CSE 6363: Assignment 1 Report

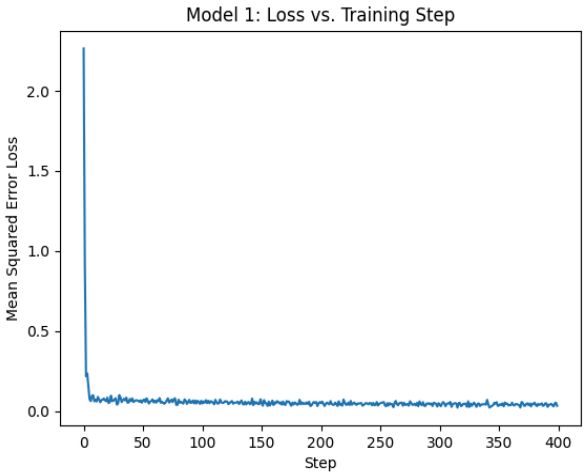
Name: Adhiraj Sen **ID:** 1002264465

Part 1: Linear Regression

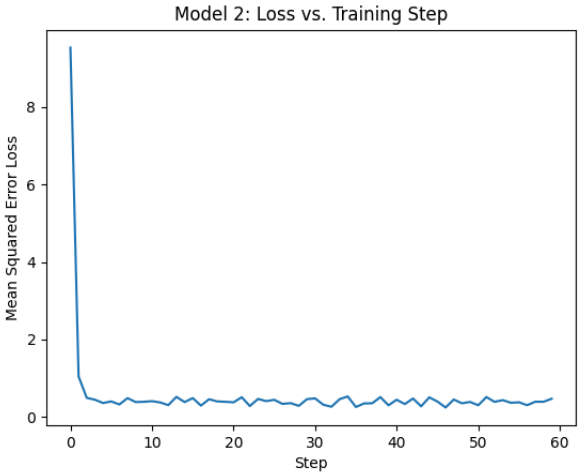
1.1 Model Training Loss

Below are the plots of MSE loss vs. training step for the four regression models.

