PREDICTING DIABETES IN PIMA INDIAN WOMEN USING MACHINE LEARNING MODELS

Capstone Project-I submitted to

Imarticus Learning

For the Completion of the

DATA SCIENCE COURSE

by

Dr. DILEEP KUMAR SHETTY

Under the Guidance of

Arun Upadhya

Data Science Course Trainer

Imarticus Learning

DECLARATION

I, **Dr.Dileep Kumar Shetty** hereby declare that the matter embodied in this project titled, **“Predicting Diabetes in Pima Indian Women using Machine Learning Models”**, is the result of my investigation and this project has been composed under the guidance and supervision of **Arun Upadhya**, Data Science Course Trainer, Imarticus Learning. I confirm that the same has not previously formed the basis for the award of any other degree.

Date: 23.12.2023
Place: Bengaluru (Dr. Dileep Kumar Shetty)

ACKNOWLEDGMENTS

I would like to express my deep-felt gratitude to my guide, Arun Upadhya, Data Science Course Trainer at Imarticus Learning, for his advice, encouragement, enduring patience, and constant support. I thank Imarticus Learning for the opportunity. I extend my heartfelt thanks to Mr. Utkarsh at Imarticus Learning for his support and continuous updates on the course.

CONTENTS

CHAPTER -1

INTRODUCTION

1.1 ABSTRACT

Diabetes is a leading cause of death worldwide, with recent statistics highlighting India as the global diabetes capital. Extensive medical research is underway to find a cure for patients affected by diabetes, although success has not been achieved thus far. Despite the absence of a cure, significant research efforts have been dedicated to managing the disease effectively. This project explores whether specific characteristics or factors in the studied population are associated with a higher likelihood of diabetes using machine learning models. This exploration aims to contribute to the development of targeted interventions or treatments.

1.2 ABOUT THE DATA

A population of women who were at least 21 years old, of Pima Indian heritage and living near Phoenix, Arizona, was tested for diabetes according to World Health Organization criteria. The data were collected by the US National Institute of Diabetes, Digestive and Kidney Diseases. Some values are not within the expected range and should be treated as missing values. What method would be more effective for filling this type of missing value? Additionally, how should further classification proceed?

npreg: Number of times pregnant

glu: Plasma glucose concentration 2 hours in an oral glucose tolerance test

bp: Diastolic blood pressure (mm Hg)

skin: Skin Thickness(Triceps skin fold thickness (mm))

insulin: 2-Hour serum insulin (mu U/ml)

bmi: Body mass index (weight in kg/(height in m)^2)

ped: Diabetes pedigree function

age: Age (years)

type –yes or no for diabetics according to WHO criteria

1.3 OBJECTIVES

This project aims to examine the factors influencing diabetes among Pima Indian women. The main objectives are as follows:

i. To analyze the data on Pima Indian women using various ML models.

ii. To explore whether specific characteristics or factors within the studied population are linked to a higher likelihood of having diabetes. Identifying such sub-groups is valuable for understanding risk factors and tailoring interventions or treatments more effectively.

1.4 SOFTWARE USED

For data analysis and representation, we utilized Python software version 3.11.

1.5 ORGANIZATION OF THE PROJECT

Chapter 1 motivates the present project. The statistical techniques used for data analysis are presented in Chapter 2. Chapter 3 details the results obtained using various classification & regression models and cluster analysis. Chapter 4 concludes the project based on the results obtained from the diabetes data.

CHAPTER-2

METHODOLOGY

2.1 DESCRIPTIVE STATISTICS

Descriptive statistics are employed to characterize the primary features of a dataset in quantitative terms. They differ from inferential statistics (or inductive statistics) in that descriptive statistics aim to provide a quantitative summary of a dataset, rather than being used to support inferential statements about the population that the data are thought to represent. Even when data analysis draws its primary conclusions using inductive statistical analysis, descriptive statistics are typically presented alongside it for a comprehensive analysis.

2.2 Box plot:

To obtain an initial overview of the data and detect unusual behavior, graphical representations like quantile plots are used. One such method for summary display is the Box-plot, which provides information on the position of the median (marker inside the box). The median line's location can suggest skewness in the distribution, with noticeable shifts indicating positive or negative skew. The 25th and 75th percentiles are represented by marks at the lowest and uppermost edges of the box, respectively. Outliers, observed outside these markers, may appear as points, circles, or asterisks and indicate extreme values significantly deviating from the rest of the sample.

2.3 Jarque-Bera test

The Jarque-Bera test is a statistical test used to assess whether a given dataset follows a normal distribution. It is named after its developers, Carlos Jarque and Anil K. Bera. The test is based on the skewness and kurtosis of the data. In a normal distribution: Skewness is expected to be close to zero. Kurtosis is expected to be close to 3 (for a normal distribution, excess kurtosis is 0).

The Jarque-Bera test calculates a test statistic that combines skewness and kurtosis, comparing it to a chi-squared distribution with 2 degrees of freedom. The null hypothesis of the test is that the data comes from a normal distribution. Therefore, a low Jarque-Bera test statistic suggests that there is no significant departure from normality, and the data can be considered approximately normally distributed.

If the p-value associated with the Jarque-Bera test is less than a chosen significance level (e.g., 0.05), the null hypothesis is rejected, indicating that the data significantly deviates from a normal distribution.

In summary, the Jarque-Bera test is a tool for assessing the normality of a dataset based on its skewness and kurtosis, providing insights into the distributional characteristics of the data.

2.4 Q-Q Plot

A Q-Q plot, or quantile-quantile plot, is a graphical tool used in statistics to assess whether a dataset follows a particular theoretical distribution, such as the normal distribution. The Q-Q plot compares the quantiles of the observed data against the quantiles of the expected distribution.

- Ordered Data: Arrange the data in ascending order.
- Theoretical Quantiles: For each data point, calculate its expected quantile based on the chosen theoretical distribution (e.g., normal distribution).

- Plotting: Plot the ordered data points against their corresponding theoretical quantiles.
- Reference Line: If the data perfectly follows the theoretical distribution, the points on the Q-Q plot would lie along a straight line. This line is often referred to as the "diagonal" or "reference" line.
- Key Interpretations: Straight Line: If the points closely align with the reference line, it suggests that the data fits the chosen distribution well.
- Deviation from Line: Departures from the reference line indicate deviations from the assumed distribution.

2.5 CORRELATION

Correlation is a statistical measure that shows how two or more variables change together. A positive correlation means the variables increase or decrease together, while a negative correlation means one variable increases as the other decreases. A correlation coefficient quantifies the degree to which changes in one variable predict changes in another. It doesn't imply causation; just because variables fluctuate together doesn't mean one causes the other. The covariance of (X, Y) is calculated by $\mathrm{cov}(X, Y) = E([X-E(X)][Y-E(Y)])$, and assuming positive variances, the correlation is given by $\mathrm{Cor}(X, Y) = \mathrm{cov}(X, Y) / (\mathrm{sd}(X) * \mathrm{sd}(Y))$, where sd is the standard deviation.

2.6 MULTIPLE LINEAR REGRESSION

In regression analysis, we explore the relationship, termed the regression function, between a variable Y (the dependent variable) and several others X_i (the independent variables). The regression function involves a set of unknown parameters β_i. If the regression function is linear in the parameters (though not necessarily in the independent variables), it is called a linear regression model. Otherwise, it is termed non-linear. Linear regression models with more than one independent variable are known as multiple linear models, in contrast to simple linear models with just one independent variable. Multiple linear regression involves one response variable Y (regressand) and two or more explanatory variables X1, X_2 ,…,X_k (regressors). The statistical analysis aims to determine the extent of dependence of the regressand on the regressors. It's important to note that regression is a method for fitting a line to data to understand the relationship between the response variable (dependent variable Y) and the independent variables (X).

The model representing multiple linear regression is given by:

$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k + \varepsilon$ Here, β_0 is the intercept term, and β_1 ,β_2 , ,…,β_k are the regression coefficients corresponding to each regressor. The errors ε's are distributed as N(0,σ^2).

2.7 BACKWARD ELIMINATION IN MODEL BUILDING

Backward elimination in multiple linear regressions involves starting with a model that includes all variables, removing the least significant variable iteratively, and re-fitting until all remaining variables are statistically significant. Unlike forward selection, it avoids rendering previously included variables non-significant. However, drawbacks include the risk of discarding variables that could be significant in the final model, suggesting a need for a compromise between forward and backward selection methods.

2.8 LOGISTIC REGRESSION

Logistic regression is used when the target variable is categorical with two categories. It's suitable for analyzing a dichotomous (binary) dependent variable. Unlike linear regression, logistic regression deals with predicting probabilities rather than continuous values. In logistic regression, the model predicts the probability of an event occurring based on one or more independent variables. The logistic model is particularly useful when the outcome variable is binary, and it estimates the probability of the event happening given the values of the predictors.

The logistic regression model is expressed as:

$p(x) = \frac{e^{\beta_0+\beta_1 X_1+\beta_2 X_2+...+\beta_k X_k}}{1+e^{\beta_0+\beta_1 X_1+\beta_2 X_2+...+\beta_k X_k}}$, where p(x) represents the probability of the event happening given the predictors value x.

The logit transformation $(g(x) = \log(\frac{p(x)}{1-p(x)}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k)$ is crucial for this model, providing properties similar to linear regression. To estimate the model parameters, the maximum likelihood method is employed. Likelihood ratio tests help assess the significance of coefficients, comparing the fitted model to a saturated model with as many parameters as data points.

The odds ratio (O.R) is a key concept in interpreting logistic regression results. It signifies the relative change in odds for an outcome with a one-unit increase in the predictor variable. The logistic regression model is a valuable tool for predicting probabilities and understanding the relationships between variables in a binary outcome context.

2.9 Different Techniques Used for Model Evaluation

i. **Cross-Validation:** Cross-validation is a more robust technique where the dataset is divided into multiple subsets or "folds." The model is trained and tested multiple times, and the results are averaged. This helps provide a more accurate estimate of a model's performance and reduces the risk of overfitting.

ii. **Confusion Matrix:** A confusion matrix is used for classification problems. It provides a clear summary of how many true positives, true negatives, false positives, and false negatives the model produced. From the confusion matrix, we can calculate various metrics like accuracy, precision, recall, and F1 score.

iii. **ROC and AUC:** Receiver Operating Characteristic (ROC) curves and Area Under the Curve (AUC) are used to evaluate the performance of binary classification models. They show how well the model distinguishes between classes, with AUC summarizing the overall performance.

iv. **Mean Absolute Error (MAE) and Mean Squared Error (MSE)**: These metrics are used to evaluate regression models. MAE measures the average absolute difference between predicted and actual values, while MSE measures the average squared difference. Lower values indicate better performance.

v. **R-squared (R2):** R-squared is another metric for regression models. It quantifies the proportion of the variance in the dependent variable that is predictable from the independent variables. A higher R-squared value indicates a better fit.

vi. **Adjusted R-squared:** An adjusted version of R-squared that accounts for the number of predictors in a regression model, helping to avoid overfitting.

2.10 Cluster Analysis:

Cluster analysis is a technique in data analysis and statistics that involves grouping similar data points into clusters or segments. K-means clustering is a popular method within cluster analysis.

K-Means clustering:

- **Initialization:** Choose the number of clusters (K) we want to create. Randomly select K data points as initial cluster centroids.
- **Assignment:** Assign each data point to the cluster whose centroid is closest. This is often done using a distance metric such as Euclidean distance.
- **Update Centroids:** Recalculate the centroids of the clusters based on the mean of the data points assigned to each cluster.
- **Repeat:** Repeat the assignment and centroid update steps until convergence. Convergence occurs when the assignment of data points to clusters no longer changes or changes very minimally.
- **Output:** The final clusters represent groups of data points with similar characteristics. The main goal of K-Means clustering is to minimize the within-cluster sum of squares, which measures how close the data points within a cluster are to the centroid of that cluster.

It's important to note that K-Means clustering has some limitations, such as sensitivity to the initial selection of centroids and the assumption that clusters are spherical and equally sized.

In practice, choosing the right number of clusters (K) can be challenging. Techniques like the elbow method or silhouette analysis are often employed to find an optimal K. The steps involved in K-Means clustering make it an iterative and computationally efficient algorithm for grouping data points into clusters. It is widely used in various fields, including customer segmentation, image segmentation, and anomaly detection.

2.11 Principal Component Analysis

Principal Component Analysis (PCA) is a dimensionality reduction technique widely used in data analysis and machine learning. It simplifies the complexity in high-dimensional data while retaining trends and patterns.

Steps in PCA:

- **Centering the Data:** For each feature, subtract the mean of that feature from all data points, centering the data around the origin.
- **Calculate Covariance Matrix:** Compute the covariance matrix of the centered data. This matrix shows the relationships between different features.
- **Eigen decomposition:** Find the eigenvectors and eigenvalues of the covariance matrix. Eigenvectors represent the directions of maximum variance, and eigenvalues indicate the magnitude of the variance in those directions.
- **Select Principal Components:** Sort the eigenvectors based on their corresponding eigenvalues in decreasing order. The eigenvectors with

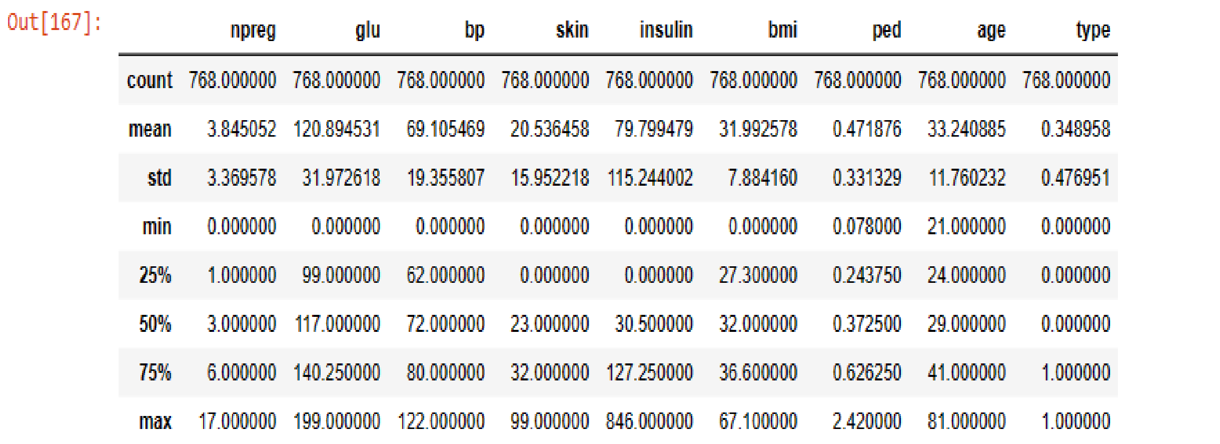
the highest eigenvalues (principal components) capture the most variance in the data.

- **Projection:** Create a projection matrix using the selected principal components. This matrix is used to transform the original data into a new coordinate system.

By selecting a subset of the principal components, one can represent the data in a lower-dimensional space while retaining most of the variance. Higher-dimensional datasets may contain noise or less relevant information. PCA helps focus on the most significant features. Reducing data to two or three dimensions allows for easier visualization and interpretation. It's essential to note that PCA assumes linear relationships in the data and may not perform well if the underlying relationships are nonlinear. Additionally, interpreting the principal components in the original feature space may not always be straightforward.

CHAPTER 3
ANALYSIS AND RESULTS

3.1 Descriptive Statistics

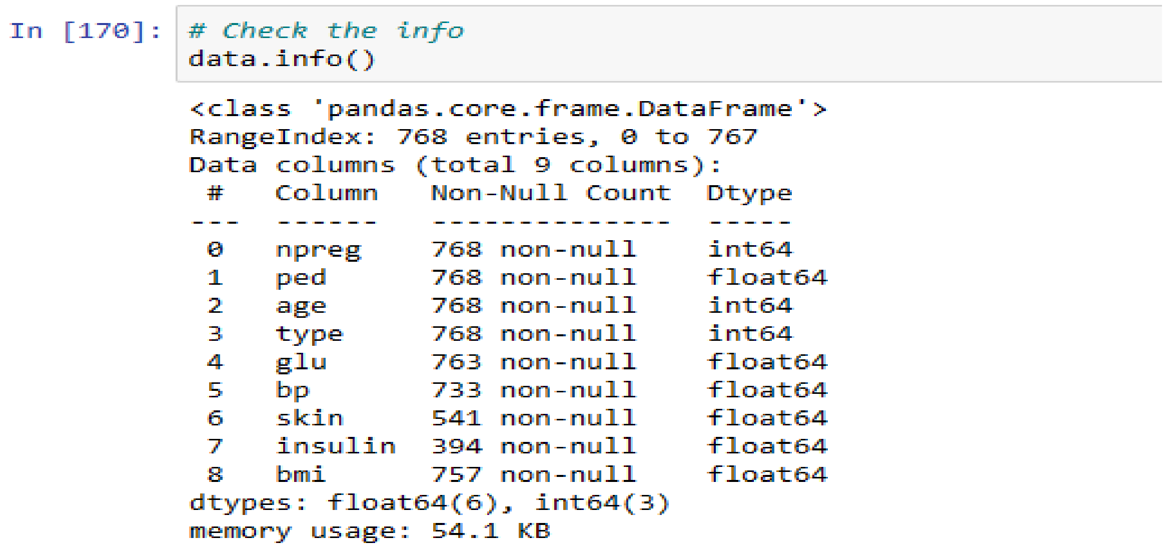


Out[167]:

	npreg	glu	bp	skin	insulin	bmi	ped	age	type
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

In the variables Glucose, BP, Skin, Insulin, and BMI, the minimum values are zero. In reality, these factors are unlikely to have a minimum value of zero; instead, these zero values are indicative of missing data. Therefore, we need to replace these zeros with NA to accurately represent the absence of valid information.

3.2 Checking Variable Characteristics'

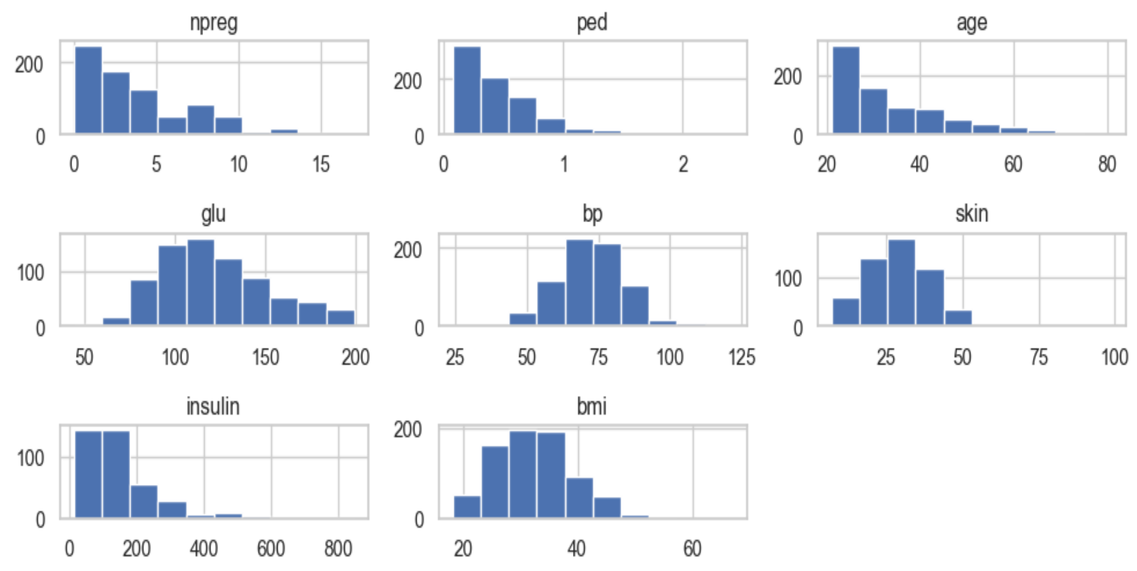


```
In [170]: # Check the info
          data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   npreg    768 non-null    int64
 1   ped      768 non-null    float64
 2   age      768 non-null    int64
 3   type     768 non-null    int64
 4   glu      763 non-null    float64
 5   bp       733 non-null    float64
 6   skin     541 non-null    float64
 7   insulin  394 non-null    float64
 8   bmi      757 non-null    float64
dtypes: float64(6), int64(3)
memory usage: 54.1 KB
```

3.3 Plot the Histogram of Numeric Features

```
In [173]: # plot the histogram of numeric features variables
          data_x.hist()
          plt.tight_layout()
          plt.show()
```



Interpretation:

Age: The variable "age" in our dataset is characterized by a right-skewed distribution. This means that the majority of individuals in our sample tend to be younger, with a few outliers who are older than the average age.

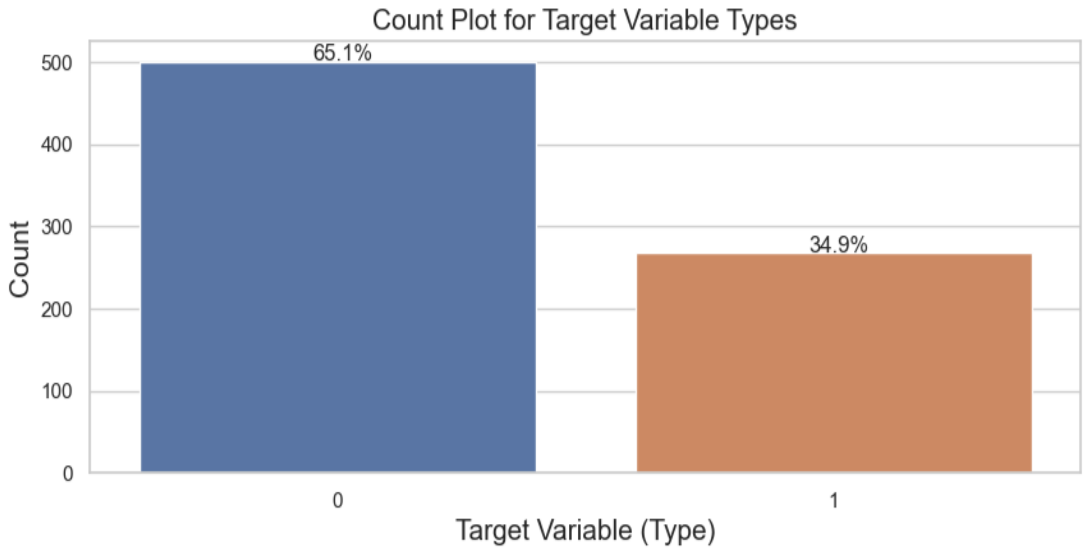
Ped: Similarly, the variable "ped" also displays a right-skewed distribution. This implies that most observations have relatively low values, while a smaller proportion of the data have higher values for this variable.

Npreg: The variable "npreg" exhibits a right-skewed distribution as well. This suggests that the majority of individuals in our dataset have a low number of pregnancies, while a smaller subset of the data includes individuals with a higher number of pregnancies.

Skin: Skin thickness follows an approximately symmetric distribution. This implies that the values of skin thickness are evenly distributed around the mean, with a similar number of individuals having both high and low skin thickness readings.

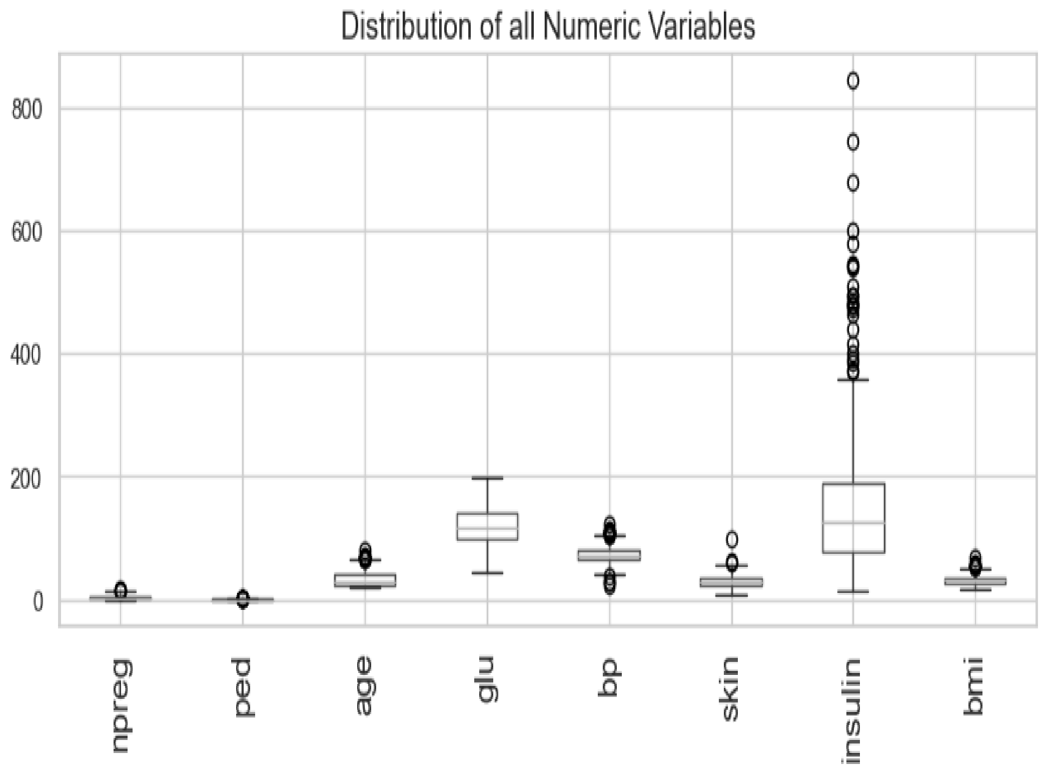
BMI: The variable "BMI" is characterized by a bimodal distribution. This means that there are two distinct peaks or modes in the data, indicating that there are two groups of individuals in our sample with different values for this variable. These groups may have different characteristics related to BMI measurements.

