The project 2: "Diabetes Patients"

About Dataset This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether a patient has diabetes based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.2 From the data set in the (.csv) File We can find several variables, some of them are independent (several medical predictor variables) and only one target dependent variable (Outcome).

In [1]:
```
# Import Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import style
import seaborn as sns
%matplotlib inline
```

In [2]:
```
# Load data from the files diabetes.csv
df = pd.DataFrame()
df = pd.read_csv('diabetes.csv')
```

In [3]:
```
df
```

Out[3]:

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

768 rows × 9 columns

In [4]:
```
df.columns
```

Out[4]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [5]:
```
df.head()
```

Out[5]:

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

In [6]:
```
df.tail(10)
```

Out[6]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction
758	1	106	76	0	0	37.5	0.197
759	6	190	92	0	0	35.5	0.278
760	2	88	58	26	16	28.4	0.766
761	9	170	74	31	0	44.0	0.403
762	9	89	62	0	0	22.5	0.142
763	10	101	76	48	180	32.9	0.171
764	2	122	70	27	0	36.8	0.340
765	5	121	72	23	112	26.2	0.245
766	1	126	60	0	0	30.1	0.349
767	1	93	70	31	0	30.4	0.315

In [7]:
```
df.sample(35)
```

Out[7]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction
742	1	109	58	18	116	28.5	0.219
135	2	125	60	20	140	33.8	0.088
599	1	109	38	18	120	23.1	0.407
104	2	85	65	0	0	39.6	0.930
31	3	158	76	36	245	31.6	0.851
65	5	99	74	27	0	29.0	0.203
152	9	156	86	28	155	34.3	1.189
302	5	77	82	41	42	35.8	0.156
337	5	115	76	0	0	31.2	0.343
288	4	96	56	17	49	20.8	0.340
621	2	92	76	20	0	24.2	1.698
73	4	129	86	20	270	35.1	0.231
598	1	173	74	0	0	36.8	0.088
620	2	112	86	42	160	38.4	0.246
427	1	181	64	30	180	34.1	0.328
556	1	97	70	40	0	38.1	0.218
530	2	122	60	18	106	29.8	0.717
5	5	116	74	0	0	25.6	0.201
390	1	100	66	29	196	32.0	0.444
183	5	73	60	0	0	26.8	0.268
578	10	133	68	0	0	27.0	0.245
581	6	109	60	27	0	25.0	0.206
77	5	95	72	33	0	37.7	0.370
255	1	113	64	35	0	33.6	0.543
340	1	130	70	13	105	25.9	0.472
153	1	153	82	42	485	40.6	0.687
468	8	120	0	0	0	30.0	0.183
406	4	115	72	0	0	28.9	0.376
433	2	139	75	0	0	25.6	0.167
20	3	126	88	41	235	39.3	0.704
401	6	137	61	0	0	24.2	0.151
638	7	97	76	32	91	40.9	0.871
494	3	80	0	0	0	0.0	0.174

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction
400	4	95	64	0	0	32.0	0.161
45	0	180	66	39	0	42.0	1.893

Columns with a value of "0"—such as glucose, blood pressure, skin thickness, insulin, and BMI—are likely to be inaccurate because a 0 for these characteristics is medically improbable. These values are probably missing or unreported. In order to indicate that they are missing, they will be replaced with NaN.

In [8]:
```
# show the number of rows and columns of the data frame
df.shape
```

Out[8]:
```
(768, 9)
```

In [9]:
```
# List the types of the columns
df.dtypes
```

Out[9]:
```
Pregnancies                 int64
Glucose                     int64
BloodPressure               int64
SkinThickness               int64
Insulin                     int64
BMI                       float64
DiabetesPedigreeFunction  float64
Age                         int64
Outcome                     int64
dtype: object
```

