Regression Models Experimentation on the Boston and Wine Datasets

Chaimae Sriti

February 4, 2024

1 Abstract

*I*n this project, we assessed linear and logistic regression models on the Boston Housing and Wine datasets, optimizing hyperparameters like learning rates, and batch sizes. For the linear regression analytical model, the best split was 90/10, with an optimal learning rate at 0.1. In the gradient descent method, the optimal learning rate was 0.1, with a batch size of 64. In the Logistic Regression model, the most generalizable split was 60/40, with optimal parameters of 0.001 for the learning rate and 16 for the batch size, yielding a 99% training accuracy and 100% test accuracy. Overall, analytical linear regression slightly outperformed gradient descent, and analytical methods with Gaussian Transformations using five basis functions produced the lowest Mean Squared Error (MSE) of 17.14. Logistic regression models exhibited consistently high accuracy of 99%.

2 Introduction

Previous work done by Sanyal et al. implemented four different regression models, including simple linear regression, polynomial regression, Ridge regression and Lasso regression, to predict the house price using the Boston Housing dataset [1]. The comparisons were made between models with fixed hyperparameters and split sizes. Here, we aim to implement two models for the Boston Housing and Wine datasets. We sought to explore the impact of hyperparameters, including learning rates, and batch sizes, on model performance. For the analytical linear regression model, an optimal 90/10 train-test split and a learning rate of 0.1 was observed. In the gradient descent method, a learning rate of 0.1, regularization strength of 0.0001, and a batch size of 64 resulted in the best performance, achieving an MSE of 26.15. Conversely, in the Gaussian Basis method, the model achieved the best model performance with an MSE of 8.92 and zero regularization. In our experiments with the Logistic Regression model, we used an 80/20 split and a learning rate of 0.001 for the gradient descent (GD), stochastic gradient descent (SGD), and 16-mini-batch SGD models. These models all reached a train accuracy of 99% and perfect test accuracy. We further adjusted the train-test split size and found a positive correlation between training set size and accuracy. We also assessed the batch size and learning rate, discovering that a batch size of 16 led to quick convergence, while the optimal learning rate was 0.001.

3 Boston Dataset and Linear Regression

3.1 Boston Dataset

The Boston Housing dataset comprises 506 observations, each characterizing a Boston suburb or town with 14 features, encompassing attributes like per capita crime rate and pupil-teacher ratio. The final column represents the median value of owner-occupied homes, our target prediction. The "B" feature was omitted due to ethical concerns. The dataset is free of missing entries and appears clean. Notably, a boxplot analysis identified one outlier in the "CHAS" feature (Appendix, Figure 7.1), which we retained to prevent the samples from becoming linearly dependent, a decision that may warrant verification. In terms of preprocessing for the Boston Housing dataset, we solely applied normalization.

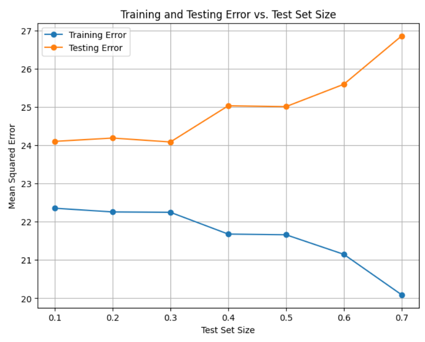
3.2 Results

For the Boston Housing dataset, we employed the following approaches:

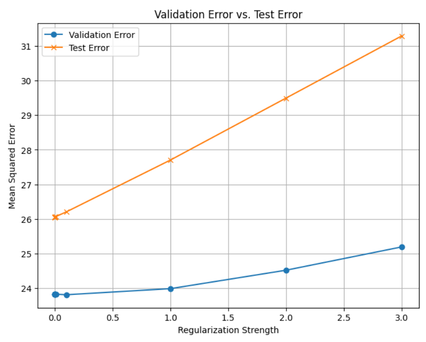
- Analytical Method
- Gradient Descent Method
- Analytical Method using Gaussian Basis Transformation

To ensure unbiased training, we randomly shuffled the dataset for each split. Both methods allowed us to configure bias (non-regularized) and regularization (known as alpha) parameters. The Gradient Descent method additionally allowed us to specify learning rates, batch sizes, epsilon, and maximum iterations.

