diabetes-prediction-system

December 29, 2023

[1]:
```
#import
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[10]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import␣
 ↪accuracy_score,classification_report,confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier,AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
```

[3]:
```
data=pd.read_csv("E:\diabetes.csv")
data
```

[3]:

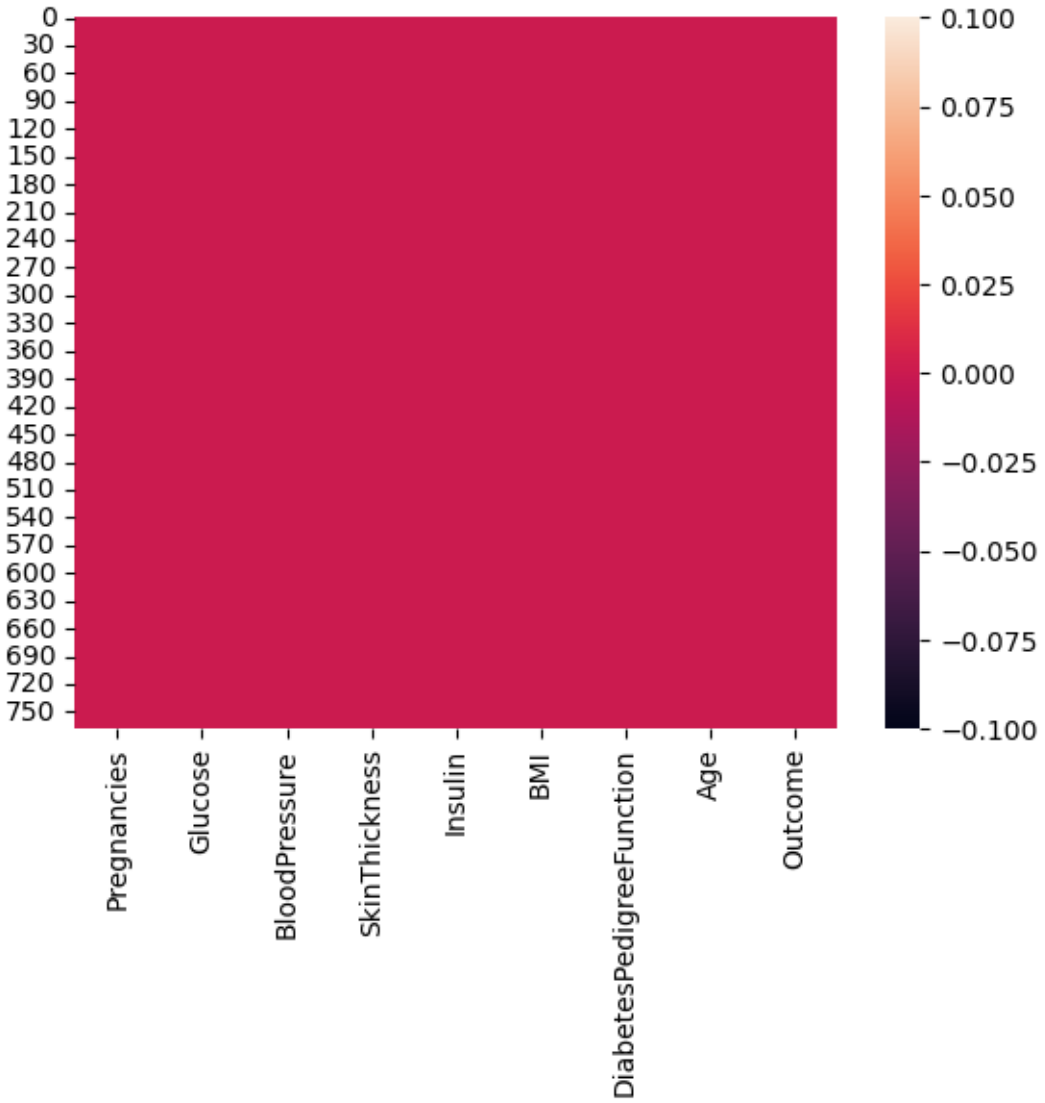
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	\
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
..	…	…	…	…	…	…	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31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

	DiabetesPedigreeFunction	Age	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

```
..                          …   …       …
763                     0.171   63      0
764                     0.340   27      0
765                     0.245   30      0
766                     0.349   47      1
767                     0.315   23      0

[768 rows x 9 columns]
```

[4]:
```
sns.heatmap(data.isnull())
```

[4]: <Axes: >

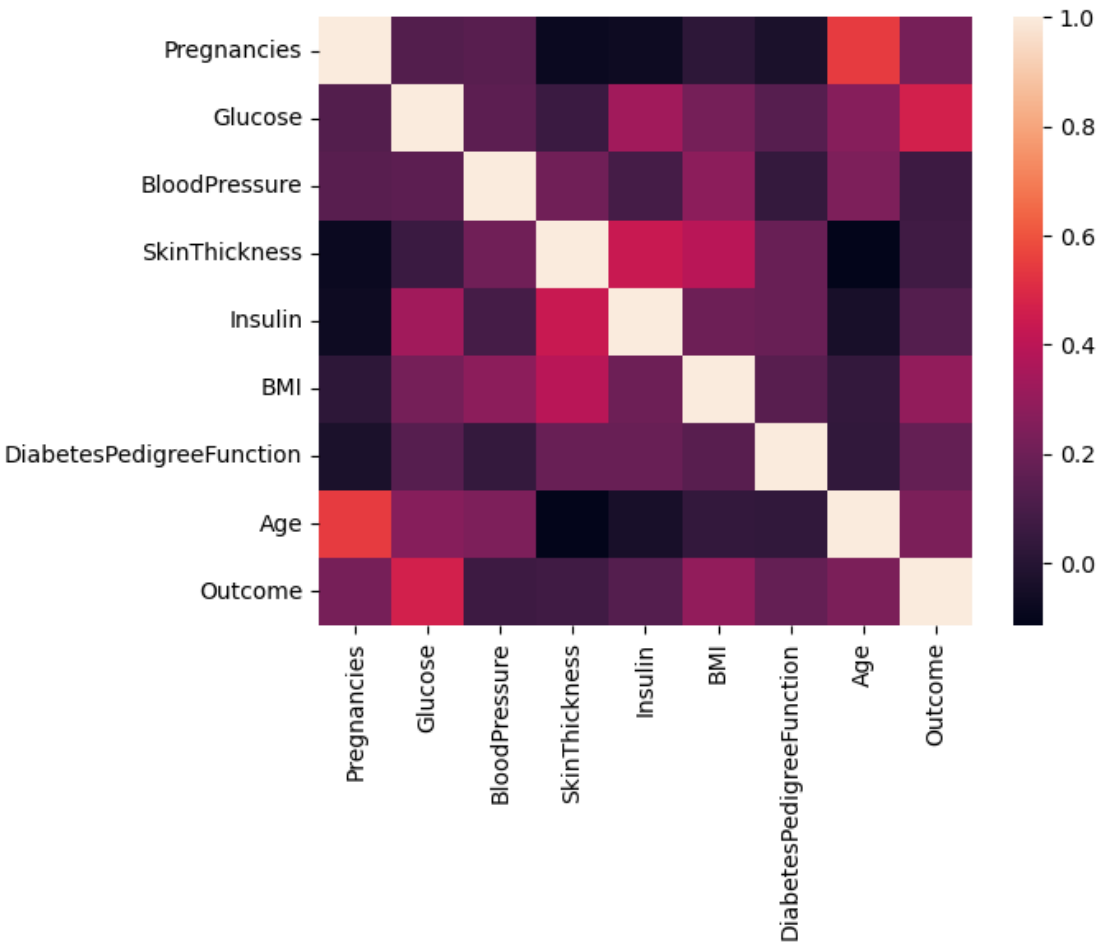


[5]:
```
correlation=data.corr()
print(correlation)
```

```
                          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
```

