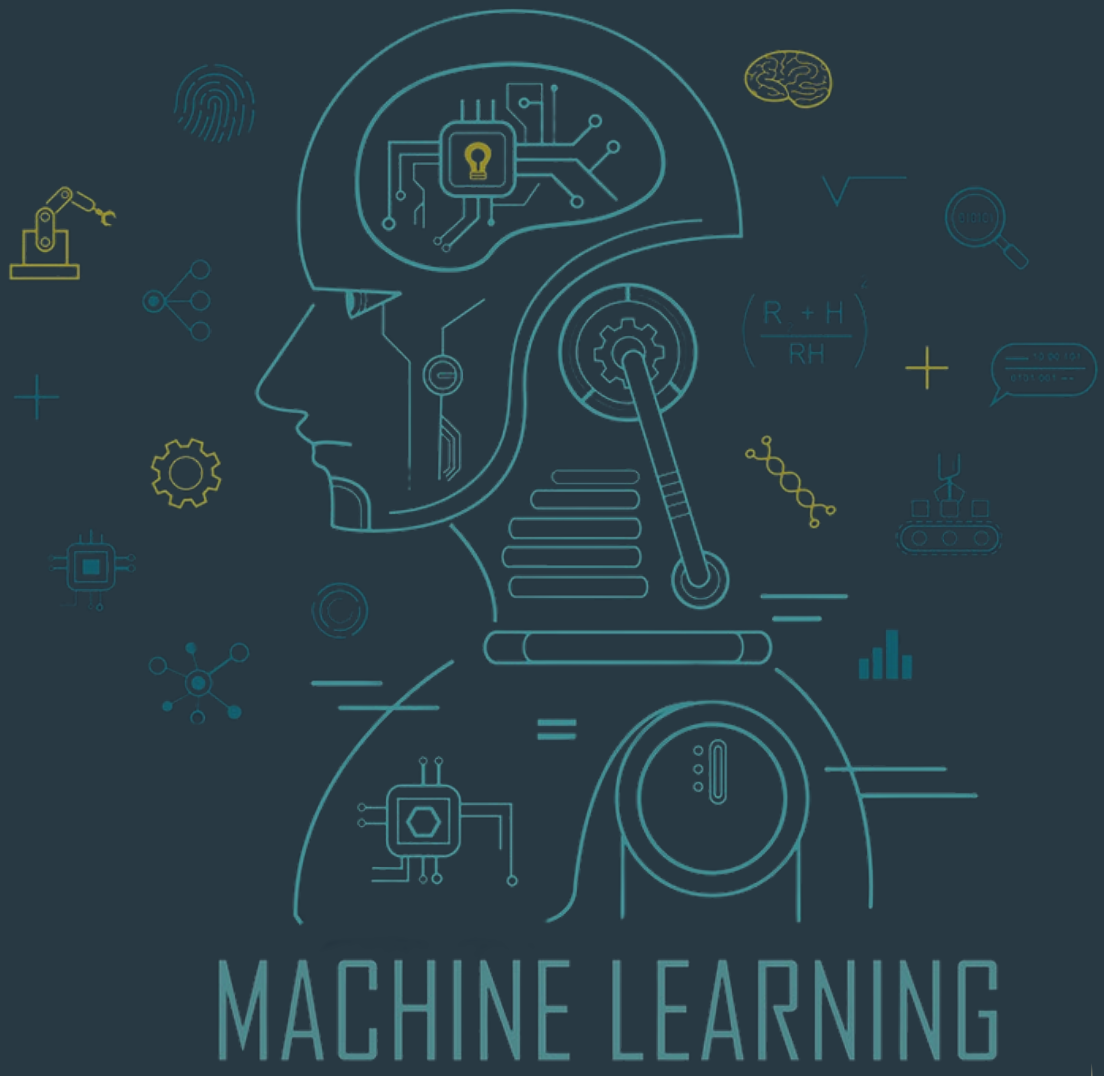
01 . 11 . 2024

Project Report



Submitted by: **EFE EROL**
No: **200101066**

1. Introduction

The primary goal of this project is to implement a simple linear regression model using gradient descent optimization. The model aims to predict house prices based on a single feature derived from a dataset. This project will process training and test data, standardize the features, and visualize the performance of the model through various plots. Ultimately, we aim to determine the optimal weights for our model while minimizing the cost function.

2. Data Processing

2.1 Reading Data

The dataset is read from text files (train.txt and test.txt), where each line contains two values: a feature (e.g., the size of the house) and the corresponding target value (e.g., the price).

- Function: read_data(file_name)
  - The function reads the data line by line, splits each line into values, and converts them to floating-point numbers.
  - The features are stored in a NumPy array X, which is reshaped into a column vector, while the target values are stored in y.
  - A bias term (a column of ones) is added to X to account for the intercept in the linear regression equation.

2.2 Standardization of Features

Before training the model, it's essential to standardize the features to improve convergence during optimization. The standardization process involves:

- Calculating the mean and standard deviation of the training data features (excluding the bias term).
- Standardizing the training and test data using the mean and standard deviation obtained from the training set.

This normalization ensures that the features contribute equally to the model, which aids in faster convergence of the gradient descent algorithm.

3. Gradient Descent Implementation

3.1 Training the Model

The train function implements the gradient descent algorithm to optimize the weights w for the linear regression model:

- Initialization: The weights are initialized to zeros, and lists are created to store the cost and root mean squared error (RMSE) values for both training and test datasets.
- Iterative Optimization: For a specified number of epochs:
  - The RMSE is calculated for both training and test datasets using the current weights.
  - The gradient of the cost function with respect to the weights is computed.
  - The weights are updated by moving in the opposite direction of the gradient.
  - The cost (using the updated weights) is computed and stored for analysis.

3.2 Cost Function and RMSE

- Compute Cost: The cost function quantifies the error in predictions, calculated as the mean of squared differences between predicted and actual values.
- Compute RMSE: The root mean squared error is a metric that indicates the average error of the model's predictions, offering a tangible measure of model performance.

4. Visualization of Results

4.1 Cost Function Over Epochs

A plot of the cost function values over the epochs is generated to visualize convergence:

- As epochs increase, the cost should decrease, indicating that the model is learning and the predictions are improving.
- The plotted curve provides insight into how effectively the gradient descent is minimizing the cost function.

4.2 RMSE Over Epochs

Separate plots for training and test RMSE are created:

- This visualization helps in understanding how the model's performance improves over time.
- The comparison between training and test RMSE can reveal potential overfitting or underfitting.

4.3 Regression Line Plot

A final plot illustrates the training and test data points along with the regression line:

- The training data is represented by blue circles, while the test data is shown with green crosses.
- The regression line (in red) indicates the model's predictions based on the learned weights.
- This visualization confirms the model's ability to generalize from the training data to the test data.

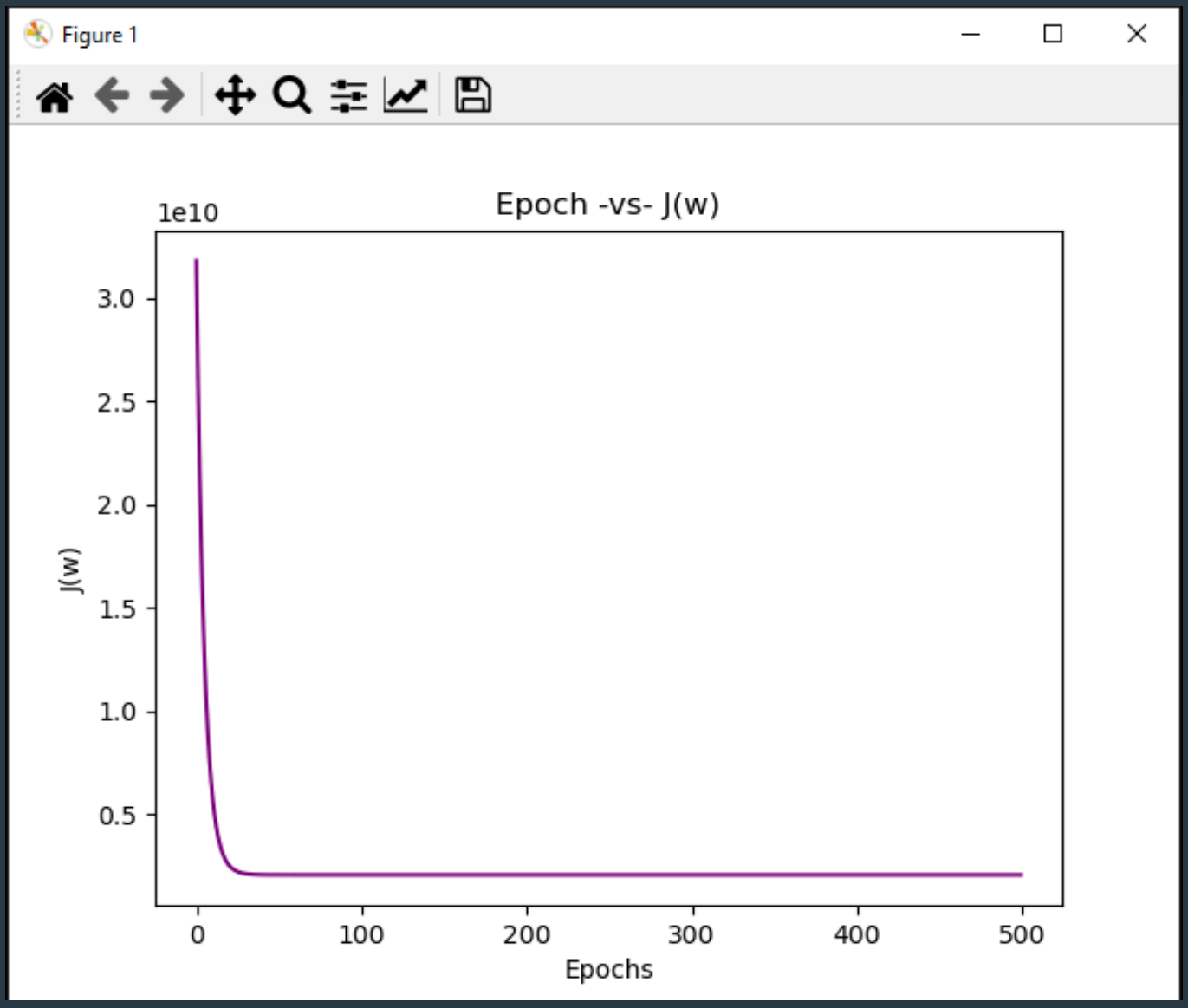
5. Results

The final weights obtained from the gradient descent optimization, as well as the weights computed using the pseudo-inverse method (np.linalg.pinv), are compared to validate the results. The RMSE values for both training and test datasets are also reported, offering an understanding of the model's predictive performance.

The results indicate that the gradient descent algorithm successfully optimizes the model parameters, leading to a lower cost function and RMSE over time. The plots provide a clear visual representation of how the model's performance evolves during the training process.

