ARTIFICIAL INTELLIGENCE LAB (CSL5402)

Name: Lakhan Kumawat

Roll: 1906055

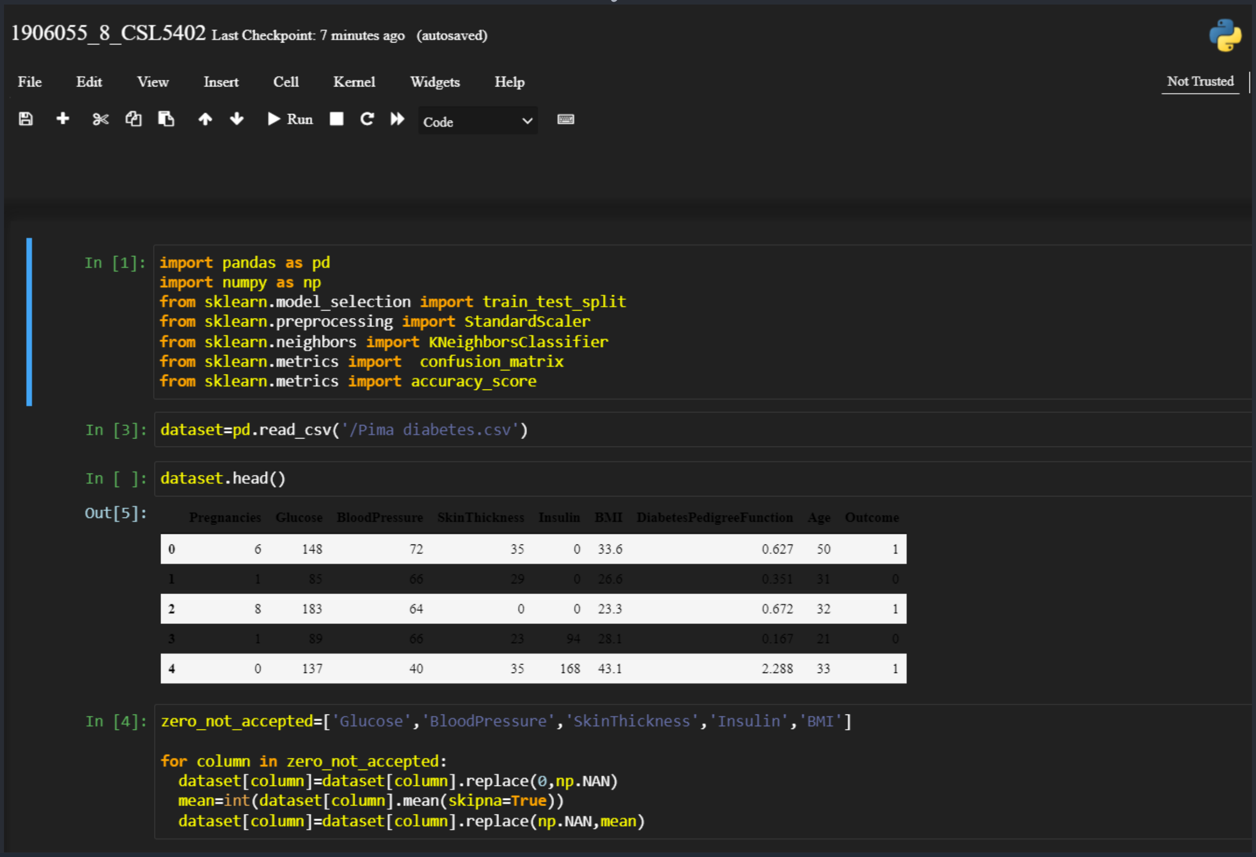
Program: B.Tech CSE
(5th Sem JUL-DEC 2021)

Assignment - 8

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score


dataset=pd.read_csv('/Pima diabetes.csv')
dataset.head()
zero_not_accepted=['Glucose','BloodPressure','SkinThickness','Insulin','BMI']

for column in zero_not_accepted:
dataset[column]=dataset[column].replace(0,np.NAN)
mean=int(dataset[column].mean(skipna=True))
dataset[column]=dataset[column].replace(np.NAN,mean)
x=dataset.iloc[:,0:8]
y=dataset.iloc[:,8]
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=0,test_size=0.2)

sc_x=StandardScaler()
x_train=sc_x.fit_transform(x_train)
x_test=sc_x.transform(x_test)
classifier=KNeighborsClassifier(n_neighbors=11,metric='euclidean')