In [10]:
```
# check whether somewhere has no value in the data
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 [11]:
```
#Check Null values
df.isna().sum()
```

Out[11]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [12]:
```
#Check Duplicate values
df.duplicated().sum()
```

Out[12]:
```
0
```

In [13]:
```
# statistic
df.describe()
```

Out[13]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPec
count	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	
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	

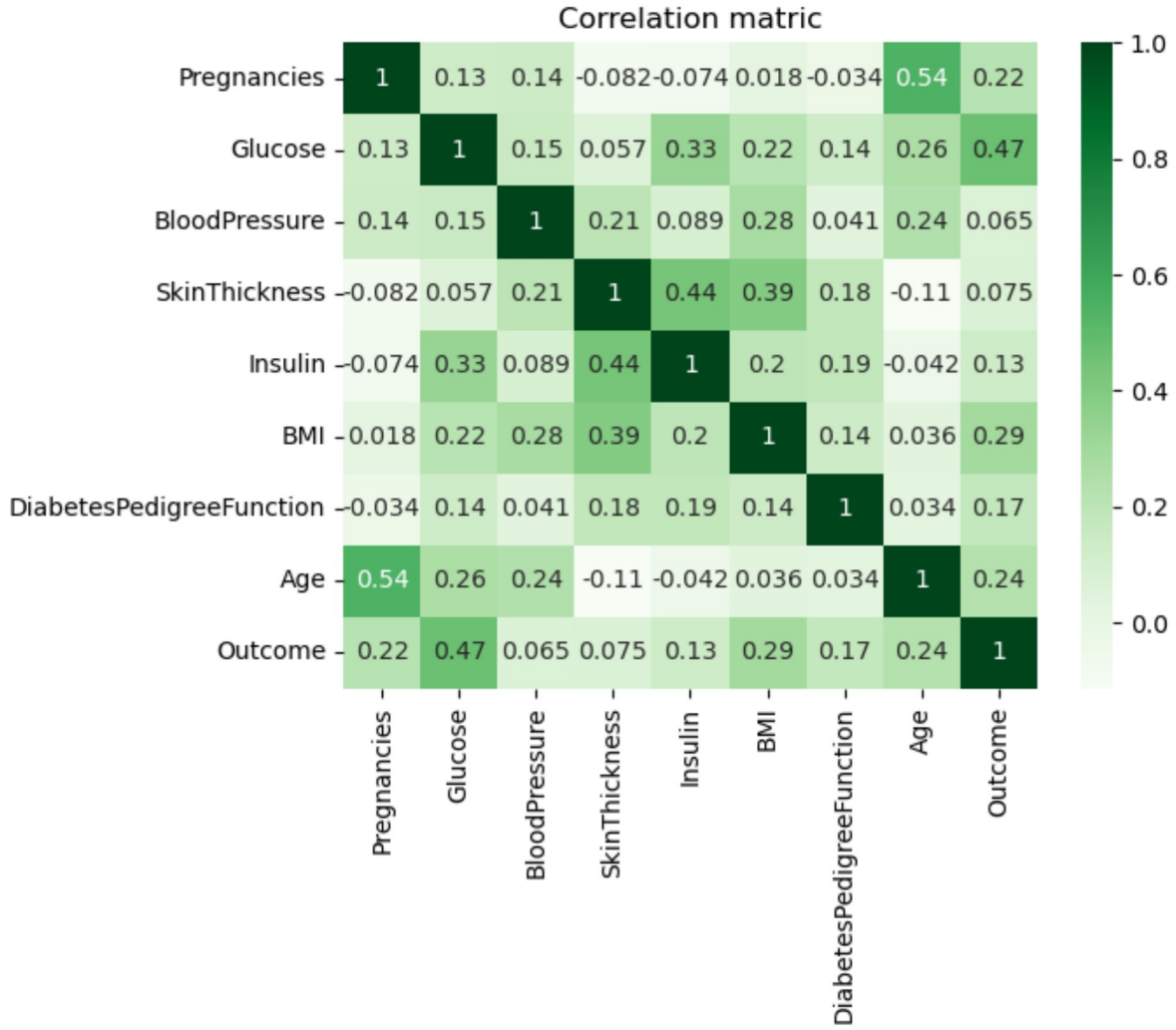
In [14]:
```
# correlation
df.corr()
```

Out[14]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BM
Pregnancies	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.01768
Glucose	0.129459	1.000000	0.152590	0.057328	0.331357	0.22107
BloodPressure	0.141282	0.152590	1.000000	0.207371	0.088933	0.28180
SkinThickness	-0.081672	0.057328	0.207371	1.000000	0.436783	0.39257
Insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.19785
