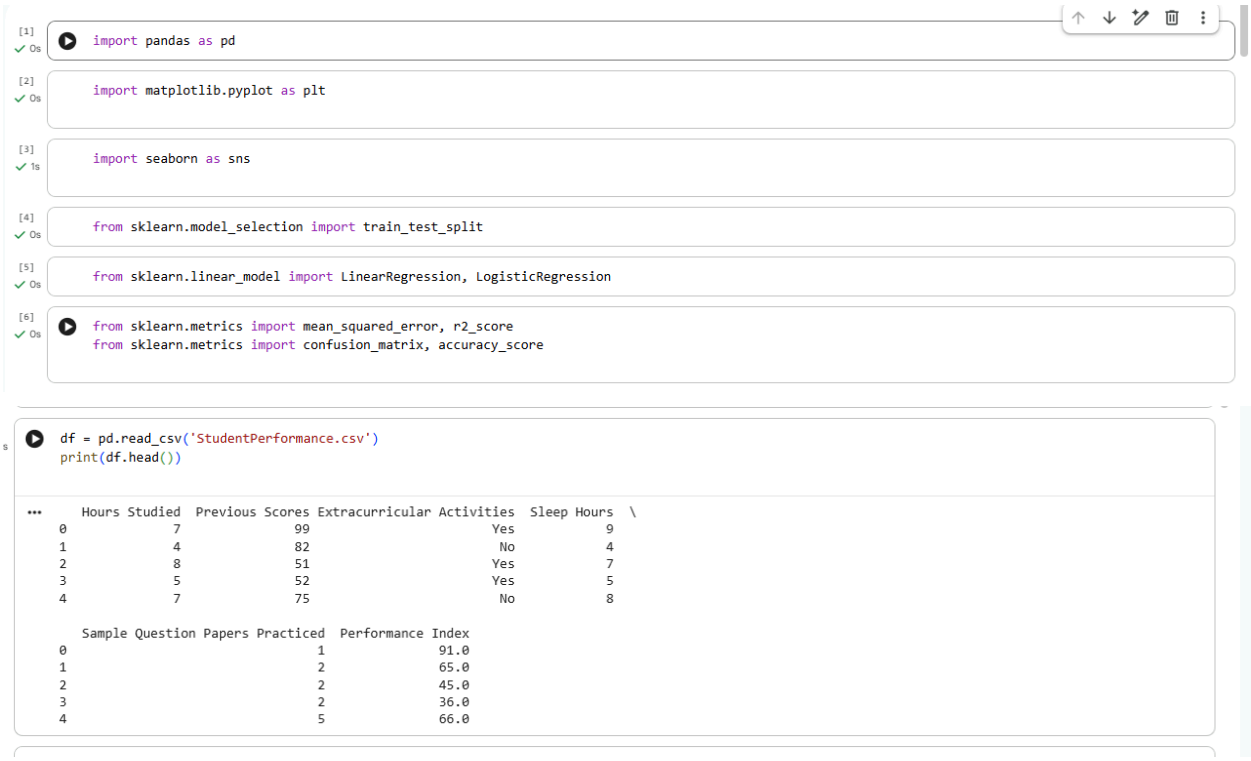
Experiment 1 : Implementation of Linear and Logistic Regression on a Real-World Dataset

```
# ================================
# STEP 1: Import Required Libraries
# ================================
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.metrics import confusion_matrix, accuracy_score

# ================================
# STEP 2: Load Dataset
# ================================
df = pd.read_csv('/mnt/data/student_performance (1).csv')
print(df.head())

# ================================
# STEP 3: LINEAR REGRESSION
# Predict Final_Score
# ================================
X = df[['Hours_Studied', 'Attendance', 'Assignment_Score', 'Midterm_Score']]
y = df['Final_Score']

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

lr = LinearRegression()
lr.fit(X_train, y_train)

y_pred = lr.predict(X_test)

# ----------------
# Graph 1: Actual vs Predicted
# ----------------
plt.figure()
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Final Score")
plt.ylabel("Predicted Final Score")
```

```
plt.title("Linear Regression: Actual vs Predicted")
plt.show()

# ----------------
# Graph 2: Residual Plot
# ----------------
residuals = y_test - y_pred

plt.figure()
plt.scatter(y_pred, residuals)
plt.axhline(0)
plt.xlabel("Predicted Final Score")
plt.ylabel("Residuals")
plt.title("Residual Plot")
plt.show()

# ================================
# STEP 4: LOGISTIC REGRESSION
# Pass / Fail Classification
# ================================
df['Pass'] = (df['Final_Score'] >= 60).astype(int)

X = df[['Hours_Studied', 'Attendance', 'Assignment_Score', 'Midterm_Score']]
y = df['Pass']

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

log_reg = LogisticRegression(max_iter=1000)
log_reg.fit(X_train, y_train)

y_pred = log_reg.predict(X_test)

# ----------------
# Graph 3: Confusion Matrix
# ----------------
cm = confusion_matrix(y_test, y_pred)

plt.figure()
sns.heatmap(cm, annot=True, fmt='d')
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Logistic Regression Confusion Matrix")
```

```
plt.show()

# ----------------
# Graph 4: Probability Distribution
# ----------------
y_prob = log_reg.predict_proba(X_test)[:, 1]

plt.figure()
plt.hist(y_prob, bins=10)
plt.xlabel("Probability of Passing")
plt.ylabel("Count")
plt.title("Pass Probability Distribution")
plt.show()

# ================================
# STEP 5: Accuracy
# ================================
accuracy = accuracy_score(y_test, y_pred)
print("Logistic Regression Accuracy:", accuracy)
```

`OUTPUT :



```
[1] import pandas as pd
[2] import matplotlib.pyplot as plt
[3] import seaborn as sns
[4] from sklearn.model_selection import train_test_split
[5] from sklearn.linear_model import LinearRegression, LogisticRegression
[6] from sklearn.metrics import mean_squared_error, r2_score
    from sklearn.metrics import confusion_matrix, accuracy_score

df = pd.read_csv('StudentPerformance.csv')
print(df.head())

   Hours Studied  Previous Scores Extracurricular Activities  Sleep Hours  \
0              7               99                         Yes            9
1              4               82                          No            4
2              8               51                         Yes            7
3              5               52                         Yes            5
4              7               75                          No            8

   Sample Question Papers Practiced  Performance Index
0                                 1               91.0
1                                 2               65.0
2                                 2               45.0
3                                 2               36.0
4                                 5               66.0
```

[11]
✓ 0s

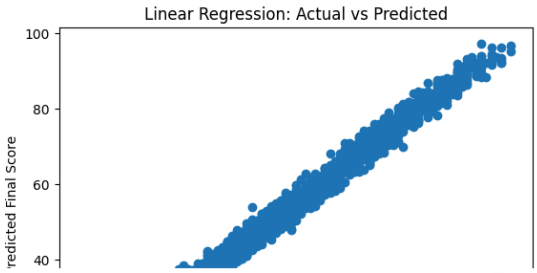
```
X = df[['Hours Studied', 'Previous Scores', 'Sleep Hours', 'Sample Question Papers Practiced']]
y = df['Performance Index']

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

lr = LinearRegression()
lr.fit(X_train, y_train)

y_pred = lr.predict(X_test)
```

[12]

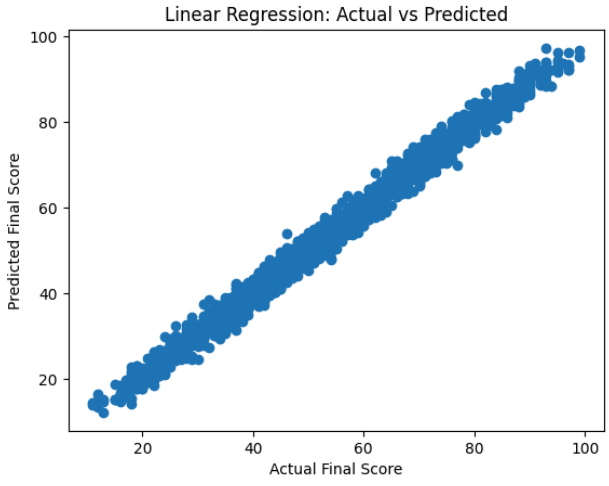
```
plt.figure()
```

[12]
✓ 0s

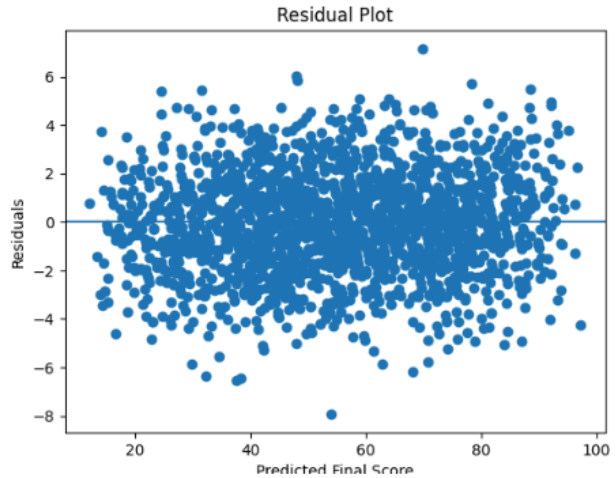
```
plt.figure()
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Final Score")
plt.ylabel("Predicted Final Score")
plt.title("Linear Regression: Actual vs Predicted")
plt.show()
```



?

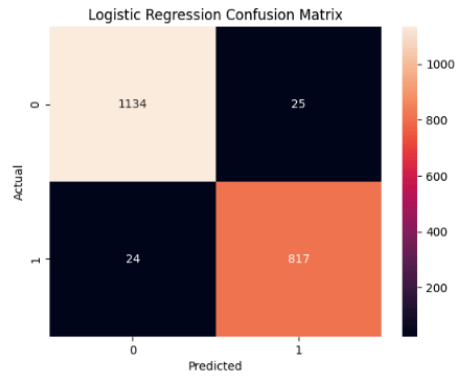


[13]
```
residuals = y_test - y_pred

plt.figure()
plt.scatter(y_pred, residuals)
plt.axhline(0)
plt.xlabel("Predicted Final Score")
plt.ylabel("Residuals")
plt.title("Residual Plot")
plt.show()
```



[15]
```
df['Pass'] = (df['Performance Index'] >= 60).astype(int)

X = df[['Hours Studied', 'Previous Scores', 'Sleep Hours', 'Sample Question Papers Practiced']]
y = df['Pass']

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

log_reg = LogisticRegression(max_iter=1000)
log_reg.fit(X_train, y_train)

y_pred = log_reg.predict(X_test)
```

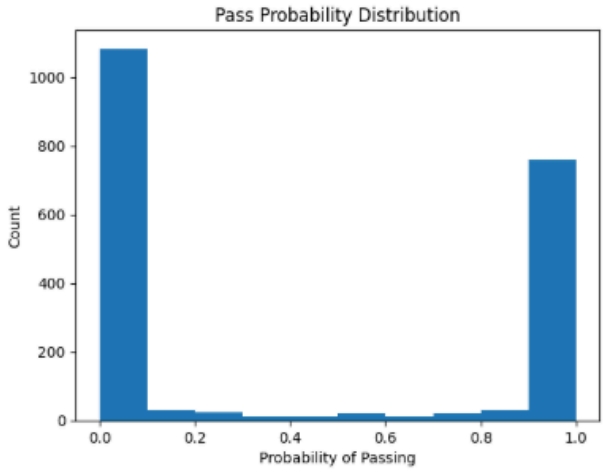
[16]
```
cm = confusion_matrix(y_test, y_pred)

plt.figure()
sns.heatmap(cm, annot=True, fmt='d')
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Logistic Regression Confusion Matrix")
plt.show()
```



[17]
```
y_prob = log_reg.predict_proba(X_test)[:, 1]
```

[17]
✓ 0s

```
y_prob = log_reg.predict_proba(X_test)[:, 1]

plt.figure()
plt.hist(y_prob, bins=10)
plt.xlabel("Probability of Passing")
plt.ylabel("Count")
plt.title("Pass Probability Distribution")
plt.show()
```



+ Code + Text

18]
✓ 0s

```
accuracy = accuracy_score(y_test, y_pred)
print("Logistic Regression Accuracy:", accuracy)
```

```
Logistic Regression Accuracy: 0.9755
```