```python
In [1]:
import pandas as pd
import numpy as np
```

Customer churn prediction is to measure why customers are leaving a business.

```python
In [2]:
df = pd.read_csv("customer_churn.csv")
```

```python
In [4]:
df.sample(5)
```

Out[4]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies
2252	1813-JLKWR	Female	0	Yes	Yes	64	Yes	No	No	No internet service	...	No internet service	No internet service	No internet service	No internet service
3679	1240-KNSEZ	Male	0	No	No	1	Yes	No	No	No internet service	...	No internet service	No internet service	No internet service	No internet service
659	2085-JVGAD	Male	0	Yes	No	37	Yes	No	No	No internet service	...	No internet service	No internet service	No internet service	No internet service
2124	7802-EFKNY	Male	0	Yes	No	5	No	No phone service	DSL	No	...	No	No	No	No
642	0970-ETWGE	Male	0	No	No	1	Yes	No	Fiber optic	No	...	No	No	Yes	Yes

5 rows × 21 columns

```python
In [5]:
df.shape
```

Out[5]: (7043, 21)

First of all, drop customerID column as it is of no use

```python
In [6]:
df.drop('customerID', axis='columns', inplace=True)
```

```python
In [7]:
df.sample(5)
```

Out[7]:

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	DeviceProtection	TechSupport	StreamingTV	StreamingMovies
5551	Male	0	No	No	20	Yes	No	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service
3023	Male	0	Yes	Yes	72	Yes	Yes	DSL	Yes	Yes	Yes	Yes	Yes	Yes
2227	Male	0	Yes	Yes	58	Yes	No	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service
1081	Male	1	Yes	No	46	Yes	Yes	Fiber optic	No	No	Yes	No	Yes	Yes
2225	Male	0	No	No	33	No	No phone service	DSL	No	Yes	No	No	Yes	No

```python
In [9]:
df.dtypes
```

```
Out[9]:
gender               object
SeniorCitizen         int64
Partner              object
Dependents           object
tenure                int64
PhoneService         object
MultipleLines        object
InternetService      object
OnlineSecurity       object
OnlineBackup         object
DeviceProtection     object
TechSupport          object
StreamingTV          object
StreamingMovies      object
Contract             object
PaperlessBilling     object
PaymentMethod        object
MonthlyCharges      float64
TotalCharges         object
Churn                object
dtype: object
```

Quick glance at above makes us realize that TotalCharges should be float but it is an object. Let's check what's going on with this column

```python
In [10]:
df.TotalCharges.values
```

```
Out[10]:
array(['29.85', '1889.5', '108.15', ..., '346.45', '306.6', '6844.5'],
      dtype=object)
```

it is string. Lets convert it to numbers

```python
In [11]:
pd.to_numeric(df.TotalCharges)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\anaconda3\lib\site-packages\pandas\_libs\lib.pyx in pandas._libs.lib.maybe_convert_numeric()

ValueError: Unable to parse string " "

During handling of the above exception, another exception occurred:

ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_19012/2112264836.py in <module>
----> 1 pd.to_numeric(df.TotalCharges)

~\anaconda3\lib\site-packages\pandas\core\tools\numeric.py in to_numeric(arg, errors, downcast)
    181         coerce_numeric = errors not in ("ignore", "raise")
    182         try:
--> 183             values, _ = lib.maybe_convert_numeric(
    184                 values, set(), coerce_numeric=coerce_numeric
    185             )

~\anaconda3\lib\site-packages\pandas\_libs\lib.pyx in pandas._libs.lib.maybe_convert_numeric()

ValueError: Unable to parse string " " at position 488
```

some values seems to be not numbers but blank string. Let's find out such rows

```python
In [13]:
df[pd.to_numeric(df.TotalCharges, errors='coerce').isnull()]
```

Out[13]:

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	DeviceProtection	TechSupport	StreamingTV	StreamingMovies
488	Female	0	Yes	Yes	0	No	No phone service	DSL	Yes	No	Yes	Yes	Yes	No
753	Male	0	No	Yes	0	Yes	No	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service
936	Female	0	Yes	Yes	0	Yes	No	DSL	Yes	Yes	Yes	No	Yes	Yes
1082	Male	0	Yes	Yes	0	Yes	Yes	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service
1340	Female	0	Yes	Yes	0	No	No phone service	DSL	Yes	Yes	Yes	Yes	Yes	No
3331	Male	0	Yes	Yes	0	Yes	No	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service
3826	Male	0	Yes	Yes	0	Yes	Yes	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service
4380	Female	0	Yes	Yes	0	Yes	No	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service
5218	Male	0	Yes	Yes	0	Yes	No	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service
6670	Female	0	Yes	Yes	0	Yes	Yes	DSL	No	Yes	Yes	Yes	Yes	No
6754	Male	0	No	Yes	0	Yes	Yes	DSL	Yes	Yes	No	Yes	No	No

```python
In [14]:
df.iloc[488].TotalCharges
```

Out[14]: ' '

Remove rows with space in TotalCharges

```python
In [15]:
df1 = df[df.TotalCharges!=' ']
```

```python
In [16]:
df1.TotalCharges = pd.to_numeric(df1.TotalCharges)
```

```
C:\Users\User\anaconda3\lib\site-packages\pandas\core\generic.py:5516: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  self[name] = value
```

```python
In [17]:
df1[df1.Churn=='No']
```

Out[17]:

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	DeviceProtection	TechSupport	StreamingTV	StreamingMovies
0	Female	0	Yes	No	1	No	No phone service	DSL	No	Yes	No	No	No	No
1	Male	0	No	No	34	Yes	No	DSL	Yes	No	Yes	No	No	No
3	Male	0	No	No	45	No	No phone service	DSL	Yes	No	Yes	Yes	No	No
6	Male	0	No	Yes	22	Yes	Yes	Fiber optic	No	Yes	No	No	Yes	No
7	Female	0	No	No	10	No	No phone service	DSL	Yes	No	No	No	No	No
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7037	Female	0	No	No	72	Yes	No	No	No internet service	No internet service	No internet service	No internet service	No internet service	No internet service
7038	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes	No	Yes	Yes	Yes	Yes
7039	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No	Yes	Yes	No	Yes	Yes
7040	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes	No	No	No	No	No
7042	Male	0	No	No	66	Yes	No	Fiber optic	Yes	No	Yes	Yes	Yes	Yes

5163 rows × 20 columns

Many of the columns are yes, no etc. Let's print unique values in object columns to see data values

```python
In [90]:
def print_unique(df1):
    for column in df1:
        if df1[column].dtypes=='object':
            print(f'{column}: {df[column].unique()}')
```

```python
In [91]:
print_unique(df1)
```

```
gender: ['Female' 'Male']
PhoneService: ['No' 'Yes']
MultipleLines: ['No' 'Yes']
InternetService: ['DSL' 'Fiber optic' 'No']
OnlineSecurity: ['No' 'Yes']
OnlineBackup: ['Yes' 'No']
DeviceProtection: ['No' 'Yes']
TechSupport: ['No' 'Yes']
StreamingTV: ['No' 'Yes']
StreamingMovies: ['No' 'Yes']
Contract: ['Month-to-month' 'One year' 'Two year']
PaymentMethod: ['Electronic check' 'Mailed check' 'Bank transfer (automatic)'
 'Credit card (automatic)']
Churn: ['No' 'Yes']
```

Some of the columns have no internet service or no phone service, that can be replaced with a simple No

```python
In [94]:
df1.replace('No internet service','No',inplace=True)
df1.replace('No phone service','No',inplace=True)
```

```python
In [95]:
print_unique(df1)
```

```
gender: ['Female' 'Male']
PhoneService: ['No' 'Yes']
MultipleLines: ['No' 'Yes']
InternetService: ['DSL' 'Fiber optic' 'No']
OnlineSecurity: ['No' 'Yes']
OnlineBackup: ['Yes' 'No']
DeviceProtection: ['No' 'Yes']
TechSupport: ['No' 'Yes']
StreamingTV: ['No' 'Yes']
StreamingMovies: ['No' 'Yes']
Contract: ['Month-to-month' 'One year' 'Two year']
PaymentMethod: ['Electronic check' 'Mailed check' 'Bank transfer (automatic)'
 'Credit card (automatic)']
Churn: ['No' 'Yes']
```

Convert Yes and No to 1 or 0

```python
In [96]:
yes_no_Column = ['Partner', 'Dependents', 'PaperlessBilling']

for col in yes_no_Column:
    df1[col].replace({'Yes': 1, 'No': 0}, inplace=True)
```

```python
In [38]:
df1.dtypes
```

```
Out[38]:
gender               object
SeniorCitizen         int64
Partner               int64
Dependents            int64
tenure                int64
PhoneService         object
MultipleLines        object
InternetService      object
OnlineSecurity       object
OnlineBackup         object
DeviceProtection     object
TechSupport          object
StreamingTV          object
StreamingMovies      object
Contract             object
PaperlessBilling      int64
PaymentMethod        object
MonthlyCharges      float64
TotalCharges        float64
Churn                object
dtype: object
```

Applying get dummies on categorical Variables

```python
In [39]:
df2 = pd.get_dummies(data=df1, columns=['InternetService', 'Contract', 'PaymentMethod'])
```

```python
In [40]:
df2
```

Out[40]:

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	OnlineSecurity	OnlineBackup	DeviceProtection	...	InternetService_DSL	InternetService_Fiber optic
0	Female	0	1	0	1	No	No	No	Yes	No	...	1	0
1	Male	0	0	0	34	Yes	No	Yes	No	Yes	...	1	0
2	Male	0	0	0	2	Yes	No	Yes	Yes	No	...	1	0
3	Male	0	0	0	45	No	No	Yes	No	Yes	...	1	0
4	Female	0	0	0	2	Yes	No	No	No	No	...	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
7038	Male	0	1	1	24	Yes	Yes	Yes	No	Yes	...	1	0
7039	Female	0	1	1	72	Yes	Yes	No	Yes	Yes	...	0	1
7040	Female	0	1	1	11	No	No	Yes	No	No	...	1	0
7041	Male	1	1	0	4	Yes	Yes	No	No	No	...	0	1
7042	Male	0	0	0	66	Yes	No	Yes	No	Yes	...	0	1

7032 rows × 27 columns

```python
In [41]:
df2.dtypes
```

```
Out[41]:
gender                                      object
SeniorCitizen                                int64
Partner                                      int64
Dependents                                   int64
tenure                                       int64
PhoneService                                object
MultipleLines                               object
OnlineSecurity                              object
OnlineBackup                                object
DeviceProtection                            object
TechSupport                                 object
StreamingTV                                 object
StreamingMovies                             object
PaperlessBilling                             int64
MonthlyCharges                             float64
TotalCharges                               float64
Churn                                       object
InternetService_DSL                          uint8
InternetService_Fiber optic                  uint8
InternetService_No                           uint8
Contract_Month-to-month                      uint8
Contract_One year                            uint8
Contract_Two year                            uint8
PaymentMethod_Bank transfer (automatic)      uint8
PaymentMethod_Credit card (automatic)        uint8
PaymentMethod_Electronic check               uint8
PaymentMethod_Mailed check                   uint8
dtype: object
```

In [ ]:

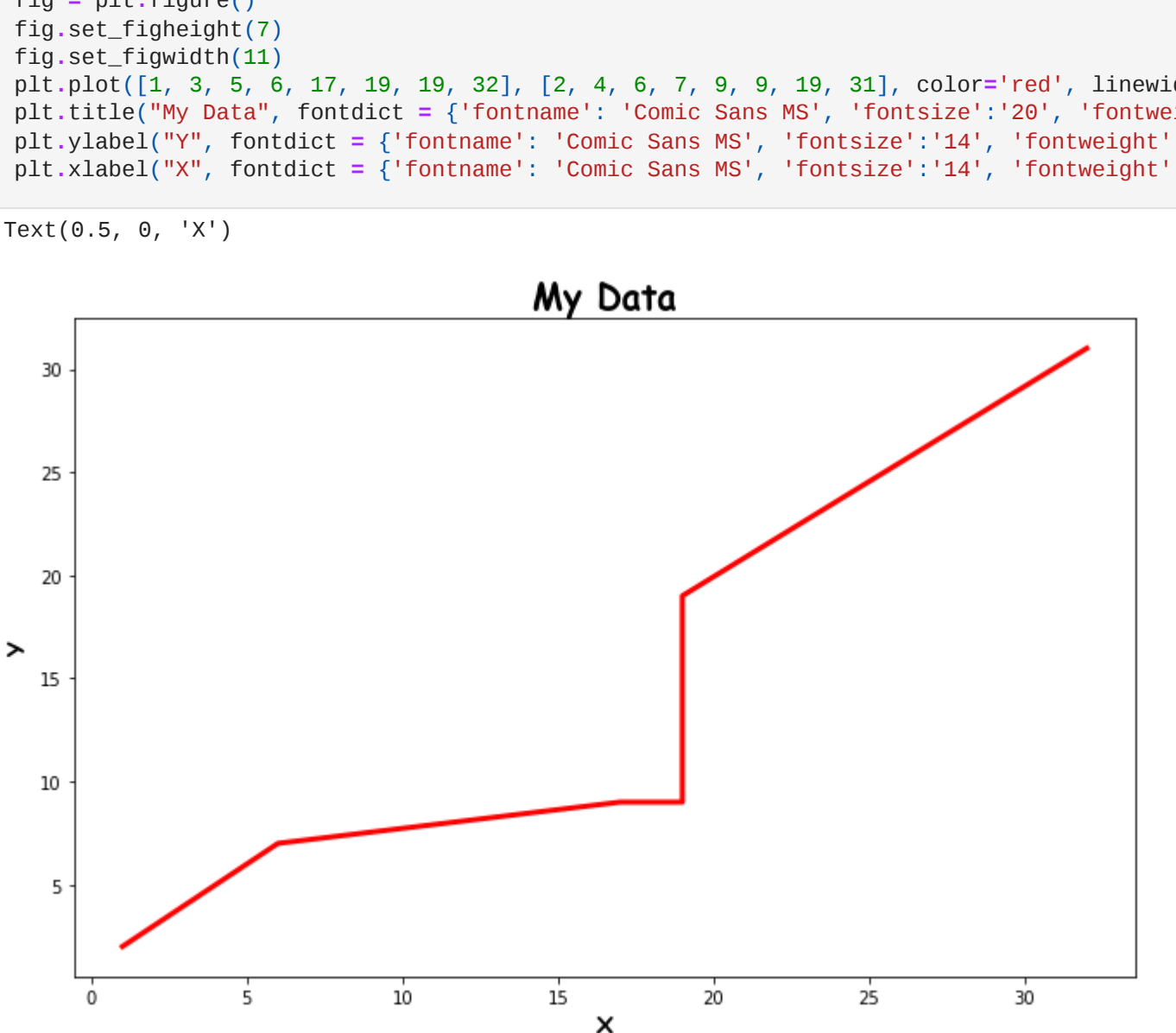
Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.

```python
In [43]:
from matplotlib import pyplot as plt
%matplotlib inline
```

In [ ]:

```python
In [58]:
fig = plt.figure()
fig.set_figheight(7)
fig.set_figwidth(11)
plt.plot([1, 3, 5, 6, 17, 19, 19, 32], [2, 4, 6, 7, 9, 9, 19, 31], color='red', linewidth=3)
plt.title("My Data", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'20', 'fontweight':'bold'})
plt.ylabel("Y", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.xlabel("X", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
```

Out[58]: Text(0.5, 0, 'X')