BMI	0.017683	0.221071	0.281805	0.392573	0.197859	1.00000
DiabetesPedigreeFunction	-0.033523	0.137337	0.041265	0.183928	0.185071	0.14064
Age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.03624
Outcome	0.221898	0.466581	0.065068	0.074752	0.130548	0.29269

In [15]:
```
# Heatmap
sns.heatmap(df.corr(),annot = True, cmap = "Greens")
plt.title("Correlation matric")
```

Out[15]:
```
Text(0.5, 1.0, 'Correlation matric')
```



In [16]:
```
#Category the Age
rep = []
for row in df['Age']:
    if row < 31:
        rep.append('Young_age')
    elif row > 50:
        rep.append('Old_age')
    else:
        rep.append('Middle_age')
```

In [17]:
```
# Add a new column 'Categories' with the values are the list 'rep'
df['Categories'] = rep
df
```

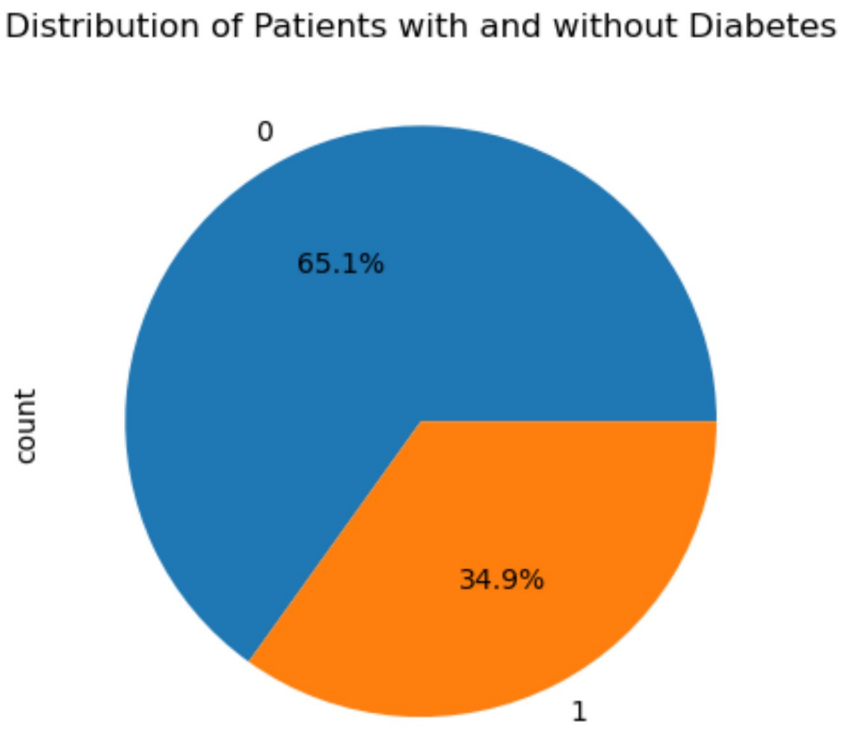
Out[17]:

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

768 rows × 10 columns

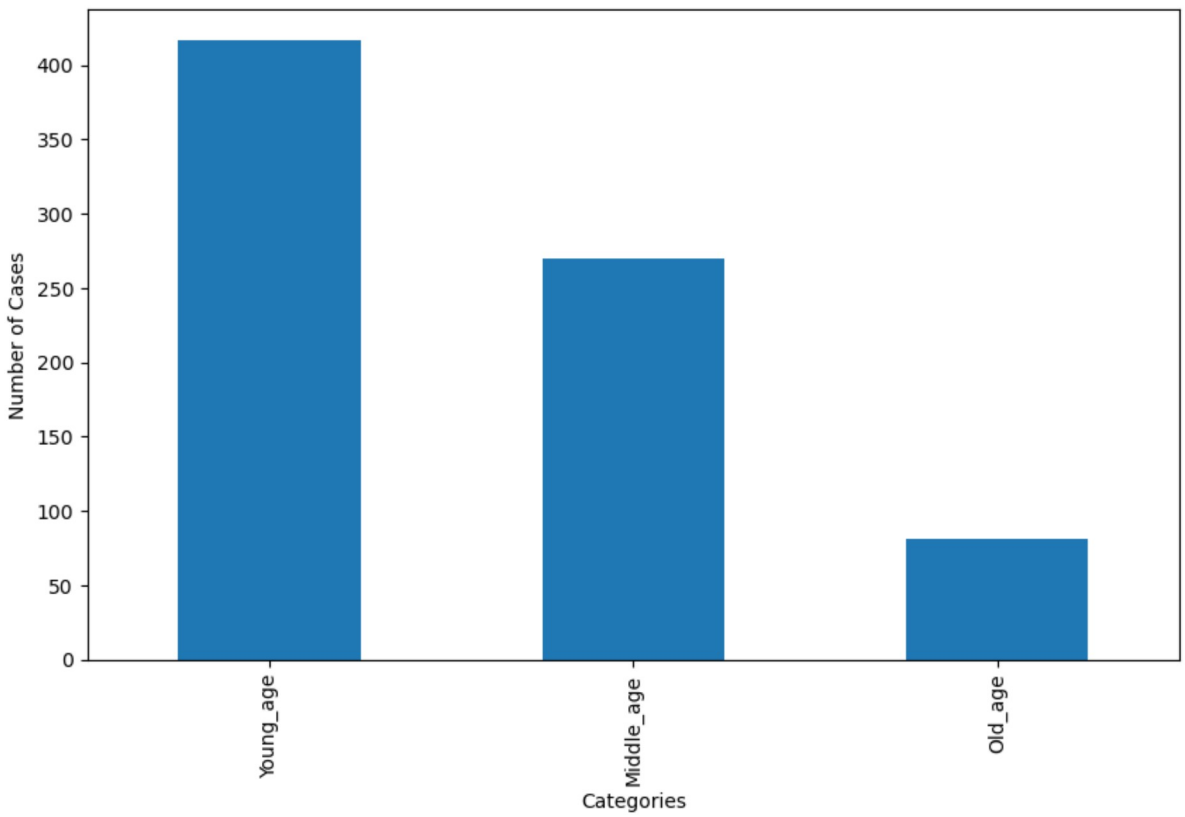
In [18]:
```
##Count Plot
# Outcome count plot

ax = df['Outcome'].value_counts().plot.pie(autopct='%1.1f%%',shadow=False)
ax.set_title('Distribution of Patients with and without Diabetes')

diabetes = df['Outcome'].value_counts()[1]
no_diabetes = df['Outcome'].value_counts()[0]

N,P = df['Outcome'].value_counts()
print('Negative (0): ',N , f'>>> There are {no_diabetes} numbers of patients withou
print('Positive (1): ',P , f'>>> There are {diabetes} numbers of patients with diab
```

```
Negative (0):  500 >>> There are 500 numbers of patients without diabetes.
Positive (1):  268 >>> There are 268 numbers of patients with diabetes.
```



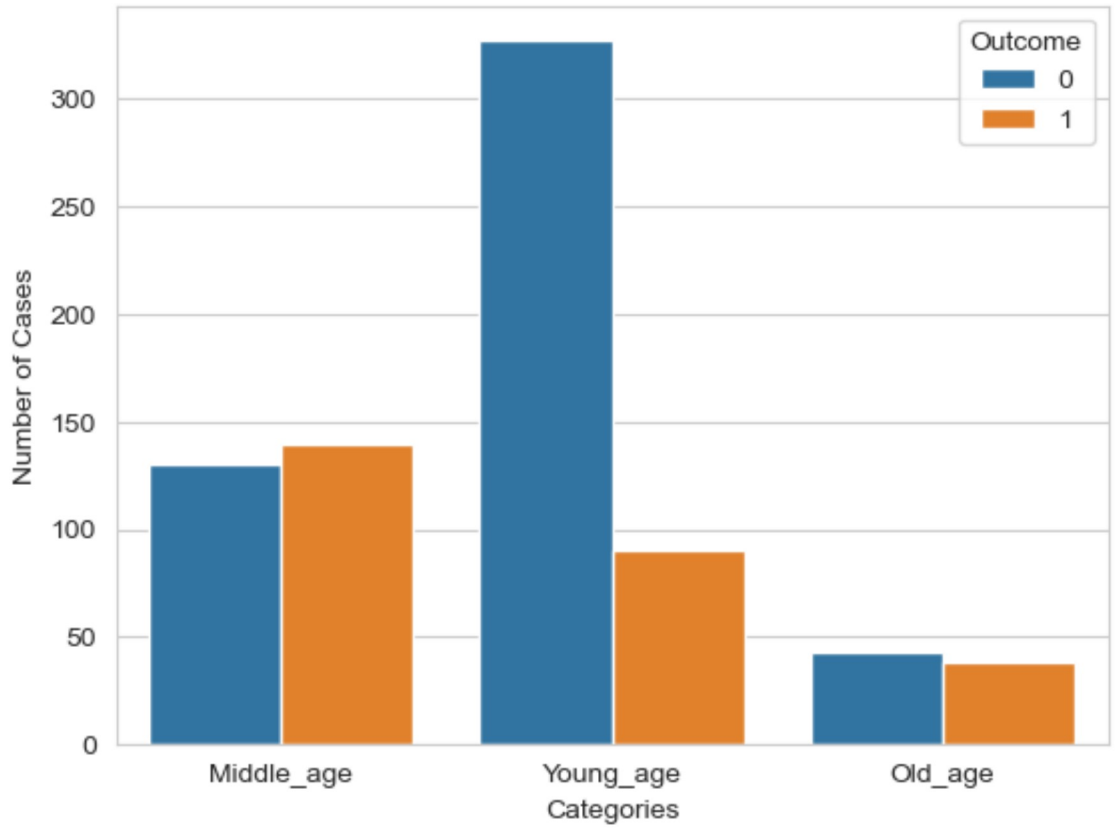
In [19]:
```
ax = df['Categories'].value_counts().plot(kind='bar', figsize=(10,6))
ax.set_ylabel('Number of Cases')
```

Out[19]:
```
Text(0, 0.5, 'Number of Cases')
```

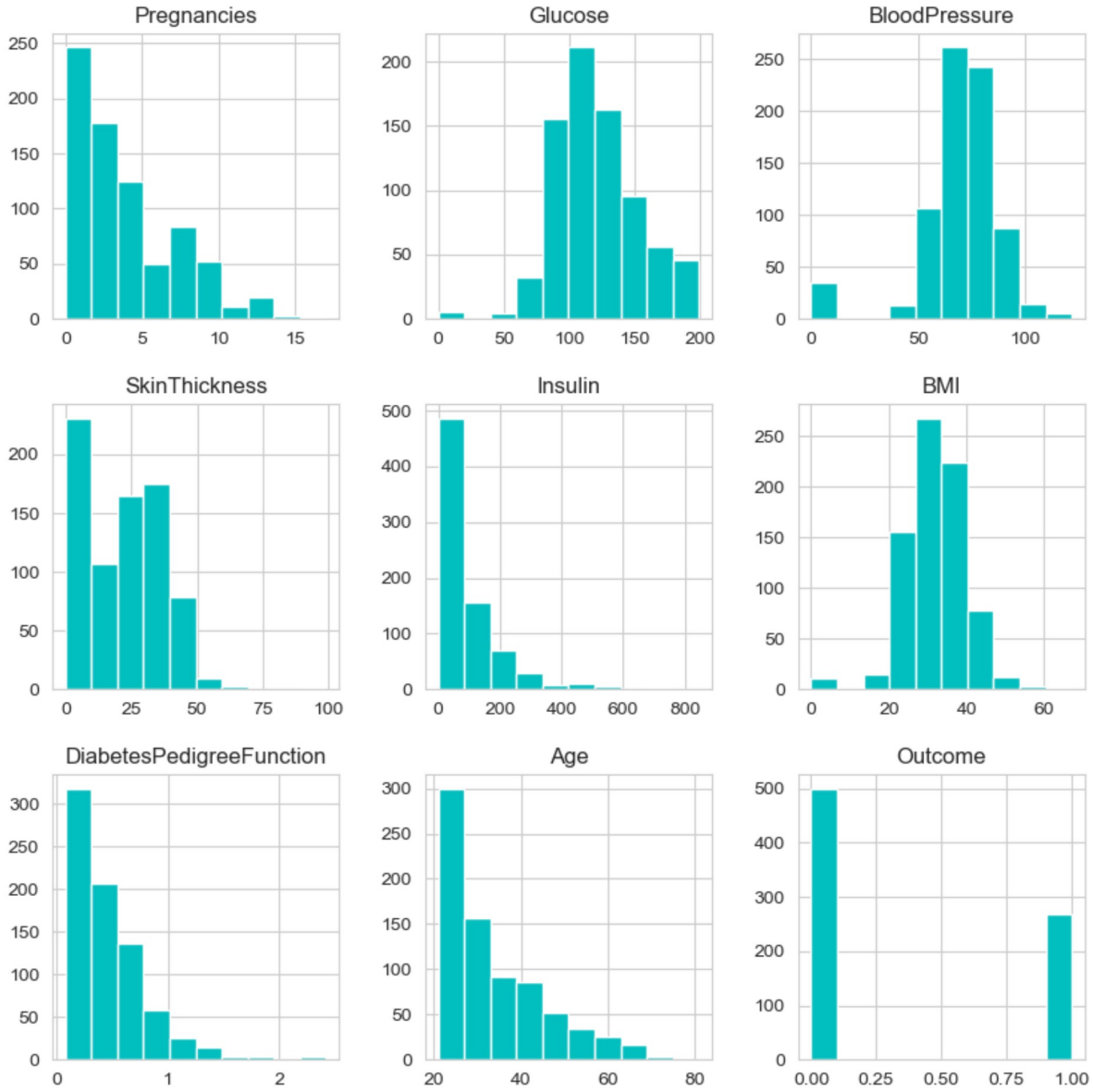


In [20]:
```
#split in group based on 'Outcome'
sns.set_style('whitegrid')
ax = sns.barplot(x='Categories', y = 'Age', hue = 'Outcome', data = df, estimator =
ax.set_ylabel('Number of Cases')
```

Out[20]:
```
Text(0, 0.5, 'Number of Cases')
```



In [21]:
```
# Histogram of each feature
df.hist(bins=10,figsize=(10,10),color='c')
plt.show()
```

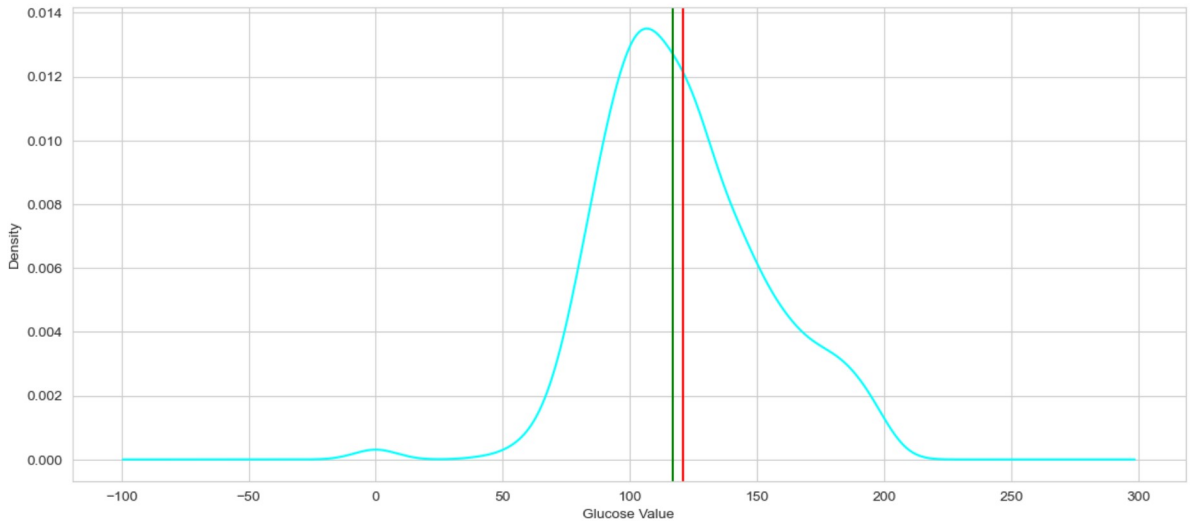


In [22]:
```
# Analyze the Glucose column
df['Glucose'].describe()
```

Out[22]:
```
count    768.000000
mean     120.894531
std       31.972618
