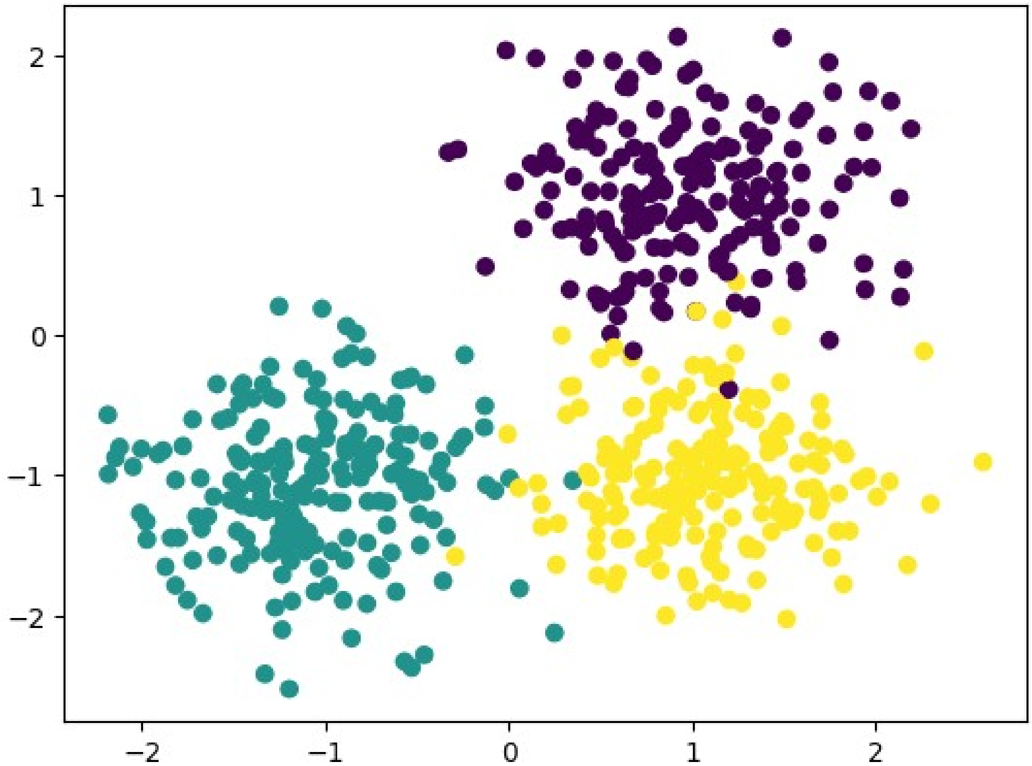
Problem 9

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

import seaborn as sns
import scipy.cluster.hierarchy as shc
from sklearn import metrics
from sklearn.datasets import make_blobs, make_circles, make_moons
from sklearn.cluster import DBSCAN
from sklearn.preprocessing import StandardScaler

seed = 0
```

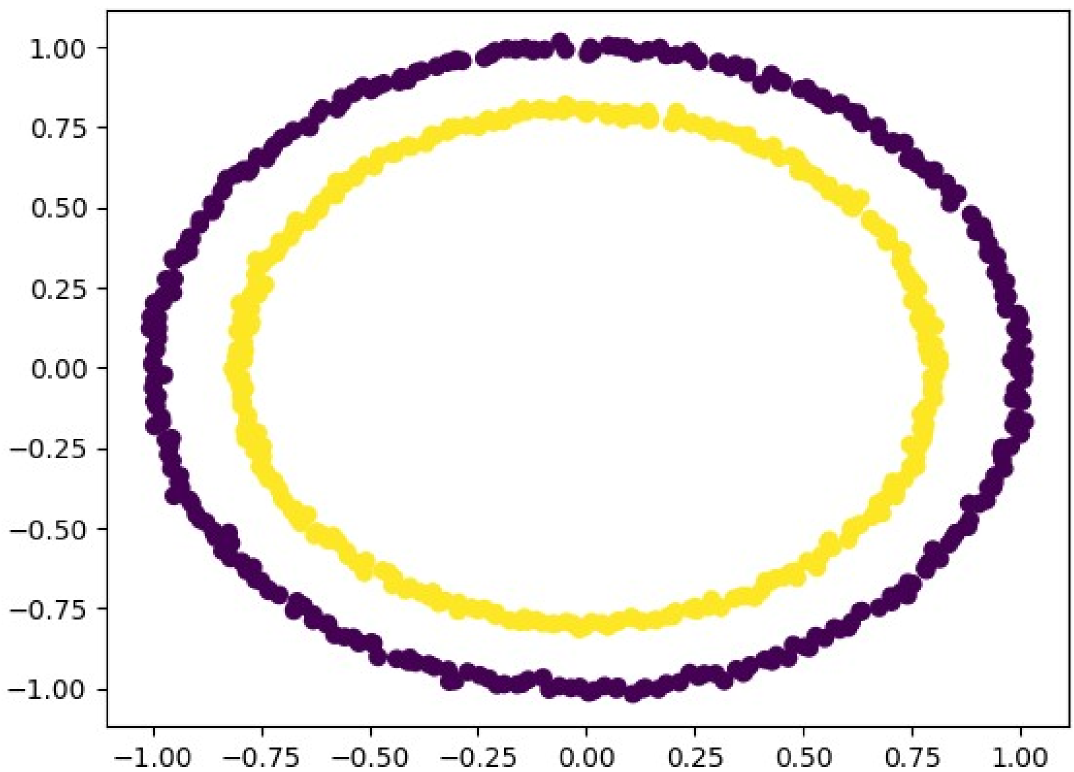
Generate a dummy toy dataset with varying densities and shapes. Set the eps (Epsilon) andmin samples (MinPts) parameters, and then fit DBSCAN to the generated dataset.

