**Homework #1**

GitHub: <https://github.com/pballou/ECGR_4106/tree/master/Homework>

1. Develop a multi-layer perceptron with three hidden layers (you pick the dimensions of the hidden layers) for the CIFAR-10 dataset.
   1. Train the model from scratch (with randomized parameters) and plot the results (training loss and accuracy, validation accuracy) after 20 epochs.

A graph of a training and valdalation accuracy

Description automatically generated with medium confidence

* + 1. Does your network need more epochs for full training?

**Yes, it is clearly still in the process of converging.**

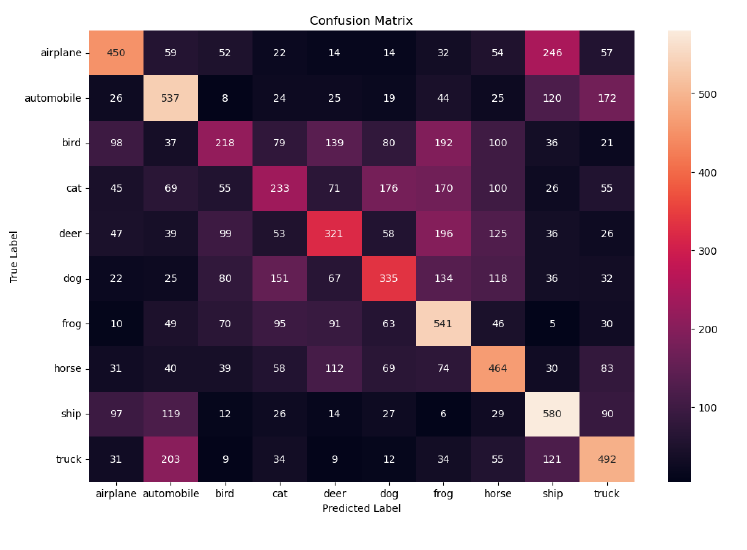
* + 1. Do you observe overfitting?

**Not too much, as validation is changing with training.**

* + 1. Make sure to save the trained parameters and model. Report and plot your training and validation results. Report precision, recall, F1 score, and confusion matrix.

A screenshot of a graph

Description automatically generated



* 1. Explore the complexity of the network by increasing its width and depth.
     1. How do the training and validation results change?

**They both got worse. Fully connected networks are not very good with structured data like images.**

A graph of training and validation loss

Description automatically generated

A graph of a line

Description automatically generated

* + 1. Compare them against the baseline. Do you see any overfitting?

Not much, but the model is much worse overall.

1. Please implement the following steps for the housing dataset we overviewed during the lectures.
   1. Build a multi-perceptron network that regresses the housing price (based on 20%, 80% split). Use the same number of features we did in the lecture without one-hot encoding. Please plot the training and validation results and report final accuracy and model complexity.

**I used a 3-layer network and here are the results:**

**A number of numbers on a white background

Description automatically generated**

**A graph of training and validation loss

Description automatically generated**

* 1. Build a multi-perceptron network that regresses the housing price (based on 20%, 80% split). Use the same number of features we did in the lecture, but this time also add one-hot encoding. Please plot the training and validation results and report the final accuracy and model complexity.

A graph with numbers and a line

Description automatically generated

* + 1. Do you see meaningful changes against 2.a?

**Yes, using one-hot encoding clearly had a positive impact on the results. Only the RMSE got a little worse, but both training and validation loss decreased significantly, showing that one-hot encoding helped the model learn the data better.**

* 1. Increase the complexity of the network for problem 2.c. and compare your results against 2.b.

**I added a couple more layers and was able to get some better results. Although the training loss got slightly worse, both validation loss and RMSE got better, so overall it performed more favorably.**

**A graph with numbers and a line

Description automatically generated**