[6]:
```
sns.heatmap(data.corr())
```

[6]: <Axes: >



[11]:
```
#standardize the feature
scaler=StandardScaler()
X_train_scaled=scaler.fit_transform(X_train)
X_test_scaled=scaler.transform(X_test)
```

[8]:
```
#preparing the data
X=data.drop("Outcome",axis=1)
Y=data['Outcome']
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.
↪2,random_state=42)
```

[12]:
```
#train a random classifier
model=RandomForestClassifier(random_state=42)
model.fit(X_train_scaled,Y_train)
```

[12]:
```
RandomForestClassifier(random_state=42)
```

[22]:
```
#make prediction on the test set
prediction=model.predict(X_test_scaled)
print(prediction)
```

```
[0 0 0 0 0 1 0 1 1 1 0 1 0 0 0 0 0 0 1 1 0 0 0 0 1 1 0 0 0 0 1 1 1 1 1 1 1
 0 0 1 0 0 1 0 0 1 0 0 0 1 0 1 1 0 0 0 1 0 0 1 1 0 0 0 0 1 0 0 0 1 1 0 0 0
 0 1 0 0 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 0 1 0 1 1 0 1 0 1 0 0 1 1 0 0 1 0 1 0
 1 0 1 0 1 1 0 0 1 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0
 0 1 0 0 0 0]
```

[23]:
```
#evaluate the model
accuracy=accuracy_score(Y_test,prediction)
print(accuracy)
```

```
0.7207792207792207
```

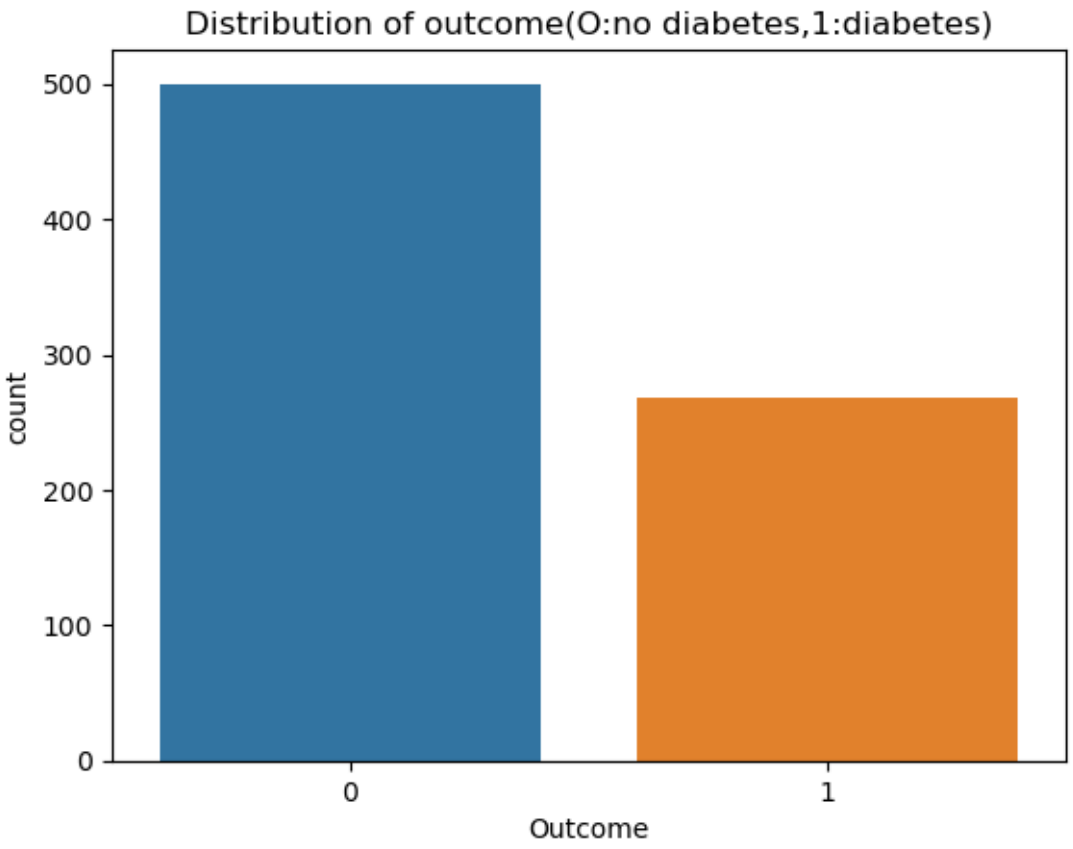
[24]:
```
#display the first few rows
print(data.head())
#checking for missing values
print(data.isnull().sum())
#statistical summary
print(data.describe())
```

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

[25]:
```
#distribution of target variable
sns.countplot(x='Outcome',data=data)
plt.title('Distribution of outcome(O:no diabetes,1:diabetes)')
plt.show()
```



[27]:
```
#additonal evaluation metrics
print('\n classification report:')
print(classification_report(Y_test,prediction))
```

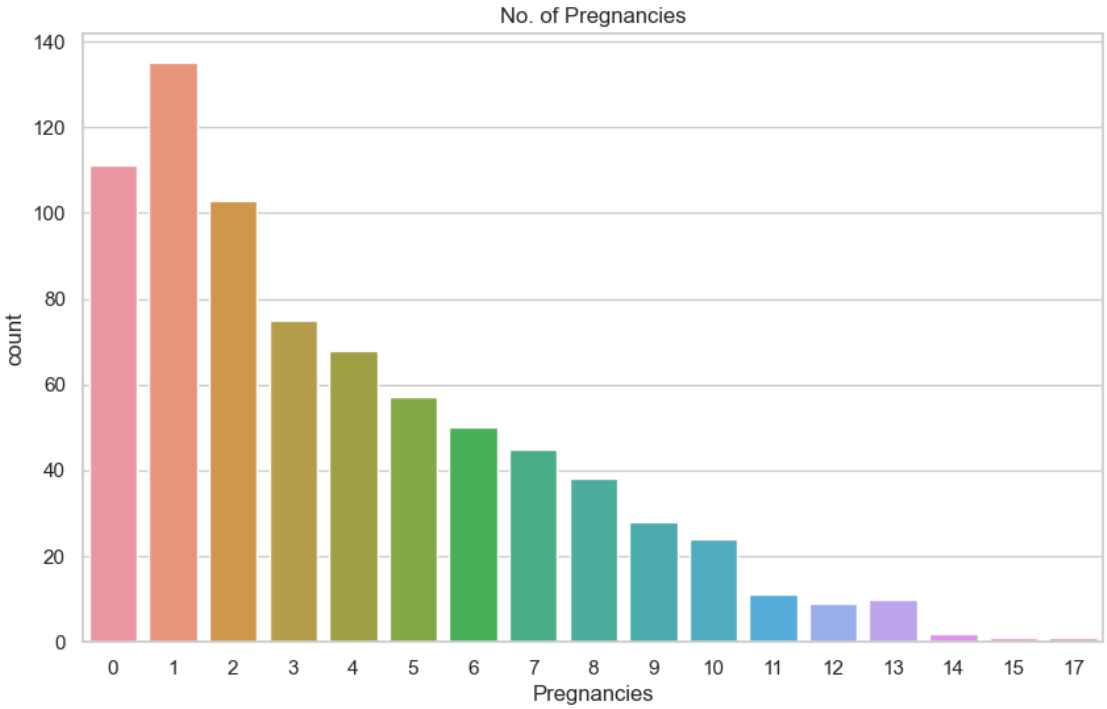
```
 classification report:
              precision    recall  f1-score   support

           0       0.79      0.78      0.78        99
           1       0.61      0.62      0.61        55

    accuracy                           0.72       154
   macro avg       0.70      0.70      0.70       154
weighted avg       0.72      0.72      0.72       154
```

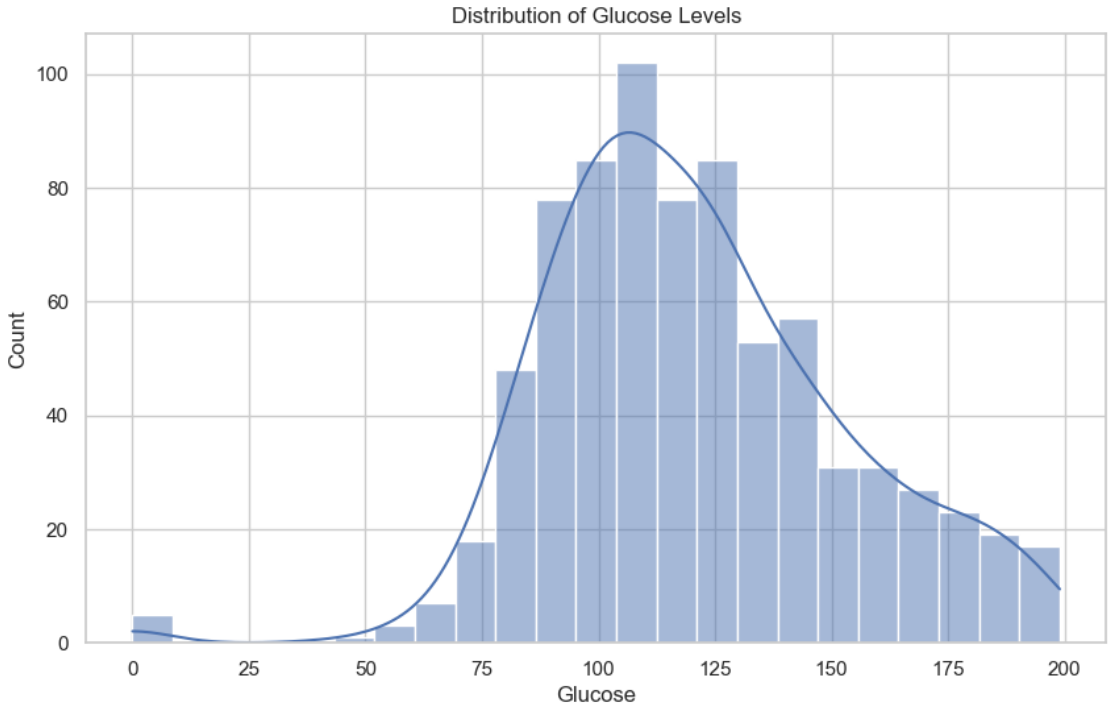
[28]:
```
print('\nconfusion matrix:')
print(confusion_matrix(Y_test,prediction))
```

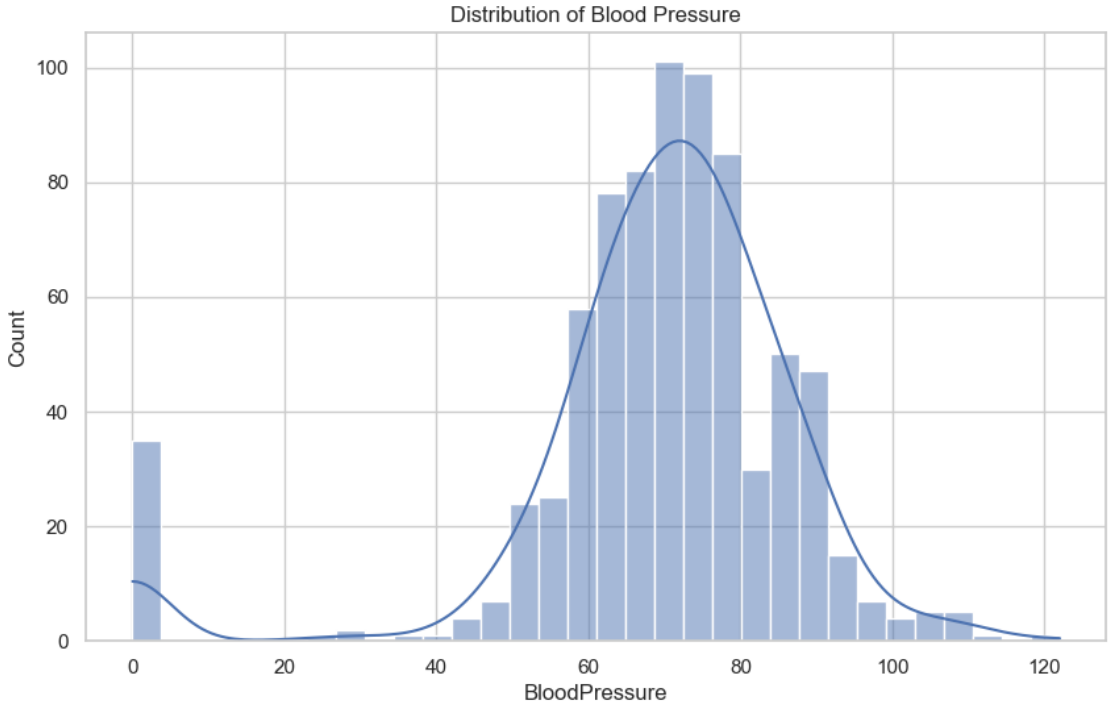
```
confusion matrix:
[[77 22]
 [21 34]]
```

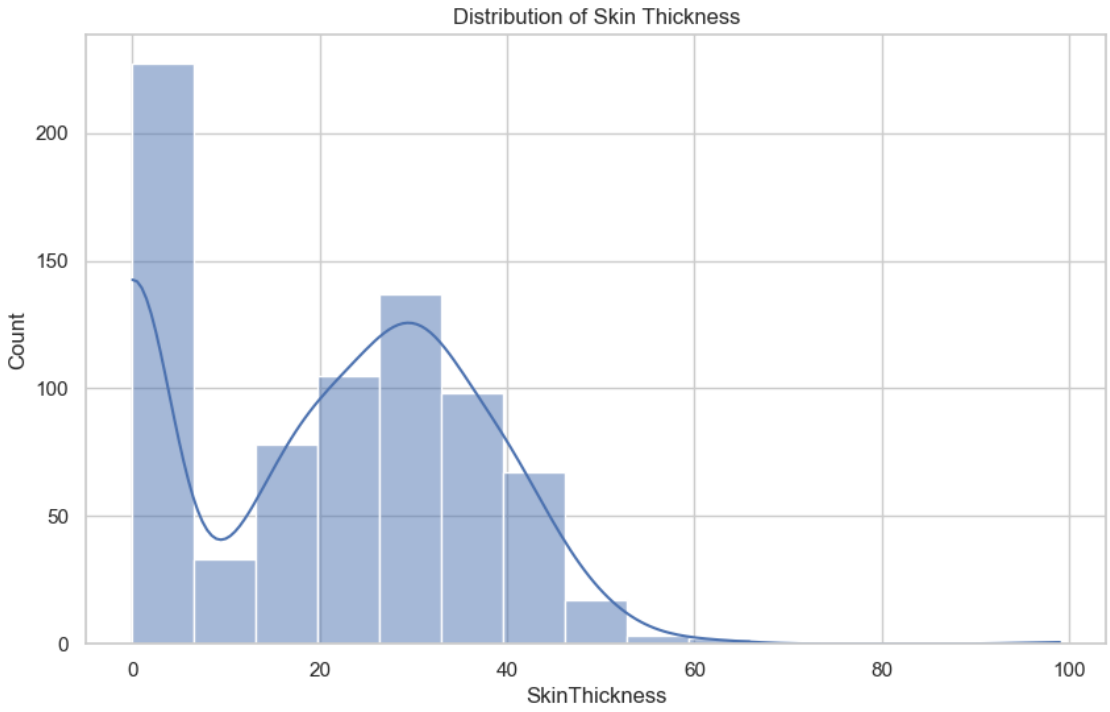
[31]:
```
sns.set(style="whitegrid")
#1.Pregnancies
plt.figure(figsize=(10,6))
sns.countplot(x='Pregnancies',data=data)
plt.title('No. of Pregnancies')
plt.show()
#2.Glucose
plt.figure(figsize=(10,6))
sns.histplot(x='Glucose',data=data,kde=True)
plt.title('Distribution of Glucose Levels')
plt.show()
#3.Blood Pressure
plt.figure(figsize=(10,6))
sns.histplot(x='BloodPressure',data=data,kde=True)
plt.title('Distribution of Blood Pressure')
plt.show()
#4.Skin THickness
plt.figure(figsize=(10,6))
sns.histplot(x='SkinThickness',data=data,kde=True)
plt.title('Distribution of Skin Thickness')
plt.show()
```

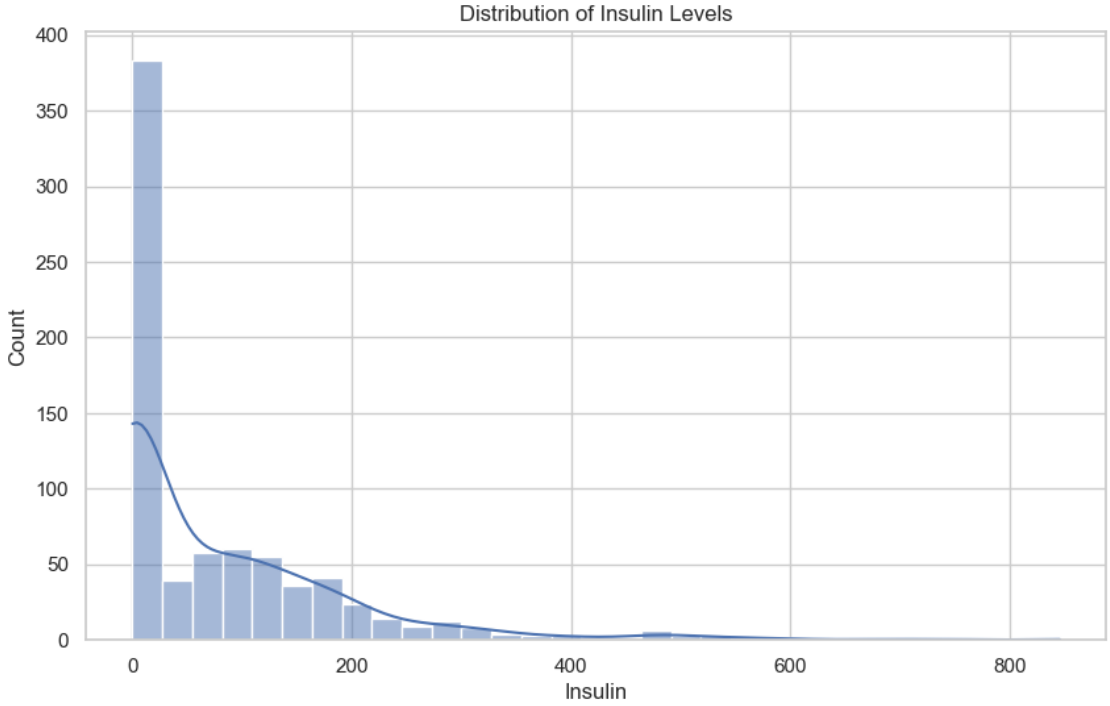
```
#5.Insulin
plt.figure(figsize=(10,6))
sns.histplot(x='Insulin',data=data,kde=True)
plt.title('Distribution of Insulin Levels')
plt.show()
#6.BMI
plt.figure(figsize=(10,6))
sns.histplot(x='BMI',data=data,kde=True)
plt.title('Distribution of BMI')
plt.show()
#7.Diabetes Pedigree Function
plt.figure(figsize=(10,6))
sns.histplot(x='DiabetesPedigreeFunction',data=data,kde=True)
plt.title('Distribution of Diabetes Pedigree Function Scores')
plt.show()
#8.Age
plt.figure(figsize=(10,6))
sns.histplot(x='Age',data=data,kde=True)
plt.title('Distribution of Age')
plt.show()
#9.Outcome
plt.figure(figsize=(10,6))
sns.countplot(x='Outcome',data=data)
plt.title('Distribution of Outcome 1: Diabetes (0: No Diabetes)')
plt.show()
```