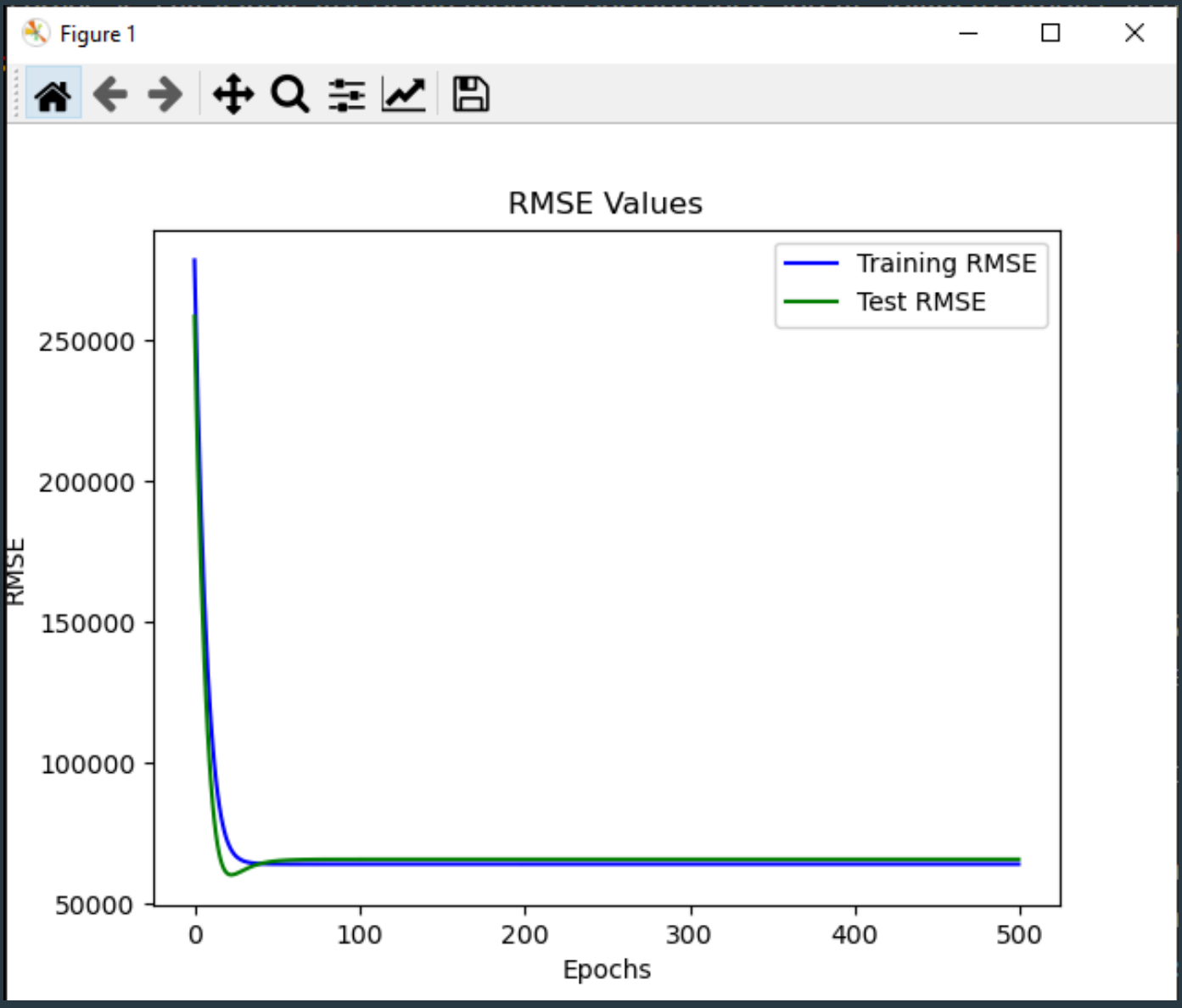
```
[My solution] ---> w = [254450.          93308.9201061]
[linalg.pinv solution] ---> w = [254450.          93308.9201061]
[RMSE] Training Data: 64083.51370385863
[RMSE] Test Data: 65773.19142920323
```

6. Change of J(w) according to the epoch



As the number of epochs increases, we observe a downward trend in the cost function, demonstrating that the algorithm is effectively minimizing the error. This trend signifies that the model is learning and improving its predictions over time. Ideally, we expect the cost to decrease and stabilize, indicating that the algorithm is approaching the optimal solution. Any fluctuations may suggest that the learning rate could be too high, while a plateau indicates convergence to a local minimum. Overall, this graph visually confirms the effectiveness of the gradient descent optimization in training the linear regression model.