min        0.000000
25%       99.000000
50%      117.000000
75%      140.250000
max      199.000000
Name: Glucose, dtype: float64
```

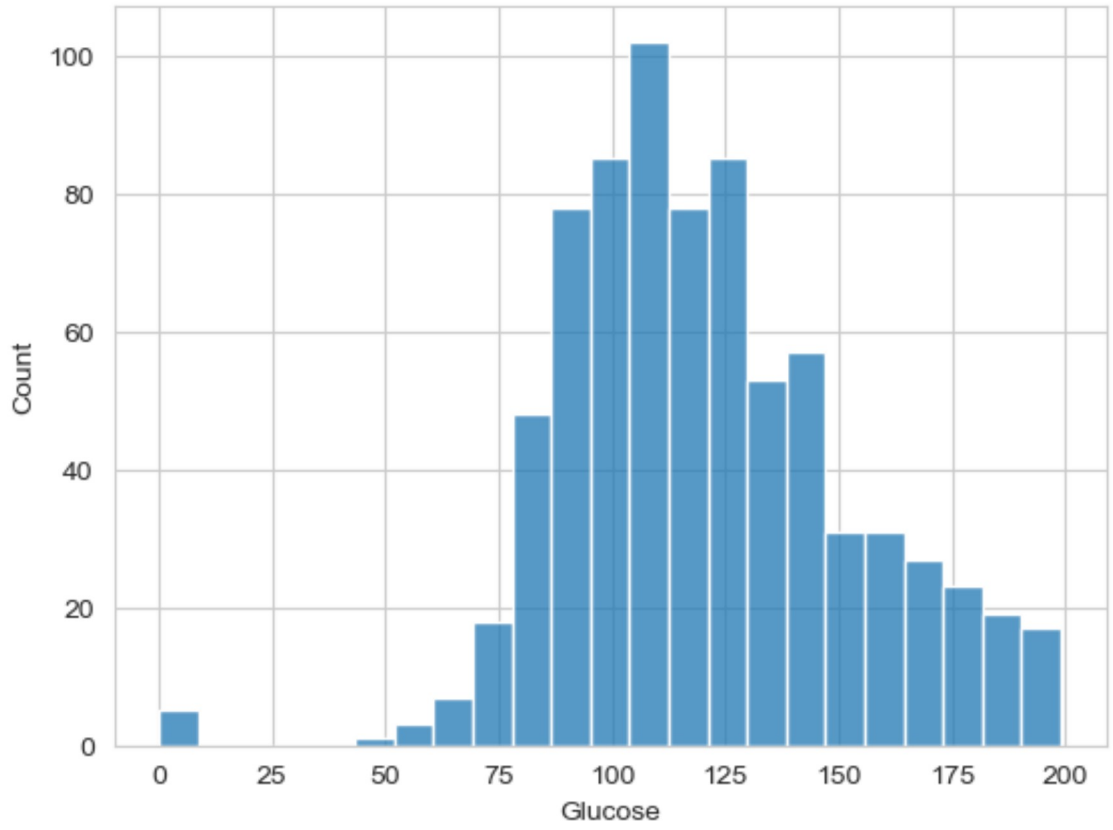
In [23]:
```
#
ax = df['Glucose'].plot(kind='density', figsize=(14,6), color ='cyan')
ax.axvline(df['Glucose'].mean(), color='red')
ax.axvline(df['Glucose'].median(), color='green')
ax.set_xlabel('Glucose Value')
```

Out[23]:
```
Text(0.5, 0, 'Glucose Value')
```



In [24]:
```
sns.histplot(df[('Glucose')])
```

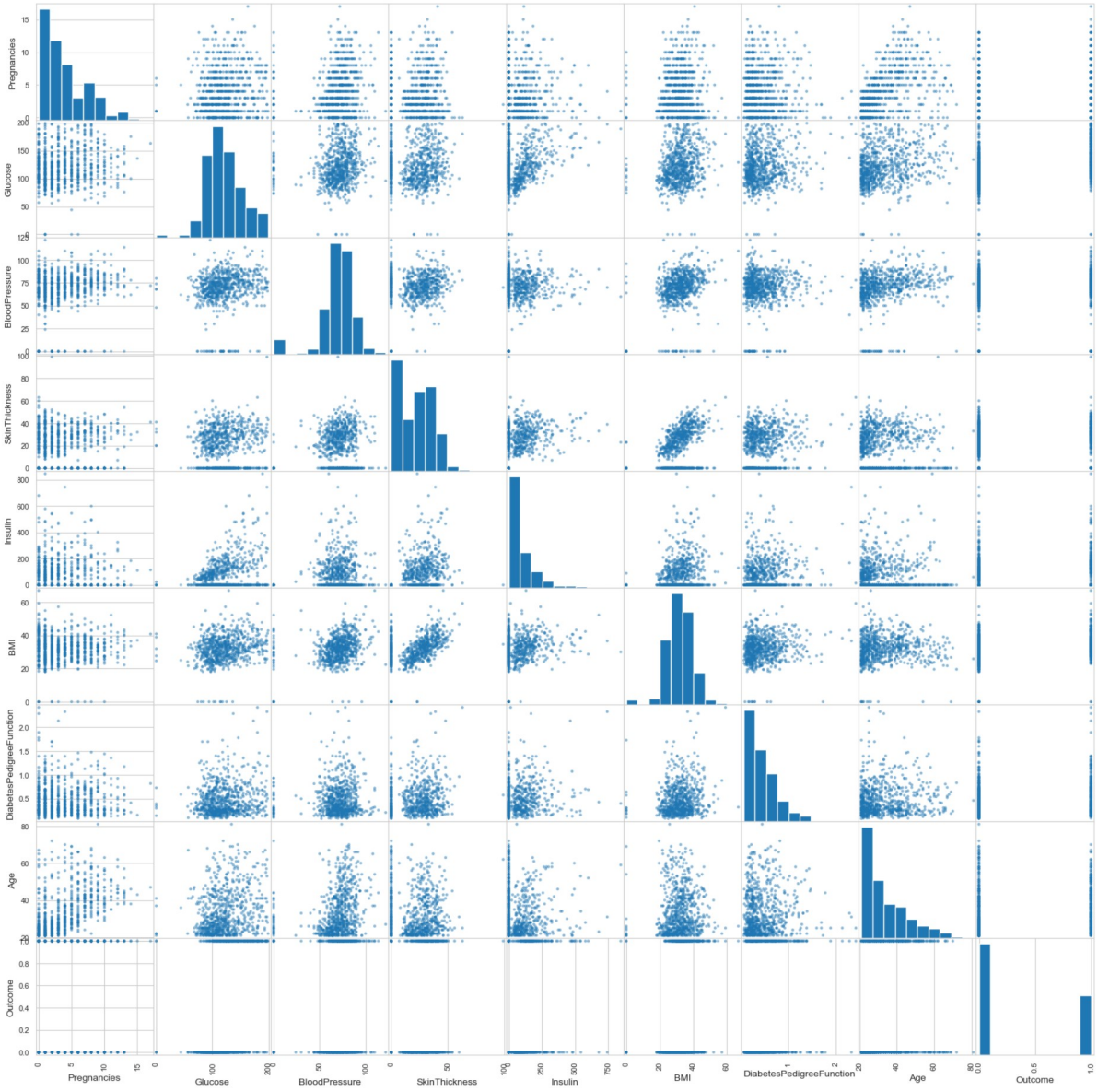
Out[24]:
```
<Axes: xlabel='Glucose', ylabel='Count'>
```



In [25]:
```
# Scatter plot matrix
from pandas.plotting import scatter_matrix
scatter_matrix(df, figsize = (20, 20))
```

Out[25]:
```
array([[<Axes: xlabel='Pregnancies', ylabel='Pregnancies'>,
        <Axes: xlabel='Glucose', ylabel='Pregnancies'>,
        <Axes: xlabel='BloodPressure', ylabel='Pregnancies'>,
        <Axes: xlabel='SkinThickness', ylabel='Pregnancies'>,
        <Axes: xlabel='Insulin', ylabel='Pregnancies'>,
        <Axes: xlabel='BMI', ylabel='Pregnancies'>,
        <Axes: xlabel='DiabetesPedigreeFunction', ylabel='Pregnancies'>,
        <Axes: xlabel='Age', ylabel='Pregnancies'>,
        <Axes: xlabel='Outcome', ylabel='Pregnancies'>],
       [<Axes: xlabel='Pregnancies', ylabel='Glucose'>,
        <Axes: xlabel='Glucose', ylabel='Glucose'>,
        <Axes: xlabel='BloodPressure', ylabel='Glucose'>,
        <Axes: xlabel='SkinThickness', ylabel='Glucose'>,
        <Axes: xlabel='Insulin', ylabel='Glucose'>,
        <Axes: xlabel='BMI', ylabel='Glucose'>,
        <Axes: xlabel='DiabetesPedigreeFunction', ylabel='Glucose'>,
        <Axes: xlabel='Age', ylabel='Glucose'>,
        <Axes: xlabel='Outcome', ylabel='Glucose'>],
       [<Axes: xlabel='Pregnancies', ylabel='BloodPressure'>,
        <Axes: xlabel='Glucose', ylabel='BloodPressure'>,
        <Axes: xlabel='BloodPressure', ylabel='BloodPressure'>,
        <Axes: xlabel='SkinThickness', ylabel='BloodPressure'>,
        <Axes: xlabel='Insulin', ylabel='BloodPressure'>,
        <Axes: xlabel='BMI', ylabel='BloodPressure'>,
        <Axes: xlabel='DiabetesPedigreeFunction', ylabel='BloodPressure'>,
        <Axes: xlabel='Age', ylabel='BloodPressure'>,
        <Axes: xlabel='Outcome', ylabel='BloodPressure'>],
       [<Axes: xlabel='Pregnancies', ylabel='SkinThickness'>,
        <Axes: xlabel='Glucose', ylabel='SkinThickness'>,
        <Axes: xlabel='BloodPressure', ylabel='SkinThickness'>,
        <Axes: xlabel='SkinThickness', ylabel='SkinThickness'>,
        <Axes: xlabel='Insulin', ylabel='SkinThickness'>,
        <Axes: xlabel='BMI', ylabel='SkinThickness'>,
        <Axes: xlabel='DiabetesPedigreeFunction', ylabel='SkinThickness'>,
        <Axes: xlabel='Age', ylabel='SkinThickness'>,
        <Axes: xlabel='Outcome', ylabel='SkinThickness'>],
       [<Axes: xlabel='Pregnancies', ylabel='Insulin'>,
        <Axes: xlabel='Glucose', ylabel='Insulin'>,
        <Axes: xlabel='BloodPressure', ylabel='Insulin'>,
        <Axes: xlabel='SkinThickness', ylabel='Insulin'>,
        <Axes: xlabel='Insulin', ylabel='Insulin'>,
        <Axes: xlabel='BMI', ylabel='Insulin'>,
        <Axes: xlabel='DiabetesPedigreeFunction', ylabel='Insulin'>,
        <Axes: xlabel='Age', ylabel='Insulin'>,
        <Axes: xlabel='Outcome', ylabel='Insulin'>],
       [<Axes: xlabel='Pregnancies', ylabel='BMI'>,
        <Axes: xlabel='Glucose', ylabel='BMI'>,
        <Axes: xlabel='BloodPressure', ylabel='BMI'>,
        <Axes: xlabel='SkinThickness', ylabel='BMI'>,
        <Axes: xlabel='Insulin', ylabel='BMI'>,
        <Axes: xlabel='BMI', ylabel='BMI'>,
        <Axes: xlabel='DiabetesPedigreeFunction', ylabel='BMI'>,
        <Axes: xlabel='Age', ylabel='BMI'>,
        <Axes: xlabel='Outcome', ylabel='BMI'>],
       [<Axes: xlabel='Pregnancies', ylabel='DiabetesPedigreeFunction'>,
        <Axes: xlabel='Glucose', ylabel='DiabetesPedigreeFunction'>,
        <Axes: xlabel='BloodPressure', ylabel='DiabetesPedigreeFunction'>,
        <Axes: xlabel='SkinThickness', ylabel='DiabetesPedigreeFunction'>,
        <Axes: xlabel='Insulin', ylabel='DiabetesPedigreeFunction'>,
```

```
        <Axes: xlabel='BMI', ylabel='DiabetesPedigreeFunction'>,
        <Axes: xlabel='DiabetesPedigreeFunction', ylabel='DiabetesPedigreeFunction'
>,
        <Axes: xlabel='Age', ylabel='DiabetesPedigreeFunction'>,
        <Axes: xlabel='Outcome', ylabel='DiabetesPedigreeFunction'>],
       [<Axes: xlabel='Pregnancies', ylabel='Age'>,
        <Axes: xlabel='Glucose', ylabel='Age'>,
        <Axes: xlabel='BloodPressure', ylabel='Age'>,
        <Axes: xlabel='SkinThickness', ylabel='Age'>,
        <Axes: xlabel='Insulin', ylabel='Age'>,
        <Axes: xlabel='BMI', ylabel='Age'>,
        <Axes: xlabel='DiabetesPedigreeFunction', ylabel='Age'>,
        <Axes: xlabel='Age', ylabel='Age'>,
        <Axes: xlabel='Outcome', ylabel='Age'>],
       [<Axes: xlabel='Pregnancies', ylabel='Outcome'>,
        <Axes: xlabel='Glucose', ylabel='Outcome'>,
        <Axes: xlabel='BloodPressure', ylabel='Outcome'>,
        <Axes: xlabel='SkinThickness', ylabel='Outcome'>,
        <Axes: xlabel='Insulin', ylabel='Outcome'>,
        <Axes: xlabel='BMI', ylabel='Outcome'>,
        <Axes: xlabel='DiabetesPedigreeFunction', ylabel='Outcome'>,
        <Axes: xlabel='Age', ylabel='Outcome'>,
        <Axes: xlabel='Outcome', ylabel='Outcome'>]], dtype=object)
```



In [26]:
```
# Replace the 0 with NaN
df[['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI']] = (
df[['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI']].replace(0, np.n
```

In [27]:
```
#
df.isnull().sum()
```

Out[27]:
```
Pregnancies                 0
Glucose                     5
BloodPressure              35
SkinThickness             227
Insulin                   374
BMI                        11
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
Categories                  0
dtype: int64
```