Charity Candidate Prediction for Alphabet Soup

via Machine Learning

Objective

Selecting candidates to receive funding for charitable causes is a challenging and involved process, requiring vetting and serious consideration. Our goal here is to establish an accurate prediction of whether or not an application should be rejected or denied via the employment of a machine learning algorithm to parse through thousands of records in a fraction of the time it would take a human to go through one of such.

This report goes over the development of a machine learning algorithm developed over the past few days to do just that.

Results

While the model developed was not able to reach the accuracy target goal of 75%, we were able to achieve reasonably close, at ~73% in the final exported model. Considering that the data contain approximately 46% of applications considered successful, this accuracy results can be confirmed as significantly better than random chance.

Methodology

Data Preprocessing

Target Variable Determination

We are looking to evaluate which applicants will be most effective with granted funds; therefore, the model is being developed with the purpose of predicting the “IS\_SUCCESSFUL” column. No other variables in the data appear to be targets.

Feature Determination

Our ideal feature set, stated in rather blunt terms, is one that provides the greatest number of useful metrics to the model and the least amount of irrelevant noise. As a result, we want to use as much information as possible to inform our model (of course, within reason, as well as ethical and legal boundaries). For this reason, all columns of provided data were initially used as features (aside from the prediction of “IS\_SUCCESSFUL”) and then picked through afterwards to remove noise in an attempt to improve accuracy.

Noise Reduction

Some good indicators of whether to drop a column is if its records are:

* Overwhelmingly dominated by one value, to the extent that any non-standard values are outliers
* The values appear to be completely uncorrelated with all other metrics across the data, *especially* with the outcome (“IS\_SUCCESSFUL” column)
* The values extremely strongly correlate with that of another column. In this case, it would be better to pick one or the other, or perhaps alternate between the two to see which produces better results.

Based primarily on the first bullet point above, the columns ‘SPECIAL\_CONSIDERATIONS” and “STATUS” were dropped. Further exploratory analysis is warranted to quantitatively determine other relationships or columns that could be dropped. Principle Component Analysis might also be a worthwhile tool in this endeavor with Component Explanation capabilities, which could help determine where correlations lie.

Model Development

Neural Network Structure

Throughout the manual process of tweaking the model for accuracy, the highest performing model contained a total of 7281 neurons and four (4) hidden layers. In order, the hidden layers performed the following activation functions:

1. Tanh
2. Sigmoid
3. Linear
4. ReLU

And the output was also run through a final Sigmoid function to collapse the answer into a binary response. While this is a somewhat arbitrary collection of a variety of methods, it was the result of numerous rounds of tweaking, which will be addressed further on in the Model Performance Tuning section of the report.

While the structure was arrived at from the aforementioned results-oriented approach, the initial principle was to run the input data through layers of “transformation” activators followed by “Boolean-approximating” activators in the hopes that the model could tweak weights more granularly in between cutoff layers to arrive at a more nuanced response.

Target Model Accuracy: Expectations vs Results

As mentioned in the Results section earlier, the model developed failed to reach the desired accuracy target, despite numerous efforts to improve performance through tweaking the model and inputs.

The Deep Learning model ostensibly plateaus at around 72-73% accuracy levels for most configurations. While greater efficiency should be possible, it’s unlikely to be achieved without a more involved evaluation of the data and machine learning techniques than was feasible for this project. This same plateau was also observed by others working on this same effort independently (namely, I heard report from Riley Capps that he was unable to achieve over 74% accuracy for his own model).

Model Performance Tuning

In the process of attempting to tune the machine model, the following techniques were used:

* Changing activation functions,
* Changing hidden layer counts,
* Tweaking hidden layer unit allocations,
* Modifying epoch iterations,
* Tweaking the feature set to experimentally determine ‘noise’ features by dropping them and seeing the resulting effect on overall accuracy.

All hidden layers for the Deep Learning model were densely connected (that is, each node was connected to every other from the preceding and *pro*ceeding layers)

Unfortunately, in the effort to shoot past 75%, the best-performing model noted down was not saved in the process. That particular model had an accuracy rate of 73.42% vs the final submission which achieved ~73.11% accuracy. While these amount to very small differences in overall predictive power, it is nonetheless unfortunate.

Of the features tweaked in the set, the “SPECIAL\_CONSIDERATIONS” and “STATUS” appeared to have negligible or weakly positive impacts on accuracy. Removal of the “AFFILIATION” column was attempted as well, but this action *significantly* harmed accuracy of the model (by ten to fifteen percentage points (10-15%)) so the feature was promptly re-inserted.

It also bears noting that performance measures were *not* taken to reduce the neuron count for the model. Over 7 thousand parameters is hardly an efficient machine learning algorithm even for arguably more complex tasks, but that step of the process is usually done *after* accuracy targets are met in the aim of optimizing resource usage to achieve the same or similar result. Since that target was never reached, this optimization process was also not undertaken.

Additionally, since we were specifically looking for a Boolean output of yes/no, it is possible that SMOTE could be used to remove some biases in the data, since it wasn’t a 50/50 split of ventures that were successful vs unsuccessful. That said, SMOTE was not used under the scope of this project.

Summary

Using machine learning and TensorFlow, we were able to develop a model to predict whether or not a venture candidate will effectively use provided funds with a 73% accuracy rate. This efficacy, while not an outright predictor of success, can still serve as a useful tool in informing decisions regarding fund allocations.

Despite the shortcomings in meeting the 75% accuracy goal, Deep Learning methods appear to be worthwhile for this project. That said, it does present the problem commonly known of being a “black box” where it is not well-known what *exactly* it’s doing to arrive at its conclusions. In the interests of greater transparency as a non-profit, it might be worth considering an alternative model such as Random Trees for these sorts of decisions, given that the evaluation process is much, much easier to audit for mistakes or ethical concerns.