```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

df = pd.read_csv("diabetes.csv")

df.head()
```

```
   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


   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1
```

```
df.columns
```

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness',
'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

```
df.dtypes
```

```
Pregnancies                   int64
Glucose                       int64
BloodPressure                 int64
SkinThickness                 int64
Insulin                       int64
BMI                         float64
DiabetesPedigreeFunction    float64
Age                           int64
Outcome                       int64
dtype: object
```

```
df.isna().sum()
```

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

```
df.isnull().sum()
```

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

```
df.head(20)
```

	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
5	5	116	74	0	0	25.6
6	3	78	50	32	88	31.0
7	10	115	0	0	0	35.3
8	2	197	70	45	543	30.5
9	8	125	96	0	0	0.0
10	4	110	92	0	0	37.6
11	10	168	74	0	0	38.0

12	10	139	80	0	0	27.1
13	1	189	60	23	846	30.1
14	5	166	72	19	175	25.8
15	7	100	0	0	0	30.0
16	0	118	84	47	230	45.8
17	7	107	74	0	0	29.6
18	1	103	30	38	83	43.3
19	1	115	70	30	96	34.6

	DiabetesPedigreeFunction	Age	Outcome
0	0.627	50	1
1	0.351	31	0
2	0.672	32	1
3	0.167	21	0
4	2.288	33	1
5	0.201	30	0
6	0.248	26	1
7	0.134	29	0
8	0.158	53	1
9	0.232	54	1
10	0.191	30	0
11	0.537	34	1
12	1.441	57	0
13	0.398	59	1
14	0.587	51	1
15	0.484	32	1
16	0.551	31	1
17	0.254	31	1
18	0.183	33	0
19	0.529	32	1

```
df.shape
```

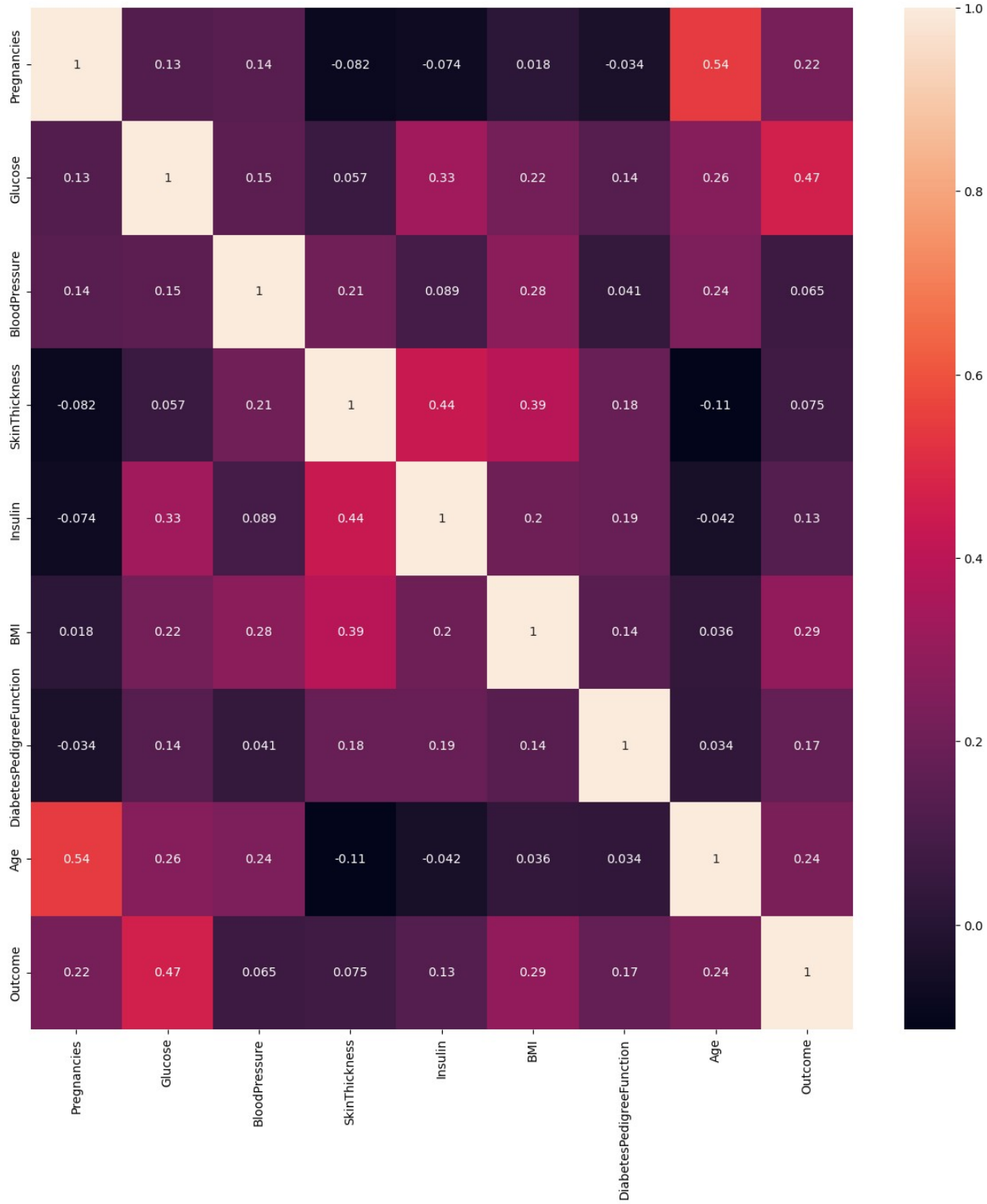
```
(768, 9)
```

```
df.corr()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness
Pregnancies	1.000000	0.129459	0.141282	-0.081672
Glucose	0.129459	1.000000	0.152590	

```
0.057328
BloodPressure                 0.141282  0.152590       1.000000
0.207371
SkinThickness                -0.081672  0.057328       0.207371
1.000000
Insulin                      -0.073535  0.331357       0.088933
0.436783
BMI                           0.017683  0.221071       0.281805
0.392573
DiabetesPedigreeFunction     -0.033523  0.137337       0.041265
0.183928
Age                           0.544341  0.263514       0.239528      -
0.113970
Outcome                       0.221898  0.466581       0.065068
0.074752

                             Insulin       BMI  DiabetesPedigreeFunction
\
Pregnancies                -0.073535  0.017683                 -0.033523

Glucose                     0.331357  0.221071                  0.137337

BloodPressure               0.088933  0.281805                  0.041265

SkinThickness               0.436783  0.392573                  0.183928

Insulin                     1.000000  0.197859                  0.185071

BMI                         0.197859  1.000000                  0.140647

DiabetesPedigreeFunction    0.185071  0.140647                  1.000000

Age                        -0.042163  0.036242                  0.033561

Outcome                     0.130548  0.292695                  0.173844


                                 Age   Outcome
Pregnancies                 0.544341  0.221898
Glucose                     0.263514  0.466581
BloodPressure               0.239528  0.065068
SkinThickness              -0.113970  0.074752
Insulin                    -0.042163  0.130548
BMI                         0.036242  0.292695
DiabetesPedigreeFunction    0.033561  0.173844
Age                         1.000000  0.238356
Outcome                     0.238356  1.000000
```

```
plt.figure(figsize = (15,15))
sns.heatmap(df.corr(),annot=True)
plt.savefig("correlationcoefficient.jpg")
```



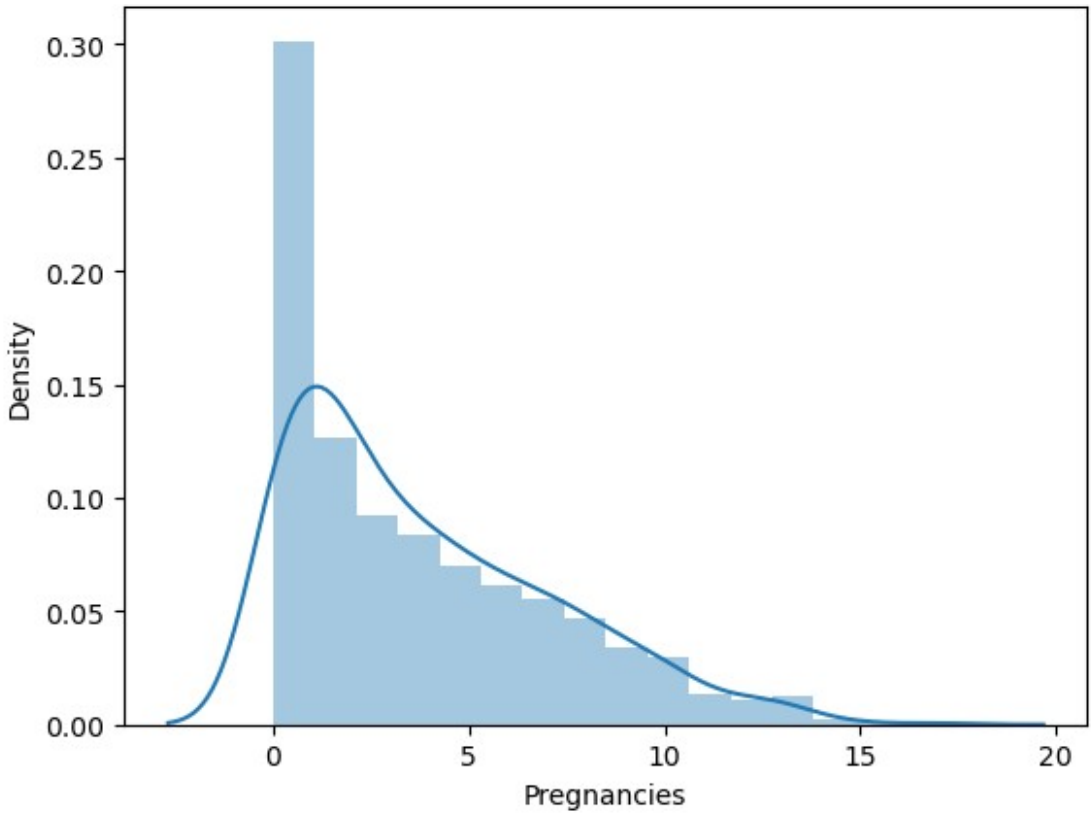

```
df.describe()
```

```
       Pregnancies     Glucose  BloodPressure  SkinThickness
Insulin  \
```

```
count   768.000000  768.000000     768.000000     768.000000
768.000000
mean      3.845052  120.894531      69.105469      20.536458
79.799479
std       3.369578   31.972618      19.355807      15.952218
115.244002
min       0.000000    0.000000       0.000000       0.000000
0.000000
25%       1.000000   99.000000      62.000000       0.000000
0.000000
50%       3.000000  117.000000      72.000000      23.000000
30.500000
75%       6.000000  140.250000      80.000000      32.000000
127.250000
max      17.000000  199.000000     122.000000      99.000000
846.000000

              BMI  DiabetesPedigreeFunction         Age     Outcome
count  768.000000                768.000000  768.000000  768.000000
mean    31.992578                  0.471876   33.240885    0.348958
std      7.884160                  0.331329   11.760232    0.476951
min      0.000000                  0.078000   21.000000    0.000000
25%     27.300000                  0.243750   24.000000    0.000000
50%     32.000000                  0.372500   29.000000    0.000000
75%     36.600000                  0.626250   41.000000    1.000000
max     67.100000                  2.420000   81.000000    1.000000
```

Data Imputations

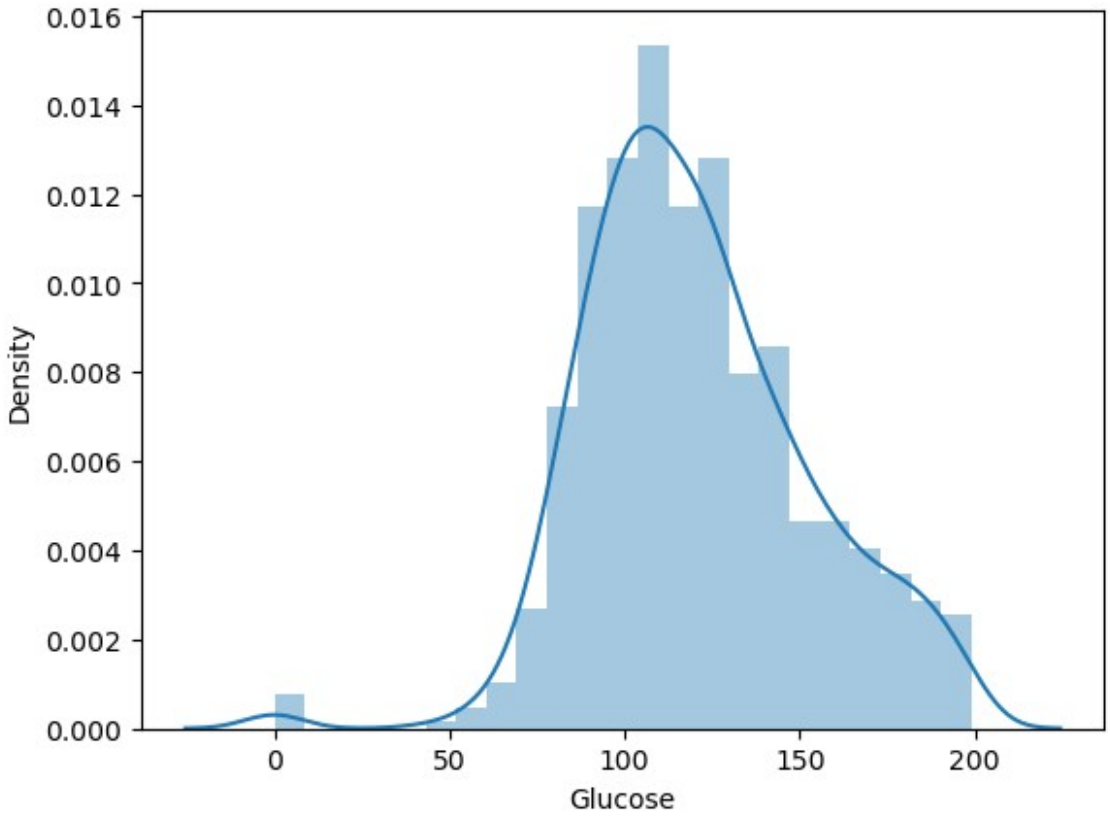
```
sns.distplot(df.Pregnancies)

