

## linearReg in py:

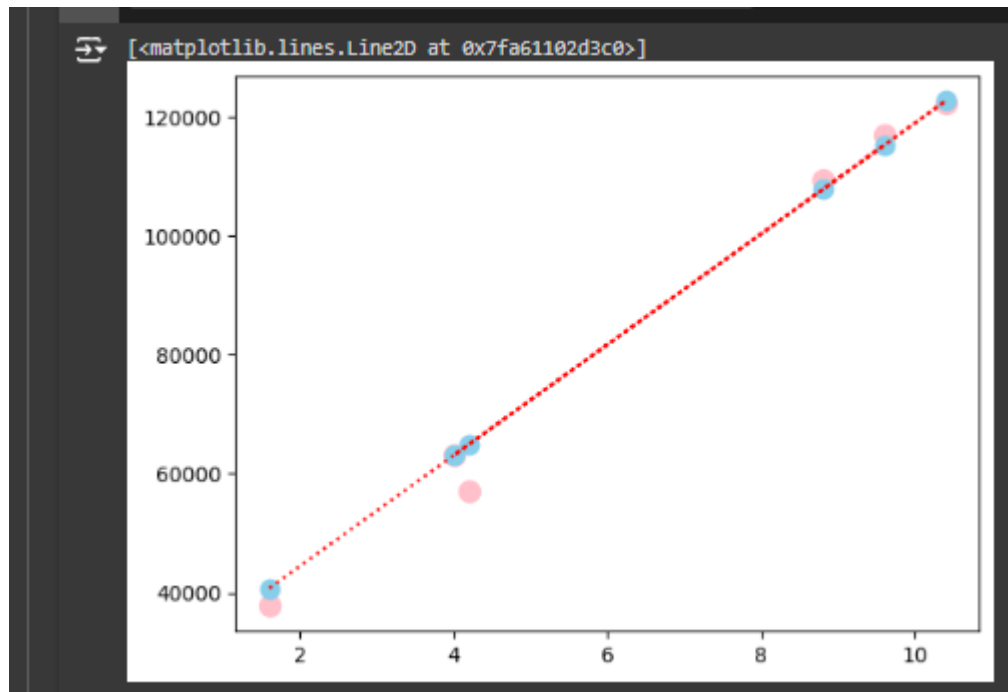
```
python code linear reg:- import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
data = pd.read_csv("/content/Salary_dataset.csv")
data.head()
data.info()
x = data["YearsExperience"]
y = data["Salary"]
#step 3
x = data[["YearsExperience"]]
y = data["Salary"].values
#step 4 - sk learn function
from sklearn.model_selection import train_test_split
x_train , x_test , y_train , y_test = train_test_split(x,y,train_size=0.8,random_state=0)
#step 5 - training model using linear reg & fit-function
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(x_train , y_train)
y_pred= model.predict(x_test)
print(y_pred)
print(y_test)
from sklearn.metrics import r2_score
score = r2_score (y_test , y_pred)
print(score)
```

```
+ Code + Text

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 3 columns):
#   Column          Non-Null Count  Dtype
---  ---          -
0   Unnamed: 0       30 non-null    int64
1   YearsExperience  30 non-null    float64
2   Salary           30 non-null    float64
dtypes: float64(2), int64(1)
memory usage: 848.0 bytes
[ 40749.96184072 122700.62295594 64962.65717022 63100.14214487
 115250.56285456 107800.50275317]
[ 37732. 122392. 57082. 63219. 116970. 109432.]
0.988169515729126
YearsExperience
0      1.2
1      1.4
2      1.6
3      2.1
4      2.3
5      3.0
6      3.1
7      3.3
8      3.3
9      3.8
10     4.0
11     4.1
12     4.1
13     4.2
14     4.6
15     5.0
16     5.2
17     5.4
18     6.0
19     6.1
20     6.9
21     7.2
22     8.0
23     8.3
24     8.8
25     9.1
26     9.6
27     9.7
28    10.4
29    10.6
[ 39344.  46206.  37732.  43526.  39892.  56643.  60151.  54446.  64446.
  57190.  63219.  55795.  56958.  57082.  61112.  67939.  66030.  83089.
  81364.  93941.  91739.  98274. 101303. 113813. 109432. 105583. 116970.
 112636. 122392. 121873.]
```

Output 1:

```
Code:- plt.scatter(x_test,y_test,color='pink',s=100)
plt.scatter(x_test,y_pred,color='skyblue',s=80)
plt.plot(x_test,y_pred,color='red',linestyle='dotted')
```



Output2:

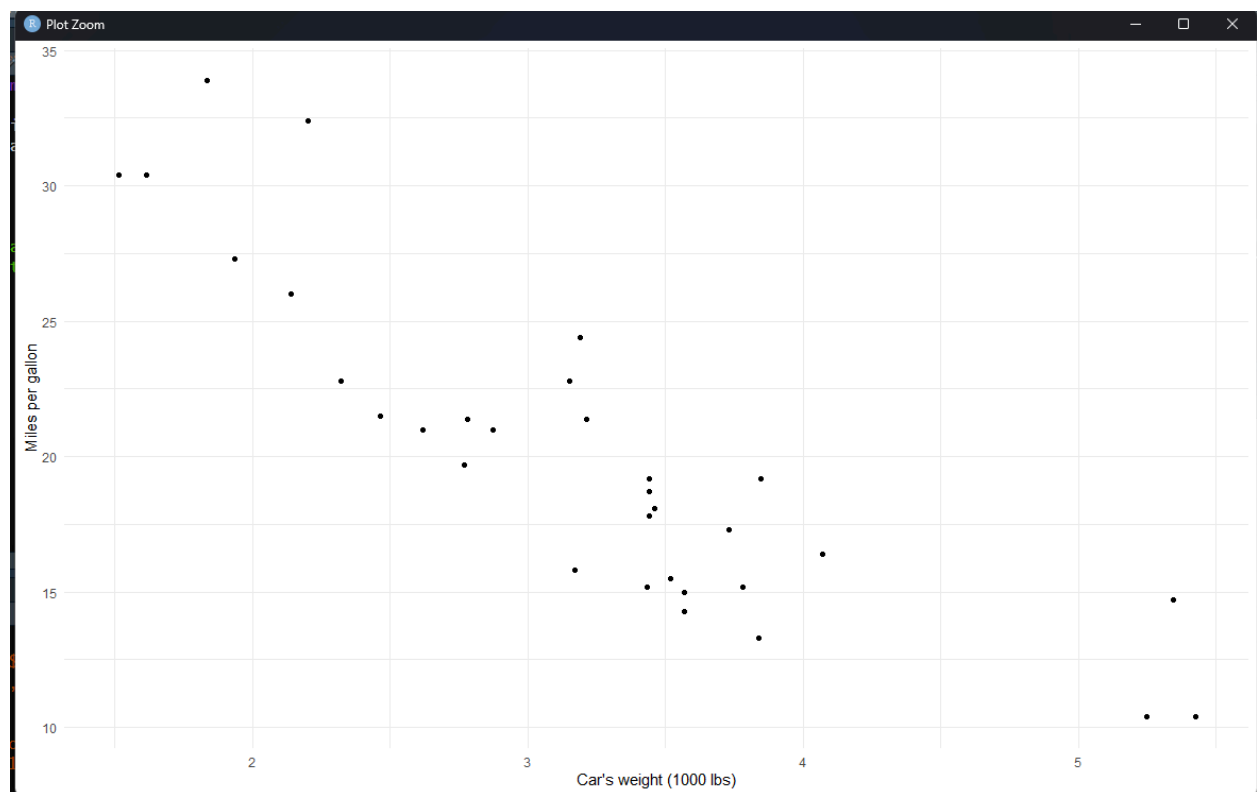
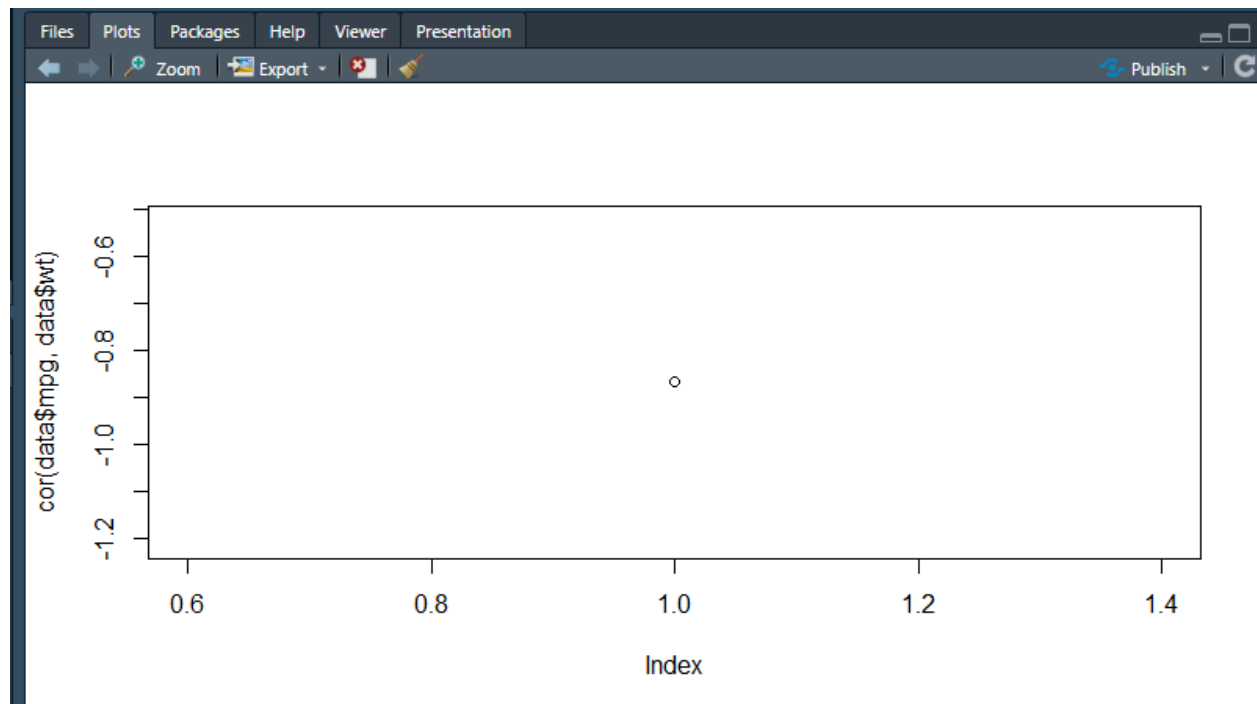
#### r code-

```
# refer statsandr.com website
#for mtcars dataset
data<-mtcars
cor(data$mpg, data$disp)
plot(cor(data$mpg, data$wt))
plot(mtcars)
library(ggplot2)
ggplot(data, aes(x = wt, y = mpg)) +
  geom_point() +
  labs(
    y = "Miles per gallon",
    x = "Car's weight (1000 lbs)"
  ) +
  theme_minimal()
#simple linear regression
model <- lm(mpg ~ wt, data = data)
summary(model)

# multiple linear regression
model1 <- lm(mpg ~ wt + hp + disp, data = data)
summary(model1)
```

solution-  
mtcars

```
> data<-mtcars
> cor(data$mpg, data$disp)
[1] -0.8475514
> cor(data$mpg, data$wt)
[1] -0.8676594
library(ggplot2)
```



```

+ theme_minimal()
> model <- lm(mpg ~ wt, data = data)
> summary(model)

```

Call:

```
lm(formula = mpg ~ wt, data = data)
```

Residuals:

Min	1Q	Median	3Q	Max
-4.5432	-2.3647	-0.1252	1.4096	6.8727

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	37.2851	1.8776	19.858	< 2e-16 ***
wt	-5.3445	0.5591	-9.559	1.29e-10 ***

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.046 on 30 degrees of freedom

Multiple R-squared: 0.7528, Adjusted R-squared: 0.7446

F-statistic: 91.38 on 1 and 30 DF, p-value: 1.294e-10

```

> model1 <- lm(mpg ~ wt + hp + disp, data = data)
> summary(model1)

```

Call:

```
lm(formula = mpg ~ wt + hp + disp, data = data)
```

Residuals:

Min	1Q	Median	3Q	Max
-3.891	-1.640	-0.172	1.061	5.861

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	37.105505	2.110815	17.579	< 2e-16 ***
wt	-3.800891	1.066191	-3.565	0.00133 **
hp	-0.031157	0.011436	-2.724	0.01097 *
disp	-0.000937	0.010350	-0.091	0.92851

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.639 on 28 degrees of freedom

Multiple R-squared: 0.8268, Adjusted R-squared: 0.8083

F-statistic: 44.57 on 3 and 28 DF, p-value: 8.65e-11