**Neural Network Model Report for Alphabet Soup**

**1. Overview of the Analysis**

The purpose of this analysis is to build a deep learning model that predicts the success of funding applications for Alphabet Soup, a charity organization. By analyzing past funding data, the goal is to create a classification model that can help the organization identify which applications are likely to be successful. The target accuracy is to achieve a performance of over 75%.

**2. Results**

**Data Preprocessing**

* **Target variable**:
  + The target variable is IS\_SUCCESSFUL, which indicates whether the funding application was successful or not (1 = successful, 0 = unsuccessful).
* **Features**:
  + The features used in the model include categorical variables that were one-hot encoded, such as:
    - APPLICATION\_TYPE (e.g., T3, T4, T5)
    - CLASSIFICATION (e.g., C1000, C2000)
    - AFFILIATION (e.g., Independent, CompanySponsored)
    - USE\_CASE (e.g., Preservation, ProductDev)
    - ORGANIZATION, STATUS, INCOME\_AMT, and SPECIAL\_CONSIDERATIONS.
  + Numerical features, such as ASK\_AMT, were also included after being scaled.
* **Removed Variables**:
  + The EIN and NAME columns were removed from the input data as they are identifiers that do not contribute meaningfully to the model's prediction.
  + These variables were dropped to ensure that only relevant features contributing to the prediction remained.

**Compiling, Training, and Evaluating the Model**

* **Model Architecture**:
  + The model was a deep neural network with:
    - **Input Layer**: The number of neurons in the input layer matched the number of features in the dataset after preprocessing (e.g., 118 features).
    - **Hidden Layers**:
      * **First hidden layer**: 128 neurons with relu activation function.
      * **Second hidden layer**: 64 neurons with relu activation function.
      * **Third hidden layer**: 32 neurons with relu activation function.
    - **Output Layer**: 1 neuron with sigmoid activation function (as this is a binary classification problem).

The choice of neurons and activation functions (relu) was made to allow the model to capture complex relationships in the data. relu helps avoid the vanishing gradient problem, ensuring faster training and better convergence.

* **Model Performance**:
  + **Initial Accuracy**: The initial accuracy of the model was approximately 72.5%.
  + **Target Accuracy**: The target accuracy was set to 75%, but the initial attempt did not meet this target.
* **Optimization Steps**: To improve the model performance, the following optimization techniques were applied:
  + **Increased number of neurons** in the hidden layers to improve the model's ability to learn complex patterns.
  + **Added a third hidden layer** to increase model depth and capture more intricate data relationships.
  + **Implemented Dropout** to prevent overfitting. This helped the model generalize better to unseen data.
  + **Batch Normalization** was applied to stabilize the learning process, reduce internal covariate shift, and improve performance.
  + **L2 regularization** was added to reduce overfitting and control model complexity.
  + **Increased the number of epochs** to give the model more training iterations, allowing it to learn better representations.

Despite these efforts, the model’s performance improved marginally, but it remained around 72.5%.

**Model Summary**

* **Neurons**: The model utilized 128, 64, and 32 neurons in its three hidden layers, respectively.
* **Layers**: Three hidden layers were selected to increase the model’s learning capacity.
* **Activation Functions**: relu was used in all hidden layers, and sigmoid in the output layer.

**3. Summary**

The deep learning model created for Alphabet Soup achieved an accuracy of 72.5%, which was slightly below the target accuracy of 75%. While various optimization techniques such as adding more neurons, using dropout, regularization, and batch normalization were applied, the model’s accuracy plateaued at 72.5%.

**Recommendation for Future Improvements:**

Given the structure of the data, other models like **Random Forests** or **Gradient Boosting Machines** (such as **XGBoost**) may provide better results. These models often perform well with tabular data, which includes both numerical and categorical variables. Random Forests, in particular, are robust to overfitting and can provide feature importance, which might help identify the most significant factors contributing to successful applications.

Thus, it is recommended to experiment with ensemble models like **Random Forests** or **XGBoost** for this classification problem to potentially improve accuracy and feature understanding.