Assignment-I: Linear Regression

Course: Machine Learning Lab- PCC-AIML 592

Topic: Linear Regression with NumPy and scikit-learn

Goal: To implement and understand linear regression through various approaches and connect theory with practice.

Expected Learning Outcomes

By completing this assignment, students should be able to:

- Explain the principle of linear regression.
- Understand the mathematical foundation behind linear models.
- Implement regression using different techniques.
- Compare analytical and iterative solutions.
- Visualize model predictions and performance.

Sub-Task 1: Understand and Explain the Theory

Write a concise theoretical summary covering:

- What is linear regression?
- How is the relationship between dependent and independent variables modeled?
- Derivation and meaning of:
  - Hypothesis function: h_θ($\mathbf{x}$) = $\theta_0 + \theta_1 x_1 + \ldots + \theta_n x_n$
  - Cost function (Mean Squared Error)
  - The Normal Equation for directly computing model parameters.
  - Gradient Descent algorithm for iterative parameter updates.
- A brief comparison of the Normal Equation and Gradient Descent.

Sub-Task 2: Data Generation and Visualization

Objective: Simulate a simple linear dataset and visualize it.

Pseudocode:

- Set random seed to ensure reproducibility
- Generate 100 random values (x) in a given range
- Create y values using a linear equation: y = slope * x + intercept + noise
- Plot the data points on a scatter plot

Expected Outcome: A cloud of points roughly aligned along a straight line.

Sub-Task 3: Solve Using the Normal Equation

Pseudocode:

```
- Add a bias term (column of ones) to the input data
- Compute theta using the Normal Equation:
  theta = inverse(transpose(X) * X) * transpose(X) * y
- Use theta to predict y for new inputs
- Plot the regression line over the original data
```

Sub-Task 4: Use scikit-learn’s LinearRegression Model

Pseudocode:

```
- Initialize a LinearRegression model
- Fit the model on the training data (x, y)
- Extract intercept and slope
- Use the model to predict values for new x inputs
```

Sub-Task 5: Use Pseudo-Inverse and SVD (SVD-based Least Squares)

Pseudocode:

```
- Use lstsq function to compute theta and residuals
- Use pinv function to compute theta as: pinv(X) * y
- Compare theta values from all methods
```

Sub-Task 6: Implement Batch Gradient Descent from Scratch

Pseudocode:

```
- Initialize theta with small random values
- Set learning rate and number of iterations
- Repeat for the given number of epochs:
    • Calculate gradients: gradient = (2/m) * transpose(X) * (X * theta - y)
    • Update theta using gradient: theta = theta - learning_rate * gradient
- Return final theta
```

Sub-Task 7: Use scikit-learn’s SGDRegressor

Pseudocode:

```
- Initialize an SGDRegressor model with:
    • learning rate
    • max number of iterations
    • no regularization (penalty=None)
- Fit the model on training data
- Extract the learned coefficients and intercept
- Predict values for new inputs
```

Report Structure

Students must prepare a report with the following:
- Title of the assignment
- Objective statement
- Theory summary (Sub-Task 1)
- For each Sub-Task:
  - Description of the task
  - Python code (well-commented for readability)
  - Explanation of outputs (including charts)
  - Observations and insights (e.g. how does model performance correlate with the size of the data?)
- Final summary comparing all approaches

Submission Instructions

Submit:
- A well-documented Jupyter Notebook (.ipynb)
- A lab-report

Use meaningful section headings. Ensure all figures are labeled and legible.