PREDICTING CREDIT CARD DEFAULT

A PREPRINT

Group Name: G
Department of Computer Science
University College London
London, WC1E 6BT

January 11, 2022

1 Introduction

We decided to choose 4 main algorithms: Random Forest, AdaBoost Classifier, support vector machine(SVM), and feed-forward neural network(NN). Before data training, the data is preprocessed with reasonable methods such as one-hot encoding, resampling, missing data removal, etc. We also fully-researched the data to comprehensively understand some significant features and structures of data. Besides, we also used the principal component analysis to reveal the drawbacks and problems within the data and enhance the current computability of our model. After optimization, all approaches achieved an accuracy of approximately 80 % on the training data.

2 Data Transformation and Exploratory

The aim of this section is to provide a coherent overview in terms of the transformation applied to the provided credit card dataset prior to the training process. The main subject of this section includes data visualisation and feature selections, aiming to provide an accurate demonstration of the nature of the problem and the dependence between different features.

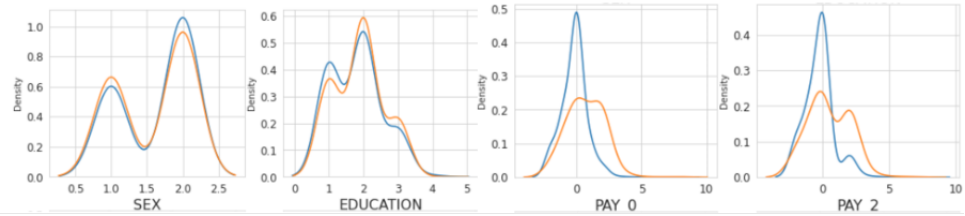
2.1 Data Visualisation

The first step of data manipulation is to perform a high-level analysis on the training data set, before diving straight into model training and construction. This allows us to gain a clear understanding in terms of the features included in the data set, the potential relationship between them and the approach to develop and train our model. To begin, we first load the provided training set using the pandas library – a fast and easy-to-use data analysis tool that help us to gain a rough overview of the training set, for example, the total number of rows and the values presented in each column. Additionally, this helps with the data cleaning process, in case when certain cells within the data frame are empty or include irrelevant values. For example, for the column 'EDUCATION', we excluded the rows with values equal to 0, 5 or 6 as they are not relevant to our research.

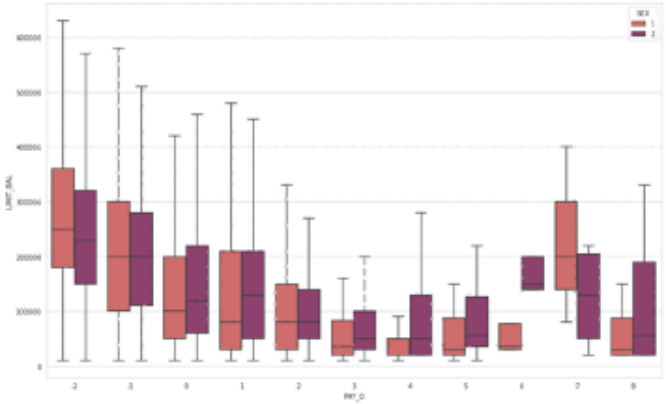
The next step of our evaluation is to create visualization based on the training set. Here, we load our pre-processed data into the h2o framework, a scalable open-source platform for machine learning for faster process purposes. After reviewing the categories of the data set with real-life examples for credit research, we decided to create visualizations using multiple plotting methods to better represent the relations between different variables. A few plots included in our research include kernel density estimate (KDE) plot, box plot, and correlation matrix.

Essentially, a KDE plot is a method for visualizing the distribution of observations in a data set, analogous to a histogram. It represents the data using a continuous probability density curve in one or more dimensions, which in our case help to show the relationship between the features. By applying the 'default payment per month' column as the base dimension, we plotted a KDE graph with each variable and were able to find the more representative attributes among all given columns. Particularly, SEX, EDUCATION and PAY shows a significant variation in curve in comparison to other

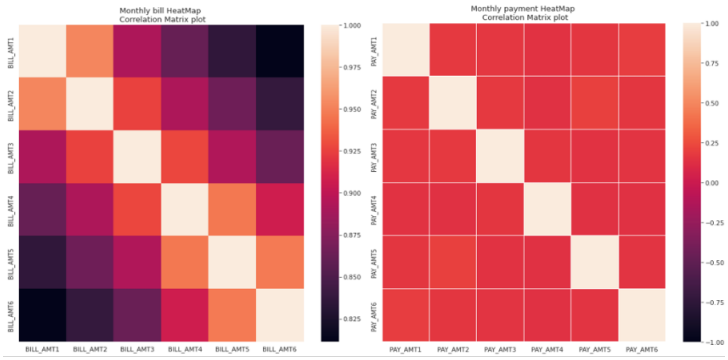
variables as shown below.



