Experiment No. 2

Title: Study and Implement Multiple Linear Regression

Objectives:

1. Develop a machine learning model to predict calories burned based on exercise-related metrics.
2. Perform data preprocessing (handling missing values, encoding categorical variables).
3. Train and evaluate a Linear Regression model for calorie prediction.
4. Measure model performance using Mean Squared Error (MSE) and R² score.
5. Visualize actual vs. predicted values to assess model accuracy.

Problem Statement:

This practical builds a data-driven approach using linear regression to predict calorie expenditure based on exercise parameters such as duration, pulse rate, and max pulse.

Outcomes:

1. A trained Linear Regression model that predicts calorie expenditure.
2. Evaluation metrics (MSE, R²) to assess model accuracy.
3. A scatter plot of actual vs. predicted values to visualize prediction quality.

Tools Required: 4GB RAM, Anaconda, Notebook

Theory:

Multiple Linear Regression (MLR) is an extension of simple linear regression, where a dependent variable (Y) is predicted based on multiple independent variables (X1, X2, ..., Xn). The equation of MLR is:

$Y=\beta_0+\beta_1X_1+\beta_2X_2+\ldots..\beta_nX_n+€$

Where:

- Y= Dependent variable
- $X_1,X_2,\ldots.X_n$= Independent variables
- β_0= Intercept
- $\beta_1,\beta_2,\ldots.\beta_n$= Coefficients of independent variables
- €= Error term

The model learns these coefficients by minimizing the Mean Squared Error (MSE):

$$MSE=\frac{1}{n}\sum_{i=1}^{n}(Y-\acute{Y})^2$$

where Y is the actual value and Ý is the predicted value.

The goodness-of-fit is measured by R² score:

$R^2=1-\frac{SS_{res}}{SS_{tot}}$

where SSres is the residual sum of squares and SStot is the total sum of squares.

Algorithm:

Step 1: Load Dataset

- Read data.csv using pandas.
- Display the first few rows to understand the structure.

Step 2: Data Preprocessing

- Handle missing values (drop or impute).
- Select independent (X) and dependent (y) variables.
- Convert categorical features using pd.get_dummies().

Step 3: Split Data

- Use train_test_split() to split into training (80%) and testing (20%) sets.

Step 4: Train Linear Regression Model

- Initialize LinearRegression() from sklearn.
- Fit the model using X_train and y_train.

Step 5: Make Predictions

- Use model.predict(X_test) to generate predictions.

Step 6: Evaluate Model

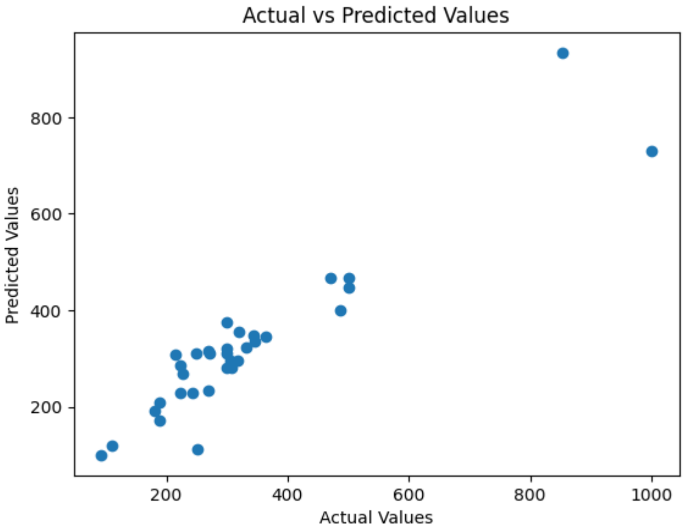
- Compute MSE and R² score to measure accuracy.

Source Code:

```
import numpy as np
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
from sklearn.metrics import mean_squared_error, r2_score
df = pd.read_csv("data.csv")  # Replace with actual dataset
print(df.head())
df = df.dropna()
X = df[['Duration', 'Pulse', 'Maxpulse',]]  # Replace with actual feature names
y = df['Calories']
X=pd.get_dummies(X, drop_first=True)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
print(f"R-squared Value: {r2}")
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Values")
plt.ylabel("Predicted Values")
plt.title("Actual vs Predicted Values")
plt.show()
```

Output:

```
Duration  Pulse  Maxpulse  Calories
0        60    110       130    409.1
1        60    117       145    479.0
2        60    103       135    340.0
3        45    109       175    282.4
4        45    117       148    406.0
Mean Squared Error: 4354.413282161514
R-squared Value: 0.8664689420685995
```



Conclusion:

____

____

____

____