Experiment No.1
Collect, Clean, Integrate and Transform Healthcare Data based on specific disease
Date of Performance:
Date of Submission:

Aim: Collect, Clean, Integrate and Transform Healthcare Data based on specific disease

Objective: The objective of this experiment is to perform basic pre processing on healthcare data set using python libraries

Theory:

Data Collection- Data collection is the process of gathering and measuring information from countless different sources. In order to use the data we collect to develop practical artificial intelligence (AI) and machine learning solutions, it must be collected and stored in a way that makes sense for the business problem at hand.

Data Cleaning: Cleaning data refers to the way of deleting wrong, corrupted, wrongly formatted, duplicate information, or incomplete information from a dataset. The possibility of duplicating or mislabelling data increases when two or more data sources are combined.

Data Integration: Data integration is the practice of consolidating data from disparate sources into a single dataset with the ultimate goal of providing users with consistent access and delivery of data across the spectrum of subjects and structure types, and to meet the information needs of all applications and business processes.

Data transformation: Data transformation is the process of converting, cleansing, and structuring data into a usable format that can be analyzed to support decision making processes, and to propel the growth of an organization. Data transformation is used when data needs to be converted to match that of the destination system.

Code: -

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler

# Define the file path and load the dataset
file_path = '/content/drive/My Drive/diabetes_data.csv'
data = pd.read_csv(file_path)

# Display the first few rows of the dataset
print("First few rows of the dataset:")
print(data.head())

# Check for missing values in the dataset
missing_values = data.isnull().sum()
print("\nMissing values in each column:")
print(missing_values)

# Get summary statistics for the dataset
summary_stats = data.describe()
print("\nSummary statistics:")
print(summary_stats)

# Calculate quartiles and interquartile range (IQR) for outlier detection
Q1 = data.quantile(0.25)
Q3 = data.quantile(0.75)
IQR = Q3 - Q1

# Detect outliers using the IQR method
```

```
outliers = (data < (Q1 - 1.5 * IQR)) | (data > (Q3 + 1.5 * IQR))
print("\nNumber of outliers in each column:")
print(outliers.sum())

# Scale the features using StandardScaler (excluding the 'Outcome' column)
scaler = StandardScaler()
scaled_data = scaler.fit_transform(data.drop('Outcome', axis=1))
scaled_df = pd.DataFrame(scaled_data, columns=data.columns[:-1])

# Add the 'Outcome' column back to the scaled DataFrame
scaled_df['Outcome'] = data['Outcome']

# Plot histograms for each feature to understand their distributions
print("\nDisplaying histograms for each feature:")
data.hist(bins=15, figsize=(15, 10))
plt.show()

# Plot box plots for each feature to visualize their spread and detect outliers
print("\nDisplaying box plots for each feature:")
data.plot(kind='box', subplots=True, layout=(3, 3), figsize=(15, 10), sharex=False, sharey=False)
plt.show()

# Print data types of each column to understand the data structure
print("\nData types of each column:")
print(data.dtypes)

# Plot the distribution of the 'Pregnancies' feature
print("\nDisplaying distribution plot for 'Pregnancies':")
sns.displot(data["Pregnancies"])
plt.title("Distribution of Pregnancies")
```

plt.show()

Outputs:

```
First few rows of the dataset:
   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
```

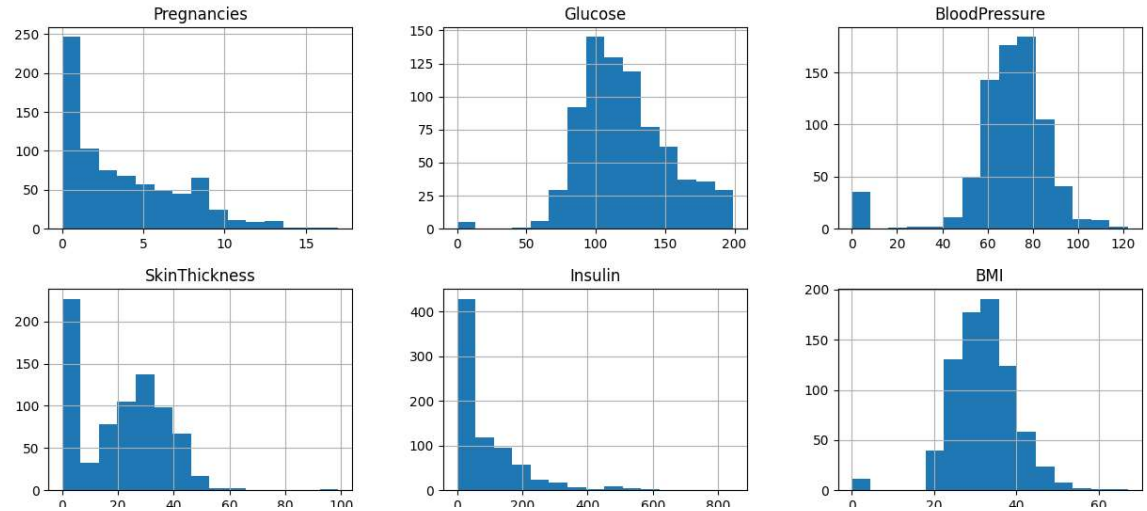
```
Missing values in each column:
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                  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

]

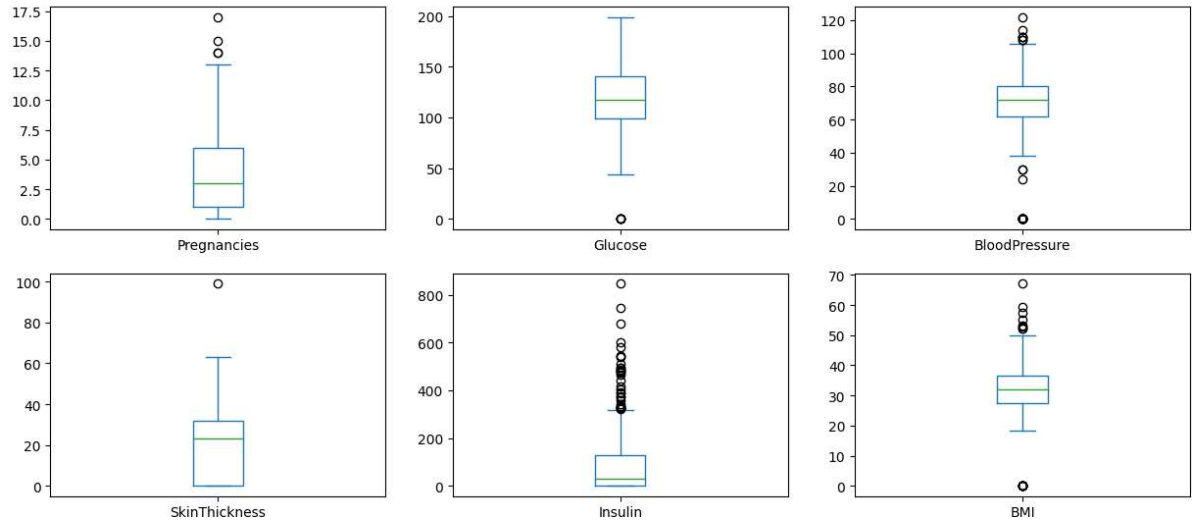
```
Summary statistics:
       Pregnancies     Glucose  BloodPressure  SkinThickness     Insulin  \
count   768.000000  768.000000     768.000000     768.000000  768.000000
mean      3.845052  120.894531      69.105469      20.536458   79.799479
std       3.369578   31.972618      19.355807      15.952218  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
```

```
Number of outliers in each column:
Pregnancies                 4
Glucose                     5
BloodPressure              45
SkinThickness               1
Insulin                    34
BMI                        19
DiabetesPedigreeFunction   29
Age                         9
Outcome                     0
dtype: int64
```

Displaying histograms for each feature:

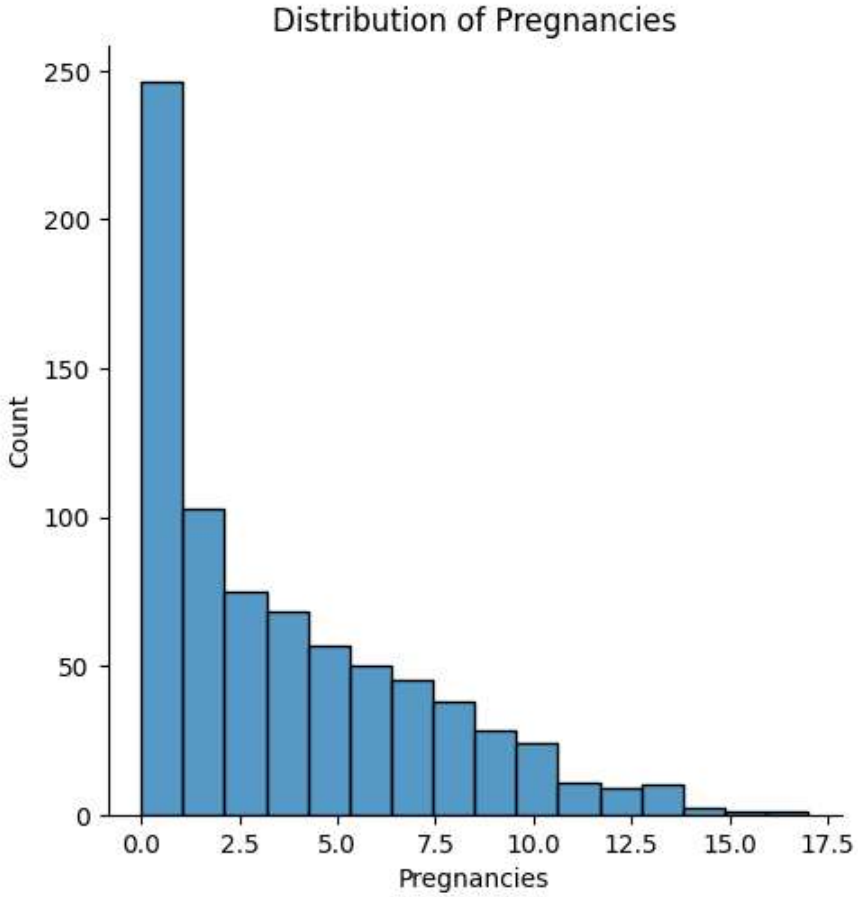


Displaying box plots for each feature:




```
Data types of each column:
Pregnancies                 int64
Glucose                     int64
BloodPressure               int64
SkinThickness               int64
Insulin                     int64
BMI                       float64
DiabetesPedigreeFunction  float64
Age                         int64
Outcome                     int64
dtype: object
```

```
Displaying distribution plot for 'Pregnancies':
```




```
Correlation between Age and BMI: 0.03624187009229416
```

Google Collaboratory Link: -

https://colab.research.google.com/drive/1XVy6T19rzyXjAsupkJVST7L5PzKkSmEk

Conclusion: -

The analysis of the diabetes dataset revealed a complete dataset with no missing values, ensuring data integrity. Summary statistics provided a detailed view of the distribution and variability of features, highlighting key measures such as mean, median, and standard deviation. Outlier detection identified several values beyond the typical range, indicating areas that may need further investigation. Feature scaling was applied to standardize the data, which is crucial for models sensitive to feature scales. Visualizations, including histograms, box plots, and distribution plots, offered valuable insights into the distribution and

relationships within the data, enhancing our understanding of the dataset's characteristics. Overall, the analysis laid a solid foundation for subsequent modeling and deeper exploration, ensuring that the data is well-prepared for further predictive analysis or machine learning applications.