MACHINE LEARNING PROJECT – CHURN

INTRODUCTION

Understanding and predicting customers' behaviours has become more and more meaningful and essential in nowadays businesses. If until a few decades ago it was unconceivable thinking that a machine could predict whether a customer would churn or not, nowadays, thanks to the advances in technology and computing power, Artificial Intelligence and Machine Learning models, this has become much more than feasible. Dealing with such kind of tasks is important for data scientists that wants to improve their skills and, therefore, as project for our "Machine Learning" course, we chose to perform a data analysis on the churn dataset of a bank, with the aim of predicting whether the customers' accounts are active or closed. In order to solve such machine learning task, we experimented different models and compared their outputs to identify which methods would guarantee the best results.

METHODS

To conduct our predictive analysis, we followed the steps that are conform to the typical data science pipeline, in particular data exploration and visualization, data processing and cleaning, feature engineering, models training and models evaluation.
The dataset was provided by Professor Italiano, and contains 10127 instances with 17 variables, including the ID of the customers, several demographic features regarding them, and several variables regarding the product to which they are associated (credit cards). The target is called 'Attrition_Flag', and it is a categorical binary variable, indicating if the customers are "Existing", with a still used bank account, or "Attrited", closed bank account. The dataset contains both numerical and categorical data, and there are some missing values in the form of unknown.
We proceeded by analysing several features that allowed us to describe better the dataset, through different plots. After some modifications on some variables, we scaled our data, and proceeded to build the models that will make the predictions.
We trained many machine learning models, all suited for a binary classification task:

- The baseline is a simple logistic regression.
- A Support Vector Machine, in which we optimized the parameters using a grid search.
- A Decision Tree, in which we also computed the optimal depth with respect to accuracy.
- Two ensemble tree models, namely an Extreme Gradient Boosting and a Random Forest.

Afterwards, we decided to look for a suitable neural network for our task, and through some research and a bit of experimentation we have obtained three variants:

- The first Neural Network is composed by a dense layer, a dropout, and again two dense, and used the Adam optimizer.
- The second has three dense layers, with respectively 120, 60 and 1 neuron, was trained for 140 epochs, and uses the RMSprop optimizer.
- The last has again three dense layers, but this time the first two with 520 neurons, was trained for 60 epochs, and uses the Adam optimizer.

All our deep learning models had in common the binary cross-entropy as loss function, were trained with a batch of 60, and all the dense layers have a 'relu' (Rectified Linear Unit) activation function, except for the output neuron that has a 'sigmoid' activation which is necessary for outputting a probability value between 0 and 1.
At first, we trained our models with the complete set of variables. Successively, looking for a better accuracy, we selected a subset of variable looking at their correlation with the target, and we tested each model again on it.

Even if our dataset has not many variables, for research purposes we performed a principal component analysis, and from our initial set of independent variables we ended up with two principal components that were chosen with the elbow method. All the models described before were tested also on this configuration.
Our work is concluded by comparing the accuracy of all these models.

CODE DESCRIPTION

This section describes all the steps performed in the code, that can be found attached as notebook with results and graph printed, and as a .ipynb file to be run.

In the notebook, we started by importing all the libraries with their modules needed to perform the machine learning tasks:

- Pandas for importing, reading, and manipulating csv file and data frames.
- Numpy to manipulate data, numbers and arrays.
- Matplotlib, Seaborn, Plotly, Dash, Jupyter-dash for data visualization.
- Sklearn with all its related modules and Xgboost, for the machine learning models.
- Tensorflow and Keras for fitting and compiling the neural networks.

Then, we imported the csv file, deleting the unknown column, and started analysing the variables, and specifically examining the summary descriptive statistics.

We produced then many plots representing several variables:

- A pie plot of customers by gender. It shows that there the customers are quite balanced by gender (47% males, 53% females)
- A bar plot representing the total year transaction amount made by the customers, grouped by their educational level. Surprisingly people who spend less are those in the top educational tier (Doctorate) while those are the second to spend the highest amount of money
- A bar plot representing the total year transaction amount made by the customers, grouped by their marital status. Singles are the ones who spend the highest amount of money, while married the least (probably because they are controlled by their partner or are saving for their children future expenses)
- Several boxplots representing the credit limit grouped by the income categories. It shows that the credit limit is highly correlated with the income category (Increasing value going from the lowest to highest category)

Successively we created an interactive dashboard for visualization purposes containing two graphs that can be filtered with sliders and dropdown menus.

The first graph is a scatterplot of 'Transaction_Amount' plotted against 'Credit_Limit' where the color is represented by the 'Card_Category' that can be filtered thorugh the dropdown menu and the symbol is represented by the 'Income_Category' that can be filtered with a slider. The second graph is again a scatter plot plotting 'Avg_Utilitization_Ratio' of the cards vs 'Customer_Age' where the color is represented by the 'Education_Level' that can be filtered with a slider.

We proceeded then to prepare out data before ingesting them in the algorithms. Several categorical features were transformed into numeric: the “Gender” column was transformed into two dummy variables, and the variables “Income_Category”, “Educational_Levels”, and “Card_Category” were

modified in numerical classes, also deleting the rows that had an “Unknown” value in these fields. We plotted then a correlation matrix that we used successively to select a subset of the variables. The last passage in this section was to scale the data with the standard scaler library.

All the models described in the previous section were trained and used for prediction with three different combinations of features. Before executing the algorithms in each of these configurations, we divided the dataset in train and test set, assigning to the test size the 0.25% of the observations; we also created a dictionary containing the accuracy of prediction on the test set for each model, in order to plot them at the end of each section.

At first, we used the complete set of variables in the dataset to predict the target, and the models gave pretty good results. Then, we decided to exclude several variables according to their correlation with the target, keeping only those with a correlation higher than 0.1 or lower than 0.1, therefore the list of deleted variables is the following:

['Income_Category', 'Card_Category', 'Months_on_book', 'Customer_Age', 'Education_Level', 'Dependent_count', 'Credit_Limit', 'Gender_F', 'Gender_M']

Finally, we decided to perform a principal component analysis, to reduce the dimensions of our data; it produced two components explaining the totality of the variability of the dataset, and we trained the same models on them.

For each machine learning model, we plotted a confusion matrix to show the rate of correct predictions; for each model and each configuration we used the accuracy as main evaluation metric, the results are showed in three scatterplots, and will be discussed in the result section.

EXPERIMENTAL DESIGN

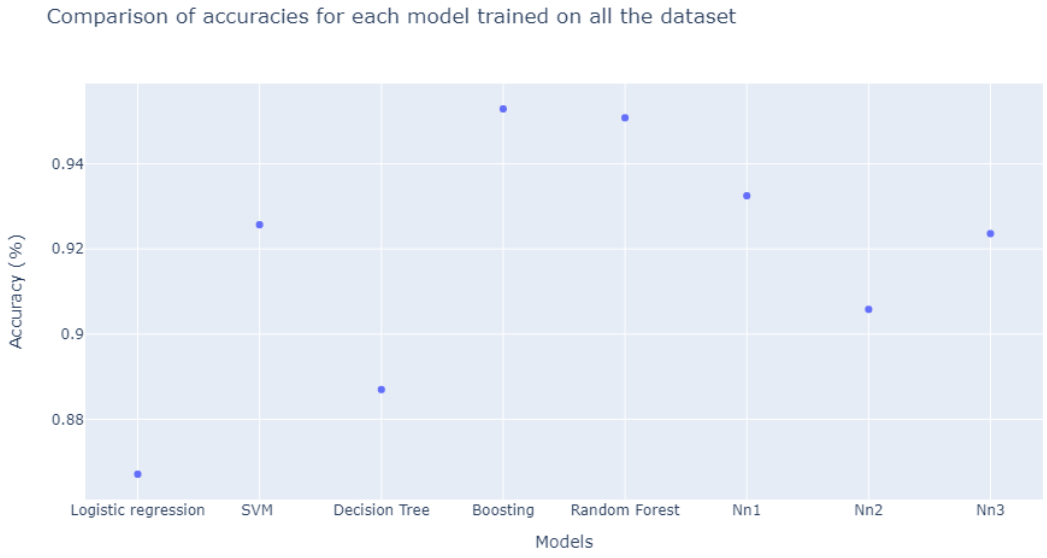
Our main purpose was to find an optimal machine learning method to predict whether a customer in our dataset has churned the company. This actually was a binary classification task, since our target variable had two possible values: “Existing Customer” or “Attrited Customer”, and for computing reasons these were transformed to 0 or 1 values. This task was accomplished by selecting first logically a set of models suitable for this kind of target, that are those described before, and applying them to our dataset after having prepared it in the best form to be fed to these algorithms. Together with the models, we tried to improve the accuracy by using several techniques capable of adjusting and fine tuning the algorithms: this was the case of the grid search, of the PCA that provided us a linear transformation of our dataset with few dimensions, and of course of the neural networks tuning by selecting the layers, the activation functions, and the optimizers.

The baseline model was a logistic regression, and as described in the next section, those which were implemented after it had mostly all a slightly improvement. Given that it was a binary classification task, as evaluation metrics we used the accuracy, and for each model we outputted a confusion matrix of the predictions and the real values.

RESULTS

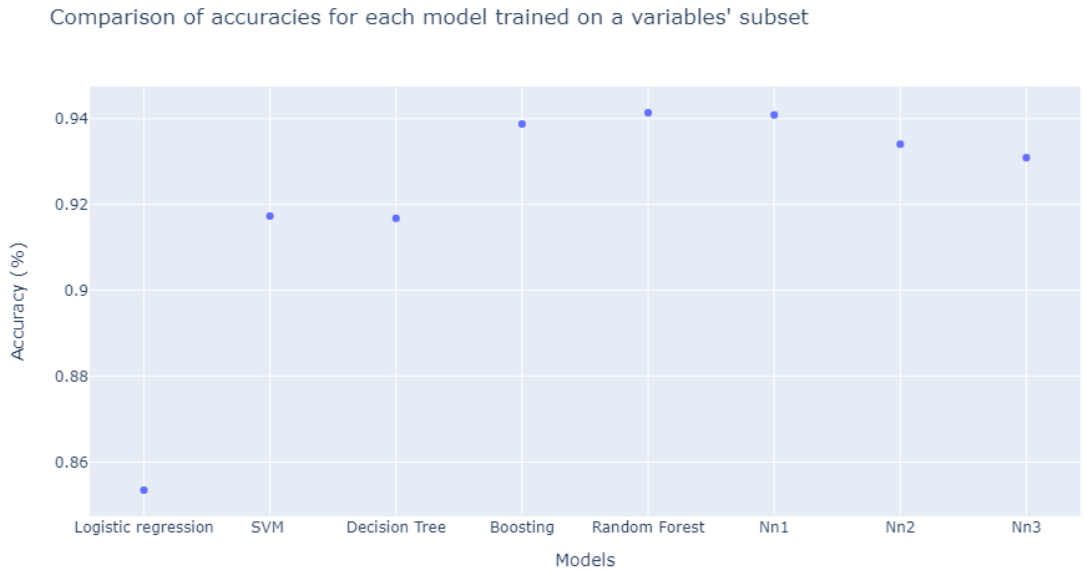
In this section we analyze the results of all the models used with the various combinations of input variables.

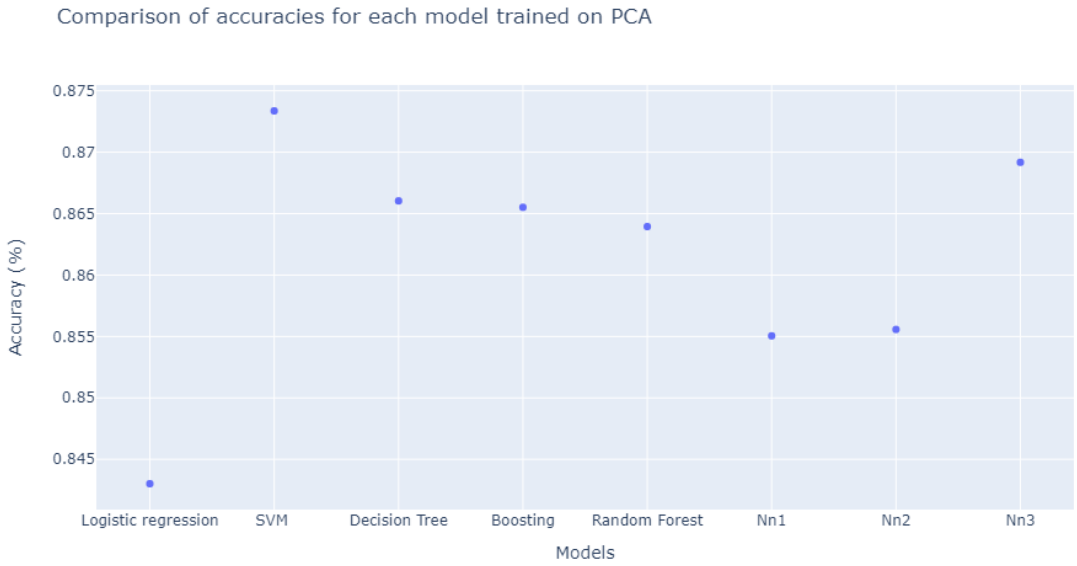
The first plot shows the accuracies for the algorithms trained on the whole set of independent variables. Our baseline model, the logistic regression, had an accuracy of 0.86, and was outperformed by all the other algorithms in this configuration. The best results here were obtained by the two ensemble tree models, both with an accuracy of around 0.95. It is interesting to look at the neural network results, that with accuracies between 0.9 and 0.93 were all outperformed by some of the previous machine learning models; their accuracies while training instead, so on the train set, were way higher than those indicated in the plot: this mean that the deep learning models probably overfitted our data.



The second plot shows the results in the configuration of only a subset of variables. Here, the overall results were not so much in a different range compared to the previous situation. The logistic regression was still the worst, with an accuracy of 0.85. The best results were those obtained by the random forest, with accuracy of 0.941, immediately followed by the first neural network, with 0.94 accuracy. In general, the three neural networks had a small improvement using less variables that led to a reduction in the variance of the model. Here, also the other simpler model, the decision tree, reached almost a 0.92 accuracy.

The third plot shows the result in the PCA configuration, that as expected, was not the best choice for our problem, since the overall accuracies were lower. Here, again the logistic regression was the worst, while the best result was obtained by the support vector machine, with an accuracy of 0.873, this was probably because the grid search allowed it to adapt to only two orthogonal variables. Here also the decision tree and the last neural network obtained quite good results with respect to the others, but still nothing impressive from an accuracy level perspective.





CONCLUSIONS

Our project can be concluded by stating that among the algorithms we tried, the best result is obtained by the Boosting in the first training configuration. Of course, this might be due to the fact that ensemble models can improve by bagging data and computing the mean of the results obtained with the different trees implemented by the model. It was interesting seeing that neural networks were outperformed by at least one of the machine learning models in every case and this confirms the fact that in order to perform well they need huge amounts of data and only a subset of variables to work with or they will likely lead to the overfitting phenomenon. As we know deep learning is not always the best solution, especially if we are looking for a solution that is interpretable and need fewer computing resources.

Our work could be improved with more instances to work with and, in a future perspective could be adapted to online learning with continuous flow of data but this would obviously require the proper

data infrastructure for storage and processing, and a constant monitoring of the performances of the models in case that bad data is fed to them.