PROJECT : DIABETES PREDICTION SYSTEM

SUBMITTED BY : RAMYA A

Mail ID : arumugammohana300@gmail.com

Phase-03

Loading and Pre-Processing the Dataset

ABOUT THIS PHASE :

In this phase we need to do loading and pre-processing the datasets. Here I explain about what are the process to do this phase.

Step 1:

Import the dependencies

In this step we import the library files which are required to run this program code, like modules (numpy , pandas, sklearn , matpltlib, seaborn)

Step 2:

Impoting the dataset

In this step I import PIMA diabetes dataset form the the sklearn module it is used to fetch the data and the data is used as the input of this project.

Step 3:

Statical measure of data

In this step I want know some statics about my dataset like mean,count,avg etc,.. and this is the important step in data preprocessing

Step4:

Data standardization:

In this step I standardized my dataset with the help of scaler function

Import the dependencies

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

Data collection and analysis

PIMA Diabetes Dataset

```
# loading the dataset to the pandas dataframe
diabetes_dataset = pd.read_csv('/content/diabetes.csv')
```

```
pd.read_csv?
```

```
# printing the first 5 rows of the dataset
diabetes_dataset.head()
```

	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

```
# number of rows and column in this dataset
diabetes_dataset.shape
```

```
(768, 9)
```

```
# getting the statistical measures of the data
diabetes_dataset.describe()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI D
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000

```
diabetes_dataset['Outcome'].value_counts()
```

```
0    500
1    268
Name: Outcome, dtype: int64
```

0--> Non-Diabetic 1-->

Diabetic

```
diabetes_dataset.groupby('Outcome').mean()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI
Outcome						
0	3.298000	109.980000	68.184000	19.664000	68.792000	30.304200
1	4.865672	141.257463	70.824627	22.164179	100.335821	35.142537

```
# seperating the data and labels
X = diabetes_dataset.drop(columns = 'Outcome', axis=1)
Y = diabetes_dataset['Outcome']
```

```
print(X)
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6
1              1       85             66             29        0  26.6
2              8      183             64              0        0  23.3
3              1       89             66             23       94  28.1
4              0      137             40             35      168  43.1
..           ...      ...            ...            ...      ...   ...
763           10      101             76             48      180  32.9
764            2      122             70             27        0  36.8
765            5      121             72             23      112  26.2
766            1      126             60              0        0  30.1
767            1       93             70             31        0  30.4

     DiabetesPedigreeFunction  Age
0                       0.627   50
1                       0.351   31
2                       0.672   32
3                       0.167   21
4                       2.288   33
..                        ...  ...
763                     0.171   63
764                     0.340   27
765                     0.245   30
766                     0.349   47
767                     0.315   23

[768 rows x 8 columns]
```

```
print(Y)
```

```
0      1
1      0
2      1
3      0
4      1       ..
763    0
764    0
765    0
766    1
767    0
Name: Outcome, Length: 768, dtype: int64
```

Data Standardization

```
scaler = StandardScaler()
```

```
scaler.fit(X)
```

```
▾StandardScaler
StandardScaler()
```

```
standardized_data = scaler.transform(X)
```

```
print(standardized_data)
```

```
[[ 0.63994726  0.84832379  0.14964075 ...  0.20401277  0.46849198
   1.4259954 ]  [-0.84488505 -1.12339636 -0.16054575 ... -
0.68442195 -0.36506078
  -0.19067191]
 [ 1.23388019  1.94372388 -0.26394125 ... -1.10325546  0.60439732
  -0.10558415]
...
 [ 0.3429808   0.00330087  0.14964075 ... -0.73518964 -0.68519336
  -0.27575966]
 [-0.84488505  0.1597866  -0.47073225 ... -0.24020459 -0.37110101
```

```
          1.17073215]  [-0.84488505 -0.8730192   0.04624525 ... -
     0.20212881 -0.47378505
       -0.87137393]]
X = standardized_data Y =
diabetes_dataset['Outcome']


print(X)
print(Y)
```

```
     [[ 0.63994726  0.84832379  0.14964075 ...  0.20401277  0.46849198
        1.4259954 ]  [-0.84488505 -1.12339636 -0.16054575 ... -
     0.68442195 -0.36506078
       -0.19067191]
      [ 1.23388019  1.94372388 -0.26394125 ... -1.10325546  0.60439732
       -0.10558415]
     ...
      [ 0.3429808   0.00330087  0.14964075 ... -0.73518964 -0.68519336
       -0.27575966]
      [-0.84488505  0.1597866  -0.47073225 ... -0.24020459 -0.37110101
        1.17073215]  [-0.84488505 -0.8730192   0.04624525 ... -
     0.20212881 -0.47378505
       -0.87137393]]
     0      1
     1      0
     2      1
     3      0
     4      1
     ..
     763    0
     764    0
     765    0
     766    1
     767    0
     Name: Outcome, Length: 768, dtype: int64
```