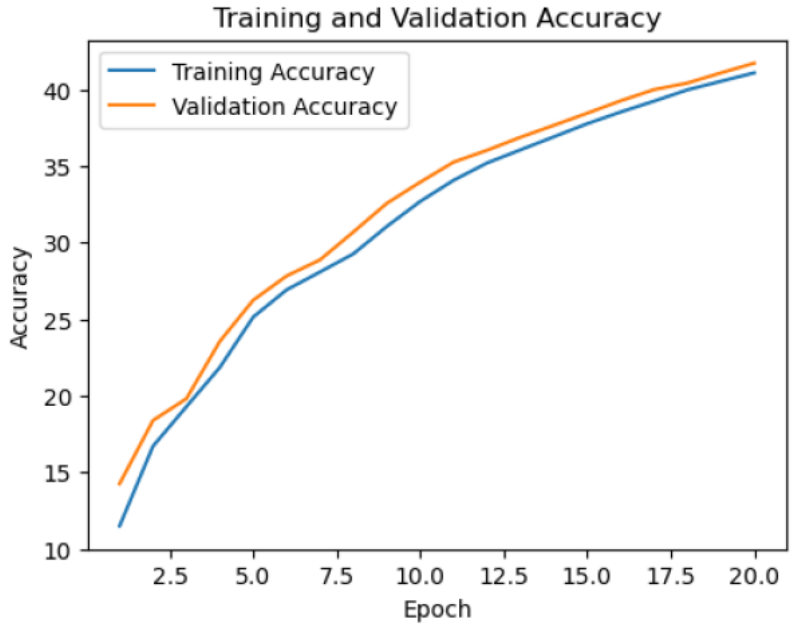
Homework #1

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.
    a. Train the model from scratch (with randomized parameters) and plot the results (training loss and accuracy, validation accuracy) after 20 epochs.

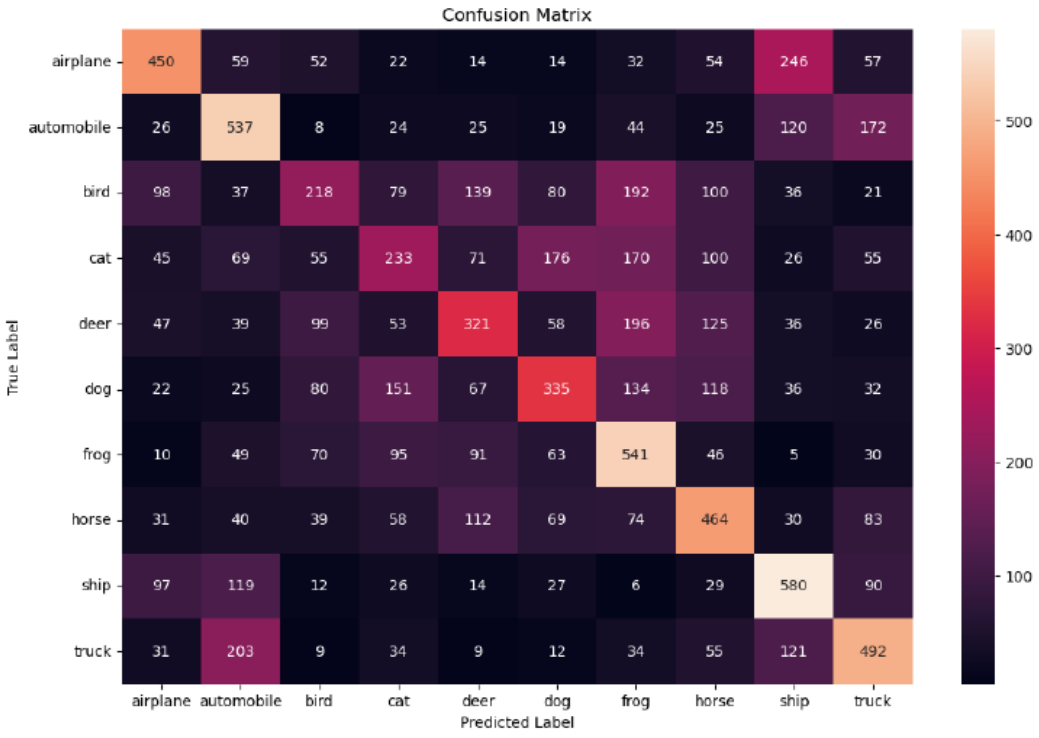


i. Does your network need more epochs for full training?
Yes, it is clearly still in the process of converging.

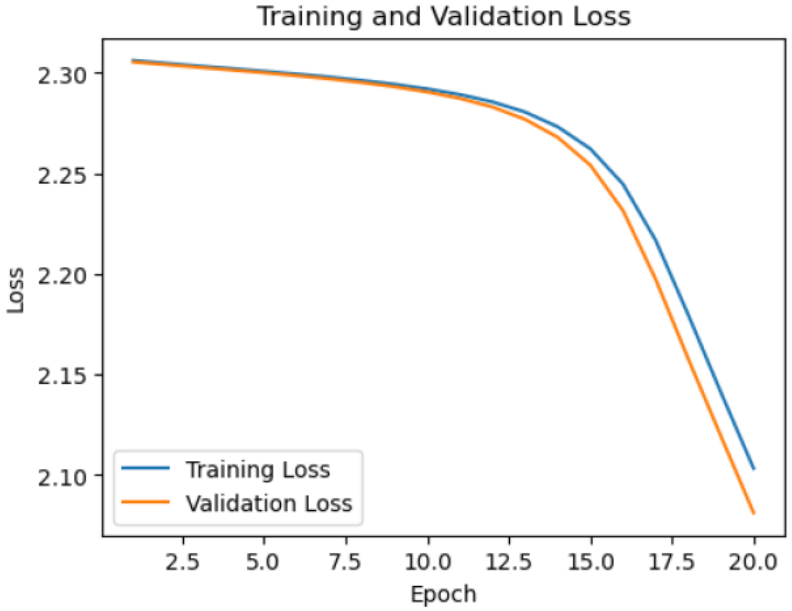
ii. Do you observe overfitting?
Not too much, as validation is changing with training.

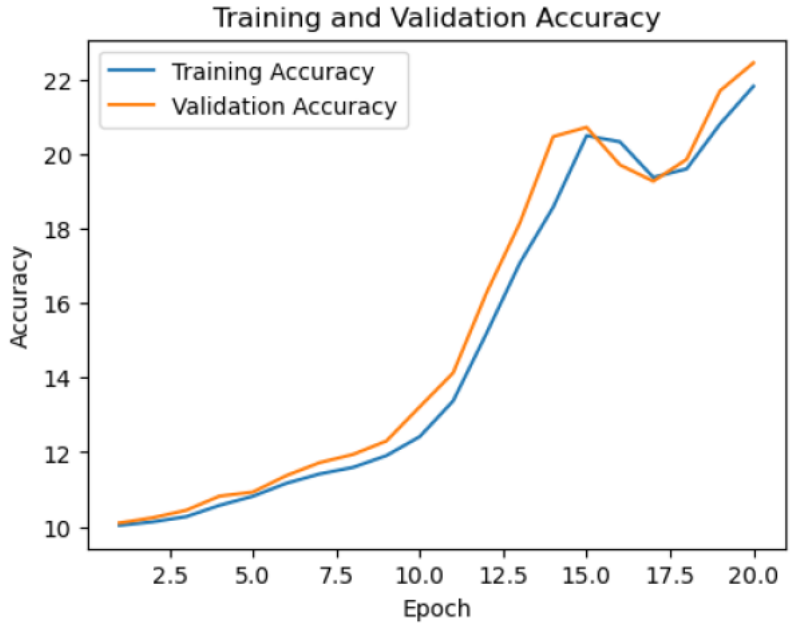
iii. Make sure to save the trained parameters and model. Report and plot your training and validation results. Report precision, recall, F1 score, and confusion matrix.

	precision	recall	f1-score
0	0.53	0.45	0.48
1	0.46	0.54	0.49
2	0.34	0.22	0.27
3	0.30	0.23	0.26
4	0.37	0.32	0.34
5	0.39	0.34	0.36
6	0.38	0.54	0.45
7	0.42	0.46	0.44
8	0.47	0.58	0.52
9	0.47	0.49	0.48
accuracy			0.42
macro avg	0.41	0.42	0.41
weighted avg	0.41	0.42	0.41



b. Explore the complexity of the network by increasing its width and depth.
   i. How do the training and validation results change?
   They both got worse. Fully connected networks are not very good with structured data like images.

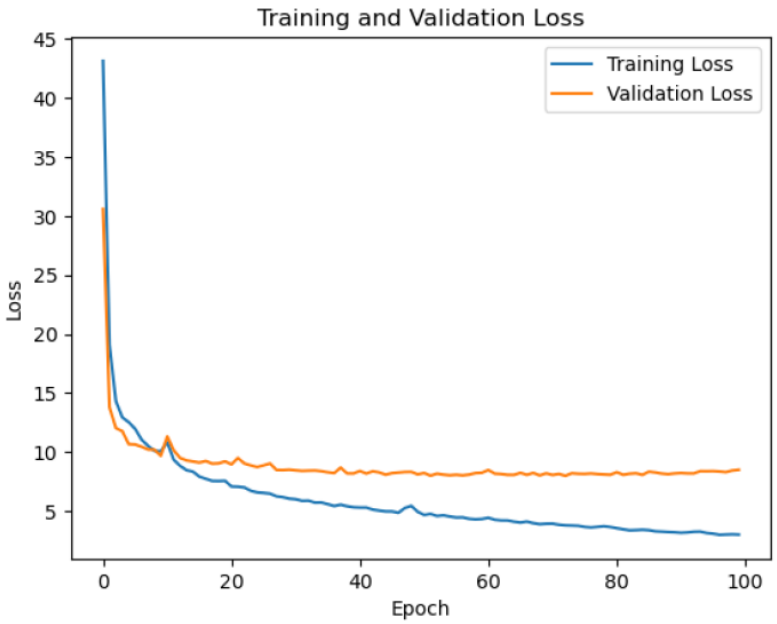




   ii. Compare them against the baseline. Do you see any overfitting?
   Not much, but the model is much worse overall.

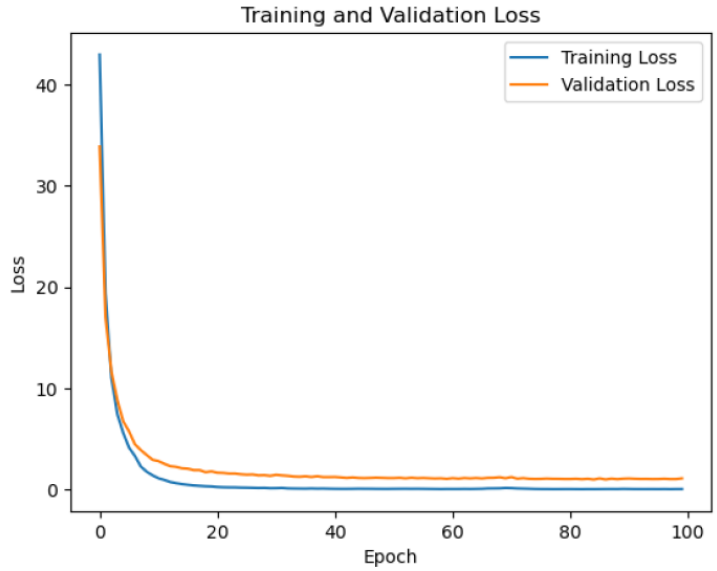
2. Please implement the following steps for the housing dataset we overviewed during the lectures.
    a. Build a multi-perceptron network that regresses the housing price (based on 20%, 80% split). Use the same number of features we did in the lecture without one-hot encoding. Please plot the training and validation results and report final accuracy and model complexity.
    I used a 3-layer network and here are the results:

```
Final Training Loss: 3.032738026819731
Final Validation Loss: 8.515102410316468
Final Validation RMSE: 0.3818462356939064
```



    b. 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.

```
Final Training Loss: 0.03129004304738421
Final Validation Loss: 1.0817846432328224
Final Validation RMSE: 1.0400887662804106
```



i. 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.

c. 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.

```
Final Training Loss: 0.14796865466786058
Final Validation Loss: 0.8522834584116936
Final Validation RMSE: 0.12080519385081238
```