Alphabet Soup Deep Learning Model Report

Overview

The goal of this project is to develop a deep learning model to predict the success of Alphabet Soup-funded organizations based on historical application data. Using a neural network model (TensorFlow/Keras), the dataset was preprocessed, trained, optimized, and evaluated to achieve the best possible accuracy.

Results

Data Preprocessing

T**arget Variable:**

• IS_SUCCESSFUL → Indicates whether an organization was successfully funded.

Feature Variables:

• Application Type, Classification, Affiliation, Organization Type, Ask Amount, and other categorical features converted via **one-hot encoding**.

Removed Columns:

• EIN and NAME → These are **identifiers** that do not contribute to predictive power.

Compiling, Training, and Evaluating the Model

Neural Network Structure:

• **Input Layer:** n input features (based on preprocessed dataset)

Hidden Layers:

• Layer 1: 80 neurons, **ReLU** activation

• Layer 2: 30 neurons, **ReLU** activation

Output Layer: 1 neuron, **Sigmoid** activation (binary classification)

Training Results:

• **Initial Model Accuracy:** ~72.99%

• **Optimized Model Accuracy:** ~72.96%

• Despite multiple adjustments, the model **did not exceed the 75% accuracy goal**.

Optimization Attempts:

• Added **more neurons and layers** to extract better features.

• Used **Leaky ReLU** activation in one model.

• Implemented **dropout layers** to reduce overfitting.

• Adjusted **learning rates and batch sizes**.

• **Feature Selection (RFE)** was applied to select the 10 most important features.

Summary

Did the model achieve the required 75% accuracy?

No, despite multiple optimization attempts, the highest accuracy achieved was ~72.99%.

Potential reasons for the accuracy limit:

• The **dataset might not contain enough distinct information** to separate successful vs. unsuccessful applicants.

• Some features **may not be strong predictors**, or better feature engineering is needed.

• Deep learning **may not be the best model** for this classification problem.

Recommendation

To solve this classification problem, **a different machine learning model might work better than a deep learning model**:

Random Forest or XGBoost

• These models performed **similarly to the neural network** and require less tuning.

While deep learning provided **moderate accuracy**, trying **Random Forest or XGBoost** in a real-world scenario might lead to a better result **with less computational cost**.