```
# blob data

centers = [[1, 1], [-1, -1], [1, -1]]
blob_X, blob_true_labels = make_blobs(n_samples = 600, centers =
centers, cluster_std= 0.5, random_state =seed)
scaled_blob_X = StandardScaler().fit_transform(blob_X)
plt.scatter(blob_X[:, 0], blob_X[:, 1], c = blob_true_labels)
```

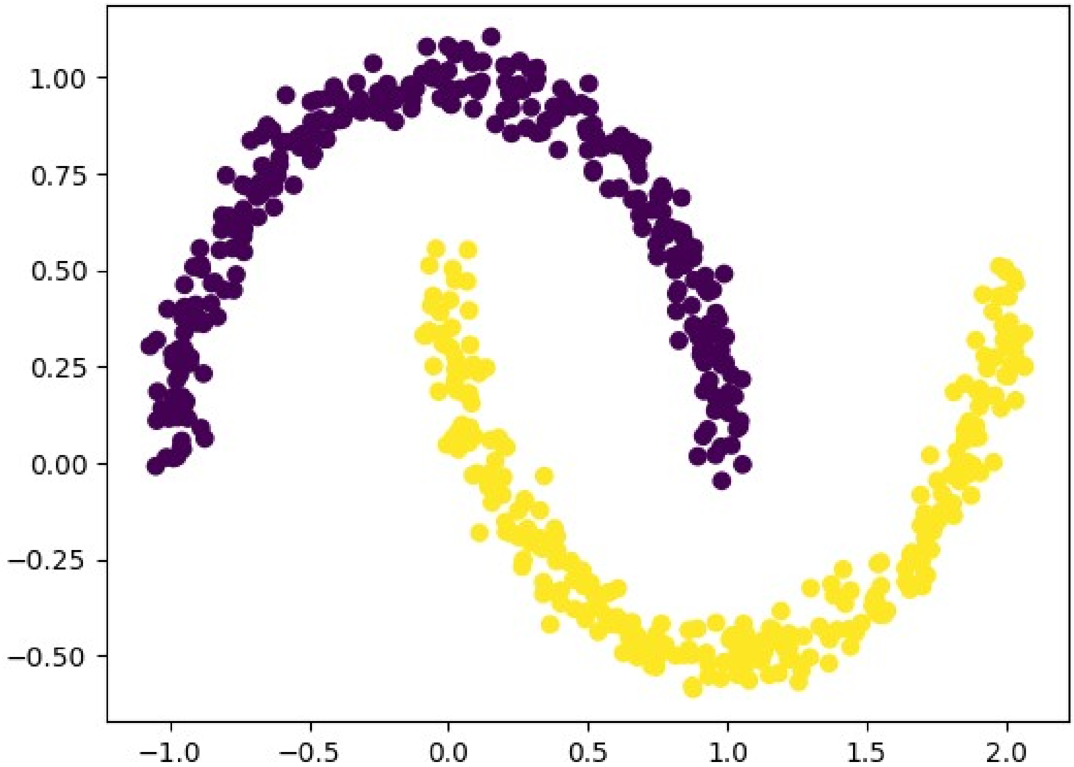
```
<matplotlib.collections.PathCollection at 0x7936dc1ae2f0>
```




```
# Concentric circles data

circle_X, circle_true_labels = make_circles(n_samples = 600, noise =
.01, random_state = seed)
scaled_circle_X = StandardScaler().fit_transform(circle_X)
plt.scatter(circle_X[:, 0], circle_X[:, 1], c = circle_true_labels)

<matplotlib.collections.PathCollection at 0x7936dbbf5600>
```




```
# Moon shaped data

moon_X, moon_true_labels = make_moons(n_samples = 600, noise = 0.05,
random_state = seed)
scaled_moon_X = StandardScaler().fit_transform(moon_X)
plt.scatter(moon_X[:, 0], moon_X[:, 1], c = moon_true_labels)

<matplotlib.collections.PathCollection at 0x7936dbc755a0>
```




```
# Helper function to plot

def plot_clusters(data, true_labels = None, cluster_labels = None,
title_true = 'True Cluster', title_cluster = 'DBSCAN clustering'):
  fig, (ax1, ax2) = plt.subplots(1, 2, figsize = (14, 6))
  ax1.scatter(data[:, 0], data[:, 1], c = true_labels)
  ax1.set_title(title_true)

  if cluster_labels is not None:
    ax2.scatter(data[:, 0], data[:, 1], c = cluster_labels)
    ax2.set_title(title_cluster)

  plt.show()
```

Objectives

1. Experiment with each combination of eps and min samples (consider at least 3 values of each) for these parameters. Report the values of the performance metrics to evaluate DBSCAN's sensitivity to parameter choices.

2. Visualize the clustering results using a scatter plot, where each cluster is assigned a different color. Additionally, use a different marker shape for noise points.

3. Calculate and report the following performance metrics: Silhouette Score, Adjusted Rand Index, Adjusted Mutual Information.

For Blob data

```
eps_values_blob = [0.2, 0.3, 0.5, 0.8, 1.3]
min_samples_values_blob = [2, 3, 5, 8, 13]

for i, eps in enumerate(eps_values_blob):
  for j, min_samples in enumerate(min_samples_values_blob):
    # Fit DBSCAN with current parameter combination
    dbscan_blob = DBSCAN(eps=eps, min_samples=min_samples)
    dbscan_blob.fit_predict(scaled_blob_X)

