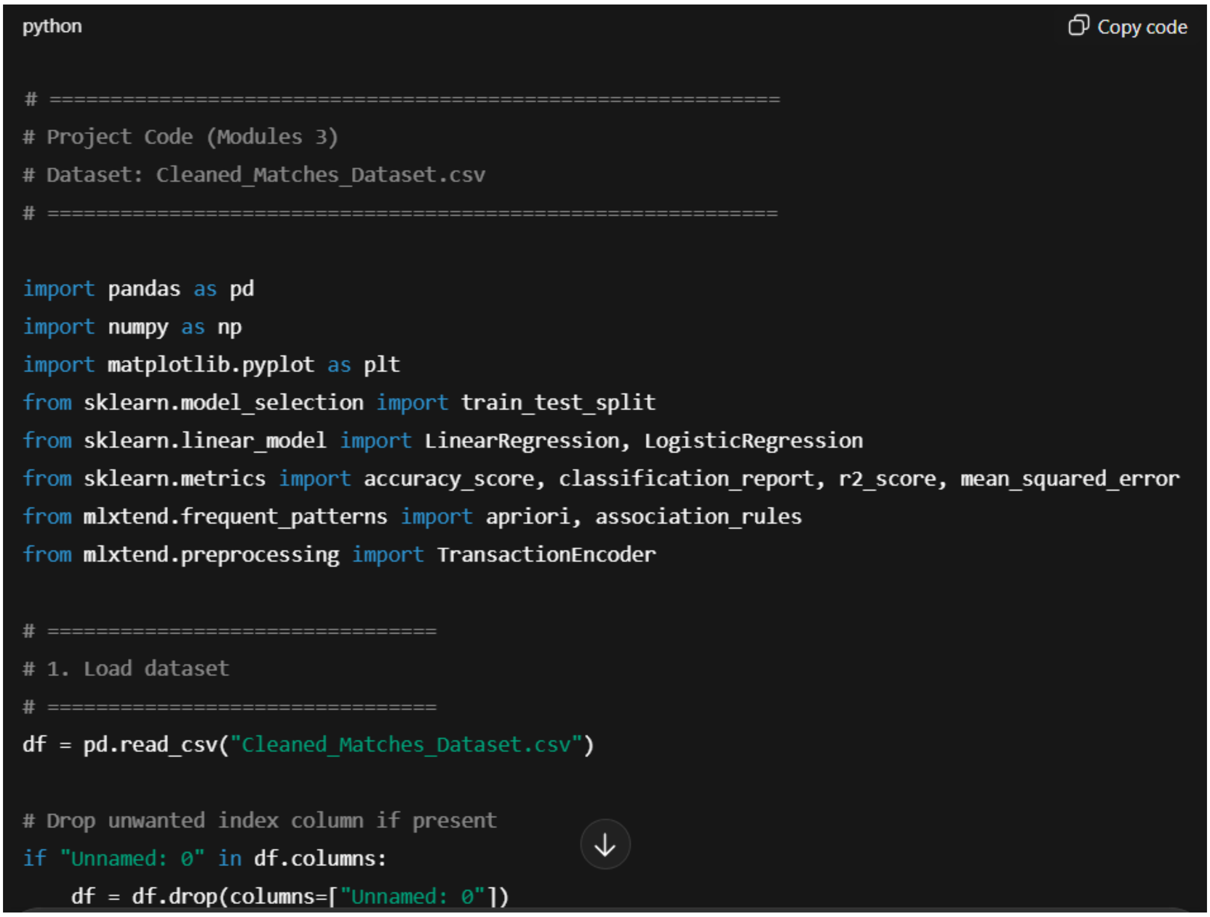
Association Rule Mining

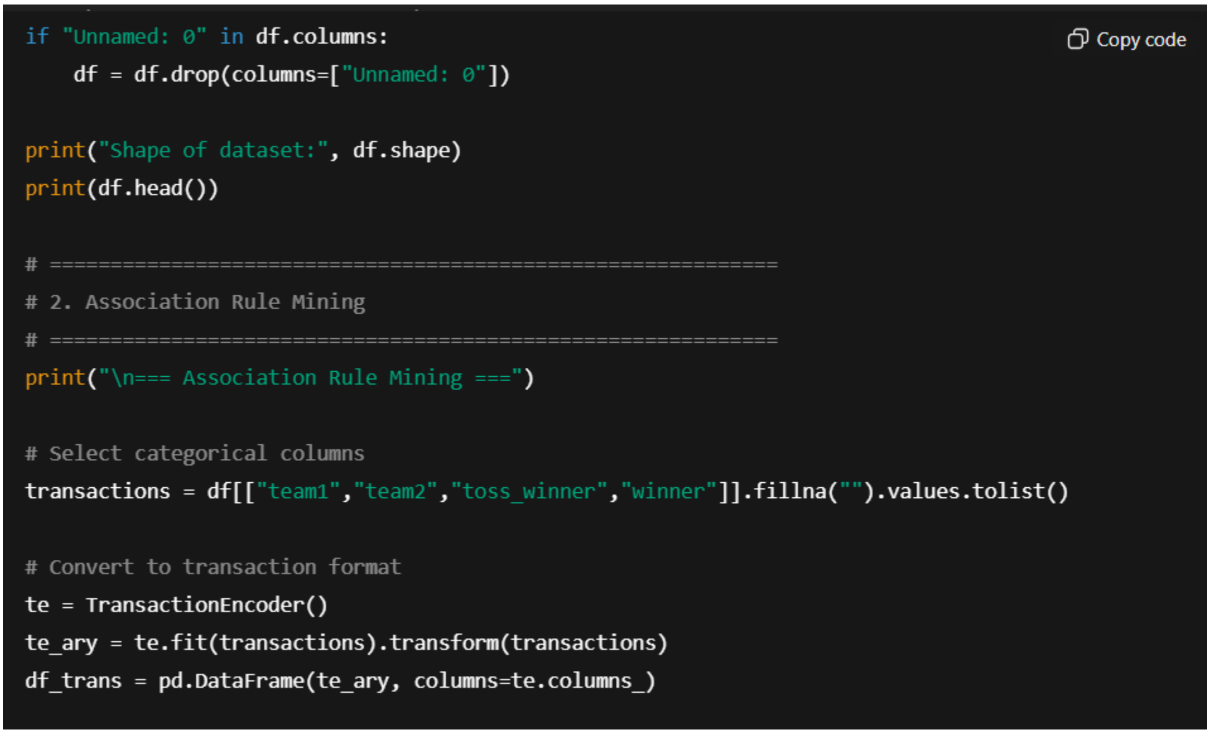
Regression

Least Square Regression

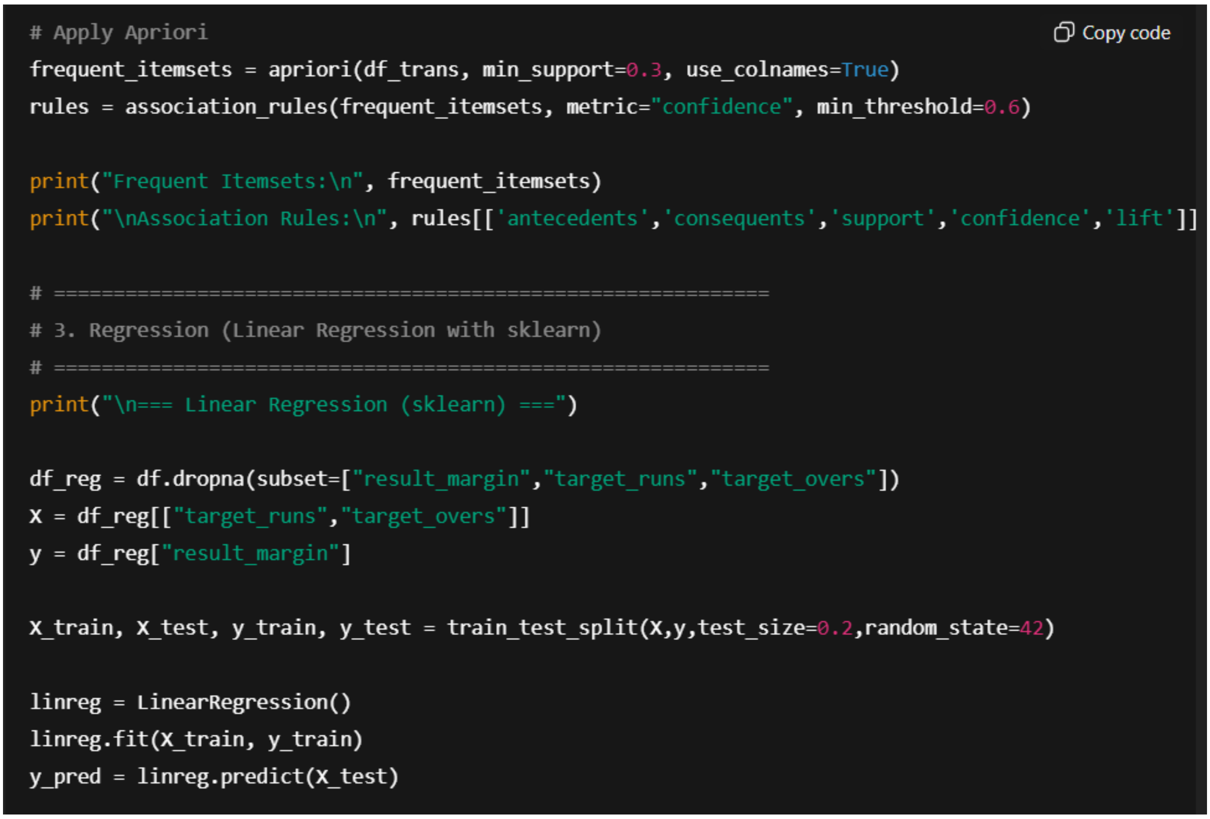
Logistic Regression



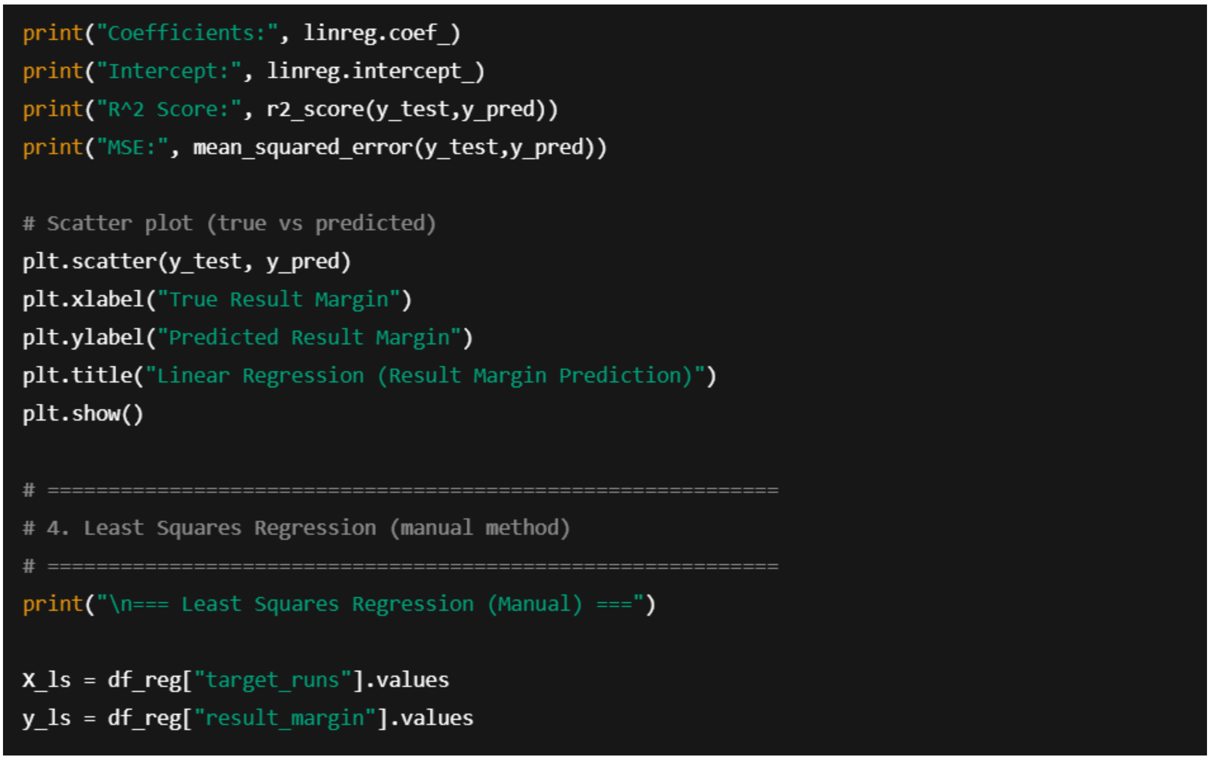
```python
# ============================================================
# Project Code (Modules 3)
# Dataset: Cleaned_Matches_Dataset.csv
# ============================================================

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, r2_score, mean_squared_error
from mlxtend.frequent_patterns import apriori, association_rules
from mlxtend.preprocessing import TransactionEncoder

# ================================
# 1. Load dataset
# ================================
df = pd.read_csv("Cleaned_Matches_Dataset.csv")

# Drop unwanted index column if present
if "Unnamed: 0" in df.columns:
    df = df.drop(columns=["Unnamed: 0"])
```



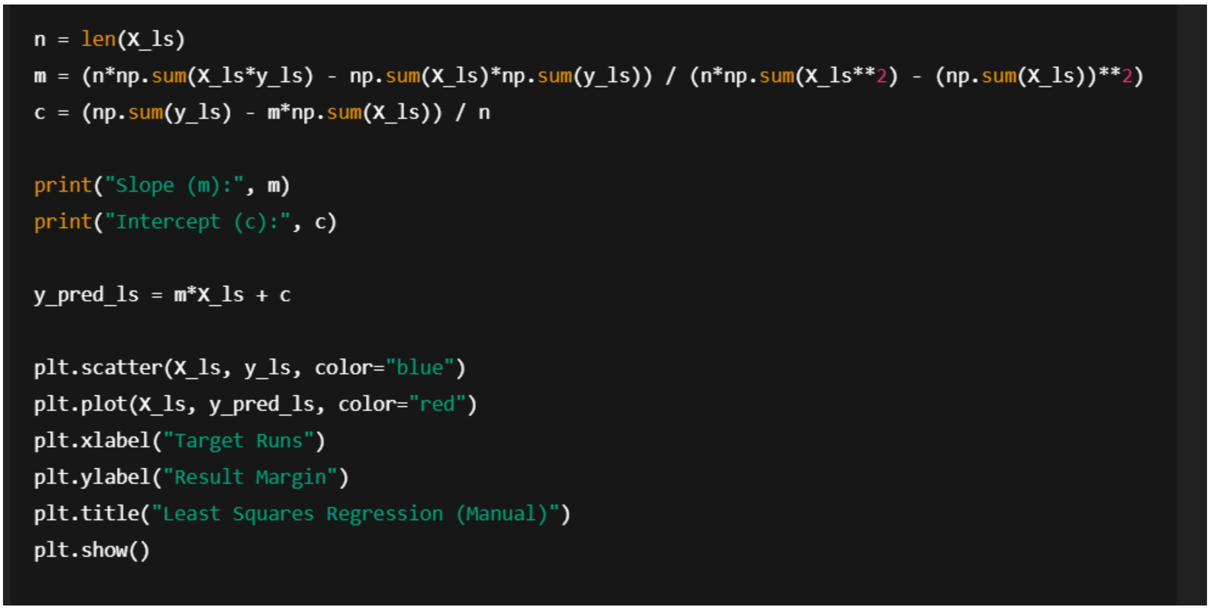
```python
if "Unnamed: 0" in df.columns:
    df = df.drop(columns=["Unnamed: 0"])

print("Shape of dataset:", df.shape)
print(df.head())

# ============================================================
# 2. Association Rule Mining
# ============================================================
print("\n=== Association Rule Mining ===")

# Select categorical columns
transactions = df[["team1","team2","toss_winner","winner"]].fillna("").values.tolist()

# Convert to transaction format
te = TransactionEncoder()
te_ary = te.fit(transactions).transform(transactions)
df_trans = pd.DataFrame(te_ary, columns=te.columns_)
```



```python
# Apply Apriori
frequent_itemsets = apriori(df_trans, min_support=0.3, use_colnames=True)
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.6)

print("Frequent Itemsets:\n", frequent_itemsets)
print("\nAssociation Rules:\n", rules[['antecedents','consequents','support','confidence','lift']]

# ============================================================
# 3. Regression (Linear Regression with sklearn)
# ============================================================
print("\n=== Linear Regression (sklearn) ===")

df_reg = df.dropna(subset=["result_margin","target_runs","target_overs"])
X = df_reg[["target_runs","target_overs"]]
y = df_reg["result_margin"]

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

linreg = LinearRegression()
linreg.fit(X_train, y_train)
y_pred = linreg.predict(X_test)
```

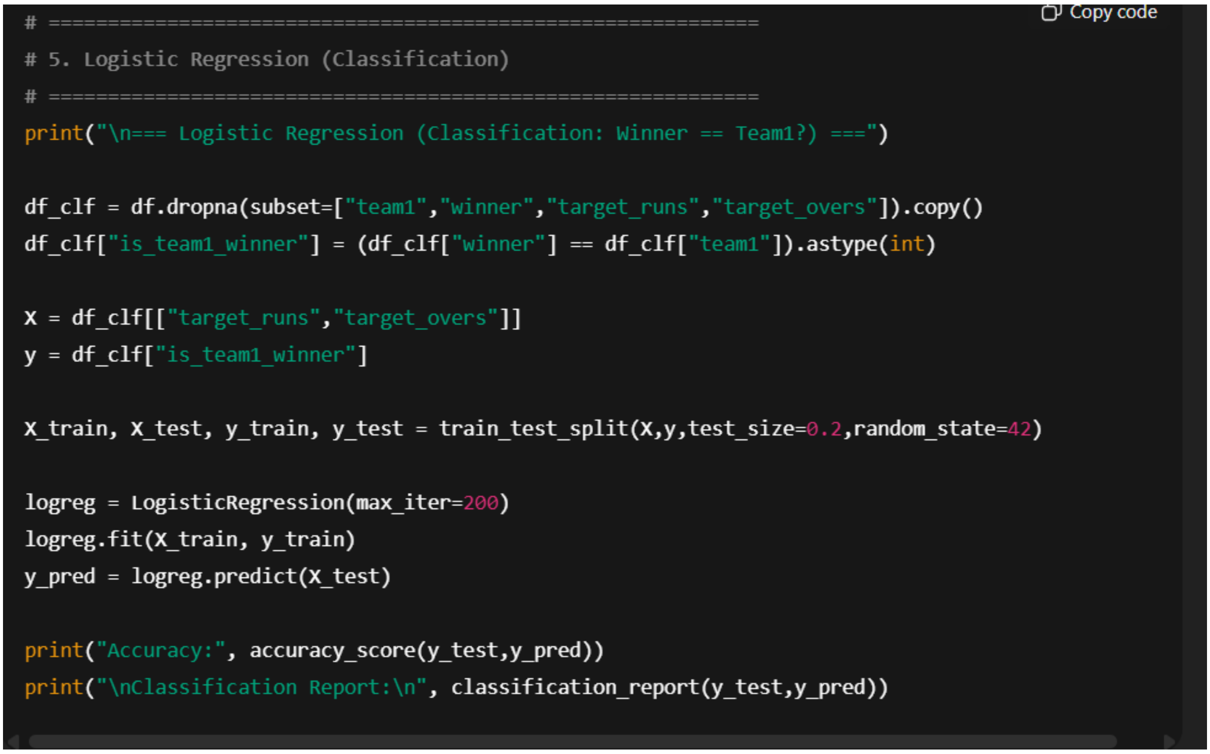


```
print("Coefficients:", linreg.coef_)
print("Intercept:", linreg.intercept_)
print("R^2 Score:", r2_score(y_test,y_pred))
print("MSE:", mean_squared_error(y_test,y_pred))

# Scatter plot (true vs predicted)
plt.scatter(y_test, y_pred)
plt.xlabel("True Result Margin")
plt.ylabel("Predicted Result Margin")
plt.title("Linear Regression (Result Margin Prediction)")
plt.show()

# ============================================================
# 4. Least Squares Regression (manual method)
# ============================================================
print("\n=== Least Squares Regression (Manual) ===")

X_ls = df_reg["target_runs"].values
y_ls = df_reg["result_margin"].values
```

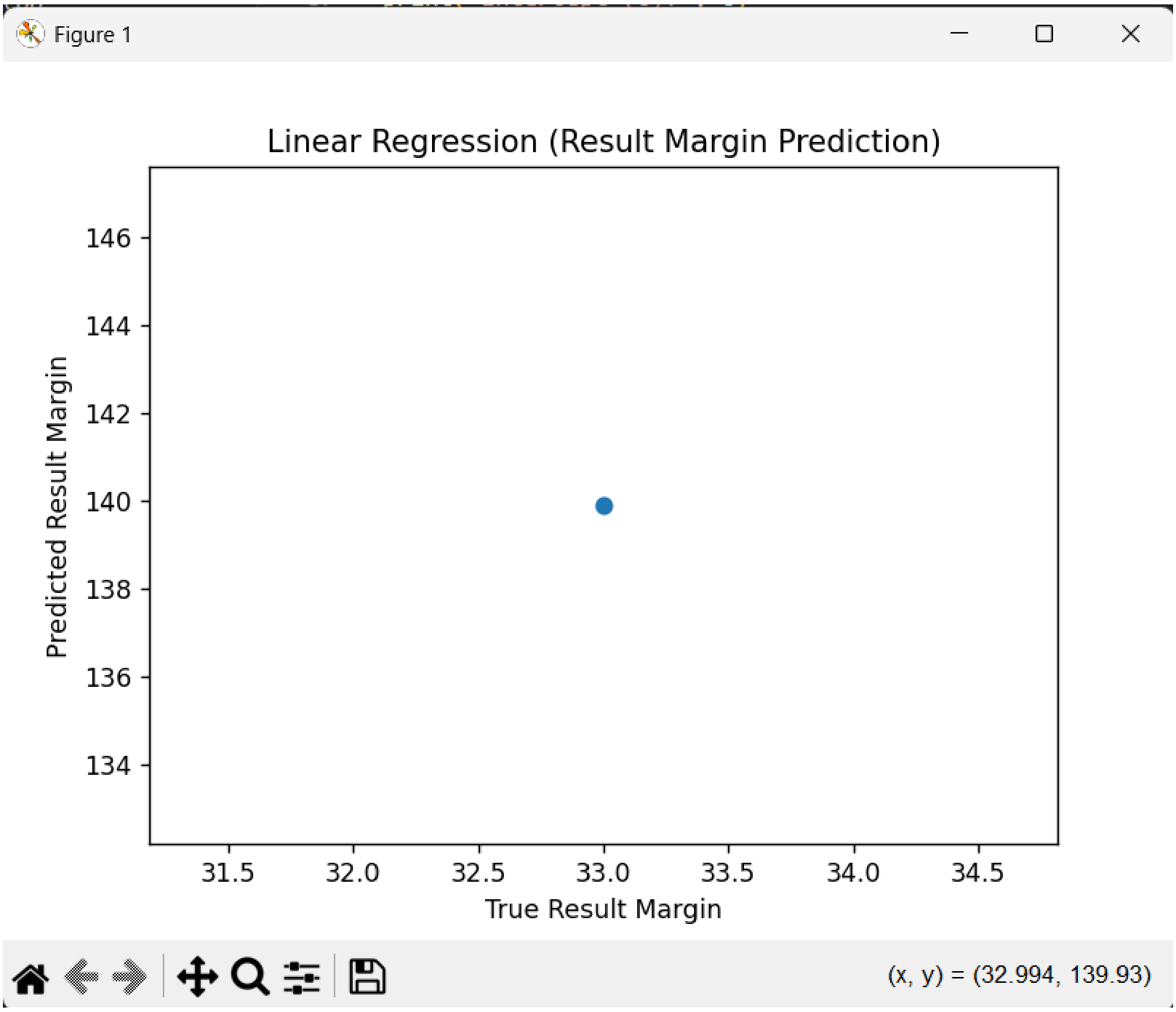


```
n = len(X_ls)
m = (n*np.sum(X_ls*y_ls) - np.sum(X_ls)*np.sum(y_ls)) / (n*np.sum(X_ls**2) - (np.sum(X_ls))**2)
c = (np.sum(y_ls) - m*np.sum(X_ls)) / n

print("Slope (m):", m)
print("Intercept (c):", c)

y_pred_ls = m*X_ls + c

plt.scatter(X_ls, y_ls, color="blue")
plt.plot(X_ls, y_pred_ls, color="red")
plt.xlabel("Target Runs")
plt.ylabel("Result Margin")
plt.title("Least Squares Regression (Manual)")
plt.show()
```

Guys this above screenshot(ss) are codes write only important functions only and for importing different library I will give you another complete text file with every requiremts.



```python
# ============================================================
# 5. Logistic Regression (Classification)
# ============================================================
print("\n=== Logistic Regression (Classification: Winner == Team1?) ===")

df_clf = df.dropna(subset=["team1","winner","target_runs","target_overs"]).copy()
df_clf["is_team1_winner"] = (df_clf["winner"] == df_clf["team1"]).astype(int)

X = df_clf[["target_runs","target_overs"]]
y = df_clf["is_team1_winner"]

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

logreg = LogisticRegression(max_iter=200)
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)

print("Accuracy:", accuracy_score(y_test,y_pred))
print("\nClassification Report:\n", classification_report(y_test,y_pred))
```



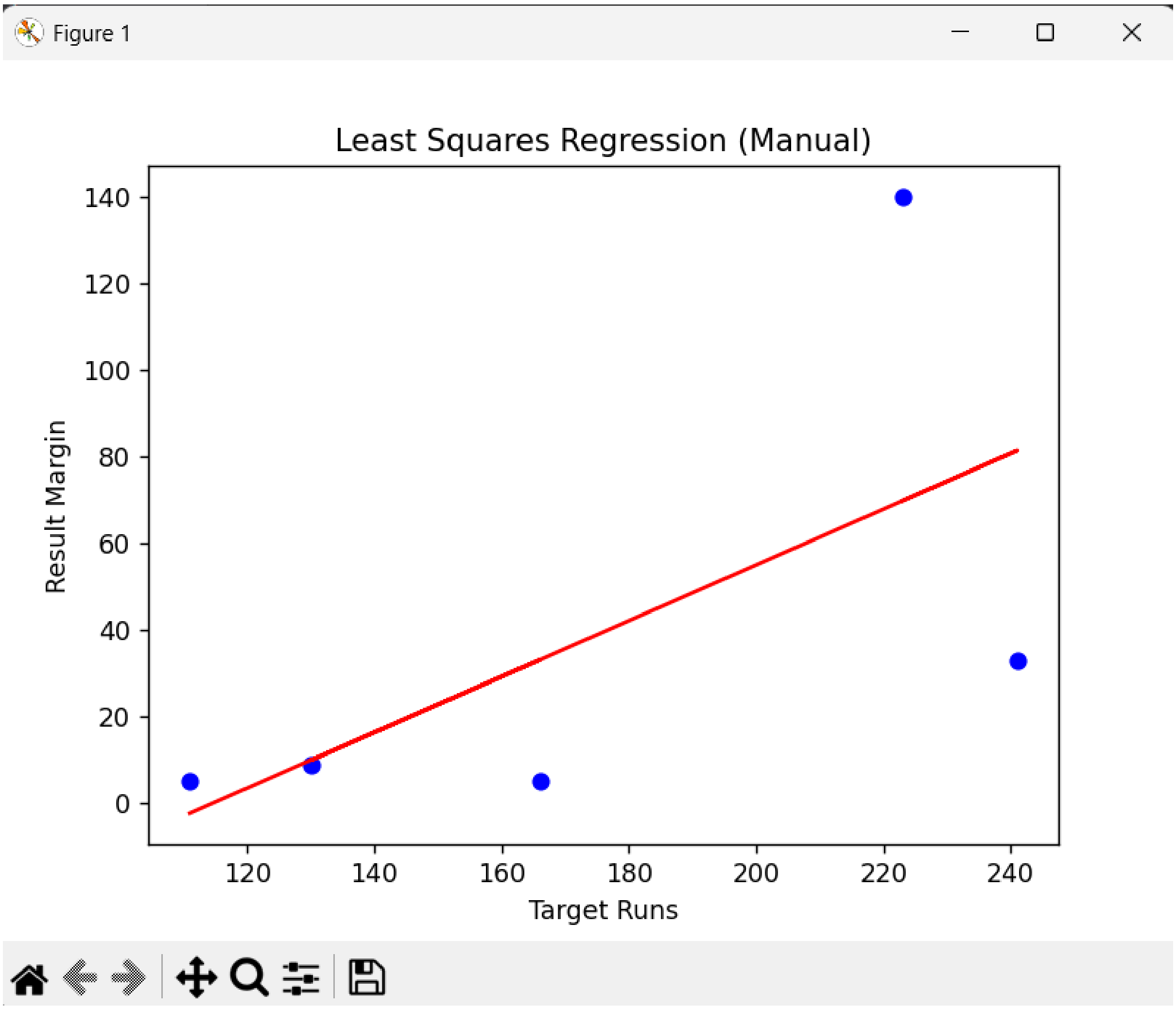


Figure 1
Least Squares Regression (Manual)
Result Margin
140
120
100
80
60
40
20
0
120
140
160
180
200
220
240
Target Runs