Assignment 1

Terms used:

lr_model_x = Linear Regression Model ; where x = 1, 2, 3, 4, multiple

lg_model_x = Logistic Regression Model; where x = 1, 2, 3

1. **Linear Regression**
   a. **Sepal length and sepal width to predict petal length**

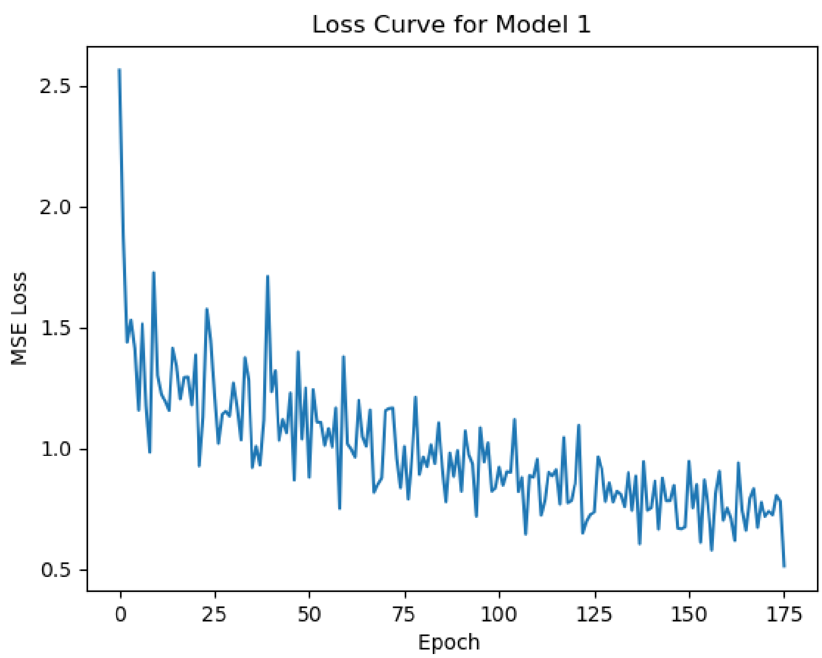
Scripts used:

Training : train_regression1.py

Testing: eval_regression1.py

Model parameters:

lr_model_1_params.npz



Weights:[[1.16068628] [-0.89237741]]

Bias: [[-0.08763533]]

MSE for Model 1: 0.9490261627621038

b. Petal length and petal width to predict sepal length

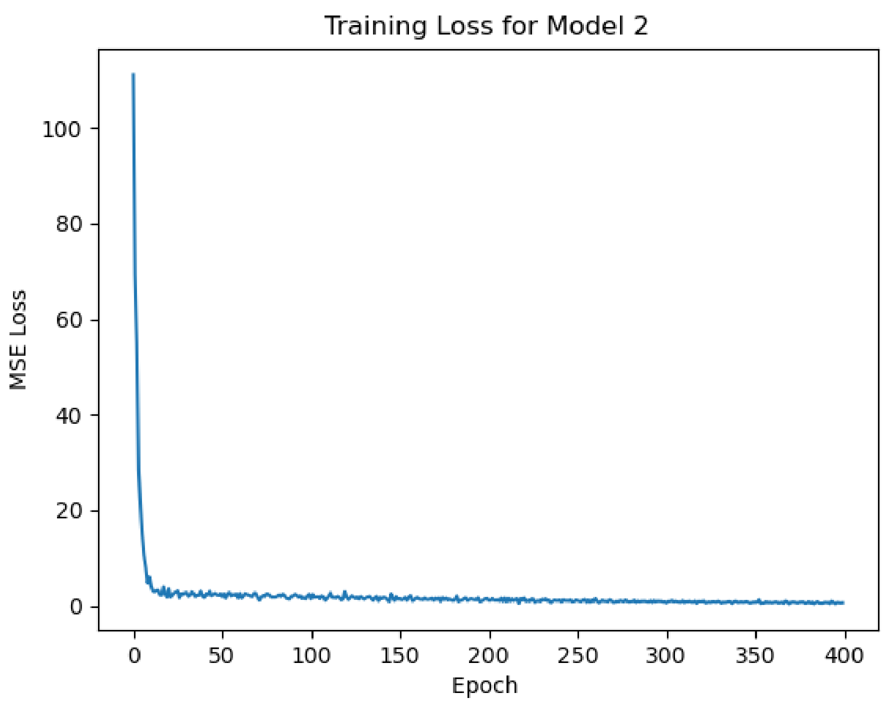
Scripts used:

Training : train_regression2.py

Testing: eval_regression2.py

Model parameters:

lr_model_2_params.npz



Weights: [[1.05499719] [-0.65426791]]

Bias: [[2.35372656]]

MSE for Model 2: 0.9787624755134201

c. Sepal length and petal length to predict sepal width

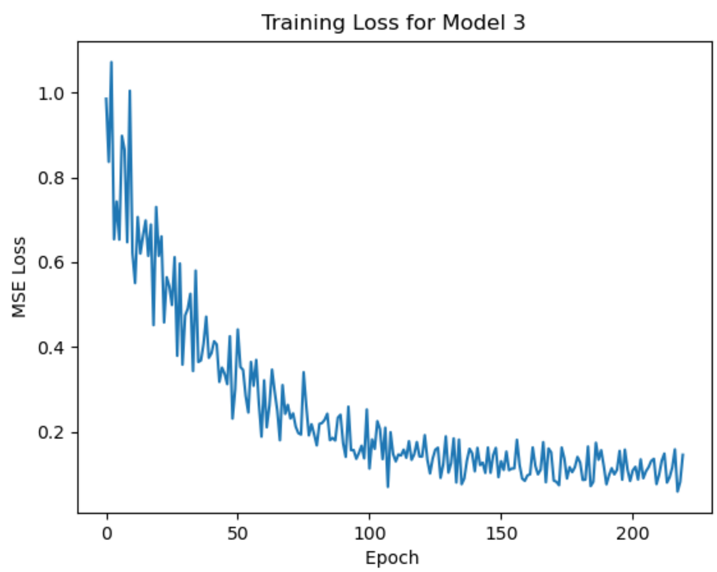
Scripts used:

Training : train_regression3.py

Testing: eval_regression3.py

Model parameters:

lr_model_3_params.npz



Weights: [[0.86673967][-0.48912179]]

Bias: [[-0.13604427]]

MSE for Model 3: 0.15172934183947107

d. Sepal width and petal width to predict sepal length

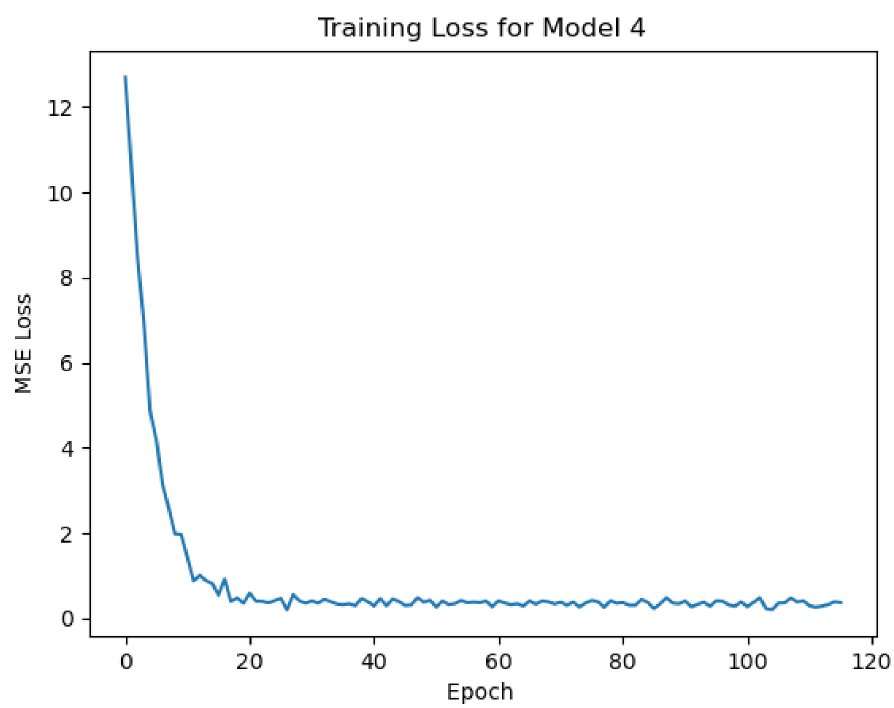
Scripts used:

Training : train_regression4.py

Testing: eval_regression4.py

Model parameters:

lr_model_4_params.npz



Weights: [[1.35679852] [1.06841669]]

Bias: [[0.3387008]]

MSE for Model 4: 0.42954057001399265

e. Regularization

Inspecting Model 3 (**Sepal length and petal length to predict sepal width)**

Without regularization:

Weights: [[0.86673967][-0.48912179]]

Bias: [[-0.13604427]]

MSE for Model 3: 0.15172934183947107

With regularization (λ):

lr_model.fit(x_train,y_train,*regularization*=0.01)

The results of varying the values of λ are shown below:

λ	Weights	Bias	MSE Loss
0.01	[[0.61075641][0.19216974]]	[[0.11286589]]	0.215145
0.05	[[0.55507437][-0.2051895]]	[[0.50107823]]	0.172046
0.1	[[0.60869781][-0.17519284]]	[[0.05839776]]	0.237135
0.5	[[0.58800414][-0.18037514]]	[[0.16748929]]	0.218567
1	[[0.54053605][-0.07587891]]	[[-0.0282996]]	0.374008
5	[[0.15032523] [0.04434984]]	[[1.83167185]]	0.336183
10	[[0.0736331][0.02650664]]	[[2.23727554]]	0.321141
50	[[0.01028876] [0.00168798]]	[[2.80267839]]	0.218465

Observations

- With low values of λ, weights are close to non-regularized model
- With moderate/high values of λ, weights shrink significantly
- With higher values of λ, the weights are almost negligible while the value of bias increases. The model may underfit and produce poor predictions since the model now relies on adjusting the bias to minimize the loss rather than relying on feature weights.
- MSE loss decreases initially when moving from λ = 0.01 to 0.05, suggesting that regularization improves loss. As λ increases further, loss starts increasing (λ @ 1, 5) due to underfitting.

f. **Input Feature Most Predictive of Its Corresponding Output Feature**

- **Model 1: Sepal length and sepal width to predict petal length**
    - Sepal Length: 1.16068628 Sepal Width: -0.89237741
    - Sepal length is highly predictive of petal length

- **Model 2: Petal length and petal width to predict sepal length**
    - Petal Length: [1.05499719] Petal Width: [-0.65426791]]
    - Petal length is more predictive of sepal length

- **Model 3: Sepal length and petal length to predict sepal width**
    - Sepal Length: [0.86673967] Petal Length: [-0.48912179]
    - Sepal length slightly more predictive of sepal width

- **Model 4: Sepal width and petal width to predict sepal length**
    - Sepal width: [1.35679852] Petal Width: [1.06841669]
    - Sepal width is slightly more predictive of sepal length

g. **Regression with Multiple Outputs**

- **Sepal length and sepal width to predict petal length and petal width**

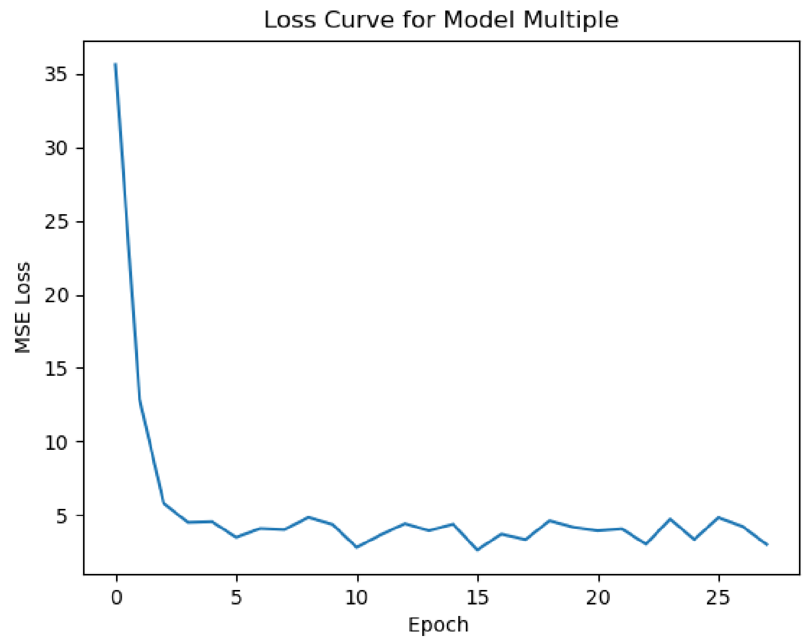
  Scripts used:

  Training : train_regression_multiple.py

  Testing: eval_regression_multiple.py

  Model parameters:

  lr_model_multiple_params.npz



Weights: [[0.81687247 0.16240173][-0.71750908 0.51966565]]

Bias: [[-0.16773643 -0.16773643]]

MSE : 2.1649863988752047

2. Logistic Regression

a. Petal length and petal width

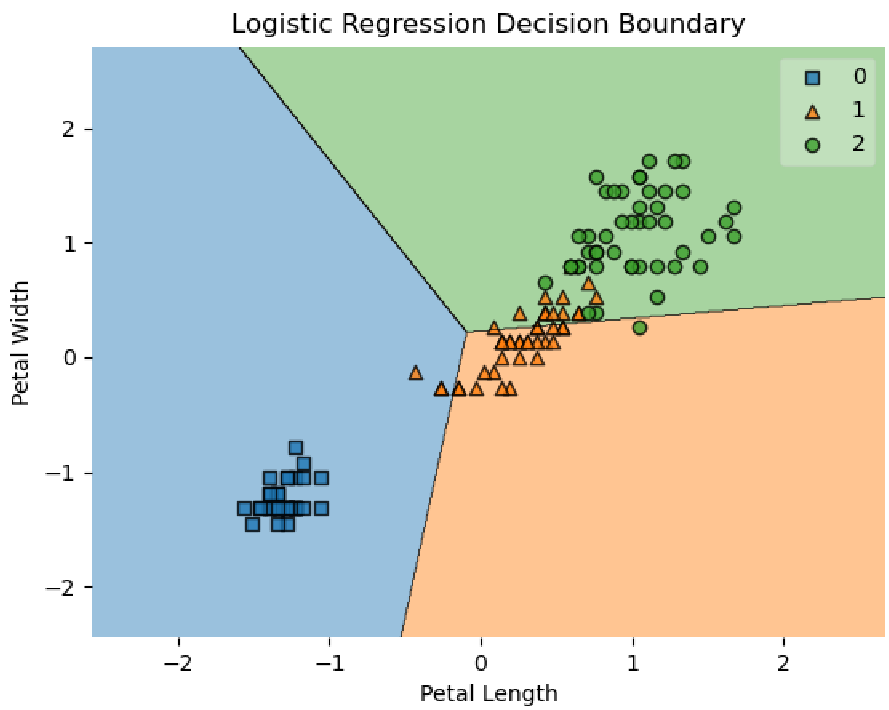
Scripts used:

Training : train_classifier1.py

Testing: eval_classifier1.py

Model parameters:

lr_model_1_params.npz



Test Accuracy: 86.67%

b. Sepal length and Sepal width

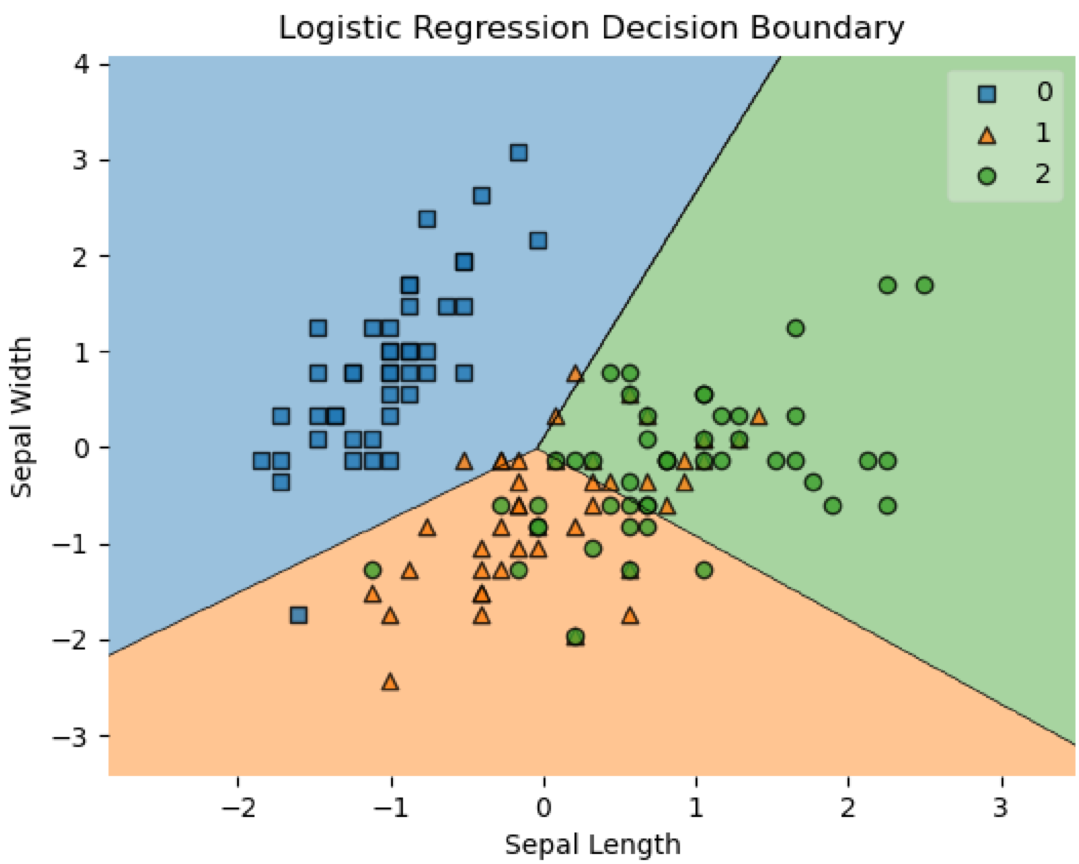
Scripts used:

Training : train_classifier2.py

Testing: eval_classifier2.py

Model parameters:

lg_model_2_params.npz



Test Accuracy: 93.33%

c. **All features**

Scripts used:

Training : train_classifier3.py

Testing: eval_classifier3.py

Model parameters:

lg_model_3_params.npz

Test Accuracy: 93.33%

Resources:

Initializing Weights for Deep Learning Models - MachineLearningMastery.com

OneHotEncoder — scikit-learn 1.6.1 documentation

rasbt/mlxtend: A library of extension and helper modules for Python's data analysis and machine learning libraries.

https://www.youtube.com/playlist?list=PLblh5JKOoLUICTaGLRoHQDuF_7q2GfuJF

Derivation of the Binary Cross Entropy Loss Gradient