Found Some insights by using heart diseases datasets

Import Libraries in python

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Pulling the dataset into python framework

```
hr = pd.read_csv("/content/heart.csv")
```

Dataset Summary

Starting Rows of dataset

```
hr.head(10)
```

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngin
0	40	M	ATA	140	289	0	Normal	172	
1	49	F	NAP	160	180	0	Normal	156	
2	37	M	ATA	130	283	0	ST	98	
3	48	F	ASY	138	214	0	Normal	108	
4	54	M	NAP	150	195	0	Normal	122	
5	39	M	NAP	120	339	0	Normal	170	
6	45	F	ATA	130	237	0	Normal	170	
7	54	M	ATA	110	208	0	Normal	142	
8	37	M	ASY	140	207	0	Normal	130	
9	48	F	ATA	120	284	0	Normal	120	

Ending Rows of dataset

```
hr.tail(10)
```

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAng
908	63	M	ASY	140	187	0	LVH	144	
909	63	F	ASY	124	197	0	Normal	136	
910	41	M	ATA	120	157	0	Normal	182	
911	59	M	ASY	164	176	1	LVH	90	
912	57	F	ASY	140	241	0	Normal	123	
913	45	M	TA	110	264	0	Normal	132	
914	68	M	ASY	144	193	1	Normal	141	
915	57	M	ASY	130	131	0	Normal	115	
916	57	F	ATA	130	236	0	LVH	174	
917	38	M	NAP	138	175	0	Normal	173	

Basic Information

```
hr.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 15 columns):
```

```
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Age             918 non-null    int64
 1   Sex             918 non-null    object
 2   ChestPainType   918 non-null    object
 3   RestingBP       918 non-null    int64
 4   Cholesterol     918 non-null    int64
 5   FastingBS       918 non-null    int64
 6   RestingECG      918 non-null    object
 7   MaxHR           918 non-null    int64
 8   ExerciseAngina  918 non-null    object
 9   Oldpeak         918 non-null    float64
 10  ST_Slope        918 non-null    object
 11  HeartDisease    918 non-null    int64
 12  Heart_Status    918 non-null    object
 13  Age_Status      918 non-null    category
 14  Age_Bracket     918 non-null    category
dtypes: category(2), float64(1), int64(6), object(6)
memory usage: 95.6+ KB
```

Finding Missing Value

```
hr.isnull().sum()
```

```
Age               0
Sex               0
ChestPainType     0
RestingBP         0
Cholesterol       0
FastingBS         0
RestingECG        0
MaxHR             0
ExerciseAngina    0
Oldpeak           0
ST_Slope          0
HeartDisease      0
dtype: int64
```

Missing values are not found

Finding "n" rows

```
hr2 = pd.read_csv("/content/heart.csv" , nrows = 6)
hr2
```

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngin
0	40	M	ATA	140	289	0	Normal	172	N
1	49	F	NAP	160	180	0	Normal	156	N
2	37	M	ATA	130	283	0	ST	98	N
3	48	F	ASY	138	214	0	Normal	108	Y
4	54	M	NAP	150	195	0	Normal	122	N
5	39	M	NAP	120	339	0	Normal	170	N

```
# we can find any rows between the dataset by using this formula
hr[5:13]
```

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngi
5	39	M	NAP	120	339	0	Normal	170	
6	45	F	ATA	130	237	0	Normal	170	
7	54	M	ATA	110	208	0	Normal	142	
8	37	M	ASY	140	207	0	Normal	130	
9	48	F	ATA	120	284	0	Normal	120	
10	37	F	NAP	130	211	0	Normal	142	
11	58	M	ATA	136	164	0	ST	99	
12	39	M	ATA	120	204	0	Normal	145	

Finding specific column

```
hr2 = pd.read_csv("/content/heart.csv" , usecols = [1,5])
hr2
```

	Sex	FastingBS
0	M	0
1	F	0
2	M	0
3	F	0
4	M	0
...	...	...
913	M	0
914	M	1
915	M	0
916	F	0
917	M	0

918 rows × 2 columns

```
# we can find specific column by another way

hr[["ChestPainType"]]
```

	ChestPainType
0	ATA
1	NAP
2	ATA
3	ASY
4	NAP
...	...
913	TA
914	ASY
915	ASY
916	ATA
917	NAP

918 rows × 1 columns

Adding Essential columns

```
# new column will add named age_bracket which will denote age status
Age_range = [0,18,30,60,100]
labels = ["young", "Adult", 'Old', 'Very Old']
label = ["0-18" , "18-30", "30-60" , "60-100"]

hr["Age_Status"] = pd.cut(hr["Age"], bins = Age_range, labels = labels)
hr["Age_Bracket"] = pd.cut(hr["Age"], bins = Age_range, labels = label)
```

```
# Heart Status column tell us about the status of heart diseases either it will be normal or heat disease.

heart_status = {0: "Normal", 1: "Heart Disease"}

hr["Heart_Status"] = hr["HeartDisease"].map(heart_status)
```

```
hr.head(100)
```

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease	Heart_Status
0	40	M	ATA	140	289	0	Normal	172	N	0.0	Up	0	Normal
1	49	F	NAP	160	180	0	Normal	156	N	1.0	Flat	1	Heart Disease
2	37	M	ATA	130	283	0	ST	98	N	0.0	Up	0	Normal
3	48	F	ASY	138	214	0	Normal	108	Y	1.5	Flat	1	Heart Disease
4	54	M	NAP	150	195	0	Normal	122	N	0.0	Up	0	Normal
...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	58	M	ASY	130	263	0	Normal	140	Y	2.0	Flat	1	Heart Disease
96	43	M	ATA	142	207	0	Normal	138	N	0.0	Up	0	Normal
97	39	M	NAP	160	147	1	Normal	160	N	0.0	Up	0	Normal
98	56	M	ASY	120	85	0	Normal	140	N	0.0	Up	0	Normal
99	41	M	ATA	125	269	0	Normal	144	N	0.0	Up	0	Normal

100 rows × 13 columns

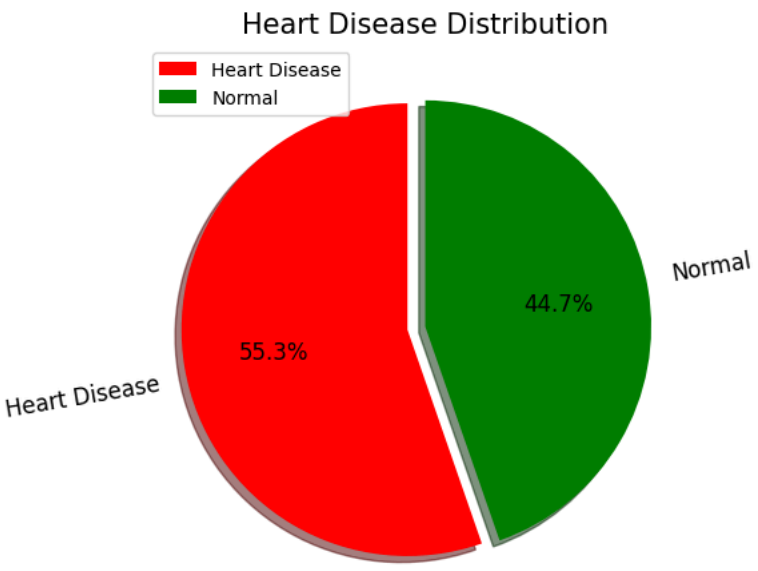
Basic Statistical Analysis

```
hr.describe()
```

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease
count	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000
mean	53.510893	132.396514	198.799564	0.233115	136.809368	0.887364	0.553377
std	9.432617	18.514154	109.384145	0.423046	25.460334	1.066570	0.497414
min	28.000000	0.000000	0.000000	0.000000	60.000000	-2.600000	0.000000
25%	47.000000	120.000000	173.250000	0.000000	120.000000	0.000000	0.000000
50%	54.000000	130.000000	223.000000	0.000000	138.000000	0.600000	1.000000
75%	60.000000	140.000000	267.000000	0.000000	156.000000	1.500000	1.000000
max	77.000000	200.000000	603.000000	1.000000	202.000000	6.200000	1.000000

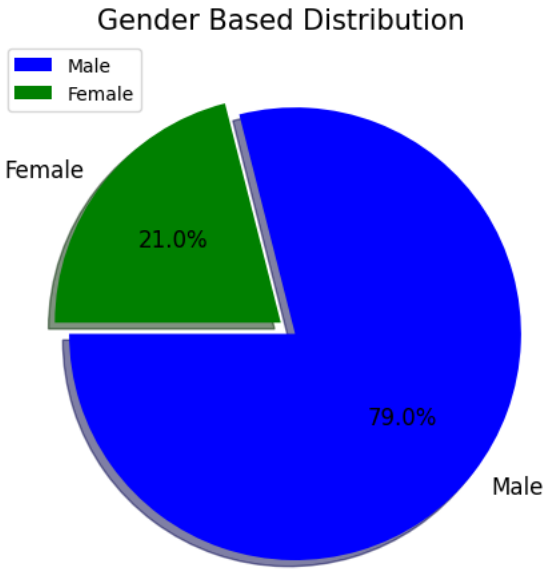
Data visualization by using Charts

```
Frequency = hr["Heart_Status"].value_counts()
labels =["Heart Disease" , "Normal"]
ex = [0.08, 0.0]
color = ['r','g']
plt.figure(figsize=(5.5,5.5))
plt.pie(Frequency, labels = labels, explode = ex, colors = color,
        autopct = "%.1f%%", shadow = True , startangle = 90,
        textprops = {"fontsize" : 12}, rotatelabels = True
        )
plt.title("Heart Disease Distribution", fontsize = 15)
plt.legend(loc ="upper left")
plt.show()
```



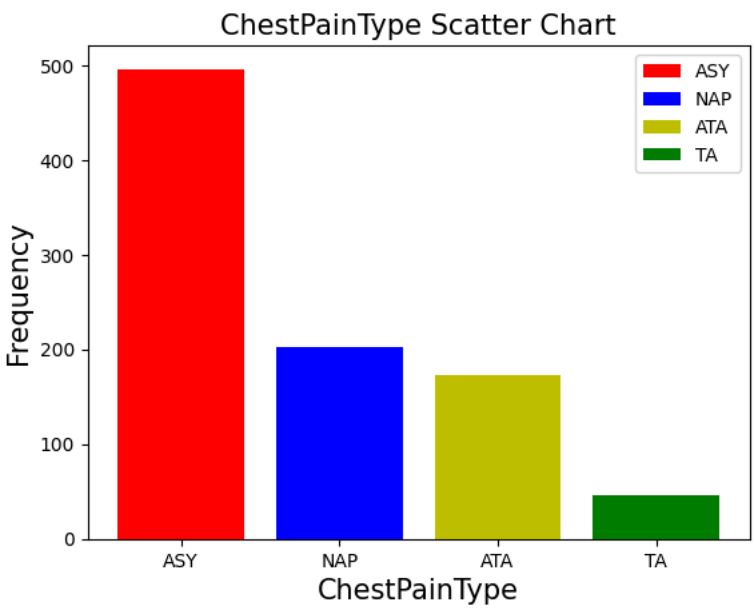
More than half of people have heart disease

```
frequency = hr["Sex"].value_counts()
labels = ["Male", "Female"]
ex = [0.08, 0.0]
color =['b','g']
plt.figure(figsize=(5.5,5.5))
plt.pie(frequency, labels = labels, explode = ex, colors = color,
        autopct = "%.1f%%", shadow = True , startangle = 180,
        textprops = {"fontsize" : 12},
        )
plt.title("Gender Based Distribution", fontsize = 15)
plt.legend(loc ="upper left")
plt.show()
```



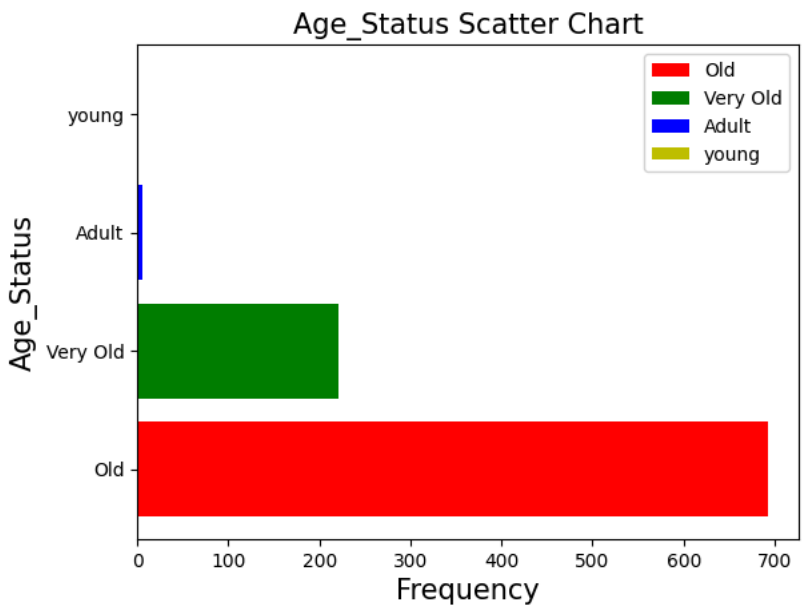
79% Male which is more than female in heart disease process

```
Frequency = hr["ChestPainType"].value_counts().values
ChestPainType = hr["ChestPainType"].value_counts().index
c = ['r', 'b', 'y', 'g']
plt.bar(ChestPainType, Frequency, color= c, label =ChestPainType)

plt.title('ChestPainType Scatter Chart', fontsize = 15)
plt.xlabel('ChestPainType', fontsize = 15)
plt.ylabel('Frequency', fontsize = 15)

plt.legend()

plt.show()
```

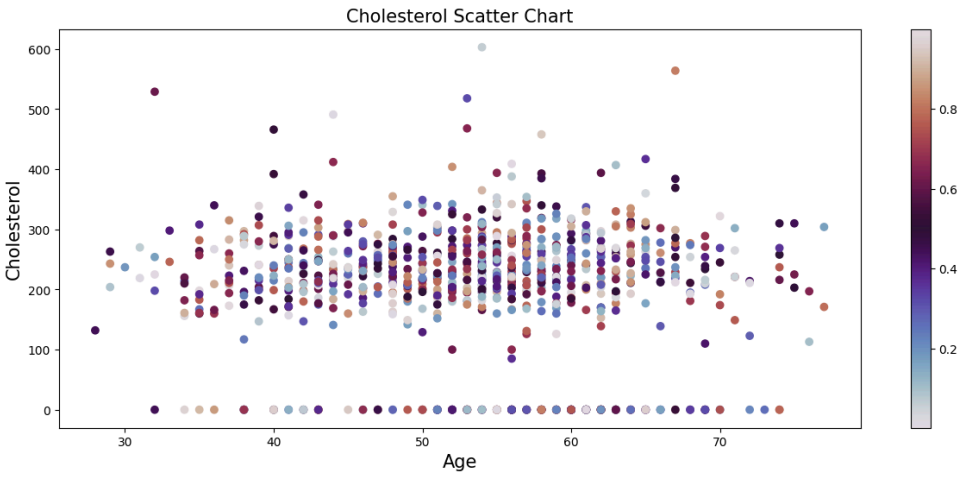


ASY(Asymptomatic) type chestpain frequency are much highest than rest of them.

```
Frequency = hr["Age_Status"].value_counts().values
Age_Bracket = hr["Age_Status"].value_counts().index

c = ['r', 'g', 'b', 'y']

plt.barh(Age_Bracket, Frequency, color = c , label = Age_Bracket )

plt.title('Age_Status Scatter Chart', fontsize = 15)
plt.ylabel('Age_Status', fontsize = 15)
plt.xlabel('Frequency', fontsize = 15)

plt.legend()

plt.show()
```




```
ages = hr['Age']
cholesterol = hr['Cholesterol']

colors = np.random.rand(len(ages))
plt.figure(figsize=(15, 6))
plt.scatter(ages, cholesterol, c = colors , cmap= "twilight")

plt.title('Cholesterol Scatter Chart',fontsize = 15)
plt.xlabel('Age',fontsize = 15)
plt.ylabel('Cholesterol', fontsize = 15)
plt.colorbar()

plt.show()
```



Most of the people lies under 400 mm/dl cholesterol