Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

Question 1.

PROBLEM STATEMENT: Write a python program to carry out preprocessing of data sets such as: Encountering or Handling Missing Values.

OBJECTIVE: To understand the preprocessing of the data sets so that the efficiency of the model is not affected.

CODE:

```
import pandas as pd
import numpy as np

data = {
 'Fruit': ['Apple','Banana','Mango','Peach'],
 'Quan': [25, np.nan, np.nan, 22],
 'Price': [50, np.nan, 42, 62]
}

df = pd.DataFrame(data)
print("Original DataFrame:")

print(df)
df_dropped = df.dropna()

print("\nAfter dropping missing values:")
print(df_dropped)

df_filled_zero = df.fillna(0)
print("\nAfter filling missing values with 0:")
print(df_filled_zero)

df_filled_mean = df.copy()
df_filled_mean['Quan'].fillna(df_filled_mean['Quan'].mean(), inplace=True)
```

```
df_filled_mean['Price'].fillna(df_filled_mean['Price'].mean(), inplace=True)
print("\nAfter filling missing values with the mean:")
print(df_filled_mean)

df_filled_median = df.copy()
df_filled_median['Quan'].fillna(df_filled_median['Quan'].median(), inplace=True)
df_filled_median['Price'].fillna(df_filled_median['Price'].median(), inplace=True)

print("\nAfter filling missing values with the median:")
print(df_filled_median)
```

Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

OUTPUT:

```
Original DataFrame:
    Fruit  Quan  Price
0   Apple  25.0   50.0
1  Banana   NaN    NaN
2   Mango   NaN   42.0
3   Peach  22.0   62.0

After dropping missing values:
   Fruit  Quan  Price
0  Apple  25.0   50.0
3  Peach  22.0   62.0

After filling missing values with 0:
    Fruit  Quan  Price
0   Apple  25.0   50.0
1  Banana   0.0    0.0
2   Mango   0.0   42.0
3   Peach  22.0   62.0

After filling missing values with the mean:
    Fruit  Quan      Price
0   Apple  25.0  50.000000
1  Banana  23.5  51.333333
2   Mango  23.5  42.000000
3   Peach  22.0  62.000000

After filling missing values with the median:
    Fruit  Quan  Price
0   Apple  25.0   50.0
1  Banana  23.5   50.0
2   Mango  23.5   42.0
3   Peach  22.0   62.0
```

Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

Question 2: Write a Python Program to implement Linear Regression predicting house crisis based on the total rules in a dataset of Real Estate Buildings.

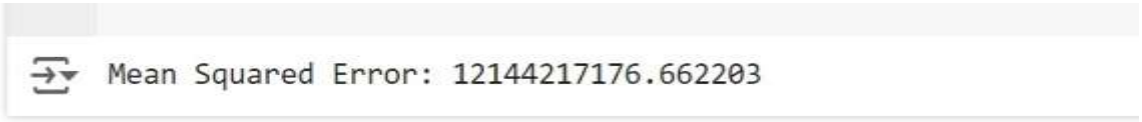
Example: California.csv determine mean square error for the model.

OBJECTIVE: Linear Regression is a popular Supervised Model, which helps to predict any future values based on the Historical State given.

CODE:

```
import pandas as pd

import numpy as np


from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_squared_error


data = pd.read_csv('/content/sample_data/california_housing_test.csv')

X = data[['total_rooms']]                # Feature

y = data['median_house_value']           # Target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


model = LinearRegression()

model.fit(X_train, y_train)


y_pred = model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
```

OUTPUT:



Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

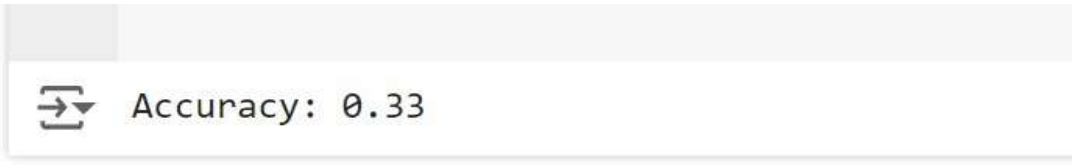
Question 3: Write a Python Program to mark E-mails from a given dataset as a Spam or Not Spam. (Logistics Regression).

