In [1]:
```python
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

In [ ]:
```python
student_df = pd.read_csv("/kaggle/input/student-performance-multiple-linear-regression/Student_Performance.csv")
```

In [ ]:
```python
student_df.head()
```

Out[ ]:

	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	7	99	Yes	9	1	91.0
1	4	82	No	4	2	65.0
2	8	51	Yes	7	2	45.0
3	5	52	Yes	5	2	36.0
4	7	75	No	8	5	66.0

In [ ]:
```python
student_df.shape
```

Out[ ]:
```
(10000, 6)
```

In [ ]:
```python
student_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 6 columns):
 #   Column                            Non-Null Count  Dtype  
---  ------                            --------------  -----  
 0   Hours Studied                     10000 non-null  int64  
 1   Previous Scores                   10000 non-null  int64  
 2   Extracurricular Activities        10000 non-null  object 
 3   Sleep Hours                       10000 non-null  int64  
 4   Sample Question Papers Practiced  10000 non-null  int64  
 5   Performance Index                 10000 non-null  float64
dtypes: float64(1), int64(4), object(1)
memory usage: 468.9+ KB
```

In [ ]:
```python
student_df.isnull().sum()
```

Out[ ]:
```
Hours Studied                       0
Previous Scores                     0
Extracurricular Activities          0
Sleep Hours                         0
Sample Question Papers Practiced    0
Performance Index                   0
dtype: int64
```

In [ ]:
```python
student_df.duplicated().sum()
```

Out[ ]:
```
127
```

In [ ]:
```python
student_df = student_df.drop_duplicates()
```

In [ ]:
```python
student_df.duplicated().sum()
```

Out[ ]:
```
0
```

In [ ]:
```python
student_df.describe()
```

Out[ ]:

	Hours Studied	Previous Scores	Sleep Hours	Sample Question Papers Practiced	Performance Index
count	9873.000000	9873.000000	9873.000000	9873.000000	9873.000000
mean	4.992100	69.441102	6.531652	4.583004	55.216651
std	2.589081	17.325601	1.697683	2.867202	19.208570
min	1.000000	40.000000	4.000000	0.000000	10.000000
25%	3.000000	54.000000	5.000000	2.000000	40.000000
50%	5.000000	69.000000	7.000000	5.000000	55.000000
75%	7.000000	85.000000	8.000000	7.000000	70.000000
max	9.000000	99.000000	9.000000	9.000000	100.000000

In [ ]:
```python
clean_df = student_df.copy()
clean_df = pd.get_dummies(clean_df, columns = ['Extracurricular Activities'])
```

In [ ]:
```python
clean_df = clean_df.replace({True: 1, False: 0})
clean_df
```

Out[ ]:

	Hours Studied	Previous Scores	Sleep Hours	Sample Question Papers Practiced	Performance Index	Extracurricular Activities_No	Extracurricular Activities_Yes
0	7	99	9	1	91.0	0	1
1	4	82	4	2	65.0	1	0
2	8	51	7	2	45.0	0	1
3	5	52	5	2	36.0	0	1
4	7	75	8	5	66.0	1	0
...	...	...	...	...	...	...	...
9995	1	49	4	2	23.0	0	1
9996	7	64	8	5	58.0	0	1
9997	6	83	8	5	74.0	0	1
9998	9	97	7	0	95.0	0	1
9999	7	74	8	1	64.0	1	0

9873 rows × 7 columns

In [ ]:
```python
X = clean_df.drop(['Performance Index'], axis = 1)
y = clean_df['Performance Index']
```

In [ ]:
```python
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 = 7)
```

In [ ]:
```python
from sklearn.linear_model import LinearRegression
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)
```

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

In [ ]:
```python
lin_pred = lin_reg.predict(X_test)
lin_pred
```

Out[ ]:
```
array([60.70228604, 47.46488821, 47.61635294, ..., 61.69489914,
       70.32626614, 44.30172536])
```

In [ ]:
```python
results_df = pd.DataFrame({'True Performance': y_test, 'Predicted Performance': lin_pred})
results_df
```

Out[ ]:

	True Performance	Predicted Performance
3265	59.0	60.702286
175	45.0	47.464888
4225	48.0	47.616353
4653	64.0	65.771226
1321	24.0	20.788038
...	...	...
541	64.0	67.257694
5355	34.0	33.915870
2544	60.0	61.694899
3888	70.0	70.326266
1935	46.0	44.301725

1975 rows × 2 columns

In [ ]:
```python
lin_mae = mean_absolute_error(y_test, lin_pred)
lin_mae
```

Out[ ]:
```
1.634425181582163
```

In [ ]:
```python
R2Score = r2_score(y_test, lin_pred)
print(f"R^2 Score is: {R2Score*100}")
```

```
R^2 Score is: 98.79370299201616
```

In [ ]:
```python
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler

lin_pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('model', LinearRegression())
])

lin_pipeline.fit(X_train, y_train)
lin_pipeline_pred = lin_pipeline.predict(X_test)
```

In [ ]:
```python
results_df = pd.DataFrame({'True Performance': y_test, 'Predicted Performance': lin_pipeline_pred})
results_df
```

Out[ ]:

	True Performance	Predicted Performance
3265	59.0	60.715747
175	45.0	47.489653
4225	48.0	47.644918
4653	64.0	65.734780
1321	24.0	20.760702
...	...	...
541	64.0	67.281626
5355	34.0	33.932692
2544	60.0	61.722495
3888	70.0	70.313385
1935	46.0	44.318302

1975 rows × 2 columns

In [ ]:
```python
lin_pipeline_mae = mean_absolute_error(y_test, lin_pipeline_pred)
lin_pipeline_mae
```

Out[ ]:
```
1.6354588740460978
```

In [ ]:
```python
R2Score = r2_score(y_test, lin_pipeline_pred)
print(f"R^2 Score is: {R2Score*100}")
```

```
R^2 Score is: 98.79291479120346
```