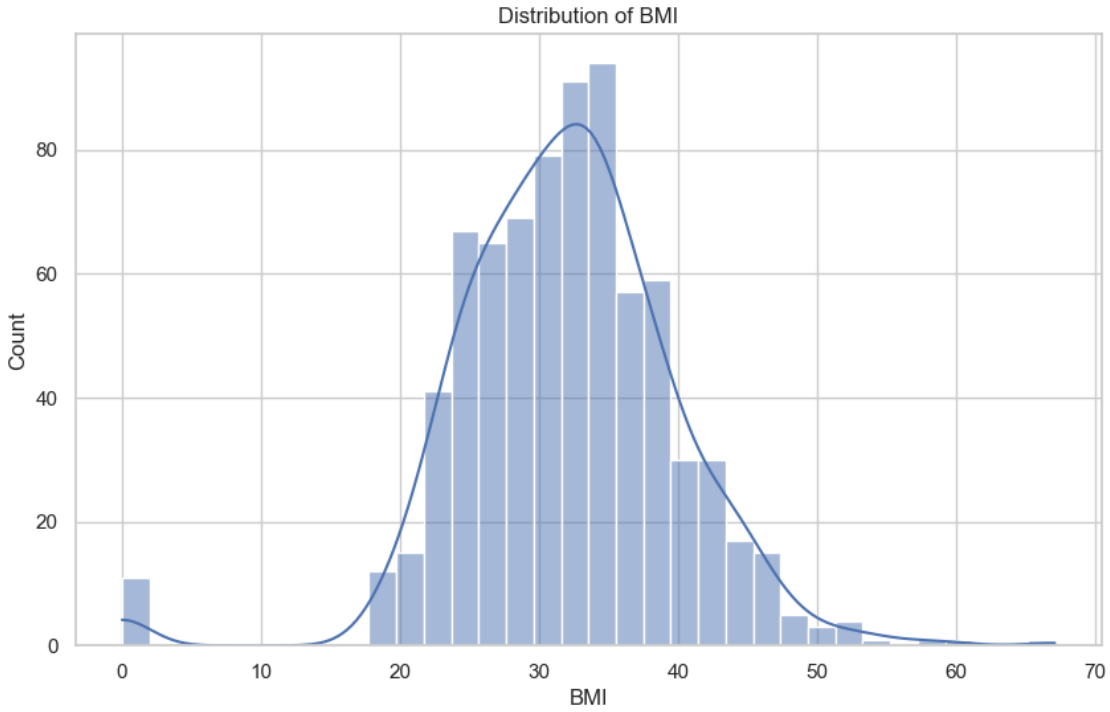


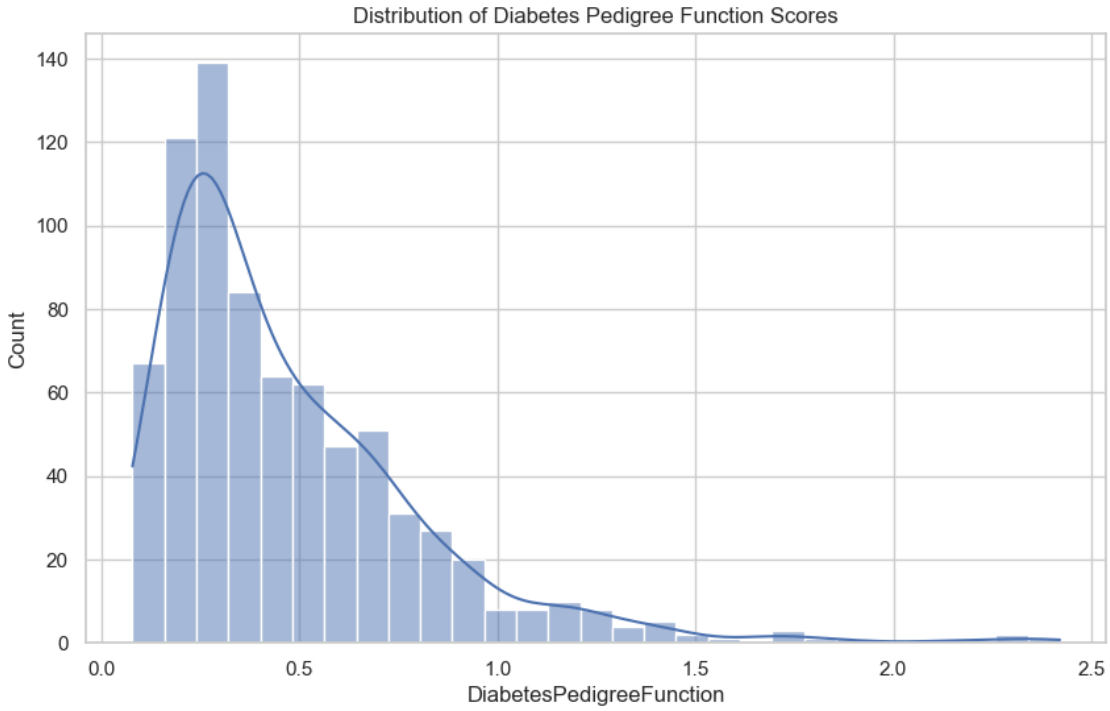

Distribution of Glucose Levels
Count
Glucose
0
20
40
60
80
100
0
25
50
75
100
125
150
175
200

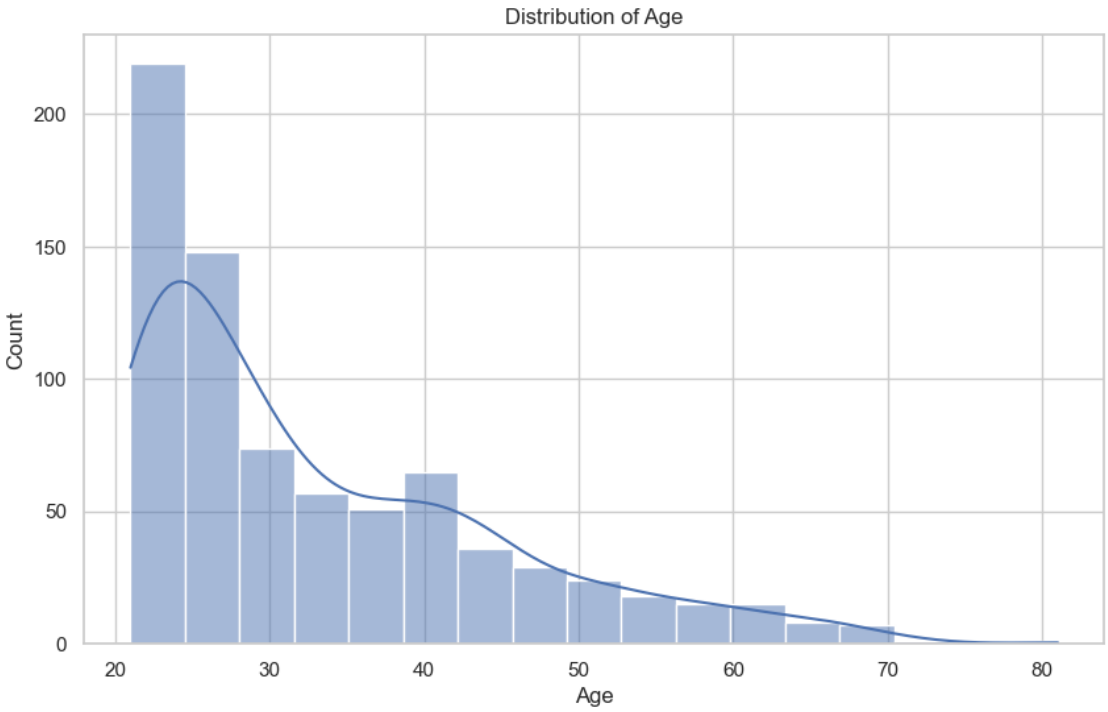

Distribution of Blood Pressure
Count
BloodPressure
0
20
40
60
80
100
0
20
40
60
80
100
120

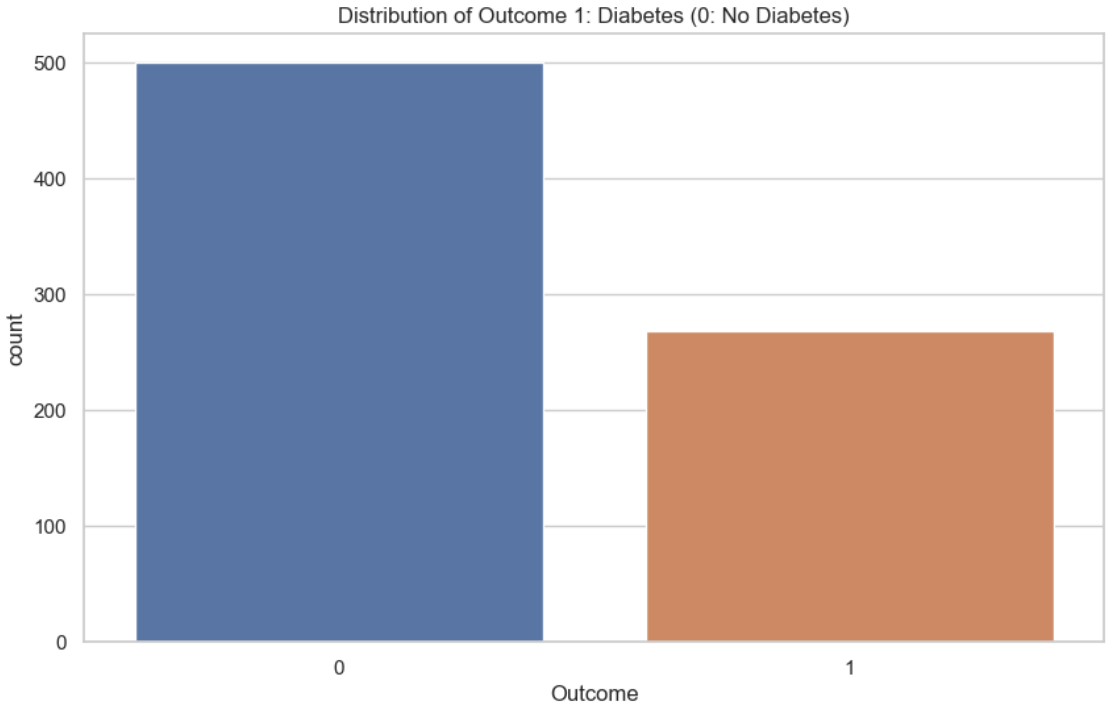

Distribution of Skin Thickness
Count
SkinThickness
0
50
100
150
200
0
20
40
60
80
100


Distribution of Insulin Levels
Count
Insulin
0
50
100
150
200
250
300
350
400
0
200
400
600
800

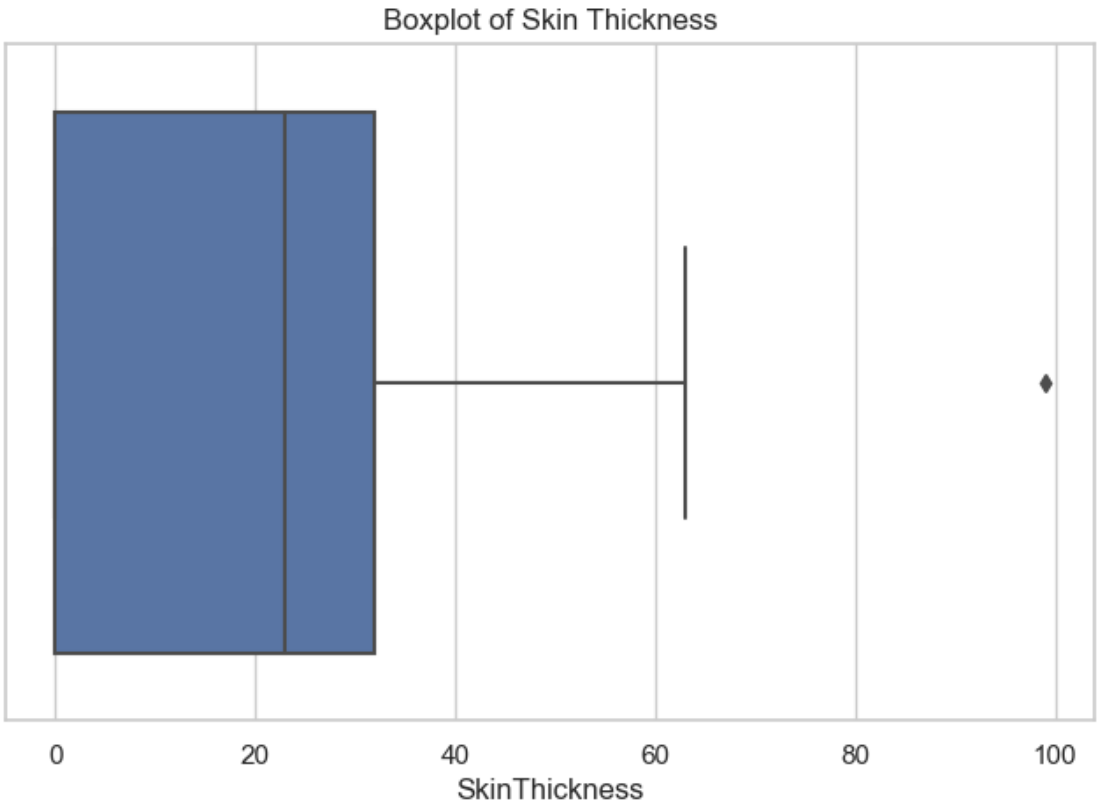

Distribution of BMI
Count
BMI
0
20
40
60
80
0
10
20
30
40
50
60
70


Distribution of Diabetes Pedigree Function Scores
Count
DiabetesPedigreeFunction
0
20
40
60
80
100
120
140
0.0
0.5
1.0
1.5
2.0
2.5


Distribution of Age
Count
Age
0
50
100
150
200
20
30
40
50
60
70
80


Distribution of Outcome 1: Diabetes (0: No Diabetes)
count
Outcome
0
100
200
300
400
500
0
1

[33]:
```
#boxplot of skin thickness
plt.figure(figsize=(8,5))
sns.boxplot(x='SkinThickness',data=data)
plt.title('Boxplot of Skin Thickness')
plt.show()
```

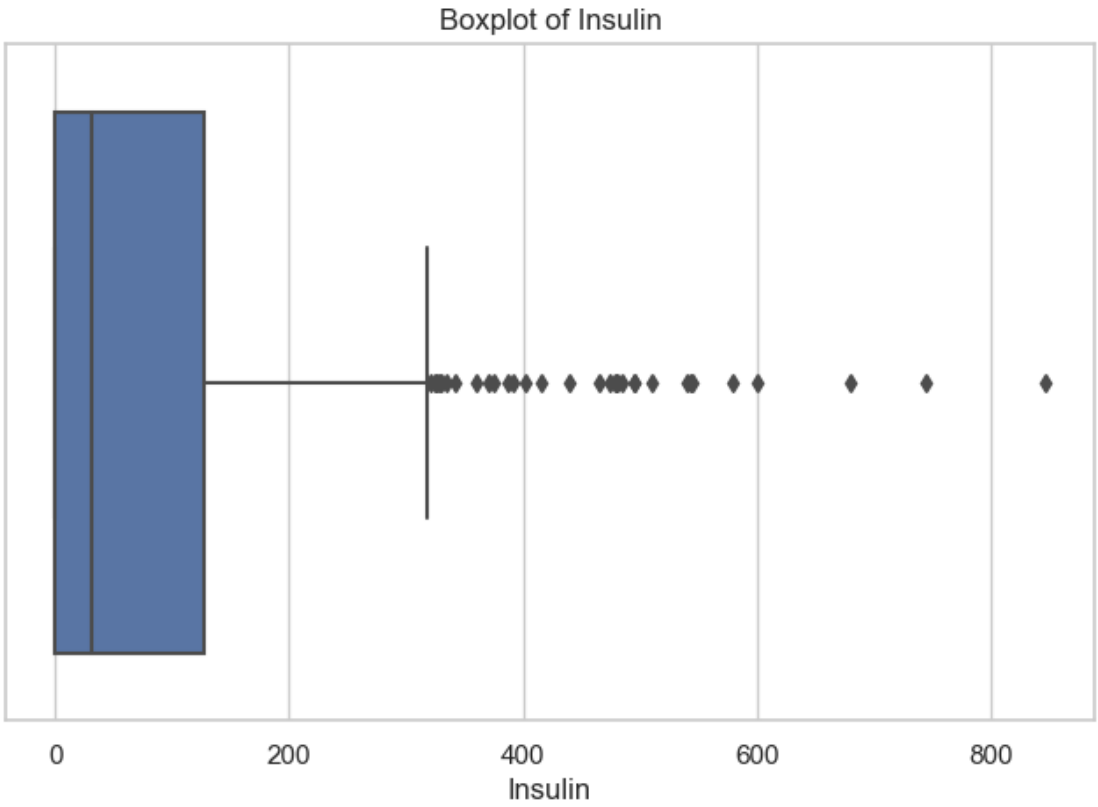


[34]:
```
#outliners in skin thickness
Q1_skin_thickness=data['SkinThickness'].quantile(0.25)
Q3_skin_thickness=data['SkinThickness'].quantile(0.75)
IQR_skin_thickness=Q3_skin_thickness-Q1_skin_thickness
lower_bound_skin_thickness=Q1_skin_thickness-1.5*IQR_skin_thickness
upper_bound_skin_thickness=Q3_skin_thickness+1.5*IQR_skin_thickness
outliers_skin_thickness=data[(data['SkinThickness']<lower_bound_skin_thickness)|(data['SkinThi
print(f'Number of outliers in SkinThickness:{len(outliers_skin_thickness)}')
```

```
Number of outliers in SkinThickness:1
```

[35]:
```
#boxplot of insulin
plt.figure(figsize=(8,5))
sns.boxplot(x='Insulin',data=data)
plt.title('Boxplot of Insulin')
```

```
plt.show()
```



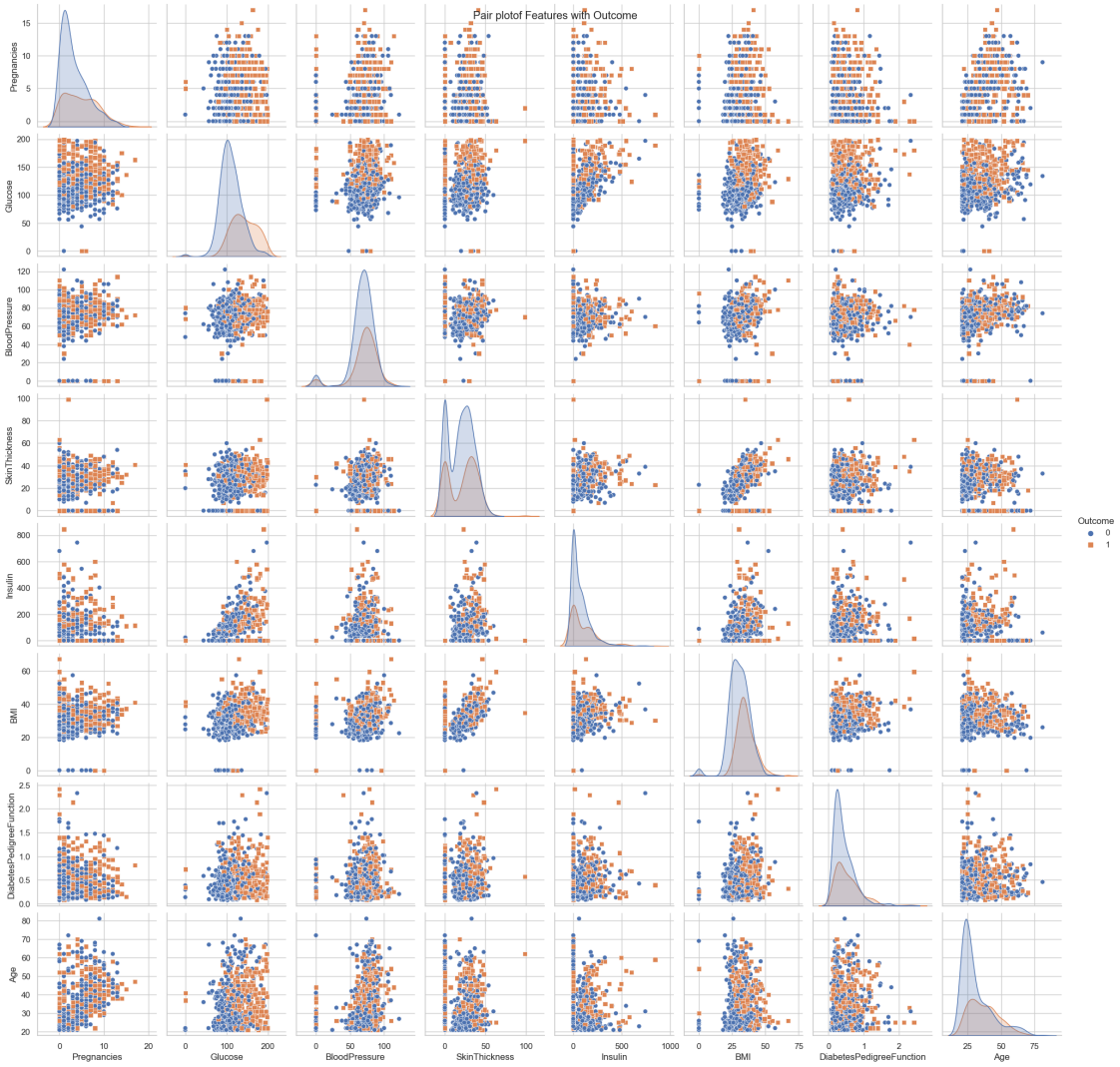
[36]:
```
#outliners in insulin
Q1_insulin = data['Insulin'].quantile(0.25)
Q3_insulin = data['Insulin'].quantile(0.75)
IQR_insulin = Q3_insulin - Q1_insulin
lower_bound_insulin = Q1_insulin - 1.5 * IQR_insulin
upper_bound_insulin = Q3_insulin + 1.5 * IQR_insulin
outliers_insulin = data[(data['Insulin'] < lower_bound_insulin)␣
 ↪|(data['Insulin'] > upper_bound_insulin)]
print(f'Number of outliers in Insulin: {len(outliers_insulin)}')
```

```
Number of outliers in Insulin: 34
```

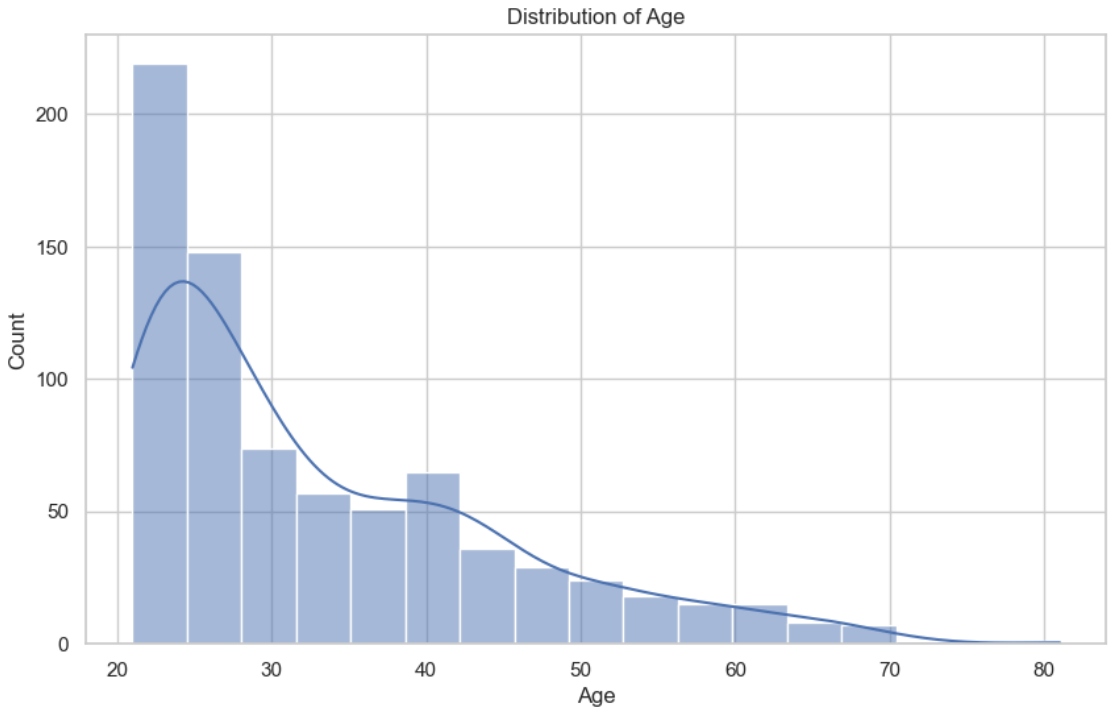
[37]:
```
#visualization between features and pair plot
sns.pairplot(data,hue='Outcome',diag_kind='kde',markers=["o","s"])
plt.suptitle('Pair plotof Features with Outcome')
plt.show()
```

```
D:\Users\RAVI\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning:
The figure layout has changed to tight
```