    # Calculate performance metrics
    X, pred_labels, true_labels = scaled_blob_X, dbscan_blob.labels_,
blob_true_labels
    if len(set(pred_labels)) > 1:
      silhouette = metrics.silhouette_score(X, pred_labels)
      ari = metrics.adjusted_rand_score(true_labels, pred_labels)
      ami = metrics.adjusted_mutual_info_score(true_labels,
pred_labels)
      print(f'For eps = {eps} and minimum samples = {min_samples}')
      print("Silhouette score: %0.3f " % silhouette)
      print('Adjusted Rand Index: %0.3f'% ari)
      print('Adjusted mutual information: %0.3f' % ami)
      plot_clusters(scaled_blob_X, true_labels, pred_labels)
      print('*********************************************')
      print('\n')

Output hidden; open in https://colab.research.google.com to view.
```

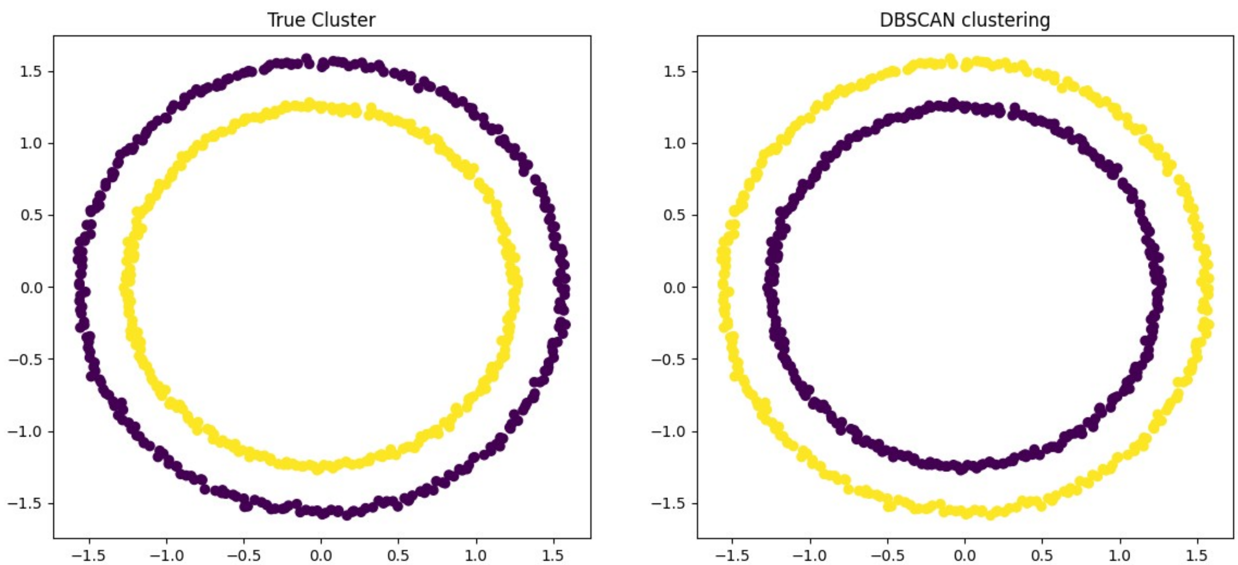
Based on the scoring for different eps and min_samples we found that the eps = 0.3 and min_samples = 13 is the best combination for blob dataset.

For Circle data

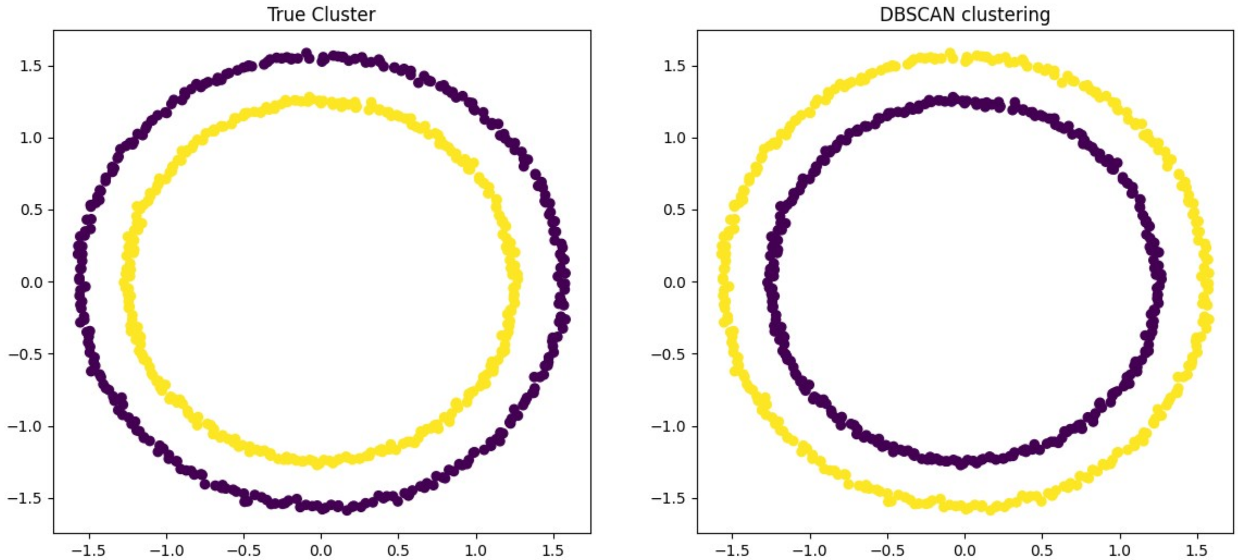
```
eps_values_circle = [0.2, 0.3, 0.5, 0.7]
min_samples_values_circle = [5, 10, 15, 20, 25]

for i, eps in enumerate(eps_values_circle):
  for j, min_samples in enumerate(min_samples_values_circle):
    # Fit DBSCAN with current parameter combination
    dbscan_circle = DBSCAN(eps=eps, min_samples=min_samples)
    dbscan_circle.fit_predict(scaled_circle_X)

    # Calculate performance metrics
    X, pred_labels, true_labels = scaled_circle_X,
dbscan_circle.labels_, circle_true_labels
    if len(set(pred_labels)) > 1:
      silhouette = metrics.silhouette_score(X, pred_labels)
      ari = metrics.adjusted_rand_score(true_labels, pred_labels)
      ami = metrics.adjusted_mutual_info_score(true_labels,
pred_labels)
```

```
        print(f'For eps = {eps} and minimum samples = {min_samples}')
        print("Silhouette score: %0.3f " % silhouette)
        print('Adjusted Rand Index: %0.3f'% ari)
        print('Adjusted mutual information: %0.3f' % ami)
        plot_clusters(X, true_labels, pred_labels)
        print('*******************************************')
        print('\n')
```

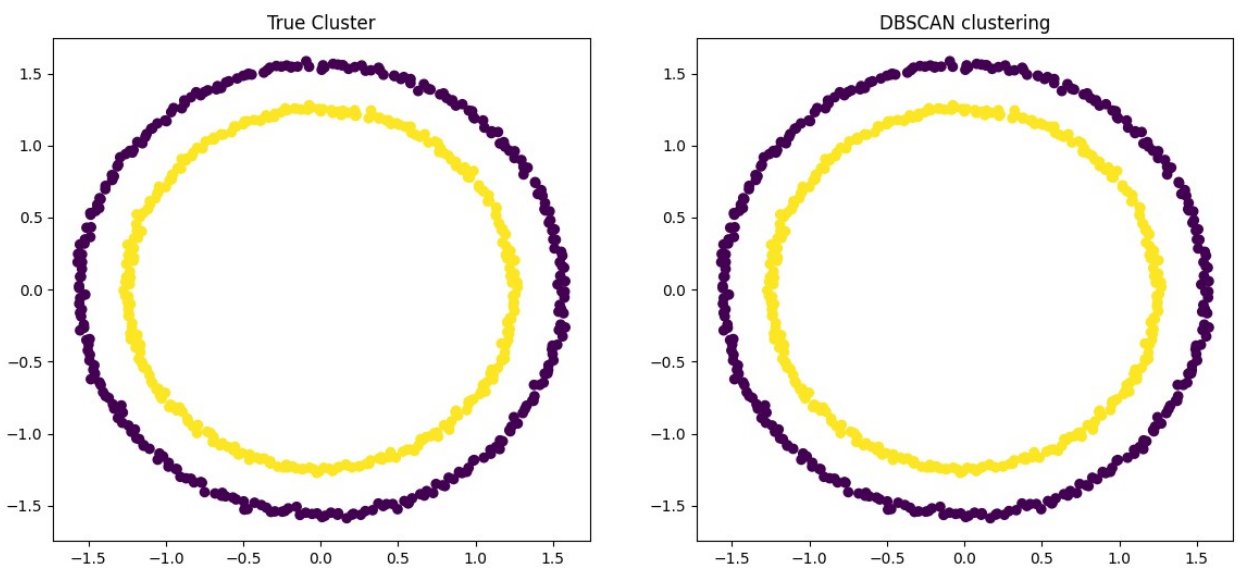
```
For eps = 0.2 and minimum samples = 5
Silhouette score: 0.019
Adjusted Rand Index: 1.000
Adjusted mutual information: 1.000
```




```
*******************************************


For eps = 0.2 and minimum samples = 10
Silhouette score: 0.019
Adjusted Rand Index: 1.000
Adjusted mutual information: 1.000
```

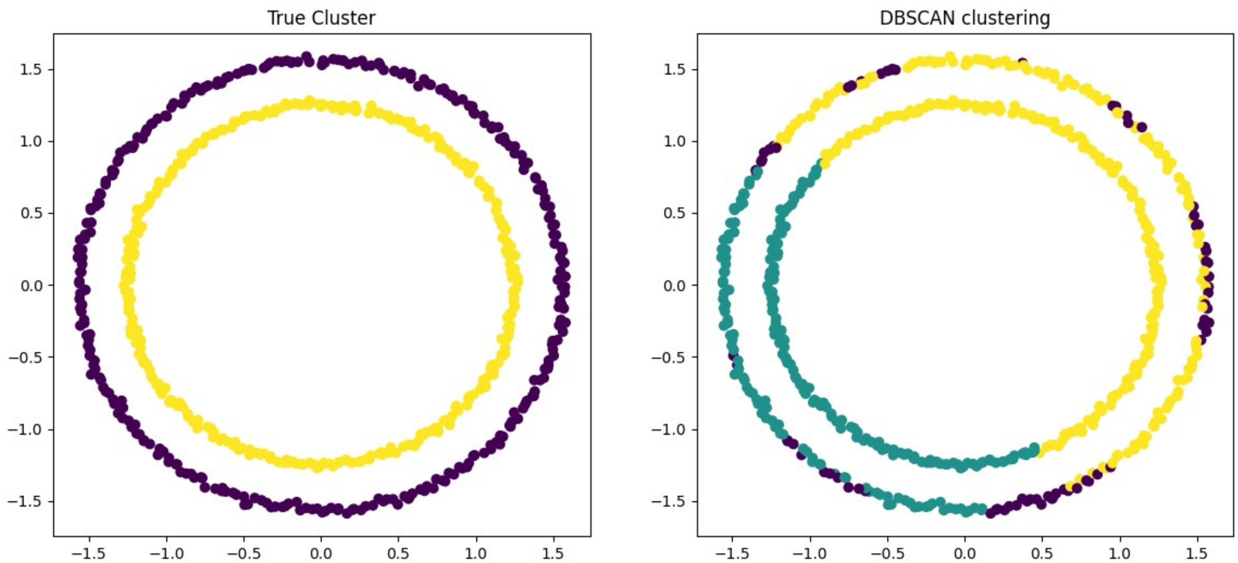



