Loading The Dataset

```
import pandas as pd
ds=pd.read_excel('/content/drive/MyDrive/DoctorVisits (2).xlsx')
df=ds
```

Exploring the Dataset

```
ds.head()   #used to retrieve first 5 columns in the dataset
```

	Unnamed: 0	visits	gender	age	income	illness	reduced	health	private	freepoor	freerepat	nchro
0	1	1	female	0.19	0.55	1	4	1	yes	no	no	
1	2	1	female	0.19	0.45	1	2	1	yes	no	no	
2	3	1	male	0.19	0.90	3	0	0	no	no	no	
3	4	1	male	0.19	0.15	1	0	0	no	no	no	
4	5	1	male	0.19	0.45	2	5	1	no	no	no	

```
ds.tail()     #used to retrieve last 5 columns in the dataset
```

	Unnamed: 0	visits	gender	age	income	illness	reduced	health	private	freepoor	freerepat	nc
5185	5186	0	female	0.22	0.55	0	0	0	no	no	no	
5186	5187	0	male	0.27	1.30	0	0	1	no	no	no	
5187	5188	0	female	0.37	0.25	1	0	1	no	no	yes	
5188	5189	0	female	0.52	0.65	0	0	0	no	no	no	
5189	5190	0	male	0.72	0.25	0	0	0	no	no	yes	

```
ds.describe()   #describe method is used to find statistics of the dataset
```

	Unnamed: 0	visits	age	income	illness	reduced	health
count	5190.000000	5190.000000	5190.000000	5190.000000	5190.000000	5190.000000	5190.000000
mean	2595.500000	0.301734	0.406385	0.583160	1.431985	0.861850	1.217534
std	1498.368279	0.798134	0.204782	0.368907	1.384152	2.887628	2.124266
min	1.000000	0.000000	0.190000	0.000000	0.000000	0.000000	0.000000
25%	1298.250000	0.000000	0.220000	0.250000	0.000000	0.000000	0.000000
50%	2595.500000	0.000000	0.320000	0.550000	1.000000	0.000000	0.000000
75%	3892.750000	0.000000	0.620000	0.900000	2.000000	0.000000	2.000000
max	5190.000000	9.000000	0.720000	1.500000	5.000000	14.000000	12.000000

```
ds.dtypes   #dtypes method is used to find the datatypes of all the columns.
```

```
Unnamed: 0      int64
visits          int64
gender         object
age           float64
income        float64
illness         int64
reduced         int64
health          int64
private        object
freepoor       object
freerepat      object
nchronic       object
lchronic       object
dtype: object
```

```
ds.info()    #info method is used to find the count of null values in each column along with the column name and its respective datatype.
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5190 entries, 0 to 5189
Data columns (total 13 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
```

```
 0   Unnamed: 0  5190 non-null   int64
 1   visits      5190 non-null   int64
 2   gender      5190 non-null   object
 3   age         5190 non-null   float64
 4   income      5190 non-null   float64
 5   illness     5190 non-null   int64
 6   reduced     5190 non-null   int64
 7   health      5190 non-null   int64
 8   private     5190 non-null   object
 9   freepoor    5190 non-null   object
 10  freerepat   5190 non-null   object
 11  nchronic    5190 non-null   object
 12  lchronic    5190 non-null   object
dtypes: float64(2), int64(5), object(6)
memory usage: 527.2+ KB
```

```
ds["illness"].value_counts()
```

```
1    1638
0    1554
2     946
3     542
4     274
5     236
Name: illness, dtype: int64
```

```
ds['gender'].value_counts()    #value_counts() method is used to find the number of unique values in a column along with their count
```

```
female    2702
male      2488
Name: gender, dtype: int64
```

```
ds.isnull().sum()      #this method is used to find the number of null values in the column.
```

```
Unnamed: 0    0
visits        0
gender        0
age           0
income        0
illness       0
reduced       0
health        0
private       0
freepoor      0
freerepat     0
nchronic      0
lchronic      0
dtype: int64
```

Data Cleaning

```
ds['age']=ds['age']*100
ds
```

	Unnamed: 0	visits	gender	age	income	illness	reduced	health	private	freepoor	freerepat	nc
0	1	1	female	19.0	0.55	1	4	1	yes	no	no	
1	2	1	female	19.0	0.45	1	2	1	yes	no	no	
2	3	1	male	19.0	0.90	3	0	0	no	no	no	
3	4	1	male	19.0	0.15	1	0	0	no	no	no	
4	5	1	male	19.0	0.45	2	5	1	no	no	no	
...	...	...	...	...	...	...	...	...	...	...	...	
5185	5186	0	female	22.0	0.55	0	0	0	no	no	no	
5186	5187	0	male	27.0	1.30	0	0	1	no	no	no	
5187	5188	0	female	37.0	0.25	1	0	1	no	no	yes	
5188	5189	0	female	52.0	0.65	0	0	0	no	no	no	
5189	5190	0	male	72.0	0.25	0	0	0	no	no	yes	

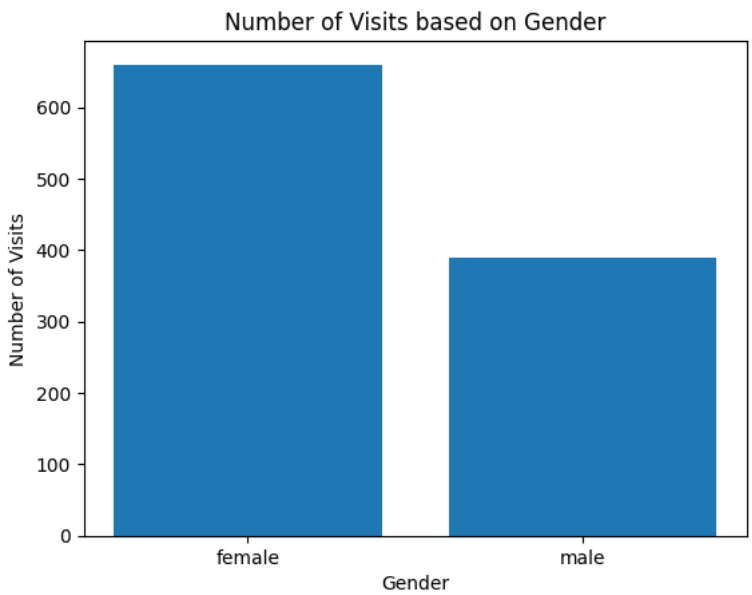
5190 rows × 13 columns

```
df = df.drop(ds[(ds.visits==0)].index)       #this df variable contains only the rows where the number of visits is not '0'.
df
```

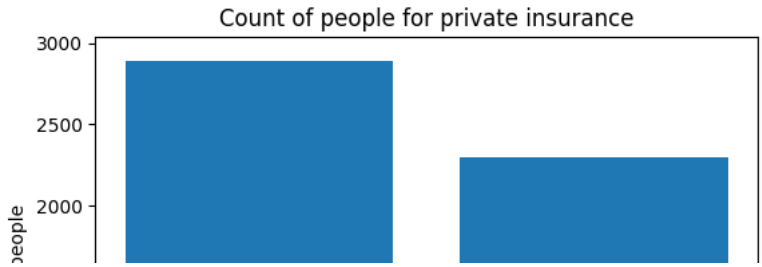
	Unnamed: 0	visits	gender	age	income	illness	reduced	health	private	freepoor	freerepat	nc
0	1	1	female	19.0	0.55	1	4	1	yes	no	no	
1	2	1	female	19.0	0.45	1	2	1	yes	no	no	
2	3	1	male	19.0	0.90	3	0	0	no	no	no	
3	4	1	male	19.0	0.15	1	0	0	no	no	no	
4	5	1	male	19.0	0.45	2	5	1	no	no	no	
...	...	...	...	...	...	...	...	...	...	...	...	
1044	1045	2	female	72.0	0.25	0	0	0	no	no	yes	
1045	1046	1	female	72.0	0.25	0	0	0	no	no	yes	
1046	1047	1	male	72.0	0.25	0	0	1	no	no	yes	
1047	1048	1	male	72.0	0.25	0	0	2	no	no	yes	
1048	1049	1	female	72.0	0.25	0	0	2	no	no	yes	

Data Visualization Using Matplotlib

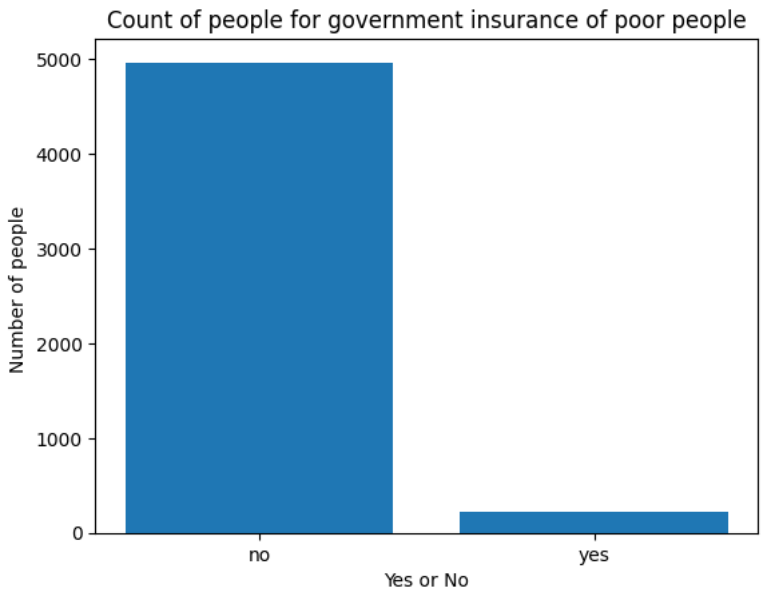
```
import matplotlib.pyplot as plt
fd=ds[ds['visits']>=1]
visit_counts=fd['gender'].value_counts()
plt.bar(visit_counts.index, visit_counts.values)
plt.xlabel('Gender')
plt.ylabel('Number of Visits')
plt.title('Number of Visits based on Gender')
plt.show()
```




```
c1=ds['private'].value_counts()
plt.bar(c1.index,c1.values)
plt.xlabel('Yes or No')
plt.ylabel('Number of people ')
plt.title('Count of people for private insurance')
plt.show()
```




```
c2=ds['freepoor'].value_counts()
plt.bar(c2.index,c2.values)
plt.xlabel('Yes or No')
plt.ylabel('Number of people ')
plt.title('Count of people for government insurance of poor people')
plt.show()
```



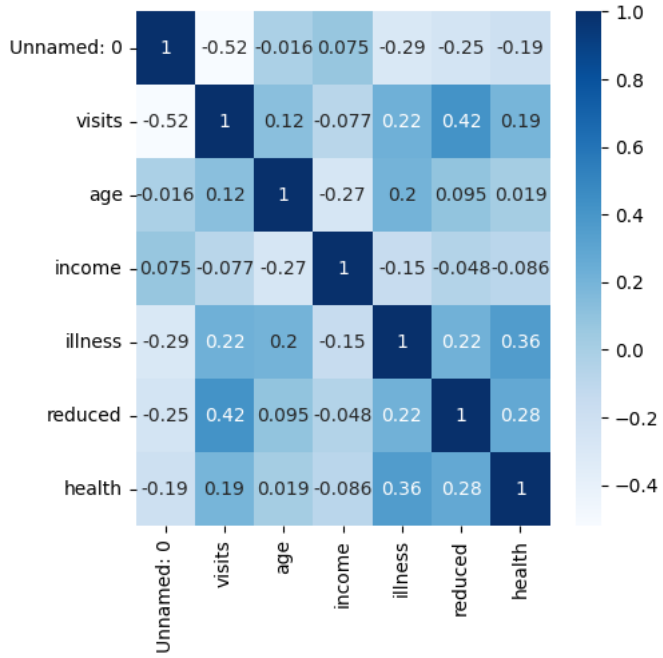

```
df.groupby(['gender','reduced']).mean()
```

```
<ipython-input-16-7beb99933e54>:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy
  df.groupby(['gender','reduced']).mean()
```

gender	reduced	Unnamed: 0	visits	age	income	illness	health
female	0	595.977117	1.199085	52.425629	0.419039	2.011442	1.624714
	1	314.642857	1.357143	32.392857	0.591071	2.642857	1.892857
	2	416.541667	1.541667	39.500000	0.517083	2.500000	1.583333
	3	405.545455	1.818182	39.121212	0.520000	2.818182	1.878788
	4	443.937500	1.437500	43.187500	0.537500	1.875000	1.875000
	5	485.250000	2.166667	43.000000	0.666667	2.166667	2.583333
	6	456.142857	2.142857	42.285714	0.578571	2.285714	1.285714
	7	468.058824	2.117647	42.411765	0.468235	2.764706	2.294118
	8	453.400000	2.400000	40.000000	0.550000	1.400000	4.000000

```
import seaborn as sns
plt.figure(figsize=(5,5))
sns.heatmap(ds.corr(),cbar=True,annot=True,cmap='Blues')
```

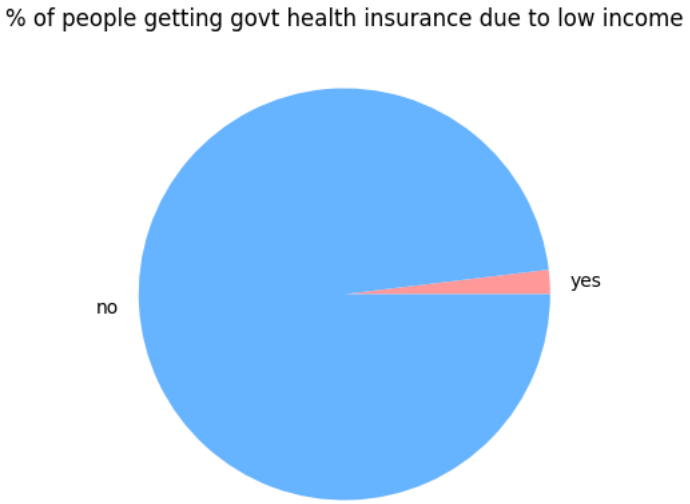
```
<ipython-input-22-ccf4191eedb4>:3: FutureWarning: The default value of numeric_only in DataFrame.corr i
  sns.heatmap(ds.corr(),cbar=True,annot=True,cmap='Blues')
<Axes: >
```



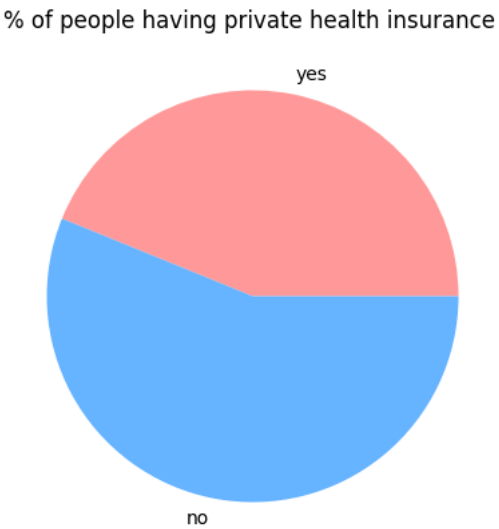

```
plt.figure(figsize=(5,5))
plt.scatter(x='income',y='visits',data=df)
plt.xlabel('income')
plt.ylabel('visits')
```

```
Text(0, 0.5, 'visits')
```


```
label=['yes','no']
Y=df[df['freepoor']=='yes']
N=df[df['freepoor']=='no']
colors = ['#ff9999','#66b3ff']
x= [Y.shape[0],N.shape[0]]
plt.figure(figsize=(5,5))
plt.pie(x,labels=label,colors=colors)
plt.title("% of people getting govt health insurance due to low income")
plt.show()
```

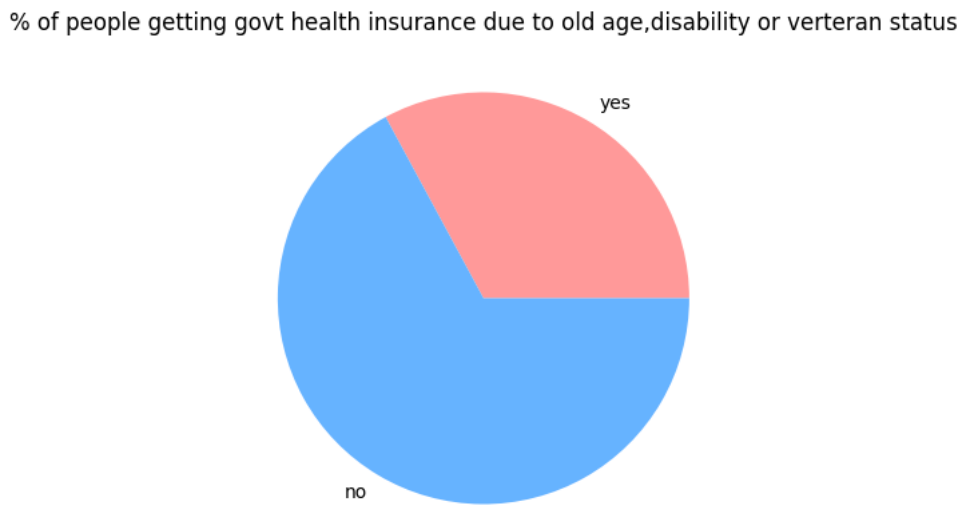



```
label=['yes','no']
Y=df[df['private']=='yes']
N=df[df['private']=='no']
x= [Y.shape[0],N.shape[0]]
colors = ['#ff9999','#66b3ff']
plt.figure(figsize=(5,5))
plt.pie(x,labels=label,colors=colors)
plt.title("% of people having private health insurance ")
plt.show()
```




```
label=['yes','no']
Y=df[df['freerepat']=='yes']
N=df[df['freerepat']=='no']
x= [Y.shape[0],N.shape[0]]
colors = ['#ff9999','#66b3ff']
plt.figure(figsize=(5,5))
plt.pie(x,labels=label,colors=colors)
```

```
plt.title("% of people getting govt health insurance due to old age,disability or verteran status")
plt.show()
```




```
db=ds.groupby('gender')['reduced'].sum().to_frame().reset_index()
plt.barh(db['gender'],db['reduced'],color=['red','blue'])
plt.title("Reduced activity vs gender")
plt.xlabel('gender')
plt.ylabel('reduced activity')
plt.show()
```