NAME:- Devarapalli Nitish Babu

DESIGN OF RANDOM FOREST ML FOR CLASSIFICATION THEORY:-

The Random Forest algorithm is a powerful ensemble learning method used for classification and regression tasks. Here's a structured overview of its theoretical design for classification:

1. Introduction to Random Forest

Random Forest is an ensemble learning technique that combines multiple decision trees to improve the accuracy and robustness of predictions. It's particularly effective for classification problems and helps mitigate issues like overfitting that can occur in single decision trees.

3. How Random Forest Works

a. Building the Forest

1. **Bootstrapping**: For each tree in the forest, a random sample of the dataset is drawn with replacement. This means some instances may appear multiple times, while others may not be included.
2. **Tree Creation**:
   - Each decision tree is built using a random subset of features at each split. This randomness helps to ensure that the trees are decorrelated.
   - A typical practice is to select a square root of the total number of features for classification tasks.
3. **Tree Depth and Growth**: Trees are grown to their maximum depth (fully grown), or stopping criteria can be set to prevent overfitting.

b. Making Predictions

- **Voting Mechanism**: When making predictions, each tree in the forest gives a class prediction, and the final class prediction is determined by majority voting among the trees.
- This helps to smooth out noise and improve accuracy.

4. Key Advantages

- **Robustness**: Less prone to overfitting compared to individual decision trees due to averaging.
- **Handling Missing Values**: Can handle missing data effectively by using the available features.
- **Feature Importance**: Provides insights into feature importance, helping in understanding the underlying data structure.

Applications

- Medical diagnosis (e.g., diabetes classification).
- Financial modeling (credit scoring).

- Image classification.
- Natural language processing tasks.

7. Conclusion

Random Forest is a versatile and powerful algorithm that leverages the strengths of decision trees while minimizing their weaknesses. Its ensemble nature and ability to handle large datasets with high dimensionality make it a popular choice in many machine learning applications.

code:-

```
import pandas as pd import
seaborn as sns import
matplotlib.pyplot as plt from
sklearn.model_selection
import train_test_split from
sklearn.ensemble import
RandomForestClassifier
from sklearn.metrics import
confusion_matrix,
ConfusionMatrixDisplay



data = pd.read_csv('diabetes.csv')
```

```
print(data.head())

X = data.iloc[:, :-1]   y
= data.iloc[:, -1]



X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)



model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

importances = model.feature_importances_ feature_names
= X.columns

importance_df = pd.DataFrame({'Feature': feature_names, 'Importance':
importances}) importance_df =
importance_df.sort_values(by='Importance',
ascending=False)

plt.figure(figsize=(10, 6)) sns.barplot(x='Importance',
y='Feature', data=importance_df) plt.title('Feature
Importance from Random Forest') plt.show()

y_pred = model.predict(X_test)

cm = confusion_matrix(y_test, y_pred) cmd =
ConfusionMatrixDisplay(confusion_matrix=cm)
```

```
cmd.plot(cmap=plt.cm.Blues) plt.title('Confusion
Matrix') plt.show()
```

Output:-

```
C:\Users\22BCE8391\Desktop\random forest algorithm> &
"C:/Program Files/Python312/python.exe"
"c:/Users/22BCE8391/Desktop/random forest algorithm/import
pandas as pd.py"
   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
```

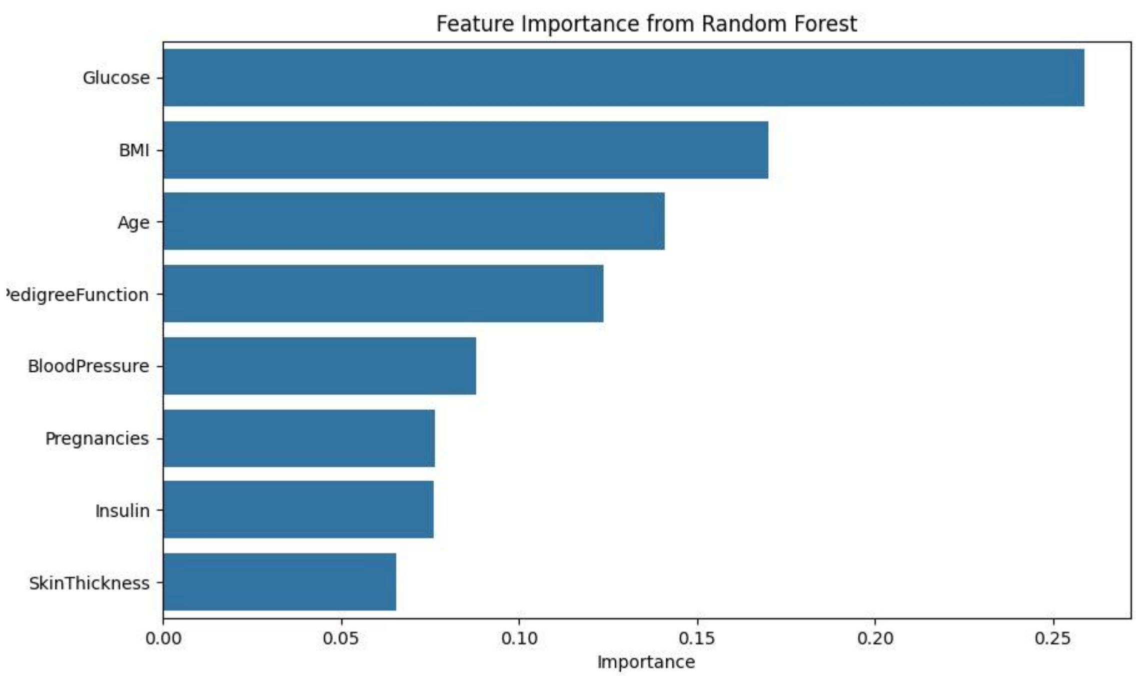
2.288 33 1

Confusion Matrix:

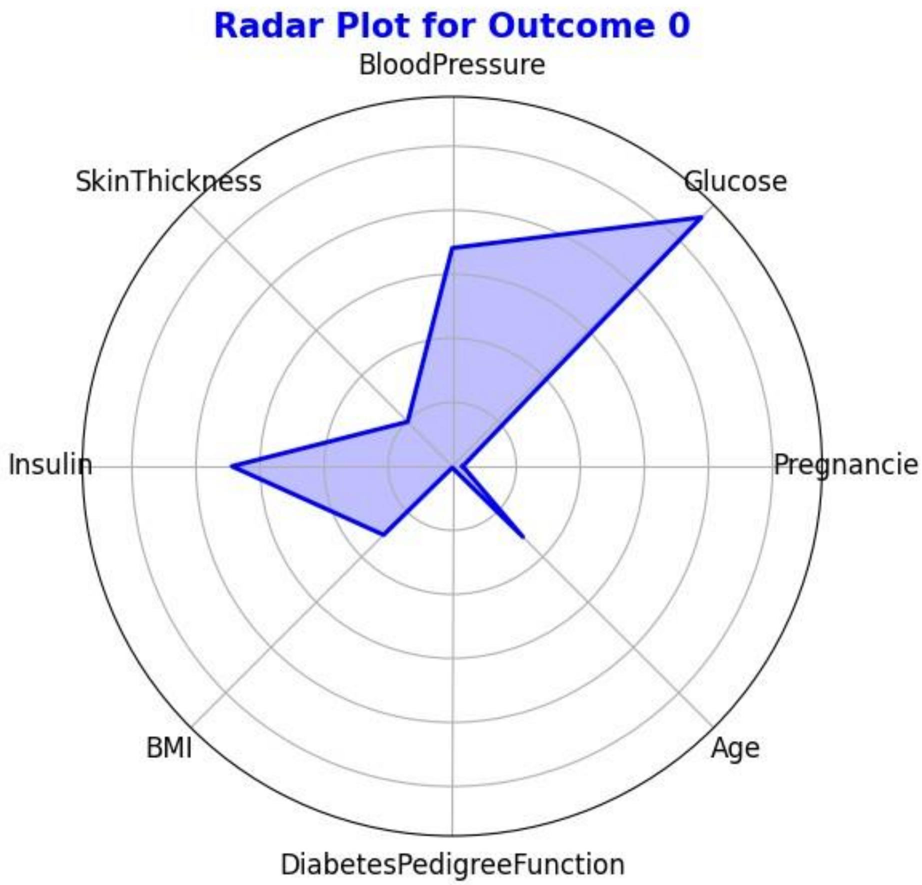
[[77 22]
 [21 34]]

Classification Report:

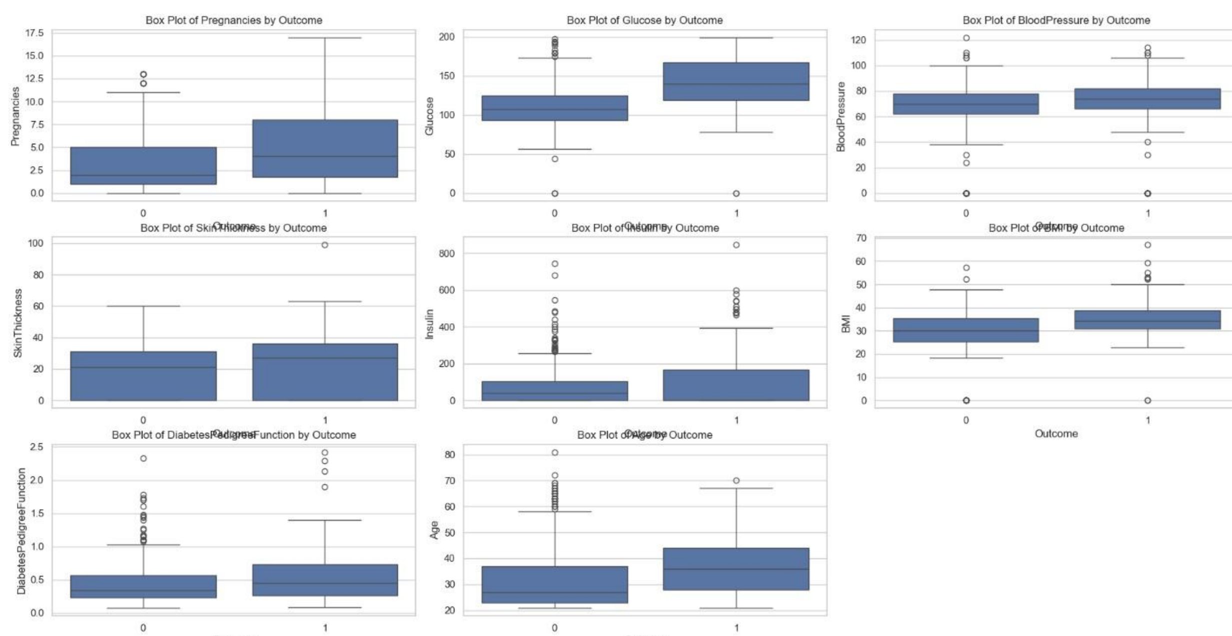
	precision	recall	f1-score	support
0	0.79	0.78	0.78	99
1	0.61	0.62	0.61	55
accuracy			0.72	154
macro avg	0.70	0.70	0.70	154
weighted avg	0.72	0.72	0.72	154



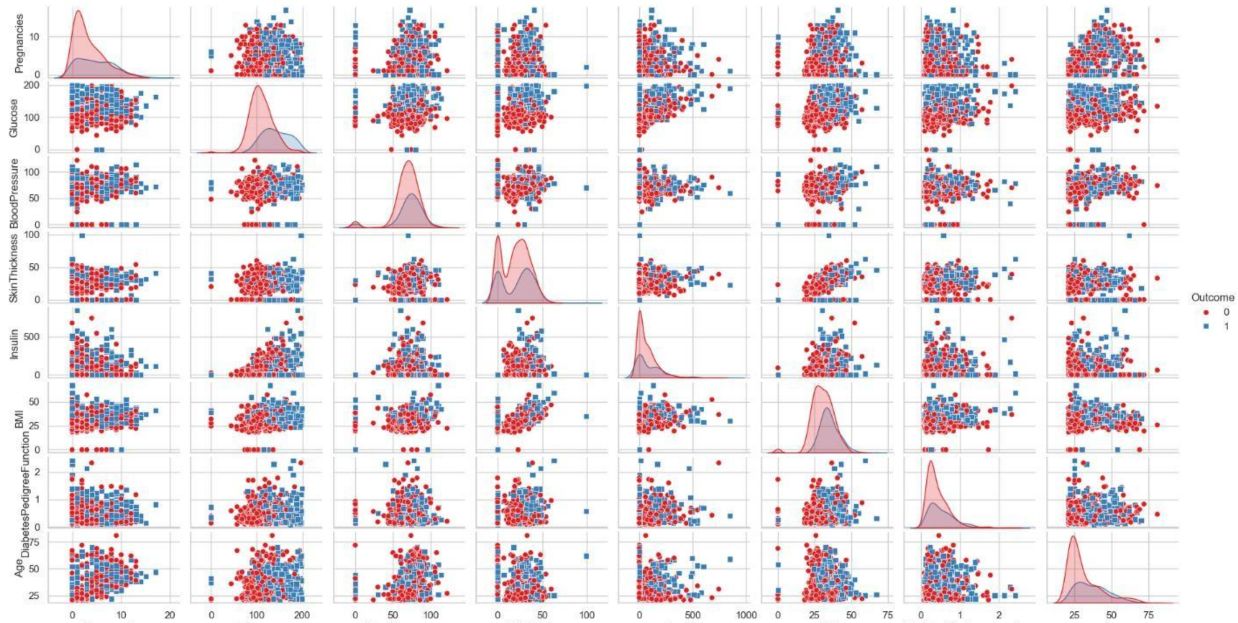
RADAR PLOT:-


Radar Plot for Outcome 0
BloodPressure
Glucose
Pregnancie
Age
DiabetesPedigreeFunction
BMI
Insulin
SkinThickness

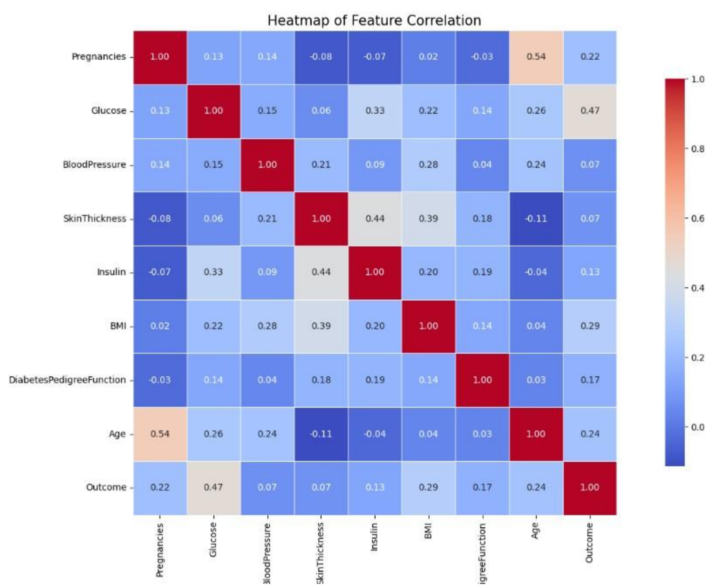
BOX PLOT:-


Box Plot of Pregnancies by Outcome
Box Plot of Glucose by Outcome
Box Plot of BloodPressure by Outcome
Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
BMI
DiabetesPedigreeFunction
Age
Outcome

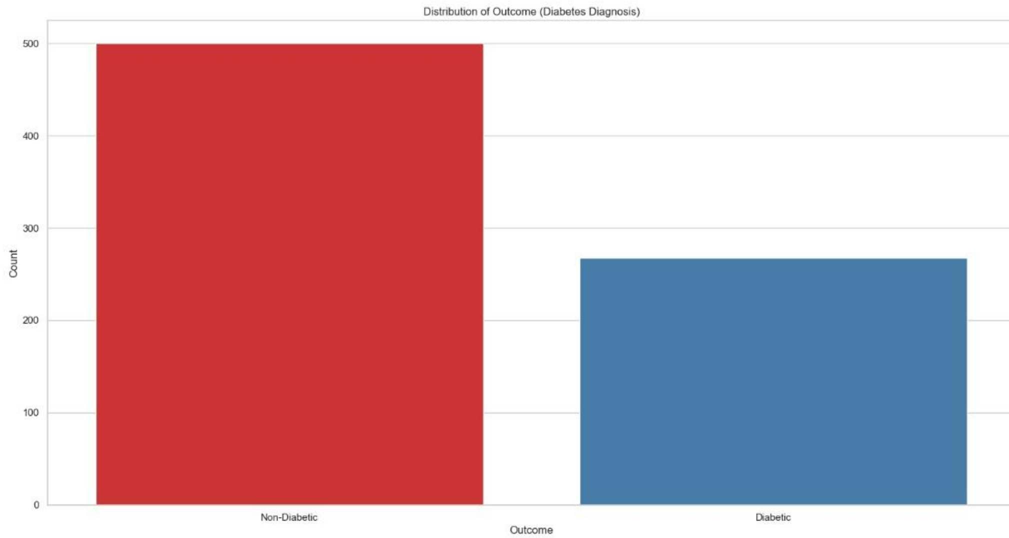
SCATTER PLOT:-



HEAPMATRIX:-



BAR GRAPH PLOT :-



XG BOOST CLASSIFICATION:-

import pandas as pd import seaborn
as sns import matplotlib.pyplot as
plt from sklearn.model_selection
import train_test_split
from sklearn.metrics import confusion_matrix,
ConfusionMatrixDisplay
from xgboost import XGBClassifier

