# LAB NO: 02

## Task 1: Simple Linear Regression

**1. Load dataset:**

Use the Diabetes dataset from sklearn.datasets.

Select one feature (bmi) to predict the target (disease progression).

**2. Perform Exploratory Data Analysis (EDA):**

o Plot scatter plot of BMI vs. Disease Progression.

o Check correlation.

**3. Implement Simple Linear Regression using sklearn.linear_model.LinearRegression:**

o Split data into training and testing sets.

o Fit the model and predict disease progression.

o Plot the regression line on the scatter plot.

**4. Evaluate the model using:**

o Mean Squared Error (MSE)

o $R^2$ score

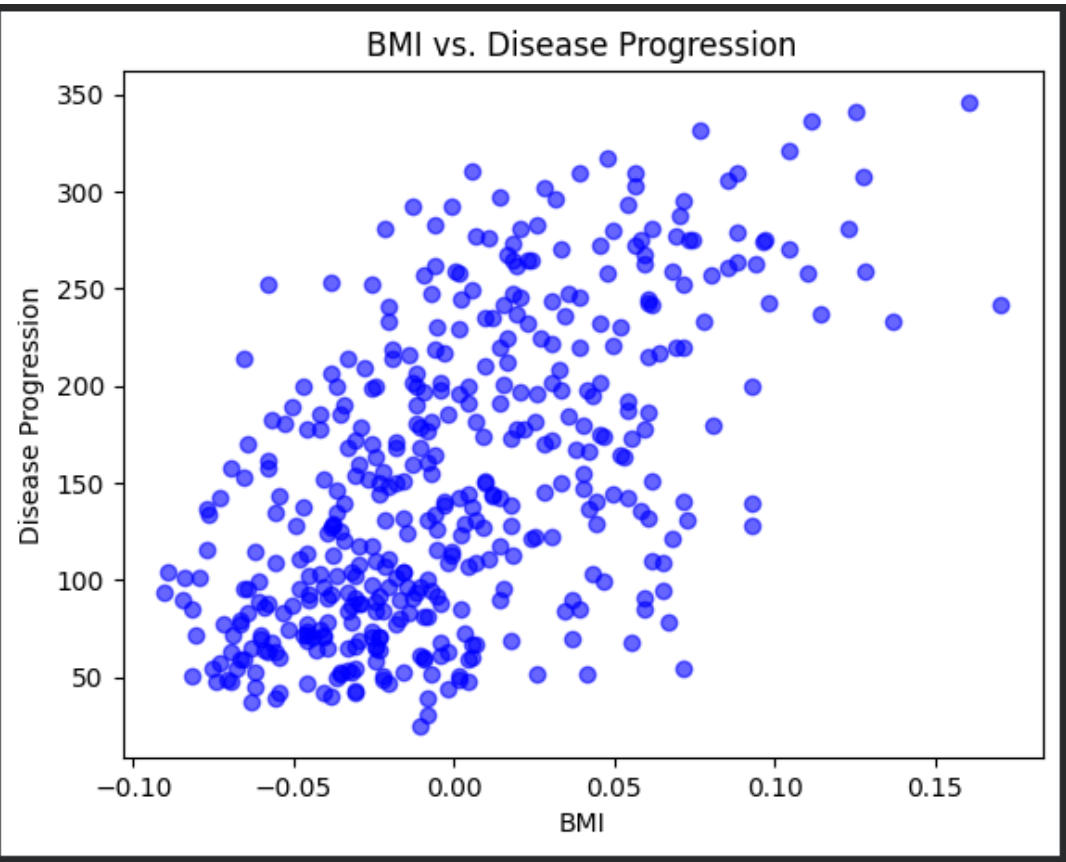
[ ]

```
import pandas as pd
import numpy as np
from sklearn.datasets import load_diabetes
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import seaborn as sns
```

```
diabetes = load_diabetes()
X = diabetes.data
y = diabetes.target
feature_names = diabetes.feature_names
```

```
X_bmi = X[:, np.newaxis, 2]
```

```
plt.scatter(X_bmi, y, color="blue", alpha=0.6)
plt.xlabel("BMI")
plt.ylabel("Disease Progression")
plt.title("BMI vs. Disease Progression")
plt.show()
```



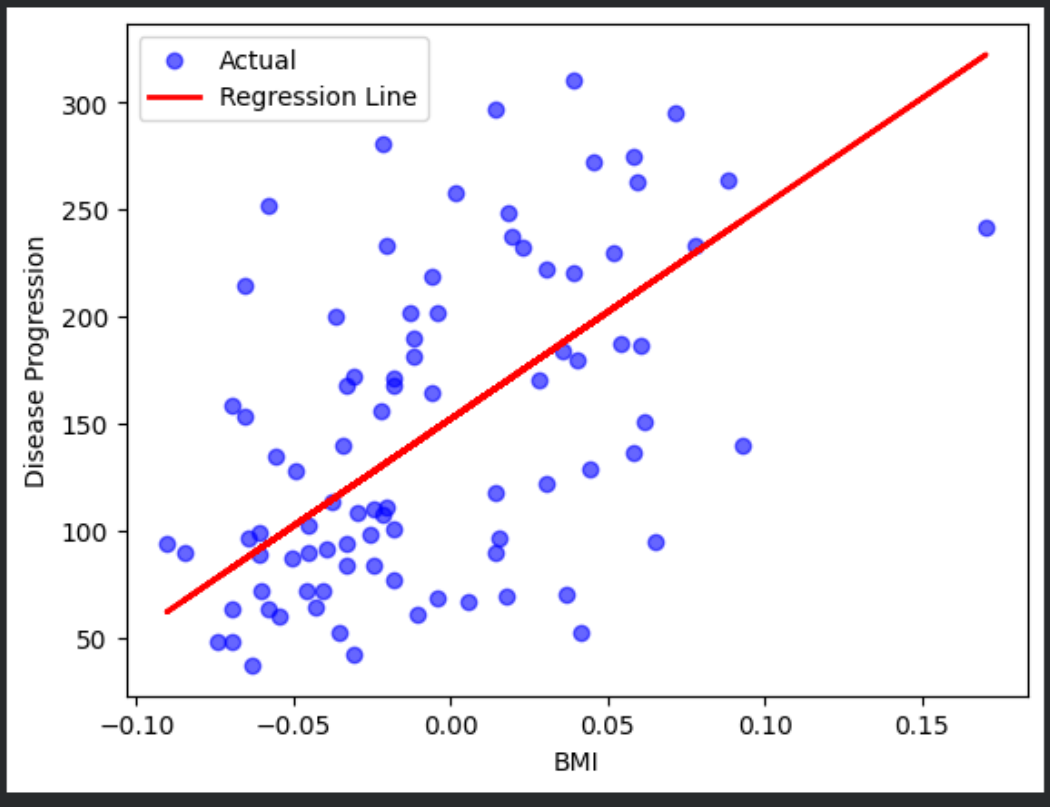
```
[ ]
bmi_corr = np.corrcoef(X_bmi.flatten(), y)[0, 1]
print("Correlation between BMI and disease progression:", bmi_corr)
```

```
Correlation between BMI and disease progression: 0.5864501344746885
```

```
[ ]
X_train, X_test, y_train, y_test = train_test_split(X_bmi, y, test_size=0.2, random_state=42)
```

```
[ ]
model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

```
plt.scatter(X_test, y_test, color="blue", alpha=0.6, label="Actual")
plt.plot(X_test, y_pred, color="red", linewidth=2, label="Regression Line")
plt.xlabel("BMI")
plt.ylabel("Disease Progression")
plt.legend()
plt.show()
```



```
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("MSE:", mse)
print("R² Score:", r2)
```

```
MSE: 4061.8259284949268
R² Score: 0.23335039815872138
```

## Task 2: Multivariate Linear Regression

**1. Load dataset:** Use the same Diabetes dataset, but include all 10 features to predict disease progression.

**2. Perform EDA:**

o Generate a correlation heatmap between features and the target.

o Create pair plots for selected features vs. target.

**3. Implement Multivariate Linear Regression:**

o Use all independent variables to predict the target.

o Fit and predict using the model.

**4. Compare actual vs. predicted values using:**

o Scatter plot (predicted vs. actual).

o Residual plot.

**5. Evaluate model performance with:**

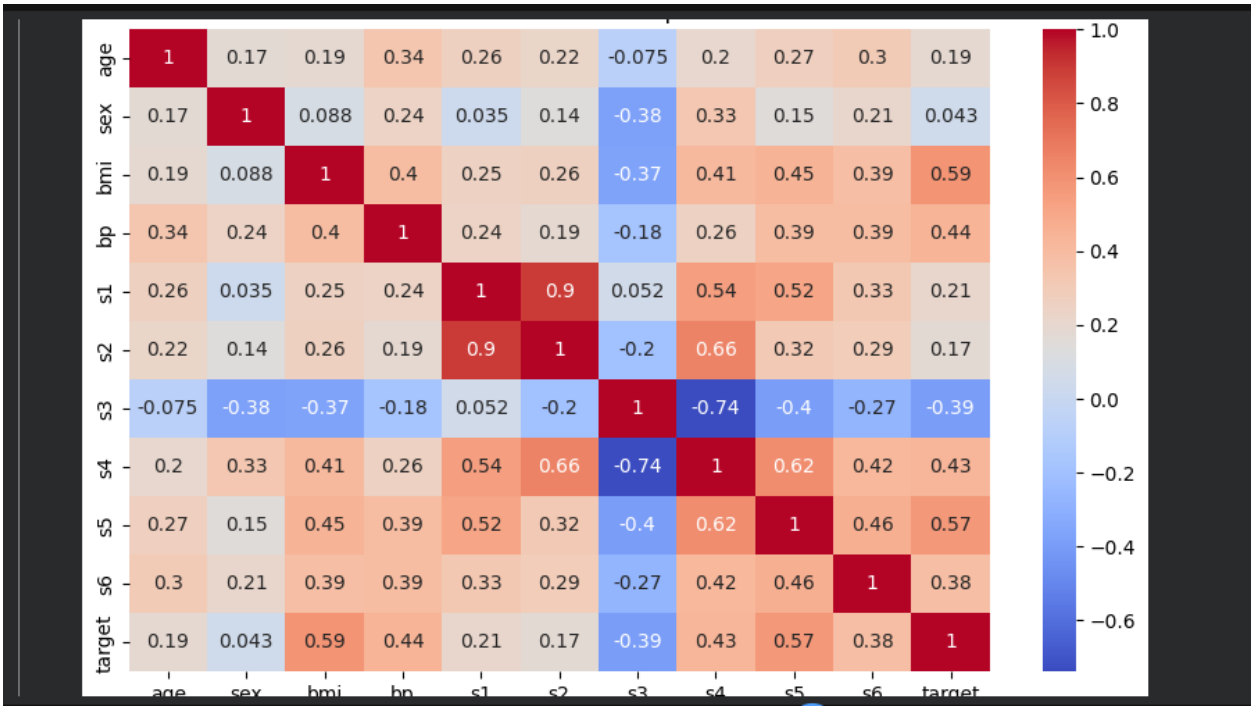
o MSE

o RMSE

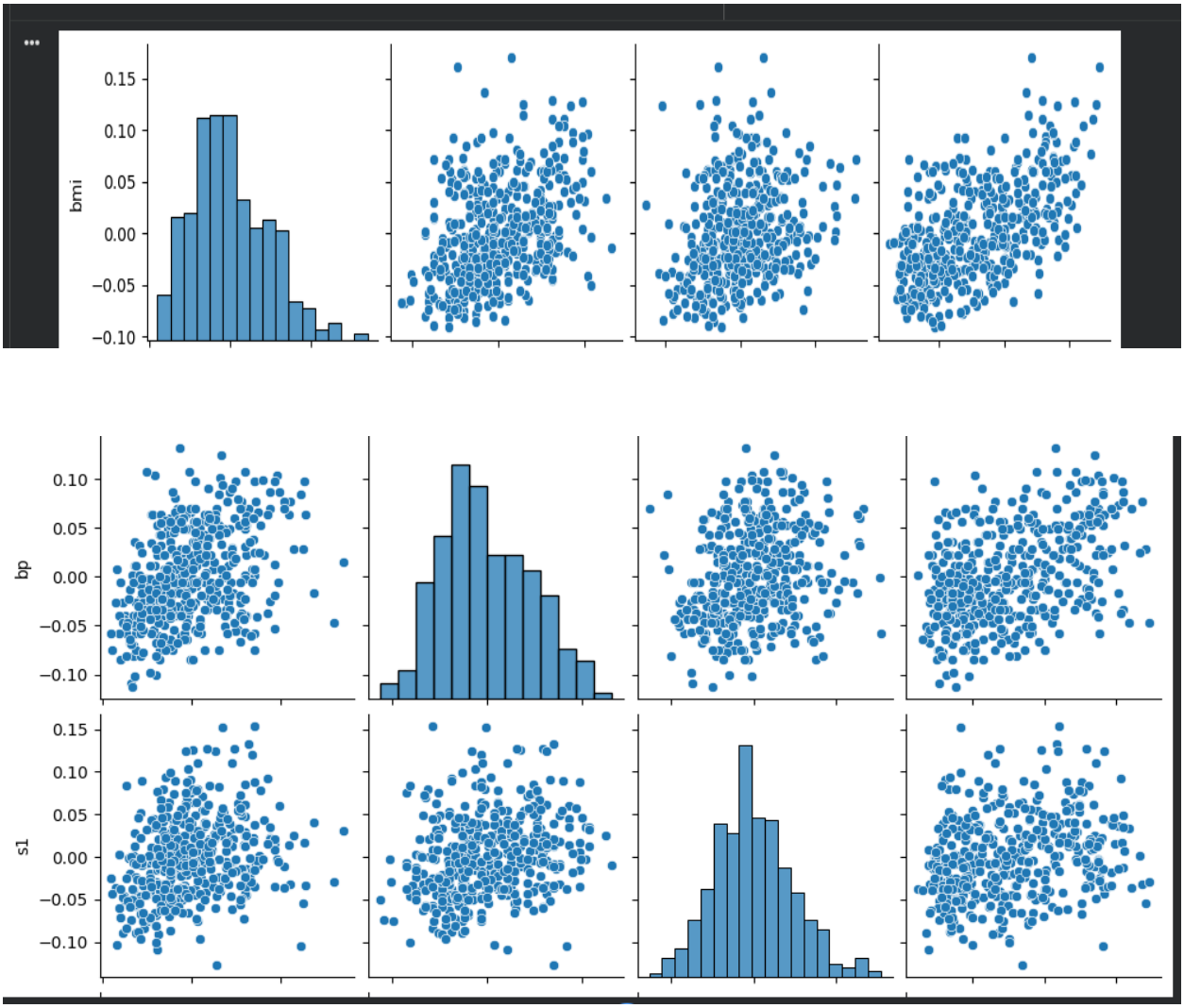
o $R^2$ score

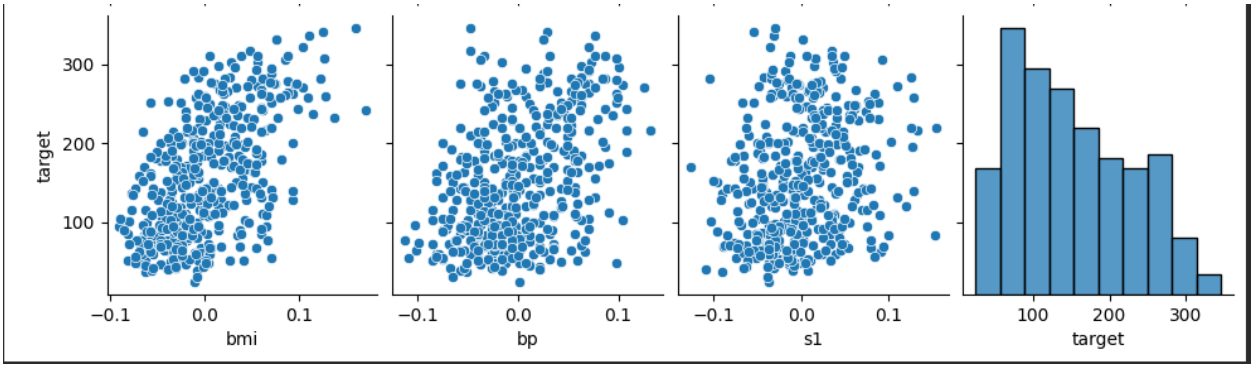
Which independent variable contributes the most to predicting disease progression?

```python
# --- Use all features
X = diabetes.data
y = diabetes.target

# --- EDA: Correlation Heatmap
df = pd.DataFrame(X, columns=feature_names)
df["target"] = y

plt.figure(figsize=(10, 6))
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.show()
```

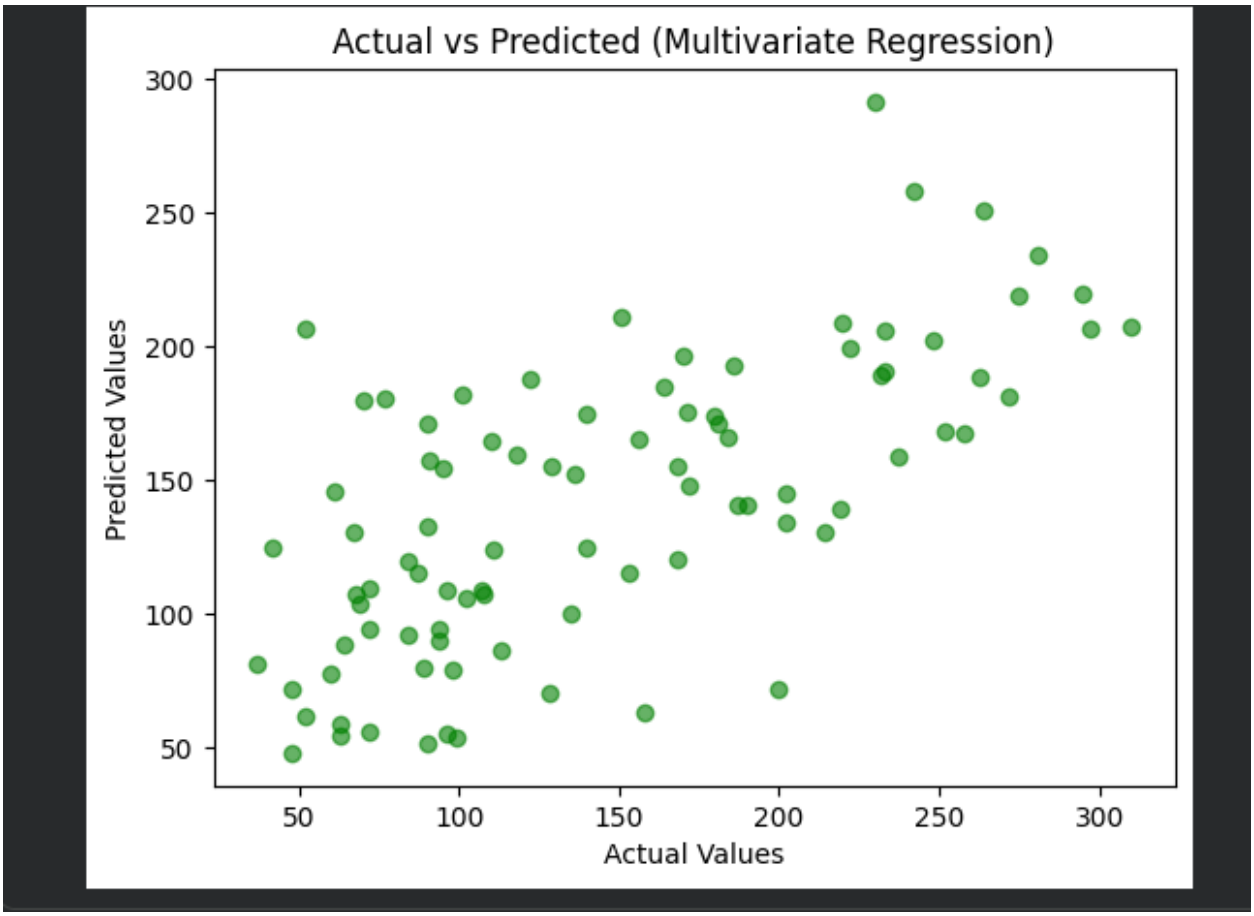


```python
# --- Pairplot for selected features
sns.pairplot(df[["bmi", "bp", "s1", "target"]])
plt.show()

# --- Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# --- Fit Multivariate Regression
multi_model = LinearRegression()
multi_model.fit(X_train, y_train)
y_pred_multi = multi_model.predict(X_test)
```

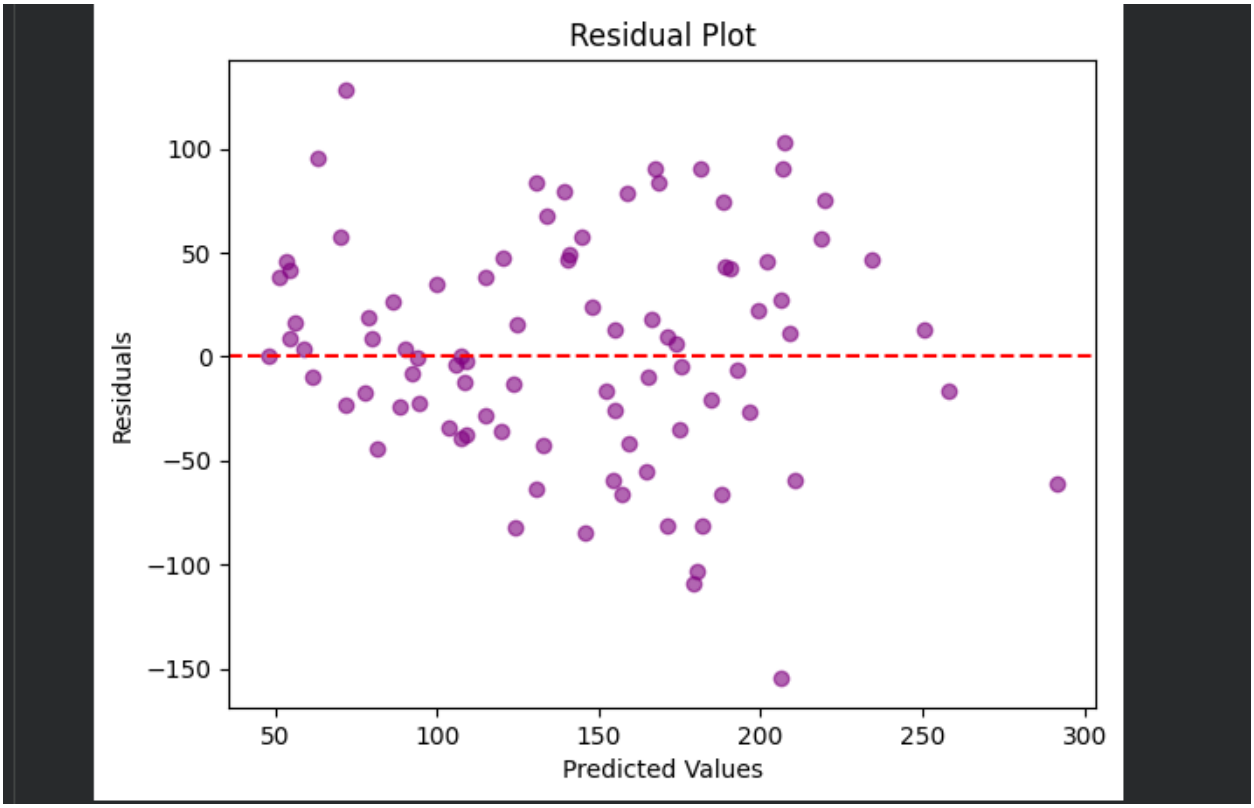





```python
# --- Scatter: Actual vs Predicted
plt.scatter(y_test, y_pred_multi, color="green", alpha=0.6)
plt.xlabel("Actual Values")
plt.ylabel("Predicted Values")
plt.title("Actual vs Predicted (Multivariate Regression)")
plt.show()
```



```python
# --- Residual Plot
residuals = y_test - y_pred_multi
plt.scatter(y_pred_multi, residuals, color="purple", alpha=0.6)
plt.axhline(y=0, color="red", linestyle="--")
plt.xlabel("Predicted Values")
plt.ylabel("Residuals")
plt.title("Residual Plot")
plt.show()
```



```python
# --- Evaluation
mse_multi = mean_squared_error(y_test, y_pred_multi)
rmse_multi = np.sqrt(mse_multi)
r2_multi = r2_score(y_test, y_pred_multi)

print("Multivariate MSE:", mse_multi)
print("Multivariate RMSE:", rmse_multi)
print("Multivariate R² Score:", r2_multi)
```

```
Multivariate MSE: 2900.193628493482
Multivariate RMSE: 53.85344583676593
Multivariate R² Score: 0.4526027629719195
```

```python
# --- Find most contributing feature (by coefficient)
coefficients = pd.DataFrame({"Feature": feature_names, "Coefficient": multi_model.coef_})
coefficients = coefficients.sort_values(by="Coefficient", ascending=False)
print(coefficients)
```

| | Feature | Coefficient |
|---|---|---|
| 8 | s5 | 736.198859 |
| 2 | bmi | 542.428759 |
| 5 | s2 | 518.062277 |
| 3 | bp | 347.703844 |
| 7 | s4 | 275.317902 |
| 6 | s3 | 163.419983 |
| 9 | s6 | 48.670657 |
| 0 | age | 37.904021 |
| 1 | sex | -241.964362 |
| 4 | s1 | -931.488846 |

# Task 3: Experimentation

1. Compare performance of simple vs. multivariate regression in terms of evaluation metrics.

```
[ ]
```

```
print("\n--- Simple Linear Regression ---")
print("MSE:", mse)
print("R² Score:", r2)

print("\n--- Multivariate Linear Regression ---")
print("MSE:", mse_multi)
print("RMSE:", rmse_multi)
print("R² Score:", r2_multi)
```

```
...
--- Simple Linear Regression ---
MSE: 4061.8259284949268
R² Score: 0.23335039815872138

--- Multivariate Linear Regression ---
MSE: 2900.193628493482
RMSE: 53.85344583676593
R² Score: 0.4526027629719195
```