Name - Kumar Rohit PRN - 21070126052

In [236…

```python
import numpy as np
import pandas as pd
import plotly.express as px
from sklearn.linear_model import LinearRegression
import plotly.graph_objects as go
```

Dataset - (Salary data based on years of experience)

Importing the dataset

In [17]:

```python
# using pandas read csv function for importing the dataframe
salary = pd.read_csv("/Users/kumarrohit/Downloads/Salary_Data.csv")
salary
```

Out[17]:

	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
5	2.9	56642.0
6	3.0	60150.0
7	3.2	54445.0
8	3.2	64445.0
9	3.7	57189.0
10	3.9	63218.0
11	4.0	55794.0
12	4.0	56957.0
13	4.1	57081.0
14	4.5	61111.0
15	4.9	67938.0
16	5.1	66029.0
17	5.3	83088.0
18	5.9	81363.0
19	6.0	93940.0
20	6.8	91738.0
21	7.1	98273.0
22	7.9	101302.0
23	8.2	113812.0
24	8.7	109431.0
25	9.0	105582.0
26	9.5	116969.0
27	9.6	112635.0
28	10.3	122391.0
29	10.5	121872.0

Checking for the null values

In [177…

```python
salary.isnull().sum()
```

Out[177…

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

Getting the statistical information of the dataset

In [19]:

```python
salary.describe()
```

Out[19]:

	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

Scaling the data -

In [152…

```python
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
salary_sc = sc.fit_transform(salary.iloc[:,0:])
salary_sc
```

Out[152…

```
array([[-1.51005294, -1.36011263],
       [-1.43837321, -1.10552744],
       [-1.36669348, -1.419919  ],
       [-1.18749416, -1.20495739],
       [-1.11581443, -1.33978143],
       [-0.86493538, -0.71830716],
       [-0.82909552, -0.58815781],
       [-0.75741579, -0.79981746],
       [-0.75741579, -0.42881019],
       [-0.57821647, -0.69801306],
       [-0.50653674, -0.47433279],
       [-0.47069688, -0.74976858],
       [-0.47069688, -0.70662043],
       [-0.43485702, -0.70201994],
       [-0.29149756, -0.55250402],
       [-0.1481381 , -0.29921736],
       [-0.07645838, -0.37004264],
       [-0.00477865,  0.26285865],
       [ 0.21026054,  0.19885989],
       [ 0.2461004 ,  0.66547573],
       [ 0.53281931,  0.58377993],
       [ 0.6403389 ,  0.82623317],
       [ 0.92705781,  0.93861127],
       [ 1.03457741,  1.40274136],
       [ 1.21377673,  1.24020308],
       [ 1.32129632,  1.09740238],
       [ 1.50049564,  1.51986835],
       [ 1.5363355 ,  1.3590738 ],
       [ 1.78721455,  1.72102849],
       [ 1.85889428,  1.70177321]])
```

In [154…

```python
salary_sc = pd.DataFrame(data=salary_sc,columns=['YearsExperience','Salary'])
salary_sc
```

Out[154…

	YearsExperience	Salary
0	-1.510053	-1.360113
1	-1.438373	-1.105527
2	-1.366693	-1.419919
3	-1.187494	-1.204957
4	-1.115814	-1.339781
5	-0.864935	-0.718307
6	-0.829096	-0.588158
7	-0.757416	-0.799817
8	-0.757416	-0.428810
9	-0.578216	-0.698013
10	-0.506537	-0.474333
11	-0.470697	-0.749769
12	-0.470697	-0.706620
13	-0.434857	-0.702020
14	-0.291498	-0.552504
15	-0.148138	-0.299217
16	-0.076458	-0.370043
17	-0.004779	0.262859
18	0.210261	0.198860
19	0.246100	0.665476
20	0.532819	0.583780
21	0.640339	0.826233
22	0.927058	0.938611
23	1.034577	1.402741
24	1.213777	1.240203
25	1.321296	1.097402
26	1.500496	1.519868
27	1.536336	1.359074
28	1.787215	1.721028
29	1.858894	1.701773