OBJECTIVE: Logistic Regression helps to classify the data into different training samples and can also be used for Regression Purposes.

CODE:

```
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
emails = [
 "Hello, I wanted to check in with you about our meeting tomorrow.",
 "Congratulations! You've won a $1,000 gift card. Click here to claim.",
 "Important information about your account update.",
 "Can we reschedule our appointment for next week?",
 "You have been selected for a free vacation package!",
 "Here's the report you requested. Let me know if you need any changes.",
"Act now! This is your last chance to save big on our products.",
]
labels = [0, 1, 0, 1, 0, 1, 0, 1]
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(emails)
X_train, X_test, y_train, y_test = train_test_split(X, labels, test_size=0.3, random_state=42)

model = LogisticRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")
```

OUTPUT:



Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

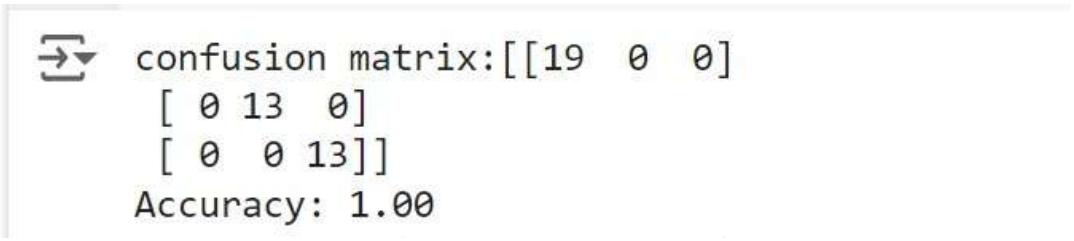
Question 4: Write a python program to implement a classification model on iris datasets into 3 species determine all the possible performance matrix such as accuracy score. (Logistics Regression).

OBJECTIVE: Models such as Logistic Regression, Decision Tree, and K-Nearest Neighbors are popular classifiers that can be used for classification purposes.

CODE:

```
import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import confusion_matrix

from sklearn.metrics import accuracy_score


data = pd.read_csv('/content/Iris.csv')


X = data.drop('Species',axis=1)

y = data['Species']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)


model = LogisticRegression()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)


cm = confusion_matrix(y_test, y_pred)

print(f'confusion matrix:{cm}')


accuracy = accuracy_score(y_test, y_pred)

print(f"Accuracy: {accuracy:.2f}")
```

OUTPUT:



Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

Question 5: Develop and implement a model to classify fruits based on color and size and determined accuracy score and confusion Matrix.

OBJECTIVE: K-Nearest Neighbors based on the number of neighbors can be implemented as an appropriate Classifier.

CODE:

```
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix

df=pd.read_csv('/content/fruits.csv')

X=df[[ 'color_score', 'mass', 'width',  'height' ]]

y=df ['fruit_name']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model=KNeighborsClassifier(n_neighbors=5)
model.fit(X_train,y_train)
y_pred=model.predict(X_test)

acc=accuracy_score(y_test,y_pred)

cf=confusion_matrix(y_test,y_pred)

print('Accuracy:',acc)
print('Confusion Matrix:')
print(cf)
```

OUTPUT:

```
Accuracy: 0.5833333333333334
Confusion Matrix:
[[1 1 0 1]
 [1 2 0 2]
 [0 0 2 0]
 [0 0 0 2]]
```

Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

Question 6: Write a python program to carry out clustering on one-dimensional data using KMeans.

