# **Machine Learning Assignment 5**

Name: Akhila Boddu

Student ID: 700742171

GitHub link: https://github.com/AkhilaBoddu/ML-Assignment-5.git

#### Video Link:

https://drive.google.com/file/d/1C2pMczVfhQ05On2QEVYYD4ZUTOfcB K/view?usp=share link

## **Clustering & Dimensionality reduction**

#### Question1

**Principal Component Analysis** 

- a. Apply PCA on CC dataset.
- b. Apply k-means algorithm on the PCA result and report your observation if the silhouette score has improved or not?
- c. Perform Scaling+PCA+K-Means and report performance.

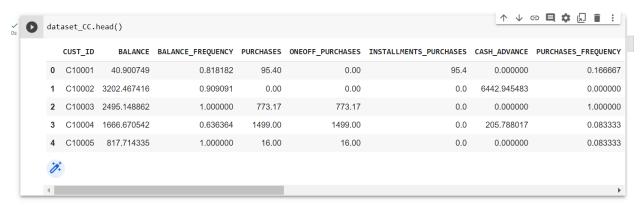
To do data analysis and apply machine learning algorithms on data, first I imported a few python libraries.

```
# importing required libraries for assignment 5 here
    import numpy as np
    import matplotlib.pyplot as plt
    import pandas as pd
    import seaborn as sns
    from sklearn import preprocessing, metrics
    from sklearn.preprocessing import StandardScaler, LabelEncoder
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
    from sklearn.decomposition import PCA
    from sklearn.cluster import KMeans
    sns.set(style="white", color_codes=True)
    import warnings
    warnings.filterwarnings("ignore")
                                                                                                       ↑ ↓ ⊖ ■ 🛊 🖟 🗎 ᠄
   # Principal Component Analysis
    # a. Apply PCA on CC dataset.
    # b. Apply k-means algorithm on the PCA result and report your observation if the silhouette score
    # has improved or not?
    # c. Perform Scaling+PCA+K-Means and report performance.
```

Using read\_csv method imported "CC GENERAL.csv" dataset.

```
↑ ↓ © 目 ‡ 見 i :
   dataset_CC = pd.read_csv('CC GENERAL.csv')
    dataset_CC.info()
<class 'pandas.core.frame.DataFrame'>
    RangeIndex: 8950 entries, 0 to 8949
   Data columns (total 18 columns):
                                          Non-Null Count Dtype
    0 CUST_ID
                                          8950 non-null
                                                          object
        BALANCE
                                          8950 non-null
                                                          float64
        BALANCE FREQUENCY
                                          8950 non-null
                                                          float64
        PURCHASES
                                          8950 non-null
                                                          float64
        ONEOFF_PURCHASES
                                          8950 non-null
                                                          float64
        INSTALLMENTS_PURCHASES
                                          8950 non-null
                                                          float64
        CASH_ADVANCE
                                          8950 non-null
                                                          float64
        PURCHASES_FREQUENCY
                                          8950 non-null
                                                          float64
        ONEOFF_PURCHASES_FREQUENCY
                                          8950 non-null
        PURCHASES_INSTALLMENTS_FREQUENCY
                                          8950 non-null
                                                          float64
     10
        CASH_ADVANCE_FREQUENCY
                                          8950 non-null
                                                          float64
     11
        CASH_ADVANCE_TRX
                                          8950 non-null
                                                          int64
        PURCHASES TRX
     12
                                          8950 non-null
                                                          int64
     13 CREDIT LIMIT
                                          8949 non-null
                                                          float64
        PAYMENTS
                                          8950 non-null
                                                          float64
     14
        MINIMUM_PAYMENTS
                                          8637 non-null
     15
                                                          float64
     16 PRC_FULL_PAYMENT
                                          8950 non-null
                                                          float64
                                          8950 non-null
    dtypes: float64(14), int64(3), object(1)
    memory usage: 1.2+ MB
```

The head() method of pandas library results top most rows of a data set.



Isnull() method of pandas library checks for any values present in dataset.

```
↑ ♥ ♥ ♥ ♥ № ■ :
dataset_CC.isnull().any()
       BALANCE
                                           False
       BALANCE_FREQUENCY
                                           False
       PURCHASES
                                           False
       ONEOFF PURCHASES
                                           False
       INSTALLMENTS PURCHASES
                                           False
       CASH_ADVANCE
                                           False
       PURCHASES_FREQUENCY
                                           False
       ONEOFF_PURCHASES_FREQUENCY
                                           False
       PURCHASES_INSTALLMENTS_FREQUENCY
                                           False
       CASH_ADVANCE_FREQUENCY
                                           False
       CASH ADVANCE TRX
                                           False
       PURCHASES TRX
                                           False
       CREDIT LIMIT
                                           True
       PAYMENTS
                                           False
       MINIMUM PAYMENTS
                                           True
       PRC_FULL_PAYMENT
                                           False
       TENURE
       dtype: bool
```

In this data set there are a few null values present in minimum payments and credit limit columns. So, these null values are replaced with their column mean value using fillna() method.

