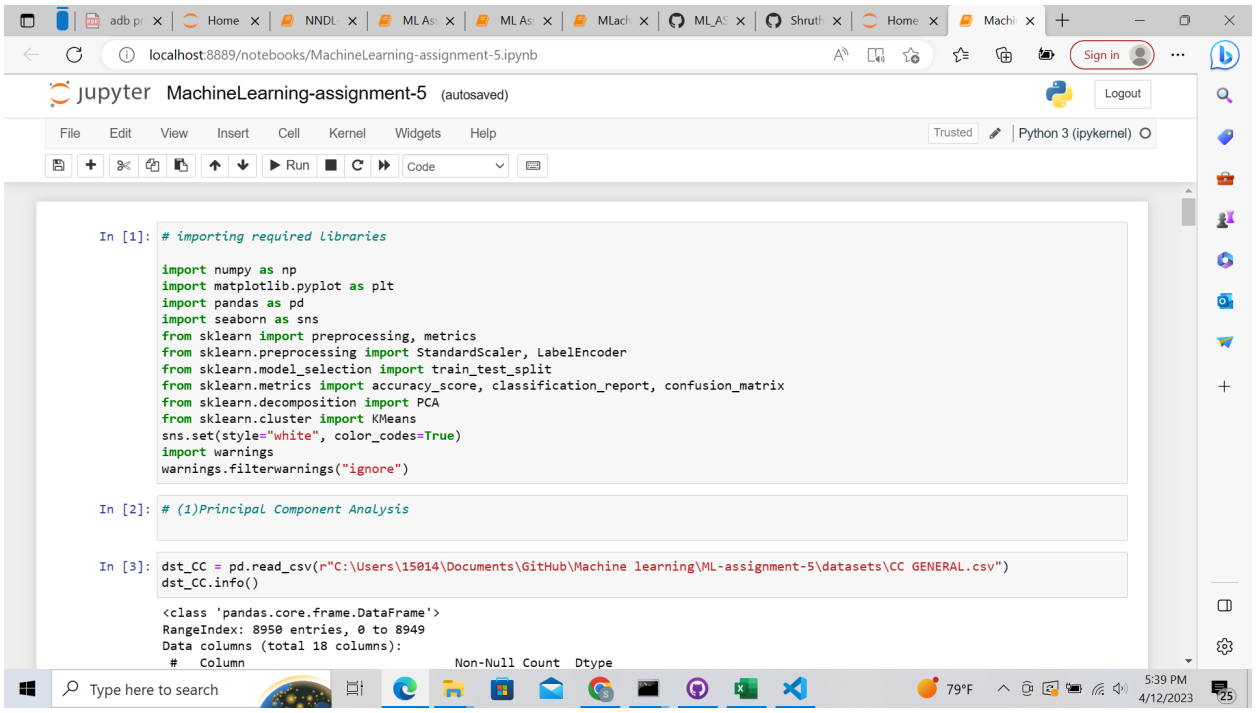
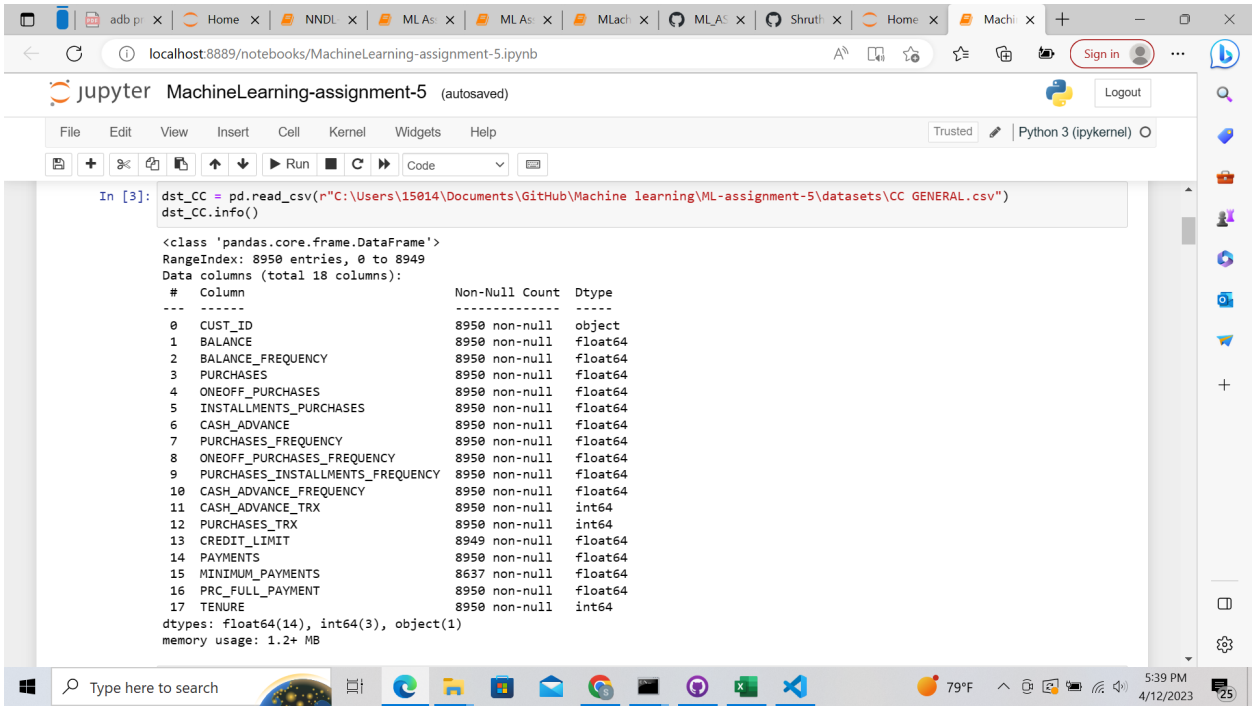
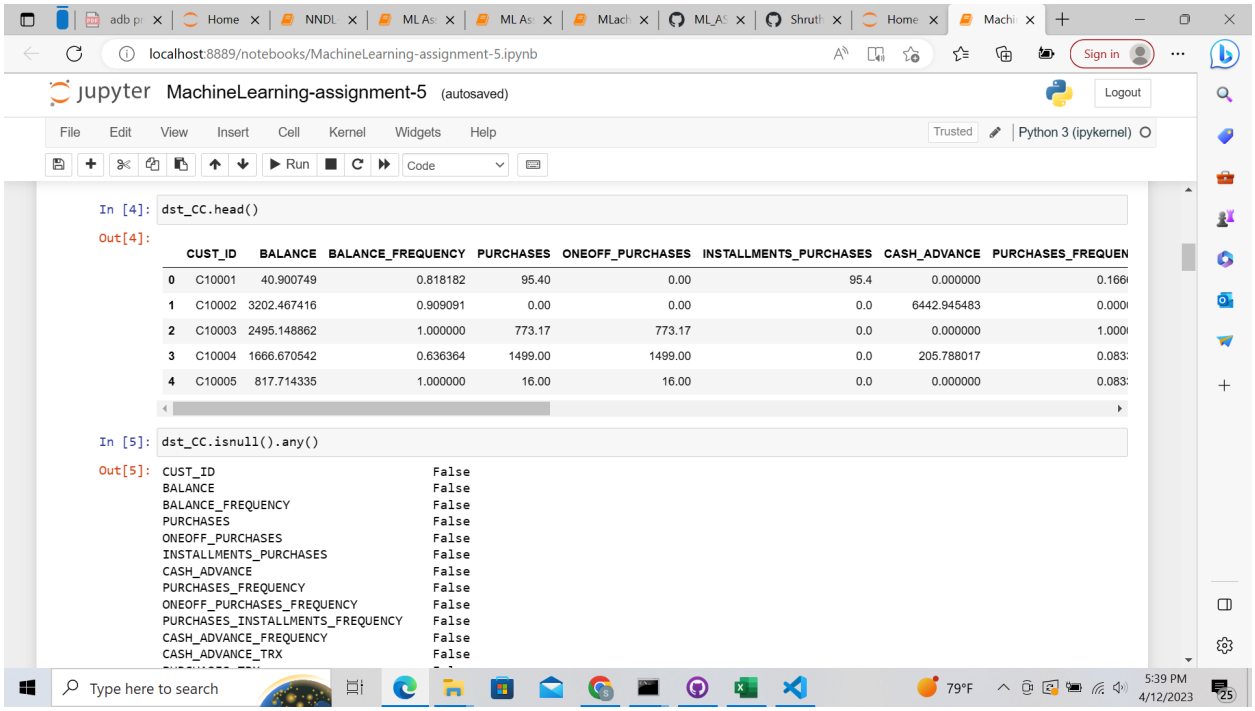
Spring 2023 5710 Machine Learning: Assignment 5

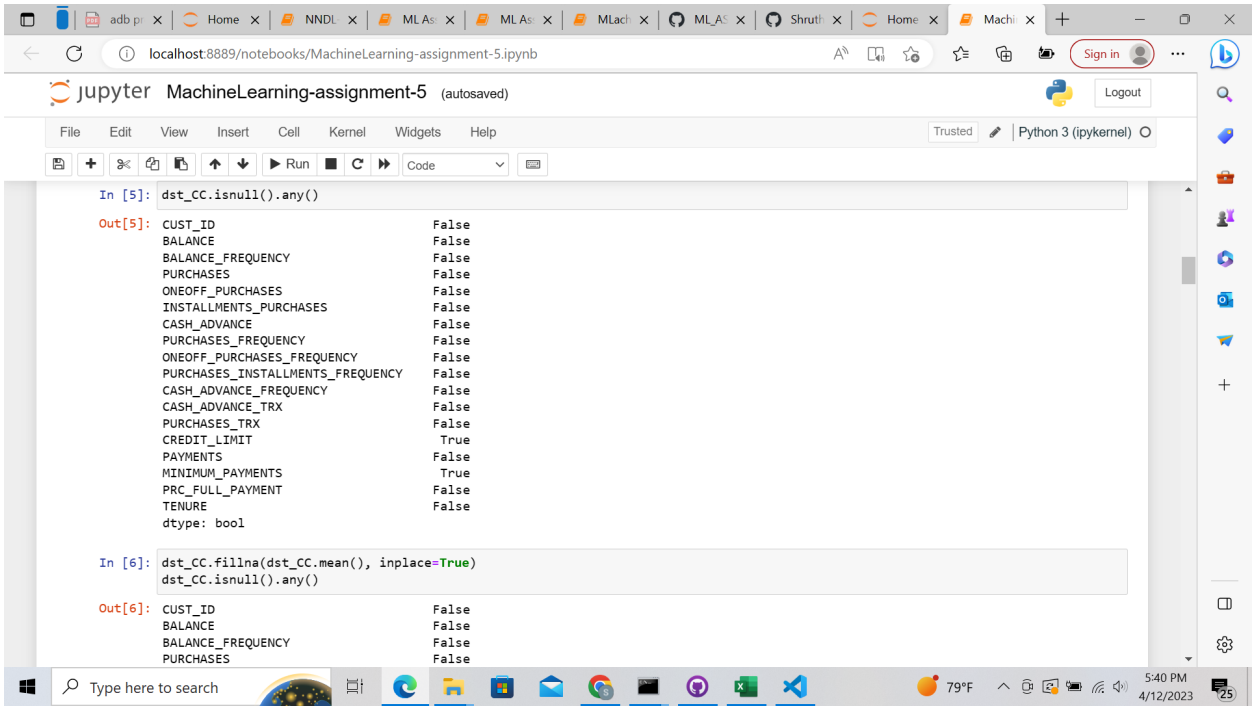
In class programming:

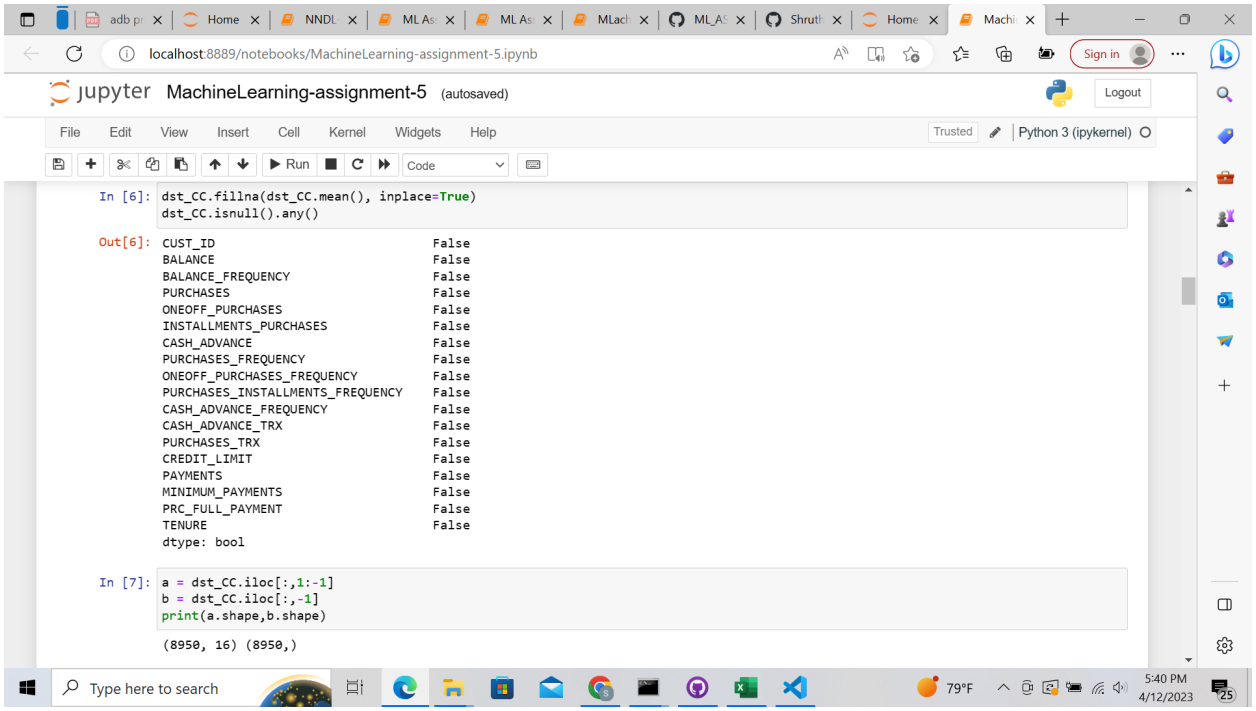
1. Principal Component Analysis
    a. Apply PCA on CC dataset.
    b. Apply k-means algorithm on the PCA result and report your observation if the silhouette score has improved or not?
    c. Perform Scaling+PCA+K-Means and report performance.

