Assignment No. 03

Roll No. : 231326
Name : Sahil Jadhav
Subject : Machine Learning

In [1]:
```
import pandas as pd
```

In [2]:
```
ds = pd.read_csv('Salary_Data.csv')
```

In [3]:
```
print(ds)
```

```
    YearsExperience  Salary
0               1.1   39343
1               1.3   46205
2               1.5   37731
3               2.0   43525
4               2.2   39891
5               2.9   56642
6               3.0   60150
7               3.2   54445
8               3.2   64445
9               3.7   57189
10              3.9   63218
11              4.0   55794
12              4.0   56957
13              4.1   57081
14              4.5   61111
15              4.9   67938
16              5.1   66029
17              5.3   83088
18              5.9   81363
19              6.0   93940
20              6.8   91738
21              7.1   98273
22              7.9  101302
23              8.2  113812
24              8.7  109431
25              9.0  105582
26              9.5  116969
27              9.6  112635
28             10.3  122391
29             10.5  121872
```

In [4]:
```
ds.shape,ds.size,ds.ndim
```

Out[4]:
```
((30, 2), 60, 2)
```

In [5]:
```
ds.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 2 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   YearsExperience  30 non-null     float64
 1   Salary           30 non-null     int64  
dtypes: float64(1), int64(1)
memory usage: 612.0 bytes
```

In [6]:
```
ds.describe()
```

Out[6]:

	Ye rsExperience	Salary
count	30.000000	30.000000
mean	5.313333	76003.000000
std	2.837888	27414.429785
min	1.100000	37731.000000
25%	3.200000	56720.750000
50%	4.700000	65237.000000
75%	7.700000	100544.750000
max	10.500000	122391.000000

In [7]:

```
ds.head()
```

Out[7]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891

In [8]:

```
ds.tail()
```

Out[8]:

	YearsExperience	Salary
25	9.0	105582
26	9.5	116969
27	9.6	112635
28	10.3	122391
29	10.5	121872

In [9]:

```
ds.isnull().sum()
```

Out[9]:

```
YearsExperience    0
Salary             0
dtype: int64
```

In [10]:

```
ds.dtypes
```

Out[10]:

```
YearsExperience    float64
Salary               int64
dtype: object
```

In [11]:

```
# Store input valiable data to x and output variable data to y
x = ds.iloc[:,:-1]
```

```
y = ds.iloc[:,-1]
```

In [12]:
```
x.shape,y.shape
```

Out[12]:
```
((30, 1), (30,))
```

In [13]:
```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size = 0.2,random_state =
```

In [14]:
```
xtrain.shape,xtest.shape,ytrain.shape,ytest.shape
```

Out[14]:
```
((24, 1), (6, 1), (24,), (6,))
```

In [15]:
```
from sklearn.linear_model import LinearRegression
# Deind Linear Regression Model
model = LinearRegression()
```

In [16]:
```
model.fit(xtrain,ytrain)
```

Out[16]:
```
▾ LinearRegr ssion  i ?

LinearRegression()
```

In [17]:
```
#print model slop and intercept of the best fit line

print(f"Slop (m) = {model.coef_}")
print(f"Intercept (b) = {model.intercept_}")
print(f"Equation of line : Salary = {model.coef_} * experience + {model.intercep
```

```
Slop (m) = [9423.81532303]
Intercept (b) = 25321.583011776813
Equation of line : Salary = [9423.81532303] * experience + 25321.583011776813
```

In [19]:
```
import numpy as np
import warnings
warnings.filterwarnings('ignore')
#predict salary of new employee
experience = float(input("Enter Experience in years"))
experience = np.array([[experience]]) # reshape(1,1)
predicted_salary = model.predict(experience)
print(f"Predicted Salary = {predicted_salary}")
```

```
Predicted Salary = [44169.21365784]
```

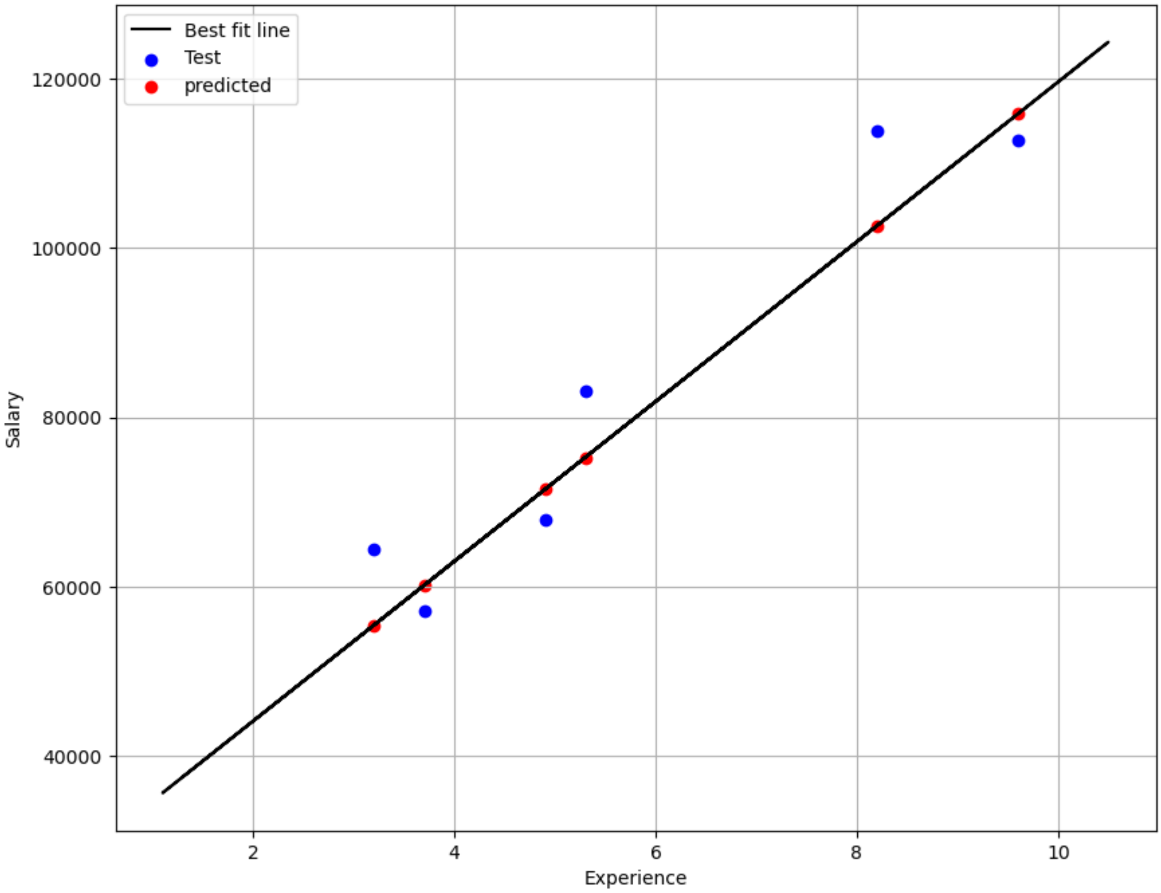
In [20]:
```
from sklearn.metrics import mean_squared_error, r2_score
# Evaluate model
pred = model.predict(xtest)
print(f"Training Score R2 =  {model.score(xtrain, ytrain)}")
print(f"Testing Score R2 = {model.score(xtest, ytest)}")
print(f"r2 score : {r2_score(ytest, model.predict(xtest))}")
print(f"Training Score MSE = {mean_squared_error(ytrain, model.predict(xtrain))}
print(f"Testing Score MSE = {mean_squared_error(ytest, model.predict(xtest))}")
```

```
Training Score R2 =  0.9645401573418146
Testing Score R2 = 0.9024461774180497
r2 score : 0.9024461774180497
Training Score MSE = 27102249.73126139
Testing Score MSE = 49830096.85590839
```

In [21]:

```
import matplotlib.pyplot as plt
#plot linear regression
plt.figure(figsize = (10,8))
#plt.scatter(xtrain, ytrain, color='g' label = 'ttraining')
plt.plot(xtrain, model.predict(xtrain), color='black', label = 'Best fit line')
plt.scatter(xtest, ytest, color='b', label = 'Test')
plt.scatter(xtest, pred, color='r', label = 'predicted')

plt.grid()
plt.legend()
plt.xlabel('Experience')
plt.ylabel('Salary')
plt.plot()
plt.show()
```



In [22]:

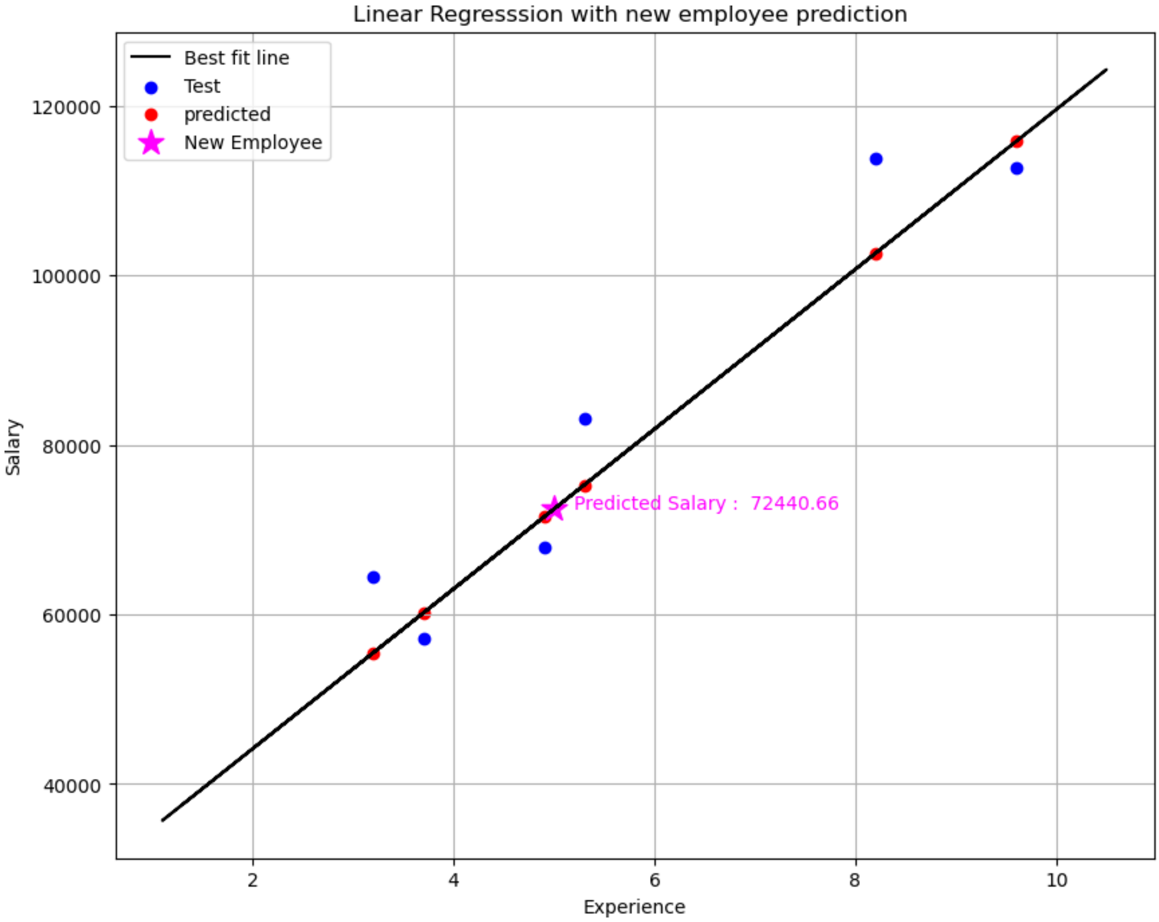
```
#plot linear regression
plt.figure(figsize = (10,8))
plt.plot(xtrain, model.predict(xtrain), color='black', label = 'Best fit line')
plt.scatter(xtest, ytest, color='b', label = 'Test')
plt.scatter(xtest, pred, color='r', label = 'predicted')

# predict the salary of the new employee
experience = float(input("Enter Experience in years : "))
experience_arr = np.array([[experience]])  #shape(1,1)
predicted_salary  = model.predict(experience_arr)[0] # take scalar
plt.scatter(experience, predicted_salary, color = 'magenta', marker='*', s=200,
#label the value

plt.text(experience + 0.2, predicted_salary, f"Predicted Salary : {predicted_sal

plt.xlabel('Experience')
plt.ylabel('Salary')
plt.legend()
plt.grid()
plt.title('Linear Regresssion with new employee prediction ')
plt.show()

print(f"Predicted Salary = {predicted_salary : .2f}")
```



```
Predicted Salary =  72440.66
```

```
In [ ]:
```