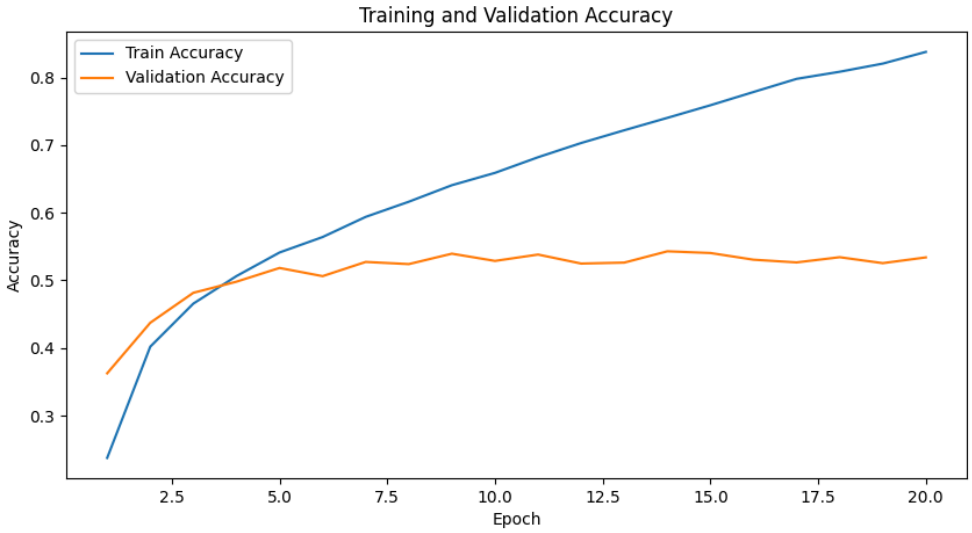
Problem 1 (50 points)

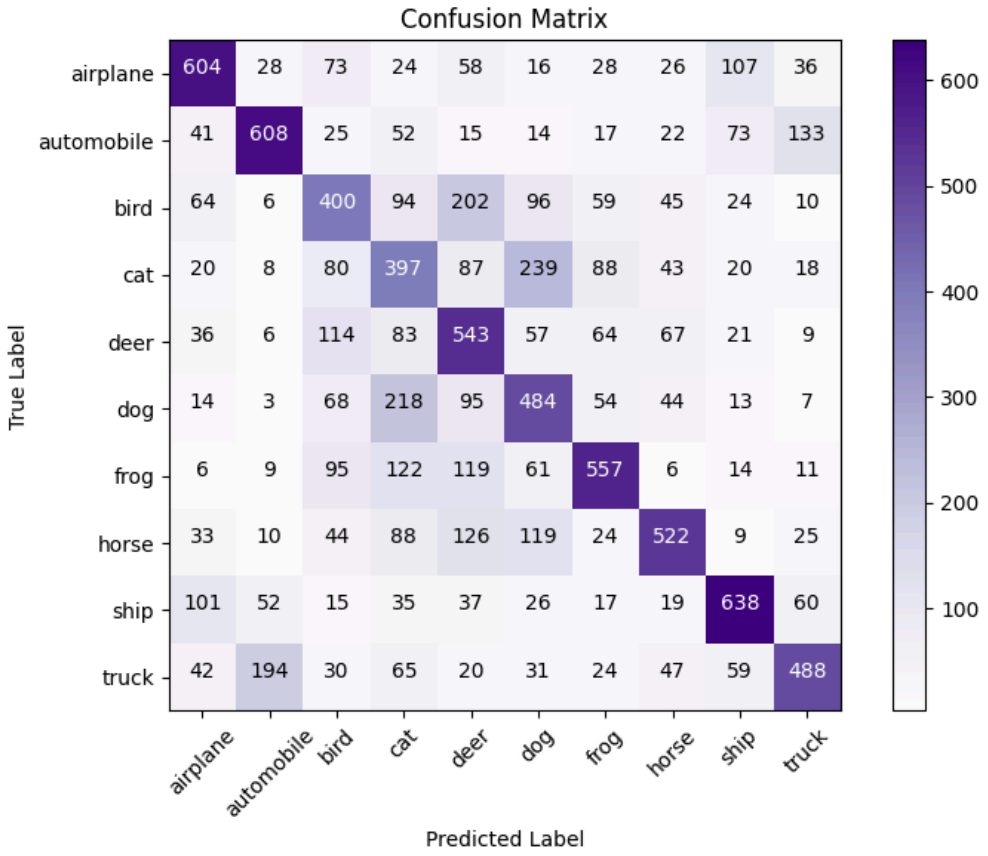
Develop a multi-layer perceptron with three hidden layers (you pick the dimensions of the hidden layers) for the CIFAR-10 dataset.

1.a)

Given the visualization of my accuracy and loss plots for testing and validation, they are indicative of overfitting. For accuracy, my training set improved over time, whereas my validation set stagnated. This is indicative of my model simply memorizing the training set and not learning anything. As such, overfitting was also observed through my loss plots, which showed the training loss decreasing over time and my validation loss increasing over time. Based on these results, it was my assumption that my network required more epochs for full training. However, increasing the epochs still showed the same patterns of overfitting, and so this is more to do with the complexity of the model than the epochs. While not included, implementing some sort of normalization or dropout layers could aid in the model's performance.


Training and Validation Accuracy
Train Accuracy
Validation Accuracy
Accuracy
Epoch
0.3
0.4
0.5
0.6
0.7
0.8
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

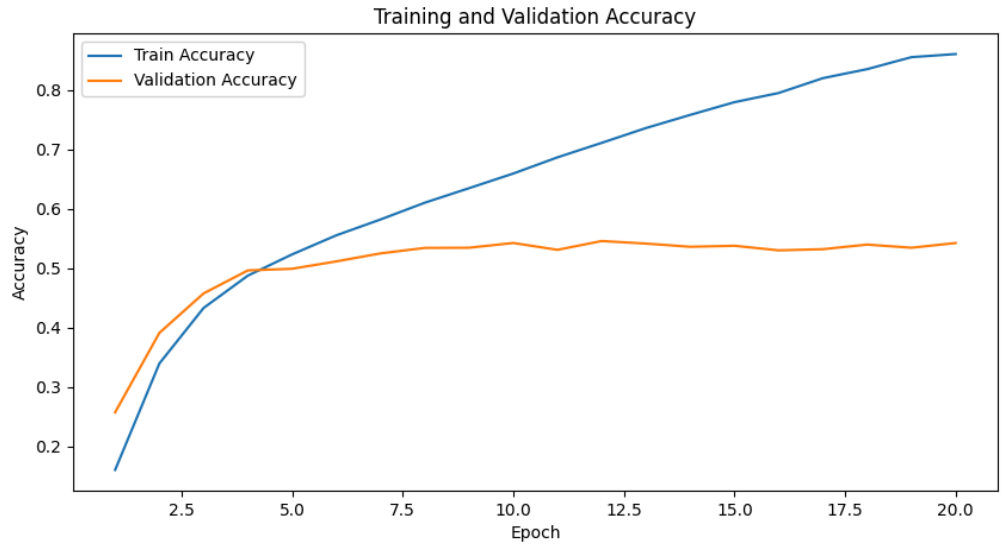

Training and Validation Loss
Train Loss
Validation Loss
Loss
Epoch
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
2.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

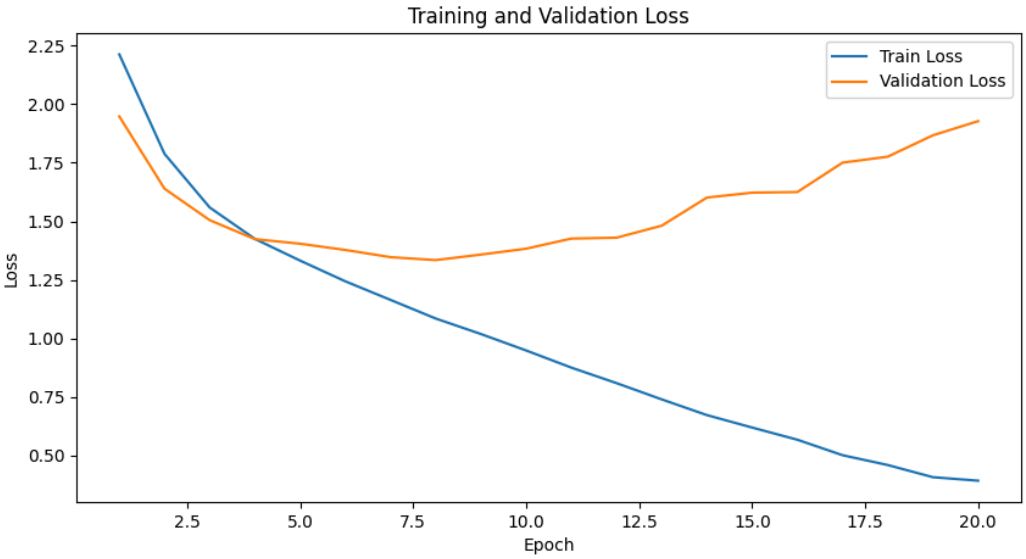


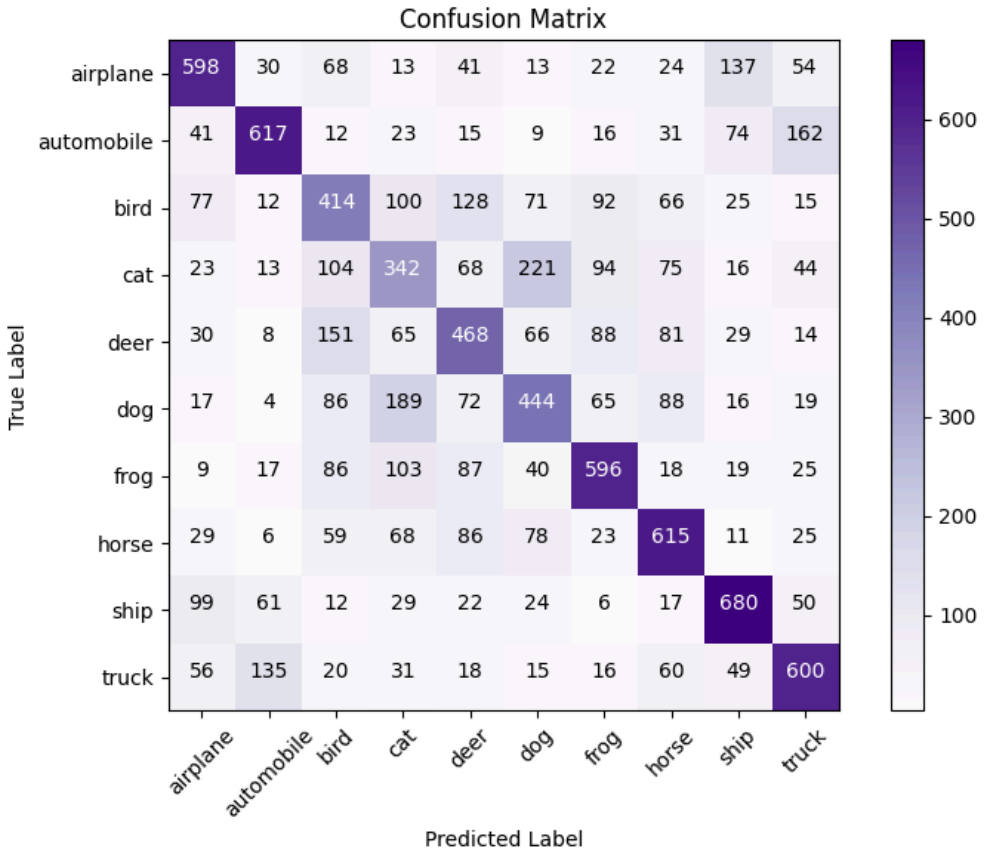
1.b)

Initially, after increasing the complexity of my network, the training and validation results did not change too drastically. Much of the code was kept the same, very barebones. I increased the complexity by including additional layers and neurons.

There was an increase in the training accuracy compared to 1.a, along with decreased loss. However, overall the model and its performance were very similar to 1.a, signs of overfitting were prevalent, and generalization worsened.

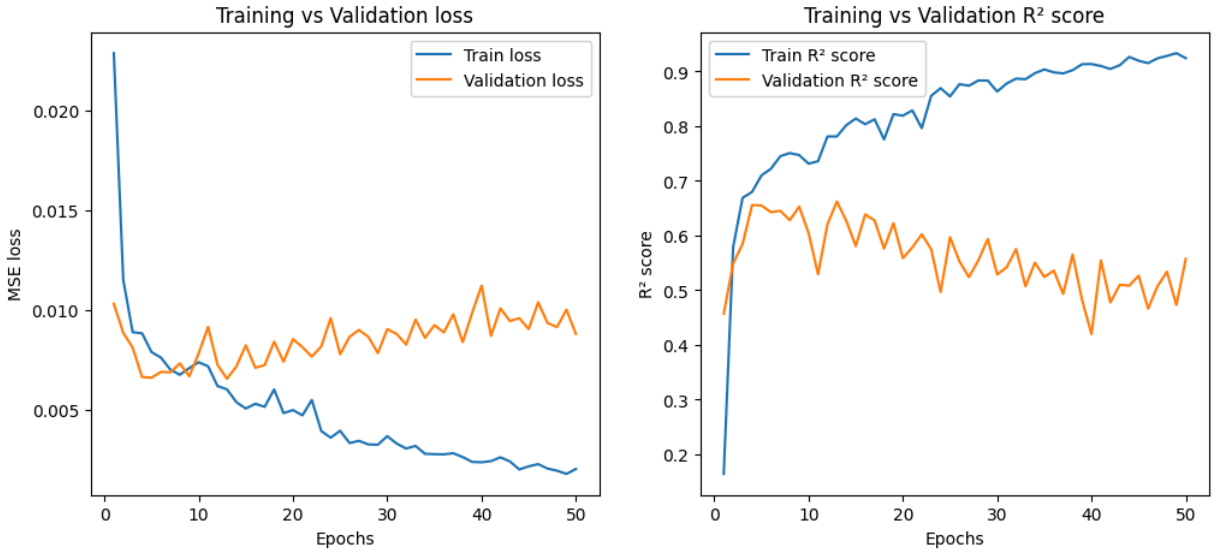

Training and Validation Accuracy
Train Accuracy
Validation Accuracy
Accuracy
Epoch
0.2
0.3
0.4
0.5
0.6
0.7
0.8
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0


Training and Validation Loss
Train Loss
Validation Loss
Loss
Epoch
0.50
0.75
1.00
1.25
1.50
1.75
2.00
2.25
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0



Problem 2 (50 points)

2.a)



Final Train R² Score: 0.9242

Final Validation R² Score: 0.5575

Results indicative of overfitting.

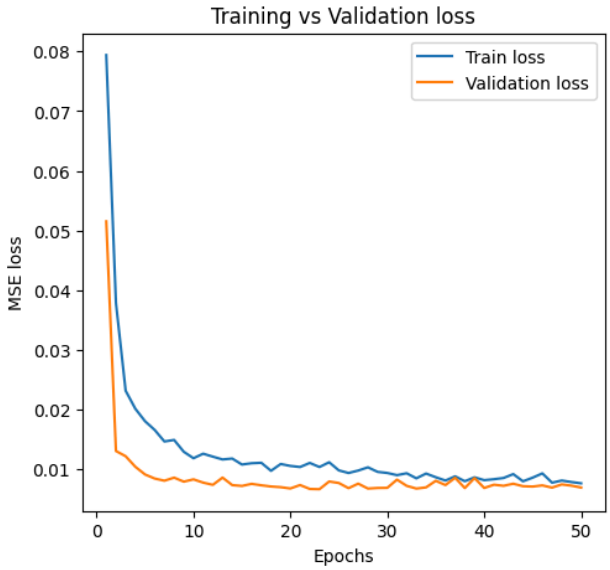
2.b)

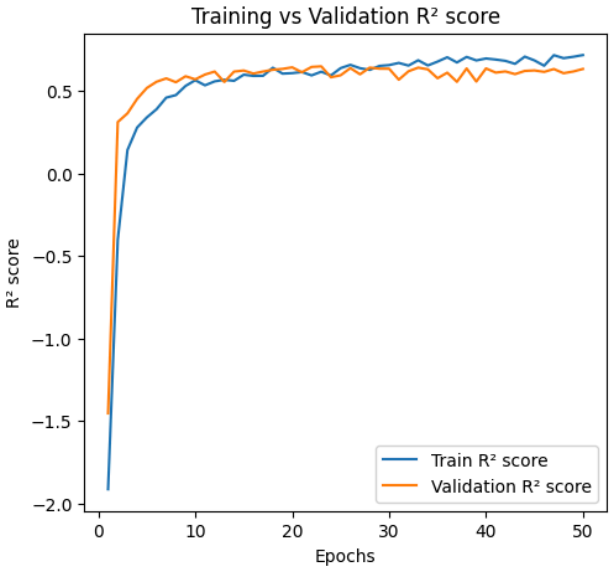
Final Train R² Score: 0.8960

Final Validation R² Score: 0.5358

Compared to 2.a, these results show that the model performed worse. It is unlikely that this is a direct result of one-hot encoding, and more to do with the model complexity capturing irrelevant patterns due to not having any regularization techniques to improve generalization.

2.c)





Final Train R² Score: 0.7184

Final Validation R² Score: 0.6340

For 2.c, I increased the complexity by having an additional layer, increased layer width/depth, and also added dropout after each layer. L2 regularization and a learning rate scheduler to optimize the training and, as shown by the results, this did lead to better generalization (compared to 2.b). The results also showed that there was less overfitting compared to the previous models, further refinements to the code could include batch normalization, experimenting with different activation functions, and fine-tuning the dropout rate and layer width/depth/.