**Alphabet Soup Deep Learning Model Report**

The purpose of this analysis was to develop a neural network capable of predicting whether nonprofit organizations would be successful in securing funding from Alphabet Soup. The analysis involved preprocessing the dataset, building a baseline model, and running multiple optimization attempts to improve accuracy.

*\*\* See screenshots of model layers at the end \*\**

**Data Preprocessing**

* The target variable for the model was IS\_SUCCESSFUL.
* All other columns, excluding identifiers, were used as features.
* EIN and NAME were dropped since they don’t contribute to prediction and are irrelevant to model performance.
* Rare categorical values in APPLICATION\_TYPE and CLASSIFICATION were grouped under “Other” to reduce dimensionality.

**Model Architecture and Training**

***Original Model***

* Layers: Dense(80), Dense(30), Dense(1)
* Activation: relu (hidden), sigmoid (output)
* Accuracy: 68.6%

***Optimization Attempt 1***

* Layers: Dense(80), Dense(30), Dense(10), Dense(1)
* Accuracy: 73.3%
* This attempt added one additional hidden layer and showed the most improvement.

***Optimization Attempt 2***

* Layers: Dense(128), Dense(64), Dense(1)
* Accuracy: 62.6%
* Although the model had more neurons, the performance dropped due to overfitting.

***Optimization Attempt 3***

* Layers: Dense(80), Dense(30), Dense(1)
* Accuracy: 68.6%
* This model matched the original and confirmed no performance gain from retrying the same structure.

**Model Settings**

* Activation functions: relu (hidden layers), sigmoid (output layer)
* Loss function: binary\_crossentropy
* Optimizer: adam
* Epochs: 100
* Scaler: StandardScaler

**Optimization Strategy:** I tested three different architectures to improve model performance. I adjusted neuron counts, added a third hidden layer in one version, and tried increasing the overall depth and width of the network. Only the model with an added third hidden layer produced a noticeable increase in performance.

**Summary**: The best-performing model reached 73.3% accuracy, just short of the 75% target. Although the model did improve, it did not exceed the benchmark. If I had more time, I would explore different algorithms altogether, such as Random Forest or XGBoost, which are often more effective for structured, tabular datasets like this one. These alternatives may offer better interpretability and less sensitivity to tuning compared to neural networks.

**Original Model**

A white sheet with black text and blue and green text

AI-generated content may be incorrect.

**Optimization Model 1**

A white sheet with text and numbers

AI-generated content may be incorrect.

**Optimization Model 2**

A screenshot of a computer

AI-generated content may be incorrect.

**Optimization Model 3**

**A white sheet with text and numbers

AI-generated content may be incorrect.**