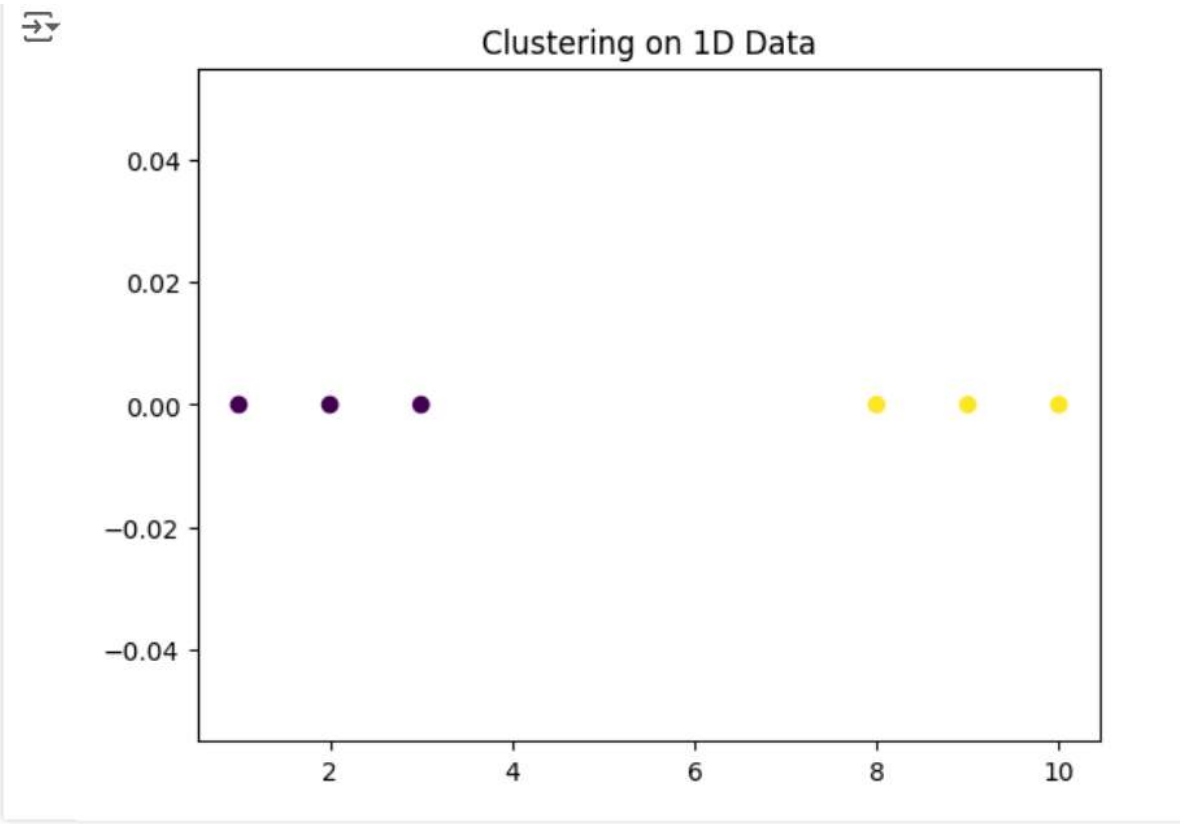
OBJECTIVE: K-means is a clustering algorithm that partitions data into k clusters based on minimizing the variance within each cluster.

CODE:

```
from sklearn.cluster import KMeans

import numpy as np
import matplotlib.pyplot as plt

data = np.array([[1], [2], [3], [8], [9], [10]])

kmeans = KMeans(n_clusters=2)
kmeans.fit(data)

labels = kmeans.predict(data)

plt.scatter(data, [0]*len(data), c=labels, cmap='viridis')

plt.title("Clustering on 1D Data")

plt.show()
```

Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

OUTPUT:



Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

Question 7: Write a python program to carry out clustering on two-dimensional data.

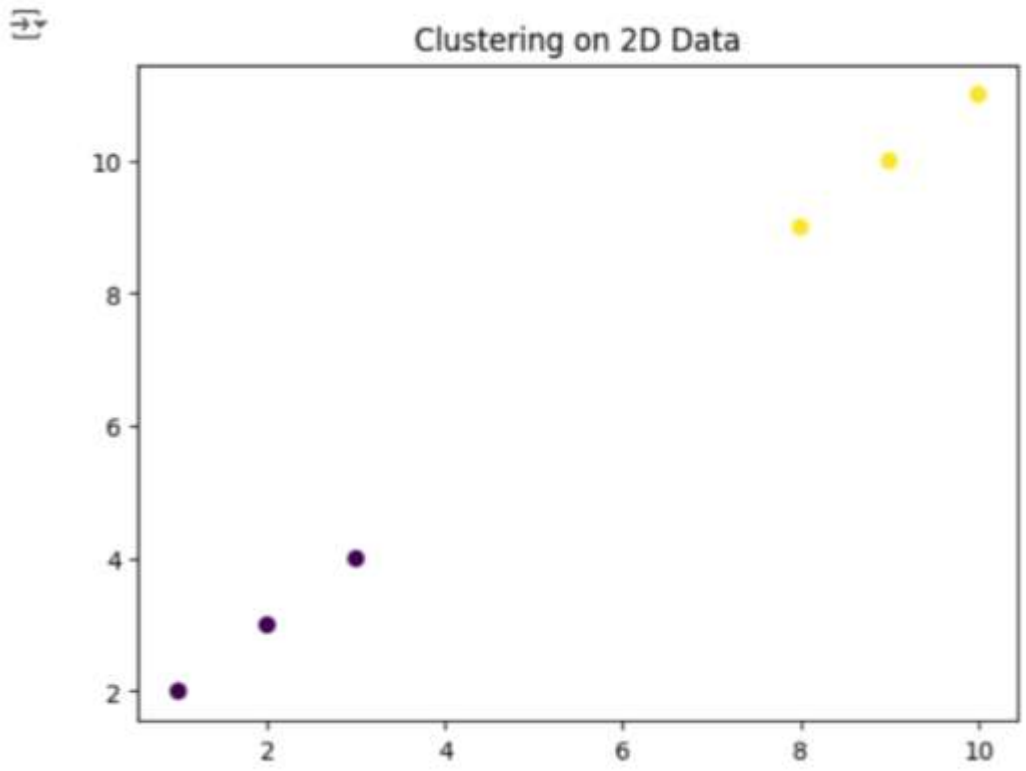
OBJECTIVE: K-means is a clustering algorithm that partitions data into k clusters based on minimizing the variance within each cluster.

CODE:

```
from sklearn.cluster import KMeans

import numpy as np
import matplotlib.pyplot as plt

data = np.array([[1, 2], [2, 3], [3, 4], [8, 9], [9, 10], [10, 11]])

kmeans = KMeans(n_clusters=2)
kmeans.fit(data)

labels = kmeans.predict(data)

plt.scatter(data[:, 0], data[:, 1], c=labels, cmap='viridis')

plt.title("Clustering on 2D Data")

plt.show()
```

Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

OUTPUT:



Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

Question 8: Write a python program to cluster random data.

CODE:

```
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs

import matplotlib.pyplot as plt

data, _ = make_blobs(n_samples=500, centers=3, random_state=42)

kmeans = KMeans(n_clusters=3)
kmeans.fit(data)

labels = kmeans.predict(data)

plt.scatter(data[:, 0], data[:, 1], c=labels, cmap='viridis')

plt.title("Clustering on Random Data")

plt.show()
```

Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

OUTPUT:

Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

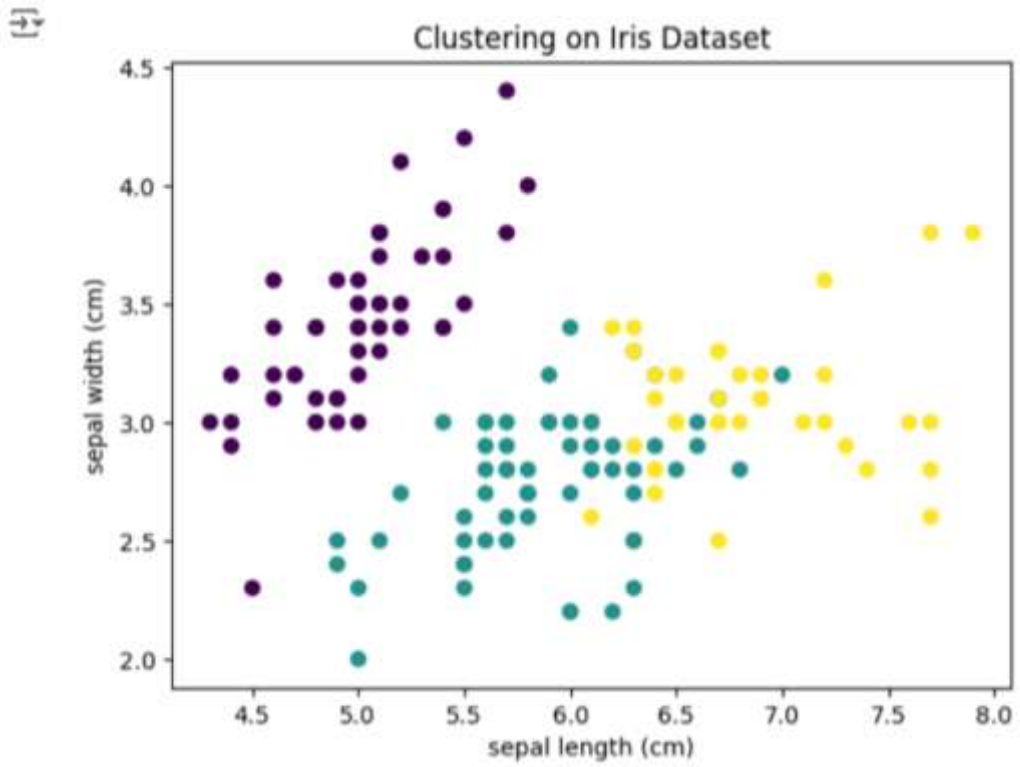
Question 9: Write a python program to implement KMeans on iris dataset.