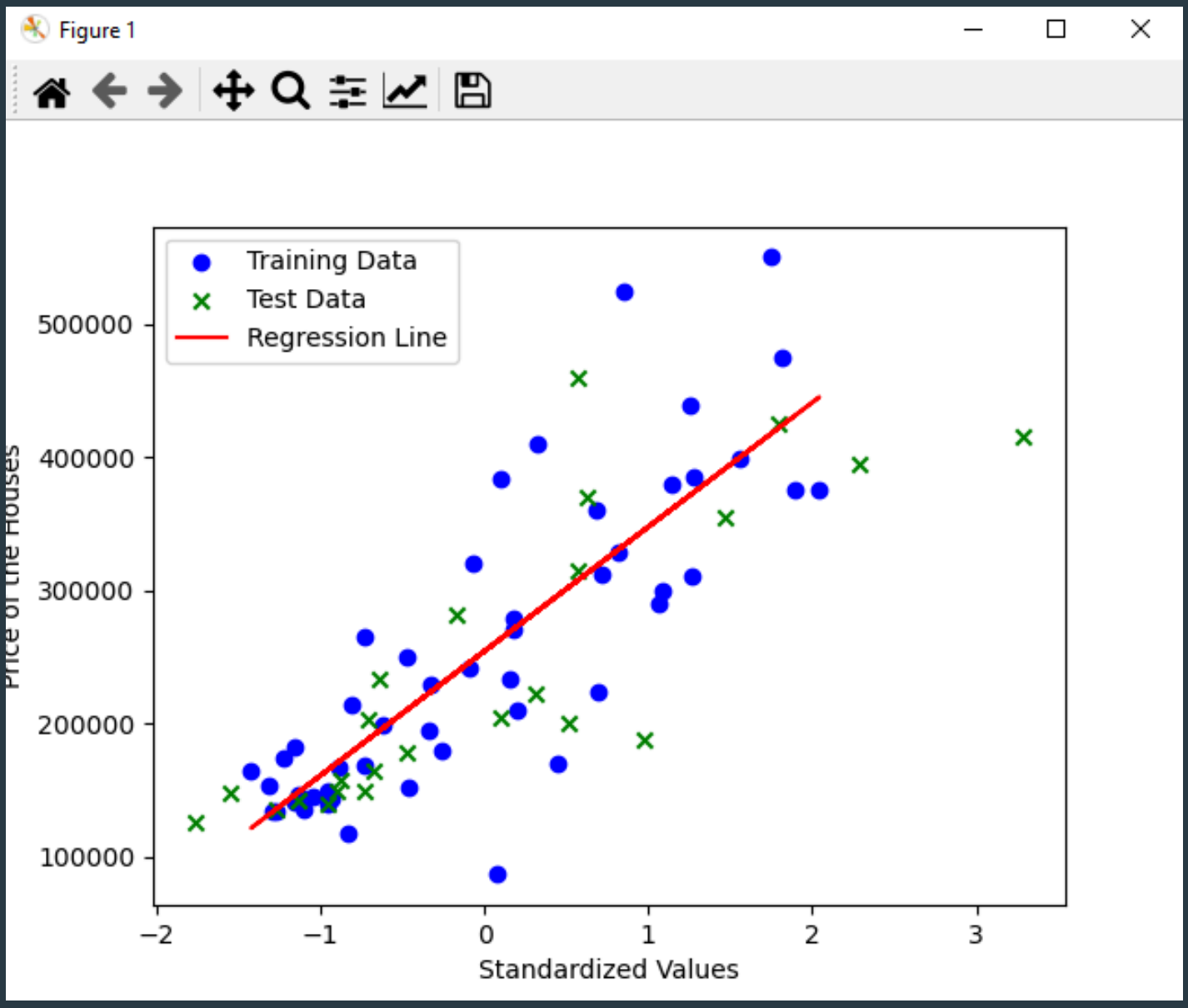
7. Change of RMSE values according to the epoch



The RMSE Values graph shows the Root Mean Squared Error (RMSE) for training and test datasets over the training epochs. The blue line indicates the training RMSE, while the green line represents the test RMSE. A decreasing trend in both lines suggests that the model is learning effectively and improving its predictions.

The convergence of training and test RMSE values indicates good generalization, meaning the model performs well on unseen data. If the training RMSE continues to decrease while the test RMSE levels off or increases, it may indicate overfitting, where the model excels on training data but struggles with new data. Overall, this graph is crucial for assessing the model's performance and predictive accuracy across both datasets.

8. Regression Line



The regression line closely follows the trend of the training data points, indicating that the model has successfully captured the underlying relationship between the input features and the target variable. The alignment of the red line with the blue circles suggests that the model is making accurate predictions for the training data.

Moreover, the proximity of the green test data points to the regression line signifies the model's ability to generalize well to new, unseen data. This close alignment indicates that the model not only fits the training data effectively but also performs reliably when applied to the test dataset.

Overall, this graph highlights the model's performance in regression tasks, demonstrating its capability to learn from training data and make accurate predictions for test data, reflecting a robust understanding of the underlying data patterns.

01 . 11 . 2024

Thank you

for taking the time to read this report.

Submitted by: **EFE EROL**
No: **200101066**