The purpose of this analysis is to find a machine learning model that will predict with the highest accuracy which charitable campaigns for the foundation Alphabet Soup are the most likely to be successful, based on previous campaigns. As with the previous challenge, using this kind of model for approving future campaigns can help reduce human bias by looking at only objective facts about the likelihood of success, although maybe likelihood of success alone might not be the most charitable parameter for deciding whether to support a charitable campaign.

* Target variables:
  + ‘IS\_SUCCESSFUL’, a column with binary values explaining whether a charitable campaign succeeded in its goals or not
* Feature variables:
  + ‘APPLICATION\_TYPE’, presumably the way the organization applied for their campaign
  + ‘AFFILIATION’, the kind of entity backing the campaign
  + ‘CLASSIFICATION’, codes for type of organization
  + ‘USE\_CASE’, what the organization needs the money for
  + ‘ORGANIZATION’, the type of organization
  + ‘STATUS’, another column with binary values indicating if the campaign is active
  + ‘INCOME\_AMT’, the amount of money the organization usually brings in
  + ‘SPECIAL\_CONSIDERATIONS’, a binary campaign indicating whether or not the application was processed with any additional supporting data
  + ‘ASK\_AMT’, the amount of money requested
* Removed variables:
  + ‘EIN’ as a code method for identifying organizations was neither the target nor something that should affect success
  + ‘NAME’ was also just an identifier for organizations that should not affect success
* Elements of final neural network model:
  + Neurons: I ended up going with 30 neurons in each of the hidden layers, as that showed an improvement in the accuracy, but not enough that it seemed worth it to keep upping the number
  + Layers: Increasing the number of layers from two to three decreased the accuracy, so I left it at two
  + Activation functions: ‘leaky\_relu’ showed a slight improvement over ‘relu,’ while ‘tanh’ showed a reduction in accuracy, so I went with the model utilizing ‘leaky\_relu’
* I was not able to achieve the target model performance
* I tried substituting activation functions, changing the number of neurons, and changing the number of layers

Overall, the accuracy of my deep learning model plateaued at 73%. This exercise increased my appreciation for Keras Tuner, because other than trying everything and stumbling across the correct answer accidentally, I don’t know how else one would go about finding the best model for a particular dataset. I did not find the best model for this dataset. I would recommend the use of Keras Tuner in a real-world application, as it can try different things and find the best result with a much greater efficiency and efficacy than myself and my 9-year-old laptop can.