EDA on the dataset

In [180…

```python
#plotting a scatter plot on the dataset
fig = px.scatter(data_frame=salary,x='YearsExperience',y='Salary',template='plotly_dark',labels={'YearsExperien
            title="Salary vs Years Of Experience")
fig.show()
```

Splitting the data into input(X) and output label(Y)

In [155…

```python
X = salary_sc['YearsExperience'].values
y = salary_sc['Salary'].values
```

In [156…

```python
# Reshaping the data

X = X.reshape(-1,1)
y = y.reshape(-1,1)
```

In [157…

```python
print(X.shape)
print(y.shape)
```

```
(30, 1)
(30, 1)
```

Splitting the data for training and testing and fitting the model

In [158…

```python
# Doing 80 - 20 split for training and testing respectively
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2,random_state=42)
lr = LinearRegression()

# Fitting the data on the training set
lr.fit(X_train,y_train)
```

Out[158…

```
▾ LinearRegression
LinearRegression()
```

Finding out the predictions from the model

In [159…

```python
y_preds = lr.predict(X_test)
y_preds
```

Out[159…

```
array([[ 1.47613439],
       [-0.16712845],
       [ 0.98665184],
       [-0.0272763 ],
       [-0.76150012],
       [-0.58668493]])
```

Finding out the slope and intercept of the regression line and forming the equation of the line

In [160…

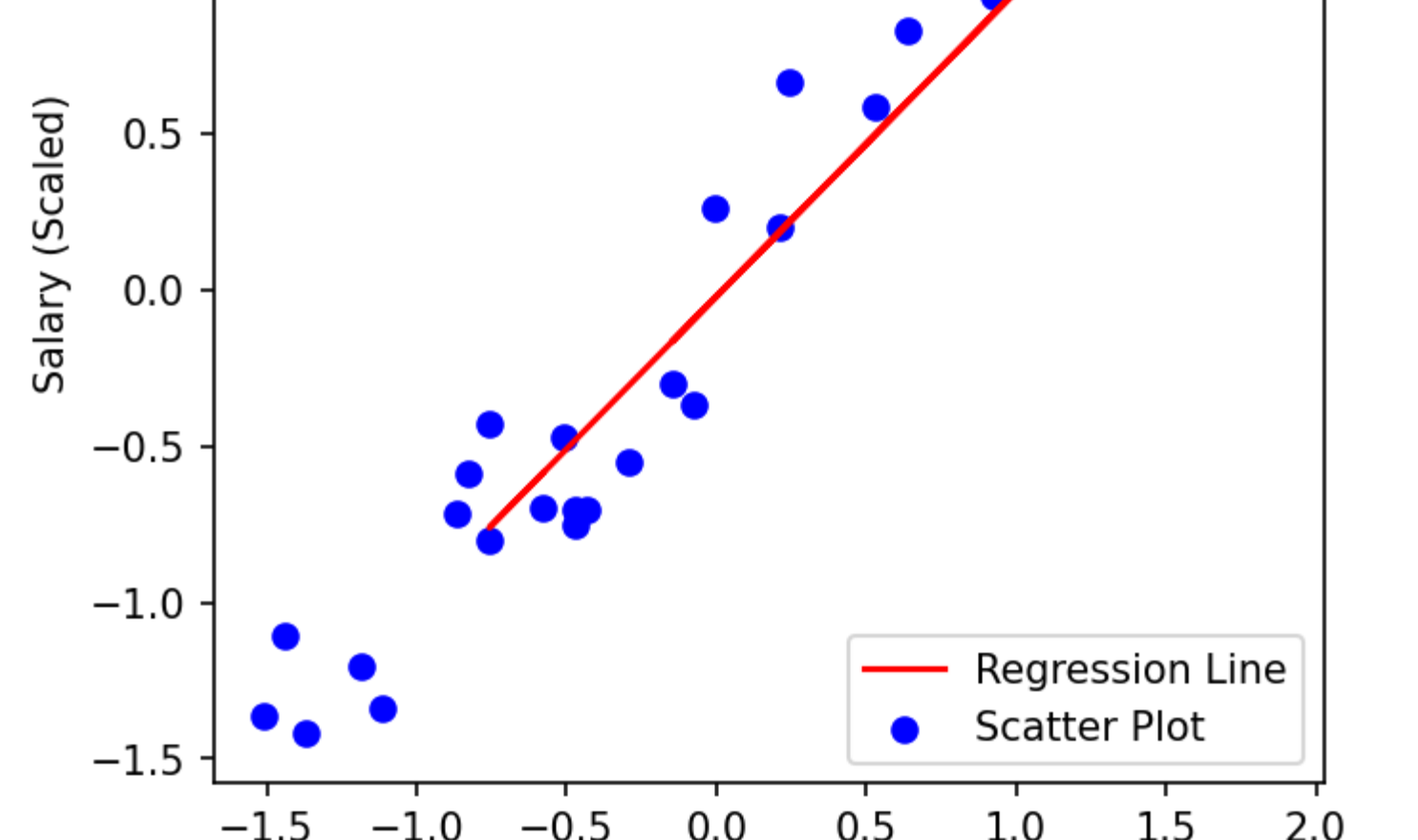
```python
print("The slope of the regression line is :" ,lr.coef_)
print("The intercept of the regression lne is:",lr.intercept_)

# Equation of the line
# y = mx + c

# y = 0.97553493(x) + -0.02261456
```

```
The slope of the regression line is : [[0.97553493]]
The intercept of the regression lne is: [-0.02261456]
```

