```
] import pandas as pd
  import numpy as np
```

```
0] #data = complete code
  data = pd.read_csv("/content/diabetes.csv")
  data.head()
```

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

Next steps: Generate code with data | View recommended plots | New interactive sheet

```
] print("\nData Types:")
  print(data.dtypes)
```

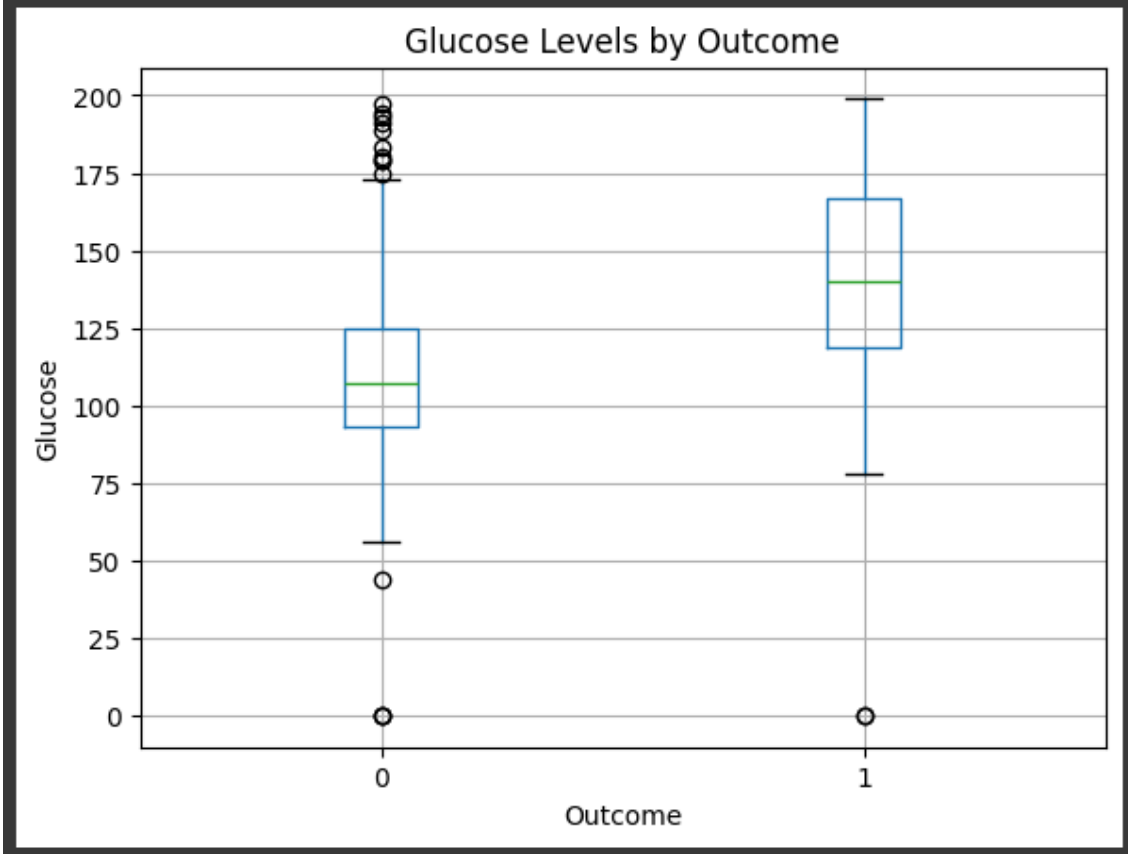
```
Data Types:
Pregnancies                 int64
Glucose                     int64
BloodPressure               int64
SkinThickness               int64
Insulin                     int64
BMI                       float64
DiabetesPedigreeFunction  float64
Age                         int64
Outcome     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

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

```python
# Summary statistics for numerical columns.

for column in data.columns:

    if data[column].isnull().sum() > 10:
        data[column].fillna(data[column].mean(), inplace=True)
    else:
        data.dropna(subset=[column], inplace=True)
print("\nSummary Statistics:\n", data.head())
print(data.describe())
print("Missing values after processing:\n", data.isnull().sum())
```

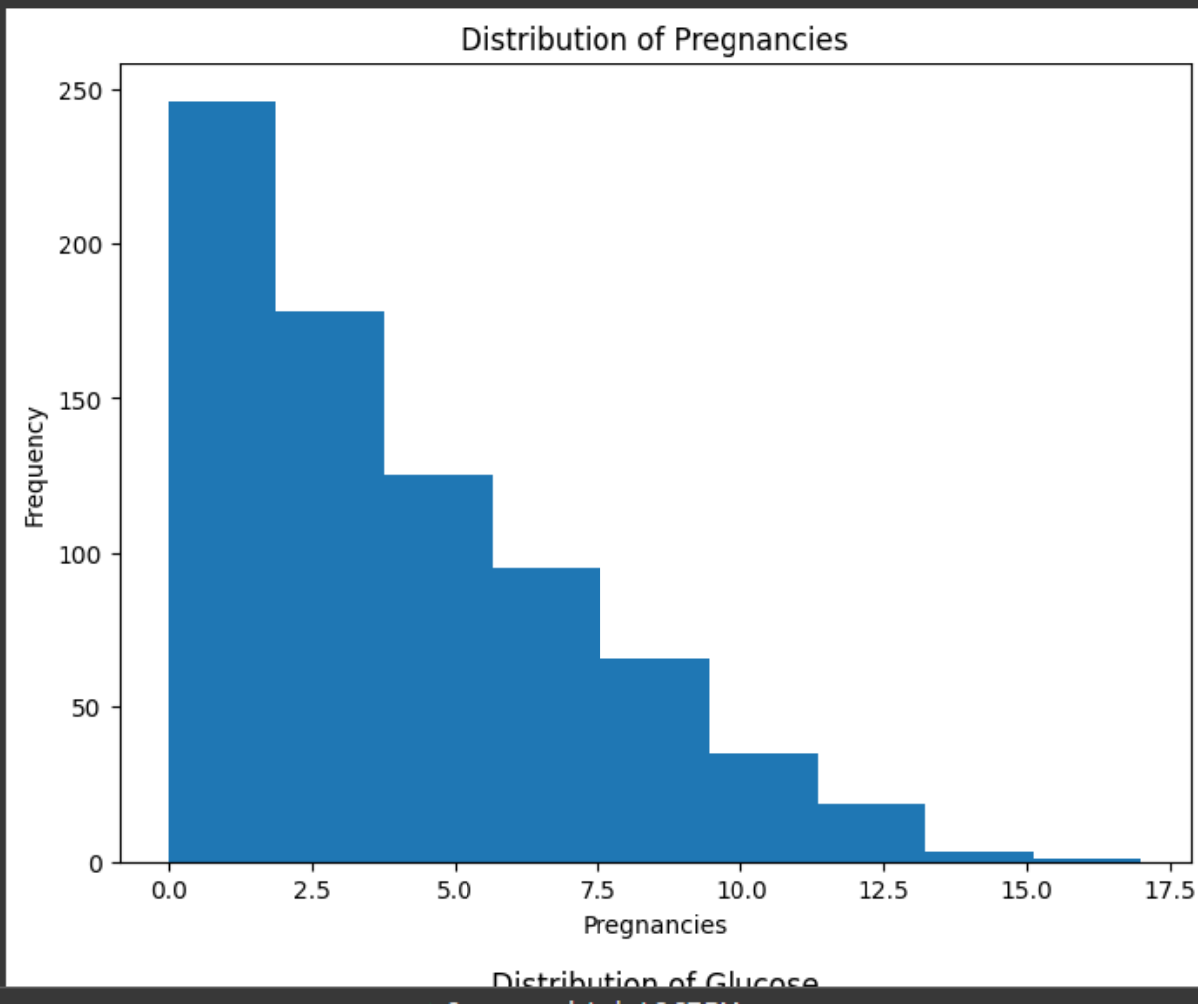
```
Summary Statistics:
    Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0            6      148             72             35        0  33.6
1            1       85             66             29        0  26.6
2            8      183             64              0        0  23.3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

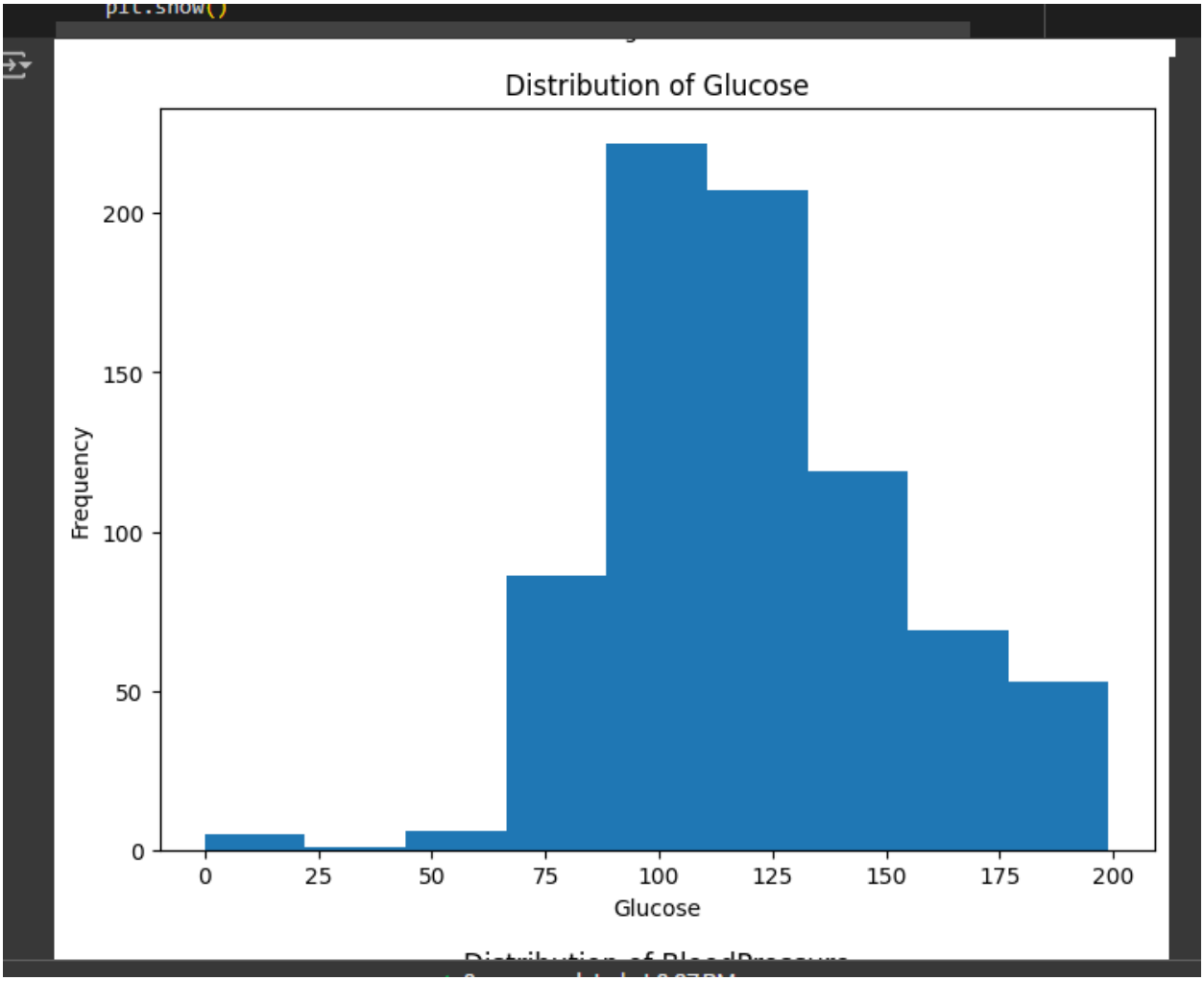
```
plt.show()
```

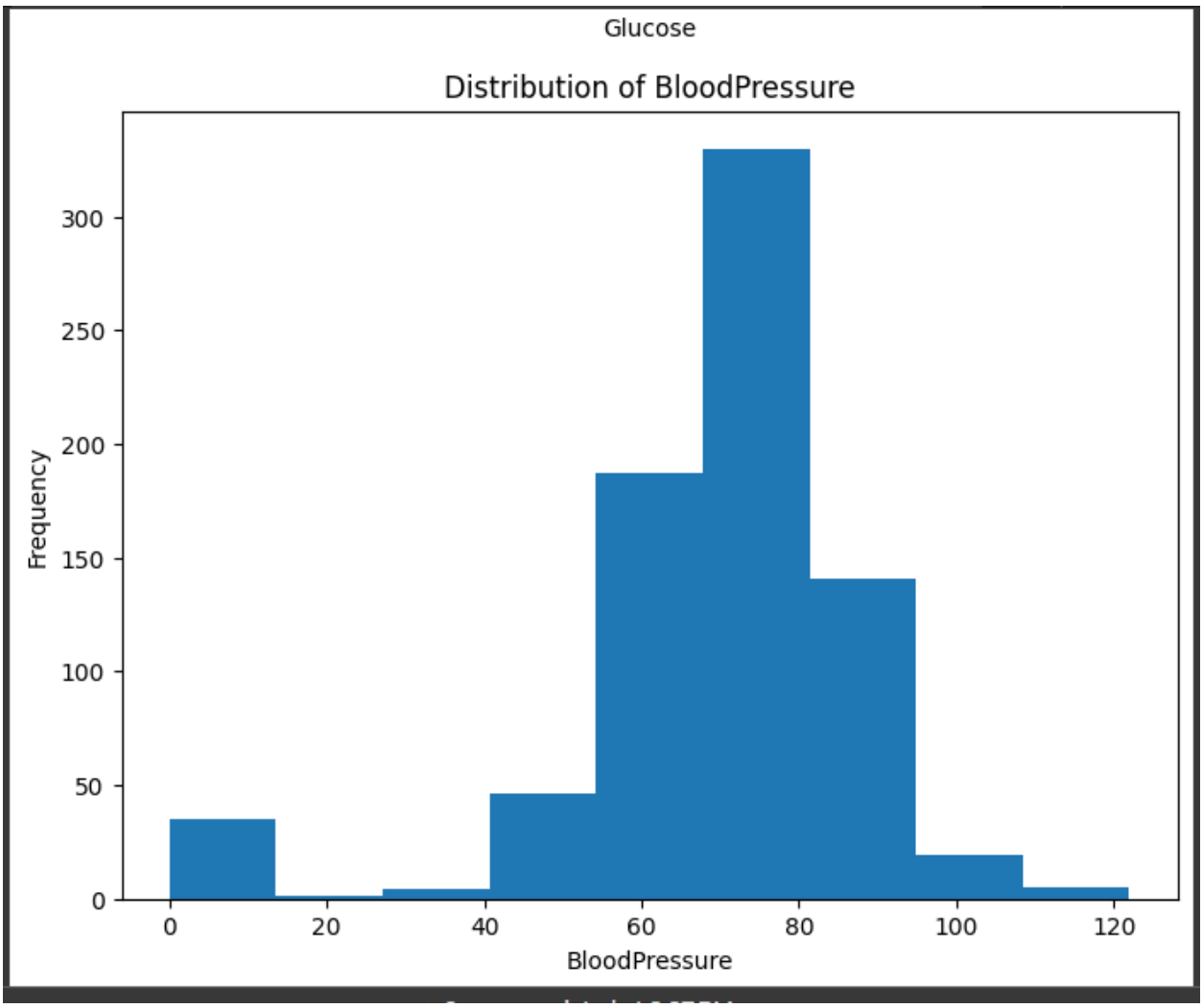


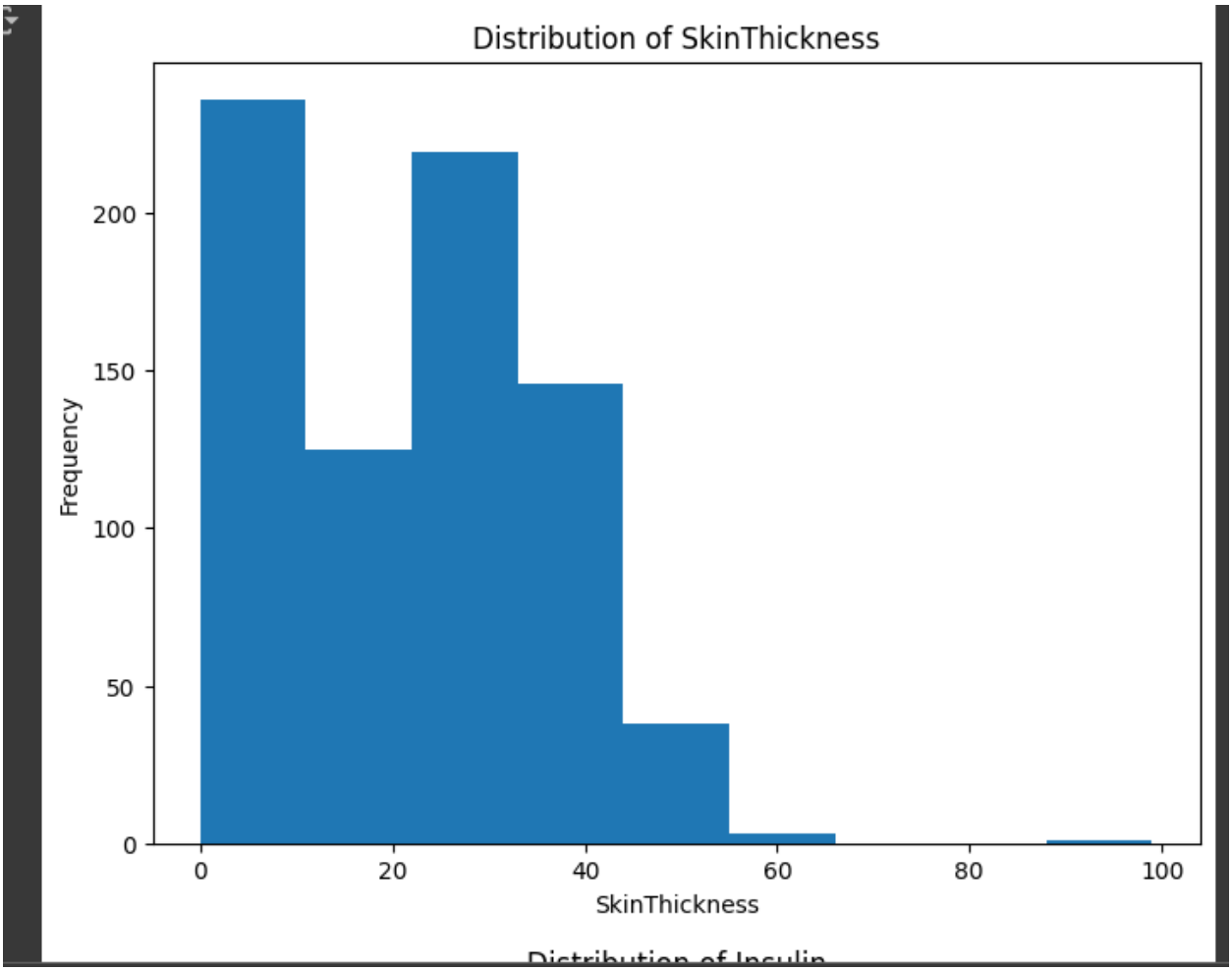

```python
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

diabetes_data = pd.read_csv("/content/diabetes.csv")

numerical_cols = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insuli
for col in numerical_cols:
    plt.figure(figsize=(8, 6))
    plt.hist(diabetes_data[col], bins=9)
    plt.title(f'Distribution of {col}')
    plt.xlabel(col)
    plt.ylabel('Frequency')
    plt.show()
```

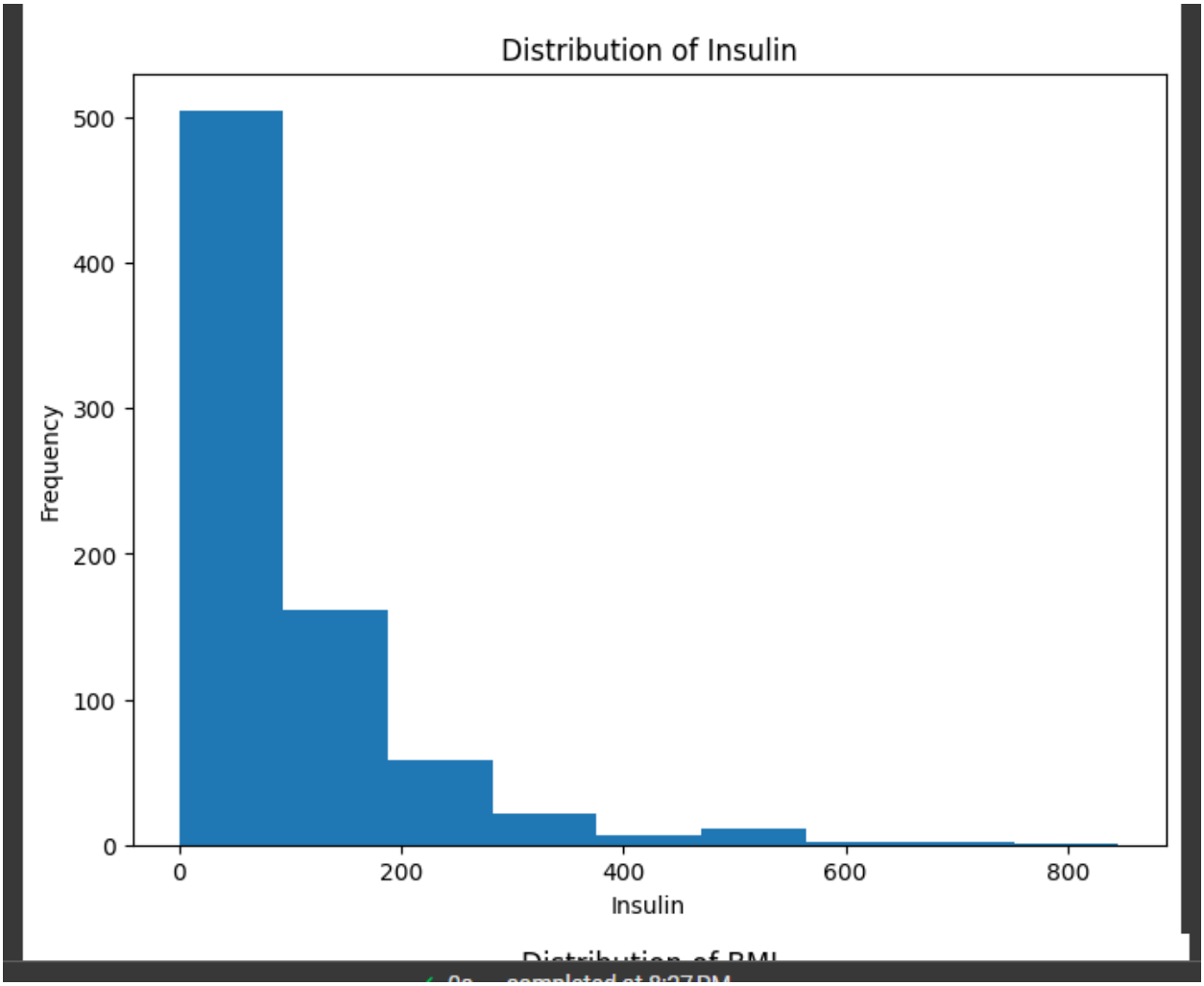
```
plt.xlabel(col)
plt.ylabel('Frequency')
plt.show()
```

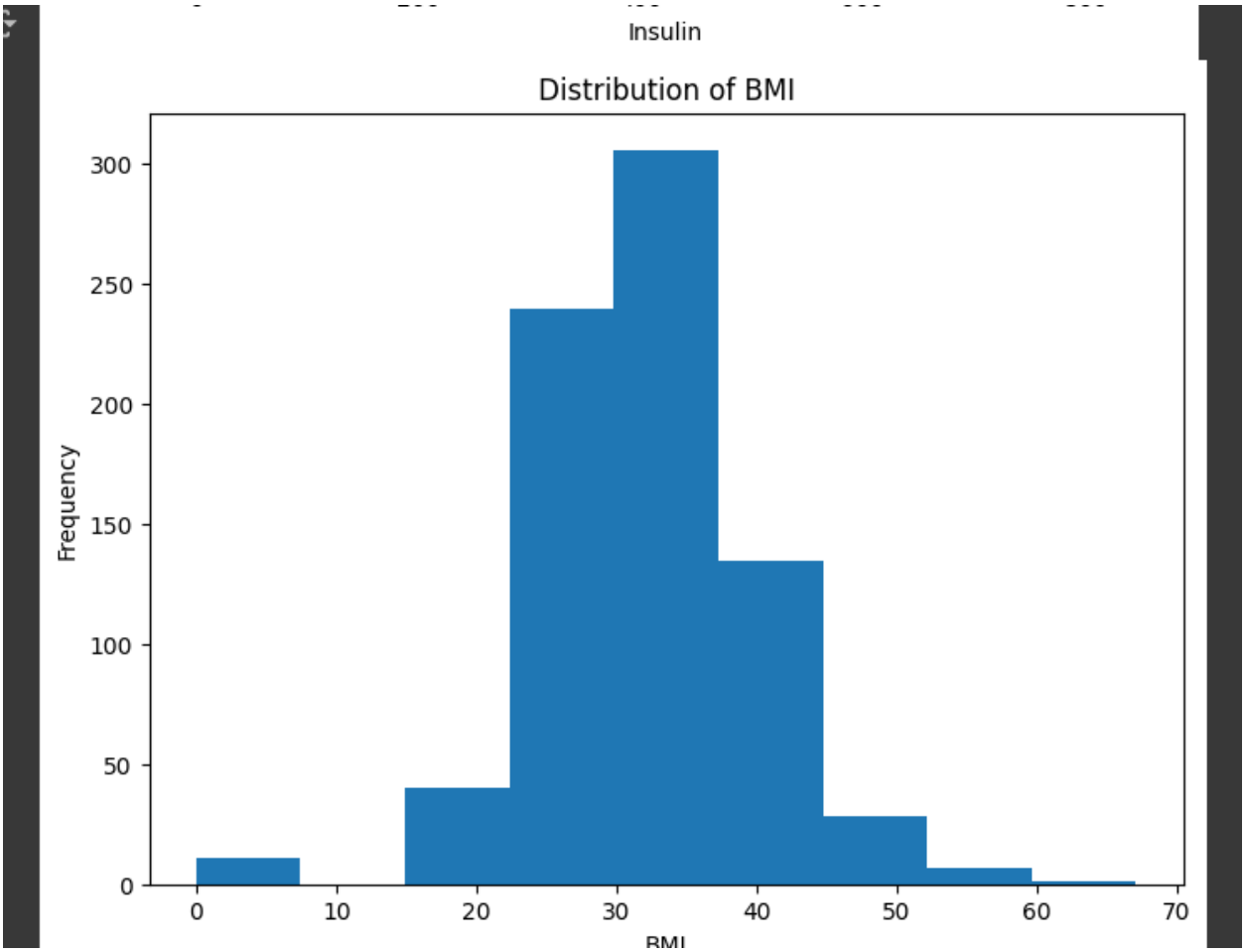


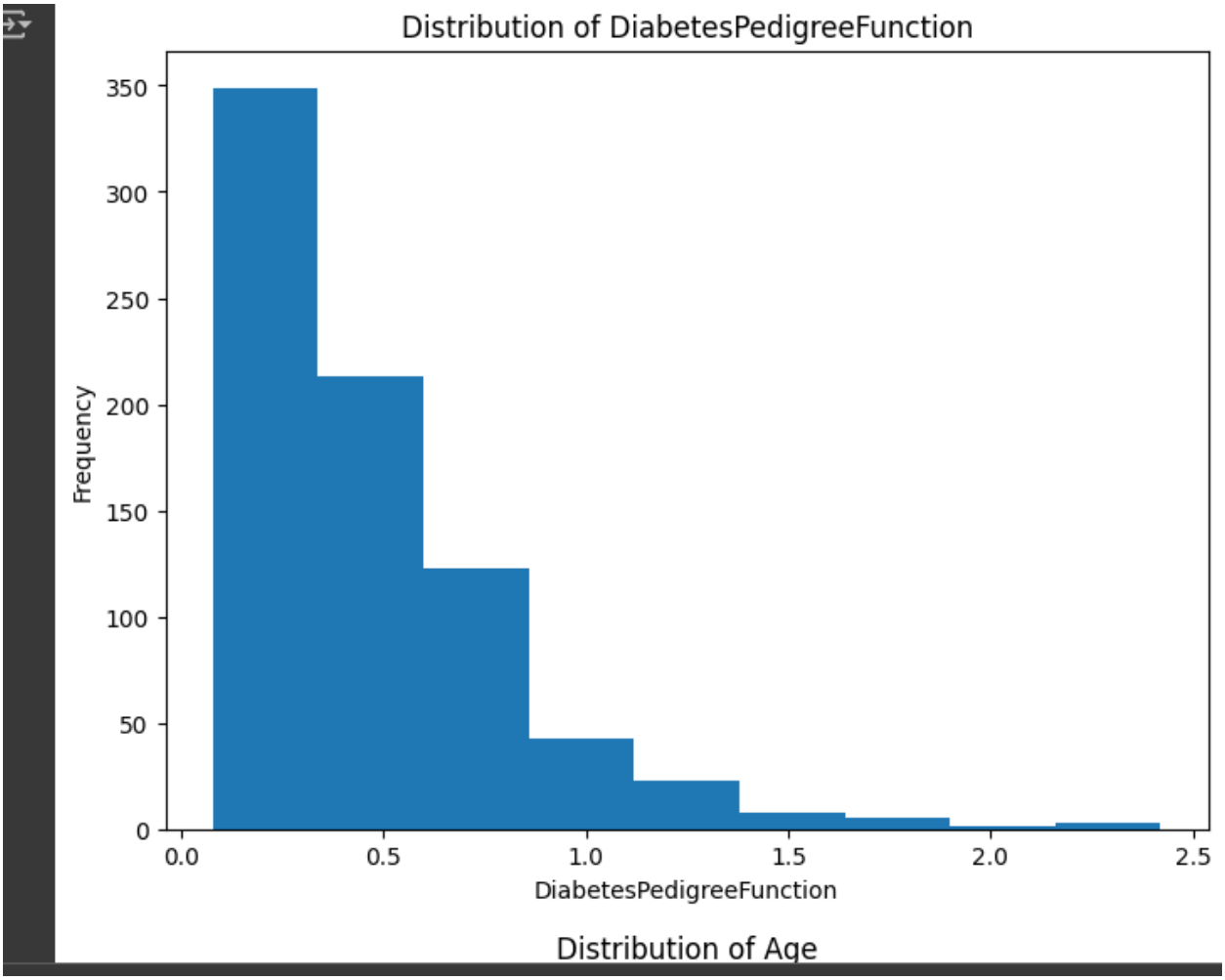

plt.show()
Distribution of Glucose
Frequency
Glucose
0
50
100
150
200
0
25
50
75
100
125
150
175
200

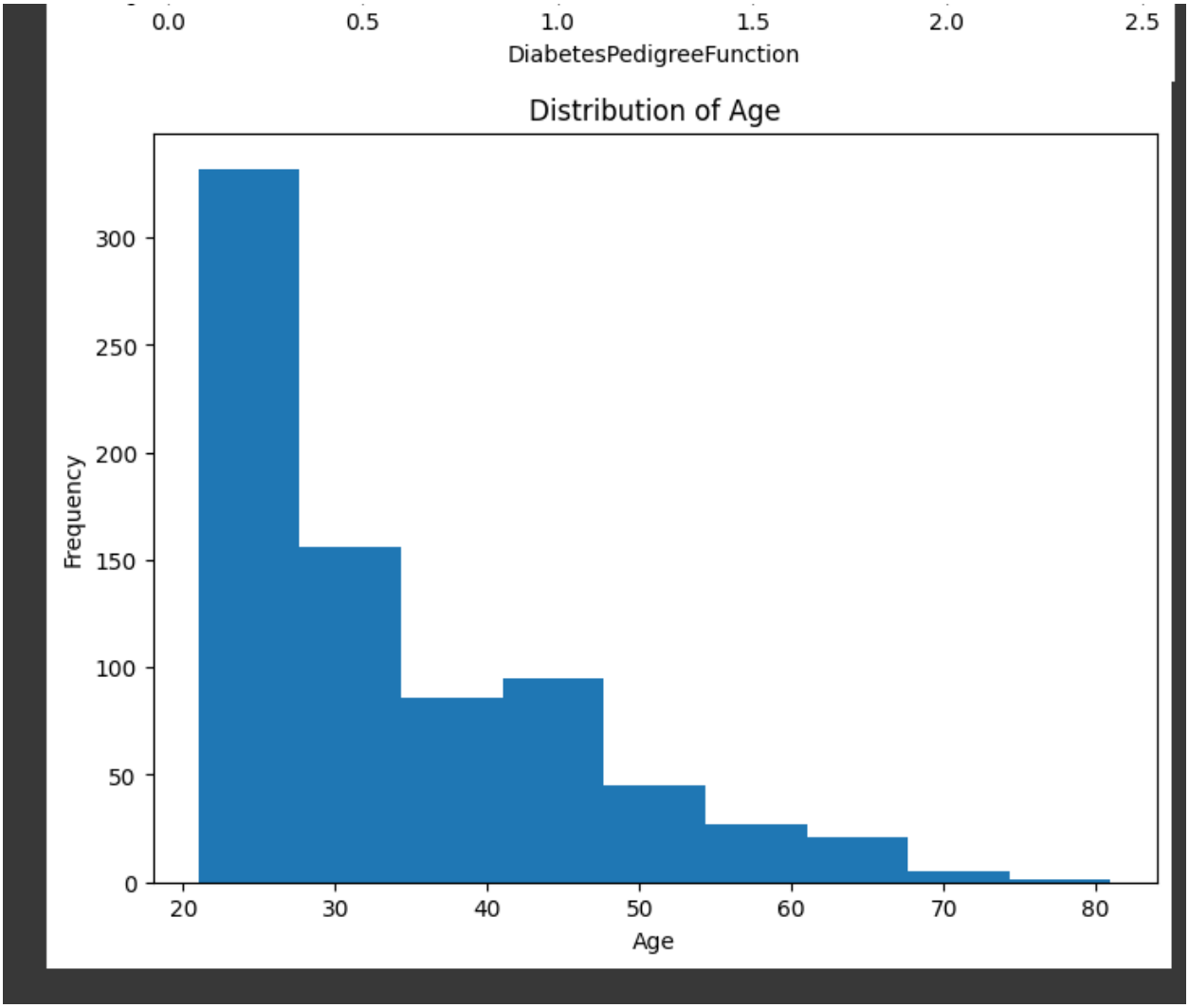

Glucose
Distribution of BloodPressure
Frequency
BloodPressure
0
50
100
150
200
250
300
0
20
40
60
80
100
120

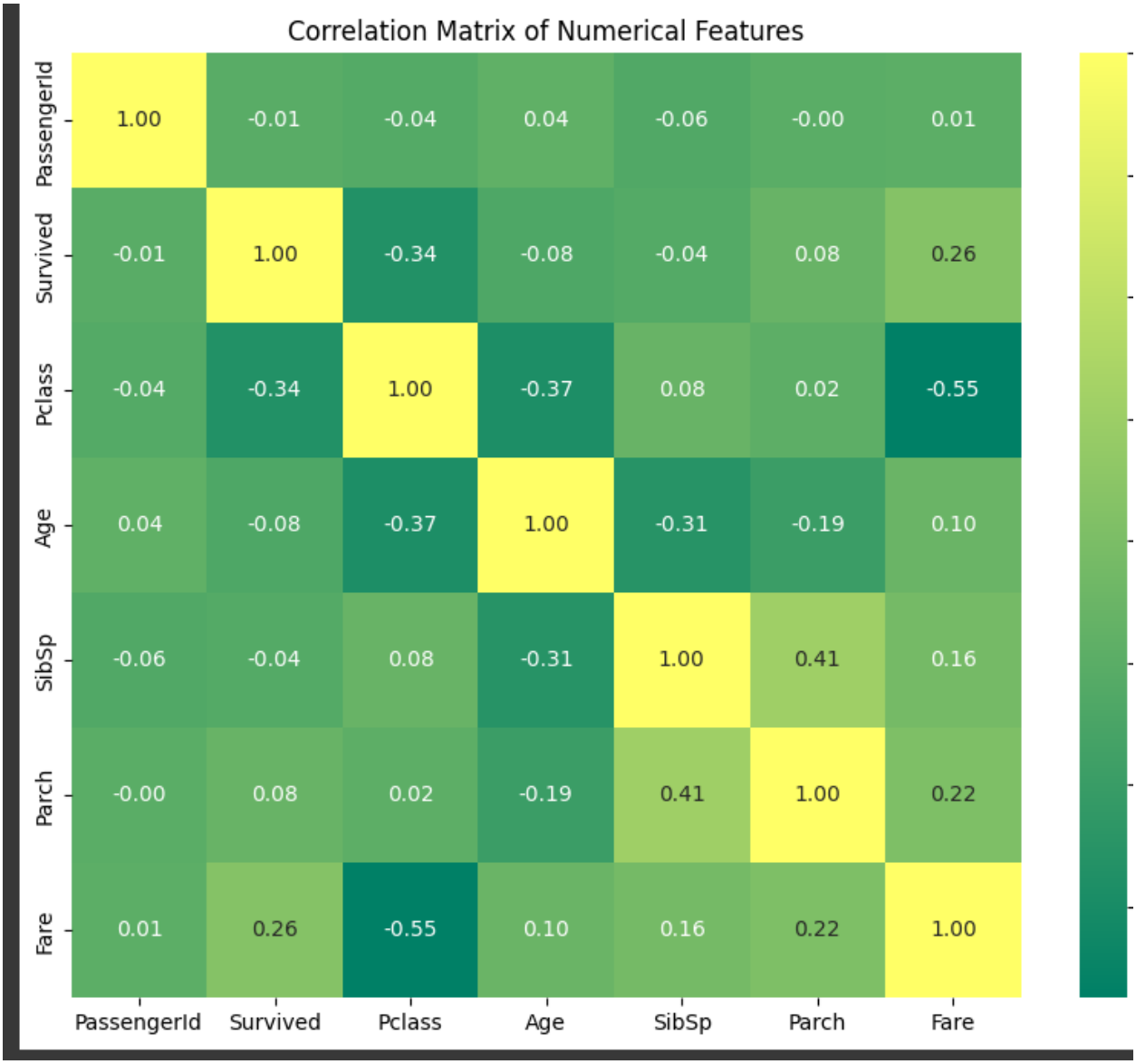

Distribution of SkinThickness
Frequency
200
150
100
50
0
0
20
40
60
80
100
SkinThickness
Distribution of Insulin


Distribution of Insulin
Frequency
Insulin
0
100
200
300
400
500
0
200
400
600
800


Insulin
Distribution of BMI
Frequency
300
250
200
150
100
50
0
0
10
20
30
40
50
60
70
BMI


Distribution of DiabetesPedigreeFunction
Frequency
DiabetesPedigreeFunction
0
50
100
150
200
250
300
350
0.0
0.5
1.0
1.5
2.0
2.5
Distribution of Age


0.0
0.5
1.0
1.5
2.0
2.5
DiabetesPedigreeFunction
Distribution of Age
Frequency
300
250
200
150
100
50
0
20
30
40
50
60
70
80
Age


Correlation Matrix of Numerical Features
PassengerId
Survived
Pclass
Age
SibSp
Parch
Fare
1.00 -0.01 -0.04 0.04 -0.06 -0.00 0.01
-0.01 1.00 -0.34 -0.08 -0.04 0.08 0.26
-0.04 -0.34 1.00 -0.37 0.08 0.02 -0.55
0.04 -0.08 -0.37 1.00 -0.31 -0.19 0.10
-0.06 -0.04 0.08 -0.31 1.00 0.41 0.16
-0.00 0.08 0.02 -0.19 0.41 1.00 0.22
0.01 0.26 -0.55 0.10 0.16 0.22 1.00
PassengerId
Survived
Pclass
Age
SibSp
Parch
Fare

```
print("Shape of X_train:", X_train.shape)
print("Shape of X_test:", X_test.shape)
print("Shape of y_train:", y_train.shape)
print("Shape of y_test:", y_test.shape)
```

```
Shape of X_train: (624, 11)
Shape of X_test: (267, 11)
Shape of y_train: (624,)
Shape of y_test: (267,)
```

```
[22] def euclidean_distance(point1, point2):
        """
        Calculate the Euclidean distance betwee

        Arguments:
        point1 : np.ndarray
            The first point as a numpy array.
        point2 : np.ndarray
            The second point as a numpy array.

        Returns:
        float
            The Euclidean distance between the

        Raises:
        ValueError: If the input points do not
        """
```

```
Test passed successfully!
```

```
4] def knn_predict_single(query, X_train, y_train, k=3)
        """
        Predict the class label for a single query using
```

```
except Exception as e:
    print(f"An unexpected error occurred during prediction or accuracy computation: {e}")
```

```
An unexpected error occurred during prediction or accuracy computation: name 'knn_predict' is not def
```

```
        print("Experiment completed. Check the plot for the accuracy trend.")
    except Exception as e:
        print(f"An unexpected error occurred during the experiment: {e}")
```

```
An unexpected error occurred during the experiment: name 'knn_predict' is not defined
```