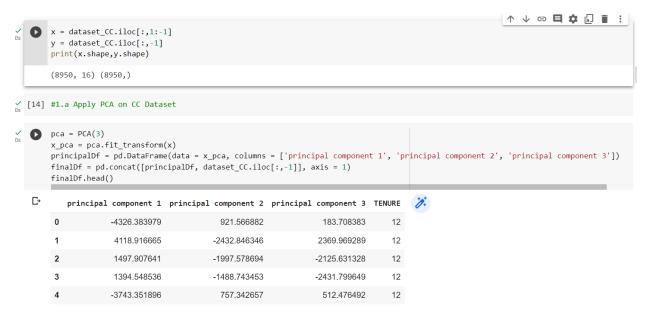
```
ΤΨΒΒΒ:

  [12] dataset_CC.fillna(dataset_CC.mean(), inplace=True)

       dataset CC.isnull().any()
       BALANCE
                                            False
       BALANCE_FREQUENCY
                                            False
       PURCHASES
       ONEOFF_PURCHASES
                                            False
       INSTALLMENTS_PURCHASES
                                            False
       CASH_ADVANCE
                                            False
       PURCHASES_FREQUENCY
                                            False
       ONEOFF PURCHASES FREQUENCY
                                            False
       PURCHASES_INSTALLMENTS_FREQUENCY
                                           False
       CASH_ADVANCE_FREQUENCY
CASH_ADVANCE_TRX
                                            False
                                            False
       PURCHASES_TRX
                                            False
       CREDIT_LIMIT
                                            False
       PAYMENTS
                                            False
       MINIMUM_PAYMENTS
       PRC_FULL_PAYMENT
                                            False
       TENURE
                                            False
       dtype: bool
```

Next we extract the input features and output labels from pandas dataframe and we print the shapes, here x is input features and y is output labels.

# a. Apply PCA on CC dataset



# b. Apply k-means algorithm on the PCA result and report your observation if the silhouette score has improved or not?

To perform k-means algorithm on a data set first we need to find the number of clusters value i.e., k value.

```
#1.b Apply K Means on PCA Result
X = finalDf.iloc[:,0:-1]
       y = finalDf.iloc[:,-1]
                                                                                                          ↑ ↓ © 目 ‡ 🖟 î :
   nclusters = 3 # this is the k in kmeans
       km = KMeans(n_clusters=nclusters)
       km.fit(X)
       # predict the cluster for each data point
       y_cluster_kmeans = km.predict(X)
       \ensuremath{\text{\#}} Summary of the predictions made by the classifier
       print(classification_report(y, y_cluster_kmeans, zero_division=1))
       print(confusion_matrix(y, y_cluster_kmeans))
       train_accuracy = accuracy_score(y, y_cluster_kmeans)
       print("\nAccuracy for our Training dataset with PCA:", train_accuracy)
       #Calculate sihouette Score
       score = metrics.silhouette_score(X, y_cluster_kmeans)
       print("Sihouette Score: ",score)
       Sihouette Score- ranges from -1 to +1 , a high value indicates that the object is well matched to its own cluster and poorly matc
                  precision recall f1-score support
  ₽
                       0.51 0.53 0.52
                 0
                                                     192
                      0.76 0.41 0.53
0.00 1.00 0.00
                 1
                                                     564
                                                        0
                                           0.44
                                                      756
                    0.42 0.65
0.70 0.44
                                            0.35
         macro avg
      weighted avg
                                           0.53
                                                      756
      [[102 74 16]
       [ 98 232 234]
       [ 0 0 0]]
      Accuracy for our Training dataset with PCA: 0.4417989417989418
      Sihouette Score: 0.30375402761352255
       '\nSihouette Score- ranges from -1 to +1 , a high value indicates that the object is well matched to its own cluster and poorly m
      atched to neighboring clusters.\n'
```

#### c. Perform Scaling+PCA+K-Means and report performance

```
#1.c Scaling +PCA + KMeans

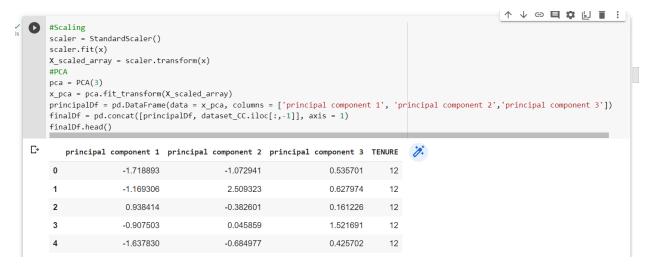
x = dataset_CC.iloc[:,1:-1]

y = dataset_CC.iloc[:,-1]

print(x.shape,y.shape)

[-> (8950, 16) (8950,)
```