```
*********************************************


For eps = 0.2 and minimum samples = 15
Silhouette score: 0.019
Adjusted Rand Index: 1.000
Adjusted mutual information: 1.000
```




```
*********************************************


For eps = 0.3 and minimum samples = 25
Silhouette score: 0.230
```

```
Adjusted Rand Index: 0.043
Adjusted mutual information: 0.110
```




```
*********************************************
```

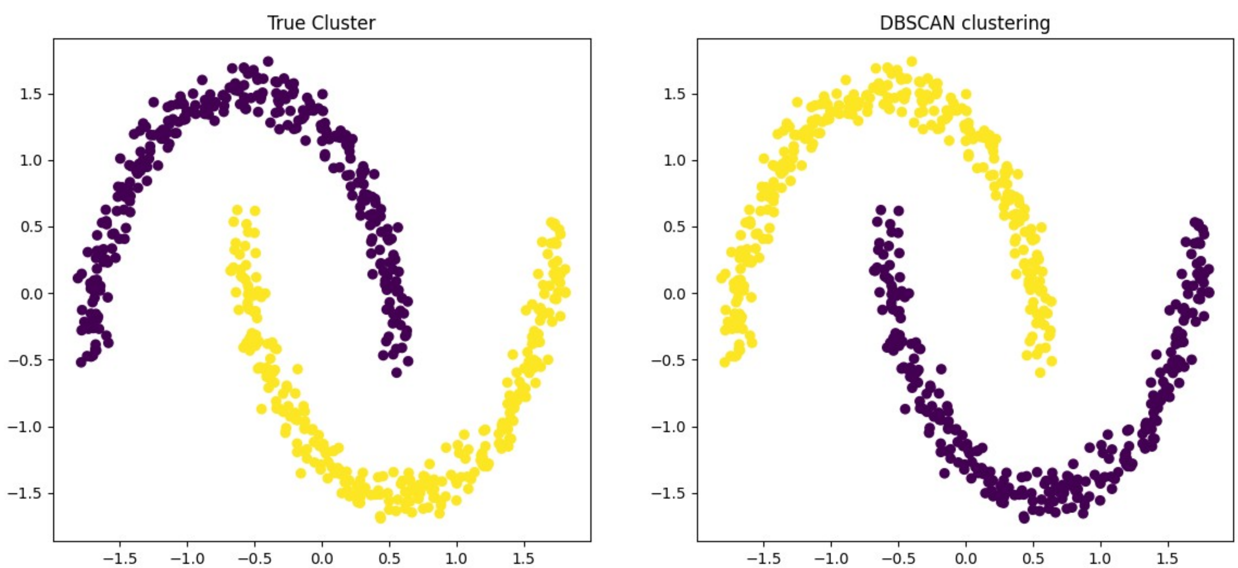
Observations

1. The scores are very bad for circular data.
2. Intrestingly, for eps = 0.2, any number of min_samples is generating a ari and ami score = 1, which implies that both the true cluster and predicted one agrees upon the placing of the points but the silhouette score is very bad.
3. We can conclude that DBSCAN clustering the data in proper number of clusters but it is not assigning correct datapoints to right cluster.

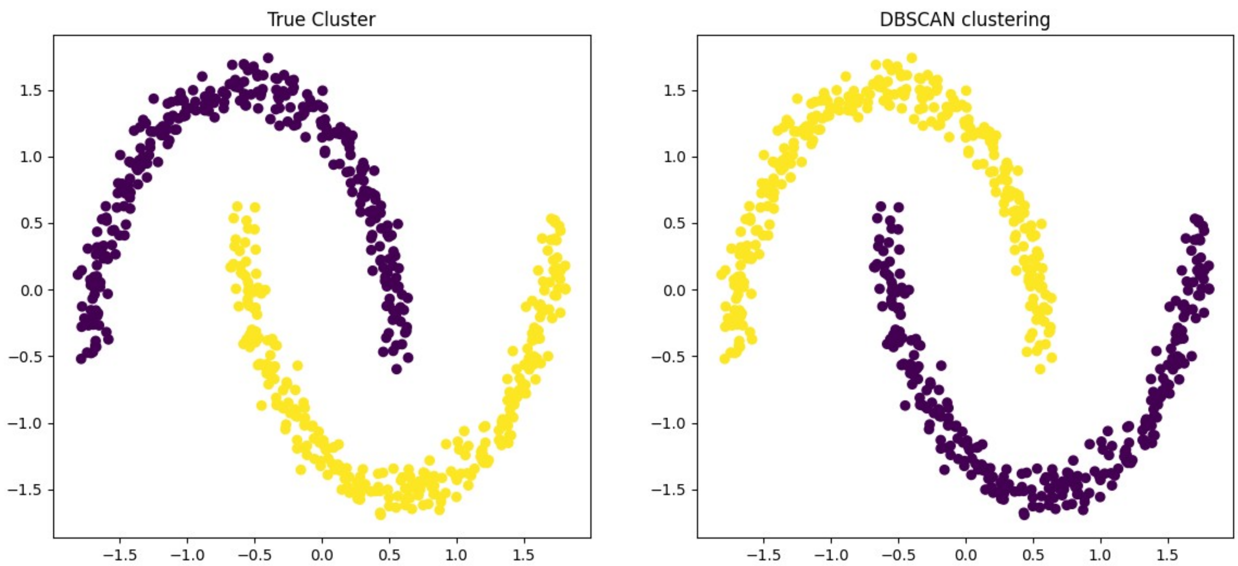
For moon data

```
eps_values_moon = [0.3, 0.5, 0.7]
min_samples_values_moon = [5, 10, 15, 20]

for i, eps in enumerate(eps_values_moon):
  for j, min_samples in enumerate(min_samples_values_moon):
    # Fit DBSCAN with current parameter combination
    dbscan_moon = DBSCAN(eps=eps, min_samples=min_samples)
    dbscan_moon.fit_predict(scaled_moon_X)

    # Calculate performance metrics
    X, pred_labels, true_labels = scaled_moon_X, dbscan_moon.labels_,
moon_true_labels
    if len(set(pred_labels)) > 1:
      silhouette = metrics.silhouette_score(X, pred_labels)
```

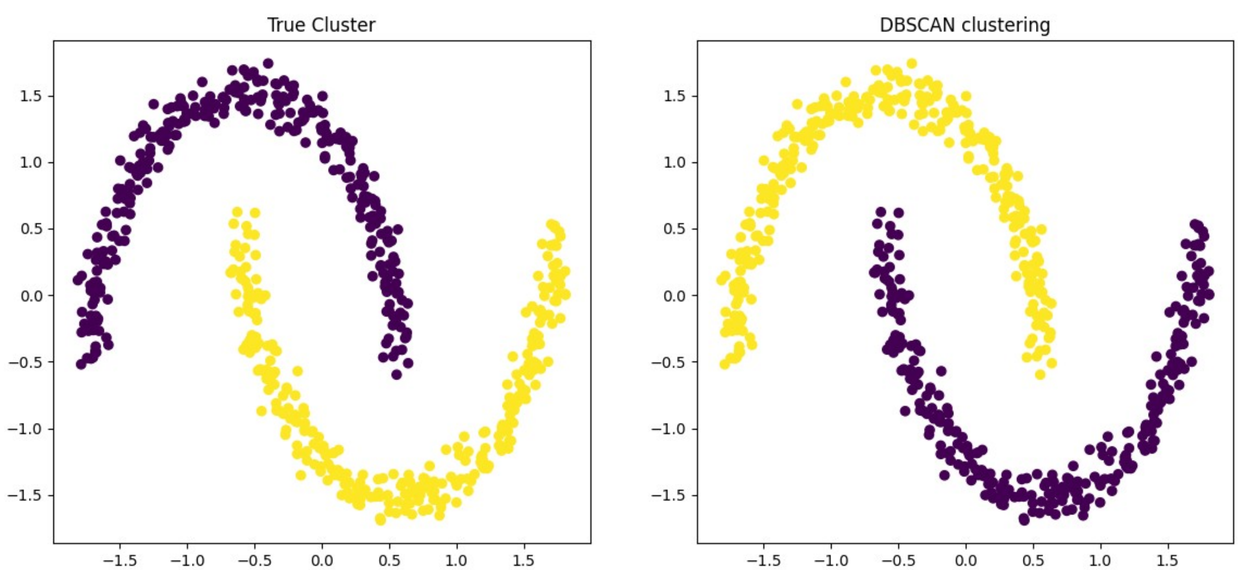
```
    ari = metrics.adjusted_rand_score(true_labels, pred_labels)
    ami = metrics.adjusted_mutual_info_score(true_labels,
pred_labels)
    print(f'For eps = {eps} and minimum samples = {min_samples}')
    print("Silhouette score: %0.3f " % silhouette)
    print('Adjusted Rand Index: %0.3f'% ari)
    print('Adjusted mutual information: %0.3f' % ami)
    plot_clusters(X, true_labels, pred_labels)
    print('*********************************************')
    print('\n')

For eps = 0.3 and minimum samples = 5
Silhouette score: 0.389
Adjusted Rand Index: 1.000
Adjusted mutual information: 1.000
```




```
*********************************************


For eps = 0.3 and minimum samples = 10
Silhouette score: 0.389
Adjusted Rand Index: 1.000
Adjusted mutual information: 1.000
```

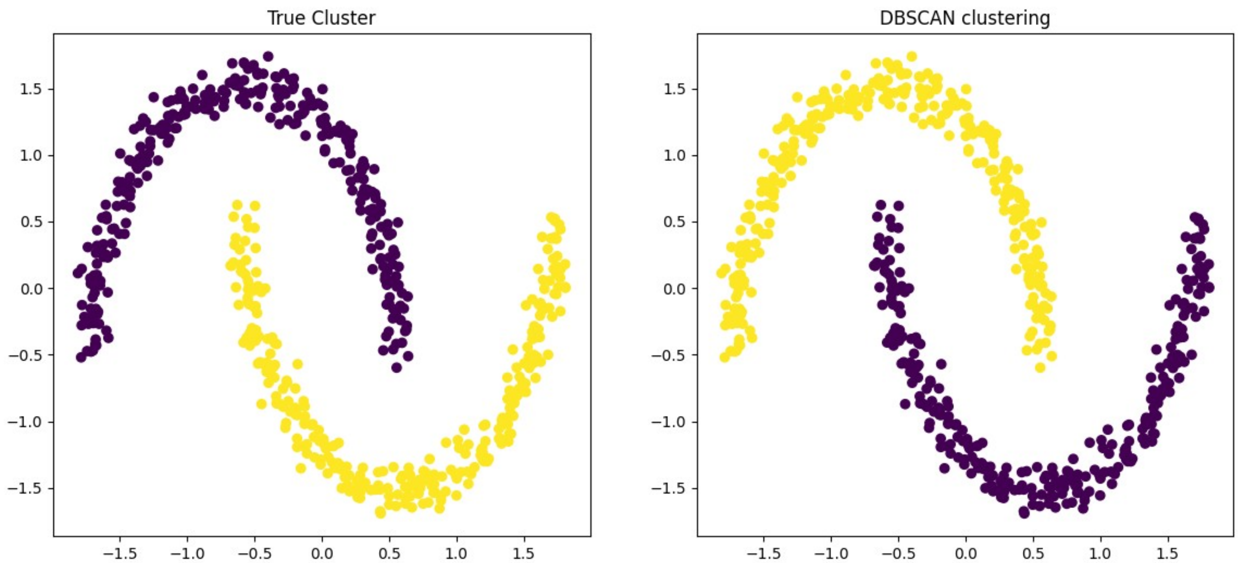



```
******************************************

For eps = 0.3 and minimum samples = 15
Silhouette score: 0.389
Adjusted Rand Index: 1.000
Adjusted mutual information: 1.000
```

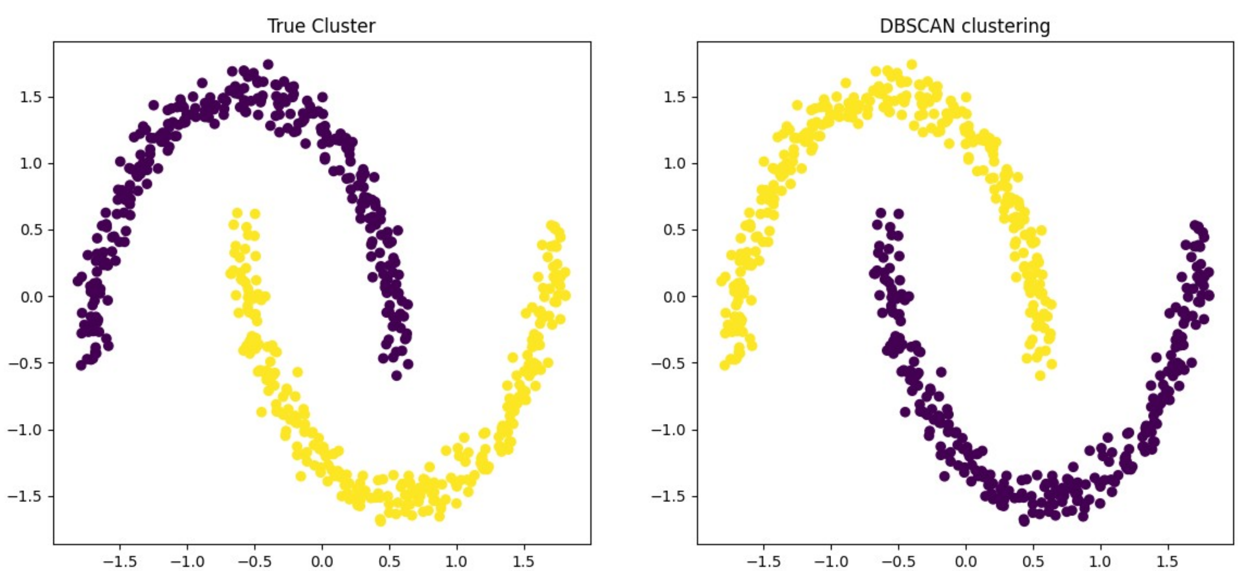



```
******************************************

For eps = 0.3 and minimum samples = 20
Silhouette score: 0.389
```

```
Adjusted Rand Index: 1.000
Adjusted mutual information: 1.000
```

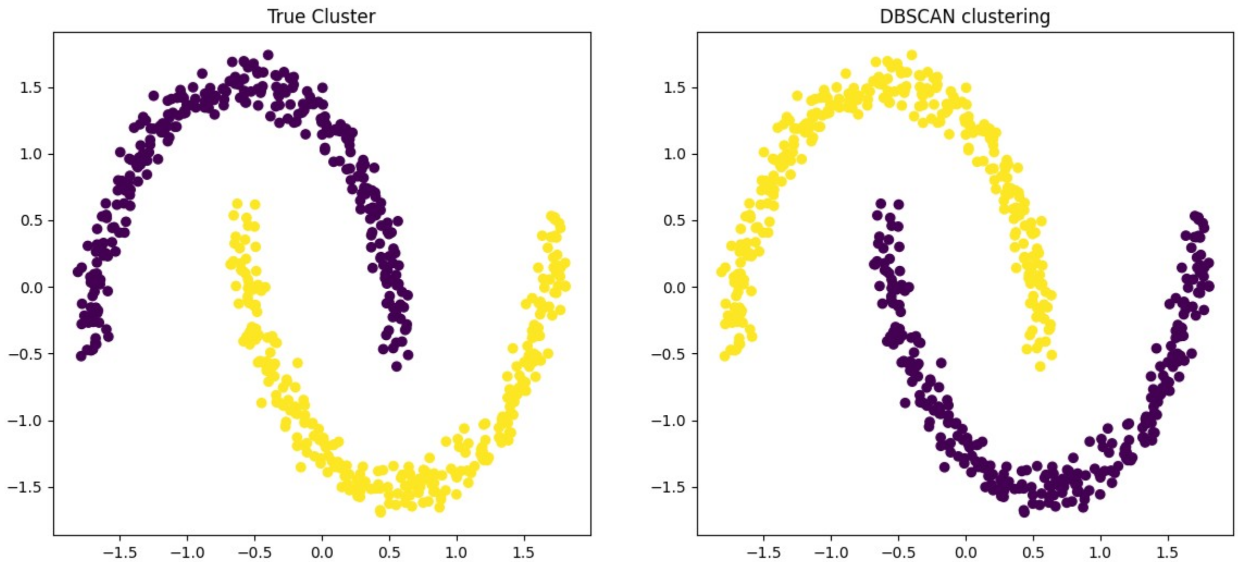



```
*********************************************


For eps = 0.5 and minimum samples = 5
Silhouette score: 0.389
Adjusted Rand Index: 1.000
Adjusted mutual information: 1.000
```

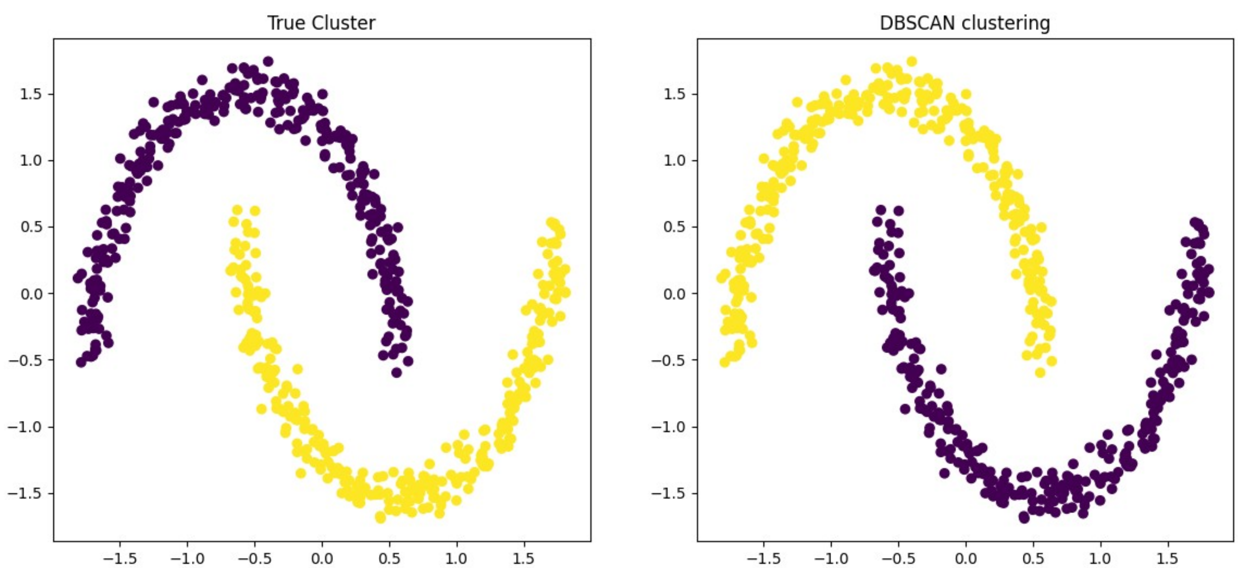



```
*********************************************
```

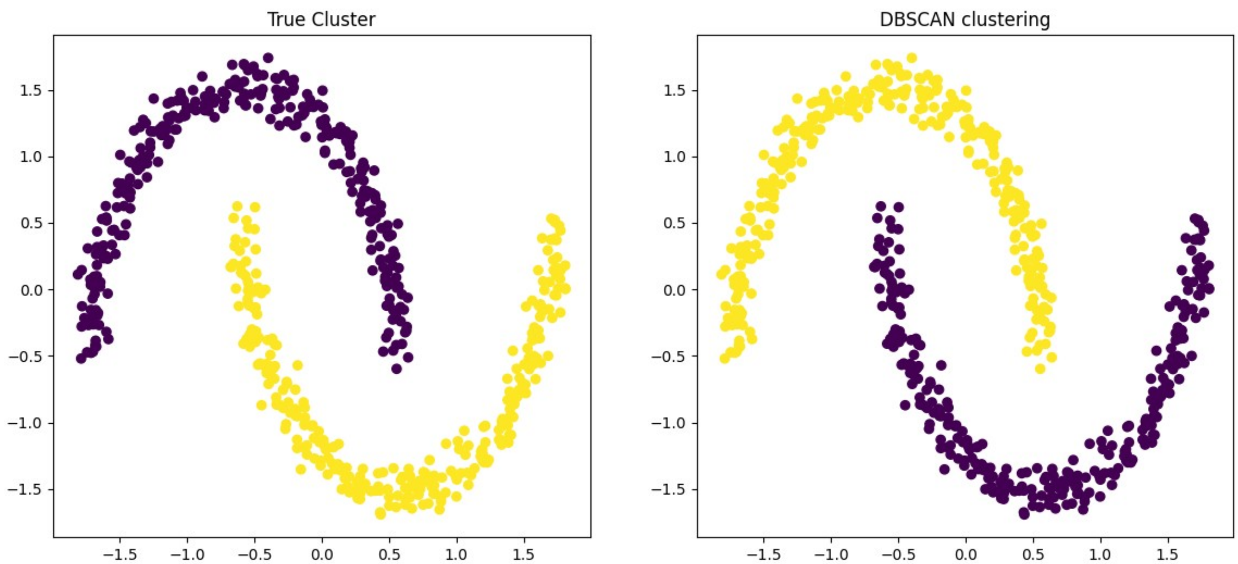
```
For eps = 0.5 and minimum samples = 10
Silhouette score: 0.389
Adjusted Rand Index: 1.000
Adjusted mutual information: 1.000
```




```
*********************************************

For eps = 0.5 and minimum samples = 15
Silhouette score: 0.389
Adjusted Rand Index: 1.000
Adjusted mutual information: 1.000
```



```
*********************************************


For eps = 0.5 and minimum samples = 20
Silhouette score: 0.389
Adjusted Rand Index: 1.000
Adjusted mutual information: 1.000
```




```
*********************************************
```

Observations

1. For moon dataset, as we can see the scores are consistent irrespective of the eps and min_samples values.
2. Again we are getting ari and ami = 1, which implies that DBSCAN is doing the placing of the points.
3. So we can take any eps and min_sample value for moon dataset.