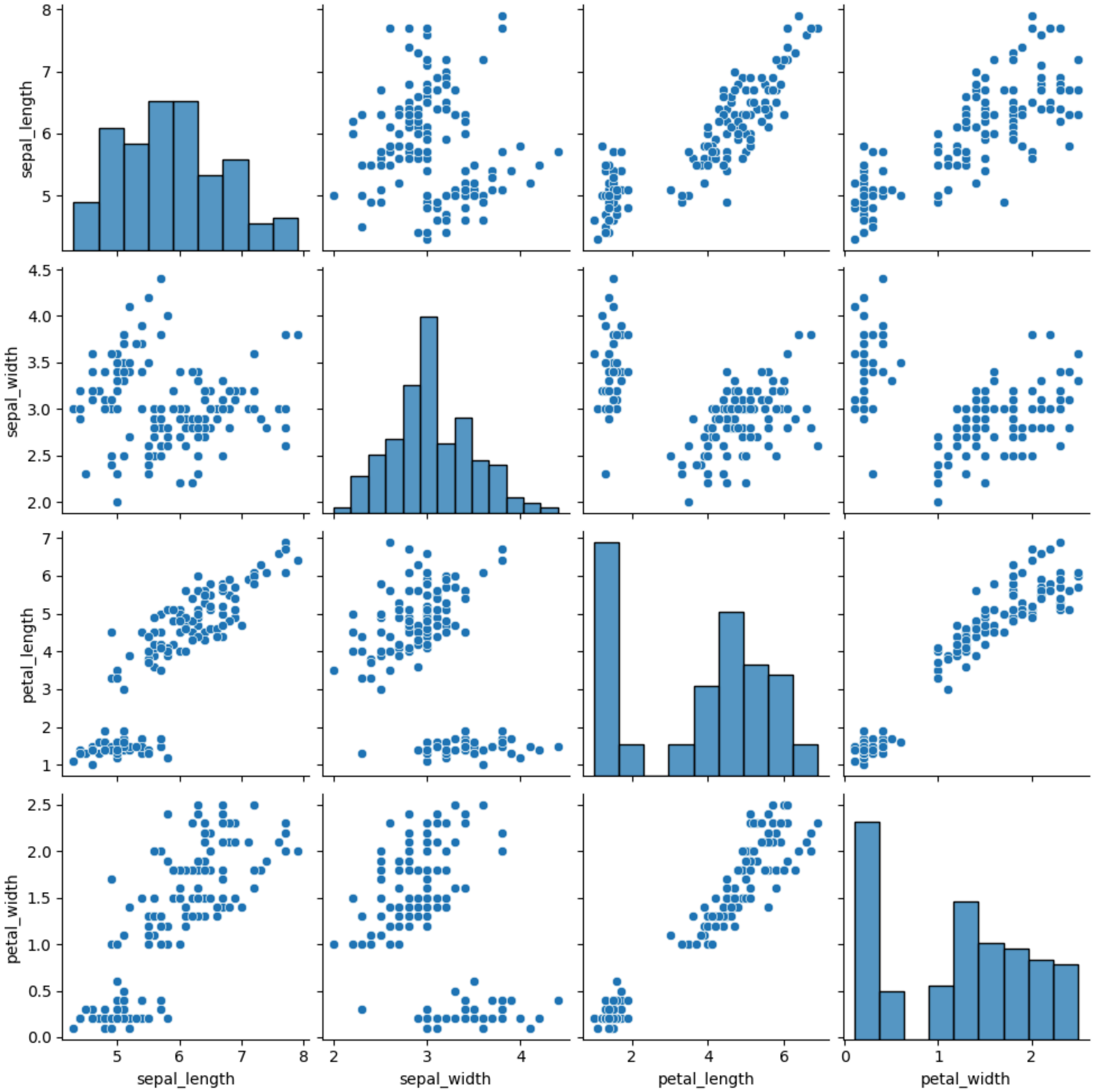
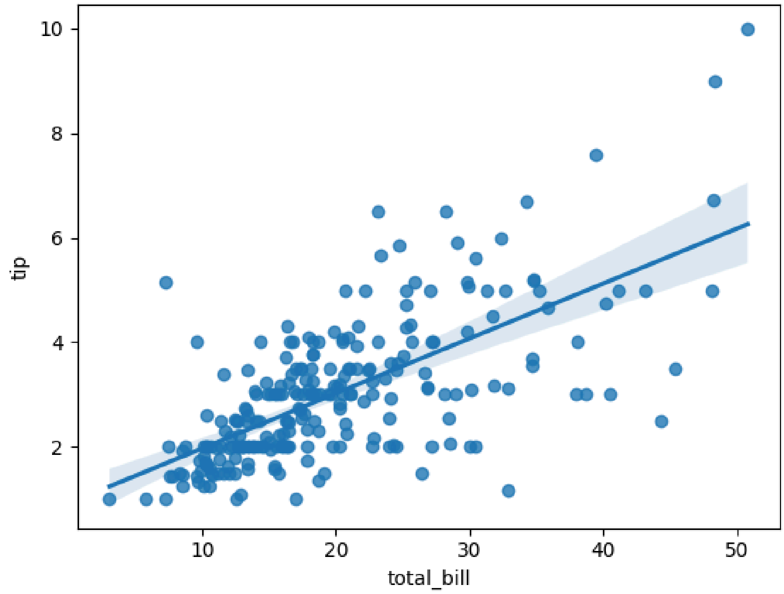
```
import matplotlib.pyplot as plt
import seaborn as sns
df = sns.load_dataset('iris')
#without regression
sns.pairplot(df,kind="scatter")
plt.show()
```

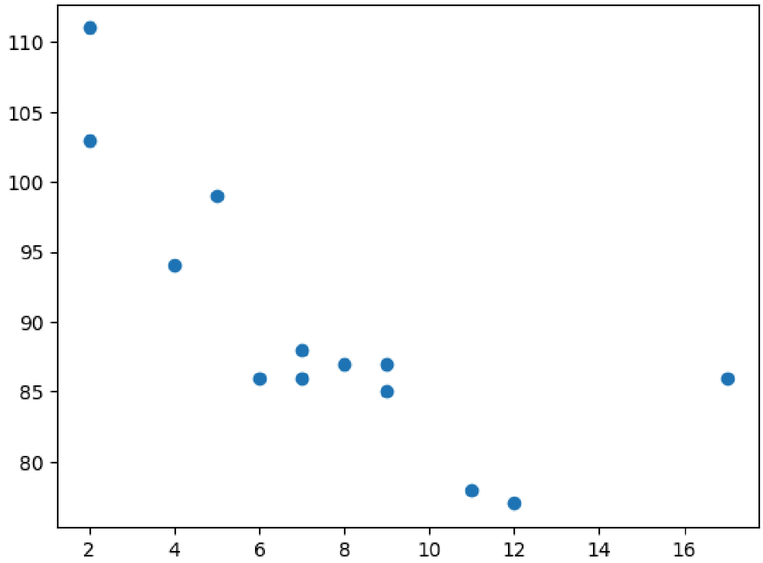


```
import seaborn as sb
from matplotlib import pyplot as plt
df=sb.load_dataset('tips')
sb.regplot(x= "total_bill", y = "tip" , data =df)
plt.show()
```



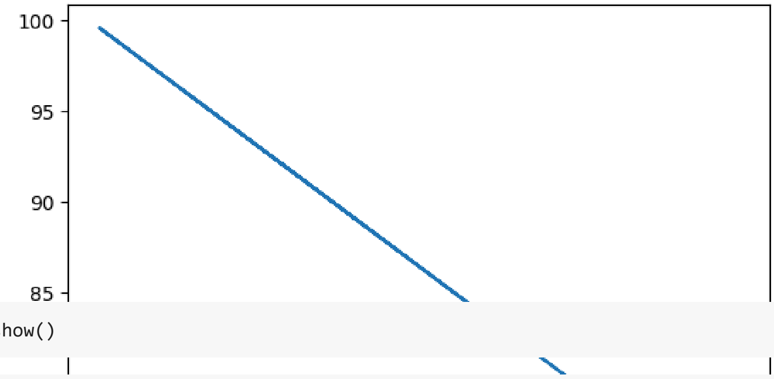
```
import matplotlib.pyplot as plt
from scipy import stats

#create an a array for X and Y axis
x =[5,7,8,7,2,17,2,9,4,11,12,9,6]
y =[99,86,87,88,111,86,103,87,94,78,77,85,86]

slope,intercept,r,p, std_err = stats.linregress(x,y)
def myfunc(x):
   return slope * x + intercept
mymodel=list(map(myfunc,x))


#Draw the original scatter plot
plt.scatter(x,y)
```

```
<matplotlib.collections.PathCollection at 0x7bdc60ad3dc0>
```



```
plt.plot(x,mymodel)
```

```
[<matplotlib.lines.Line2D at 0x7bdc60b028f0>]
```



```
plt.show()
```

```
import numpy as nmp
import matplotlib.pyplot as plt
def estimate_coeff(p,q):
#Here, we will estimate the total numbers of points or observation
  n1= nmp.size(p)
#Now, we will calculate the mean of a and b vector
  m_p = nmp.mean(p)
  m_q = nmp.mean(q)
#Here,we wil calculate the cross deviation and deviation about a
  SS_pq = nmp.sum(q * p) - n1 * m_q * m_p
  SS_pp = nmp.sum(q * p) - n1 * m_p * m_q
#Here , we will calculate the regression coeffients
  b_1 = SS_pq / SS_pp
  b_0 = m_q - b_1 * m_p
  return(b_0, b_1)
def plot_regression_line(p,q,b):
   mtplt.scatter(p,q,color = "m",marker = "o", s=30)
   q_pred = b[0] +b[1] *p
#here we will plot the regression line
   mtplt.plot(p,q_pred,color = "g")
#here,we will put the labels
   mtplt.xlabel('p')
   mtplt.ylabel('q')
```

Start coding or generate with AI.

Start coding or generate with AI.