CODE:

```
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris

import matplotlib.pyplot as plt
import pandas as pd

iris = load_iris()
data = iris.data

kmeans = KMeans(n_clusters=3)
kmeans.fit(data)

labels = kmeans.predict(data)

plt.scatter(data[:, 0], data[:, 1], c=labels, cmap='viridis')

plt.title("Clustering on Iris Dataset")

plt.xlabel(iris.feature_names[0])
plt.ylabel(iris.feature_names[1])

plt.show()
```

Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

OUTPUT:



Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

Question 10: Write a python program to implement Elbow Method on random dataset.

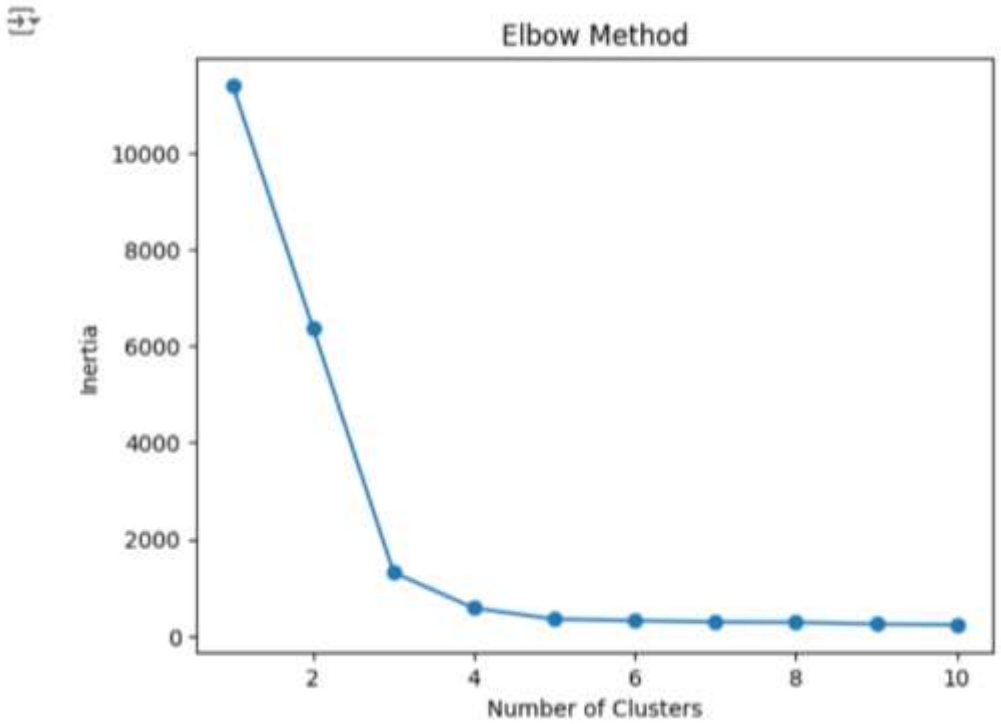
OBJECTIVE: The elbow method identifies the optimal number of clusters in a dataset by plotting the explained variance against the number of clusters and finding the point where the rate of decrease sharply changes, resembling an "elbow."

CODE:

```
from sklearn.cluster import KMeans

from sklearn.datasets import make_blobs


import matplotlib.pyplot as plt


data, _ = make_blobs(n_samples=200, centers=5, random_state=42)


inertia = []

for k in range(1, 11):

    kmeans = KMeans(n_clusters=k)

    kmeans.fit(data)

    inertia.append(kmeans.inertia_)


plt.plot(range(1, 11), inertia, marker='o')


plt.title("Elbow Method")


plt.xlabel("Number of Clusters")

plt.ylabel("Inertia")


plt.show()
```

Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

OUTPUT:



Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

Question 11: Write a Python Program to implement KMeans, Hierarchical Clustering and DBScan on a random dataset, where the dataset has 3 circles of data points.

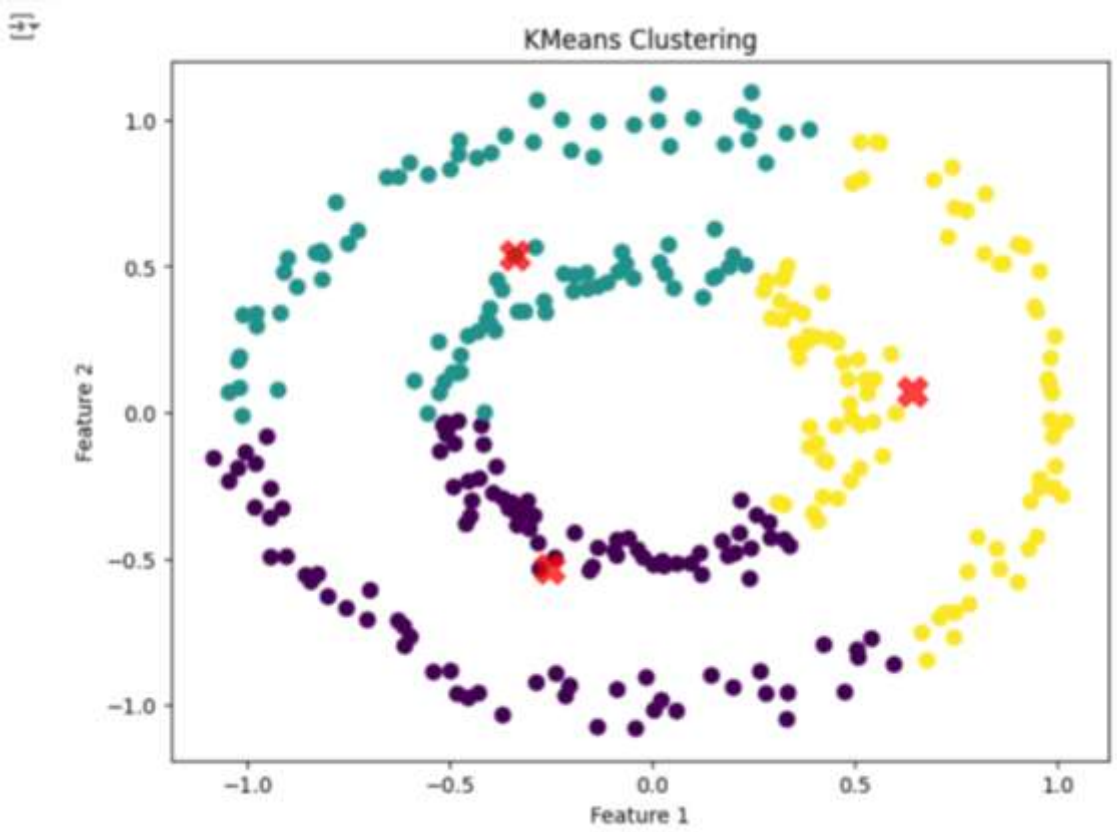
CODE:

1. K-Means:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_circles
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score

X, _ = make_circles(n_samples=300, noise=0.05, factor=0.5)

kmeans = KMeans(n_clusters=3)
kmeans.fit(X)
y_kmeans = kmeans.predict(X)

plt.figure(figsize=(8, 6))
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=0.75, marker='X')
plt.title('KMeans Clustering')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.show()

silhouette_kmeans = silhouette_score(X, y_kmeans)
print(f'Silhouette Score for KMeans: {silhouette_kmeans}')
```

Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

OUTPUT:




```
Silhouette Score for KMeans: 0.3802863325871913
```

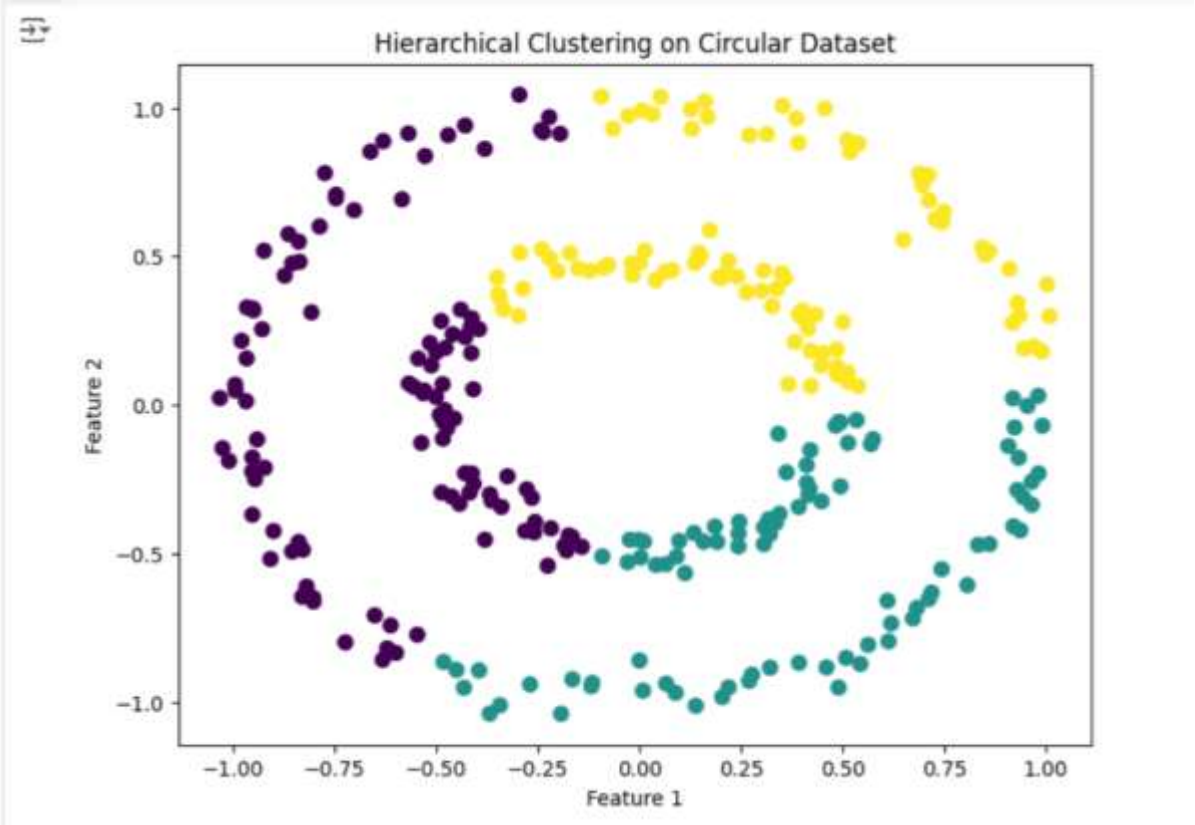
2. Hierarchical:

CODE:

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

from sklearn.datasets import make_circles
from sklearn.cluster import AgglomerativeClustering

X, _ = make_circles(n_samples=300, noise=0.05, factor=0.5)

hc = AgglomerativeClustering(n_clusters=3)
y_hc = hc.fit_predict(X)

plt.figure(figsize=(8, 6))
plt.scatter(X[:, 0], X[:, 1], c=y_hc, s=50, cmap='viridis')

plt.title('Hierarchical Clustering on Circular Dataset')

plt.xlabel('Feature 1')
plt.ylabel('Feature 2')

plt.show()
```

Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

OUTPUT:



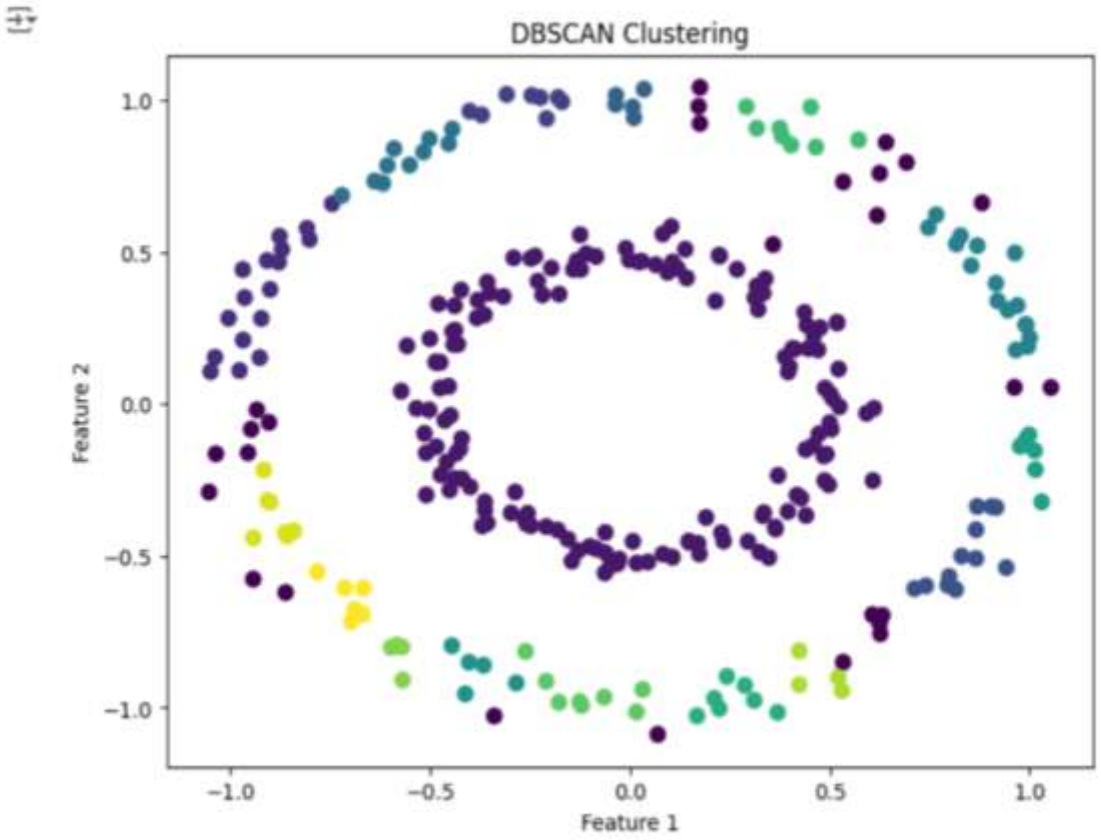
3. DBSCAN:

CODE:

```
from sklearn.cluster import DBSCAN
from sklearn.preprocessing import StandardScaler

X_scaled = StandardScaler().fit_transform(X)

dbscan = DBSCAN(eps=0.2, min_samples=5)
y_dbscan = dbscan.fit_predict(X_scaled)

plt.figure(figsize=(8, 6))
plt.scatter(X[:, 0], X[:, 1], c=y_dbscan, s=50, cmap='viridis')

plt.title('DBSCAN Clustering')

plt.xlabel('Feature 1')
plt.ylabel('Feature 2')

plt.show()

n_clusters_dbscan = len(set(y_dbscan)) - (1 if -1 in y_dbscan else 0)

print(f'Estimated number of clusters for DBSCAN: {n_clusters_dbscan}')
```

Name: Chhaya Bhandari **Section: A (MCA)** **Roll Number: 20(2301123)**

OUTPUT:




```
Estimated number of clusters for DBSCAN: 16
```