DIABETES DATASET

In [42]:
```
import pandas as pd
import numpy as np
```

The datasets consists of several medical predictor variables and one target variable, Outcome.

Pregnancies :- Number of times pregnant

Glucose:- Plasma glucose concentration a 2 hours in an oral glucose tolerance test

BloodPressure:- Diastolic blood pressure

SkinThickness:- Triceps skin fold thickness

Insulin:- 2-Hour serum insulin

BMI:- Body mass index

DiabetesPedigreeFunction:- Diabetes pedigree function

Age:-Age in years

Outcome:- Class variable (0 or 1)

In [43]:
```
my_data=pd.read_csv('diabetes.csv')
```

In [44]:
```
my_data
```

Out[44]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Ag
0	6	148	72	35	0	33.6	0.627	5
1	1	85	66	29	0	26.6	0.351	3
2	8	183	64	0	0	23.3	0.672	3
3	1	89	66	23	94	28.1	0.167	2
4	0	137	40	35	168	43.1	2.288	3
...	...	...	...	...	...	...	...	.
763	10	101	76	48	180	32.9	0.171	6
764	2	122	70	27	0	36.8	0.340	2
765	5	121	72	23	112	26.2	0.245	3
766	1	126	60	0	0	30.1	0.349	4
767	1	93	70	31	0	30.4	0.315	2

768 rows × 9 columns

In [45]:
```
my_data.shape
```

Out[45]:
```
(768, 9)
```

In [46]:
```
types=my_data.dtypes
types
```

Out[46]:
```
Pregnancies                 int64
Glucose                     int64
BloodPressure               int64
SkinThickness               int64
Insulin                     int64
BMI                       float64
DiabetesPedigreeFunction  float64
Age                         int64
Outcome                     int64
dtype: object
```

In [47]:
```
#coloumns
my_data.columns
```

Out[47]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [48]:
```
#Top 5 rows in dataset
my_data.head()
```

Out[48]:

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

MISSING VALUES

In [49]:
```
my_data.isnull().sum()
```

Out[49]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [50]:
```
my_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [51]:
```
#describing the data
my_data.describe()
```

Out[51]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigr
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	

In [52]:
```
my_data.groupby('Outcome').size()
```

Out[52]:
```
Outcome
0    500
1    268
dtype: int64
```

In [53]:
```
#Mean
my_data.groupby('Outcome').mean()
```

Out[53]:

Outcome	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedi
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	

In [54]:
```
#Median
my_data.groupby('Outcome').median()
```

Out[54]:

Outcome	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunctio
0	2.0	107.0	70.0	21.0	39.0	30.05	0.33
1	4.0	140.0	74.0	27.0	0.0	34.25	0.44

In [55]:
```
#Standard deviation
my_data.groupby('Outcome').std()
```

Out[55]:

Outcome	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigr
0	3.017185	26.141200	18.063075	14.889947	98.865289	7.689855	
1	3.741239	31.939622	21.491812	17.679711	138.689125	7.262967	

In [56]:
```
#skew calculation
my_data.groupby('Outcome').skew()
```

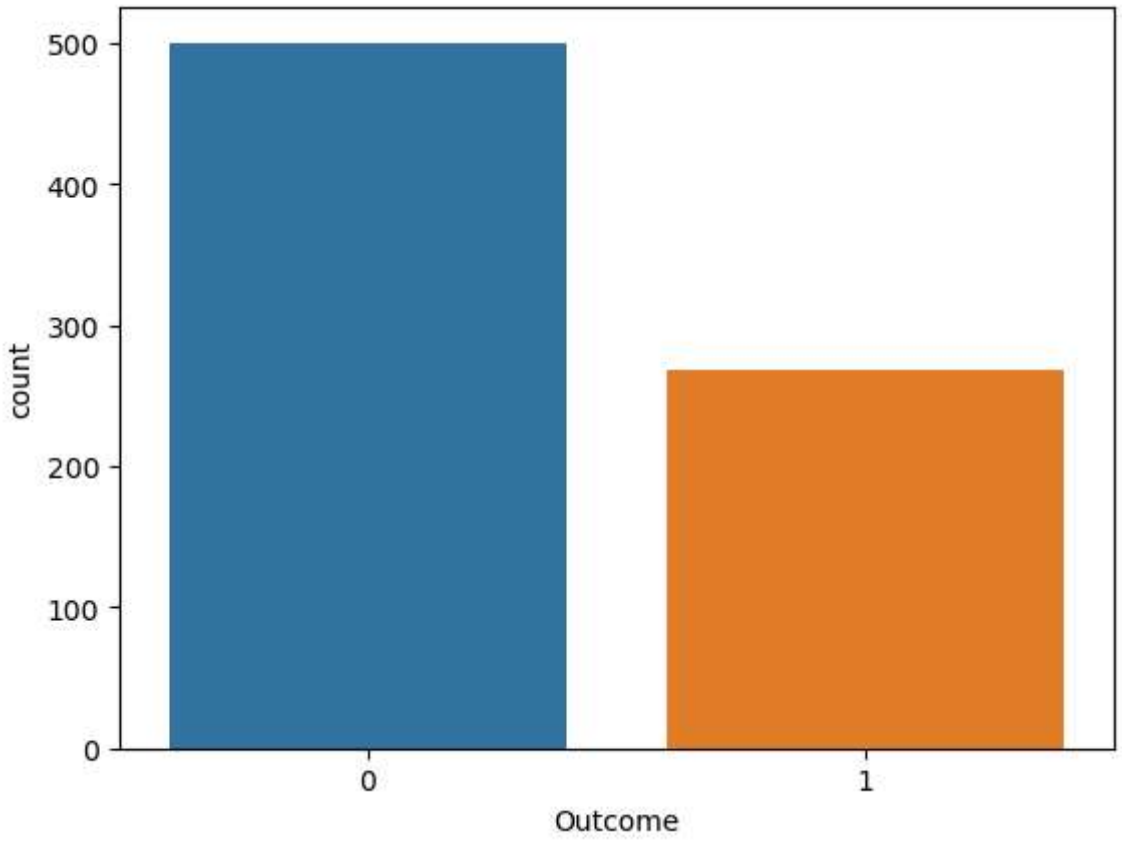
Out[56]:

Outcome	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigree
0	1.114105	0.173111	-1.809825	0.031155	2.498741	-0.665902	
1	0.503749	-0.495557	-1.943633	0.115910	1.843831	0.000597	

In [57]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [58]:
```
import seaborn as sns
sns.countplot(my_data['Outcome'],label="count")
```

Out[58]:
```
<AxesSubplot:xlabel='Outcome', ylabel='count'>
```



In [59]:
```
#correlation of the data
corr = my_data.corr()
```

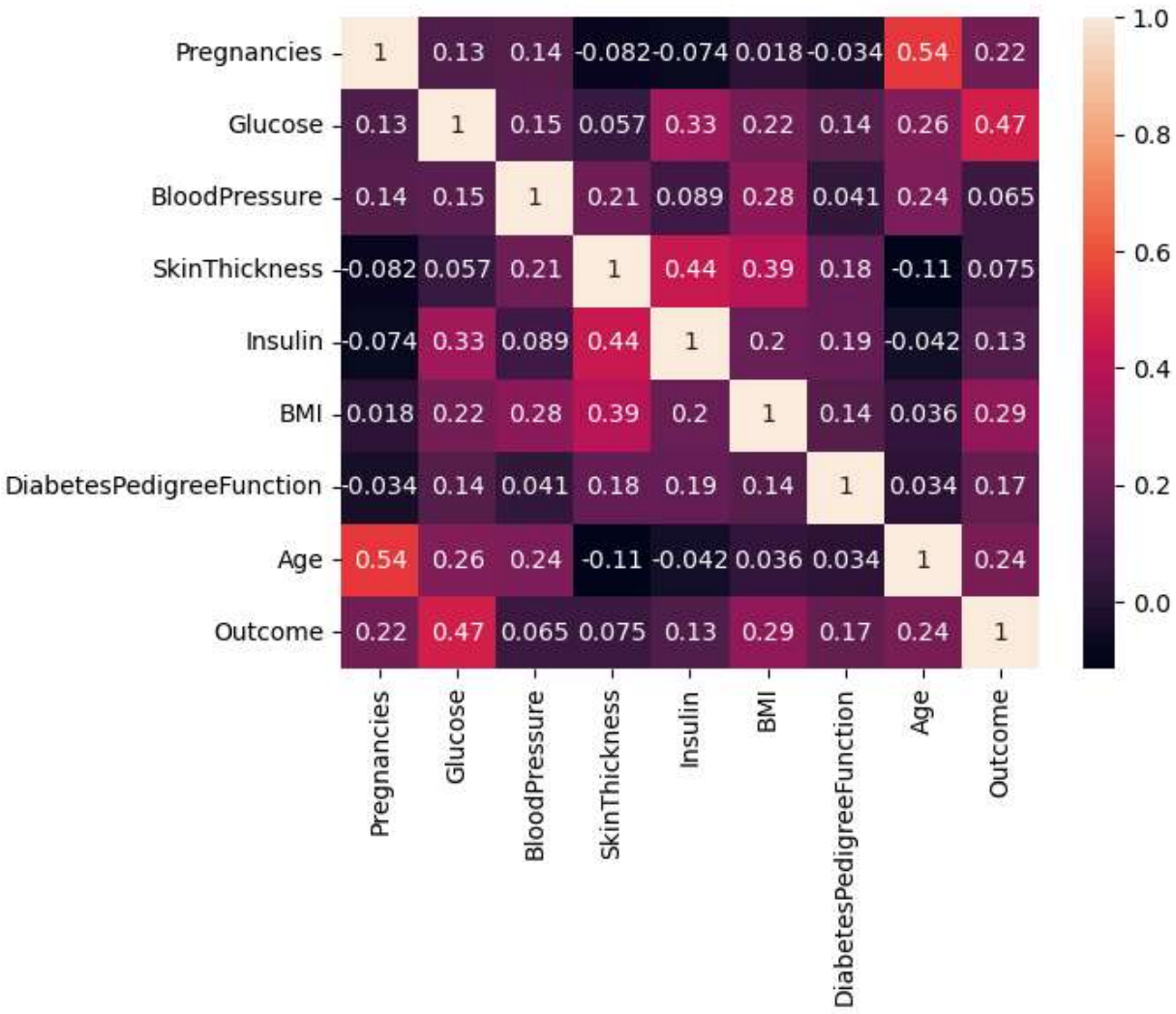
```
corr
```

Out[59]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	D
Pregnancies	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	
Glucose	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	
BloodPressure	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	
SkinThickness	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	
Insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	
BMI	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	
DiabetesPedigreeFunction	-0.033523	0.137337	0.041265	0.183928	0.185071	0.140647	
Age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	
Outcome	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	

In [60]:
```
sns.heatmap(corr,annot=True)
```

Out[60]:
```
<AxesSubplot:>
```



In [61]:
```
#Blood pressure : By observing the data we can see that there are 0 values for blood p
# And it is evident that the readings of the data set seems wrong because a living per
# cannot have diastolic blood pressure of zero.
print("Total: ",my_data[my_data.BloodPressure == 0].shape[0])
print(my_data[my_data.BloodPressure == 0].groupby('Outcome')['Age'].count())
```

