Charity Organizations Analysis Using Neural Networks

Bird's eye view:

The purpose of this exercise is to develop, with the use of neural networks, a binary classifier that is capable of determining whether or not a particular application would be successful in obtaining funding from a charitable organization known as "Alphabet Soup". We were given a CSV file by the business team at Alphabet Soup that included the names of more than 34,000 groups that have been successful in obtaining funding throughout the years.

Results:

The columns of the data set are:

- 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

Data Preprocessing:

- What variable(s) are considered the target(s) for the model?

  - Target, T, is the correct or desired value for the response associated to one input, X. This value will be compared with the output (the response from the neural network), Y to guide the learning process involving the weight changes. The difference between the desired result (the target, T) and the actual output, Y, is the error. The objective of training the neural network is to minimize the error.

  - In our case, the objective is that the Neural Network be able to predict if an organization is going to be successful or not, using the funds received, so the IS_SUCCESSFUL column contains the target variable. Target variables are also known as dependent variable and we are using this variable to train our model.

- What variable(s) are considered to be the features for your model?
  - Input values are defined as features for the model and are also referred to as independent variables. All the columns in the CSV except the target variable IS_SUCCESSFUL and the ones we dropped — EIN and NAME are included in those variables.

- What variable(s) are neither targets nor features, and should be removed from the input data?
  - The columns EIN and NAME do not contain data that gives additional information to the model. They would just add noise to the problem and were therefore removed from the dataset using the drop function from Pandas.

  - In the same way, variables with too many unique values would be removed. In our example, the column ASK_AMT has 8747 unique values, so this variable should also be eliminated, or at least "binned" in order to reduce the number of variables that the model will have to deal with.

What is Binning?:

- Binning is a technique that accomplishes exactly what it sounds like. It will take a column with continuous numbers and place the numbers in “bins” or categories based on ranges that we determine. This will give us a new categorical variable feature.

Compiling, Training, and Evaluating the Model:

- How many neurons, layers, and activation functions did you select for your neural network model, and why?
    - A good rule of thumb for a basic neural network is to have two to three times the amount of neurons in the hidden layer as the number of inputs. In the first run, the model had two hidden layers, the first layer had 80 neurons and the second layer had 30 neurons. These parameters were changed in subsequent runs, but they will be explained later on.

    - Other parameters used for the first run were the RELU activation function and the adam optimizer. Adam (the name Adam is derived from adaptive moment estimation) is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data.

    - The binary crossentropy was used as the loss function. Binary crossentropy is a loss function that is used in binary classification tasks. These are tasks that answer a question with only two choices (yes or no, A or B, 0 or 1, left or right). Several independent such questions can be answered at the same time

- Were you able to achieve the target model performance?
    - In the instructions for this challenge it is stated that "The accuracy for the solution is designed to be lower than 75 %", so the objective of the exercise is to optimize the Tensorflow model in order to achieve a target predictive accuracy higher than 75 %.

- What steps did you take to try and increase model performance?
    - There were four attempts to improve the model's accuracy. The first three attempts involved changing the activation function, and the fourth attempt involved changing the number of hiden layers and neurons.

Summary:

What is the relationship between the accuracy and the loss in deep learning?: There is no relationship between these two metrics.

Loss can be defined as the difference between the problem's true values and the values predicted by the model. The greater the loss, the greater the magnitude of the data errors.

The number of errors made on the data can be used to calculate accuracy.

That means:

- A low accuracy and large loss indicates that there are numerous errors on a large amount of data.

- A low accuracy but low loss indicates that there are minor mistakes on a large amount of data.

- A high accuracy with low loss indicates few errors on a small set of data (best case scenario).

The results show that it was not possible to exceed the level of 75 percent accuracy, even with the original run's settings. The loss results in the five cases presented are dismal. In each case, the loss function is greater than 50%. The RELU function performed the worst in this regard, with a loss value of 70.64 percent.

In terms of accuracy, the difference between runs was marginal. The difference between the lowest and highest accuracy is only 0.11 percent (72.58 % - 72.47 % = 0.11 %), indicating that the model was not improved by the changes made.

So, how could the model be improved?

- Considering gathering more data.
- Testing additional activation functions like Leaky RELU, Parametric RELU, ELU, Softmax, Swish, GELU or SELU.
- Testing additional loss functions for binary classification like Hinge Loss or Squared Hinge Loss.
- Testing additional optimizer functions like Adadelta, Adagrad, RMSprop, SGD with Momentum or SGD.
- Increasing the number of hidden layers.
- Increasing the number of neurons per layer.