Report on the Neural Network Model for Alphabet Soup

Overview of the Analysis

The purpose of this analysis is to build and evaluate a deep learning model to predict whether applicants for funding from Alphabet Soup will be successful. The dataset includes a variety of features that are used to train the model to identify patterns that correlate with successful outcomes. This report will detail the steps taken to preprocess the data, the architecture of the neural network model, the results obtained, and potential improvements for future models.

Results

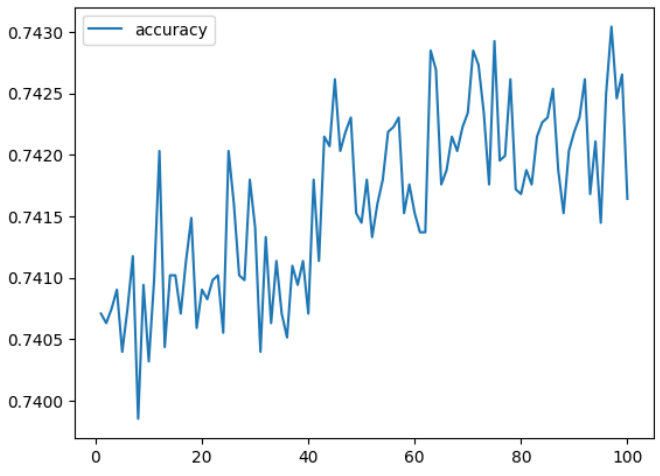
Data Preprocessing

- **Target Variable(s):**
  - The target variable for the model is IS_SUCCESSFUL, which indicates whether the funding applicant was successful (1) or not (0).
- **Feature Variable(s):**
  - The features for the model include:
    - **APPLICATION_TYPE** (17 categories, binned): The type of application submitted by the applicant. Infrequent application types were grouped into an "Other" category to reduce noise.
    - **AFFILIATION** (6 categories): The applicant's affiliation.
    - **CLASSIFICATION** (71 categories, binned): The classification of the application. Infrequent classifications were grouped into an "Other" category.
    - **USE_CASE** (5 categories): The intended use of the funding.
    - **ORGANIZATION** (4 categories): The type of organization applying for funding.
    - **STATUS** (2 categories): The current status of the application.
    - **INCOME_AMT** (9 categories): The amount of income reported by the applicant.

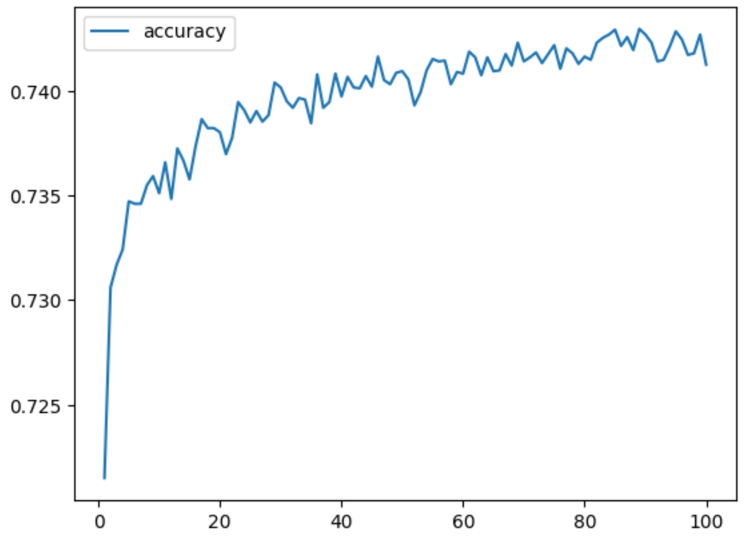
        - **SPECIAL_CONSIDERATIONS** (2 categories): Indicates whether special considerations were requested.
        - **ASK_AMT** (Continuous variable): The amount of funding requested by the applicant.
- **Variables to Remove:**
    - The EIN and NAME columns were dropped as they were non-beneficial to the model. These columns likely contained unique identifiers that did not contribute to the prediction of the target variable.
- **Binning and Encoding:**
    - **Binning:** Application types with fewer than 600 occurrences and classifications with fewer than 100 occurrences were grouped into "Other" categories to reduce noise and improve model performance.
    - **Encoding:** All categorical variables were converted to a numeric format using one-hot encoding (pd.get_dummies). This step ensured that the neural network could process the data efficiently.

Compiling, Training, and Evaluating the Model

- **Neurons, Layers, and Activation Functions:**
    - **Initial Model:**
        - **Layers and Neurons:**
            - First Hidden Layer: 80 neurons, ReLU activation
            - Second Hidden Layer: 30 neurons, ReLU activation
        - **Output Layer:** 1 neuron, Sigmoid activation
        - **Rationale:** ReLU was chosen for the hidden layers to handle non-linear relationships, while Sigmoid was selected for the output layer due to the binary nature of the classification problem.



- **Optimized Model:**
    - **Layers and Neurons:**
        - First Hidden Layer: 90 neurons, ReLU activation
        - Second Hidden Layer: 30 neurons, ReLU activation
        - Third Hidden Layer: 20 neurons, Tanh activation
    - **Output Layer:** 1 neuron, Sigmoid activation
    - **Rationale:** An additional layer with Tanh activation was introduced to capture more complex patterns and interactions between features, with the goal of improving model performance.



- **Model Performance:**
    - **Initial Model Results:**
        - **Loss:** 0.5751
        - **Accuracy:** 72.45%
    - **Optimized Model Results:**
        - **Loss:** 0.5531
        - **Accuracy:** 73.58%
    - **Rationale:** The addition of another hidden layer and tweaking the activation function to Tanh in the third layer slightly improved model

performance, suggesting that a more complex model can capture the intricacies of the dataset more effectively.

- **Steps to Increase Performance:**
    - Added a third hidden layer to increase the model's complexity.
    - Experimented with different activation functions (ReLU and Tanh) to find the most effective combination.
    - Adjusted the number of neurons in the hidden layers to optimize the balance between underfitting and overfitting.

Summary

The deep learning model developed for predicting funding success in the Alphabet Soup dataset showed promising results, with an accuracy of approximately 73.58% after optimization. While the optimized model outperformed the initial model, there is still room for improvement.

Recommendation:

For further improvement, a different model could be considered, such as a **Random Forest Classifier** or **Gradient Boosting Machine (GBM)**. These models are known for handling large feature spaces and capturing non-linear relationships effectively. Additionally, implementing **hyperparameter tuning** using techniques like **Grid Search** or **Random Search** could further refine model performance.