
Experiment No: 05

Code 01 :-

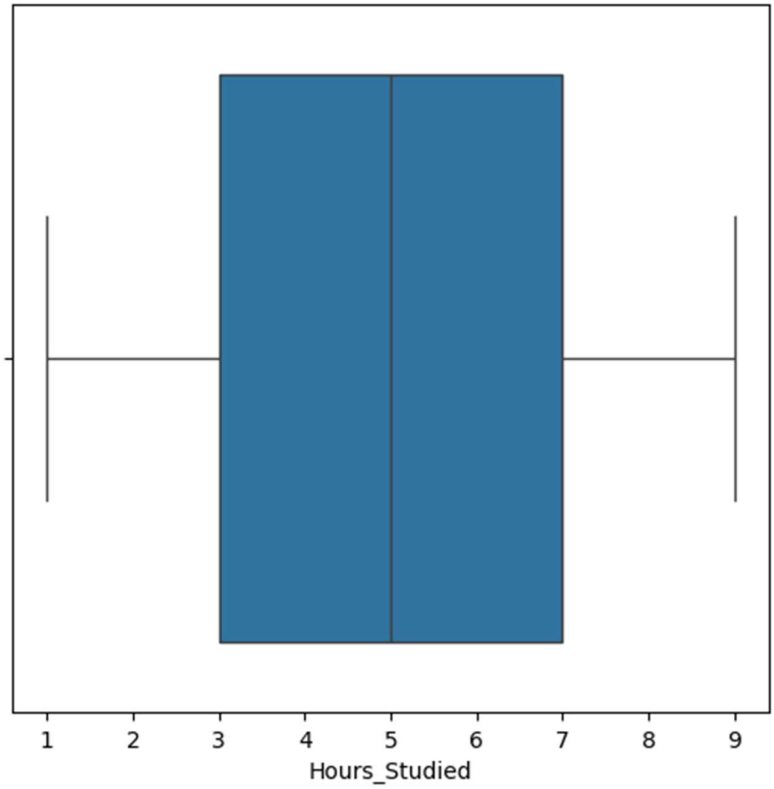
```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns


# Import warnings

import warnings

warnings.filterwarnings("ignore")


# We will use some methods from the sklearn module

from sklearn import linear_model

from sklearn.linear_model import LinearRegression

from sklearn import metrics

from sklearn.metrics import mean_squared_error, mean_absolute_error

from sklearn.model_selection import train_test_split, cross_val_score


# Load the dataset

df = pd.read_csv("Student_Performance.csv")

print(df.corr(numeric_only=True))  # Use numeric_only=True to only get numeric
correlations

print(df.describe())


# Setting the value for X and y

X = df[['Hours_Studied', 'Previous_Scores']]  # Adjusted features
```

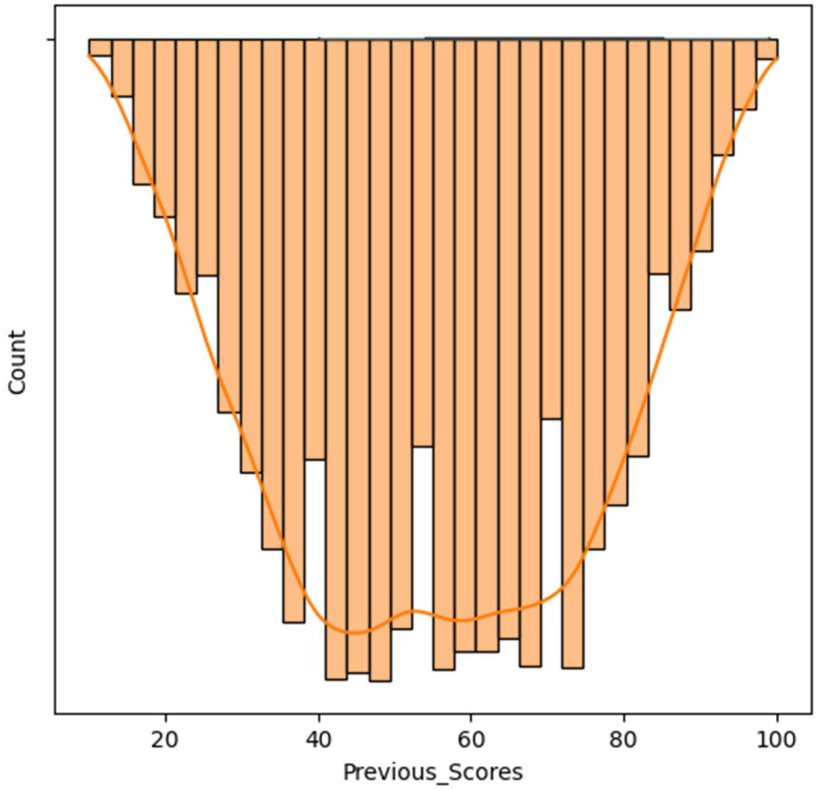
```
y = df['Performance_Index']  # Adjusted target variable


# Visualizing the features
fig, axs = plt.subplots(2, figsize=(5, 10))  # Adjusted height for better spacing
plt1 = sns.boxplot(x=df['Hours_Studied'], ax=axs[0])  # Adjusted for the new feature
plt2 = sns.boxplot(x=df['Previous_Scores'], ax=axs[1])  # Adjusted for the new feature
plt.tight_layout()


sns.histplot(df['Performance_Index'], kde=True)  # Changed to histplot with kde for better visualization


sns.pairplot(df, x_vars=['Hours_Studied', 'Previous_Scores'], y_vars='Performance_Index', height=4, aspect=1, kind='scatter')
plt.show()


# Create the correlation matrix and represent it as a heatmap.
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap='coolwarm')  # Use numeric_only=True for heatmap
plt.show()


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=100)


# Fitting the Multiple Linear Regression model
reg_model = LinearRegression().fit(X_train, y_train)


# Printing the model coefficients
print('Intercept: ', reg_model.intercept_)
```

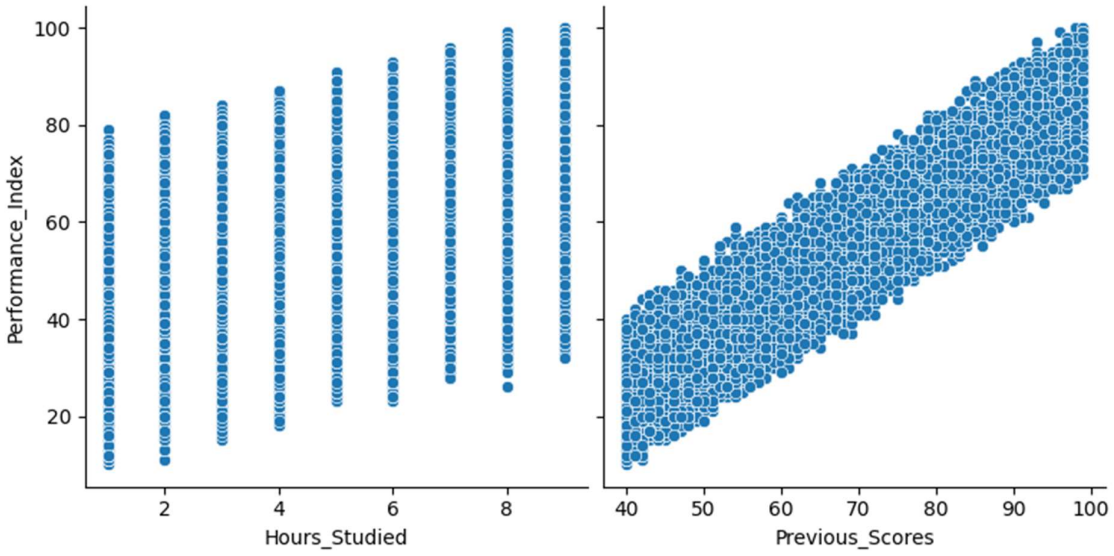
```
# Pair the feature names with the coefficients
print(list(zip(X.columns, reg_model.coef_)))  # Corrected to use .columns for feature names


# Predicting the Test and Train set result
y_pred = reg_model.predict(X_test)
x_pred = reg_model.predict(X_train)  # This is actually predicting on the train set


print("Prediction for test set: {}".format(y_pred))


# Actual value and the predicted value
reg_model_diff = pd.DataFrame({'Actual value': y_test, 'Predicted value': y_pred})
print(reg_model_diff)  # Added print to display the DataFrame


mae = metrics.mean_absolute_error(y_test, y_pred)
mse = metrics.mean_squared_error(y_test, y_pred)
r2 = np.sqrt(mse)  # Changed to use mse instead of mean_squared_error directly


print('Mean Absolute Error:', mae)
print('Mean Square Error:', mse)
print('Root Mean Square Error:', r2)
```

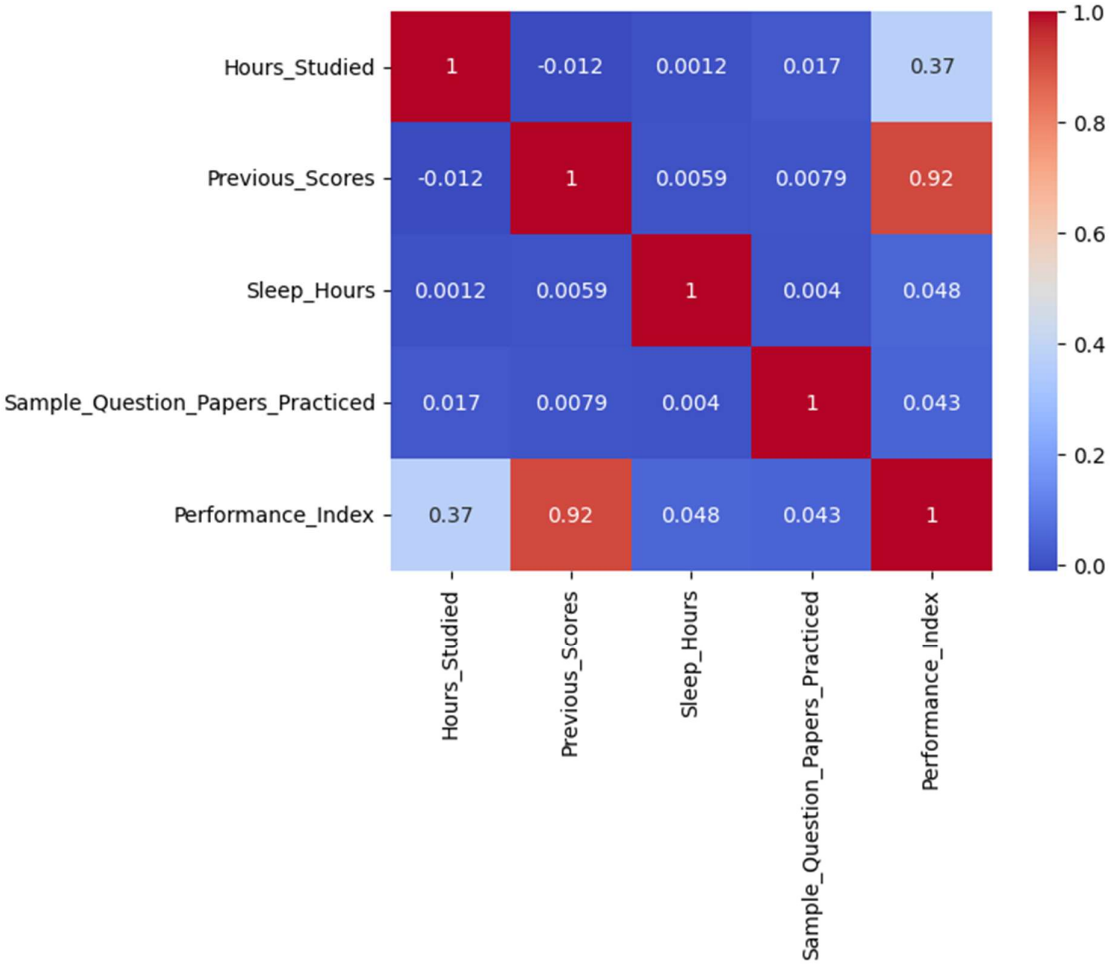
Code 01 Output :-

```
                                  Hours_Studied  Previous_Scores  Sleep_Hours  \
Hours_Studied                          1.000000        -0.012390     0.001245
Previous_Scores                       -0.012390         1.000000     0.005944
Sleep_Hours                            0.001245         0.005944     1.000000
Sample_Question_Papers_Practiced       0.017463         0.007888     0.003990
Performance_Index                      0.373730         0.915189     0.048106

                                  Sample_Question_Papers_Practiced  \
Hours_Studied                                             0.017463
Previous_Scores                                           0.007888
Sleep_Hours                                               0.003990
Sample_Question_Papers_Practiced                          1.000000
Performance_Index                                         0.043268

                                  Performance_Index
Hours_Studied                              0.373730
Previous_Scores                            0.915189
Sleep_Hours                                0.048106
Sample_Question_Papers_Practiced           0.043268
Performance_Index                          1.000000
       Hours_Studied  Previous_Scores   Sleep_Hours  \
count   10000.000000     10000.000000  10000.000000
mean        4.992900        69.445700      6.530600
std         2.589309        17.343152      1.695863
min         1.000000        40.000000      4.000000
25%         3.000000        54.000000      5.000000
50%         5.000000        69.000000      7.000000
75%         7.000000        85.000000      8.000000
max         9.000000        99.000000      9.000000

       Sample_Question_Papers_Practiced  Performance_Index
count                      10000.000000       10000.000000
mean                           4.583300          55.224800
std                            2.867348          19.212558
min                            0.000000          10.000000
25%                            2.000000          40.000000
50%                            5.000000          55.000000
75%                            7.000000          71.000000
max                            9.000000         100.000000
```


1
2
3
4
5
6
7
8
9
Hours_Studied


Count
20
40
60
80
100
Previous_Scores


Performance_Index
100
80
60
40
20
2
4
6
8
Hours_Studied
40
50
60
70
80
90
100
Previous_Scores


Hours_Studied
Previous_Scores
Sleep_Hours
Sample_Question_Papers_Practiced
Performance_Index
1
-0.012
0.0012
0.017
0.37
-0.012
1
0.0059
0.0079
0.92
0.0012
0.0059
1
0.004
0.048
0.017
0.0079
0.004
1
0.043
0.37
0.92
0.048
0.043
1
1.0
0.8
0.6
0.4
0.2
0.0

```
Intercept:  -29.759696031285607
[('Hours_Studied', np.float64(2.8458785973597283)), ('Previous_Scores', np.float64(1.0192750395624741))]
Prediction for test set: [65.62826464 49.70329949 34.41417389 ... 71.9558614  35.22150241
 14.87645919]
      Actual value  Predicted value
8018            65        65.628265
9225            51        49.703299
3854            35        34.414174
2029            97        92.896287
3539            87        84.530140
...            ...              ...
4208            69        67.242922
5335            60        58.705286
2554            73        71.955861
13              33        35.221502
4468            15        14.876459

[3000 rows x 2 columns]
Mean Absolute Error: 1.8372246647574142
Mean Square Error: 5.3409802420571
Root Mean Square Error: 2.3110560880379127
```

Code 02 :-

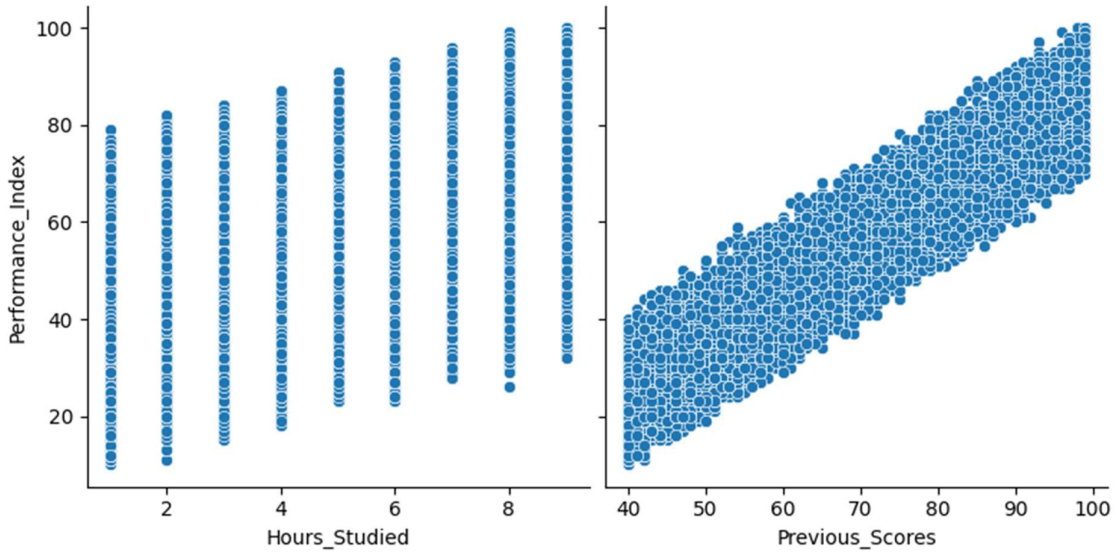
```
# Import necessary libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn import linear_model

from sklearn.model_selection import train_test_split

from sklearn import metrics


# Load the dataset

df = pd.read_csv("Student_Performance.csv")


# Prepare the features and target variable

X = df[['Hours_Studied', 'Previous_Scores']]  # Adjusted features
```

```
y = df['Performance_Index']  # Adjusted target variable


# Perform exploratory data analysis

print(df.corr(numeric_only=True))  # Only get numeric correlations

sns.pairplot(df, x_vars=['Hours_Studied', 'Previous_Scores'], y_vars='Performance_Index',
height=4, aspect=1, kind='scatter')

plt.show()


# Split the data into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)


# Create and fit the multivariable regression model

reg_model = linear_model.LinearRegression()

reg_model.fit(X_train, y_train)


# Print model coefficients

print('Intercept:', reg_model.intercept_)

print('Coefficients:', reg_model.coef_)


# Make predictions

y_pred = reg_model.predict(X_test)


# Evaluate the model

mae = metrics.mean_absolute_error(y_test, y_pred)

mse = metrics.mean_squared_error(y_test, y_pred)

r2 = metrics.r2_score(y_test, y_pred)
```

```
print('Mean Absolute Error:', mae)

print('Mean Squared Error:', mse)

print('R-squared:', r2)


# Visualize actual vs predicted values

plt.scatter(y_test, y_pred)

plt.xlabel('Actual Performance Index')

plt.ylabel('Predicted Performance Index')

plt.title('Actual vs Predicted Performance Index')

plt.plot([y.min(), y.max()], [y.min(), y.max()], 'r--')  # 45-degree line

plt.show()
```

Code 02 Output: -

```
                                Hours_Studied  Previous_Scores  Sleep_Hours  \
Hours_Studied                        1.000000        -0.012390     0.001245
Previous_Scores                     -0.012390         1.000000     0.005944
Sleep_Hours                          0.001245         0.005944     1.000000
Sample_Question_Papers_Practiced     0.017463         0.007888     0.003990
Performance_Index                    0.373730         0.915189     0.048106

                                Sample_Question_Papers_Practiced  \
Hours_Studied                                           0.017463
Previous_Scores                                         0.007888
Sleep_Hours                                             0.003990
Sample_Question_Papers_Practiced                        1.000000
Performance_Index                                       0.043268

                                Performance_Index
Hours_Studied                            0.373730
Previous_Scores                          0.915189
Sleep_Hours                              0.048106
Sample_Question_Papers_Practiced         0.043268
Performance_Index                        1.000000
```




```
Intercept: -29.60536090045995
Coefficients: [2.86449217 1.01615225]
Mean Absolute Error: 1.8101437492369952
Mean Squared Error: 5.201346140631689
R-squared: 0.9859445764337783
```