Marco Galindo
ID: 310540013
2021, November 2nd
Deep Learning Fall 2021

Homework 1

Regression

a) Neural network

The neural network consists of the input layer with 17 features, 2 hidden layers of 5 neurons each, and a single neuron as the output layer. The activation function is the ReLU for the hidden layers, and the output layer does not have an activation function, as this is a regression problem. All the hyperparameters used can be seen in Table 1.

Table 1 Configuration for the neural network for the regression problem

Network architecture	17, 5, 5, 1
Learning rate	0.0001
Epochs	10,000
Activation function	ReLU
Output activation function	None
Cost function	Sum-of-squares
Mini-batch size	15
Final Training E_{RMS}	2.93509
Final Test E_{RMS}	3.20139

There is an early stopping algorithm in place, so if the test error starts to go up, the training is stopped.

The data was pre-processed by encoding the Glazing Area Distribution and Orientation features into one-hot vectors that represent the various values these two features have. To normalize the features vector, the Surface Area, Wall Area, Roof Area and Overall Height have been divided by their maximum value.

After the training process, the RMS for training and test data can be seen in Table 2.

Table 2 RMS for training and test data for the regression problem

Final Training E_{RMS}	2.95414
Final Test E_{RMS}	3.21589

b) Learning curve

The training loss was plotted (Figure 1) and to check for overfitting, the test loss during every 100 epochs was also plotted (Figure 2).

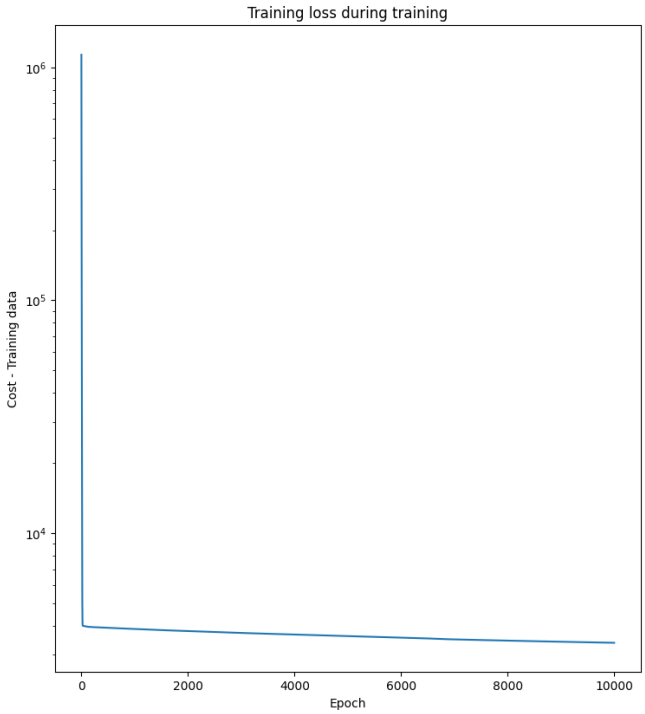


Figure 1 Training loss for the regression problem

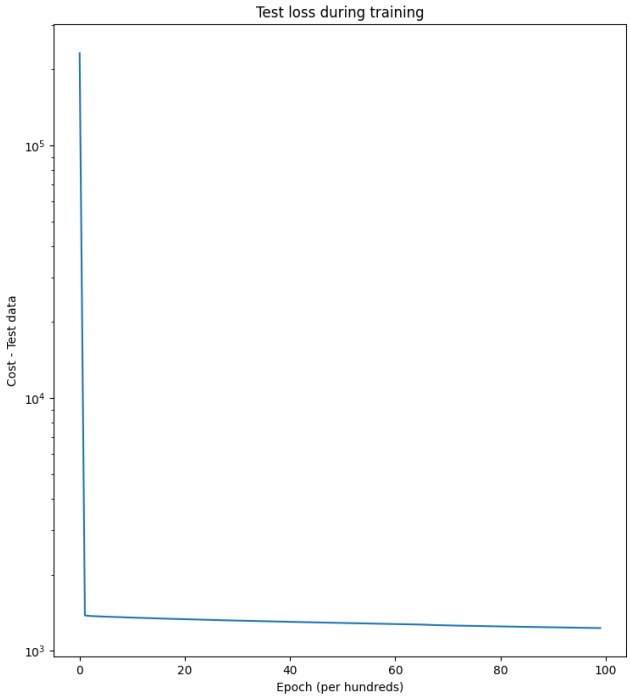


Figure 2 Test loss for the regression problem

c) Regression result with training labels

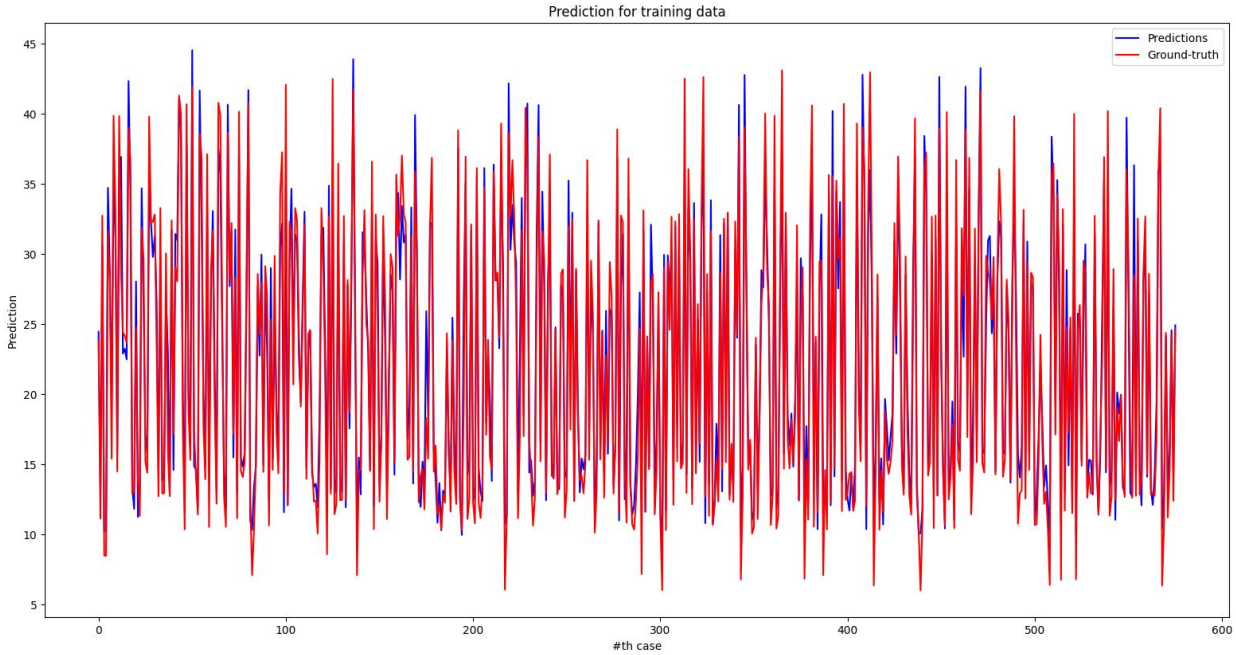


Figure 3 Regression result with training labels

d) Regression result with test labels

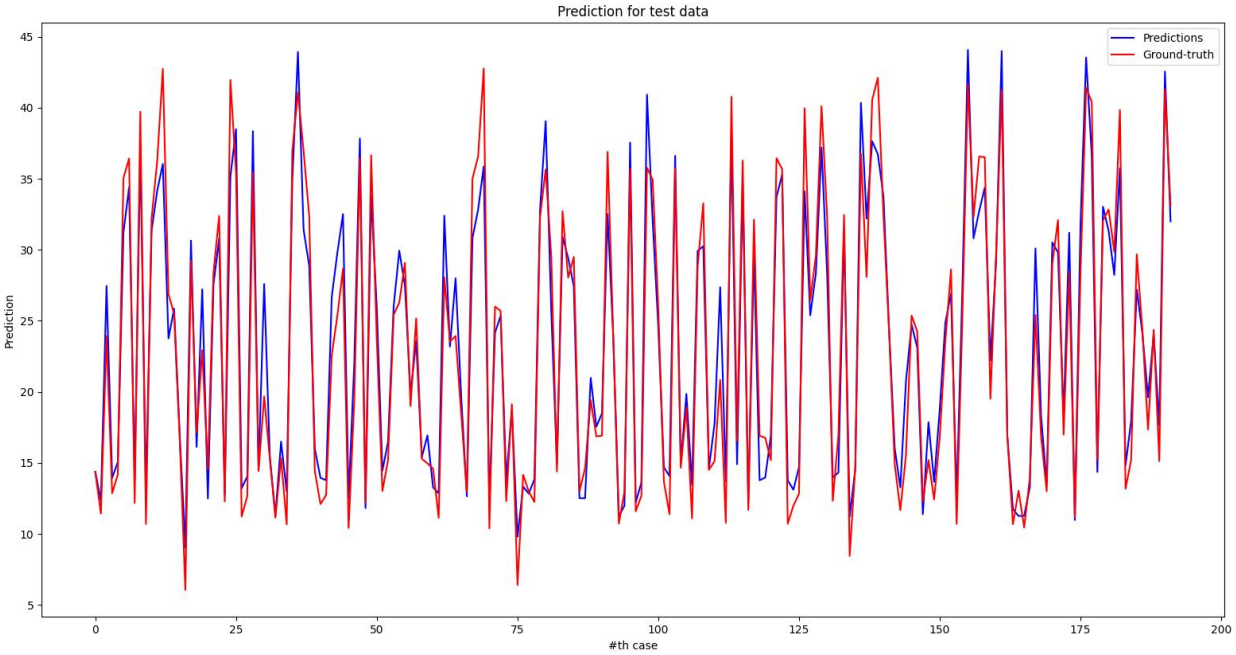


Figure 4 Regression result with test labels

e) Feature selection

The feature selection process was implemented by first removing individual features and comparing the RMS for the training and test data. This list of values was sorted starting with the features that when removed resulted in the best RMS for the test data. The results are seen in Table 3.

Table 3 Individual feature removal RMS Error results

RMS training	RMS test	# of the feature removed #
2.453120631	2.853143638	4
2.461525557	2.868791203	5
2.5567727	2.905372179	16
2.596346349	2.969884947	3
2.626535424	2.99439971	2
2.6191539	2.996292254	0
2.634491104	3.021738745	15
2.545601436	3.032456934	9
2.541257928	3.041581459	14
2.695192927	3.051978265	1
2.586997335	3.171383277	8
2.683461648	3.180242372	10
2.646531732	3.200789551	12
2.678193177	3.205612927	11
2.767389137	3.218071337	13
2.841630623	3.348428401	7
5.540765461	6.545384195	6

With this table, a feature selection process and training were carried out, first training the data with only the 10 most important features, and then eliminating more and more features until the RMS was not good anymore. The result of this process can be seen in Table 4.

Table 4 Several features removal RMS Error results

RMS training	RMS test	Features removed
2.95414	3.21589	All features
2.67494	3.06980	4, 5, 16, 3, 2, 0, 15
2.67893	2.95527	4, 5, 16, 3, 2, 0, 15, 9
2.21996	2.62049	4, 5, 16, 3, 2, 0, 15, 9, 14
3.11332	3.37432	4, 5, 16, 3, 2, 0, 15, 9, 14, 1

The combination that gave the best results was when the feature vector was missing the features number 4, 5, 16, 3, 2, 0, 15, 9 and 14, with an RMS for training data of 2.21996 and an RMS for test data of 2.62049, an improvement of 18%.

The feature selection process works because it eliminates the features that are the most irrelevant to the overall result of the neural network, allowing it to converge faster and give a better result with less noise in the input. Without extra information, the algorithm does not need to learn that those inputs are not important for the desired result.

Classification

a) Neural network

The neural network implemented for the classification problem consists of the input layer with 34 features, two hidden layers of 3 and 2 neurons respectively and the output layer with a single neuron. The activation function is the ReLU for the hidden layers, and the sigmoid function for the output layer. All the hyperparameters used can be seen in Table 5.

Table 5 Configuration for neural network for the classification problem

Network architecture	34,3,2,1
Learning rate	0.01
Epochs	20,000
Activation function	ReLU
Output activation function	Sigmoid
Cost function	Cross entropy

b) Learning curve

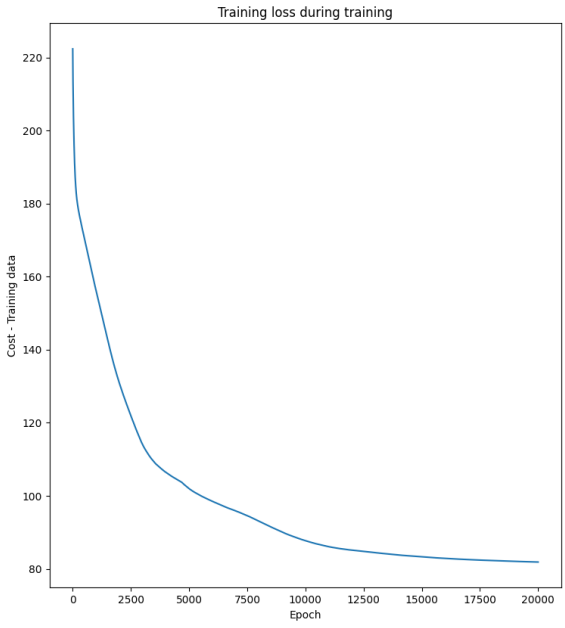


Figure 5 Training loss for the classification problem

c) Training error rate

Table 6 Training error rate

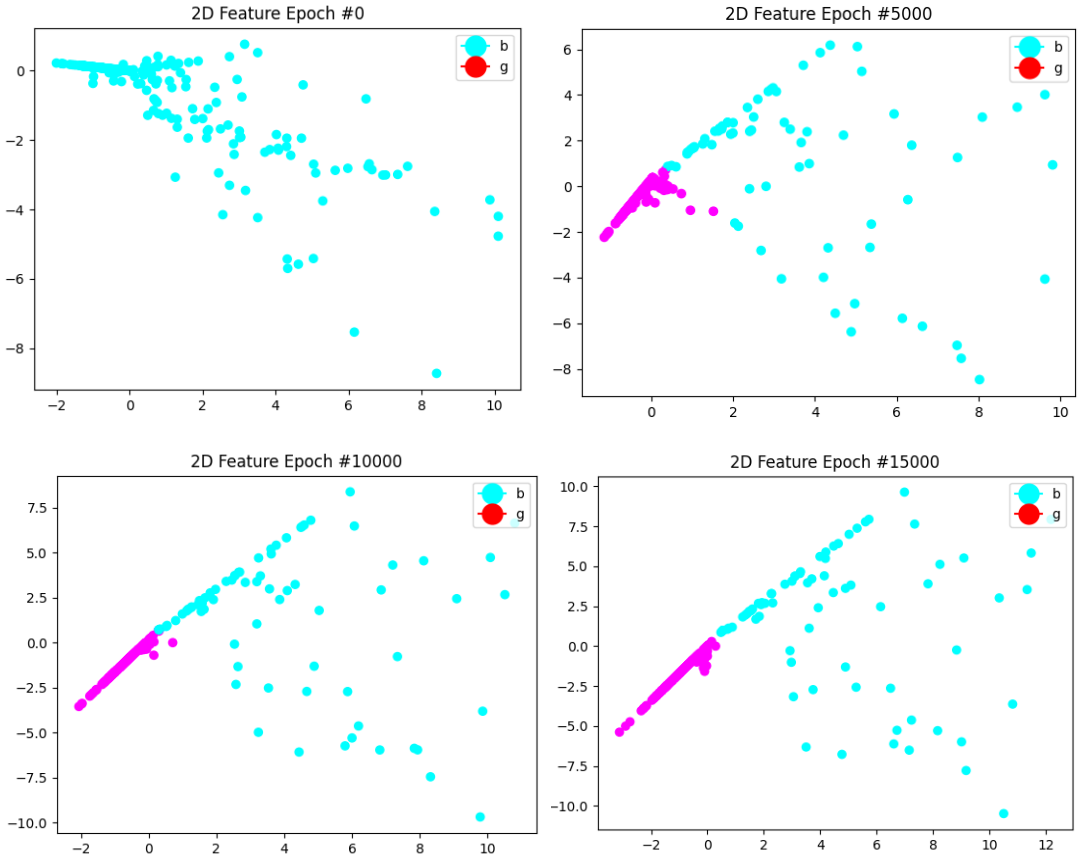
Total cases	280
Correct predictions	252
False positives	27
Missed positives	1
Correct rate	90%

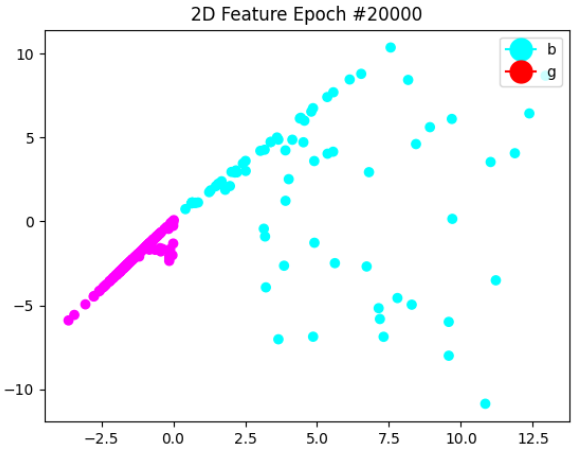
d) Test error rate

Table 7 Test error rate

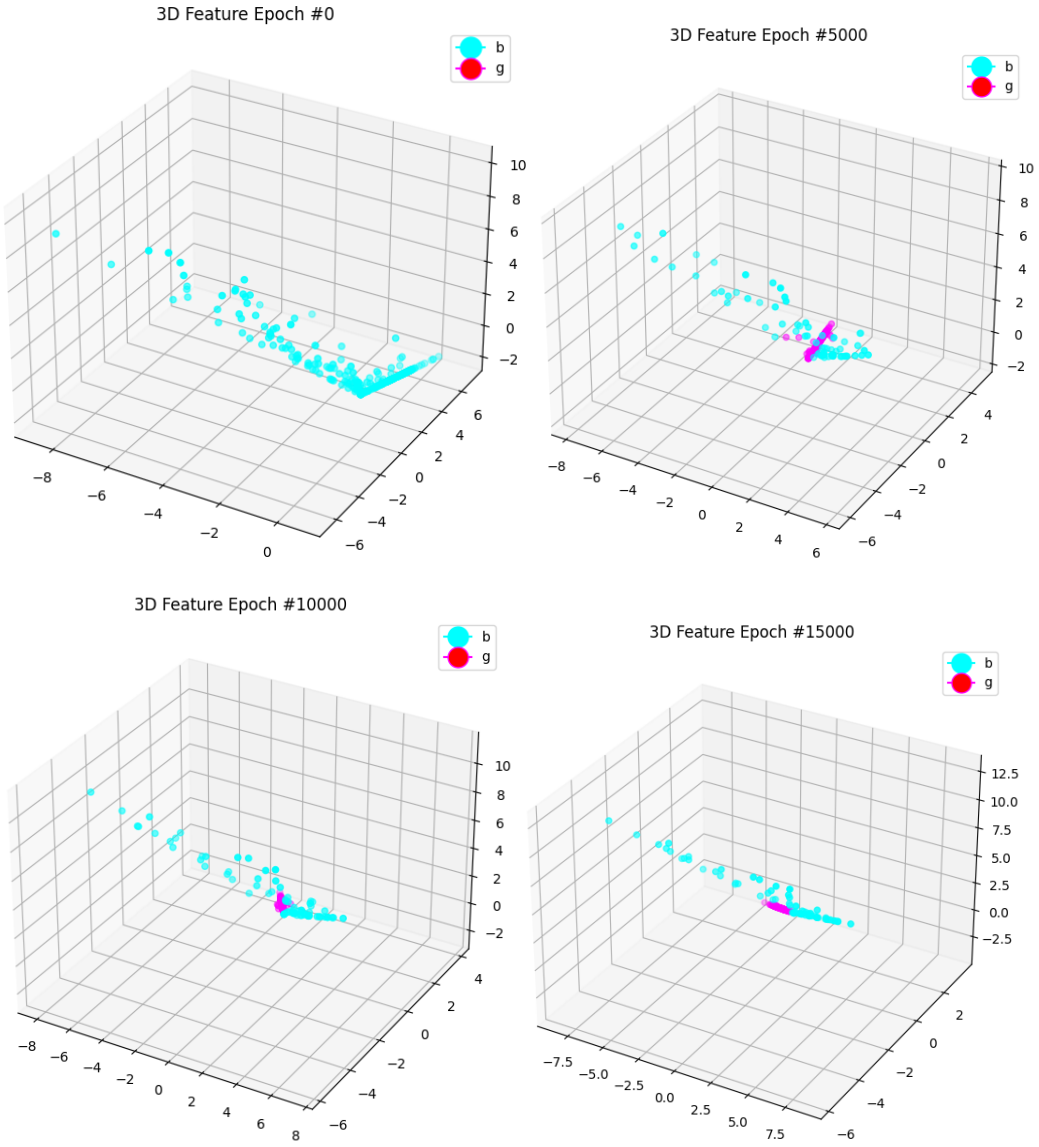
Total cases	70
Correct predictions	65
False positives	3
Missed positives	2
Correct rate	93%

e) Comparisons for a layer of 2 neurons in the layer before the output layer.

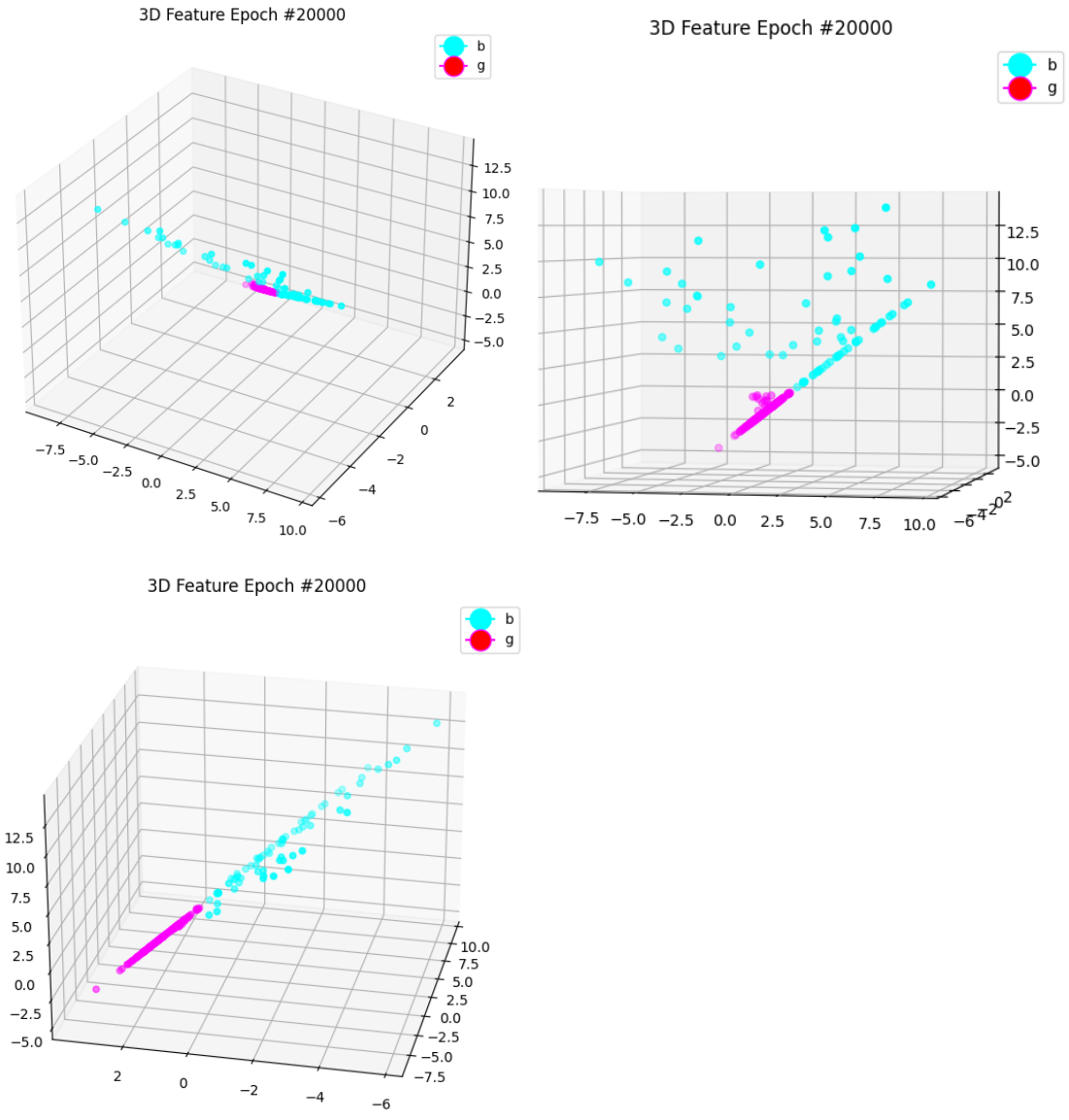




f) Comparisons for a layer of 3 neurons in the layer before the output layer.



For easier display, the last graph is presented in different views. One can appreciate that from a 2D perspective, the final graph resembles that of the model that only had 2 neurons in the last hidden layer.



Main References

- Andrew Ng, *Neural Networks and Deep Learning* [MOOC]. Coursera. https://www.coursera.org/learn/neural-networks-deep-learning/
- Andrew Ng. *Machine Learning* [MOOC]. https://www.coursera.org/learn/machine-learning

- Peter Roelants, *How to implement a neural network (1/5) - gradient descent*, 2015. https://peterroelants.github.io/posts/neural-network-implementation-part01/ (Accessed November 2021).
- Matt Mazur, *A Step by Step Backpropagation Example*, 2015. https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/. (Accessed November 2021).
- Piotr Skalski, *Let's code a Neural Network in plain NumPy*, 2018. https://towardsdatascience.com/lets-code-a-neural-network-in-plain-numpy-ae7e74410795. (Accessed November 2021).
- Zhuochen Wu, *Feature Selection in Convolutional Neural Network with MNIST Handwritten Digits*, 2018. http://users.cecs.anu.edu.au/~Tom.Gedeon/conf/ABCs2018/paper/ABCs2018_paper_156.pdf (Accessed November 2021).
  Arseny Turin, Gradient Descent From Scratch, 2020. https://towardsdatascience.com/gradient-descent-from-scratch-e8b75fa986cc . (Accessed November 2021).