np.where(c == 10)[0]
f_eps = 0
f_num_p = 0
try:
    f_eps = e[c_10[0]]
    f_num_p = p[c_10[0]]
except:
    f_eps = e[c_9[0]]
    f_num_p = p[c_9[0]]

print('the nearest paramters to get 10 clusters after tuning:')
print('eps=', f_eps)
print('min_points=', f_num_p)
```

The nearest paramters to get 10 clusters after tuning:
Epsilon = **4.90999999999994**
Minpoints = **5**

```
fig = go.Figure(data=[go.Scatter3d(x=e, y=p, z=c)])
fig.update_layout(scene = dict(
    xaxis_title='Epsilon Values',
    yaxis_title='Min Number Of Point Per Cluster',
    zaxis_title='Number Of Cluster'),
    width=700,
    margin=dict(r=20, b=10, l=10, t=10)
)
fig.show()
```



- **Plot DBSCAN clusters.**

```
dbscan = DBSCAN(eps = f_eps, min_samples = f_num_p)
y_predict = dbscan.fit_predict(X)
y_clusters = getTrueLabel(y_predict, y)
print("Silhouette Coefficient:", metrics.silhouette_score(X, y_predict))
print("Accuracy:", metrics.accuracy_score(y_clusters, y))
plt.scatter(X_reduced[:, 0], X_reduced[:, 1], c=y_predict, cmap="plasma")
```

```
Silhouette Coefficient: 0.06272461850083305
Accuracy: 0.23166666666666666
```