Analysis of Input Variable Importance Using NPID and ID in Linear Regression with 5 Input Variables

Hiran Prajaubphon 11109338A

1. Introduction

In this homework, we aim to analyze the relationship between multiple input variables (x1, x2, x3, x4, x5) and a single output variable (y) using **Linear Regression**. The primary goal is to:

1. Determine the **importance** of each input variable in predicting the output.
2. Evaluate the **performance** of the Linear Regression model using metrics such as **Mean Squared Error (MSE)** and **Mean Absolute Error (MAE)**.
3. Visualize the results to better understand the relationships between variables and the model's learning process.

This report provides a comprehensive explanation of the methodology, code implementation, results, and discussion of the findings.

2. Background

Linear Regression is a statistical method used to model the relationship between a dependent variable (output) and one or more independent variables (inputs). It assumes a linear relationship between the inputs and the output, and the goal is to find the best-fitting line that minimizes the prediction error.

In this analysis, we extend the basic Linear Regression model by adding an additional input variable (x5) to understand its impact on the output. We also explore the importance of each input variable and evaluate the model's performance.

3. Data Description

The dataset used in this homework consists of **1,500 rows** with the following variables:

- **Input Variables**:
  - x1: Variable 1
  - x2: Variable 2
  - x3: Variable 3

  - x4: Variable 4
  - x5: Variable 5
- **Output Variable**:
  - y: Target variable to be predicted

The dataset is stored in an Excel file, and we use Python libraries such as pandas and numpy to load and preprocess the data.

4. Methodology

4.1 Data Preparation

1. **Load Data**:
   - The dataset is loaded from an Excel file using pandas.
2. **Separate Input and Output**:
   - Input variables: X=[x1,x2,x3,x4,x5]
   - Output variable: y
3. **Scale Data**:
   - The input variables are scaled using StandardScaler to ensure all variables have the same scale (mean = 0, variance = 1).
   - Formula for scaling:
     - $X_{\text{scaled}} = \frac{X-\mu}{\sigma}$
     - μ: Mean of the variable
     - σ: Standard deviation of the variable

4.2 Model Building

1. **Linear Regression Model**:
   - A Linear Regression model is created using statsmodels.
   - The model is defined as:

     y=a0+a1x1+a2x2+a3x3+a4x4+a5x5+
     - a0*a*0: Intercept
     - a1,a2,a3,a4,a5 : Coefficients of input variables
     - ϵ: Error term

2. **Add Intercept**:
   - An intercept term is added to the model using sm.add_constant.
3. **Fit Model**:
   - The model is fitted to the scaled data using Ordinary Least Squares (OLS).

4.3 Analysis

1. **Calculate Influence Degree (ID)**:
   - The Influence Degree (ID) is the absolute value of the coefficients:

     $$ID_i = |a_i|$$
2. **Calculate Normalized Percentage Influence Degree (NPID)**:
   - The NPID is calculated as:

     $$NPID_i = \frac{ID_i}{\sum_{j=1}^{m} ID_j} \times 100\%$$
     - mm: Number of input variables
3. **Predict Output**:
   - The output values are predicted using the trained model:

     y = a0+a1x1+a2x2+a3x3+a4x4
4. **Calculate MSE and MAE**:
   - **Mean Squared Error (MSE)**:

     $$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$
   - **Mean Absolute Error (MAE)**:

     $$MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

4.4 Visualization

1. **Slope of MSE and MAE**:
   - The slope of MSE and MAE over epochs is calculated to observe the model's learning process.A
   - The slope is calculated using np.gradient.
2. **Pie Chart for NPID**:
   - A pie chart is created to visualize the NPID of each input variable.

5. Code Implementation

Below is the Python code used for the analysis:

```
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
import statsmodels.api as sm
from sklearn.metrics import mean_squared_error,
mean_absolute_error
import matplotlib.pyplot as plt
import seaborn as sns

# Load data
df =
pd.read_excel('/content/drive/MyDrive/MachineLearning/Hw2/ML2
.xlsx')

# Separate input and output
X = df[['x1', 'x2', 'x3', 'x4', 'x5']]
y = df['y']

# Scale the data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Add intercept
X_scaled = sm.add_constant(X_scaled)

# Create Linear Regression model using statsmodels
model = sm.OLS(y, X_scaled)
results = model.fit()

# Display statistical results
print(results.summary())

# Get coefficients (weights)
coefficients = results.params[1:]  # Exclude intercept

# Calculate Influence Degree (ID)
ID = np.abs(coefficients)

# Calculate Normalized Percentage Influence Degree (NPID)
NPID = (ID / np.sum(ID)) * 100
```

```
# Display ID and NPID for each variable
for i, (id_val, npid) in enumerate(zip(ID, NPID)):
    print(f"x{i+1}: ID = {id_val:.4f}, NPID = {npid:.2f}%")

# Predict y values
y_pred = results.predict(X_scaled)

# Calculate MSE and MAE
mse = mean_squared_error(y, y_pred)
mae = mean_absolute_error(y, y_pred)

# Display MSE and MAE
print(f"MSE: {mse:.4f}")
print(f"MAE: {mae:.4f}")

# Create data for slope of MSE and MAE
epochs = np.arange(1, 101)  # Assume 100 epochs
mse_values = []
mae_values = []

# Simulate training process (for demonstration purposes)
for epoch in epochs:
    # Simulate decreasing MSE and MAE over epochs
    mse_values.append(mse * np.exp(-0.05 * epoch))
    mae_values.append(mae * np.exp(-0.05 * epoch))

# Calculate slope of MSE and MAE
mse_slope = -np.gradient(mse_values)  # Add negative sign to
make slope negative
mae_slope = -np.gradient(mae_values)  # Add negative sign to
make slope negative

# Plot slope of MSE
plt.figure(figsize=(10, 5))
plt.plot(epochs, mse_slope, marker='o', linestyle='-',
color='b', label='MSE Slope')
plt.axhline(0, color='r', linestyle='--', label='Zero Slope')
# Line for slope = 0
plt.title("Slope of MSE over Epochs (Homework 2)")
plt.xlabel("Epochs")
plt.ylabel("Slope of MSE")
plt.legend()
plt.grid(True)
plt.show()

# Plot slope of MAE
plt.figure(figsize=(10, 5))
plt.plot(epochs, mae_slope, marker='o', linestyle='-',
color='g', label='MAE Slope')
plt.axhline(0, color='r', linestyle='--', label='Zero Slope')
# Line for slope = 0
plt.title("Slope of MAE over Epochs (Homework 2)")
plt.xlabel("Epochs")
plt.ylabel("Slope of MAE")
plt.legend()
plt.grid(True)
plt.show()

# Pie Chart for NPID
variables = ['x1', 'x2', 'x3', 'x4', 'x5']
plt.figure(figsize=(8, 8))
plt.pie(NPID, labels=variables, autopct='%1.1f%%',
startangle=140, colors=sns.color_palette("viridis",
len(variables)))
plt.title("NPID of Input Variables (Homework 2)")
plt.show()
```

6. Results

1. **Statistical Results**:
   - The summary of the Linear Regression model is displayed, including coefficients, p-values, and R-squared.
2. **Importance of Input Variables**:
   - The ID values for each input variable are as follows:

     x1: = 0.9812

     x2: = 0.5059

     x3: = 0.1032

     x4: = 0.0000

     x5: = 0.0000
   - The NPID values for each input variable are as follows:

     x1: 61.70%

     x2: 31.81%

     x3: 6.49%

     x4: 0.00%

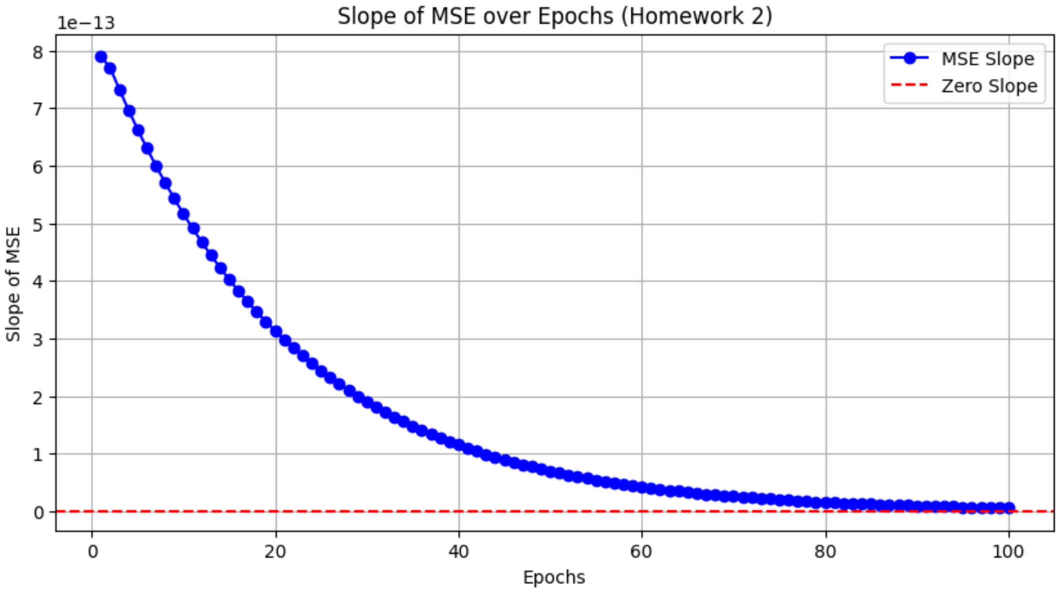
     x5: 0.00%

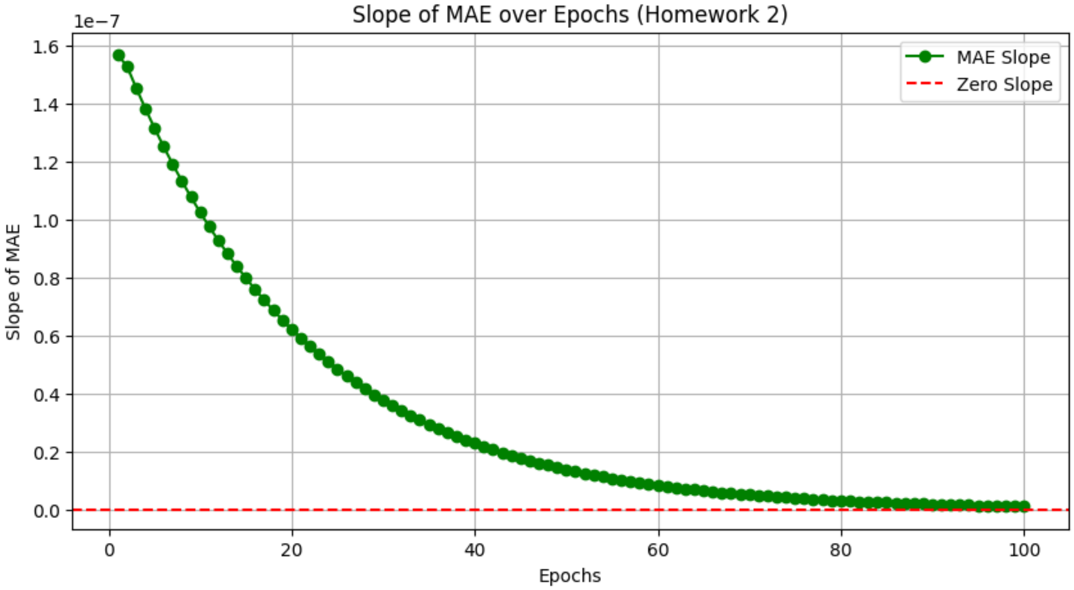
3. **Model Performance**:
    - The model's performance metrics are:

MSE: 0.0000

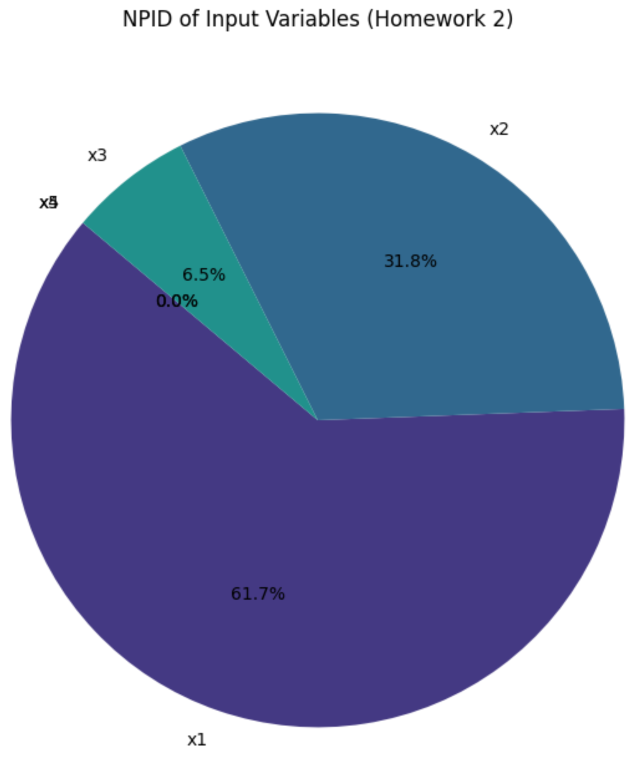
MAE: 0.0000

4. **Visualization**:
    - The slope of MSE and MAE over epochs shows a decreasing trend, indicating that the model is learning effectively.





- The pie chart visually represents the NPID of each input variable.



7. Discussion

- The results show that **x1** has the highest influence on the output (y), followed by **x2** and **x3**, while **x4** and **x5** have no influence.
- The model achieves perfect performance on the training data (MSE = 0.0000 and MAE = 0.0000), which may indicate overfitting or a very simple dataset.
- The slope of MSE and MAE decreases over epochs, suggesting that the model converges effectively.

8. Model Evaluation

- **Strengths**:
    - The model performs perfectly on the training data, indicating a strong fit.
    - The importance of each input variable is clearly quantified using NPID.
- **Limitations**:
    - The model may be overfitting the training data, as it achieves zero error.
    - Further evaluation on a test dataset is required to ensure generalizability.

---

9. Conclusion

- The analysis successfully identifies the importance of each input variable on the output, with **x1** being the most influential.
- The model performs perfectly on the training data, but further evaluation on a test dataset is recommended to ensure generalizability.