

Forward School

Program Code: J620-002-4:2020

Program Name: FRONT-END SOFTWARE DEVELOPMENT

Title : Exe25 - k-Means Exercise

Name: Ooi Caaron

IC Number:990701-07-5837

Date :29/7/23

Introduction :

Conclusion :

Exercise 1: Build and Plot k-Means

In [22]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Step 1: create blobs with the size of 500, and center of 3

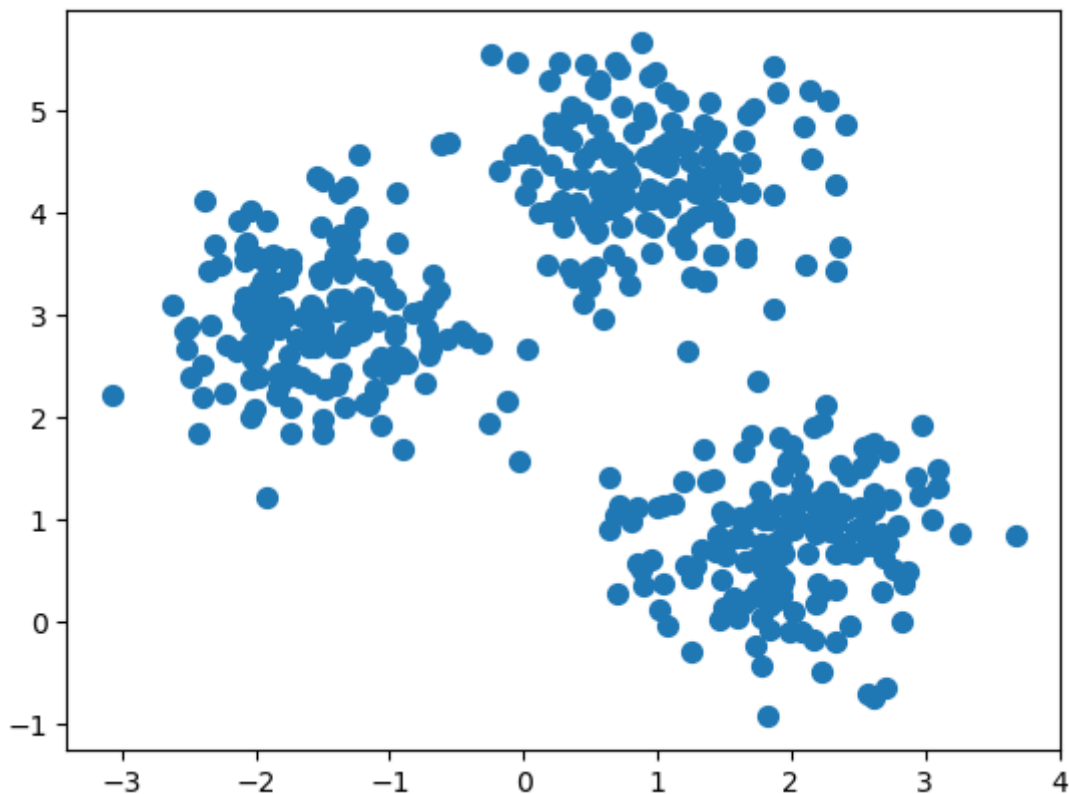
In [23]:

```
from sklearn.datasets import make_blobs
X, y_true = make_blobs(n_samples=500, centers=3,
                       cluster_std=0.60, random_state=0)
```

Step 2: Plot the distribution of the blobs

In [24]:

```
plt.scatter(X[:, 0], X[:, 1], s=50);
```



Step 3: Use K-means, find the centers of these clusters

In [25]:

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=4)
kmeans.fit(X)
y_kmeans = kmeans.predict(X)
```

C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster_kmeans.py:1412: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning

```
super()._check_params_vs_input(X, default_n_init=10)
```

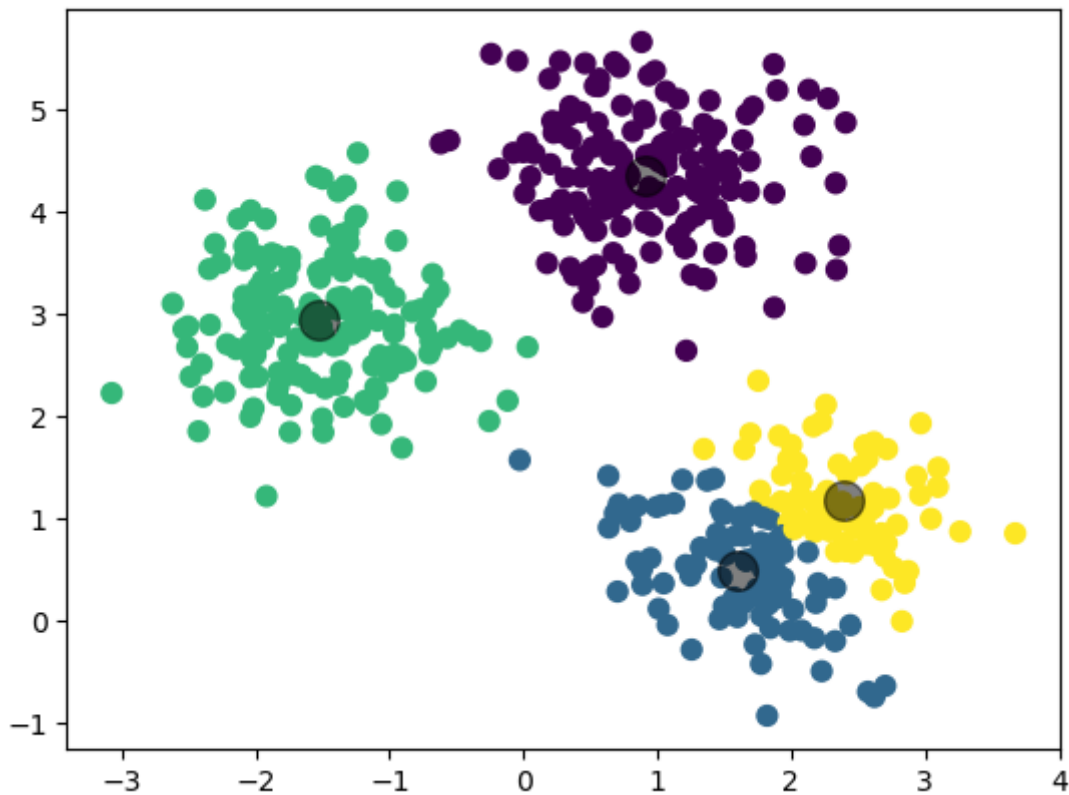
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=2.

```
warnings.warn(
```

Step 4: Plot the blobs with the found centers

In [26]:

```
plt.scatter(X[:, 0], X[:, 1], c=y_kmeans, s=50, cmap='viridis')  
  
centers = kmeans.cluster_centers_  
plt.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5);
```



Additional/Optional:

Step 5: How can you find out the automatically assigned "labels" in the produced clusters?

In []:

Step 6: How about classes? How to find out where there are classes.

In []:

Exercise 2: k-Means with the Iris dataset

Step 1: Load the iris dataset from sklearn and other necessary libraries

In [27]:

```
import numpy as np
from sklearn.ensemble import RandomForestClassifier
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import SelectFromModel
from sklearn.metrics import accuracy_score
```

Step 2: Set the training and target data as X and y respectively. Display the targets.

In [28]:

```
# Load the iris dataset
iris = datasets.load_iris()

X = iris.data

y = iris.target
```

Introducing - *the Elbow Method*: A technique to allow you to identify the best K

General idea: iterate the creation of k-Means clusters with increasing sizes, and record down the value of `kmeans.inertia_` (`inertia_`: Sum of squared distances of samples to their closest cluster center.)

Step 3: create a list named `wcss` and store the inertia values for a selected range of ks.

In [29]:

```
from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters=2, random_state=0)

kmeans.fit(X)
kmeans.inertia_
```

C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster_kmeans.py:1412: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning

```
super()._check_params_vs_input(X, default_n_init=10)
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
warnings.warn(
```

Out[29]:

152.34795176035792

Step 4: Plot a graph to look at 'The elbow'

In [30]:

```
from sklearn.cluster import KMeans
cs = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', max_iter = 300, n_init = 10, ran
    kmeans.fit(X)
    cs.append(kmeans.inertia_)
plt.plot(range(1, 11), cs)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('CS')
plt.show()
```

```
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
```

```
warnings.warn(
```

```
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
```

```
warnings.warn(
```

```
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
```

```
warnings.warn(
```

```
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
```

```
warnings.warn(
```

```
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
```

```
warnings.warn(
```

```
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
```

```
warnings.warn(
```

```
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
```

```
warnings.warn(
```

```
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
```

```
warnings.warn(
```

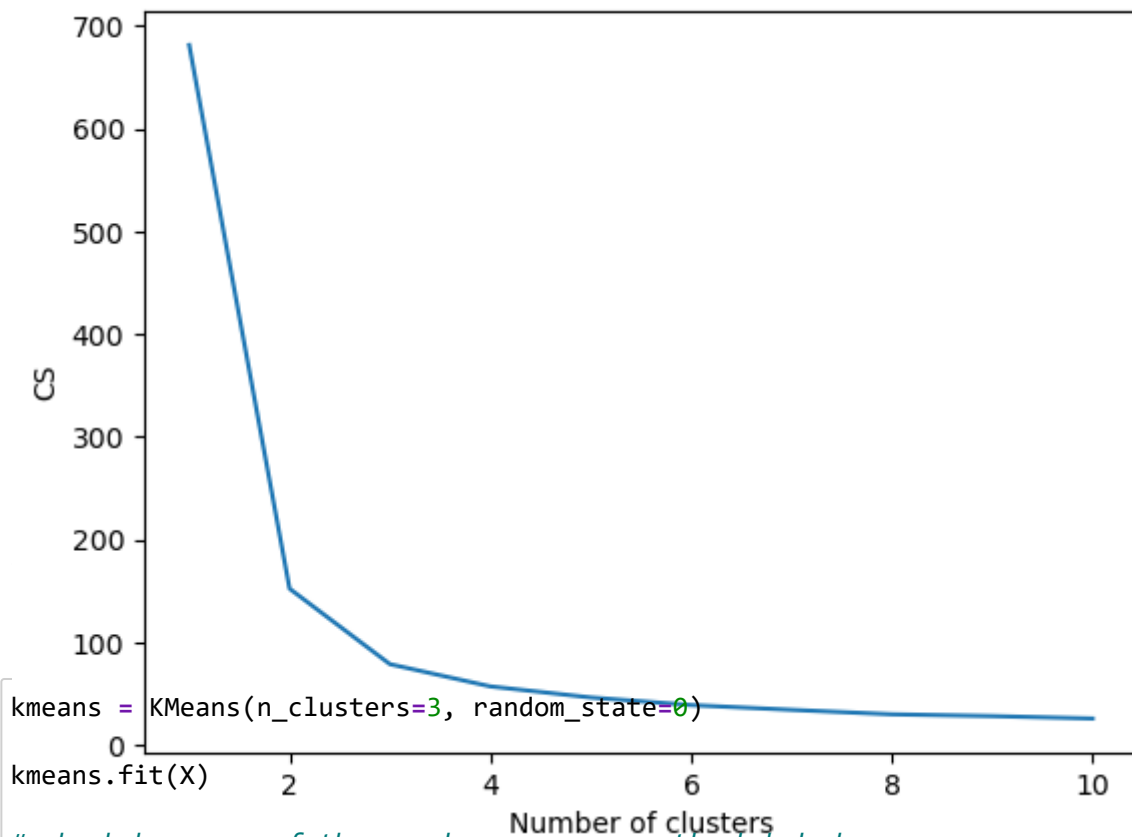
```
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
```

```
warnings.warn(
```

```
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
```

```
warnings.warn(
```

The Elbow Method



```
kmeans = KMeans(n_clusters=3, random_state=0)
kmeans.fit(X)

# check how many of the samples were correctly labeled
labels = kmeans.labels_

correct_labels = sum(y == labels)
print("Result: %d out of %d samples were correctly labeled." % (correct_labels, y.size))
print('Accuracy score: {0:0.2f}'.format(correct_labels/float(y.size)))
```

Result: 36 out of 150 samples were correctly labeled.
Accuracy score: 0.24

C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster_kmeans.py:1412: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning

```
super()._check_params_vs_input(X, default_n_init=10)
C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster\_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
warnings.warn(
```

Step 6: Visualize the clusters. Name the clusters accordingly, and also plot the centroids.

In [37]:

```

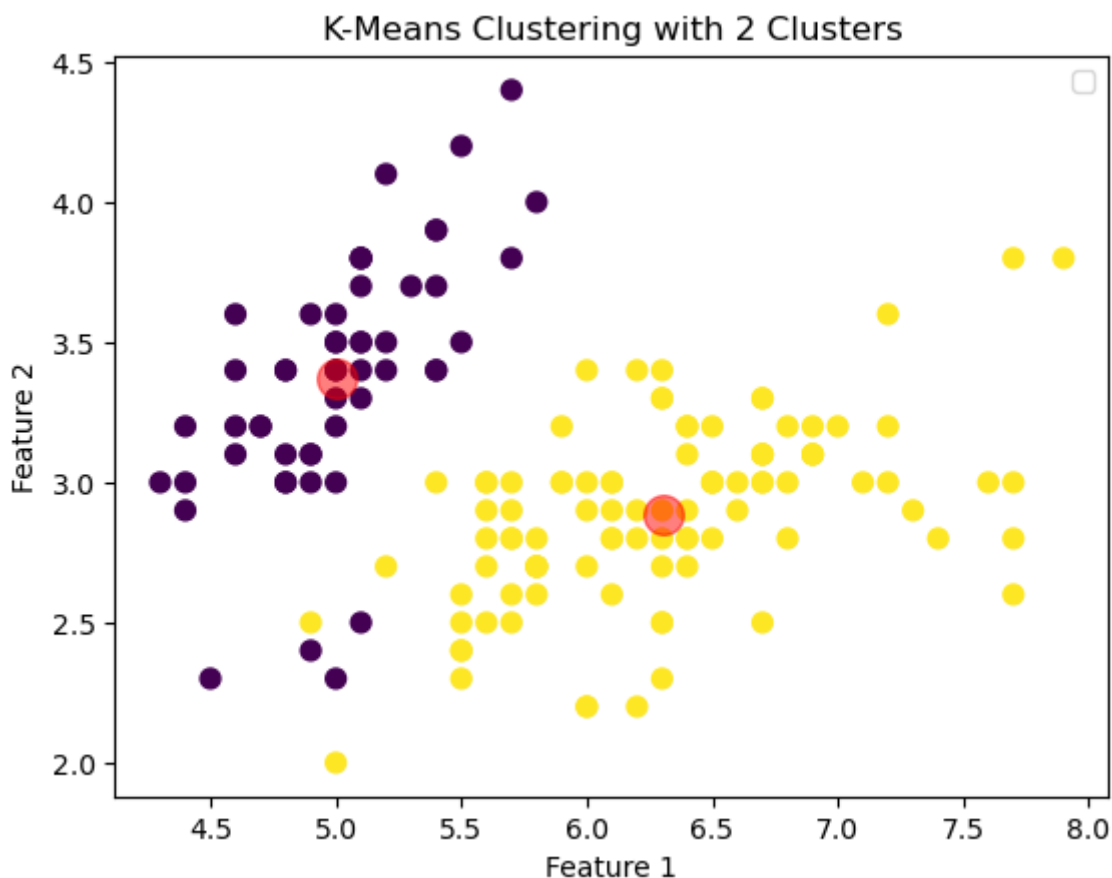
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
# Best K value obtained from the Elbow Method
best_k = 2
# Initialize and fit the KMeans model with the best K value
kmeans = KMeans(n_clusters=best_k, init='k-means++', max_iter=300, n_init=10, random_state=0)
y_kmeans = kmeans.fit_predict(X)
# Visualize the clusters using a scatter plot
plt.scatter(X[:, 0], X[:, 1], c=y_kmeans, s=50, cmap='viridis')
centers = kmeans.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], c='red', s=200, alpha=0.5);
# Plot the centroids of the clusters
plt.title('K-Means Clustering with 2 Clusters')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.legend()
plt.show()

```

C:\Users\User\anaconda3\envs\python-dscourse\Lib\site-packages\sklearn\cluster_kmeans.py:1436: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.

warnings.warn(

No artists with labels found to put in legend. Note that artists whose label start with an underscore are ignored when legend() is called with no argument.



Additional/Optional:

In []: