Final Course Project: Predicting Credit Card Default

14.04.2021
by Tom K. Walter

"In order to increase market share, card-issuing banks in Taiwan over-issued cash and credit cards to unqualified applicants. At the same time, most cardholders, irrespective of their repayment ability, overused credit cards for consumption and accumulated heavy credit card debts. The crisis caused [a] blow to consumer finance confidence and it is a big challenge for both banks and card-holders."
I-Cheng and Che-hui, 2009.[1]

1. Dataset & Goal

For my project, I have chosen the Taiwan Credit Card Default Dataset, which was originally aggregated and analyzed by I-Cheng Yeh and Che-hui Lien in their paper *"The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients"* in the journal Expert Systems with Applications (2009).[2] The dataset is available on the UCI Machine Learning Repository.[3] It contains 30,000 observations, which correspond to clients that have been issued credit cards, and 25 columns that provide demographic information and the payment history of each client. The demographic information include marital status, sex, age, and education level. The payment record variables track the timely payment (PAY) over the last 6 months, the amount (BILL_AMT) due for each of the 6 months, and the amount paid (PAY_AMT) for each of the 6 months. The target variable, which I have renamed into DEFAULT, indicates whether or not a client has defaulted in the month following the observed time period. Out of 30,000 credit card holders, 23,272 have continued to pay off their bill and 6,625 did default. This makes it an unbalanced dataset. For more details on these variables see Table 1.

Table 1:

Name	**Description**	**Type**
ID	index number assigned to each customer	(useless for ML)
LIMIT_BAL	amount of the given credit in New Taiwan Dollar (NT$)	continuous
GENDER	gender (1= male; 2= female)	categorical
EDUCATION	level of education (1= graduate school; 2= university; 3= high school; 4= others)	categorical
MARRIAGE	marital status (1= married; 2= single; 3= others)	categorical
AGE	age in years	continuous
PAY_n	history of past payment; *n* tracks the 6 months of payment record (from April to September, 2005); the repayment status is: -1= pay duly, 1= payment delay for one month, 2= payment delay for two months, etc.	ordinal
BILL_AMT_n	amount of the bill in NT$ (*n* tracks the 6 months)	continuous
PAY_AMT_n	amount paid in NT$ (*n* tracks the 6 months)	continuous

[1] Yeh, I-Cheng, and Che-hui Lien. "The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients." Expert Systems with Applications 36.2 (2009): 2473-2480.
[2] Ibid.
[3] https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients.

DEFAULT	whether or not client has defaulted (Yes= 1, No= 0)	binary target

The research task associated with the dataset is to predict whether or not clients will default on their credit card, and to gain a better understanding of credible or non-credible clients in the future. In order to complete this task, I will first develop a Random Forest classifier with Python's Sklearn module.[4] This will serve as a baseline in terms of classification performance to compare the Neural Networks against that I will implement with the Keras module.[5] Particularly, I will develop a simple, single hidden layer Neural Network as well as a deep Neural Network with a set of hyperparameters for optimization. Ideally, this will elucidate whether more or less complexity is needed when modelling the classification of (non-)credible clients.

However, before modelling, I will briefly address the preprocessing step. Then, I intend to downsample in order to balance the dataset. Moreover, I will also engage in an exploratory analysis to ensure that the downsampled dataset accurately reflects on information provided in the original dataset (otherwise it would be useless for modelling). Lastly, after the building and optimizing the ML and DL models, I will evaluate and compare their performances.

2. Preprocessing, Balancing, & EDA

2.1 Data Cleaning and Preprocessing

To prepare a dataset for ML analysis, it must be checked for duplicate and missing values. All variables must also be encoded and normalized to be ready for the algorithm. The ID column has been dropped because it does not contain relevant information for modelling.

Given the large size of the dataset, missing and duplicate values have been dropped. 35 duplicate values have been dropped and only their original instances have been kept. For the categorical variables EDUCATION and MARRIAGE, I have found additional values to those mentioned in Table 1's description. I am assuming that a 0 stands for missing entries and thus I have dropped 68 instances where EDUCATION or MARRIAGE were equal to 0. Further, I assume that the values 5 and 6 represent further education or degree types that are just not mentioned in the description. So, they have been kept. Similarly, the columns PAY_*n* contained the value -2, which I assume also stands for *very* punctual payment given the ordinal nature of the variable. Therefore, I have kept them in the dataset. I have documented this here because neither the dataset's description on the UCI website nor I-Cheng's and Che-hui's original paper provide any information about this.

The categorical variables EDUCATION and MARRIAGE have been encoded as dummies. For the variable GENDER, I have replaced the values to 0 for male and 1 for female. For the Random Forest model, the dataset will not be normalized. Given the presence of ordinal and dummy features, I have chosen min-max scaling as normalization for the Neural Networks.

2.2 Balancing the Dataset

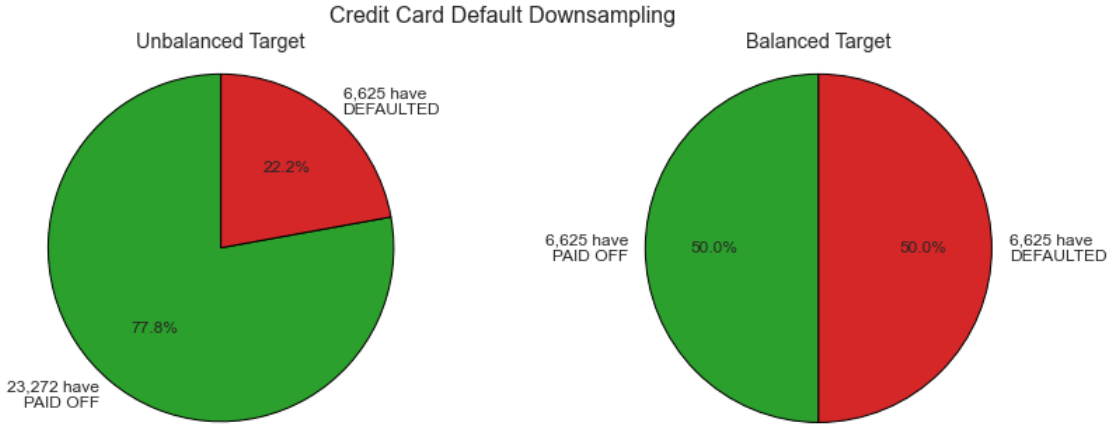
The classification of "rare events", e.g. disease occurrence, anomaly detection, accident prediction, or credit card default, is a special type of classification problem as its datasets are often highly imbalanced. This means that the occurrence of a rare event are the minority of samples and the non-occurrence are the majority of samples. The same issue is given in the Taiwan Credit Card Default dataset, where 77.8% of cardholders did not default (PAID OFF) and 22.8% did (DEFAULT) as illustrated in Figure 1.

---

[4] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.
[5] https://keras.io/api/models/sequential/.

This is an issue because most ML algorithms for classification expect the target to be balanced. A common method of dealing with unbalanced classes is resampling, i.e. to balance the target by either downsampling the majority class or by upsampling the minority class. Another method is to only use algorithms that can adjust the weight given to each class. As mentioned before, I am choosing to downsample so that PAID OFF and DEFAULT match 1:1 as shown in Figure 1. However, the downsampled dataset has the potential risk to no longer reflect the information given in the original dataset such as the distribution of the input features or their relationship with the target. The downsampled dataset has a total 13,250 samples, which will be split into a training set with 9,937 (~75%) and a hold-out set with 3,313 rows of observations (~25%).

Figure 1:



2.3 Exploratory Data Analysis

To ensure that the downsampled dataset has retained the same information as the original dataset, I have compared distributions, correlations, and balances of selected variables, compiled here in a set of visual representations. Although one can hardly expect the exact same characteristics between the original and downsampled dataset, similar ones indicate that it can be used for modelling. If the distributions are too different, it means the downsampled dataset does not accurately mirror the underlying relation between input and output, and should not be used for modelling.

Figure 2 shows the relative and absolute share of male and female credit card holders in the original and downsampled dataset. Especially, the relative share of ~60% female and ~40% male holders illustrate that the downsampled dataset closely reflects the original dataset.

Figure 2:

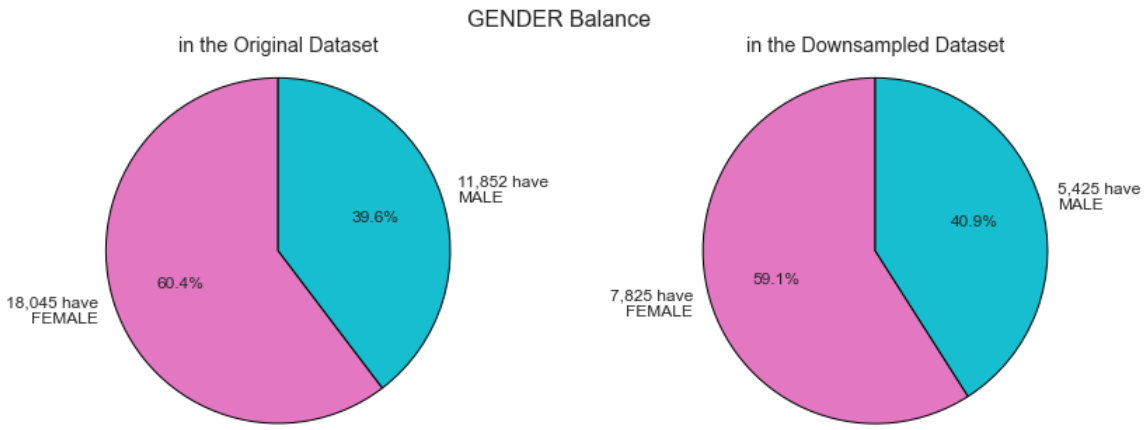


Figure 3 shows the correlation of all input features with the target variable across both datasets. Although the size of coefficient is slightly stronger in most cases, all downsampled features have a similar Pearson correlation coefficient compared to the ones in the original dataset.

Figure 3:

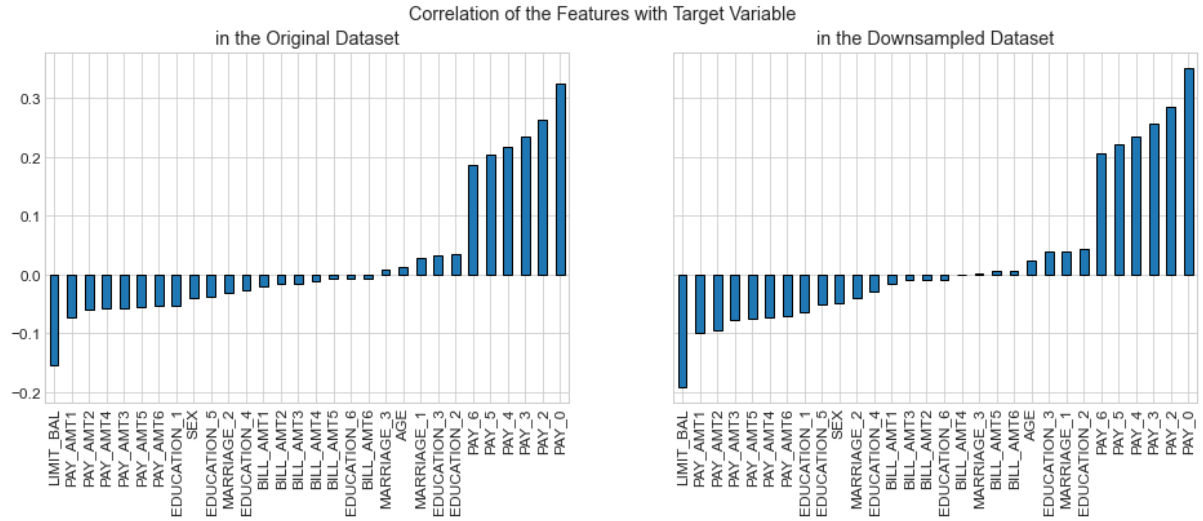
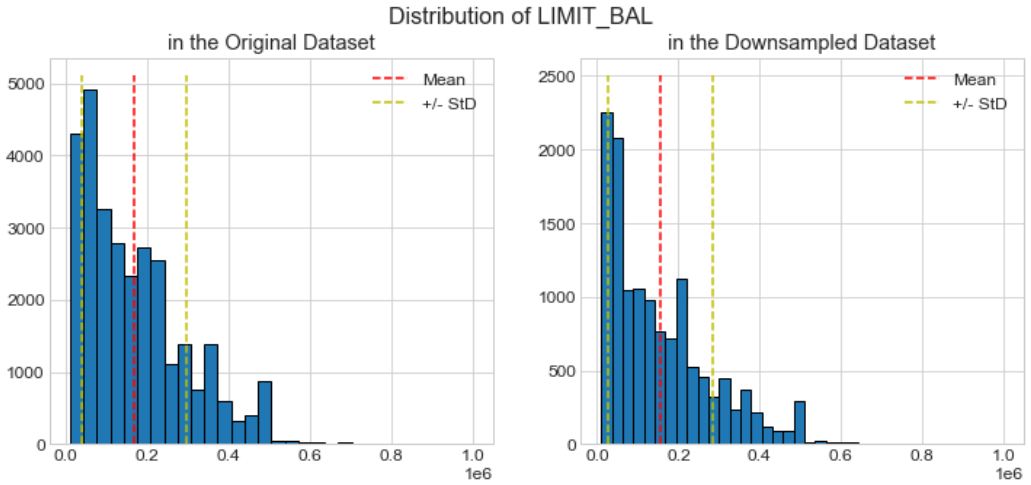


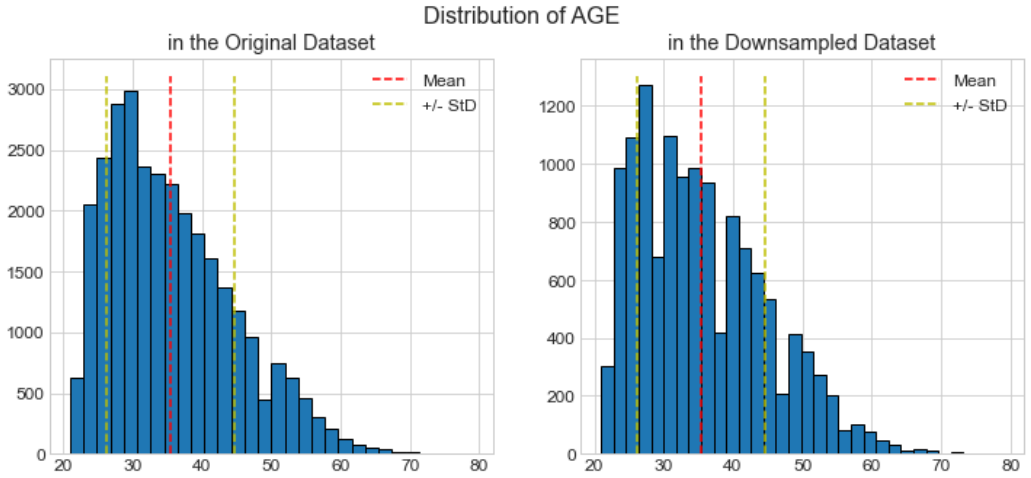
Figure 4 shows a distribution histogram of the credit amount given to the cardholders. Both distributions are skewed to the right, which means that most credit card holders had a low credit limit relative to all cardholders. As the x-axis are on the same scale, the fact that there is only a minor discrepancy between original and downsampled dataset is confirmed by the same positions of the means and standard deviations of both distributions.

Figure 4:



The distribution of the cardholders' age is displayed in Figure 5. Although AGE is a bit more normally distributed than LIMIT_BAL, it is also right skewed in both the original and the downsampled dataset. A cursory observation confirms that ~68% of cardholders are between the age of 25 and 45 in both datasets (i.e. within one standard deviation from the mean).

Figure 5:



For my in-depth analysis, I have checked more features, but these figures prove the point. Therefore, I can conclude that the downsampled dataset reflects the original dataset well enough to be used for modelling.

3. Random Forest as Baseline

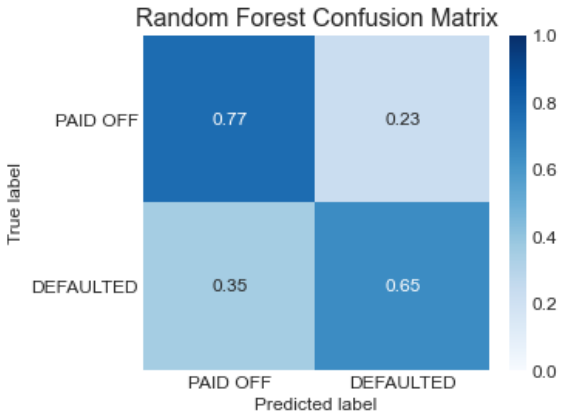
Since Neural Networks as DL models are able to find complex, non-linear decision boundaries, a Random Forest classifier will be a good ML runner-up in terms of complexity for fitting and also hyperparameters. Therefore, a Random Forest classifier is employed here to obtain a baseline of classification metrics for later comparison with the Neural Networks. The following range of hyperparameters have been used for the Random Forest classifier's optimization with GridSearchCV and 5-fold cross validation:[6]

- number of estimators/ trees: [10, 50, 100, 200, 300],
- maximum number of features: [2, 3, 4, 5, 6, 8, 10, 20, 30].

This optimization has determined 200 estimators and 2 features, also preventing overfitting.

A final test on the hold-out set has found that 77% of PAID OFF and 65% of DEFAULTED cases have been correctly identified as showcased in Figure 6. But a confusion matrix is not the only metric important for classification of unbalanced problems. I have listed further classification metrics obtained from each classifier on the hold-out set in Table 2 and will discuss them in section 6.

Figure 6:



[6] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

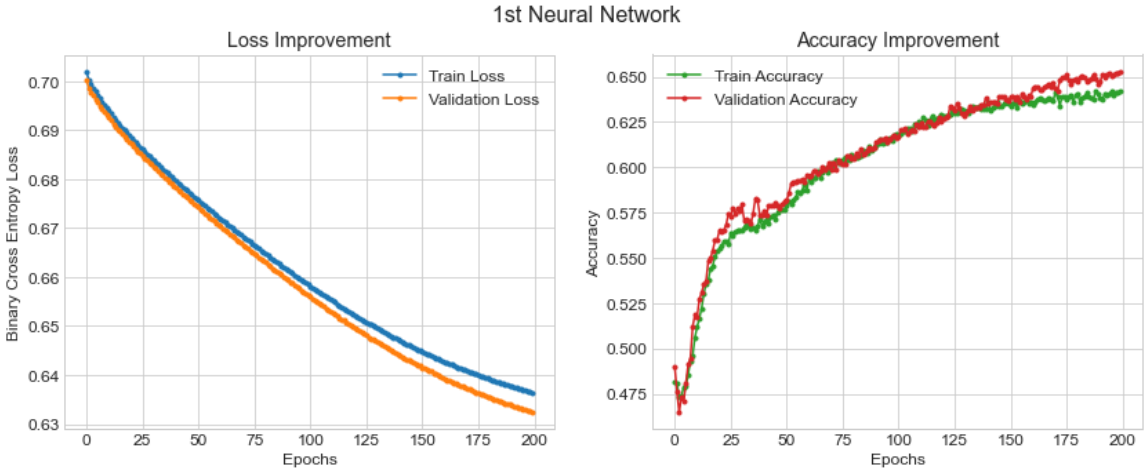
4. Simple Neural Network

As a starting point for Neural Networks (NN), I have built a single hidden-layer model with the Keras Sequential API in order to understand if a simple NN would already perform better than the Random Forest. Since the training and implementation of NNs are computationally intensive and thus can also be financially expensive (compared to ML models), it is important to test how complex a NN model really needs to be. The following hyperparameters define the simple NN:

- number of hidden layers: 1,
- number of neurons per layer: 15,
- activation: sigmoid,
- optimizer: Stochastic Gradient Descent (SGD),
- batch size: 30,
- epochs: 200.

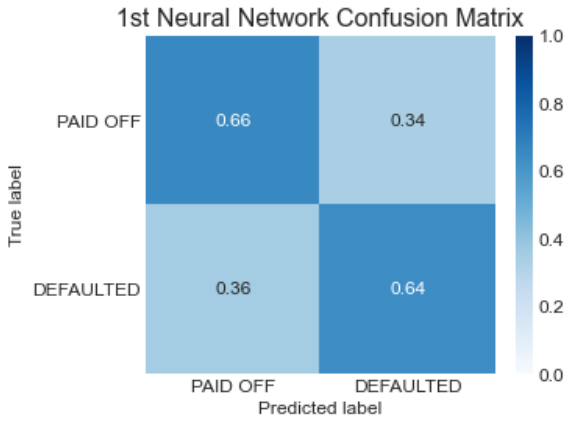
As Figure 7 illustrates, loss and accuracy scores improve only within a small range and do not seem to run into diminishing returns at 200 epochs. Other classification metrics are listed in Table 2.

Figure 7:



Moreover, Figure 8 showcases that the first NN did perform worse on the hold-out set than the Random Forest. Only 66% of PAID OFF cases and 64% of DEFAULTED cases were predicted correctly.

Figure 8:



5. Optimizing Neural Network Hyperparameters

Since these results for the first NN are far from optimal to be used in a business environment where the decision of issuing credit cards to financially responsible clients is dependent on such a model, my next step is to define a range of hyperparameters and optimize a second NN. The optimization of NN hyperparameters in a Keras model can also be achieved with Sklearn's GridSearchCV function.[7] I have set the following range of hyperparameters for the second Keras classifier's optimization with GridSearchCV and 5-fold cross validation:

- number of hidden layers: [1, 2, 3, 4],
- number of neurons per layer: [15, 20, 30],
- activation function: [sigmoid],
- optimizers: [SGD, RMSprop, Adam],
- batch size: [30, 60, 120],
- epochs: [50, 100, 200].

Of course, there are more potential hyperparameters for optimization such as learning rate, other activation functions, kernel initialization, etc. but each additional value drives up the permutations and thus explodes the computation time and resources. This optimization has determined the best hyperparameters to be:

- number of hidden layers: 3,
- number of neurons per layer: 30,
- activation: sigmoid,
- optimizer: Adam,
- batch size: 30,
- epochs: 200.

Below, Figure 9 illustrates the training process of the second, optimized NN, where the loss makes the largest improvement within the first 25 epochs but accuracy only starts to have diminishing returns after 125 epochs. This contrasts the shallow NN, which has not reached any inflection point where these metrics would plateau. Additionally, with just 2 more hidden layers, the deep NN has become less prone to misclassifying PAID OFF cases of which 72% in the hold-out set were correctly predicted. Yet, classifying DEFAULT cases has improved only by 5% as Figure 10 demonstrates. Nonetheless, Figure 10 attests the best confusion matrix of all algorithms to the deep NN.

[7] https://machinelearningmastery.com/grid-search-hyperparameters-deep-learning-models-python-keras/.

Figure 9:

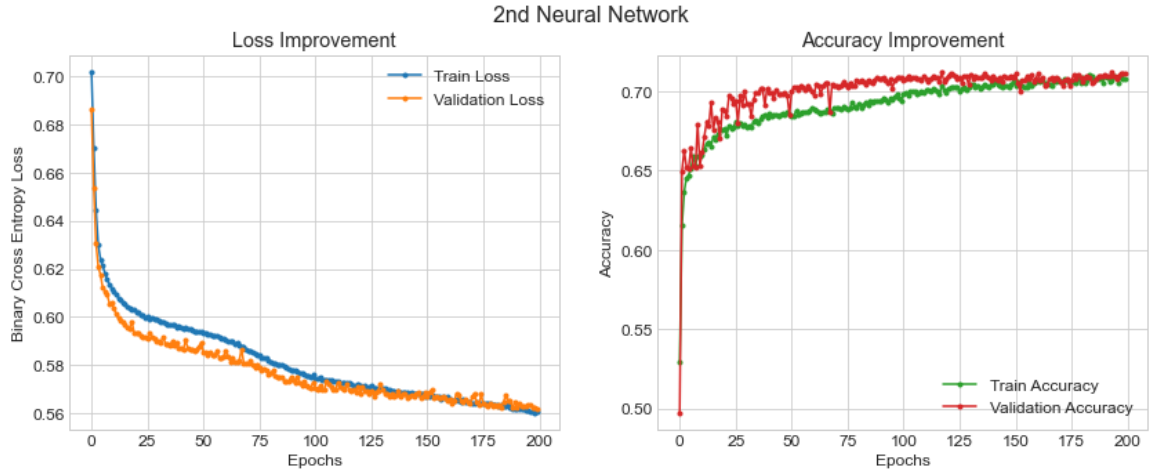
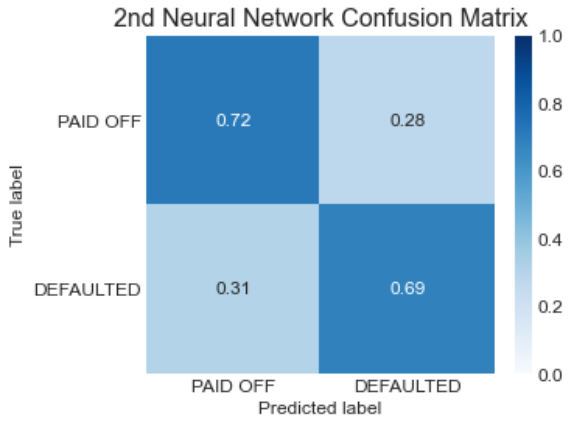


Figure 10:



6. Final Evaluation & Remarks

The ML and DL models developed for this project of predicting credit card default have attained differing classification performance scores as summarized in Table 2. Usually, downsampling leads to a higher recall but a lower precision score as it gives more weight to the minority class. However, for this downsampled dataset precision remains higher than recall for all models, which means they are still giving more weight to the majority class. This is why the F1-score is important when the balance of classes is an issue as it describes the harmonic mean of precision and recall.

Among all models the simple NN has performed worst on all metrics, even worse than the Random Forest classifier, which has set a relatively high performance baseline for the NNs to achieve (cf. Table 2). In fact, the Random Forest classifier even beats the deep NN in terms of recall and precision. Only a close comparison of the confusion matrices of the Random Forest (Figure 6) and the deep NN (Figure 10) reveals that the latter has actually a much better classifying performance than Table 2 would suggest.

As the deep NN is the winner in this close competition with the Random Forest, I would recommend using a deep NN model as learning on new data, in the long-term, will be better handled by NNs. As the amount of DEFAULT cases may decline in the future and as the concomitant nature of the dataset may change, it is also important to consider weighting the classes. This weighting of DEFAULT over

PAID OFF cases for classification may assist in maintaining an accurate performance in NNs, even as samples on DEFAULT cases become sparse.

For future analysis, it may also be helpful to have features, i.e. data collected on the amount of monthly income and monthly personal expenses, which each credit cardholder has. For both machines and humans this would be an easy-to-understand indicator showing whether or not clients are prone to overspending (and thus spiralling into debt). But, as I-Cheng and Che-hui wrote, this is probably missing because banks over-issued credit cards without fully checking the credibility of applicants. To prevent another credit card debt crisis, checking the credibility of applicants with deep learning is a crucial step, but in turn, the accuracy of these models should be continuously monitored too.

Table 2:

	Random Forest	**Simple NN**	**Deep NN**
Accuracy	0.7081	0.6529	0.7111
Precision	0.7331	0.6523	0.7522
Recall	0.6480	0.6444	0.6237
F1 Score	0.6880	0.6483	0.6820