1. Why Do We Need Data Visualization? Explain with Examples

Data visualization is the practice of translating information into a visual context, such as a map or graph, to make data easier for the human brain to understand and pull insights from. The main goal of data visualization is to make it easier to identify patterns, trends and outliers in large data sets. The term is often used interchangeably with others, including information graphics, information visualization and statistical graphics.

Outcome from Data Visualization

1.Desicion Meking 2.Finding solution of Problems 3.For Understanding the data clearly 4.To find relationship among the data 5.Comparative analysis

In [1]:
```
import seaborn as sns
```

In [2]:
```
tips=sns.load_dataset('tips')
```

In [3]:
```
tips
```

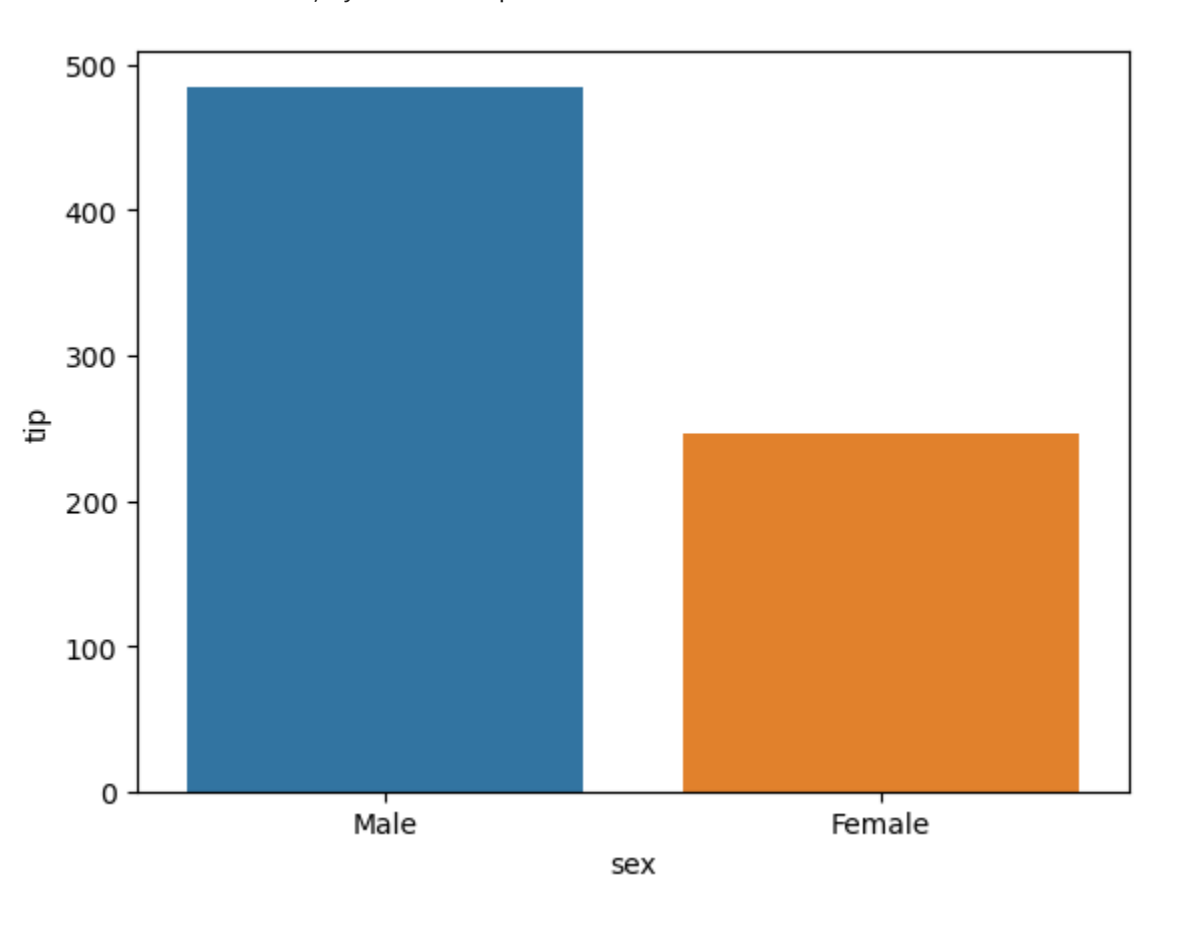
Out[3]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

In [4]:
```
# example for DataVisualization
dx=tips.groupby(tips["sex"])["tip"].sum().reset_index()

sns.barplot(x="sex",y="tip",data=dx)
```

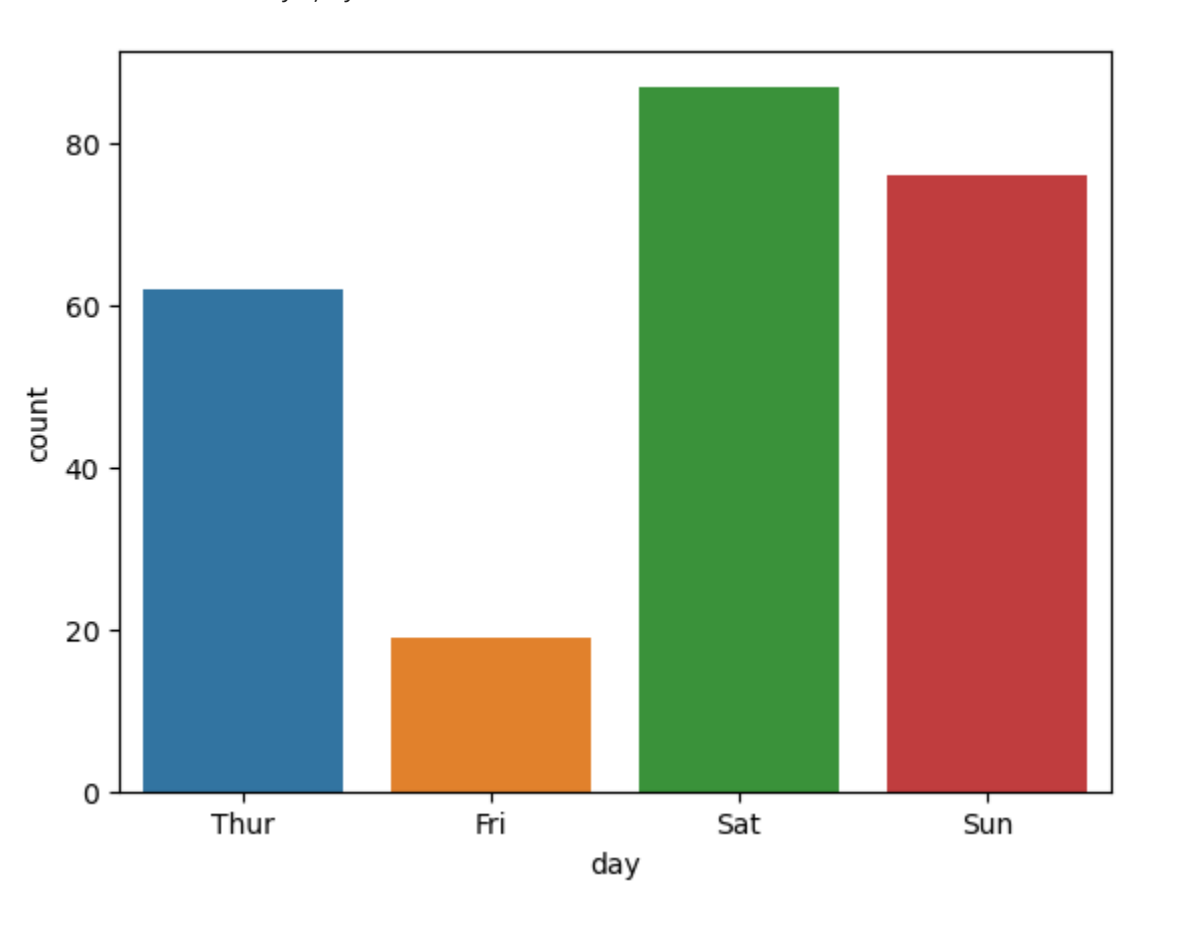
Out[4]: `<Axes: xlabel='sex', ylabel='tip'>`



Conclusion: 1.number of male spend more money for tips than famale

In [5]:
```
sns.countplot(x="day",data=tips)
```

Out[5]: `<Axes: xlabel='day', ylabel='count'>`



Conclusion: 1.Most of time people use to go for lunch or dinner on saturday and sunday

2. Give a Brief Note on Libraries used for Data Visualization.

1.Matplotlib

Matplotlib is a cross-platform, data visualization and graphical plotting library (histograms, scatter plots, bar charts, etc) for Python and its numerical extension NumPy. As such, it offers a viable open source alternative to MATLAB. Developers can also use matplotlib's APIs (Application Programming Interfaces) to embed plots in GUI applications.

A Python matplotlib script is structured so that a few lines of code are all that is required in most instances to generate a visual data plot. The matplotlib scripting layer overlays two APIs:

The pyplot API is a hierarchy of Python code objects topped by matplotlib.pyplot An OO (Object-Oriented) API collection of objects that can be assembled with greater flexibility than pyplot. This API provides direct access to Matplotlib's backend layers.

2.Seaborn

Seaborn is an amazing visualization library for statistical graphics plotting in Python. It provides beautiful default styles and color palettes to make statistical plots more attractive. It is built on top matplotlib library and is also closely integrated with the data structures from pandas. Seaborn aims to make visualization the central part of exploring and understanding data. It provides dataset-oriented APIs so that we can switch between different visual representations for the same variables for a better understanding of the dataset.

Different categories of plot in Seaborn

Plots are basically used for visualizing the relationship between variables. Those variables can be either completely numerical or a category like a group, class, or division. Seaborn divides the plot into the below categories –

- Relational plots: This plot is used to understand the relation between two variables.
- Categorical plots: This plot deals with categorical variables and how they can be visualized.
- Distribution plots: This plot is used for examining univariate and bivariate distributions
- Regression plots: The regression plots in Seaborn are primarily intended to add a visual guide that helps to emphasize patterns in a dataset during exploratory data analyses.
- Matrix plots: A matrix plot is an array of scatterplots.
- Multi-plot grids: It is a useful approach to draw multiple instances of the same plot on different subsets of the dataset.

3.Plotly

Python Plotly Library is an open-source library that can be used for data visualization and understanding data simply and easily. Plotly supports various types of plots like line charts, scatter plots, histograms, cox plots, etc. So you all must be wondering why Plotly over other visualization tools or libraries? Here's the answer –

- Plotly has hover tool capabilities that allow us to detect any outliers or anomalies in a large number of data points.
- It is visually attractive that can be accepted by a wide range of audiences.
- It allows us for the endless customization of our graphs that makes our plot more meaningful and understandable for others.

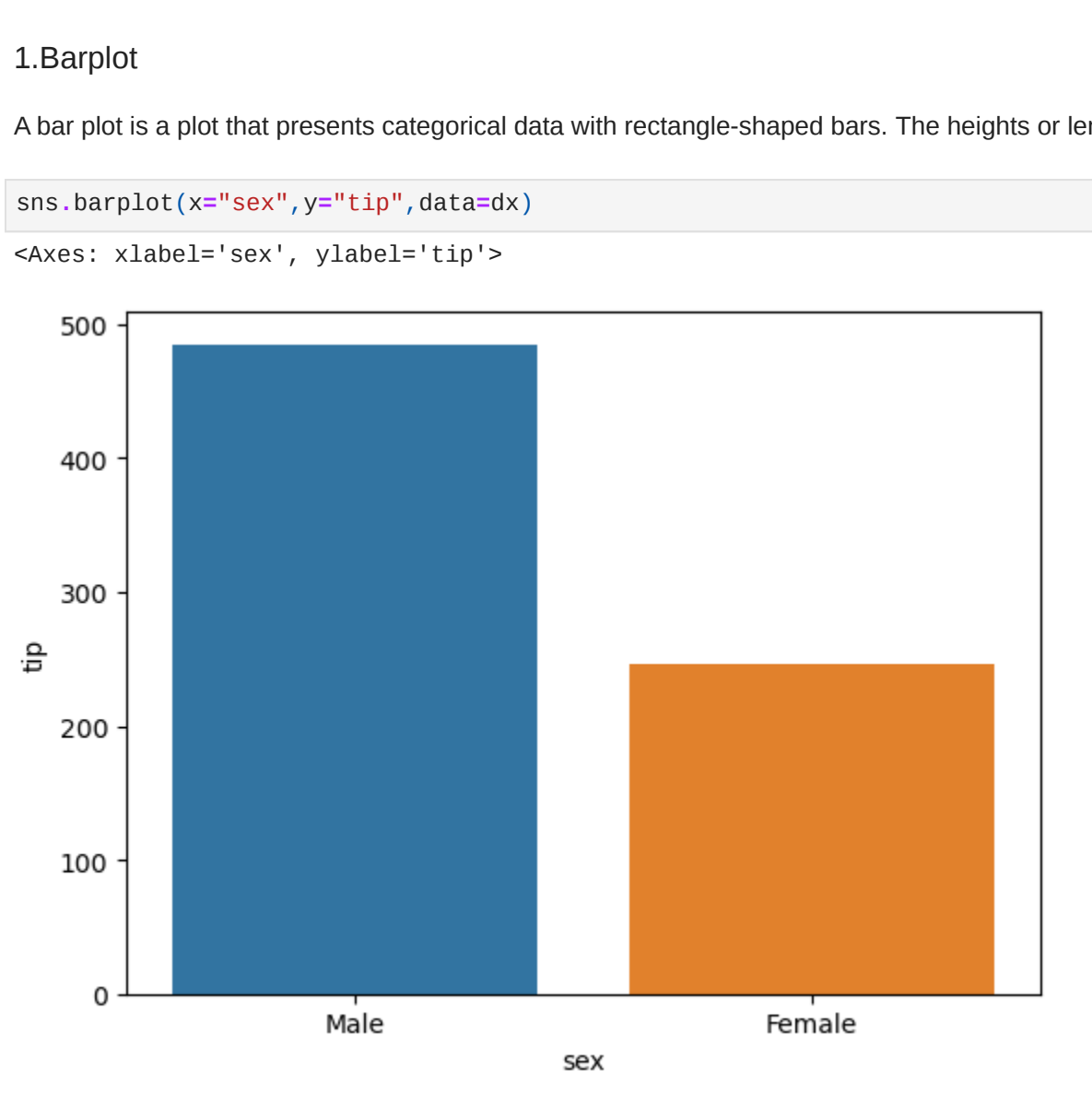
3. Explain Different types of Plot with Syntax.

1.Barplot

A bar plot is a plot that presents categorical data with rectangle-shaped bars. The heights or lengths of these bars are proportional to the values that they represent. The bars can be vertical or horizontal.

In [6]:
```
sns.barplot(x="sex",y="tip",data=dx)
```

Out[6]: `<Axes: xlabel='sex', ylabel='tip'>`

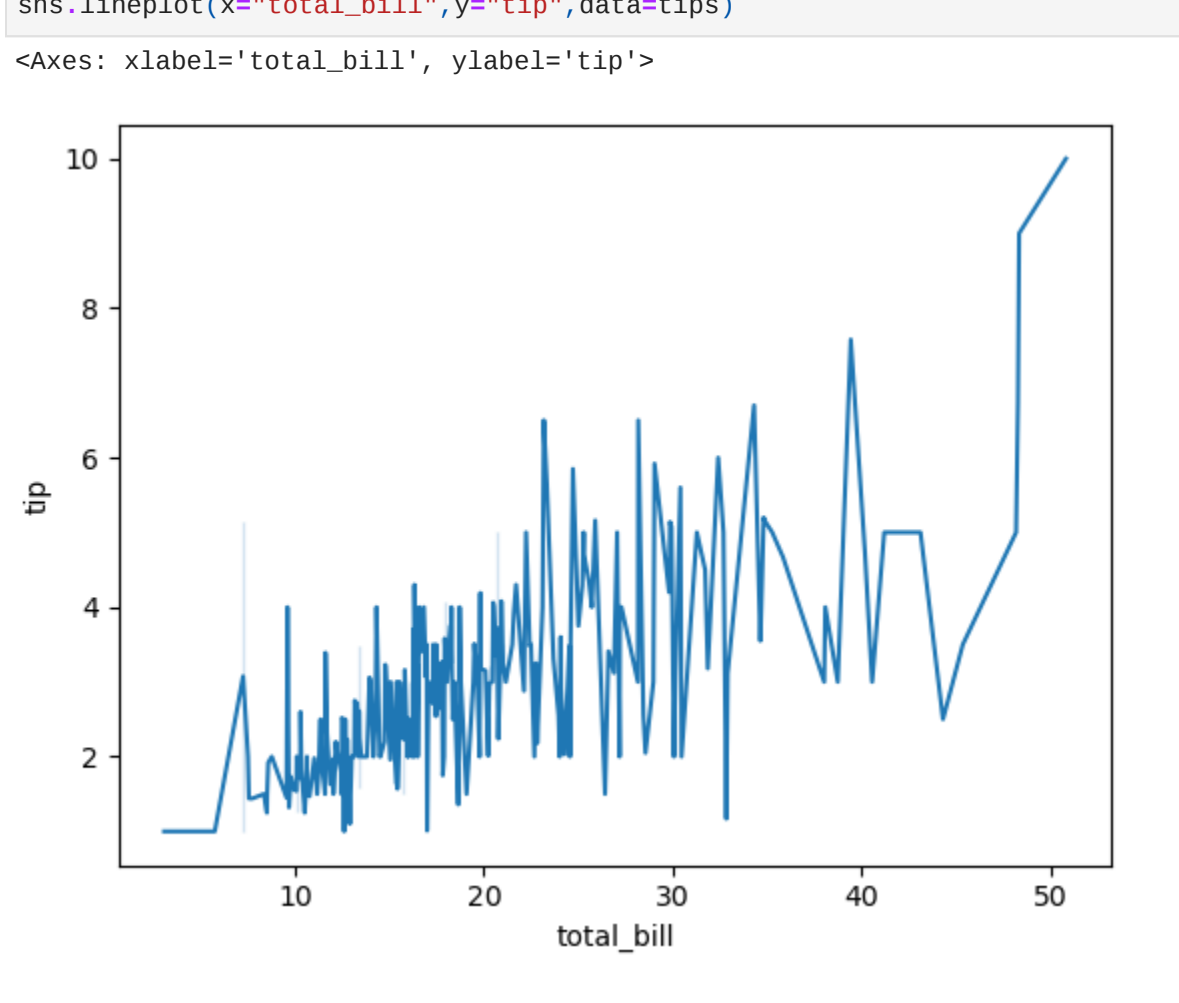


2.Line plot

It displays a sequence of data points as markers. The points are ordered typically by their x-axis value. These points are joined with straight line segments. A line plot is used to visualize a trend in data over intervals of time.

In [7]:
```
sns.lineplot(x="total_bill",y="tip",data=tips)
```

Out[7]: `<Axes: xlabel='total_bill', ylabel='tip'>`



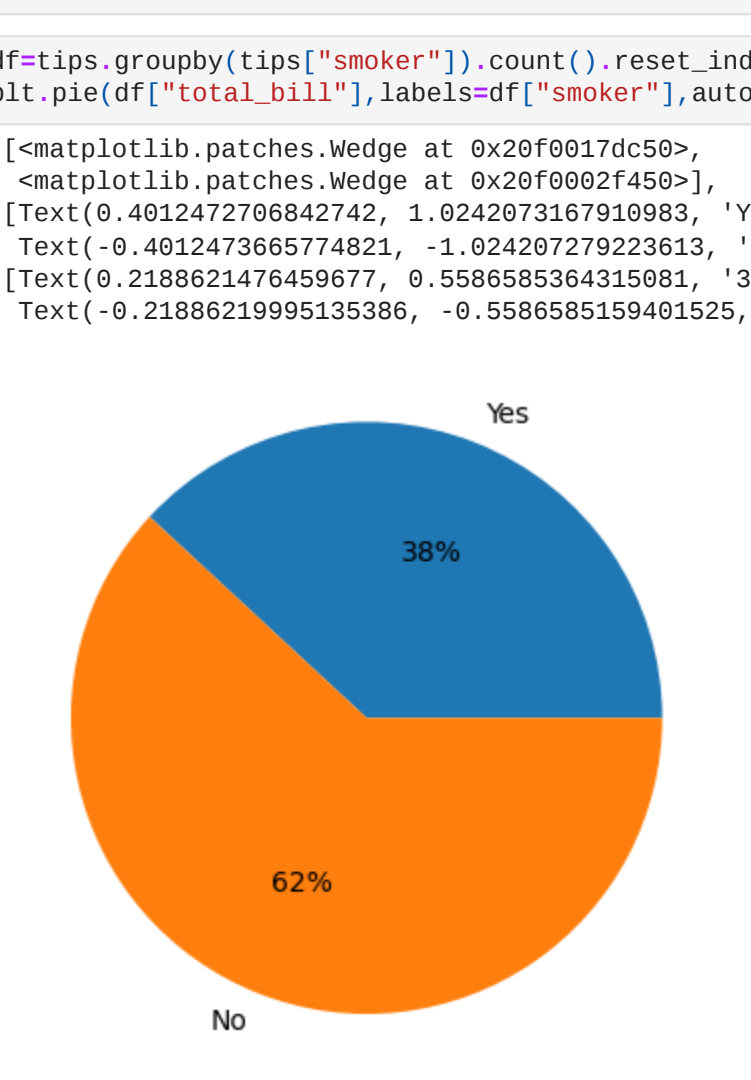
3.Pie Chart

A pie chart is a circular statistical graphic. To illustrate numerical proportion, it is divided into slices. In a pie chart, for every slice, each of its arc lengths is proportional to the amount it represents. The central angles, and area are also proportional. It is named after a sliced pie.

In [8]:
```
import matplotlib.pyplot as plt
```

In [9]:
```
df=tips.groupby(tips["smoker"]).count().reset_index()
plt.pie(df["total_bill"],labels=df["smoker"],autopct='%.0f%%')
```

Out[9]:
```
([<matplotlib.patches.Wedge at 0x20f0017dc50>,
  <matplotlib.patches.Wedge at 0x20f0002f450>],
 [Text(0.4012472706842742, 1.0242073167910983, 'Yes'),
  Text(-0.4012473665774821, -1.024207279223613, 'No')],
 [Text(0.2188621476459677, 0.5586585364315081, '38%'),
  Text(-0.21886219995135386, -0.5586585159401525, '62%')])
```

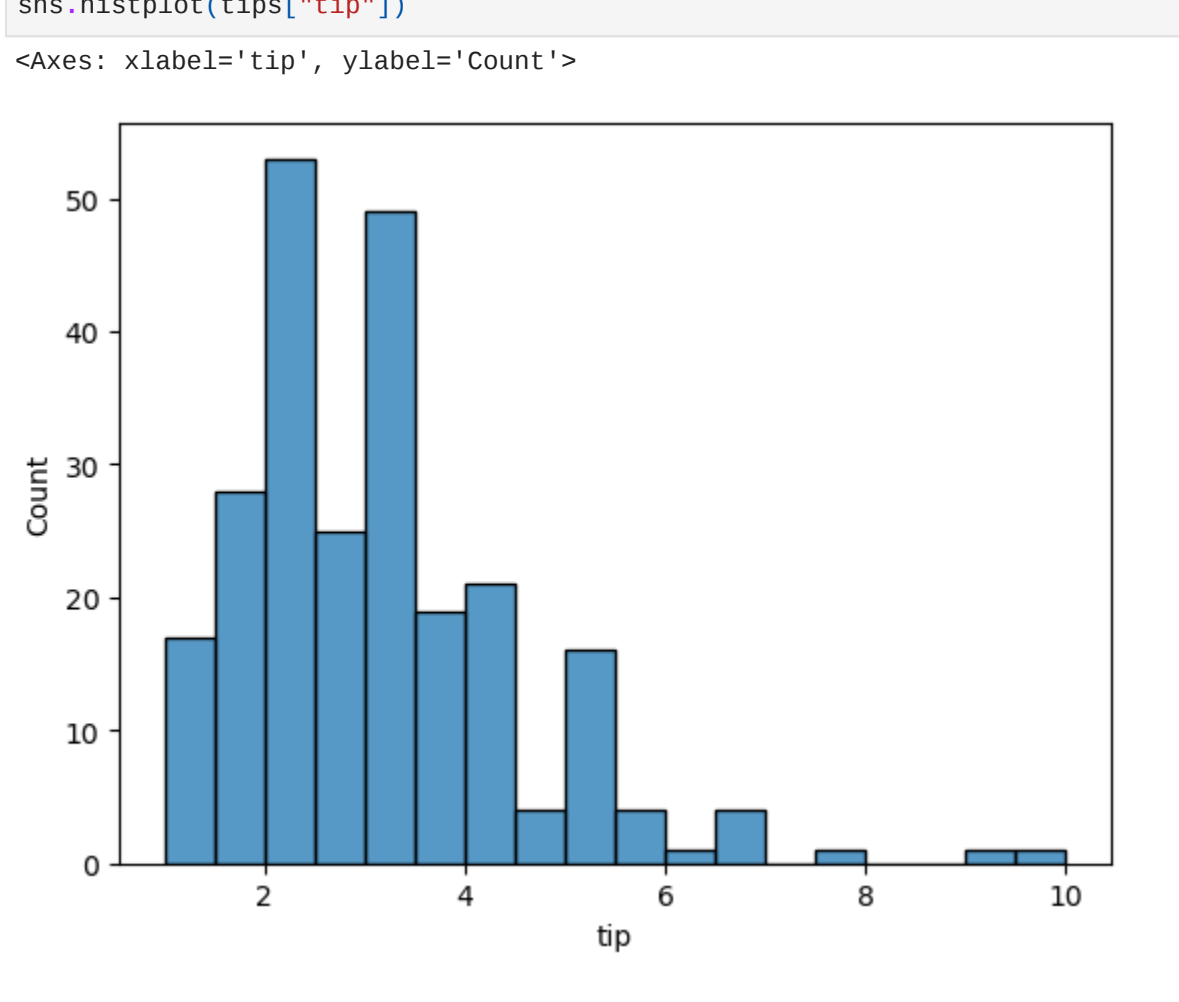


4.Histogram

A histogram is an approximate representation of the distribution of numerical data. The data is divided into non-overlapping intervals called bins and buckets. A rectangle is erected over a bin whose height is proportional to the number of data points in the bin. Histograms give a feel of the density of the distribution of the underlying data

In [10]:
```
sns.histplot(tips["tip"])
```

Out[10]: `<Axes: xlabel='tip', ylabel='Count'>`

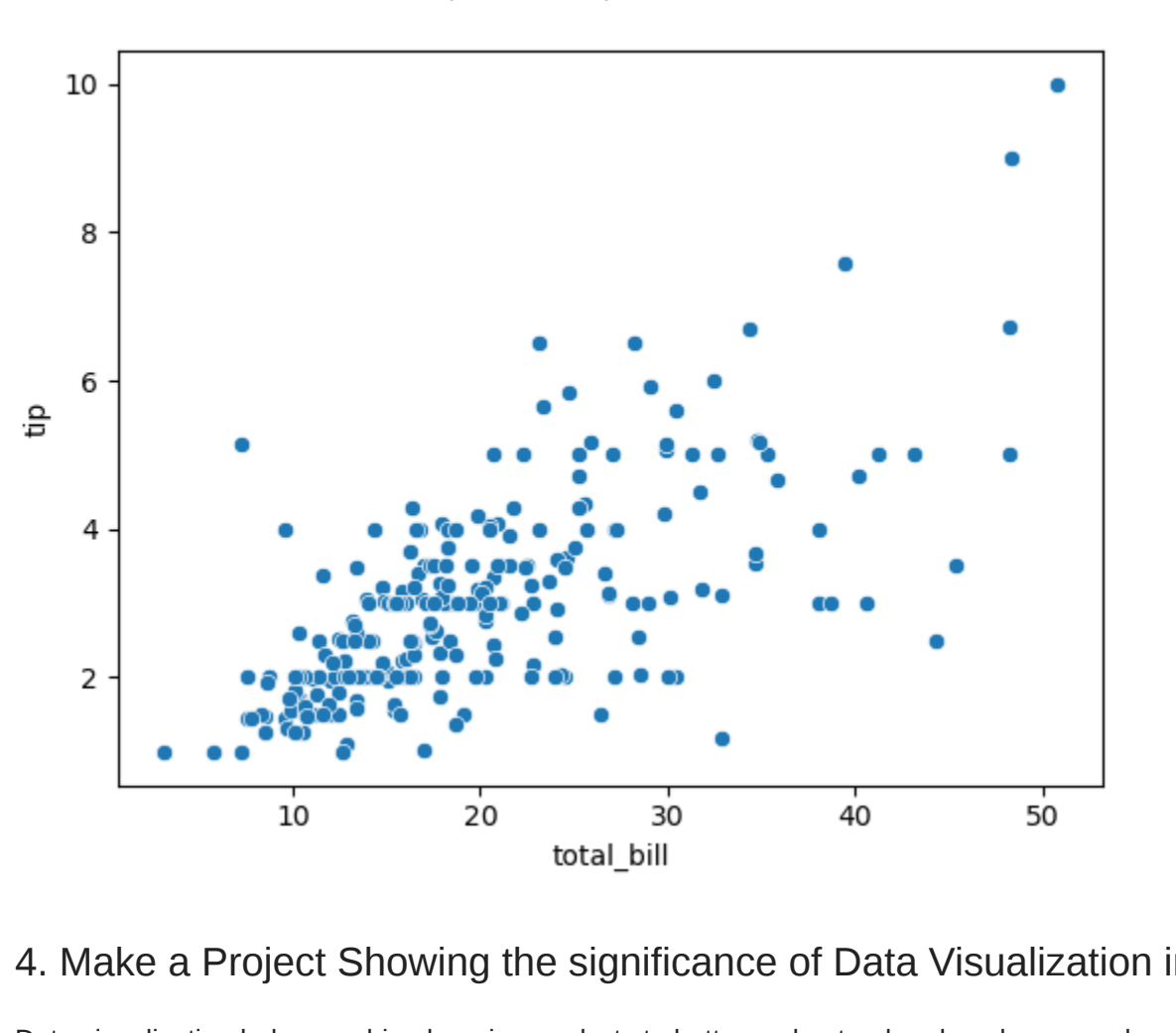


5.Scatter

Scatterplot can be used with several semantic groupings which can help to understand well in a graph. They can plot two-dimensional graphics that can be enhanced by mapping up to three additional variables while using the semantics of hue, size, and style parameters. All the parameter control visual semantic which are used to identify the different subsets. Using redundant semantics can be helpful for making graphics more accessible

In [11]:
```
sns.scatterplot(data=tips,y="tip",x="total_bill")
```

Out[11]: `<Axes: xlabel='total_bill', ylabel='tip'>`



4. Make a Project Showing the significance of Data Visualization in machine learning.

Data visualization helps machine learning analysts to better understand and analyze complex data sets by presenting them in an easily understandable format. Data visualization is an essential step in data preparation and analysis as it helps to identify outliers, trends, and patterns in the data that may be missed by other forms of analysis.

With the increasing availability of big data, it has become more important than ever to use data visualization techniques to explore and understand the data. Machine learning algorithms work best when they have high-quality and clean data, and data visualization can help to identify and remove any inconsistencies or anomalies in the data.

In [12]:
```
# chinese automobile company  aspires to enter the US market
```

In [13]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [15]:
```
tips
```

Out[15]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

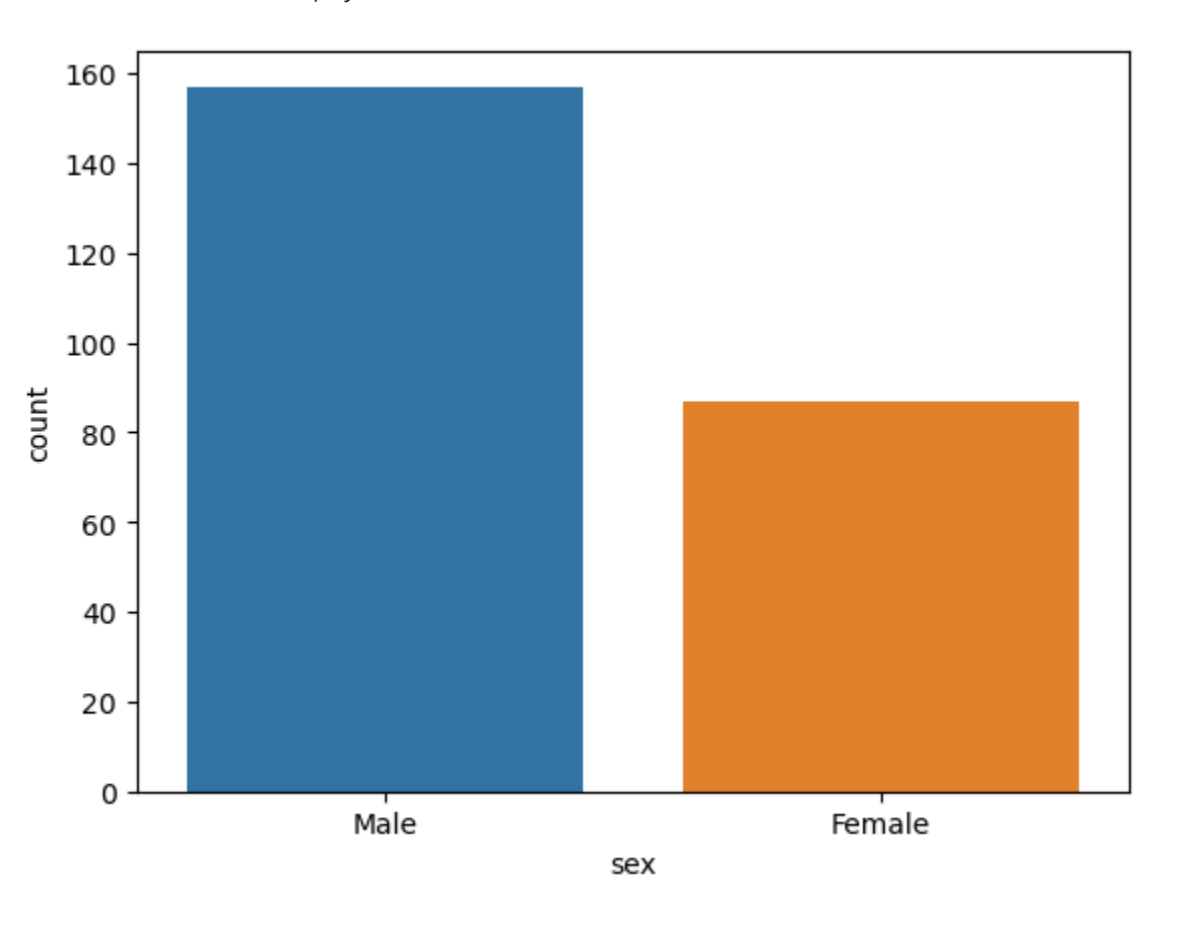
In [54]:
```
tips.head()
```

Out[54]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

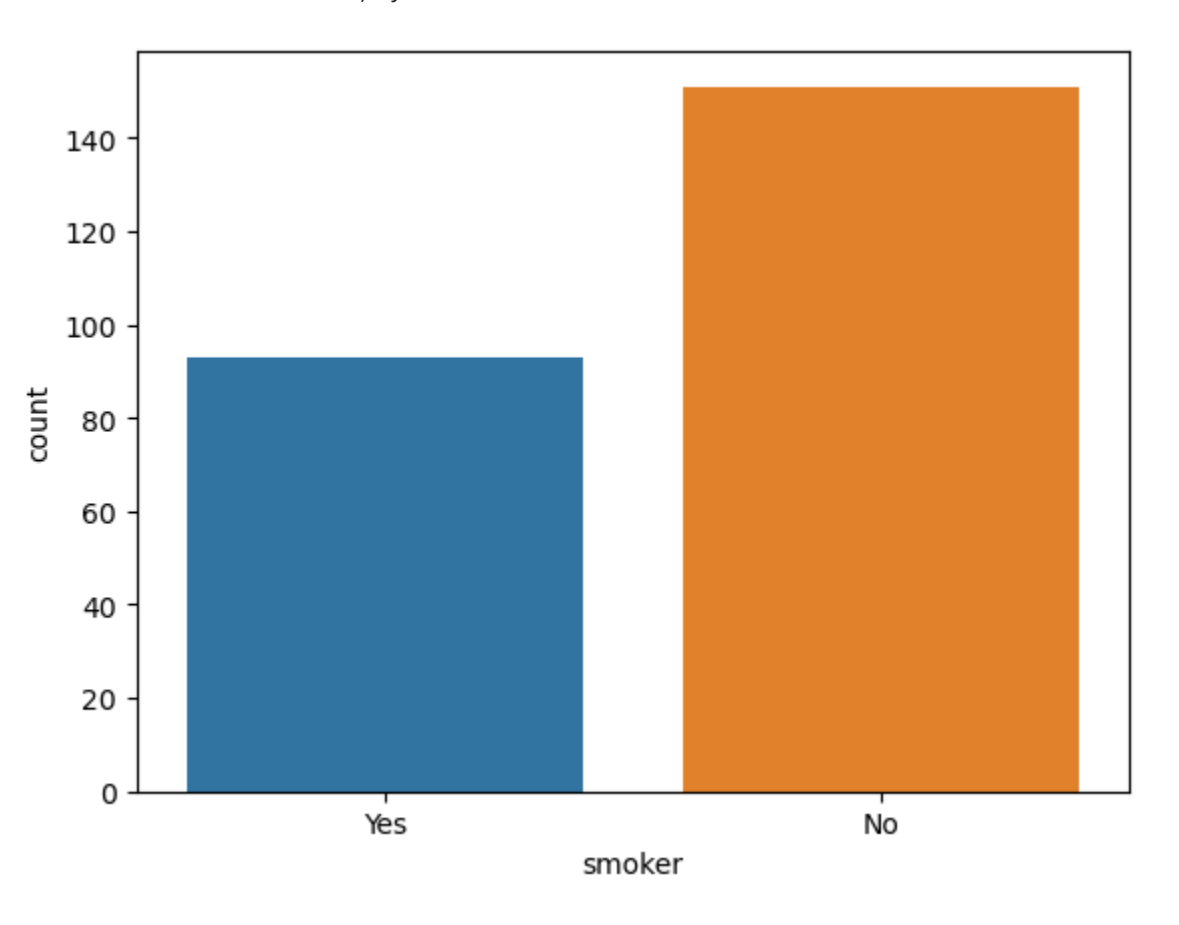
In [55]:
```
sns.countplot(x='sex',data=tips)
```

Out[55]: `<Axes: xlabel='sex', ylabel='count'>`



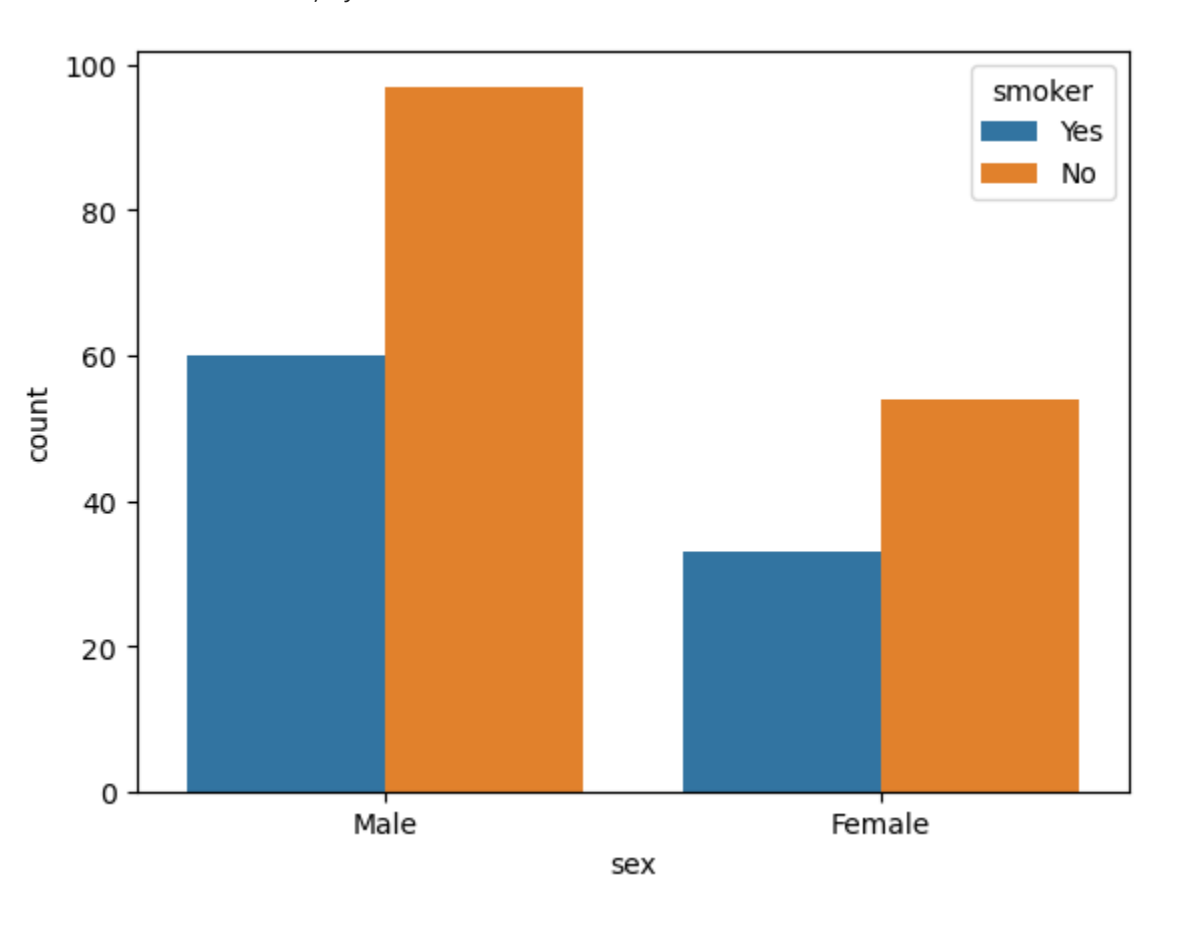
In [56]:
```
sns.countplot(x='smoker',data=tips)
```

Out[56]: `<Axes: xlabel='smoker', ylabel='count'>`



In [57]:
```
sns.countplot(x='sex',data=tips,hue="smoker")
```

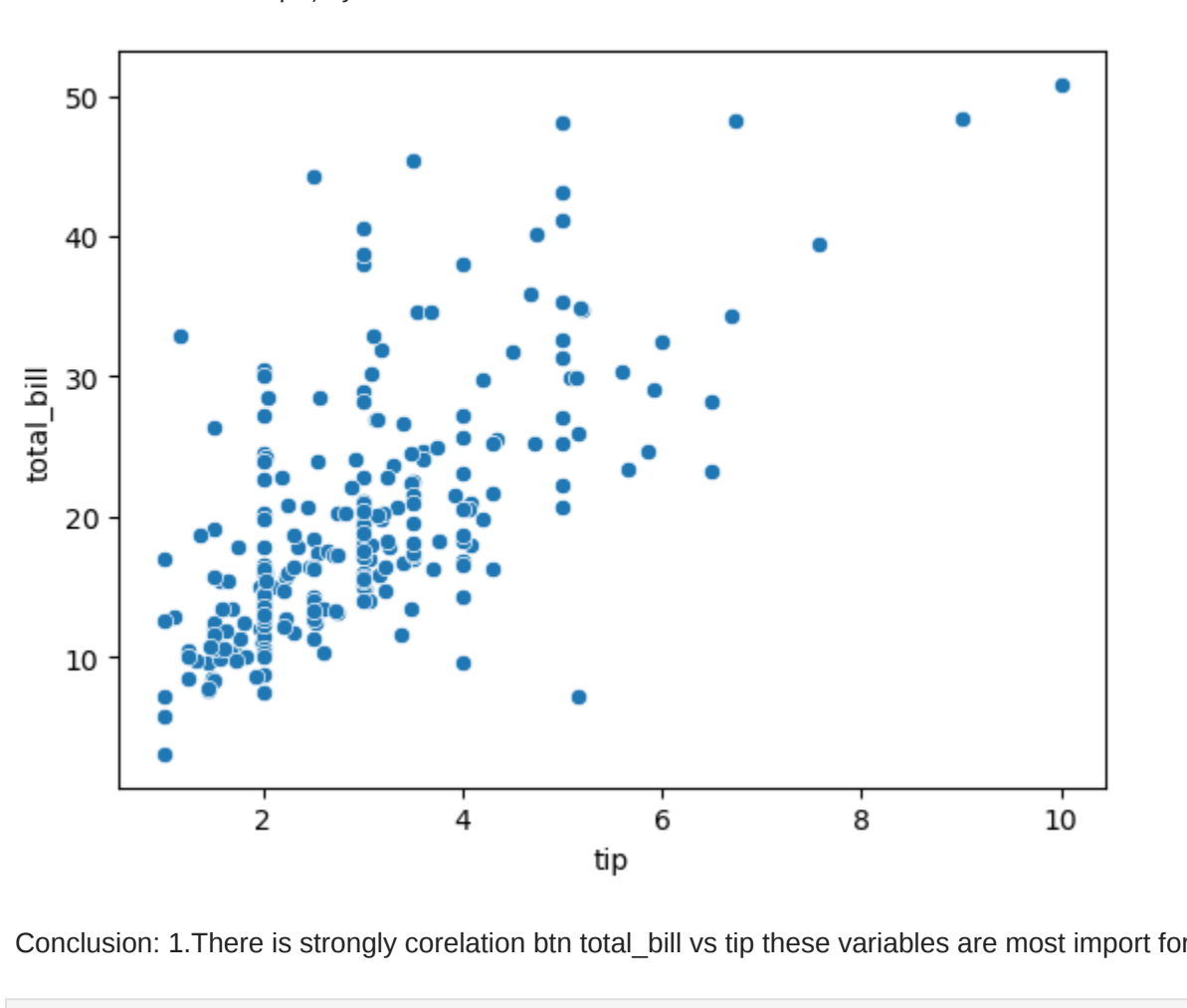
Out[57]: `<Axes: xlabel='sex', ylabel='count'>`



Conclusion: 1.male are pay mostly 2.non smoker paid more than smoker 3.in female non smoker paid mostly

In [61]:
```
sns.scatterplot(x='tip',y="total_bill",data=tips)
```

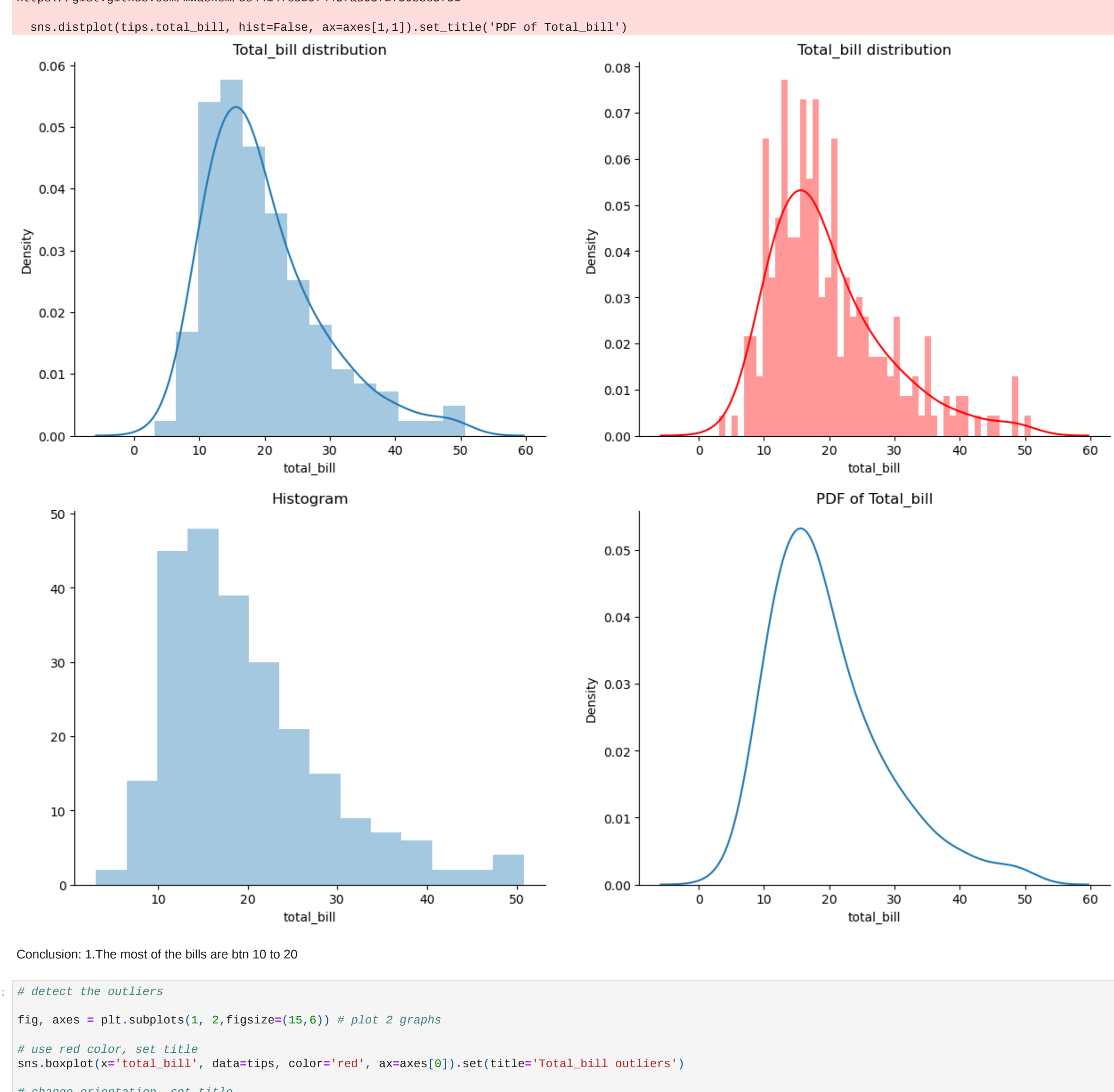
Out[61]: `<Axes: xlabel='tip', ylabel='total_bill'>`



Conclusion: 1.There is strongly corelation btn total_bill vs tip these variables are most import for ML

In [63]:
```
fig, axes = plt.subplots(2, 2, figsize=(15,12)) # plot 4 graphs

# histogram and density function, set title
sns.distplot(tips.total_bill, ax=axes[0,0]).set_title('Total_bill distribution')

#set number of bins and color, set title
sns.distplot(tips.total_bill, bins=50, color='r', ax=axes[0,1]).set_title('Total_bill distribution')

# only histogram, without density function, set title
sns.distplot(tips.total_bill, kde=False, ax=axes[1,0]).set_title('Histogram')

# only density function, without histogram, set title
sns.distplot(tips.total_bill, hist=False, ax=axes[1,1]).set_title('PDF of Total_bill')
sns.despine() # no top and right axes spine
```

```
C:\Users\adina\AppData\Local\Temp\ipykernel_51288\2178433952.py:4: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(tips.total_bill, ax=axes[0,0]).set_title('Total_bill distribution')
C:\Users\adina\AppData\Local\Temp\ipykernel_51288\2178433952.py:7: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(tips.total_bill, bins=50, color='r', ax=axes[0,1]).set_title('Total_bill distribution')
C:\Users\adina\AppData\Local\Temp\ipykernel_51288\2178433952.py:10: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(tips.total_bill, kde=False, ax=axes[1,0]).set_title('Histogram')
C:\Users\adina\AppData\Local\Temp\ipykernel_51288\2178433952.py:13: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(tips.total_bill, hist=False, ax=axes[1,1]).set_title('PDF of Total_bill')
```

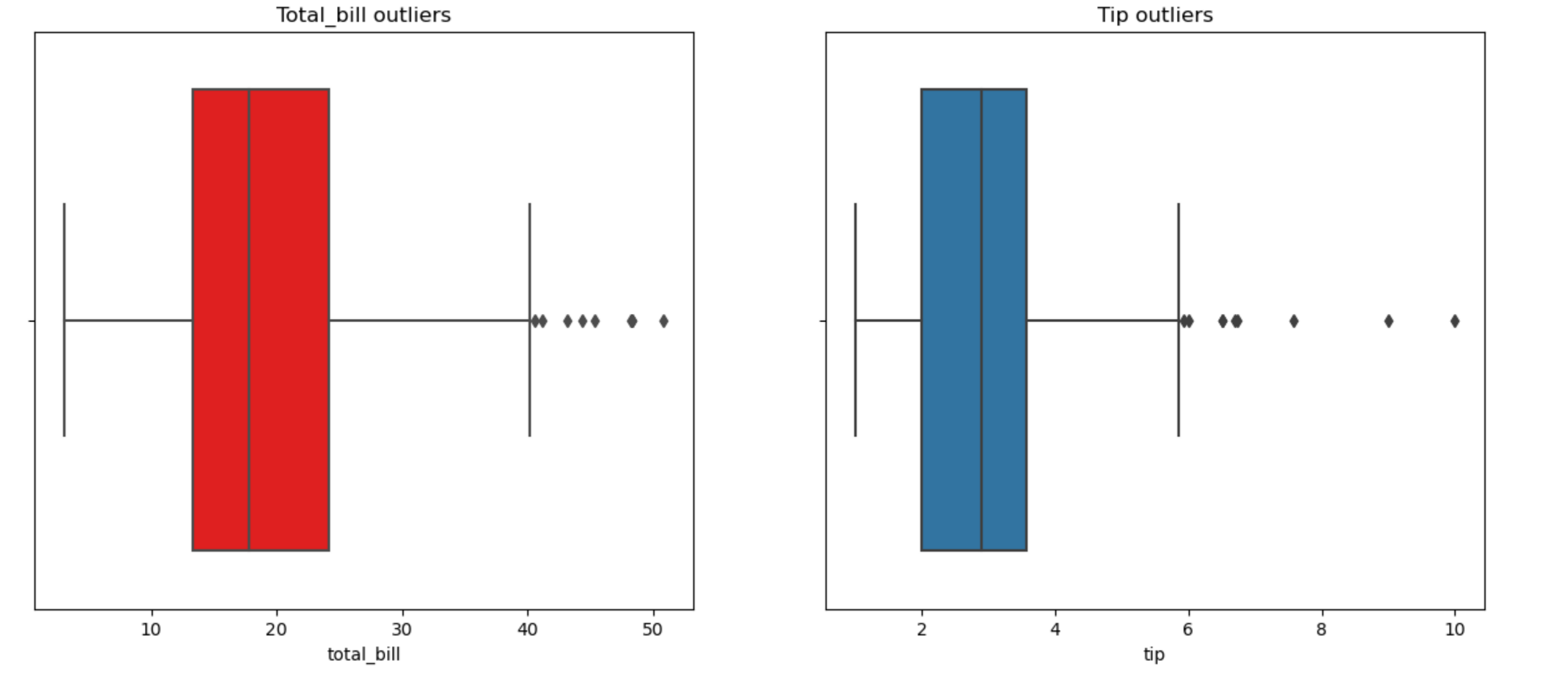


Conclusion: 1.The most of the bills are btn 10 to 20

In [65]:
```
# detect the outliers

fig, axes = plt.subplots(1, 2,figsize=(15,6)) # plot 2 graphs

# use red color, set title
sns.boxplot(x='total_bill', data=tips, color='red', ax=axes[0]).set(title='Total_bill outliers')

# change orientation, set title
sns.boxplot(x='tip', data=tips, orient='v', ax=axes[1]).set_title('Tip outliers')
```

```
C:\Users\adina\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1599: UserWarning: Vertical orientation ignored with only `x` specified.
  warnings.warn(single_var_warning.format("Vertical", "x"))
```

Out[65]: `Text(0.5, 1.0, 'Tip outliers')`



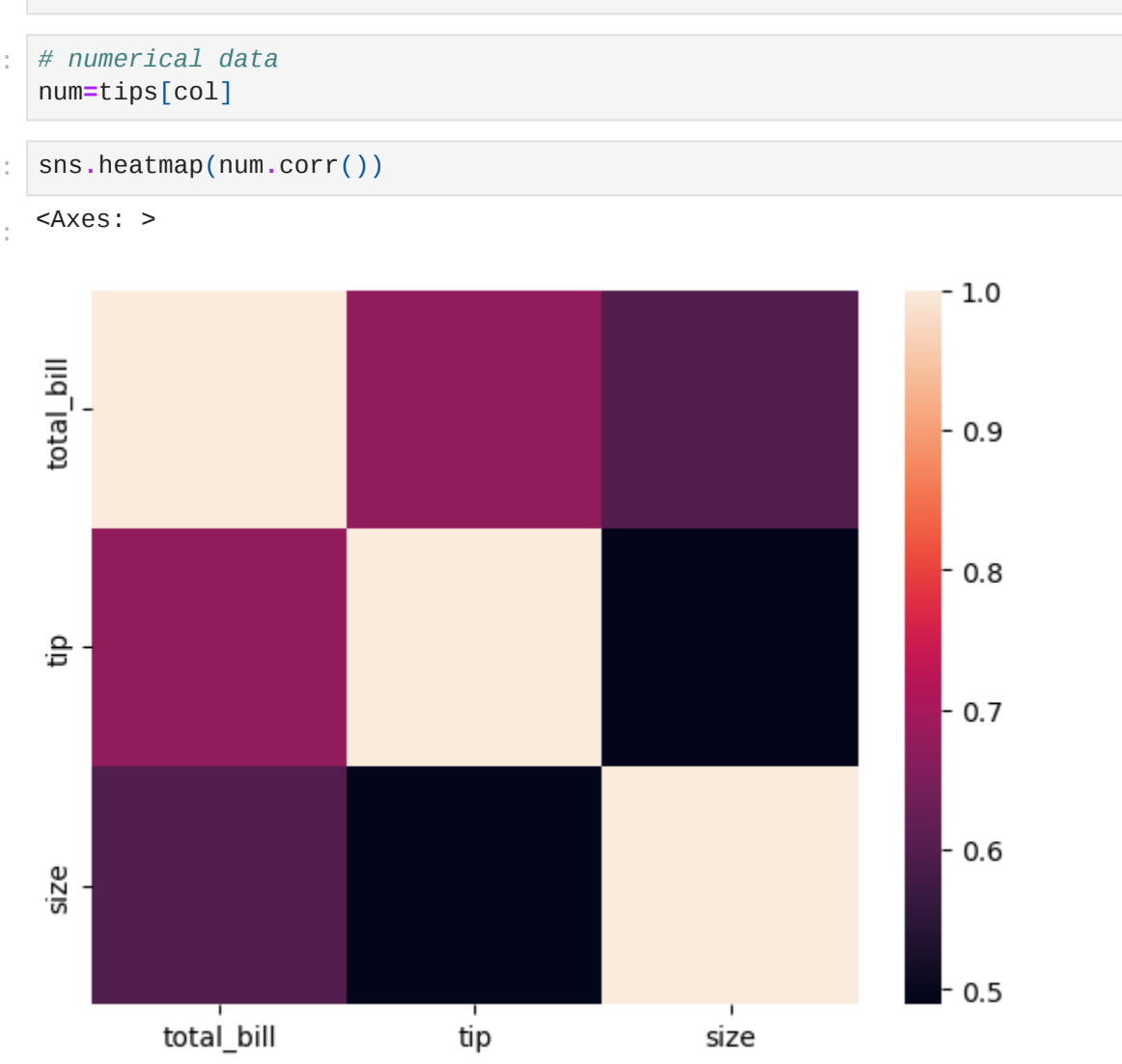
Conclusion: 1.There is outlier in total_bill and tips.On the basis of this we will take or remove the outliers

In [43]:
```
col=tips.columns[tips.dtypes!='category']
```

In [47]:
```
# numerical data
num=tips[col]
```

In [52]:
```
sns.heatmap(num.corr())
```

Out[52]: `<Axes: >`



conclusin: 1.The coralation between total bill and tip is positive coralation

In [ ]: