

42047: Data Processing with Python

Assignment (Part C)

Report: Data Analysis and Visualization

Student Name and ID: [Syed Jazil Hussain 14369793]

Date: [28/10/2022]

Contents

Abstract	2
1. Introduction and Background	2
1.1 The problem you tried to solve	3
1.2 Business Question	3
1.3 Dataset	3
2. Overview of the Data Analysis Pipeline	4
2.1 Flow Diagram/Flowchart/Workflow	4
2.2 Data preparation	4
2.3 Missing values exploration	6
2.4 Outlier Identification	8
2.5 Data Visualization	10
3. Discussion and Conclusions.....	10
4. References	14

Abstract

To help comprehend trends and patterns in a specific data set, data analysis which is the process of organizing, gathering, and analyzing raw data is used. When making decisions or resolving business issues, the insights gathered are helpful to both individuals and corporations. Contrarily, data visualization is the act of obtaining unprocessed data and presenting it graphically through the use of diagrams, tables, and graphs.

1. Introduction and Background

This report and tasks are done to interpret raw data and visualize it in a way, where an individual may retrieve valuable information from it. Native Americans known as Pima Indians reside in Arizona, USA, and Mexico. It was determined that the incidence of diabetes mellitus was high in this group. They were therefore considered important to and representative of world health in the studies conducted around them. A well-known benchmark dataset is the Pima Indian Diabetes dataset, which includes Pima Indian females aged 21 and older. This group is also significant to other indigenous and misrepresented minorities.

1.1 The problem you tried to solve:

It's crucial to estimate a person's likelihood of being at risk for and susceptible to a chronic condition like diabetes. Early detection of chronic conditions lowers medical expenses and lowers the possibility of developing more severe health issues. It is essential that conclusions drawn from immediately measurable medical indicators can be made with accuracy, even in emergency situations where a patient may be unconscious or unable to communicate. This will aid clinicians in making better decisions regarding patient treatment in high-risk circumstances. In this dataset, we're trying to consider all of the possible causations of diabetes to female Pima Indians above the age of 21 and which of those variables have the largest effect (Chang et al., 2022).

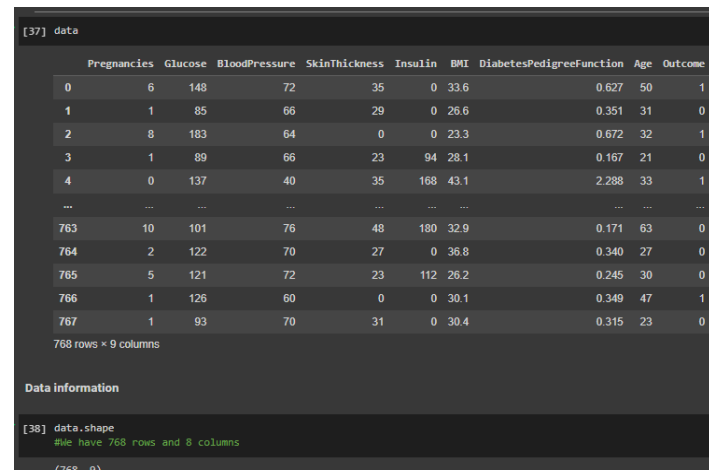
1.2 Business Question:

Using exploratory analysis of the dataset, we are trying to solve the question of which variables have a direct and large causation/effect on the probability of a Female Pima Indian over the age of 21 being diagnosed with 21.

1.3 Dataset:

<https://www.kaggle.com/code/niteshsahujhansi/diabetes-data-analysis/data>

The dataset used has been named as 'diabetes2', after uploading the csv file and loading the dataset, we can see that it contains 9 columns/attributes such as Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age and the Outcome (whether or not an individual has diabetes or not). The function data.shape is also used which tells us there are 9 columns and 768 samples of data. To solve the business question and using our common sense initially, the variables we would like to take a closer look at would be 'Pregnancies', 'Glucose', 'BMI', 'Age' as these tend to have a direct positive relationship with, and individual being diagnosed with diabetes or not.



[37] data

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

768 rows x 9 columns

Data information

[38] data.shape
#We have 768 rows and 8 columns
(768, 9)

Pregnancies: the number of times pregnant.

Glucose: The plasma glucose concentration in the oral glucose tolerance test after two hours.

