PIMA_Daibetes_Data_Analysis

October 12, 2023

1 Diabetes Patients Data Analysis Using Python

1.0.1 (1) Importing Libraries

[155]:
```
# import the required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

1.0.2 (2) Introduction

Diabetes is a health condition that affects how your body turns food into energy. Most of the food you eat is broken down into sugar (also called glucose) and released into your bloodstream. When your blood sugar goes up, it signals your pancreas to release insulin.

Without ongoing, careful management, diabetes can lead to a buildup of sugars in the blood, which can increase the risk of dangerous complications, including stroke and heart disease.

Objectives (1) Predict if person is diabetes patient or not (2) Find most indicative features of diabetes

1.0.3 (3) Importing DataFrame

[156]:
```
# Load the dataset
df = pd.read_csv(r"E:\MeriSkill\Project 2 - Diabetes Data\Project 2␣
 ↪MeriSKILL\diabetes.csv")
df
```

[156]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	\
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
..	…	…	…	…	…	…	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	

```
766            1      126             60              0        0  30.1
767            1       93             70             31        0  30.4

     DiabetesPedigreeFunction  Age  Outcome
0                       0.627   50        1
1                       0.351   31        0
2                       0.672   32        1
3                       0.167   21        0
4                       2.288   33        1
..                        …  …      …
763                     0.171   63        0
764                     0.340   27        0
765                     0.245   30        0
766                     0.349   47        1
767                     0.315   23        0

[768 rows x 9 columns]
```

1.0.4 (4) Data Dictionary

(1)Pregnancies: Number of times pregnant

(2) Glucose: The plasma glucose concentration in the oral glucose tolerance test after two hours

(3) BloodPressure: Diastolic blood pressure (mm Hg)

(4) SkinThickness: Triceps skin fold thickness (mm)

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

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

(7) DiabetesPedigreeFunction: This function calculates the likelihood of having diabetes based on the lineage of a descendant

(8) Age: Age (years)

(9) Outcome: Class variable (have the disease (1) or not (0))

[157]:
```
df.shape # it gives the rows and columns
```

[157]:
```
(768, 9)
```

[158]:
```
df.head() # Top 5 records of the DataFrame
```

[158]:
```
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0            6      148             72             35        0  33.6
1            1       85             66             29        0  26.6
2            8      183             64              0        0  23.3
3            1       89             66             23       94  28.1
4            0      137             40             35      168  43.1
```

```
   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1
```

[159]:
```
df.tail() # it gives last 5 records of the DataFrame
```

[159]:
```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
763           10      101             76             48      180  32.9
764            2      122             70             27        0  36.8
765            5      121             72             23      112  26.2
766            1      126             60              0        0  30.1
767            1       93             70             31        0  30.4

     DiabetesPedigreeFunction  Age  Outcome
763                     0.171   63        0
764                     0.340   27        0
765                     0.245   30        0
766                     0.349   47        1
767                     0.315   23        0
```

[160]:
```
df.info() # it gives the basic info about the DataFrame
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

[161]:
```
df.describe() # it gives the basic statistics of the DataFrame
```

[161]:
```
       Pregnancies     Glucose  BloodPressure  SkinThickness     Insulin  \
count   768.000000  768.000000     768.000000     768.000000  768.000000
mean      3.845052  120.894531      69.105469      20.536458   79.799479
std       3.369578   31.972618      19.355807      15.952218  115.244002
```

```
min       0.000000    0.000000       0.000000       0.000000     0.000000
25%       1.000000   99.000000      62.000000       0.000000     0.000000
50%       3.000000  117.000000      72.000000      23.000000    30.500000
75%       6.000000  140.250000      80.000000      32.000000   127.250000
max      17.000000  199.000000     122.000000      99.000000   846.000000
```

```
              BMI  DiabetesPedigreeFunction         Age     Outcome
count  768.000000                768.000000  768.000000  768.000000
mean    31.992578                  0.471876   33.240885    0.348958
std      7.884160                  0.331329   11.760232    0.476951
min      0.000000                  0.078000   21.000000    0.000000
25%     27.300000                  0.243750   24.000000    0.000000
50%     32.000000                  0.372500   29.000000    0.000000
75%     36.600000                  0.626250   41.000000    1.000000
max     67.100000                  2.420000   81.000000    1.000000
```

1.0.5 (5) Data Pre-Processing

[162]:
```
# Check for null values
df.isnull().sum()
```

[162]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

I observed that there is no missing values in dataset however the features like Glucose, BloodPressure, Insulin, SkinThickness has 0 values which is not possible. We have to replace 0 values with either mean or median values of specific column.

[163]:
```
df['Glucose'] = df['Glucose'].replace(0, df['Glucose'].mean())

df['BloodPressure'] = df['BloodPressure'].replace(0, df['BloodPressure'].
 ↪mean()) # There are 35 records with 0 BloodPressure in dataset

df['BMI'] = df['BMI'].replace(0, df['BMI'].median())

df['SkinThickness'] = df['SkinThickness'].replace(0, df['SkinThickness'].
 ↪median())

df['Insulin'] = df['Insulin'].replace(0, df['Insulin'].median())
```

[164]:
```
df.head()
```

[164]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148.0	72.0	35	30.5	33.6	0.627	50	1
1	1	85.0	66.0	29	30.5	26.6	0.351	31	0
2	8	183.0	64.0	23	30.5	23.3	0.672	32	1
3	1	89.0	66.0	23	94.0	28.1	0.167	21	0
4	0	137.0	40.0	35	168.0	43.1	2.288	33	1

[165]:
```
# Check for Duplicated Values
df.duplicated().sum()
```

[165]:
```
0
```

- There is "No Duplicated" values in the Diabetes DataFrame

[166]:
```
# Check for Zero Variance and Near Zero Variance Features
df.var()==0
```

[166]:
```
Pregnancies                 False
Glucose                     False
BloodPressure               False
SkinThickness               False
Insulin                     False
BMI                         False
DiabetesPedigreeFunction    False
Age                         False
Outcome                     False
dtype: bool
```

- There is No Zero Variance and Near Zero Varince Features

1.0.6 (6) Exploratory Data Analysis

- In this EDA (1) Uni Variate Analysis (2) Bi Variate Analysis (3) Multi Variate Analysis

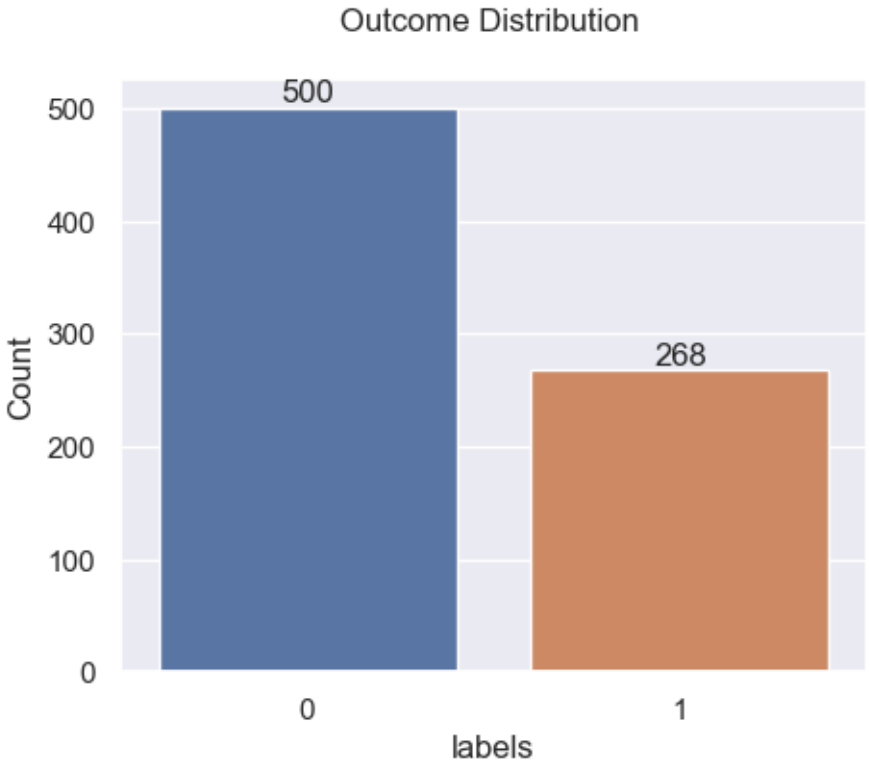
(6.1) Univariate Analysis

- In this Uni variate analysis we can consider single variable only

[167]:
```
df.columns
```

[167]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

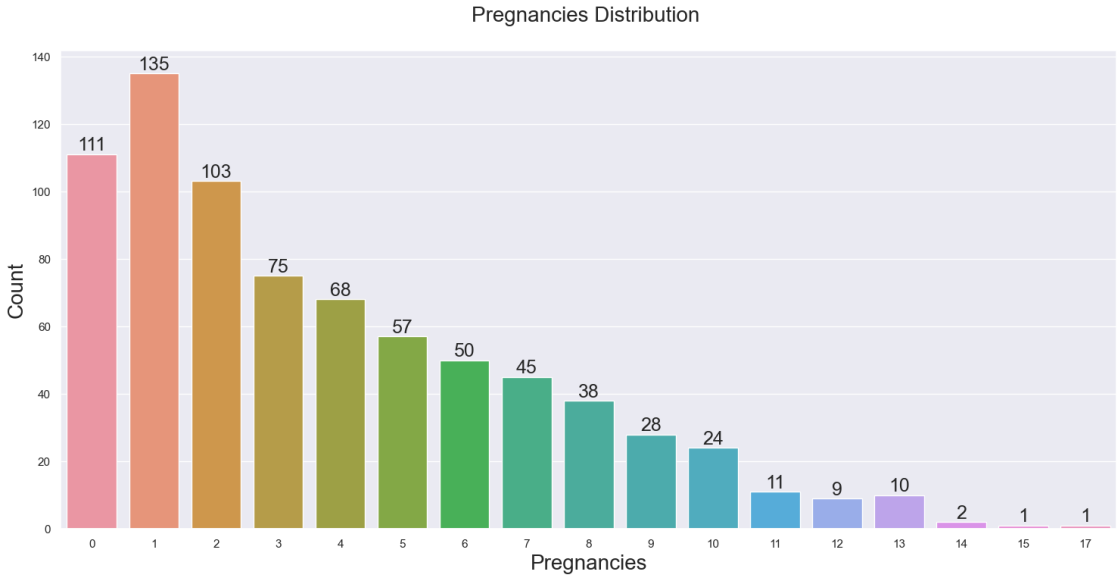
[168]:
```
# Check the Distribution of Outcome Feature
plt.figure(figsize= (5, 4))
ax = sns.barplot(x=df['Outcome'].value_counts().index,   y=df['Outcome'].
 ↪value_counts())
for bars in ax.containers:
    ax.bar_label(bars)
plt.xlabel('labels', size = 12)
plt.ylabel('Count', size = 12)
plt.title('Outcome Distribution \n', size = 12)
plt.show()
```



- In this above Graph 0 means don't have diabetes and 1 means have the diabets
- we can observe Above Graph , there is more people don't have diabetes
- 500 people don't have diabetes
- 268 people have the diabetes

[169]:
```
# Pregnancies Distribution
```

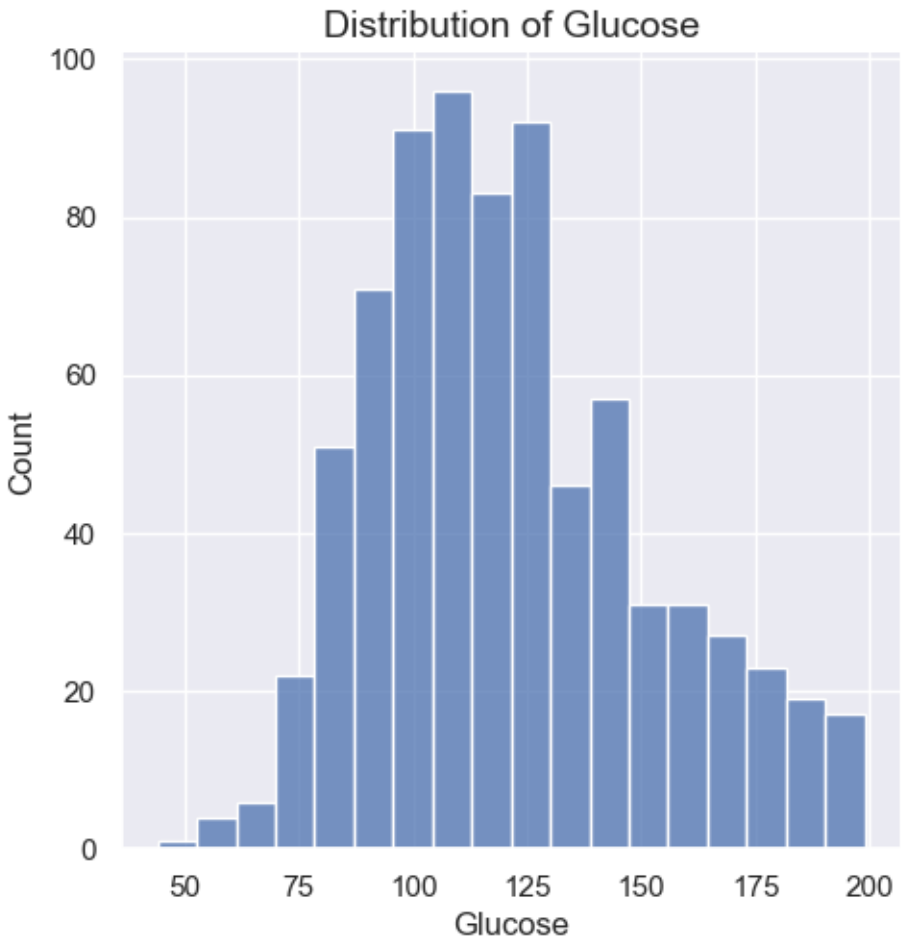
```
ax = sns.barplot(x=df['Pregnancies'].value_counts().index,␣
 ↪y=df['Pregnancies'].value_counts())
for bars in ax.containers:
    ax.bar_label(bars,size = 18)
plt.xlabel('Pregnancies', size = 20)
plt.ylabel('Count', size = 20)
plt.title('Pregnancies Distribution \n', size = 20)
plt.show()
```



Insights:

- The Above Graph Represents people got No.of times pregnancies
- (1) the pregnancies range from 0 to 17
- (2) People who get the pregnancie 1 time are Hige when compare to the Reaming
- (3) 111 people didn't get pregnancie
- (4) Only one people got Pregnancie at 15 times and 17 times

