STAT 4850G/9850B FINAL PROJECT

Akila Balasubramaniam, Daniel Molson, Hwang Lee, Yanru Wang, Yufei Xia, and Yumeng Chen

3/31/2020

Abstract

Credit cards have been a massive success over the years for many banks since almost every client of a bank holds a credit card. Due to this, credit cards have become an essential part of a bank's profits. Banks are interested in whether or not a client is likely to default on their credit card payments. Identifying risky and non-risky customers has been the interest of many banks for years. In this paper, the factors that are needed to analyze the riskiness of a client will be determined. The article will also help banks to predict whether or not if a customer has the potential to repay the used credit of the bank by using various models. (the result was noted to be….)

Introduction

When a client uses the credit-card issued to them by the bank, the bank expects the client to pay back their credit loan with interests. However, not all clients can pay back their loans. Some cardholders overuse the credit-card for consumption, which leads them into massive credit debt. This was the issue that was facing Taiwan in 2006, where debt from credit cards and other loans reached $268 billion US, and people were struggling to repay their loans[1]. The Taiwan media called people who were struggling to pay back their loans the "credit card slaves" as they were struggling to pay even the minimum balance on their credit card debt every month[1]. This issue resulted in significant societal problems such as debtors committing suicide because of the debtor. Some became homeless[1]. To prevent this, banks use client's information to decide whether they will default in their payments. Identifying risky and non-risky customers has been the interest of many banks for years. In this article, the dataset containing information on default payments of clients in Taiwan from April 2005 to September 2005 will be studied. This data would be used to analyze what factors place a role in why a client will default on their payment as well to predict whether or not a client will default with given information.

There is an article by Cheng Yeh and Che-hui Lien, where the default payments in Taiwan were looked at and explored. In this article, six methods were looked at when analyzing the data; which are: K-nearest neighbor classifiers (KNN), Logistic regression (LR), Discriminant analysis (DA), Naive Bayesian classifier (NB), Artificial neural networks (ANNs) and Classification trees (CTs). Cheng Yeh and Che-hui Lien found that artificial neural networks achieve the best performance with relatively low error rate[2]. In the article by Cheng Yeh and Che-hui Lien, the classification method was run on all the variables. One way our paper differs from Cheng Yeh and Che-hui Lien is that we will first use variable selection methods to reduce the number of variables then use classification methods. We

also saw that more customers don't default their payment in this data. This can be seen as an unbalance in the category in this data set. This needs to be considered when looking to see which model is better.

Notation and Model

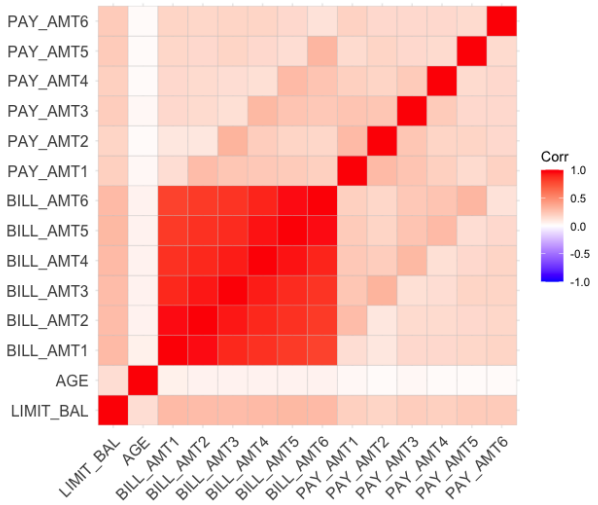
The dataset consists of 30000 observations and 24 variables. All the variables and their description can be found in table 1. In this data set, we noticed that there were some unknown categories in some of the variables, such as education, marital status, etc. Instead of removing all the unknown data, we merged it with the other category. Since the response variable is either yes or no depending on whether or not a client default payment, therefore this is a classification study. So all the models that would be looked in this paper will be a classification of a binary variable. We will be looking at ….model>>>

First, we conducted an exploratory data analysis by plotting various graphs. We started by plotting the correlation matrix and explored how different variables are correlated. This result can be seen in figure 1. As seen in figure 1, some of the variables are highly correlated which can influence our analysis. It can be seen that the amount of bill statement payment for period one to period six are highly correlated. Whereas, the repayment amounts for different time periods are not strongly correlated. From this we think there might exist multicollinearity which can affect data analysis.

```
str(df)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    30000 obs. of  24 variables:
##  $ LIMIT_BAL: num  20000 120000 90000 50000 50000 50000 500000 100000
140000 20000 ...
##  $ SEX      : Factor w/ 2 levels "1","2": 2 2 2 2 1 1 1 2 2 1 ...
##  $ EDUCATION: Factor w/ 5 levels "0","1","2","3",..: 3 3 3 3 3 2 2 3 4 4
...
##  $ MARRIAGE : Factor w/ 3 levels "1","2","3": 1 2 2 1 1 2 2 2 1 2 ...
##  $ AGE      : num  24 26 34 37 57 37 29 23 28 35 ...
##  $ PAY_1    : num  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
##  $ PAY_2    : num  2 2 -1 -1 -1 -1 -1 -1 -1 -1 ...
##  $ PAY_3    : num  -1 -1 -1 -1 -1 -1 -1 -1 2 -1 ...
##  $ PAY_4    : num  -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
##  $ PAY_5    : num  -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
##  $ PAY_6    : num  -1 2 -1 -1 -1 -1 -1 -1 -1 -1 ...
##  $ BILL_AMT1: num  3913 2682 29239 46990 8617 ...
##  $ BILL_AMT2: num  3102 1725 14027 48233 5670 ...
##  $ BILL_AMT3: num  689 2682 13559 49291 35835 ...
##  $ BILL_AMT4: num  0 3272 14331 28314 20940 ...
##  $ BILL_AMT5: num  0 3455 14948 28959 19146 ...
##  $ BILL_AMT6: num  0 3261 15549 29547 19131 ...
##  $ PAY_AMT1 : num  0 0 1518 2000 2000 ...
##  $ PAY_AMT2 : num  689 1000 1500 2019 36681 ...
##  $ PAY_AMT3 : num  0 1000 1000 1200 10000 657 38000 0 432 0 ...
##  $ PAY_AMT4 : num  0 1000 1000 1100 9000 ...
##  $ PAY_AMT5 : num  0 0 1000 1069 689 ...
```

```
##  $ PAY_AMT6 : num  0 2000 5000 1000 679 ...
##  $ dpnm     : Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1 1 1 ...
```



from the correlation plot we can see that the amount of bill statement payment for period1,period2...period6 are most highly correlated. followed by repayment status for different time period. so it might exsit the multicollinearity and will effect our futher data analyze. so we need lasso regression to reduce the highly correlated variables.

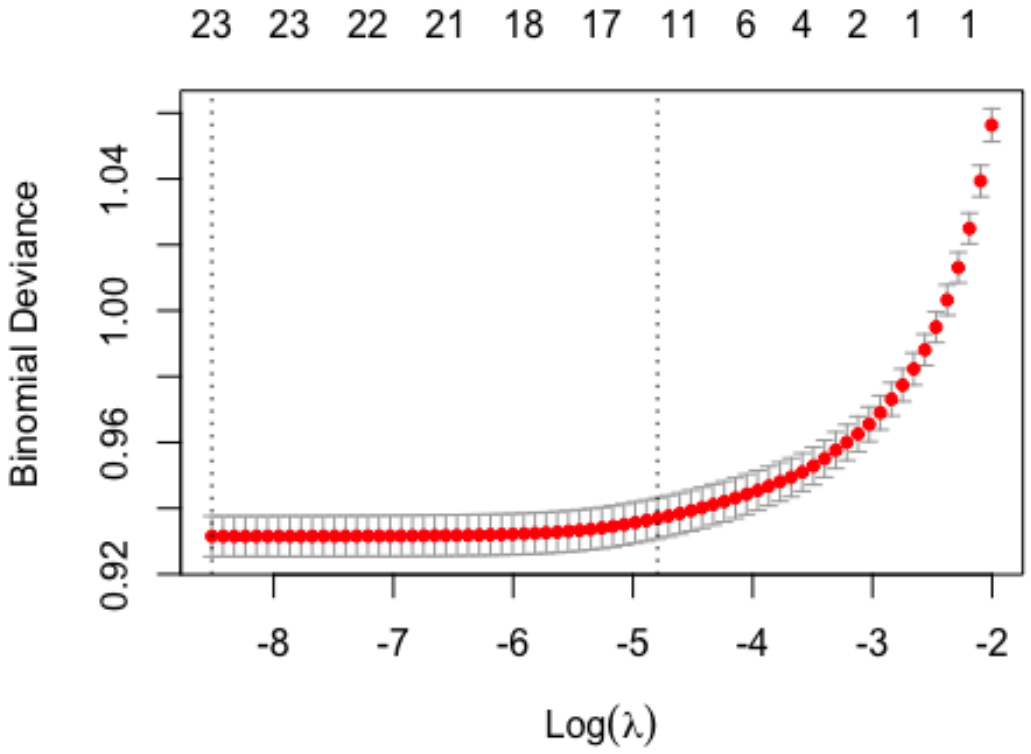
Methodology

Section 1: Variable Selection

To reduce multicollinearity and overfitting, we went through a variable selection process. We primarily used the cross-validation lasso regression. The Least Absolute Shrinkage and Selection Operator (LASSO) is a method of variable selection that uses the penalization of the regression coefficients based on the value of a tuning parameter λ. In the cross-validated lasso regression, the penalty parameter λ is chosen using the K-fold cross-validation. This enables the lasso to automatically reduce the coefficients of irrelevant variables to zero. Based on the selected variables from the cross-validation lasso, we aimed to further simplify the model by obtaining the best AIC. This was done through two different approaches: using glmulti package and stepwise variable selection. Glmulti is a R package for automated model selection based on information criteria. In this approach, only the main effects were used to build the candidate set because the interactions were already reduced through the cross-validation lasso. Stepwise variable selection is a method that repeatedly adds the most contributive predictors and removes variables that no longer improves the model fit.

```
set.seed(111)
y = df$dpnm
x = model.matrix(dpnm~.,dfm)

cv.out <- cv.glmnet(x,y,alpha=1,family="binomial",type.measure = "deviance" )
plot(cv.out)
```




```
lambda_min <- cv.out$lambda.min
lambda_1se <- cv.out$lambda.1se
coef(cv.out,s=lambda_1se)
```

```
## 26 x 1 sparse Matrix of class "dgCMatrix"
##                              1
## (Intercept) -1.127071e+00
## (Intercept)  .
## ID           .
## LIMIT_BAL   -7.042723e-07
## SEX         -6.534529e-03
## EDUCATION    .
## MARRIAGE    -5.421898e-02
## AGE          1.999301e-03
## PAY_1        5.611599e-01
## PAY_2        7.407625e-02
## PAY_3        7.132704e-02
## PAY_4        1.296048e-02
## PAY_5        2.122928e-02
## PAY_6        .
## BILL_AMT1   -1.281042e-06
## BILL_AMT2    .
## BILL_AMT3    .
```

```
## BILL_AMT4    .
## BILL_AMT5    .
## BILL_AMT6    .
## PAY_AMT1    -4.438892e-06
## PAY_AMT2    -1.087985e-06
## PAY_AMT3     .
## PAY_AMT4    -2.638372e-07
## PAY_AMT5    -2.578755e-07
## PAY_AMT6     .
```

through the lasso regression crossvalidtion, we find that we only left one variable for the bill payment amount(bill amount at time 1). so does the previous payment amount (payment amount at time 1) however we keep repayment status variables for different time period except for pay_6. besides this, LIMIT_BAL, MARRIAGE are also been selected. through lassso regression, we reduce some highly correlated variables.

Section 2: Model Selection

Method1: Logistic Regression

In a classification model, one would model the conditional probability $p(x) = Pr(Y = 1|X = x)$ as a function of x. One way to look at the probability is to let p(x) be a linear function of x, which is the base behind logistic regression. Logistic regression is a specific case of linear regression models where it is used to model the probability of a class or event existing [3]. In this dataset, the response variable (default payment (Yes = 1, No = 0)), is a binary variable; therefore, the binary logistic model would be used for the analysis. The logistic regression model is

The advantage of this model is that it can produce a simple probabilistic formula of classification, and its disadvantage is that logistic regression cannot deal with non-linear and interactive effects of explanatory variables [2].

Method2: Linear and Quadratic Discriminant Analysis (LDA & QDA)

Suppose we want to classify an observation into K classes. Let pik defines a prior probability that a randomly chosen observation belong to kth class, also let ,according to bayes theorem 4.10. $Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$ (1)
We can express a p dimensional random variable x has multivariate gaussian distribution ,we write x $X \sim N(\mu, \Sigma)$, the gaussian density is defined as (2). $\frac{1}{(2\pi)^{p/2} det(\Sigma)^{1/2}} \exp(-\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu))$ (2)
and we can plug (2) into (1) and we can get the bayes classifier for an observation x is (3). $\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + log \pi_k$ (3) values x for which $\delta_k(x) = \delta_l(x)$;i.e. $x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k = x^T \Sigma^{-1} \mu_l - \frac{1}{2} \mu_l^T \Sigma^{-1} \mu_l$(4)
#LDA on full model

```
lda_fit = lda(dpnm~.,train)
pred_lda = factor(predict(lda_fit,x_test)$class,levels = c("1","0"))
```

```
confusionMatrix(pred_lda, factor(y_test,levels = c("1","0")), positive =
NULL)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    1    0
##          1  501  274
##          0  789 4436
##
##                Accuracy : 0.8228
##                  95% CI : (0.8129, 0.8324)
##     No Information Rate : 0.785
##     P-Value [Acc > NIR] : 1.667e-13
##
##                   Kappa : 0.3862
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.3884
##             Specificity : 0.9418
##          Pos Pred Value : 0.6465
##          Neg Pred Value : 0.8490
##              Prevalence : 0.2150
##          Detection Rate : 0.0835
##    Detection Prevalence : 0.1292
##       Balanced Accuracy : 0.6651
##
##        'Positive' Class : 1
##
```

from the lda, we can see that the overall accuracy is 0.8197. which is pretty much decent. however the sensitivity is only 0.36695. which means only 36% of default cases are detected by lda. in the contrast of its high specificity. nearly 94% of non-default cases are detected. #LDA on selected model

```
lda_fit_select =
lda(dpnm~LIMIT_BAL+MARRIAGE+PAY_1+PAY_2+PAY_3+PAY_4+PAY_5+BILL_AMT1+PAY_AMT1,
train)
pred_lda_select = factor(predict(lda_fit_select,x_test)$class,levels =
c("1","0"))
confusionMatrix(pred_lda_select, factor(y_test,levels = c("1","0")), positive
= NULL)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    1    0
##          1  512  292
##          0  778 4418
```

```
##
##                Accuracy : 0.8217
##                  95% CI : (0.8117, 0.8313)
##     No Information Rate : 0.785
##     P-Value [Acc > NIR] : 8.967e-13
##
##                   Kappa : 0.388
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.39690
##             Specificity : 0.93800
##          Pos Pred Value : 0.63682
##          Neg Pred Value : 0.85027
##              Prevalence : 0.21500
##          Detection Rate : 0.08533
##    Detection Prevalence : 0.13400
##       Balanced Accuracy : 0.66745
##
##        'Positive' Class : 1
##
```

from the lda method toward selected model, we can see that the overall accuracy is 0.8202. which is slightly better than full model. and sensitivity and specificity are 0.3701 and 0.9433,which are also close to full model's sensitivity and specificity. we can say that the selected model has similiar performance with full mo del.

QDA on full model

For the qda we have $X \sim N(\mu_k, \Sigma_k)$ the Σ_k is covariate matrix For the kth class. And based on this, the bayes classifier for observations X =x to the class k is $\delta_k(x) = -\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k) - \frac{1}{2} log|\Sigma_k| + log\pi_k$

$= -\frac{1}{2} x^T \Sigma_k^{-1} x + x^T \Sigma_k^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma_k^{-1} \mu_k - \frac{1}{2} log|\Sigma_k| + log\pi_k$

for qda, we first fit it on full model using below code. qda_fit = qda(dpnm~.,train) pred_qda = factor(predict(qda_fit,x_test)$class,levels = c("1","0")) confusionMatrix(pred_qda, factor(y_test,levels = c("1","0")), positive = NULL) however, it return the rank deficiency error, which means we don't have enough data points to estimate all varibles.It also means that there is still some collinearity there. so let's try to fit with less variables in reduced model. #QDA on selected model

```
qda_fit_se =
qda(dpnm~LIMIT_BAL+MARRIAGE+PAY_1+PAY_2+PAY_3+PAY_4+PAY_5+BILL_AMT1+PAY_AMT1,
train)
pred_qda_se = factor(predict(qda_fit_se,x_test)$class,levels = c("1","0"))
```

```
confusionMatrix(pred_qda_se, factor(y_test,levels = c("1","0")), positive =
NULL)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    1    0
##          1  734  819
##          0  556 3891
##
##                Accuracy : 0.7708
##                  95% CI : (0.76, 0.7814)
##     No Information Rate : 0.785
##     P-Value [Acc > NIR] : 0.9962
##
##                   Kappa : 0.3679
##
##  Mcnemar's Test P-Value : 1.599e-12
##
##             Sensitivity : 0.5690
##             Specificity : 0.8261
##          Pos Pred Value : 0.4726
##          Neg Pred Value : 0.8750
##              Prevalence : 0.2150
##          Detection Rate : 0.1223
##    Detection Prevalence : 0.2588
##       Balanced Accuracy : 0.6976
##
##        'Positive' Class : 1
##
```

Our QDA on selected model worked, the overall accuracy is 0.77.which is little lower than LDA, however, the sensitivity is 0.58, which is 20% percentage higher than the lda. which means 58% of default cases are detected by qda. it's specificity is almost 82%,that means 82% of non-default cases are detected. compared to LDA, the QDA impose a more strict rule to decide if the credit card default.

Method3: Deep Learning Model

Deep Learning is a technique for an approach to artificial intelligence called neural networks. It is also a subset of machine learning, where a computer learns to perform required tasks by analyzing training data. After going over the entire dataset many times, it would find patterns that consistently correlate with input variables/labels. Most of the neural nets are organized into layers of nodes; one node might have several nodes connected, data passes through the succeeding layers with weights applied during this process, after receiving all kinds of transformations like complex multiplication and additives, it arrives at the output layer. It is worth mentioning that the weights applied are adjusted to find patterns in order to make better predictions, users do not need to specify what patterns to look for — the neural network learns on its own. For this part of the

analysis, we decided to use Keras-a user-friendly neural network library written in Python to complete the task.

Method 4: Support Vector Machine (SVM)

SVM is a machine learning algorithm used to classify categorical data (binary data will only be considered in this paper). Given a labeled training set, SVM defines a hyperplane that best separates the data according to their classes. Classifications are then determined by where data points lie in relation to the hyperplane. Consider, for example, the following plot consisting of two classes, blue circle and red square.

As shown by the green lines, there are many ways to construct a classifier that separates the classes. The optimal classifier maximizes the distance between the classes or, in other words, produces the maximum margin.

The optimal classifier is an $N-1$ dimensional hyperplane, where N denotes the number of features. In the above example, data points that lie to the upper right of the hyperplane are classified as blue circles while data points that lie to the bottom left of it are classified as red squares. The disposition of the hyperplane is influenced most by the closest data points. These data points, which are called the support vectors, delineate the classes and help determine the margin to be maximized. In the above example, the support vectors lie on the dashed lines. Mathematically, this can be interpreted as follows. Let x denote a matrix of features, $y \in -1,1$ denote a vector of binary responses, and w & b denote two parameters (vector and scalar, respectively). The i^{th} response is then determined as $y_i = 1$ if $w^T x_i + b \geq 1$ otherwise, $y_i = -1$ if $w^T x_i \leq -1$. This can be rewritten as $y_i(w^T x_i + b) \geq 1$. SVM maximizes the distance between the two classes and mathematically, this translates to maximizing the distance between $w^T x_i + b = 1$ and $w^T x_i + b = -1$. The distance between these two hyperlanes is $\frac{2}{||w||}$, where $||.||$ denotes the 1-norm of the argument. To maximize this margin, we solve $\max_w \frac{2}{||w||}$, or equivalently $\min_w \frac{||w||}{2}$. Also note that we'd like SVM to correctly classify all data points or equivalently, satisfy $y_i(w^T x_i + b) \geq 1 \forall i \in 1, \dots, N$. Consequently, the optimal hyperplane is obtained by solving

$$\min\limits_{w} \frac{||w||}{2} \\ s.t. ~~~ y_{i}(w^{T}x_{i} + b) \ge 1 ~~~ \forall i \in \{1, ..., N\}$$

SVM is advantageous because it focuses on data points that are closest to the decision boundary. These points are used to construct the separating hyperplane while points that are far from the boundary have little to no influence. The intuition here is that if this model can classify the most difficult data points (ones that are near the boundary), then it should be fairly good at classifying the easier data points (ones that are far from the boundary).

Data Analysis

Section 2: Model Selection Method

From the result above, we used the selected variables to run different modeling methods. We used the training data set to generate the models; then, we tested how well it is at

predicting the classes using the testing data set. When LR was tested, it found to have an accuracy of 81.82%.

Now we are going to build a Deep Learning Model using a Kera based neural network for predicting default of credit card clients.
In this method, we set the number of epochs to 100, which means it will go through the entire dataset 100 times. We define a fully-connected network structure with three layers in Keras. The first hidden layer has 8 nodes and uses the relu activation function. The second hidden layer has five nodes and uses the relu activation function, and then the output layer has one node with the sigmoid activation function. The activation function determines the output a node will generate, based upon its input.
Next, we compile the model, and the best way to represent the network for training and making predictions to run on our hardware is automatically chosen by the backend. We will define the optimizer as the effective stochastic gradient descent algorithm adam, this algorithm tunes itself and gives good results in a wide range of problems.
Finally, because this is a classification problem, we will collect and report the classification accuracy, defined via the metrics argument.
Move on to the fitting of Keras Model. We want to train the model many times until it learns a good enough mapping of rows of input data to the output classification. Finally, we can evaluate the performance of the network on the testing dataset. Just like a black box, this method will only give us an idea of how well we have modeled the dataset and provide predictions, but not of how exactly the model made the predictions. Through the Keras Model, we can get an accuracy of 78.5%.

Now, the SVM algorithm is run on a reduced dataset obtained from lasso regression and stepwise selection. A model is trained on a training set and predictions are made on a seperate test set. The following confusion table summarizes the prediction results, where "1" represents a client defaulting and "0" represents the converse.

```
##            Reference
## Prediction    0    1
##          0 4535  844
##          1  190  431
```

The SVM model achieves an accuracy of 81.48%, a sensitivity of 0.338, a specificity of 0.948, and an F-measure of 0.444. The most notable metric is the specificity (0.948). This measure tells us the model is very proficient at accurately predicting reputable clients (i.e. responses classified as 0 will largely be predicted as 0). In spite of this, it is reasonable to assume that a lending institution is more interested in correctly classifying risky customers than safe customers. It is, after all, the risky customers that can potentially default on their loans. In this sense, the model does not perform very well as it achieves a low sensitivity (0.338). This measure tells us the model inadequately predicts risky customers (i.e. responses classified as 1 will not always be predicted as 1). The sensitivity drawback is also represented in the F-measure (0.444). The model's high specificity and low sensitivity translate into a mediocre F-measure. Since this metric provides a general summary of the model, we can conclude that while the SVM fit isn't perfect, it does provide an added benefit in estimating customer credit risk.