Brian Silfer

Module 21 Challenge

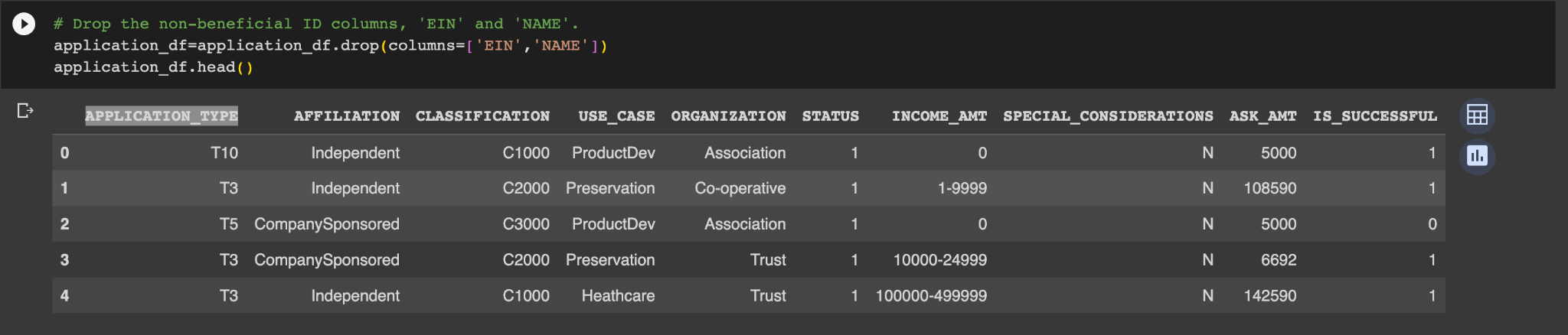
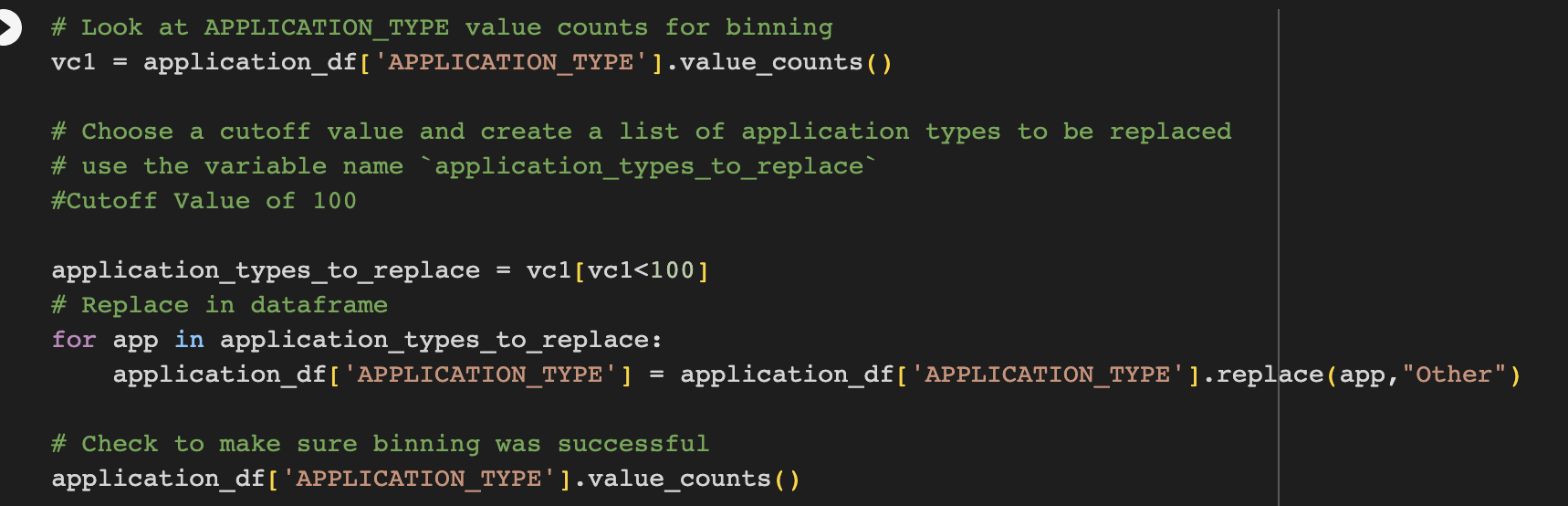
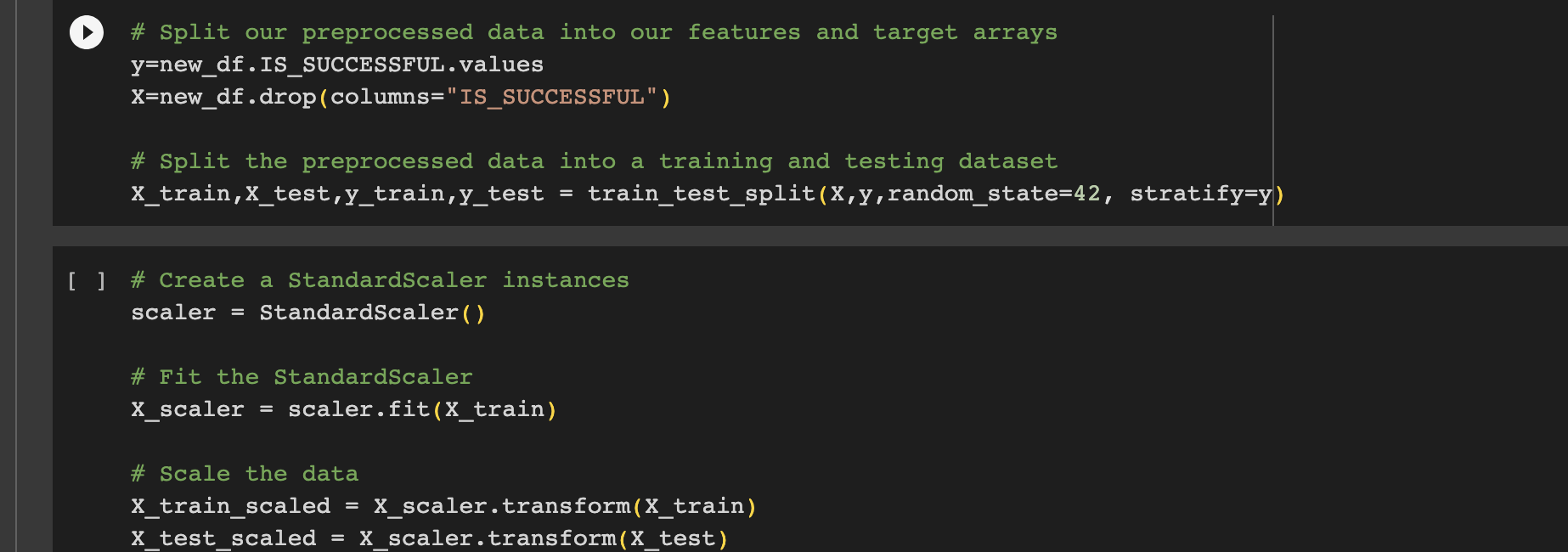
Neural Net Analysis

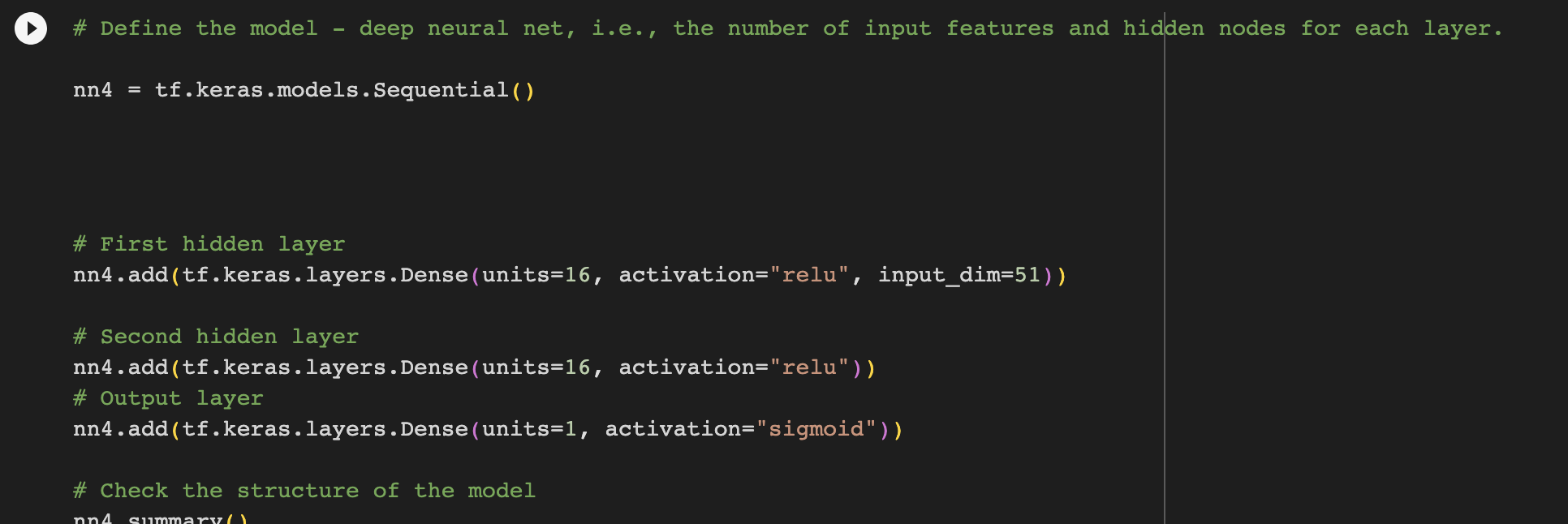
Overview:

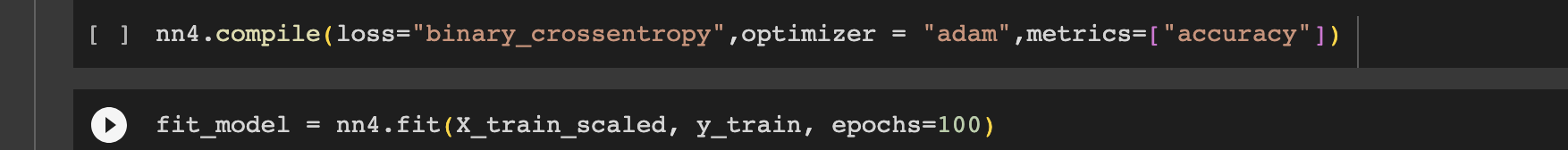
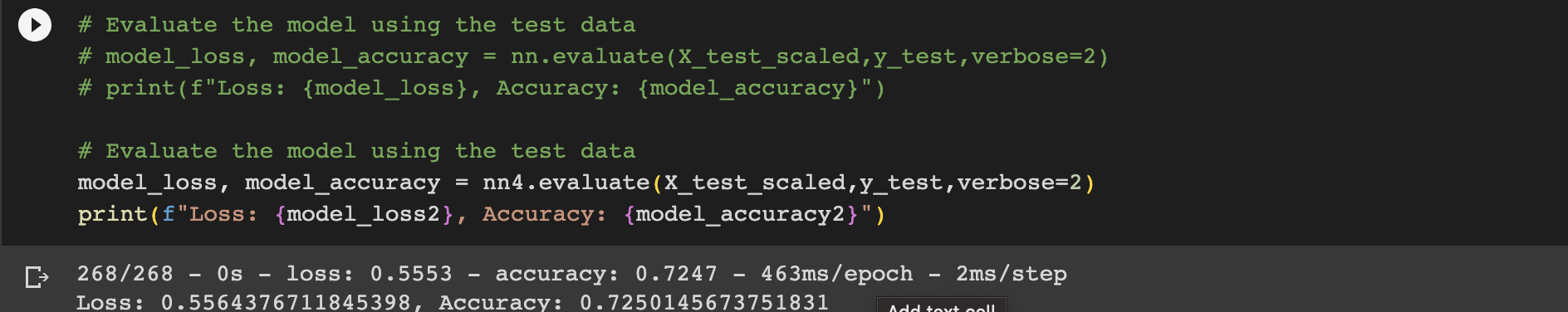
The purpose of this analysis was to take a dataset, and create a neural network that would predict a binary outcome of whether or not someone would be successful if funded by Alphabet Soup. Before compiling, training, and evaluating our model, we first preprocessed the data to remove unnecessary columns, categorizing application types and classifications into groups and then counting how many times each variable appears in the data to group insignificant output into its own group. Lastly, we converted the categorical variables into numerics so they can be read by the neural network model. We then scaled the data using StandardScaler to put all of our numerical data on a scaled playing field making it easier to read by the model. Our model was taken through 3 steps to attempt to optimize it to predict success with an accuracy score of 75%.

Results:

Data Preprocessing:

* + The target variable in this model is IS\_Successful
  + The feature variables are: Application\_type, Affiliation, Classification,Use\_case, Organization, Status, income\_amt, special\_considerations, and ask\_amt.
  + The variables EIN and Name were removed from the dataset as they were purely identifier variables and held no predictive power in this model.
  + We also processed the Classification and Application\_Type column to reduce the types and classifications with only a few rows. Doing this makes the conversion of categorical variables into numerics easier for the model to interpret as there will be less categories to interpret and more meaningful data present.
  + Lastly, we scaled our data and converted categorical data into numerics so the neural network model can interpret it.

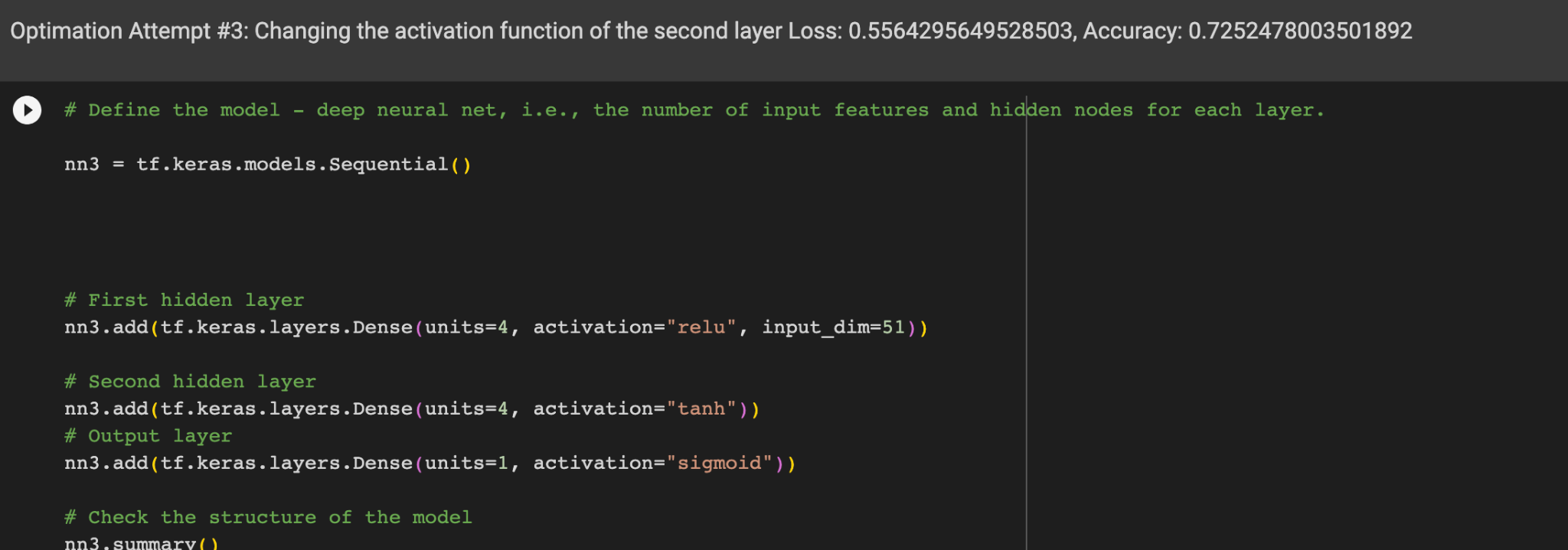
Compiling, Training, and Evaluating the Model (Attempt 1)

* + I selected the activation function, number of units, and number of layers by testing out which number of units and layers yielded the most accurate output. I chose Relu as the activation function as it is most common and changing it to tanh had a worsening effect on the models accuracy.
  + I was not able to achieve the target performance of 75% accuracy as my models hovered around 73%.
  + In order to increase the models performance, I chose to change the original model by changing the amount of units of the model, the type of activation functions used, and by adding a third layer to the model. I also tested the original model under conditions of 500 epochs instead of 100. These changes yielded small increases or decreases in the accuracy score of the model.

Model Optimizations Attempt 2



Model Optimization Attempt 3



Overall Results:

The overall results of the model were that the model yielded about a 73% accuracy rate in predicting whether or not someone being funded by AlphabetSoup would be successful or not. I was not able to find a combination of model factors that would yield the 75% accuracy score, but I learned that changing certain elements such as doubling the number of epochs can have little to no impact on the accuracy rating. I fit the model on 100, 200 and 500 epochs and the most the model accuracy increased was by a few tenths of a percentage point from 0.721 to .725. It's possible that an even higher number of epochs would have increased the model accuracy further, but it seemed to be less effective as an optimization method then I previously thought. Other methods such as changing the second hidden layer’s activation function from Relu to tanh had little effect on the accuracy. The same effect occurred when I changed the amount of inputs to 8 versus 16. It would seem based on this challenge as well as our in class examples that understanding the combinations of activation functions and what they do is the best way to optimize a model, and not plug and play with inputs, layers, and epoch numbers. If I could choose a different model to predict success rate, I would pick a regression model, as it would be more straightforward and rely less on mixing and matching model factors and more on pre-processing the data to eliminate errors and deficiencies in the data. Using neural networks requires a deep understandings of activation functions and regression models are simpler to utilize.