**Overview**:

The deep-learning-challenge project aims to develop a supervised machine learning model using deep learning neural networks to classify applicants for a funding scheme into those likely to succeed (i.e., 1) or fail (i.e., 0) in their ventures. The goal is to assist decision-makers within the Alphabet Soup Charity in identifying applicants with the highest chances of success, thus improving the allocation of resources and maximizing the impact of their funding efforts. The project involves preprocessing the provided data, building an initial neural network model, optimizing the model using hyperparameter tuning, and evaluating its performance to ensure its efficacy.

**Optimisation Methodology:**

Data preprocessing optimisation

1. The initial step was to find if any variables columns could be excluded from the model to improve accuracy.
   1. Drop columns from the application data frame.
   2. Run model using initial parameters *(See Appendix Figure 1)* to see if accuracy improved from initial test with all parameters included.
   3. Noted any parameters that increased accuracy of the model *(See appendix Figure 2)*.
2. Step two was implemented to find the ideal cutoff value for the application types and classification types. i.e. any classification types with a frequency that did not occur over a certain amount of times (threshold) was grouped together in the model and classified as ‘other’.
   1. Change the classification threshold for both application types and classification types and observe any increase in accuracy *(See appendix Figure 3)*.

Keras-Tuner Model Optimisation

The model was calibrated to optimal level from the two above steps. Then using Keras Tuner search through the different combinations of; number of neurons in each hidden layer; the activation function to be used in each hidden layer and the learning rate.

* 1. **Define Hyperparameter Search Space**: Define the ranges of hyperparameters to search through:
     1. **Function build\_model**: Defines the neural network model with hyperparameters.
     2. Units- Number of neurons in each hidden layer, ranging from 5 to 200 in steps of 5.
     3. Activation functions for each hidden layer (relu, tanh, elu).
     4. learning\_rate: Learning rate for the optimizer (0.01, 0.001, 0.0001).
  2. **Build and Compile Model**: Use the hyperparameters to build and compile the model.
  3. **Set Up Tuner**: Set up the tuner with the model-building function, search space, and optimization objective.
     1. **Tuner Setup**: Uses RandomSearch to search through the hyperparameter space defined in build\_model.
     2. **Objective**: Optimizes for validation accuracy (val\_accuracy).
     3. **Executions**: Executes each trial 3 times to account for variability (executions\_per\_trial=3)
     4. **Directory**: Saves the search results in the my\_dir/intro\_to\_kt directory.
  4. **Perform Hyperparameter Tuning**: Run the search on the training data.
     1. Searches for the best hyperparameters by training the model on the training data and validating it on the test data for 10 epochs.
  5. **Get Best Hyperparameters**: Retrieve and use the best hyperparameters to build the final model.
  6. **Evaluate Model**: Train and evaluate the model with the optimal hyperparameters.

The number of hidden layers is not optimised be the Keras-Tuner, therefore the tuner was run with numerous iterations of:

Different minimum and maximum values for the neurons within each hidden layer;

A different number of Hidden layers.

1. **Results**:

Data Preprocessing

* + The model target variable is the ‘IS\_SUCCESSFUL’ column. This variable indicates whether a charity donation request was successful (1) or not (0).
  + The features of the model are all columns from the data frame after dropping EIN', 'NAME', 'SPECIAL\_CONSIDERATIONS’ and IS\_SUCCESSFUL. This includes; Categorical variables converted to numeric through one-hot encoding and; Numerical variables (INCOME\_AMT, ASK\_AMT).
  + Model was found to increase accuracy when ‘SPECIAL\_CONSIDERATIONS’ column is dropped from the Data frame *(See Appendix Figure 2).*
  + It can be seen that the model accuracy is not impacted significantly by changing the classification bins (See Appendix Figure 3).
  + The greatest accuracy was found by introducing an application type cutoff threshold of 600 and a classification type cutoff threshold of 1000.

Compiling, Training, and Evaluating the Model

**Initial Model**:

* + Configuration: Two hidden layers (80 neurons with ReLU, 30 neurons with ReLU), output layer with sigmoid activation.
  + Accuracy: 73%.

**Optimized Model**:

After trying 1 – 6 hidden layers and minimum value of 1 neuron up to 200 neurons for each layer Keras-Tuner optmised the model to an Accuracy of ## % with the following configuration:

* + Six hidden layers with reduced complexity (neurons between 5-9, sigmoid activation for all layers), output layer with sigmoid activation.
  + Accuracy: 73%.

**Recommendations**

Despite efforts to optimize the neural network, the model's accuracy plateaued at 73%. This suggests that the current deep learning model may not be capturing the complexities of the data adequately. To potentially improve classification accuracy, a more complex model such as a Random Forest classifier is recommended.

**Reasons for Current Model Limitations:**

* + **Model Complexity**: The neural network may not be complex enough to capture intricate patterns in the data, despite optimization.
  + **Overfitting**: With increased layers and neurons, the model might be overfitting the training data, leading to poor generalization on test data.
  + **Feature Engineering**: The current feature set might lack important information or interactions that a more advanced model like Random Forest could capture.

**Proposed Model: Random Forest**

A Random Forest classifier could be more effective due to its ability to handle many input features and interactions between them. It can also provide insights into feature importance, helping to identify which variables contribute most to predicting funding success. This model might offer better accuracy and robustness, making it a suitable alternative to the current deep learning approach.

**APPENDIX**

**A close up of a list

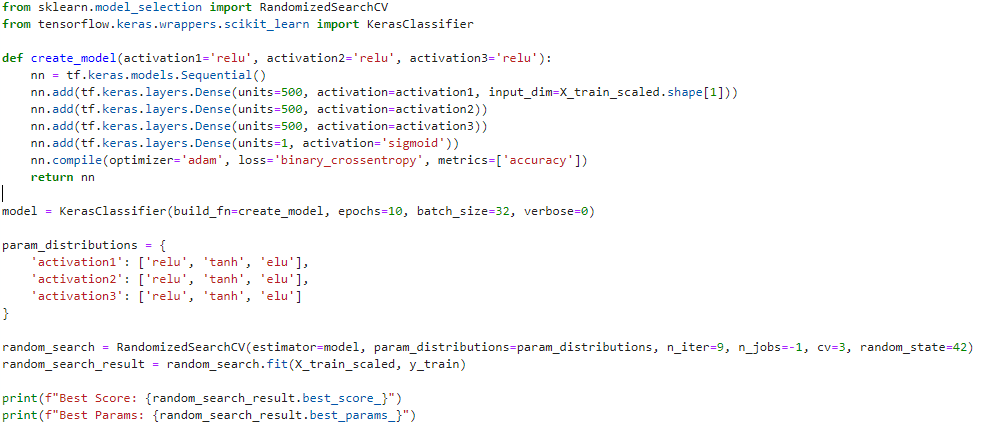
Description automatically generated**

**Figure 2**

**A screenshot of a spreadsheet

Description automatically generated**

**Figure 3**

****

**Figure 4**