Practical No.6

Aim:- Regression and Its Types

- Implement Simple Linear Regression using a given dataset.
- Explore and interpret the Regression Model coeffiecients and goodness-of-fit measures.

(A)

Simple Linear Regression

In [1]:
```
#Importing Libraries
import numpy as nm
import matplotlib.pyplot as plt
import pandas as pd
from sklearn import metrics
```

In [4]:
```
#Loading Dataset
data_set = pd.read_csv('SalaryData.csv')
```

In [5]:
```
#Independent variable
X = data_set.iloc[:,:-1].values

#Dependent variable
Y = data_set.iloc[:,1].values
```

In [7]:
```
#Splitting the dataset into training and test set.
from sklearn.model_selection import  train_test_split
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.2,random_state=0)
```

In [9]:
```
#Fitting the Simple Linear Regression model to the training dataset
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train,Y_train)
```

Out[9]:
```
LinearRegression()
```

In [10]:
```
#Prediction of Test and Training set result
y_pred = regressor.predict(X_test)
x_pred = regressor.predict(X_train)
```

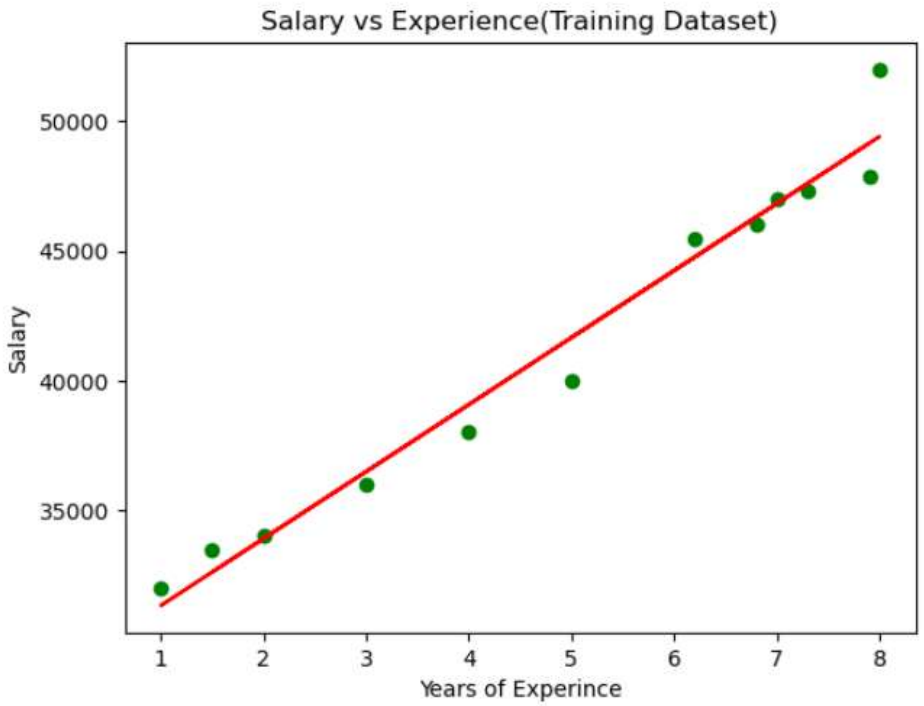
In [16]:
```
#Evaluation of Model

print('R squared:{:.2f}'.format(regressor.score(X,Y)*100))
print('Mean Absolute Error:',metrics.mean_absolute_error(Y_test,y_pred))
print('Mean Squared Error:',metrics.mean_squared_error(Y_test,y_pred))
print('Root Mean Squared Error:',metrics.mean_squared_error(Y_test,y_pred,squared=False))
```

```
R squared:97.30
Mean Absolute Error: 417.5303066318029
Mean Squared Error: 260049.2063071443
Root Mean Squared Error: 509.95019983047786
```

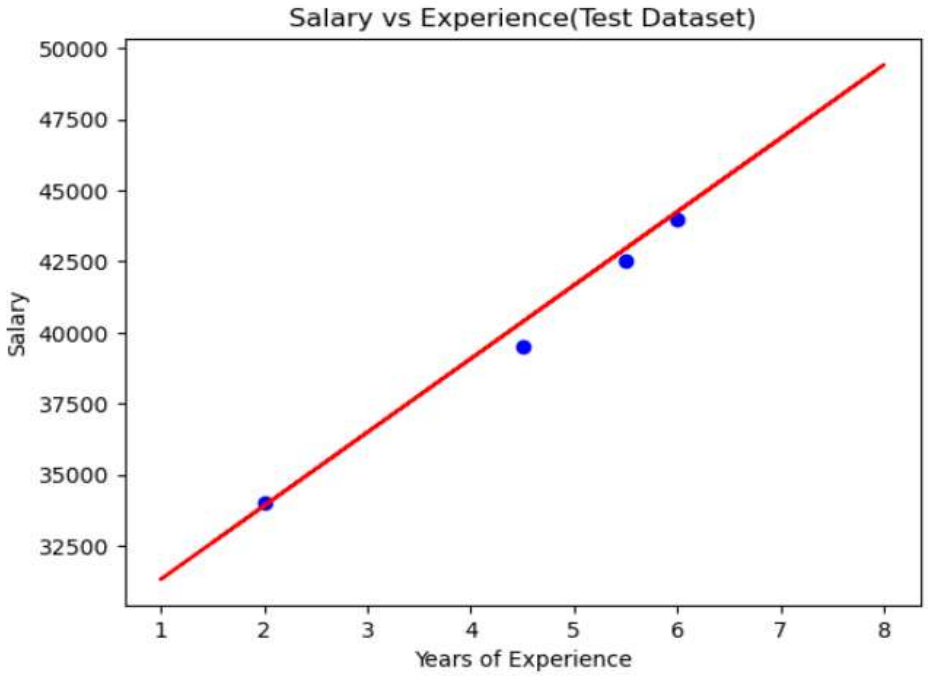
In [17]:
```
#Print the coefficients and intercept
print('Coefficients:',regressor.coef_)
print('Intercept:',regressor.intercept_)
```

```
Coefficients: [2585.21511766]
Intercept: 28738.55478963632
```

In [18]:
```
plt.scatter(X_train,Y_train,color="green")
plt.plot(X_train,x_pred,color="red")
plt.title("Salary vs Experience(Training Dataset)")
plt.xlabel("Years of Experince")
plt.ylabel("Salary")
plt.show()
```



```
In [19]: #visualizing the Test set results
plt.scatter(X_test,Y_test,color="blue")
plt.plot(X_train,x_pred,color="red")
plt.title("Salary vs Experience(Test Dataset)")
plt.xlabel("Years of Experience")
plt.ylabel("Salary")
plt.show()
```



(B)

- Implement Multiple Linear Regression and assess the impact of additional predictors.

MULTIPLE LINEAR REGRESSION

Importing the libraries

In [30]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn import metrics
```

Importing the dataset

In [36]:
```
dataset = pd.read_csv('50_Startups.csv')
X = dataset.iloc[:,:-1].values
Y = dataset.iloc[:,-1].values
```

In [37]:
```
print(X)
```

```
[[165349.2 136897.8 471784.1 'New York']
 [162597.7 151377.59 443898.53 'California']
 [153441.51 101145.55 407934.54 'Florida']
 [144372.41 118671.85 383199.62 'New York']
 [142107.34 91391.77 366168.42 'Florida']
 [131876.9 99814.71 362861.36 'New York']
 [134615.46 147198.87 127716.82 'California']
 [130298.13 145530.06 323876.68 'Florida']
 [120542.52 148718.95 311613.29 'New York']
 [123334.88 108679.17 304981.62 'California']]
```

Encoding categorical data

In [39]:
```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
ct = ColumnTransformer(transformers=[('encoder',OneHotEncoder(),[3])],remainder='passthrough')
X = np.array(ct.fit_transform(X))
```

In [40]:
```
print(X)
```

```
[[0.0 0.0 1.0 165349.2 136897.8 471784.1]
 [1.0 0.0 0.0 162597.7 151377.59 443898.53]
 [0.0 1.0 0.0 153441.51 101145.55 407934.54]
 [0.0 0.0 1.0 144372.41 118671.85 383199.62]
 [0.0 1.0 0.0 142107.34 91391.77 366168.42]
 [0.0 0.0 1.0 131876.9 99814.71 362861.36]
 [1.0 0.0 0.0 134615.46 147198.87 127716.82]
 [0.0 1.0 0.0 130298.13 145530.06 323876.68]
 [0.0 0.0 1.0 120542.52 148718.95 311613.29]
 [1.0 0.0 0.0 123334.88 108679.17 304981.62]]
```

Splitting the dataset into the training set and test set

In [41]:
```
from sklearn.model_selection import  train_test_split
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.2,random_state=0)
```

Training the Multiple Linear Regression model on the Training set

In [42]:
```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train,Y_train)
```

Out[42]:
```
LinearRegression()
```

Predicting the Test set results

In [44]:
```
y_pred = regressor.predict(X_test)
print("Prediction{}:".format(y_pred))
```

```
Prediction[103015.20159796 132582.27760816 132447.73845174  71976.09851258
 178537.48221055 116161.24230166  67851.69209676  98791.73374686
 113969.43533013 167921.06569551]:
```

In [45]:
```
mir_diff = pd.DataFrame({'Actual value': Y_test,'Predicted value': y_pred})
mir_diff.head()
```

Out[45]:

	Actual value	Predicted value
0	103282.38	103015.201598
1	144259.40	132582.277608
2	146121.95	132447.738452
3	77798.83	71976.098513
4	191050.39	178537.482211

In [46]:
```
#Evaluation of Model

print('R squared:{:.2f}'.format(regressor.score(X,Y)*100))
print('Mean Absolute Error:',metrics.mean_absolute_error(Y_test,y_pred))
print('Mean Squared Error:',metrics.mean_squared_error(Y_test,y_pred))
print('Root Mean Squared Error:',metrics.mean_squared_error(Y_test,y_pred,squared=False))
```

```
R squared:94.85
Mean Absolute Error: 7514.2936596413765
Mean Squared Error: 83502864.03259295
Root Mean Squared Error: 9137.990152795797
```