Pandas task

```
import pandas as pd
```

```
data =  pd.read_csv("https://raw.githubusercontent.com/RAHULJOGI-CODE/ML-TASKS/main/ML%20ASSETS/diabetes.csv")

# Displaying the first few rows of the dataset
print(data.head())
```

```
   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
```

```
#Displaying first 10 data points
data.head(10)
```

	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
5	5	116	74	0	0	25.6	0.201	30	0
6	3	78	50	32	88	31.0	0.248	26	1
7	10	115	0	0	0	35.3	0.134	29	0
8	2	197	70	45	543	30.5	0.158	53	1
9	8	125	96	0	0	0.0	0.232	54	1

```
#Displaying last 10 data points
data.tail(10)
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
758	1	106	76	0	0	37.5	0.197	26	0
759	6	190	92	0	0	35.5	0.278	66	1
760	2	88	58	26	16	28.4	0.766	22	0
761	9	170	74	31	0	44.0	0.403	43	1
762	9	89	62	0	0	22.5	0.142	33	0
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

```
len(data)
```

```
768
```

```
#dimension of data
data.shape
```

```
(768, 9)
```

```
(768, 9)
```

```
#determinig types of data included in the dataset
data.dtypes
```

```
Pregnancies                 int64
Glucose                     int64
BloodPressure               int64
SkinThickness               int64
Insulin                     int64
BMI                       float64
DiabetesPedigreeFunction  float64
Age                         int64
Outcome                     int64
dtype: object
```

```
#knowing the mean & s.d of each feature involved
data.describe()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Ou
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.0
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.3
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.4
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.0
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.0
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.0
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.0
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.0

From the above described data we can see the various factors are impacting the samples diagnosed with diabetes

```
#knowing about null values present in each feature
data.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
```

```
# To check wether null values present or not
data.isnull().sum()
```

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

```
#we are going to drop the null values row
data.dropna(inplace=True)
```

```
#As null values are not present, in order to handle them we can find mean of each feature and replace the null value with mea
data["Glucose"].fillna(data["Glucose"].mean(), inplace=True)
data["BloodPressure"].fillna(data["BloodPressure"].mean(), inplace=True)
data["SkinThickness"].fillna(data["SkinThickness"].median(), inplace=True)
data["Insulin"].fillna(data["Insulin"].median(), inplace=True)
data["BMI"].fillna(data["BMI"].mean(), inplace=True)
```

```
data.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
```

```
#we are treating X as input and Y as Output...which says diabetes is present or not
X = data[['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI','DiabetesPedigreeFunction','Age']]
X
```

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

768 rows × 8 columns

```
Y =data[['Outcome']]
Y
```

	Outcome
0	1
1	0
2	1
3	0
4	1
...	...
763	0
764	0
765	0
766	1
767	0

768 rows × 1 columns

```
data.Age.describe()
```

```
count    768.000000
mean      33.240885
std       11.760232
min       21.000000
25%       24.000000
50%       29.000000
75%       41.000000
max       81.000000
Name: Age, dtype: float64
```

```
data.corr()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	
Pregnancies	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	-0.033523	0.5
Glucose	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	0.137337	0.2
BloodPressure	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	0.041265	0.2
SkinThickness	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.183928	-0.1
Insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	0.185071	-0.0
BMI	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	0.140647	0.0
DiabetesPedigreeFunction	-0.033523	0.137337	0.041265	0.183928	0.185071	0.140647	1.000000	0.0
Age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	0.033561	1.0
Outcome	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	0.173844	0.2

Analysing Glucose

```
mean_glucose_diabetic = data[data["Outcome"] == 1]["Glucose"].mean()
mean_glucose_non_diabetic = data[data["Outcome"] == 0]["Glucose"].mean()

print("Mean Glucose Level:")
print("Diabetic Patients:", mean_glucose_diabetic)
print("Non-Diabetic Patients:", mean_glucose_non_diabetic)
```

```
Mean Glucose Level:
Diabetic Patients: 141.25746268656715
Non-Diabetic Patients: 109.98
```

We are anlaysing the age factors for diabetes

```
average_age_diabetic = data[data["Outcome"] == 1]["Age"].mean()
average_age_non_diabetic = data[data["Outcome"] == 0]["Age"].mean()

print("Average Age:")
print("Diabetic Patients:", average_age_diabetic)
print("Non-Diabetic Patients:", average_age_non_diabetic)
```

```
Average Age:
Diabetic Patients: 37.06716417910448
Non-Diabetic Patients: 31.19
```

Total Diagnosed

```
percentage_diabetic = (data["Outcome"].sum() / data["Outcome"].count()) * 100
print("\nPercentage of Diabetic Patients in the Dataset:", percentage_diabetic, "%")
```

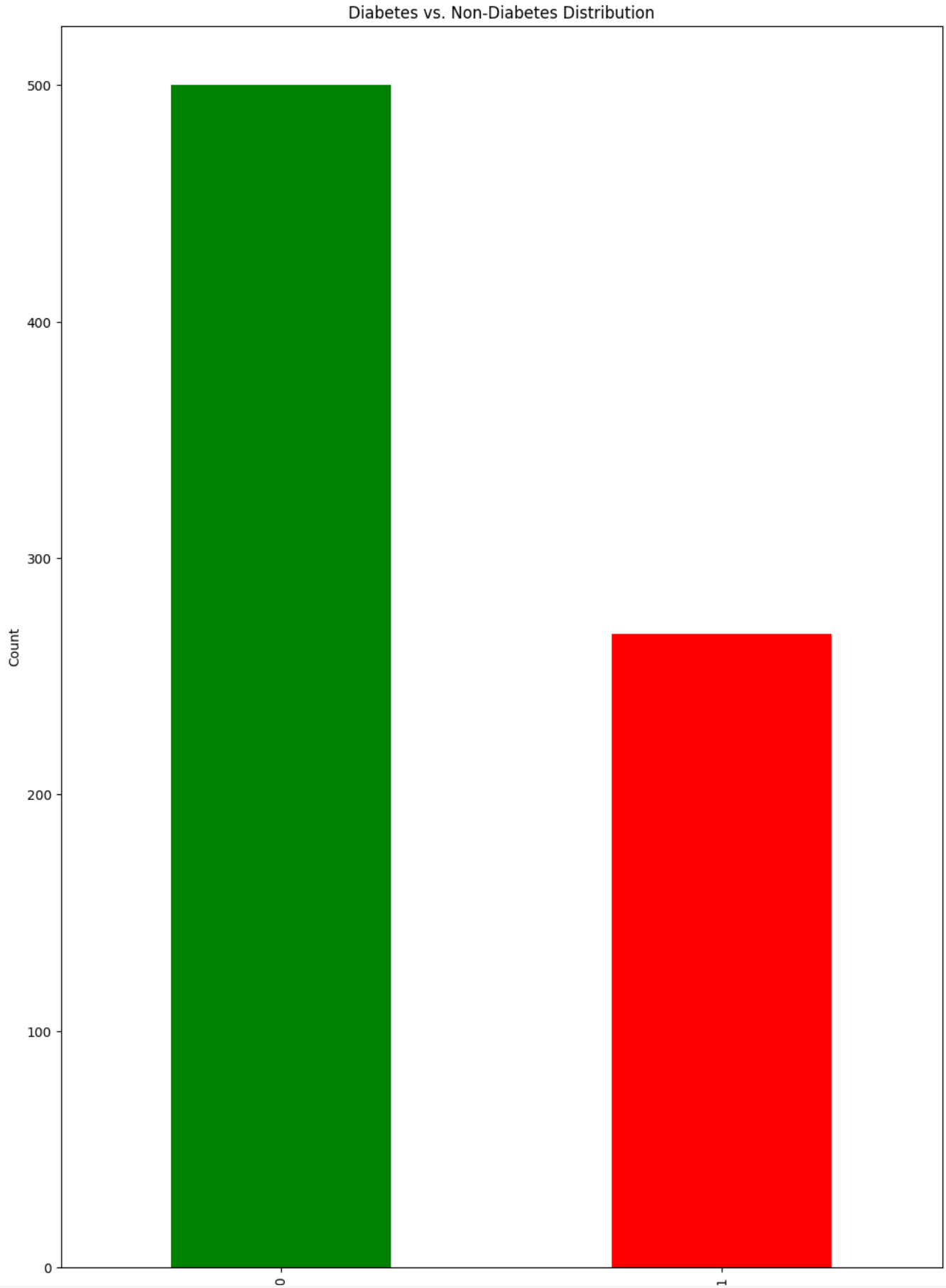
```
Percentage of Diabetic Patients in the Dataset: 34.89583333333333 %
```

PLOTTING THE DATA

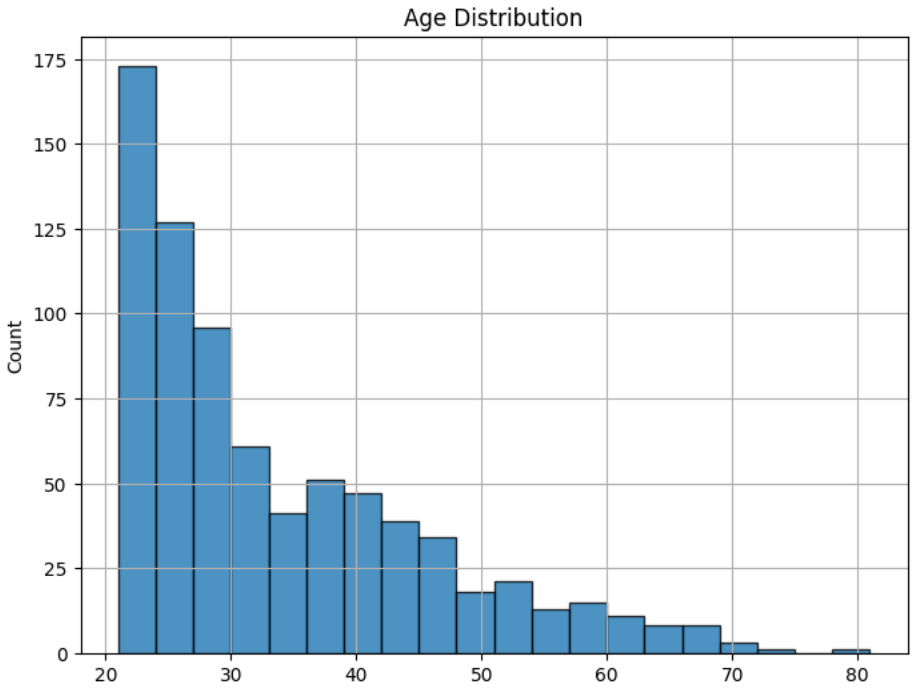
```
import matplotlib.pyplot as plt
```

```
#green representing----No Diabetes
#red representing----- Diabetes

plt.figure(figsize=(12,17))
data['Outcome'].value_counts().plot(kind='bar', color=['green', 'red'])
plt.xlabel('Outcome (0: No Diabetes, 1: Diabetes)')
plt.ylabel('Count')
plt.title('Diabetes vs. Non-Diabetes Distribution')
plt.show()
```




```
#Age Distribution
plt.figure(figsize=(8, 6))
data['Age'].hist(edgecolor='black', bins=20, alpha=0.8)
plt.xlabel('Age')
plt.ylabel('Count')
plt.title('Age Distribution')
plt.show()
```



Chances of being diagnosed with diabetes with Pregnancies

```
#Chances of being diagnosed with diabetes with Pregnancies
plt.figure(figsize=(8, 6))
data.groupby('Pregnancies')['Outcome'].mean().plot(kind='bar')
plt.xlabel('Number of Pregnancies')
plt.ylabel('Diabetes Probability')
plt.title('Diabetes Probability based on Number of Pregnancies')
plt.show()
```