Analysis of the Neural Network Framework for Alphabet Soup Funding Applications

Introduction

This report delves into the creation and assessment of a deep learning model using TensorFlow and Keras for predicting funding outcomes in Alphabet Soup applications. The neural network model aims to anticipate funding application success based on a variety of applicant attributes.

Data Preprocessing Target Variable(s)

The primary target variable indicates the success or failure of a funding application. Feature Variable(s): Applicant characteristics such as 'APPLICATION\_TYPE', 'AFFILIATION', 'CLASSIFICATION', 'USE\_CASE', 'ORGANIZATION', 'INCOME\_AMT', and 'SPECIAL\_CONSIDERATIONS' are included as features. Variable(s) to Remove: To optimize model performance, highly correlated or irrelevant variables were identified and eliminated. Columns with a correlation coefficient greater than 0.8 were removed, and initial variables 'EIN' and 'NAME' were also excluded.

Model Design and Training Neurons, Layers, and Activation Functions

The neural network architecture consists of an input layer, three hidden layers with 100, 50, and 30 neurons, respectively, and an output layer. ReLU activation functions are applied to the hidden layers, while a sigmoid activation function is used for the output layer. The chosen configuration balances model complexity and performance, considering the classification problem's specifics. Increasing neuron numbers allows capturing complex data patterns, while decreasing neurons across layers facilitates hierarchical feature representation. Multiple hidden layers enable learning intricate data patterns, with ReLU aiding non-linear learning and sigmoid suitable for binary classification, representing success probability. Achieving Target Model Performance: The model reached approximately 72.5% accuracy, slightly below the 75% target. Despite reasonable performance, further optimization opportunities exist through adjustments in architecture, hyperparameters, or preprocessing. Steps to Increase Model Performance: Strategies include dropping highly correlated columns, adjusting neural network architecture (e.g., increasing neuron count, adding hidden layers), and experimenting with training epochs to balance underfitting and overfitting. Despite efforts, the model fell short of the target, suggesting complex data patterns or the need for hyperparameter exploration. Continued experimentation is recommended for improvement.

Summary

The deep learning model shows promising results, with optimizations in architecture and preprocessing. Exploring alternative models like Random Forest or Gradient Boosting, adept at feature engineering and pattern recognition, could provide fresh insights. Combining traditional machine learning with neural networks might offer a more robust solution.