

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
data = pd.read_csv('p_iris.csv')
```


```
x = data['SepalLengthCm'].values
y = data['SepalWidthCm'].values
```

```
# 1. Analytical Method
# Calculate means
N = len(x)
mean_x = np.mean(x)
mean_y = np.mean(y)

# Calculate coefficients
beta_1 = (N * np.sum(x * y) - np.sum(x) * np.sum(y)) / (N * np.sum(x ** 2) - np.sum(x) ** 2)
beta_0 = mean_y - beta_1 * mean_x

print(f"Analytical Method: Intercept ( $\beta_0$ ) = {beta_0}, Slope ( $\beta_1$ ) = {beta_1}")

# Make predictions using the derived coefficients
y_pred_analytical = beta_0 + beta_1 * x
```

 Analytical Method: Intercept ( $\beta_0$ ) = 0.47599332256678795, Slope ( $\beta_1$ ) = -0.08590235069574728

```
# 2. Machine Learning Method (Gradient Descent)
# Initialize parameters
alpha = 0.01 # Learning rate
iterations = 1000
m = len(x) # Number of observations
```


```
# Initialize weights
theta_0 = 0 # Intercept
theta_1 = 0 # Slope
```

```
# Gradient Descent
for _ in range(iterations):
    # Predictions
    y_pred_ml = theta_0 + theta_1 * x
    # Calculate gradients
    error = y_pred_ml - y
    gradient_0 = (1/m) * np.sum(error)
    gradient_1 = (1/m) * np.sum(error * x)
```

```
# Update weights
theta_0 -= alpha * gradient_0
theta_1 -= alpha * gradient_1
```

```
print(f"Machine Learning Method: Intercept ( $\theta_0$ ) = {theta_0}, Slope ( $\theta_1$ ) = {theta_1}")
```

```
# Make predictions using the ML method
y_pred_ml = theta_0 + theta_1 * x
```

 Machine Learning Method: Intercept ( $\theta_0$ ) = 0.40406666573537464, Slope ( $\theta_1$ ) = 0.07447297472687159

```
# Plotting the results
plt.figure(figsize=(12, 6))
plt.scatter(x, y, color='blue', label='Data points', alpha=0.5)
plt.plot(x, y_pred_analytical, color='red', label='Analytical Prediction', linewidth=2)
plt.plot(x, y_pred_ml, color='green', label='ML Prediction', linewidth=2)
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Sepal Width (cm)')
plt.title('Simple Linear Regression')
plt.legend()
plt.show()
```