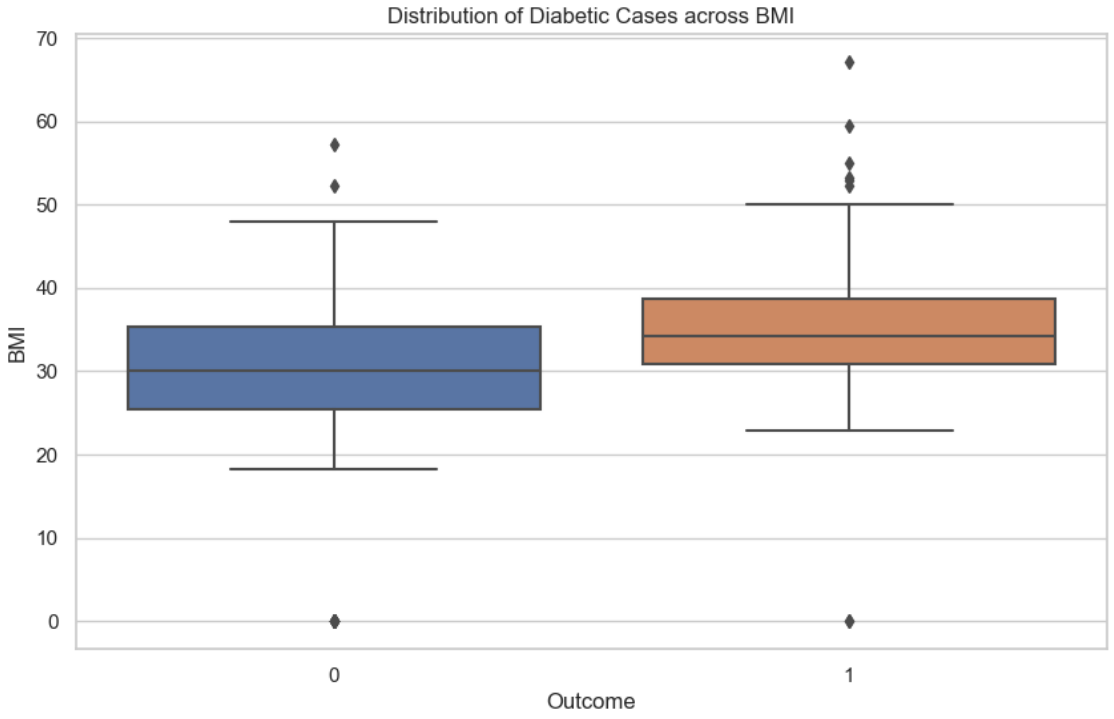
```
self._figure.tight_layout(*args, **kwargs)
```



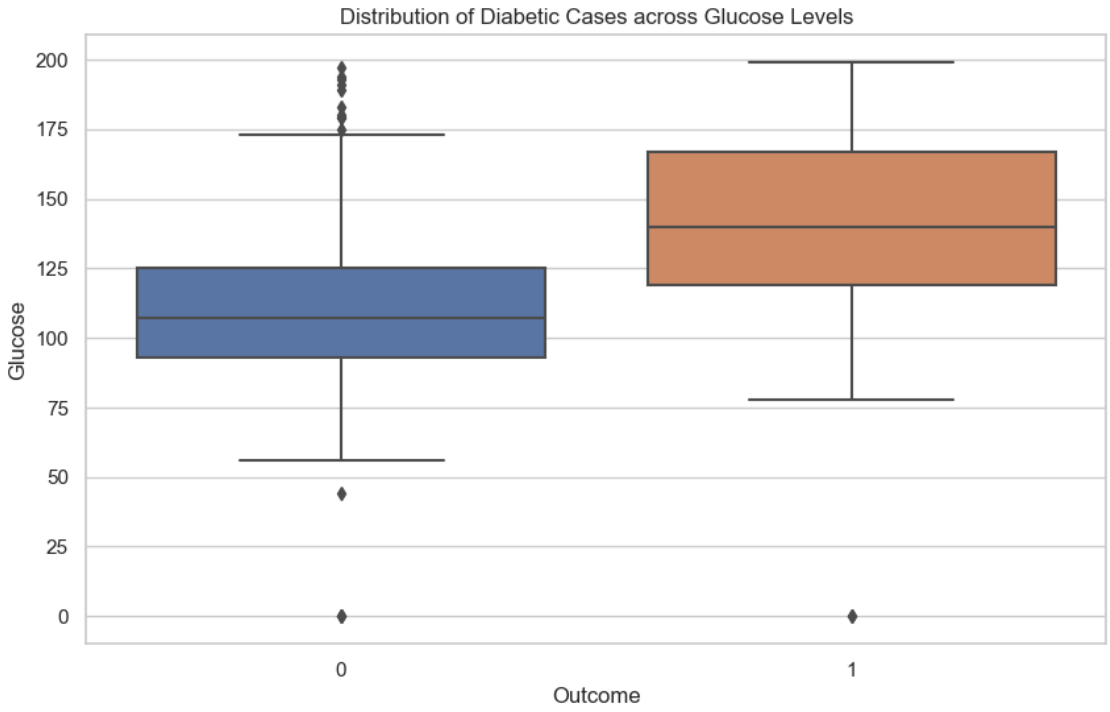
[39]:
```
#Distribution of Age
plt.figure(figsize=(10, 6))
sns.histplot(x='Age', data=data, kde=True)
plt.title('Distribution of Age')
plt.show()
```



[40]:
```
#Distribution of cases accross BMI
plt.figure(figsize=(10, 6))
sns.boxplot(x='Outcome', y='BMI', data=data)
plt.title('Distribution of Diabetic Cases across BMI')
plt.show()
```



[41]:
```
#Distribution of cases accross glucose level
plt.figure(figsize=(10, 6))
sns.boxplot(x='Outcome', y='Glucose', data=data)
plt.title('Distribution of Diabetic Cases across Glucose Levels')
plt.show()
```


Distribution of Diabetic Cases across Glucose Levels
Glucose
200
175
150
125
100
75
50
25
0
0
1
Outcome

[ ]: