ML ASSIGNMENT NO.-3

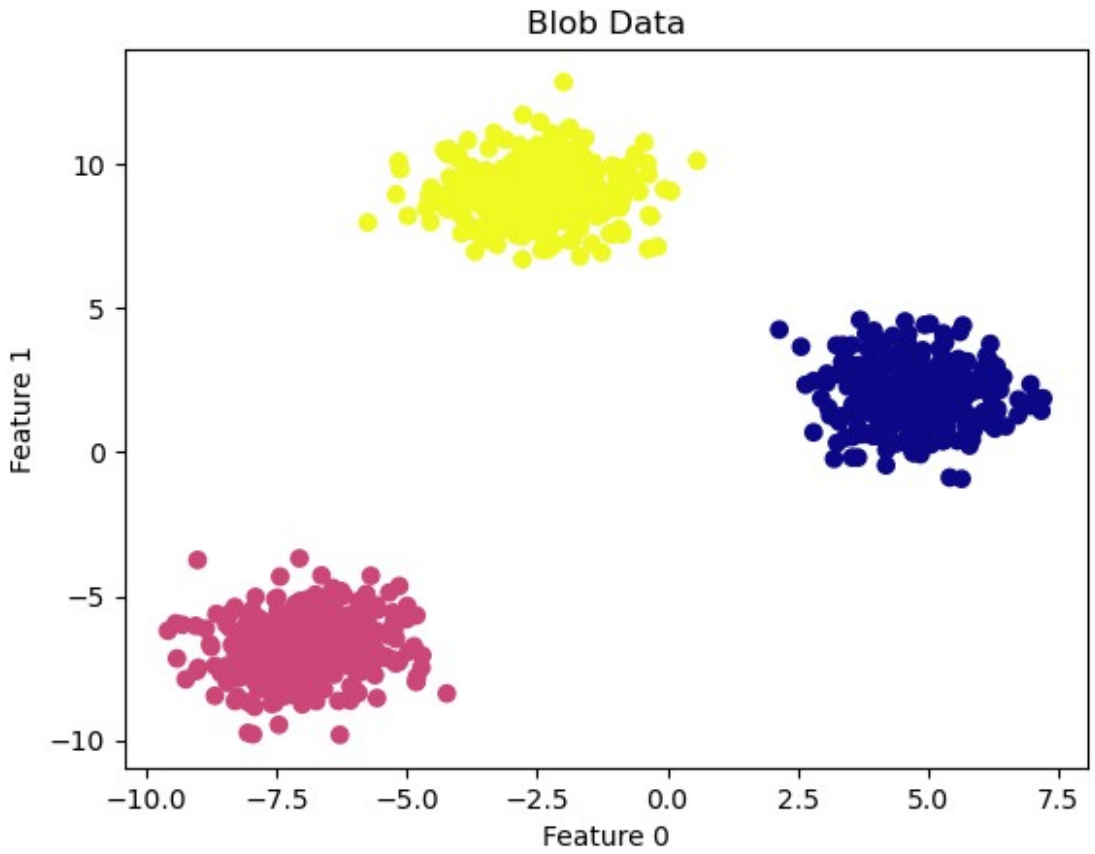
Question NO.1- Using Make_blob generate data of 1000 data points with three cluster apply kmeans on it with k = 3 and use the metrics and get the accuracy (For Accuracy take reference of DBSCAN evaluation) ● Apply DBscan on Cust Segmentation Data

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

X, _ = make_blobs(n_samples=1000, centers=3, random_state=42)

kmeans = KMeans(n_clusters=3, random_state=77)
y_pred = kmeans.fit_predict(X)
Accuracy = silhouette_score(X, y_pred)
print("Silhouette Score:", Accuracy)
plt.scatter(X[:,0],X[:,1],c = y_pred,cmap="plasma")
plt.xlabel("Feature 0")
plt.ylabel("Feature 1")
plt.title("Blob Data")
plt.show()
```

```
Silhouette Score: 0.8435705873891368
```




```
#DBSCAN On Cust Segmentation Data

import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import DBSCAN

cust = pd.read_csv('Cust_Segmentation.csv')
cust.head()
```

```
   Customer Id  Age  Edu  Years Employed  Income  Card Debt  Other
Debt  \
0            1   41    2               6      19      0.124
1.073
1            2   47    1              26     100      4.582
8.218
2            3   33    2              10      57      6.111
5.802
3            4   29    2               4      19      0.681
0.516
4            5   47    1              31     253      9.308
8.908

   Defaulted Address  DebtIncomeRatio
```

```
0        0.0  NBA001              6.3
1        0.0  NBA021             12.8
2        1.0  NBA013             20.9
3        0.0  NBA009              6.3
4        0.0  NBA008              7.2
```

```
cust.columns
```

```
Index(['Customer Id', 'Age', 'Edu', 'Years Employed', 'Income', 'Card
Debt',
       'Other Debt', 'Defaulted', 'Address', 'DebtIncomeRatio'],
      dtype='object')
```

```
cust['Card Debt']=cust['Card Debt'].astype(int)
cust['Other Debt']=cust['Other Debt'].astype(int)
object_columns = cust.select_dtypes(include = ['object']).columns
cust= cust.drop(object_columns, axis =1)
```

```
cust.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 850 entries, 0 to 849
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Customer Id      850 non-null    int64
 1   Age              850 non-null    int64
 2   Edu              850 non-null    int64
 3   Years Employed   850 non-null    int64
 4   Income           850 non-null    int64
 5   Card Debt        850 non-null    int32
 6   Other Debt       850 non-null    int32
 7   Defaulted        700 non-null    float64
 8   DebtIncomeRatio  850 non-null    float64
dtypes: float64(2), int32(2), int64(5)
memory usage: 53.2 KB
```

```
cust = cust[[ 'Age', 'Edu', 'Years Employed', 'Income', 'Card
Debt','Other Debt']]
cust.head()
```

```
   Age  Edu  Years Employed  Income  Card Debt  Other Debt
0   41    2               6      19          0           1
1   47    1              26     100          4           8
2   33    2              10      57          6           5
3   29    2               4      19          0           0
4   47    1              31     253          9           8
```

```
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import DBSCAN
from sklearn.metrics import silhouette_score
ss = StandardScaler()
```

```
data = ss.fit_transform(cust)
clust = DBSCAN(eps = 4, min_samples = 20)
clust.fit(data)
print("Accuracy for Cust Segmentation Data :",silhouette_score
(cust,clust.labels_))
```

```
Accuracy for Cust Segmentation Data : 0.778946043283898
```

```
#2-Using dirtydata.csv Demonstrate all the techniques for removing the
null values
#● Replace by MEAN
#● Replace by MEDIAN
#● Replace by MODE
#● Replace by ARBITUARY VALUE
#● Replace by 0
```

```
data = pd.read_csv("dirtydata.csv")
data
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	NaN
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	280.0

```
30        60  '2020/12/30'    102       129     380.3
31        60  '2020/12/31'     92       115     243.0
```

```
data.isnull().sum()
```

```
Duration    0
Date        1
Pulse       0
Maxpulse    0
Calories    2
dtype: int64
```

```
#by MEAN

print(data['Calories'].mean())
data['Calories'] = data['Calories'].fillna(data['Calories'].mean())
data
```

```
304.68
```

```
    Duration          Date  Pulse  Maxpulse  Calories
0         60  '2020/12/01'    110       130    409.10
1         60  '2020/12/02'    117       145    479.00
2         60  '2020/12/03'    103       135    340.00
3         45  '2020/12/04'    109       175    282.40
4         45  '2020/12/05'    117       148    406.00
5         60  '2020/12/06'    102       127    300.00
6         60  '2020/12/07'    110       136    374.00
7        450  '2020/12/08'    104       134    253.30
8         30  '2020/12/09'    109       133    195.10
9         60  '2020/12/10'     98       124    269.00
10        60  '2020/12/11'    103       147    329.30
11        60  '2020/12/12'    100       120    250.70
12        60  '2020/12/12'    100       120    250.70
13        60  '2020/12/13'    106       128    345.30
14        60  '2020/12/14'    104       132    379.30
15        60  '2020/12/15'     98       123    275.00
16        60  '2020/12/16'     98       120    215.20
17        60  '2020/12/17'    100       120    300.00
18        45  '2020/12/18'     90       112    304.68
19        60  '2020/12/19'    103       123    323.00
20        45  '2020/12/20'     97       125    243.00
21        60  '2020/12/21'    108       131    364.20
22        45           NaN    100       119    282.00
23        60  '2020/12/23'    130       101    300.00
24        45  '2020/12/24'    105       132    246.00
25        60  '2020/12/25'    102       126    334.50
26        60      20201226    100       120    250.00
27        60  '2020/12/27'     92       118    241.00
28        60  '2020/12/28'    103       132    304.68
29        60  '2020/12/29'    100       132    280.00
```

```
30        60  '2020/12/30'    102       129    380.30
31        60  '2020/12/31'     92       115    243.00
```

```
# by MEDIAN
```

```
data2 = pd.read_csv("dirtydata.csv")
print(data1['Calories'].median())
```

```
291.2
```

```
data2['Calories'] =
data2['Calories'].fillna(data2['Calories'].median())
data2
```

```
    Duration          Date  Pulse  Maxpulse  Calories
0         60  '2020/12/01'    110       130     409.1
1         60  '2020/12/02'    117       145     479.0
2         60  '2020/12/03'    103       135     340.0
3         45  '2020/12/04'    109       175     282.4
4         45  '2020/12/05'    117       148     406.0
5         60  '2020/12/06'    102       127     300.0
6         60  '2020/12/07'    110       136     374.0
7        450  '2020/12/08'    104       134     253.3
8         30  '2020/12/09'    109       133     195.1
9         60  '2020/12/10'     98       124     269.0
10        60  '2020/12/11'    103       147     329.3
11        60  '2020/12/12'    100       120     250.7
12        60  '2020/12/12'    100       120     250.7
13        60  '2020/12/13'    106       128     345.3
14        60  '2020/12/14'    104       132     379.3
15        60  '2020/12/15'     98       123     275.0
16        60  '2020/12/16'     98       120     215.2
17        60  '2020/12/17'    100       120     300.0
18        45  '2020/12/18'     90       112     291.2
19        60  '2020/12/19'    103       123     323.0
20        45  '2020/12/20'     97       125     243.0
21        60  '2020/12/21'    108       131     364.2
22        45           NaN    100       119     282.0
23        60  '2020/12/23'    130       101     300.0
24        45  '2020/12/24'    105       132     246.0
25        60  '2020/12/25'    102       126     334.5
26        60      20201226    100       120     250.0
27        60  '2020/12/27'     92       118     241.0
28        60  '2020/12/28'    103       132     291.2
29        60  '2020/12/29'    100       132     280.0
30        60  '2020/12/30'    102       129     380.3
31        60  '2020/12/31'     92       115     243.0
```

```
#by MODE
```

```
data2 = pd.read_csv("dirtydata.csv")
print(data1['Calories'].mode()[0])
```

```
300.0
```

```
print(data1['Calories'].mode()[0])
```

```
data1['Calories'] = data1['Calories'].fillna(data1['Calories'].mode()
[0])
```

```
data1
```

```
300.0
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	300.0
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	300.0
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

```
# by ARBITARY VALUE
```

```
data4 = pd.read_csv("dirtydata.csv")
```

```
arbitary_value =350
```

```
data4['Calories'] = data4['Calories'].fillna(arbitary_value)
data4
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	350.0
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	350.0
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

```
# by ZERO{0}

data5 = pd.read_csv("dirtydata.csv")
data5['Calories'] = data5['Calories'].fillna(0)
data5
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0

7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	0.0
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	0.0
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0