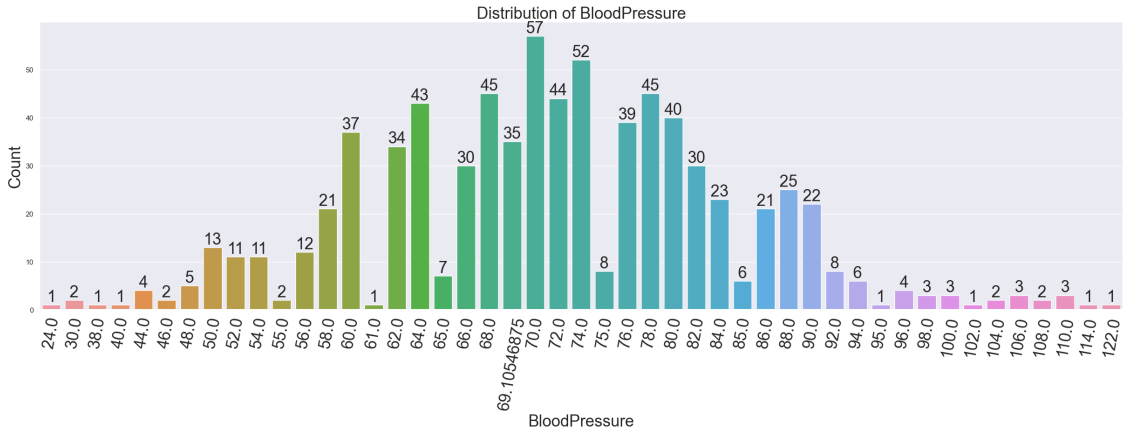
[170]:
```
# Distribution of Glucose
sns.displot(df, x="Glucose")
plt.title("Distribution of Glucose", size  = 14)
plt.show()
```



Insights:

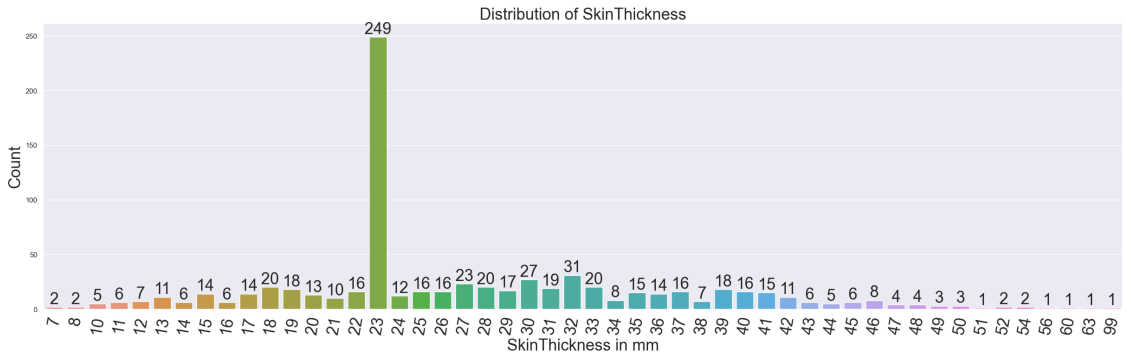
- In this Above Distribution plot I Observed
- (1) Glucose Range From 50 to 200
- (2) Most of the people had Glucose level from 80 to 120

[171]:
```
plt.figure(figsize= (30, 8))

ax = sns.countplot(data = df, x=df.BloodPressure)
for bars in ax.containers:
    ax.bar_label(bars,size = 25)
plt.title('Distribution of BloodPressure', size= 25)
plt.xlabel("BloodPressure",size = 25)
plt.ylabel("Count",size = 25)
plt.xticks(rotation = 80,size= 25)
plt.show()
```



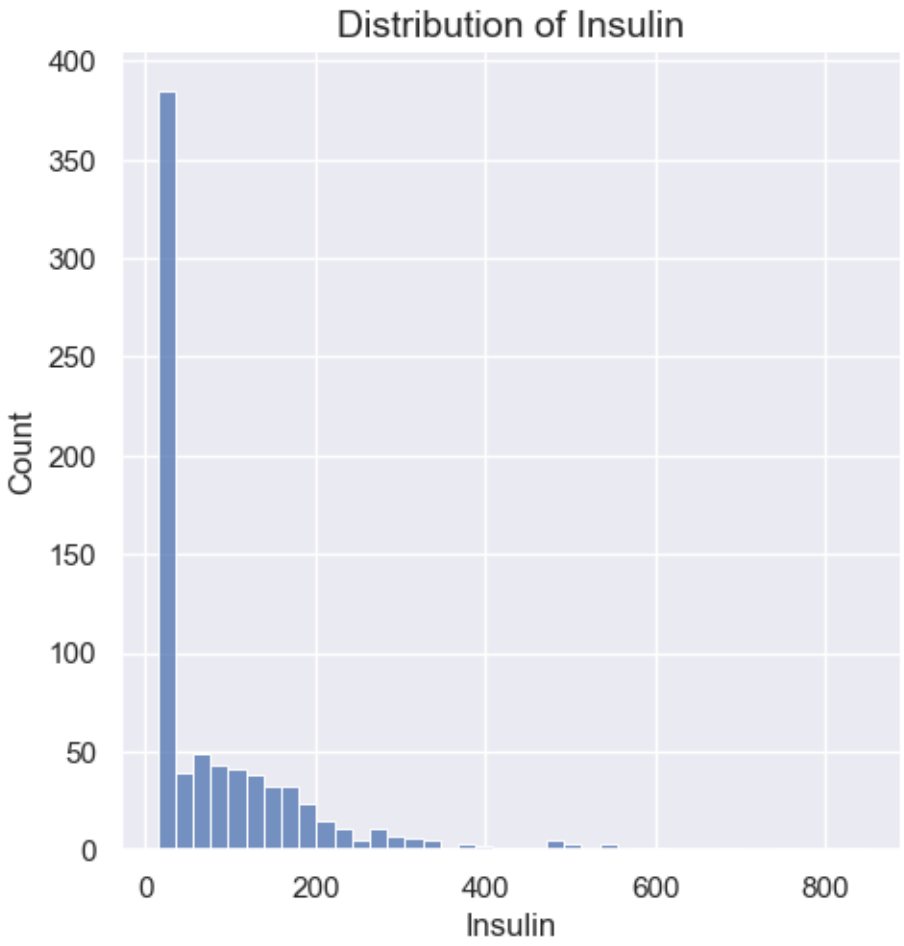
- In this we can observe BloodPressure Range from 24 to 122
- 57 peoples have the BloodPressure 57
- Most of the people have BloodPressure Range from 60-80
- Less no of the people have BloodPressure Range from 24-60 and 90-122

[172]:
```
plt.figure(figsize= (30, 8))

ax = sns.countplot(data = df, x=df.SkinThickness)
for bars in ax.containers:
    ax.bar_label(bars,size = 25)
plt.title('Distribution of SkinThickness', size= 25)
plt.xlabel("SkinThickness in mm",size = 25)
plt.ylabel("Count",size = 25)
plt.xticks(rotation = 80,size= 25)
plt.show()
```



