**Report on the Neural Network Model**

**Overview**: The purpose of this analysis is to create a predictive model that helps Alphabet Soup, a nonprofit foundation, select applicants for funding with the highest likelihood of success in their ventures. By developing a binary classifier, this project will aid in determining whether an applicant, if funded, will use the money effectively and be successful.

The analysis is important

1. Improved Decision Making: By predicting which applicants are likely to succeed, Alphabet Soup can allocate its resources more efficiently. This maximizes the impact of its funding and ensures better outcomes for the funded ventures.
2. Data-Driven Insights: The dataset provided contains information about more than 34,000 organizations that have received funding in the past. Analyzing this historical data allows Alphabet Soup to identify patterns and key factors associated with successful ventures.
3. Optimizing Funding: The binary classifier model will automate the selection process, saving time and ensuring that applicants are evaluated based on consistent and data-driven criteria. It helps to minimize human bias and errors in the selection process.
4. Potential for Scaling: As Alphabet Soup grows and supports more ventures, the model can be scaled and adjusted to accommodate new data. The model can be retrained over time to reflect changing dynamics or new insights.

Analysis aspects

1. Target Variable: The variable IS\_SUCCESSFUL will be the target, where the model will predict if the applicant succeeded (1) or failed (0) after receiving funding.
2. Feature Engineering: The analysis includes preprocessing the dataset by cleaning, binning rare categories, and encoding categorical variables, ensuring the data is well-prepared for training a machine learning model.
3. Neural Network Model: A deep learning model with multiple hidden layers will be used to capture complex relationships between the features. The model will be optimized to achieve accuracy higher than 75%, utilizing methods like tuning activation functions, adjusting layers, and modifying input data.
4. Evaluation and Optimization: After training, the model’s performance will be evaluated using loss and accuracy metrics. Optimization steps will be taken to improve the model if the accuracy target is not met initially.

In summary, this analysis will empower Alphabet Soup with an AI-based tool to improve its funding decisions, leading to better resource allocation and a higher success rate for the ventures it supports.

**Results**: The bulleted lists and images support the answers.,

1. Data Preprocessing
   1. The target variables in the model is “IS\_SUCCESSFUL**”** It is a **binary** variable indicating whether a charity application was **successful** (1) or **unsuccessful** (0).:”
   2. The features variables are the attributes or properties used by the model to make predictions about the target variable. [ IS\_SUCCESSFUL,] which indicates whether a charity application was successful (1) or not (0).

These are columns containing numerical values that represent measurable quantities

ASK\_AMT—Funding amount requested

|  |  |
| --- | --- |
| Column name | Description |
| EIN | 9 or 8 digit number identtifying the request |
| NAME | Test description of the mname of the company requesting funds |
| APPLICATION\_TYPE | 3 digit test. 1 of 17 codes d=identify the application type |
| AFFILIATION | description 1 of 7 affiliation types: sector of industry |
| CLASSIFICATION | 4 digit test 1 of 70 classifications |
| USE\_CASE | test description 1 of 5 types of use cases |
| ORGANIZATION | test description 1 of 4 types of organizations |
| STATUS | Binary 1 or 0 |
| INCOME\_AMT | Ranges of income amounts |
| SPECIAL\_CONSIDERATIONS | Binary Yes or No |
| ASK\_AMT | currency amount ranges from 5K to 8.6Billion |
| IS\_SUCCESSFUL | Binary 1 or 0 |

The features include:

APPLICATION\_TYPE

AFFILIATION

CLASSIFICATION

USE\_CASE

ORGANIZATION

STATUS

INCOME\_AMT

SPECIAL\_CONSIDERATIONS

ASK\_AMT

* 1. Variables to Remove: The following variables should be removed because they are identification columns and do not provide meaningful information for prediction:

EIN (Employer Identification Number)

NAME (Organization Name)

1. Compiling, Training, and Evaluating the Model
   1. How many neurons, layers, and activation functions did you select for your neural network model, and why?

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test | Neurons | Layers | Activation function | Epochs | Batch size | Reason |
| 0 | 1st = 80  2nd = 30 | 1st  2nd  Output | 1st = relu  2nd =relu  Output = signmoid | 100 | 80 | Given in assignment |
| 1 | 1st = 80  2nd = 30 | 1st  2nd Output | 1st = relu  2nd =relu  Output = signmoid | 100 | 80 | Reducing bin sizes can reduce complexity, prevent overfitting, improve convergence, and enhance ability to learn from the data effectively. |
| 2 | 1st = 80  2nd= 30 3rd =15 | 1st  2nd  3rd Output | 1st = relu  2nd =relu  3rd = tanh  Output = signmoid | 100 | 80 | Adding hidden layers increases the model's capacity to learn more complex representations  tanh can be particularly useful when you expect a balanced distribution between positive and negative inputs |
| 3 | 1st =400  2nd=200 | 1st  2nd Output | 1st = relu  2nd =relu  Output = signmoid | 600 | 200 | More neurons  -model more complex relationships i.e. learning a greater number of parameters.  -capture fine-grained details for richer internal representations  -reduces underfitting  More epochs  -learn and refine its understanding of the data. |

* 1. Were you able to achieve the target model performance? **NO**
     1. AlphabetSoupCharity: Test Loss: 0.5434 Test Accuracy: 0.7299
     2. 1 Test Lo ss: 0.5848 Test Accuracy: 0.7086
     3. 2 Test Lo ss: 0.5627 Test Accuracy: 0.7255
     4. 3 Test Loss: 0.6284 Test Accuracy: 0.7255
  2. What steps did you take in your attempts to increase model performance?
     1. Attempt0 as prescribed by the assignment
     2. Attempt1 changed the bins
     3. Attempt2 changed layers & model
     4. Attempt3 changed neurons/epochs/batch size

1. **Summary**: Summarize the overall results of the deep learning model.

The 3 additional attempts failed to increase the model accuracy to 75%.

**Additional attempts could include**

* 1. Try less layers/epochs/neurons/batch sizes
  2. Try an alternative to converting categorical data to numeric using label encoding or ordinal encoding
  3. Try other scaling beside StandardScaler such as MinMaxScaler, RobustScaler ,
  4. Change the proportion of test size to train
  5. Don’t use call back

Two recommendations for how a different model could solve this classification problem, and then explain your recommendation.

A Random Forest model is an alternative for this binary classification problem because it can efficiently handle complex feature interactions, avoid overfitting, provide feature importance insights, and perform well without extensive hyperparameter tuning. Its robustness and flexibility make it an excellent choice for solving classification tasks like predicting the success of applicants based on various organizational and funding-related features.

Using an SVM with an RBF kernel could solve the classification problem of predicting the success of applicants by effectively capturing non-linear relationships and providing robust classification boundaries. Its focus on support vectors ensures that the model generalizes well, making it a reliable option for this classification task where the aim is to make accurate predictions for future funding applicants based on historical data.

**Summary of outputs for the four model versions**

AlphabetSoupCharity.ipynb

**Model: "sequential"**

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┓

┃ **Layer (type)** ┃ **Output Shape** ┃ **Param #** ┃

┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━┩

│ dense (Dense) │ (None, 80) │ 3,520 │

├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤

│ dense\_1 (Dense) │ (None, 30) │ 2,430 │

├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤

│ dense\_2 (Dense) │ (None, 1) │ 31 │

└──────────────────────────────────────┴─────────────────────────────┴─────────────────┘

**Total params:** 5,981 (23.36 KB)

**Trainable params:** 5,981 (23.36 KB)

**Non-trainable params:** 0 (0.00 B)

first\_neuron = 80 activation='relu'

second\_neuron = 30 activation='relu'

epochs = 100 batch\_size = 80

**Test Loss: 0.5634**

**Test Accuracy: 0.7299**

**AlphabetSoupCharity-O1.ipynb**

Updated 'APPLICATION\_TYPE' Value Counts After Replacement:

APPLICATION\_TYPE

T3 27037

Other 7262

CLASSIFICATION

C1000 17326

C2000 6074

Other 6062

C1200 4837

Initialize the Sequential Model

first\_neuron = 80, activation='relu'

second\_neuron = 30, activation='relu'))

**Model: "sequential"**

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┓

┃ **Layer (type)** ┃ **Output Shape** ┃ **Param #** ┃

┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━┩

│ dense (Dense) │ (None, 80) │ 2,800 │

├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤

│ dense\_1 (Dense) │ (None, 30) │ 2,430 │

├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤

│ dense\_2 (Dense) │ (None, 1) │ 31 │

└──────────────────────────────────────┴─────────────────────────────┴─────────────────┘

**Total params:** 5,261 (20.55 KB)

**Trainable params:** 5,261 (20.55 KB)

**Non-trainable params:** 0 (0.00 B)

Train the Neural Network Model

epochs = 100

batch\_size = 80

**Test Loss: 0.5848**

**Test Accuracy: 0.7086**

**AlphabetSoupCharity-O2.ipynb**

Initialize the Sequential Model

first\_neuron = 80 Activation function: ReLU

second\_neuron = 30  Activation function: ReLU

third\_neuron = 15 Using tanh  activation function.

**Model: "sequential"**

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┓

┃ **Layer (type)** ┃ **Output Shape** ┃ **Param #** ┃

┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━┩

│ dense (Dense) │ (None, 80) │ 3,520 │

├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤

│ dense\_1 (Dense) │ (None, 30) │ 2,430 │

├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤

│ dense\_2 (Dense) │ (None, 15) │ 465 │

├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤

│ dense\_3 (Dense) │ (None, 1) │ 16 │

└──────────────────────────────────────┴─────────────────────────────┴─────────────────┘

**Total params:** 6,431 (25.12 KB)

**Trainable params:** 6,431 (25.12 KB)

**Non-trainable params:** 0 (0.00 B)

Train the Neural Network Model

epochs = 100

batch\_size = 80

**Test Loss: 0.5627**

**Test Accuracy: 0.7255**

**AlphabetSoupCharity-O3.ipynb**

**Model: "sequential"**

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┓

┃ **Layer (type)** ┃ **Output Shape** ┃ **Param #** ┃

┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━┩

│ dense (Dense) │ (None, 400) │ 17,600 │

├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤

│ dense\_1 (Dense) │ (None, 200) │ 89,200 │

├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤

│ dense\_2 (Dense) │ (None, 1) │ 201 │

└──────────────────────────────────────┴─────────────────────────────┴─────────────────┘

**Total params:** 98,001 (382.82 KB)

**Trainable params:** 98,001 (382.82 KB)

**Non-trainable params:** 0 (0.00 B)

Train the Neural Network Model

epochs = 600

batch\_size = 200

**Test Loss: 0.6284**

**Test Accuracy: 0.7255**