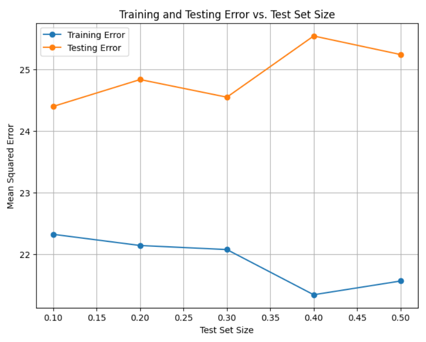
Our analysis aimed to identify the best hyperparameters for all methods. We used 5-fold cross-validation to avoid overfitting.



[figure 1]



[figure 2]



[figure 3]

3.2.1 Choosing Best Parameters for Linear Regression Analytical Method

We conducted 1000 runs with splits (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7). The results, shown in Figure 1, indicate that the model's performance starts to degrade beyond a split of 0.4. Hence, we selected a split of 0.1 to ensure a sufficient amount of data for training.

We used 5-fold cross-validation to determine the optimal regularization strength, as demonstrated in Figure 2. The tested alpha values ranged were (0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 2, 3) . We observed that large alpha values performed poorly, while there was no definitive best value for small alphas. We settled on alpha=0.1.

3.2.2 Choosing Best Parameters for Linear Regression Gradient Descent Method

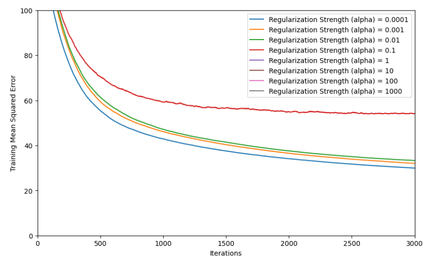
We ran 100 runs for test splits (0.1, 0.2, 0.3, 0.4, 0.5) and averaged the loss (results shown in Figure 3). We chose a split size of 0.1 for the same reasons as in the analytical method.

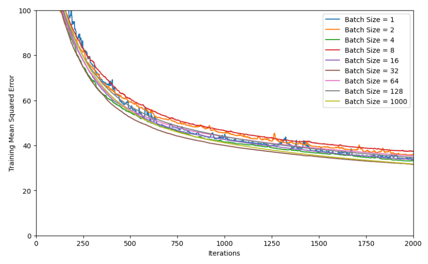
Similarly, we employed 5-fold cross-validation to select the best combination of hyperparameters, including learning rates, regularization strengths, and batch sizes:

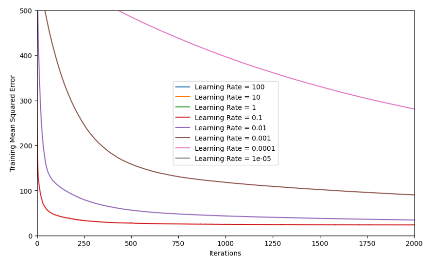
- Learning Rates = [100, 10, 1, 0.1, 0.01, 0.001, 0.0001, 0.00001]
- Regularization Strengths = [0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000]
- Batch Sizes = [1, 2, 4, 8, 16, 32, 64, 128, full-dataset]

Best Learning Rate	0.1
Best Regularization Strength	0.0001
Best Batch Size	64
Best Validation MSE	23.97
Training MSE	22.24
Test MSE	26.15

We also visualized the impact of changing a single parameter while keeping others constant, as shown in the figures below:







3.2.3 Choosing Best Parameters for Transformed Linear Regression Analytical Method

In this approach, we split the data and used five center vectors for transformation. However, the initial results were suboptimal, with a best validation MSE of around 44.70. To improve performance, we experimented with varying numbers of center vectors.

Using 5-fold cross-validation, we explored the optimal combination of regularization strength and the number of center vectors:

:

- Learning Rates = [0, 0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 2, 3]
- Number of Centers = [1 to 30]

Best Learning Rate	0
Best number of center vectors	28
Best Batch Size	64
Best Validation MSE	16.14
Test MSE	8.92

3.2.4 Comparing Everything

We compared the three best models and their respective parameters:

- Analytical Method (alpha=0.1)
- Gradient Descent Method (alpha=0.1, learning rate = 0.0001, batch size = 64)
- Analytical Method using Gaussian Transformation (alpha=0, number of centers = 28)

To ensure a fair comparison, we randomly split the data and ran each model 100 times, averaging the test loss. The results are summarized in Table 3:

Table 1: Models Average Error (100 runs)

Analytical Method	24.36
Gradient Descent Method	24.44
Analytical Method using Gaussian Transformation	17.14

4 Wine Dataset and Multi-Class Logistic Regression

In this section, we will provide a brief overview of the Wine dataset followed by a summary of the results from the various experiments conducted using Multi-Class Logistic Regression (Softmax).

4.1 Wine Dataset

The Wine dataset contains 178 observations of 13 chemicals in three wine types. 11 attributes are continuous and two (Magnesium and Proline) are discrete. Distribution is roughly 33% class 1, 40% class 2, and 27% class 3, showing minimal class imbalance. The dataset is complete without missing values. Due to varied feature magnitudes seen in boxplots (see Appendix 7.2), normalization and outlier removal will be applied during preprocessing. In the correlation analysis, `Falavanoids` shows strong correlations with `Total Phenols` (0.86) and `OD280_OD315_of_diluted_wines` (0.79), suggesting potential multicollinearity. This could inform decisions on feature removal in model experimentation. Further feature details are in `a1-Softmax.ipynb`.

4.2 Results

During preprocessing, we normalized the dataset and removed outliers, but retained the full dataset for six experiments due to negligible outlier impact. We used an 80/20% train-test split, normalized, then applied multiclass logistic regression.

For tasks 1-4, with a learning rate of 0.001, GD, SGD, and 16-minibatch SGD produced similar accuracies(Table 1). Using 5-fold cross-validation on the 80/20% split, all models had nearly identical performance on the Wine Dataset, indicating good generalization.

Wine Dataset (100 Epochs)				
Model	Train Accuracy	Test Accuracy	5-cv Train	5-cv Test
GD/SGD/16Batch	0.9929	1	0.9929	0.9831

Table 2: Accuracy for GD, SGD and 16-Batch Softmax Models on the Wine Dataset

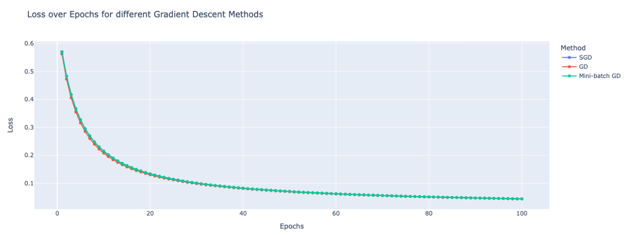


Figure 1: Loss by Epoch for different Softmax methods.

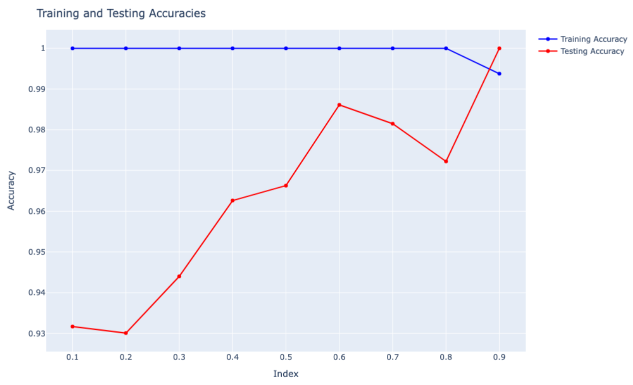


Figure 2: Train and Test Accuracies by Epoch for different Split Size.

All three methods seem to effectively reduce the loss as the number of epochs increases and the loss for all three methods converges towards a similar point by the end of the epochs shown (Figure 4).

Next, We assessed test accuracy based on split size, adjusting the Softmax model. After repeating each split 200 times, we observed improved test performance as training size increased, peaking at 90%. The 60% split offered optimal generalization.

We then evaluated the effect of batch size on model performance and speed, testing six sizes from 8 to 178 (the latter being fully-batched). Training scores rise quickly in initial epochs for all batch sizes, then stabilize, showing the model quickly grasps data patterns. The 16-batch size converges fastest and is optimal, although test scores for all sizes are consistently close. Next, we'll test the model's response to varying learning rates using the 16 minibatch model: `[0.5, 0.4, 0.3, 0.2, 0.1, 0.01, 0.001, 0.0001, 0.00001]`. Check plotly figures in `a1-Softmax.ipynb`.

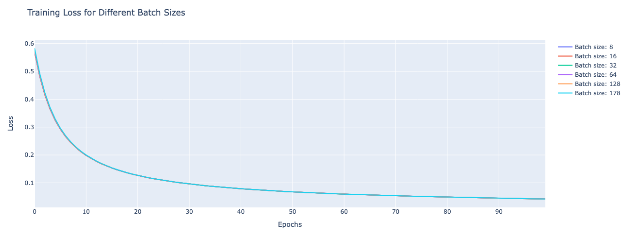


Figure 3: Training Loss by Different Batch Sizes

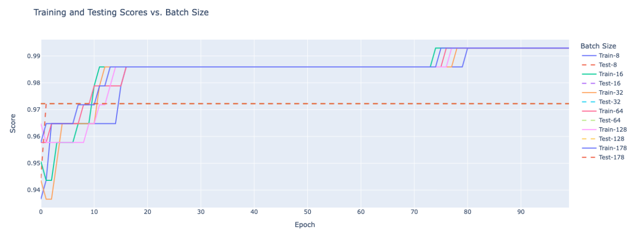


Figure 4: Training and Testing Scores by Batch-Size

Higher learning rates like 0.5 and 0.4 show quick loss reduction but then stabilize early, sign of overshooting and overfitting. Lower rates (0.01, 0.001) offer steadier loss decrease and better generalization. The very low

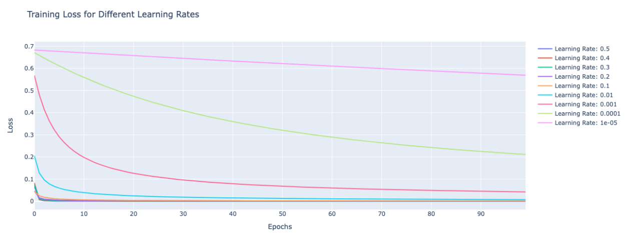


Figure 5: Training Loss by Different Learning Rates

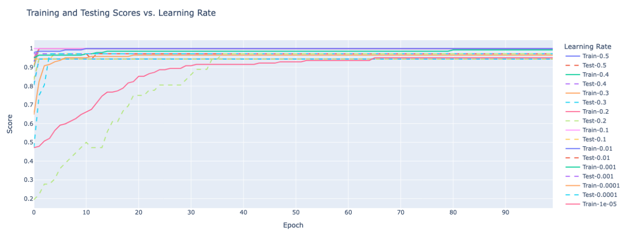


Figure 6: Training and Testing Scores by Learning Rates

rate of 1e-05 results in negligible loss decline, showing ineffective learning. The optimal learning rate, balancing convergence and generalization, is 0.001.

To determine the best model, we evaluated parameters using train and test accuracy, and generalization. Given the classification problem and balanced classes, accuracy was prioritized over metrics like Precision or Recall. Analysis from Figure 10 shows batch-size has minimal impact on performance, making learning rate the key factor. The optimal configuration (lr=0.001, mini-batch=16) train and test accuracies of 0.9929 and 1, respectively (see decisions figures in the notebook).

We conducted extra experiments involving l2-regularization, outliers removal and highly correlated feature removal. To keep the report concise, we will report some of the key results in the conclusion and discussion. Complete results can be consulted in the notebook.

5 Conclusion and Discussion

5.1 Boston Dataset

We've found some interesting things while working with our Regression models. First off, the Analytical method did better than the Gradient Descent method, which wasn't surprising because the Analytical method has an advantage in finding the best solution directly. But what really caught our attention was how much better our model got when we used something called Gaussian basis transformation – it improved by about 30%! Looking forward, we plan to keep improving our model. We'll explore different ways to make it even better, like trying out other methods such as sigmoid, trigonometry, and cross-multiplication transformations. It's worth noting that while the Analytical method worked well for our small dataset, it may not be as feasible for larger datasets due to computational limitations.

5.2 Wine Dataset

For multi-class logistic regression on the wine dataset, it is clear that the over simplistic structure of the data in addition to its thin size helped fitting a near-to-perfect model. Using an 80/20% train-test split with softmax regression, the best model-with a 0.001 learning rate and 16-batch size—achieved 5-fold cross-validation train and test accuracies, respectively. However, L2 regularization reduced test accuracy to 0.9444, likely due to its constraining effect on the model. Future work will delve into alternative outlier treatments, further preprocessing, and the exploration of other classifiers on this dataset.

6 Appendix and References

6.1 Boston Dataset Boxplot

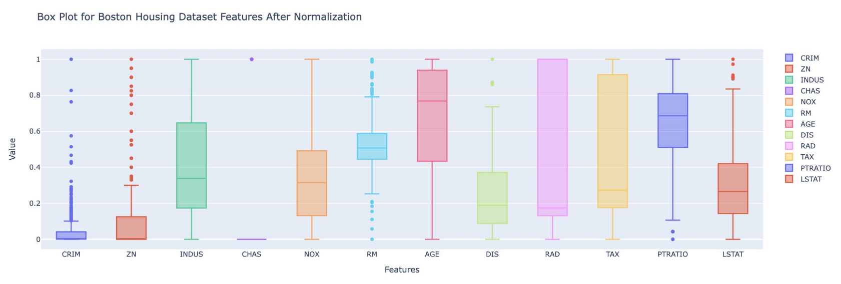


Figure 7: Boxplot summarizing the distributions of the features after normalization.

6.2 Wine Dataset Boxplots

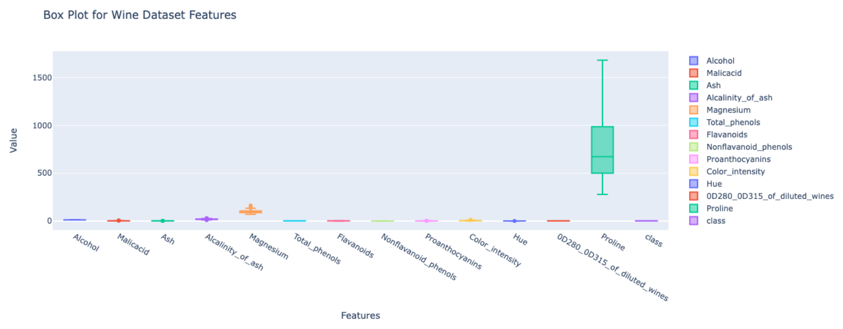


Figure 8: Boxplot summarizing the distributions of the features before normalization.

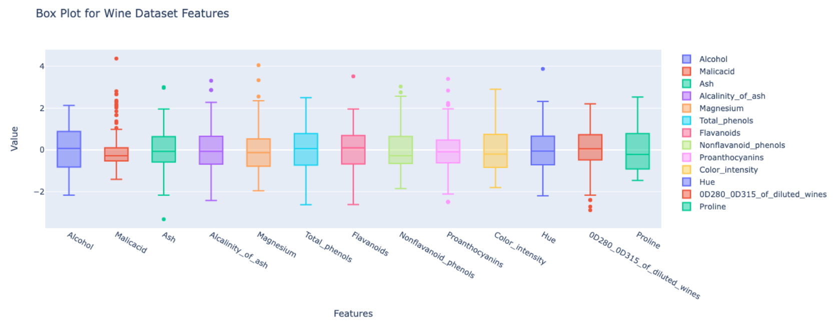


Figure 9: Boxplot summarizing the distributions of the features after normalization.

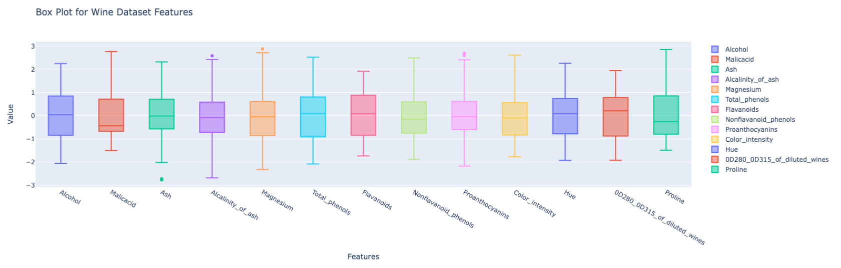


Figure 10: Boxplot summarizing the normalized distributions of the features after outliers removal (IQR).

6.3 Reference

[1] S. Sanyal, S. Kumar Biswas, D. Das, M. Chakraborty and B. Purkayastha, ”Boston House Price Prediction Using Regression Models,” 2022 2nd International Conference on Intelligent Technologies (CONIT), Hubli, India, 2022, pp. 1-6, doi: 10.1109/CONIT55038.2022.9848309.