```
Total:  35
Outcome
0    19
1    16
Name: Age, dtype: int64
```

In [62]:
```
#Insulin : In a rare situation a person can have zero insulin
print("Total: ",my_data[my_data.Insulin == 0].shape[0])
print(my_data[my_data.Insulin == 0].groupby('Outcome')['Age'].count())
```

```
Total:  374
Outcome
0    236
1    138
Name: Age, dtype: int64
```

In [63]:
```
# Skin Fold Thickness : For normal people skin fold thickness can’t be less than 10 mm
print("Total: ",my_data[my_data.SkinThickness == 0].shape[0])
print(my_data[my_data.SkinThickness == 0].groupby('Outcome')['Age'].count())
```

```
Total:  227
Outcome
0    139
1     88
Name: Age, dtype: int64
```

In [64]:
```
#BMI : Should not be 0 or close to zero unless the person is really underweight which
print("Total: ",my_data[my_data.BMI == 0].shape[0])
print(my_data[my_data.BMI == 0].groupby('Outcome')['Age'].count())
```

```
Total:  11
Outcome
0    9
1    2
Name: Age, dtype: int64
```

In [65]:
```
# Plasma glucose levels : Even after fasting glucose level would not be as low as zero
print("Total: ",my_data[my_data.Glucose == 0].shape[0])
print(my_data[my_data.Glucose == 0].groupby('Outcome')['Age'].count())
```

```
Total:  5
Outcome
0    3
1    2
Name: Age, dtype: int64
```

HANDLING INVALID DATA VALUES:

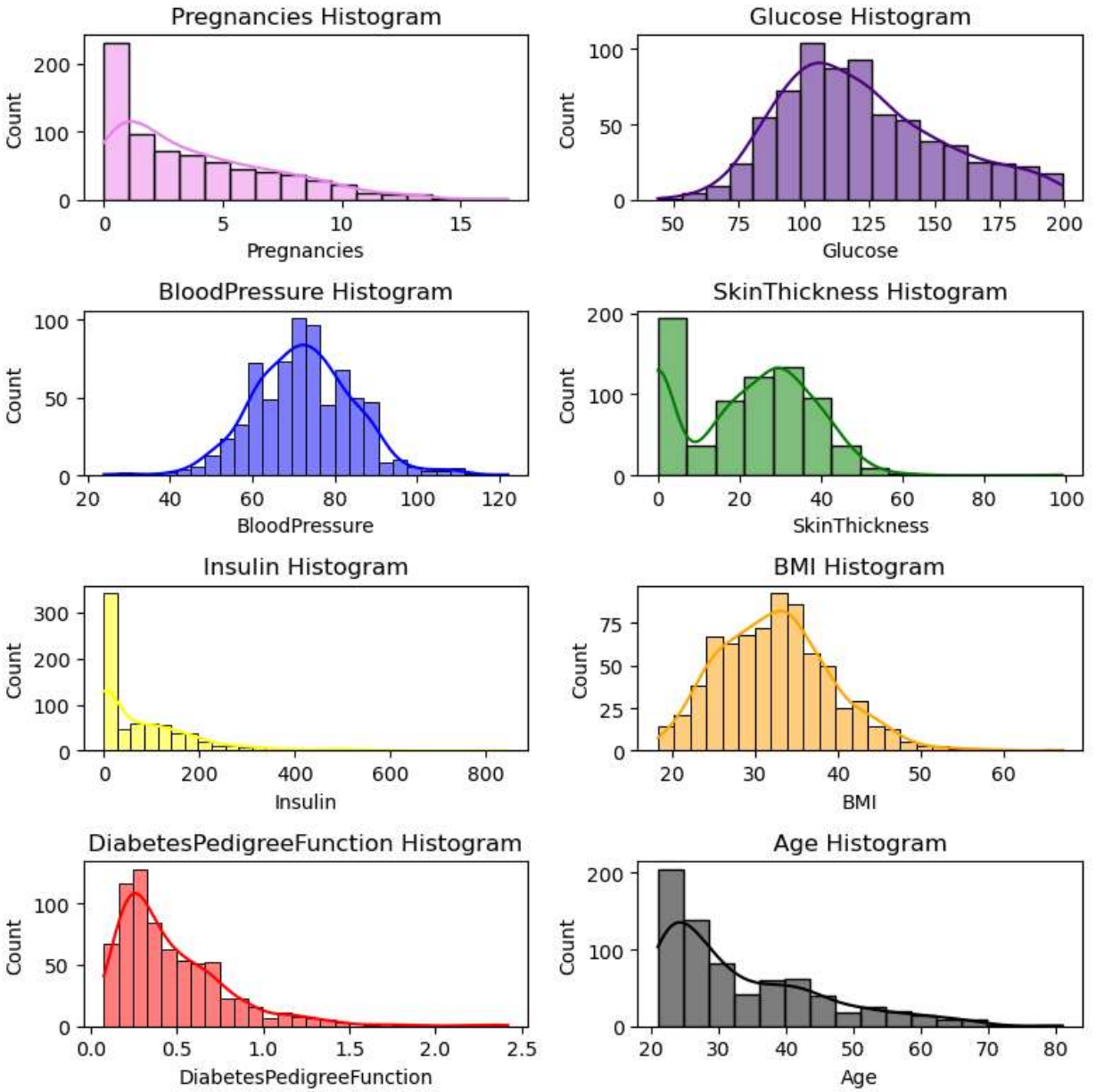
In [66]:
```
#remove the rows which the “BloodPressure”, “BMI” and “Glucose” are zero.

my_data=my_data[(my_data.BloodPressure !=0) & (my_data.BMI !=0) & (my_data.Glucose !=0
print(my_data.shape)
```

```
(724, 9)
```

In [67]:
```
from matplotlib import pyplot
import matplotlib.pyplot as plt
```

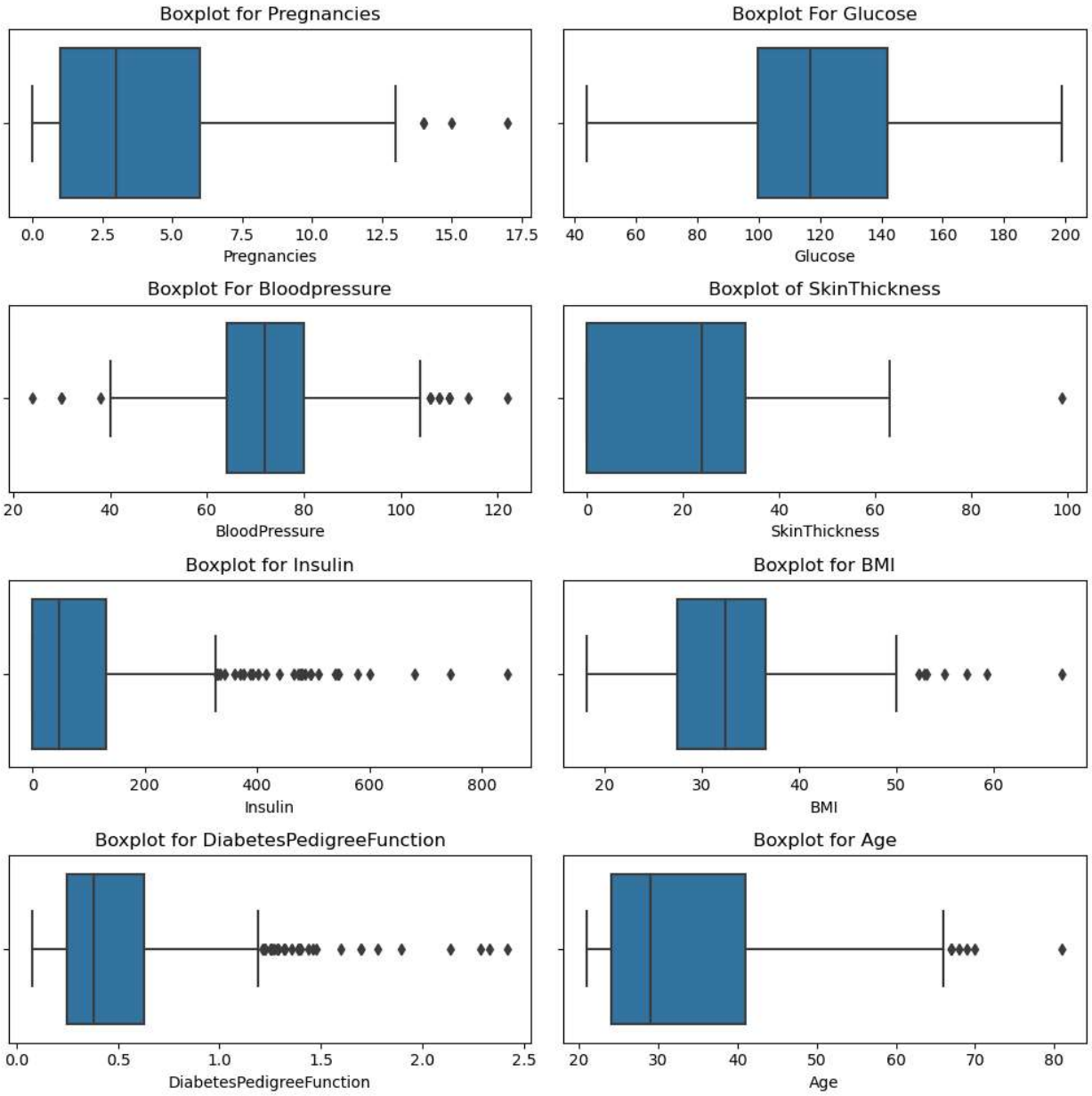
HISTOGRAM:

In [68]:
```
fig, axes =plt.subplots(4,2,figsize=(8,8))

sns.histplot(data=my_data["Pregnancies"],kde=True,ax=axes[0,0],color='violet').set(tit
sns.histplot(data=my_data["Glucose"],kde=True,ax=axes[0,1],color='indigo').set(title='
sns.histplot(data=my_data["BloodPressure"],kde=True,ax=axes[1,0],color='blue').set(tit
sns.histplot(data=my_data["SkinThickness"],kde=True,ax=axes[1,1],color='green').set(ti
sns.histplot(data=my_data["Insulin"],kde=True,ax=axes[2,0],color='yellow').set(title='
sns.histplot(data=my_data["BMI"],kde=True,ax=axes[2,1],color='orange').set(title='BMI
sns.histplot(data=my_data["DiabetesPedigreeFunction"],kde=True,ax=axes[3,0],color='red
sns.histplot(data=my_data["Age"],kde=True,ax=axes[3,1],color='black').set(title='Age H
plt.tight_layout()
plt.show()
```



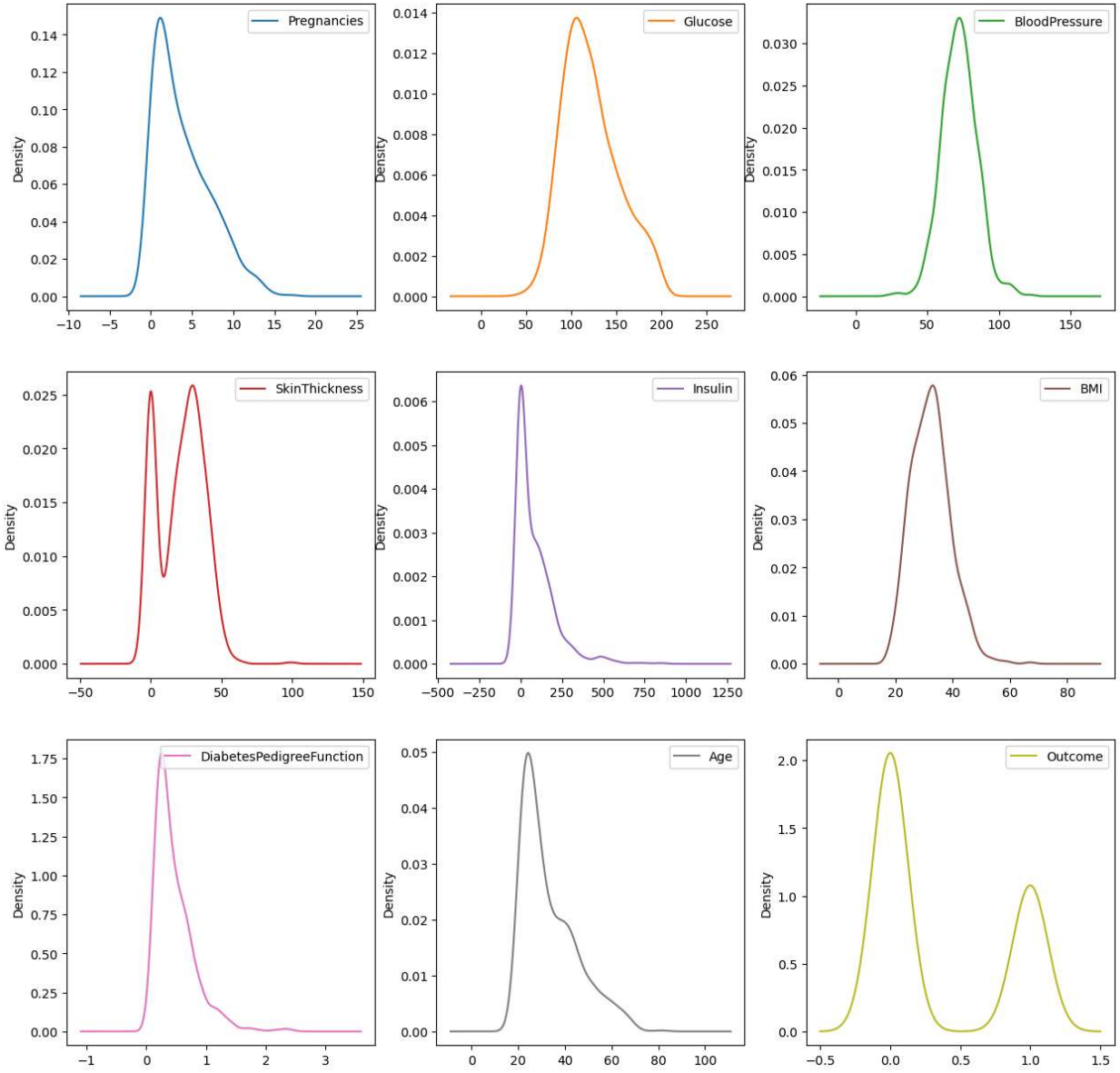
BOXPLOT:

In [69]:
```
fig, axes=plt.subplots(4,2,figsize=(10,10))

sns.boxplot(x=my_data['Pregnancies'], ax=axes[0,0]).set(title='Boxplot for Pregnancies
sns.boxplot(x=my_data['Glucose'],ax=axes[0,1]).set(title='Boxplot For Glucose')
sns.boxplot(x=my_data['BloodPressure'],ax=axes[1,0]).set(title='Boxplot For Bloodpress
sns.boxplot(x=my_data['SkinThickness'],ax=axes[1,1]).set(title='Boxplot of SkinThickne
sns.boxplot(x=my_data['Insulin'],ax=axes[2,0]).set(title='Boxplot for Insulin')
sns.boxplot(x=my_data['BMI'],ax=axes[2,1]).set(title='Boxplot for BMI')
sns.boxplot(x=my_data['DiabetesPedigreeFunction'],ax=axes[3,0]).set(title='Boxplot for
sns.boxplot(x=my_data['Age'],ax=axes[3,1]).set(title='Boxplot for Age')
plt.tight_layout()
plt.show()
```



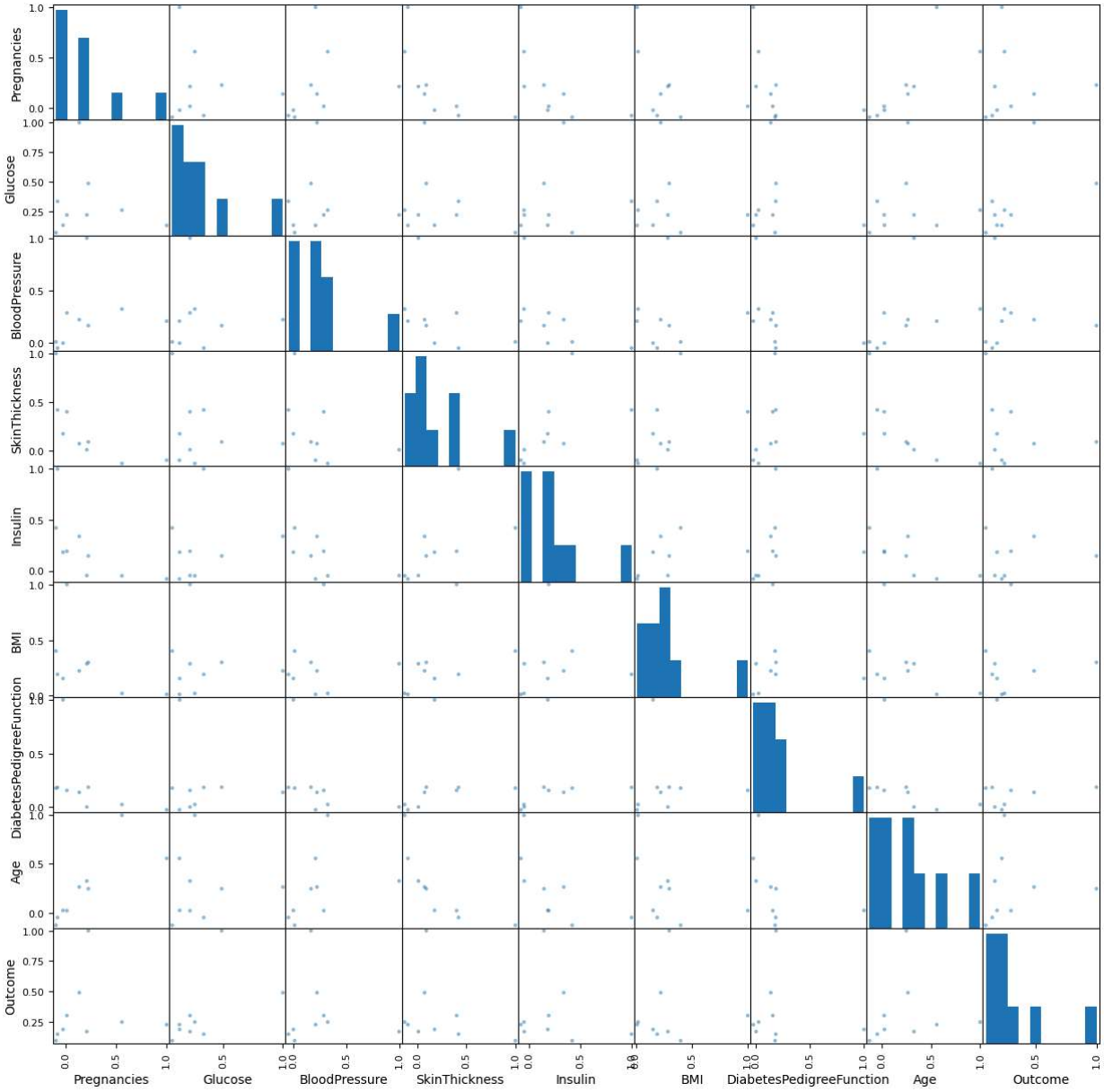
DENSITY PLOT

In [70]:
```
my_data.plot(kind='density',subplots=True,layout=(3,3),sharex=False,figsize=(15,15))
pyplot.show()
```



SCATTER_MATRIX:

In [71]:
```
import pandas
from pandas.plotting import scatter_matrix

dataCorr = my_data.corr()
pandas.plotting.scatter_matrix(dataCorr,figsize=(15,15))
pyplot.show()
```


Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
BMI
DiabetesPedigreeFunction
Age
Outcome