classifier.fit(x_train,y_train)
y_pred=classifier.predict(x_test)

y_pred
```

```
cm=confusion_matrix(y_test,y_pred)
cm
print(accuracy_score(y_test,y_pred))
```

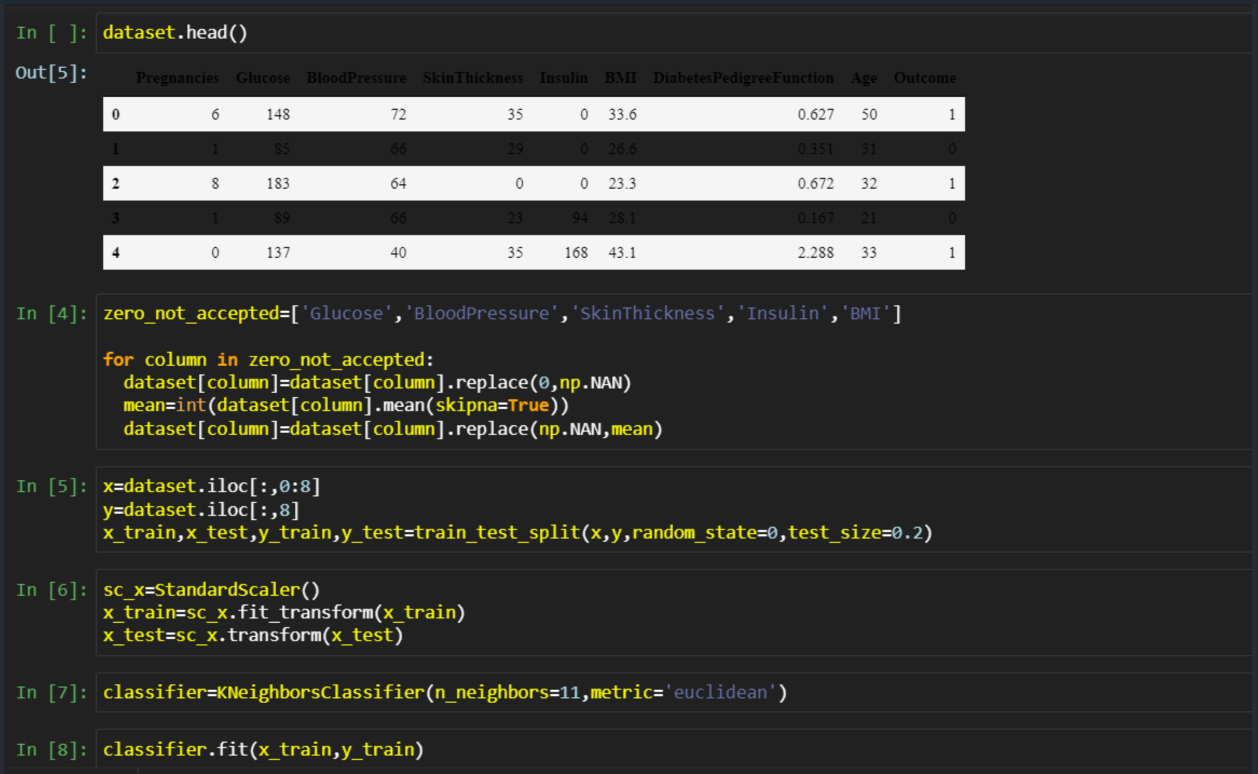
Outputs:


1906055_8_CSL5402 Last Checkpoint: 7 minutes ago (autosaved)
Not Trusted
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Run
Code

```
In [1]: import pandas as pd
        import numpy as np
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import StandardScaler
        from sklearn.neighbors import KNeighborsClassifier
        from sklearn.metrics import  confusion_matrix
        from sklearn.metrics import accuracy_score

In [3]: dataset=pd.read_csv('/Pima diabetes.csv')

In [ ]: dataset.head()

Out[5]:
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
In [4]: zero_not_accepted=['Glucose','BloodPressure','SkinThickness','Insulin','BMI']

        for column in zero_not_accepted:
          dataset[column]=dataset[column].replace(0,np.NAN)
          mean=int(dataset[column].mean(skipna=True))
          dataset[column]=dataset[column].replace(np.NAN,mean)
```

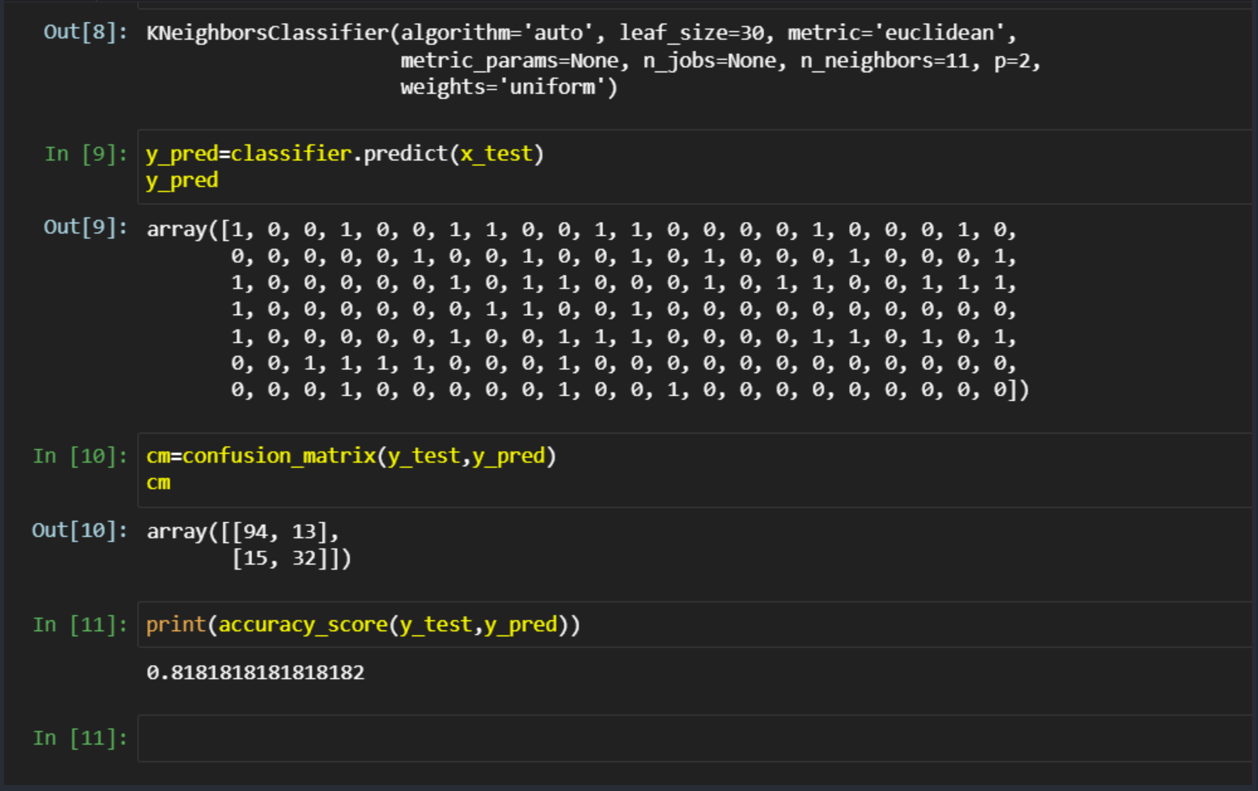


```
In [ ]: dataset.head()
```

Out[5]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
In [4]: zero_not_accepted=['Glucose','BloodPressure','SkinThickness','Insulin','BMI']

        for column in zero_not_accepted:
          dataset[column]=dataset[column].replace(0,np.NAN)
          mean=int(dataset[column].mean(skipna=True))
          dataset[column]=dataset[column].replace(np.NAN,mean)

In [5]: x=dataset.iloc[:,0:8]
        y=dataset.iloc[:,8]
        x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=0,test_size=0.2)

In [6]: sc_x=StandardScaler()
        x_train=sc_x.fit_transform(x_train)
        x_test=sc_x.transform(x_test)

In [7]: classifier=KNeighborsClassifier(n_neighbors=11,metric='euclidean')

In [8]: classifier.fit(x_train,y_train)
```



```
Out[8]: KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='euclidean',
                             metric_params=None, n_jobs=None, n_neighbors=11, p=2,
                             weights='uniform')

 In [9]: y_pred=classifier.predict(x_test)
         y_pred

Out[9]: array([1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0,
               0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1,
               1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1,
               1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1,
               0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0])

In [10]: cm=confusion_matrix(y_test,y_pred)
         cm

Out[10]: array([[94, 13],
                [15, 32]])

In [11]: print(accuracy_score(y_test,y_pred))

         0.8181818181818182

In [11]:
```

End Of Assignment