```
# import all the lib
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

```
# read the dataset using pandas
data = pd.read_csv('/content/Salary_Data.csv')
```

```
# This displays the top 5 rows of the data
data.head()
```

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

```
# Provides some information regarding the columns in the data
data.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     float64
dtypes: float64(2)
memory usage: 608.0 bytes
```

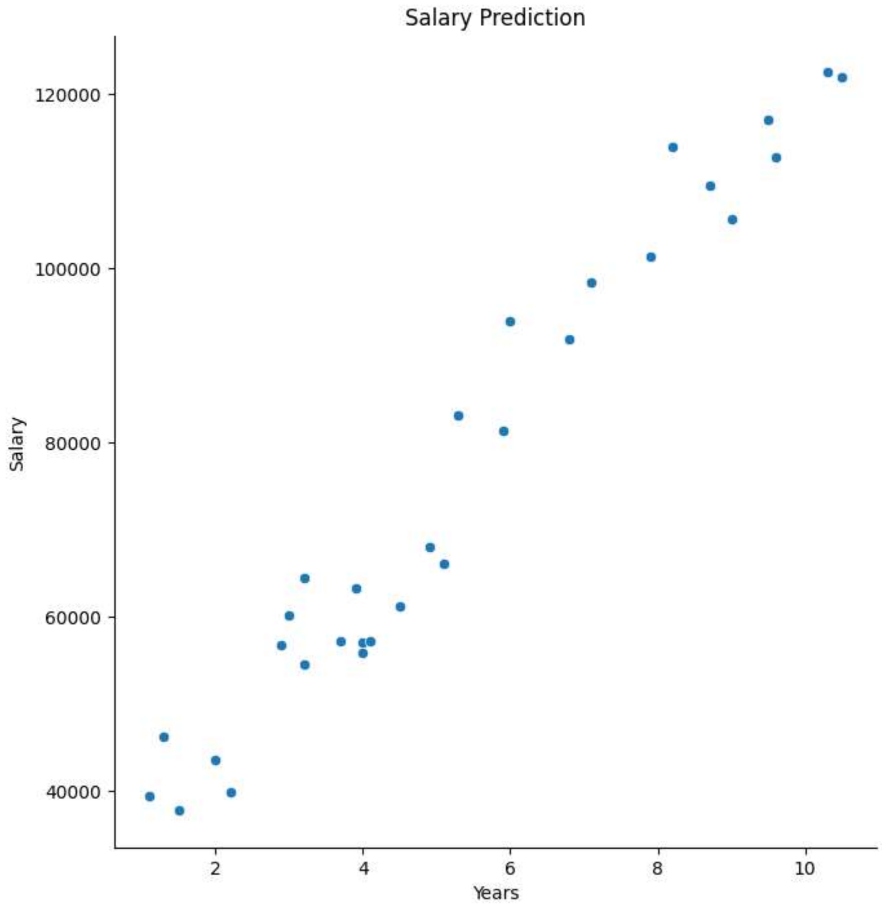
```
# this describes the basic stat behind the dataset used
data.describe()
```

	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

```
# These Plots help to explain the values and how they are scattered

plt.figure(figsize=(12,6))
sns.pairplot(data,x_vars=['YearsExperience'],y_vars=['Salary'],size=7,kind='scatter')
plt.xlabel('Years')
plt.ylabel('Salary')
plt.title('Salary Prediction')
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/axisgrid.py:2100: UserWarning: The `siz
  warnings.warn(msg, UserWarning)
<Figure size 1200x600 with 0 Axes>
```




```
# Cooking the data
X = data['YearsExperience']
X.head()
```

```
0    1.1
1    1.3
2    1.5
3    2.0
4    2.2
Name: YearsExperience, dtype: float64
```

```
# Cooking the data
y = data['Salary']
y.head()
```

```
0    39343.0
1    46205.0
2    37731.0
3    43525.0
4    39891.0
Name: Salary, dtype: float64
```

```
# Import Segregating data from scikit learn
from sklearn.model_selection import train_test_split
```

```
# Split the data for train and test
X_train,X_test,y_train,y_test = train_test_split(X,y,train_size=0.7,random_state=100)
```

```
# Create new axis for x column
X_train = X_train[:,np.newaxis]
X_test = X_test[:,np.newaxis]
```

```
<ipython-input-14-d825d65695ed>:2: FutureWarning: Support for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will b
  X_train = X_train[:,np.newaxis]
<ipython-input-14-d825d65695ed>:3: FutureWarning: Support for multi-dimensional indexing (e.g. `obj[:, None]`) is deprecated and will b
  X_test = X_test[:,np.newaxis]
```

```
# Importing Linear Regression model from scikit learn
from sklearn.linear_model import LinearRegression
```

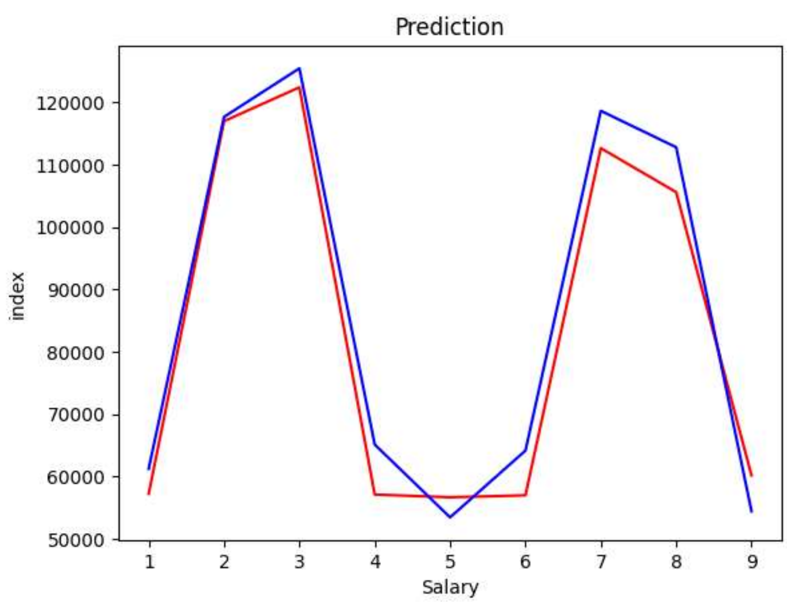
```
# Fitting the model
lr = LinearRegression()
lr.fit(X_train,y_train)
```

```
▾ LinearRegression
LinearRegression()
```

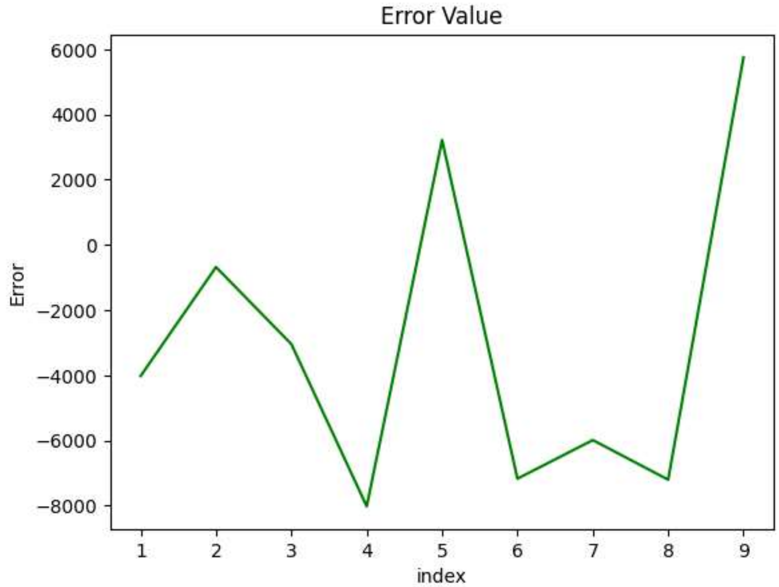
```
# Predicting the Salary for the Test values
y_pred = lr.predict(X_test)
```

```
# Plotting the actual and predicted values

c = [i for i in range (1,len(y_test)+1,1)]
plt.plot(c,y_test,color='r',linestyle='-')
plt.plot(c,y_pred,color='b',linestyle='-')
plt.xlabel('Salary')
plt.ylabel('index')
plt.title('Prediction')
plt.show()
```




```
# plotting the error
c = [i for i in range(1,len(y_test)+1,1)]
plt.plot(c,y_test-y_pred,color='green',linestyle='-')
plt.xlabel('index')
plt.ylabel('Error')
plt.title('Error Value')
plt.show()
```




```
# Importing metrics for the evaluation of the model
from sklearn.metrics import r2_score,mean_squared_error
```

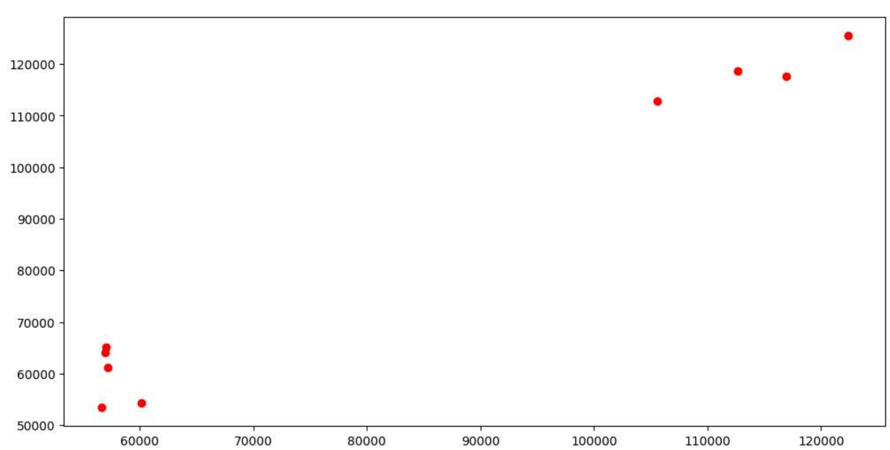
```
# calculate Mean square error
mse = mean_squared_error(y_test,y_pred)
```

```
# Calculate R square vale
rsq = r2_score(y_test,y_pred)
```

```
print('mean squared error :',mse)
print('r square :',rsq)
```

```
mean squared error : 30310299.043402452
r square : 0.9627668685473267
```

```
# Just plot actual and predicted values for more insights
plt.figure(figsize=(12,6))
plt.scatter(y_test,y_pred,color='r',linestyle='-')
plt.show()
```




```
# Intecept and coeff of the line
```

```
# intercept and coef. of the line
print('Intercept of the model:',lr.intercept_)
print('Coefficient of the line:',lr.coef_)
```

```
Intercept of the model: 25202.887786154883
Coefficient of the line: [9731.20383825]
```

Then it is said to form a line with

$$y = 25202.8 + 9731.2x$$