Name : Yash Chikhale
Class : D15C
Batch : C
Rollno: 50

DMBI - 3

Aim: To perform Exploratory Data Analysis and Visualization using python.

Theory:
Exploratory Data Analysis (EDA) is the process of exploring, summarizing, and visualizing data to understand its main characteristics before applying statistical models or machine learning. It helps researchers **detect underlying structures, spot anomalies, identify relationships, and test hypotheses.**

The major steps include:

1. **Descriptive Statistics**
   ○ Provides numerical summaries such as mean, median, variance, min/max values, and standard deviation.
   ○ Helps detect skewness, outliers, and unusual distributions

2. **Target Variable Analysis**
   ○ The dependent variable (here: heart_disease) is visualized with count plots to check class balance.
   ○ If the dataset is highly imbalanced, it may affect classification performance.

3. **Correlation Analysis**
   ○ Pearson's correlation coefficient is calculated between numeric features.
   ○ A heatmap helps identify strong positive/negative correlations (e.g., thalach vs. age, chol vs. bmi).
   ○ Useful for detecting multicollinearity or redundant features.

4. **Feature Distribution Analysis**
   ○ Histograms/KDE plots show how features like age, cholesterol, and bmi are distributed (normal, skewed, multimodal).
   ○ Boxplots grouped by target show how continuous features vary between patients with and without heart disease.

5. **Categorical Feature Analysis**
   ○ Countplots and bar charts show how categorical features (e.g., sex, cp, thal) are distributed across target classes.
   ○ This helps assess the predictive power of categorical variables (e.g., chest pain type has strong association with heart disease).

6. **Multivariate Visualization**
   ○ Pairplots allow simultaneous visualization of multiple variables, highlighting clusters and class separation.
   ○ Useful to see which combinations of features separate patients with vs. without heart disease.

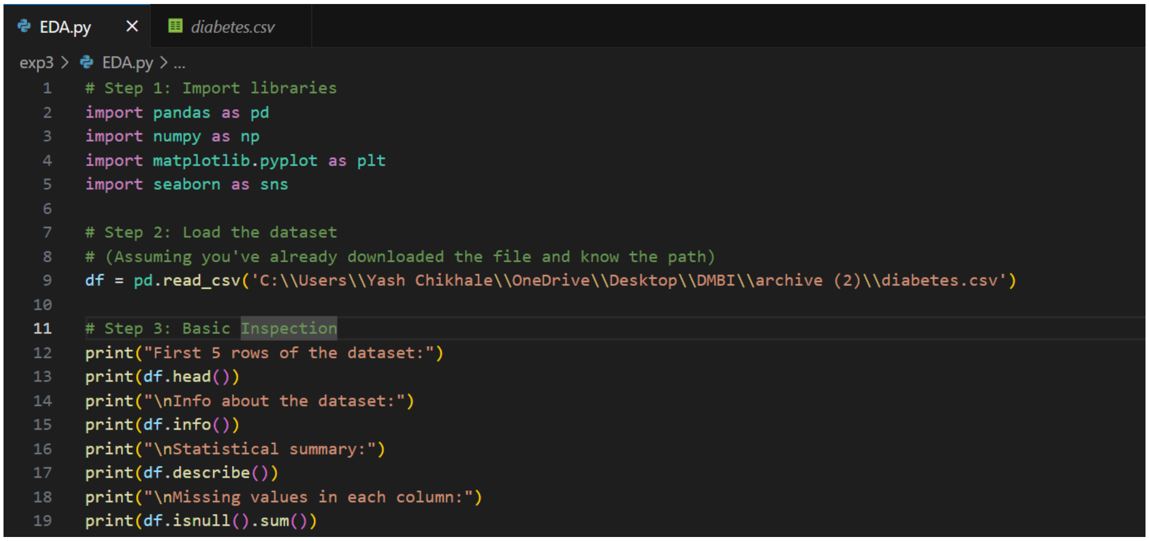
Importance:

- Provides deeper insight into dataset structure.
- Helps select features that are most relevant for prediction.
- Reveals outliers or errors that might need special treatment.
- Builds intuition about how independent variables influence the target outcome.

Conclusion:

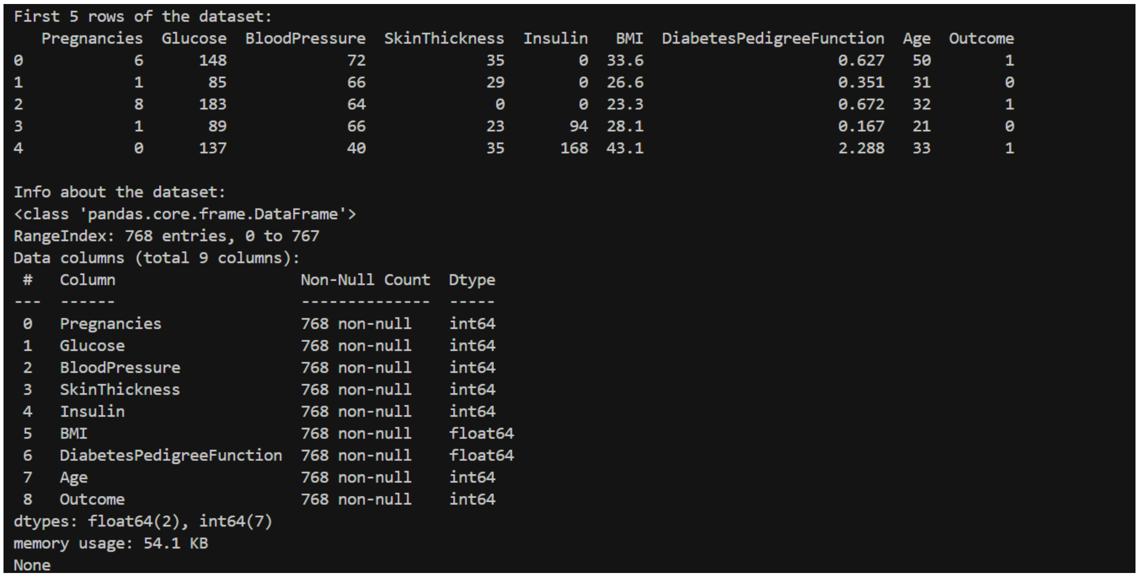
- Key Predictors: Glucose, BMI, and Age are significant in predicting diabetes.
- Class Balance: The dataset has a moderate class imbalance, which should be considered in modeling.
- Data Quality: Some features have implausible zero values (e.g., BloodPressure, Glucose) indicating possible data entry errors.
- Feature Relationships: Most features are weakly correlated, reducing multicollinearity concerns.
- Modeling Readiness: The dataset, after handling anomalies, is suitable for building predictive models for diabetes.

Code and Output:

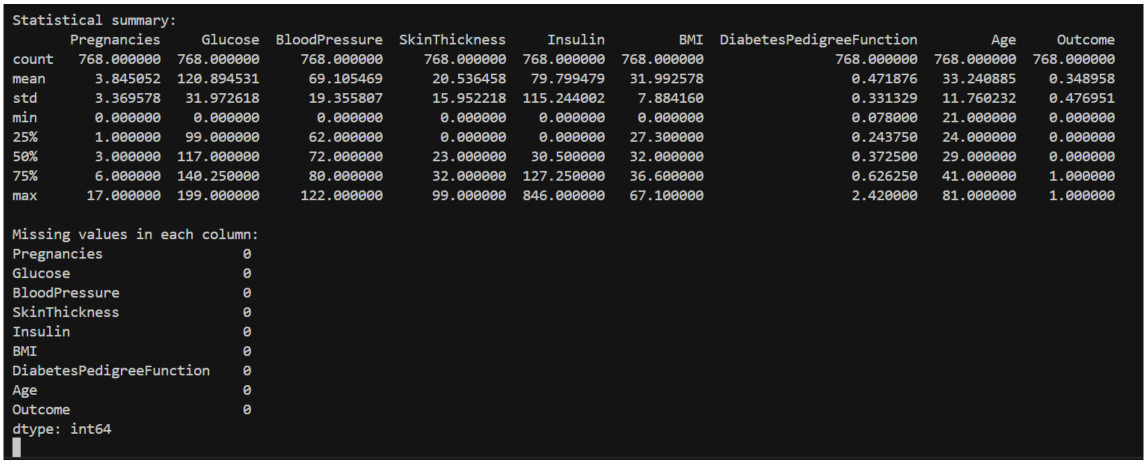


```python
# Step 1: Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Step 2: Load the dataset
# (Assuming you've already downloaded the file and know the path)
df = pd.read_csv('C:\\Users\\Yash Chikhale\\OneDrive\\Desktop\\DMBI\\archive (2)\\diabetes.csv')

# Step 3: Basic Inspection
print("First 5 rows of the dataset:")
print(df.head())
print("\nInfo about the dataset:")
print(df.info())
print("\nStatistical summary:")
print(df.describe())
print("\nMissing values in each column:")
print(df.isnull().sum())
```

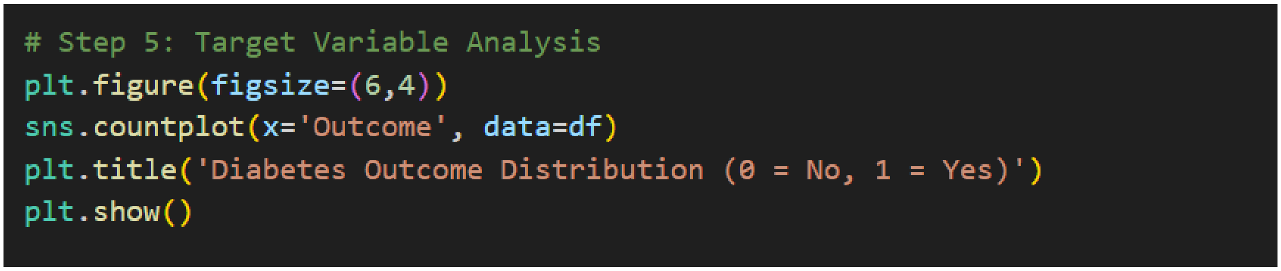
👇



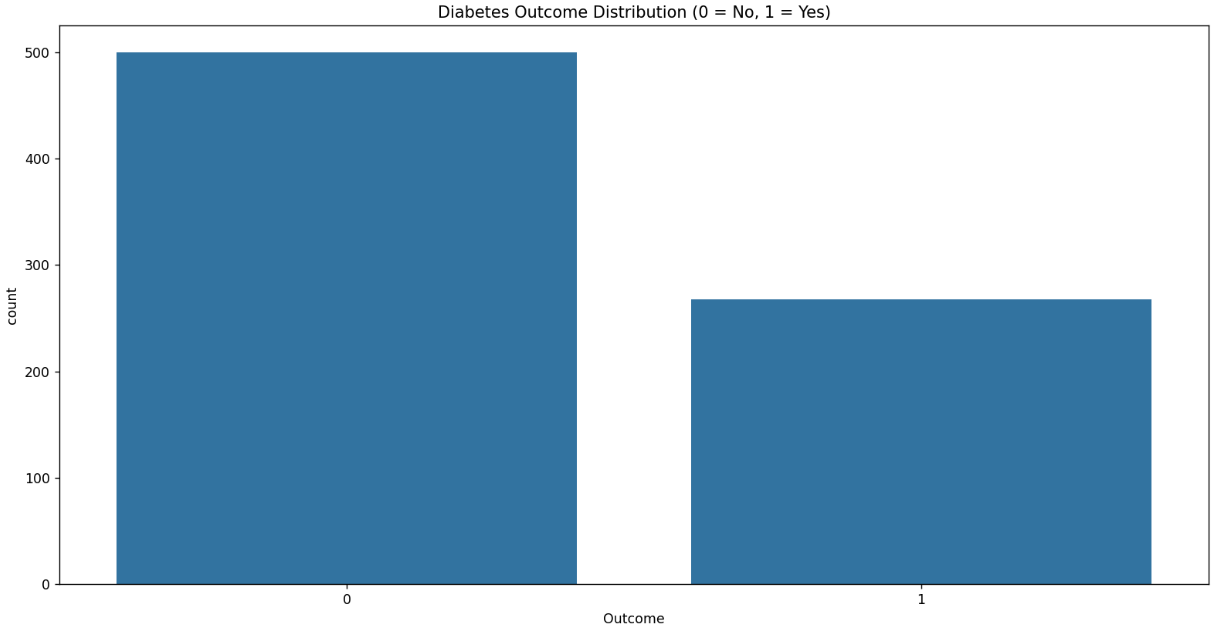
```
First 5 rows of the dataset:
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  DiabetesPedigreeFunction  Age  Outcome
0            6      148             72             35        0  33.6                     0.627   50        1
1            1       85             66             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

Info about the dataset:
<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
None
```

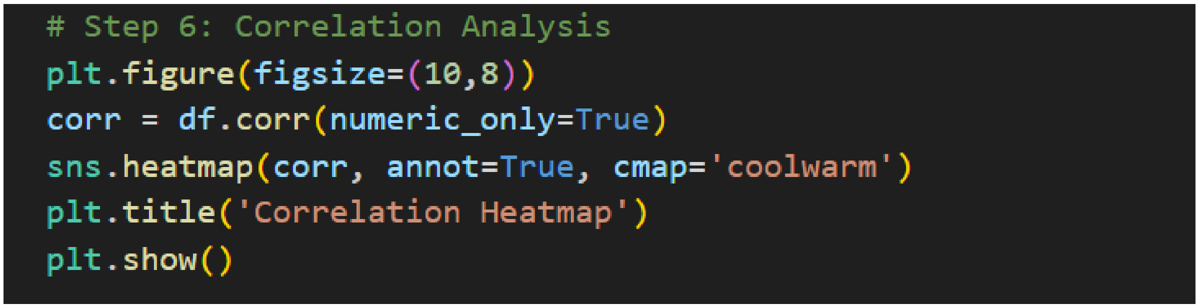


```
Statistical summary:
       Pregnancies     Glucose  BloodPressure  SkinThickness     Insulin         BMI  DiabetesPedigreeFunction         Age     Outcome
count   768.000000  768.000000     768.000000     768.000000  768.000000  768.000000                768.000000  768.000000  768.000000
mean      3.845052  120.894531      69.105469      20.536458   79.799479   31.992578                  0.471876   33.240885    0.348958
std       3.369578   31.972618      19.355807      15.952218  115.244002    7.884160                  0.331329   11.760232    0.476951
min       0.000000    0.000000       0.000000       0.000000    0.000000    0.000000                  0.078000   21.000000    0.000000
25%       1.000000   99.000000      62.000000       0.000000    0.000000   27.300000                  0.243750   24.000000    0.000000
50%       3.000000  117.000000      72.000000      23.000000   30.500000   32.000000                  0.372500   29.000000    0.000000
75%       6.000000  140.250000      80.000000      32.000000  127.250000   36.600000                  0.626250   41.000000    1.000000
max      17.000000  199.000000     122.000000      99.000000  846.000000   67.100000                  2.420000   81.000000    1.000000

Missing values in each column:
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

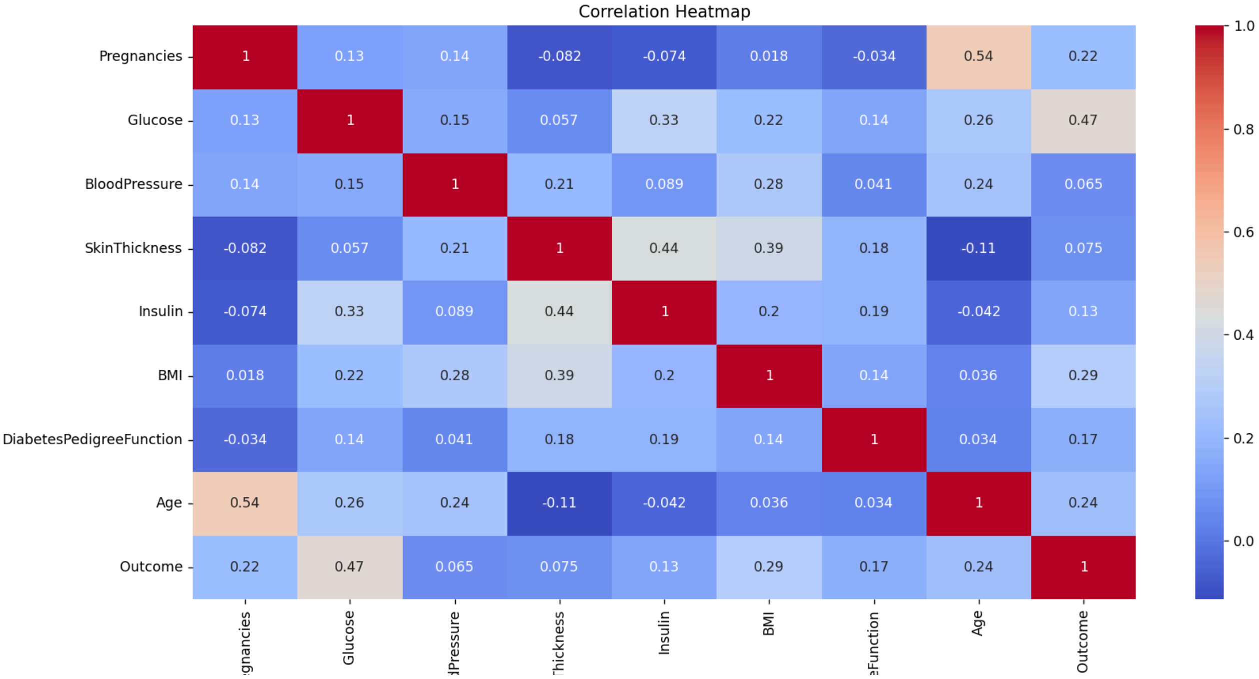


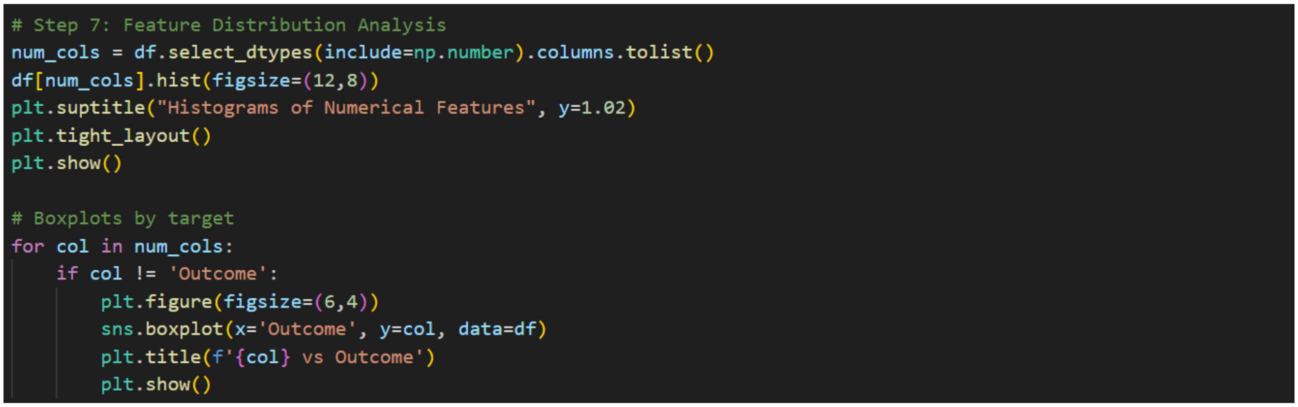
```
# Step 5: Target Variable Analysis
plt.figure(figsize=(6,4))
sns.countplot(x='Outcome', data=df)
plt.title('Diabetes Outcome Distribution (0 = No, 1 = Yes)')
plt.show()
```



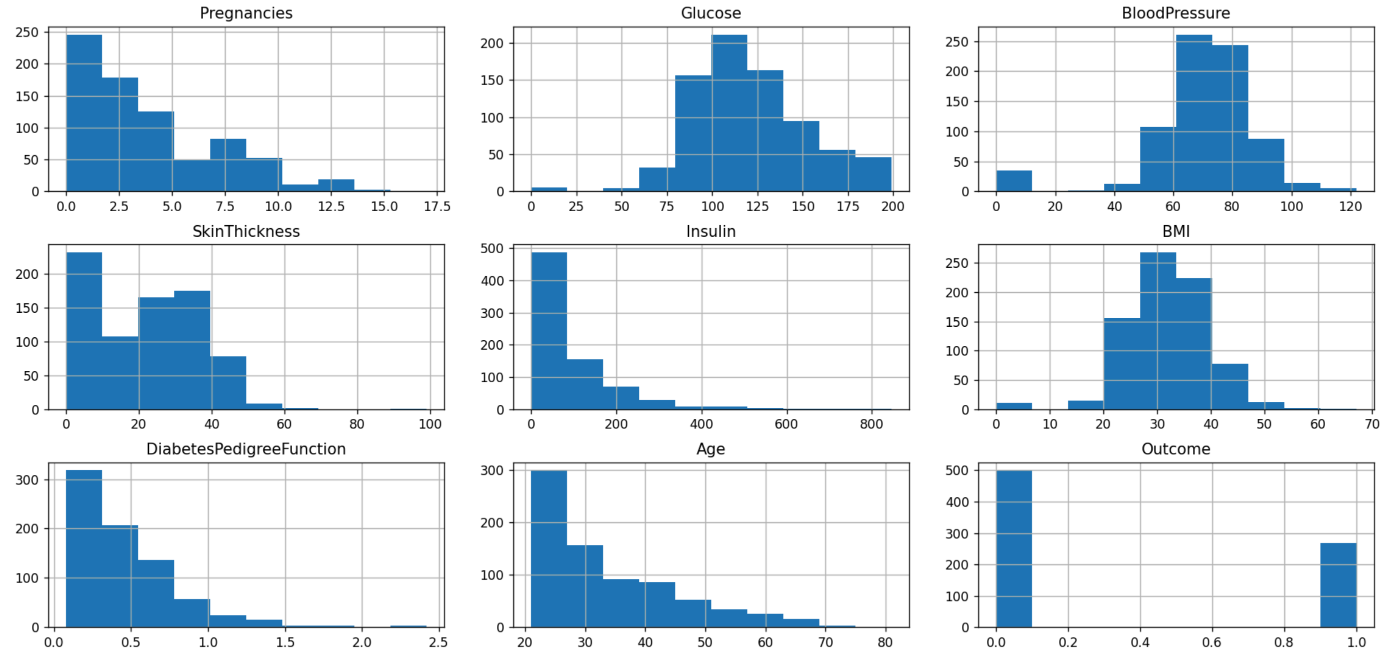


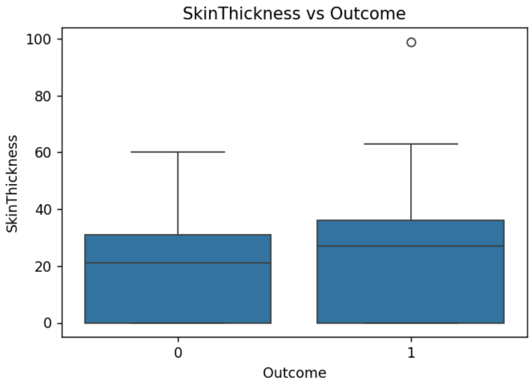
```
# Step 6: Correlation Analysis
plt.figure(figsize=(10,8))
corr = df.corr(numeric_only=True)
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

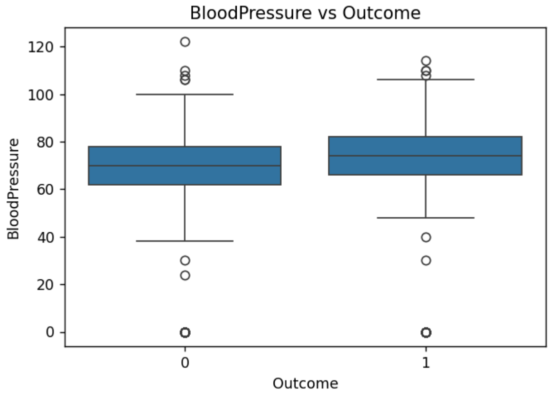


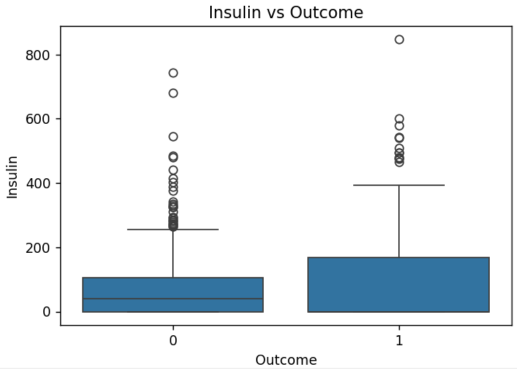


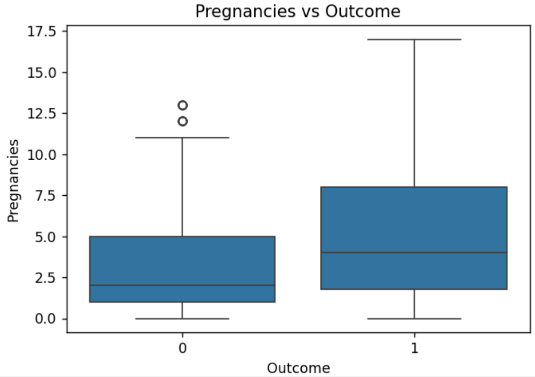
```python
# Step 7: Feature Distribution Analysis
num_cols = df.select_dtypes(include=np.number).columns.tolist()
df[num_cols].hist(figsize=(12,8))
plt.suptitle("Histograms of Numerical Features", y=1.02)
plt.tight_layout()
plt.show()

# Boxplots by target
for col in num_cols:
    if col != 'Outcome':
        plt.figure(figsize=(6,4))
        sns.boxplot(x='Outcome', y=col, data=df)
        plt.title(f'{col} vs Outcome')
        plt.show()
```

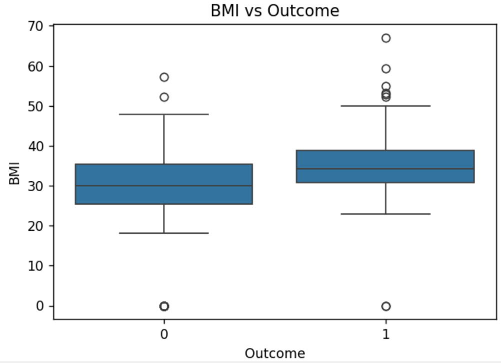


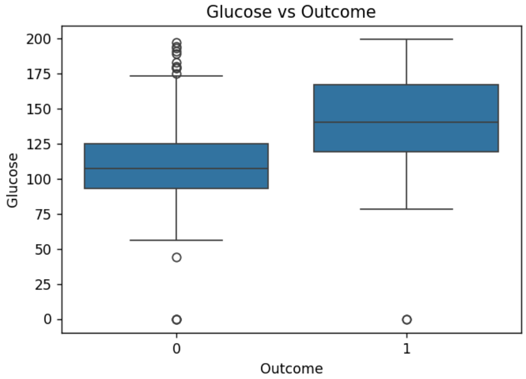

SkinThickness vs Outcome
SkinThickness
100
80
60
40
20
0
0
1
Outcome

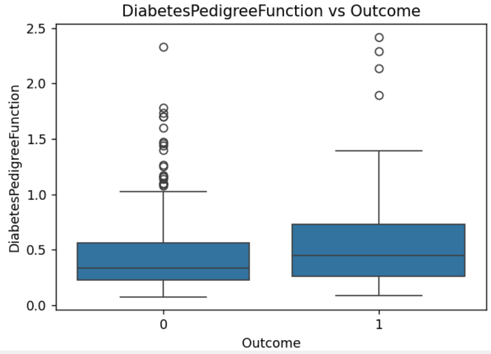

BloodPressure vs Outcome
BloodPressure
120
100
80
60
40
20
0
0
1
Outcome

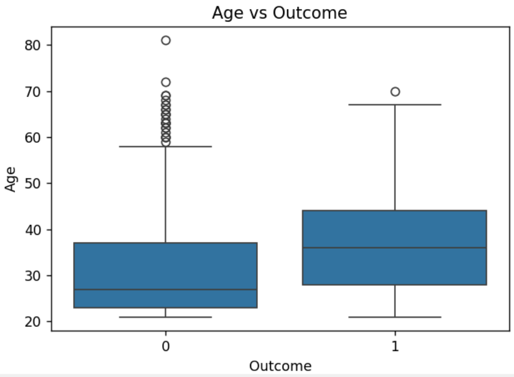

Insulin vs Outcome
Insulin
800
600
400
200
0
0
1
Outcome

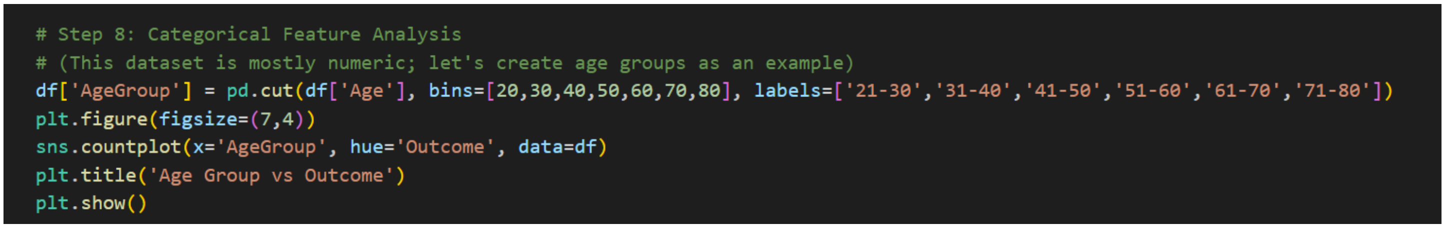

Pregnancies vs Outcome
Pregnancies
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
1
Outcome


BMI vs Outcome
BMI
70
60
50
40
30
20
10
0
0
1
Outcome

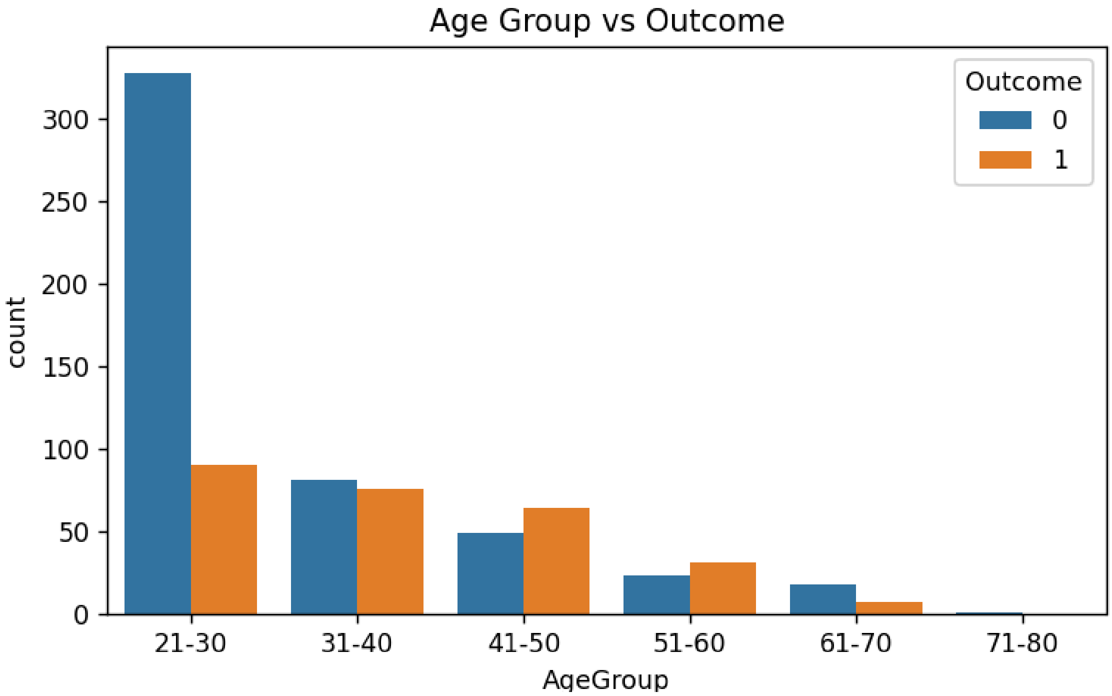

Glucose vs Outcome
Glucose
200
175
150
125
100
75
50
25
0
0
1
Outcome

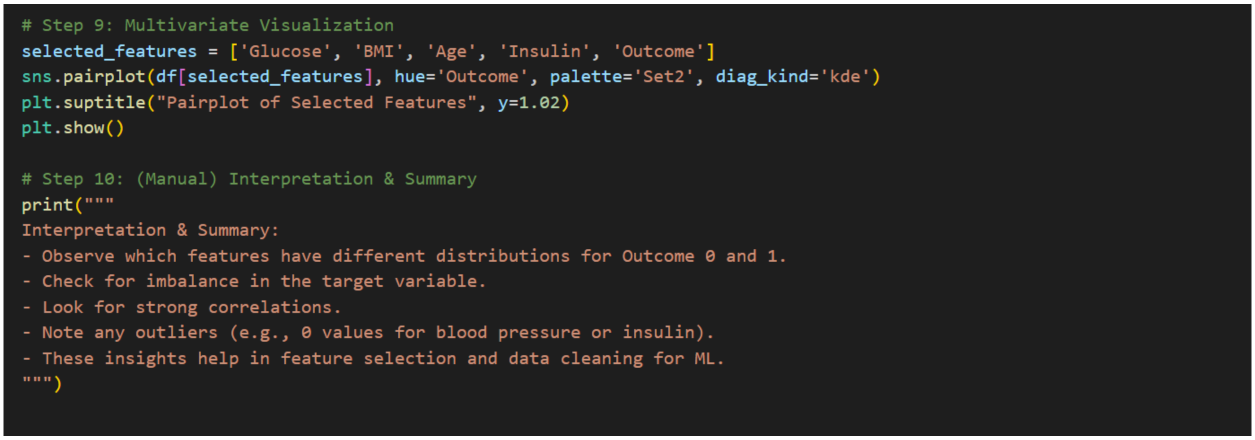

DiabetesPedigreeFunction vs Outcome
DiabetesPedigreeFunction
2.5
2.0
1.5
1.0
0.5
0.0
0
1
Outcome


Age vs Outcome
Age
80
70
60
50
40
30
20
0
1
Outcome

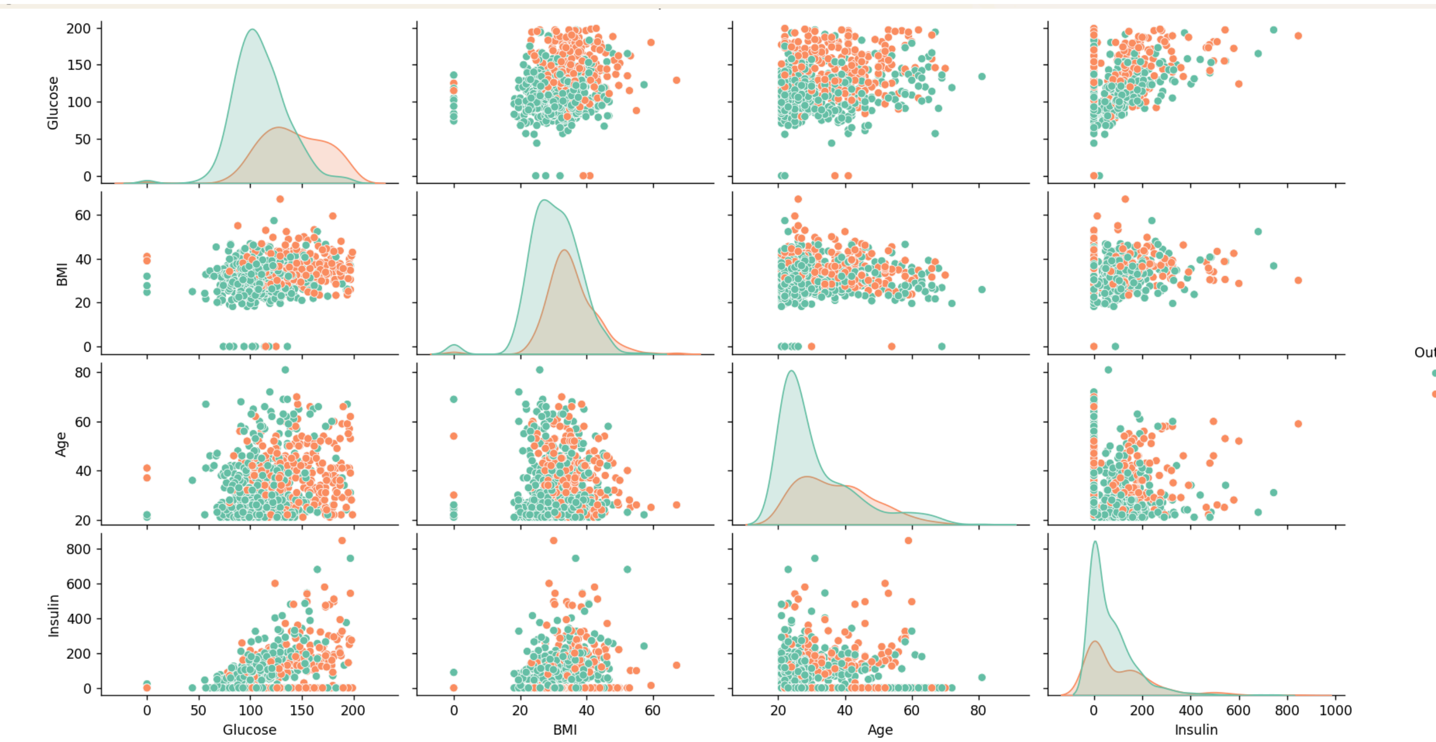


```
# Step 8: Categorical Feature Analysis
# (This dataset is mostly numeric; let's create age groups as an example)
df['AgeGroup'] = pd.cut(df['Age'], bins=[20,30,40,50,60,70,80], labels=['21-30','31-40','41-50','51-60','61-70','71-80'])
plt.figure(figsize=(7,4))
sns.countplot(x='AgeGroup', hue='Outcome', data=df)
plt.title('Age Group vs Outcome')
plt.show()
```





```
# Step 9: Multivariate Visualization
selected_features = ['Glucose', 'BMI', 'Age', 'Insulin', 'Outcome']
sns.pairplot(df[selected_features], hue='Outcome', palette='Set2', diag_kind='kde')
plt.suptitle("Pairplot of Selected Features", y=1.02)
plt.show()

# Step 10: (Manual) Interpretation & Summary
print("""
Interpretation & Summary:
- Observe which features have different distributions for Outcome 0 and 1.
- Check for imbalance in the target variable.
- Look for strong correlations.
- Note any outliers (e.g., 0 values for blood pressure or insulin).
- These insights help in feature selection and data cleaning for ML.
""")
```


Glucose
BMI
Age
Insulin
Glucose
BMI
Age
Insulin
Out