In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt

```

In [5]:

```
df = pd.read_csv("Food-Truck(For Linear Regression Program).csv")
df
```

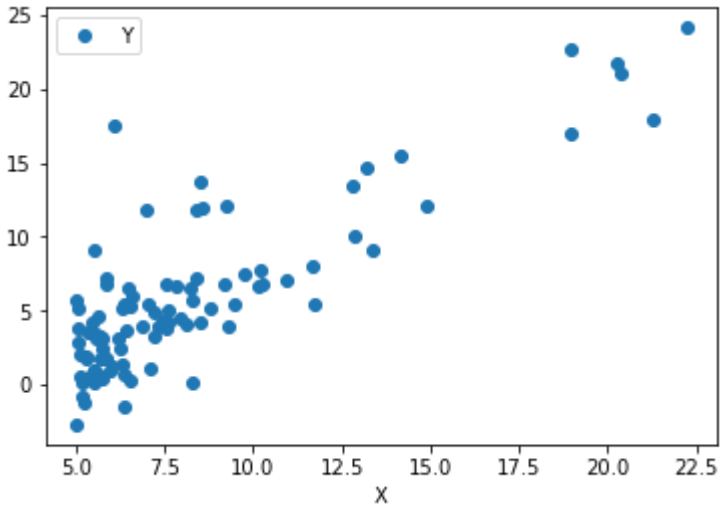
Out[5]:

	X	Y
0	6.1101	17.59200
1	5.5277	9.13020
2	8.5186	13.66200
3	7.0032	11.85400
4	5.8598	6.82330
...	...	...
92	5.8707	7.20290
93	5.3054	1.98690
94	8.2934	0.14454
95	13.3940	9.05510
96	5.4369	0.61705

97 rows × 2 columns

In [6]:

```
df.plot(x="X", y="Y", style="o")
plt.show()
```



In [7]:

```
x_mean = df["X"].mean()
y_mean = df["Y"].mean()
print(x_mean,y_mean)
```

```
8.159799999999999 5.839135051546393
```

In [8]:

```
df["x"] = df["X"] - x_mean
df["y"] = df["Y"] - y_mean
df["x*y"] = df["x"] * df["y"]
df["x^2"] = df["x"]**2
df["y^2"] = df["y"]**2
df
```

Out[8]:

	X	Y	x	y	x*y	x^2	y^2
0	6.1101	17.59200	-2.0497	11.752865	-24.089847	4.201270	138.129834
1	5.5277	9.13020	-2.6321	3.291065	-8.662412	6.927950	10.831108
2	8.5186	13.66200	0.3588	7.822865	2.806844	0.128737	61.197216
3	7.0032	11.85400	-1.1566	6.014865	-6.956793	1.337724	36.178600
4	5.8598	6.82330	-2.3000	0.984165	-2.263579	5.290000	0.968581
...	...	...	...	...	...	...	...
92	5.8707	7.20290	-2.2891	1.363765	-3.121794	5.239979	1.859855
93	5.3054	1.98690	-2.8544	-3.852235	10.995820	8.147599	14.839715
94	8.2934	0.14454	0.1336	-5.694595	-0.760798	0.017849	32.428413
95	13.3940	9.05510	5.2342	3.215965	16.833004	27.396850	10.342431
96	5.4369	0.61705	-2.7229	-5.222085	14.219215	7.414184	27.270172

97 rows × 7 columns

In [9]:

```
summation_x_y = df["x*y"].sum()
summation_x_squared = df["x^2"].sum()
summation_y_squared = df["y^2"].sum()
print(summation_x_y, summation_x_squared, summation_y_squared)
```

```
1715.219528539 1437.69585786 2914.8470516572247
```

In [11]:

```
correlation = summation_x_y / (summation_x_squared * summation_y_squared)**0.5
correlation
```

Out[11]:

```
0.8378732325263409
```

In [20]:

```
def getMean(num):
 if len(num) == 0:
  return None
 else:
  sum = 0
  for i in num:
   sum += i
   avg = sum/len(num)
  return avg
```

In [21]:

```
def getStandardDeviation(num):
 if len(num) == 0:
  return 0
 else:
  mean = getMean(num)
  std_deviation = 0
  for i in num:
   std_deviation += (i - mean)**2
  return (std_deviation/len(num))**0.5
```

In [22]:

```
std_deviation_x = getStandardDeviation(df["x"].tolist())
std_deviation_y = getStandardDeviation(df["y"].tolist())
print(std_deviation_x, std_deviation_y)
```

```
3.8498839959227156 5.48178525707763
```

In [23]:

```
m = correlation * (std_deviation_y / std_deviation_x)
m
```

Out[23]:

```
1.1930336441895937
```

In [24]:

```
c = df["Y"].mean() - m * df["X"].mean()
c
```

Out[24]:

```
-3.895780878311852
```

In [31]:

```
df["y_predict"] = m * df["X"] + c
df
```

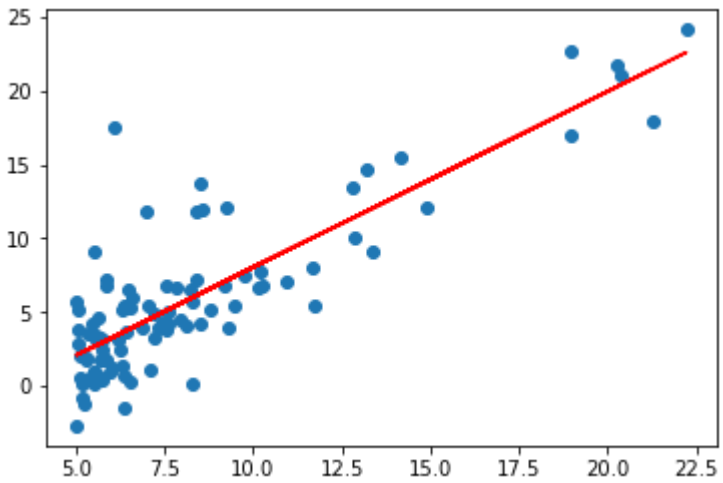
Out[31]:

	X	Y	x	y	x*y	x^2	y^2	y_predict
0	6.1101	17.59200	-2.0497	11.752865	-24.089847	4.201270	138.129834	3.393774
1	5.5277	9.13020	-2.6321	3.291065	-8.662412	6.927950	10.831108	2.698951
2	8.5186	13.66200	0.3588	7.822865	2.806844	0.128737	61.197216	6.267196
3	7.0032	11.85400	-1.1566	6.014865	-6.956793	1.337724	36.178600	4.459272
4	5.8598	6.82330	-2.3000	0.984165	-2.263579	5.290000	0.968581	3.095158
...	...	...	...	...	...	...	...	...
92	5.8707	7.20290	-2.2891	1.363765	-3.121794	5.239979	1.859855	3.108162
93	5.3054	1.98690	-2.8544	-3.852235	10.995820	8.147599	14.839715	2.433740
94	8.2934	0.14454	0.1336	-5.694595	-0.760798	0.017849	32.428413	5.998524
95	13.3940	9.05510	5.2342	3.215965	16.833004	27.396850	10.342431	12.083712
96	5.4369	0.61705	-2.7229	-5.222085	14.219215	7.414184	27.270172	2.590624

97 rows × 8 columns

In [33]:

```
plot1 = plt.scatter(df["X"], df["Y"])
plot2 = plt.plot(df["X"], df["y_predict"],color="red")
plt.show()
```



In [34]:

```
ssr = sum((df["y_predict"]-y_mean)**2)
ssr
```

Out[34]:

2046.3146047180405

In [35]:

```
sse = ((df["Y"]-df["y_predict"])**2).sum()
sse
```

Out[35]:

868.5324469391845

In [36]:

```
sst = sum((df["Y"]-y_mean)**2)
sst
```

Out[36]:

2914.8470516572247

In [37]:

```
tss_new =sse+ssr
tss_new
```

Out[37]:

2914.847051657225

In [38]:

```
correlation**2
```

Out[38]:

0.7020315537841398

In [39]:

```
cost = sse/96
cost
```

Out[39]:

9.04721298894984

In [ ]:

```

```