The second visualization tool used is box plot, which shows the distribution of quantitative data in a way that facilitates comparisons between variables or across levels of a categorical variable. The box shows the quartiles of the data set while the whiskers extend to show the rest of the distribution, except for points that are determined to be "outliers" using a method that is a function of the inter-quartile range. Here, we plotted the relationship between PAY and LIMIT_BAL, grouped by the variable SEX to reveal the potential difference based on users' sex.



Next, we created a correlation matrix based on the existing columns. A heat map is a data visualisation technique that shows magnitude of a phenomenon as colours in two-dimension space. Here in our research, the correlation matrix finds the relations between the variables by taking BILL_AMT and PAY_AMT as input and compare them with the users' default payment status. As shown in the figure below, there is a strong positive correlation between the BILL_AMT variable and the default payment status whereas the link between PAY_AMT is rather weak in comparison. It is worth mentioning that although some phenomenon could be found by creating the heat map, the analysis only gives a surface level conclusion. This led to our next steps of analysing the data by constructing a machine learning model and applying different methodologies.



2.2 One-hot-encoding

In many data files, features are not always continuous values and maybe categorical values. It is much more efficient if the categorical features are represented as numbers. However, after conversion to a numerical representation, the above data cannot be used directly in our classifier. This is because, by default, classifiers tend to have continuous and ordered

data. However, the numbers as represented above are not ordered, are randomly assigned. One way to solve the above problem is to use One-Hot Encoding. It can be understood that for each feature, if it has m possible values, then after the One-Hot Encoding, it becomes m binary features. And, these features are mutually exclusive, with only one activation at a time. As a result, the data becomes sparse. Also, it ensures that machine learning does not assume that higher numbers are more important. In our case, we basically consider those columns as Categorical Variable: 'SEX', 'EDUCATION', 'MARRIAGE','PAY_0', 'PAY_2', 'PAY_3', 'PAY_4', 'PAY_5', 'PAY_6' And we consider the rest of the columns as Ordinal Variable:'LIMIT_BAL','BILL_AMT1', 'BILL_AMT2','BILL_AMT3', 'BILL_AMT4', 'BILL_AMT5', 'BILL_AMT6', 'PAY_AMT1','PAY_AMT2', 'PAY_AMT3', 'PAY_AMT4', 'PAY_AMT5', 'PAY_AMT6'

Moreover, there existing the complexity trade-off when we considering the Categorical Variables such as variables 'PAY_0', 'PAY_2', 'PAY_3', 'PAY_4', 'PAY_5', 'PAY_6', there increasing huge running time after we consider these 6 variables into one-hot encoding. In our case with downsampling data size(default = 1750, no default= 1000) and default parameter value(C = 1.0, gamma = 0.01), we get the higher running time from 0.4 to 0.8 seconds when consider those variable into the one-hot encoding. But in order to build a more reasonable calculate, we finally decide to take consideration into these 6 variables into one-hot encoding.

3 Methodology Overview

3.1 Overall mechanism

The overall mechanism started with the observation of the data using different displaying techniques and checking the distribution and patterns of clients' features[1]. We first perform data reprocessing which includes removing invalid values, normalization of the data. Meanwhile, since some features are categorical and hence contribute less to the prediction result. The one-hot encoder is used to translate these features to numerical ones. Then PCA is applied in order to reduce strongly correlated features. After that we evaluated four classification models by comparing their accuracies and selected one for our final prediction. Different hyper-parameter techniques, such as backpropagation, greedy algorithm, GridsearchCV are utilized during the training.

3.2 Concepts used in the models

3.2.1 PCA

The main idea of Principle Component Analysis is transform the n-dimensional features to smaller dimensional features. The problem here is which features we should select. By doing this we explain the maximum amount of variance with the fewest number of principal components. The PCA mechanism computes the contribution to the variance of each feature by calculating their eigenvalues and then the corresponding eigenvectors. These vectors are synthesized to find the target features.

3.2.2 GridsearchCV

This function is imported from another sklearn library called *sklearn.model_selection*. Through this function, the model will be continuously looped until all possibilities have been computed and will then return the model with the highest level of accuracy depending on the specified options.

3.2.3 SelectKBest

SelectKBest is imported from the library named sklearn.feature_selection. SelectKBest. and it is refers to a dimensionality reduction method similar to PCA that uses k highest score as a way of feature selection. In comparison to PCA, SelectKbest uses the k parameter, which is the number of top features to select, which differs from the PCA's approach of using n_components.

In our case, we use this method in SVM part.

4 Model Training and Validation

Four algorithms have been trained and validated, the steps of training are elaborated below.

4.1 Feed Forward Neural Network

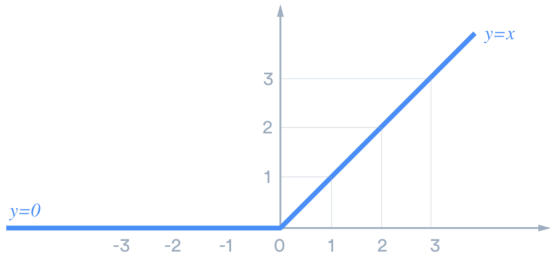
Before choosing the proper one among so many architectures of neural networks, we would like to understand the features' insights of the data and then put into the deep learning models.

We have 30000 rows of data and 23 features. The problem is formed as a binary classification problem, with client default=1 and no client default=0. We divide the features into two categories: Static Features and Dynamic Features, as some of the features are time correlated. Thus, we could choose RNN/LSTM/1D-CNN to extract latent from features. And in the end, I decided to choose 1D-CNN as our dataset is a combination of categorical and numeric data, and data of time series is missing, we don't need RNN to exhibit its dynamic behavior.

We use the following hyperparameters to extract time-correlated features from the dynamic features and then concatenate with the static features and feed into a DNN. Conv1d(input feature=3,output feature=16,kernel size = 2) Batch size = 1024 Epoch = 200 Learning Rate = 0.0005 Activation function = ReLU We start the initial attempt by feeding the data with batch size of 1024, 1 Convolution layer. ReLU[4] function is selected as the activation function between hidden layers.

$$y = \max(0, x)$$

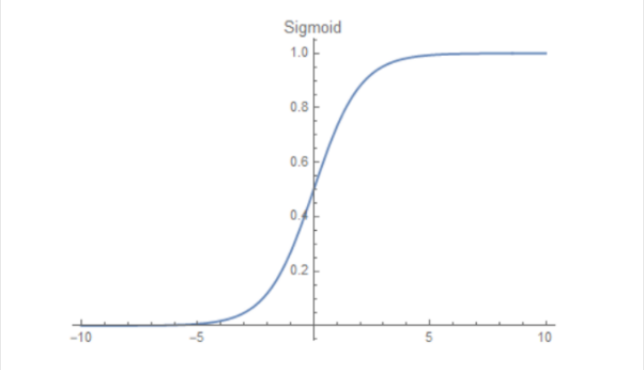
And the graph of ReLU function is:



And we choose sigmoid function as the activation function for the output layer:

$$f(x) = \frac{1}{1 + e^{-x}}$$

The graph of sigmoid function is:



After obtaining the predict results of our neural network model, we decided to use cross entropy loss to find the difference between the predicted result and labels.

During the training process, we used Adam instead of SGD, Adam is a combination of RMSprop and Stochastic Gradient Descent with momentum. Essentially Adam is an algorithm for gradient-based optimization of stochastic objective functions. It combines the advantages of two SGD extensions — Root Mean Square Propagation (RMSprop) and Adaptive Gradient Algorithm (AdaGrad) — and computes individual adaptive learning rates for different parameters.

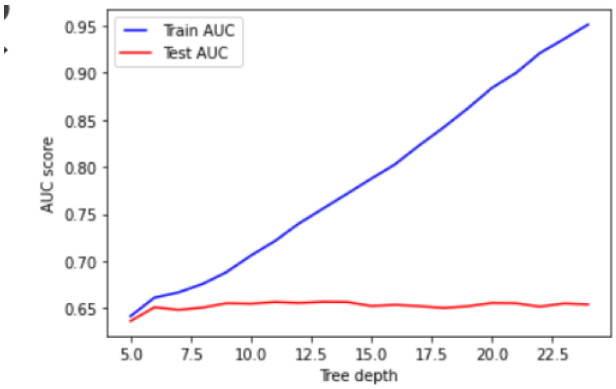
4.2 Random Forest

Random Forest algorithm (RF) is an ensemble learning methods, which is a collection of multiple decision trees. In order to classify an object, the algorithm grows many classification trees in the intermediate stage, and then theirs

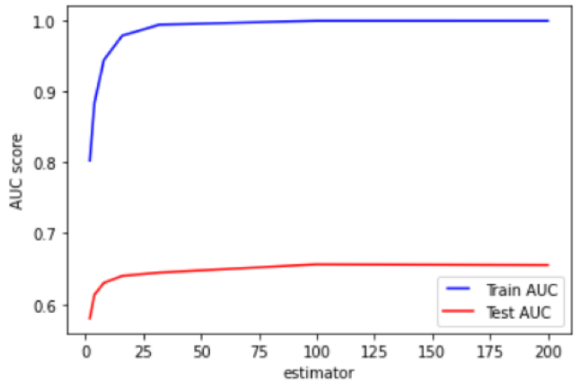
predictions are aggregated to identify the most voted result. During the learning, in order to the reduce the variance which prevents overfitting in an early stage, the bootstrap aggregation methods is applied: random samples and features are taken from the dataset with replacement. Hence the correlation among decision trees is restricted.

4.2.1 Finetuning Hyperparameters

The depth of each tree(max_depth), and the number of trees(n_estimators) are two key parameters of the random forest model. Although larger depth of tree and larger number of trees both make positive effect on the model, they only function in a small range. In other words, once the two parameters exceed their "sweet point", it has no significant performance gain but increase the computational cost.[5] This is also reflected by our experiment[figure 1] using greedy search. Our graphs have shown that the roc_auc scores of both parameters converge after a very small increase. We found that the maximum tree depth with 13 and number of estimators with 100 have the optimum performance in our experiment range.



The optimum maximum depth of the tree is 13.

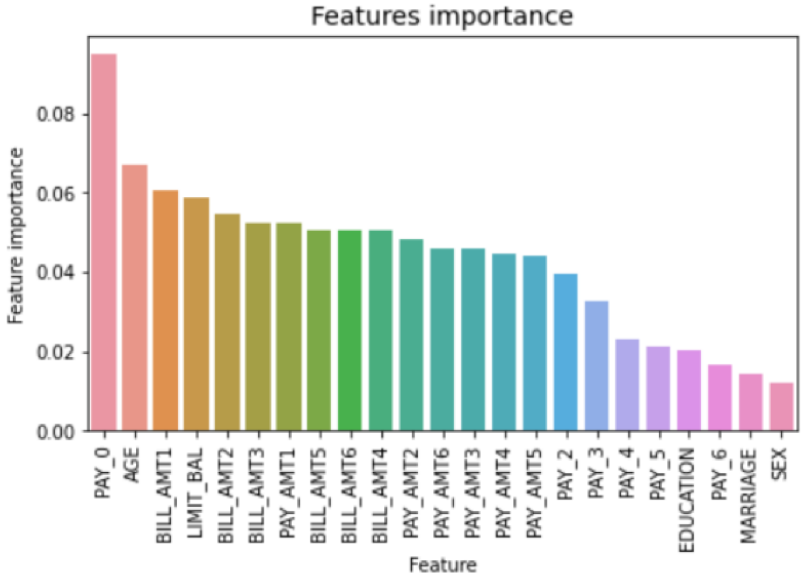


The optimum number of estimators of the tree is 100.

4.2.2 Feature Importance

We also plot the feature importance graph for dimensionality reduction.[Figure 2] The high-importance features are kept for training which reduces the computational cost. We experiment by selecting the top 15 features for a re-training

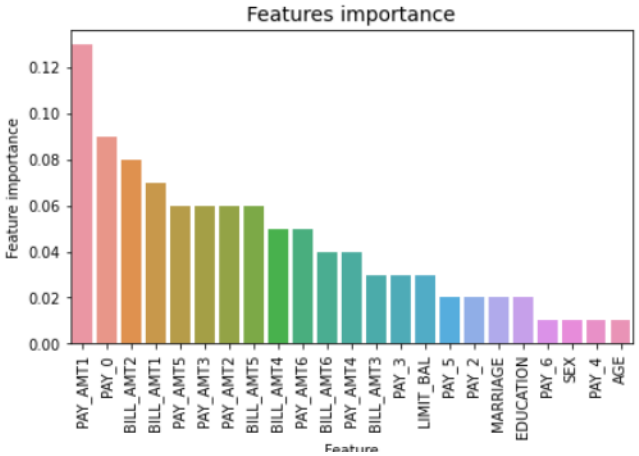
of the model. The accuracy is only decreased by 1.2% but we removed 10 relatively insignificant variables.



4.3 AdaBoost classifier

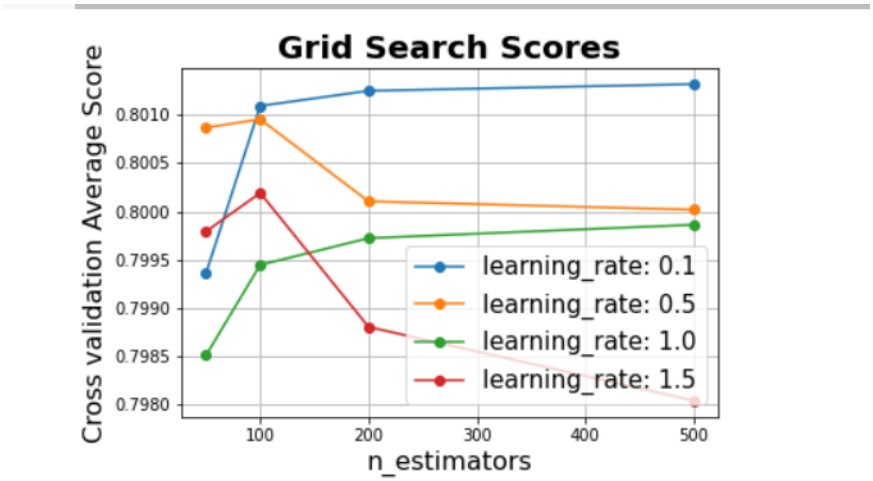
The AdaBoost algorithm is an integrated learning algorithm like Random Forest[2] , and the classifier of Boosting consists of multiple weak classifiers. Each weak classifier is trained in turn, and each weak classifier has weights. The weighted sum of the weak classifiers' predictions forms the final prediction result. During training, the training samples also have weights, which are dynamically adjusted during training. Samples that are misclassified by the previous weak classifiers are weighted more, so the algorithm focuses on the hard-to-classify samples.

The Adaboost classifier is implemented by *sklearn.ensemble.AdaBoostClassifier()*, and set its algorithm, number of classifiers, and learning rate. Then, as the overfitting problem often occurs in the training, the model is evaluated by averaging across the repeats run of classified k-folds cross-validation, then generating a cross-validation score.[3]Initially, we set the model with a learning rate of 1.0 and a number of classifiers of 100, the mean cross-validation score would be 0.8135.



As there are 23 features used as input data to the model, then we trying to reduce dimension for the features selection. Firstly, we generated a feature map shown above, which shows the relative importance of each feature to the outcome variable. This allows further reduction in the dimension of the data, as only the top two highest-scoring variables were then kept for training the algorithm. Then fit the model with the top two features after computation of PCA, the

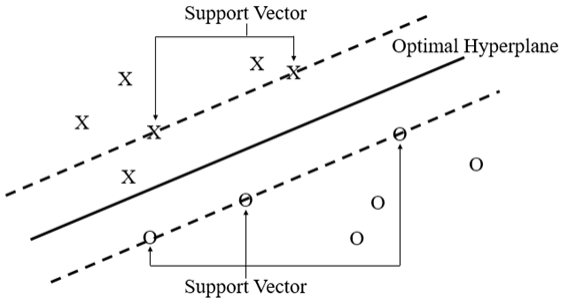
accuracy of the model only decreases by 1.27% and the validation score becomes lower than 80. Thus, for a small cost in accuracy and validation score, the number of features is decreased to only the top two left.



To optimize the results, the next step is tuning the parameters. In specific, we would like to investigate which combination of a number of classifiers and learning rate would result in the best performance of the model. As there is more than one hyperparameter in the model, GridSearchCV is considered an efficient method of tuning. The GridSearchCV function will loop the model until all the possibilities are achieved and return the model with the highest level of performance. The running process is listed in the Figure shown above. The best cross-validation score of the model is 0.8013 with parameter set{learning rate:0.1, n_estimators:500} and the accuracy becomes 81.56%.

4.4 Support Vector Machine Classifier

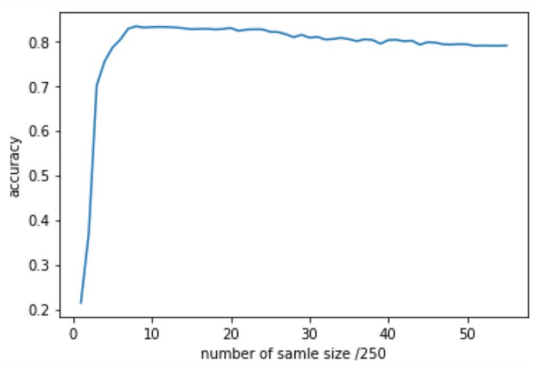
Support Vector Machine is a supervised machine learning algorithm that uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs. In the most case SVM has better performance than other the machine learning methods but is sensitive to parameter settings and training samples. For our model, SVM is imported from the sklearn.svm library.



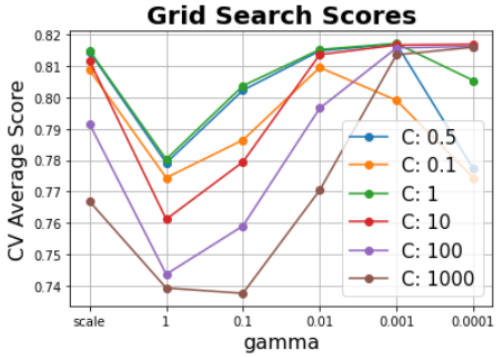
Specifically, The SVM is performed by finding the optimal hyperplane that categorizes training input data into to two classes. The machine conceptually implements the following idea: input data are non-linearly mapped to a high-dimension feature space. In this feature space a linear decision surface is constructed. Special properties of the decision surface ensures high generalization ability of the learning machine.

For our method SVM, it perform well for a data matrix with lots of features, and not too many instances. To fully understand the characteristic of SVM, which is it works better in a reasonable database, we basically use the method Data-Resampling to downsample our dataset into a smaller scale to find the reasonable scale of instances to fit in our SVM model. In an experiment, We make all parameters of SVM and the down constant. Due to the small amount of data in the default sample, the number of samples in the default sample was kept constant at 1000 in each cycle, while 250 non-default samples were added for training in each cycle. The final graph below shows that the accuracy peaks at 1750 non-default samples and 1000 default samples for training, and then remain stable and even decreases slightly as

the number of samples increases, below is the graphing showing the relationship between accuracy and desampling size.



After we desampling the data, we then using the Support Vector Machine Classifier imported from *sklearn.svm.SVC* . And We basically apply GridSearchCV on the SVM model in order to find out the suitable Regularization parameter 'C' and Kernel Coefficient 'gamma' value. Below are the Grid Search Scores graphing[8] to measure the different average test scores when using different values of 'C' and 'gamma'. Based on the following example graph we can conclude that the optimal 'C' is 1.0 and optimal 'gamma' is 0.001 when the desampling rate equals to 0.5, and finally we gain 83.27% accuracy by using this method.



Moreover, we also implement two dimensional reduction techniques which are SelectKBest and PCA(Principle Component analysis) to better represent the underlying problem existing in SVM model and improve the algorithm's learning capabilities for the data.

5 Results

	Random Forest	AdaBoost	SVM	NN
Accuracy on training data(%)	83.1	82.83	83.27	77.73

Table.1

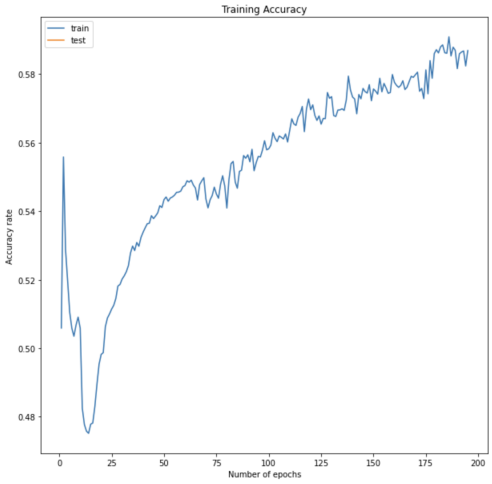
Each algorithm we chose was trained using the data from the provided file: CreditCardtrain.csv. The results are presented above in Table.1. The algorithms were compared in terms of complexity, reliability, and workload.

The testing data provided is tabular, and all four algorithms can process well. And neural network shows more reliable than the others. After the neural network learns from the learning data, neural network can then deduce more hidden relationships on unseen data, thus making the model generalization and predict on test data. From the table, we can see that Random Forest and SVM have the highest accuracy. However, the reliability ultimately increase the potential of neural network. It will produce a higher accuracy than other algorithms with a longer training time.

We could easily find out that neural network has the greatest workload, as neural network had the longest processing time, as the data must be transferred from categorical into numerical before feeding to the model. Random forest, SVM have far simpler and less time-consuming preprocessing than neural network.

5.1 Feed Forward Neural Network

The accuracy of the neural network:



The accuracy of model trained in Google colab reached 60%. However, the maximum accuracy could reach 78% when the model is trained with GPU and a longer time.

6 Final Predictions on Test Set

Both the random forest and the Adaboost model require tiny correlations between features. In our example, even though we have applied the PCA technique, features such as the history payments still preserve a strong correlation and hence are having negative effect in the training. Meanwhile, although Support Vector Machine provided the highest accuracy among these four models in our training process, it does not perform well when the training dataset contains more noise and more data. [7]

We finally select the neural network as our final technique in predicting the testing dataset. As shown by the accuracy-iterations graph posted in section 5.1, the accuracy of the neural network model keeps increasing with more iterations and more training dataset provided. The model is far away from being overfitting with our limited calculating resources and data. Apart from our work on jupyte notebook, we have trained our neural network model with more advanced GPU and more training epoches and the accuracy can reach over 85%. One small drawback of this model is that all features are considered equally during the training, and hence we cannot perform dimensionality reduction to reduce calculation.

7 Conclusion

In conclusion, neural network model will generate a more accurate and reliable result in the long term. As the intricacy of feed forward neural network is far greater and more sophisticated than the other three methods.

Besides of that, there are still many high-level algorithms worthy to use to predict the default of customers. There is a wide range of options to predict the customer default, and they could all produce similar results due to the training data is not overcomplicated.

References

[1] Sagar Tupkar. Predicting Credit Card Defaults using Machine Learning Algorithms In*https://www.slideshare.net/sagarvtupkar/predicting-credit-card-defaults-using-machine-learning-algorithms*

[2] Pan Jinquan. Adaboost algorithm concept explanation and code implementation. In *https://blog.csdn.net/guyuealian/article/details/70995333*.

[3] Srijani Chaudhury. Tuning of Adaboost with Computational Complexity. In *https://medium.com/@chaudhurysrijani/tuning-of-adaboost-with-computational-complexity-8727d01a9d20*.

[4] DanQing Liu. A Practical Guide to ReLU. In *https://medium.com/@danqing/a-practical-guide-to-relu-b83ca804f1f7*.

[5] Thais Mayumi Oshiro, Pedro Santoro Perez and José Augusto Baranauskas How Many Trees in a Random Forest? In *https://www.researchgate.net/publication/230766603_How_Many_Trees_in_a_Random_Forest*

[6] Neil Liberman Decision Trees and Random Forests In *https://towardsdatascience.com/decision-trees-and-random-forests-df0c3123f991*

[7] Dhiraj K Top 4 advantages and disadvantages of Support Vector Machine or SVM In *https://dhirajkumarblog.medium.com/top-4-advantages-and-disadvantages-of-support-vector-machine-or-svm-a3c06a2b107*

[8] David Alvarez How to graph grid scores from GridSearchCV? In *https://stackoverflow.com/questions/37161563/how-to-graph-grid-scores-from-gridsearchcv*