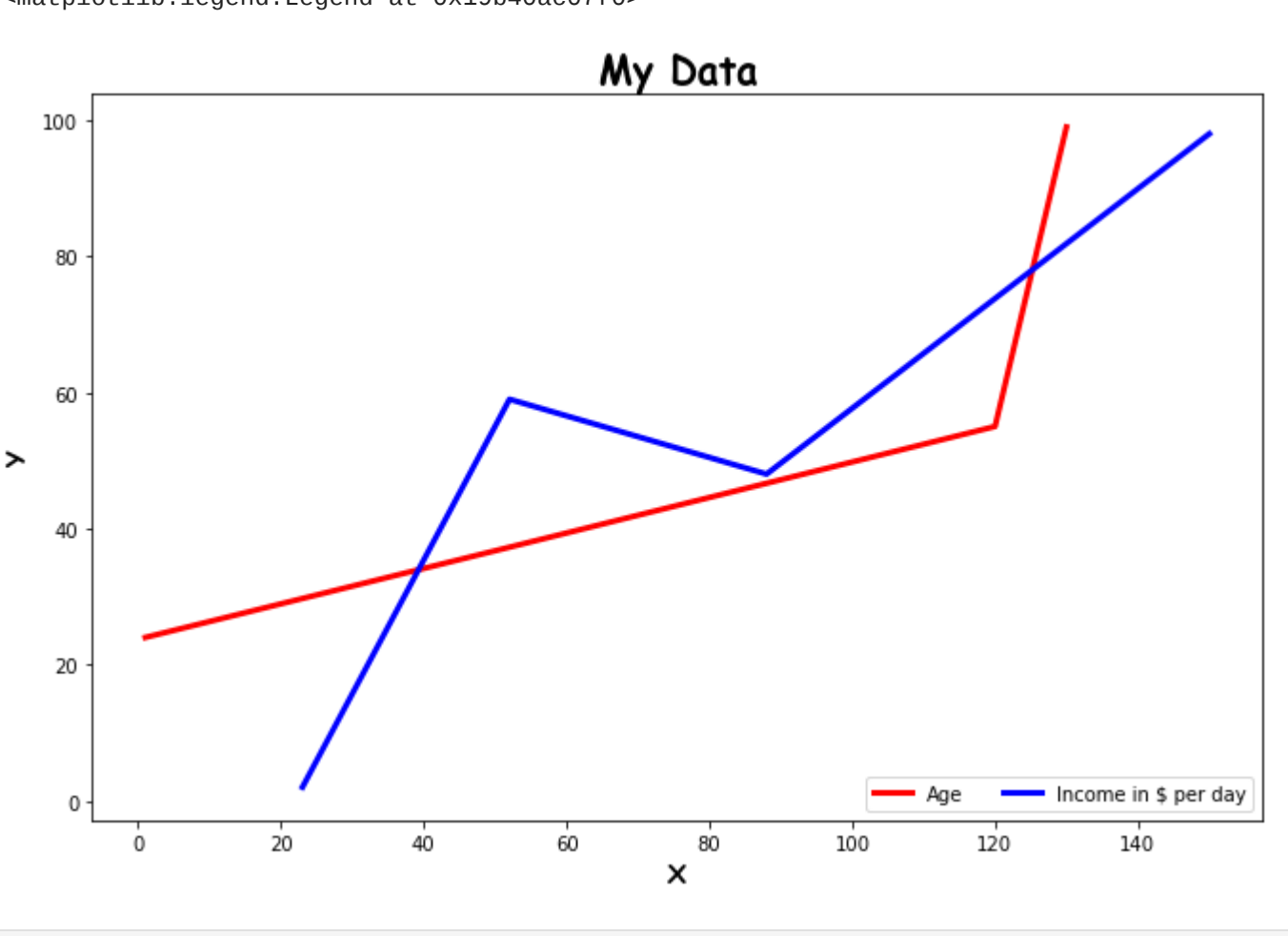



```python
In [60]:
fig = plt.figure()
fig.set_figheight(7)
fig.set_figwidth(11)
x = [1, 51, 120, 130]
y = [24, 37, 55, 99]
x1 = [23, 52, 88, 150]
y1 = [2, 59, 48, 98]
plt.plot(x, y, color='red', linewidth=3)
plt.plot(x1, y1, color='blue', linewidth=3)
plt.title("My Data", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'20', 'fontweight':'bold'})
plt.ylabel("Y", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.xlabel("X", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
```

Out[60]: Text(0.5, 0, 'X')

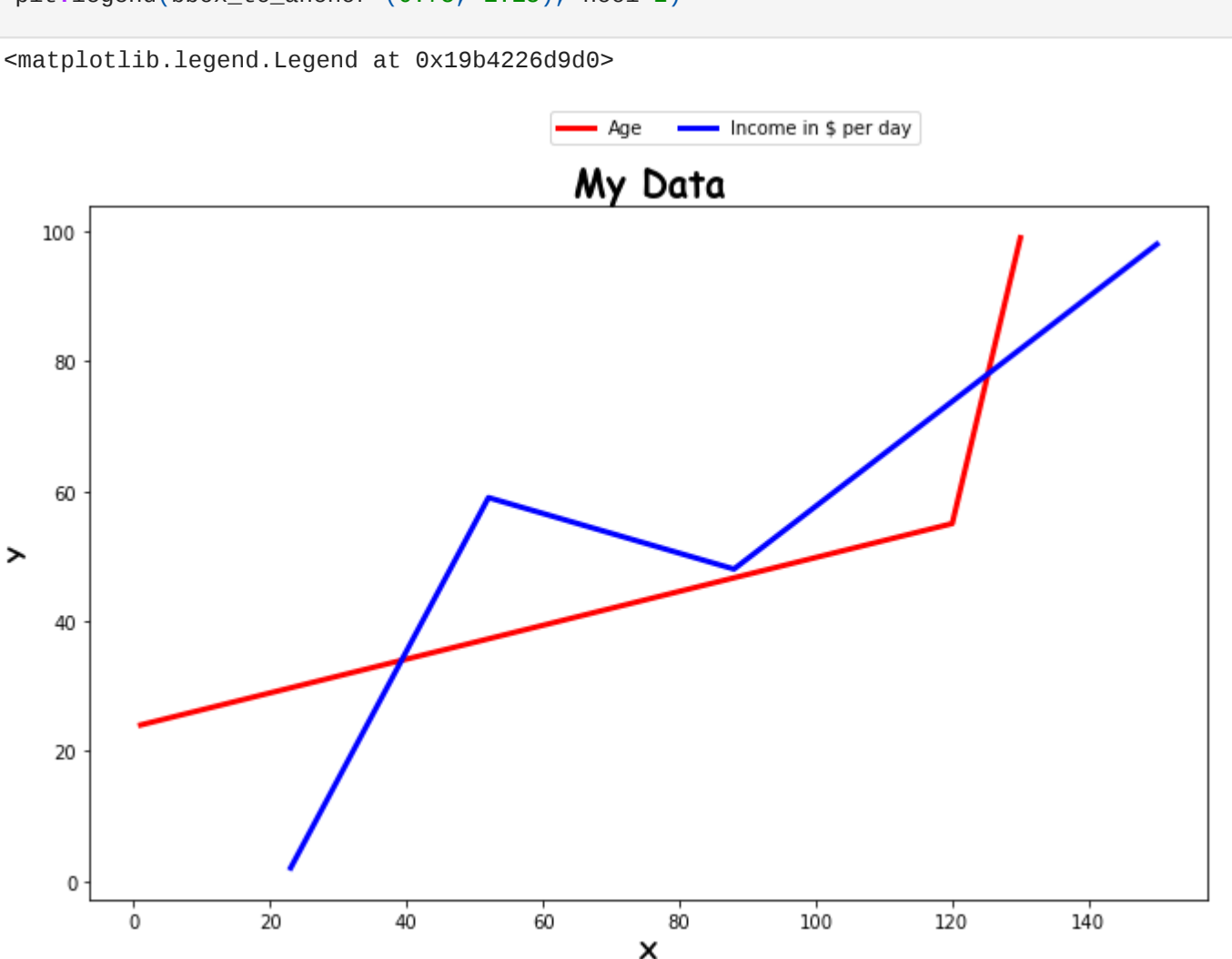

```python
In [64]:
fig = plt.figure()
fig.set_figheight(7)
fig.set_figwidth(11)
x = [1, 51, 120, 130]
y = [24, 37, 55, 99]
x1 = [23, 52, 88, 150]
y1 = [2, 59, 48, 98]
plt.plot(x, y, color='red', linewidth=3, label="Age")
plt.plot(x1, y1, color='blue', linewidth=3, label="Income in $ per day")
plt.title("My Data", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'20', 'fontweight':'bold'})
plt.ylabel("Y", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.xlabel("X", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.legend(loc="lower right", ncol=2)
```

Out[64]: <matplotlib.legend.Legend at 0x19b40ac07f0>



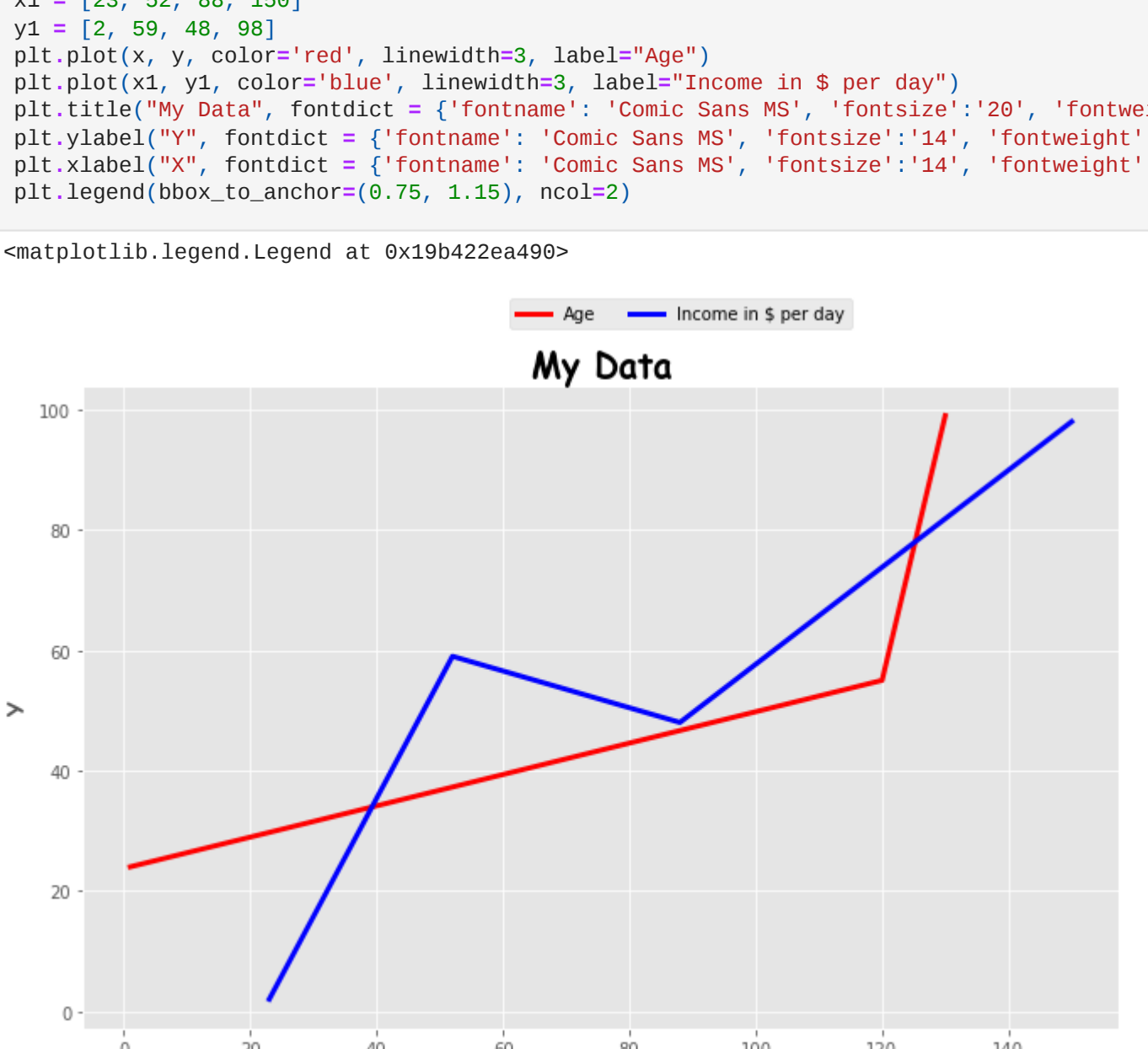

```python
In [69]:
fig = plt.figure()
fig.set_figheight(7)
fig.set_figwidth(11)
x = [1, 51, 120, 130]
y = [24, 37, 55, 99]
x1 = [23, 52, 88, 150]
y1 = [2, 59, 48, 98]
plt.plot(x, y, color='red', linewidth=3, label="Age")
plt.plot(x1, y1, color='blue', linewidth=3, label="Income in $ per day")
plt.title("My Data", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'20', 'fontweight':'bold'})
plt.ylabel("Y", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.xlabel("X", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.legend(bbox_to_anchor=(0.75, 1.15), ncol=2)
```

Out[69]: <matplotlib.legend.Legend at 0x19b4226d9d0>




```python
In [70]:
from matplotlib import style
style.use('ggplot')
fig = plt.figure()
fig.set_figheight(7)
fig.set_figwidth(11)
x = [1, 51, 120, 130]
y = [24, 37, 55, 99]
x1 = [23, 52, 88, 150]
y1 = [2, 59, 48, 98]
plt.plot(x, y, color='red', linewidth=3, label="Age")
plt.plot(x1, y1, color='blue', linewidth=3, label="Income in $ per day")
plt.title("My Data", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'20', 'fontweight':'bold'})
plt.ylabel("Y", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.xlabel("X", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.legend(bbox_to_anchor=(0.75, 1.15), ncol=2)
```

Out[70]: <matplotlib.legend.Legend at 0x19b422ea490>




```python
In [71]:
style.use('dark_background')
fig = plt.figure()
fig.set_figheight(7)
fig.set_figwidth(11)
x = [1, 51, 120, 130]
y = [24, 37, 55, 99]
x1 = [23, 52, 88, 150]
y1 = [2, 59, 48, 98]
plt.plot(x, y, color='red', linewidth=3, label="Age")
plt.plot(x1, y1, color='blue', linewidth=3, label="Income in $ per day")
plt.title("My Data", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'20', 'fontweight':'bold'})
plt.ylabel("Y", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.xlabel("X", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.legend(bbox_to_anchor=(0.75, 1.15), ncol=2)
```

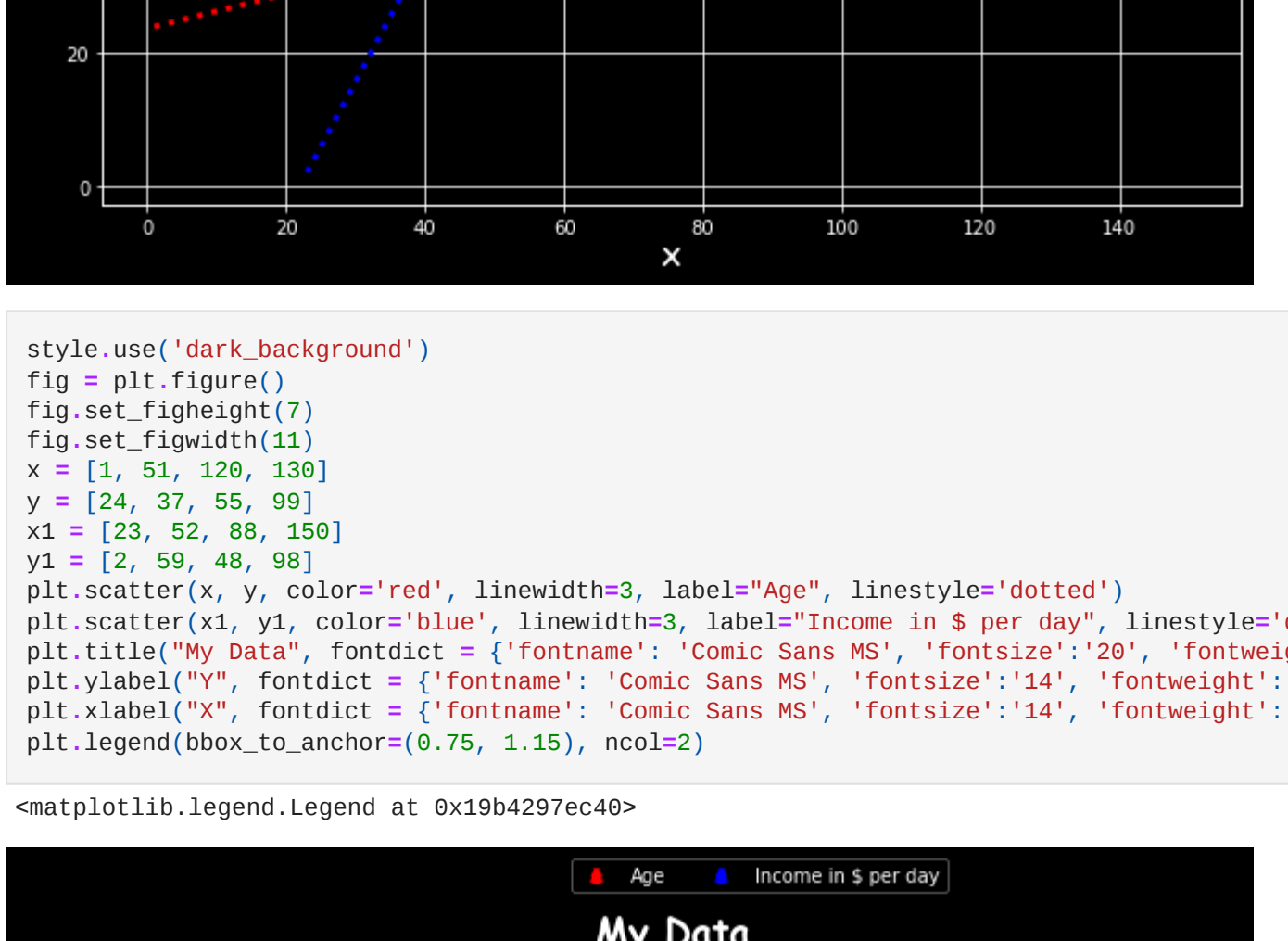
Out[71]: <matplotlib.legend.Legend at 0x19b42364910>


```python
In [73]:
style.use('dark_background')
fig = plt.figure()
fig.set_figheight(7)
fig.set_figwidth(11)
x = [1, 51, 120, 130]
y = [24, 37, 55, 99]
x1 = [23, 52, 88, 150]
y1 = [2, 59, 48, 98]
plt.plot(x, y, color='red', linewidth=3, label="Age", linestyle='dotted')
plt.plot(x1, y1, color='blue', linewidth=3, label="Income in $ per day", linestyle='dotted')
plt.title("My Data", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'20', 'fontweight':'bold'})
plt.ylabel("Y", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.xlabel("X", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.legend(bbox_to_anchor=(0.75, 1.15), ncol=2)
```

Out[73]: <matplotlib.legend.Legend at 0x19b3f5da2e0>


```python
In [77]:
style.use('dark_background')
fig = plt.figure()
fig.set_figheight(7)
fig.set_figwidth(11)
x = [1, 51, 120, 130]
y = [24, 37, 55, 99]
x1 = [23, 52, 88, 150]
y1 = [2, 59, 48, 98]
plt.scatter(x, y, color='red', linewidth=3, label="Age", linestyle='dotted')
plt.scatter(x1, y1, color='blue', linewidth=3, label="Income in $ per day", linestyle='dotted')
plt.title("My Data", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'20', 'fontweight':'bold'})
plt.ylabel("Y", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.xlabel("X", fontdict = {'fontname': 'Comic Sans MS', 'fontsize':'14', 'fontweight':'bold'})
plt.legend(bbox_to_anchor=(0.75, 1.15), ncol=2)
```

Out[77]: <matplotlib.legend.Legend at 0x19b4297ec40>



In [ ]:

```python
In [79]:
df = pd.read_csv("customer_churn.csv")
```

```python
In [81]:
df
```

Out[81]:

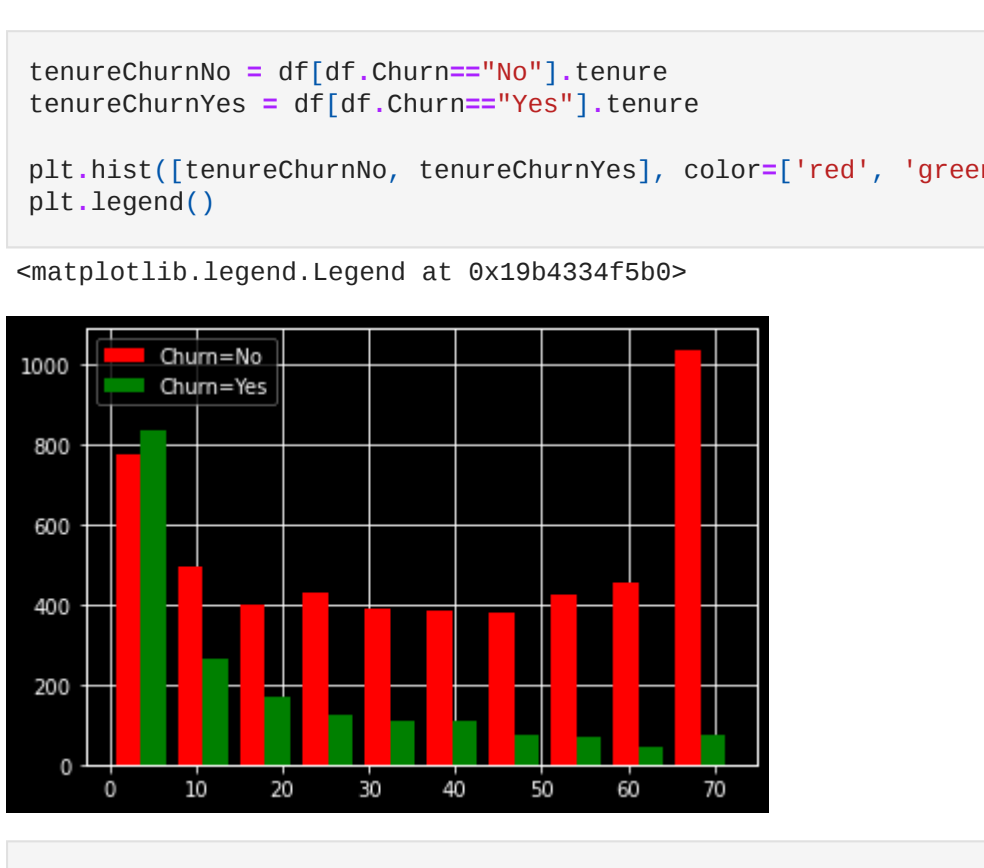
	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	No	No	No
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	No	No	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	No	No	No
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	Yes	No	No
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	No	No	No
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes	...	Yes	Yes	Yes	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No	...	Yes	No	Yes	Yes
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes	...	No	No	No	No
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No	...	No	No	No	No
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	Fiber optic	Yes	...	Yes	Yes	Yes	Yes

7043 rows × 21 columns

```python
In [82]:
tenureChurnNo = df[df.Churn=="No"].tenure
tenureChurnYes = df[df.Churn=="Yes"].tenure

plt.hist([tenureChurnNo, tenureChurnYes], color=['red', 'green'], label=['Churn=No', 'Churn=Yes'])
plt.legend()
```

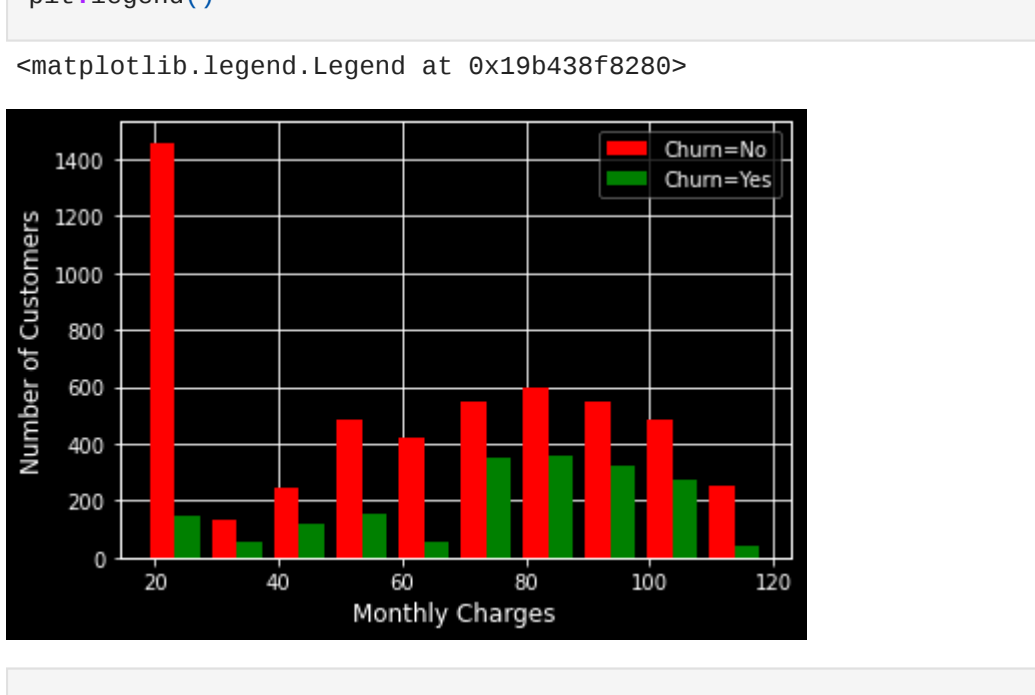
Out[82]: <matplotlib.legend.Legend at 0x19b4334f5b0>




```python
In [85]:
tenureChurnNo = df[df.Churn=="No"].MonthlyCharges
tenureChurnYes = df[df.Churn=="Yes"].MonthlyCharges

plt.xlabel("Monthly Charges")
plt.ylabel("Number of Customers")
plt.hist([tenureChurnNo, tenureChurnYes], color=['red', 'green'], label=['Churn=No', 'Churn=Yes'])
plt.legend()
```

Out[85]: <matplotlib.legend.Legend at 0x19b438f8280>



In [ ]: