SEABORN KRISH NAIK

June 29, 2022

[2]:
```
import seaborn as sns
```

[3]:
```
#inbuilt functions:
#load_dataset()-it loads the data set
#go through the notes to understand features..
```

[4]:
```
#seaborn has inbuilt dataset called as 'tips'
df=sns.load_dataset("tips")
print(df.head())
```

```
   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
```

[5]:
```
#heat correlation map:
#A correlation heat map used colour cells ,typically in a monochromatic scale .
 ↪It is very important in feature engineering.
#seaborn has a property called as heatmap which helps us to find the␣
 ↪correlation between each and every feature in a
#DataFrame.
```

[6]:
```
#there is a difference between correlation and covarience..to understand it..we␣
 ↪can understand the interrelationship between each feature
#using correlation but using covarience we cannot compare between features.
#and the correlation values occur from -1 to +1
```

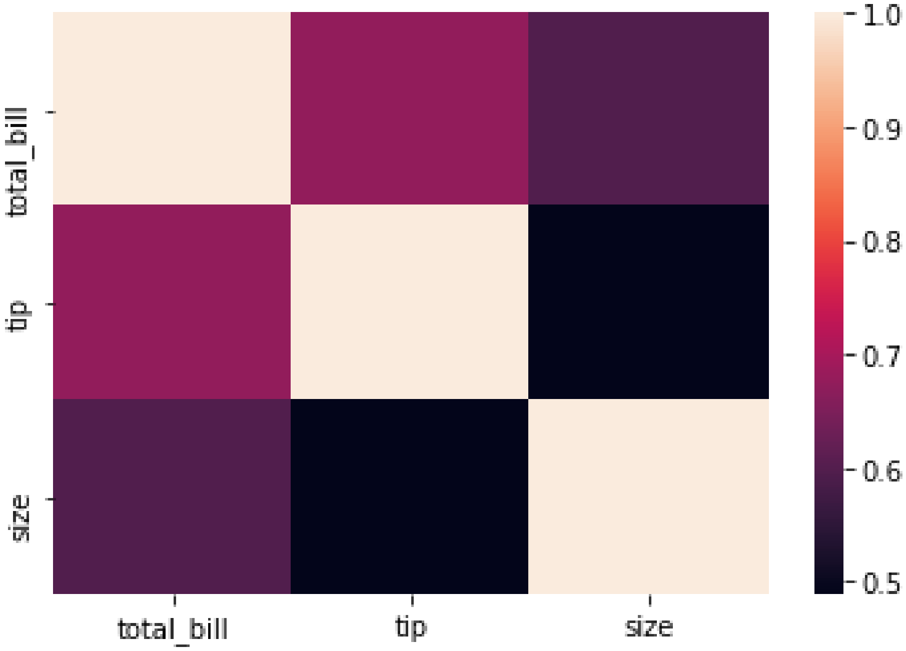
[7]:
```
df.corr()
#go through the notes to understand this data better.
```

[7]:
```
            total_bill       tip      size
total_bill    1.000000  0.675734  0.598315
tip           0.675734  1.000000  0.489299
size          0.598315  0.489299  1.000000
```

[8]:
```
#using heatmap we can understand the positive correlation of the above data.
```

[9]:
```
sns.heatmap(df.corr())
```

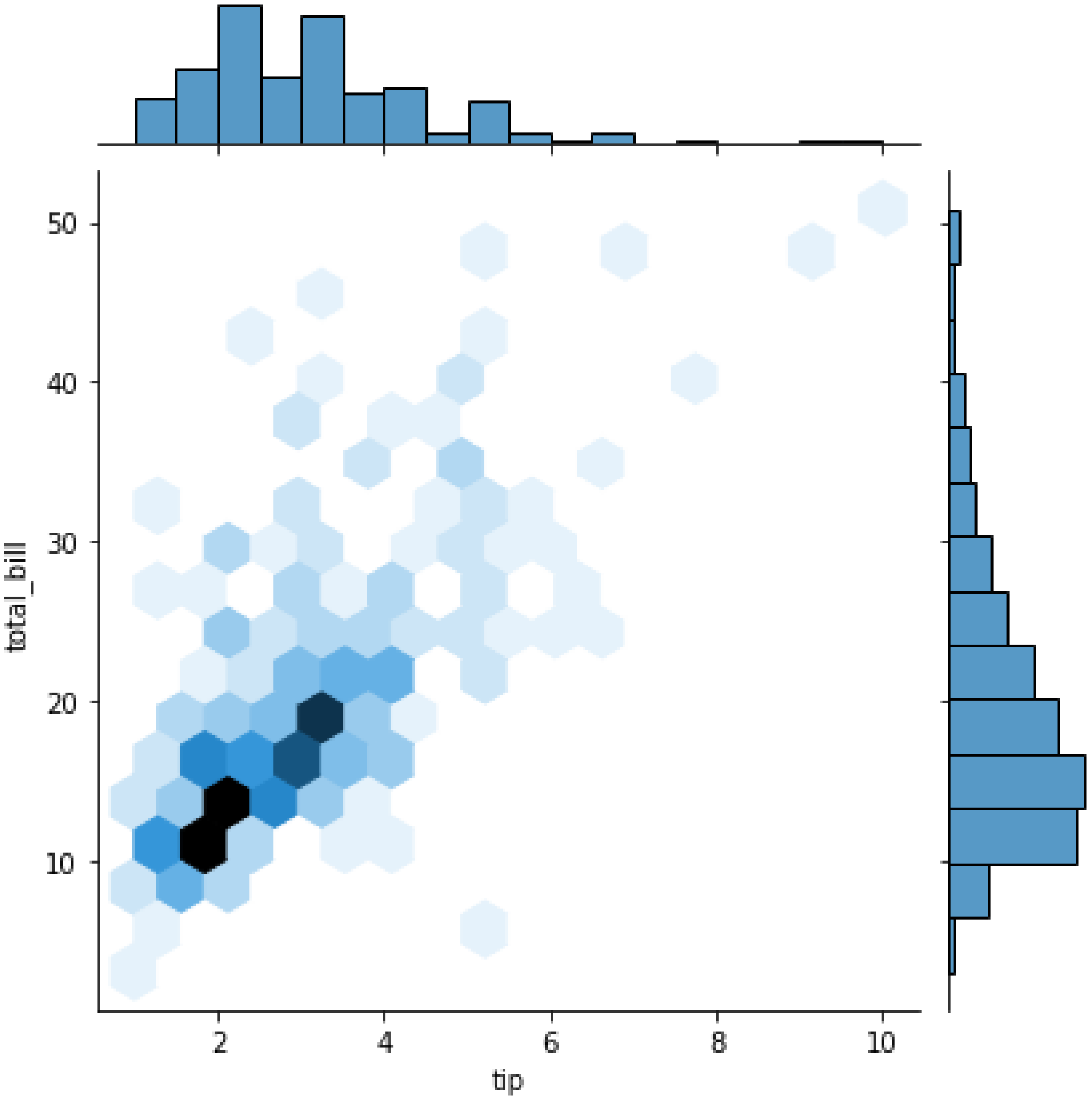
[9]:
```
<AxesSubplot:>
```



[10]:
```
#from the above dataframe you can see the correlation between tip and size is␣
↪somewhere around 0.489299..so its marked
#u=in black colour.
#correlation is very very important in exploratory data analysis.
#this is all about the heat map.
```

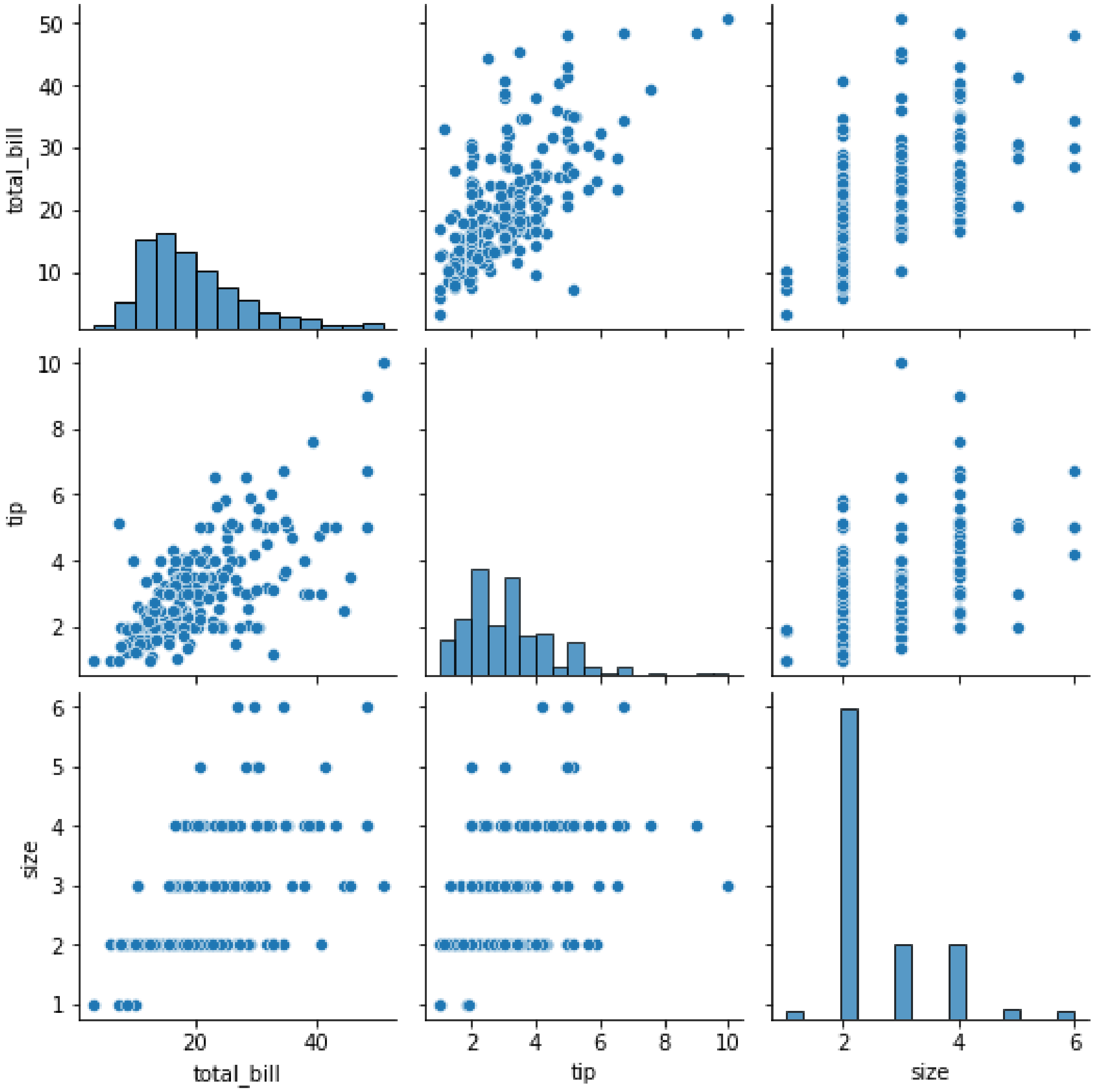
[11]:
```
#moving on to jointplot.
#jointplot - helps us to do univarient analysis
#we are only going to take two features over here.
sns.jointplot(x="tip",y="total_bill",data=df,kind="hex")
#so we are taking two features for the analysis one is tip and the other one is␣
↪total_bill for analysis.
#go through the notes to understand this data better.
```

[11]:
```
<seaborn.axisgrid.JointGrid at 0x2c56ef7bf70>
```



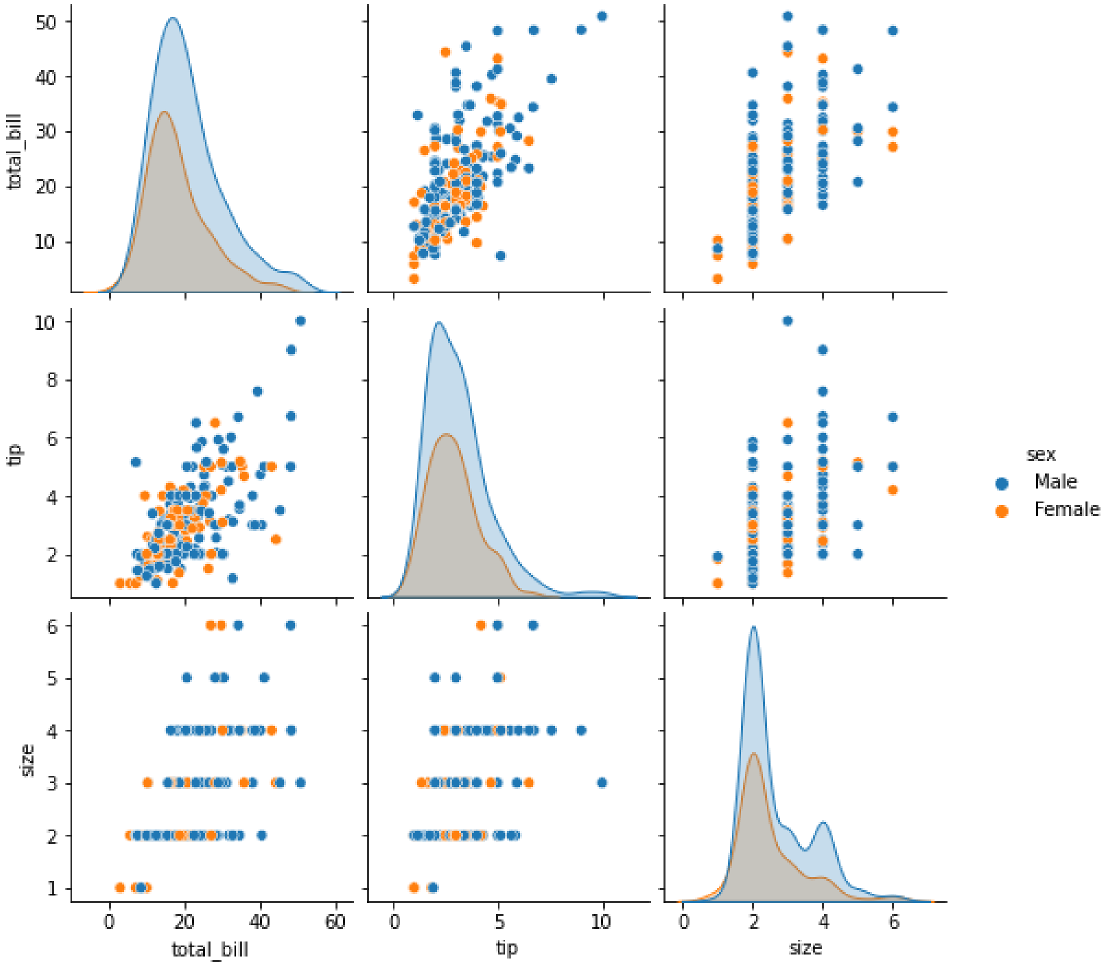
```
[12]: #pair plot:
      #we will use this kind of plotting when we use more than 2 features
      #pair plot is also called as scatter plot , in which one variable in the same␣
       ↪data is matched with another variable in
      #another variable
      sns.pairplot(df)
```

```
[12]: <seaborn.axisgrid.PairGrid at 0x2c570188f70>
```



```
[13]: #if we want to scatterplot considering another feature called sex then we have␣
       ↪to follow the below code
      sns.pairplot(df,hue="sex")
```

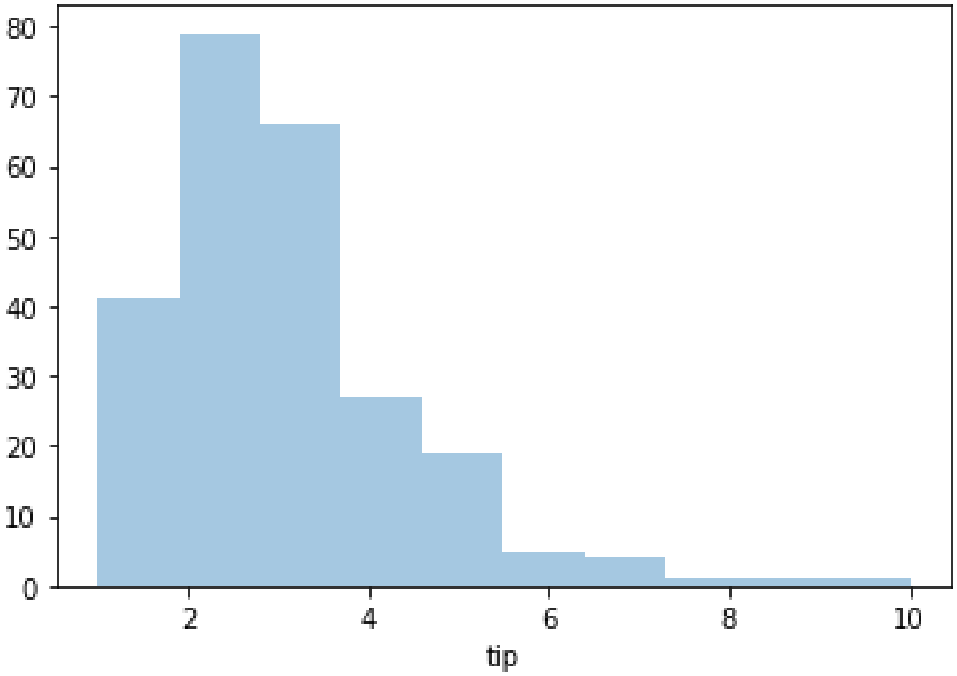
```
[13]: <seaborn.axisgrid.PairGrid at 0x2c5701bdc40>
```



[14]:
```
#using distplot:
#distplot helps us to check the distribution of the columns feature
sns.distplot(df['tip'],kde=False,bins=10)
```

```
C:\Users\91936\anaconda3\lib\site-packages\seaborn\distributions.py:2557:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```

[14]:
```
<AxesSubplot:xlabel='tip'>
```

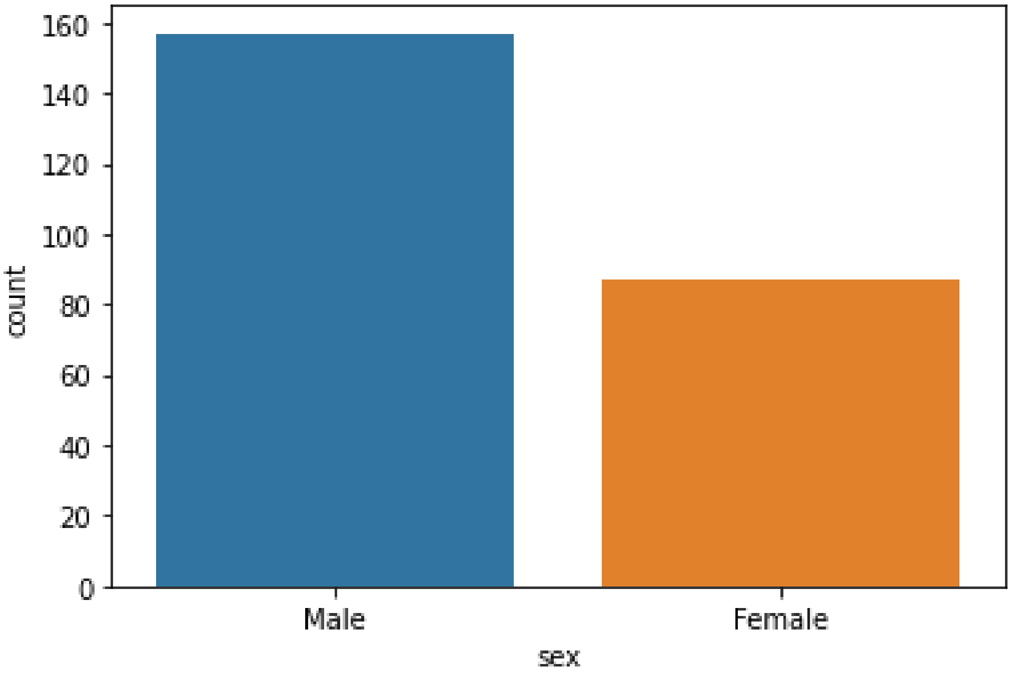


[15]:
```
#CATEGORICAL PLOTS
#1) BOXPLOT
#2) VIOLINPLOT
#3) COUNTPLOT
#4) BAR PLOT
```

[16]:
```
#countplot
#it shows the count of observations in each categorical bin using bars.
sns.countplot('sex',data=df)
```

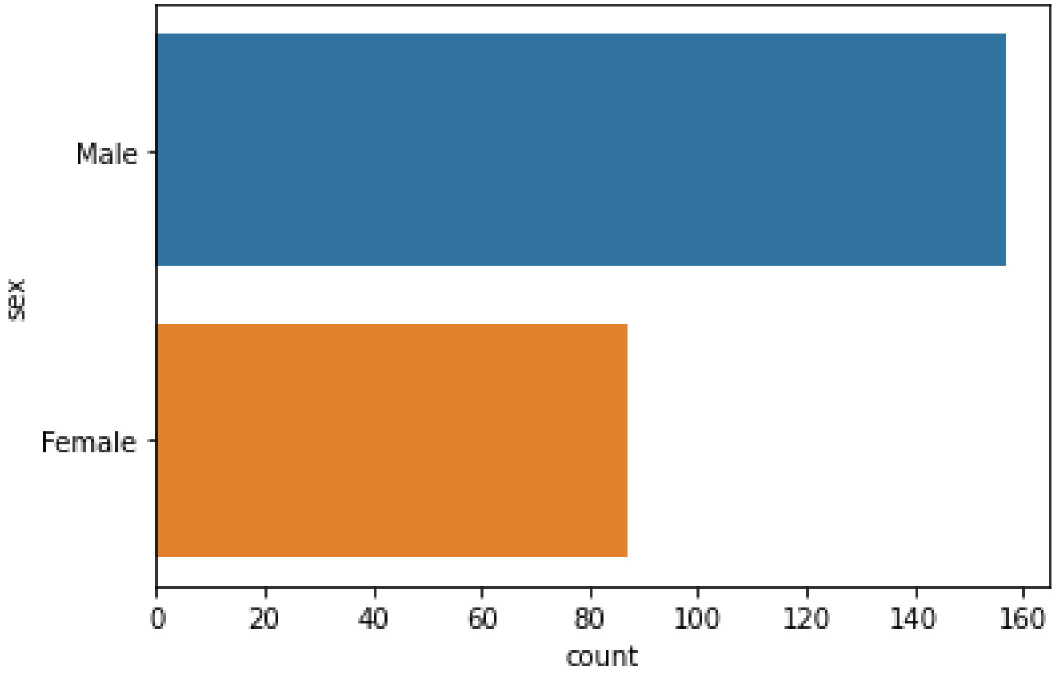
```
C:\Users\91936\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

[16]:
```
<AxesSubplot:xlabel='sex', ylabel='count'>
```



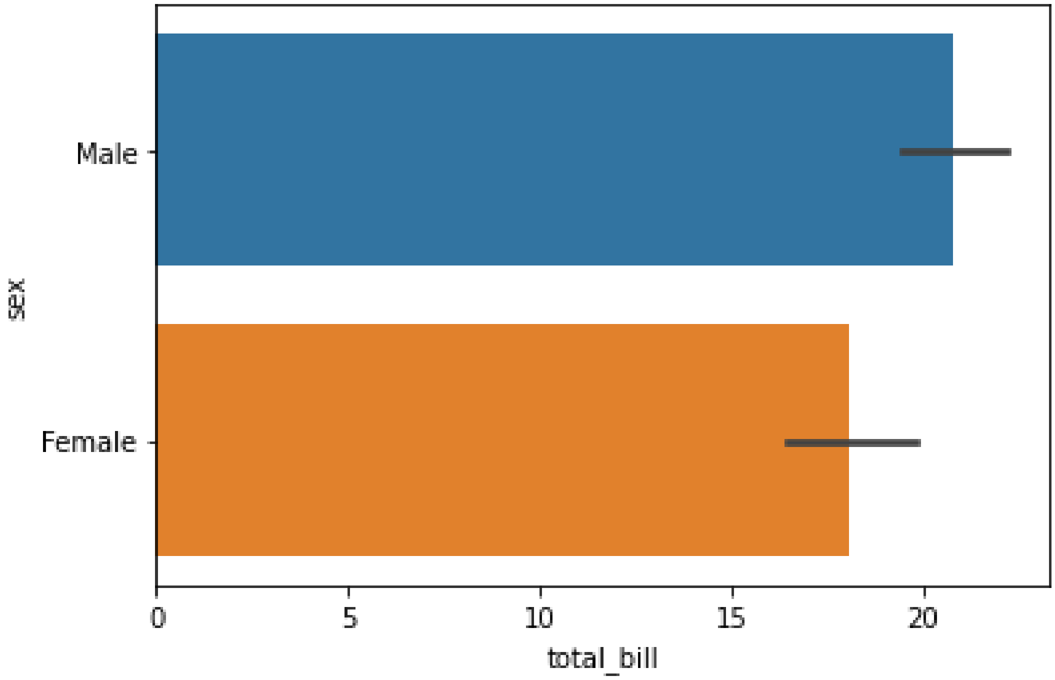
```
[17]: #now if you want to plot in y axis
      sns.countplot(y='sex',data=df)
```

```
[17]: <AxesSubplot:xlabel='count', ylabel='sex'>
```



```
[18]: #using barplot
      #both countplot and barplot are similar but we have to give x and y values
      sns.barplot(x="total_bill",y="sex",data=df)
```

```
[18]: <AxesSubplot:xlabel='total_bill', ylabel='sex'>
```

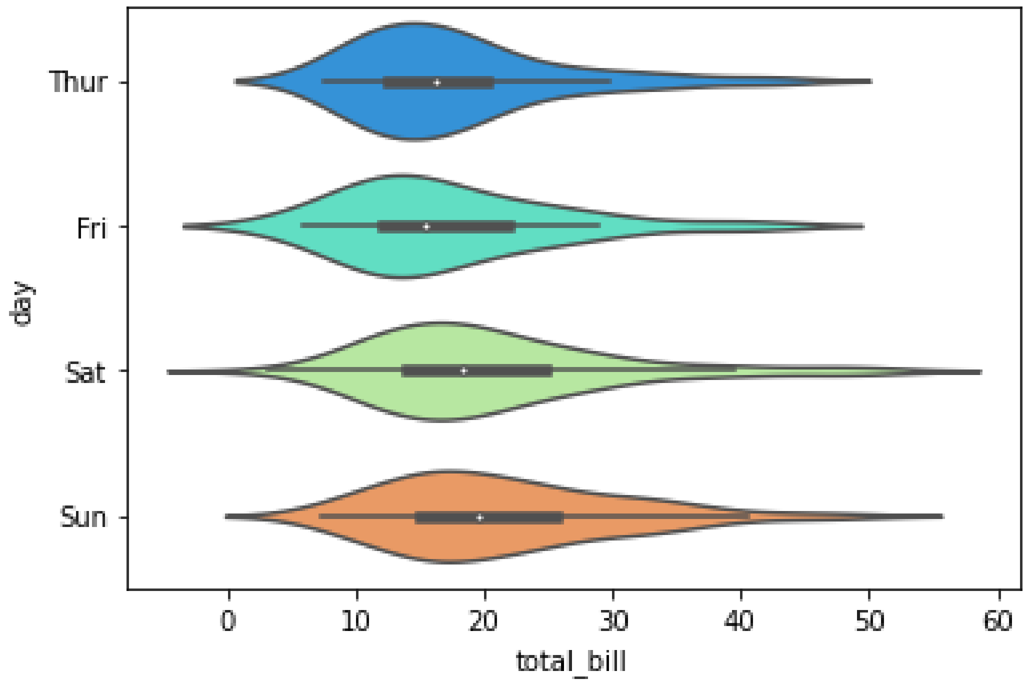


```
[ ]: #violin plot
```

```
[ ]: #it helps us to see both the distribution of data in terms of kernel density␣
      ↪estimation and the box plot
```

```
[19]: sns.violinplot(x="total_bill",y="day",data=df,palette="rainbow")
```

```
[19]: <AxesSubplot:xlabel='total_bill', ylabel='day'>
```


Thur
Fri
Sat
Sun
day
0
10
20
30
40
50
60
total_bill

[ ]: