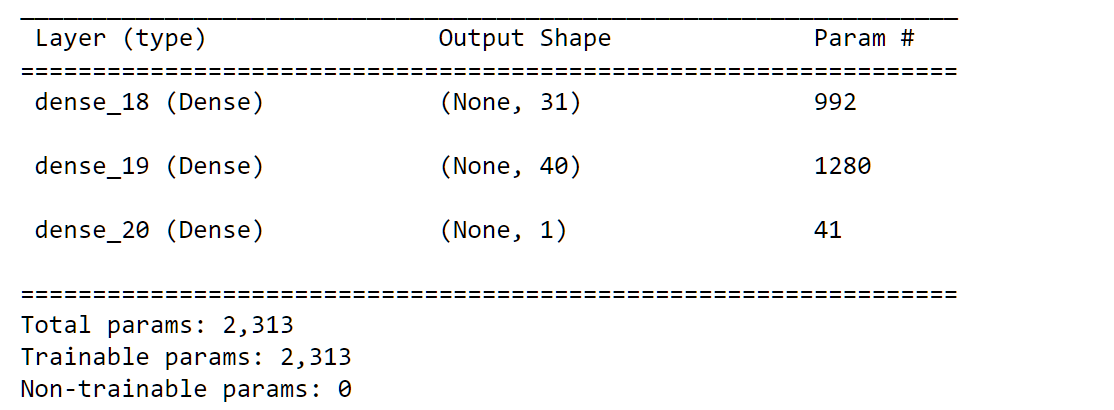
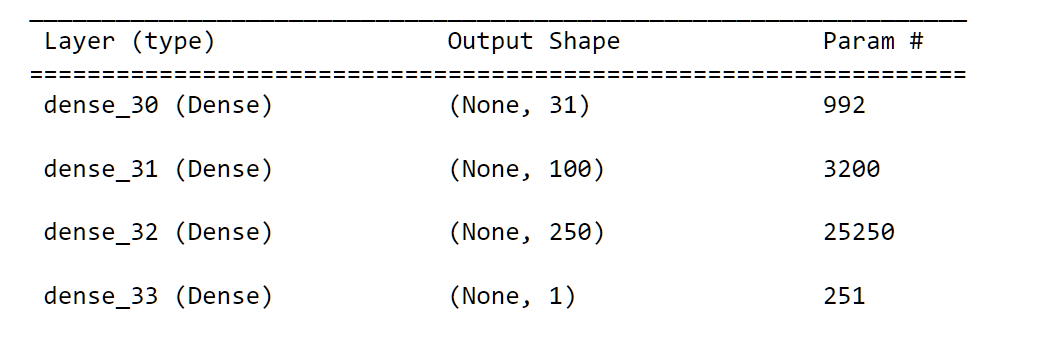
Phuong Dinh

**Task 3**

* Starting with the two classification models using SVM:
  + The 1st SVM model I got an MAE score of 1.6895238095238096 and an accuracy score of 0.2547619047619048 using the rbf kernel from the 10k fold assessment.
  + The 2nd SVM model I got an MAE score of 1.706190476190476 and an accuracy score of 0.26 using the linear kernel from the 10k fold assessment.
* Following the two SVM models, I also created two classification models using NN:
  + The 1st NN model I got an MAE score of 2.3290476190476186 and an accuracy score of 0.24095238095238095 using the following NN model:
    - In this model, I used relu for the first two layers and tanh for the output layer. The loss function is Mean Absolute Error while the optimizer is adam.
  + The 2nd NN model I also got an MAE score of 2.3290476190476186 and an accuracy score of 0.24095238095238095 using a different NN model but with the same loss and optimizer:



* + - In this model, I used sigmoid for the first three layers and tanh for the output layer.
* From the four classification experiments I have done, it is clear that the SVM model outperformed the NN model with an average decrease of 0.6 in MAE score. Meanwhile the accuracy stays low for all four training models with all of them below 30% accuracy. I suspect the difference is due to the low amount of data given by the dataset. Since SVM can effectively use only a subset of a dataset as training data, for well-separated classes, the number of observations required to train an SVM isn’t high. On the other hand, NN would be more robust for larger dataset when it is not possible to draw boundaries for SVM. Furthermore, for the NN model, I specifically used the Mean Absolute Error loss which is considered a loss function for linear regression models. However, the dataset by nature leans toward a classification problem. Thus, this could have negatively affected the MAE score for the NN as it is not meant to process this kind of dataset.
* In this task, I utilized the Grid search method to identify the best parameters for both SVC and NN to minimize MAE. Both grid search methods are included in the source code for SVC and NN; however, they are commented out from the main run since it is computationally expensive. For SVC, I created a grid search to test between different kernels: linear, rbf, poly, sigmoid and different C ranging from 1 to 100. At the end of grid search, I concluded that the best MAE score I could achieve is 1.2742857142857145 using kernel rbf and C = 8.
* Table

  Description automatically generatedFor NN, I tested between different activation, optimizers, epochs, and batches to minimize the MAE score. I first declare a new neural network with the Sparse Categorical Cross Entropy loss function with a place holder optimizer and two place holder activation functions. The grid search will test out each combination of optimizer, activation function, batch size, and epochs to check the lowest possible MAE score. After search, the lowest possible MAE score I could achieve is 1.5957142857142856 using the following NN model:
  + - In this model, I used relu for the first two layers and sigmoid for the output layer. The batch size is 100, the optimizer is adam, and the epochs are 200.
    - The new result is significantly lower compared to the original two neural networks.
* All in all, there are three noteworthy findings in task 3. Firstly, for smaller dataset, it is more beneficial to run SVM instead of using NN. This will save on processing time while providing better result. Secondly, Using NN Mean Absolute Error is not preferable for a classification model. I believe that using a loss function such as sparse categorical cross entropy would have yield a better MAE score for this problem. Thirdly, through doing grid search on both my SVC implementation and NN implementation, I saw that some parameters have much higher effects on minimizing the overall MAE score. For SVC changing kernel and C values seem to yield lower MAE score rather than changing the gamma. For NN it is not possible to change the loss as it is tied to the output layer, but changing the epochs and optimizers yield better result overall.