BMI: Body Mass Index (weight in kg/(height in m)²

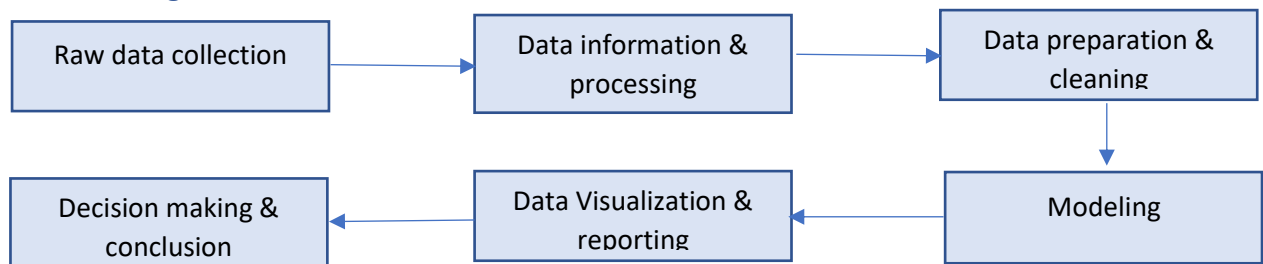
Age: Age in years.

```
data.columns
#Variable names in dataset

Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

2. Overview of the Data Analysis Pipeline

2.1 Flow Diagram/Flowchart/Workflow:



Exploratory data analysis is a method of studying data sets to highlight their key features, frequently using visual techniques. In the context of data science, EDA refers to the crucial process of conducting first investigations on data to identify patterns, notice anomalies, test hypotheses, and double-check assumptions with the aid of graphical representations and summary statistics. The first step in data analysis is the collection of raw data, which in our case has already been collected and provided to us in the form of a CSV file. The second step of this is viewing the data information provided and processing it and is carried out in a step-by-step manner (Ige, 2018). The unprocessed data is gathered, sorted, processed, examined, and stored before being provided in a legible way. The third step of the flowchart is data preparation & cleaning which involves modifying, fixing, and arranging a data set's data to make it more uniform and ready for analysis. Referring to data modelling, entities, or the items or ideas we want to track data about, are the building blocks of data models. These entities become the tables in a database. As information about an entity that we wish to keep track of, attributes are like the columns in a database. Relationships are the links between the entities in a data model. Data visualisation uses visual representations to show patterns and trends in data and give readers easy access to key insights. Lastly, these reports and visualizations are used to derive useful conclusions to solve a business problem provided alongside a dataset (Duggal, 2022).

2.2 Data preparation

As mentioned above, data preparation and cleaning are the procedure of finding, fixing, and guaranteeing that your supplied data set is error-free, consistent, and useable by locating any flaws or corruptions in the data, correcting, or deleting them, or manually processing them as necessary to prevent the error from tainting our final analysis. Our diabetes dataset was cleaned using several techniques below:

1. Data visual inspection: To get a sense of how the data is presented, we look at the first five and last five rows of the data frame.

```
[44] df.head(5)
#First 5 rows
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Diabetes
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

The function `df.head(5)` is used to view the first 5 rows or samples in the data frame.

```
[45] df.tail(5)
#Last 5 rows
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Diabetes
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

The function `df.tail(5)` is used to view the last 5 rows or samples in the data frame.

2. Carryout Summary Statistics: This provides us with a brief and straightforward summary of the diabetes dataset. It will provide data such as mean, median, mode, and variance. We can also identify outliers. It can be applied to columns with numerical data.

```
[43] df.describe()
#Describing the dataset
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Diabetes
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

The function `df.describe()` is used to get a summary statistic of each of the attributes/columns of the dataset. This includes the count of each attribute, mean, standard deviation etc.

3. Renaming variables: Renaming variables or attributes to make them easier for users to understand or read can also be practices.

```
[42] #Renaming the columns
df = data.rename(columns={'Outcome': 'Diabetes'})
```

The function `rename` has been used to rename the attribute outcome to diabetes to help users understand the data in a more efficient and easier way.

4. Converting to appropriate data types: Data types sometimes arrive in the wrong format, which makes analysis challenging. For example, a column that should be storing numeric data might be storing strings, and a column that should be storing categorical data might be storing strings. Therefore, to increase the simplicity of analysis, it is always necessary to convert to acceptable data formats.

The function `df.dtypes` has been used to view the data types for each attribute, we can determine with the use of this function that all the variable types have been in the correct format.

```
[143] df.dtypes

Pregnancies    float64
Glucose         float64
BloodPressure   float64
SkinThickness   float64
Insulin         float64
BMI            float64
DiabetesPedigreeFunction  float64
Age            float64
Diabetes        int64
BMIClass        object
AgeClass        object
dtype: object
```

2.3 Missing values exploration

Duplicate and missing data: Duplicate and missing data might have unfavourable effects on the analysis, which is why they are problematic. Regardless of the cause, it is always required to identify missing or duplicate data so that it can be discarded or filled. Using a summary statistic or binary values, such as ones and zeros, missing data can be filled in. It is always preferable to understand why the data is being duplicated or missing, regardless of the strategy we use.

```
[46] df.duplicated().sum()
#Checking number of duplicated values
#No duplicated values

0
```

The function `df.duplicated().sum()` is used to count the number of duplicate values from the dataset.