data = pd.read_csv('diabetes.csv')
print(data.head())

```
X = data.iloc[:, :-1]   y
= data.iloc[:, -1]

X_train, X_test, y_train, y_test = train_test_split(X,
y, test_size=0.2, random_state=42)

model = XGBClassifier(use_label_encoder=False,
eval_metric='logloss', random_state=42)
model.fit(X_train, y_train)

importances = model.feature_importances_
feature_names = X.columns

importance_df = pd.DataFrame({'Feature':
feature_names, 'Importance': importances})
importance_df =
importance_df.sort_values(by='Importance',
ascending=False)

plt.figure(figsize=(10, 6))
sns.barplot(x='Importance', y='Feature',
data=importance_df)
```

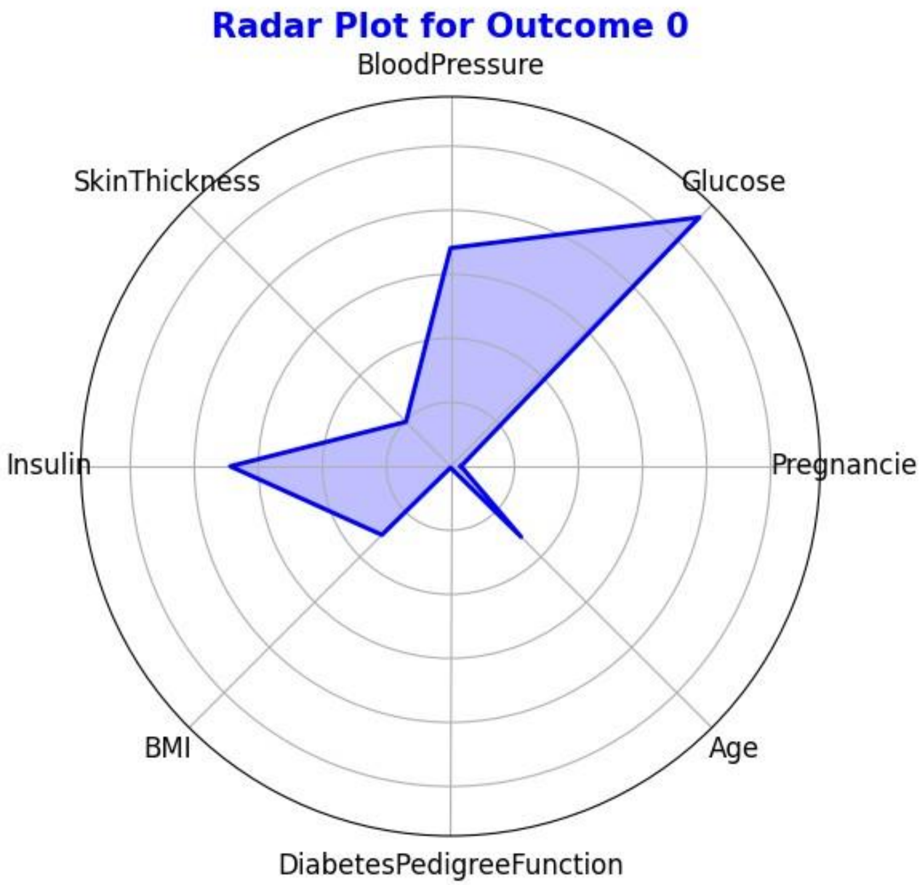
```
plt.title('Feature Importance from XGBoost')
plt.show()

y_pred = model.predict(X_test)

cm = confusion_matrix(y_test, y_pred) cmd
=
ConfusionMatrixDisplay(confusion_matrix=cm)
cmd.plot(cmap=plt.cm.Blues)
plt.title('Confusion Matrix - XGBoost') plt.show()
```

OUTPUT:-

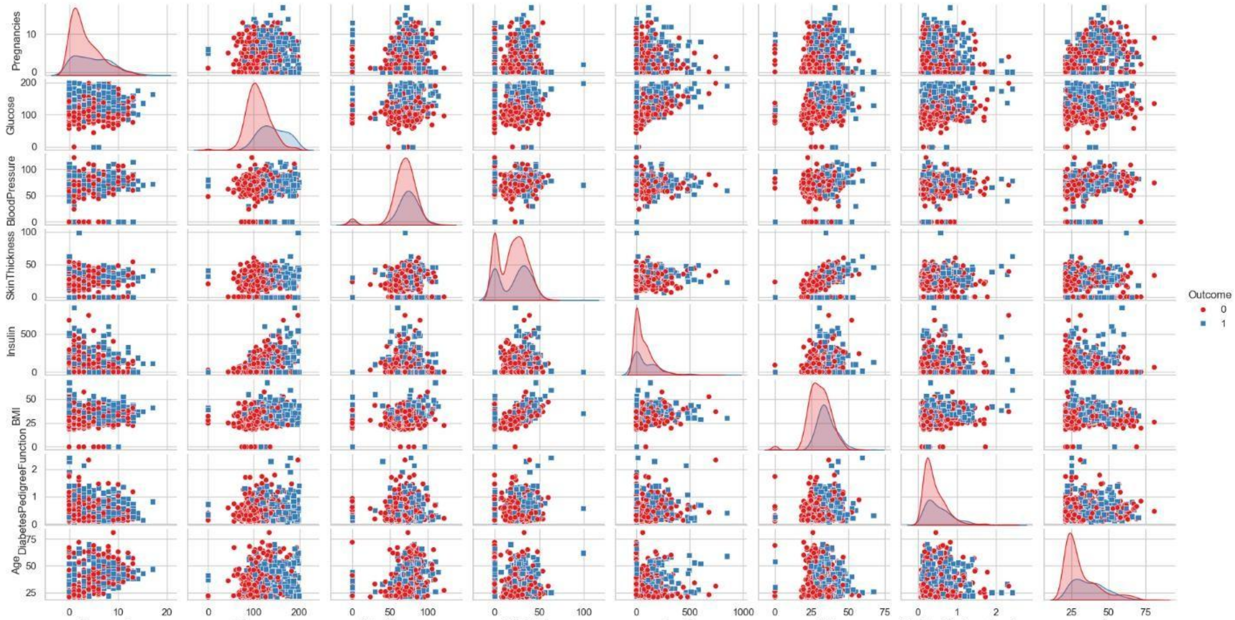
```
  Pregnancies  Glucose  BloodPressure
SkinThickness  Insulin  BMI  \
0            6      148             72             35        0
33.6
1            1       85             66             29        0  26.6
2            8      183             64              0        0  23.3
3            1       89             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

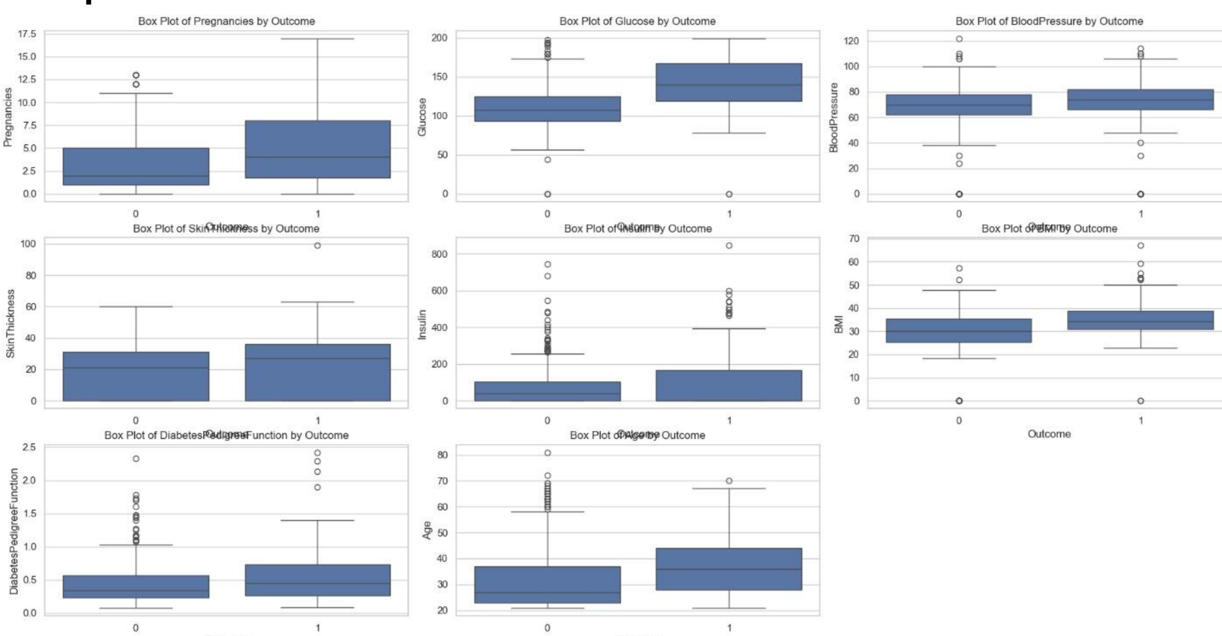
Radar plots:-



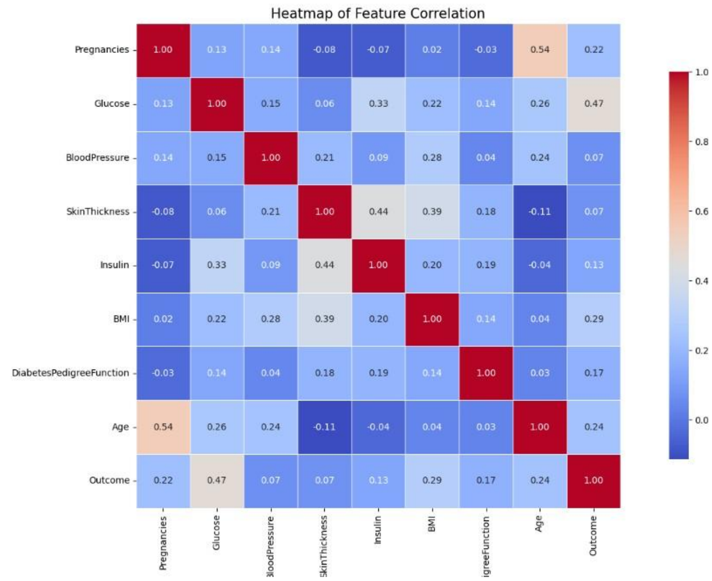
Scatter plots:-



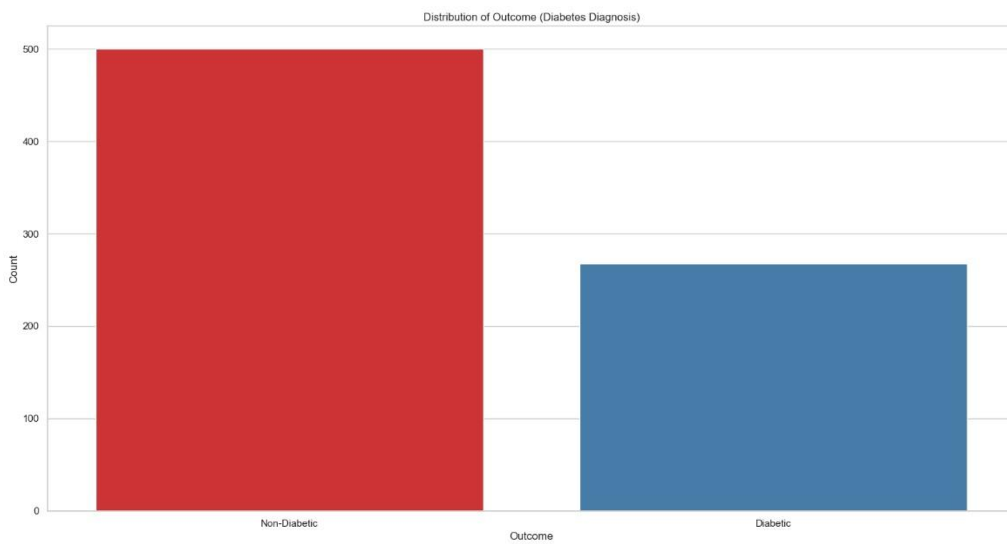
box plots:-



heap matrix


Heatmap of Feature Correlation
Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
BMI
DiabetesPedigreeFunction
Age
Outcome

bar graphs


Distribution of Outcome (Diabetes Diagnosis)
Count
Non-Diabetic
Diabetic
Outcome