### Here we perform Scaling and PCA



Here we extract the features and target variable from finalDf dataframe and X contains all columns of dataframe except last one, y contains values from the last column

```
y [69] X = finalDf.iloc[:,0:-1]
y = finalDf["TENURE"]
print(X.shape,y.shape)

(8950, 3) (8950,)
```

#### Here we perform k-means

```
↑ ↓ ⑤ 🛢 🛊 🎵 🔋
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.34,random_state=0)
 nclusters = 3
 # this is the k in kmeans
 km = KMeans(n_clusters=nclusters)
 km.fit(X_train,y_train)
 # predict the cluster for each training data point
 y_clus_train = km.predict(X_train)
 # Summary of the predictions made by the classifier
 print(classification_report(y_train, y_clus_train, zero_division=1))
 print(confusion_matrix(y_train, y_clus_train))
 train_accuracy = accuracy_score(y_train, y_clus_train)
 print("Accuracy for our Training dataset with PCA:", train_accuracy)
 #Calculate sihouette Score
 score = metrics.silhouette_score(X_train, y_clus_train)
 print("Sihouette Score: ",score)
 Sihouette Score- ranges from -1 to +1 , a high value indicates that the object is well matched to its own cluster and poorly matc
```

```
C→
                  precision
                               recall f1-score support
                       0.00
                                 1.00
                                            0.00
                                                       0.0
                                  1.00
                                                       0.0
                       0.00
                                            0.00
                       1.00
                                  0.00
                                            0.00
                                                      139.0
                                  0.00
                                            0.00
                        1.00
                                  0.00
                                            0.00
                                                      128.0
                                  0.00
                        1.00
                                            0.00
                                                      118.0
              10
                       1.00
                                  0.00
                                            0.00
                                                      151.0
              11
                       1.00
                                  0.00
                                            0.00
                                                      262.0
                                                     4974.0
        accuracy
                                                     5907.0
                       0.70
       macro avg
                                 0.30
                                            0.00
                                                     5907.0
    weighted avg
                       1.00
                                                     5907.0
        0
              0
                                   0
                                        0
                                                       0]
                                                       0]
       105
            30
                        a
                                                       0]
0]
       108
             26
                                                       0]
0]
        89
             27
                             0
                                        0
       107
       185
             66
                 11
                             0
                                   0
                                        0
                                                       01
     [3397 842 735
                                                       0]]
    Accuracy for our Training dataset with PCA: 0.0 Sihouette Score: 0.38140434191330214
     \nSihouette Score- ranges from -1 to +1 , a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring c
    lusters.\n'
```

```
# predict the cluster for each testing data point
y_clus_test = km.predict(X_test)

# Summary of the predictions made by the classifier
print(classification_report(y_test, y_clus_test, zero_division=1))
print(confusion_matrix(y_test, y_clus_test))

train_accuracy = accuracy_score(y_test, y_clus_test)
print("\nAccuracy for our Training dataset with PCA:", train_accuracy)

#Calculate sihouette Score
score = metrics.silhouette_score(X_test, y_clus_test)
print("Sihouette Score: ",score)

"""

Sihouette Score- ranges from -1 to +1 , a high value indicates that the object is well matched to its own clust"
"""
```

		pre	cision	r	recall	f1-sc	ore	support	↑ ↓ ፡□ 🗖 💠
		0	0.00		1.00	6	.00	0.0	
		1	0.00		1.00		.00	0.0	
		2	0.00		1.00		.00	0.0	
		6	1.00		0.00		.00	65.0	
		7	1.00		0.00		.00	55.0	
		8	1.00		0.00		.00	68.0	
		9	1.00		0.00		.00	57.0	
	1		1.00		0.00		.00	85.0	
	1		1.00		0.00		.00	103.0	
	1	2	1.00		0.00	6	.00	2610.0	
	accurac	.,					.00	3043.0	
	icro av	,	0.70		0.30		.00	3043.0	
weighted av			1.00		0.00		.00	3043.0	
		ь	2.00		0.00			501510	
]]	0 0	0	0	0	0	0	0	0 0]	
[	0 0	0	0	0	0	0	0	0 0]	
[	0 0	0	0	0	0	0	0	0 0]	
[ 4	1 21	3	0	0	0	0	0	0 0]	
[ 4	13 12	0	0	0	0	0	0	0 0]	
[ 5	7 10	1	0	0	0	0	0	0 0]	
[ :	5 22	0	0	0	0	0	0	0 0]	
[ 6	3 17	5	0	0	0	0	0	0 0]	
[ 6	9 30	4	0	0	0	0	0	0 0]	
[176	5 450	395	0	0	0	0	0	0 0]]	

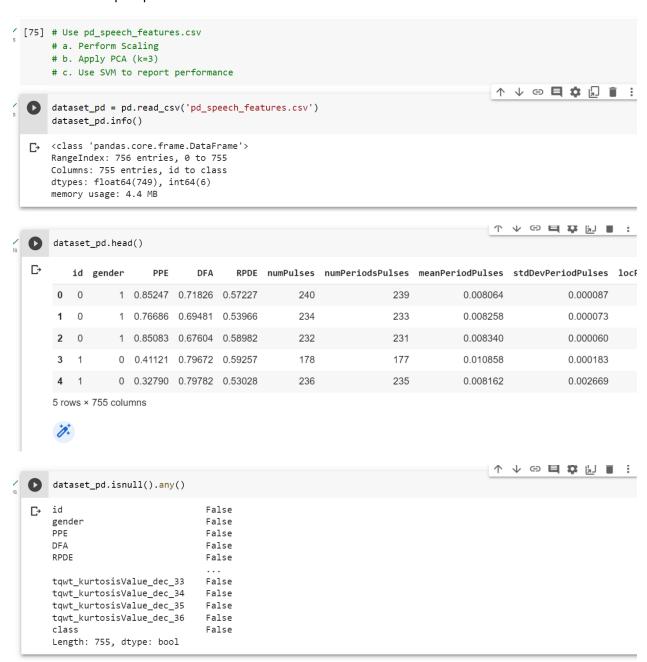
'\nSihouette Score- ranges from -1 to +1 , a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring c

lusters.\n'

#### Question2

Use pd\_speech\_features.csv

- a. Perform Scaling
- b. Apply PCA (k=3)
- c. Use SVM to report performance



# a. Perform Scaling

```
#Scaling Data
scaler = StandardScaler()
X_Scale = scaler.fit_transform(X)
X = dataset_pd.drop('class',axis=1).values
y = dataset_pd['class'].values
```

# b. Apply PCA (k=3)



```
X = finalDf.drop('class',axis=1).values
y = finalDf['class'].values
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.34,random_state=0)
```

### c. Use SVM to report performance

```
#2.c Support Vector Machine's

from sklearn.svm import SVC

svmClassifier = SVC()
svmClassifier.fit(X_train, y_train)

y_pred = svmClassifier.predict(X_test)

# Summary of the predictions made by the classifier
print(classification_report(y_test, y_pred, zero_division=1))
print(confusion_matrix(y_test, y_pred))
# Accuracy score
glass_acc_svc = accuracy_score(y_pred,y_test)
print('accuracy is',glass_acc_svc )

#Calculate sihouette Score
score = metrics.silhouette_score(X_test, y_pred)
print("Sihouette Score: ",score)
```

```
₽
             precision recall f1-score support
                0.67 0.42 0.51
                                         62
           0
                0.84 0.93 0.88
                                        196
     accuracy
                                0.81
                                         258
              0.75 0.68 0.70
                                         258
     macro avg
   weighted avg
                0.80 0.81 0.79
                                         258
   [[ 26 36]
   [ 13 183]]
   accuracy is 0.810077519379845
   Sihouette Score: 0.25044638820643456
```

#### Question3

# Apply Linear Discriminant Analysis (LDA) on Iris.csv dataset to reduce dimensionality of data tok=2.

```
T ▼ 면 택 및 된 : ]
   #3.Apply Linear Discriminant Analysis (LDA) on Iris.csv dataset to reduce dimensionality of data to k=2.
    from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
    dataset_iris = pd.read_csv('Iris.csv')
    dataset_iris.info()
class 'pandas.core.frame.DataFrame'>
    RangeIndex: 150 entries, 0 to 149
    Data columns (total 6 columns):
    # Column Non-Null Count Dtype
                      150 non-null int64
     1 SepalLengthCm 150 non-null float64
2 SepalWidthCm 150 non-null float64
     3 PetalLengthCm 150 non-null float64
    4 PetalWidthCm 150 non-null float64
5 Species 150 non-null object
    dtypes: float64(4), int64(1), object(1)
    memory usage: 7.2+ KB
dataset_iris.isnull().any()
[→ Id
    SepalLengthCm
                    False
    SepalWidthCm
                    False
    PetalLengthCm
                    False
    PetalWidthCm
                    False
    Species
                    False
    dtype: bool
                                                                                         V - - V E .
x = dataset_iris.iloc[:,1:-1]
    y = dataset_iris.iloc[:,-1]
    print(x.shape,y.shape)
    (150, 4) (150,)
```

#### Question4

#### Briefly identify the difference between PCA and LDA

