Neural Network Model Analysis for Alphabet Soup

Overview of the Analysis

The objective of this analysis was to develop a deep learning model to predict whether charitable donations would be successful (`IS_SUCCESSFUL` = 1) or not (`IS_SUCCESSFUL` = 0). Using a dataset containing organizational and funding-related information, the goal was to preprocess the data, design a neural network, and evaluate its performance. Multiple approaches were attempted to improve accuracy and minimize the loss, resulting in two distinct iterations of the model.

Results

1. Data Preprocessing

First Attempt:

- **Target Variable:**
  - The target variable was `IS_SUCCESSFUL`, which indicates whether a donation was successful.
- **Features:**
  - Features included:
    - `APPLICATION_TYPE`, `AFFILIATION`, `CLASSIFICATION`, `USE_CASE`, `ORGANIZATION`, `STATUS`, `INCOME_AMT`, `SPECIAL_CONSIDERATIONS`, and `ASK_AMT`.
    - Categorical variables were one-hot encoded.
  - `APPLICATION_TYPE` and `CLASSIFICATION` were grouped using a cutoff to reduce the number of unique categories.
- **Removed Variables:**
  - `EIN` and `NAME` were removed as they were non-beneficial identifiers.

Second Attempt:

- **Features and Target:**
  - Further feature selection was applied by removing `STATUS`, `SPECIAL_CONSIDERATIONS`, and `ASK_AMT`, in addition to `EIN` and `NAME`.
  - A new cutoff value was applied to `APPLICATION_TYPE` and `CLASSIFICATION` to replace infrequent categories with "Other."
- **Transformation:**
  - All numerical features were standardized using `StandardScaler` to improve model performance.

---

2. Compiling, Training, and Evaluating the Model

First Attempt:

- **Model Architecture:**
  - **Input Layer:** 43 features after one-hot encoding.
  - **Hidden Layers:**
    - First layer: 80 neurons, ReLU activation.
    - Second layer: 30 neurons, ReLU activation.
  - **Output Layer:** 1 neuron, sigmoid activation for binary classification.
  - Total Parameters: 5,981.
- **Training Results:**
  - The model was trained for 100 epochs with a batch size of 1.
  - **Training Accuracy:** ~73.6%.
  - **Validation Accuracy:** ~72.6%.
  - **Validation Loss:** 0.558.
- **Evaluation:**
  - Test Loss: 0.558.
  - Test Accuracy: 72.63%.
- **Challenges:**
  - The model plateaued in performance and did not achieve the desired accuracy of 75%.

Second Attempt:

- **Model Architecture:**
  - **Input Layer:** 30 features after revised preprocessing.
  - **Hidden Layers:**
    - First layer: 10 neurons, ReLU activation.
    - Second layer: 10 neurons, ReLU activation.
  - **Output Layer:** 1 neuron, sigmoid activation for binary classification.
  - Total Parameters: 4,611.
- **Training Results:**
  - The model was trained for 100 epochs with a validation split of 15%.
  - **Training Accuracy:** ~80.8%.
  - **Validation Accuracy:** ~80.4%.
  - **Validation Loss:** 0.457.
- **Evaluation:**
  - Test Loss: 0.473.
  - Test Accuracy: 78.86%.
- **Key Improvements:**
  - Reduced model complexity by reducing the number of neurons and features.
  - Applied stricter binning thresholds for categorical features.

---

3. Summary

Overall Results:

- **First Attempt:** The model achieved a test accuracy of 72.63% with a relatively high validation loss, indicating possible overfitting or suboptimal preprocessing.
- **Second Attempt:** The optimized model improved test accuracy to 78.86% with reduced loss, indicating a more generalized model.

Recommendations:

While the second attempt showed significant improvement, further optimization is possible. Alternative models could also be explored to improve performance:

1. **Random Forest or Gradient Boosting (e.g., XGBoost):**
    - These models are often more effective for tabular data and can handle categorical variables without requiring extensive one-hot encoding.
    - Feature importance analysis from tree-based models could offer valuable insights for feature selection.
2. **Logistic Regression with Feature Engineering:**
    - Simpler and interpretable, logistic regression can achieve similar performance with fewer computational resources.

Why Use These Models?

- Neural networks are not always the most suitable option for small-to-medium tabular datasets due to their complexity and sensitivity to hyperparameter tuning.
- Tree-based models are robust to noise and require less preprocessing, making them ideal for datasets with mixed data types.