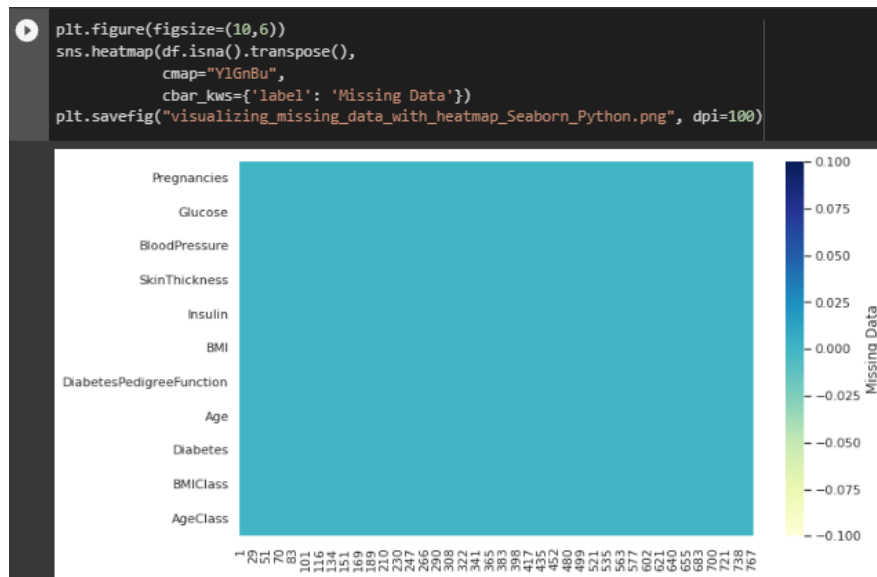
The data value that is not kept for a variable in the relevant observation is known as missing data (or missing values). Nearly all research encounters the issue of missing data, which can significantly affect the inferences that can be made from the data.

The function `df.isnull().sum()` is used to check the number of missing values for each of the attributes, as we can see from the use of this function, there are no missing values in the dataset.

```
Checking for Null Values

df.isnull().sum()
#Number of missing values for each column
#No null values in the dataset

Pregnancies    0
Glucose         0
BloodPressure   0
SkinThickness   0
Insulin         0
BMI            0
DiabetesPedigreeFunction  0
Age            0
Diabetes        0
dtype: int64
```



A heatmap can also be created to visualize the missing values in the dataset, as seen above, our diabetes dataset has no missing values as all values are equivalent to 0. We used Seaborn's `heatmap()` to create a heatmap of the data and see where each variable's missing data is located. Here, we provide Seaborn's `heatmap()` method the transposed boolean dataframe from `isna()`.



Making a stacked barplot using the percentage of missing data for each variable in the data is another method for visualising missing data. The `displot()` method from Seaborn can be used. Using the `melt()` to `displot()` tool, we provide the data here in lengthy form.

```
[49] (df[df.columns] == 0).sum()
#Checking 0 values in the dataset
```

Pregnancies	111
Glucose	5
BloodPressure	35
SkinThickness	227
Insulin	374
BMI	11
DiabetesPedigreeFunction	0
Age	0
Diabetes	500

```
dtype: int64
```

```
[149] for i in ["Glucose","BMI","Insulin","BloodPressure","SkinThickness"]:
df[i].replace({0:df[i].median()},inplace = True)
#Replacing 0 values with the median
```

```
(df[df.columns] == 0).sum()
#Checking for 0 values again
```

Pregnancies	81
Glucose	0
BloodPressure	0
SkinThickness	0
Insulin	0
BMI	0
DiabetesPedigreeFunction	0
Age	0
Diabetes	357
BMIClass	0
AgeClass	0

```
dtype: int64
```

Not having missing values may not always be a good sign for a dataset, this may be because instead of having missing/null values, they have been replaced with 0.

The function on the left is used to count the number of 0's for each of the attributes in the dataset, for important variables, the zero values have been replaced with the median of their corresponding attribute to help create a more reliable dataset for further visualization. As we can see on the left, after swapping all the zero values for the median values, only the variable 'Pregnancies' has 0 values which can be possible if an individual has not been pregnant before. And the variable 'Diabetes' has 0 values which tells us that an individual does not have diabetes. Other attributes such as 'Age', 'BMI', 'Insulin' etc. Did not make logical sense if their value was equivalent to 0.

2.4 Outlier Identification:

An observation that appears to differ significantly from other observations in the sample is known as an outlier. It's critical to identify probable outliers for the reasons listed below. An outlier might represent flawed data. For instance, an experiment might not have been properly run or the data might have been coded wrongly. An outlying value should be removed from the analysis if it can be established that the outlying point is in fact incorrect. It might not always be feasible to tell whether an outlying point contains bad data. In addition to being the result of random fluctuation, outliers can also point to important scientific phenomena. Whatever the case, we usually do not want to just delete the outlier observation.

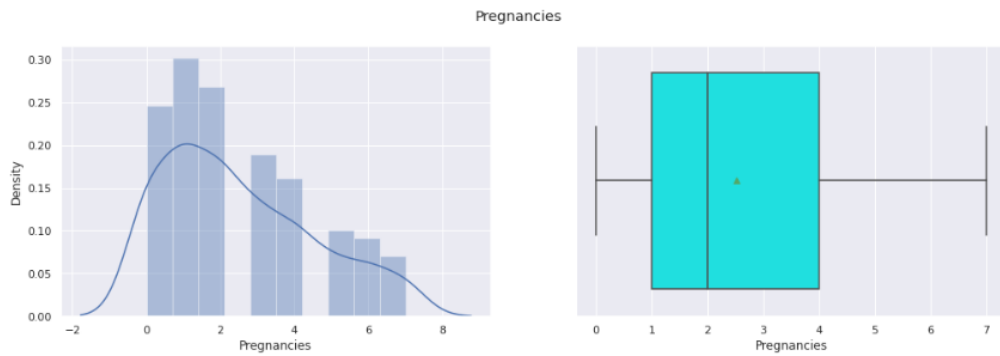
```
Outlier Observation
```

```
#Bifurcating columns into columns with character or numeric values
obj_col=df.select_dtypes(include=["object"]).columns.to_list()
num_col=df.select_dtypes(exclude=["object"]).columns.to_list()

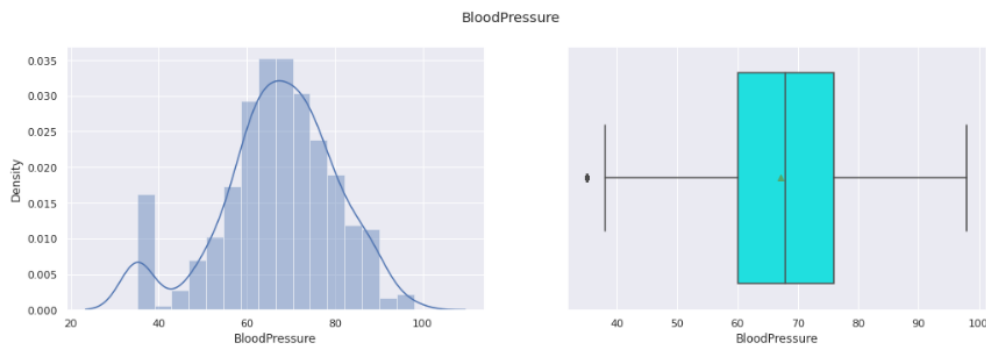
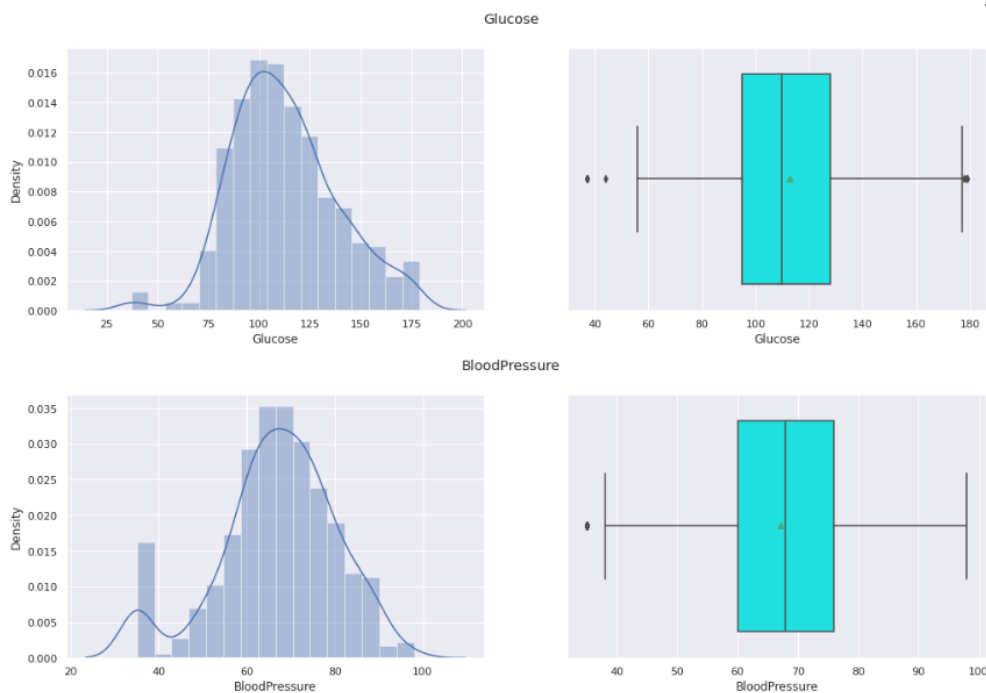
#Box plot and Distribution plot
col=num_col

for x in col:
    fig, axes=plt.subplots(1,2,figsize=(17,5))
    sns.distplot(df[x], ax=axes[0], kde=True)
    sns.boxplot(df[x], ax=axes[1], showmeans=True,color='cyan')
    fig.suptitle(x)
```

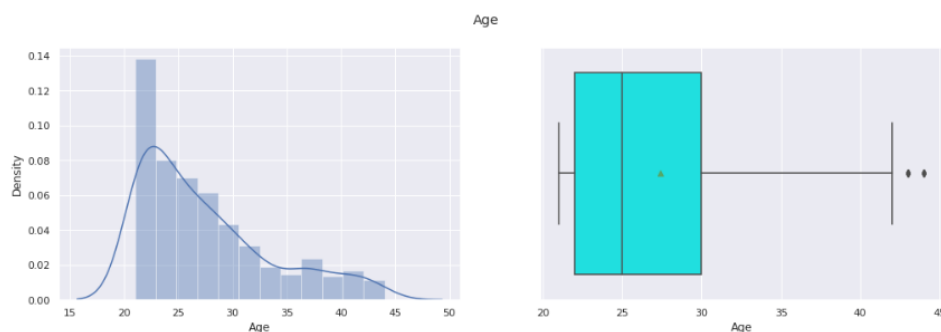
The following function is used to determine and visualize the outliers for each of the attributes present in the dataset as well as creating a distribution plot for each of the attributes based on the observations in the dataset.



The attribute 'Pregnancies' has visual outliers in the boxplot made, and the data in this attribute are skewed to the left based on the distribution plot.



Another example is the 'Glucose' and 'BloodPressure' attributes/variables which do contain visual outliers based on the box plot graphs made, <50 and <35 mm Hg respectively. Both attributes are also normally distributed.



Lastly, the attribute 'Age' has been displayed because of the outliers present, greater than the age of 43. The data present for the variable 'Age' are also skewed to the left.

```
[139] #Skewness from all the variables
print(df["Pregnancies"].skew())
print(df["Glucose"].skew())
print(df["BloodPressure"].skew())
print(df["SkinThickness"].skew())
print(df["Insulin"].skew())
print(df["BMI"].skew())
print(df["Age"].skew())
```

```
0.6156396496213958
0.3221788274295189
-0.5451701511979015
-0.22870793055250085
1.3933281074585542
0.15944615792022407
1.040155129127571
```

To deal with the outliers, we first check the skewness of each of the important variables to us. The function `print .skew` is used to calculate this. If the skewness is between -0.5 and 0.5, the data is symmetrical, but if the skewness is between -1 to -0.5 or 0.5 to 1, the data is moderately skewed. To resolve this, all the outliers should be dropped, this is done by the following function.

```
[136] #Dropping skewed values from Age
index = df[(df["Age"] >=45)].index
df.drop(index, inplace=True)
df["Age"].describe()
print(df["Age"].skew())
```

```
1.040155129127571
```

```
[137] #Dropping skewed values from BloodPressure
index = df[(df["BloodPressure"] >=100)].index
df.drop(index, inplace=True)
df["BloodPressure"].describe()
print(df["BloodPressure"].skew())
```

```
-0.5451701511979015
```

```
[138] print(df["BMI"].skew())
#Skewness of BMI after dropping outliers for other variables
```

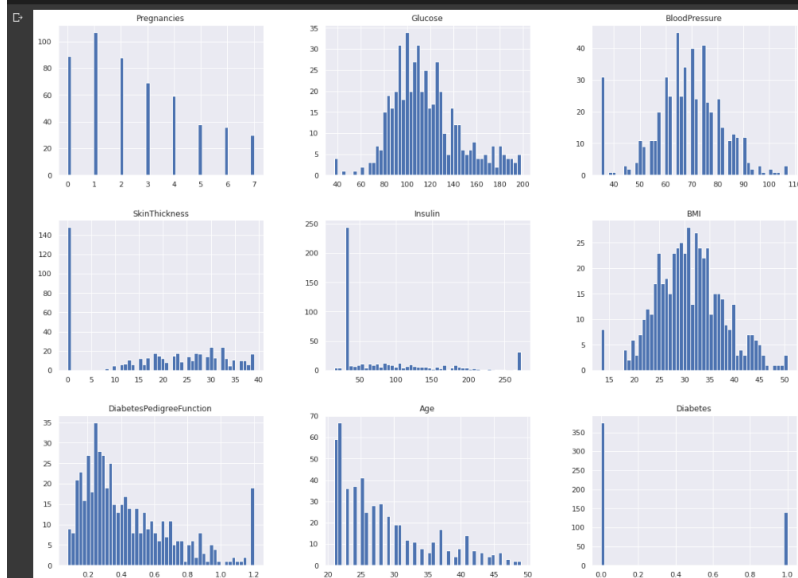
```
0.15944615792022407
```

Since the outliers for age are greater than 45, all observations above the age of 45 are dropped. The same is done for the diastolic blood pressure of less than 100 mm Hg. Similarly, this is done with all variables containing outliers. The result of this can be seen by calculating the skewness of BMI (had no outliers) which has been improved and as a result, it is now normally distributed. The dataset is not a very large one, so it would be better to avoid removing rows unnecessarily.

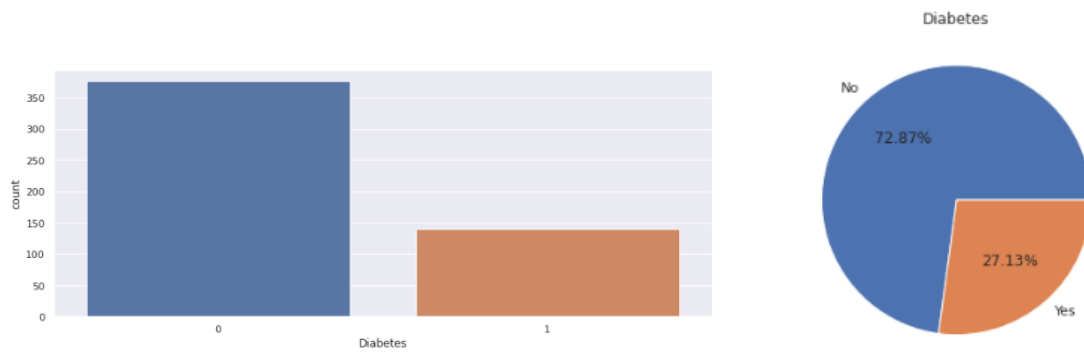
2.5 Data Visualization:

To make data easier for the user to grasp and draw conclusions from, data visualisation is the practise of putting information into a visual context, like a map or graph. Data visualization's major objective is to make it simpler to spot patterns, trends, and outliers in big data sets. One of the processes in the data analysis pipeline, data visualisation states that after data has been gathered, processed, and modelled, it must be represented to draw conclusions. A component of the larger field of data presentation architecture (DPA), which tries to search, locate, manipulate, format, and transmit data as effectively as possible, is data visualisation.

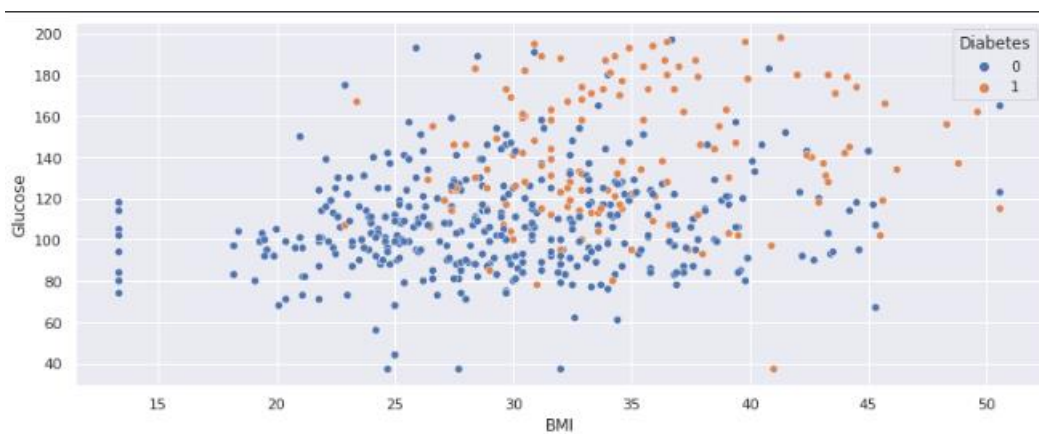
```
#Data distribution of all variables
df.hist(bins=50, figsize=(20,15));
```



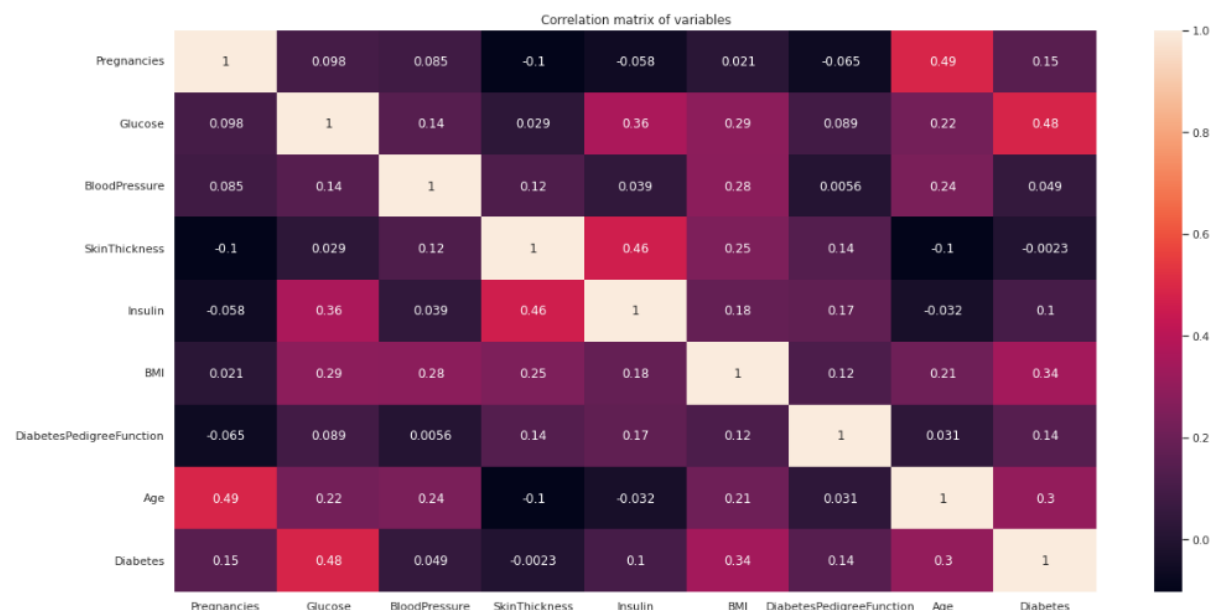
The function `df.hist` is used to visualize the distribution of the data throughout the dataset for each of the different attributes. The histograms give us an idea as to which variables have a normal distribution (Glucose, BloodPressure, BMI), and which ones are skewed.



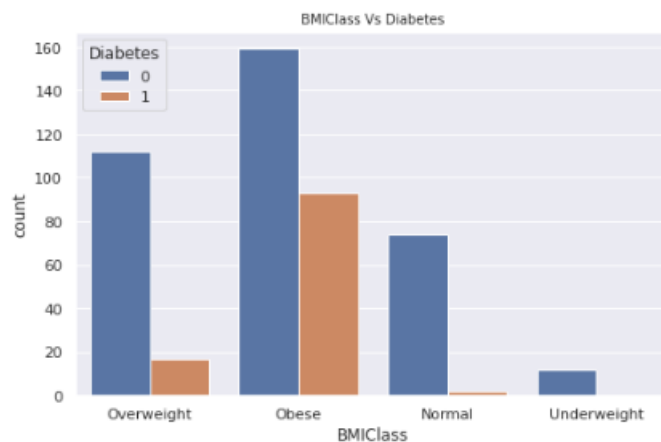
A bar chart and pie chart have been created to visualize the number of individuals in the dataset that have and have not been diagnosed with diabetes. The bar chart shows us the count of the number of Prime Indians have that been diagnosed with diabetes and that have not. Around 370 have not been diagnosed with diabetes while nearly 150 have been diagnosed. The pie chart helps visualize this in a percentage, 27% of all the sample have been diagnosed with diabetes while 73% have not been diagnosed.



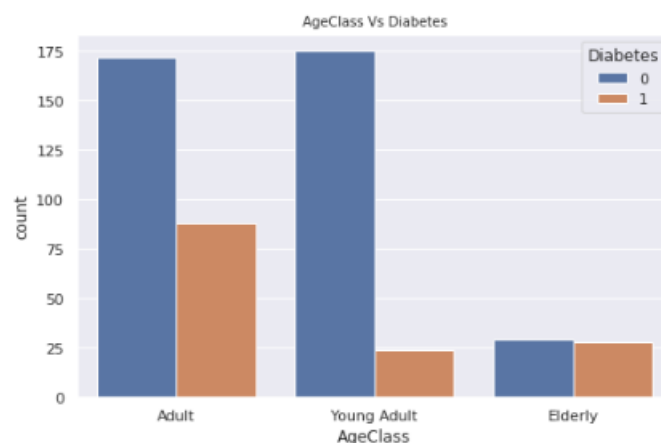
The scatterplot above shows that patients with glucose levels below 120 and BMI values a little about 20(healthy/normal weight) rarely have diabetes. That is a patient high BMI value and high Glucose level is most likely to be diagnosed with diabetes.



By Observing Outcome Ratios of Correlation: Features Prevalent - BMI, Age, Pregnancies (External Factors) Glucose, Insulin (Internal Factors) -- Descending Order Features Not Critical - Skin Thickness, Blood Pressure



A new variable called 'BMIClass' has been also created to further divide the numerical variable 'BMI' and create a categorical variable. This helps to give a better understanding of the relationship between the two variables BMI and Diabetes. As we can see from the bar chart, Individuals that are Obese with a BMI > 29.9 are most likely to have diabetes. More than 33% of Obese individuals are diagnosed with diabetes, while only around 15% of Overweight individuals are diagnosed with diabetes. Less than 2% of individuals with a normal BMI are diagnosed with diabetes, and lastly, 0% of individuals that are underweight have been diagnosed with diabetes.



The last visualization plot and variable created was 'AgeClass' which has been created to make age a categorical variable, making it easier to visualize and understand. This helps give a better understand of the relationship between Age and Diabetes. As we can see from the bar chart, individuals > than the age of 40 who are considered elderly, are most likely to have diabetes with nearly 50% of all the elderly individuals being diagnosed

with diabetes. Nearly 30% of all adults have been diagnosed with diabetes, and lastly, only 12% of all young adults have been diagnosed with diabetes.

3. Discussion and Conclusions:

To conclude, the data preparation and cleaning steps have been done to help better prepare our data for visualization use. It is important that all these steps are used for our analysis, and it is important to holistically know which of these techniques to apply and when.

From the above visualization techniques, we can assume that a high glucose level indicates that an individual will most likely be diagnosed with diabetes. People with a high number of pregnancies are also more prone to being diagnosed with diabetes. Underweight people (very low BMI) are also prone to getting diabetes if their skin thickness is relatively high. Moreover, overweight people (high BMI) who are of an elderly are at the highest risk of getting diabetes.

Furthermore, Prima Indians should keep their BMI in check and low to avoid high levels of Glucose and Insulin. Keep their Glucose and Insulin levels in check as age progresses. Pregnant ladies also should keep a close watch on their glucose and insulin levels especially if they have been pregnant

more than one times. High glucose level in pregnancies increase the risk of diabetes, a BMI greater than 30 and high levels of glucose together increase the risk of diabetes. We can see from this report that an increasing glucose level is the key factor which increase risk of diabetes, and alongside an increase in other variables increases the probability of being diagnosed with diabetes the most. This report helps us answer our business problem of which variables have the greatest effect on the prediction of a Prima Indian being diagnose with diabetes.

4. References:

- Duggal, N. (2022) *What is Data Processing System? definition, cycle, types & methods [updated]*, *Simplilearn.com*. Simplilearn. Available at: <https://www.simplilearn.com/what-is-data-processing-article#:~:text=It%20is%20usually%20performed%20in,presented%20in%20a%20readable%20format>. (Accessed: October 29, 2022).
- Ige, O. (2018) *Pipeline for exploratory data analysis and data cleaning.*, *Medium*. Medium. Available at: <https://medium.com/@oluwabukunmige/pipeline-for-exploratory-data-analysis-and-data-cleaning-6adce7ac0594> (Accessed: October 29, 2022).
- Chang, V. et al. (2022) *Pima Indians diabetes mellitus classification based on machine learning (ML) algorithms*, *Neural computing & applications*. Springer London. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8943493/#:~:text=Pima%20Indians%20are%20a%20Native,of%20global%20health%20%5B4%5D>. (Accessed: October 29, 2022).
- Data Cleaning Steps & process to prep your data for Success* (2021) *MonkeyLearn Blog*. Available at: <https://monkeylearn.com/blog/data-cleaning-steps/> (Accessed: October 29, 2022).
- (no date) *1.3.5.17. detection of outliers*. Available at: <https://www.itl.nist.gov/div898/handbook/eda/section3/eda35h.htm#:~:text=An%20outlier%20is%20an%20observation,not%20have%20been%20run%20correctly>. (Accessed: October 29, 2022).