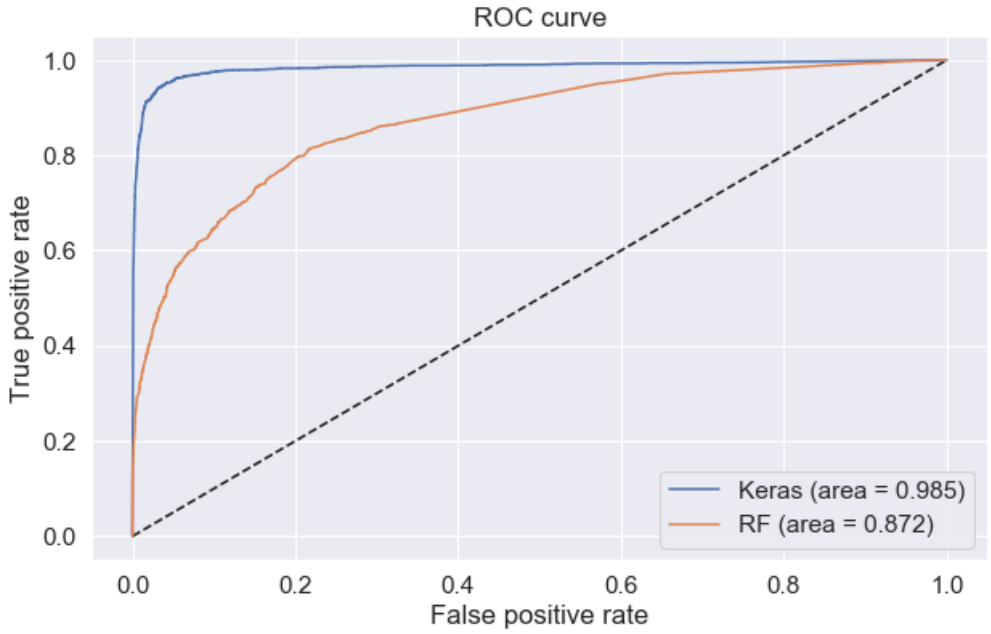
Mohsen Kheirabadi (Mohsen.kheirabadi@gmail.com)

Step 4 - Compare your modeling approaches:

Please prepare a relatively short write-up comparing the pros and cons of the two modeling techniques you used (PDF preferred). Are there choices you made in the context of the exercise that might be different in a business context?

I applied a Random Forest model (a conventional method) vs a Deep Neural Network model (newer trend in machine learning). When we look at the classification reports of these two model and compare the results we could easily find out that the deep neural network outperformed the Random Forest method. The Area Under the ROC Curve (AUC) in DNN model is around 0.985 while this area for RF is 0.872.



An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

- True Positive Rate
- False Positive Rate

AUC ranges in value from 0 to 1. A model whose predictions are 100% wrong has an AUC of 0.0; one whose predictions are 100% correct has an AUC of 1.0.

In our case Keras (DNN) model performed better than RF, because its AUC value is bigger.

Step 5 - Submit your work:

Your submission should consist of all the code used for EDA, cleaning, prepping, and modeling (text, html, or pdf preferred), the two result files (.csv format - each containing 10,000 decimal probabilities), and your write-up comparing the pros and cons of the two modeling techniques used (text, html, or pdf preferred).

Comparing the pros and cons of the two modeling techniques:

The Random Forests can only work with tabular data while a Neural Network model can work many different data types.

In theory, RF can work with huge datasets, but in real-life applications, after preprocessing, data will become sparse and RF will be stuck. In this case a Neural Network work much better than a RF model.

In theory, the Random Forests should work with missing and categorical data. However, the sklearn implementation doesn't handle this, therefore to prepare data for Random Forests (in python and sklearn package) we need to make sure that:

- There are no missing values in our data
- Convert categorical data into numerical (Using methods like One-hot Encoding)

Data preprocessing for Neural Networks requires more works. For a Neural Network filling missing values and converting categorical data into numerical data along with feature scaling are essential needs. In the case of different ranges of features, there will be problems with a Neural Network model training.

For Random Forests, we set the number of trees in the ensemble and accept the defaults but in order to train a Neural Network we need to define the NN architecture. We need to define the number of layers, the number of neurons, the type of activation function, training algorithm and in general we need to have a deep understanding of Neural Network architecture in order to set it up.

One of the problems with a Neural Network is that each of the NN hyper-parameters mentioned above can be critical. For example, if we set too large learning rate or not enough neurons in second hidden-layer and our NN training will be stuck in a local minimum.

To sum up, if we are going to work with tabular data, it is worth to check the Random Forests first because it is easier. The Random Forests requires less preprocessing and the training process is simpler. Therefore, it is simpler to use RF in the production system. If we are not happy with the model performance we definitely should try to tune and train a Neural Network. We need enough knowledge and experience to tune so many hyper-parameters in NN and instead we can obtain very good results with NN.