CS564 : Foundations of Machine Learning
Assignment 2

By Priyanka Sachan (1901CS43)

Problem Statement

The assignment targets to implement DBScan algorithms to cluster the 3 datasets with blob, moon and circle structures. Additionally, compare it Kmeans algorithm in terms of cluster visualization and Silhouette index score.

Installation

Install the following dependencies either using pip or through conda in a Python 3.5+ environment:

```
python3 -m pip install pandas matplotlib
```

Running the program

Use the following command to run the program :

```
python3 cluster_algos.py
```

Implementation

Code added in zip file or check Notebook.

Import libraries and packages

```
# Import libraries and packages
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline

# scaling
from sklearn.preprocessing import StandardScaler

# KMeans clustering
from sklearn.cluster import KMeans
# DBSCAN clustering
from sklearn.cluster import DBSCAN

# silhouette score
from sklearn.metrics import silhouette_score
```

Import data

```
# Read csv file
blob_df = pd.read_csv(f'data/cluster_blobs.csv')
circle_df = pd.read_csv(f'data/cluster_circles.csv')
moon_df = pd.read_csv(f'data/cluster_moons.csv')

# Creating a dictionary for dataframes
df={'blob':blob_df,'circle':circle_df,'moon':moon_df}
```

Scaling data

Our dataset is not scaled some values are much bigger than others,if we do not scale our data our model will not going to perform well.So now we are going to scale our data using StandardScaler↗
It standardises features by removing the mean and scaling to unit variance.

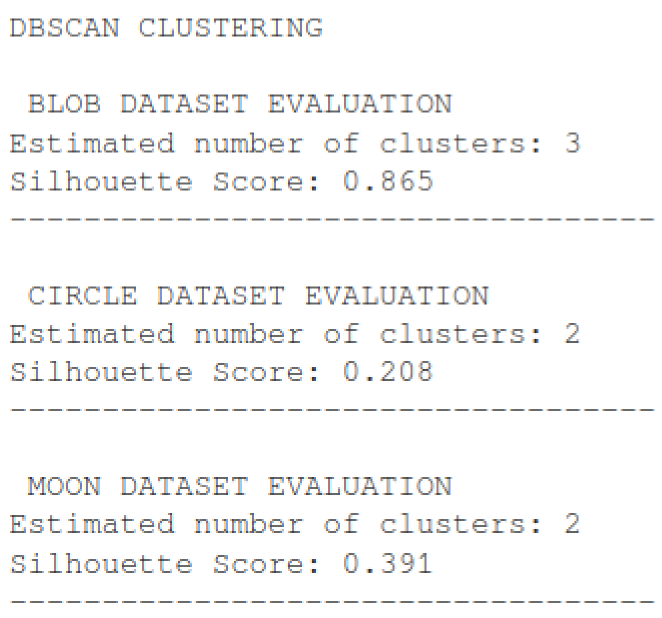
```
# Scaling data
scaled_df={}
for shape in ['blob','circle','moon']:
 print('\n',shape.upper(),'SCALED DATA')
 scaled=StandardScaler().fit_transform(df[shape])
 scaled_df[shape]=pd.DataFrame(scaled,columns=df[shape].columns)
 print(scaled_df[shape])
 print('-------------------------')
```

DBSCAN algorithm

Fit data using DBSCAN algorithm

```python
dbscan_labels={}
dbscan_clusters={}
print('DBSCAN CLUSTERING')

for shape in ['blob','circle','moon']:
 # Training datset on dbscan model
 dbscan = DBSCAN(eps=0.3, min_samples=10).fit(scaled_df[shape])

 # Get predicted labels
 dbscan_labels[shape] = dbscan.labels_
 # Number of clusters in labels
 dbscan_clusters[shape]=len(set(dbscan_labels[shape])) - (1 if -1 in
dbscan_labels[shape] else 0)

 print('\n',shape.upper(),'DATASET EVALUATION')
 print("Estimated number of clusters: %d" % dbscan_clusters[shape])
 print("Silhouette Score: %0.3f" % silhouette_score(scaled_df[shape],
dbscan_labels[shape]))
 print('-----------------------------------')
```