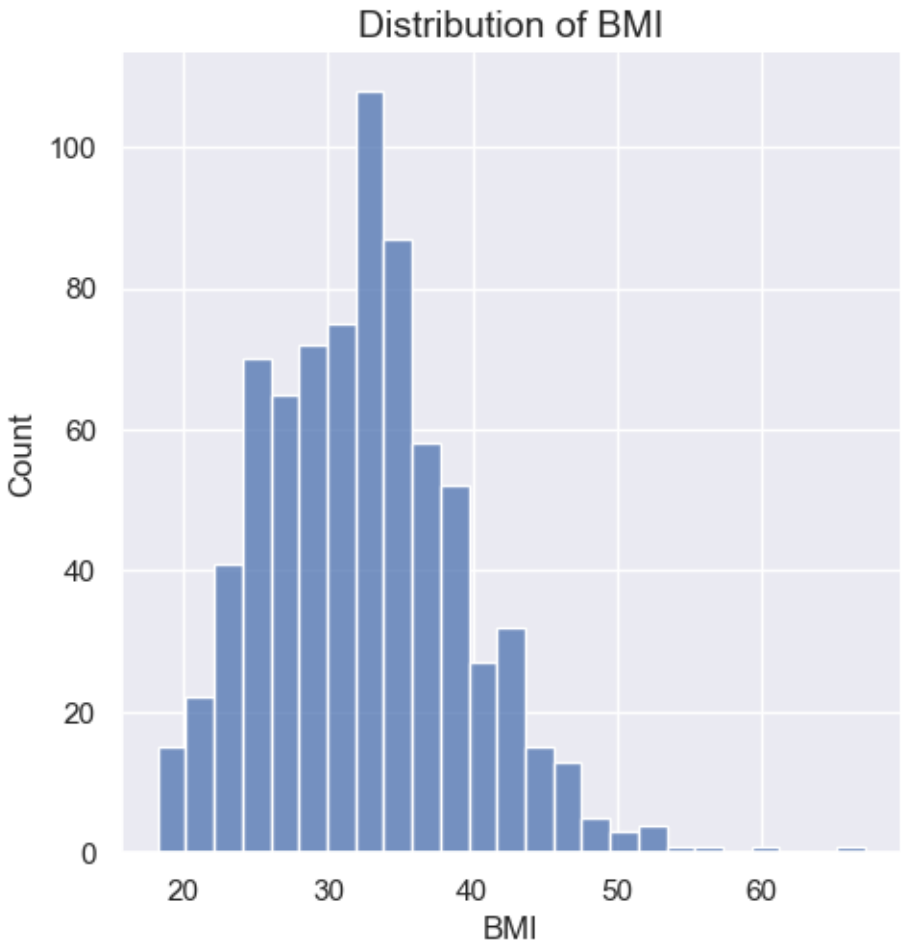
- In this “SkinThickness in mm” Graph representation,Skinthickness Range From 7 to 99
- More people had skinthickness is 23mm
- I Observed 249 people had 23mm Skinthickness

[173]:
```
# Distribution of Insulin
sns.displot(df, x="Insulin")
plt.title("Distribution of Insulin", size  = 14)
plt.show()
```



- In this I observed Insulin Range from 0 to 800
- most of the people had the insulin range between o to 200

[174]:
```
sns.displot(df, x="BMI")
plt.title("Distribution of BMI", size  = 14)
plt.show()
```

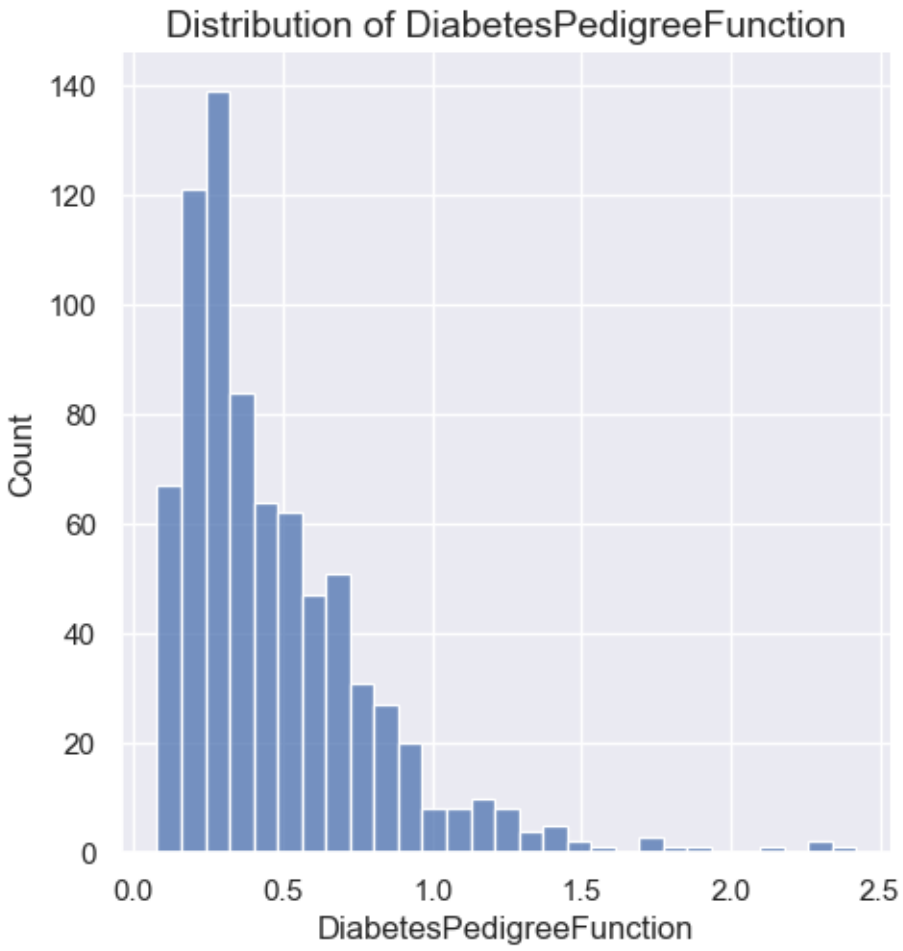


- According to the above Graph Representation BMI Range Between 20 to 70
- More people had the BMI Range from 25 to 40

[175]:
```
df.columns
```

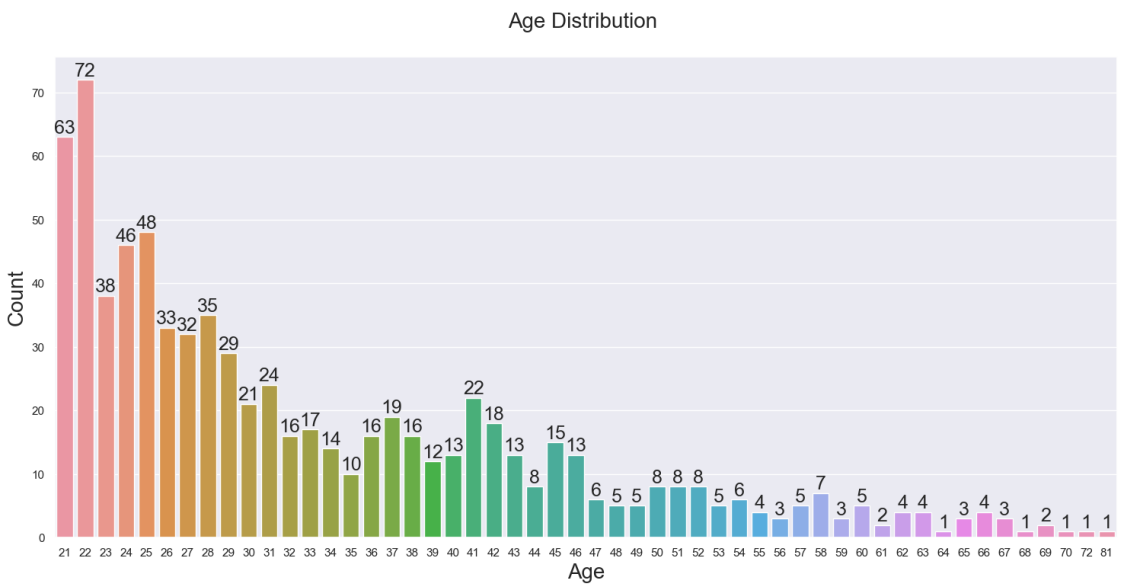
[175]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

[176]:
```
# Distribution of DiabetesPedigreeFunction
sns.displot(df, x="DiabetesPedigreeFunction")
plt.title("Distribution of DiabetesPedigreeFunction", size  = 14)
plt.show()
```



- According to the above Graph Representation I Observed
- (1) DiabetesPedigreeFunction Range Between from 0 to 2.5
- (2) And More people had DiabetesPedigreeFunction value from o to 1 only
- (3)0.3,0.4 had the more people

[177]:
```
# Distribution of Age
ax = sns.barplot(x=df['Age'].value_counts().index,   y=df['Age'].value_counts())
for bars in ax.containers:
    ax.bar_label(bars,size = 18)
plt.xlabel('Age', size = 20)
plt.ylabel('Count', size = 20)
plt.title('Age Distribution \n', size = 20)
plt.show()
```

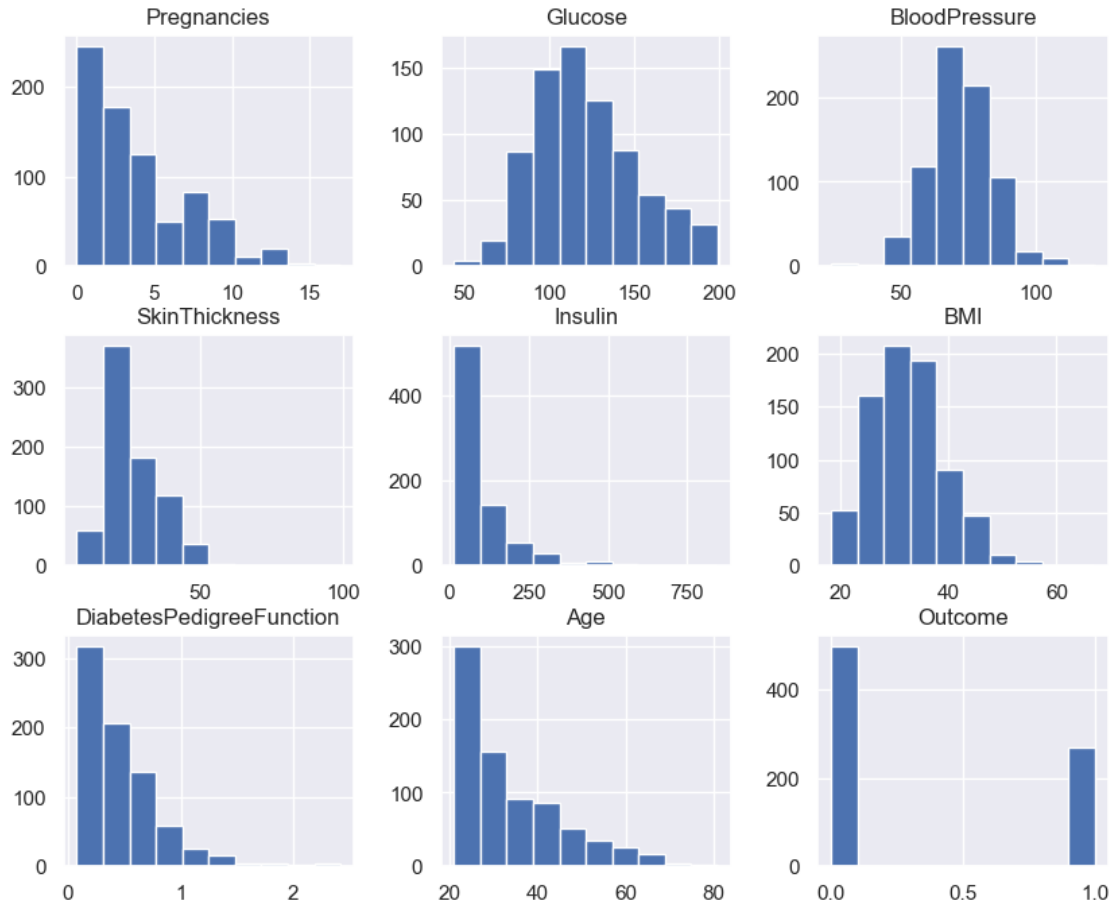


Insights:

- we can observe above Graph, the Age Range from 21 to 81
- In this given dataframe,the people who had age 22 are higher when compare to the Remaing age people
- 72 people had age 22 and followed 63 people had age 21
- Less no people had age from 47 to 81

[178]:
```
# Histogram for the entire DataFrame
df.hist(figsize=(10,8))
```

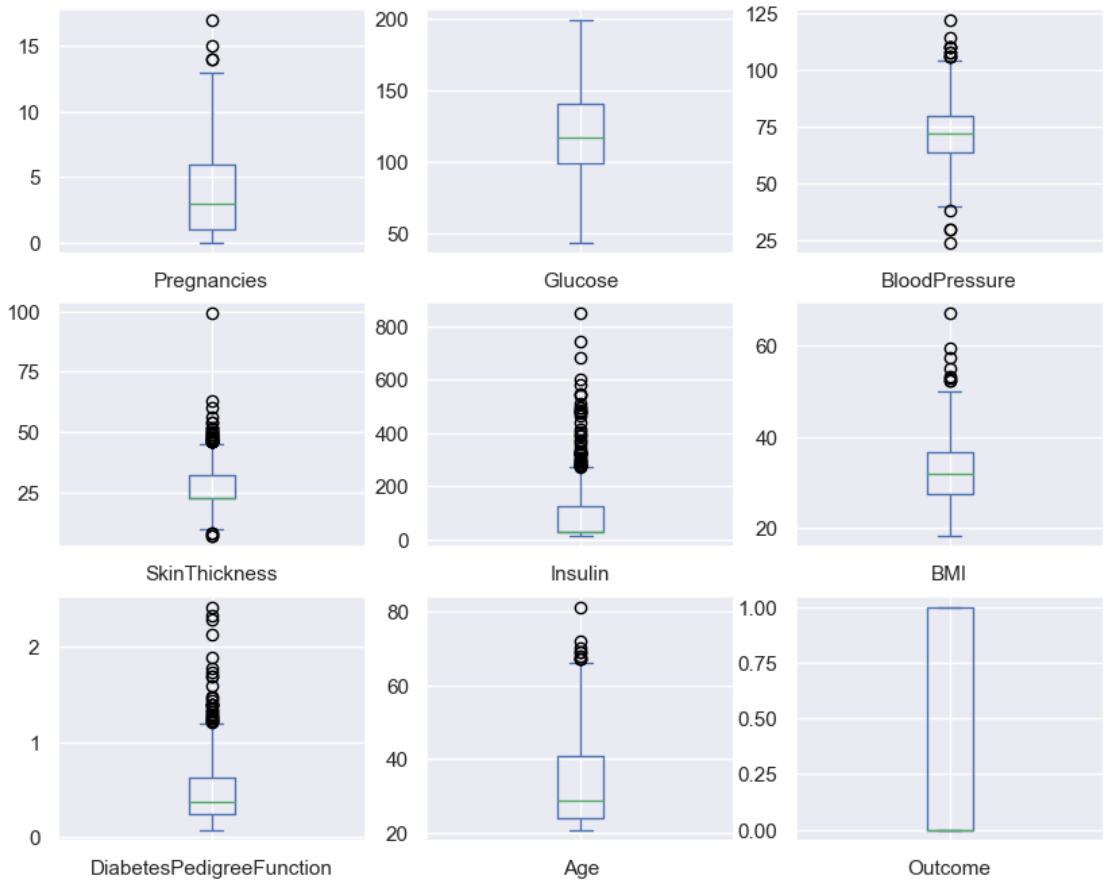
[178]:
```
array([[<Axes: title={'center': 'Pregnancies'}>,
        <Axes: title={'center': 'Glucose'}>,
        <Axes: title={'center': 'BloodPressure'}>],
       [<Axes: title={'center': 'SkinThickness'}>,
        <Axes: title={'center': 'Insulin'}>,
        <Axes: title={'center': 'BMI'}>],
       [<Axes: title={'center': 'DiabetesPedigreeFunction'}>,
        <Axes: title={'center': 'Age'}>,
        <Axes: title={'center': 'Outcome'}>]], dtype=object)
```