3.4 Count Plot for Target Variable (type)



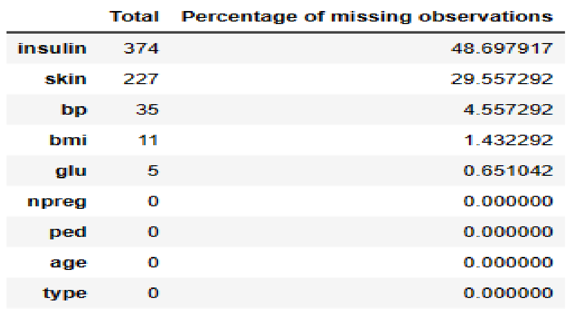
Interpretations:
In our research study, we have a target variable called "type," which serves as an indicator of an individual's diabetes status.
Based on our analysis of the data, we have found that 34.90% of the study participants fall into the diabetic category (type = 1), indicating that they have diabetes. In contrast, 65.1% of the participants fall into the non-diabetic category (type = 0), signifying that they do not have diabetes.
These percentages are important findings as they provide insights into the prevalence of diabetes within our study population. It suggests that a significant portion of the individuals in our study have been diagnosed with diabetes (34.90%), while the majority are non-diabetic (65.10%). This information is crucial for understanding the distribution of diabetes within the studied group and may have implications for healthcare interventions or further research related to diabetes management and prevention.

3.5 Box- Plot for Features



Interpretation: In our analysis, many variables exhibit data points surpassing the upper distribution limit, indicating significant high values. Notably, in blood pressure (BP), we observe both high and low extreme values. Given the medical context, retaining these outliers is crucial as they may signify critical health conditions or unique responses to treatment. This decision aligns with comprehensive medical research, aiming to encompass all potential variations and conditions in the studied population.

3.6 Missing Values Analysis



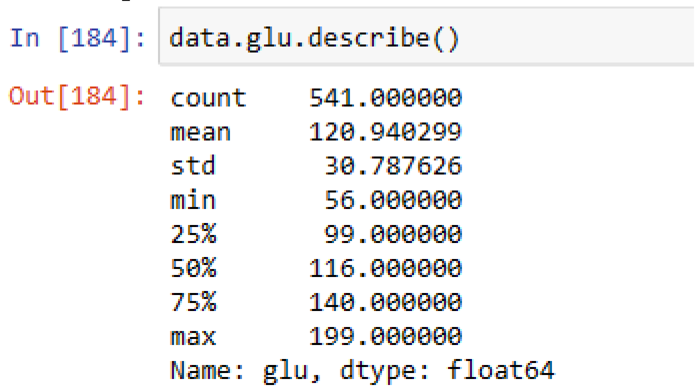
	Total	Percentage of missing observations
insulin	374	48.697917
skin	227	29.557292
bp	35	4.557292
bmi	11	1.432292
glu	5	0.651042
npreg	0	0.000000
ped	0	0.000000
age	0	0.000000
type	0	0.000000

Interpretation: With insulin having nearly 50% missing values in the dataset, it is advice to removing this variable instead of filling or dropping missing values. Filling missing values with any central tendency measure risks losing originality, and dropping them results in losing almost half of the data. For other variables like skin, blood pressure (bp), and body mass index (BMI), excluding glucose (glu), filling missing values with the median is suitable due to the presence of outliers. For glu, filling missing values with the mean is a reasonable approach.

3.7 Univariate Analysis

i.glu

a.Descriptive Statistics



```
In [184]: data.glu.describe()

Out[184]: count    541.000000
          mean     120.940299
          std       30.787626
          min       56.000000
          25%       99.000000
          50%      116.000000
          75%      140.000000
          max      199.000000
          Name: glu, dtype: float64
```

The average plasma glucose concentration in an oral glucose tolerance test is 120.89 mg/dL, ranging from a minimum of 56 mg/dL to a maximum of 199 mg/dL. Among the observations, 25% show a glucose level less than or equal to 99 mg/dL, and 50% have a level less than or equal to 117 mg/dL. A 2-hour plasma glucose level below 140 mg/dL is considered normal. The range of 140-199 mg/dL indicates impaired glucose tolerance, while a level of 200 mg/dL or higher indicates diabetes.

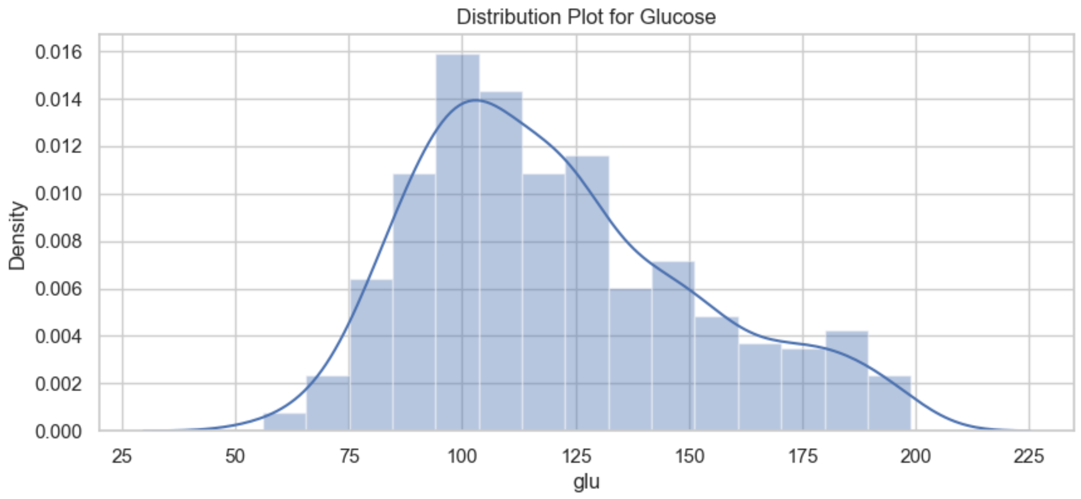
b.Skewness and Kurtosis

Skewness: 0.617323

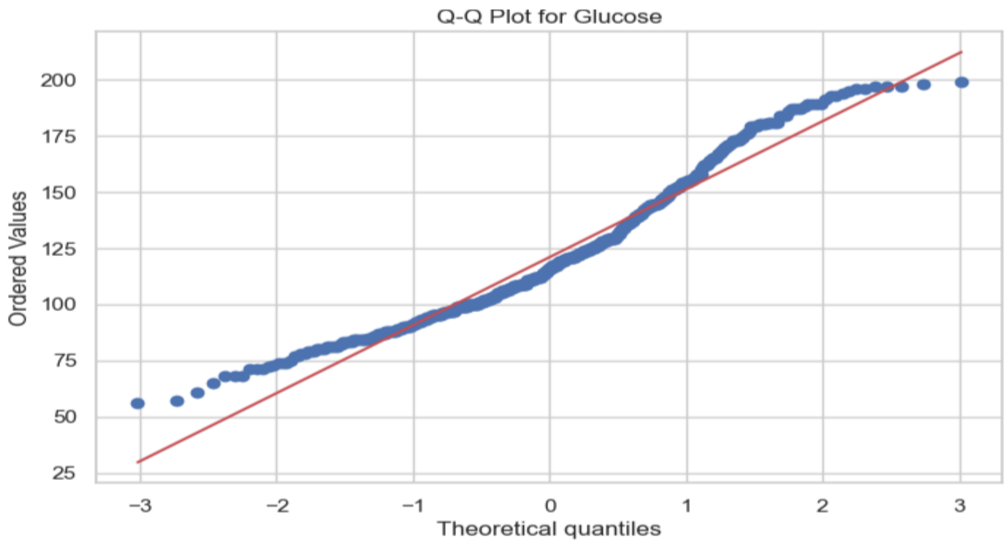
Kurtosis: -0.294508

In summary, the glucose variable's distribution is slightly right-skewed, implying the presence of some higher values, and it is platykurtic, indicating a distribution with lighter tails and a flatter peak compared to a normal distribution.

c. Distribution Plot for Glucose



d. Q-Q Plot for Glucose



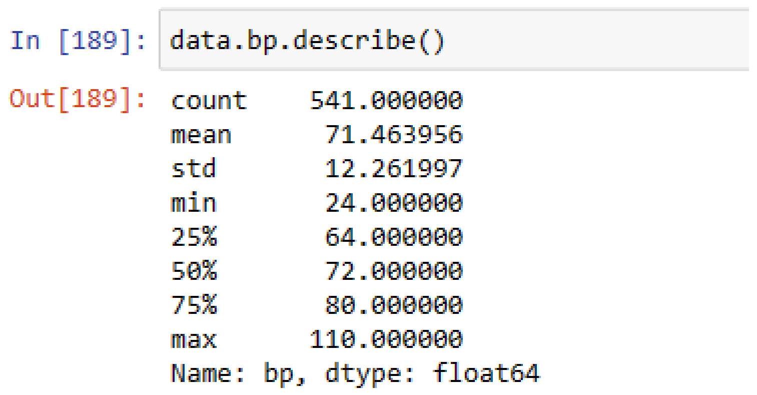
e. Jarque-Bera Test

```
Jarque-Bera statistic: 36.23866327547381
P-value: 1.351681139466535e-08
The glu does not come from a normal distribution (reject the null
hypothesis).
```

Interpretation: The confirmation of non-normal distribution for plasma glucose concentration in an oral glucose tolerance test is supported by the density plot, Q-Q plot, and Jarque-Bera test.

ii.BP

a.Descriptive Statistics



```
In [189]: data.bp.describe()

Out[189]: count    541.000000
          mean      71.463956
          std       12.261997
          min       24.000000
          25%       64.000000
          50%       72.000000
          75%       80.000000
          max      110.000000
          Name: bp, dtype: float64
```

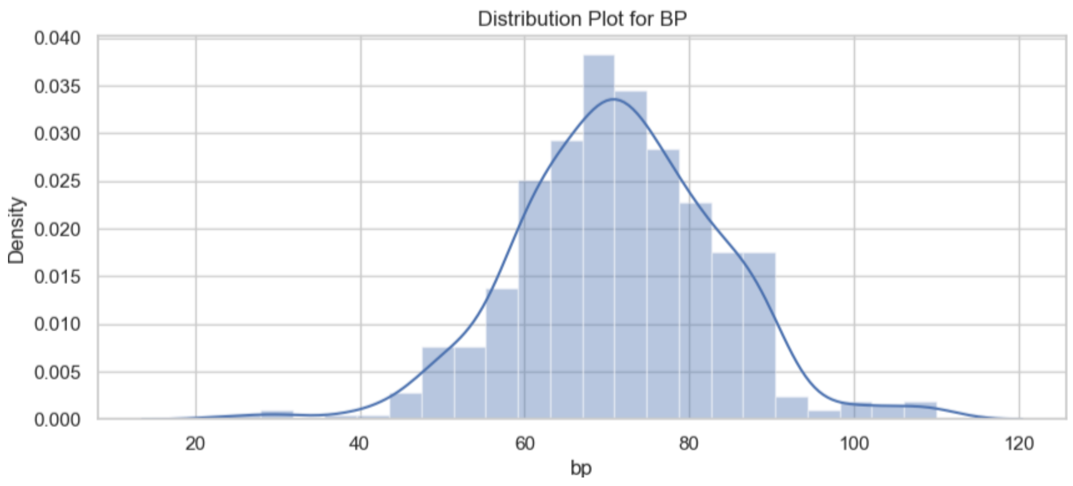
The average blood pressure is approximately 71.46, reflecting a central tendency, but there is notable variability in the dataset. The minimum value of 24 suggests potential risks, as extremely low blood pressure is associated with conditions such as severe hypotension, diabetes and aortic dissection. The mean value of 71.46 falls within the normal range for blood pressure.

b.Skewness and Kurtosis

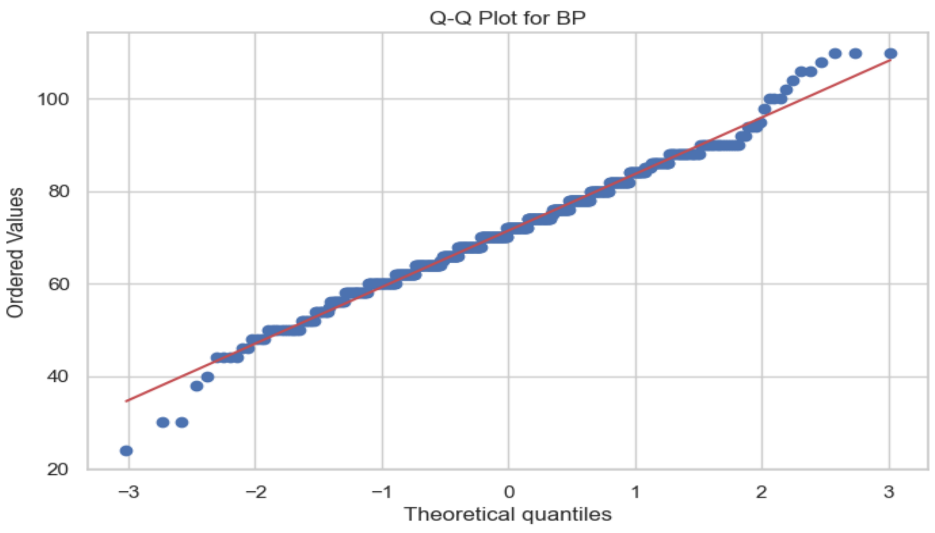
```
Skewness: -0.007283
Kurtosis: 0.823193
```

The skewness near zero suggests a nearly symmetric distribution of blood pressure, and the mesokurtic kurtosis indicates a moderate peak and tails, similar to what would be expected in a normal distribution. These characteristics provide insights into the shape and symmetry of the blood pressure distribution in your dataset.

c. Distribution Plot for BP



d. Q-Q Plot for BP



e. Jarque-Bera Test

```
Jarque-Bera statistic: 14.595424282189091
P-value: 0.0006770860819314788
The BP does not come from a normal distribution (reject the null hypothesis).
```

Interpretation: The confirmation of non-normal distribution for Blood Pressure is supported by the density plot, Q-Q plot, and Jarque-Bera test.

iii.Skin Thickness

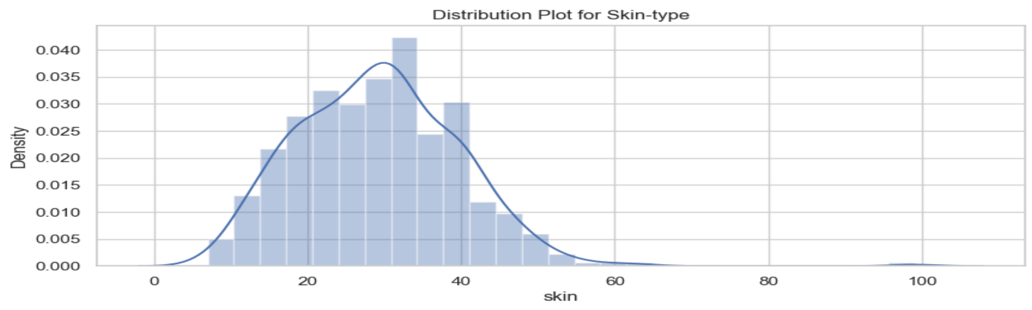
```
In [194]: data.skin.describe()

Out[194]: count     541.000000
          mean       29.153420
          std        10.476982
          min         7.000000
          25%        22.000000
          50%        29.000000
          75%        36.000000
          max        99.000000
          Name: skin, dtype: float64
```

b.Skewness and Kurtosis

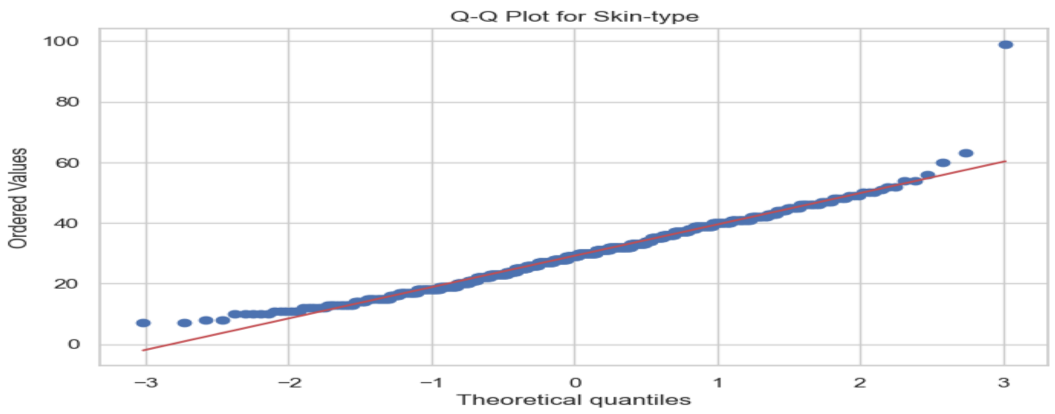
```
Skewness: 0.690619
Kurtosis: 2.935491
```

The positive skewness suggests a right-skewed distribution of skin thickness, and the leptokurtic kurtosis indicates heavier tails and a sharper peak. These characteristics provide insights into the asymmetry and concentration of values in the skin thickness distribution in your dataset.

c. Distribution Plot for Skin Thickness



d. Q-Q Plot for Skin Thickness



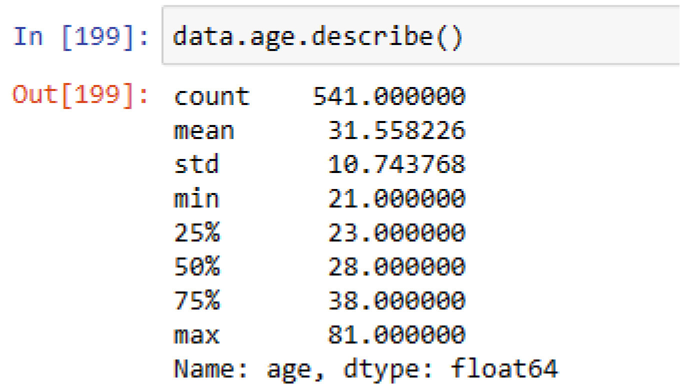
e. Jarque-Bera Test

```
Jarque-Bera statistic: 231.99762905260965
P-value: 4.191359802100761e-51
The Skin-type does not come from a normal distribution (reject the null
hypothesis).
```

Interpretation: The confirmation of non-normal distribution for Skin-type is supported by the density plot, Q-Q plot, and Jarque-Bera test.

iv.Age

a.Descriptive Statistics



```
In [199]: data.age.describe()

Out[199]: count    541.000000
          mean      31.558226
          std       10.743768
          min       21.000000
          25%       23.000000
          50%       28.000000
          75%       38.000000
          max       81.000000
          Name: age, dtype: float64
```

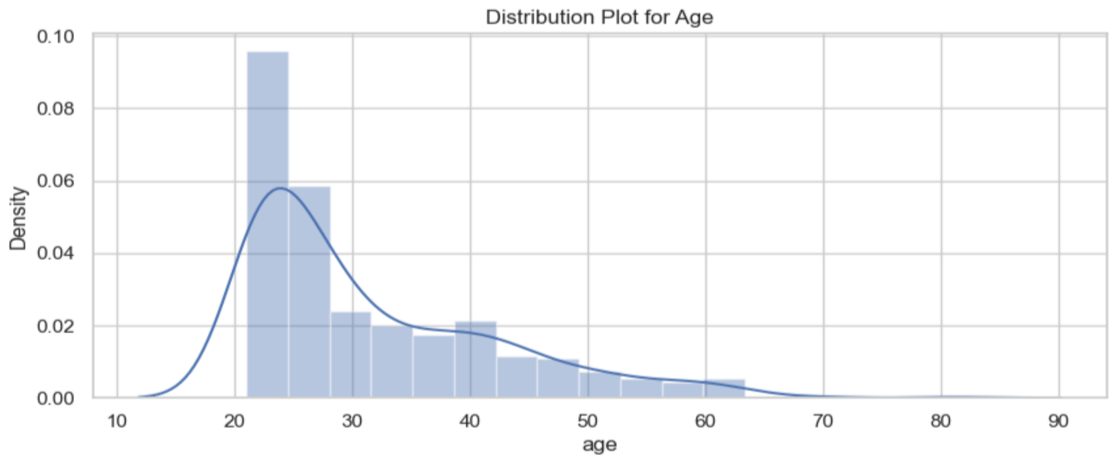
The average age in the study is 31.50 years, ranging from a minimum of 21 years to a maximum of 81 years. Twenty-five percent of women are younger than 23 years, and 50% of women are 28 years old.

b.Skewness and Kurtosis

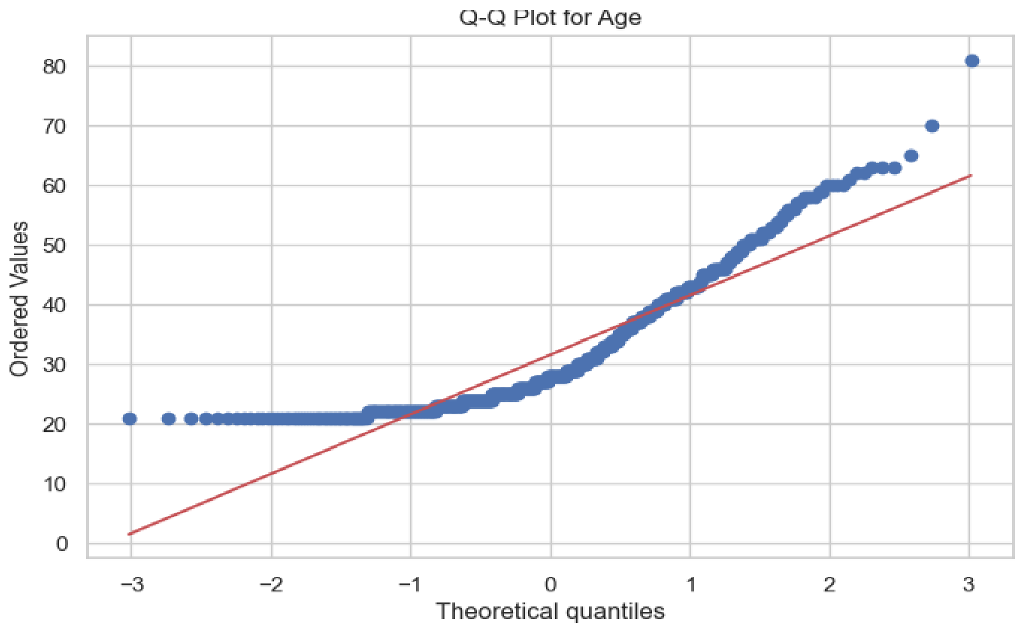
```
Skewness: 1.269905
Kurtosis: 1.180848
```

Based on these values, the age distribution in our study is likely to be somewhat skewed to the right with a few older individuals having ages considerably higher than the mean. Additionally, the distribution may exhibit heavier tails and a slightly more peaked shape compared to a normal distribution. Further analysis and visualization of the age distribution, such as a histogram or a density plot, could provide a clearer picture of the data distribution.

c. Distribution Plot



d. Q-Q



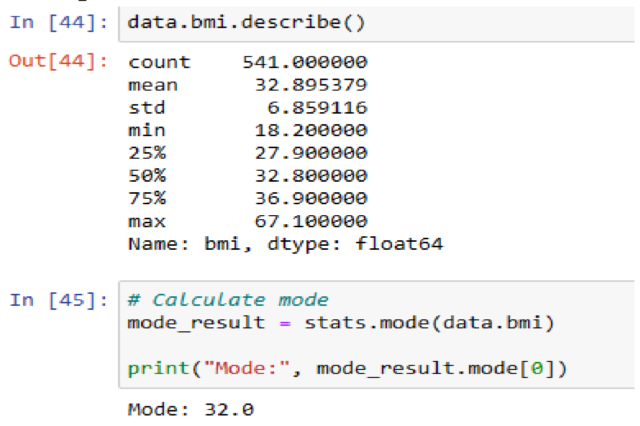
e. Jarque-Bera Test

```
Jarque-Bera statistic: 174.87643785709037
P-value: 1.061852055024055e-38
The Age does not come from a normal distribution (reject the
null hypothesis).
```

Interpretation: The confirmation of non-normal distribution for age is supported by the density plot, Q-Q plot, and Jarque-Bera test.

v. BMI

a.Descriptive Statistics



```
In [44]: data.bmi.describe()

Out[44]: count     541.000000
         mean       32.895379
         std         6.859116
         min        18.200000
         25%        27.900000
         50%        32.800000
         75%        36.900000
         max        67.100000
         Name: bmi, dtype: float64

In [45]: # Calculate mode
         mode_result = stats.mode(data.bmi)

         print("Mode:", mode_result.mode[0])

         Mode: 32.0
```

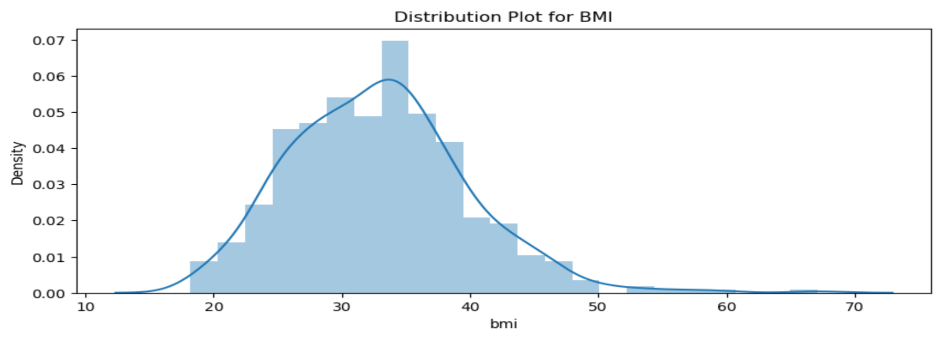
The mean Body Mass Index (BMI) is 32.89. The BMI ranges from a minimum of 18.20 to a maximum of 67.10. Notably, approximately 50% of women exhibit a BMI below 32.80, a value that aligns closely with both the mean and mode of the BMI. This indicates a close similarity between the mean, median, and mode, suggesting a distribution with a central tendency.

b.Skewness and Kurtosis

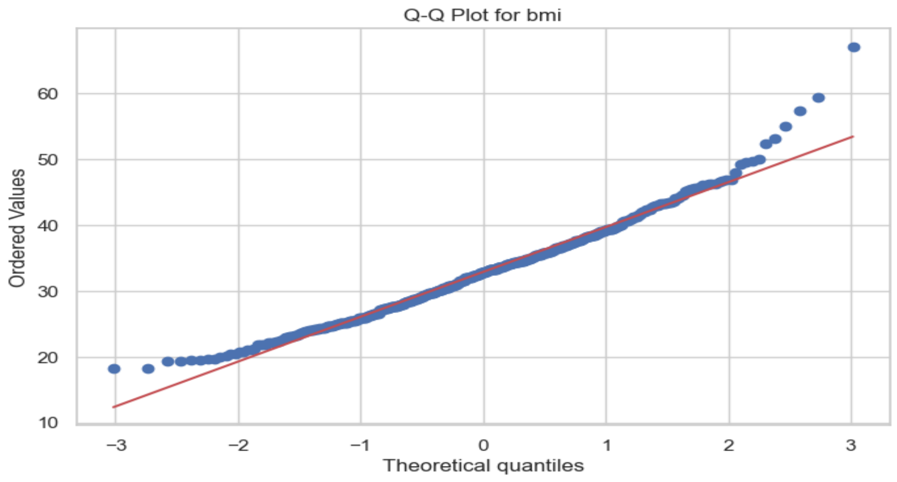
```
Skewness: 0.626830
Kurtosis: 1.272595
```

The distribution is right-skewed, suggesting a possible concentration of individuals with higher BMI values. The kurtosis indicates that the distribution has heavier tails and is more peaked than a normal distribution, possibly indicating the presence of more extreme BMI values.

c. Distribution Plot



d. Q-Q Plot

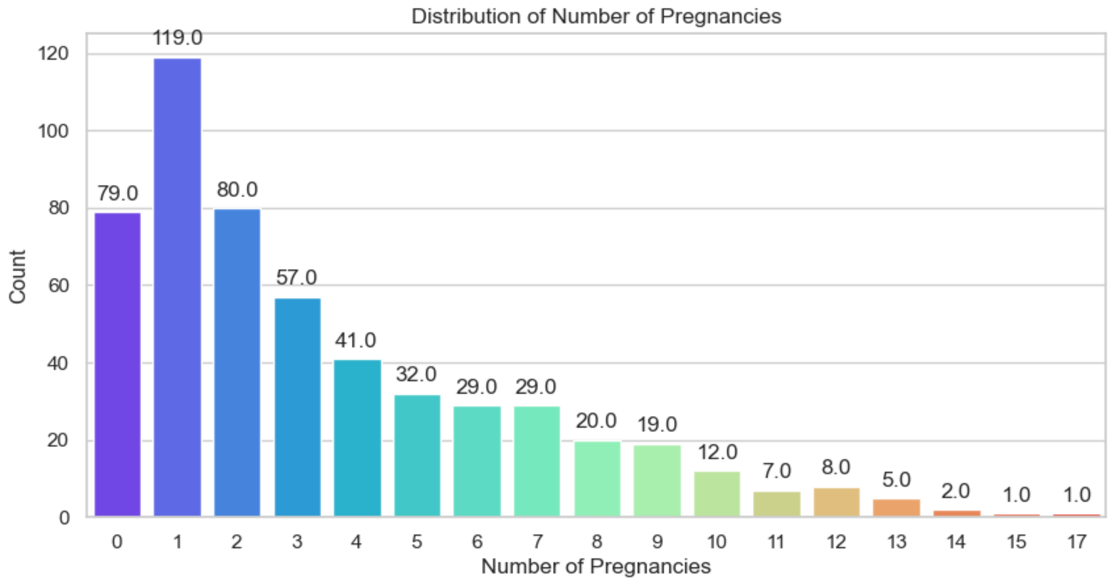


e. Jarque-Bera Test

```
Jarque-Bera statistic: 70.44127573826687
P-value: 5.056748150849164e-16
The BMI does not come from a normal distribution (reject the
null hypothesis).
```

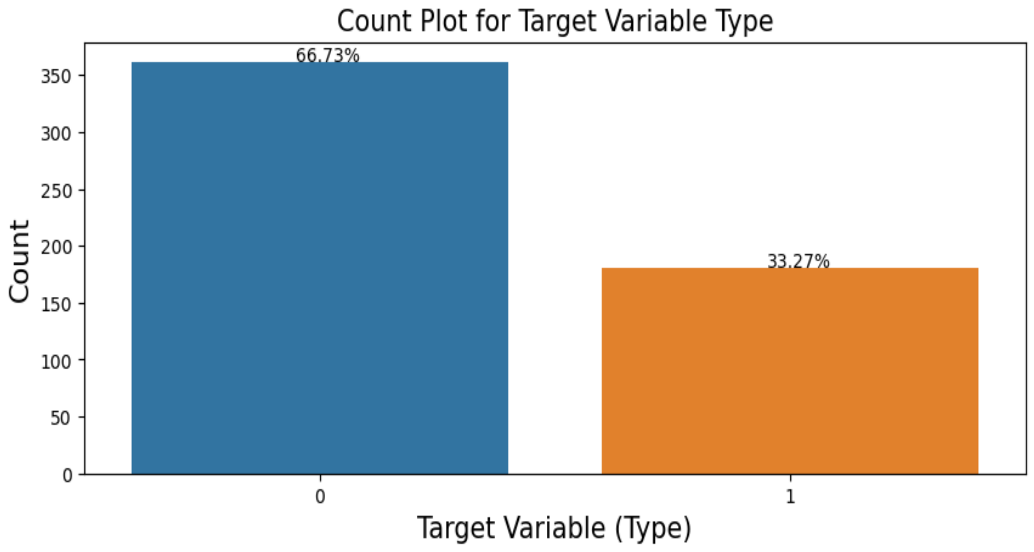
<u>Interpretations:</u> The confirmation of non-normal distribution for BMI is supported by the density plot, Q-Q plot, and Jarque-Bera test.

vi. Number of Pregnancies



Where 79 women have no children and 119 women have one child, the mode of the distribution is determined to be 1.

vii. Type (outcome): Diabetic (Yes or No)

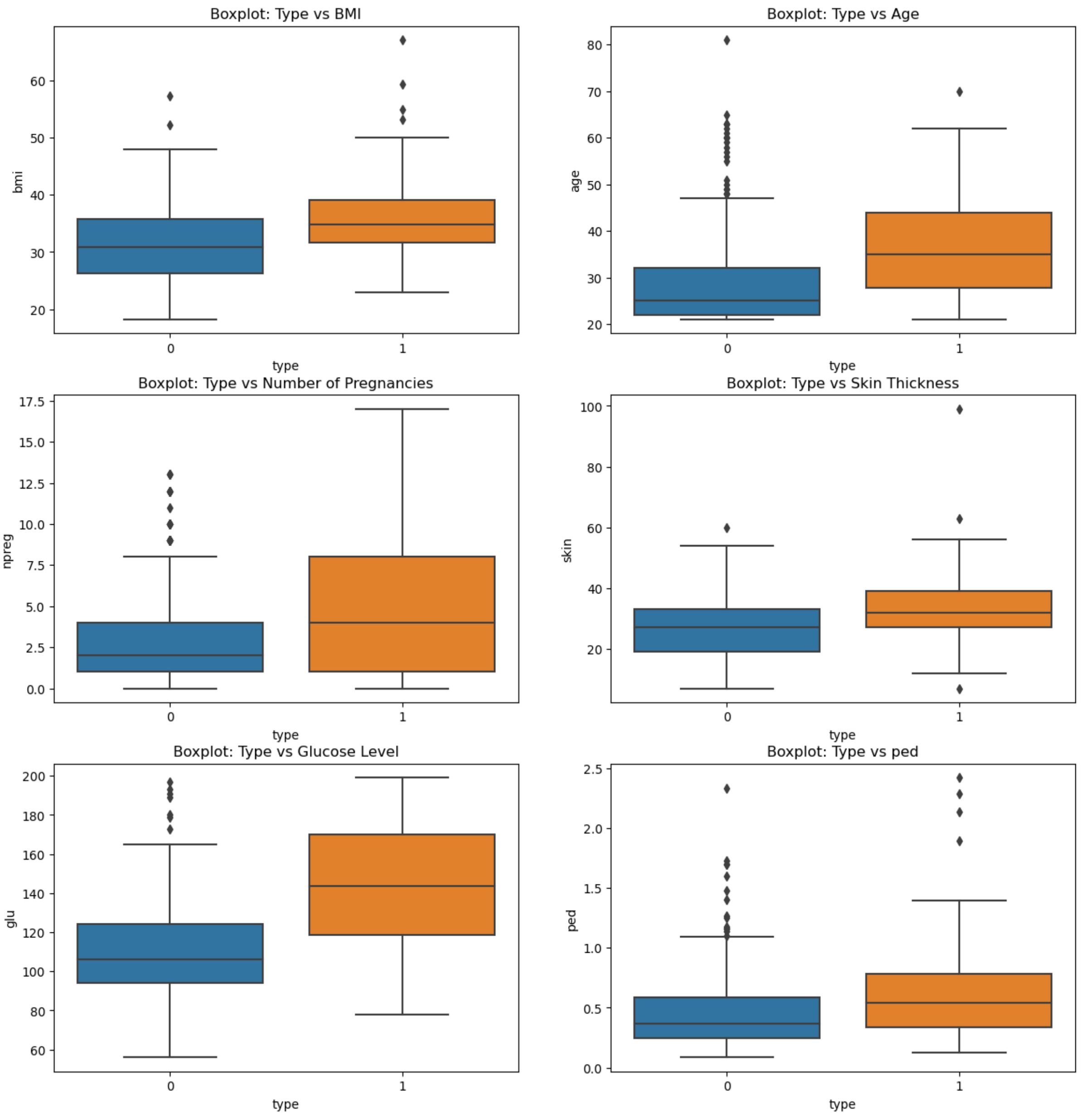


Based on our analysis of the data, we have found that 33.27% of the study participants fall into the diabetic category (type = 1), indicating that they have diabetes. In contrast, 66.73% of the participants fall into the non-diabetic category (type = 0), signifying that they do not have diabetes.

These percentages are important findings as they provide insights into the prevalence of diabetes within our study population. It suggests that a significant portion of the individuals in our study have been diagnosed with diabetes (33.27%), while the majority are non-diabetic (66.73%). This information is crucial for understanding the distribution of diabetes within the studied group and may have implications for healthcare interventions or further research related to diabetes management and prevention.

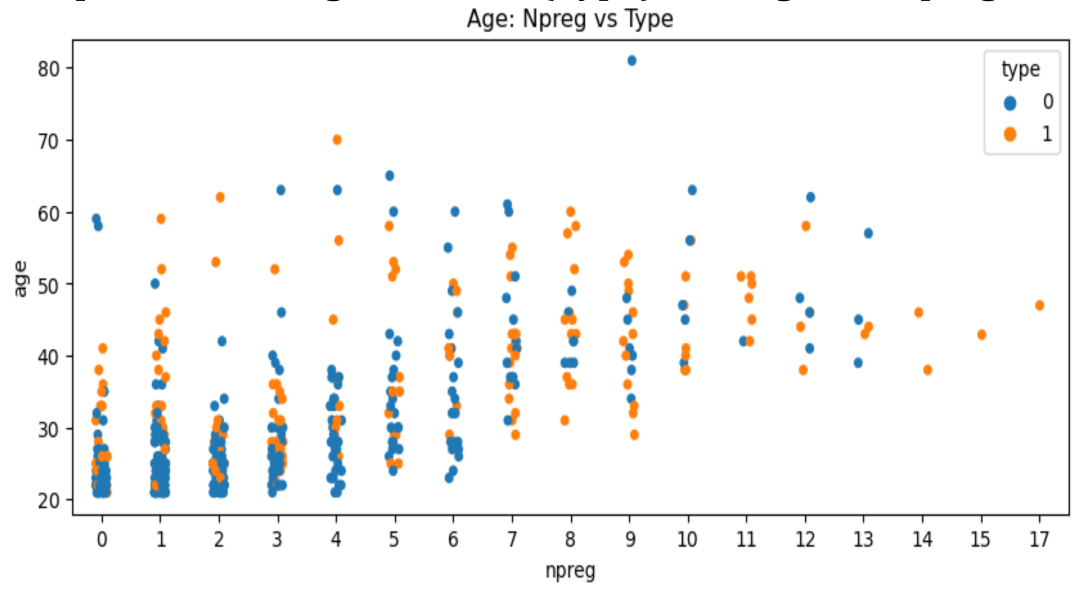
3.8 Multivariate Analysis

i.Box Plots for Target Variable (Type) with Different Features



From the multiple boxplots presented above, a notable observation emerges individuals with diabetes exhibit significantly higher mean values for BMI, Glucose, Pedigree Function (ped), and age compared to those without diabetes. Specifically, the boxplots reveal a clear distinction in these variables between the two groups. The higher mean values in BMI, Glucose, ped, and age for individuals with diabetes suggest potential associations or characteristics that differentiate the diabetic and non-diabetic populations in the dataset.

ii. Strip-Plots for Target Variable (Type) with Age and Npreg



The likelihood of diabetes appears to be higher in cases with a greater number of pregnancies than in instances with fewer pregnancies. This observation implies a potential correlation between the number of pregnancies and the risk of diabetes among women. Further analysis and statistical testing would be needed to establish a conclusive relationship and determine potential contributing factors.

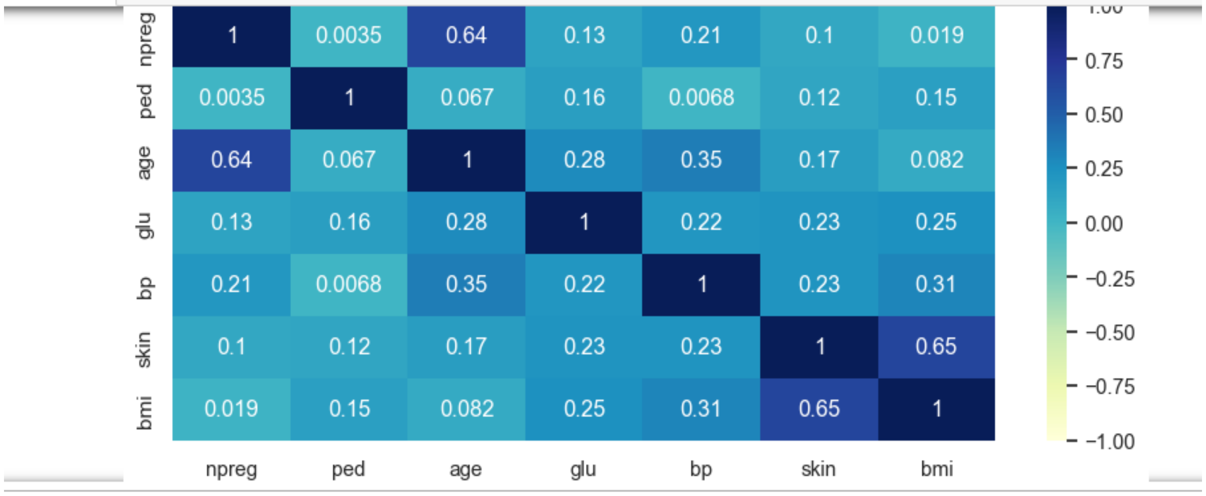
iii. Correlation Matrix and Heat Map

```
In [215]: data_x.corr()
```

```
Out[215]:
```

	npreg	ped	age	glu	bp	skin	bmi
npreg	1.000000	0.003539	0.644686	0.127319	0.205066	0.100239	0.019495
ped	0.003539	1.000000	0.066834	0.160832	0.006823	0.115016	0.150885
age	0.644686	0.066834	1.000000	0.279718	0.347147	0.166816	0.081580
glu	0.127319	0.160832	0.279718	1.000000	0.217581	0.227369	0.247116
bp	0.205066	0.006823	0.347147	0.217581	1.000000	0.226723	0.309508
skin	0.100239	0.115016	0.166816	0.227369	0.226723	1.000000	0.647828
bmi	0.019495	0.150885	0.081580	0.247116	0.309508	0.647828	1.000000

```
In [58]: corr=data_x.corr()
         sns.heatmap(corr, cmap = 'YlGnBu', vmax = 1.0, vmin = -1.0, annot = True, annot_kws = {"size": 12})
```



	npreg	ped	age	glu	bp	skin	bmi
npreg	1	0.0035	0.64	0.13	0.21	0.1	0.019
ped	0.0035	1	0.067	0.16	0.0068	0.12	0.15
age	0.64	0.067	1	0.28	0.35	0.17	0.082
glu	0.13	0.16	0.28	1	0.22	0.23	0.25
bp	0.21	0.0068	0.35	0.22	1	0.23	0.31
skin	0.1	0.12	0.17	0.23	0.23	1	0.65
bmi	0.019	0.15	0.082	0.25	0.31	0.65	1

Interpretation: Age and number of pregnancies: There is a positive correlation between Age and number of pregnancy. This means that those who are having higher age have more number of children.

BMI and Skin thickness: There is a positive correlation between BMI and Skin thickness. This means that those who have having higher BMI with higher skin thickness.

Remaining Variables: For the other variables in our dataset, there are weak or no significant correlations between them. This implies that changes in one of these variables do not strongly influence or predict changes in the others. They are relatively independent of each other in terms of their relationships within the dataset.

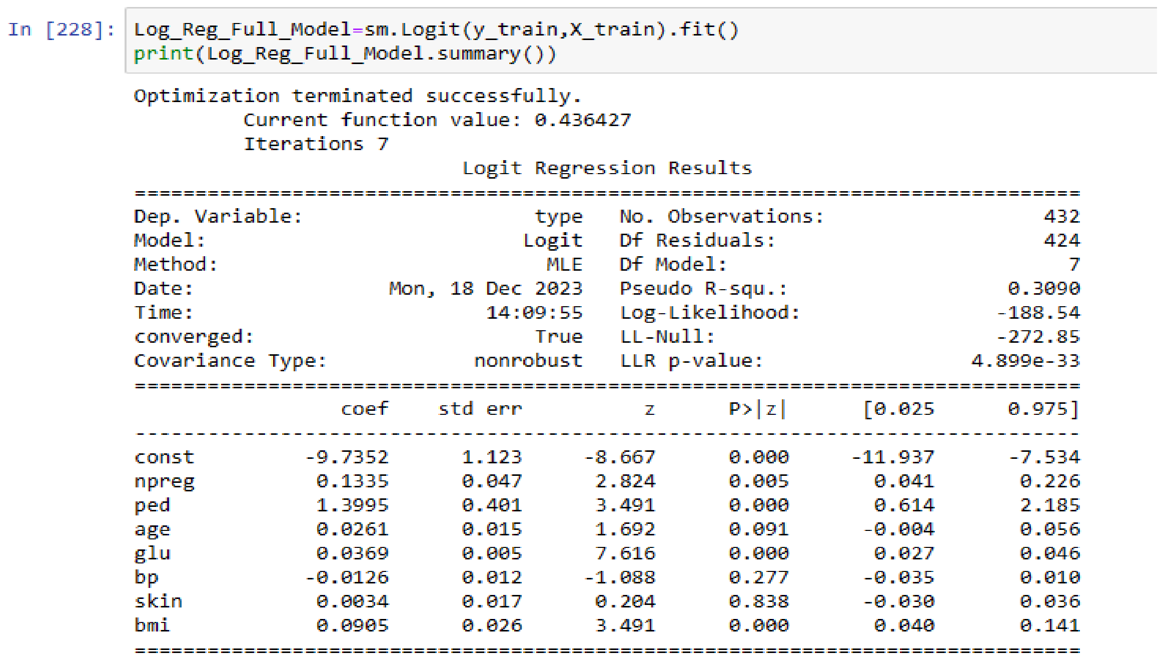
3.9 ML Models for Classification Problem

After completing data cleaning and certain exploratory data analysis (EDA) steps, we partitioned the data into two sets: a training set comprising 80% of the observations and a test set with 20% of the observations to assess the model's accuracy.

In this phase, we applied various machine learning models, namely Logistic Regression, Decision Tree, Naive Bayes, and Support Vector Machine. Subsequently, we compared the accuracy of these different models, selecting the best-performing ones for deployment.

3.9.1 Logistic Regression Model with All Features

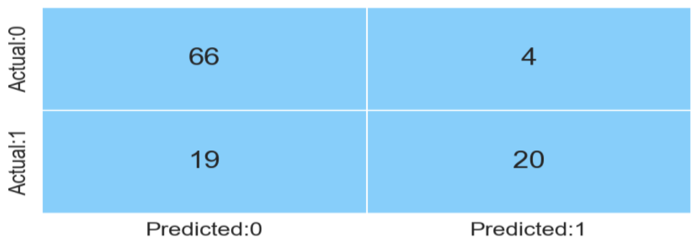
a.Model Summary:



```
In [228]: Log_Reg_Full_Model=sm.Logit(y_train,X_train).fit()
          print(Log_Reg_Full_Model.summary())

Optimization terminated successfully.
         Current function value: 0.436427
         Iterations 7
                           Logit Regression Results
==============================================================================
Dep. Variable:                   type   No. Observations:                  432
Model:                          Logit   Df Residuals:                      424
Method:                           MLE   Df Model:                            7
Date:                Mon, 18 Dec 2023   Pseudo R-squ.:                  0.3090
Time:                        14:09:55   Log-Likelihood:                -188.54
converged:                       True   LL-Null:                       -272.85
Covariance Type:            nonrobust   LLR p-value:                 4.899e-33
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const         -9.7352      1.123     -8.667      0.000     -11.937      -7.534
npreg          0.1335      0.047      2.824      0.005       0.041       0.226
ped            1.3995      0.401      3.491      0.000       0.614       2.185
age            0.0261      0.015      1.692      0.091      -0.004       0.056
glu            0.0369      0.005      7.616      0.000       0.027       0.046
bp            -0.0126      0.012     -1.088      0.277      -0.035       0.010
skin           0.0034      0.017      0.204      0.838      -0.030       0.036
bmi            0.0905      0.026      3.491      0.000       0.040       0.141
==============================================================================
```

Except for BP and skin thickness, all variables are deemed statistically significant, as their p-values exceed 0.05 for intercept, npreg, ped, age, glu, and bmi. This implies significance at the 5% level. However, the inclusion of non-significant variables raises concerns about potential overfitting. To mitigate this, we intend to select only significant variables using forward or backward elimination in feature selection methods and subsequently assess the model's accuracy.

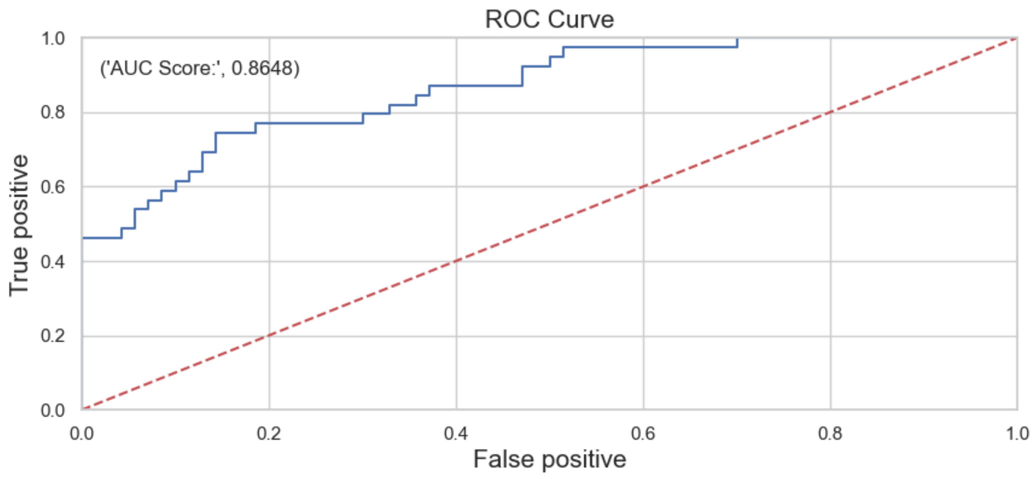
b.Confusion Matrix:



	Predicted:0	Predicted:1
Actual:0	66	4
Actual:1	19	20

The confusion matrix indicates a 17.43% false negative and a 3.6% false positive, resulting in a total accuracy of 78.89%.

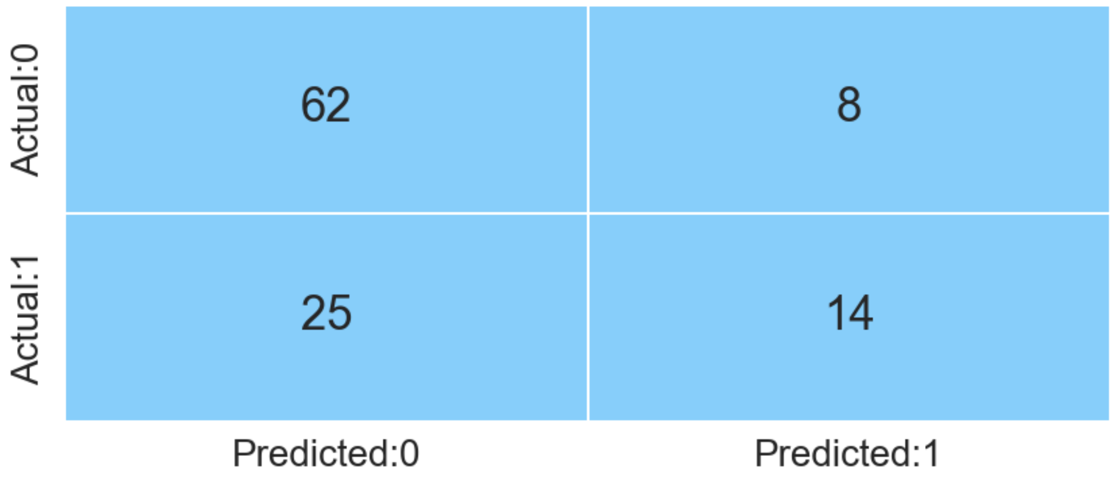
c. ROC-Curve:



The AUC score of 0.8648 indicates that there is a high probability that the model will assign a higher predicted probability to a randomly chosen positive instance compared to a randomly chosen negative instance. The closer the AUC score is to 1, the better the model's ability to distinguish between positive and negative instances.

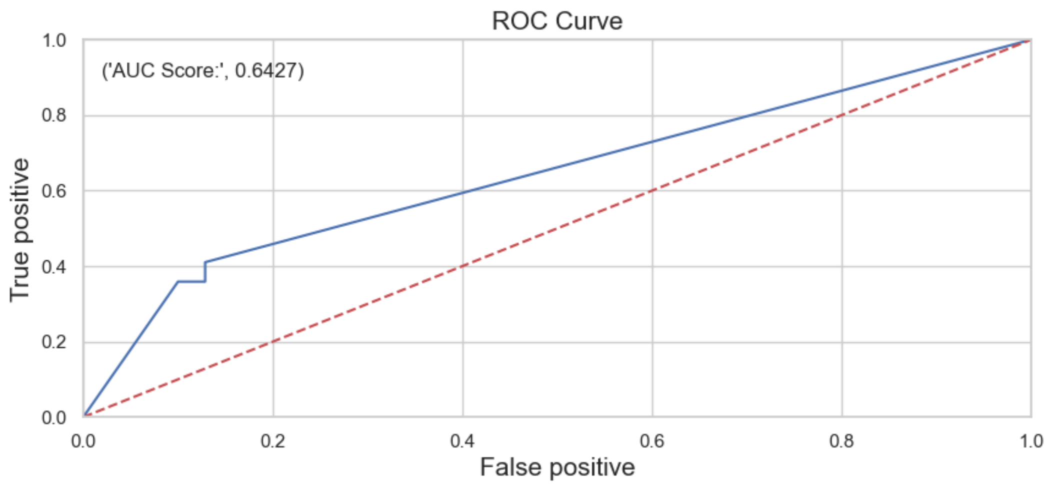
3.9.2 SGDC Classifier with Constant(Intercept term)

a.Confusion Matrix:



The confusion matrix reveals a 22.93% false negative rate and a 7.3% false positive rate, leading to an overall accuracy of 69.72%. This accuracy is comparatively lower than that of the previous model**.**

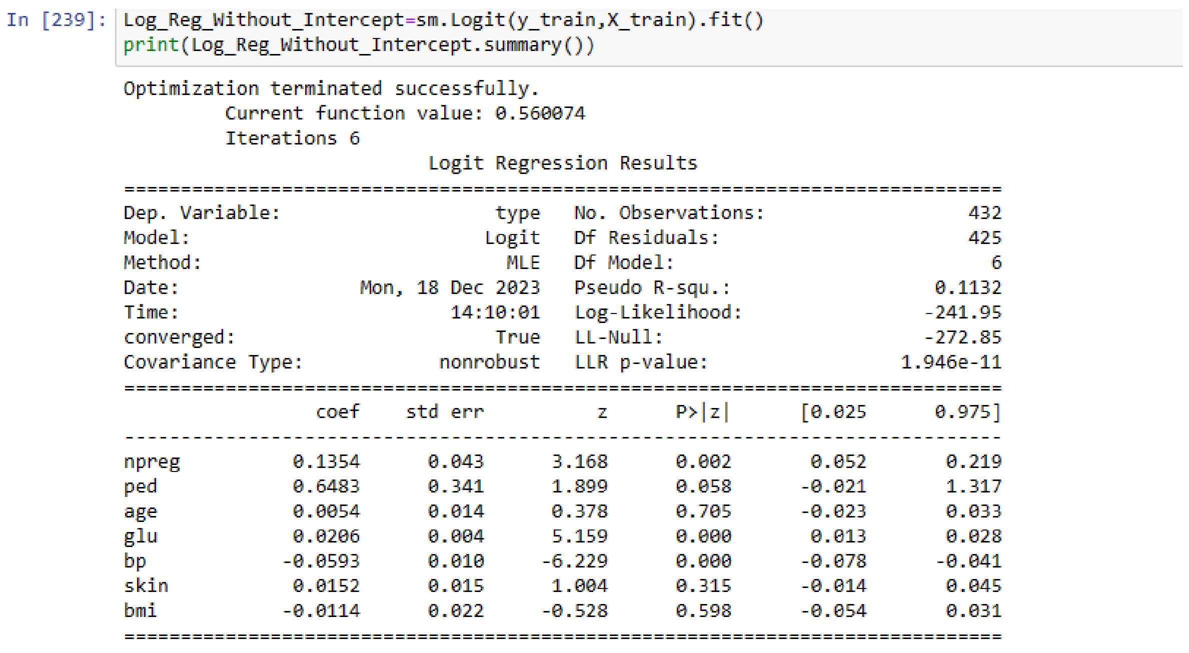
b.ROC Curve:



An Area Under the Curve (AUC) score of 0.6427 on the Receiver Operating Characteristic (ROC) curve suggests a moderate discriminatory performance of the model. The ROC curve illustrates the trade-off between the true positive rate (sensitivity) and the false positive rate (1-specificity) across various threshold values.

3.9.3 Logistic Regression Without Intercept

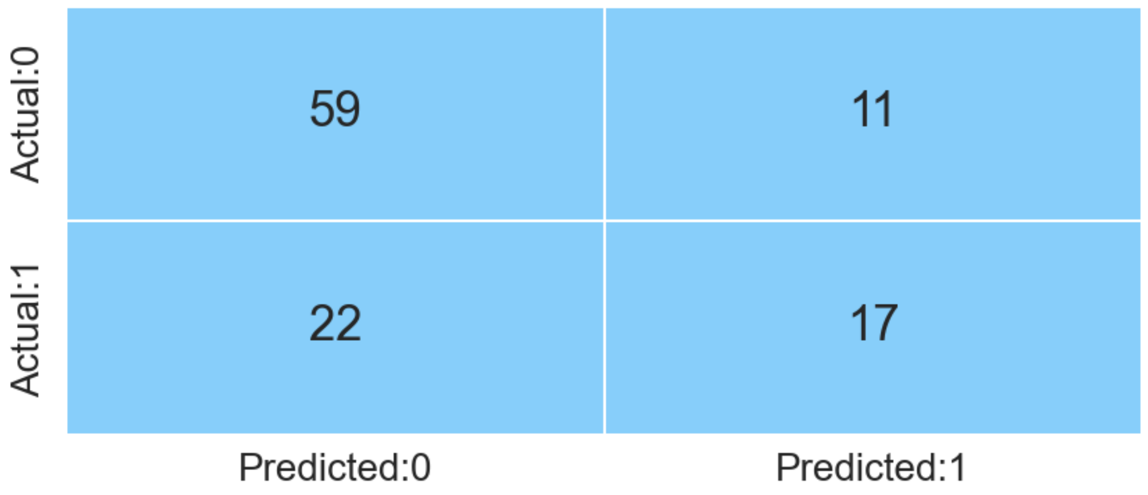
a.Model Summary



```
In [239]: Log_Reg_Without_Intercept=sm.Logit(y_train,X_train).fit()
          print(Log_Reg_Without_Intercept.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.560074
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                   type   No. Observations:                  432
Model:                          Logit   Df Residuals:                      425
Method:                           MLE   Df Model:                            6
Date:                Mon, 18 Dec 2023   Pseudo R-squ.:                  0.1132
Time:                        14:10:01   Log-Likelihood:                -241.95
converged:                       True   LL-Null:                       -272.85
Covariance Type:            nonrobust   LLR p-value:                 1.946e-11
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
npreg          0.1354      0.043      3.168      0.002       0.052       0.219
ped            0.6483      0.341      1.899      0.058      -0.021       1.317
age            0.0054      0.014      0.378      0.705      -0.023       0.033
glu            0.0206      0.004      5.159      0.000       0.013       0.028
bp            -0.0593      0.010     -6.229      0.000      -0.078      -0.041
skin           0.0152      0.015      1.004      0.315      -0.014       0.045
bmi           -0.0114      0.022     -0.528      0.598      -0.054       0.031
==============================================================================
```

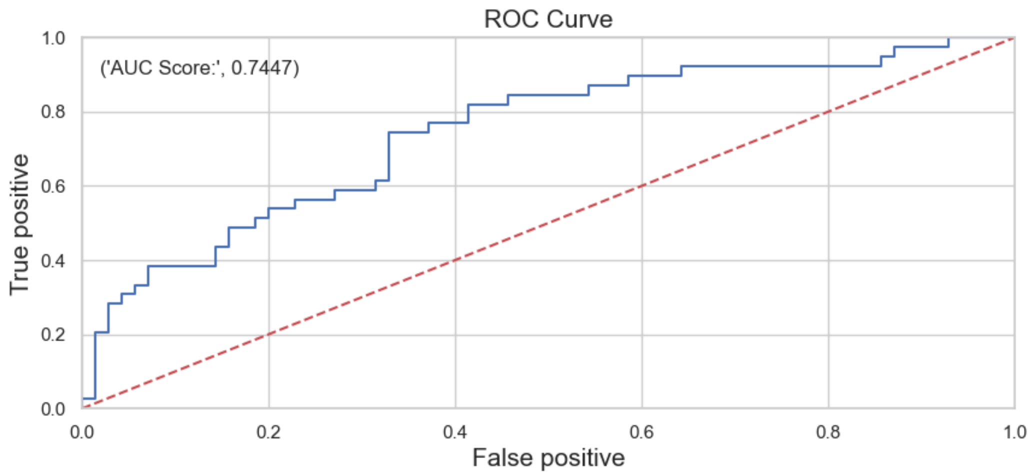
b. Confusion Matrix:



	Predicted:0	Predicted:1
Actual:0	59	11
Actual:1	22	17

The confusion matrix reveals a 20.18% false negative rate and a 10.09% false positive rate, resulting in an overall accuracy of 69.72%. This accuracy is comparatively lower than that of the first model, suggesting a potential need to consider adding the intercept term.

c. ROC Curve:



An AUC score of 0.74 implies a fair discriminatory performance, and further analysis or adjustments may be considered to potentially enhance the model's effectiveness.

3.9.4 Backward Elimination Method Using Univariate Statistical Testing

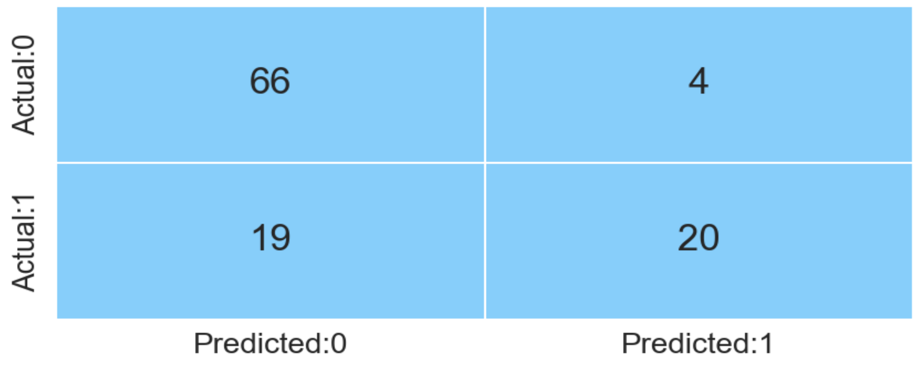
```
Selected Features (Backward): ['npreg', 'ped', 'glu', 'bmi'] by
Backward Elimination Method
```

a. Model Summary

```
Optimization terminated successfully.
         Current function value: 0.440269
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                   type   No. Observations:                  432
Model:                          Logit   Df Residuals:                      427
Method:                           MLE   Df Model:                            4
Date:                Mon, 18 Dec 2023   Pseudo R-squ.:                  0.3029
Time:                        14:10:04   Log-Likelihood:                -190.20
converged:                       True   LL-Null:                       -272.85
Covariance Type:            nonrobust   LLR p-value:                 1.065e-34
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const         -9.8095      1.004     -9.766      0.000     -11.778      -7.841
npreg          0.1795      0.037      4.836      0.000       0.107       0.252
ped            1.4627      0.395      3.705      0.000       0.689       2.236
glu            0.0377      0.005      8.004      0.000       0.028       0.047
bmi            0.0856      0.020      4.293      0.000       0.047       0.125
==============================================================================
```

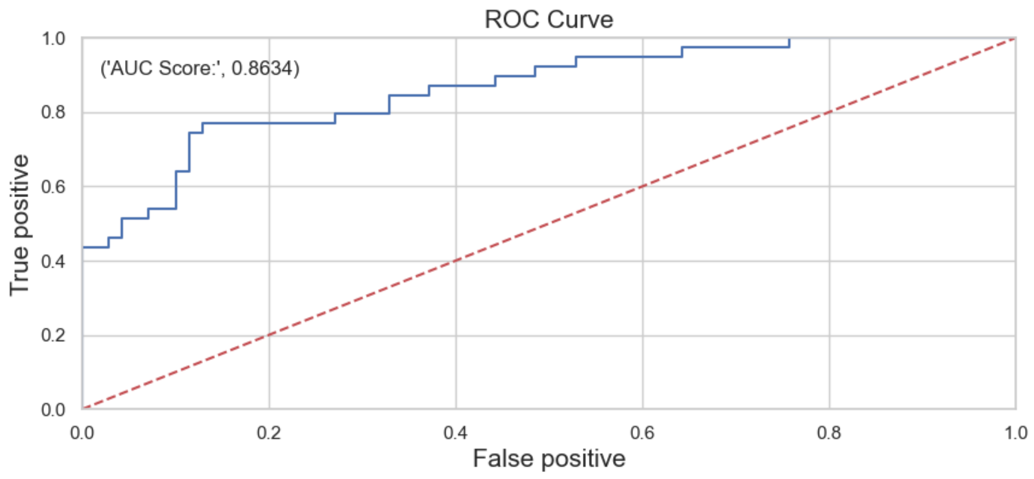
In the model summary table, all variables are found to be significant at any given level of significance. Models of this type are less prone to overfitting. If the model performs well under the performance metrics, we can consider finalizing it for real-world predictions of diabetes.

b. Confusion Matrix:



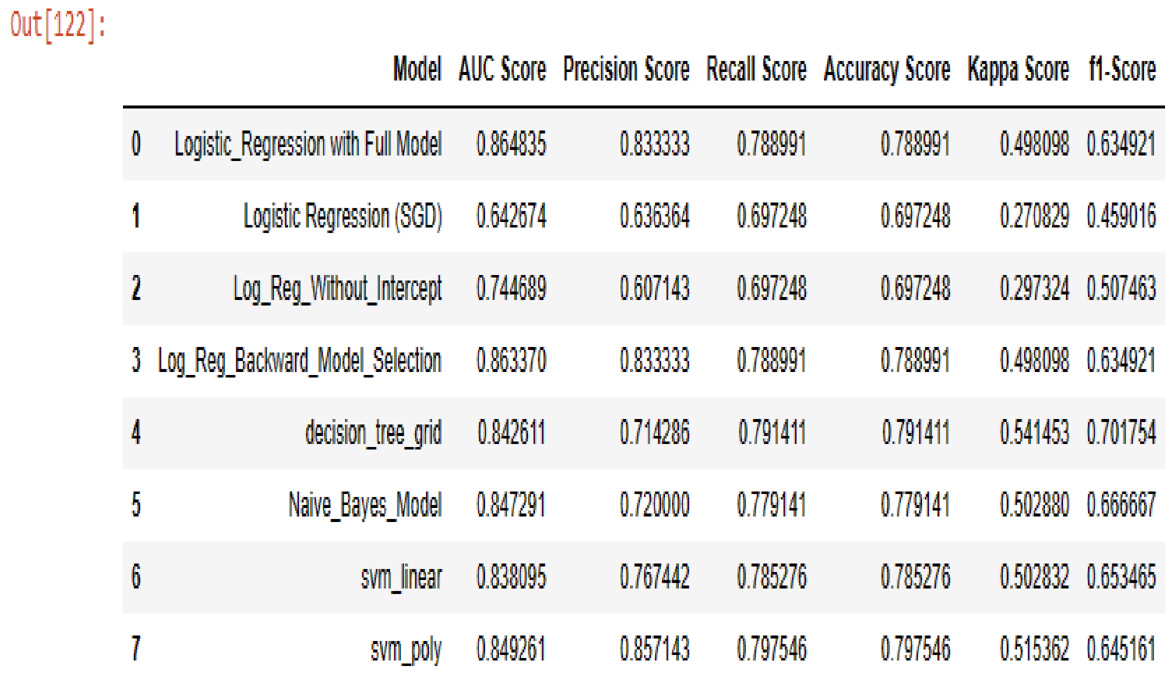
The confusion matrix reveals a 17.43% false negative rate and a 3.6% false positive rate, leading to an overall accuracy of 78.89%. The performance of this model is comparable to that of the first model.

c. ROC Curve



An AUC score of 0.8648 suggests that the model has a strong ability to discriminate between classes, making it a promising indicator of its overall performance.

Similarly, we applied other models, namely Decision Tree Classifier, Support Vector Machine, and Naive Bayes Classifier. We calculated the performance metrics to compare the accuracy of these models. The accuracy table is provided below:



Out[122]:

	Model	AUC Score	Precision Score	Recall Score	Accuracy Score	Kappa Score	f1-Score
0	Logistic_Regression with Full Model	0.864835	0.833333	0.788991	0.788991	0.498098	0.634921
1	Logistic Regression (SGD)	0.642674	0.636364	0.697248	0.697248	0.270829	0.459016
2	Log_Reg_Without_Intercept	0.744689	0.607143	0.697248	0.697248	0.297324	0.507463
3	Log_Reg_Backward_Model_Selection	0.863370	0.833333	0.788991	0.788991	0.498098	0.634921
4	decision_tree_grid	0.842611	0.714286	0.791411	0.791411	0.541453	0.701754
5	Naive_Bayes_Model	0.847291	0.720000	0.779141	0.779141	0.502880	0.666667
6	svm_linear	0.838095	0.767442	0.785276	0.785276	0.502832	0.653465
7	svm_poly	0.849261	0.857143	0.797546	0.797546	0.515362	0.645161

Conclusions on Classification Models:

The table above compares the performance of eight models. Among these models, except for the Logistic model with the SGD classifier, the remaining models exhibit nearly equal performance. Some models, such as decision tree, SVM, and naive Bayes, are complex (high variance) and may be prone to overfitting. When simple models (low variance) perform nearly as well as complex models, it is preferable to choose the simpler ones. We are selecting Logistic Regression with the Backward Elimination method because all variables are significant, and it involves fewer features. To mitigate the risk of high variance, opting for simpler models is advisable, especially when the model does not exhibit signs of underfitting.

3.10 Multiple Linear Regression Model Building for ped

3.10.1Multiple Regression Model Building for ped using PCA

a. Perform PCA with the following steps:

i. Filter the numerical variables

ii. Scale the data to get variables on the same scale

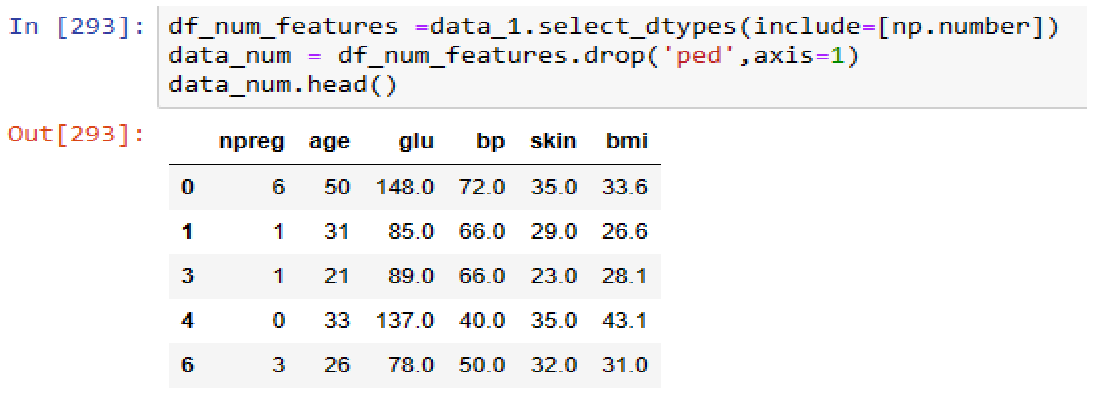
iii. Compute covariance matrix

iv. Calculate eigenvalues and eigenvectors of the covariance matrix

v. Decide the number of principal components

vi. Obtain principal components

i. Filter the Numerical Variables:

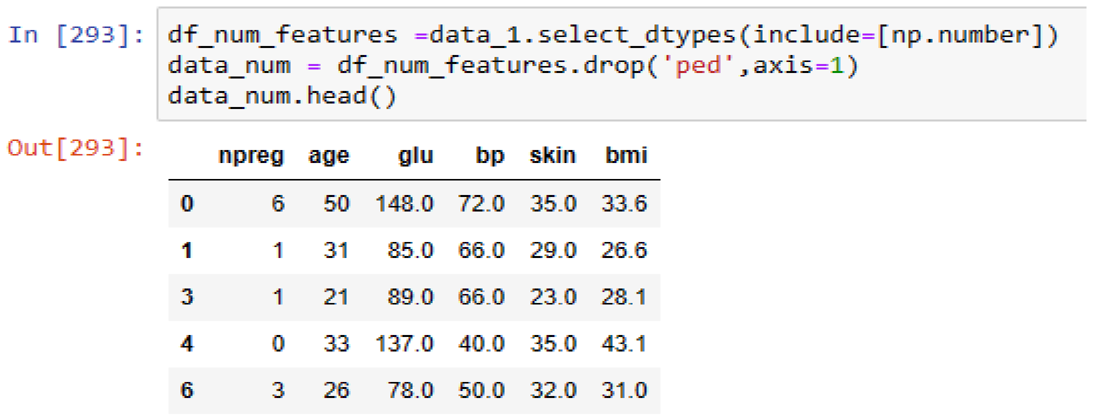


```
In [293]: df_num_features =data_1.select_dtypes(include=[np.number])
          data_num = df_num_features.drop('ped',axis=1)
          data_num.head()
```

Out[293]:

	npreg	age	glu	bp	skin	bmi
0	6	50	148.0	72.0	35.0	33.6
1	1	31	85.0	66.0	29.0	26.6
3	1	21	89.0	66.0	23.0	28.1
4	0	33	137.0	40.0	35.0	43.1
6	3	26	78.0	50.0	32.0	31.0

ii. Scale the Data to Get Variables on the Same Scale

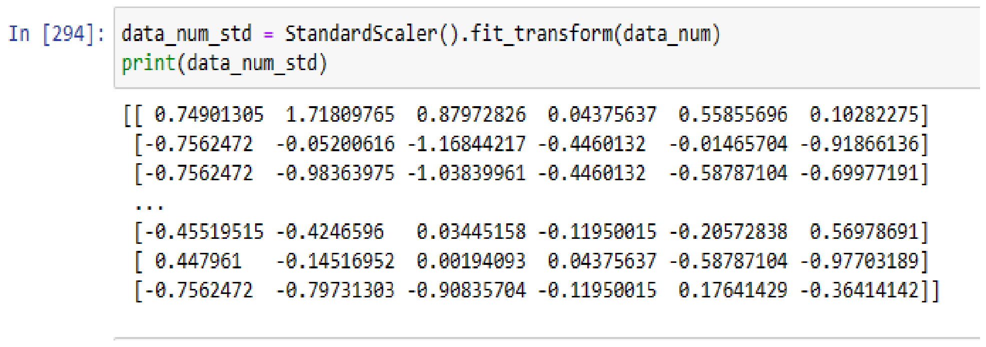


```
In [293]: df_num_features =data_1.select_dtypes(include=[np.number])
          data_num = df_num_features.drop('ped',axis=1)
          data_num.head()
```

Out[293]:

	npreg	age	glu	bp	skin	bmi
0	6	50	148.0	72.0	35.0	33.6
1	1	31	85.0	66.0	29.0	26.6
3	1	21	89.0	66.0	23.0	28.1
4	0	33	137.0	40.0	35.0	43.1
6	3	26	78.0	50.0	32.0	31.0

iii. Compute Covariance Matrix



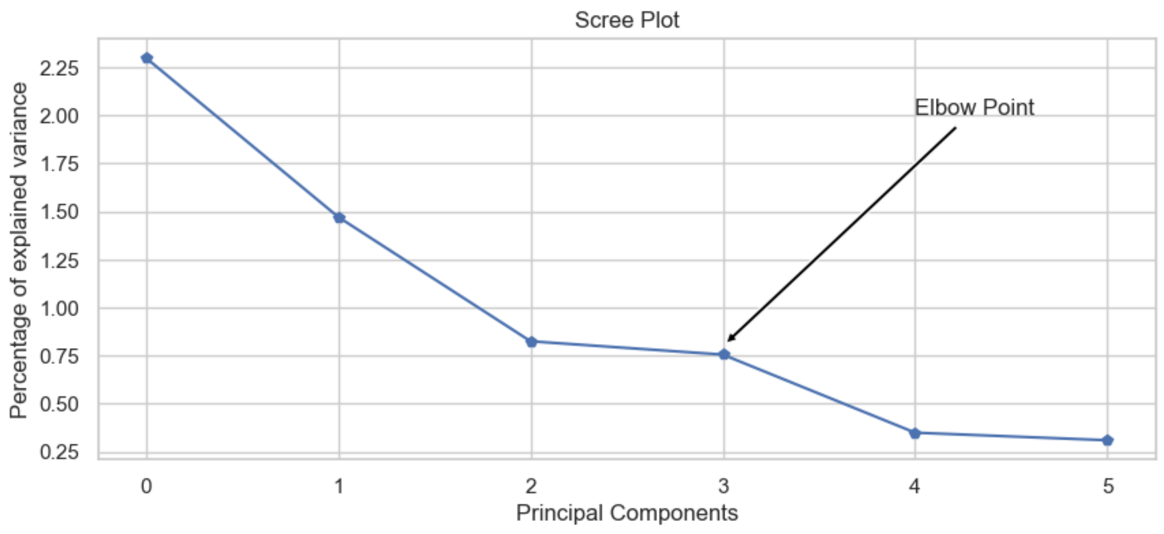
```
In [294]: data_num_std = StandardScaler().fit_transform(data_num)
          print(data_num_std)

[[ 0.74901305  1.71809765  0.87972826  0.04375637  0.55855696  0.10282275]
 [-0.7562472  -0.05200616 -1.16844217 -0.4460132  -0.01465704 -0.91866136]
 [-0.7562472  -0.98363975 -1.03839961 -0.4460132  -0.58787104 -0.69977191]
 ...
 [-0.45519515 -0.4246596   0.03445158 -0.11950015 -0.20572838  0.56978691]
 [ 0.447961   -0.14516952  0.00194093  0.04375637 -0.58787104 -0.97703189]
 [-0.7562472  -0.79731303 -0.90835704 -0.11950015  0.17641429 -0.36414142]]
```

iv. Calculate Eigen Values and Eigen Vectors of the Covariance Matrix

```
In [296]: eig_values, eig_vector = np.linalg.eig(cov_mat)
          print('Eigen values:','\n','\n', eig_values,"\n")
          print('Eigen vectors:','\n','\n',eig_vector,'\n')
```

```
Eigen values:

 [2.2991948  1.47162588 0.31010656 0.34945851 0.82499884 0.75572652]

Eigen vectors:

 [[-0.36607157 -0.54358524 -0.44030593  0.47421335  0.2818529  -0.26892012]
 [-0.45281473 -0.4839029   0.61269163 -0.41773334  0.0524138  -0.0903184 ]
 [-0.35885404  0.0456002  -0.13289832  0.07190307 -0.8978727  -0.2003422 ]
 [-0.41488941 -0.01663631 -0.20873069 -0.02897317 -0.00391307  0.88496709]
 [-0.43110956  0.44434121 -0.40069046 -0.53421526  0.27934406 -0.30452184]
 [-0.4173554   0.52015173  0.45711743  0.55608014  0.18327011 -0.05905298]]
```

v. Decide the Number of Principal Components

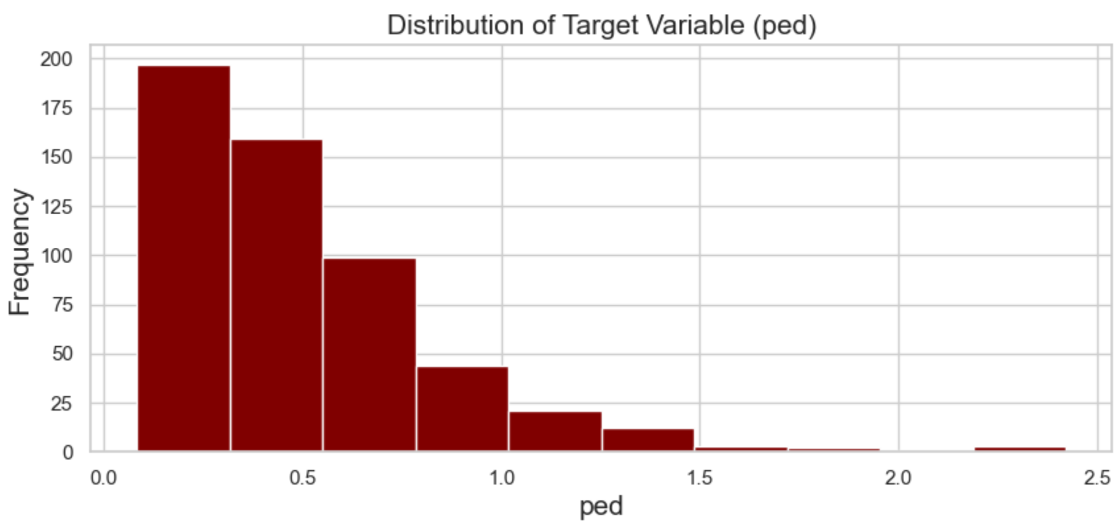


Using the scree plot alone, it is challenging to decide on the number of principal components to select for extracting maximum variations in the data. Therefore, we employed a manual approach, calculating the percentage of variance explained by the first p eigen values. In our case, the first four eigen values exceed 90%. Consequently, we have selected the first four components for further analysis.

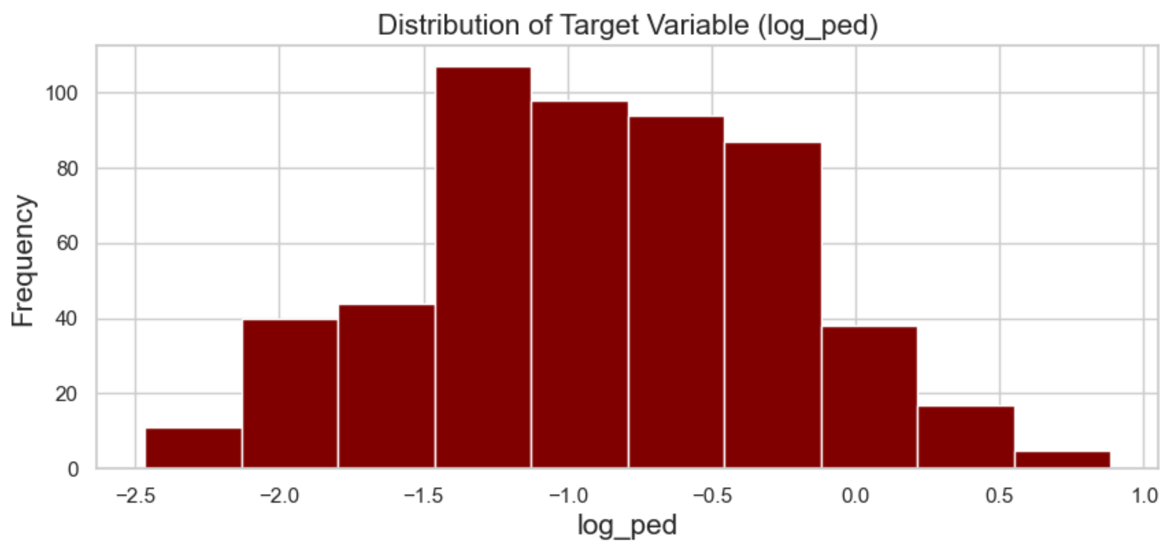
vi. Obtain Principal Components

```
In [299]: eigenvector = eig_vector[:,0:4]
          eigenvector

Out[299]: array([[-0.36607157, -0.54358524, -0.44030593,  0.47421335],
                 [-0.45281473, -0.4839029 ,  0.61269163, -0.41773334],
                 [-0.35885404,  0.0456002 , -0.13289832,  0.07190307],
                 [-0.41488941, -0.01663631, -0.20873069, -0.02897317],
                 [-0.43110956,  0.44434121, -0.40069046, -0.53421526],
                 [-0.4173554 ,  0.52015173,  0.45711743,  0.55608014]])

In [300]: pca_data = pd.DataFrame(data_num_std.dot(eigenvector), columns= ['PC1','PC2','PC3','PC4'])
```

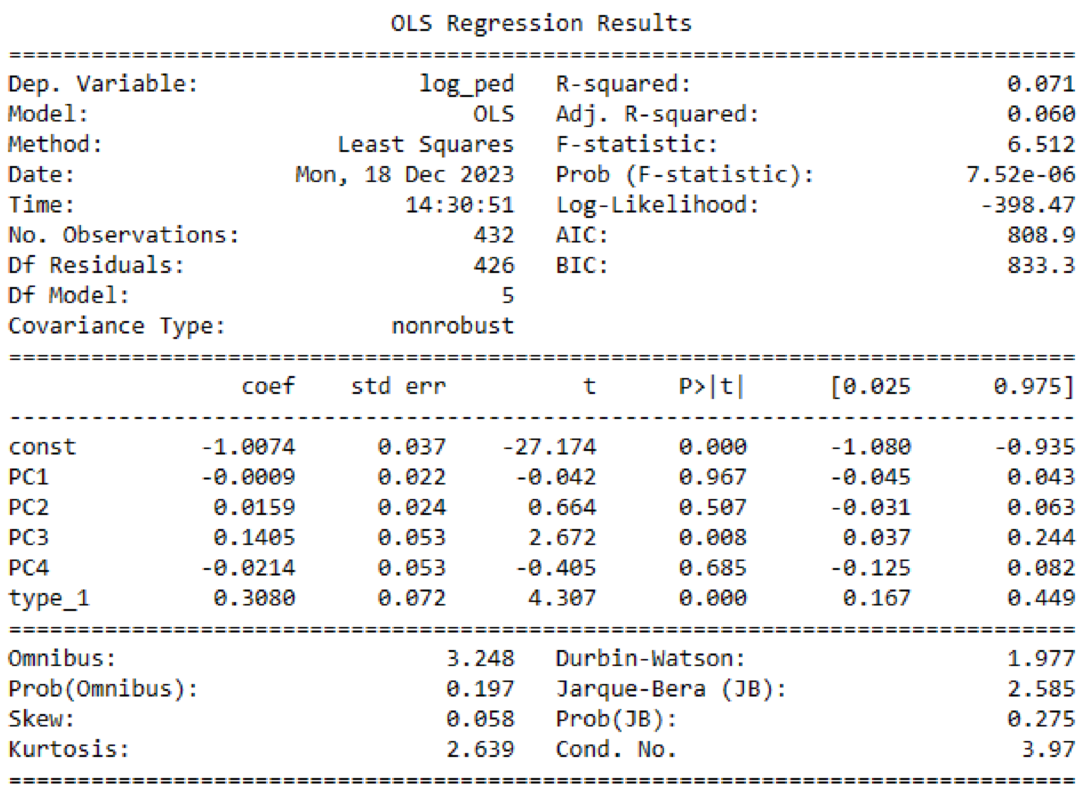
b. Distribution of Target Variable (ped)



The distribution of the target variable is right-skewed. However, the assumption for the target variable in linear regression is a normal distribution. To address the right-skewed nature of the variable, a log transformation is applied to achieve normality.



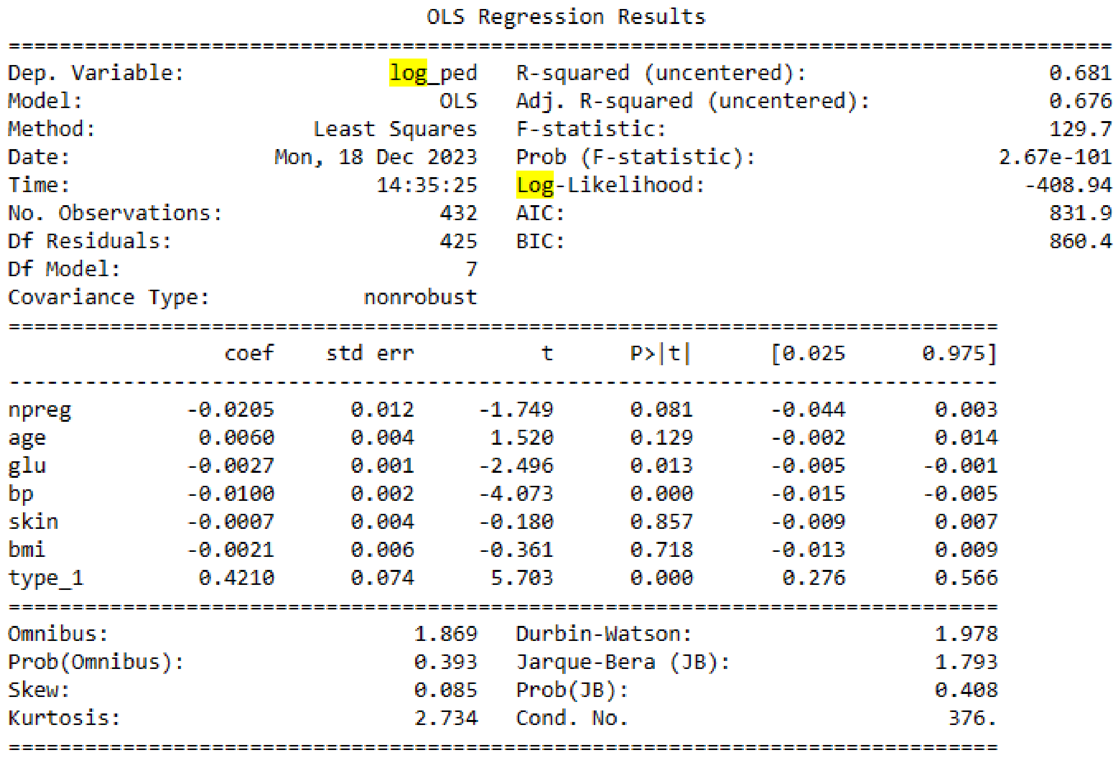
c. Linear Regression Model for PCA Features



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                log_ped   R-squared:                       0.071
Model:                            OLS   Adj. R-squared:                  0.060
Method:                 Least Squares   F-statistic:                     6.512
Date:                Mon, 18 Dec 2023   Prob (F-statistic):           7.52e-06
Time:                        14:30:51   Log-Likelihood:                -398.47
No. Observations:                 432   AIC:                             808.9
Df Residuals:                     426   BIC:                             833.3
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -1.0074      0.037    -27.174      0.000      -1.080      -0.935
PC1           -0.0009      0.022     -0.042      0.967      -0.045       0.043
PC2            0.0159      0.024      0.664      0.507      -0.031       0.063
PC3            0.1405      0.053      2.672      0.008       0.037       0.244
PC4           -0.0214      0.053     -0.405      0.685      -0.125       0.082
type_1         0.3080      0.072      4.307      0.000       0.167       0.449
==============================================================================
Omnibus:                        3.248   Durbin-Watson:                   1.977
Prob(Omnibus):                  0.197   Jarque-Bera (JB):                2.585
Skew:                           0.058   Prob(JB):                        0.275
Kurtosis:                       2.639   Cond. No.                         3.97
==============================================================================
```

The R-squared and adjusted R-squared values are considerably low, indicating poor performance of the model.

3.10.2 Linear Regression Model without Intercept

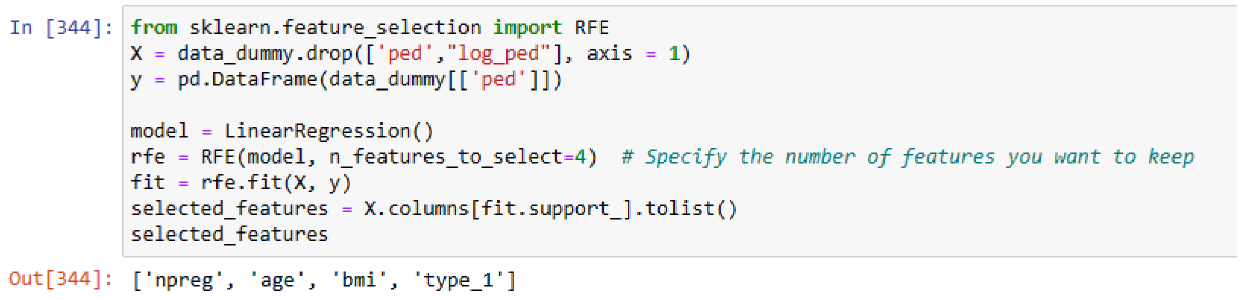


```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                log_ped   R-squared (uncentered):                   0.681
Model:                            OLS   Adj. R-squared (uncentered):              0.676
Method:                 Least Squares   F-statistic:                              129.7
Date:                Mon, 18 Dec 2023   Prob (F-statistic):                   2.67e-101
Time:                        14:35:25   Log-Likelihood:                         -408.94
No. Observations:                 432   AIC:                                      831.9
Df Residuals:                     425   BIC:                                      860.4
Df Model:                           7
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
npreg         -0.0205      0.012     -1.749      0.081      -0.044       0.003
age            0.0060      0.004      1.520      0.129      -0.002       0.014
glu           -0.0027      0.001     -2.496      0.013      -0.005      -0.001
bp            -0.0100      0.002     -4.073      0.000      -0.015      -0.005
skin          -0.0007      0.004     -0.180      0.857      -0.009       0.007
bmi           -0.0021      0.006     -0.361      0.718      -0.013       0.009
type_1         0.4210      0.074      5.703      0.000       0.276       0.566
==============================================================================
Omnibus:                        1.869   Durbin-Watson:                   1.978
Prob(Omnibus):                  0.393   Jarque-Bera (JB):                1.793
Skew:                           0.085   Prob(JB):                        0.408
Kurtosis:                       2.734   Cond. No.                         376.
==============================================================================
```

The model exhibits an R-squared of 0.681 and an adjusted R-squared of 0.676, indicating a performance level of 68%. However, certain features, namely age, skin, and BMI, are found to be insignificant. To address this, feature selection techniques such as Lasso regression, k-best features, and backward elimination methods should be employed to identify and retain the most relevant features.

3.10.3 Linear Regression Model for Recursive Features Elimination

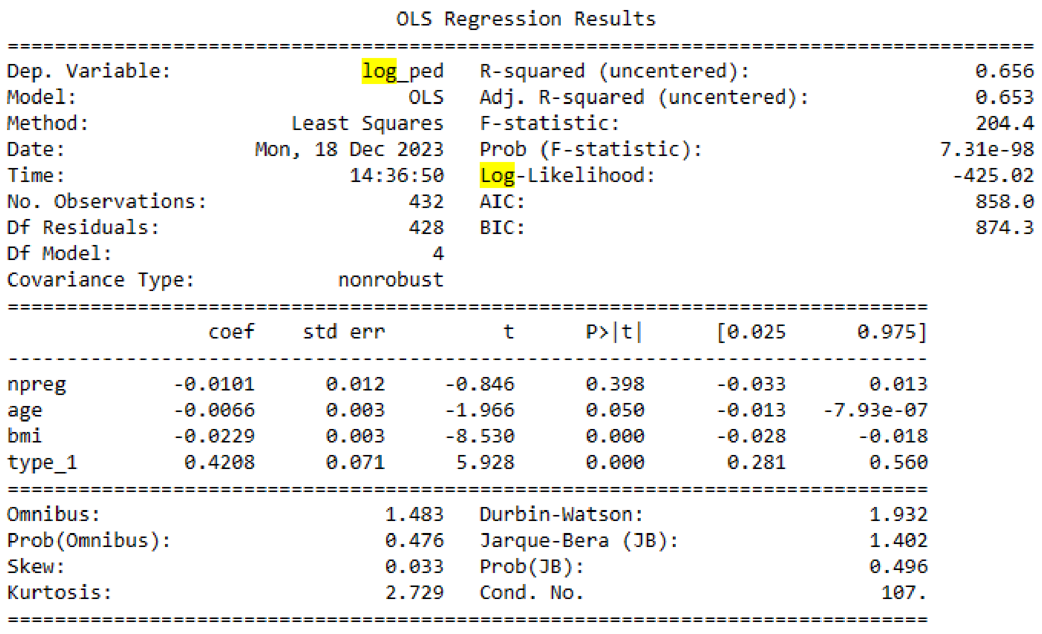
a. Recursive Feature Elimination (RFE):



```
In [344]: from sklearn.feature_selection import RFE
          X = data_dummy.drop(['ped',"log_ped"], axis = 1)
          y = pd.DataFrame(data_dummy[['ped']])

          model = LinearRegression()
          rfe = RFE(model, n_features_to_select=4)  # Specify the number of features you want to keep
          fit = rfe.fit(X, y)
          selected_features = X.columns[fit.support_].tolist()
          selected_features

Out[344]: ['npreg', 'age', 'bmi', 'type_1']
```

The RFE technique has identified the features npreg, age, bmi, and type. Using these selected features, we need to construct a regression model and assess both its performance and the significance of the chosen features.

b. Model Summary

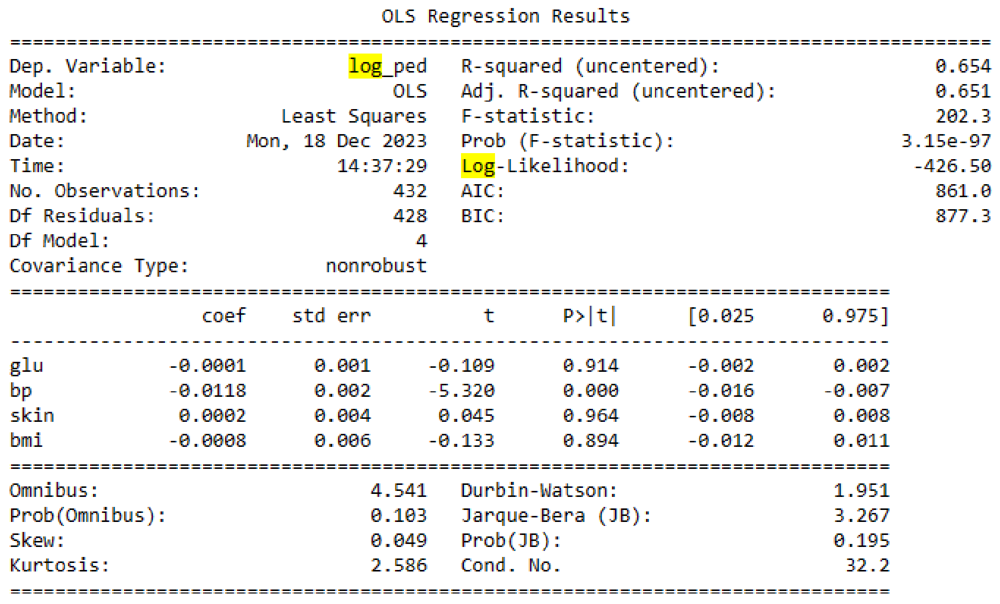


```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                log_ped   R-squared (uncentered):                   0.656
Model:                            OLS   Adj. R-squared (uncentered):              0.653
Method:                 Least Squares   F-statistic:                              204.4
Date:                Mon, 18 Dec 2023   Prob (F-statistic):                    7.31e-98
Time:                        14:36:50   Log-Likelihood:                         -425.02
No. Observations:                 432   AIC:                                      858.0
Df Residuals:                     428   BIC:                                      874.3
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
npreg         -0.0101      0.012     -0.846      0.398      -0.033       0.013
age           -0.0066      0.003     -1.966      0.050      -0.013   -7.93e-07
bmi           -0.0229      0.003     -8.530      0.000      -0.028      -0.018
type_1         0.4208      0.071      5.928      0.000       0.281       0.560
==============================================================================
Omnibus:                        1.483   Durbin-Watson:                   1.932
Prob(Omnibus):                  0.476   Jarque-Bera (JB):                1.402
Skew:                           0.033   Prob(JB):                        0.496
Kurtosis:                       2.729   Cond. No.                         107.
==============================================================================
```

The model demonstrates an R-squared of 0.656 and an adjusted R-squared of 0.653, representing a performance level of 66%. However, it was observed that the feature npreg is deemed insignificant in this context.

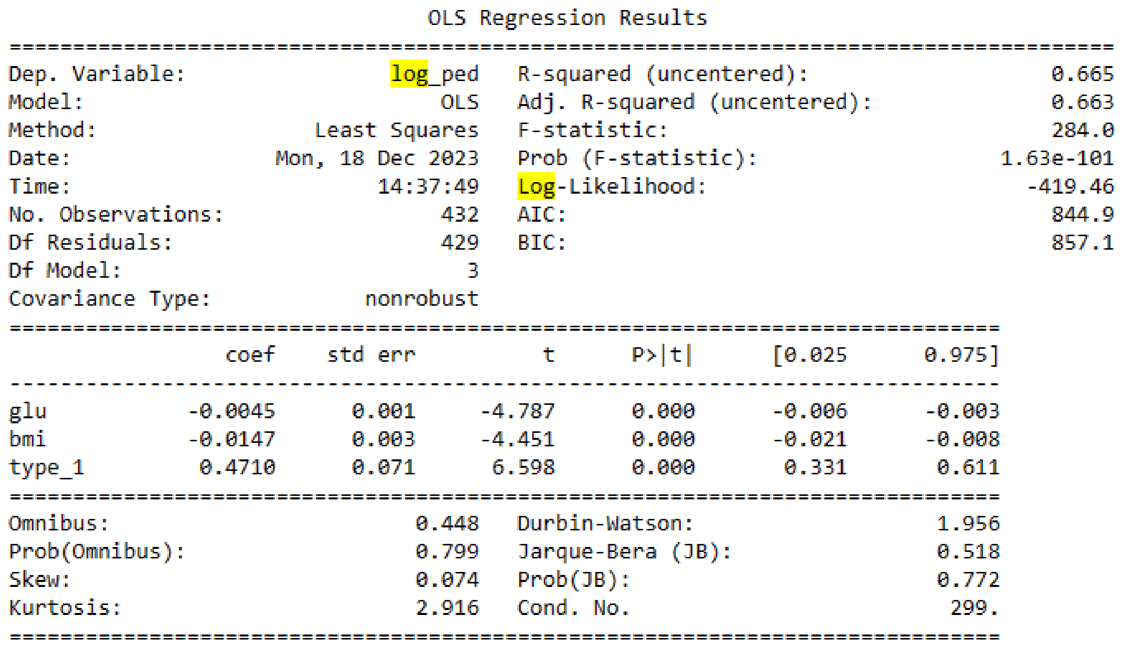
Similarly, we applied Lasso and k-best techniques to select the optimal features, and subsequently, regression models were employed. The tables below provide the model summaries for the Lasso and k-best features models.

3.10.4 Multiple Linear Regression for LASSO (L1 Regularization)



```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                log_ped   R-squared (uncentered):                   0.654
Model:                            OLS   Adj. R-squared (uncentered):              0.651
Method:                 Least Squares   F-statistic:                              202.3
Date:                Mon, 18 Dec 2023   Prob (F-statistic):                    3.15e-97
Time:                        14:37:29   Log-Likelihood:                         -426.50
No. Observations:                 432   AIC:                                      861.0
Df Residuals:                     428   BIC:                                      877.3
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
glu           -0.0001      0.001     -0.109      0.914      -0.002       0.002
bp            -0.0118      0.002     -5.320      0.000      -0.016      -0.007
skin           0.0002      0.004      0.045      0.964      -0.008       0.008
bmi           -0.0008      0.006     -0.133      0.894      -0.012       0.011
==============================================================================
Omnibus:                        4.541   Durbin-Watson:                   1.951
Prob(Omnibus):                  0.103   Jarque-Bera (JB):                3.267
Skew:                           0.049   Prob(JB):                        0.195
Kurtosis:                       2.586   Cond. No.                         32.2
==============================================================================
```

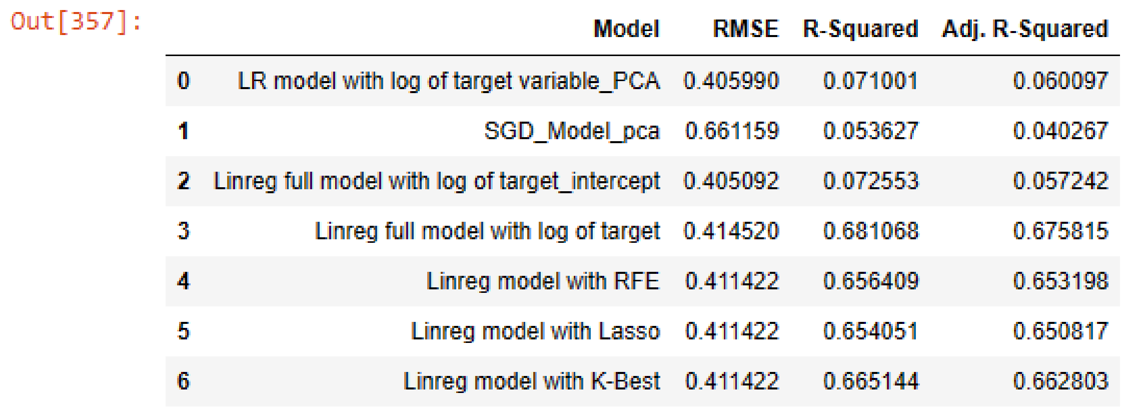
3.10.5 Multiple Linear Regression for K Best Features



```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                log_ped   R-squared (uncentered):                   0.665
Model:                            OLS   Adj. R-squared (uncentered):              0.663
Method:                 Least Squares   F-statistic:                              284.0
Date:                Mon, 18 Dec 2023   Prob (F-statistic):                   1.63e-101
Time:                        14:37:49   Log-Likelihood:                         -419.46
No. Observations:                 432   AIC:                                      844.9
Df Residuals:                     429   BIC:                                      857.1
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
glu           -0.0045      0.001     -4.787      0.000      -0.006      -0.003
bmi           -0.0147      0.003     -4.451      0.000      -0.021      -0.008
type_1         0.4710      0.071      6.598      0.000       0.331       0.611
==============================================================================
Omnibus:                        0.448   Durbin-Watson:                   1.956
Prob(Omnibus):                  0.799   Jarque-Bera (JB):                0.518
Skew:                           0.074   Prob(JB):                        0.772
Kurtosis:                       2.916   Cond. No.                         299.
==============================================================================
```

From the two tables above, it is evident that the K-best features selection model outperforms the Lasso features regression model. In the K-best model, all features are significant at any level of significance, whereas in the Lasso regression model, certain features do not exhibit significance.

3.10.6 Linear Regression Model Performance Metrics



Out[357]:

	Model	RMSE	R-Squared	Adj. R-Squared
0	LR model with log of target variable_PCA	0.405990	0.071001	0.060097
1	SGD_Model_pca	0.661159	0.053627	0.040267
2	Linreg full model with log of target_intercept	0.405092	0.072553	0.057242
3	Linreg full model with log of target	0.414520	0.681068	0.675815
4	Linreg model with RFE	0.411422	0.656409	0.653198
5	Linreg model with Lasso	0.411422	0.654051	0.650817
6	Linreg model with K-Best	0.411422	0.665144	0.662803

From the above metrics, we can observe that the linear regression model for K-best features performs better compared to other models. In this model, all features are significant. However, in the standard linear regression model, some features are found to be insignificant. Therefore, we choose the linear regression model with K-best features for further analysis and deployment.

3.10.7 Performing k-Fold Cross-Validation on Linear Regression with Best Features Model

Performing k-fold cross-validation for linear regression involves splitting the dataset into k folds, training the model on k-1 folds, and testing on the remaining fold. This process is repeated k times, each time using a different fold as the test set, and then the performance metrics are averaged over the k folds.

```
Cross-Validation Scores: [-0.21852189 -0.22947818 -0.32575763 -0.25001804]
Mean CV Score: -0.2559439342203844
Std CV Score: 0.04186261537915351
```

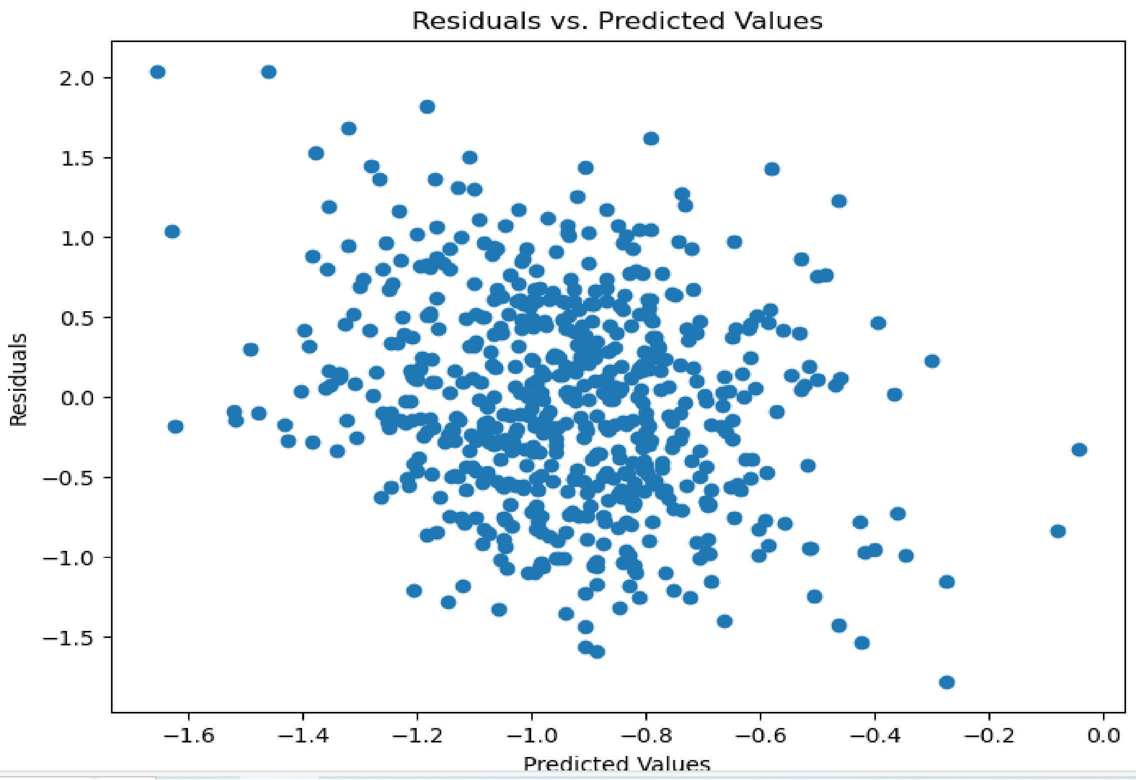
In this case, the mean squared error values are relatively low, indicating that the model is performing well on the cross-validated data. The standard deviation is also relatively small, suggesting consistency across folds.

3.10. Residual Analysis

Create residual plots to visually assess the assumptions of linear regression. Common plots include the scatter plot of residuals against predicted values, a histogram of residuals, and a Q-Q plot:

a.Evaluate Homoscedasticity:

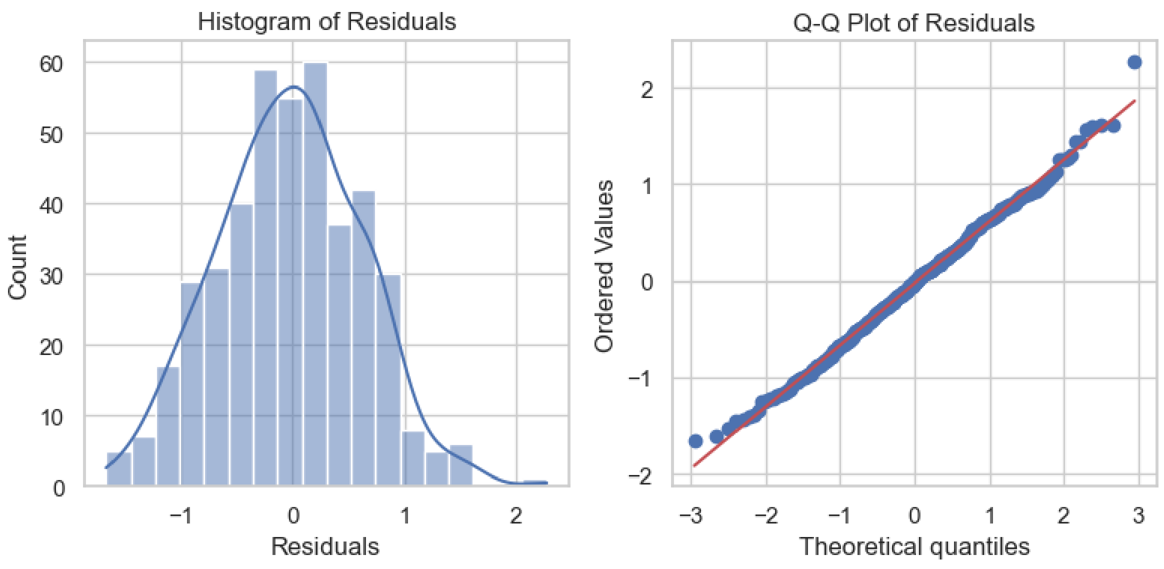
Check for homoscedasticity using a plot of residuals against predicted values. This type of plot is useful for identifying patterns or trends in the residuals. Ideally, we want to see a random scatter of points with no clear pattern, indicating that the residuals are homoscedastic. If we observe any systematic patterns (e.g., a funnel shape, curvature), it may suggest that the assumption of homoscedasticity is violated, and we might need to consider transformations or other model adjustments.



In the scatter plot above, we can observe a random scatter of points with no clear pattern, suggesting that the residuals exhibit homoscedasticity. The absence of any systematic trend in the distribution of residuals indicates that the variance of the errors remains constant across different levels of predicted values.

c. Check for Normality:

Assess the normality of residuals using a Jarque-Bera test, histogram and a Q-Q plot



```
Jarque-Bera statistic: 0.5183927359564329
P-value: 0.7716714765970492
The Residuals follow Normal distribution.
```

Based on the distribution plot, Q-Q plot, and the Jarque-Bera test, it has been confirmed that the residuals follow a normal distribution. This observation supports the assumption that the residuals are normally distributed, a key requirement in linear regression analysis.

d. Durbin-Watson Test:

The Durbin-Watson test checks for the presence of autocorrelation in the residuals. A value around 2 suggests no autocorrelation.

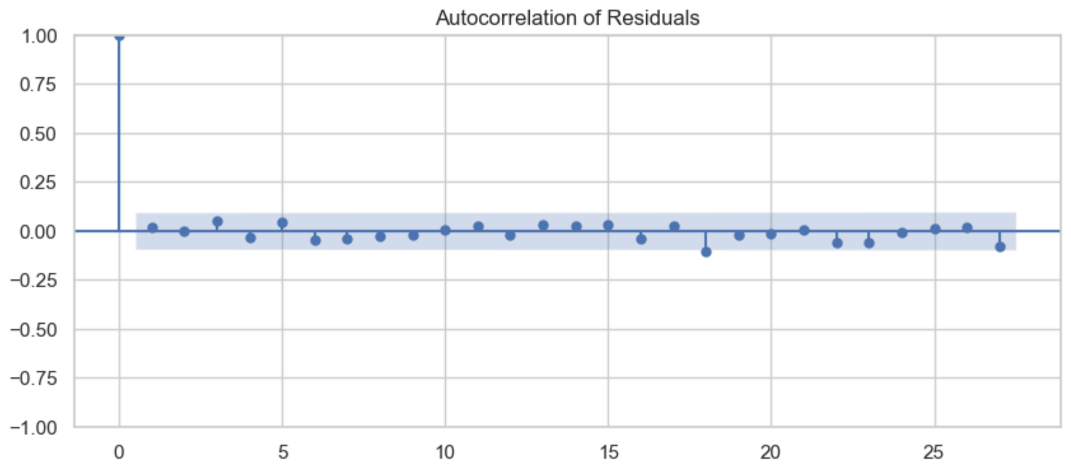
```
In [195]: from statsmodels.stats.stattools import durbin_watson

          # Calculate Durbin-Watson statistic
          durbin_watson_stat = durbin_watson(linreg_logmodel_Kbest.resid)
          print("Durbin-Watson Statistic:", durbin_watson_stat)
```

```
Durbin-Watson Statistic: 1.9485260033637735
```

In this case, a Durbin-Watson statistic of 1.95 is very close to 2, suggesting that there is no strong evidence of significant autocorrelation in the residuals. The value being close to 2 indicates that the residuals are approximately uncorrelated with each other.

e. Autocorrelation Plot:

Visualize the autocorrelation of residuals using an autocorrelation plot:



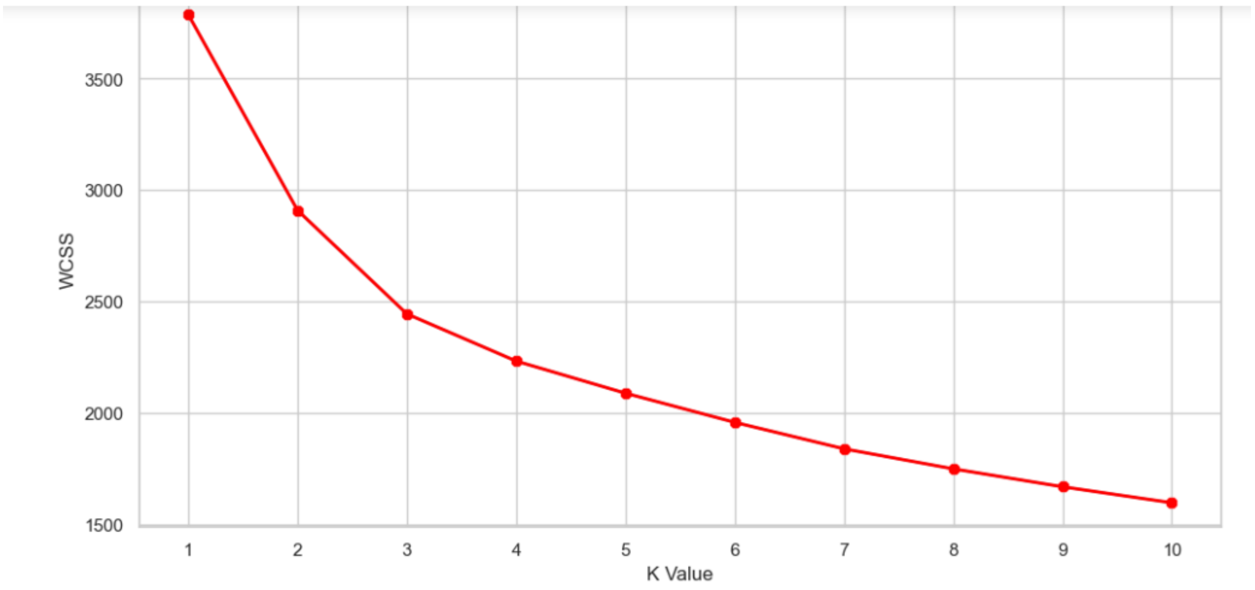
The Durbin-Watson test statistic is approximately equal to 2, suggesting that the autocorrelation function (ACF) of the error term is nearly zero. This indicates the absence of autocorrelation in the error term. Additionally, the Jarque-Bera (JB) test yields a p-value of 0.77. As the p-value is greater than the significance level (commonly 0.05), we accept the null hypothesis. This result implies that the residuals are normally distributed. Altogether, the Durbin-Watson test indicates no autocorrelation, and the JB test supports the normality assumption of the residuals.

3.12 Cluster Analysis

3.12.1 Build a Model with Multiple K

We constructed our models using the silhouette score method. Silhouette is a technique for interpreting and validating the consistency within clusters of data. We do not know the optimal number of clusters that would yield the most useful results. Therefore, we create clusters by varying K from 2 to 8 and subsequently determine the optimum number of clusters (K) with the assistance of the silhouette score.

```
In [384]: import warnings
          warnings.filterwarnings("ignore")
          from sklearn.cluster import KMeans
          from sklearn.metrics import silhouette_score
          n_clusters=[2,3,4,5,6,7,8]
          for K in n_clusters:
              cluster=KMeans(n_clusters=K,random_state=10)
              predict=cluster.fit_predict(features_scaled)
              score=silhouette_score(features_scaled,predict,random_state=10)
              print("For n_clusters={}, silhoutte score is {}".format(K,score))

          For n_clusters=2, silhoutte score is 0.23295225438615771
          For n_clusters=3, silhoutte score is 0.2162184537793981
          For n_clusters=4, silhoutte score is 0.17184136416592352
          For n_clusters=5, silhoutte score is 0.17201048775548997
          For n_clusters=6, silhoutte score is 0.16968199563891143
          For n_clusters=7, silhoutte score is 0.17378025698538757
          For n_clusters=8, silhoutte score is 0.17441664982213687
```

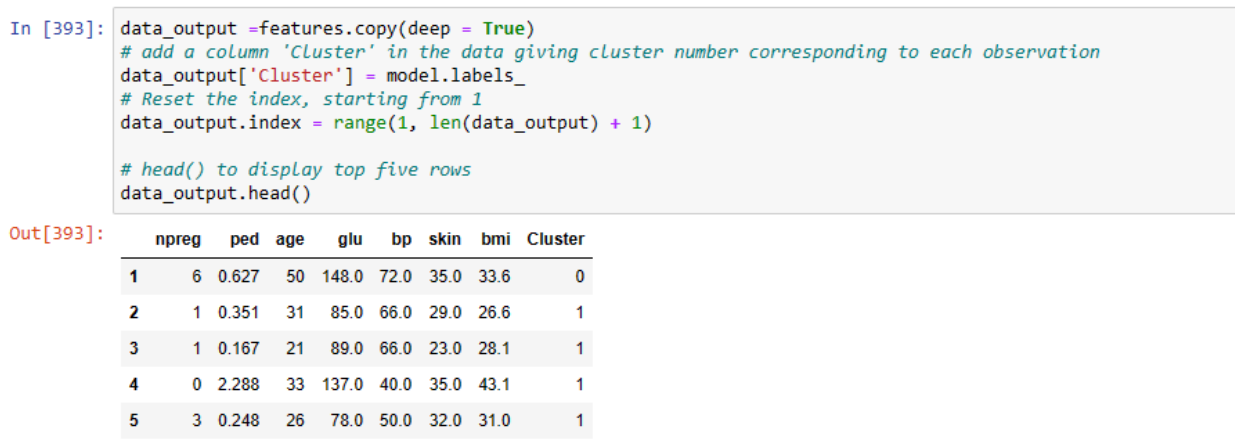


The optimal value for K is identified by the highest silhouette score. From the above output, it is evident that, for K = 2, the silhouette score is the highest. Consequently, we construct the clusters with K = 2.

3.12.2 Building a K-Means model for K = 2

```
In [385]: # building a K-Means model for K = 2
          model = KMeans(n_clusters= 2, random_state= 10)

          # fit the model
          model.fit(features_scaled)

Out[385]: KMeans(n_clusters=2, random_state=10)
```

3.12.3 Retrieve the Clusters

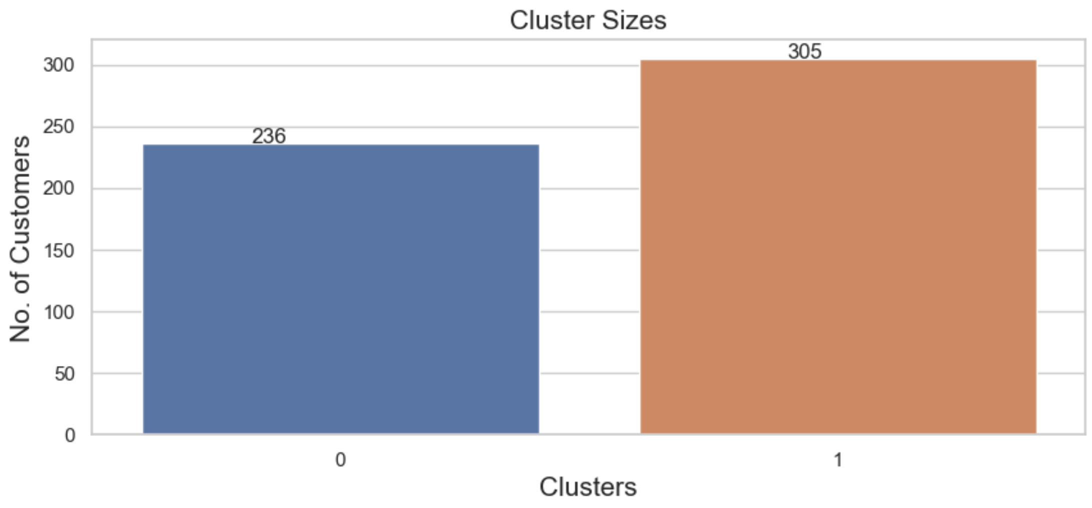


```
In [393]: data_output =features.copy(deep = True)
          # add a column 'Cluster' in the data giving cluster number corresponding to each observation
          data_output['Cluster'] = model.labels_
          # Reset the index, starting from 1
          data_output.index = range(1, len(data_output) + 1)

          # head() to display top five rows
          data_output.head()
```

```
Out[393]:
     npreg    ped  age    glu    bp  skin   bmi  Cluster
  1      6  0.627   50  148.0  72.0  35.0  33.6        0
  2      1  0.351   31   85.0  66.0  29.0  26.6        1
  3      1  0.167   21   89.0  66.0  23.0  28.1        1
  4      0  2.288   33  137.0  40.0  35.0  43.1        1
  5      3  0.248   26   78.0  50.0  32.0  31.0        1
```

We have added a column named 'cluster' to the data frame, indicating the cluster number for each observation.

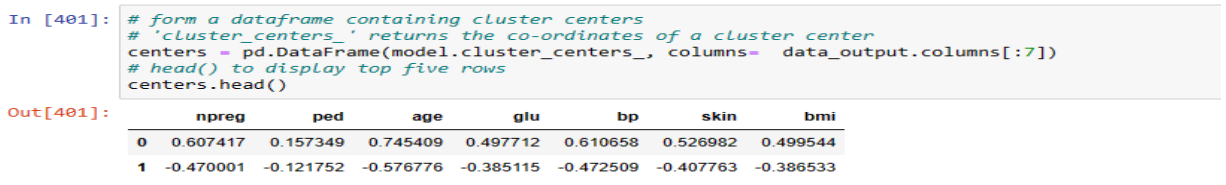
3.12.4 Plot a Bar Plot to Visualize the Cluster Sizes



The second cluster is slightly larger, containing 305 customers.

3.12.5 Cluster Centers

The cluster centers can give information about the variables belonging to the clusters



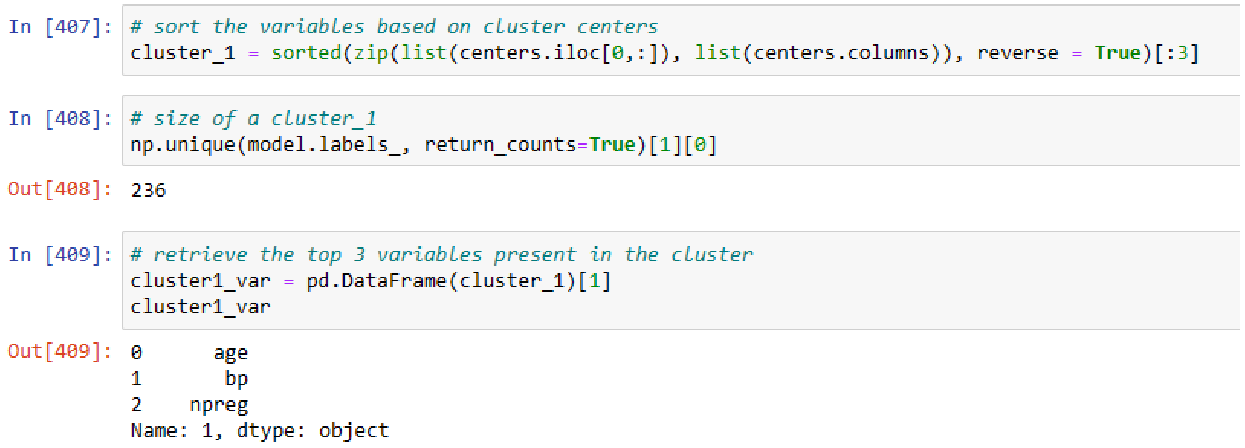
```
In [401]: # form a dataframe containing cluster centers
          # 'cluster_centers_' returns the co-ordinates of a cluster center
          centers = pd.DataFrame(model.cluster_centers_, columns=  data_output.columns[:7])
          # head() to display top five rows
          centers.head()
```

```
Out[401]:
        npreg        ped        age        glu         bp       skin        bmi
  0  0.607417   0.157349   0.745409   0.497712   0.610658   0.526982   0.499544
  1 -0.470001  -0.121752  -0.576776  -0.385115  -0.472509  -0.407763  -0.386533
```

Now, extract the variables in each of the clusters and attempt to assign a name to each cluster based on the variables

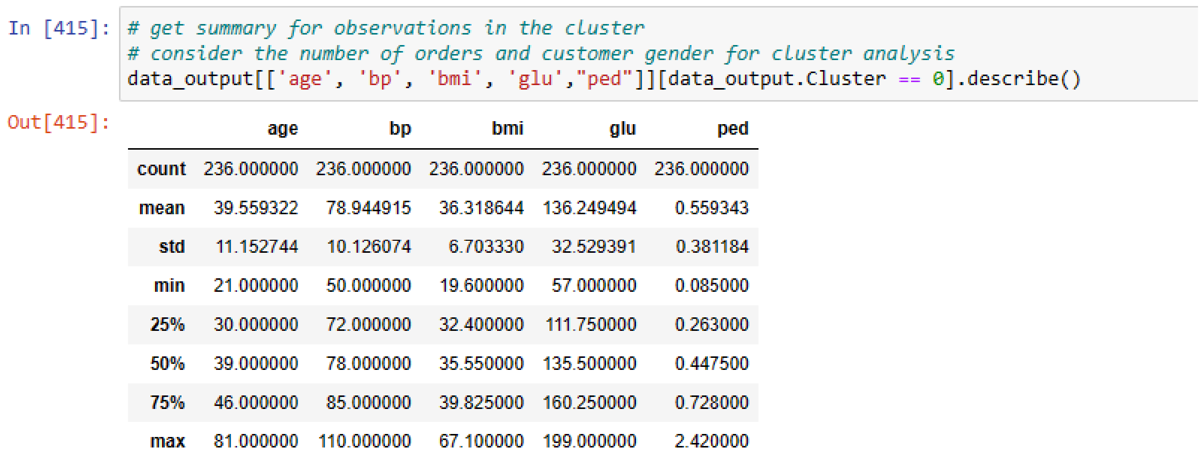
3.12.6 Analysis of Cluster-1

Here, we analyze the first cluster, initially examining its size, followed by sorting the variables that belong to the cluster. Subsequently, we compute a statistical summary for the observations within the cluster.



```
In [407]: # sort the variables based on cluster centers
          cluster_1 = sorted(zip(list(centers.iloc[0,:]), list(centers.columns)), reverse = True)[:3]

In [408]: # size of a cluster_1
          np.unique(model.labels_, return_counts=True)[1][0]

Out[408]: 236

In [409]: # retrieve the top 3 variables present in the cluster
          cluster1_var = pd.DataFrame(cluster_1)[1]
          cluster1_var

Out[409]: 0      age
          1       bp
          2    npreg
          Name: 1, dtype: object
```

The first cluster comprises 236 observations. The top three variables in this cluster, ranked by importance, are age, blood pressure (bp), and number of pregnancies (npreg). This suggests that these factors play a significant role within the cluster and may warrant further investigation or attention in the context of the overall dataset.

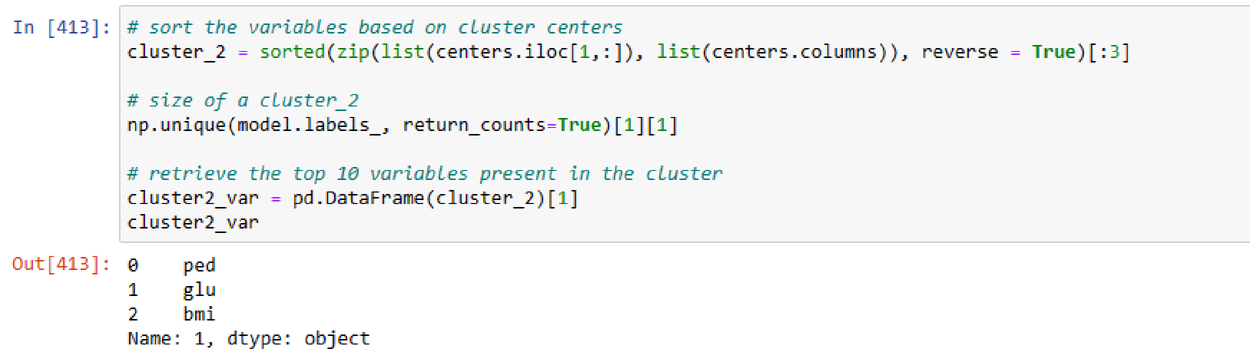


```
In [415]: # get summary for observations in the cluster
          # consider the number of orders and customer gender for cluster analysis
          data_output[['age', 'bp', 'bmi', 'glu',"ped"]][data_output.Cluster == 0].describe()

Out[415]:
```

	age	bp	bmi	glu	ped
count	236.000000	236.000000	236.000000	236.000000	236.000000
mean	39.559322	78.944915	36.318644	136.249494	0.559343
std	11.152744	10.126074	6.703330	32.529391	0.381184
min	21.000000	50.000000	19.600000	57.000000	0.085000
25%	30.000000	72.000000	32.400000	111.750000	0.263000
50%	39.000000	78.000000	35.550000	135.500000	0.447500
75%	46.000000	85.000000	39.825000	160.250000	0.728000
max	81.000000	110.000000	67.100000	199.000000	2.420000

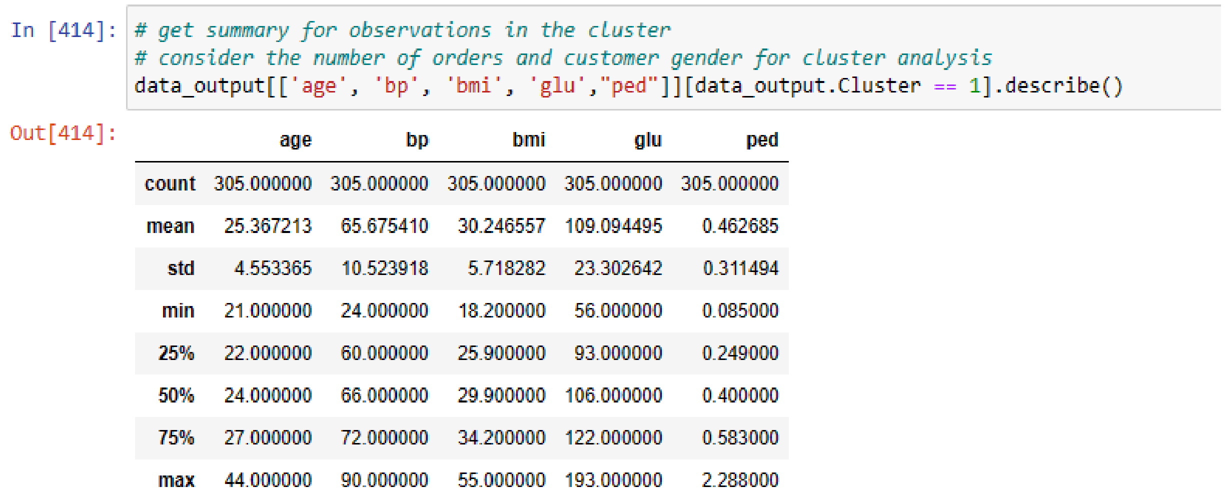
3.12.7 Analysis of Cluster-2

Here, we analyze the second cluster, sorting the variables that belong to the cluster. Subsequently, we compute a statistical summary for the observations within the cluster.



```
In [413]: # sort the variables based on cluster centers
          cluster_2 = sorted(zip(list(centers.iloc[1,:]), list(centers.columns)), reverse = True)[:3]

          # size of a cluster_2
          np.unique(model.labels_, return_counts=True)[1][1]

          # retrieve the top 10 variables present in the cluster
          cluster2_var = pd.DataFrame(cluster_2)[1]
          cluster2_var

Out[413]: 0    ped
          1    glu
          2    bmi
          Name: 1, dtype: object
```

The top three variables in this cluster, ranked by importance, are ped, glu and bmi. This suggests that these factors play a significant role within the cluster and may warrant further investigation or attention in the context of the overall dataset.



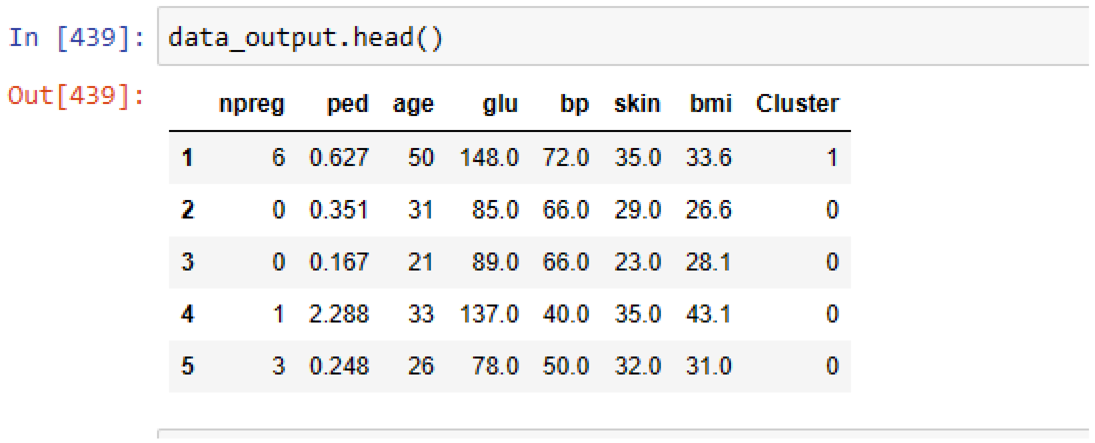
```
In [414]: # get summary for observations in the cluster
          # consider the number of orders and customer gender for cluster analysis
          data_output[['age', 'bp', 'bmi', 'glu',"ped"]][data_output.Cluster == 1].describe()

Out[414]:
```

	age	bp	bmi	glu	ped
count	305.000000	305.000000	305.000000	305.000000	305.000000
mean	25.367213	65.675410	30.246557	109.094495	0.462685
std	4.553365	10.523918	5.718282	23.302642	0.311494
min	21.000000	24.000000	18.200000	56.000000	0.085000
25%	22.000000	60.000000	25.900000	93.000000	0.249000
50%	24.000000	66.000000	29.900000	106.000000	0.400000
75%	27.000000	72.000000	34.200000	122.000000	0.583000
max	44.000000	90.000000	55.000000	193.000000	2.288000

It can be observed that, in the second cluster, most women exhibit lower mean values for features such as ped, age, bp, bmi, and glucose compared to the first cluster. Higher values in these features are often associated with diabetes. Therefore, we may categorize the first cluster as the 'diabetes group' and the second cluster as the 'non-diabetes group,' suggesting potential differences in health characteristics between the two clusters.

3.12.8 Data Frame of Cluster Analysis

a.Data Frame



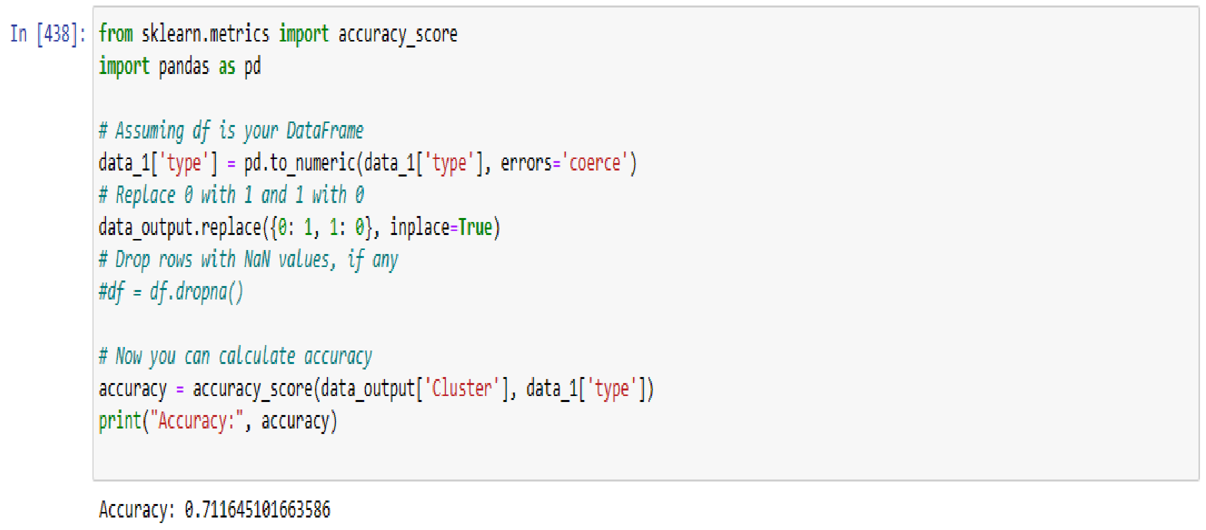
In [439]:
```
data_output.head()
```

Out[439]:

	npreg	ped	age	glu	bp	skin	bmi	Cluster
1	6	0.627	50	148.0	72.0	35.0	33.6	1
2	0	0.351	31	85.0	66.0	29.0	26.6	0
3	0	0.167	21	89.0	66.0	23.0	28.1	0
4	1	2.288	33	137.0	40.0	35.0	43.1	0
5	3	0.248	26	78.0	50.0	32.0	31.0	0

In this data frame, '0' represents non-diabetes, and '1' represents diabetes. These labels were assigned through cluster analysis. However, we have the actual labels available, allowing us to compare them with the cluster-assigned labels and calculate the accuracy score.

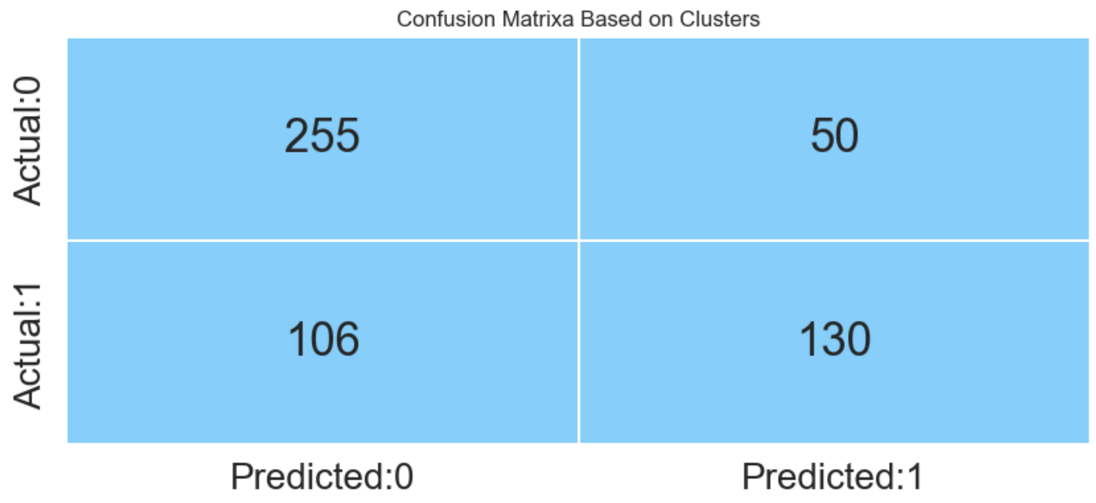
b. Accuracy Score



In [438]:
```
from sklearn.metrics import accuracy_score
import pandas as pd

# Assuming df is your DataFrame
data_1['type'] = pd.to_numeric(data_1['type'], errors='coerce')
# Replace 0 with 1 and 1 with 0
data_output.replace({0: 1, 1: 0}, inplace=True)
# Drop rows with NaN values, if any
#df = df.dropna()

# Now you can calculate accuracy
accuracy = accuracy_score(data_output['Cluster'], data_1['type'])
print("Accuracy:", accuracy)
```

