Tableau Public - Data Science Capstone - Healthcare Project

File Data Worksheet Dashboard Story Analysis Map Format Server Window Help

Entire View | Show Me

Data | Analytics

health care diabetes

Search

Tables

- Age Group
- Blood Pressure (bin)
- BMI (bin)
- Diabetes Pedigree Fun...
- Diabetic or Not
- Glucose (bin)
- Insulin (bin)
- Pregnancies (bin)
- Skin Thickness (bin)
- *Measure Names*
- Age
- Blood Pressure
- BMI
- Bubble
- Diabetes Pedigree Fun...
- Glucose
- Insulin
- Outcome
- Pregnancies
- Skin Thickness

Parameters

- Parameter 1

Pages

Columns: Parameter 1

Rows

Filters: Measure Names

Marks: Circle

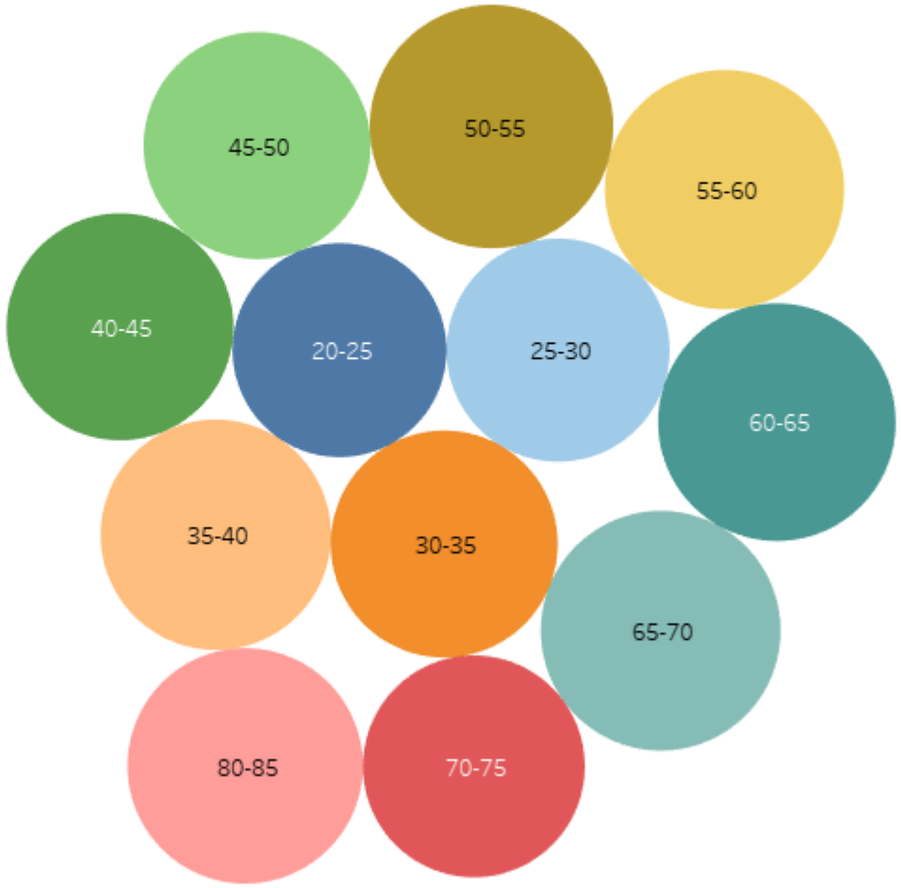
Color | Size | Label | Detail | Tooltip

Age Group

AVG(Bubble)

Age Group

Bubble Chart



Age Group
- 20-25
- 25-30
- 30-35
- 35-40
- 40-45
- 45-50
- 50-55
- 55-60
- 60-65
- 65-70
- 70-75
- 80-85

Data Source | Pie Chart | Scatter | Hist_BP | Hist_BMI | Hist_Glucose | Hist_Insulin | Correlation of Heatmap | **Bubble Chart** | Dashboard 1

12 marks | 1 row by 1 column | SUM of AVG(Bubble): 1,543.13 | kasif ali

Tableau Public - Data Science Capstone - Healthcare Project

File Data Worksheet Dashboard Story Analysis Map Format Server Window Help

Standard Show Me

Data Analytics

health care diabetes

Search

Tables

- Age Group
- Blood Pressure (bin)
- BMI (bin)
- Diabetes Pedigree Fun...
- Diabetic or Not
- Glucose (bin)
- Insulin (bin)
- Pregnancies (bin)
- Skin Thickness (bin)
- *Measure Names*
- Age
- Blood Pressure
- BMI
- Bubble
- Diabetes Pedigree Fun...
- Glucose
- Insulin
- Outcome
- Pregnancies
- Skin Thickness

Parameters

- Parameter 1

Pages

Filters

- Measure Names
- Age Group

Marks

Automatic

Color Size Label Detail Tooltip

Measure Va..

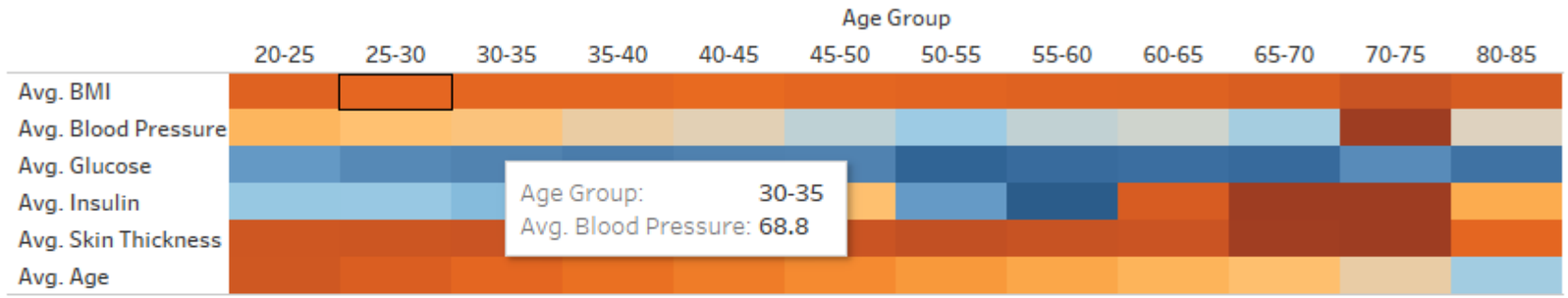
Measure Values

- AVG(BMI)
- AVG(Blood Pressure)
- AVG(Glucose)
- AVG(Insulin)
- AVG(Skin Thickness)
- AVG(Age)

Columns: Age Group

Rows: Measure Names

Correlation of Heatmap



Data Source | Pie Chart | Scatter | Hist_BP | Hist_BMI | Hist_Glucose | Hist_Insulin | Correlation of Heatmap | Bubble Chart | Dashboard 1

72 marks 6 rows by 12 columns SUM of Measure Values: 4,329.3 kasif ali

Tableau Public - Data Science Capstone - Healthcare Project

File Data Worksheet Dashboard Story Analysis Map Format Server Window Help

Entire View | Show Me

Dashboard | Layout

Device Preview

Size

Automatic

Sheets

- Hist_BP
- Hist_BMI
- Hist_Glucose
- Hist_Insulin
- Correlation of ...
- Bubble Chart

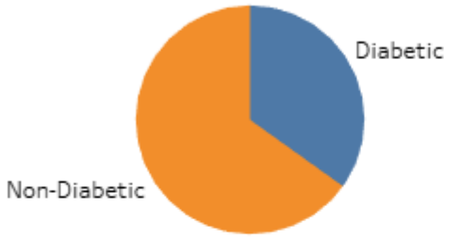
Objects

Horizontal, Blank, Vertical, Navigation, Text, Download, Image, Extension, Web Page, Ask Data, Data Story, Workflow

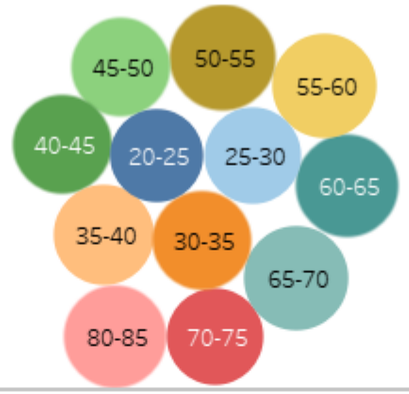
Tiled | Floating

Show dashboard title

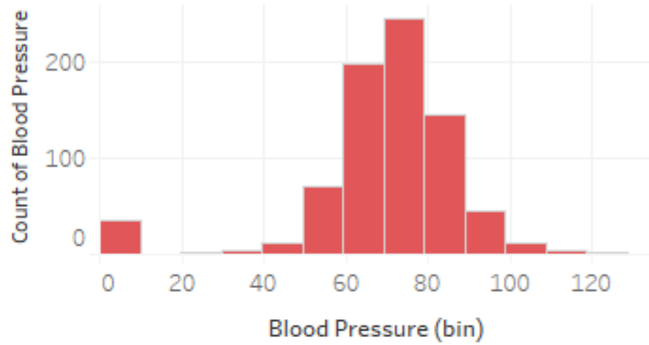
Analysis of Diabetes Population

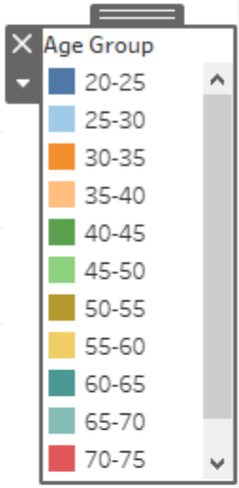


Bubble Chart

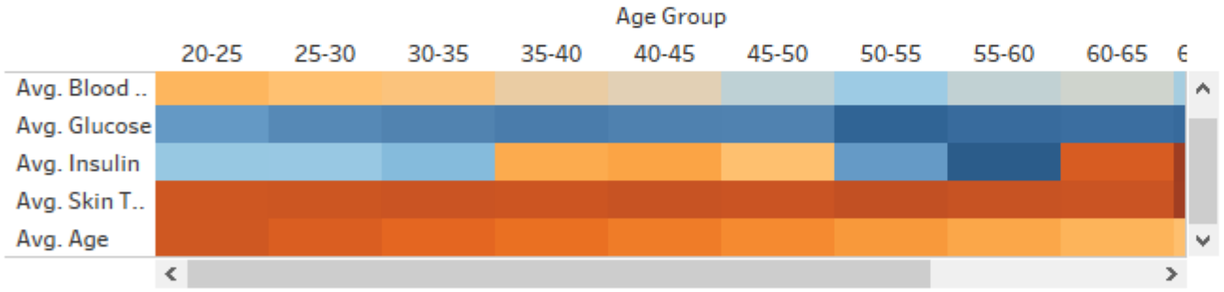


Histogram of Blood Pressure

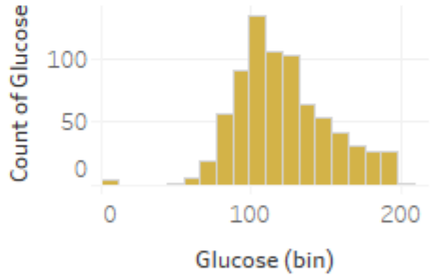




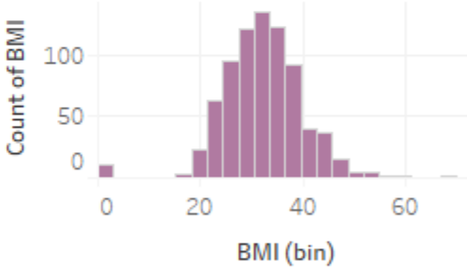
Correlation of Heatmap



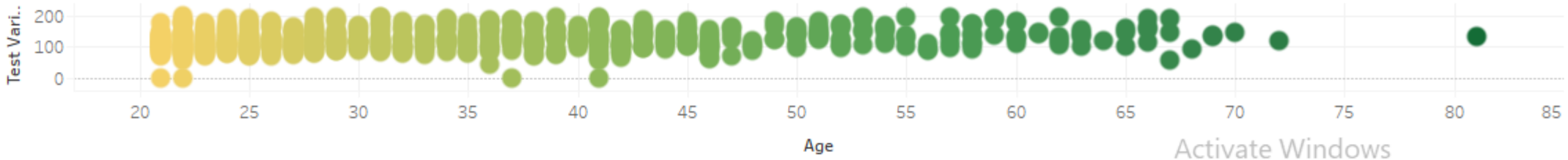
Hist_Glucose



Hist_BMI



Scatter Chart - Analysis of Variable Relationship



Data Source | Pie Chart | Scatter | Hist_BP | Hist_BMI | Hist_Glucose | Hist_Insulin | Correlation of Heatmap | Bubble Chart | Dashboard 1

12 marks 1 row by 1 column SUM of AVG(Bubble): 1,543.13

kasif ali

Tableau Public - Data Science Capstone - Healthcare Project

File Data Worksheet Dashboard Story Analysis Map Format Server Window Help

Standard | Show Me

Data | Analytics

health care diabetes

Search

Tables

- Age Group
- Blood Pressure (bin)
- BMI (bin)
- Diabetes Pedigree Fun...
- Diabetic or Not
- Glucose (bin)
- Insulin (bin)
- Pregnancies (bin)
- Skin Thickness (bin)
- *Measure Names*
- Age
- Blood Pressure
- BMI
- Bubble
- Diabetes Pedigree Fun...
- Glucose
- Insulin
- Outcome
- Pregnancies
- Skin Thickness

Parameters

- Parameter 1

Pages

Filters

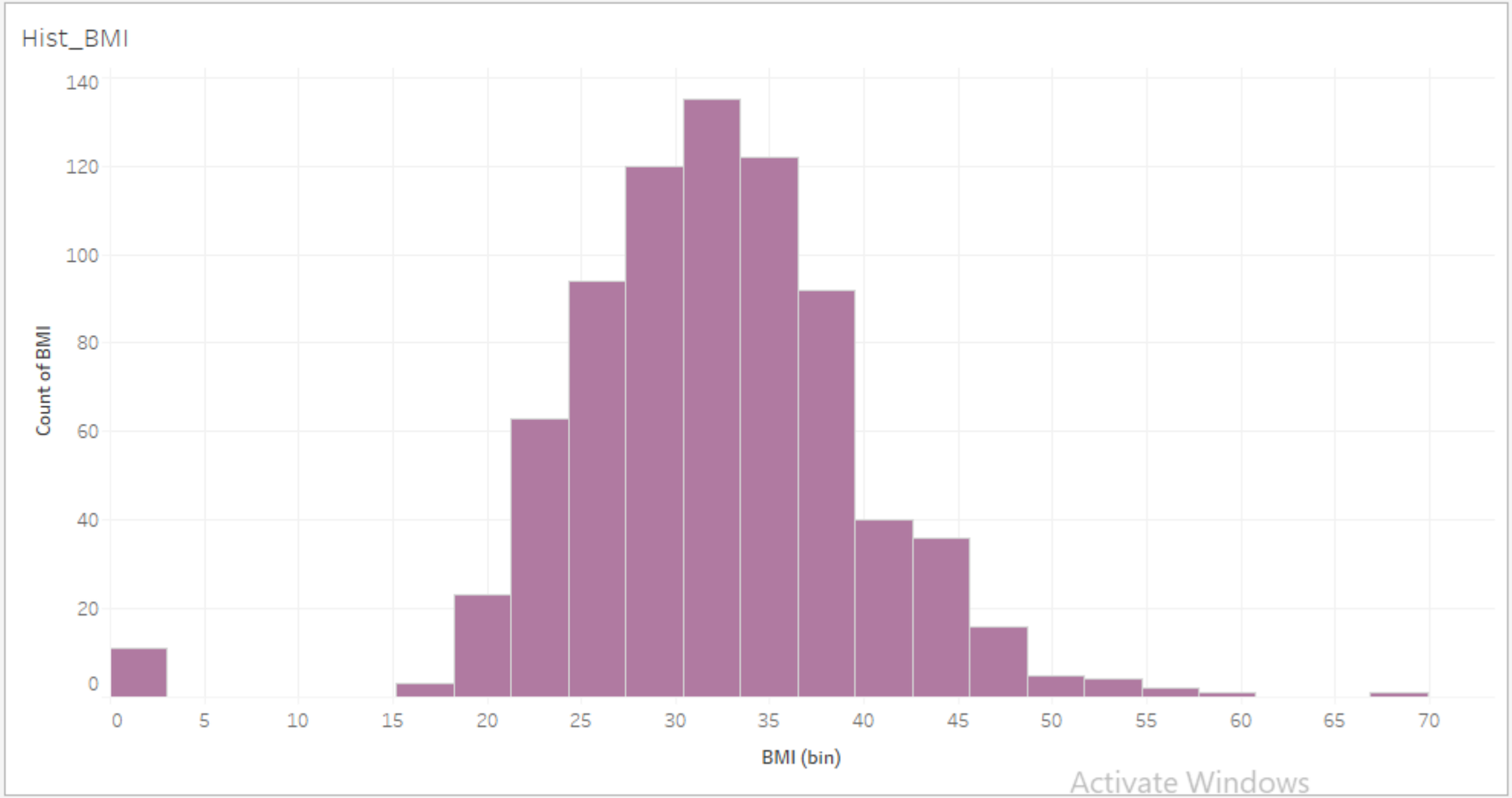
Marks

Automatic

Color | Size | Label | Detail | Tooltip

Columns: BMI (bin)

Rows: CNT(BMI)



Data Source | Pie Chart | Scatter | Hist_BP | Hist_BMI | Hist_DPF | Hist_Glucose | Hist_Insulin | Hist_Pragnancies | Hist_Skin | Correlation of Heatmap | Bubble Chart | Dashboard 1

17 marks 1 row by 1 column SUM of CNT(BMI): 768

kasif ali

Tableau Public - Data Science Capstone - Healthcare Project

File Data Worksheet Dashboard Story Analysis Map Format Server Window Help

Standard | Show Me

Data | Analytics

health care diabetes

Search

Tables

- Age Group
- Blood Pressure (bin)
- BMI (bin)
- Diabetes Pedigree Fun...
- Diabetic or Not
- Glucose (bin)
- Insulin (bin)
- Pregnancies (bin)
- Skin Thickness (bin)
- *Measure Names*
- Age
- Blood Pressure
- BMI
- Bubble
- Diabetes Pedigree Fun...
- Glucose
- Insulin
- Outcome
- Pregnancies
- Skin Thickness

Parameters

- Parameter 1

Pages

Filters

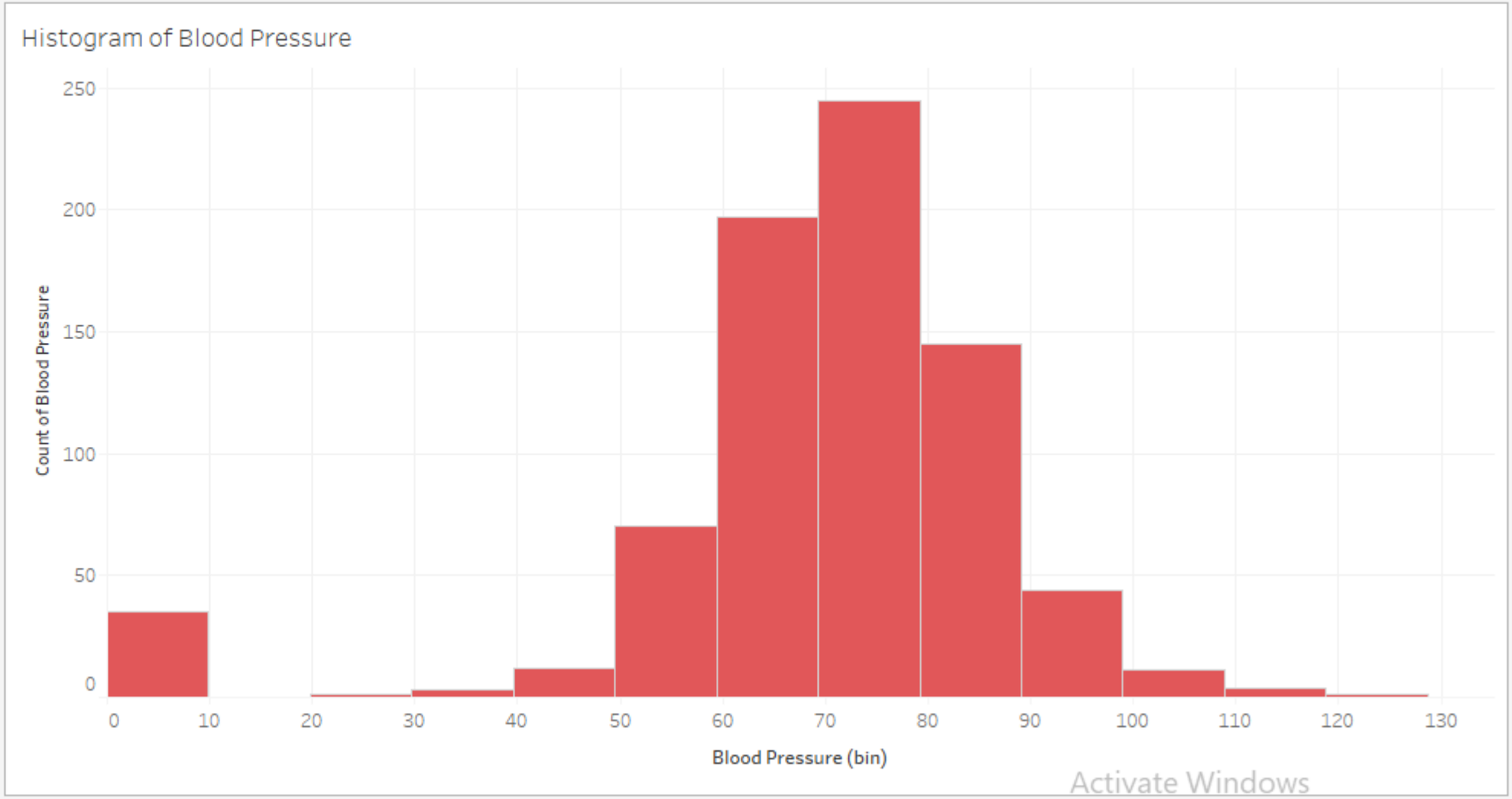
Marks

Automatic

Color | Size | Label | Detail | Tooltip

Columns: Blood Pressure (bin)

Rows: CNT(Blood Pressure)



Data Source | Pie Chart | Scatter | Hist_BP | Hist_BMI | Hist_DPF | Hist_Glucose | Hist_Insulin | Hist_Pragnancies | Hist_Skin | Correlation of Heatmap | Bubble Chart | Dashboard 1

12 marks | 1 row by 1 column | SUM of CNT(Blood Pressure): 768 | kasif ali

Tableau Public - Data Science Capstone - Healthcare Project

File Data Worksheet Dashboard Story Analysis Map Format Server Window Help

Standard Show Me

Data Analytics

health care diabetes

Search

Tables

- Age Group
- Blood Pressure (bin)
- BMI (bin)
- Diabetes Pedigree Fun...
- Diabetic or Not
- Glucose (bin)
- Insulin (bin)
- Pregnancies (bin)
- Skin Thickness (bin)
- *Measure Names*
- Age
- Blood Pressure
- BMI
- Bubble
- Diabetes Pedigree Fun...
- Glucose
- Insulin
- Outcome
- Pregnancies
- Skin Thickness

Parameters

- Parameter 1

Pages

Filters

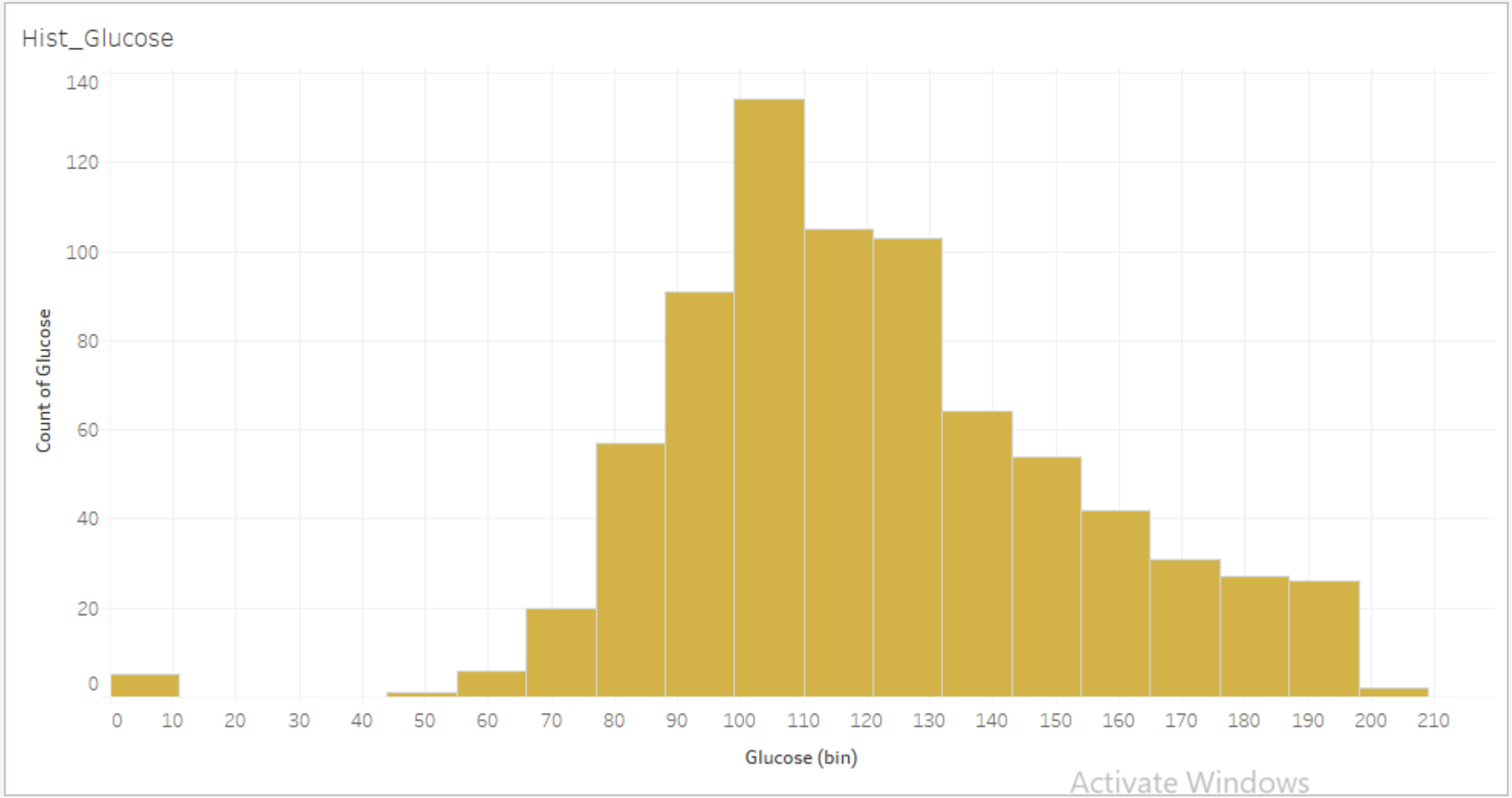
Marks

Automatic

Color Size Label Detail Tooltip

Columns: Glucose (bin)

Rows: CNT(Glucose)



Data Source | Pie Chart | Scatter | Hist_BP | Hist_BMI | Hist_Glucose | Hist_Insulin | Hist_Pragnancies | Hist_Skin | Correlation of Heatmap | Bubble Chart | Dashboard 1

16 marks 1 row by 1 column SUM of CNT(Glucose): 768

kasif ali

Tableau Public - Data Science Capstone - Healthcare Project

File Data Worksheet Dashboard Story Analysis Map Format Server Window Help

Standard | Show Me

Data | Analytics

health care diabetes

Search

Tables

- Age Group
- Blood Pressure (bin)
- BMI (bin)
- Diabetes Pedigree Fun...
- Diabetic or Not
- Glucose (bin)
- Insulin (bin)
- Pregnancies (bin)
- Skin Thickness (bin)
- *Measure Names*
- Age
- Blood Pressure
- BMI
- Bubble
- Diabetes Pedigree Fun...
- Glucose
- Insulin
- Outcome
- Pregnancies
- Skin Thickness

Parameters

- Parameter 1

Pages

Filters

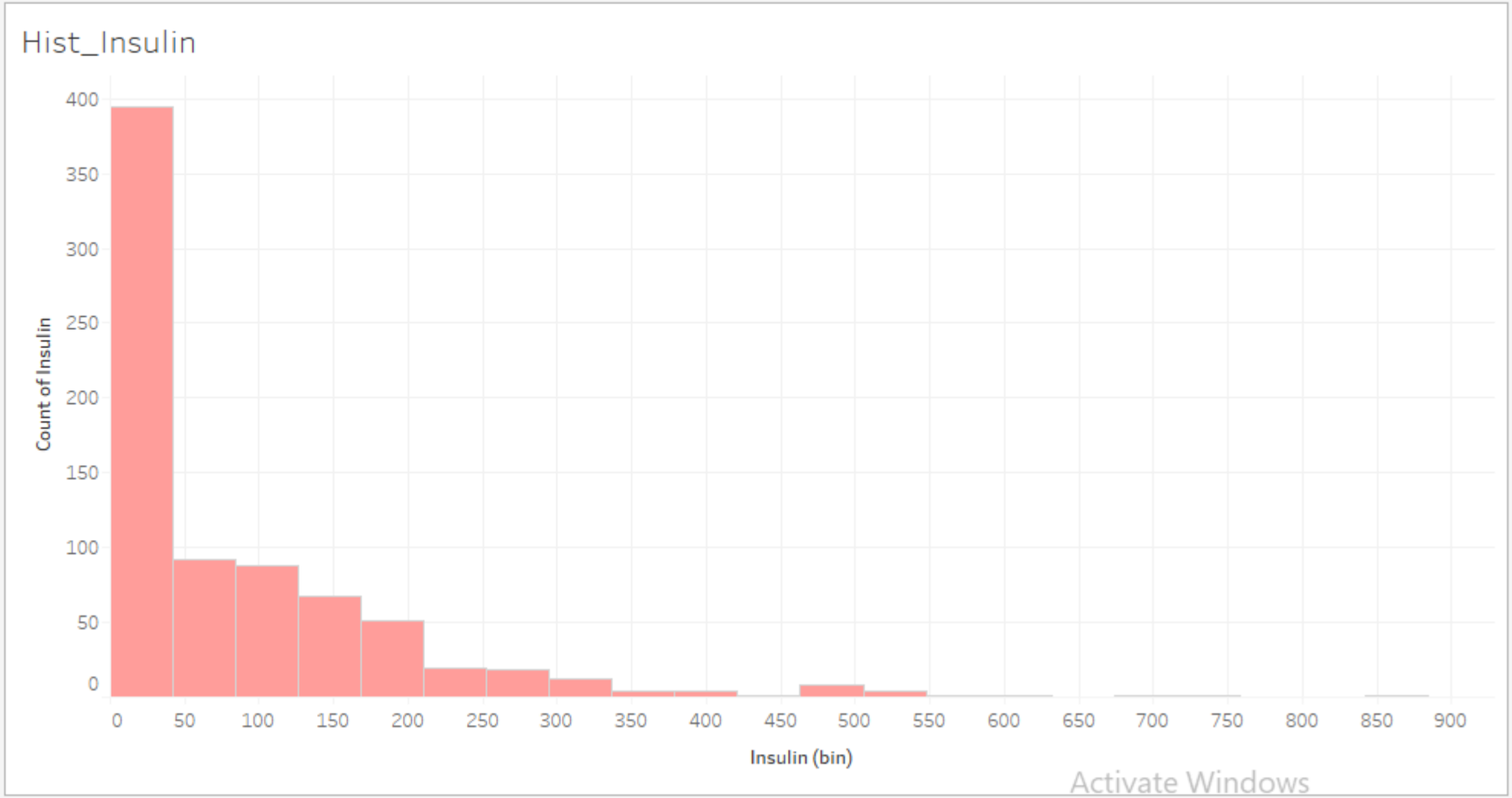
Marks

Automatic

Color | Size | Label | Detail | Tooltip

Columns: Insulin (bin)

Rows: CNT(Insulin)



Data Source | Pie Chart | Scatter | Hist_BP | Hist_BMI | Hist_Glucose | Hist_Insulin | Hist_Pragnancies | Hist_Skin | Correlation of Heatmap | Bubble Chart | Dashboard 1

18 marks 1 row by 1 column SUM of CNT(Insulin): 768

kasif ali

Tableau Public - Data Science Capstone - Healthcare Project

File Data Worksheet Dashboard Story Analysis Map Format Server Window Help

Entire View | Show Me

Data | Analytics

health care diabetes

Search

Tables

- Abc Age Group
- Blood Pressure (bin)
- BMI (bin)
- Diabetes Pedigree Fun...
- Abc Diabetic or Not
- Glucose (bin)
- Insulin (bin)
- Pregnancies (bin)
- Skin Thickness (bin)
- Abc *Measure Names*
- # Age
- # Blood Pressure
- # BMI
- # Bubble
- # Diabetes Pedigree Fun...
- # Glucose
- # Insulin
- # Outcome
- # Pregnancies
- # Skin Thickness

Parameters

- Abc Parameter 1

Pages

Filters

Marks

Pie

Color | Size | Label | Detail | Tooltip | Angle

Diabetic or Not

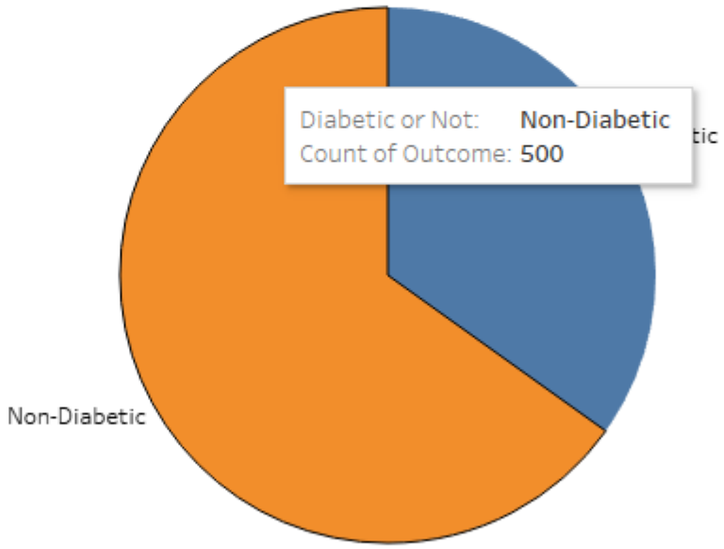
CNT(Outcome)

Diabetic or Not

Columns

Rows

Analysis of Diabetes Population



Data Source | Pie Chart | Scatter | Hist_BP | Hist_BMI | Hist_DPF | Hist_Glucose | Hist_Insulin | Hist_Pragnancies | Hist_Skin | Correlation of Heatmap | Bubble Chart | Dashboard 1

2 marks 1 row by 1 column SUM of CNT(Outcome): 768

kasif ali

Tableau Public - Data Science Capstone - Healthcare Project

File Data Worksheet Dashboard Story Analysis Map Format Server Window Help

Standard | Show Me

Data | Analytics

health care diabetes

Search

Tables

- Age Group
- Blood Pressure (bin)
- BMI (bin)
- Diabetes Pedigree Fun...
- Diabetic or Not
- Glucose (bin)
- Insulin (bin)
- Pregnancies (bin)
- Skin Thickness (bin)
- *Measure Names*
- Age
- Blood Pressure
- BMI
- Bubble
- Diabetes Pedigree Fun...
- Glucose
- Insulin
- Outcome
- Pregnancies
- Skin Thickness

Parameters

- Parameter 1

Pages

Filters

SUM(Test Variable)

Marks

Automatic

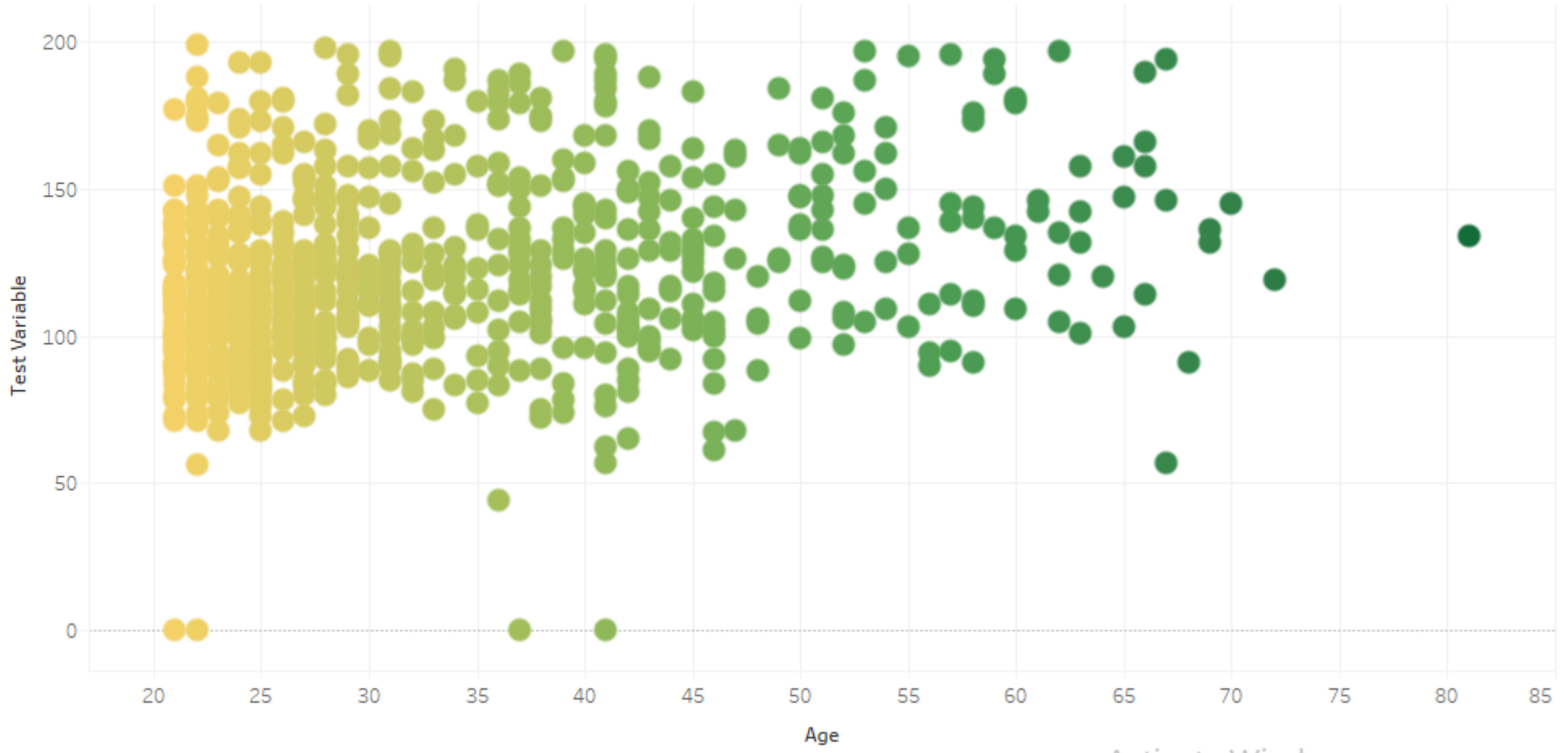
Color | Size | Label | Detail | Tooltip | Shape

Age

Columns: Age

Rows: Test Variable

Scatter Chart - Analysis of Variable Relationship



Data Source | Pie Chart | Scatter | Hist_BP | Hist_BMI | Hist_DPF | Hist_Glucose | Hist_Insulin | Hist_Pragnancies | Hist_Skin | Correlation of Heatmap | Bubble Chart | Dashboard 1

664 marks 1 row by 1 column

kasif ali

jupyter Data Science Capstone Project Healthcare Last Checkpoint: a few seconds ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help   Trusted | Python 3 (ipykernel)

WEEK 1
Reading Data and Performing descriptive analysis

In [48]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
```

In [4]:
```python
import os
print(os.getcwd())
os.listdir(os.getcwd())
```

```
C:\Users\User1\Desktop\Capstone Project 2 Healthcare\Project 2
```

Out[4]:
```
['.ipynb_checkpoints',
 'Case Study - healthcare Industry.docx',
 'Data Science Capstone Project Healthcare.ipynb',
 'health care diabetes.csv',
 'Healthcare - Diabetes',
 'healthcare appointment data.csv',
 'train.csv']
```

In [5]:
```python
df=pd.read_csv('health care diabetes.csv')
df.head()
```

Out[5]:

	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

In [6]:
```python
# It shows how many null values are present and their data types
df.info()
```

```
<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
```

In [7]:
```python
# it displays unique value in each column
df.nunique()
```

Out[7]:
```
Pregnancies                  17
Glucose                     136
BloodPressure                47
SkinThickness                51
Insulin                     186
BMI                         248
DiabetesPedigreeFunction    517
Age                          52
Outcome                       2
dtype: int64
```

In [8]:
```python
# here Min column shows 0 which means missing values
df.describe()
```

Out[8]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
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

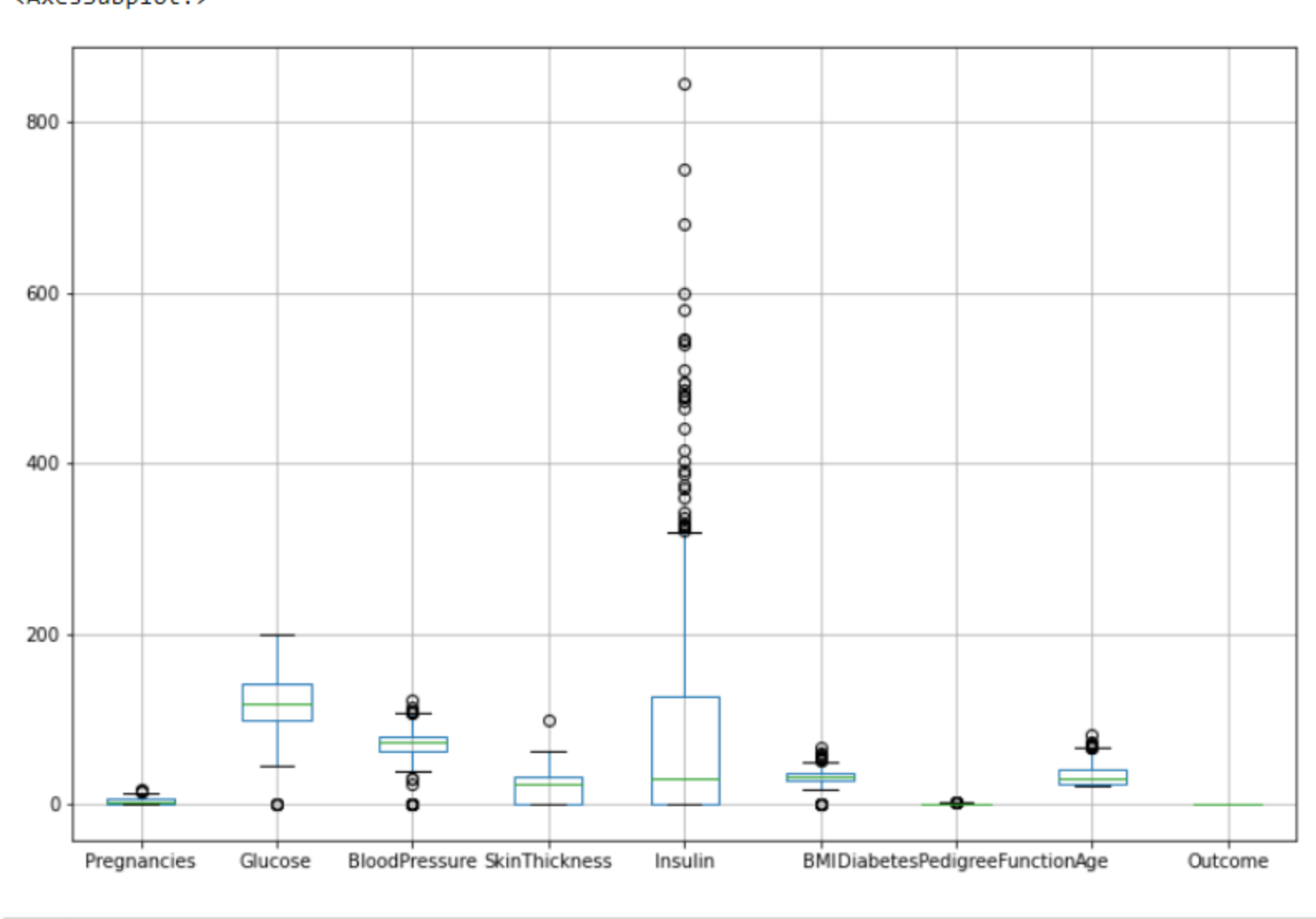
In [9]:
```python
df.corr()
```

Out[9]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	-0.033523	0.544341	0.221898
Glucose	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	0.137337	0.263514	0.466581
BloodPressure	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	0.041265	0.239528	0.065068
SkinThickness	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.183928	-0.113970	0.074752
Insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	0.185071	-0.042163	0.130548
BMI	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	0.140647	0.036242	0.292695
DiabetesPedigreeFunction	-0.033523	0.137337	0.041265	0.183928	0.185071	0.140647	1.000000	0.033561	0.173844
Age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	0.033561	1.000000	0.238356
Outcome	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	0.173844	0.238356	1.000000

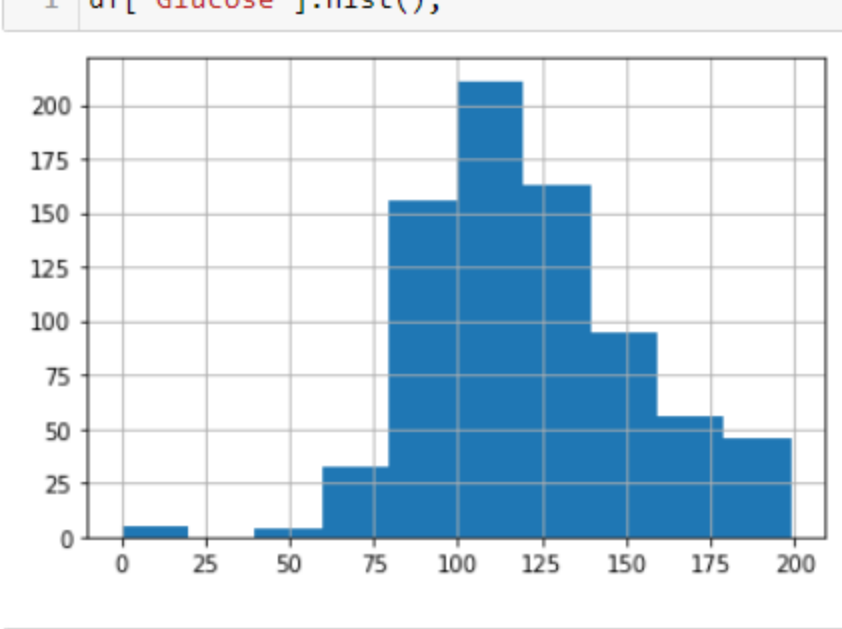
In [10]:
```python
plt.figure(figsize=(12,8))
df.boxplot()
```

Out[10]: <AxesSubplot:>

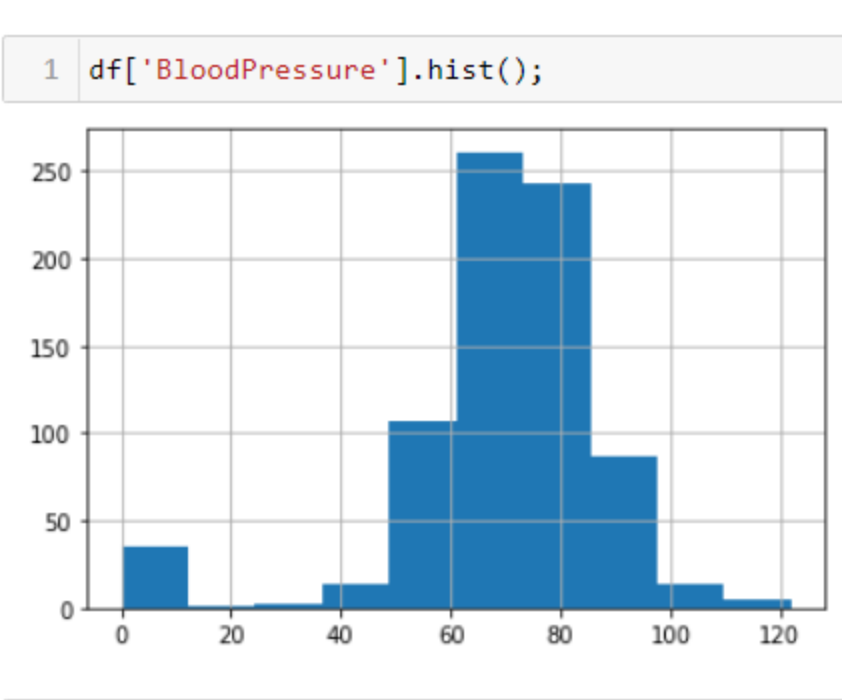


In [ ]:
```python
# Visually exploring these variables using histograms. Treating the missing values accordingly
```

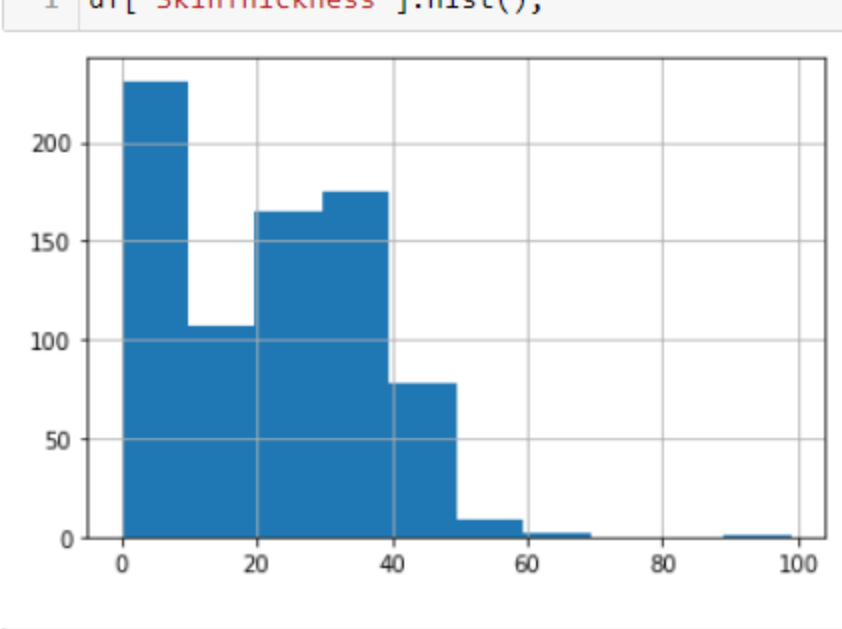
In [11]:
```python
df['Glucose'].hist();
```



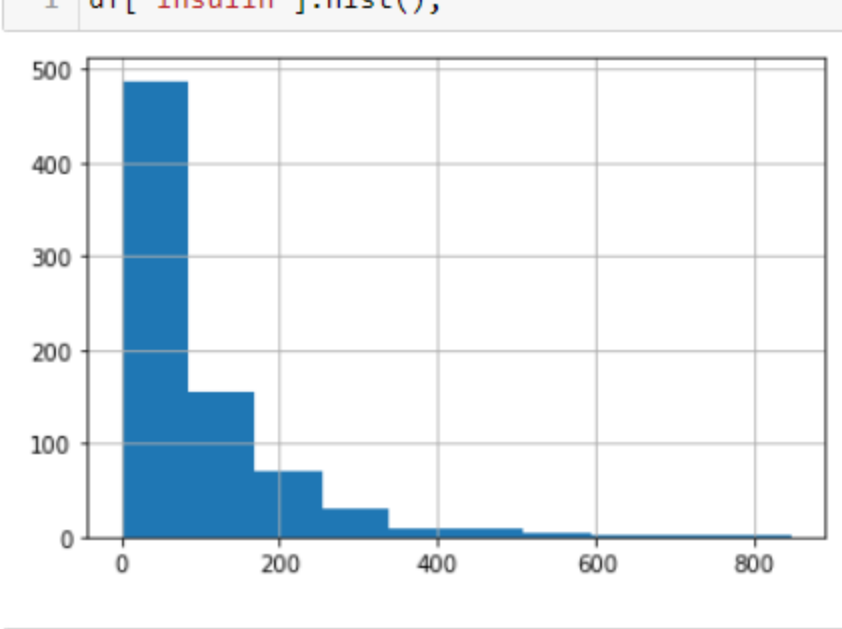
In [12]:
```python
df['BloodPressure'].hist();
```



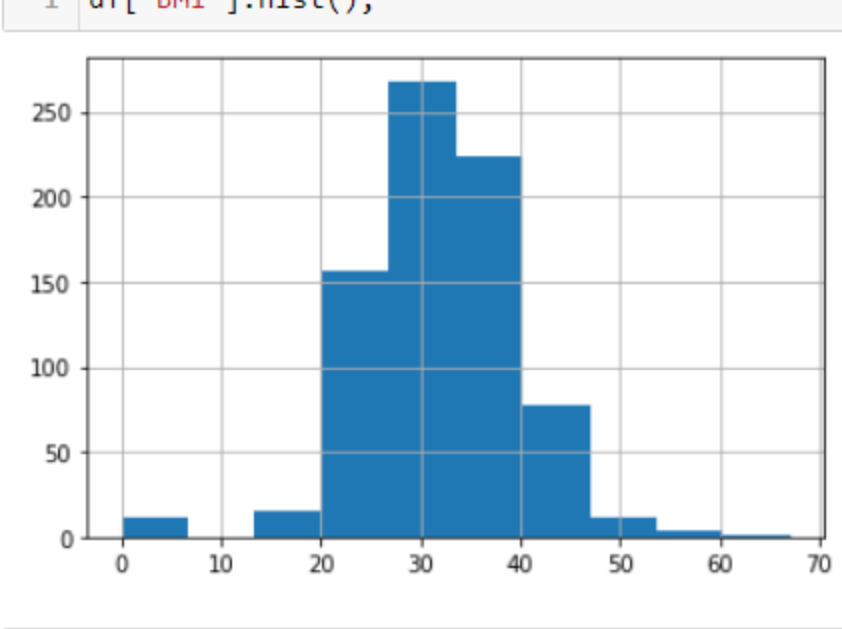
In [13]:
```python
df['SkinThickness'].hist();
```



In [14]:
```python
df['Insulin'].hist();
```



In [15]:
```python
df['BMI'].hist();
```



In [ ]:
```python
# From above histograms, it is clear that Insulin has highly skewed data distribution and remaining 4 variables have
# relatively balanced data distribution therefore we will treat missing values in these 5 variables
```

In [16]:
```python
df.columns
```

Out[16]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [17]:
```python
# No Null values shows because isnull.sum captures null for Nan values.
df.isnull().sum()
```

Out[17]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [18]:
```python
df.shape
```

Out[18]: (768, 9)

In [19]:
```python
# Here we are identifying how much %tage of 0 is there in each columns.
(df[df[['Glucose','BloodPressure','SkinThickness','Insulin',"BMI"]]==0]).count()/len(df)*100
```

Out[19]:
```
Pregnancies                  0.000000
Glucose                      0.651042
BloodPressure                4.557292
SkinThickness               29.557292
Insulin                     48.697917
BMI                          1.432292
DiabetesPedigreeFunction     0.000000
Age                          0.000000
Outcome                      0.000000
dtype: float64
```

In [ ]:
```python
# Treating the missing value
# Here taken Median bcz we havenot done the outlier treatment if we would have done outlier then we can take Mean also.
```

In [20]:
```python
# This median calcln is considering the zeros present in the columns thus providing the wrong values
df['Insulin'].median()
```

Out[20]: 30.5

In [21]:
```python
# The median value for column Insulin before removing 0 is 30.50 and after removing o is 125.00
df[df['Insulin']!=0]['Insulin'].median()
```

Out[21]: 125.0

In [25]:
```python
for i in ['Glucose','BloodPressure','SkinThickness','Insulin','BMI']:
    print (i, "old median:", df[i].median())
    median_value = df[df[i]!=0][i].median()
    print (median_value,'\n')
    df[i].replace(0,median_value, inplace=True)
```

```
Glucose old median: 117.0
117.0

BloodPressure old median: 72.0
72.0

SkinThickness old median: 23.0
29.0

Insulin old median: 30.5
125.0

BMI old median: 32.0
32.3
```

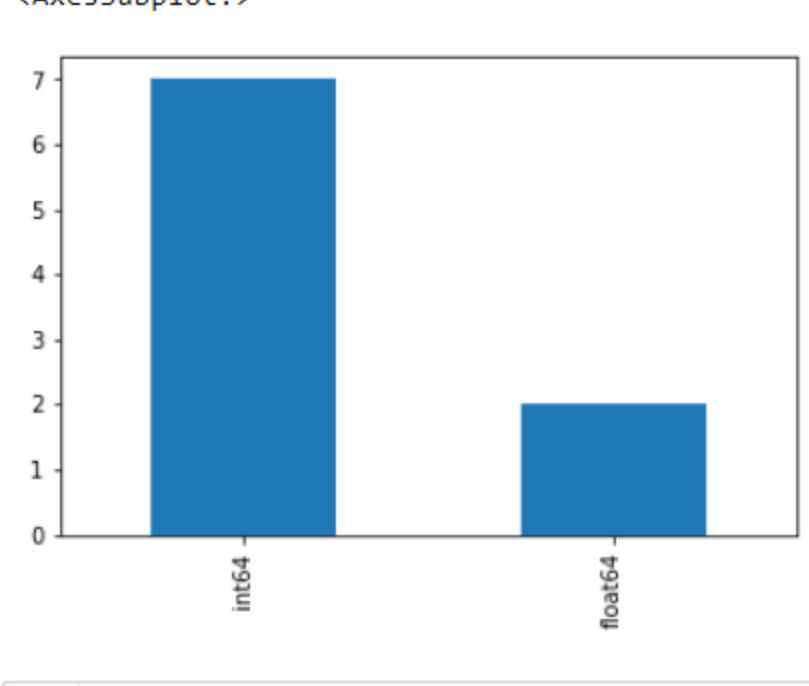
In [26]:
```python
# here it shows that in all columns whereever the value is 0 the count is shown for all columns
(df[df[['Glucose','BloodPressure','SkinThickness','Insulin',"BMI"]]==0]).count()
```

Out[26]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [ ]:
```python
# Creating a count (frequency) plot describing the data types and the count of variables:
```

In [27]:
```python
df.dtypes.value_counts().plot(kind='bar')
```

Out[27]: <AxesSubplot:>



WEEK 2

In [ ]:
```python
# DATA EXPLORATION:
# Checking the balance of the data by plotting the count of outcomes by their value. Describe your findings and plan
# future course of action:
```

In [28]:
```python
df.columns
```

Out[28]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [29]:
```python
df.shape
```

Out[29]: (768, 9)

In [30]:
```python
# It will tell me the number of Counts in percentage distribution
df['Outcome'].value_counts(normalize=True)*100
```

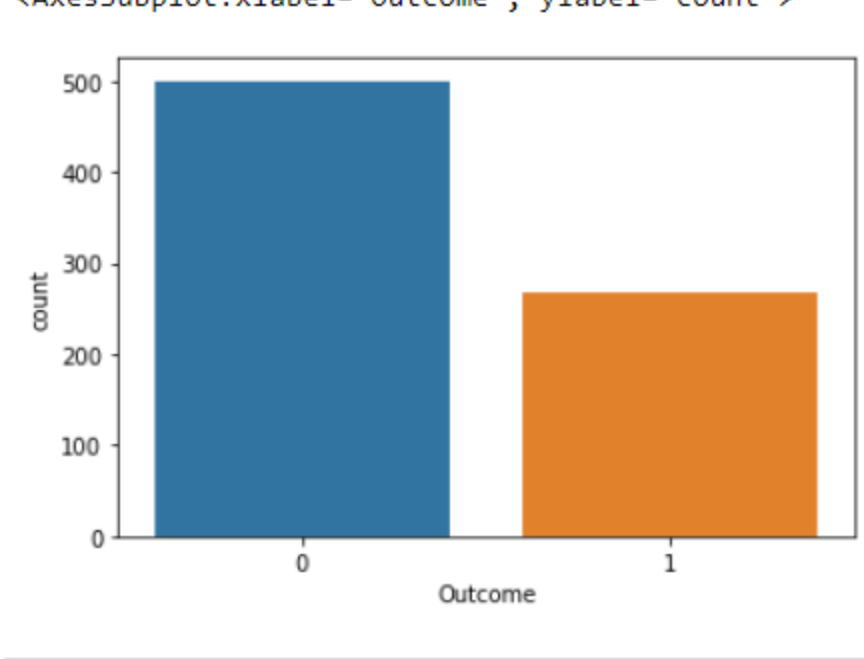
Out[30]:
```
0    65.104167
1    34.895833
Name: Outcome, dtype: float64
```

In [31]:
```python
# It will tell me the number of Counts without percentage directly showing no of rows having value 0 & 1.
df['Outcome'].value_counts()
```

Out[31]:
```
0    500
1    268
Name: Outcome, dtype: int64
```

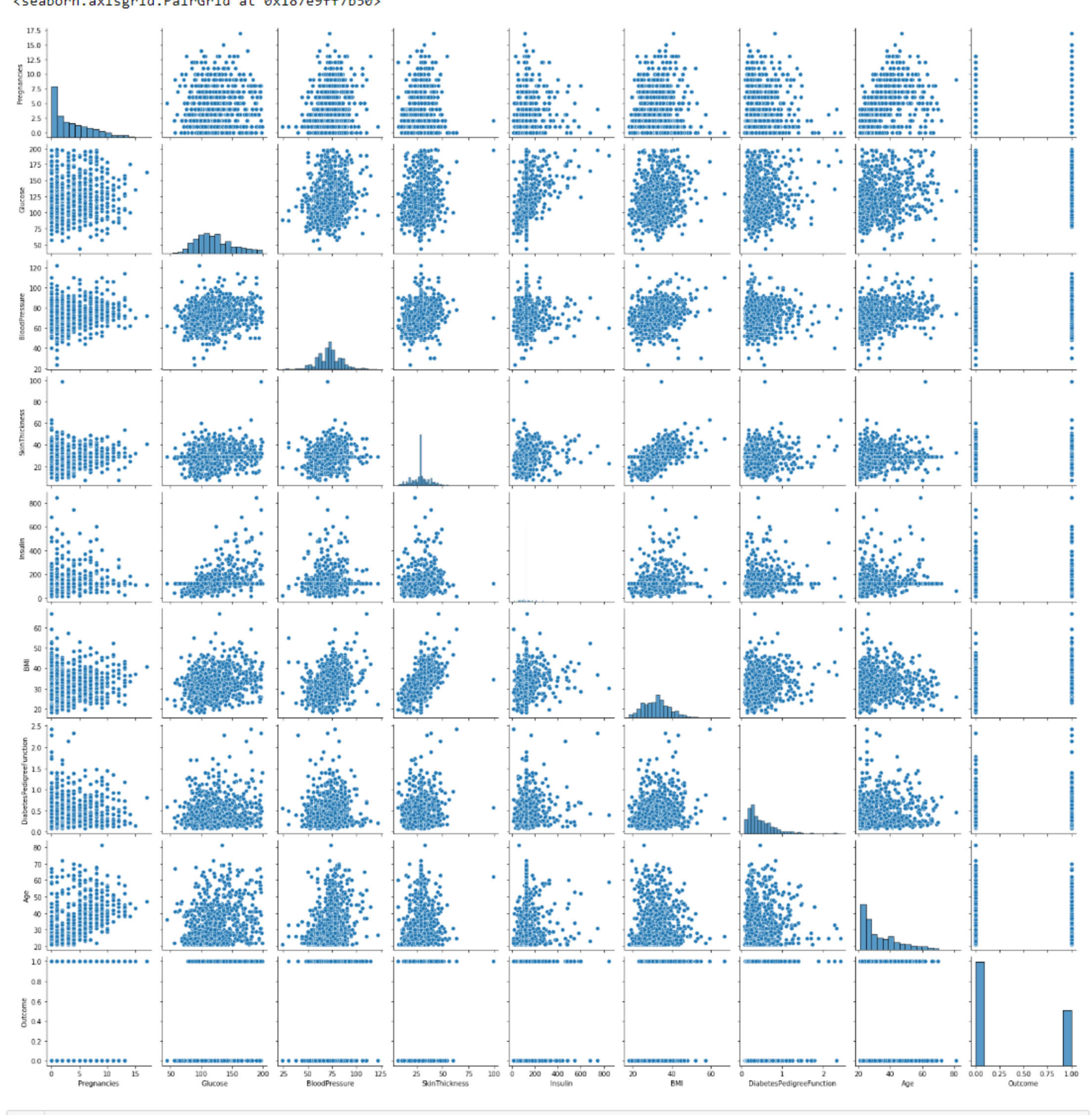
In [33]:
```python
# This is just showing the above cell in bar plot
import warnings
warnings.simplefilter('ignore')
sns.countplot(df.Outcome)
```

Out[33]: <AxesSubplot:xlabel='Outcome', ylabel='count'>



In [34]:
```python
# Creating scatter charts between the pair of variables to understand the relationships.
sns.pairplot(df)
```

Out[34]: <seaborn.axisgrid.PairGrid at 0x187e9ff7b50>



###
We have some observations from above scatter plot of pairs of features:
(1) Glucose alone is impressively good to distinguish between the Outcome classes.
(2) Age alone is also able to distinguish between classes to some extent.
(3) It seems none of pairs in the dataset is able to clealry distinguish between the Outcome classes.
(4) With increase in skin thickiness the BMI value also increases.
###

In [ ]:
```python
# Performing correlation analysis. Visually exploring it using a heat map:
```

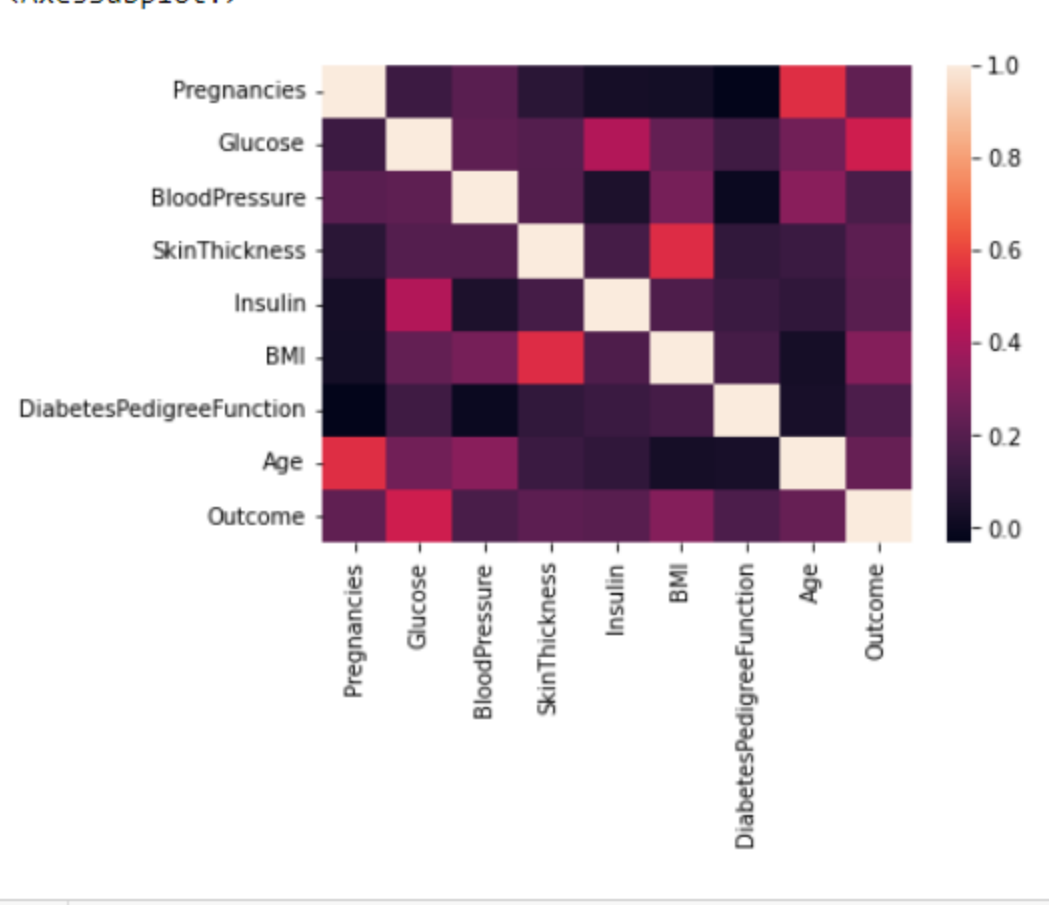
In [35]:
```python
df.corr()
```

Out[35]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1.000000	0.128213	0.208615	0.081770	0.025047	0.021559	-0.033523	0.544341	0.221898
Glucose	0.128213	1.000000	0.218937	0.192615	0.419451	0.231049	0.137327	0.266909	0.492782
BloodPressure	0.208615	0.218937	1.000000	0.191892	0.045363	0.281257	-0.002378	0.324915	0.165723
SkinThickness	0.081770	0.192615	0.191892	1.000000	0.155610	0.543205	0.102188	0.126107	0.214873
Insulin	0.025047	0.419451	0.045363	0.155610	1.000000	0.180241	0.126503	0.097101	0.203790
BMI	0.021559	0.231049	0.281257	0.543205	0.180241	1.000000	0.153438	0.025597	0.312038
DiabetesPedigreeFunction	-0.033523	0.137327	-0.002378	0.102188	0.126503	0.153438	1.000000	0.033561	0.173844
Age	0.544341	0.266909	0.324915	0.126107	0.097101	0.025597	0.033561	1.000000	0.238356
Outcome	0.221898	0.492782	0.165723	0.214873	0.203790	0.312038	0.173844	0.238356	1.000000

In [36]:
```python
# in heat map Dark color shows good corr and light color shows bad corr.
sns.heatmap(df.corr())
```

Out[36]: <AxesSubplot:>



###
It appears from correlation matrix and heatmap that there exists significant correlation between some pairs such as -
Age-Pregnancies
BMI-SkinThickness
Also we can see that no pair of variables have negative correlation
###

WEEK 3:

Since this is a classification problem, we will be building all popular classification models for our training data and then compare performance of each model on test data to accurately predict target variable (Outcome):

In [37]:
```python
df.columns
```

Out[37]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [40]:
```python
x=df.drop('Outcome',axis=1) # This is Featured
y=df['Outcome']             # This is label
```

In [41]:
```python
df['Outcome'].value_counts(normalize=True)
```

Out[41]:
```
0    0.651042
1    0.348958
Name: Outcome, dtype: float64
```

In [42]:
```python
# Here Stratify do the divison b/w train & test so that the balance is maintained of outcome of above line.
# in stratify=y Y means the target variable or the label.
train_x,test_x,train_y,test_y=train_test_split(x,y,test_size=.25,random_state=42,stratify=y)
```

In [43]:
```python
train_x.shape
```

Out[43]: (576, 8)

In [44]:
```python
test_x.shape
```

Out[44]: (192, 8)

In [45]:
```python
# here the split or ratio is now same compare to the overall ratio
train_y.value_counts(normalize=True)
```

Out[45]:
```
0    0.651042
1    0.348958
Name: Outcome, dtype: float64
```

In [46]:
```python
test_y.value_counts(normalize=True)
```

Out[46]:
```
0    0.651042
1    0.348958
Name: Outcome, dtype: float64
```

KNN ALGORITHM

In [50]:
```python
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import StandardScaler
```

In [51]:
```python
scaler=StandardScaler()
```

In [52]:
```python
scalled_train_x = scaler.fit_transform(train_x)
```

In [53]:
```python
scalled_test_x = scaler.fit_transform(test_x)
```

In [54]:
```python
# (n_neighbors = 5) means no of neighbors is equal to 5
model = KNeighborsClassifier(n_neighbors = 5)
```

In [55]:
```python
model.fit(scalled_train_x,train_y)
```

Out[55]:
```
KNeighborsClassifier
KNeighborsClassifier()
```

In [56]:
```python
# Accuracy on Training Data
model.score(scalled_train_x,train_y)
```

Out[56]: 0.8368055555555556

In [57]:
```python
# Accuracy on Testing Data
model.score(scalled_test_x,test_y)
```

Out[57]: 0.75

In [58]:
```python
from sklearn.metrics import classification_report,confusion_matrix
```

In [59]:
```python
y_pred=model.predict(scalled_test_x)
```

In [61]:
```python
# Checking the counts in prediction
pd.DataFrame(y_pred).value_counts()
```

Out[61]:
```
0    137
1     55
dtype: int64
```

In [62]:
```python
# in Healthcare domain Recall is the widely used metric
# Sensitivity- Recall of class 1 = out of actual 1 how many we predict correctly
# Specificity- Recall of class 0

print(classification_report(test_y,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.78      0.86      0.82       125
           1       0.67      0.55      0.61        67

    accuracy                           0.75       192
   macro avg       0.73      0.70      0.71       192
weighted avg       0.74      0.75      0.74       192
```

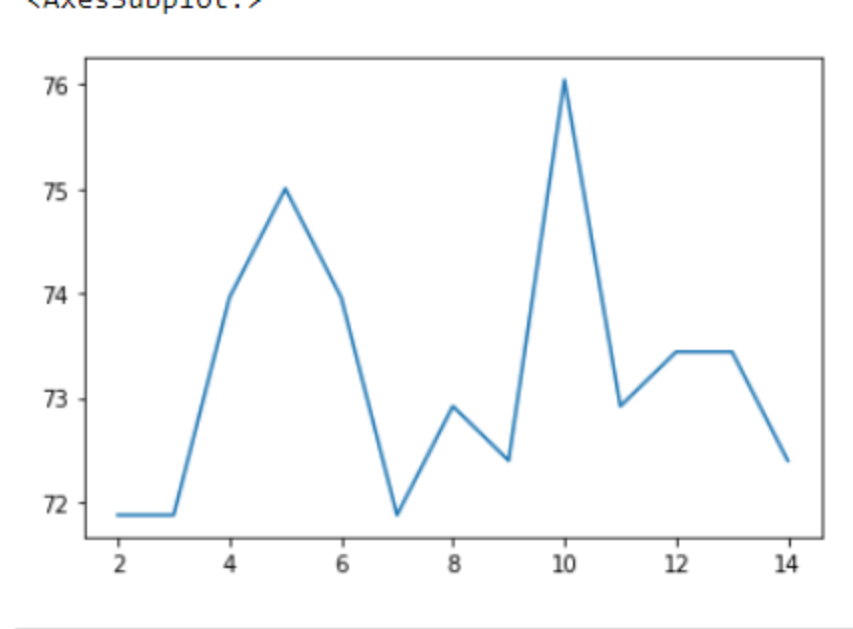
In [63]:
```python
print(confusion_matrix(test_y,y_pred))
```

```
[[107  18]
 [ 30  37]]
```

In [64]:
```python
acc=[]
ran=range(2,15)
for k in ran:
    model=KNeighborsClassifier(n_neighbors=k)
    model.fit(scalled_train_x,train_y)
    test_acc=model.score(scalled_test_x,test_y)
    acc.append(test_acc*100)
```

In [66]:
```python
sns.lineplot(ran,acc)
```

Out[66]: <AxesSubplot:>



In [ ]:
```python
# From above line graph we come to know that we are betting the best accuracy ot k=8
```

RANDOM FOREST ALGORITHM

In [68]:
```python
from sklearn.ensemble import RandomForestClassifier
RF_model = RandomForestClassifier(n_estimators=500,max_depth=3)
```

In [70]:
```python
RF_model.fit(scalled_train_x,train_y)
```

Out[70]:
```
RandomForestClassifier
RandomForestClassifier(max_depth=3, n_estimators=500)
```

In [71]:
```python
RF_model.score(scalled_train_x,train_y)
```

Out[71]: 0.8020833333333334

In [72]:
```python
RF_model.score(scalled_test_x,test_y)
```

Out[72]: 0.7239583333333334

In [73]:
```python
rf_y_pred=RF_model.predict(scalled_test_x)
```

In [74]:
```python
pd.DataFrame(rf_y_pred).value_counts()
```

Out[74]:
```
0    152
1     40
dtype: int64
```

In [75]:
```python
# in Healthcare domain Recall is the widely used metric
# Sensitivity- Recall of class 1
# Specificity- Recall of class 0

print(classification_report(test_y,rf_y_pred))
```

```
              precision    recall  f1-score   support

           0       0.74      0.90      0.81       125
           1       0.68      0.40      0.50        67

    accuracy                           0.72       192
   macro avg       0.71      0.65      0.66       192
weighted avg       0.72      0.72      0.70       192
```

Among all models, RandomForest has given best accuracy and f1_score. Therefore we will build final model using RandomForest.