Machine Learning

Lesson 6: Unsupervised Learning

Concepts Covered

- ✓ Unsupervised Learning
- ✓ Hierarchical Clustering
- ✓ Dendrogram
- ✓ K means clustering

Learning Objectives

By the end of this lesson, you will be able to:

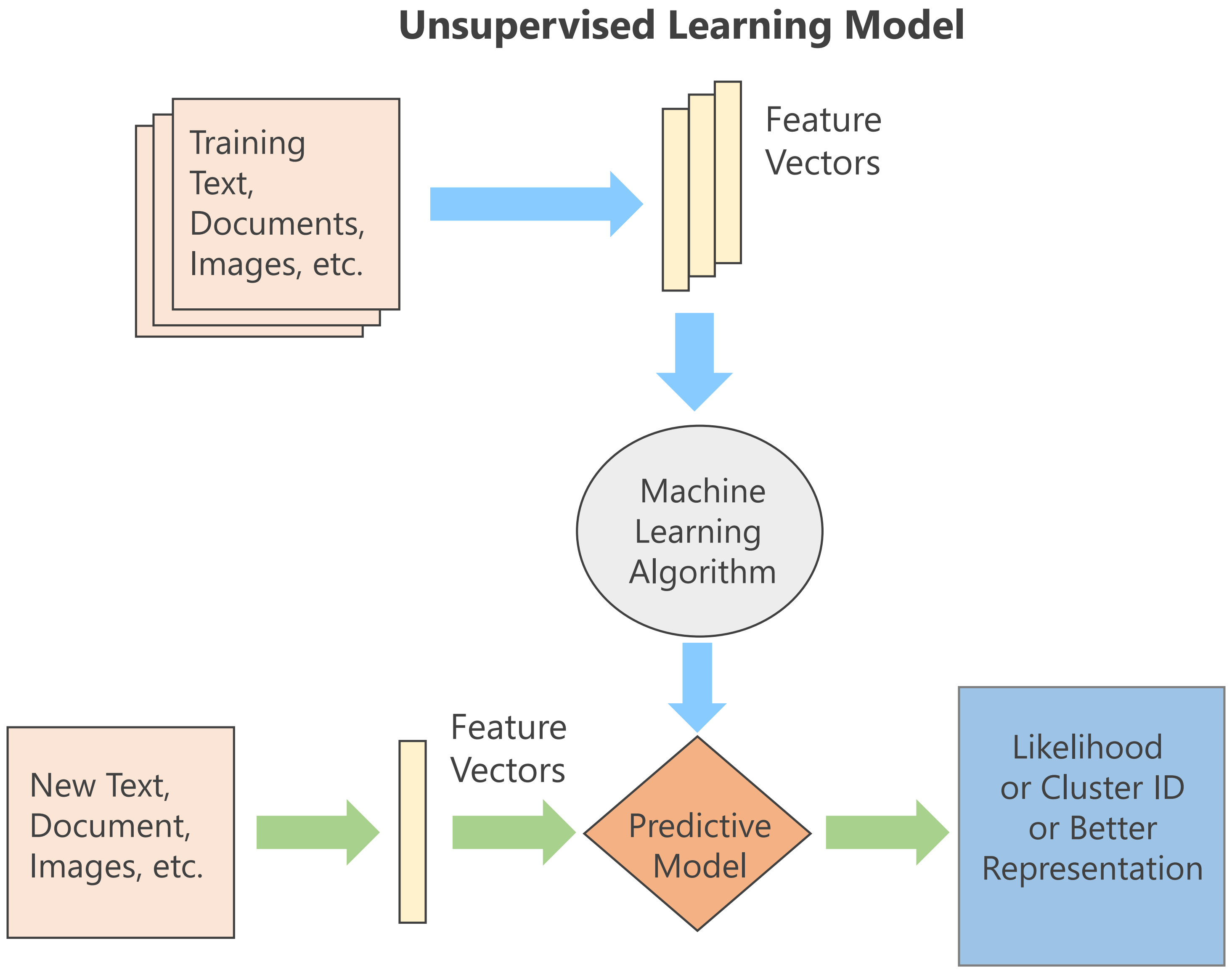
- Explain the mechanism of unsupervised learning
- Practice different clustering techniques in Python

Unsupervised Learning

Topic 1: Overview

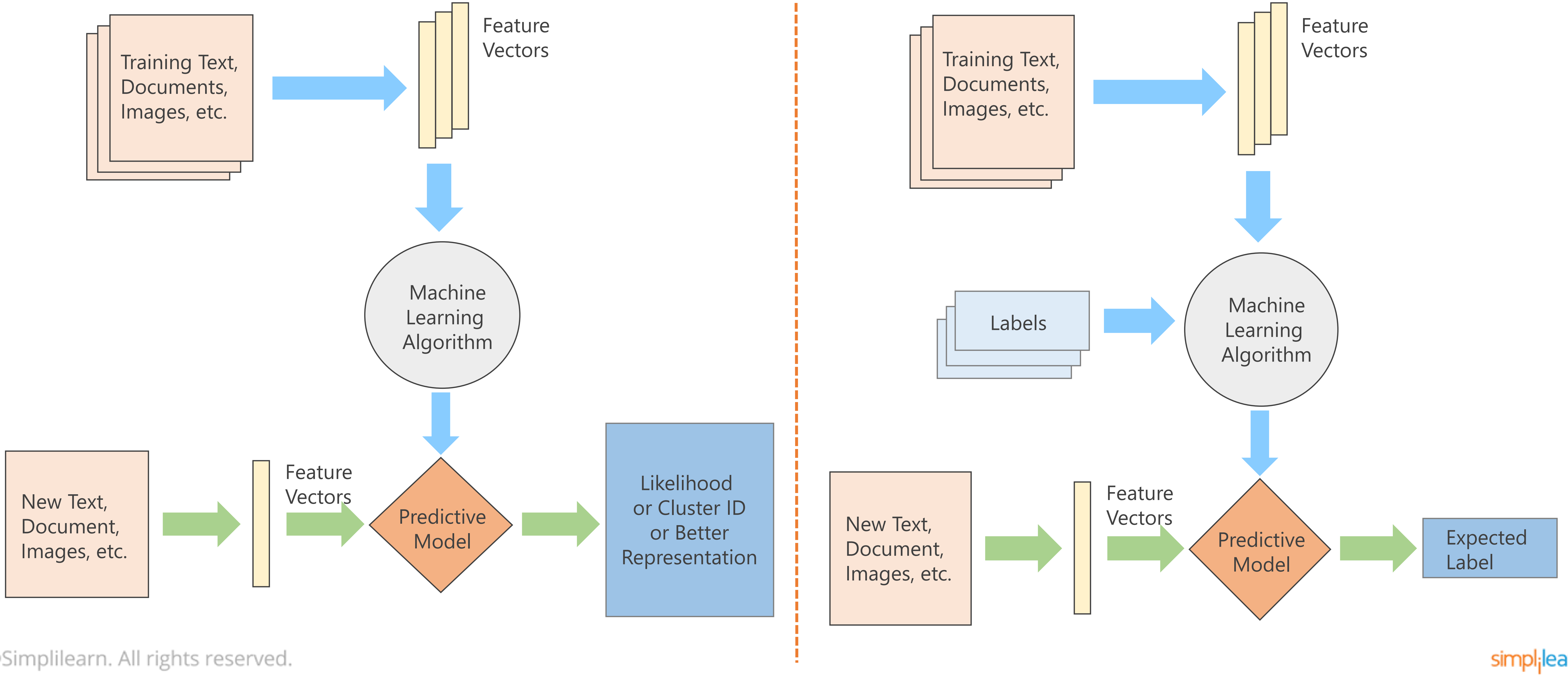
Unsupervised Learning Process Flow

The data has no labels. The machine just looks for whatever patterns it can find.


Unsupervised Learning Model
Training Text, Documents, Images, etc.
Feature Vectors
Machine Learning Algorithm
New Text, Document, Images, etc.
Feature Vectors
Predictive Model
Likelihood or Cluster ID or Better Representation

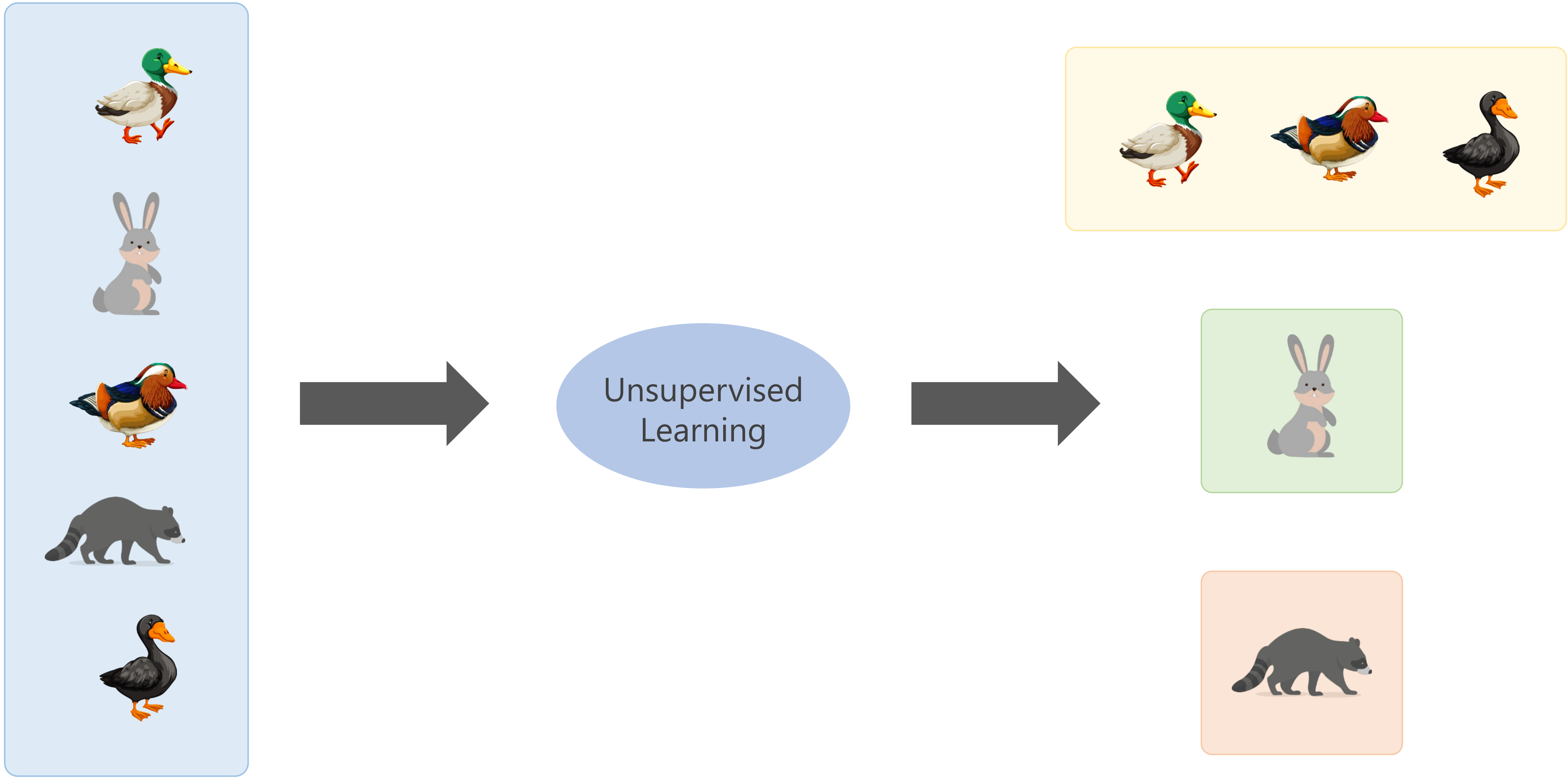
Unsupervised Learning vs. Supervised Learning

The only difference is the labels in the training data


Training Text, Documents, Images, etc.
Feature Vectors
Machine Learning Algorithm
New Text, Document, Images, etc.
Feature Vectors
Predictive Model
Likelihood or Cluster ID or Better Representation
Training Text, Documents, Images, etc.
Feature Vectors
Labels
Machine Learning Algorithm
New Text, Document, Images, etc.
Feature Vectors
Predictive Model
Expected Label

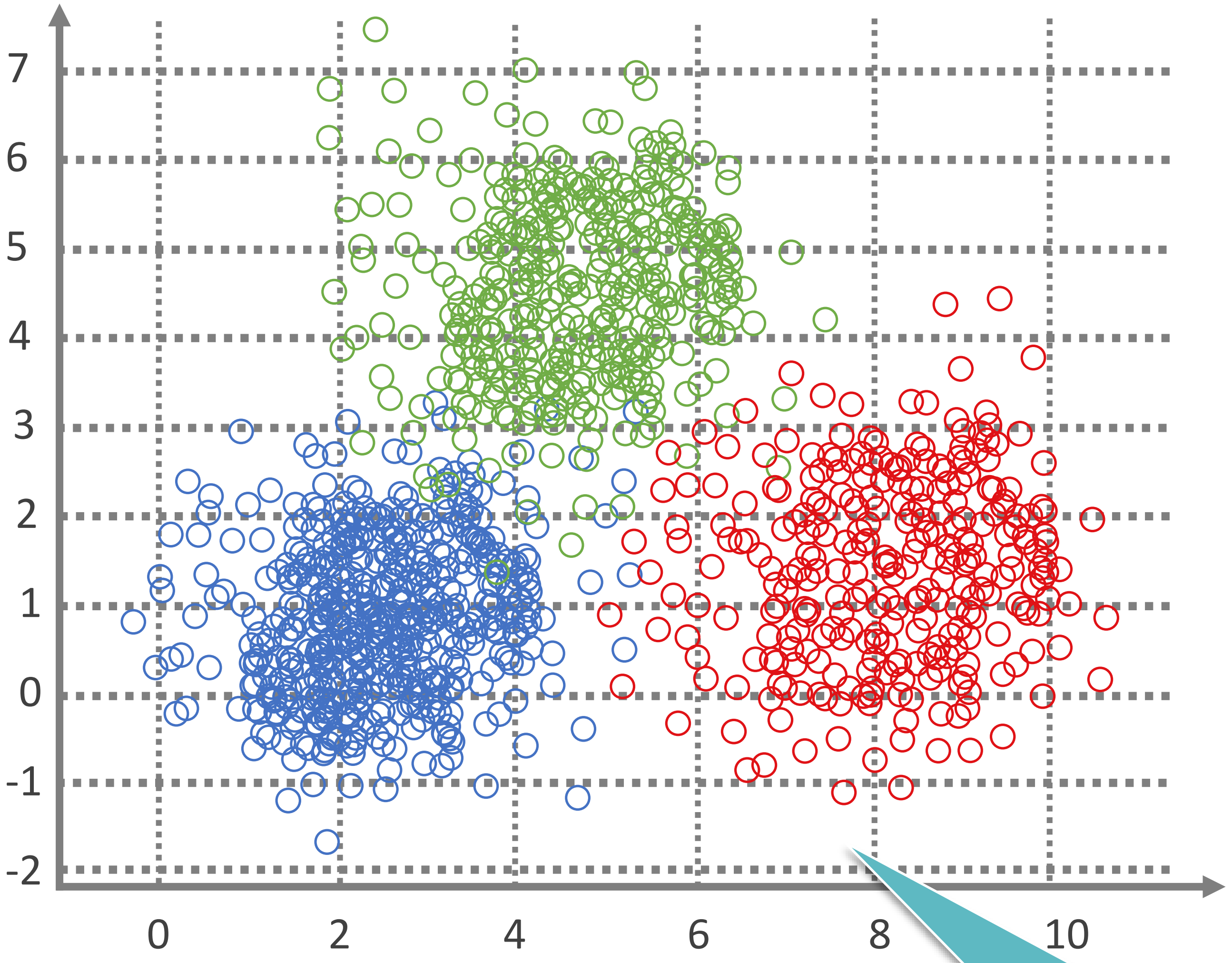
Unsupervised Learning: Example

Clustering like-looking birds/animals based on their features

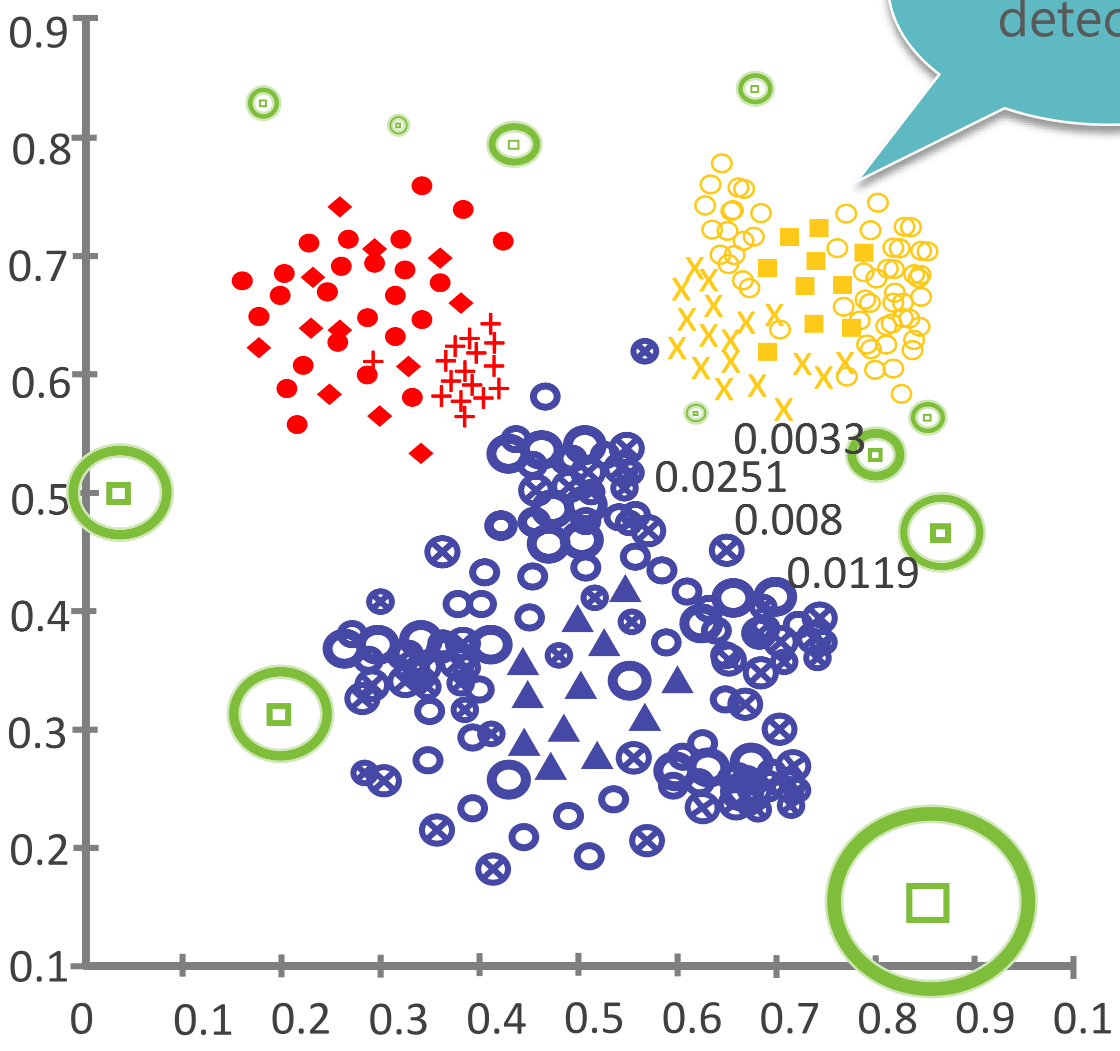


Application of Unsupervised Learning

Unsupervised learning can be used for anomaly detection as well as clustering



Identifying similarities in groups (Clustering)



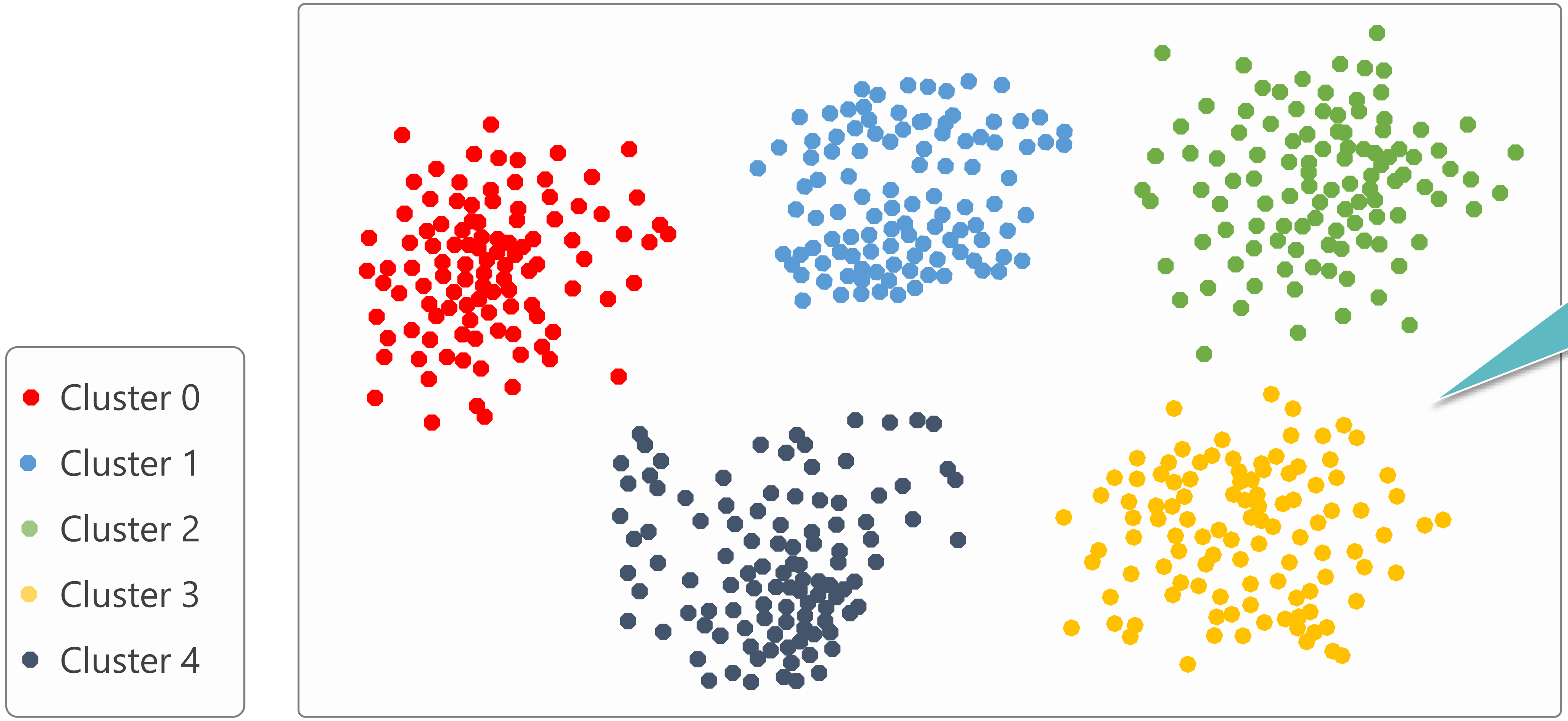
Anomaly detection

Unsupervised Learning

Topic 2: Clustering

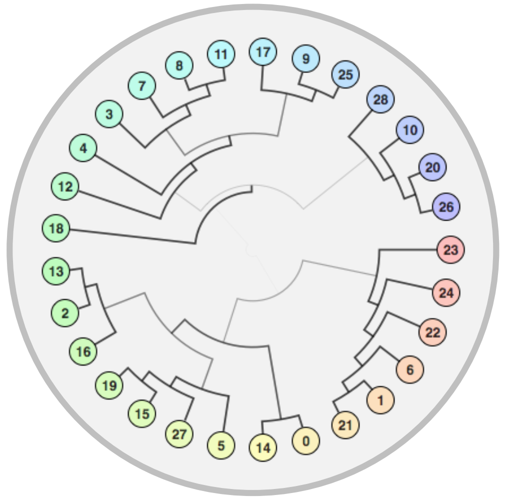
Clustering

Grouping objects based on the information found in data that describes the objects or their relationship



The goal is to see that similar objects are grouped into one cluster and different from objects in another cluster

Need of Clustering



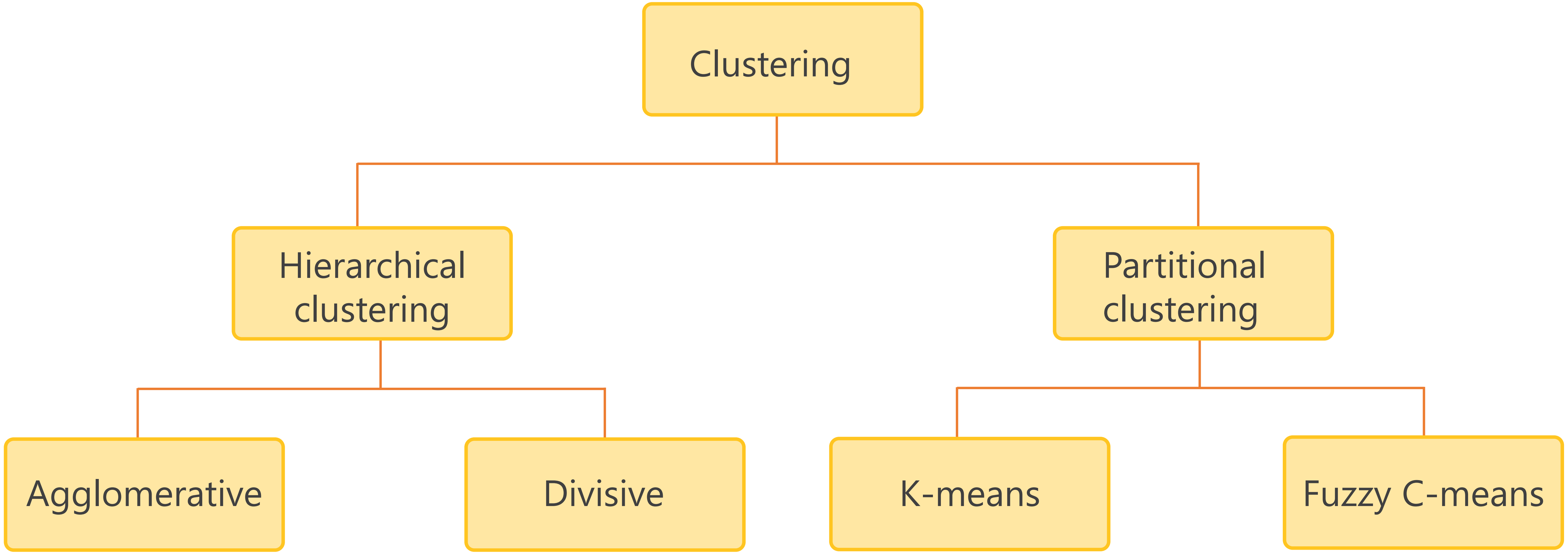
To determine the intrinsic grouping in a set of unlabeled data

To organize data into clusters showing internal structure of the data

To partition the data points

To understand and extract value from large sets of structured and unstructured data

Types of Clustering



Clustering

Hierarchical clustering

Partitional clustering

Agglomerative

Divisive

K-means

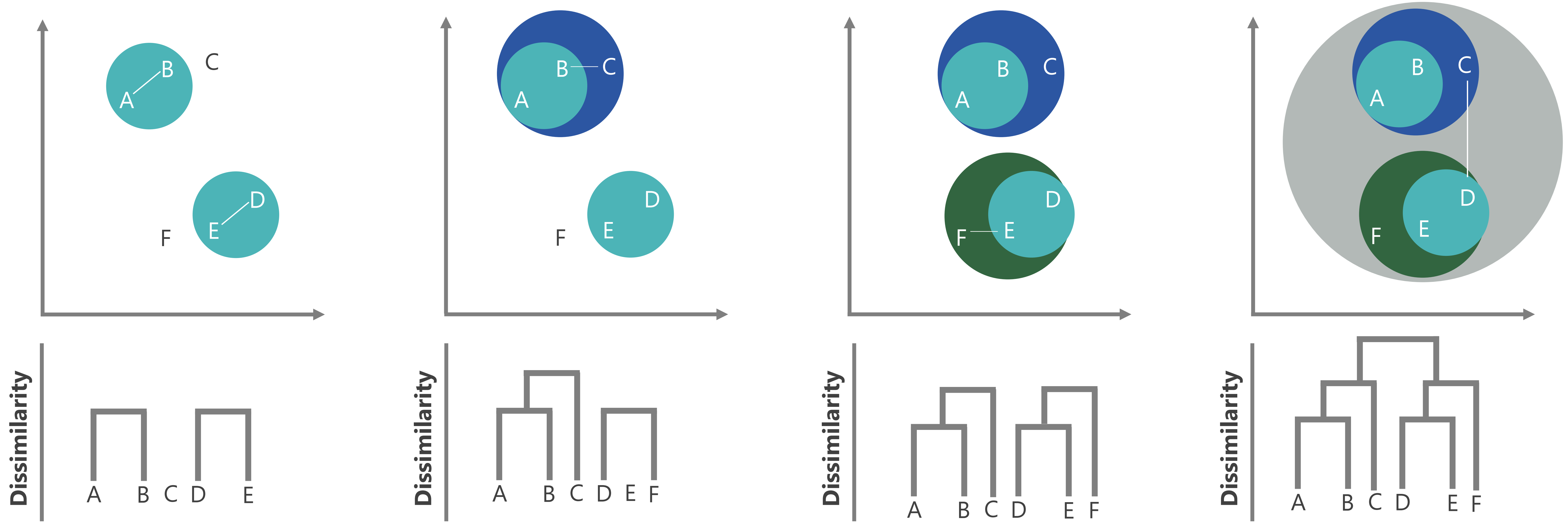
Fuzzy C-means

Unsupervised Learning

Topic 3: Hierarchical Clustering

Hierarchical Clustering

Outputs a hierarchy, a structure that is more informative than the unstructured set of clusters returned by flat clustering



Combine A and B based on similarity
Combine D and E based on similarity

1

Combination of A and B is combined
with C

2

Combination of D and E is combined
with F

3

Final tree contains all clusters
Combined into a single cluster

4

Working: Hierarchical Clustering

Step 1

Assign each item to its own cluster, such that if you have N number of items, you now have N number of clusters

Step 2

Find the closest (most similar) pair of clusters and merge them into a single cluster. Now you have one less cluster

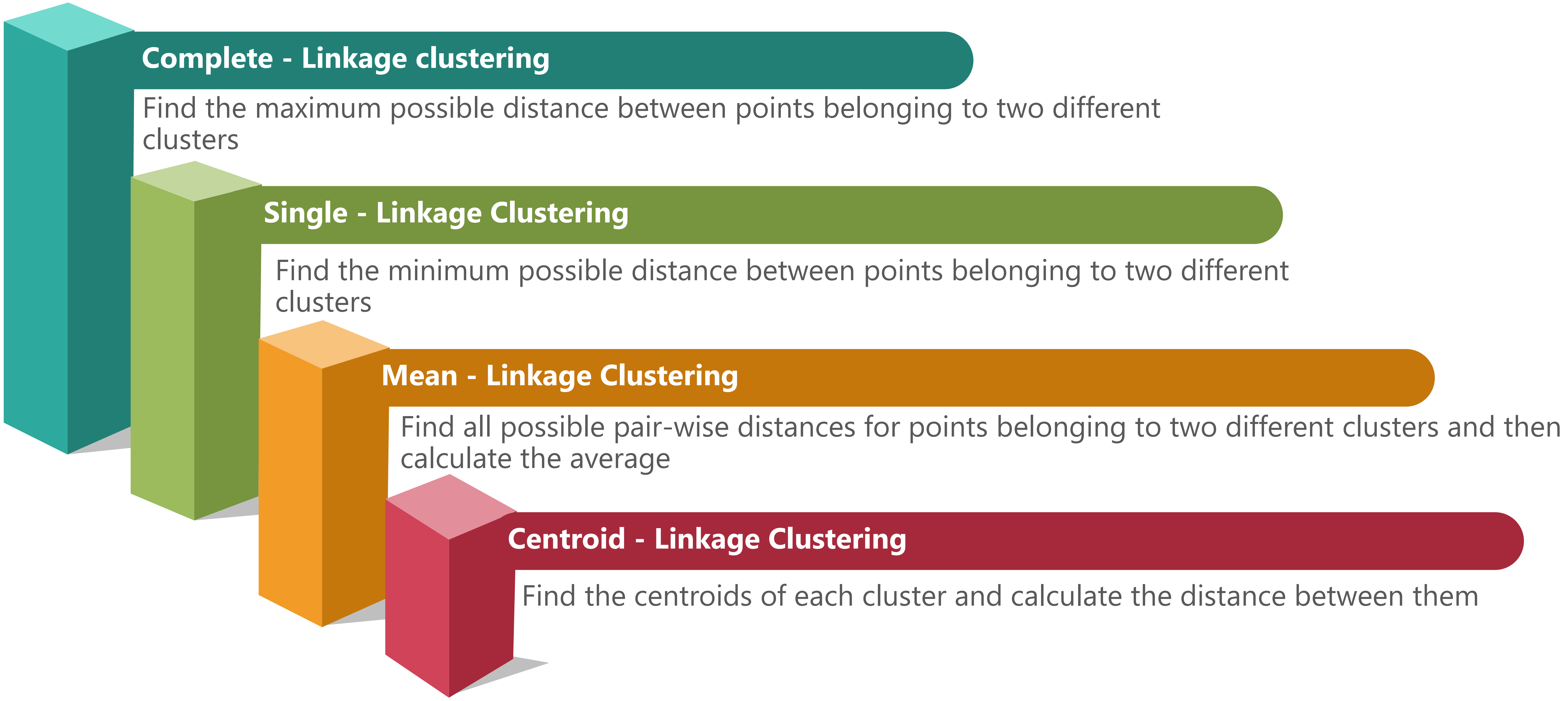
Step 3

Compute distances (similarities) between the new cluster and every old cluster

Step 4

Repeat steps 2 and 3 until all items are clustered into a single cluster of size N

Distance Measures



Complete - Linkage clustering

Find the maximum possible distance between points belonging to two different clusters

Single - Linkage Clustering

Find the minimum possible distance between points belonging to two different clusters

Mean - Linkage Clustering

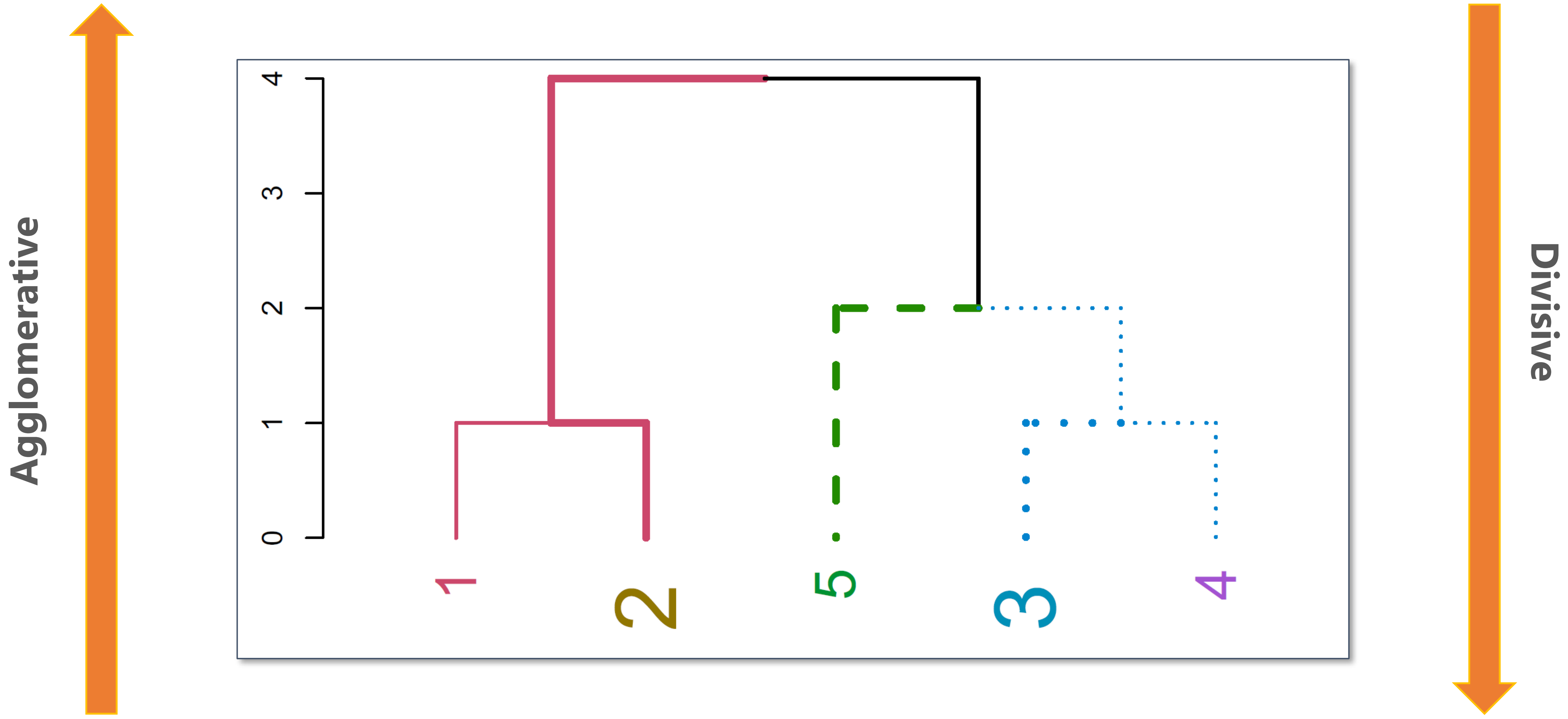
Find all possible pair-wise distances for points belonging to two different clusters and then calculate the average

Centroid - Linkage Clustering

Find the centroids of each cluster and calculate the distance between them

The Dendrogram

Dendrogram ((in Greek, *dendro* means tree and *gramma* means drawing) is a tree diagram frequently used to illustrate the arrangement of the clusters produced by hierarchical clustering.

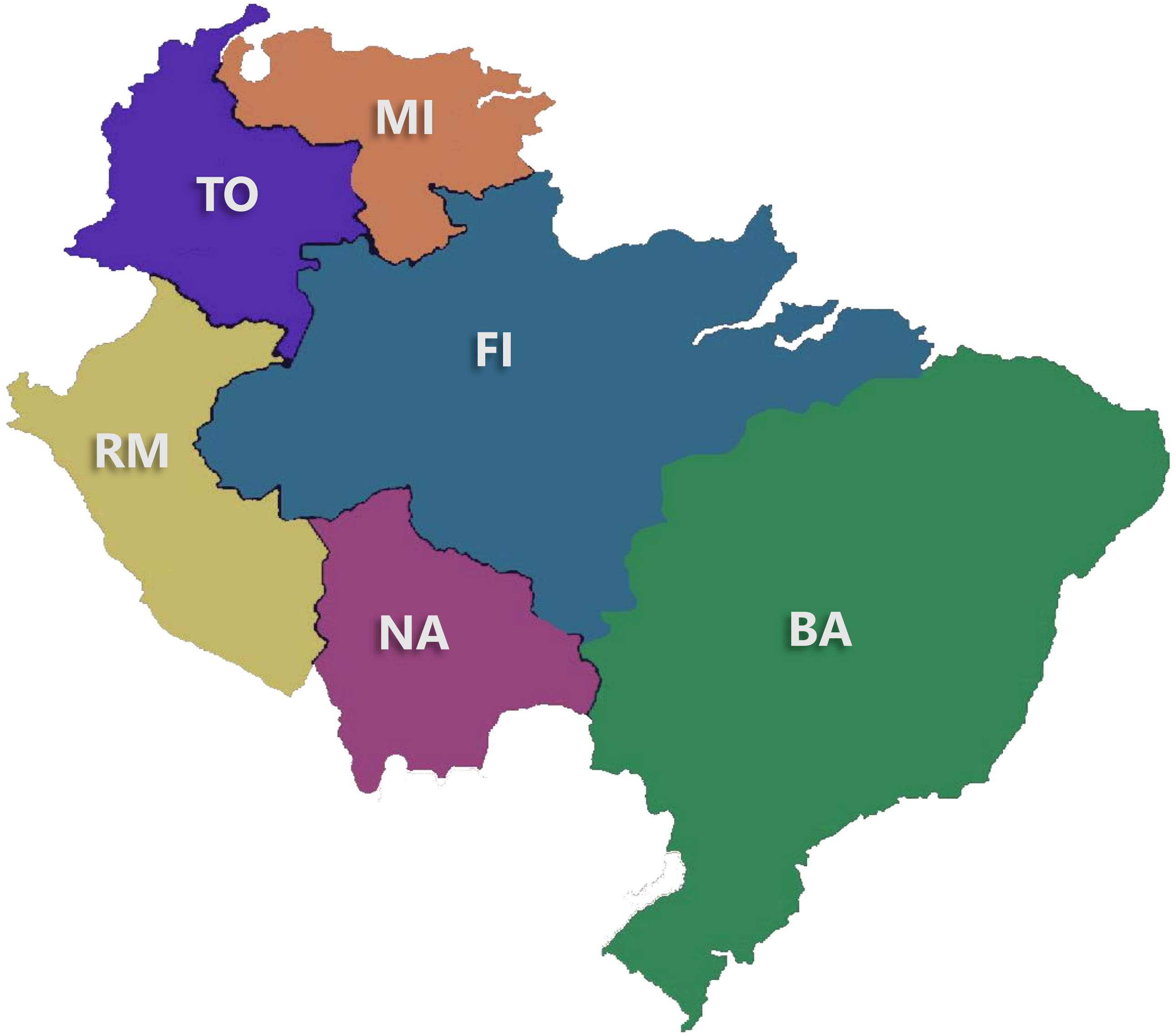


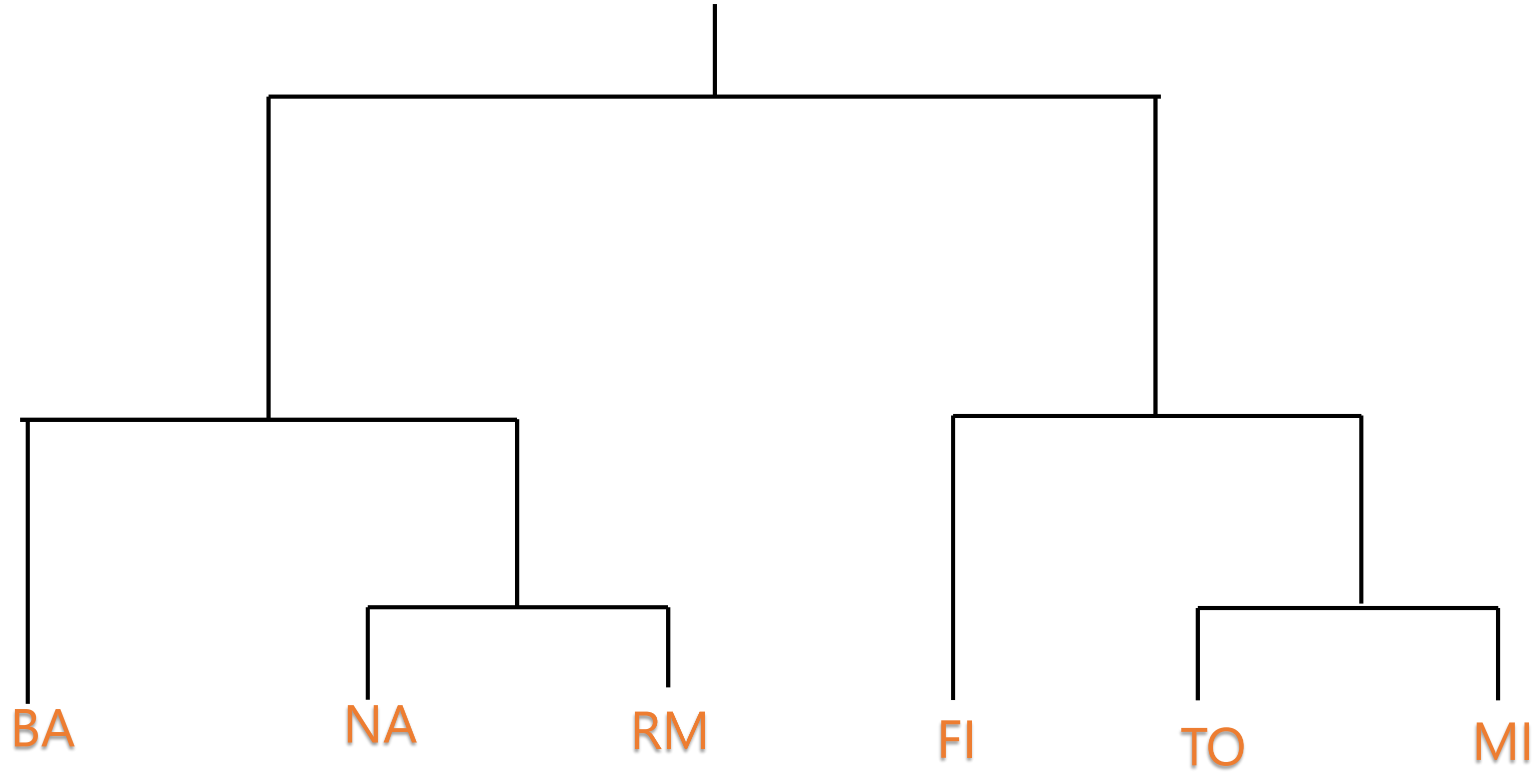
Agglomerative

Divisive

Hierarchical Clustering: Example

A hierarchical clustering of distances between cities in kilometers





Hierarchical Clustering: Step 1

Create distance matrix of data

	BA	FI	MI	NA	RM	TO
BA	0	662	877	255	412	996
FI	662	0	295	468	268	400
MI	877	295	0	754	564	138
NA	255	468	754	0	219	869
RM	412	268	564	219	0	669
TO	996	400	138	869	669	0

Distance between TO and MI

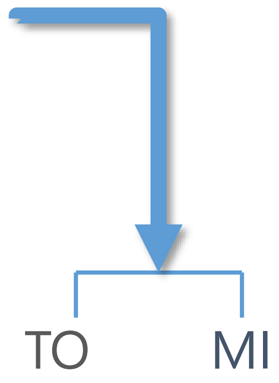
Distance Matrix

Hierarchical Clustering: Step 2

From the distance matrix, you can see that MI has least distance with TO and they form a cluster together

	BA	FI	MI	NA	RM	TO
BA	0	662	877	255	412	996
FI	662	0	295	468	268	400
MI	877	295	0	754	564	138
NA	255	468	754	0	219	869
RM	412	268	564	219	0	669
TO	996	400	138	869	669	0

	BA	FI	MI/TO	NA	RM
BA	0	662	877	255	412
FI	662	0	295	468	268
MI/TO	877	295	0	754	564
NA	255	468	754	0	219
RM	412	268	564	219	0



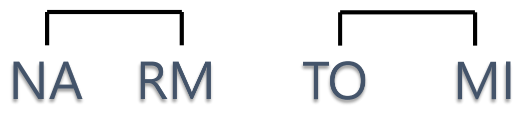
As the MI column has lower values than TO column, MI/TO consists of MI column values

Hierarchical Clustering: Step 3

Repeat clustering until a single cluster is obtained with all the members in it

	BA	FI	MI/TO	NA	RM
BA	0	662	877	255	412
FI	662	0	295	468	268
MI/TO	877	295	0	754	564
NA	255	468	754	0	219
RM	412	268	564	219	0

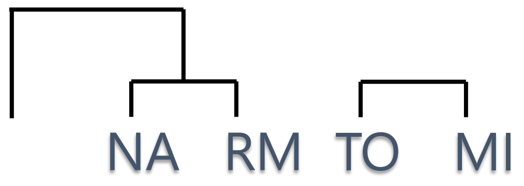
	BA	FI	MI/TO	NA/RM
BA	0	662	877	255
FI	662	0	295	268
MI/TO	877	295	0	564
NA/RM	255	468	564	0



Hierarchical Clustering: Step 3 (Contd.)

	BA	FI	MI/TO	NA/RM
BA	0	662	877	255
FI	662	0	295	268
MI/TO	877	295	0	564
NA/RM	255	268	564	0

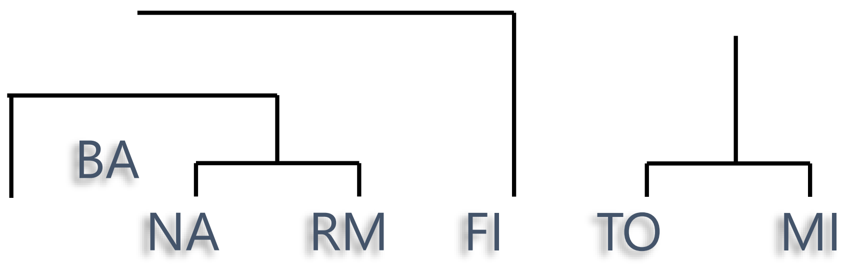
	BA/(NA/RM)	FI	MI/TO
BA/(NA/RM)	0	268	564
FI	268	0	295
MI/TO	564	295	0



Hierarchical Clustering: Step 3 (Contd.)

	BA/(NA/RM)	FI	MI/TO
BA/(NA/RM)	0	268	564
FI	268	0	295
MI/TO	564	295	0

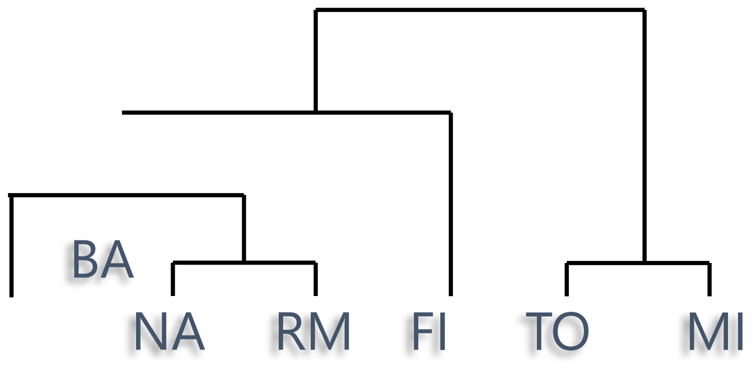
	BA/(NA/RM)/FI	(MI/TO)
BA/(NA/RM)/FI	0	295
(MI/TO)	295	0



Hierarchical Clustering: Step 4

Derive the final dendrogram

	BA/(NA/RM)/FI	(MI/TO)
BA/(NA/RM)/FI	0	295
(MI/TO)	295	0



Assisted Practice

Hierarchical Clustering

Duration: 15 mins.

Problem Statement: Consider the dataset “zoo.data” and look at the info in the first five rows. The first column denotes the animal name and the last one specifies a high-level class for the corresponding animal. Find a solution to the following questions:

- Unique number of high-level class
- Perform agglomerative clustering using the 16 intermediate features

  [Hint: Refer to the agglomerative clustering (Hierarchical Clustering) module in Scikit learn and set the number of clusters appropriately]

  Refer the below link for further documentation:
  http://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html
- Compute the mean squared error by comparing the actual class and predicted high-level class.

Objective: Perform agglomerative clustering with appropriate MSE value.

Access: Click on the Labs tab on the left side panel of the LMS. Copy or note the username and password that are generated. Click on the Launch Lab button. On the page that appears, enter the username and password in the respective fields, and click Login.

Unassisted Practice

Hierarchical Clustering

Duration: 10 mins.

Problem Statement: An ecommerce company has prepared a rough dataset containing shopping details of their customers, which includes CustomerID, Genre, Age, Annual Income (k$), Spending Score (1-100). The company is unable to target a specific set of customers with a particular set of SKUs.

Objective: Segment customers into different groups based on their shopping trends.

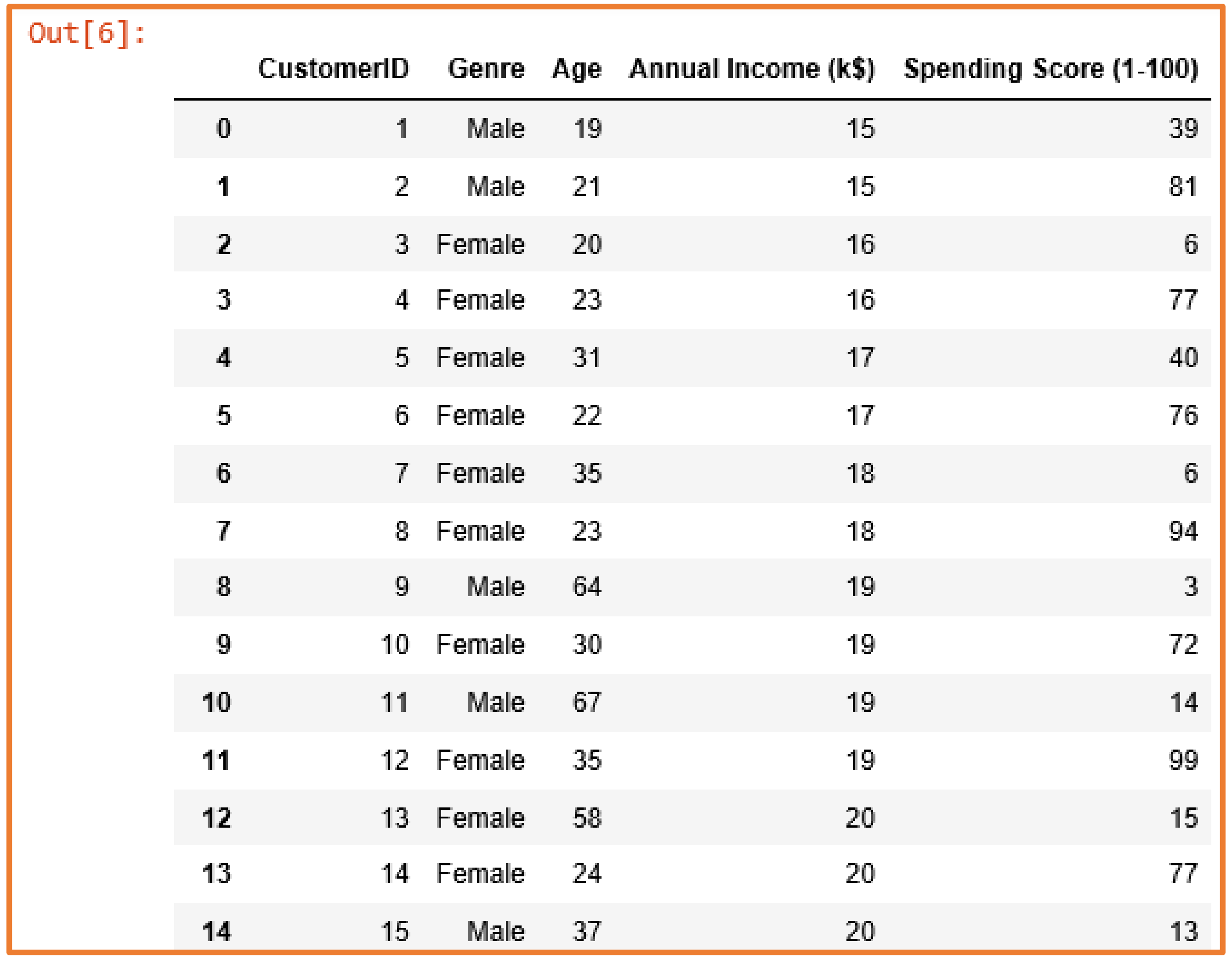
Note: This practice is not graded. It is only intended for you to apply the knowledge you have gained to solve real-world problems.

Access: Click on the Labs tab on the left side panel of the LMS. Copy or note the username and password that are generated. Click on the Launch Lab button. On the page that appears, enter the username and password in the respective fields, and click Login.

Step1: Data Import

Code

```
import pandas as pd
import numpy as np
customer_data = pd.read_csv('shopping_data.csv’)
customer_data
```



Out[6]:

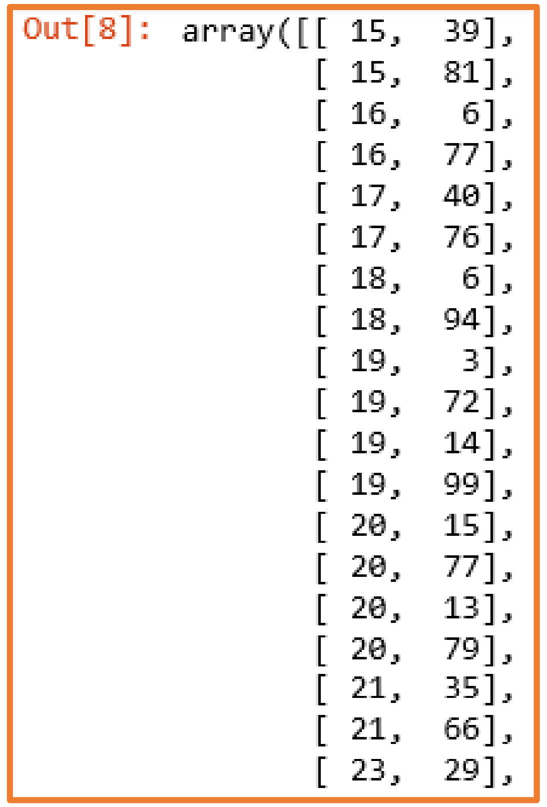
	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
5	6	Female	22	17	76
6	7	Female	35	18	6
7	8	Female	23	18	94
8	9	Male	64	19	3
9	10	Female	30	19	72
10	11	Male	67	19	14
11	12	Female	35	19	99
12	13	Female	58	20	15
13	14	Female	24	20	77
14	15	Male	37	20	13

Step 2: Filter Columns

Discard all the data, except annual income (in thousands of dollars) and spending score (1-100)

Code

```
data = customer_data.iloc[:,3:5].values
data
```

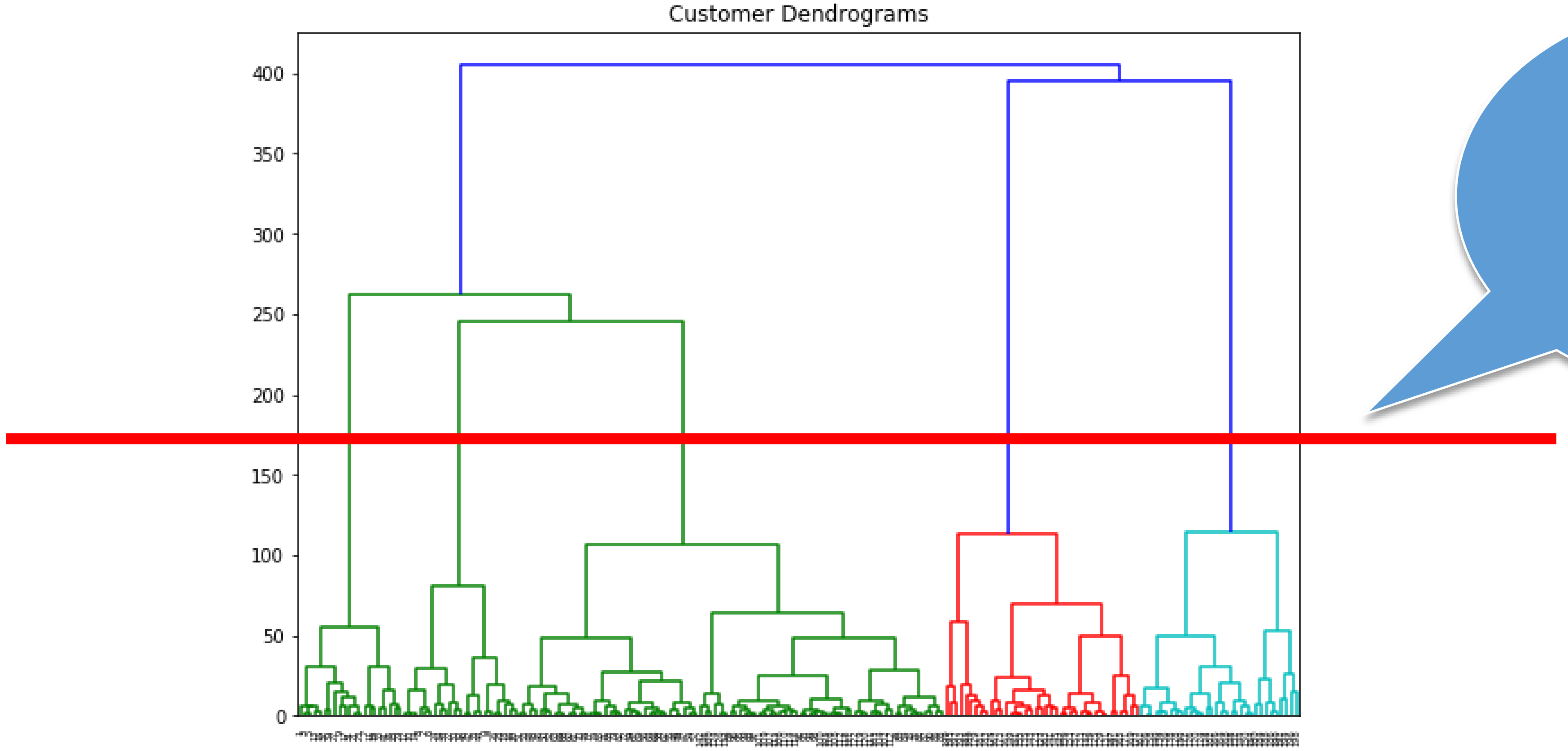


```
Out[8]: array([[ 15,  39],
               [ 15,  81],
               [ 16,   6],
               [ 16,  77],
               [ 17,  40],
               [ 17,  76],
               [ 18,   6],
               [ 18,  94],
               [ 19,   3],
               [ 19,  72],
               [ 19,  14],
               [ 19,  99],
               [ 20,  15],
               [ 20,  77],
               [ 20,  13],
               [ 20,  79],
               [ 21,  35],
               [ 21,  66],
               [ 23,  29],
```

Step 3: Create Dendrograms

Code

```
import matplotlib.pyplot as plt
%matplotlib inline
import scipy.cluster.hierarchy as shc
plt.figure(figsize=(10,7))
plt.title('Customer Dendrograms')
dend = shc.dendrogram(shc.linkage(data,method='ward'))
```



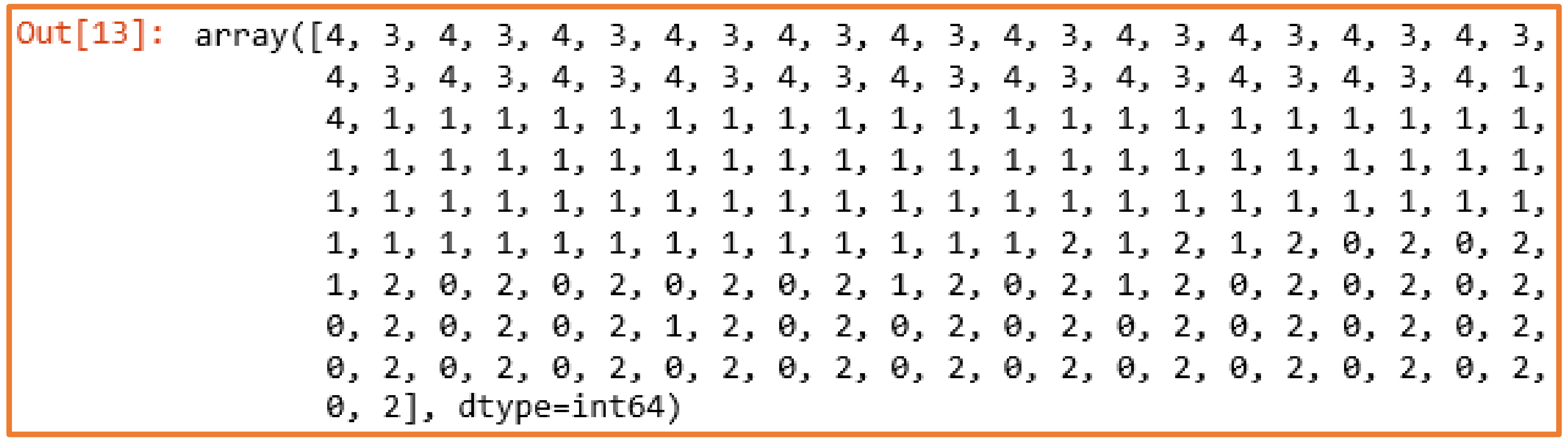
5 Clusters

Step 4: Agglomerative Clustering

Since there are five clusters, group the data points into these five clusters

Code

```
from sklearn.cluster import AgglomerativeClustering
cluster = AgglomerativeClustering(n_clusters=5, affinity='euclidean',
linkage='ward')
cluster.fit_predict(data)
```



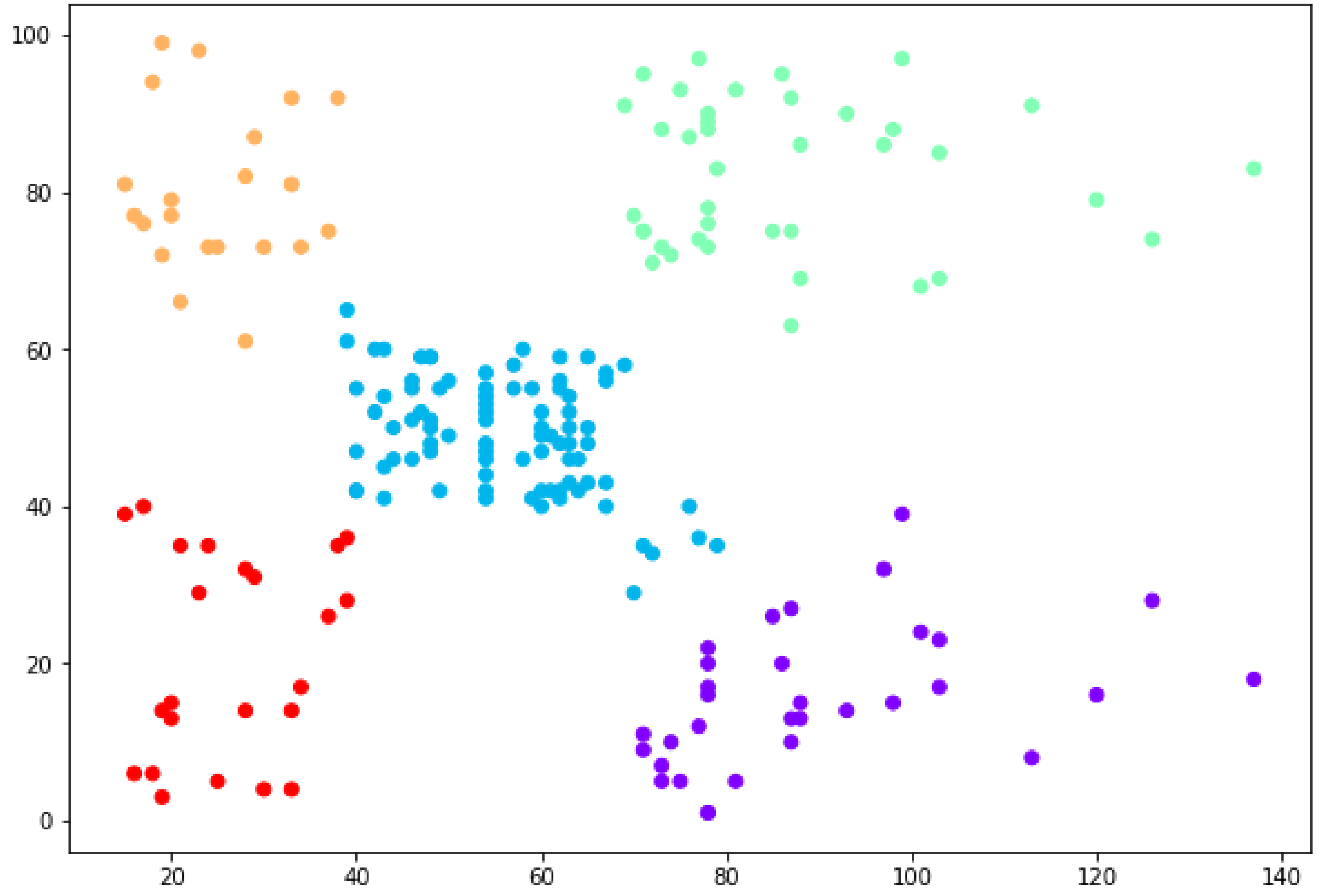
```
Out[13]: array([4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3,
                4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 1,
                4, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 2, 0, 2, 0, 2,
                1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 1, 2, 0, 2, 1, 2, 0, 2, 0, 2, 0, 2,
                0, 2, 0, 2, 0, 2, 1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
                0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
                0, 2], dtype=int64)
```

Step 4: Plotting the Clusters

Code

```
plt.figure(figsize=(10, 7))
plt.scatter(data[:,0], data[:,1], c=cluster.labels_, cmap='rainbow')
```

```
Out[14]: <matplotlib.collections.PathCollection at 0x18268c22cc0>
```



Unsupervised Learning

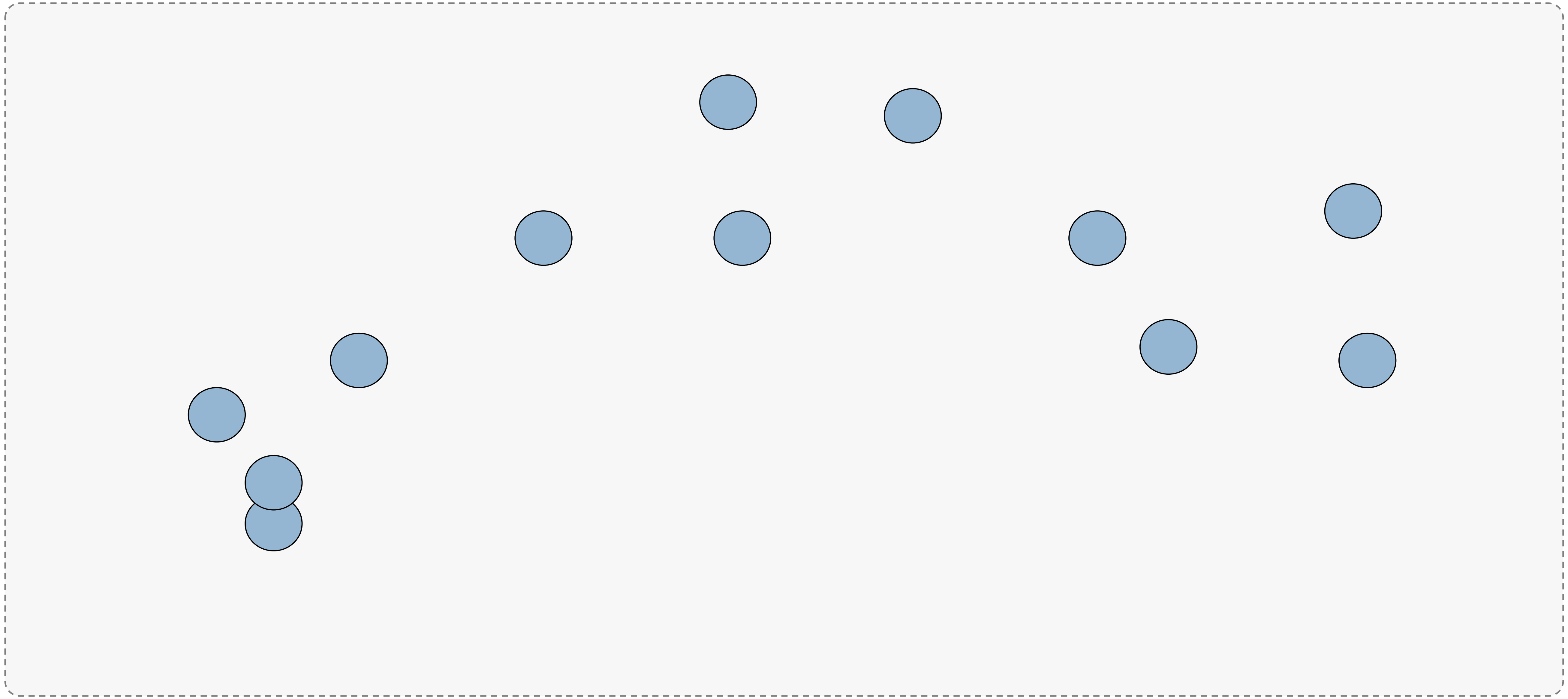
Topic 4: K-means Clustering

K-means Algorithm: Steps

1. Randomly chooses k datapoints as initial centroids
2. Assigns each datapoint closest to the centroid
3. Calculates new cluster centroids
4. Checks if the convergence criterion is met

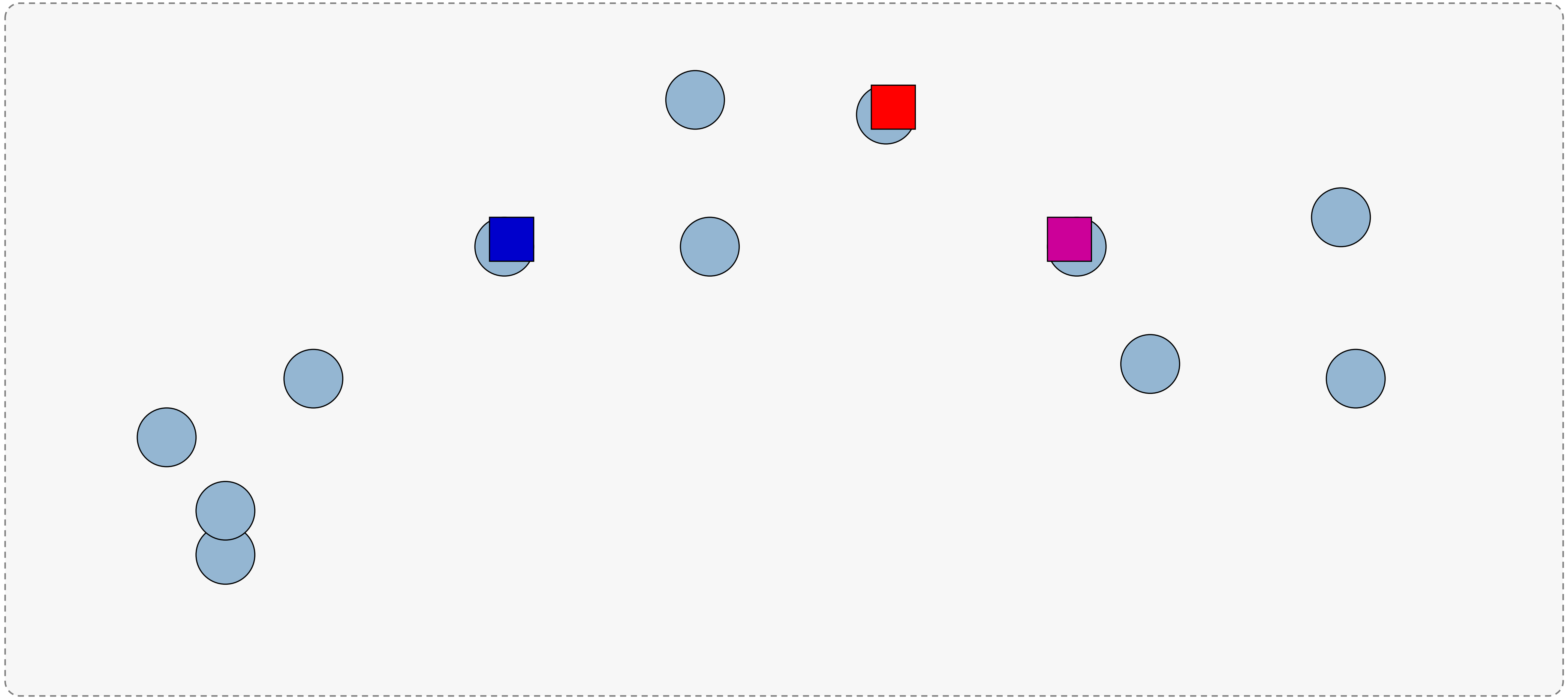
K-means: Example

Consider the below datapoints



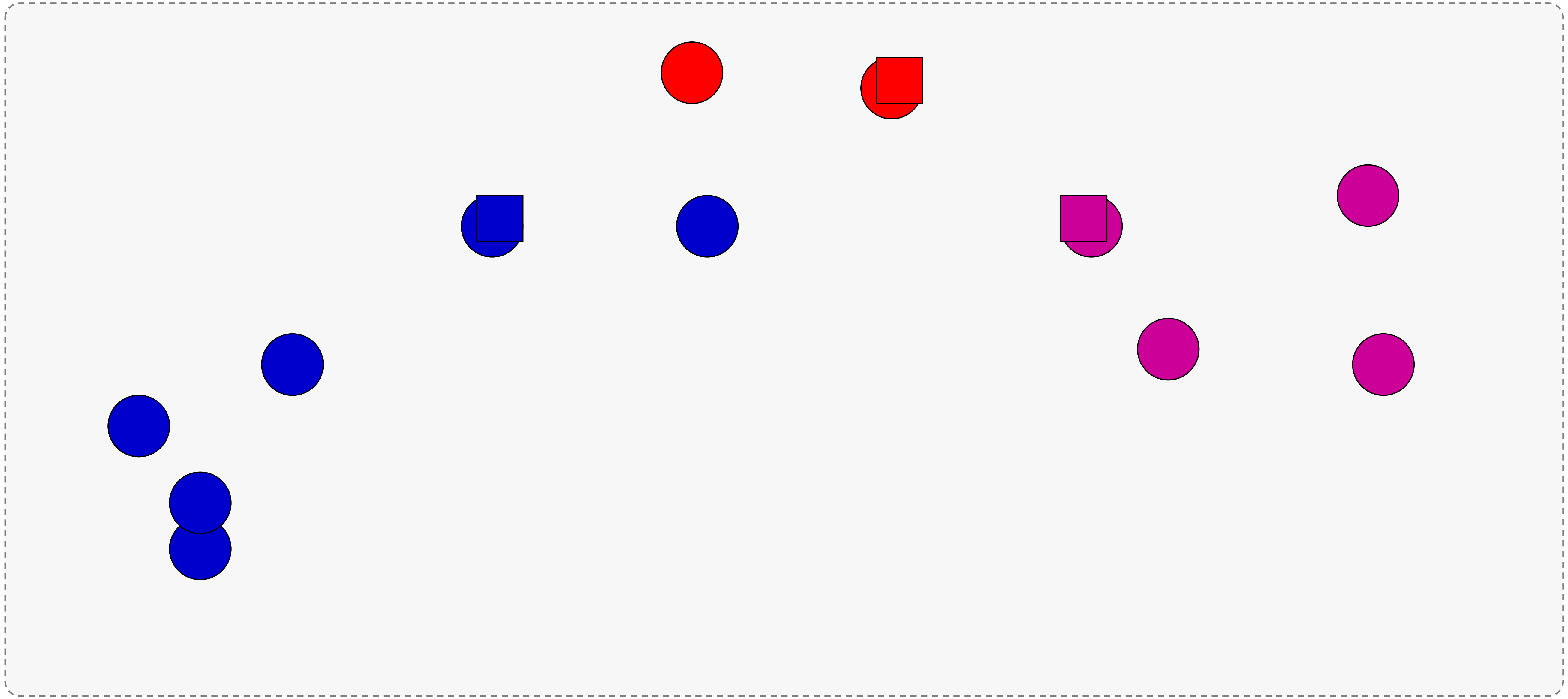
K-means: Example (Contd.)

Initialize centers randomly



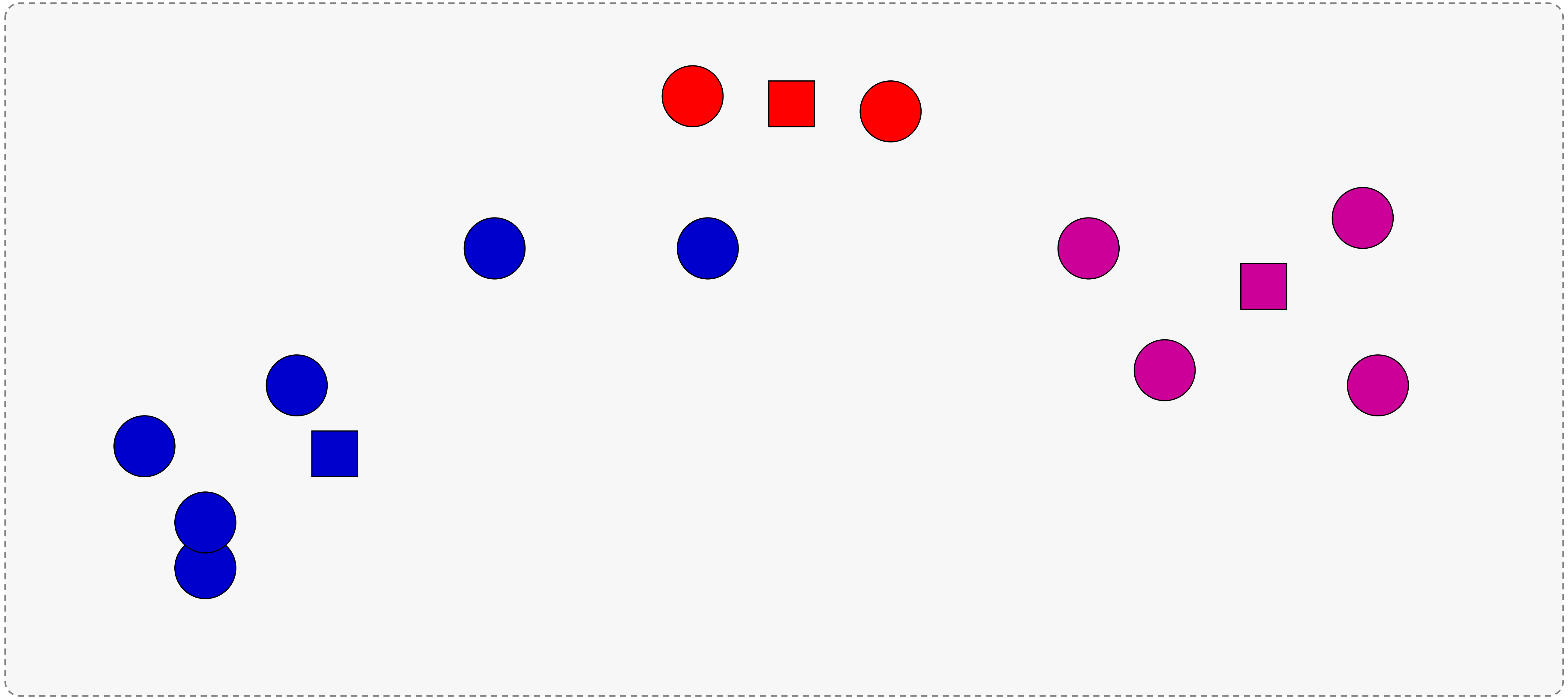
K-means: Example (Contd.)

Assign points to the nearest center



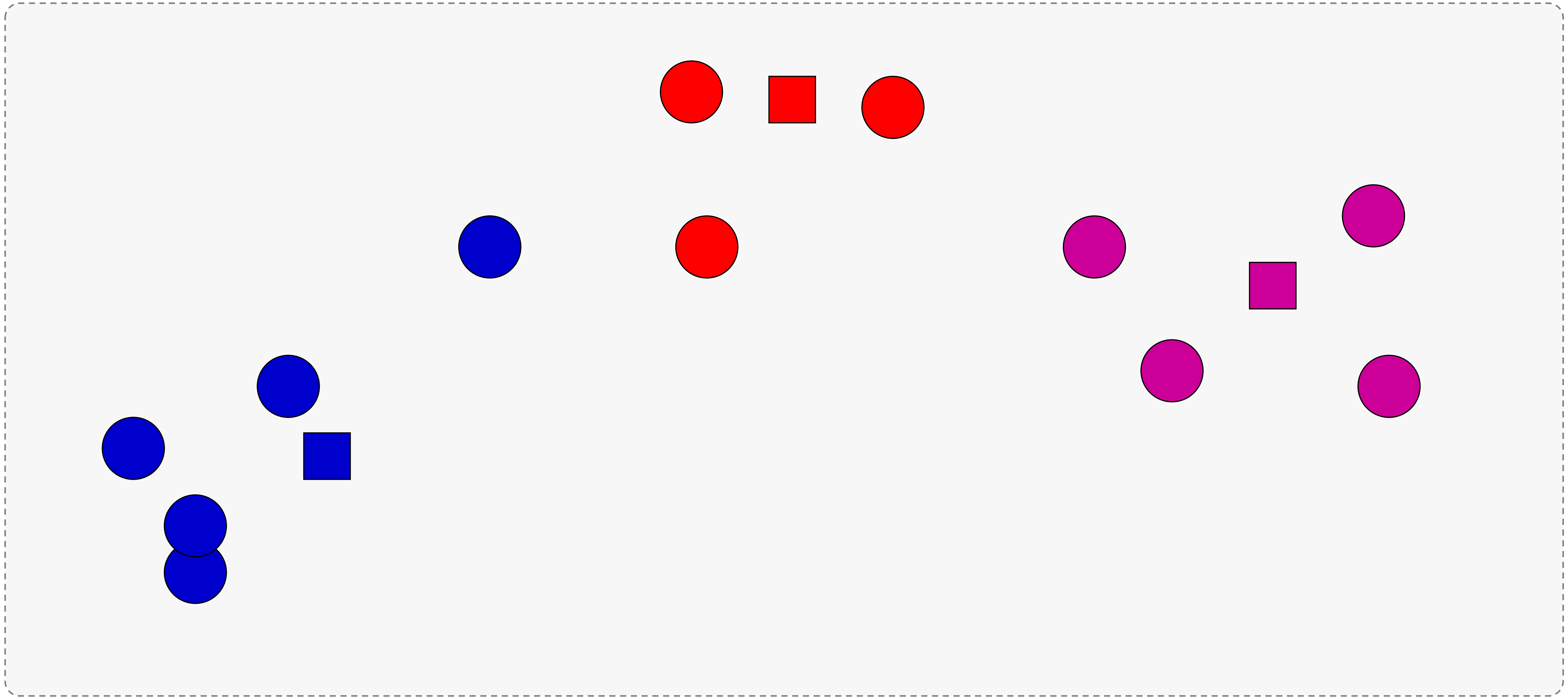
K-means: Example (Contd.)

Readjust centers



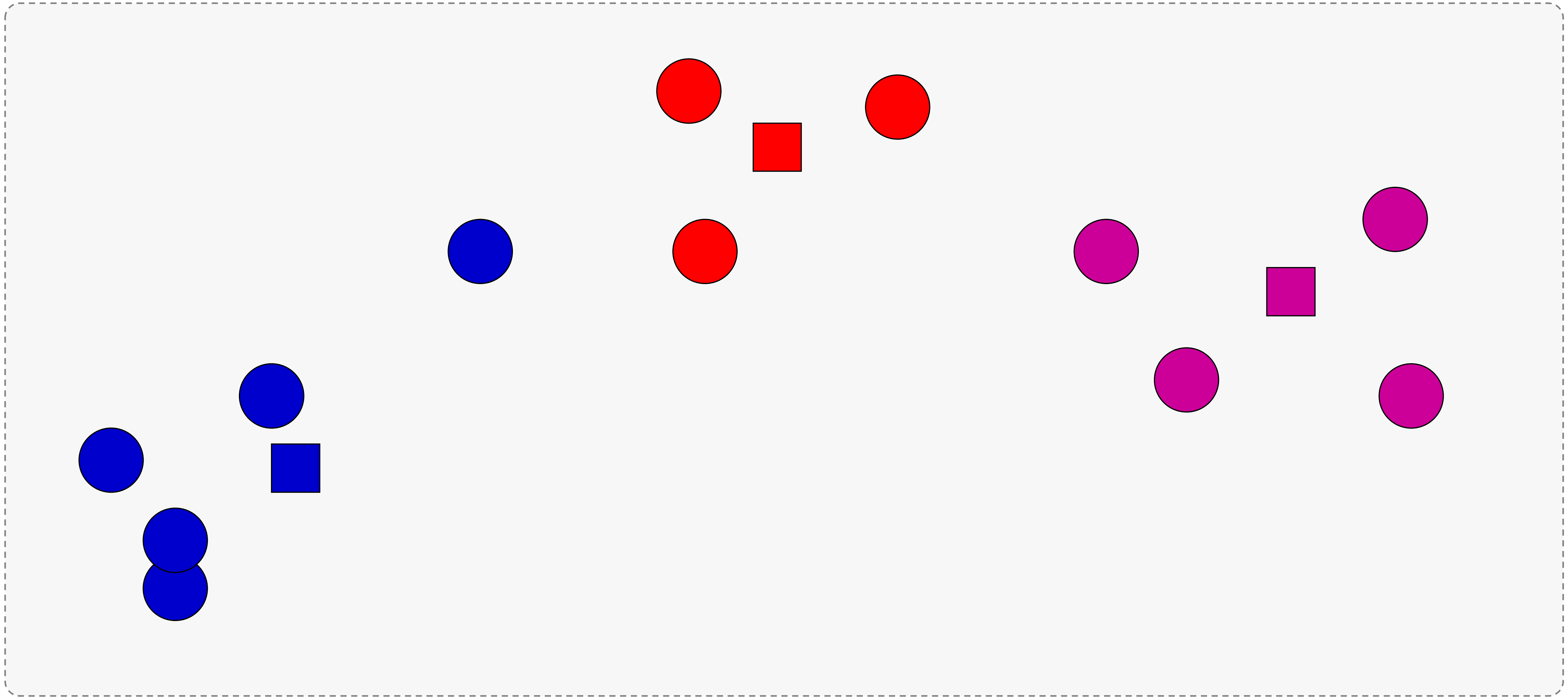
K-means: Example (Contd.)

Assign points to the nearest center



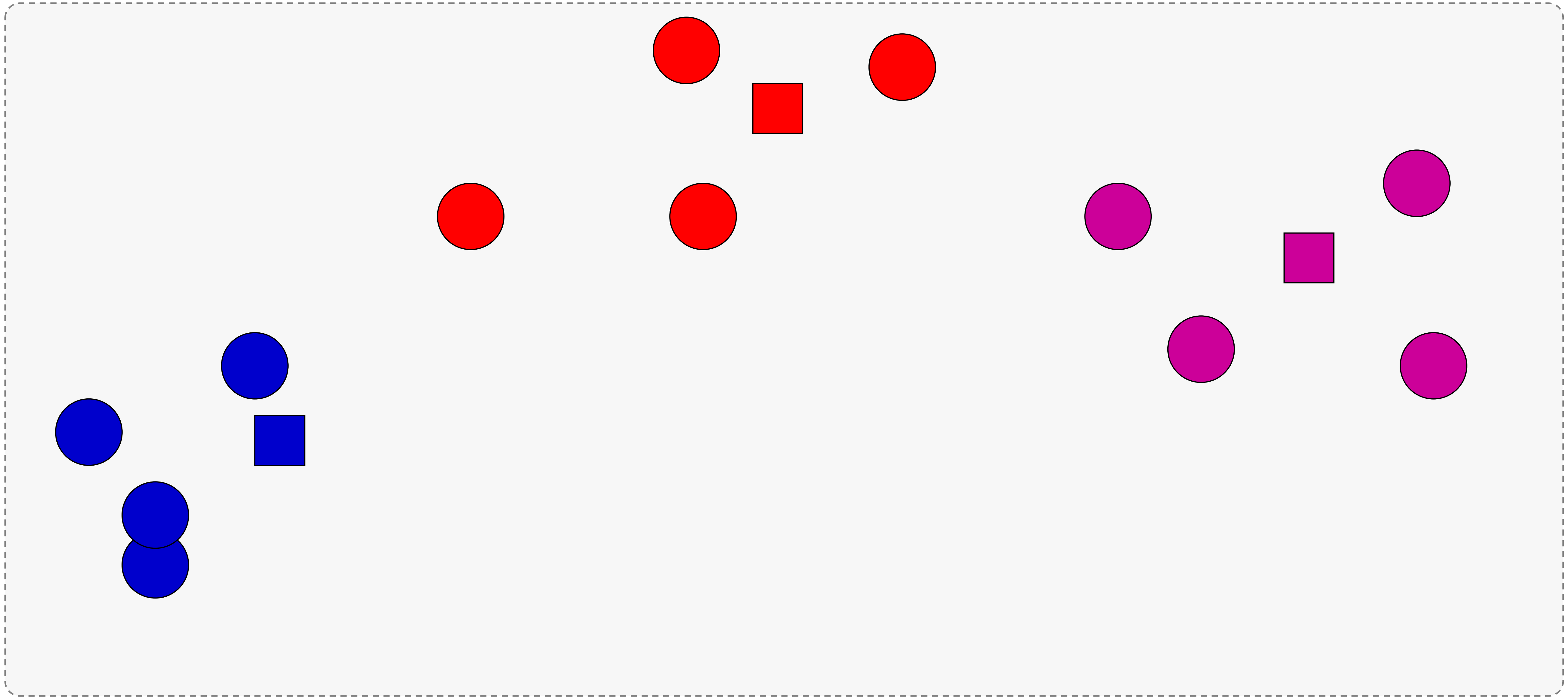
K-means: Example (Contd.)

Re-adjust centres



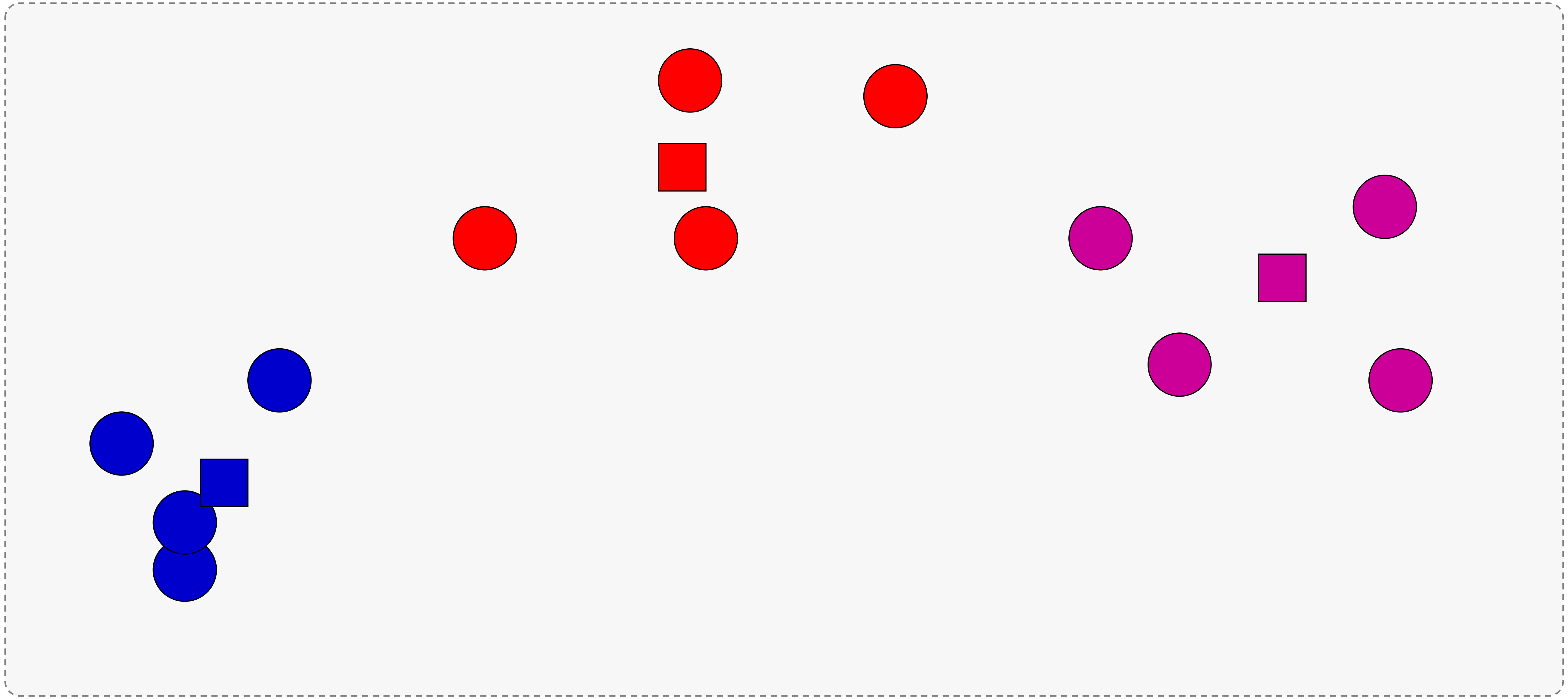
K-means: Example (Contd.)

Assign points to the nearest center



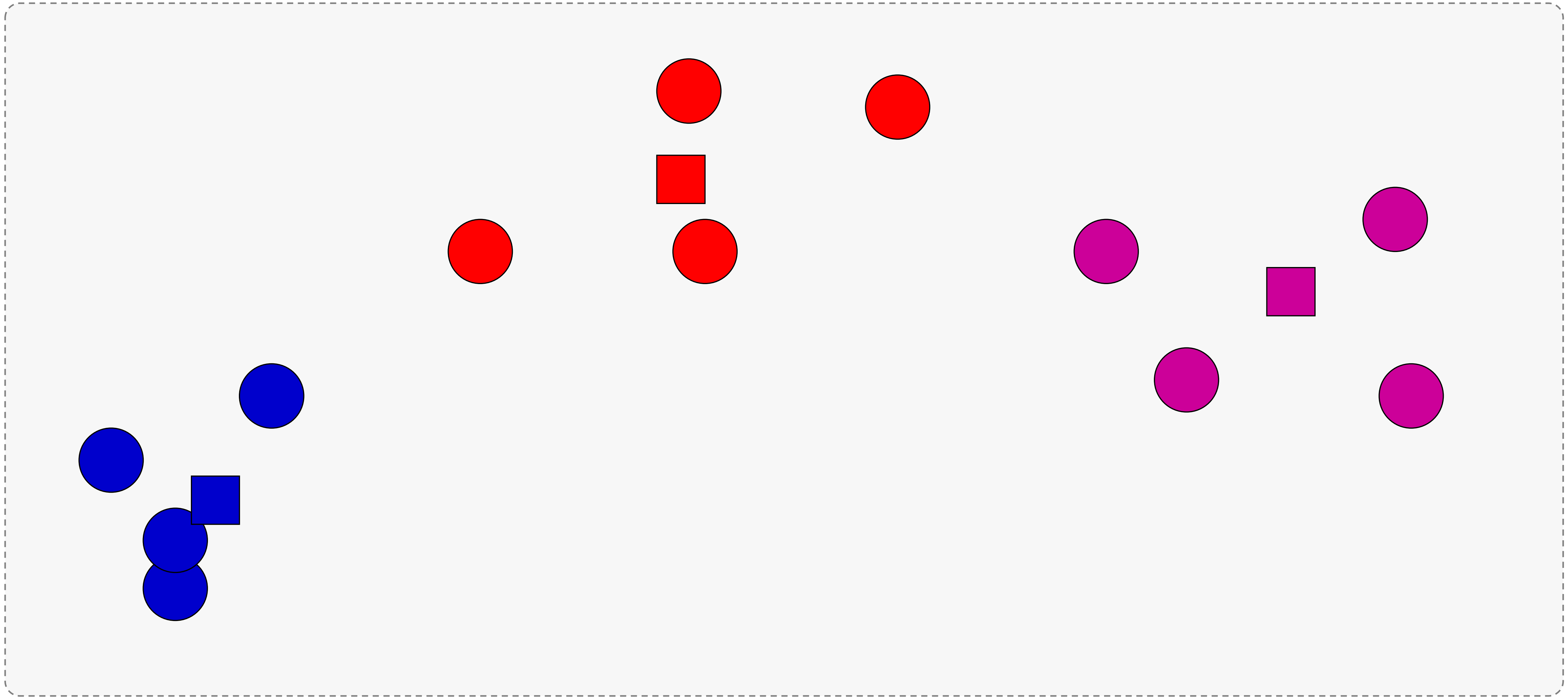
K-means: Example (Contd.)

Readjust centers



K-means: Example (Contd.)

Assign points to the nearest center

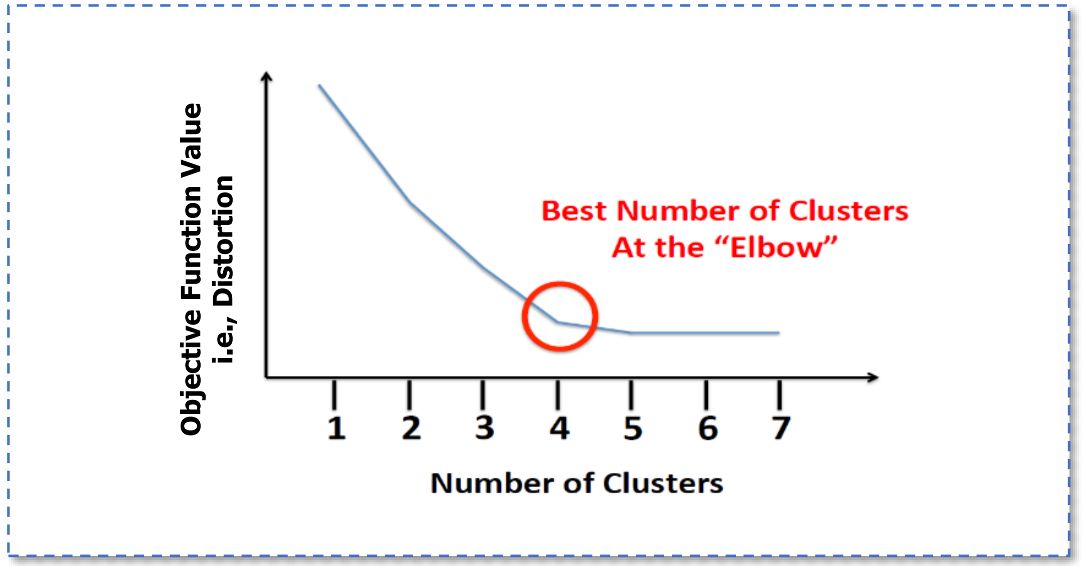


Optimal Number of Clusters

If you plot k against the SSE, you will see that the error decreases as k increases

This is because their size decreases and hence distortion is also smaller"

The goal of elbow method is to choose k where SSE decreases abruptly



Elbow Plot

Assisted Practice

K-means Clustering

Duration: 15 mins.

Problem Statement: Lithionpower is the largest provider of electric vehicle(e-vehicle) batteries.
It provides battery on a rental model to e-vehicle drivers. Drivers rent battery typically for a day and then replace it with a charged battery from the company.
Lithionpower has a variable pricing model based on driver's driving history. Battery life depends on factors such as over speeding, distance driven per day, etc.

Objective:

- Create a cluster model where drivers can be grouped together based on the driving data.
- Group the datapoints so that drivers will be incentivized based on the cluster.

Access: Click on the Labs tab on the left side panel of the LMS. Copy or note the username and password that are generated. Click on the Launch Lab button. On the page that appears, enter the username and password in the respective fields, and click Login.

Unassisted Practice

K-means Clustering

Duration: 10 mins.

Problem Statement: There is an image in the name of “tiger.png”. Use k-means clustering with k set to 16 and cluster the image, which means that you want to keep just 16 colors in our compressed image.

Objective: Open and display the image “tiger.png”. Convert the image into numpy array, so that it can be used in further processing. Find out the dimensions of the image and convert it into a two-dimensional array (Use k-means clustering for image segmentation, reducing the image into 16 colors).

Note: This practice is not graded. It is only intended for you to apply the knowledge you have gained to solve real-world problems.

Access: Click on the Labs tab on the left side panel of the LMS. Copy or note the username and password that are generated. Click on the Launch Lab button. On the page that appears, enter the username and password in the respective fields, and click Login.

Step 1: Import Libraries

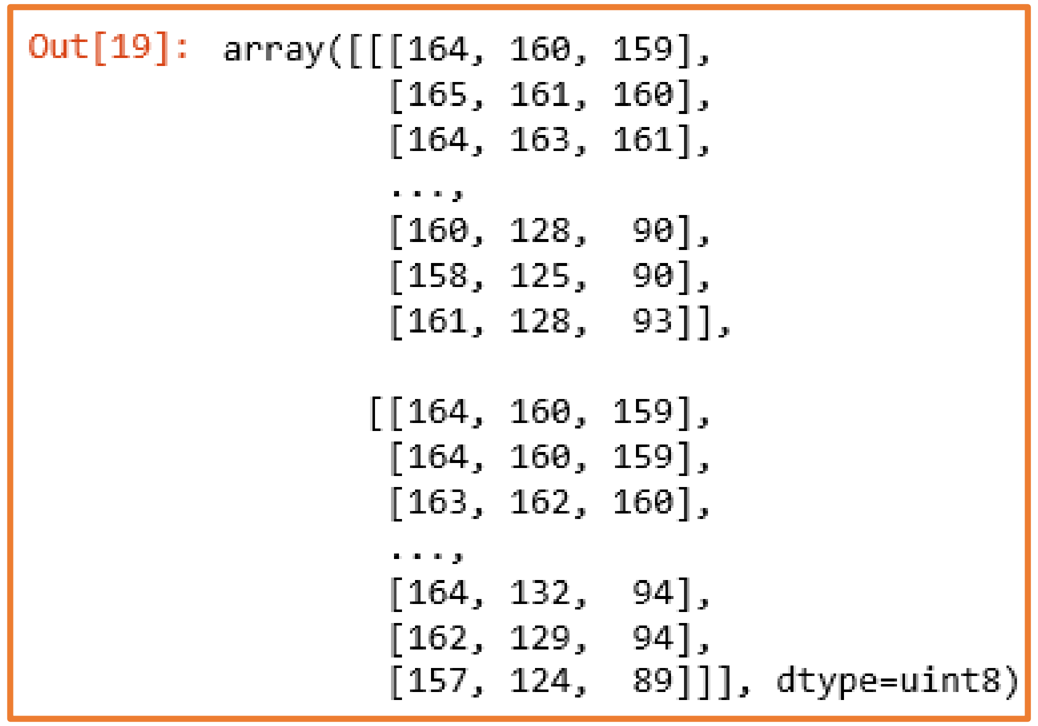
Code

```
from sklearn.cluster import KMeans
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import os
%matplotlib inline
```

Step 2: Get the Image and its Corresponding RGB Values

Code

```
img = Image.open('tiger.png’)
img_np=np.asarray(img)
img_np[0:2]
```



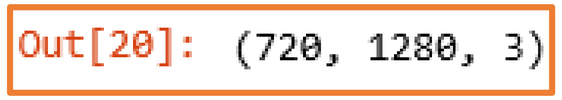
```
Out[19]: array([[[164, 160, 159],
                 [165, 161, 160],
                 [164, 163, 161],
                 ...,
                 [160, 128,  90],
                 [158, 125,  90],
                 [161, 128,  93]],

                [[164, 160, 159],
                 [164, 160, 159],
                 [163, 162, 160],
                 ...,
                 [164, 132,  94],
                 [162, 129,  94],
                 [157, 124,  89]]], dtype=uint8)
```

Step 3: Get the Image Dimensions

Code

```
img_np.shape
```



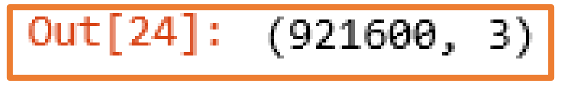
```
Out[20]:  (720, 1280, 3)
```

For feeding this data into the algorithm, you must change the shape of this data into a dataset with 720*1280 = 921600 rows and 3 columns

Step 4: Reshape the Data

Code

```
pixels=img_np.reshape(img_np.shape[0]*img_np.shape[1],img_np.shape[2])
pixels.shape
```

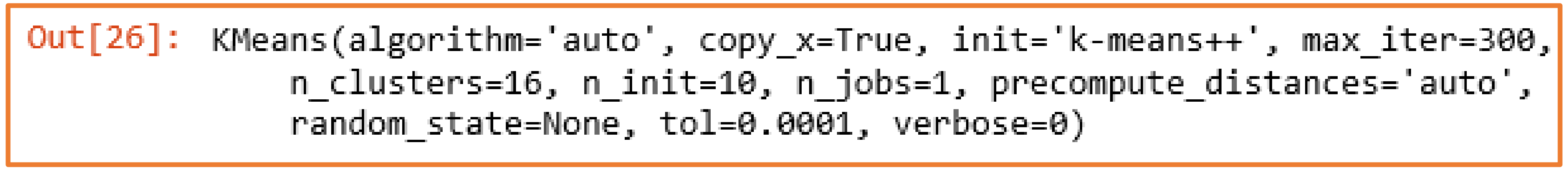


```
Out[24]:  (921600, 3)
```

Step 5: Define the K-means Model

Code

```
model=KMeans(n_clusters=16)
model.fit(pixels)
```



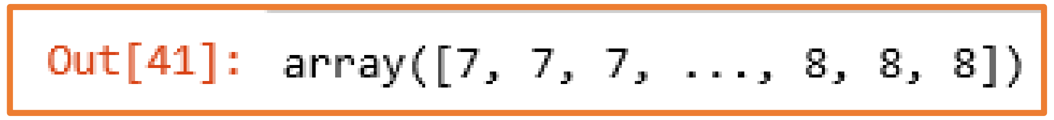
```
Out[26]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
             n_clusters=16, n_init=10, n_jobs=1, precompute_distances='auto',
             random_state=None, tol=0.0001, verbose=0)
```

After the model is trained, model.labels_ is used to obtain the number of cluster that is assigned to each data point or each pixel.
model.cluster_centers_ gives us the coordinates or the RGB values of the 16 cluster centers.

Step 6: Define the Cluster Centres

Code

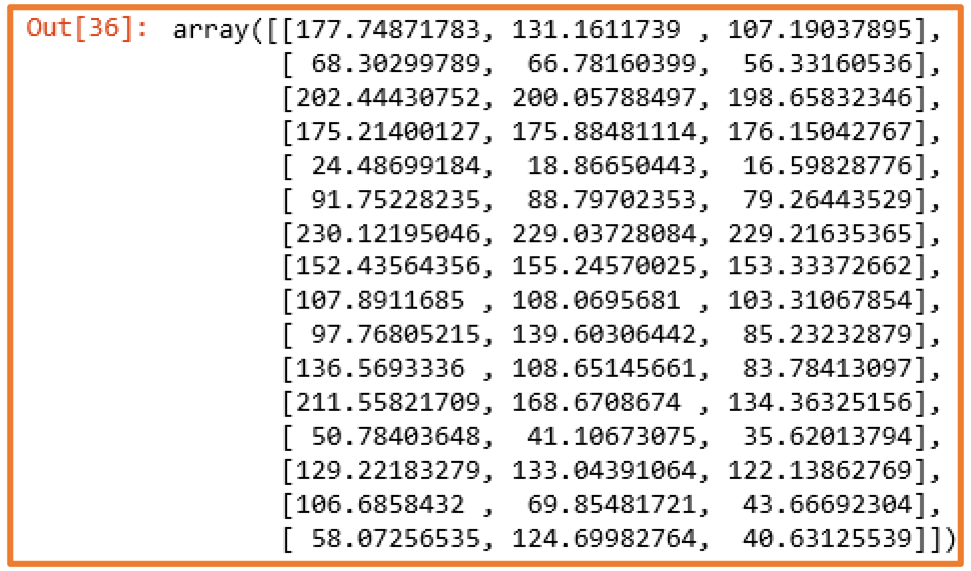
```
pixel_centroids = model.labels_
cluster_centers=model.cluster_centers_
pixel_centroids
```



```
Out[41]:  array([7, 7, 7, ..., 8, 8, 8])
```

Code

```
cluster_centers
```

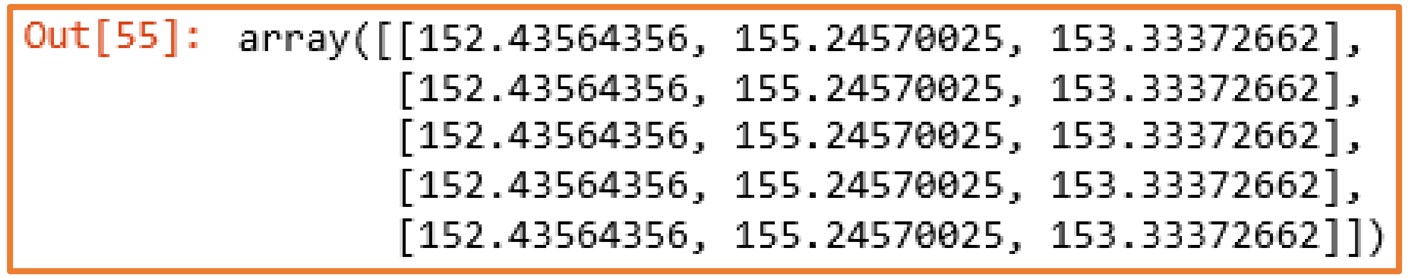


```
Out[36]:  array([[177.74871783, 131.1611739 , 107.19037895],
                 [ 68.30299789,  66.78160399,  56.33160536],
                 [202.44430752, 200.05788497, 198.65832346],
                 [175.21400127, 175.88481114, 176.15042767],
                 [ 24.48699184,  18.86650443,  16.59828776],
                 [ 91.75228235,  88.79702353,  79.26443529],
                 [230.12195046, 229.03728084, 229.21635365],
                 [152.43564356, 155.24570025, 153.33372662],
                 [107.8911685 , 108.0695681 , 103.31067854],
                 [ 97.76805215, 139.60306442,  85.23232879],
                 [136.5693336 , 108.65145661,  83.78413097],
                 [211.55821709, 168.6708674 , 134.36325156],
                 [ 50.78403648,  41.10673075,  35.62013794],
                 [129.22183279, 133.04391064, 122.13862769],
                 [106.6858432 ,  69.85481721,  43.66692304],
                 [ 58.07256535, 124.69982764,  40.63125539]])
```

Step 7: Cluster Assignment

Code

```
final=np.zeros((pixel_centroids.shape[0],3))
for cluster_no in range(16):
    final[pixel_centroids==cluster_no]=cluster_centers[cluster_no]
final[0:5]
```

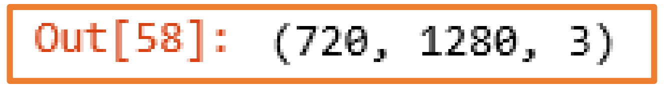


```
Out[55]: array([[152.43564356, 155.24570025, 153.33372662],
                [152.43564356, 155.24570025, 153.33372662],
                [152.43564356, 155.24570025, 153.33372662],
                [152.43564356, 155.24570025, 153.33372662],
                [152.43564356, 155.24570025, 153.33372662]])
```

Step 8: Reshape to Original Dimensions

Code

```
comp_image=final.reshape(img_np.shape[0],img_np.shape[1],3)
comp_image.shape
```



```
Out[58]:  (720, 1280, 3)
```

Step 9: Convert the Pixel Values to Image

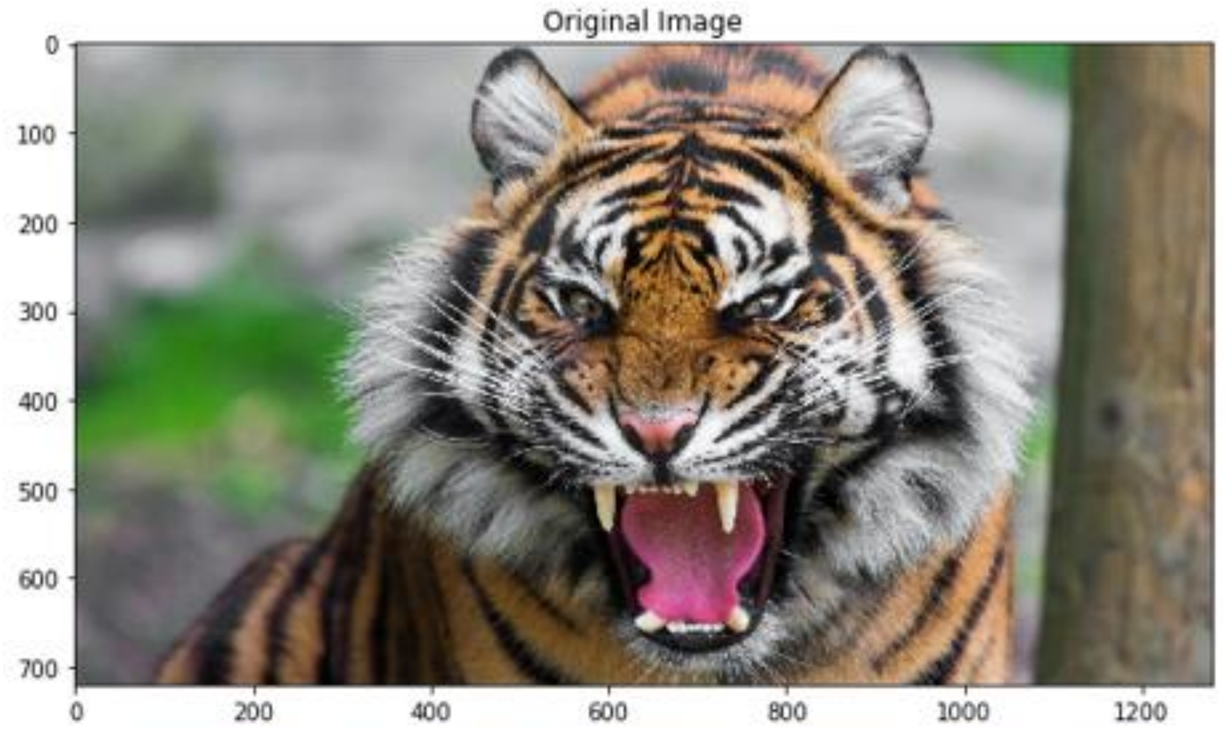
Code

```
comp_image=Image.fromarray(np.uint8(comp_image))
comp_image.save('tiger_compressed.png’)
img_1 = mpimg.imread('tiger.png')
img_2 = mpimg.imread('tiger_compressed.png')
```

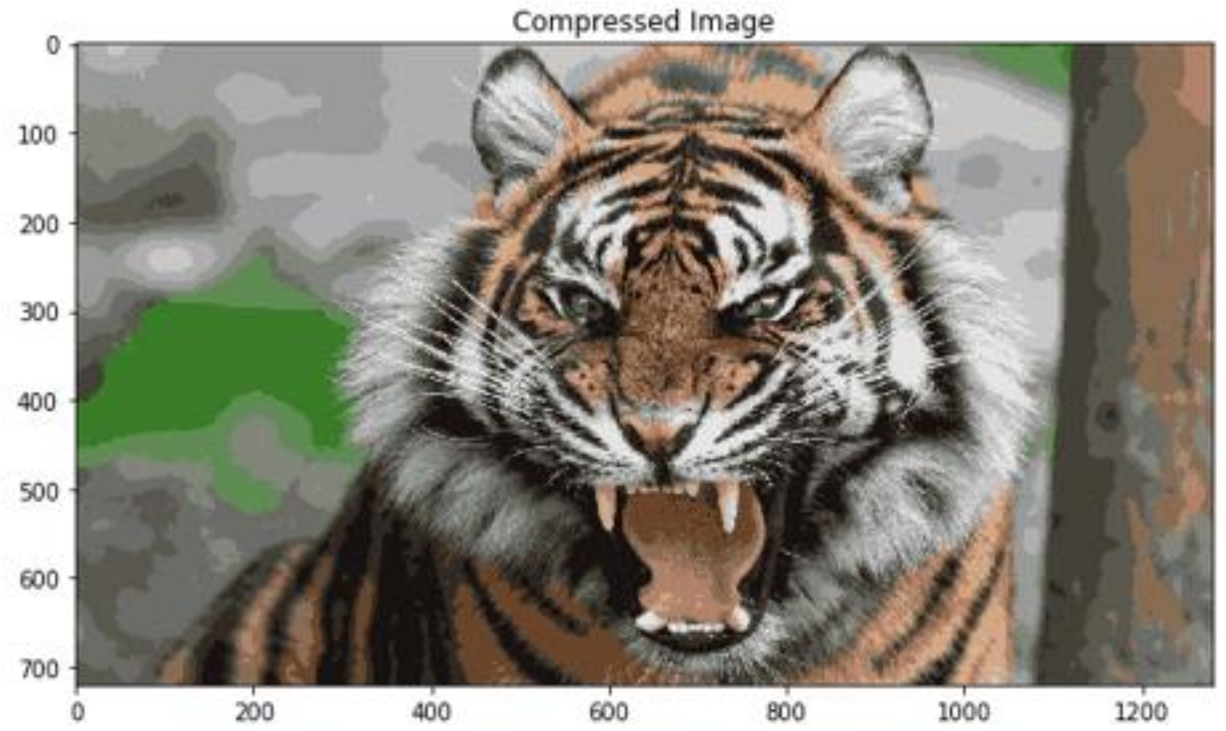
Step 10: Original Plot vs. Compressed Image

Code

```
fig,(ax1,ax2) = plt.subplots(1,2, figsize=(20,20))
ax1.imshow(img_1)
ax1.set_title('Original Image')
ax2.imshow(img_2)
ax2.set_title('Compressed Image')
plt.show()
```



Original Image



Compressed Image

Key Takeaways

Now, you are able to:

- ✓ Explain the mechanism of unsupervised learning
- ✓ Practice different clustering techniques in Python

Knowledge Check

Knowledge Check

1

Can decision trees be used for performing clustering?

a. True

b. False

Knowledge Check

1

Can decision trees be used for performing clustering?

a. True

b. False

The correct answer is **a. True**

Decision trees can also be used to for clusters in the data, but it often generates natural clusters and is not dependent on any objective function.

Knowledge Check

2

Which of the following can act as possible termination condition in K-Means?

1. Fixed number of iterations.
2. Assigning observations to clusters such that they don’t change between iterations, except for cases with a bad local minimum.
3. Stationary centroids appear between successive iterations.
4. When RSS falls below a threshold.

a. 1,3, and 4

b. 1, 2, and 3

c. 1, 2, and 4

d. All the above

Knowledge Check

2

Which of the following can act as possible termination condition in K-Means?

1. Fixed number of iterations.
2. Assigning observations to clusters such that they don’t change between iterations, except for cases with a bad local minimum.
3. Stationary centroids appear between successive iterations.
4. When RSS falls below a threshold.

a. 1,3, and 4

b. 1, 2, and 3

c. 1, 2, and 4

d. All the above

The correct answer is **d. All the above**

All the above options are true.

Lesson-End Project

Duration: 20 mins.

Problem Statement: Open and display the image “dog.jpeg”. The image has to be converted in to numpy array, so that it can be used in further processing. The major challenge is to identify the dominant color in the image [Hint: Refer the following url for image processing documentation: http://omz-software.com/pythonista/docs/ios/PIL.html]

Objective: Use K-means clustering for image segmentation, which will include the following steps:

- Find out the dimensions of the image and convert it in to a two-dimensional array.
- Use k-means clustering with k set to 3 and cluster the image.
  [Hint: Refer to k-means module of scikit learn]
- Predict the cluster label of every pixel in the image and plot it back as an image.
- Find out the three dominant color in the image.
  [Hint: The cluster centers should correspond to three dominant colors]

Access: Click the Labs tab in the left side panel of the LMS. Copy or note the username and password that are generated. Click the Launch Lab button. On the page that appears, enter the username and password in the respective fields and click Login.

Thank You