```
Accuracy: 0.711645101663586
```

c. Confusion Matrix



The accuracy score is 71.16%, suggesting that the cluster labeling is correct in approximately 71 out of 100 instances. This indicates a reasonably good performance of the cluster labeling method.

CHAPTER 4

CONCLUSION

Diabetes, a leading global cause of death, is notably prevalent in India. Despite extensive medical research, a definitive cure remains difficult to track down.

This project explores associations between specific characteristics in the studied population and a higher likelihood of diabetes, utilizing machine learning models. Identifying these sub-groups (features) is valuable for understanding risk factors.

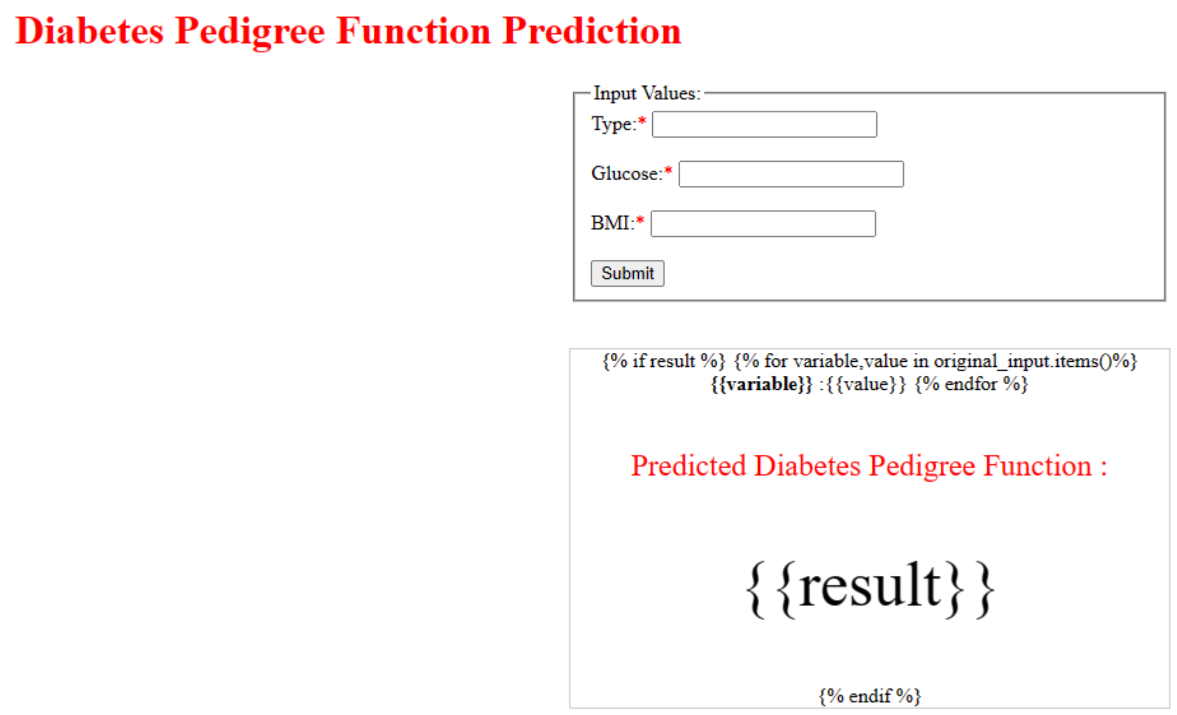
We chose Logistic Regression with the Backward Elimination method to predict the probability of the target outcome—whether a person is diabetic. The model suggests that the number of pregnancies, Diabetes Pedigree Function (ped), glucose (glu), and BMI are significant variables contributing to diabetes factors.

Next, we applied several models to estimate ped, selecting a regression model for the K-best features approach, which outperformed other models. This model confirmed that glu, bmi, and diabetes status (whether a person is diabetic or not) are significant contributions to the ped.

Further research and exploration of these identified risk factors are essential for advancing our efforts in preventing and managing diabetes, a global health concern with particular prominence in India.

CHAPTER VI

DEPLOYMENTS



Diabetes Pedigree Function Prediction

Input Values:

Type:*

Glucose:*

BMI:*

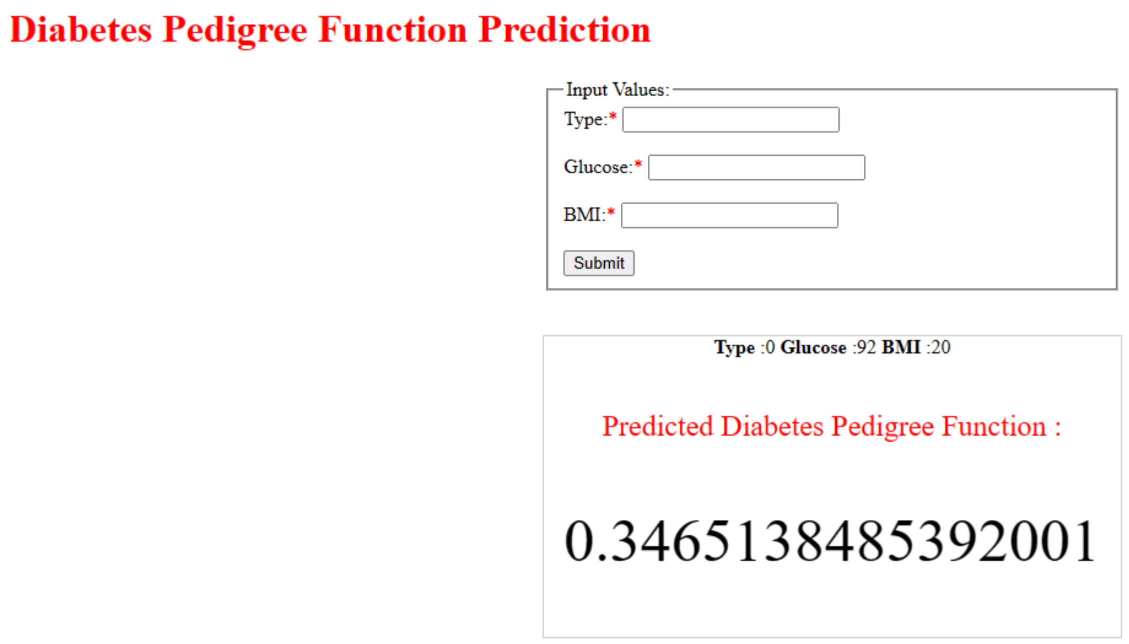
Submit

{% if result %} {% for variable,value in original_input.items()%} **{{variable}}** :{{value}} {% endfor %}

Predicted Diabetes Pedigree Function :

{{result}}

{% endif %}



Diabetes Pedigree Function Prediction

Input Values:

Type:*

Glucose:*

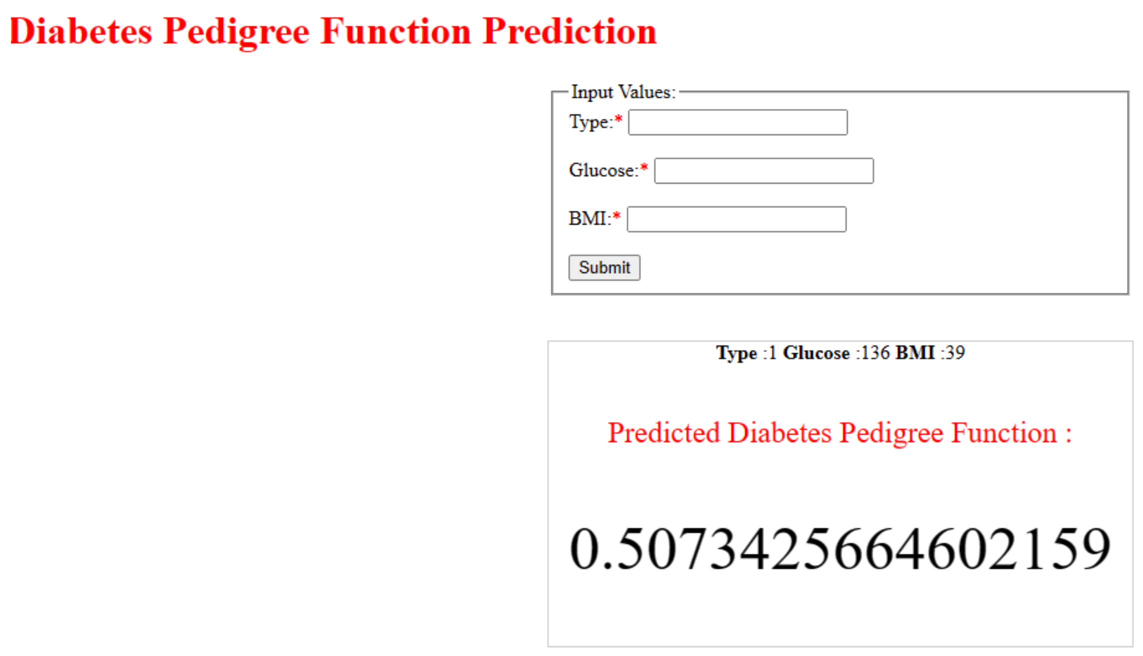
BMI:*

Submit

Type :0 **Glucose** :92 **BMI** :20

Predicted Diabetes Pedigree Function :

0.3465138485392001



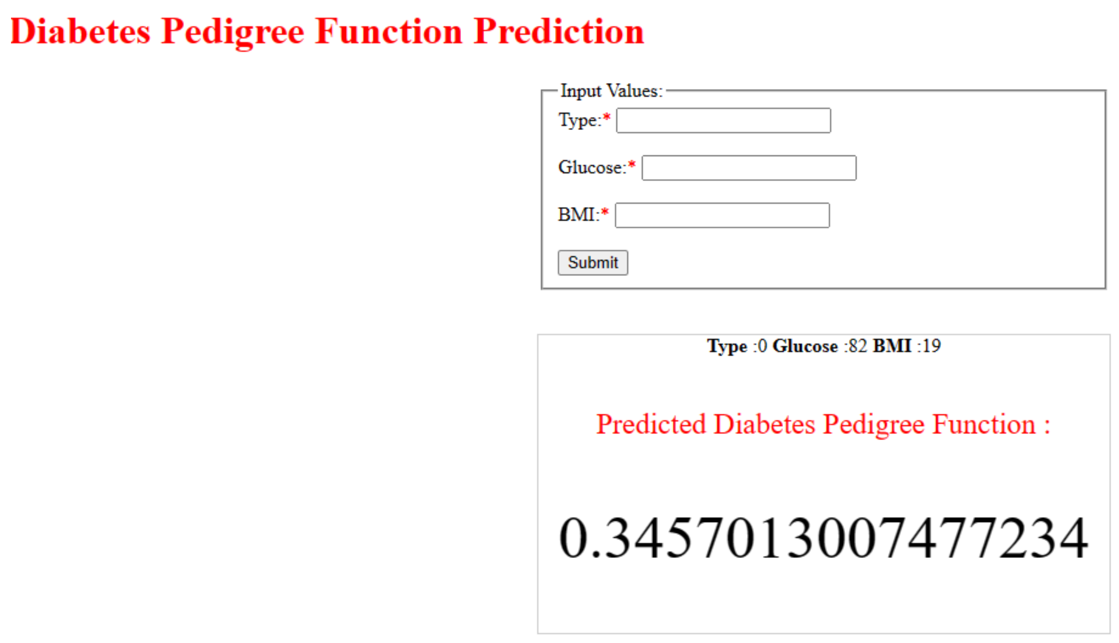
Diabetes Pedigree Function Prediction

Input Values:
Type:*
Glucose:*
BMI:*
Submit

Type :1 **Glucose** :136 **BMI** :39

Predicted Diabetes Pedigree Function :

0.5073425664602159



Diabetes Pedigree Function Prediction

Input Values:
Type:*
Glucose:*
BMI:*
Submit

Type :0 **Glucose** :82 **BMI** :19

Predicted Diabetes Pedigree Function :

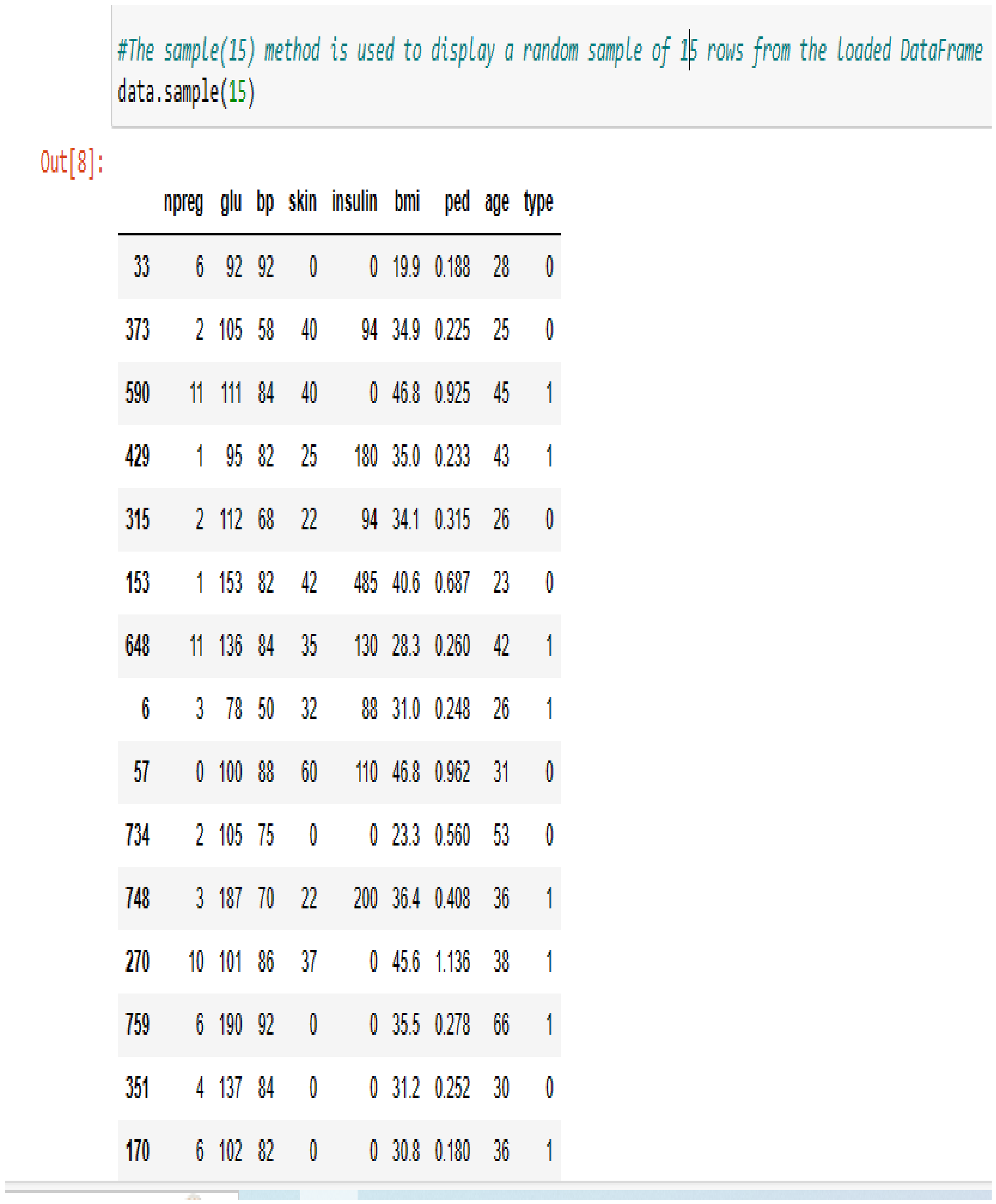
0.3457013007477234

REFERENCES:

- Hosmer, D. &Lemeshow, S. (2000). *Applied Logistic Regression (Second Edition)*. New York: John Wiley & Sons, Inc.
- Long, J. Scott (1997). Regression Models for Categorical and Limited Dependent Variables. *Thousand Oaks, CA: Sage Publications*.
- Everitt, B.S., Landau, S. and Leese, M. (2001), *Cluster Analysis, Fourth edition, Arnold.*
- Manly, B.F.J. (2005), *Multivariate Statistical Methods: A primer*, Third edition, Chapman and Hall.
- Rencher, A.C. (2002), *Methods of Multivariate Analysis*, Second edition, Wiley.
- Jack Johnston and John Dinardo(1984), *Econometric Methods*,John Wiley
- M.Strano; B.M. Colosimo (2006), International Journal of Machine Tools and Manufacture. 46 (6): 673–682. *doi:10.1016/j.ijmachtools.2005.07.005.*

APPENDIX

I. Sample Data:



```
#The sample(15) method is used to display a random sample of 15 rows from the loaded DataFrame
data.sample(15)
```

Out[8]:

	npreg	glu	bp	skin	insulin	bmi	ped	age	type
33	6	92	92	0	0	19.9	0.188	28	0
373	2	105	58	40	94	34.9	0.225	25	0
590	11	111	84	40	0	46.8	0.925	45	1
429	1	95	82	25	180	35.0	0.233	43	1
315	2	112	68	22	94	34.1	0.315	26	0
153	1	153	82	42	485	40.6	0.687	23	0
648	11	136	84	35	130	28.3	0.260	42	1
6	3	78	50	32	88	31.0	0.248	26	1
57	0	100	88	60	110	46.8	0.962	31	0
734	2	105	75	0	0	23.3	0.560	53	0
748	3	187	70	22	200	36.4	0.408	36	1
270	10	101	86	37	0	45.6	1.136	38	1
759	6	190	92	0	0	35.5	0.278	66	1
351	4	137	84	0	0	31.2	0.252	30	0
170	6	102	82	0	0	30.8	0.180	36	1

II. Sample Program

```
#Pandas is a powerful data manipulation library for Python.
import pandas as pd

#NumPy is a numerical computing library for Python.
import numpy as np

#Matplotlib is a plotting library for creating static, interactive, and animated visualizations in Python.
import matplotlib.pyplot as plt

#ListedColormap is a class in Matplotlib used to create a colormap from a list of colors.
from matplotlib.colors import ListedColormap

#Seaborn is a statistical data visualization library based on Matplotlib.
import seaborn as sns

#is_string_dtype is a function from Pandas used to check if a dtype is of object type.
from pandas.api.types import is_string_dtype

#StandardScaler is a preprocessing technique used to standardize features by removing the mean and scaling to
unit variance.
from sklearn.preprocessing import StandardScaler

#train_test_split is a function in scikit-learn used for splitting a dataset into training and testing sets.
from sklearn.model_selection import train_test_split
                                                                                    In [10]:

#The metrics module in scikit-learn provides various metrics for evaluating model performance.
from sklearn import metrics

#LogisticRegression is a class in scikit-learn used for logistic regression modeling.
from sklearn.linear_model import LogisticRegression

#classification_report is a function in scikit-learn that generates a text report showing the main classification
metrics.
from sklearn.metrics import classification_report

#cohen_kappa_score is a function in scikit-learn used for calculating the Cohen's kappa statistic.
from sklearn.metrics import cohen_kappa_score

#confusion_matrix is a function in scikit-learn that computes the confusion matrix to evaluate the accuracy of a
classification.
from sklearn.metrics import confusion_matrix
```

```
#roc_auc_score is a function in scikit-learn used for computing the area under the ROC AUC.
from sklearn.metrics import roc_auc_score

#roc_curve is a function in scikit-learn used for generating receiver operating characteristic (ROC) curves.
from sklearn.metrics import roc_curve

#SGDClassifier is a class in scikit-learn implementing linear classifiers with Stochastic Gradient Descent training.
from sklearn.linear_model import SGDClassifier

#DecisionTreeClassifier is a class in scikit-learn for building decision tree models.
from sklearn.tree import DecisionTreeClassifier

#GridSearchCV is a class in scikit-learn for hyperparameter tuning using grid search.
from sklearn.model_selection import GridSearchCV

#The tree module in scikit-learn provides tools for working with decision trees.
from sklearn import tree

#export_graphviz is a function in scikit-learn for exporting decision tree models to Graphviz format.
from sklearn.tree import export_graphviz
```

In [11]:

```
#Statsmodels is a library for estimating and testing statistical models.
import statsmodels
import statsmodels.api as sm

#SVC is a class in scikit-learn implementing Support Vector Classification.
from sklearn.svm import SVC

#GaussianNB is a class in scikit-learn implementing Gaussian Naive Bayes classification.
from sklearn.naive_bayes import GaussianNB

#KNeighborsClassifier is a class in scikit-learn for k-nearest neighbors classification.
from sklearn.neighbors import KNeighborsClassifier
```

In [12]:

```
#Ignore Warnings:
import warnings
from warnings import filterwarnings
filterwarnings('ignore')

#Adjust Figure Size for Matplotlib:
plt.rcParams['figure.figsize'] = [10,4]
```

In [13]:

```
#Adjusting some display and print options for Pandas and NumPy
#max_columns to None, Pandas not to truncate the display of columns.
pd.options.display.max_columns = None

##max_rows to None, Pandas not to truncate the display of rows.
pd.options.display.max_rows = None

# To see the full numeric values without exponential notation.
np.set_printoptions(suppress=True)
```

In [14]:

```
#The os.chdir function is used to change the current working directory to the specified path.
import os
os.chdir("C:\DKS\Machine_Learning\Extra_Projects")

##Load the Dataset
data = pd.read_csv('diabetes.csv')

#The sample(15) method is used to display a random sample of 15 rows from the loaded DataFrame
data.sample(15)
```

nformation.

In [10]:

```
#This lines selects specific columns from the original DataFrame (data)
data_Correct=data.iloc[:,[0,6,7,8]]
data_missing=data.iloc[:,1:6]

#This line replaces all occurrences of 0 in the data_missing DataFrame with NaN.
#The inplace=True argument modifies the DataFrame in place.
data_missing.replace(0, np.nan, inplace=True)

#Concatenates (pd.concat) the two DataFrames (data_Correct and data_missing) along the columns (axis=1).
#The result is stored in the variable data.
data= pd.concat([data_Correct,data_missing], axis=1)
data.head()
```

```
ory usage: 54.1 KB
```

In [13]:

```
#Splitting the DataFrame into feature variables (data_x) and the target variable (data_y).
data_x = data.iloc[:, data.columns != 'type']
data_y = data.iloc[:,data.columns == 'type']
print(data_y.head(2))
print(data_x.head(2))
```

III. DEPLOYMENTS Codes

a. Program Script

```
import numpy as np
import pandas as pd
pd.set_option("display.max_column",None)
import warnings
warnings.filterwarnings("ignore")
import statsmodels.api as sm
from statsmodels.formula.api import ols
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import os
os.chdir(r'C:\DKS\Machine_Learning\Deployments4')
df= pd.read_csv('diabetes.csv')
data=pd.DataFrame(df)
data_Correct=data.iloc[:,[0,6,7,8]]
data_missing=data.iloc[:,1:6]
data_missing.replace(0, np.nan, inplace=True)
data= pd.concat([data_Correct,data_missing], axis=1)
data.drop(['insulin'],axis=1, inplace=True)
data.dropna(subset=['skin'], inplace=True)
data['glu'].fillna(data["glu"].mean() , inplace = True)
data['bp'].fillna(data["bp"].median() , inplace = True)
data['bmi'].fillna(data["bmi"].median() , inplace = True)
data2 = data.copy(deep = True)
data2['log_ped'] = np.log(data2['ped'])
data_dummy=data2
X = data_dummy.drop(['ped',"log_ped","npreg","bp","age","skin"], axis = 1)
y = pd.DataFrame(data_dummy['log_ped'])
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 1)
from sklearn import linear_model
model=linear_model.LinearRegression()
model.fit(X_train,y_train['log_ped'])
print(model.score(X_train, y_train['log_ped']))
prediction_test=model.predict(X_test)
print(y_test, prediction_test)
import pickle
pickle.dump(model,open('model.pkl','wb'))
```

b. Template Script

```
<!doctype html>
<html>
<style>
form {
    margin:auto;
    width:35%;
}
.result {
    margin:auto;
    width:35%;
    border: 1px solid #ccc
}
</style>
<head>
<h1 style="font-size:200%;color:red;">Diabetes Pedigree Function Prediction</h1>
</head>
<body>
<form action="{{ url_for('main') }}" method="POST">
    <fieldset>
        <legend>Input Values:</legend>
        Type:<span style="color:red">*</span>
       <input name="type" type="number" step="0.01" required>
       <br>
       <br>Glucose:<span style="color:red">*</span>
       <input name="glu" type="number" step="0.01" required>
       <br>
       <br>BMI:<span style="color:red">*</span>
       <input name="bmi" type="number" step="0.01" required>
       <br>
       <br>
       <input type="submit">
       </fieldset>
       </form>
<br>
<br>
<div class="result" align="center">
{% if result %}
    {% for variable,value in original_input.items()%}
         <b>{{variable}}</b> :{{value}}
         {% endfor %}
         <br>
         <br><p style="font-size:150%;color:red;">Predicted Diabetes Pedigree
Function :</p>
         <p style="font-size:50px">{{result}}</p>
         {% endif %}
</div>
</html>
```

<u>c.App Script</u>

```
# -*- coding: utf-8 -*-
"""
Created on Thu Dec 21 20:29:37 2023

@author: acer
"""


# -*- coding: utf-8 -*-
"""
Created on Sun Nov 26 20:59:08 2023

@author: acer
"""


#!/usr/bin/env python
```

```
# coding: utf-8
import flask
from flask import Flask,request,render_template
import pickle
import pandas as pd
import numpy as np
#import os
#os.chdir(r'C:\DKS\Machine_Learning\Deployments4')
#with open(f'models/model.pkl','rb') as f:
#    model=pickle.load(f)
import os
os.chdir(r'C:\DKS\Machine_Learning\Deployments4')
with open(r'models/model.pkl', 'rb') as f:
    pass ## Placeholder, replace with your actual code
    model=pickle.load(f)
app=flask.Flask(__name__, template_folder="templates")

@app.route('/', methods=['GET','POST'])

def main():
    if flask.request.method=="GET":
        return (flask.render_template("index.html"))

    if flask.request.method=="POST":
        type=flask.request.form['type']
        glu=flask.request.form['glu']
        bmi=flask.request.form['bmi']



input_variables=pd.DataFrame([[type,glu,bmi]],columns=['type','glu','bmi'],index=['Input'])
        prediction=model.predict(input_variables)[0]
    return
flask.render_template('index.html',original_input={'Type':type,'Glucose':glu,'BMI':bmi},result=n
p.exp(prediction),)


if __name__=='__main__':
        app.run()
```