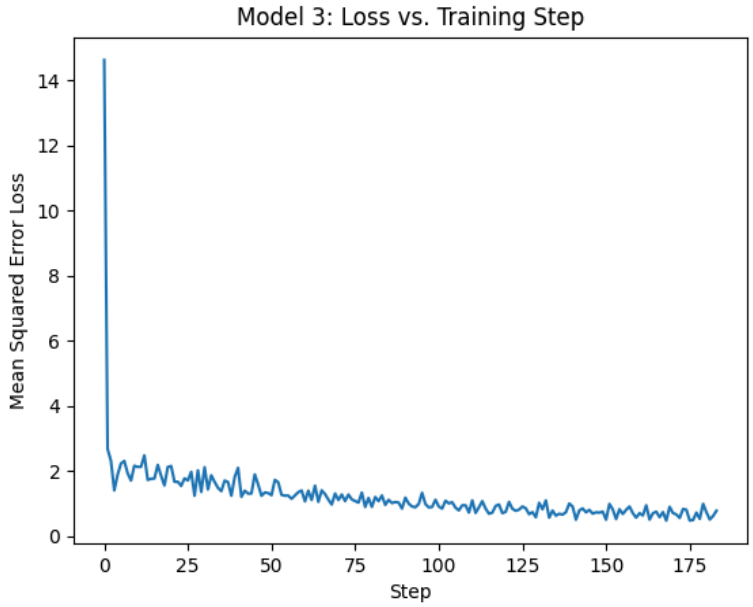
Model 1: Petal Length -> Petal Width



Model 2: Sepal Length -> Sepal Width

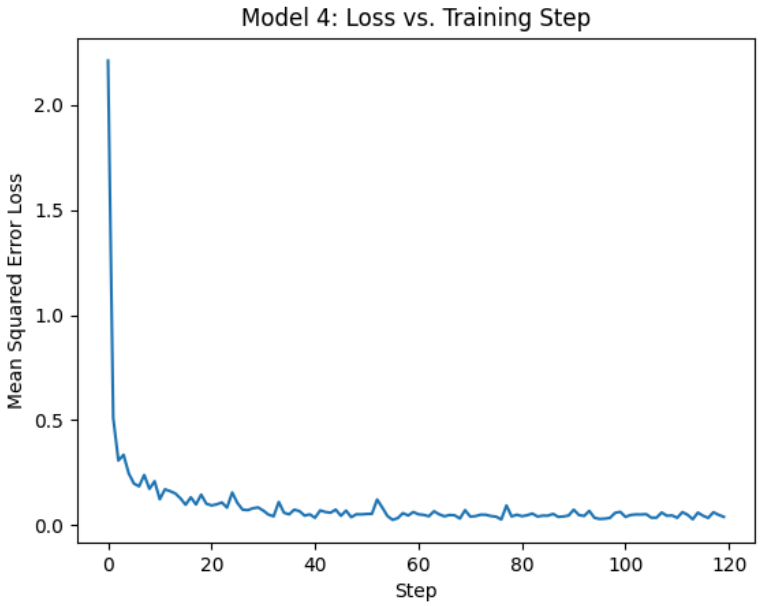


Model 3: Sepal Features -> Petal Length



Model 4: All other Features -> Petal Width

1.2 Regularization Effects



Here is a comparison of the learned model parameters for a task with and without L2 regularization. The weights for the regularized model are smaller , which helps prevent overfitting.

Model	**Weights**	**Bias**
No Regularization	0.39973418	-0.26487918
L2 Regularization	0.39189157	-0.23808083

1.3 Model Evaluation

The following Mean Squared Error (MSE) values were recorded on the unseen test set.

Model Task	**Test MSE**
1. Petal Length -> Petal Width	1.0719
2. Sepal Length -> Sepal Width	0.2862
3. Sepal Features -> Petal Length	4.7714
4. All others -> Petal Width	1.0994
5. Multi-Output (Sepal -> Petal)	0.4819

Analysis: Based on the results, the model with the lowest MSE was **Model 2**. This indicates that **Sepal length is the most predictive feature for determining Sepal width** in the Iris dataset, as their relationship is highly linear.

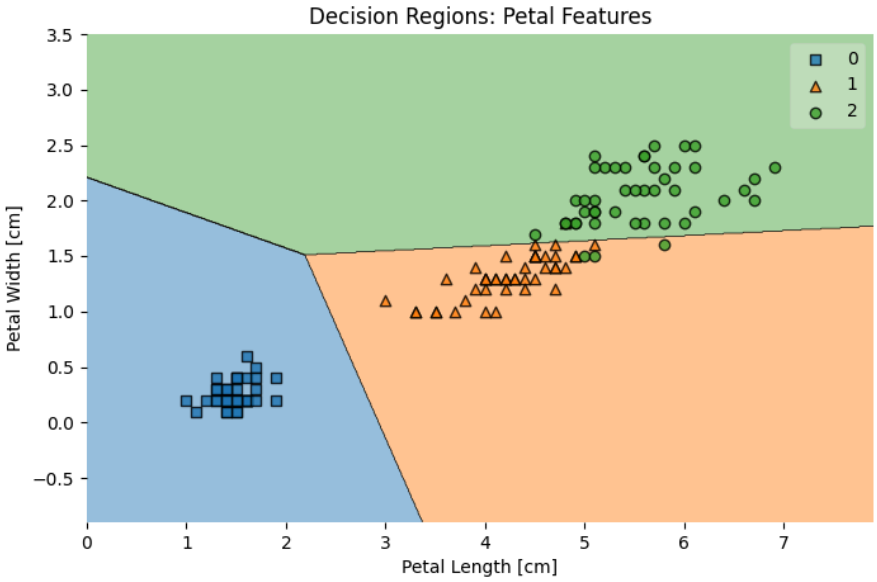
Part 2: Logistic Regression

2.1 Model Evaluation & Decision Regions

Variant 1: Petal Features

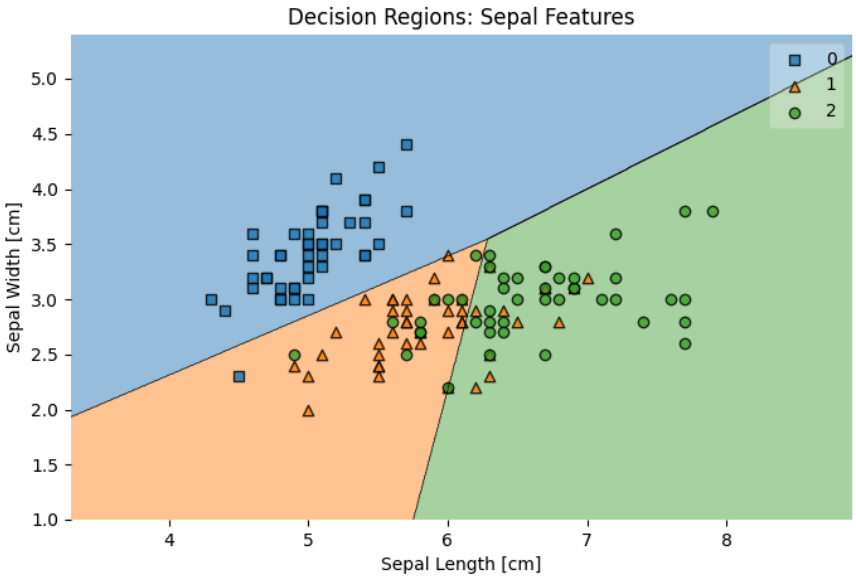
- **Test Accuracy:** 87%

- **Decision Regions:** The decision boundaries show a clean separation between the classes, explaining the high accuracy.



Variant 2: Sepal Features

- **Test Accuracy:** 73%
- **Decision Regions:** The classes are not as well-separated using sepal features, leading to lower accuracy. *[INSERT classifier2_regions.png HERE]*



Variant 3: All Features

- **Test Accuracy:** 100%

Brief Analysis: The petal features resulted in a very high accuracy and a clean separation between classes. The sepal features performed less effectively. Using all features yielded the highest accuracy, showing that even the less-predictive sepal features provide some useful information for the classifier.

Part 3: Questions

Linear Regression

What are the pros and cons of using the normal equation to solve for the weights in linear regression as opposed to using gradient descent?

The **normal equation** is a direct, analytical method that solves for the optimal weights in a single calculation, whereas **gradient descent** is an iterative optimization algorithm that gradually approaches the optimal solution.

- **Pros of Normal Equation:** It requires no hyper-parameter tuning (like a learning rate) and is guaranteed to find the global minimum. It's very efficient for datasets with a small number of features.

- **Cons of Normal Equation:** It is computationally very slow for datasets with a large number of features (due to the high cost of matrix inversion) and fails if the feature matrix is non-invertible.

- **Pros of Gradient Descent:** It scales efficiently to datasets with a very large number of features and can handle datasets that are too large to fit in memory.

- **Cons of Gradient Descent:** It requires careful tuning of the learning rate and other hyperparameters. It also benefits from feature scaling to ensure faster convergence.

In summary, the normal equation is ideal for smaller-scale problems, while gradient descent is necessary for large-scale applications.

Logistic Regression

Why is the softmax function used in multi-class logistic regression (Hint: the model itself produces logits)?

The softmax function is essential in multi-class logistic regression because it **converts the raw, unbounded scores from the model (called logits) into a meaningful probability distribution.**

A linear model produces a raw score, or logit, for each possible class. These scores are not probabilities; they can be any real number and do not sum to 1. To make a final classification, we need to interpret these scores as the probability that the input belongs to each class.

The softmax function accomplishes this by taking the vector of logits and performing two steps:

1. It exponentiates each logit, which makes all values positive.
2. It normalizes these positive values by dividing each by their sum.

The result is a new vector where each element is between 0 and 1 and the sum of all elements equals 1. This allows us to interpret the model's output as a set of probabilities and confidently select the class with the highest probability as our prediction.