PROJECT TITLE:

BUILDING A SMARTER AI BASED DIABETES PREDICTION SYSTEM

Phase 3-Development Part -1

Overview:

AI-based diabetes prediction system, data can be collected from diverse sources, such as electronic health records, wearable devices, and surveys. This data is then stored in a secure, structured database, ensuring data privacy . Additionally, cloud-based solutions can be employed for scalable and accessible storage. Proper data preprocessing techniques are applied to clean, standardize, and anonymize the information, removing any inconsistencies or personally identifiable details. The data is organized into features (e.g., glucose levels, family history) and used to train predictive models while keeping the database continually updated for real-time assessments.

Step 1: **PROJECT PLANNING AND DEFINITION**

- **Project Scope**: Clearly define the scope and objectives of the project, including the target population (e.g., adults, children), geographic region, and any specific data sources or technologies to be used.
- **Data Sources**: Identify the sources of medical data, such as electronic health records (EHRs), surveys, wearable devices, and genetic databases.
- **Regulatory Compliance**: Ensure compliance with healthcare regulations and data privacy laws (e.g., HIPAA in the United States).

Step 2: **DATA COLLECTION AND INTEGRATION**

- **Data Gathering**: Collect a diverse dataset including patient demographics, medical history, lifestyle factors, and genetic information. Collaborate with healthcare institutions and obtain necessary permissions.
- **Data Integration**: Integrate data from various sources into a unified database or data warehouse.

Step 3**: DATA PREPROCESSING**

- **Data Cleaning**: Remove duplicates, handle missing values, and address outliers in the dataset.

- **Data Transformation**: Normalize and standardize data to ensure uniformity and improve model performance.

Step 4: **EXPLORATORY DATA ANALYSIS (EDA**)

- **Feature Selection**: Use EDA to identify relevant features or variables that may influence diabetes risk. This can help streamline the modeling process by focusing on the most important factors.
- **Data Visualization**: Create data visualizations such as histograms, scatter plots, and box plots to understand data distributions and uncover patterns or trends.
- **Summary Statistics**: Calculate and examine summary statistics like mean, median, standard deviation, and quantiles to gain insights into data characteristics.
- **Correlation Analysis**: Perform correlation analysis to understand relationships between features, helping identify potential multicollinearity and influential variables.

Step 5: **DATA SPLITTING**

- **Training and Validation Sets**: Split the preprocessed dataset into training and validation sets. The training set is used to train the predictive model, while the validation set is used to assess model performance during development.

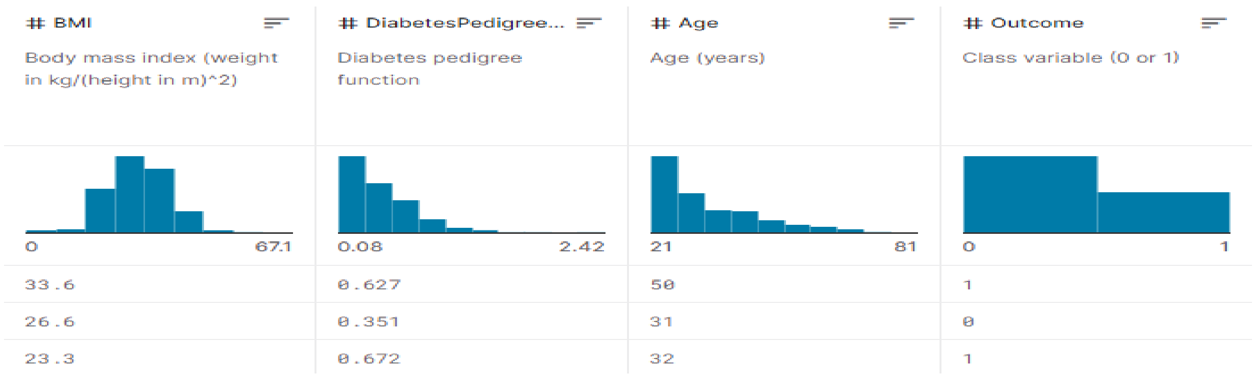
- **Test Set**: Reserve a separate test set, which will be used to evaluate the final model's performance. It should not be used for model development.
- **Cross-Validation (Optional)**: Consider using cross-validation techniques like k-fold cross-validation to assess model generalization and robustness. This involves repeatedly splitting the data into training and validation subsets to produce multiple model evaluation

INPUT DATA SET:

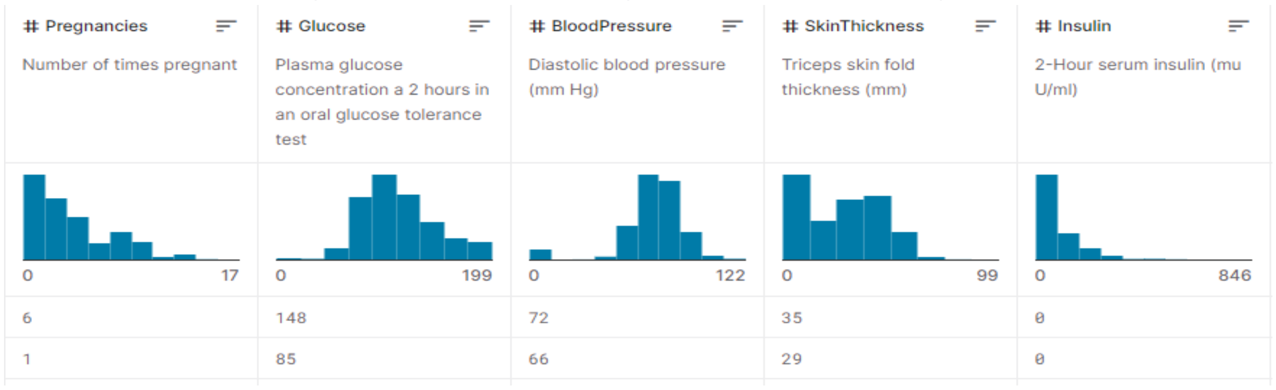
About this file

- Pregnancies: Number of times pregnant(if female)
- Glucose: Plasma glucose concentration a 2 hours in an oral glucose tolerance test
- BloodPressure: Diastolic blood pressure (mm Hg)
- SkinThickness: Triceps skin fold thickness (mm)
- Insulin: 2-Hour serum insulin (mu U/ml)
- BMI: Body mass index (weight in kg/(height in m)^2)
- DiabetesPedigreeFunction: Diabetes pedigree function
- Age: Age (years)
- Outcome: Class variable (0 or 1)

which is stored as (diabetes .csv)

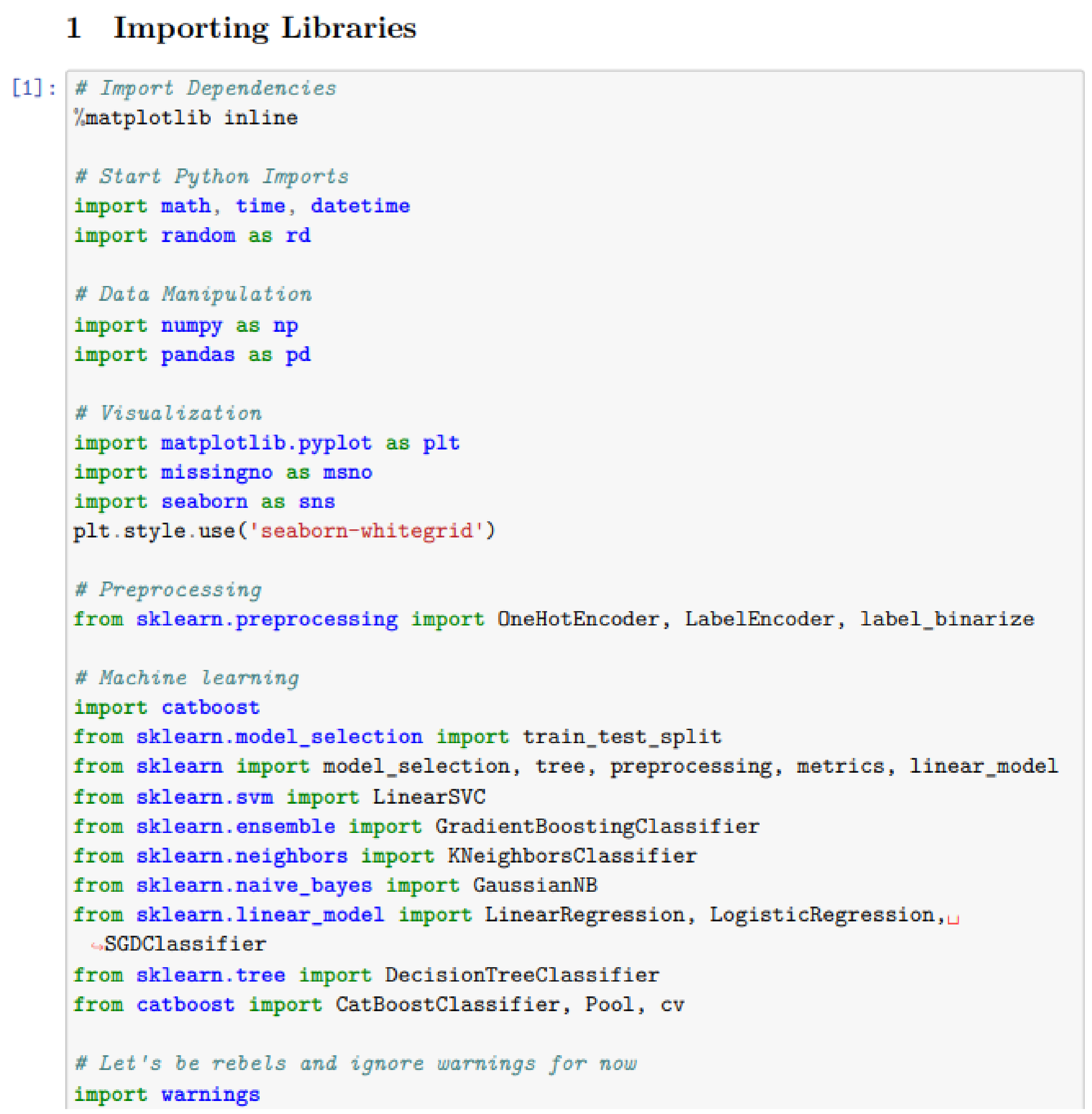


# BMI	# DiabetesPedigree...	# Age	# Outcome
Body mass index (weight in kg/(height in m)^2)	Diabetes pedigree function	Age (years)	Class variable (0 or 1)
0 – 67.1	0.08 – 2.42	21 – 81	0 – 1
33.6	0.627	50	1
26.6	0.351	31	0
23.3	0.672	32	1



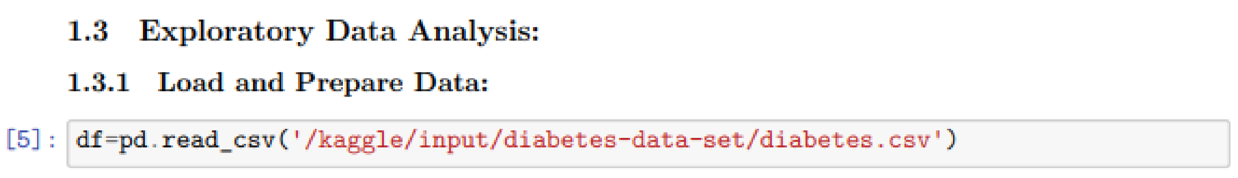
# Pregnancies	# Glucose	# BloodPressure	# SkinThickness	# Insulin
Number of times pregnant	Plasma glucose concentration a 2 hours in an oral glucose tolerance test	Diastolic blood pressure (mm Hg)	Triceps skin fold thickness (mm)	2-Hour serum insulin (mu U/ml)
0 – 17	0 – 199	0 – 122	0 – 99	0 – 846
6	148	72	35	0
1	85	66	29	0

CODING:



1 Importing Libraries

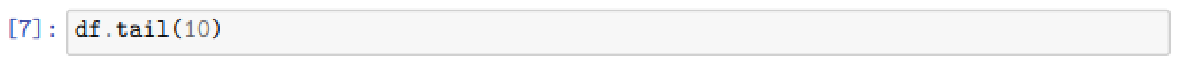
```
[1]: # Import Dependencies
     %matplotlib inline

     # Start Python Imports
     import math, time, datetime
     import random as rd

     # Data Manipulation
     import numpy as np
     import pandas as pd

     # Visualization
     import matplotlib.pyplot as plt
     import missingno as msno
     import seaborn as sns
     plt.style.use('seaborn-whitegrid')

     # Preprocessing
     from sklearn.preprocessing import OneHotEncoder, LabelEncoder, label_binarize

     # Machine learning
     import catboost
     from sklearn.model_selection import train_test_split
     from sklearn import model_selection, tree, preprocessing, metrics, linear_model
     from sklearn.svm import LinearSVC
     from sklearn.ensemble import GradientBoostingClassifier
     from sklearn.neighbors import KNeighborsClassifier
     from sklearn.naive_bayes import GaussianNB
     from sklearn.linear_model import LinearRegression, LogisticRegression,␣
       ↪SGDClassifier
     from sklearn.tree import DecisionTreeClassifier
     from catboost import CatBoostClassifier, Pool, cv

     # Let's be rebels and ignore warnings for now
     import warnings
```



1.3 Exploratory Data Analysis:

1.3.1 Load and Prepare Data:

```
[5]: df=pd.read_csv('/kaggle/input/diabetes-data-set/diabetes.csv')
```



```
[7]: df.tail(10)
```

1.3.3 Data Cleaning:

```
[14]: df.shape
```

```
[14]: (768, 9)
```

```
[15]: df=df.drop_duplicates()
```

```
[16]: df.shape
```

```
[16]: (768, 9)
```

Check null Values

```
[17]: df.isnull().sum()
```

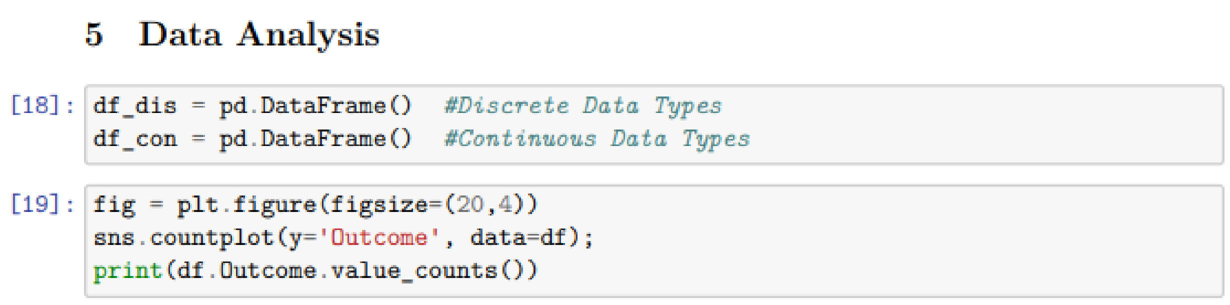
```
Missing Values :
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64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

3 Data Loading

```
[8]: df = pd.read_csv("/kaggle/input/diabetes-data-set/diabetes.csv")
```

4 Data Inspection

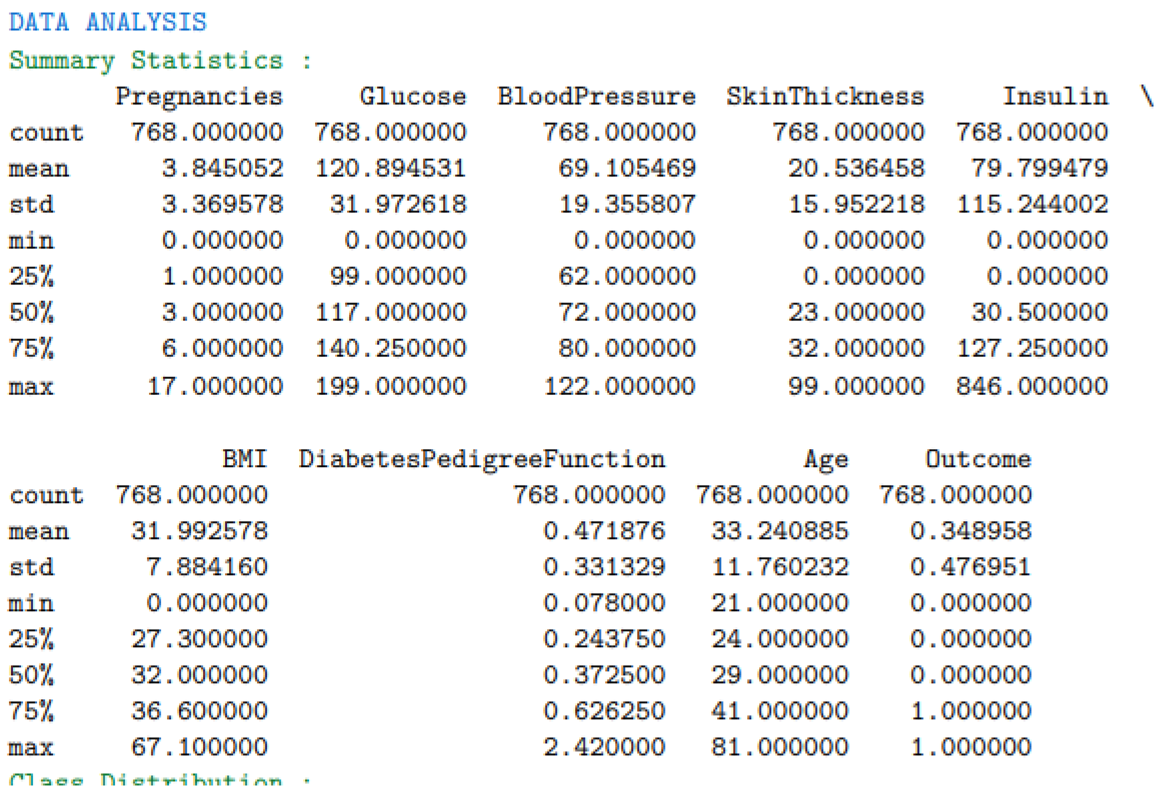
```
[9]: systematic_sample(df, 5)
```



5 Data Analysis

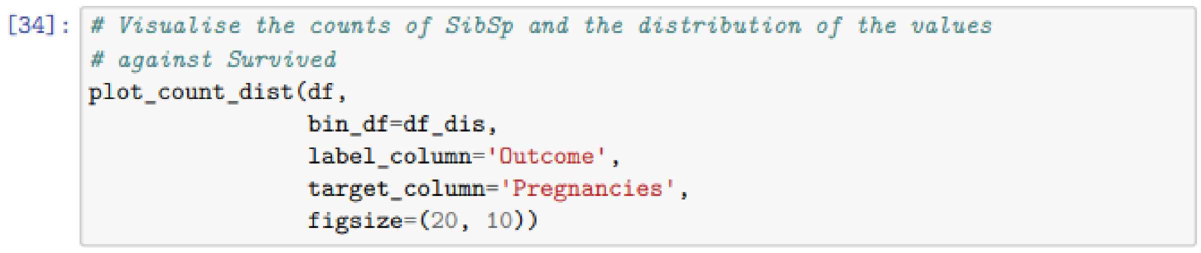
```
[18]: df_dis = pd.DataFrame()   #Discrete Data Types
      df_con = pd.DataFrame()   #Continuous Data Types
```

```
[19]: fig = plt.figure(figsize=(20,4))
      sns.countplot(y='Outcome', data=df);
      print(df.Outcome.value_counts())
```

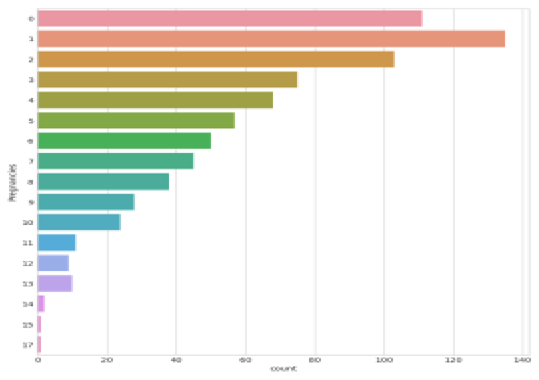


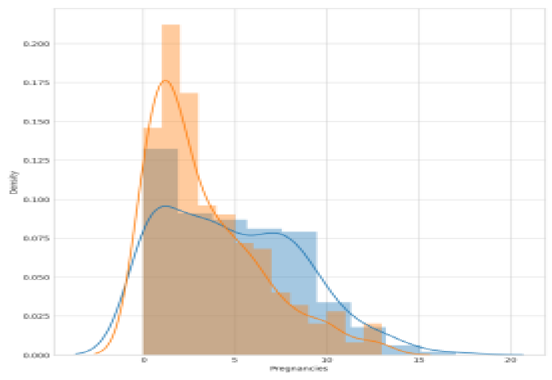
```
DATA ANALYSIS
Summary Statistics :
       Pregnancies     Glucose  BloodPressure  SkinThickness     Insulin  \
count   768.000000  768.000000     768.000000     768.000000  768.000000
mean      3.845052  120.894531      69.105469      20.536458   79.799479
std       3.369578   31.972618      19.355807      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
Class Distribution :
```

Data visualization:

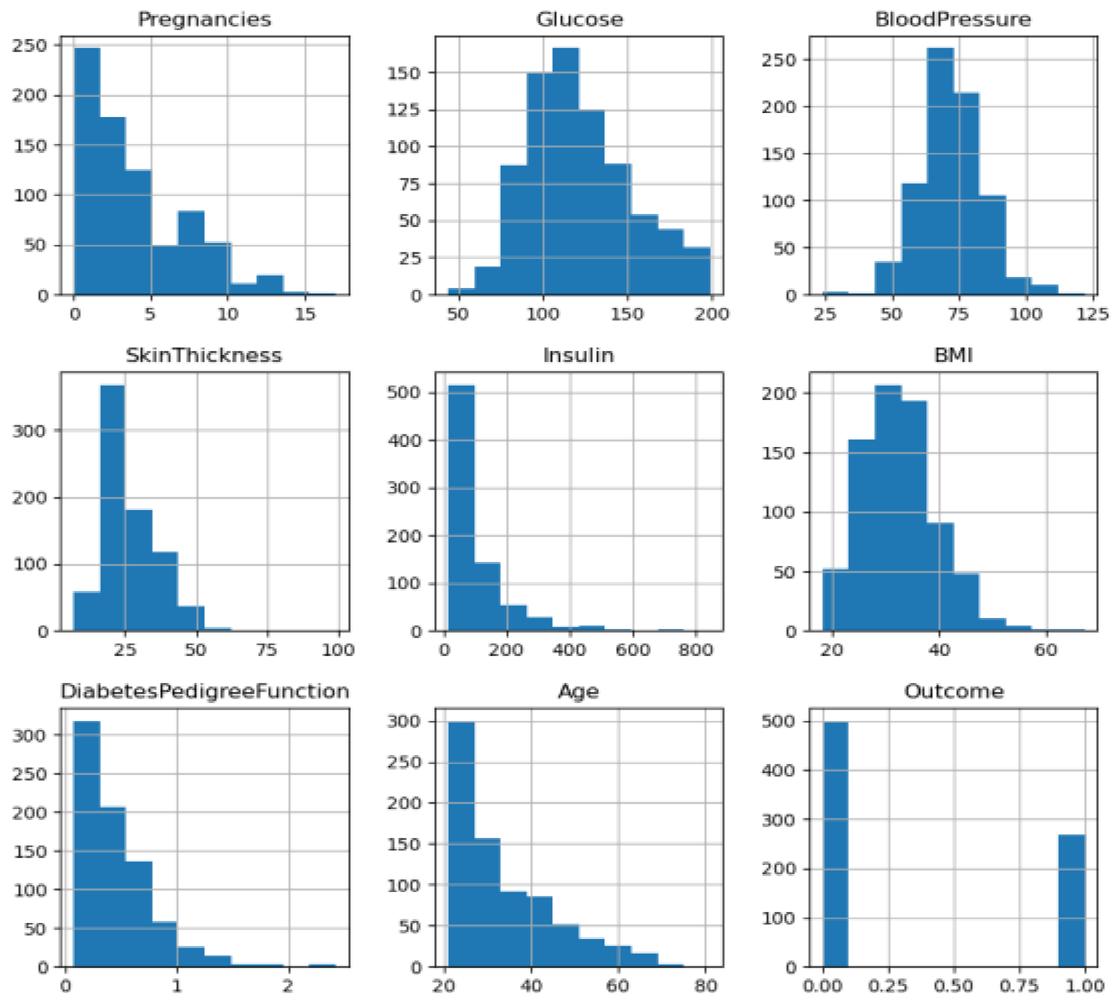


```
[34]: # Visualise the counts of SibSp and the distribution of the values
      # against Survived
      plot_count_dist(df,
                      bin_df=df_dis,
                      label_column='Outcome',
                      target_column='Pregnancies',
                      figsize=(20, 10))
```





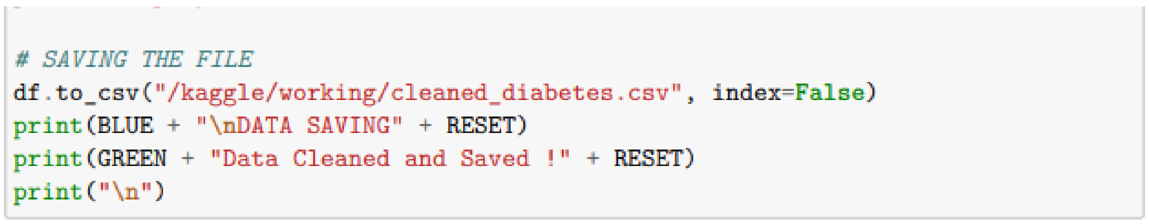
Like wise we can check visualize outcome for Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age and so on.



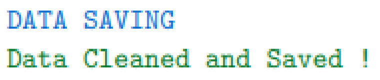
Training the data set:

```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
# Load your preprocessed dataset
data = pd.read_csv(‘diabetes.csv')
# Separate features (X) and target variable (y)
X = data.drop('diabetes_label', axis=1)  # Adjust the target variable name
y = data['diabetes_label']
# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
# Standardize the feature data
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
print(f'Accuracy: {accuracy}')
print(f'Confusion Matrix:\n{confusion}')
print(f'Classification Report:\n{report}')
```

we have ,to save the data set with (.csv)file extension.



```
# SAVING THE FILE
df.to_csv("/kaggle/working/cleaned_diabetes.csv", index=False)
print(BLUE + "\nDATA SAVING" + RESET)
print(GREEN + "Data Cleaned and Saved !" + RESET)
print("\n")
```



```
DATA SAVING
Data Cleaned and Saved !
```