Evaluation

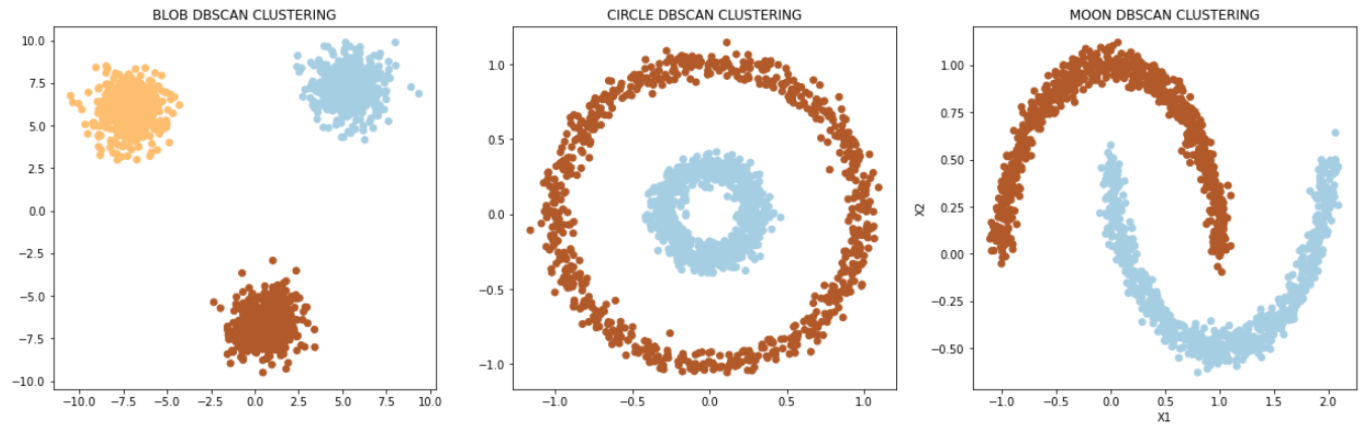


```
DBSCAN CLUSTERING

 BLOB DATASET EVALUATION
Estimated number of clusters: 3
Silhouette Score: 0.865
-----------------------------------

 CIRCLE DATASET EVALUATION
Estimated number of clusters: 2
Silhouette Score: 0.208
-----------------------------------

 MOON DATASET EVALUATION
Estimated number of clusters: 2
Silhouette Score: 0.391
-----------------------------------
```

Visualization

```
print('CLUSTERING VISUALIZATION')
fig, ax = plt.subplots(1, 3, figsize=(21,6))
id=0

for shape in ['blob','circle','moon']:
 ax[id].scatter(df[shape].iloc[:,0],df[shape].iloc[:,1],c=dbscan_labels[shape],
cmap='Paired')
 ax[id].set_title(shape.upper()+' DBSCAN CLUSTERING')
 id+=1

plt.xlabel("X1") # X-axis label
plt.ylabel("X2") # Y-axis label
plt.show() # showing the plot
```

```
CLUSTERING VISUALIZATION
```

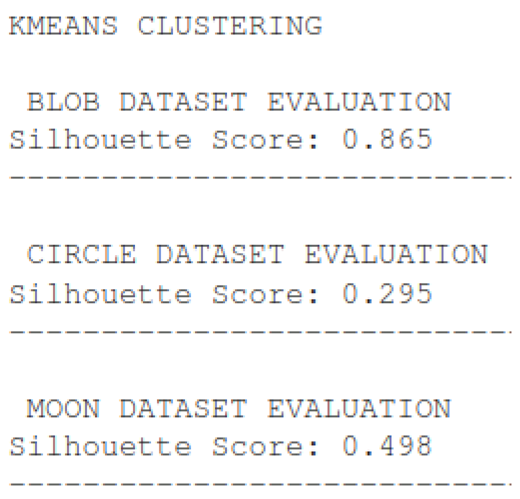


K-Means algorithm

Fit data using K-means algorithm

```
kmeans_labels={}
print('KMEANS CLUSTERING')

for shape in ['blob','circle','moon']:
 # Training datset on dbscan model
 kmeans = KMeans(n_clusters = dbscan_clusters[shape],random_state =
0).fit(scaled_df[shape])

 # Get predicted labels
 kmeans_labels[shape] = kmeans.labels_

 print('\n',shape.upper(),'DATASET EVALUATION')
 print("Silhouette Score: %0.3f" % silhouette_score(scaled_df[shape],
kmeans_labels[shape]))
 print('------------------------------------')
```

Evaluation

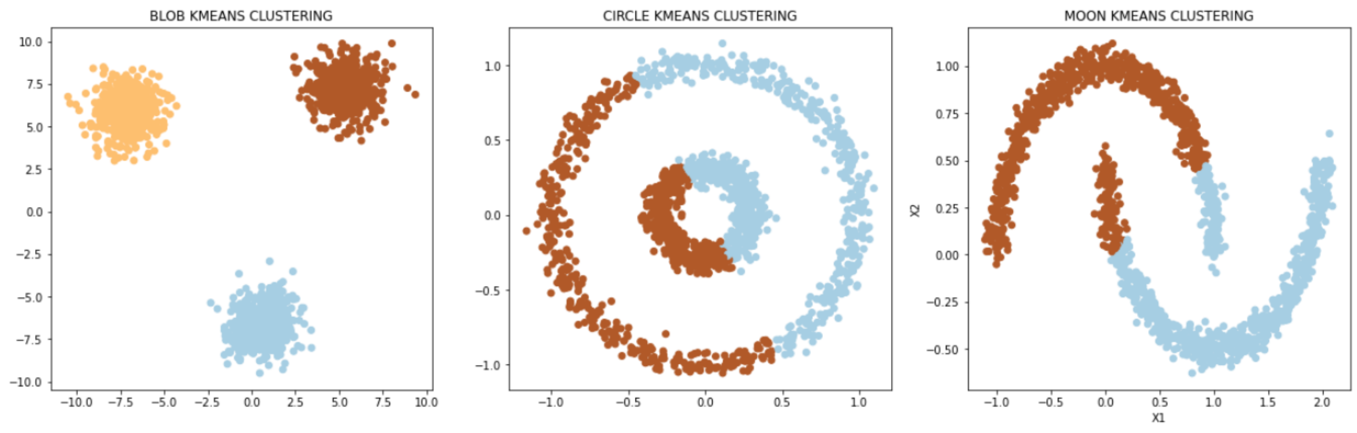


```
KMEANS CLUSTERING

 BLOB DATASET EVALUATION
Silhouette Score: 0.865
----------------------------

 CIRCLE DATASET EVALUATION
Silhouette Score: 0.295
----------------------------

 MOON DATASET EVALUATION
Silhouette Score: 0.498
----------------------------
```

Visualization

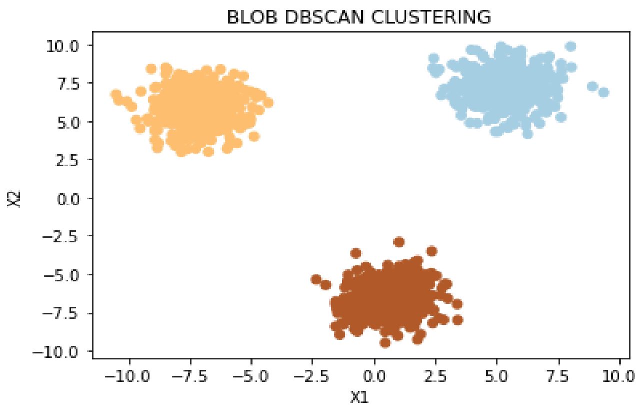
```
print('CLUSTERING VISUALIZATION')
fig, ax = plt.subplots(1, 3, figsize=(21,6))
id=0

for shape in ['blob','circle','moon']:
 ax[id].scatter(df[shape].iloc[:,0],df[shape].iloc[:,1],c=kmeans_labels[shape],
cmap='Paired')
 ax[id].set_title(shape.upper()+' KMEANS CLUSTERING')
 id+=1

plt.xlabel("X1") # X-axis label
plt.ylabel("X2") # Y-axis label
plt.show() # showing the plot
```

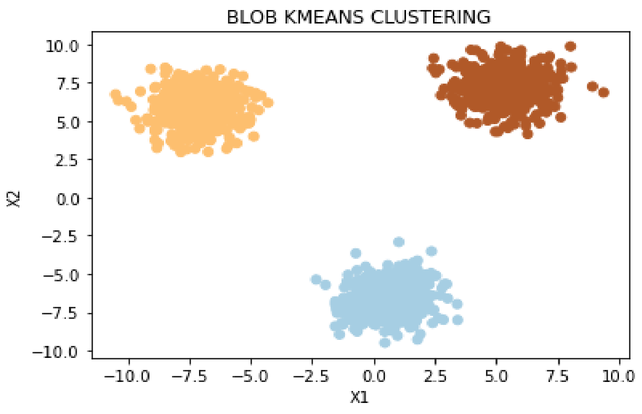
```
CLUSTERING VISUALIZATION
```



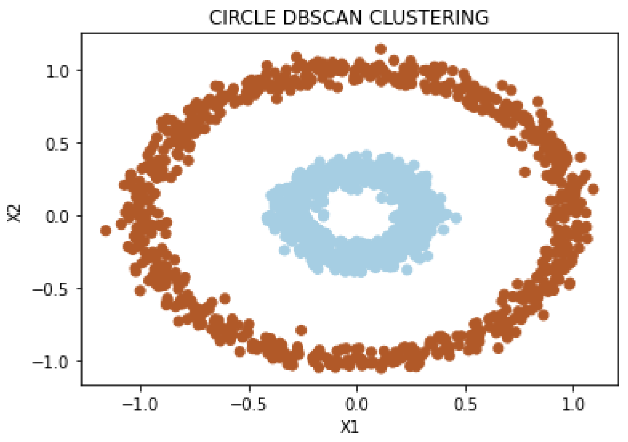
Comparison between DBSCAN and K-Means algorithm in terms of cluster visualization and Silhouette index score.

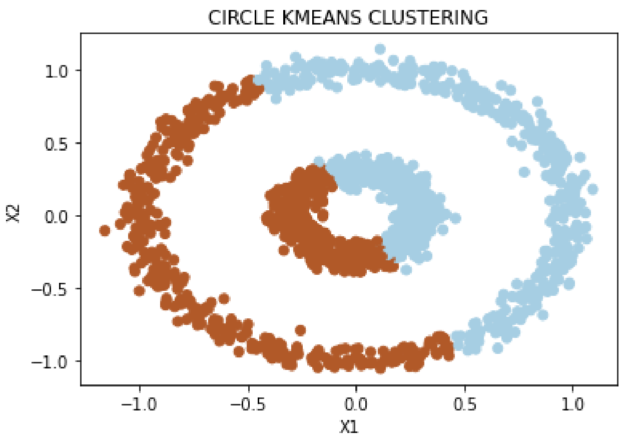
Quality of clustering measured by silhouette score approach determines how well each object lies within its cluster, high average silhouette width implies good clustering.



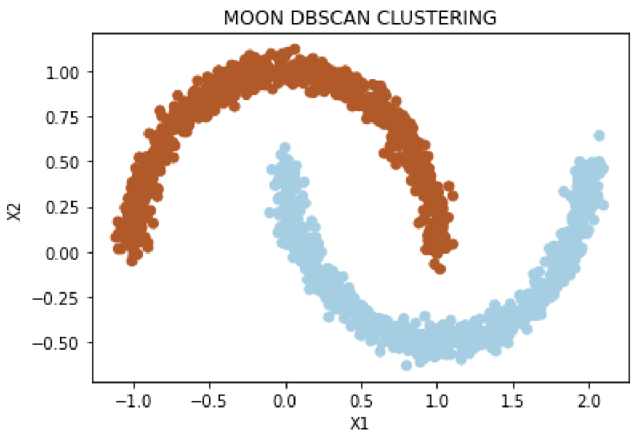


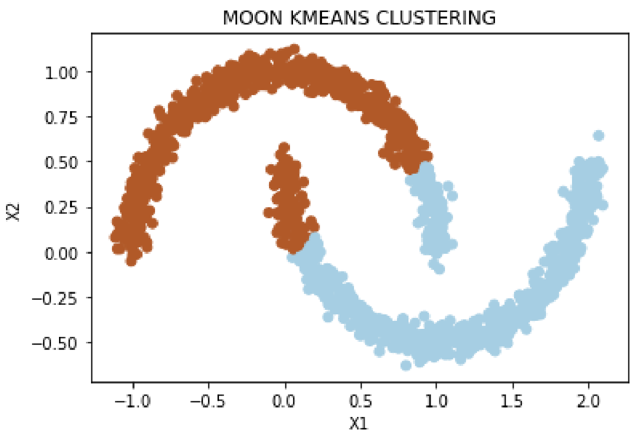
For blob shaped dataset, both DBSCAN and K-means algorithm has a 0.865 silhouette score and visualization too appears the same.





For circle shaped dataset, DBSCAN algorithm has a 0.208 silhouette score and K-means algorithm has a score of 0.295, however according to the plot, DBSCAN is better in clustering non-globular shapes .





For moon shaped dataset, DBSCAN algorithm has a 0.391 silhouette score and K-means algorithm has a score of 0.498, however according to the plot, DBSCAN is better.