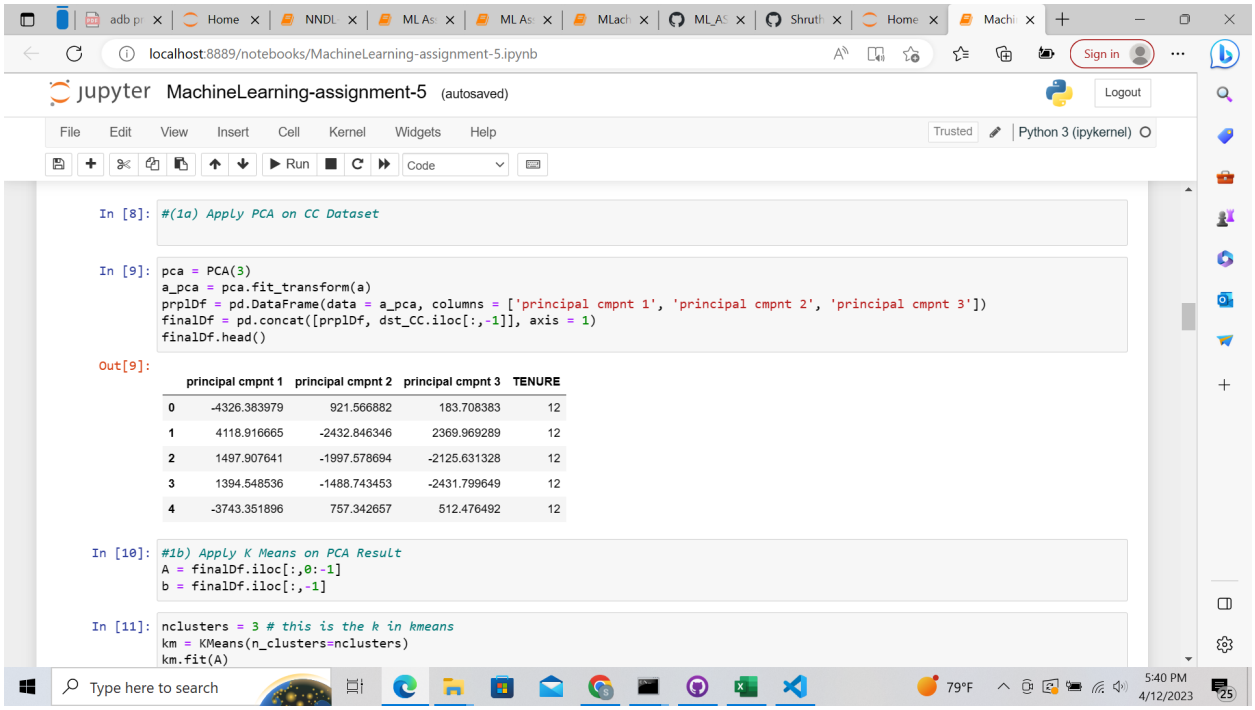


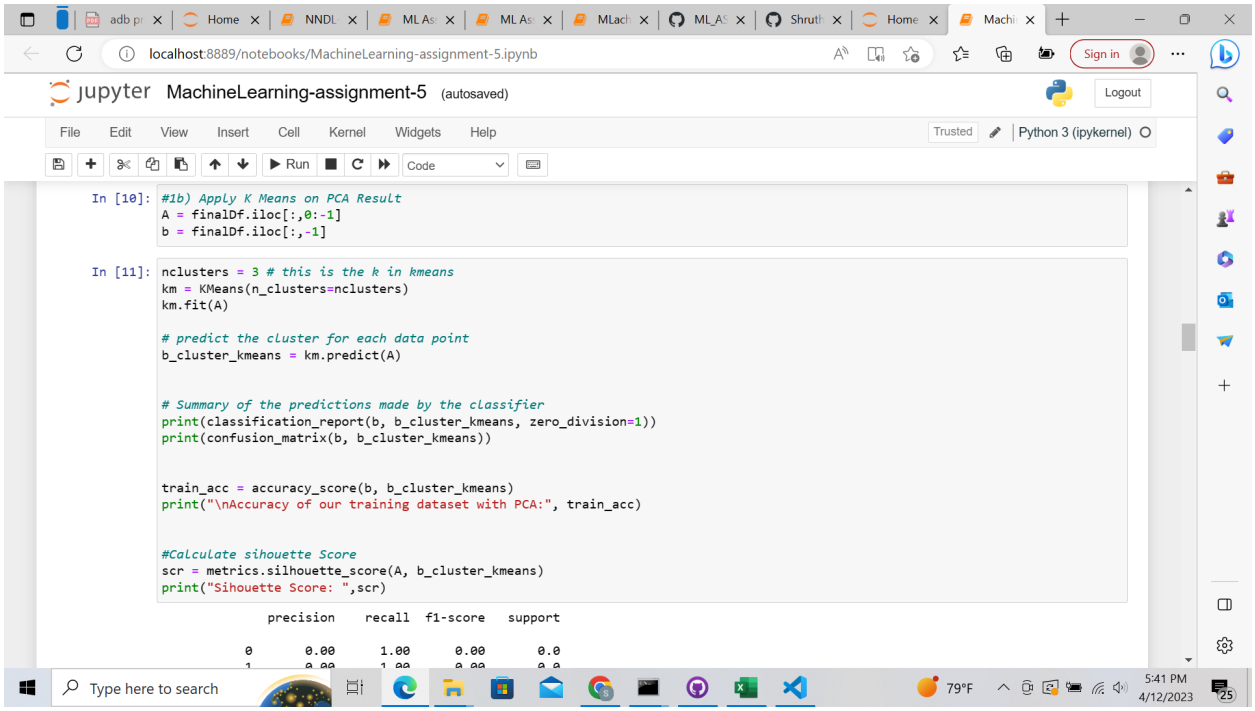

jupyter MachineLearning-assignment-5 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
In [3]:
dst_CC = pd.read_csv(r"C:\Users\15014\Documents\GitHub\Machine learning\ML-assignment-5\datasets\CC GENERAL.csv")
dst_CC.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8950 entries, 0 to 8949
Data columns (total 18 columns):
Column Non-Null Count Dtype
--- ------ -------------- -----
0 CUST_ID 8950 non-null object
1 BALANCE 8950 non-null float64
2 BALANCE_FREQUENCY 8950 non-null float64
3 PURCHASES 8950 non-null float64
4 ONEOFF_PURCHASES 8950 non-null float64
5 INSTALLMENTS_PURCHASES 8950 non-null float64
6 CASH_ADVANCE 8950 non-null float64
7 PURCHASES_FREQUENCY 8950 non-null float64
8 ONEOFF_PURCHASES_FREQUENCY 8950 non-null float64
9 PURCHASES_INSTALLMENTS_FREQUENCY 8950 non-null float64
10 CASH_ADVANCE_FREQUENCY 8950 non-null float64
11 CASH_ADVANCE_TRX 8950 non-null int64
12 PURCHASES_TRX 8950 non-null int64
13 CREDIT_LIMIT 8949 non-null float64
14 PAYMENTS 8950 non-null float64
15 MINIMUM_PAYMENTS 8637 non-null float64
16 PRC_FULL_PAYMENT 8950 non-null float64
17 TENURE 8950 non-null int64
dtypes: float64(14), int64(3), object(1)
memory usage: 1.2+ MB
Type here to search
79°F
5:39 PM
4/12/2023

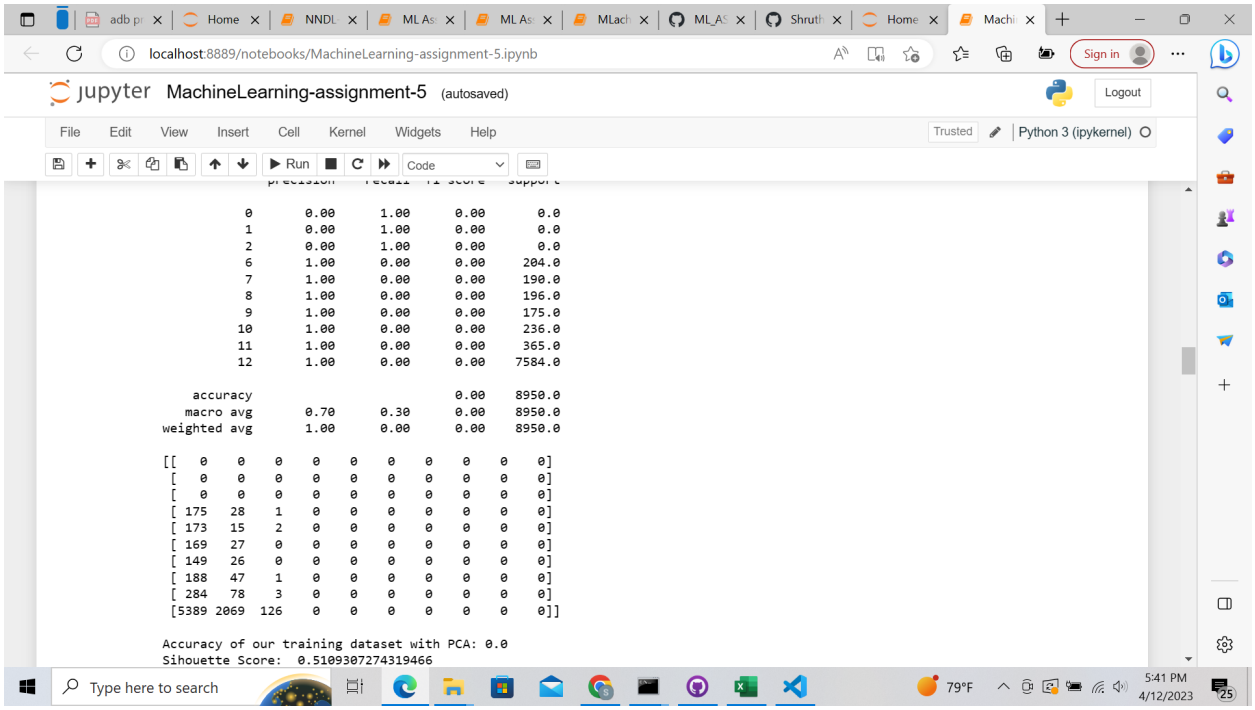

jupyter MachineLearning-assignment-5 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
In [4]:
dst_CC.head()
Out[4]:
CUST_ID BALANCE BALANCE_FREQUENCY PURCHASES ONEOFF_PURCHASES INSTALLMENTS_PURCHASES CASH_ADVANCE PURCHASES_FREQUEN
0 C10001 40.900749 0.818182 95.40 0.00 95.4 0.000000 0.166
1 C10002 3202.467416 0.909091 0.00 0.00 0.0 6442.945483 0.000
2 C10003 2495.148862 1.000000 773.17 773.17 0.0 0.000000 1.000
3 C10004 1666.670542 0.636364 1499.00 1499.00 0.0 205.788017 0.083
4 C10005 817.714335 1.000000 16.00 16.00 0.0 0.000000 0.083
In [5]:
dst_CC.isnull().any()
Out[5]:
CUST_ID False
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
Type here to search
79°F
5:39 PM
4/12/2023


localhost:8889/notebooks/MachineLearning-assignment-5.ipynb
jupyter MachineLearning-assignment-5 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
In [5]: dst_CC.isnull().any()
Out[5]:
CUST_ID False
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 False
dtype: bool
In [6]: dst_CC.fillna(dst_CC.mean(), inplace=True)
dst_CC.isnull().any()
Out[6]:
CUST_ID False
BALANCE False
BALANCE_FREQUENCY False
PURCHASES False
Type here to search
79°F
5:40 PM
4/12/2023

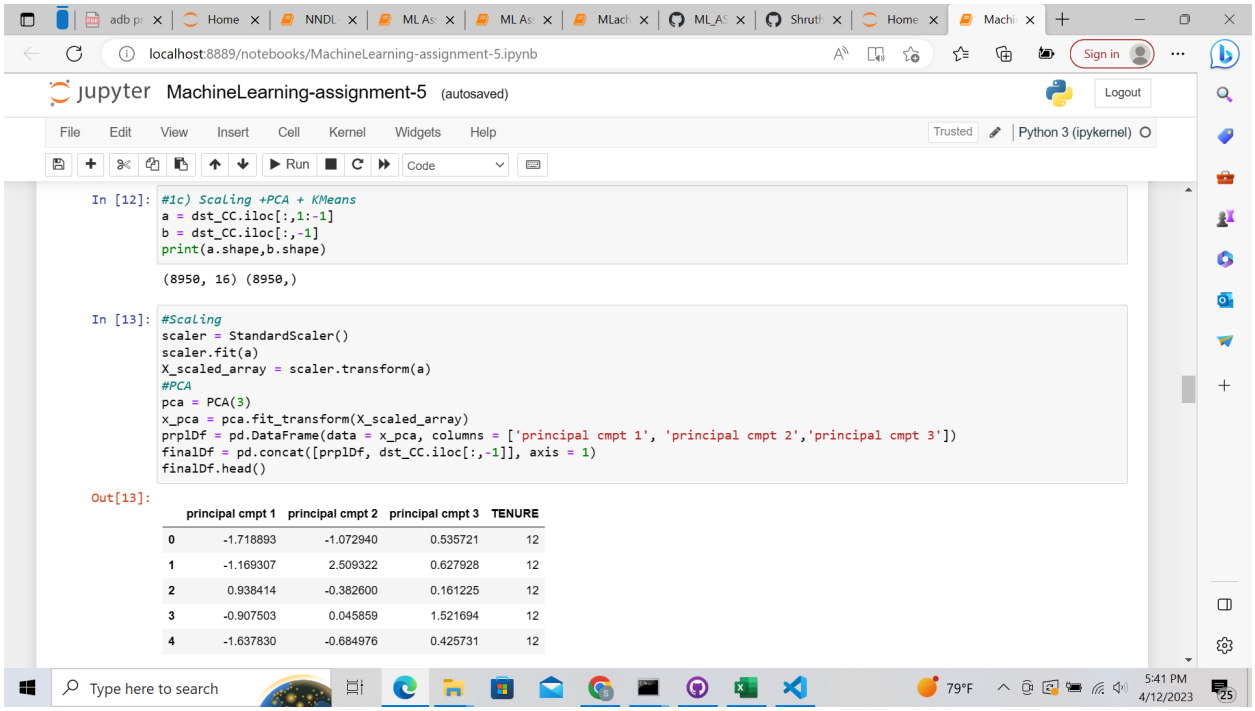

localhost:8889/notebooks/MachineLearning-assignment-5.ipynb
jupyter MachineLearning-assignment-5 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
In [6]: dst_CC.fillna(dst_CC.mean(), inplace=True)
dst_CC.isnull().any()
Out[6]:
CUST_ID False
BALANCE 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 False
PAYMENTS False
MINIMUM_PAYMENTS False
PRC_FULL_PAYMENT False
TENURE False
dtype: bool
In [7]: a = dst_CC.iloc[:,1:-1]
b = dst_CC.iloc[:,-1]
print(a.shape,b.shape)
(8950, 16) (8950,)
Type here to search
79°F
5:40 PM
4/12/2023


localhost:8889/notebooks/MachineLearning-assignment-5.ipynb
jupyter MachineLearning-assignment-5 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
In [8]:
#(1a) Apply PCA on CC Dataset
In [9]:
pca = PCA(3)
a_pca = pca.fit_transform(a)
prplDf = pd.DataFrame(data = a_pca, columns = ['principal cmpnt 1', 'principal cmpnt 2', 'principal cmpnt 3'])
finalDf = pd.concat([prplDf, dst_CC.iloc[:,-1]], axis = 1)
finalDf.head()
Out[9]:
principal cmpnt 1 principal cmpnt 2 principal cmpnt 3 TENURE
0 -4326.383979 921.566882 183.708383 12
1 4118.916665 -2432.846346 2369.969289 12
2 1497.907641 -1997.578694 -2125.631328 12
3 1394.548536 -1488.743453 -2431.799649 12
4 -3743.351896 757.342657 512.476492 12
In [10]:
#1b) Apply K Means on PCA Result
A = finalDf.iloc[:,0:-1]
b = finalDf.iloc[:,-1]
In [11]:
nclusters = 3 # this is the k in kmeans
km = KMeans(n_clusters=nclusters)
km.fit(A)
Type here to search
79°F
5:40 PM
4/12/2023


localhost:8889/notebooks/MachineLearning-assignment-5.ipynb
jupyter MachineLearning-assignment-5 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
In [10]:
#1b) Apply K Means on PCA Result
A = finalDf.iloc[:,0:-1]
b = finalDf.iloc[:,-1]
In [11]:
nclusters = 3 # this is the k in kmeans
km = KMeans(n_clusters=nclusters)
km.fit(A)
# predict the cluster for each data point
b_cluster_kmeans = km.predict(A)
# Summary of the predictions made by the classifier
print(classification_report(b, b_cluster_kmeans, zero_division=1))
print(confusion_matrix(b, b_cluster_kmeans))
train_acc = accuracy_score(b, b_cluster_kmeans)
print("\nAccuracy of our training dataset with PCA:", train_acc)
#Calculate sihouette Score
scr = metrics.silhouette_score(A, b_cluster_kmeans)
print("Sihouette Score: ",scr)
precision recall f1-score support
0 0.00 1.00 0.00 0.0
Type here to search
79°F
5:41 PM
4/12/2023



```
                   precision    recall  f1-score   support

               0       0.00      1.00      0.00       0.0
               1       0.00      1.00      0.00       0.0
               2       0.00      1.00      0.00       0.0
               6       1.00      0.00      0.00     204.0
               7       1.00      0.00      0.00     190.0
               8       1.00      0.00      0.00     196.0
               9       1.00      0.00      0.00     175.0
              10       1.00      0.00      0.00     236.0
              11       1.00      0.00      0.00     365.0
              12       1.00      0.00      0.00    7584.0

        accuracy                           0.00    8950.0
       macro avg       0.70      0.30      0.00    8950.0
    weighted avg       1.00      0.00      0.00    8950.0

[[   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    0]
 [ 175   28    1    0    0    0    0    0    0    0]
 [ 173   15    2    0    0    0    0    0    0    0]
 [ 169   27    0    0    0    0    0    0    0    0]
 [ 149   26    0    0    0    0    0    0    0    0]
 [ 188   47    1    0    0    0    0    0    0    0]
 [ 284   78    3    0    0    0    0    0    0    0]
 [5389 2069  126    0    0    0    0    0    0    0]]

Accuracy of our training dataset with PCA: 0.0
Sihouette Score:  0.5109307274319466
```



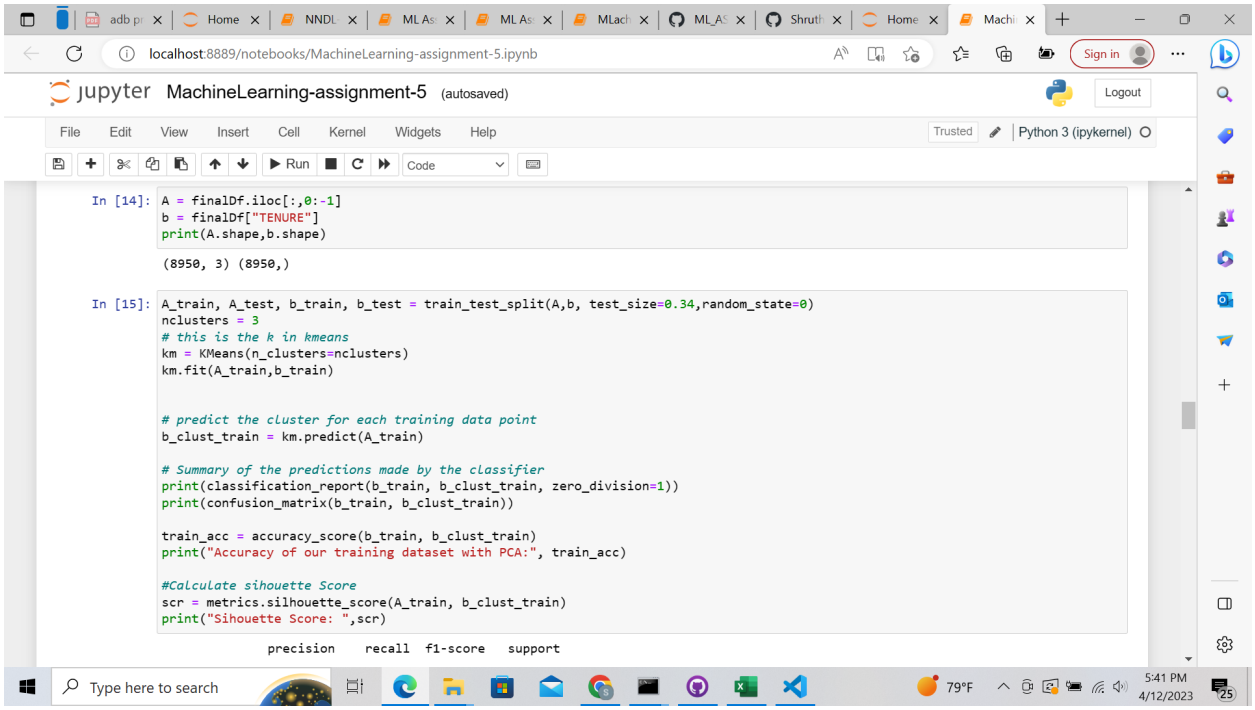
In [12]:
```
#1c) Scaling +PCA + KMeans
a = dst_CC.iloc[:,1:-1]
b = dst_CC.iloc[:,-1]
print(a.shape,b.shape)
```

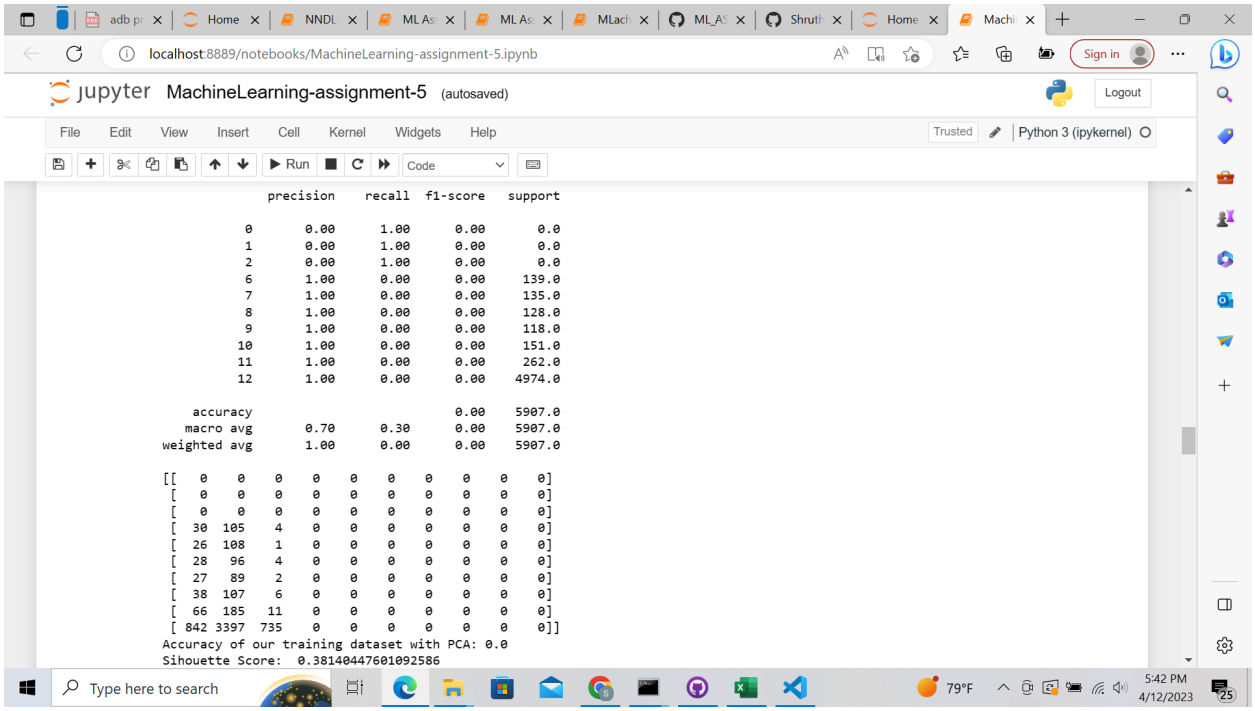
```
(8950, 16) (8950,)
```

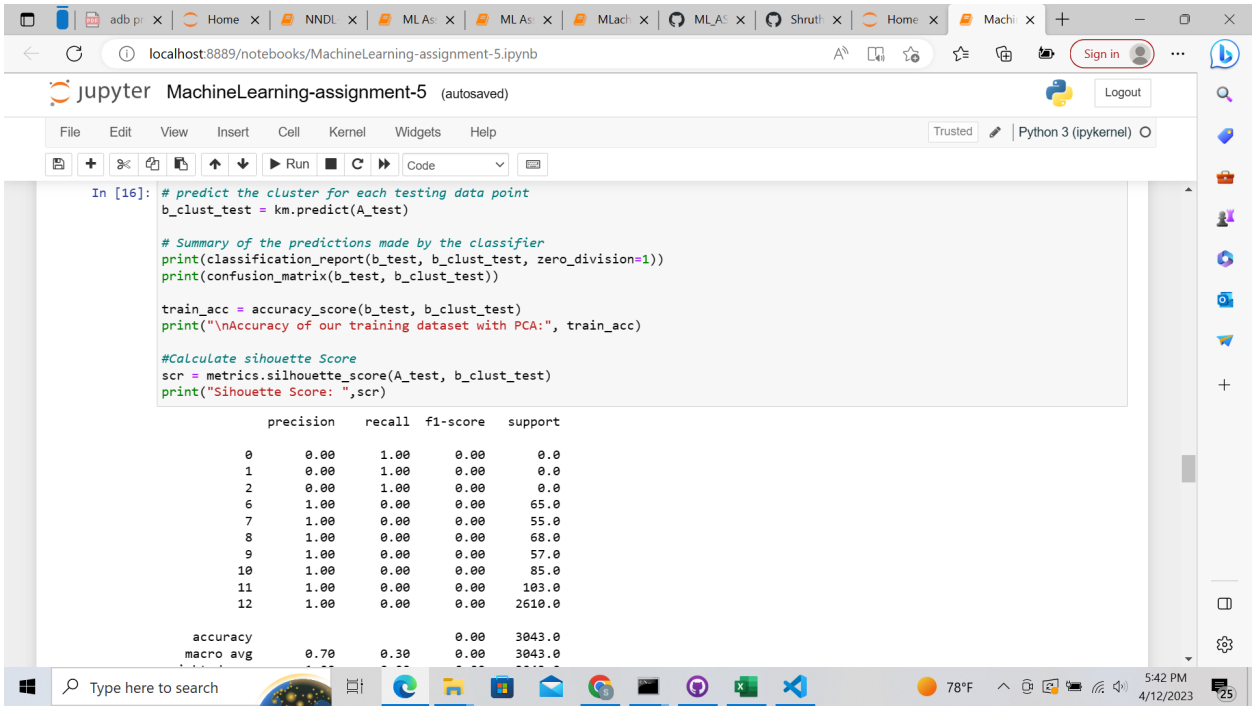
In [13]:
```
#Scaling
scaler = StandardScaler()
scaler.fit(a)
X_scaled_array = scaler.transform(a)
#PCA
pca = PCA(3)
x_pca = pca.fit_transform(X_scaled_array)
prplDf = pd.DataFrame(data = x_pca, columns = ['principal cmpt 1', 'principal cmpt 2','principal cmpt 3'])
finalDf = pd.concat([prplDf, dst_CC.iloc[:,-1]], axis = 1)
finalDf.head()
```

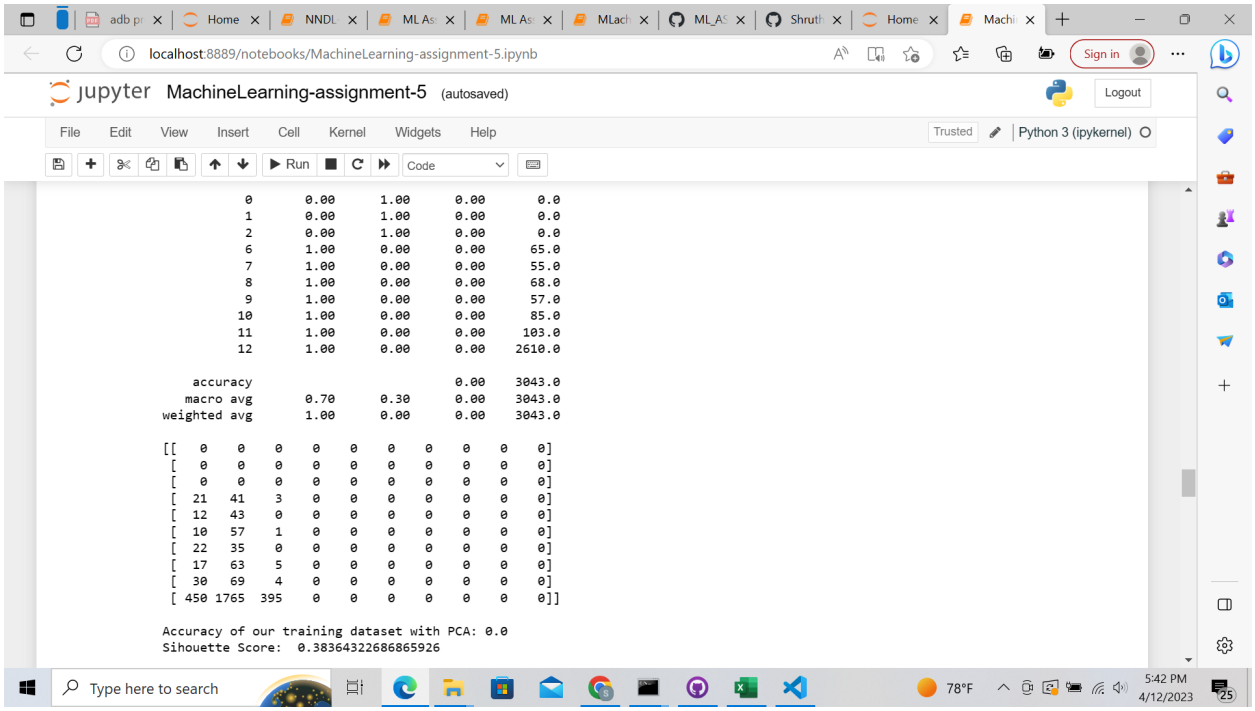
Out[13]:

	principal cmpt 1	principal cmpt 2	principal cmpt 3	TENURE
0	-1.718893	-1.072940	0.535721	12
1	-1.169307	2.509322	0.627928	12
2	0.938414	-0.382600	0.161225	12
3	-0.907503	0.045859	1.521694	12
4	-1.637830	-0.684976	0.425731	12

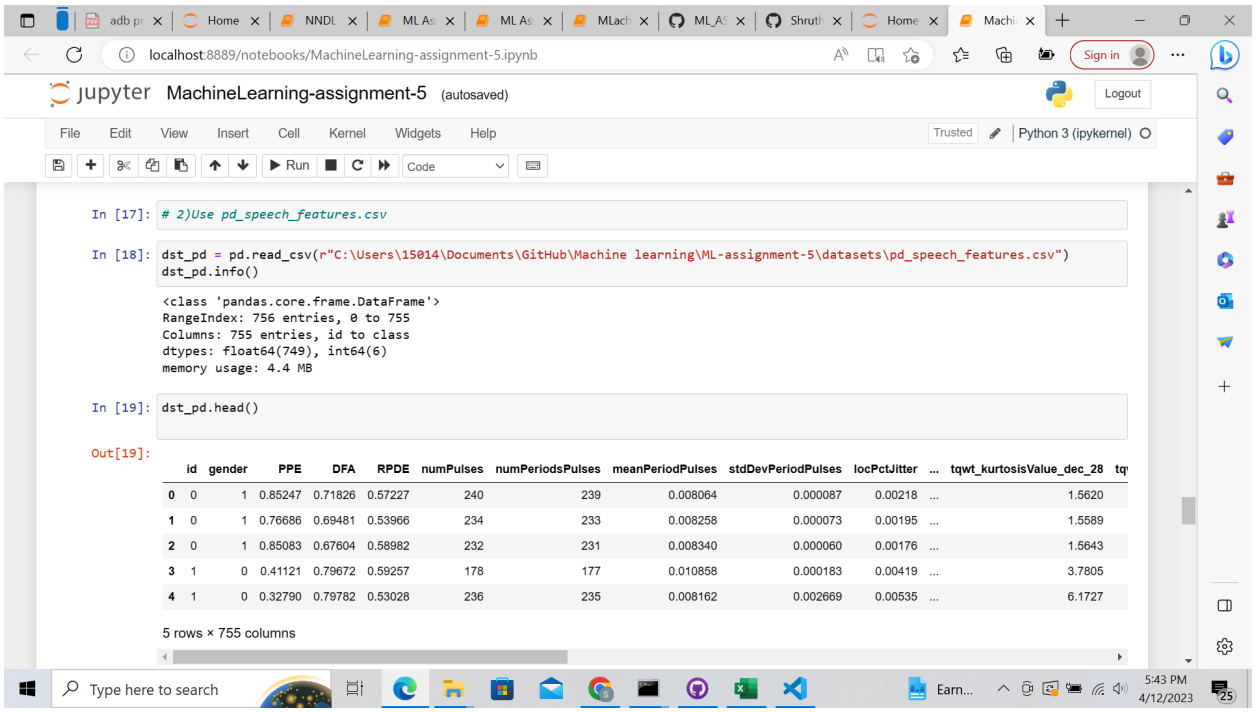

localhost:8889/notebooks/MachineLearning-assignment-5.ipynb
jupyter MachineLearning-assignment-5 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
In [14]:
A = finalDf.iloc[:,0:-1]
b = finalDf["TENURE"]
print(A.shape,b.shape)
(8950, 3) (8950,)
In [15]:
A_train, A_test, b_train, b_test = train_test_split(A,b, test_size=0.34,random_state=0)
nclusters = 3
# this is the k in kmeans
km = KMeans(n_clusters=nclusters)
km.fit(A_train,b_train)
# predict the cluster for each training data point
b_clust_train = km.predict(A_train)
# Summary of the predictions made by the classifier
print(classification_report(b_train, b_clust_train, zero_division=1))
print(confusion_matrix(b_train, b_clust_train))
train_acc = accuracy_score(b_train, b_clust_train)
print("Accuracy of our training dataset with PCA:", train_acc)
#Calculate sihouette Score
scr = metrics.silhouette_score(A_train, b_clust_train)
print("Sihouette Score: ",scr)
precision recall f1-score support
Type here to search
79°F
5:41 PM
4/12/2023


localhost:8889/notebooks/MachineLearning-assignment-5.ipynb
jupyter MachineLearning-assignment-5 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
precision recall f1-score support
0 0.00 1.00 0.00 0.0
1 0.00 1.00 0.00 0.0
2 0.00 1.00 0.00 0.0
6 1.00 0.00 0.00 139.0
7 1.00 0.00 0.00 135.0
8 1.00 0.00 0.00 128.0
9 1.00 0.00 0.00 118.0
10 1.00 0.00 0.00 151.0
11 1.00 0.00 0.00 262.0
12 1.00 0.00 0.00 4974.0
accuracy 0.00 5907.0
macro avg 0.70 0.30 0.00 5907.0
weighted avg 1.00 0.00 0.00 5907.0
[[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 0]
[30 105 4 0 0 0 0 0 0 0]
[26 108 1 0 0 0 0 0 0 0]
[28 96 4 0 0 0 0 0 0 0]
[27 89 2 0 0 0 0 0 0 0]
[38 107 6 0 0 0 0 0 0 0]
[66 185 11 0 0 0 0 0 0 0]
[842 3397 735 0 0 0 0 0 0 0]]
Accuracy of our training dataset with PCA: 0.0
Sihouette Score: 0.38140447601092586
Type here to search
79°F
5:42 PM
4/12/2023


localhost:8889/notebooks/MachineLearning-assignment-5.ipynb
jupyter MachineLearning-assignment-5 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
In [16]:
# predict the cluster for each testing data point
b_clust_test = km.predict(A_test)
# Summary of the predictions made by the classifier
print(classification_report(b_test, b_clust_test, zero_division=1))
print(confusion_matrix(b_test, b_clust_test))
train_acc = accuracy_score(b_test, b_clust_test)
print("\nAccuracy of our training dataset with PCA:", train_acc)
#Calculate sihouette Score
scr = metrics.silhouette_score(A_test, b_clust_test)
print("Sihouette Score: ",scr)
precision recall f1-score support
0 0.00 1.00 0.00 0.0
1 0.00 1.00 0.00 0.0
2 0.00 1.00 0.00 0.0
6 1.00 0.00 0.00 65.0
7 1.00 0.00 0.00 55.0
8 1.00 0.00 0.00 68.0
9 1.00 0.00 0.00 57.0
10 1.00 0.00 0.00 85.0
11 1.00 0.00 0.00 103.0
12 1.00 0.00 0.00 2610.0
accuracy 0.00 3043.0
macro avg 0.70 0.30 0.00 3043.0
Type here to search
78°F
5:42 PM
4/12/2023


localhost:8889/notebooks/MachineLearning-assignment-5.ipynb
jupyter MachineLearning-assignment-5 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
0 0.00 1.00 0.00 0.0
1 0.00 1.00 0.00 0.0
2 0.00 1.00 0.00 0.0
6 1.00 0.00 0.00 65.0
7 1.00 0.00 0.00 55.0
8 1.00 0.00 0.00 68.0
9 1.00 0.00 0.00 57.0
10 1.00 0.00 0.00 85.0
11 1.00 0.00 0.00 103.0
12 1.00 0.00 0.00 2610.0
accuracy 0.00 3043.0
macro avg 0.70 0.30 0.00 3043.0
weighted avg 1.00 0.00 0.00 3043.0
[[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 0]
[21 41 3 0 0 0 0 0 0 0]
[12 43 0 0 0 0 0 0 0 0]
[10 57 1 0 0 0 0 0 0 0]
[22 35 0 0 0 0 0 0 0 0]
[17 63 5 0 0 0 0 0 0 0]
[30 69 4 0 0 0 0 0 0 0]
[450 1765 395 0 0 0 0 0 0 0]]
Accuracy of our training dataset with PCA: 0.0
Sihouette Score: 0.38364322686865926
Type here to search
78°F
5:42 PM
4/12/2023

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



localhost:8889/notebooks/MachineLearning-assignment-5.ipynb

jupyter MachineLearning-assignment-5 (autosaved)

File Edit View Insert Cell Kernel Widgets Help — Trusted — Python 3 (ipykernel)

Run — Code

In [17]:
```
# 2)Use pd_speech_features.csv
```

In [18]:
```
dst_pd = pd.read_csv(r"C:\Users\15014\Documents\GitHub\Machine learning\ML-assignment-5\datasets\pd_speech_features.csv")
dst_pd.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 756 entries, 0 to 755
Columns: 755 entries, id to class
dtypes: float64(749), int64(6)
memory usage: 4.4 MB
```

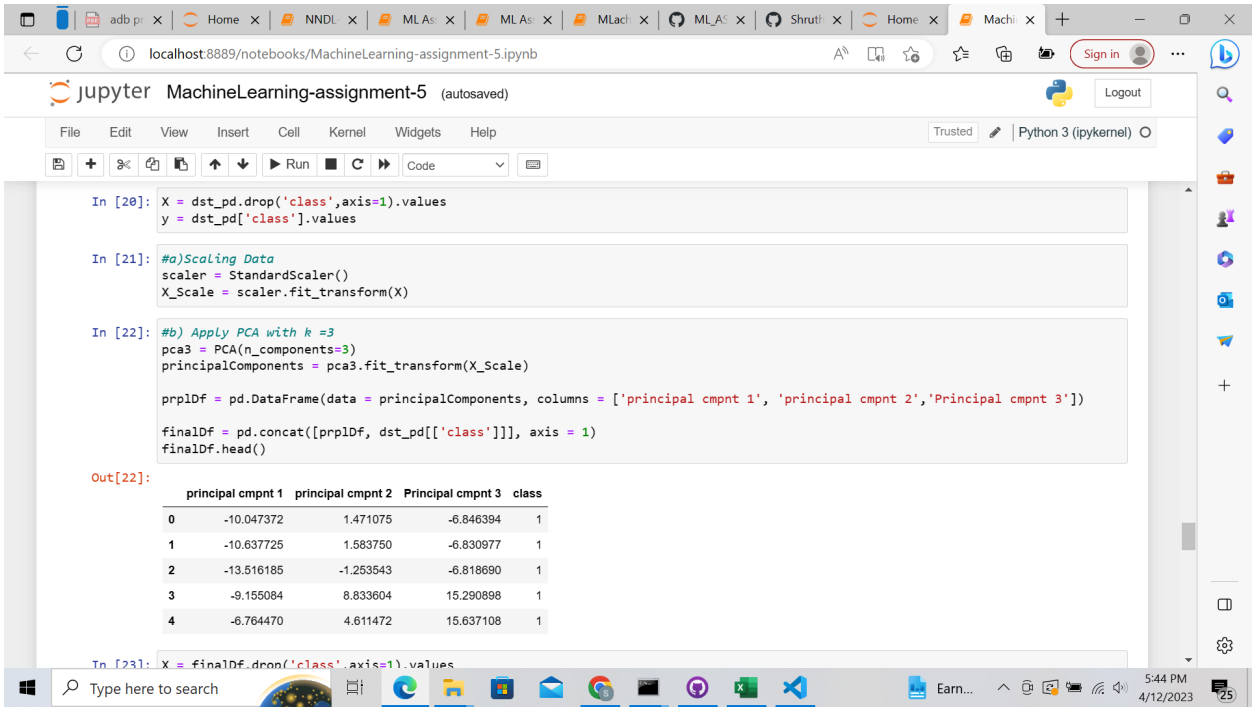
In [19]:
```
dst_pd.head()
```

Out[19]:

	id	gender	PPE	DFA	RPDE	numPulses	numPeriodsPulses	meanPeriodPulses	stdDevPeriodPulses	locPctJitter	...	tqwt_kurtosisValue_dec_28	tq
0	0	1	0.85247	0.71826	0.57227	240	239	0.008064	0.000087	0.00218	...	1.5620	
1	0	1	0.76686	0.69481	0.53966	234	233	0.008258	0.000073	0.00195	...	1.5589	
2	0	1	0.85083	0.67604	0.58982	232	231	0.008340	0.000060	0.00176	...	1.5643	
3	1	0	0.41121	0.79672	0.59257	178	177	0.010858	0.000183	0.00419	...	3.7805	
4	1	0	0.32790	0.79782	0.53028	236	235	0.008162	0.002669	0.00535	...	6.1727	

5 rows × 755 columns

Type here to search — Earn... — 5:43 PM 4/12/2023



localhost:8889/notebooks/MachineLearning-assignment-5.ipynb

jupyter MachineLearning-assignment-5 (autosaved)

File Edit View Insert Cell Kernel Widgets Help — Trusted — Python 3 (ipykernel)

Run — Code

In [20]:
```
X = dst_pd.drop('class',axis=1).values
y = dst_pd['class'].values
```

In [21]:
```
#a)Scaling Data
scaler = StandardScaler()
X_Scale = scaler.fit_transform(X)
```

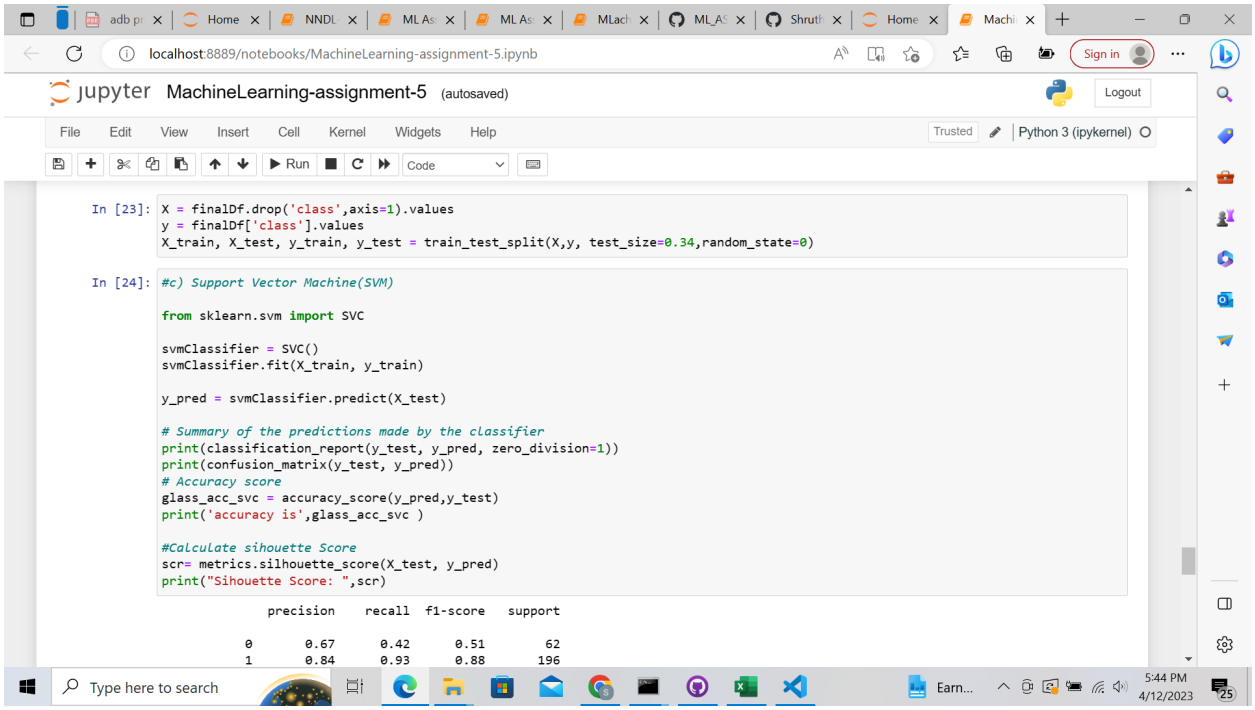
In [22]:
```
#b) Apply PCA with k =3
pca3 = PCA(n_components=3)
principalComponents = pca3.fit_transform(X_Scale)

prplDf = pd.DataFrame(data = principalComponents, columns = ['principal cmpnt 1', 'principal cmpnt 2','Principal cmpnt 3'])

finalDf = pd.concat([prplDf, dst_pd[['class']]], axis = 1)
finalDf.head()
```

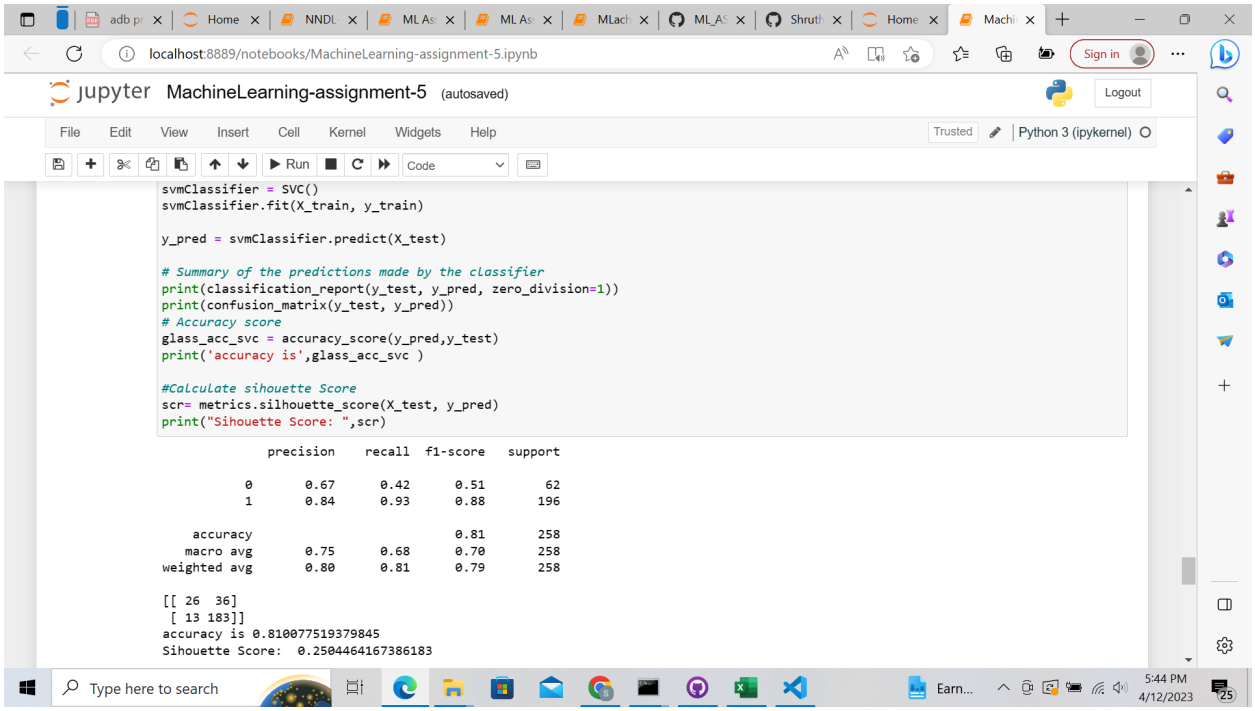
Out[22]:

	principal cmpnt 1	principal cmpnt 2	Principal cmpnt 3	class
0	-10.047372	1.471075	-6.846394	1
1	-10.637725	1.583750	-6.830977	1
2	-13.516185	-1.253543	-6.818690	1
3	-9.155084	8.833604	15.290898	1
4	-6.764470	4.611472	15.637108	1

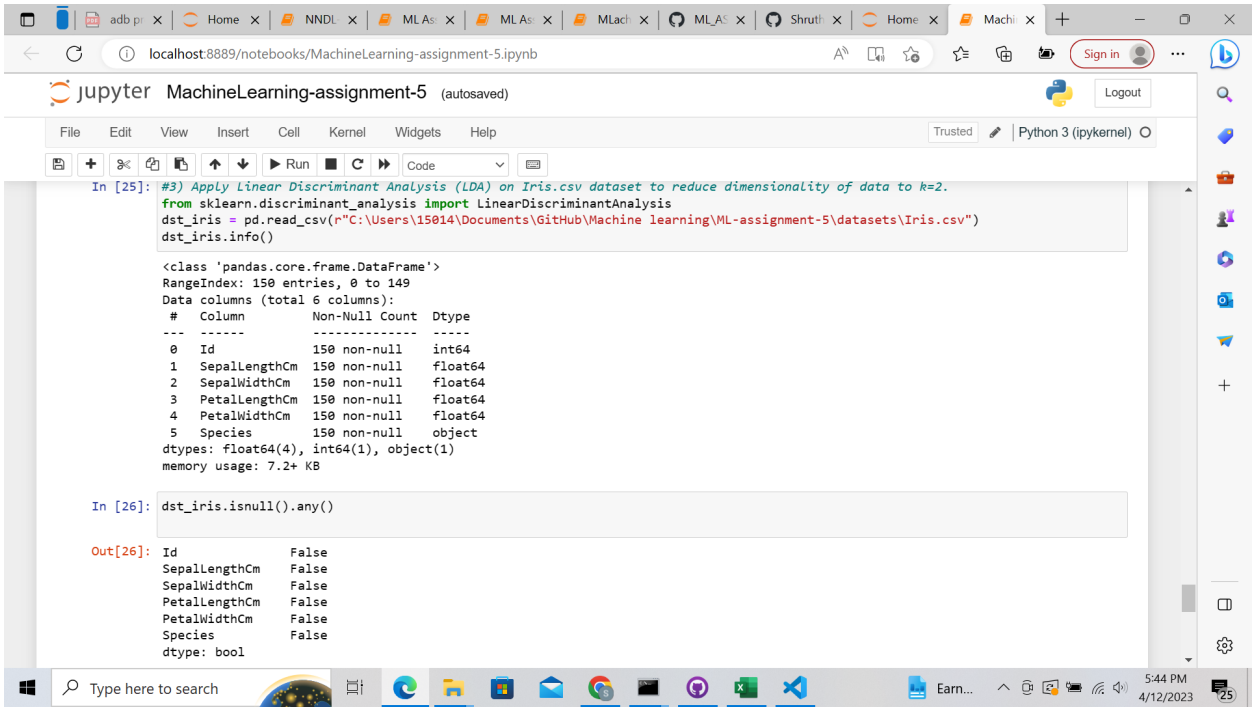
In [23]:
```
X = finalDf.drop('class',axis=1).values
```

Type here to search — Earn... — 5:44 PM 4/12/2023


localhost:8889/notebooks/MachineLearning-assignment-5.ipynb
jupyter MachineLearning-assignment-5 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
In [23]:
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)
In [24]:
#c) Support Vector Machine(SVM)
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
scr= metrics.silhouette_score(X_test, y_pred)
print("Sihouette Score: ",scr)
precision recall f1-score support
0 0.67 0.42 0.51 62
1 0.84 0.93 0.88 196
Type here to search
5:44 PM
4/12/2023


localhost:8889/notebooks/MachineLearning-assignment-5.ipynb
jupyter MachineLearning-assignment-5 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
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
scr= metrics.silhouette_score(X_test, y_pred)
print("Sihouette Score: ",scr)
precision recall f1-score support
0 0.67 0.42 0.51 62
1 0.84 0.93 0.88 196
accuracy 0.81 258
macro avg 0.75 0.68 0.70 258
weighted avg 0.80 0.81 0.79 258
[[26 36]
[13 183]]
accuracy is 0.8100775193798450
Sihouette Score: 0.2504464167386183
Type here to search
5:44 PM
4/12/2023

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

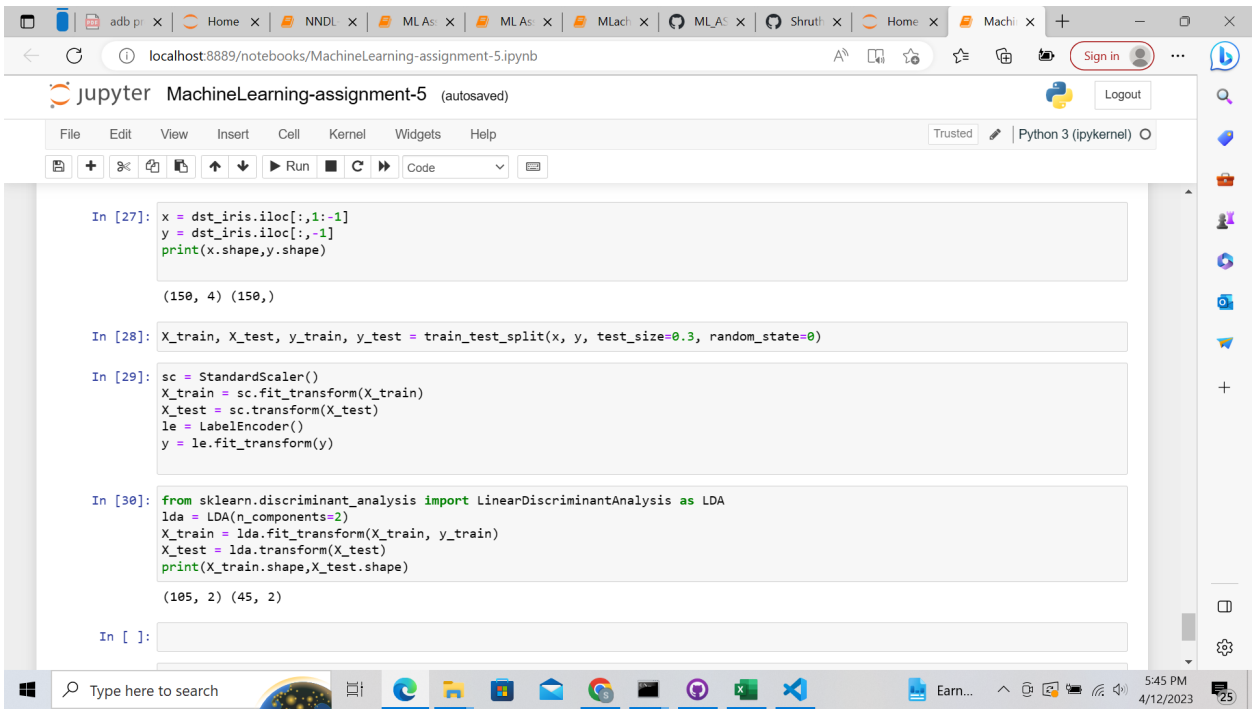


```
In [25]:
#3) Apply Linear Discriminant Analysis (LDA) on Iris.csv dataset to reduce dimensionality of data to k=2.
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
dst_iris = pd.read_csv(r"C:\Users\15014\Documents\GitHub\Machine learning\ML-assignment-5\datasets\Iris.csv")
dst_iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             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
```

```
In [26]:
dst_iris.isnull().any()
```

```
Out[26]:
Id               False
SepalLengthCm    False
SepalWidthCm     False
PetalLengthCm    False
PetalWidthCm     False
Species          False
dtype: bool
```



```
In [27]:
x = dst_iris.iloc[:,1:-1]
y = dst_iris.iloc[:,-1]
print(x.shape,y.shape)
```

```
(150, 4) (150,)
```

```
In [28]:
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=0)
```

```
In [29]:
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
le = LabelEncoder()
y = le.fit_transform(y)
```

```
In [30]:
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
lda = LDA(n_components=2)
X_train = lda.fit_transform(X_train, y_train)
X_test = lda.transform(X_test)
print(X_train.shape,X_test.shape)
```

```
(105, 2) (45, 2)
```

```
In [ ]:
```

4. Briefly identify the difference between PCA and LDA
Both LDA and PCA are based on linear transformations and aim to maximize the variance of the lower dimension. PCA is an unsupervised learning algorithm while LDA is a supervised learning algorithm.That is, PCA finds indications of maximum variance regardless of class labels, while LDA finds indications of maximum class separation

GitHub Repo link : https://github.com/ShruthiVallapReddy/ML-assignment-5.git