C:\Users\apurv\AppData\Local\Temp\ipykernel_19356\3462734468.py:1:
UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.Pregnancies)

<Axes: xlabel='Pregnancies', ylabel='Density'>
```




```
sns.distplot(df.Glucose)
```

```
C:\Users\apurv\AppData\Local\Temp\ipykernel_19356\2035962260.py:1:
UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.Glucose)
```

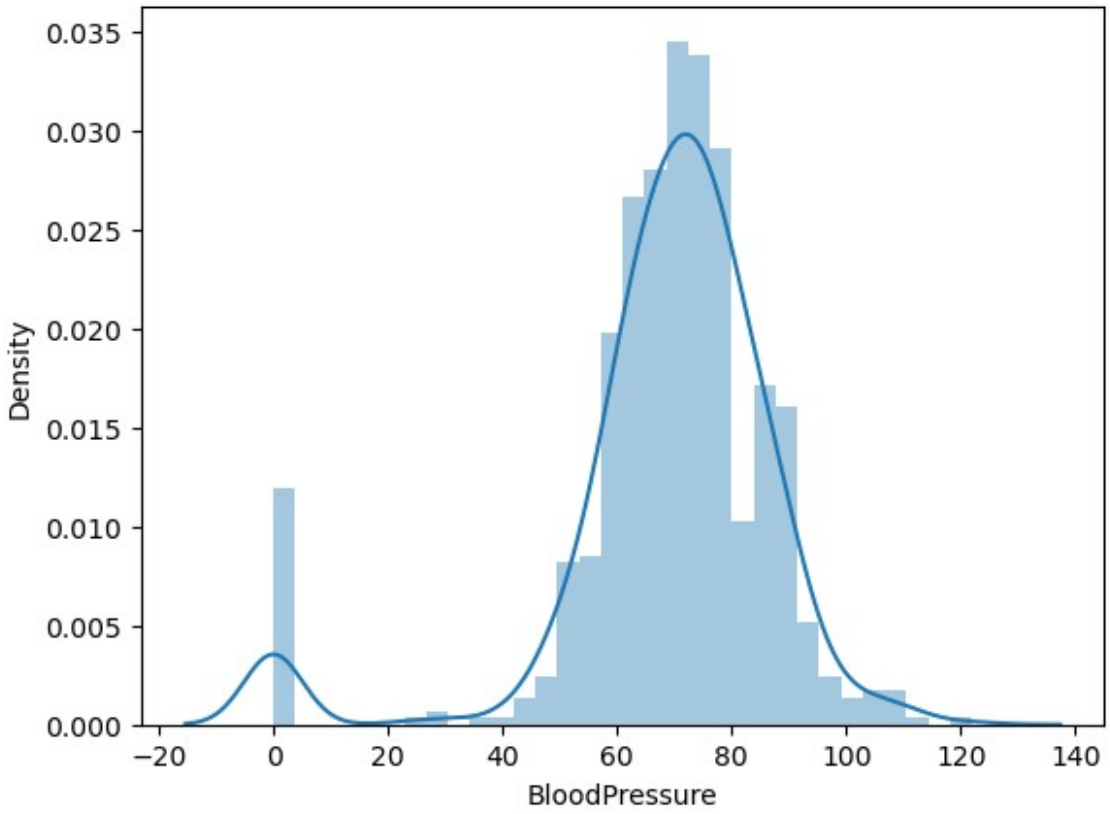
```
<Axes: xlabel='Glucose', ylabel='Density'>
```




```
sns.distplot(df.BloodPressure)
```

```
C:\Users\apurv\AppData\Local\Temp\ipykernel_19356\3755031075.py:1:
UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.BloodPressure)
```

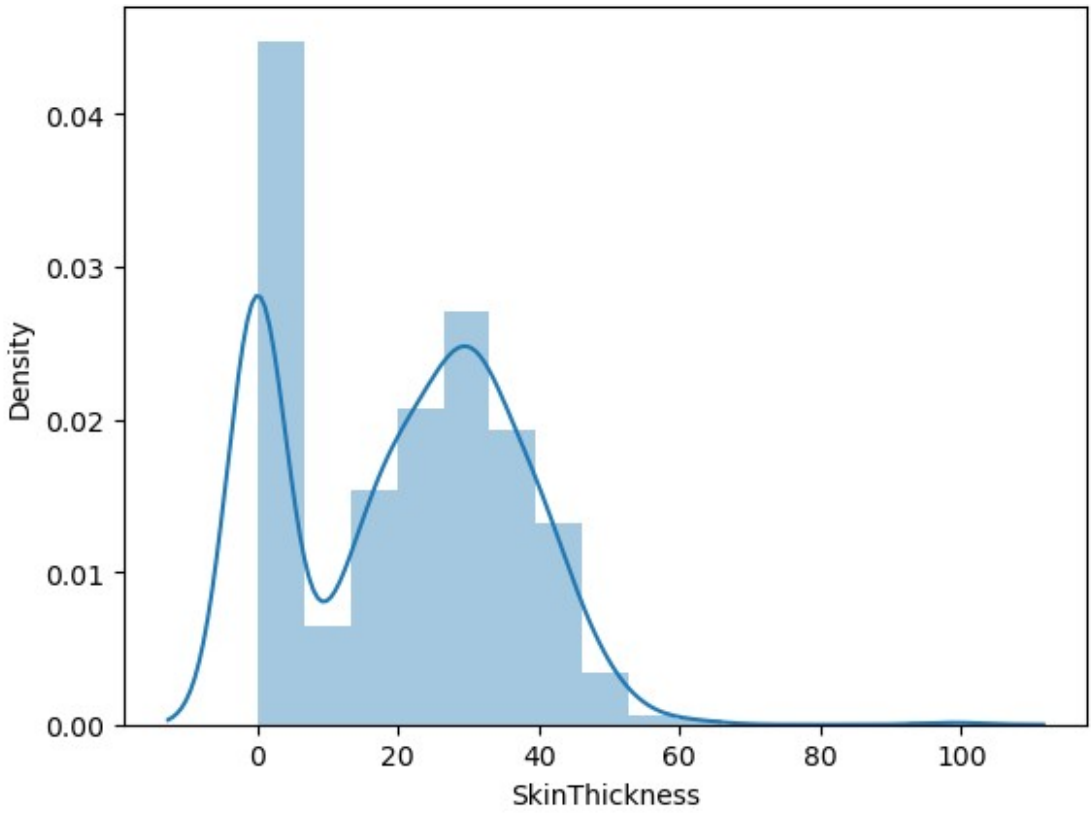
```
<Axes: xlabel='BloodPressure', ylabel='Density'>
```



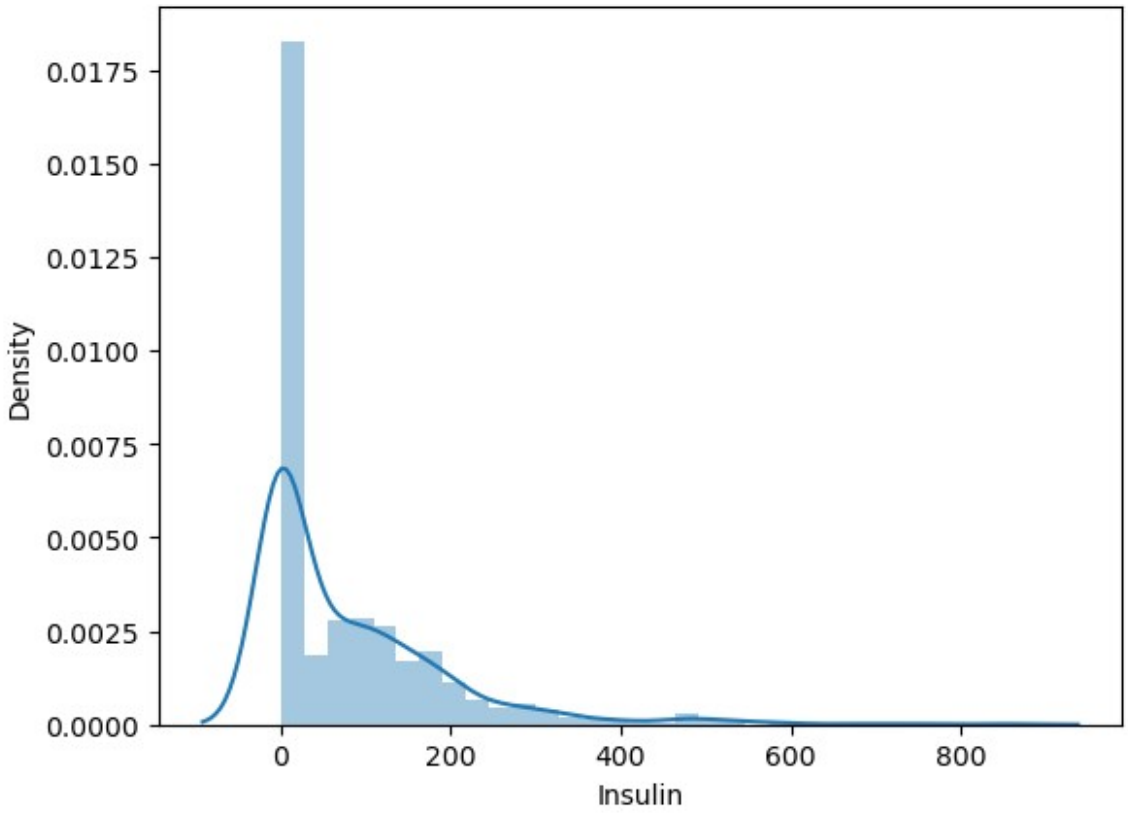

```
sns.distplot(df.SkinThickness)
```

```
C:\Users\apurv\AppData\Local\Temp\ipykernel_19356\1815010915.py:1:
UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.SkinThickness)
```

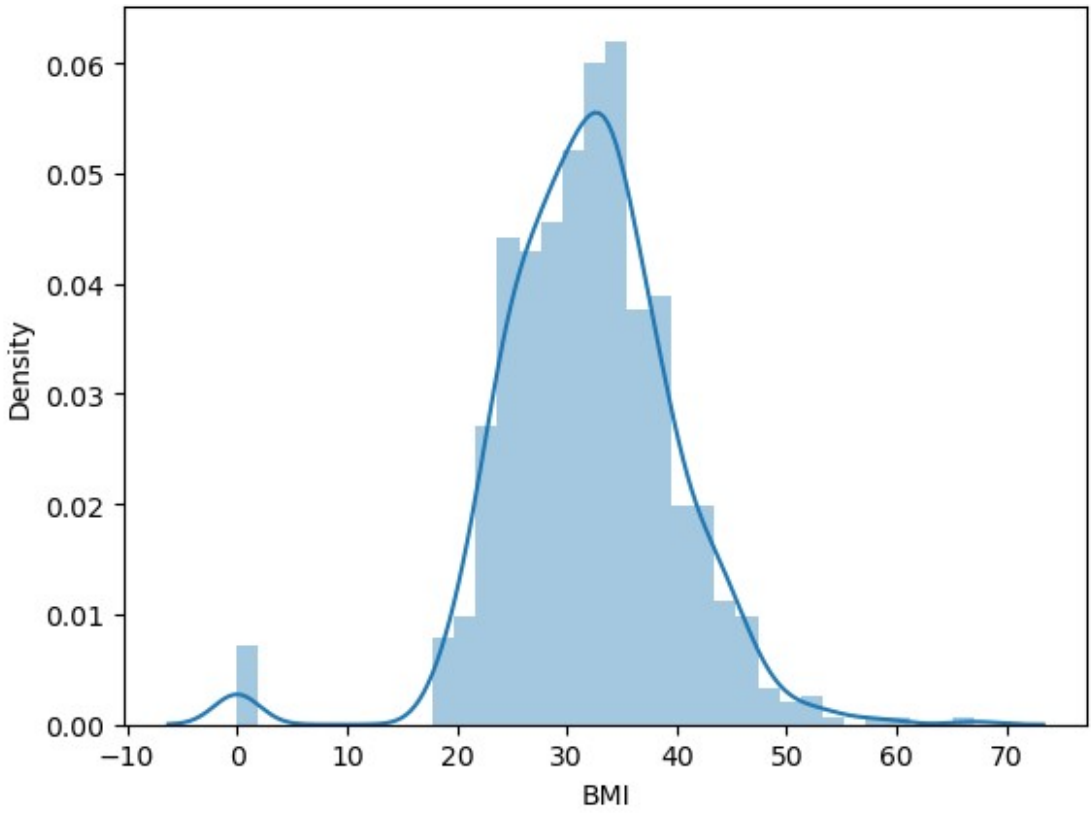
```
<Axes: xlabel='SkinThickness', ylabel='Density'>
```




```
sns.distplot(df.Insulin)

C:\Users\apurv\AppData\Local\Temp\ipykernel_19356\2622307985.py:1:
UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.Insulin)

<Axes: xlabel='Insulin', ylabel='Density'>
```



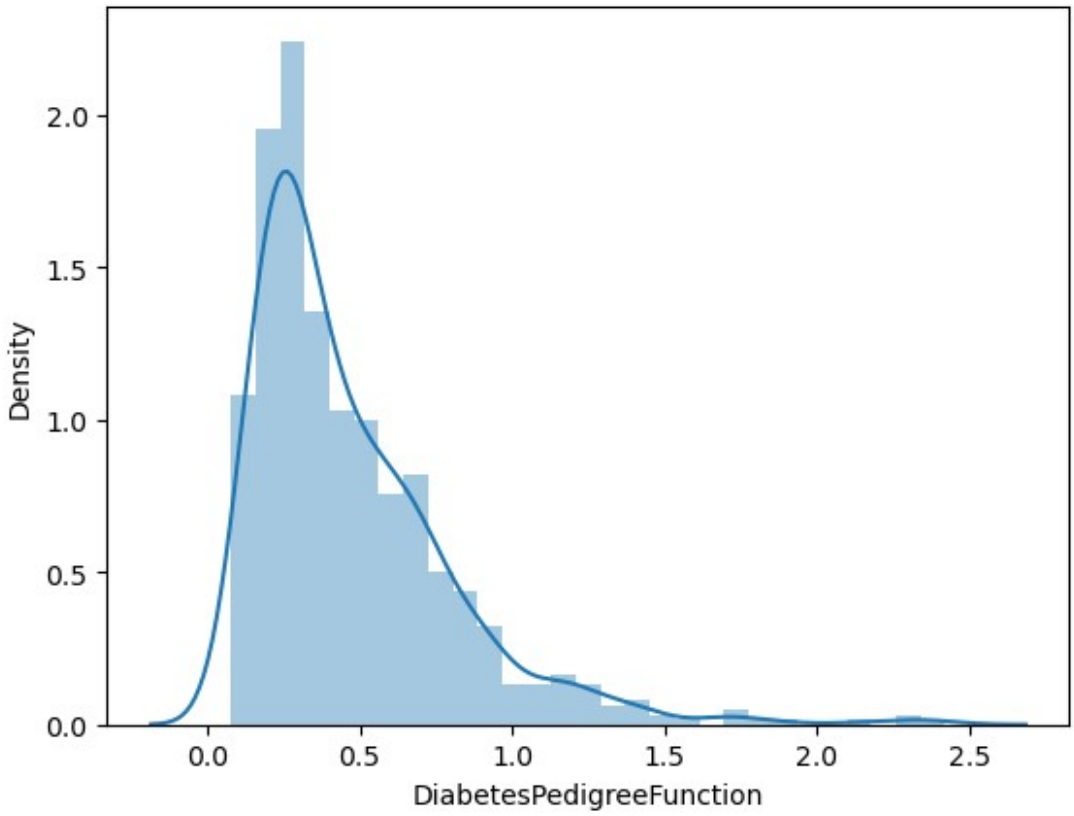

```
sns.distplot(df.BMI)

C:\Users\apurv\AppData\Local\Temp\ipykernel_19356\1689980233.py:1:
UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.BMI)

<Axes: xlabel='BMI', ylabel='Density'>
```




```
sns.distplot(df.DiabetesPedigreeFunction)
```

```
C:\Users\apurv\AppData\Local\Temp\ipykernel_19356\2655324800.py:1:
UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.DiabetesPedigreeFunction)
```

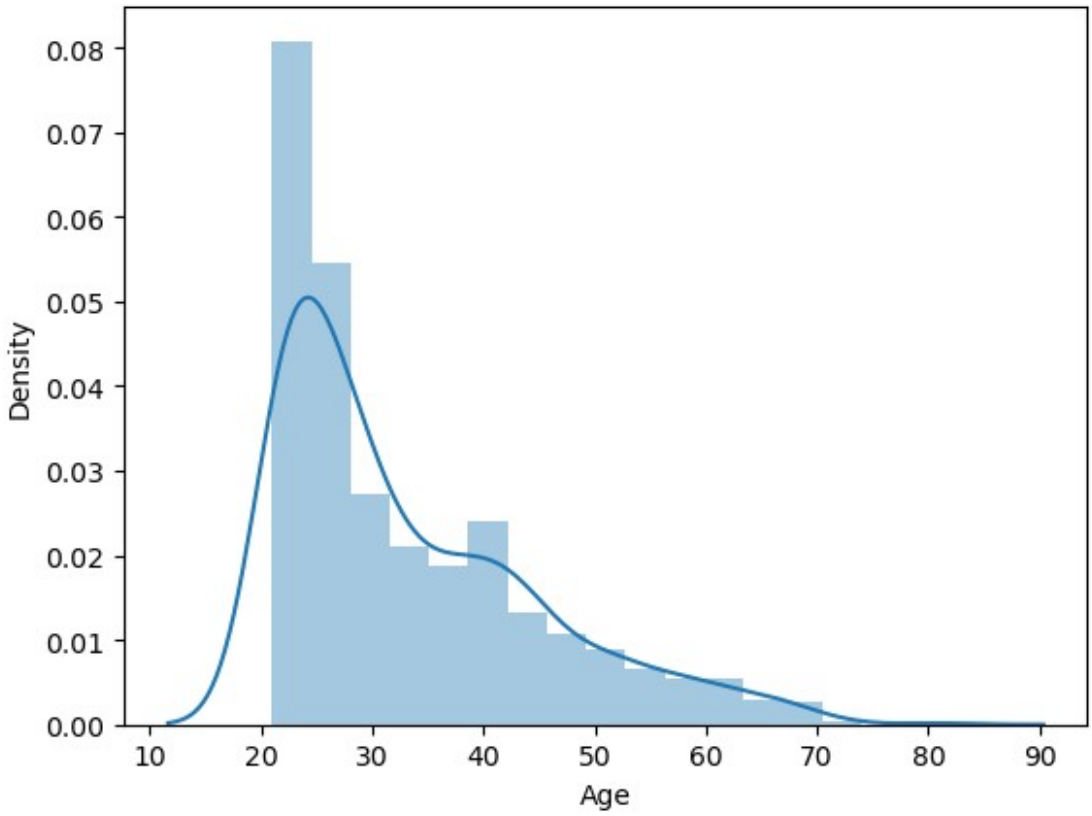
```
<Axes: xlabel='DiabetesPedigreeFunction', ylabel='Density'>
```




```
sns.distplot(df.Age)
```

```
C:\Users\apurv\AppData\Local\Temp\ipykernel_19356\1239919984.py:1:
UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.Age)
```

```
<Axes: xlabel='Age', ylabel='Density'>
```




```
df['Insulin']=df['Insulin'].replace(0,df['Insulin'].median())
df['Pregnancies']=df['Pregnancies'].replace(0,df['Pregnancies'].median
())
df['Glucose']=df['Glucose'].replace(0,df['Glucose'].mean())
df['BloodPressure']=df['BloodPressure'].replace(0,df['BloodPressure'].
mean())
df['SkinThickness']=df['SkinThickness'].replace(0,df['SkinThickness'].
median())
df['BMI']=df['BMI'].replace(0,df['BMI'].mean())
df['DiabetesPedigreeFunction']=df['DiabetesPedigreeFunction'].replace(
0,df['DiabetesPedigreeFunction'].median())
df['Age']=df['Age'].replace(0,df['Age'].median())

df.head(20)
```

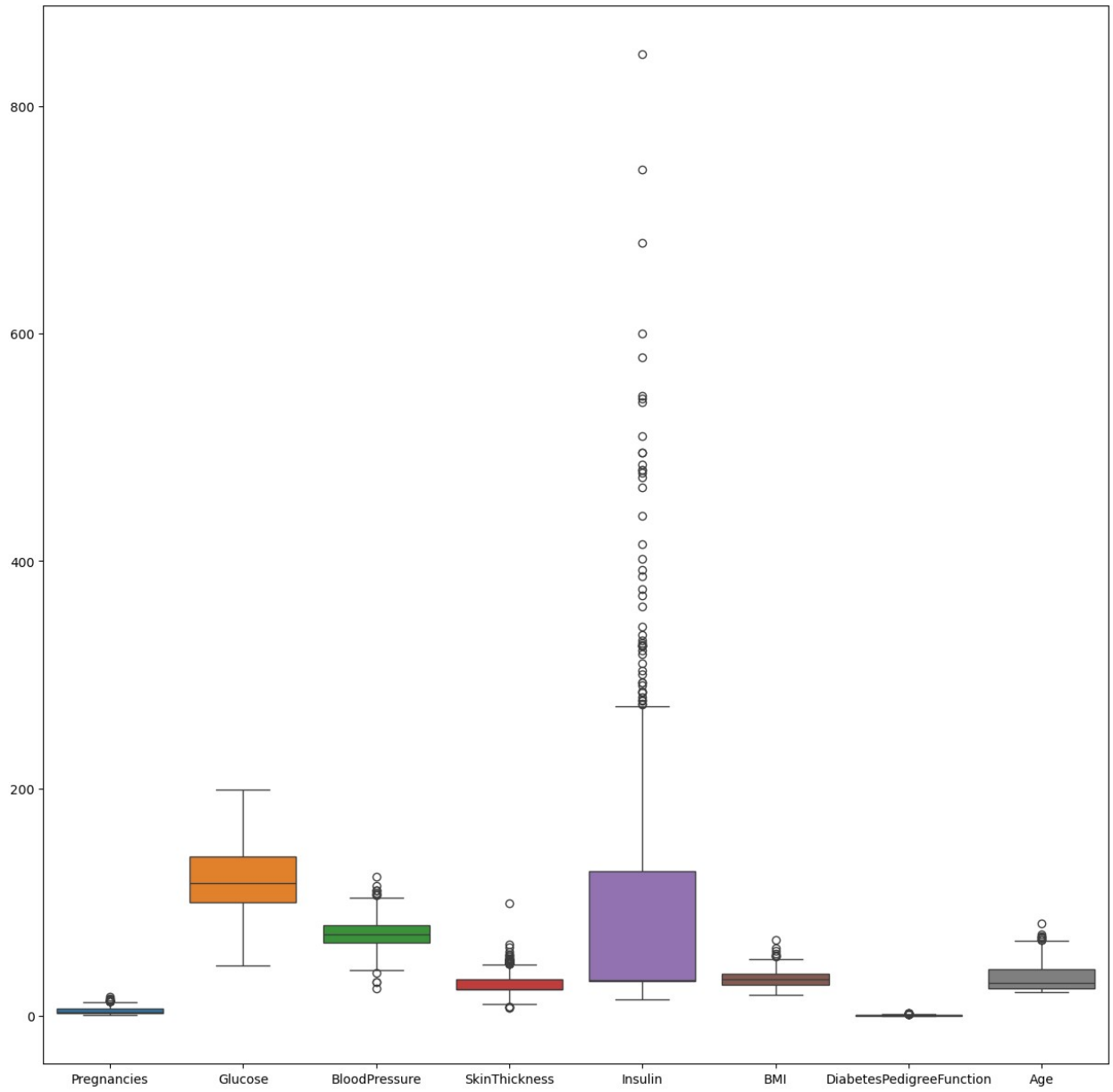
```
    Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin
BMI  \
0             6    148.0      72.000000             35     30.5
33.600000
1             1     85.0      66.000000             29     30.5
26.600000
2             8    183.0      64.000000             23     30.5
23.300000
3             1     89.0      66.000000             23     94.0
```

```
28.100000
4             3    137.0      40.000000             35    168.0
43.100000
5             5    116.0      74.000000             23     30.5
25.600000
6             3     78.0      50.000000             32     88.0
31.000000
7            10    115.0      69.105469             23     30.5
35.300000
8             2    197.0      70.000000             45    543.0
30.500000
9             8    125.0      96.000000             23     30.5
31.992578
10            4    110.0      92.000000             23     30.5
37.600000
11           10    168.0      74.000000             23     30.5
38.000000
12           10    139.0      80.000000             23     30.5
27.100000
13            1    189.0      60.000000             23    846.0
30.100000
14            5    166.0      72.000000             19    175.0
25.800000
15            7    100.0      69.105469             23     30.5
30.000000
16            3    118.0      84.000000             47    230.0
45.800000
17            7    107.0      74.000000             23     30.5
29.600000
18            1    103.0      30.000000             38     83.0
43.300000
19            1    115.0      70.000000             30     96.0
34.600000

    DiabetesPedigreeFunction  Age  Outcome
0                      0.627   50        1
1                      0.351   31        0
2                      0.672   32        1
3                      0.167   21        0
4                      2.288   33        1
5                      0.201   30        0
6                      0.248   26        1
7                      0.134   29        0
8                      0.158   53        1
9                      0.232   54        1
10                     0.191   30        0
11                     0.537   34        1
12                     1.441   57        0
13                     0.398   59        1
```

```
14                       0.587   51        1
15                       0.484   32        1
16                       0.551   31        1
17                       0.254   31        1
18                       0.183   33        0
19                       0.529   32        1
```

Outliers Detection

```
X = df.drop(columns='Outcome',axis=1)
Y = df['Outcome']

fig,ax = plt.subplots(figsize = (15,15))
sns.boxplot(data = X,ax=ax)
plt.savefig('boxplot.jpg')
```




```
cols = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness',
'Insulin','BMI', 'DiabetesPedigreeFunction', 'Age']

for col in cols:
    Q1 = X[col].quantile(0.25)
    Q3 = X[col].quantile(0.75)
    IQR = Q3-Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    mask = (X[col]>=lower_bound) & (X[col]<=upper_bound)

X_after_outlier_detection = X[mask]
Y_after_outlier_detection = Y[mask]
```

```
X_after_outlier_detection.shape
```

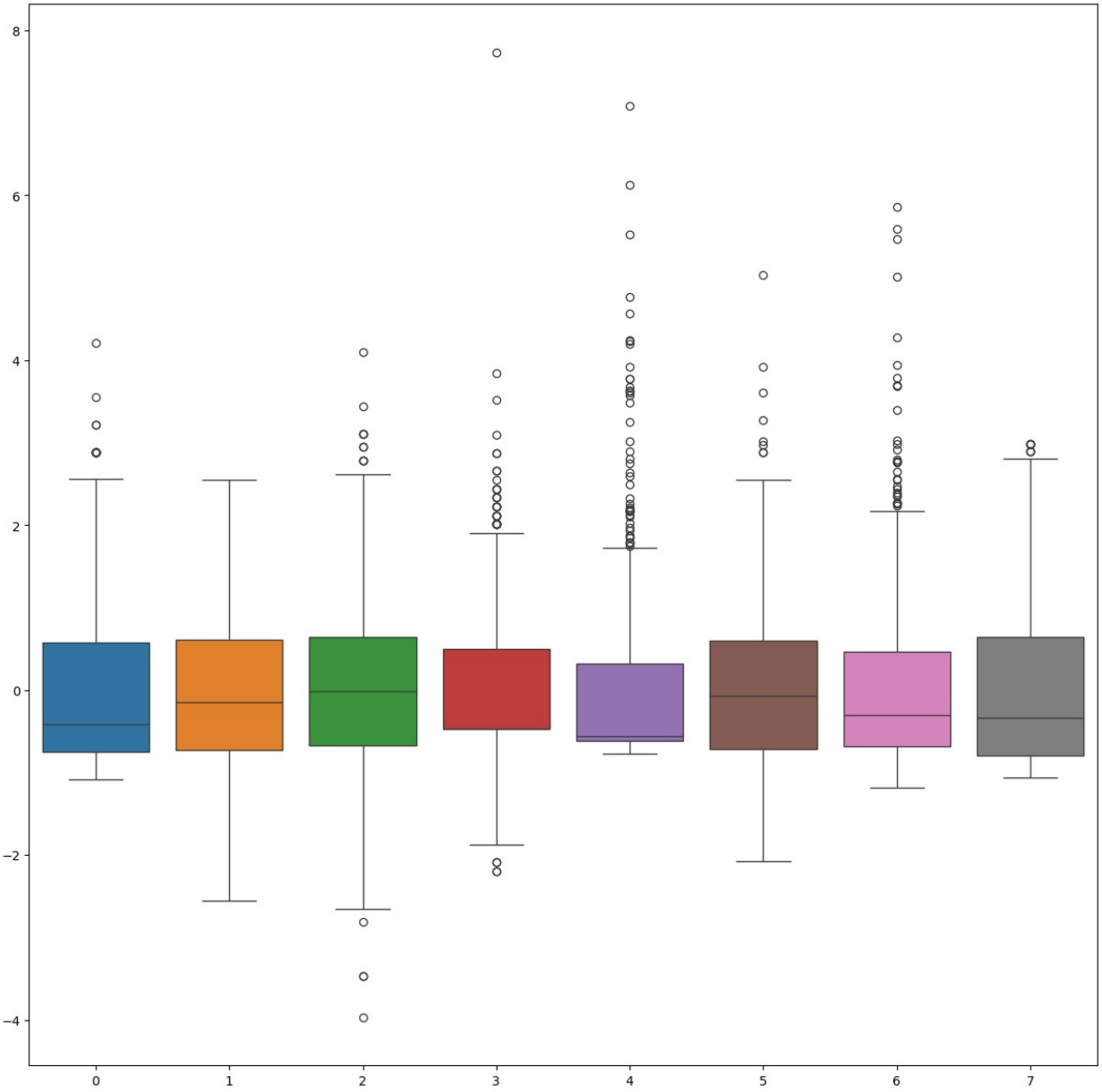
```
(759, 8)
```

```
Y_after_outlier_detection.shape
```

```
(759,)
```

Standardization

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_Scaled = scaler.fit_transform(X_after_outlier_detection)

fig,ax = plt.subplots(figsize = (15,15))
sns.boxplot(data = X_Scaled,ax=ax)
plt.savefig('boxplot.jpg')
```




```
cols = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness',
'Insulin','BMI', 'DiabetesPedigreeFunction', 'Age']

X_Scaled = pd.DataFrame(X_Scaled,columns=cols)
X_Scaled.describe()
```

```
       Pregnancies       Glucose  BloodPressure  SkinThickness
Insulin  \
count  7.590000e+02  7.590000e+02   7.590000e+02   7.590000e+02
7.590000e+02
mean   1.029772e-16 -3.978665e-17  -3.042508e-17  -1.509552e-16 -
4.329724e-17
std    1.000659e+00  1.000659e+00   1.000659e+00   1.000659e+00
```

```
1.000659e+00
min   -1.079800e+00 -2.558042e+00  -3.968588e+00  -2.200901e+00 -
7.684941e-01
25%   -7.491956e-01 -7.286101e-01  -6.755847e-01  -4.729631e-01 -
6.126688e-01
50%   -4.185912e-01 -1.517621e-01  -1.698412e-02  -4.729631e-01 -
5.607270e-01
75%    5.732217e-01  6.063810e-01   6.416165e-01   4.990017e-01
3.222827e-01
max    4.209869e+00  2.551183e+00   4.099270e+00   7.734740e+00
7.088876e+00
```

	BMI	DiabetesPedigreeFunction	Age
count	7.590000e+02	7.590000e+02	7.590000e+02
mean	5.546727e-16	4.914821e-17	1.591466e-16
std	1.000659e+00	1.000659e+00	1.000659e+00
min	-2.081038e+00	-1.183313e+00	-1.062953e+00
25%	-7.125819e-01	-6.852739e-01	-7.928253e-01
50%	-7.202795e-02	-3.045975e-01	-3.426125e-01
75%	5.976421e-01	4.627740e-01	6.478556e-01
max	5.037846e+00	5.864467e+00	2.988962e+00

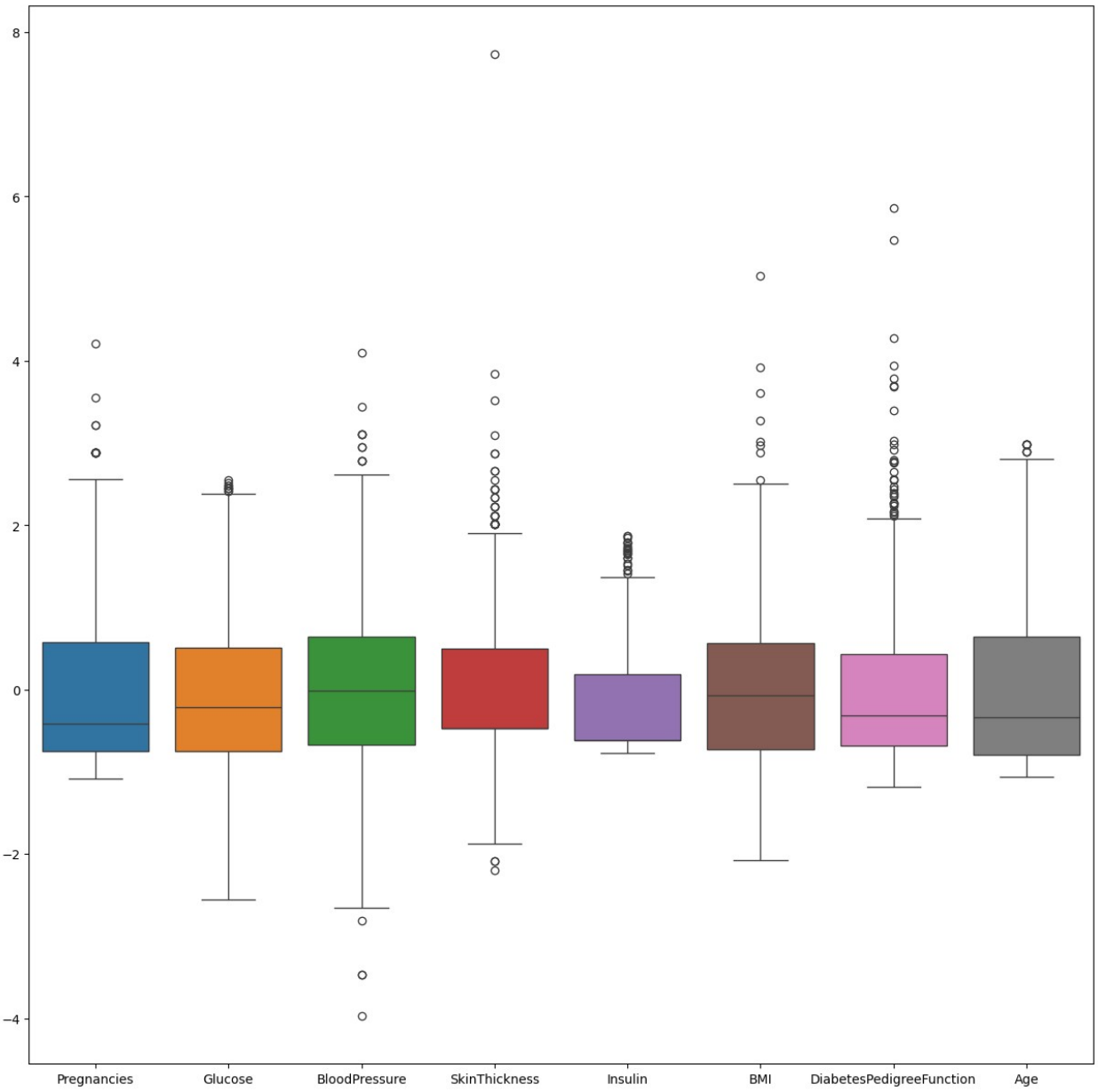
```
X_Scaled.reset_index(drop=True, inplace = True)
Y_after_outlier_detection.reset_index(drop=True,inplace = True)

q = X_Scaled['Insulin'].quantile(0.95)
mask = X_Scaled['Insulin']<q
dataNew = X_Scaled[mask]
Y_after_outlier_detection = Y_after_outlier_detection[mask]

dataNew.shape
```

```
(721, 8)
```

```
Y_after_outlier_detection.shape
```

```
(721,)
```

```
fig,ax = plt.subplots(figsize = (15,15))
sns.boxplot(data = dataNew,ax=ax)
plt.savefig('boxplot.jpg')
```




```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_Train,Y_test =
train_test_split(dataNew,Y_after_outlier_detection,test_size=0.35,rand
om_state=43)

X_train.shape
```

```
(468, 8)
```

```
X_test.shape
```

```
(253, 8)
```

```
Y_Train.value_counts()
```

```
Outcome
0    312
1    156
Name: count, dtype: int64
```

```
from imblearn.over_sampling import SMOTE
smote = SMOTE(random_state=42)
X_trained_resample, Y_trained_resample =
smote.fit_resample(X_train,Y_Train)
print(pd.Series(Y_trained_resample).value_counts())
```

```
Outcome
0    312
1    312
Name: count, dtype: int64
```

```
from sklearn.linear_model import LogisticRegression
Classification = LogisticRegression()
Classification.fit(X_trained_resample,Y_trained_resample)
```

```
LogisticRegression()
```

```
Y_prediction = Classification.predict(X_test)
Y_prediction
```

```
array([0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1,
0,
       0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0,
1,
       1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1,
0,
       0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0,
1,
       0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0,
0,
       1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0,
1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0,
1,
       0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
0,
       0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0,
0,
       0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1,
1,
       0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0,
1,
       1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0], dtype=int64)
```

```
from sklearn.metrics import accuracy_score
print(accuracy_score(Y_test,Y_prediction))
```

```
0.7786561264822134
```

```
from sklearn.metrics import classification_report
target_names = ['Non-Diabetic','Diabetic']
print(classification_report(Y_test,Y_prediction,target_names =
target_names))
```

```
              precision    recall  f1-score   support

Non-Diabetic       0.84      0.81      0.83       165
    Diabetic       0.67      0.72      0.69        88

    accuracy                           0.78       253
   macro avg       0.76      0.76      0.76       253
weighted avg       0.78      0.78      0.78       253
```

```
import pickle
pickle.dump(Classification,open("Classification_model.pkl","wb"))
```