Checking Outliers in DataFrame

[179]:
```
df.plot(kind = 'box',subplots = True, layout = (3,3),sharex=False,␣
↪sharey=False,figsize=(10,8))
```

[179]:
```
Pregnancies                 Axes(0.125,0.653529;0.227941x0.226471)
Glucose                  Axes(0.398529,0.653529;0.227941x0.226471)
BloodPressure            Axes(0.672059,0.653529;0.227941x0.226471)
SkinThickness               Axes(0.125,0.381765;0.227941x0.226471)
Insulin                  Axes(0.398529,0.381765;0.227941x0.226471)
BMI                      Axes(0.672059,0.381765;0.227941x0.226471)
DiabetesPedigreeFunction            Axes(0.125,0.11;0.227941x0.226471)
Age                         Axes(0.398529,0.11;0.227941x0.226471)
Outcome                     Axes(0.672059,0.11;0.227941x0.226471)
dtype: object
```



- I observed Above Boxplot Representation
- Most of the features have the outliers so can use one of the outliers Techiques Winosorization to deal with Outliers

[180]:
```
from scipy.stats.mstats import winsorize

df['Pregnancies']=winsorize(df.Pregnancies,limits=[0.07, 0.093])
df['BloodPressure']=winsorize(df.BloodPressure,limits=[0.06, 0.094])
df['SkinThickness']=winsorize(df.SkinThickness,limits=[0.07, 0.093])
df['Insulin']=winsorize(df.Insulin,limits=[0.06, 0.094])
df['BMI']=winsorize(df.BMI,limits=[0.07, 0.093])
df['DiabetesPedigreeFunction']=winsorize(df.DiabetesPedigreeFunction,limits=[0.
↪06, 0.094])
df['Age']=winsorize(df.Age,limits=[0.07, 0.093])
```

[181]:
```
df.columns
```

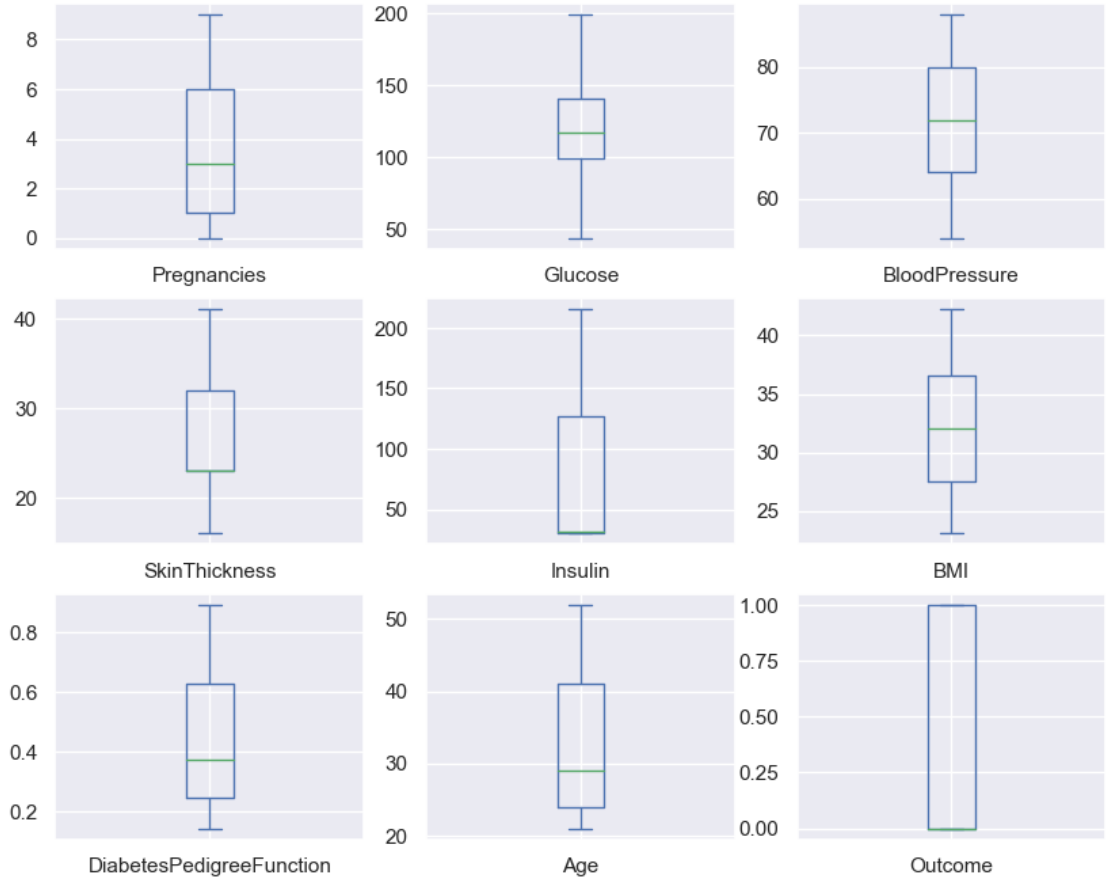
[181]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
```

```
      dtype='object')
```

Once again check for Outliers after winsorization

[182]:
```
df.plot(kind = 'box',subplots = True, layout = (3,3),sharex=False,sharey=False,figsize=(10,8))
```

[182]:
```
Pregnancies                 Axes(0.125,0.653529;0.227941x0.226471)
Glucose                  Axes(0.398529,0.653529;0.227941x0.226471)
BloodPressure            Axes(0.672059,0.653529;0.227941x0.226471)
SkinThickness               Axes(0.125,0.381765;0.227941x0.226471)
Insulin                  Axes(0.398529,0.381765;0.227941x0.226471)
BMI                      Axes(0.672059,0.381765;0.227941x0.226471)
DiabetesPedigreeFunction        Axes(0.125,0.11;0.227941x0.226471)
Age                      Axes(0.398529,0.11;0.227941x0.226471)
Outcome                  Axes(0.672059,0.11;0.227941x0.226471)
dtype: object
```



- Now There is no outliers in the dataFrame

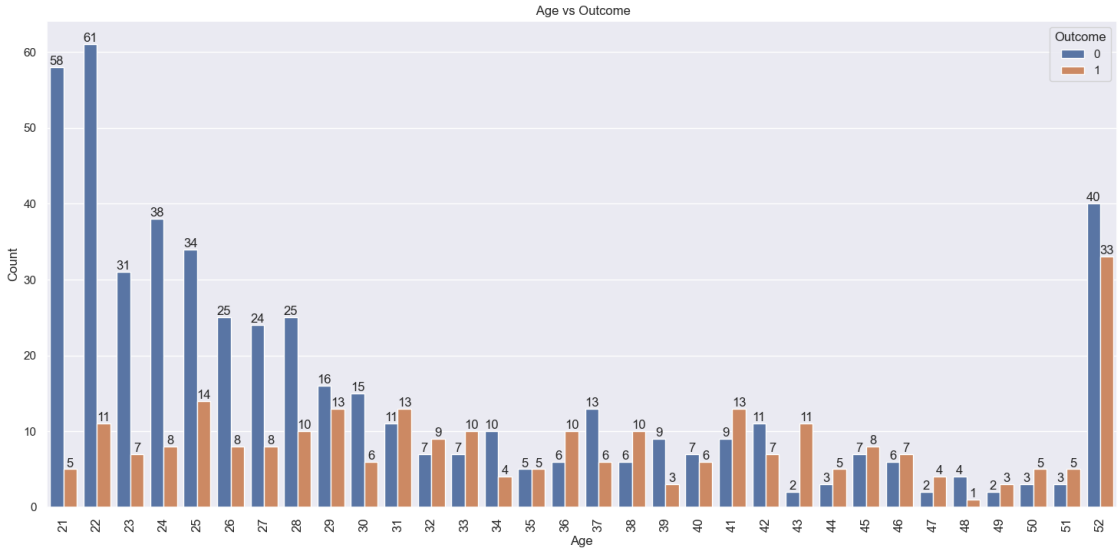
[183]:
```
df.columns
```

[183]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

(6.1) Bi Variate Analysis

Age Vs Outcome

[184]:
```
ax = sns.countplot( x = df['Age'],hue = df['Outcome'],data = df)
for bars in ax.containers:
    ax.bar_label(bars)
plt.xlabel("Age", size = 12)
plt.xticks(rotation = 90, size = 12)
plt.ylabel("Count", size = 12)
plt.title(" Age vs Outcome",size = 12)
plt.show()
```



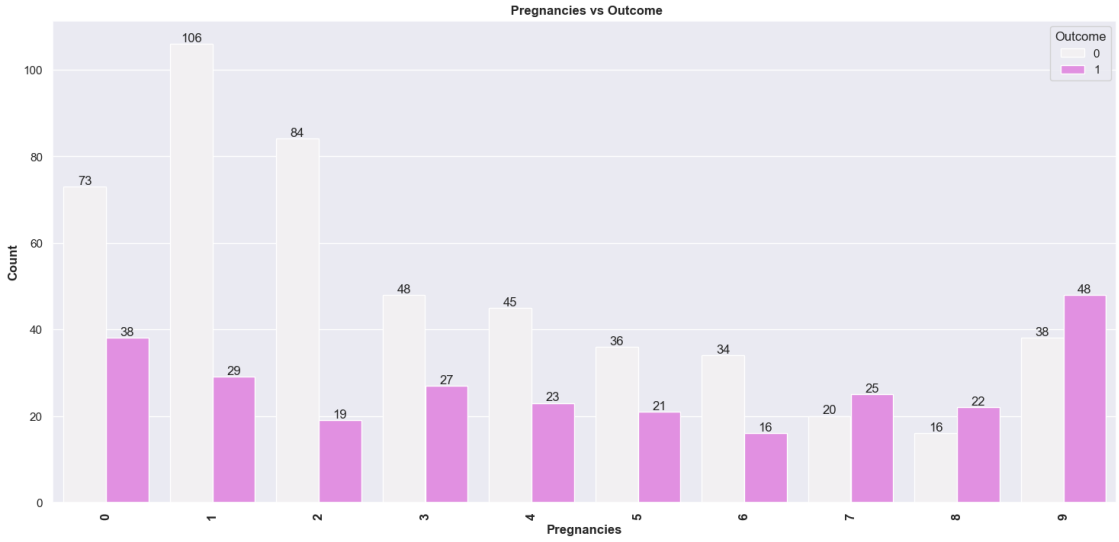
Insights:

- The Above Graph Represents Relationship between Age and Outcome Variables
- Most of the people have the diabetes who had age is 52 Years.
- Only One person have the diabetes who had age is 48 years

Pregnancies Vs Outcome

[185]:
```
ax = sns.countplot( x = df['Pregnancies'],hue = df['Outcome'],data = df,color =␣
↪'violet')
for bars in ax.containers:
```

```
    ax.bar_label(bars)
plt.xlabel("Pregnancies", size = 12, fontweight='bold')
plt.xticks(rotation = 90, size = 12, fontweight='bold')
plt.ylabel("Count", size = 12,fontweight='bold')
plt.title(" Pregnancies vs Outcome",size = 12, fontweight='bold')
plt.show()
```



Insights :

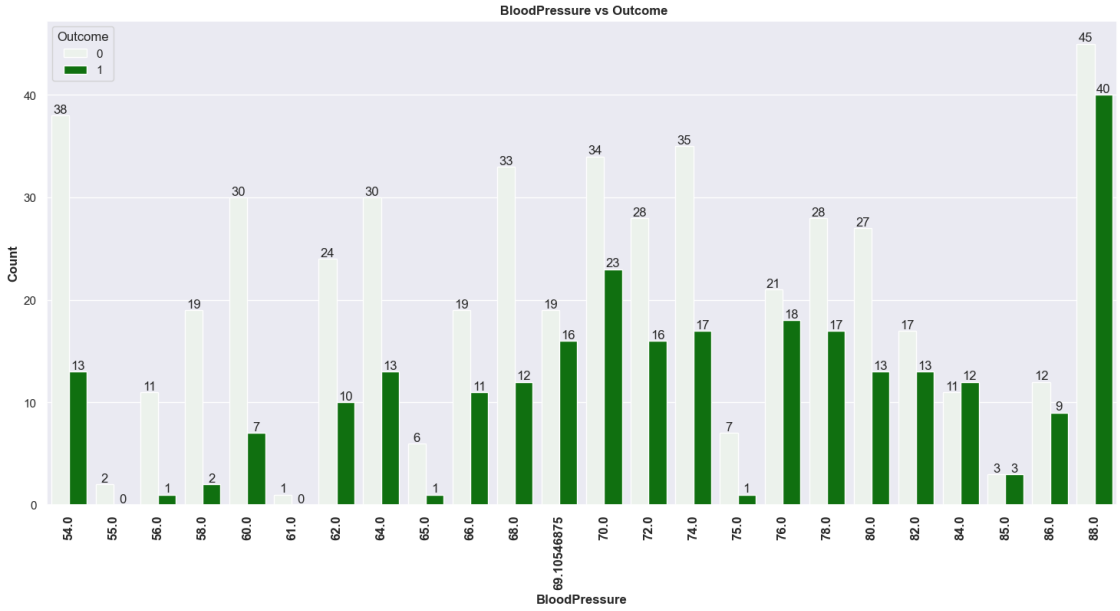
- According above Graph we can observe most of people had diabetes who had got Pregnancie at 9 times followed by No Preganancie

[186]:
```
df.columns
```

[186]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

Pregnancies Vs Outcome

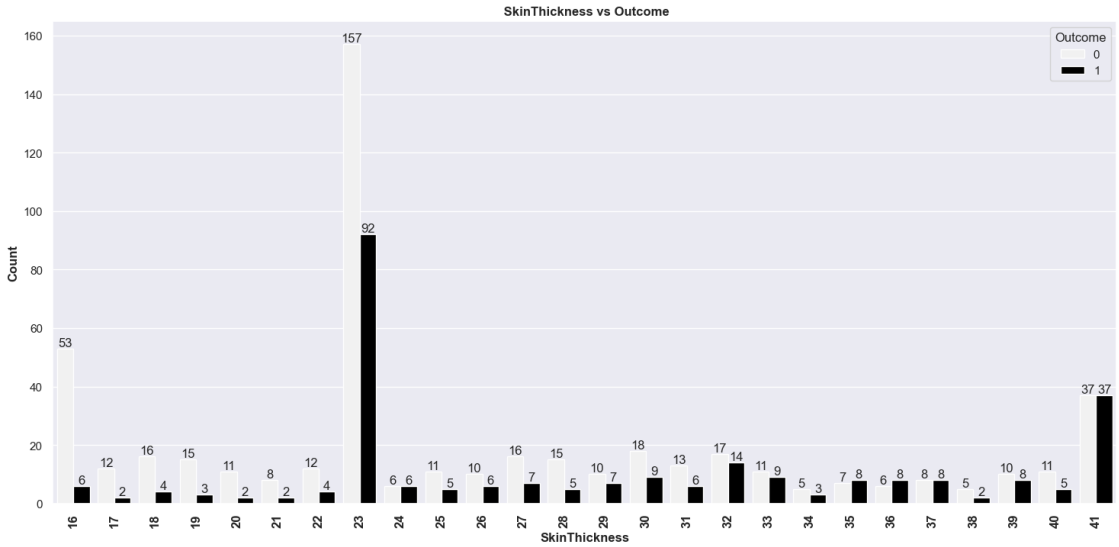
[187]:
```
ax = sns.countplot( x = df['BloodPressure'],hue = df['Outcome'],data = df,color␣
 ↪= 'green')
for bars in ax.containers:
    ax.bar_label(bars)
plt.xlabel("BloodPressure", size = 12, fontweight='bold')
plt.xticks(rotation = 90, size = 12, fontweight='bold')
plt.ylabel("Count", size = 12,fontweight='bold')
plt.title(" BloodPressure vs Outcome",size = 12, fontweight='bold')
plt.show()
```



- According above I observed:
- most of the people had diabetes who had Blood pressure is 88 when compared to remiaing

SkinThickness Vs Outcome

[188]:
```
ax = sns.countplot( x = df['SkinThickness'],hue = df['Outcome'],data = df,color␣
↪= 'black')
for bars in ax.containers:
    ax.bar_label(bars)
plt.xlabel("SkinThickness", size = 12, fontweight='bold')
plt.xticks(rotation = 90, size = 12, fontweight='bold')
plt.ylabel("Count", size = 12,fontweight='bold')
plt.title(" SkinThickness vs Outcome",size = 12, fontweight='bold')
plt.show()
```



Insights:

- The above graph represents the relationship between SkinThickness and Outcome
- more people had diabetes which who had the SkinThickness is 23 mm followed by 41mm

[189]:
```
df.columns
```

[189]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

(6.3) Multivariate Analysis

[190]:
```
sns.pairplot(df)
```

[190]:
```
<seaborn.axisgrid.PairGrid at 0x1e10c1c2fd0>
```

- correlation matrix
- A correlation matrix is a statistical technique used to evaluate the relationship between two variables in a data set. The matrix is a table in which every cell contains a correlation coefficient, where 1 is considered a strong relationship between variables, 0 a neutral relationship and -1 a not strong relationship.

[191]:
```
corr = df.corr()
corr
```

[191]:

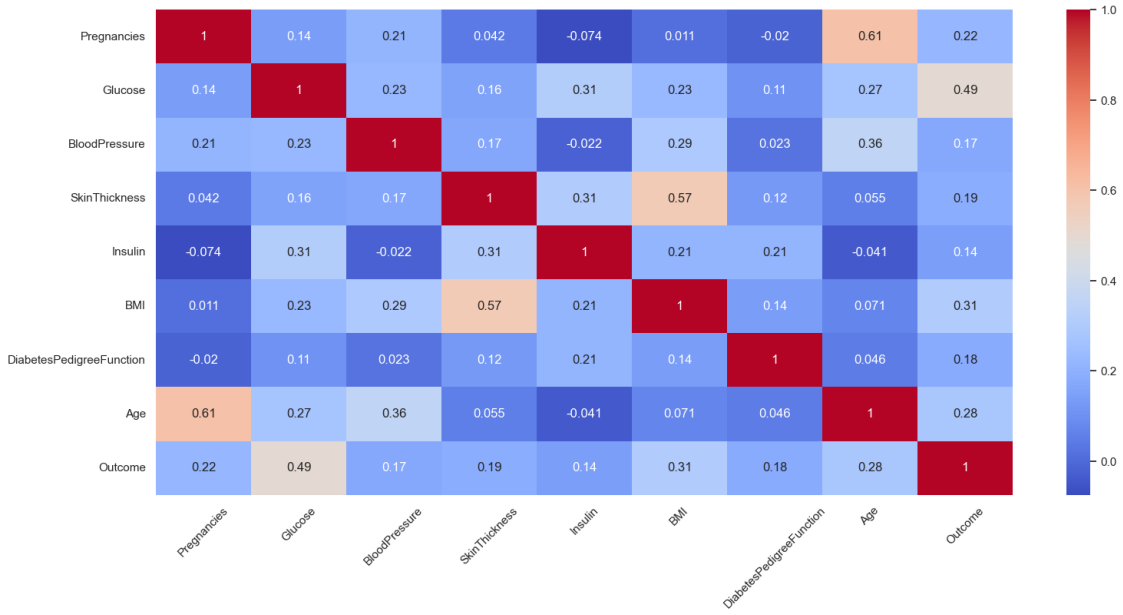
	Pregnancies	Glucose	BloodPressure	SkinThickness	\
Pregnancies	1.000000	0.136131	0.213259	0.042077	
Glucose	0.136131	1.000000	0.226211	0.155425	
BloodPressure	0.213259	0.226211	1.000000	0.167052	

```
SkinThickness                0.042077  0.155425       0.167052       1.000000
Insulin                     -0.074346  0.314589      -0.021627       0.307417
BMI                          0.011103  0.228126       0.293312       0.568028
DiabetesPedigreeFunction    -0.019510  0.106038       0.023420       0.123840
Age                          0.609083  0.272314       0.357636       0.055250
Outcome                      0.221354  0.492908       0.169715       0.187046

                           Insulin       BMI  DiabetesPedigreeFunction  \
Pregnancies              -0.074346  0.011103                 -0.019510
Glucose                   0.314589  0.228126                  0.106038
BloodPressure            -0.021627  0.293312                  0.023420
SkinThickness             0.307417  0.568028                  0.123840
Insulin                   1.000000  0.214567                  0.213582
BMI                       0.214567  1.000000                  0.137851
DiabetesPedigreeFunction  0.213582  0.137851                  1.000000
Age                      -0.040506  0.071340                  0.045905
Outcome                   0.142288  0.306664                  0.179747

                               Age   Outcome
Pregnancies               0.609083  0.221354
Glucose                   0.272314  0.492908
BloodPressure             0.357636  0.169715
SkinThickness             0.055250  0.187046
Insulin                  -0.040506  0.142288
BMI                       0.071340  0.306664
DiabetesPedigreeFunction  0.045905  0.179747
Age                       1.000000  0.282376
Outcome                   0.282376  1.000000
```

- Heat Map
- A heatmap is a graphical representation of data that uses a system of color coding to represent different values. Heatmaps are used in various forms of analytics

[192]:
```
sns.heatmap(corr,cmap = 'coolwarm', annot = True)
plt.xticks(rotation = 45)
```

[192]:
```
(array([0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5]),
 [Text(0.5, 0, 'Pregnancies'),
  Text(1.5, 0, 'Glucose'),
  Text(2.5, 0, 'BloodPressure'),
  Text(3.5, 0, 'SkinThickness'),
  Text(4.5, 0, 'Insulin'),
  Text(5.5, 0, 'BMI'),
  Text(6.5, 0, 'DiabetesPedigreeFunction'),
  Text(7.5, 0, 'Age'),
  Text(8.5, 0, 'Outcome')])
```



Insights:

- In This Heat Map i observed :
- (1) Pregnancies feature Highly correlated with Age which is 0.61
- (2) BMI and SkinThickness correlation value is 0.57
- (3) Insulin had negative correlation with age
- (4) Glucose correlation with outcome which is 0.49

[ ]: