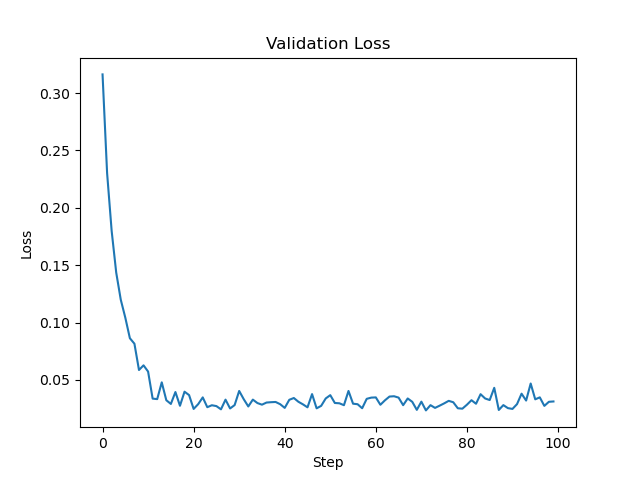
What is the most predictive output feature?

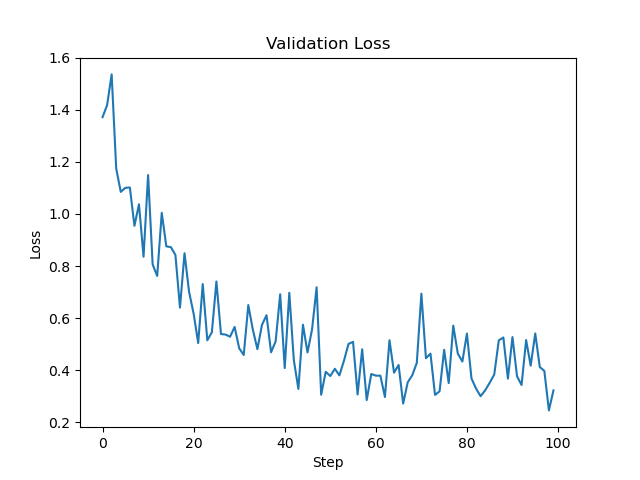
The most predictive output feature is trying to predict petal width given petal length and sepal width out of all the models I’ve tested. I’m guessing the other models like predicting petal length given sepal length and width aren’t as predictive because those features don’t work as best together, and so forth with other models. Train\_regression1 has the lowest mean squared error.

**Train\_regression1 # Predict petal width given petal length and sepal width**

****

**Mean squared error of the 'linreg\_sepalwidth\_petallength\_target\_petalwidth\_model.npz' model on the test set: 0.05**

**Train\_regression2 # Predict petal length given sepal length and sepal width**

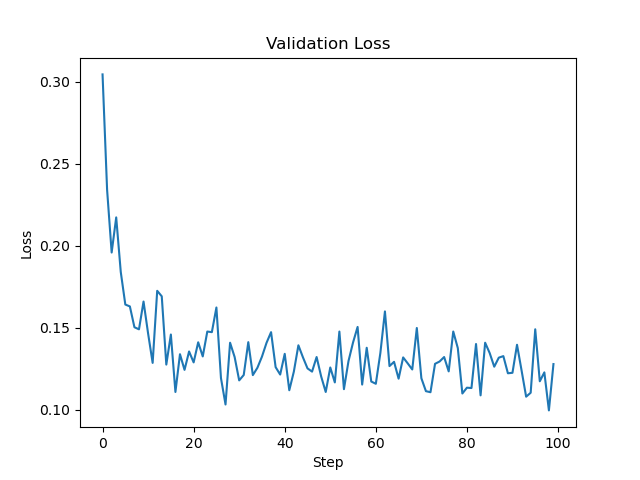
****

**Mean squared error of the 'linreg\_sepallength\_sepalwidth\_target\_petallength\_model.npz' model on the test set: 0.37**

**USED REGULARIZATION FOR MODEL #2**

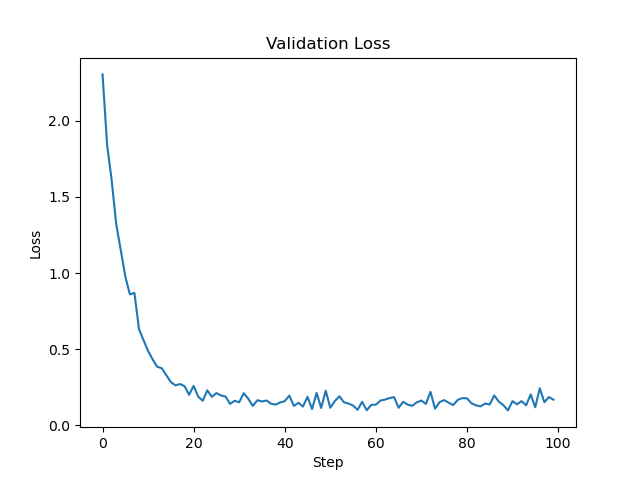
By adding value of 0.001 for regularization, it slightly increased my mse by .10, perhaps increasing regularization for this set of features is harmful to prediction. Perhaps it is too simple or dataset is too small?

**Train\_regression3 # Predict sepal length given petal width, petal length and sepal width**

**2**

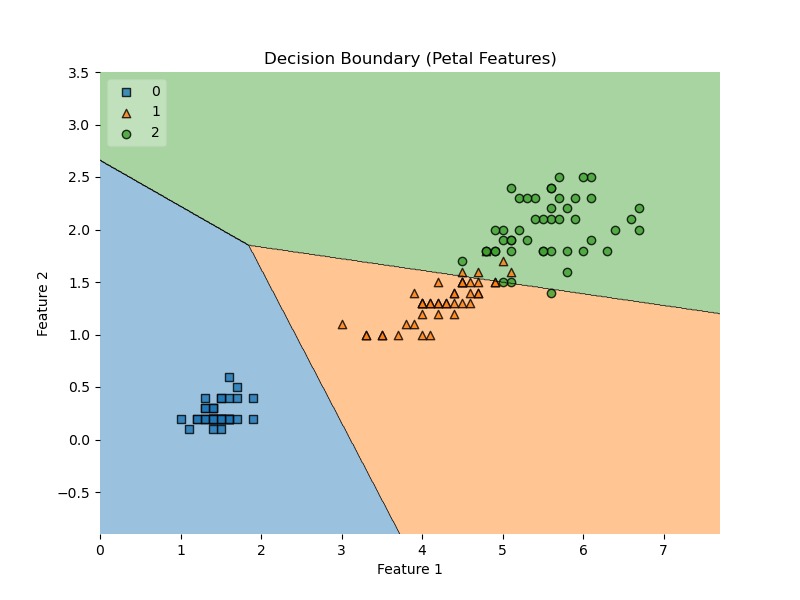
**Mean squared error of the 'linreg\_sepalwidth\_petallength\_petalwidth\_target\_sepallength\_model.npz' model on the test set: 0.15**

**Train\_regression4 # Predict sepal width given petal length and sepal length**

****

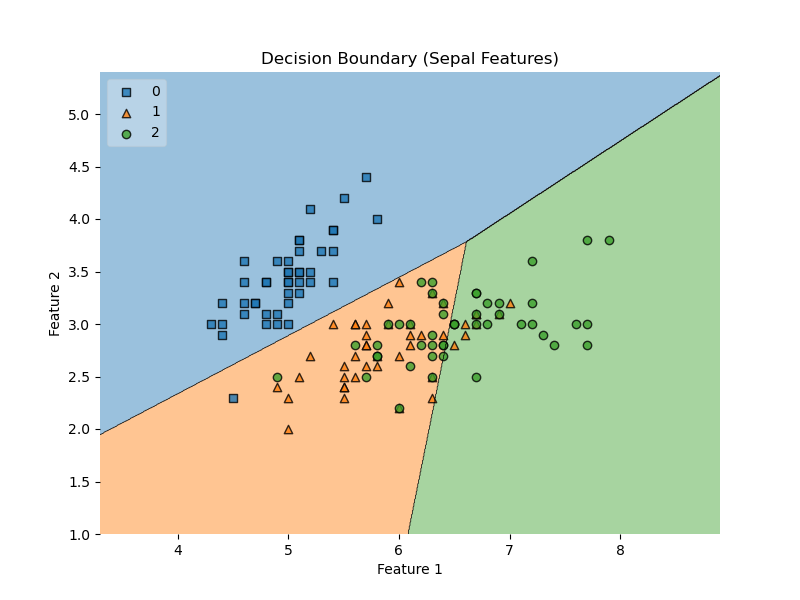
**Mean squared error of the 'linreg\_sepallength\_petallength\_target\_sepalwidth\_model.npz' model on the test set: 0.08**

**Train\_classifer1 # Classification with petal length and width**

****

**Model accuracy: 0.8666666666666667**

**Train\_classifer2 # Classification with sepal length and sepal width**

****

**Model accuracy: 0.6666666666666666**

**Train\_classifer3 # Classification with all features**

**Model accuracy: 1.0**

**# Linear Regression**

1. 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?

Pros of a normal equation is that its fast for smaller datasets and you don’t need to tweak any hyperparameters. However normal equations store the entire transposed X times X matrix in memory, which is difficult to do if you are using a large data set. Once again, if you use a large data set, its computationally expensive to do the multiplication of the transposed X and X and then inversing.

**# Logistic Regression**

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

The softmax function is used in multi-class logistic regression because the model produces logits. By taking the vector of logits, which are raw scores and using softmax to transform them into probabilities, it produces more useful output that can be used for classification. Softmax unlike sigmoid allows classification of more than two classes, hence why using softmax is best for multiclass.