Plotting the regression line

In [184…

```python
import matplotlib.pyplot as plt
plt.figure(figsize=(5,5),dpi=150)
# plt.plot(X,y,color='red')
plt.scatter(X,y,color='blue',label='Scatter Plot')
plt.title('Relationship between Sales and Years of Experience')
plt.xlabel('Years of Experience (Scaled)')
plt.ylabel('Salary (Scaled)')
plt.plot(X_test,y_preds,label='Regression Line',color='red')
plt.legend(loc=4)
plt.show()
```



Metrics - Checking the performance of the model using RMSE and R2 score

In [231…

```python
# Importing the required libraries
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error

mse = mean_squared_error(y_test,y_preds)
rmse = np.sqrt(mse)

print(f'The R2 score the model is: {r2_score(y_test,y_preds)}')
print(f'The RMSE value is {rmse }')
```

```
The R2 score the model is: 0.9024461774180498
The RMSE value is 0.2618956451815691
```

In [167…

```python
# Checking for Overfitting and Underfitting

print("Training set score: {:.4f}".format(lr.score(X_train,y_train)))
print("Test set score: {:.4f}".format(lr.score(X_test,y_test)))
```

```
Training set score: 0.9645
Test set score: 0.9024
```

Checking for individual prediction

In [175…

```python
years_of_exp = int(input("Enter the number of years of experience : "))
predSalary = lr.predict([[years_of_exp]])
print(f"Predicted Salary is :{predSalary}")
```

```
Enter the number of years of experience : 19
Predicted Salary is :[[18.51254916]]
```

Dataset - (Position Salaries)

Importing the dataset

In [176…

```python
position = pd.read_csv("/Users/kumarrohit/Downloads/position_salaries.csv")
position
```

Out[176…

	Position	Level	Salary
0	Business Analyst	1	45000
1	Junior Consultant	2	50000
2	Senior Consultant	3	60000
3	Manager	4	80000
4	Country Manager	5	110000
5	Region Manager	6	150000
6	Partner	7	200000
7	Senior Partner	8	300000
8	C-level	9	500000
9	CEO	10	1000000

Checking for the null values

In [178…

```python
position.isnull().sum()
```

Out[178…

```
Position    0
Level       0
Salary      0
dtype: int64
```

Getting the statistical information of the dataset

In [179…

```python
position.describe()
```

Out[179…

	Level	Salary
count	10.00000	10.000000
mean	5.50000	249500.000000
std	3.02765	299373.883668
min	1.00000	45000.000000
25%	3.25000	65000.000000
50%	5.50000	130000.000000
75%	7.75000	275000.000000
max	10.00000	1000000.000000

EDA on the dataset

In [195…

```python
# Poltting a bar chart to analyse the salaries at different positions
fig = px.bar(data_frame=position,x='Position',y='Salary',text='Salary',template='plotly_dark')
fig.update_layout(xaxis=dict(showgrid=False),yaxis=dict(showgrid=False))
fig.show()
```

In [199…

```python
# Scatter Plot between Level and Salary

fig = px.scatter(data_frame=position,x='Level',y='Salary',template='plotly_dark')
# fig.update_layout(xaxis=dict(showgrid=Fa))
fig.show()
```

Scaling the dataset for modelling and applying Linear Regression Algorithm

In [223…

```python
# Creating an object of Standard Scaler
sc2 = StandardScaler()
position_sc = sc2.fit_transform(position.iloc[:,1:])

# Converting this into a dataframe -
position_sc = pd.DataFrame(data=position_sc,columns=['Level','Salary'])
position_sc
```

Out[223…

	Level	Salary
0	-1.566699	-0.720043
1	-1.218544	-0.702438
2	-0.870388	-0.667228
3	-0.522233	-0.596808
4	-0.174078	-0.491178
5	0.174078	-0.350339
6	0.522233	-0.174289
7	0.870388	0.177810
8	1.218544	0.882008
9	1.566699	2.642503

Splitting the data into input (X) and output label (Y)

In [224…

```python
X_2 = position_sc['Level'].values
y_2 = position_sc['Salary'].values
```

In [225…

```python
# Reshaping the data
X_2 = X.reshape(-1,1)
y_2 = y.reshape(-1,1)
```

Splitting the data for training and testing

In [226…

```python
X_train_2 ,X_test_2, y_train_2, y_test_2 = train_test_split(X_2,y_2,test_size=0.2,random_state=42)

lr2 = LinearRegression()
# Fitting the data for training the model
lr2.fit(X_train,y_train)
```

Out[226…

```
▾ LinearRegression
LinearRegression()
```

Finding out the predictions from the model

In [227…

```python
y_preds_2 = lr2.predict(X_test_2)
y_preds_2
```

Out[227…

```
array([[ 1.16611728],
       [-1.2113464 ]])
```

Finding out the slope and intercept

In [228…

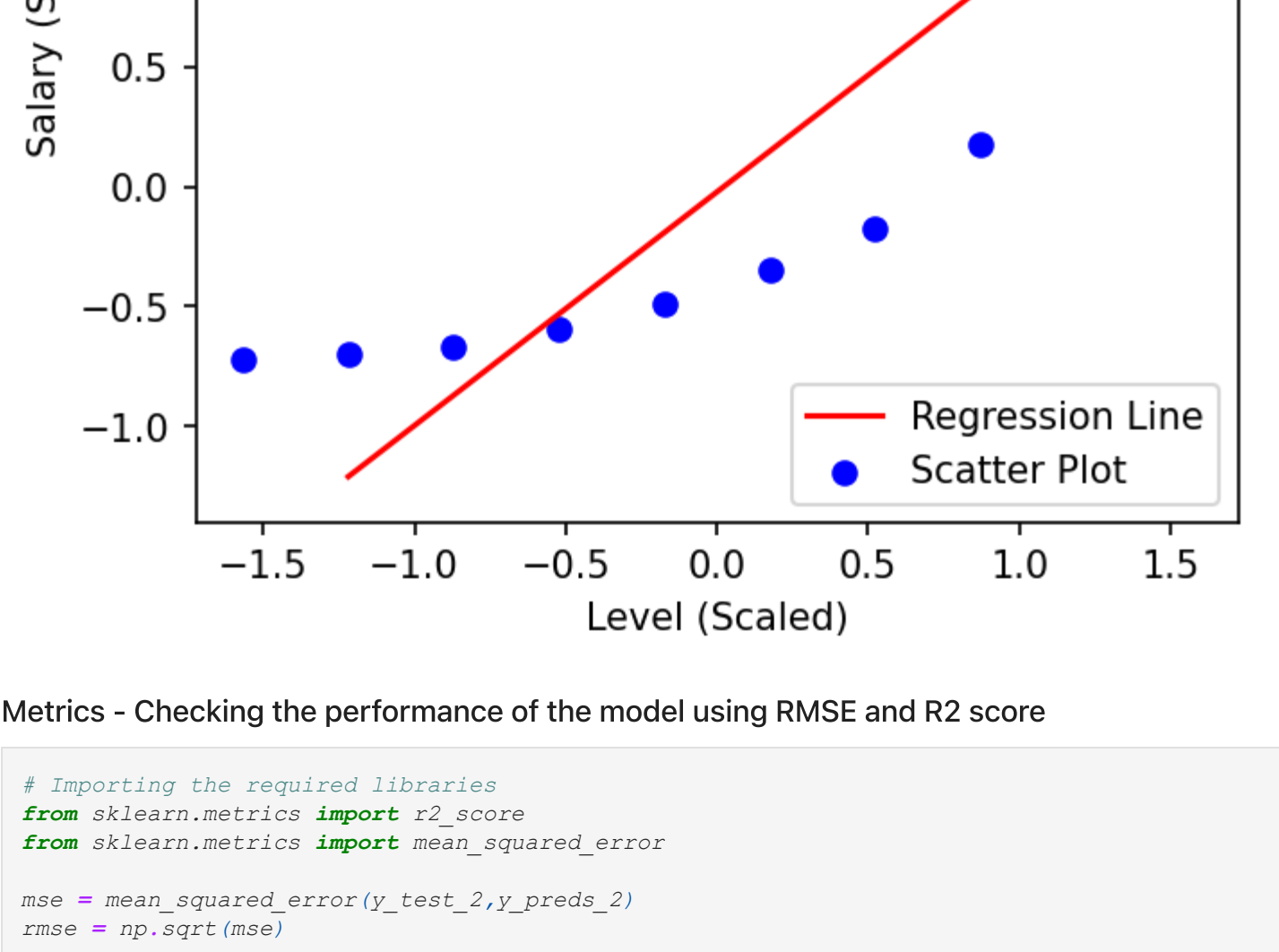
```python
print("The slope of the regression line is :" ,lr2.coef_)
print("The intercept of the regression lne is:",lr2.intercept_)
```

```
The slope of the regression line is : [[0.97553493]]
The intercept of the regression lne is: [-0.02261456]
```

Plotting the regression line

In [230…

```python
import matplotlib.pyplot as plt
plt.figure(figsize=(5,5),dpi=150)
# plt.plot(X,y,color='red')
plt.scatter(X_2,y_2,color='blue',label='Scatter Plot')
plt.title('Relationship between Level and Salary')
plt.xlabel('Level (Scaled)')
plt.ylabel('Salary (Scaled)')
plt.plot(X_test_2,y_preds_2,label='Regression Line',color='red')
plt.legend(loc=4)
plt.show()
```



Metrics - Checking the performance of the model using RMSE and R2 score

In [233…

```python
# Importing the required libraries
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error

mse = mean_squared_error(y_test_2,y_preds_2)
rmse = np.sqrt(mse)

print(f'The R2 score the model is: {r2_score(y_test_2,y_preds_2)}')
print(f'The RMSE value is {rmse}')
```

```
The R2 score the model is: 0.7293682084360357
The RMSE value is 0.4121324001812414
```

In [234…

```python
# Checking for Overfitting and Underfitting

print("Training set score: {:.4f}".format(lr.score(X_train_2,y_train_2)))
print("Test set score: {:.4f}".format(lr.score(X_test_2,y_test_2)))
```

```
Training set score: 0.6305
Test set score: 0.7294
```

In [ ]: