IT1244 Project - Credit Card Application Enhancement

Yeo Hiong Wu; Teo Ren Jun, Lincoln; Lee Zhan Peng; Jennifer Chue Jing Wen

e0726132@u.nus.edu; teo_lincoln@u.nus.edu; leezp@u.nus.edu; jenniferchue@u.nus.edu

Introduction

In the current financial context, credit scores are utilised by institutions to gauge the likelihood of a consumer's ability to return loans. The scores are generated by credit bureaus and are based on the consumer's financial history. Unfortunately, individuals that do not use credits are shortchanged since they have no credit transactional history (Clements, 2015). As a result, banks are unable to make a decisive choice whether to approve their credit card application.

As such, this project aims to tackle the problem using different Machine Learning techniques [Linear Regression (LR), Logistic Regression (LogR), Neural Network (NN), Decision Tree (DT), Random Forest (RF)] on the Credit Card Approval Dataset by generating a score that banks can refer to, using non-financial personal attributes of clients such as age & job type, on top of their usual financial information. In addition, we hope to determine the feasibility of creating such model and evaluate its pros and cons.

Related Analysis

There have been analyses conducted by the public using similar datasets to classify credit card applicants as "good" or "bad". The figure below shows some of the different models used and their respective accuracy records.

References	LR[1]	DT[2]	RF[3]	LGBM[4]	KNN[5]
Seanny, 2020	0.61	0.83	0.85	0.90	-
Suman, 2022	0.50	1	1	-	0.99
Khujat, 2021	0.53	0.74	0.75	-	0.44
Hisham, 2022	0.89	0.99	0.99	0.99	0.94

Table: Results of analyses by multiple Kaggle users

As seen, the results of the analyses vary. In terms of project structure, none is seen to make full use of the data. As only 10% of the dataset is labelled, the 90% unlabelled data were completely neglected. To add, their models only classified applicants. If scores were generated instead, classification could be more flexible with varying thresholds.

Dataset Preprocessing & Cleaning

For the application dataset, binary attributes are adjusted to '1' and '0', categorical attributes are encoded with One-Hot Encoding (OHE), and continuous attributes are normalised to maintain all attribute scales to between 0 and 1. In addition, the 'own_mobile' attribute is dropped as it is invariant.

There are no explicit y-labels in the dataset. To generate our scores between 0 and 1 as y-labels, we proportionated our scoring model, with 70% influenced by the credit record status and 30% from total record count **(ANNEX A)**. Our base score used is 0.35, and a calculated score higher than 0.35 would imply that the client had good record history.

Our dataset now consists of 36457 (~10%) labelled and 402100 (~90%) unlabelled data. We further split it into 2% testing and 8% training data (labelled), and 3 x 30% unlabelled data in preparation for our intended learning methods.

Method and Modelling (Regression)

With only 10% of labelled data, we used Semi-Supervised Learning (SSL)[6] **(ANNEX B)** for our models to combat the overratio of unlabelled to labelled data (Brownlee, 2020).

We decided to deploy LN and NN within our SSL method to regress approval scores for applicants.

As one of the only few regression models available, **LN** is used due to its simplicity in implementation and its quick

[1] Logistic Regression
[2] Decision Tree
[3] Random Forest
[4] LightGBM
[5] K-Nearest Neighbour
[6] SSL is an approach to ML that combines small amount of labelled data with large amounts of unlabelled data during training.

computation. This can quickly give us an idea of the nature of our dataset and its relation with the credit scores.

With many attributes in the dataset generated from OHE, we believe that **NN** will be useful in capturing non-linear relationships that may exist between them and the scores.

Mean squared error (MSE) is then used for the evaluation of the model for both techniques using the testing data.

Results of Regression

We expected the MSE to drop after every training in SSL. However, the MSE did not decrease for both model and maintained at 0.01 **(ANNEX C)**. SSL seemed ineffective in reducing the MSE using pseudo-labelled data.

For further verification, we ran a more sophisticated NN model solely on the labelled data, and it performed better than when SSL was deployed.

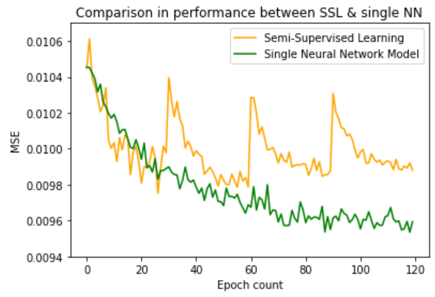


Fig: Change in MSE between SSL & single NN model at every epoch

We believe that the inadequacy in SSL comes from overfitting. Since all pseudo-labelled data were used in future iterations, it may have impaired the learning of the model.

Apart from the effectiveness of the learning method, the MSE value of 0.01 implies that the modelling of credit score using the given dataset may not be a feasible idea. On average, the error between predicted and actual score is roughly √(0.01) = **0.1** (10% of the full scale). As such, the classification of clients may have been a better approach.

Modelling (Classification) & Data adjustment

Since classification may instead yield a better result, we proceed to implement LogR, DT, RF & NN for our classification model.

Similar to LR, the easy implementation of **LogR** makes it an ideal classification model to begin with for our multiple modelling. We believe that **DT & RF** will be good models to handle our dataset which contains a large amount of categorical data. Lastly, as mentioned previously, since **NN** is useful for capturing any non-linear relationships between attributes and score labels, it was implemented as well.

Before classification, we converted our scores(y-labels) to ‘0’ & ‘1’. As 0.35 was our base score, we used it as threshold and set scores above it as ‘1’ and ‘0’ otherwise. As there would be an imbalance in the number of ‘1’s and ‘0’s, there is a need to stablilise both classes of data (Microsoft, n.d.). We utilised SMOTE from imblearn library to oversample our minority class **(ANNEX D)**. Proceeding from there, SSL was utilised, but in this case only pseudo-labels that are above 0.99 and below 0.01 were added to the pool of training data. Naturally, they were converted to ‘0’s and ‘1’s before training.

Results of Classification

	Trng 1	Trng 2	Trng 3	Trng 4	Non-SSL
LogR	0.507	0.502	0.500	0.490	0.507
DT	0.905	0.905	0.904	0.905	0.905
RF	0.913	0.916	0.916	0.915	0.913
NN	0.818	0.752	0.727	0.719	0.864

Table: Accuracy of trained models on test data

Like previously, the use of SSL has deteriorated the accuracy of models (in RED). This made its results unreliable for comparison. Instead, from the results generated purely from labelled data (in GREEN), it is noted that RF has performed the best amongst all. Apart from being the most accurate, its high accuracy of 0.913 presents a positive light towards the use of non-financial attributes in influencing credit card application.

Areas for improvement

Unfortunately, we were insufficiently tactful in identifying areas for further adjustments to make SSL work. In particular, we could have fine-tuned our NN model better if given more time. For instance, using more “Dropout” layers **(ANNEX E)** in Keras to reduce overfitting, or just in general spending more time to find the best permutation of layers for modelling. Apart from that, we could have improved our data preprocessing by reducing its dimensions with Principal Component Analysis (PCA). Lastly, as we generated our own scores (y-values), it could have had flaws interpreting the credit records. More research could have been done in creating our scores.

Conclusion

While generating scores that banks can refer to may be idealistic, it did not work as well as our classification models. In terms of the technical skills, we were enlightened with the exposure to the details of machine learning that were not covered in the module, such as the importance of oversampling the minority class with SMOTE to prevent data imbalance, and also the use of SSL to combat the little labelled data we have. Nevertheless, it is a relief that some models were able to perform, which shows a possibility of non-financial data being leveraged and used at a global financial level in the near future.

References

- **Clements, N. (2015, April 1).** 3 Problems With The New FICO Credit Score, from https://www.forbes.com/sites/nickclements/2015/04/01/3-problems-with-the-new-fico-score/?sh=1f4cabe16ff2
- **Vinaykhujat. (2021, June 5).** *Credit card approval prediction*. Kaggle, from https://www.kaggle.com/code/vinaykhujat/credit-card-approval-prediction
- **saurav12suman. (2022, October 6).** *Credit_card_approval*. Kaggle, from https://www.kaggle.com/code/saurav12suman/credit-card-approval
- **omarhisham19. (2022, February 9).** *Credit card approval prediction*. Kaggle, from https://www.kaggle.com/code/omarhisham19/credit-card-approval-prediction
- **Seanny. (2020, November 14).** *Credit Card Approval Prediction using ML*. Kaggle, from https://www.kaggle.com/code/rikdifos/credit-card-approval-prediction-using-ml
- **Brownlee, J. (2020, December 16).** *What is semi-supervised learning*. Machine Learning Mastery, from https://machinelearningmastery.com/what-is-semi-supervised-learning/
- **Microsoft. (n.d.).** *Smote - Azure Machine Learning*. Azure Machine Learning | Microsoft Learn, from https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/smote

Appendix

A: Generated score for y-label

Previously mentioned, 70% of score is influenced by credit record status ('X', 'C', '0', '1-5') and 30% by total record counts.

How are scores influenced based on every level of credit record status? (Note that 'X' is neutral):

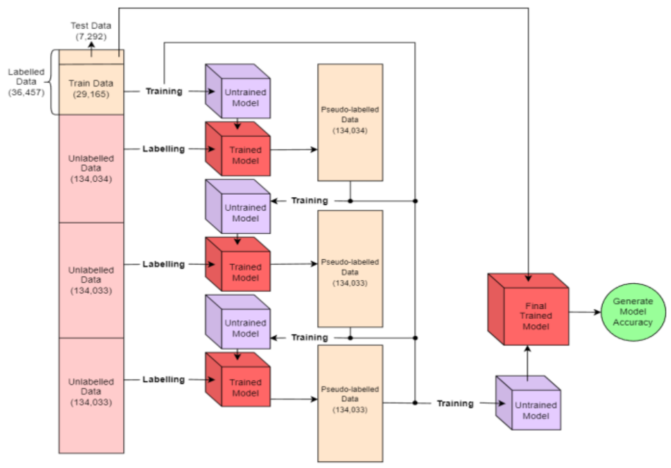
Current Score	C, 0	1	2	3	4	5
>0.35	+0.00625	-10% of 70%	-15% of 70%	-20% of 70%	-25% of 70%	-30% of 70%
<=0.35	+0.005					

How about 30% of record counts?:
+0.00263 every record, only begins after 6 months of records, stops increasing after 10 years of records.

E.g. Client A has score of **0.4** generated from 12 records (1 year), 0.38422 & 0.01578 contributed from record status and record counts respectively.
His 13th record is **'1'** → 0.38422 – 10% = 0.345798
Add record count → 0.01578 + 0.00263 = 0.01841
His updated score → 0.345798 + 0.01841 = **0.364208**

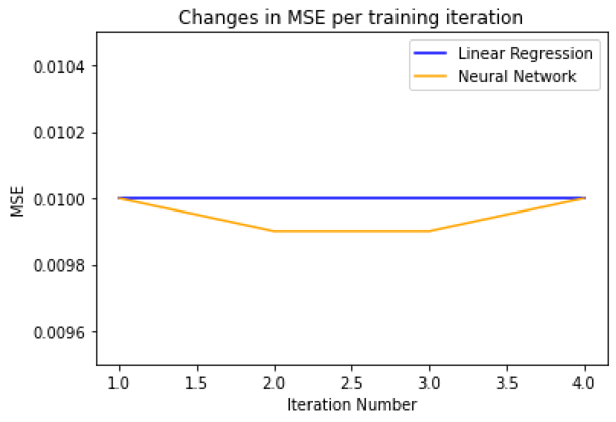
B: Structure of Semi-supervised learning

We first train our model using labelled data. We then use the model to pseudo-label our first batch of unlabelled data. These pseudo-labelled data is added to the pool of labelled data. Process repeats until all data is used. The final trained model is the model of use.



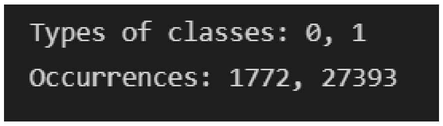
C: Changes in MSE at every SSL training iteration

No significant decrease in MSE is seen. SSL has been ineffective.



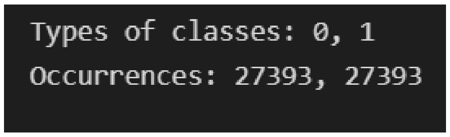
D: SMOTE library for oversampling of minority class

Before oversampling with SMOTE:



```
Types of classes: 0, 1
Occurrences: 1772, 27393
```

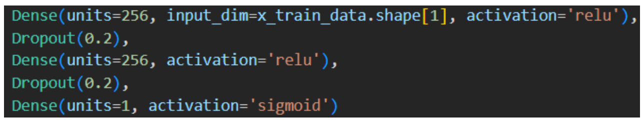
After oversampling with SMOTE:



```
Types of classes: 0, 1
Occurrences: 27393, 27393
```

E: Dropout layers in Keras Neural Network

Dropout layers serve to ignore a portion of the output from the previous layer, so as to train on data that are not similar at every epoch. This reduces overfitting to a certain extent.



```
Dense(units=256, input_dim=x_train_data.shape[1], activation='relu'),
Dropout(0.2),
Dense(units=256, activation='relu'),
Dropout(0.2),
Dense(units=1, activation='sigmoid')
```