EXPERIMENT 7: LINEAR REGRESSION

Aim:

To predict the relationship between a dependent variable and one or more independent variables using Linear Regression.

Procedure :

1. **Collect Data:** Gather a dataset containing the dependent variable and independent variable(s).
2. **Preprocess Data:** Handle missing values, encode categorical variables, and scale features if necessary.
3. **Split Dataset:** Divide the data into training and testing sets (e.g., 80% train, 20% test).
4. **Train Model:** Apply Linear Regression on the training data to learn the relationship between variables.
5. **Evaluate Model:** Predict on test data and assess performance using metrics like R^2, MAE, or RMSE.

In [66]:

```
import numpy as np
import pandas as pd
df=pd.read_csv("C:\\Users\\kaviy\\Downloads\\Salary_data.csv")
df
df.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: 608.0 bytes
```

In [67]:

```
df.dropna(inplace=True)
df.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: 608.0 bytes
```

In [68]:

```
df.describe()
```

Out[68]:

	YearsExperience	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 [71]:

```
features=df.iloc[:,[0]].values
label=df.iloc[:,[1]].values
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(features,label,test_size=0.2
,random_state=42)
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(x_train,y_train)
```

Out[71]:

```
LinearRegression
 LinearRegression()
```

In [72]:

```
model.score(x_train,y_train)
```

Out[72]:

```
0.9645401573418146
```

In [73]:

```
model.score(x_test,y_test)
```

Out[73]:

```
0.9024461774180497
```

In [74]:

```
model.coef_
```

Out[74]:

```
array([[9423.81532303]])
```

EXPERIMENT 7: LINEAR REGRESSION

In [75]:

```
model.intercept_
```

Out[75]:

```
array([25321.58301178])
```

In [ ]:

```
import pickle
pickle.dump(model,open('SalaryPred.model','wb'))
model=pickle.load(open('SalaryPred.model','rb'))
yr_of_exp=float(input("Enter Years of Experience: "))
yr_of_exp_NP=np.array([[yr_of_exp]])
Salary=model.predict(yr_of_exp_NP)
```

In [ ]:

```
print("Estimated Salary for {} years of experience is {}: "
.format(yr_of_exp,Salary)
```

Result :

- The Linear Regression model successfully learned the relationship between input and output variables.
- Predicted values on the test set were close to actual values, indicating good model performance.
- Evaluation metrics (e.g., R^2 score) confirmed the model's accuracy in predicting outcomes.