Text

Description automatically generated

Changing layers to hidden: 20,10 from 8,5

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generatedTanh

Text

Description automatically generated

Fourth layer

Text

Description automatically generated

2nd layer outer

Text

Description automatically generated

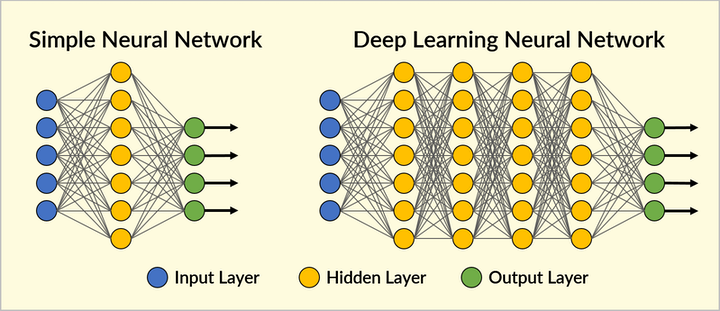
**Neural Network Charity Analysis**

**Overview of Project**

Bek’s come a long way since her first day at that boot camp five years ago—and since earlier this week, when she started learning about neural networks! Now, she is finally ready to put her skills to work to help the foundation predict where to make investments.

With your knowledge of machine learning and neural networks, you’ll use the features in the provided dataset to help Beks create a binary classifier that is capable of predicting whether applicants will be successful if funded by Alphabet Soup.

From Alphabet Soup’s business team, Beks received a CSV containing more than 34,000 organizations that have received funding from Alphabet Soup over the years. Within this dataset are a number of columns that capture metadata about each organization, such as the following:

[](https://github.com/emmanuelmartinezs/Neural_Network_Charity_Analysis/blob/main/Resources/Images/header.png?raw=true)

* **EIN** and **NAME**—Identification columns
* **APPLICATION\_TYPE**—Alphabet Soup application type
* **AFFILIATION**—Affiliated sector of industry
* **CLASSIFICATION**—Government organization classification
* **USE\_CASE**—Use case for funding
* **ORGANIZATION**—Organization type
* **STATUS**—Active status
* **INCOME**\_AMT—Income classification
* **SPECIAL**\_CONSIDERATIONS—Special consideration for application
* **ASK\_AMT**—Funding amount requested
* **IS\_SUCCESSFUL**—Was the money used effectively

**Deliverables:**

This new assignment consists of two technical analysis deliverables and a written report.

1. ***Deliverable 1:*** Preprocessing Data for a Neural Network Model
2. ***Deliverable 2:*** Compile, Train, and Evaluate the Model
3. ***Deliverable 3:*** Optimize the Model
4. ***Deliverable 4:*** A Written Report on the Analysis [README.md](https://github.com/emmanuelmartinezs/Neural_Network_Charity_Analysis)

**Resources:**

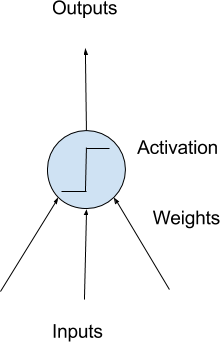
This new assignment consists of three technical analysis deliverables and a proposal for further statistical study. You’ll submit the following:

* Data Source: charity\_data.csv, AlphabetSoupCharity.h5 and AlphabetSoupCharity\_Optimzation.h5
* Data Tools: AlphabetSoupCharity\_starter\_code.ipynb, AlphabetSoupCharity.ipynb and AlphabetSoupCharity\_Optimzation.ipynb.
* Software: Python 3.9, Visual Studio Code 1.50.0, Anaconda 4.8.5, Jupyter Notebook 6.1.4 and Pandas

**Before Start:**

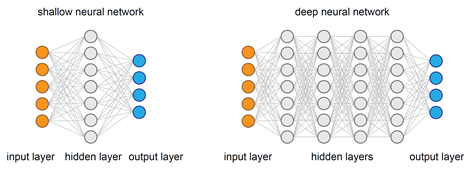
**Extra Note**

Basically, there are 3 important layers which are the input layer, hidden layer, and output layer. In each layer, there are neurons that are added with weight to produce the outputs through the activation functions (i.e: Sigmoid, ReLU, softmax) which define the output. The neural networks are further divided into shallow and deep neural networks as shown above. Depend on the dataset you are dueling with, you could add many hidden layers to generate a better result.

[](https://github.com/emmanuelmartinezs/Neural_Network_Charity_Analysis/blob/main/Resources/Images/neurona.png)

**Extra Note**

We should import the necessary library such as Pandas and PyTorch Library. Then, we get the dataset and determine the input and target for prediction. These input and target are converted into NumPy array and split into training, testing and validation set. You can visit my notebook if you are interested to get the details of the source code.

[](https://github.com/emmanuelmartinezs/Neural_Network_Charity_Analysis/blob/main/Resources/Images/mini_header.png)

Let's move on!

**Deliverable 1:**

**Preprocessing Data for a Neural Network Model**

**Deliverable Requirements:**

Using your knowledge of Pandas and the Scikit-Learn’s StandardScaler(), you’ll need to preprocess the dataset in order to compile, train, and evaluate the neural network model later in Deliverable 2.

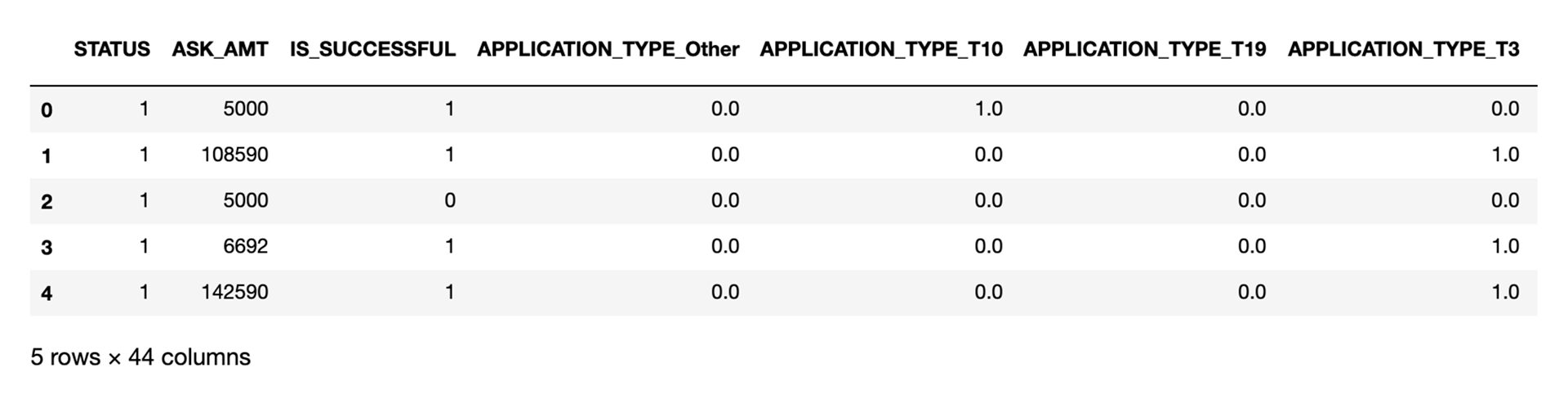
To Deliver.

**Follow the instructions below:**

Using the information we have provided in the starter code, follow the instructions to complete the preprocessing steps.

1. Read in the charity\_data.csv to a Pandas DataFrame, and be sure to identify the following in your dataset:
   * What variable(s) are considered the target(s) for your model?
   * What variable(s) are considered the feature(s) for your model?
2. Drop the EIN and NAME columns.
3. Determine the number of unique values for each column.
4. For those columns that have more than 10 unique values, determine the number of data points for each unique value.
5. Create a density plot to determine the distribution of the column values.
6. Use the density plot to create a cutoff point to bin "rare" categorical variables together in a new column, Other, and then check if the binning was successful.
7. Generate a list of categorical variables.
8. Encode categorical variables using one-hot encoding, and place the variables in a new DataFrame.
9. Merge the one-hot encoding DataFrame with the original DataFrame, and drop the originals.

At this point, your merged DataFrame should look like this:

[](https://github.com/emmanuelmartinezs/Neural_Network_Charity_Analysis/blob/main/Resources/Images/e1.png)

1. Split the preprocessed data into features and target arrays.
2. Split the preprocessed data into training and testing datasets.
3. Standardize numerical variables using Scikit-Learn’s StandardScaler class, then scale the data.

**Deliverable 1 Requirements**

You will earn a perfect score for Deliverable 1 by completing all requirements below:

* The following preprocessing steps have been performed:
  + The EIN and NAME columns have been dropped
  + The columns with more than 10 unique values have been grouped together
  + The categorical variables have been encoded using one-hot encoding
  + The preprocessed data is split into features and target arrays
  + The preprocessed data is split into training and testing datasets
  + The numerical values have been standardized using the StandardScaler() module

**Deliverable 2:**

**Compile, Train, and Evaluate the Model**

**Deliverable Requirements:**

Using your knowledge of TensorFlow, you’ll design a neural network, or deep learning model, to create a binary classification model that can predict if an Alphabet Soup–funded organization will be successful based on the features in the dataset. You’ll need to think about how many inputs there are before determining the number of neurons and layers in your model. Once you’ve completed that step, you’ll compile, train, and evaluate your binary classification model to calculate the model’s loss and accuracy.

To Deliver.

**Follow the instructions below:**

Using the information we have provided in the starter code, follow the instructions to complete the preprocessing steps.

1. Continue using the AlphabetSoupCharity.ipynb file where you’ve already performed the preprocessing steps from Deliverable 1.
2. Create a neural network model by assigning the number of input features and nodes for each layer using Tensorflow Keras.
3. Create the first hidden layer and choose an appropriate activation function.
4. If necessary, add a second hidden layer with an appropriate activation function.
5. Create an output layer with an appropriate activation function.
6. Check the structure of the model.
7. Compile and train the model.
8. Create a callback that saves the model's weights every 5 epochs.
9. Evaluate the model using the test data to determine the loss and accuracy.
10. Save and export your results to an **HDF5** file, and name it AlphabetSoupCharity.h5.

**Deliverable 2 Requirements**

You will earn a perfect score for Deliverable 1 by completing all requirements below:

* The neural network model using Tensorflow Keras contains working code that performs the following steps:
  + The number of layers, the number of neurons per layer, and activation function are defined
  + An output layer with an activation function is created
  + There is an output for the structure of the model
  + There is an output of the model’s loss and accuracy
  + The model's weights are saved every 5 epochs
  + The results are saved to an HDF5 file

**Deliverable 3:**

**Optimize the Model**

**Deliverable Requirements:**

Using your knowledge of TensorFlow, optimize your model in order to achieve a target predictive accuracy higher than 75%. If you can't achieve an accuracy higher than 75%, you'll need to make at least three attempts to do so.

**NOTE**

The accuracy for the solution is designed to be lower than 75% after completing the requirements for Deliverables 1 and 2.

Optimize your model in order to achieve a target predictive accuracy higher than 75% by using any or all of the following:

* Adjusting the input data to ensure that there are no variables or outliers that are causing confusion in the model, such as:
  + Dropping more or fewer columns.
  + Creating more bins for rare occurrences in columns.
  + Increasing or decreasing the number of values for each bin.
* Adding more neurons to a hidden layer.
* Adding more hidden layers.
* Using different activation functions for the hidden layers.
* Adding or reducing the number of epochs to the training regimen.

To Deliver.

**Follow the instructions below:**

Using the information we have provided in the starter code, follow the instructions to complete the preprocessing steps.

1. Create a new Jupyter Notebook file and name it AlphabetSoupCharity\_Optimzation.ipynb.
2. Import your dependencies, and read in the charity\_data.csv to a Pandas DataFrame.
3. Preprocess the dataset like you did in Deliverable 1, taking into account any modifications to optimize the model.
4. Design a neural network model, taking into account any modifications that will optimize the model to achieve higher than 75% accuracy.
5. Create a callback that saves the model's weights every 5 epochs.
6. Save and export your results to an HDF5 file, and name it AlphabetSoupCharity\_Optimization.h5.

**Deliverable 2 Requirements**

You will earn a perfect score for Deliverable 1 by completing all requirements below:

* The model is optimized, and the predictive accuracy is increased to over 75%, or there is working code that makes three attempts to increase model performance using the following steps:
  + Noisy variables are removed from features
  + Additional neurons are added to hidden layers
  + Additional hidden layers are added
  + The activation function of hidden layers or output layers is changed for optimization
  + The model's weights are saved every 5 epochs
  + The results are saved to an HDF5 file

**DELIVERABLE RESULTS:**

**Data Preprocessing**

For this analysis and model, the target is held in IS\_SUCCESSFUL field.

**The following variable(s) should be considered on features model**

* ORGANIZATION
* STATUS
* INCOME\_AMT
* SPECIAL\_CONSIDERATIONS
* ASK\_AMT
* APPLICATION\_TYPE
* AFFILIATION
* CLASSIFICATION
* USE\_CASE

**The following variable(s) should be removed from input and data.**

* NAME
* EIN

**Compiling, Training, and Evaluating the Model**

**Model Configuration:**

* hidden\_nodes\_layer1 = 80
* hidden\_nodes\_layer2 = 30
* number\_input\_features = 43

**This model acheived 63.8% accuracy with several attempts to incraese the accuracy including:**

* Increasing the number of hidden nodes in layer 1 (3 X number of input features)
* Increasing the number of hidden layers to include a 3rd
* Changing the activation functions: tried linear, tanh, sigmoid for a combination of hidden layers and output layer

**SUMMARY**

Our Analysis and Deep Learning Model Results include a recommendation for how a different model could solve this classification and results.