**Evaluation of Credit Card Default**

**Payments Using Only Excel**

**09/11/2020**

**James (Nick) Newsome**

**I. Introduction**

**II. Data Set**

**A. Description**

**B. Issues**

**C. Excel Formatting**

**III. Excel Analysis**

**A. Basic Analysis**

**B. Five Attempts**

**C. Results**

**IV. Conclusion**

**Introduction**

In this report, I evaluate data to determine if I can predict customers likely to default on their next credit card payment. For this report, I wanted to use only Microsoft Excel in order to demonstrate my some of my ability using multiple sheets and formulas as well as the Solver add-in. The data is evaluated using Solver as well as normalizing some of the data. The effectiveness of the solutions was determined by using a confusion matrix. I was not able to find a good solution and better solutions can probably be found using Python with machine learning techniques.

**Data**

**Description**

The data set was obtained from <https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients> and is a compilation of 30,000 credit card clients in Taiwan in 2005. There are 25 columns explained by the website as:

This research employed a binary variable, default payment (Yes = 1, No = 0), as the response variable. This study reviewed the literature and used the following 23 variables as explanatory variables:   
**X1**: Amount of the given credit (NT dollar): it includes both the individual consumer credit and his/her family (supplementary) credit.   
**X2**: Gender (1 = male; 2 = female).   
**X3**: Education (1 = graduate school; 2 = university; 3 = high school; 4 = others).   
**X4**: Marital status (1 = married; 2 = single; 3 = others).   
**X5**: Age (year).   
**X6 - X11**: History of past payment. We tracked the past monthly payment records (from April to September, 2005) as follows: X6 = the repayment status in September, 2005; X7 = the repayment status in August, 2005; . . .;X11 = the repayment status in April, 2005. The measurement scale for the repayment status is: -1 = pay duly; 1 = payment delay for one month; 2 = payment delay for two months; . . .; 8 = payment delay for eight months; 9 = payment delay for nine months and above.   
**X12-X17**: Amount of bill statement (NT dollar). X12 = amount of bill statement in September, 2005; X13 = amount of bill statement in August, 2005; . . .; X17 = amount of bill statement in April, 2005.   
**X18-X23**: Amount of previous payment (NT dollar). X18 = amount paid in September, 2005; X19 = amount paid in August, 2005; . . .;X23 = amount paid in April, 2005.

These columns have also been labeled in **bold** below their "X" designation. The labels, which is how they will be referred henceforth are as follows:

**ID**: Unique identifier given to each card holder. Does not have an "X" designation.

**LIMIT**: Credit Limit (X1)

**SEX**: Gender (X2)

**EDU**: Education Level (X3)

**MAR**: Marital status (X4)

**AGE**: Age (X5)

**PAY1 - PAY6:** Payment history (X6-X11)

**BILLAMT1 - BILLAMT6**: Bill amount history (X12-X17)

**PAYAMT1 - PAYAMT6**: Payment amount history (X18-X23)

**DEFAULT:** This is the binomial response where 1 signifies a default payment and 0 signifies non-default.

These columns are enclosed in light green border.

**Issues**

There are a couple of issues or discrepancies with this data set. The most notable is that of the values of PAY1-PAY6. According to the description of the data, the accepted values for these columns are -1 for paid on time or an integer between 1 and 9 based on how many months behind they are. However, -2 and 0 also show up in each of these columns and there is no key for determining what these mean. Additionally, there are bill amounts in the most recent month that are negative, but the next month the account is in default. This does not seem to make sense. I have looked for more information regarding this data set to see if these issues could be resolved, but I could find nothing more on them.

**Excel Formatting**

In the Excel document, the first tab (Data) contains the original data. It was formatted to fit the screen better and all rows except for the first 3 and last 3 are hidden. Underneath the original data are four rows that calculate the minimum, maximum, mean, and standard deviation of each column. This data is surrounded by a dark green border. Beneath this original data, I separated out just the default payments in order to look for differences amongst the four calculations. These are formatted the same as the original data. To the right of the original and default only data is data that is enclosed with a blue border. This data is calculations on the original data. The columns are:

**PAYAVG:** Average of PAY1-PAY6

**PAYAMTAVG:** Average of PAYAMT1-PAYAMT6

**BILLAVG:** Average of BILLAMT1-BILLAMT6

**NPAYAMTAVG:** This normalizes PAYAMTAVG

**PAY/BILL:** PAYAMTAVG / BILLAVG, the average payment as a percentage of the average bill amount

**BILLDIFF:** Difference between largest bill and smallest bill

**NBILLDIFF:** BILLDIFF normalized

**PAYDIFF:** Difference between largest payment and smallest payment

**NPAYDIFF:** PAYDIFF normalized

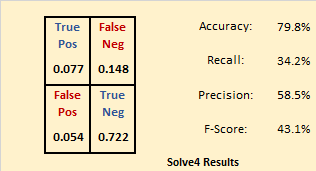
Between the original data and the default only data are three rows where I calculated the total default out of the 30,000, the total non-default, and the percentage of non-default to total. Out of 30,000 card holders, 77.9% are non-defaulters which translate to negatives in the confusion matrix. In order to break out a train and test set, I created a column and assigned a random number to each card holder, then sorted the data by the random numbers in ascending order. The first 25,000 rows became the train set and the final 5000 were the test set. The train set were put in the second tab (Train) and the test set were separated into the third tab (Test).

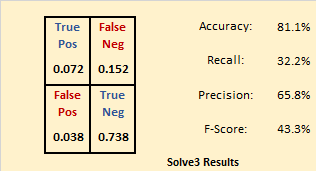
**Excel Analysis**

**Basic Analysis**

To run Solver and look for a new sheet was added (Solvei). In this sheet, only the column names were included and the data would be referenced from the train data set. The concept of Solver is to use a basic slope intercept formula to determine coefficients to each variable to find the best fit. To accomplish this, I would assign 0.01 as the coefficient of each variable that I would be testing plus as well as assigning one to DEFAULT. The DEFAULT column is the value I am trying to predict and does not get assigned a coefficient, so this space was used as the intercept. The coefficients were highlighted in a light orange. For clarity, any variable not being considered had their coefficients set as 'NA'. Using the slope intercept formula (y = m1x1 + m2x2 + … + mkxk + b), Y was calculated for each slope (all set to 0.01 to begin with). This is the first column in the light orange area with the heading 'Use Solver to Find Equation' beneath the coefficients. This Y is labeled 'Logit' and was calculated for each row, but all rows except for the first and last two are hidden. From here, the column to the right calculates eLogit, the probability P(x) of the event and finally the log likelihood. This is done for each row. The best estimates for the coefficients will be when we maximize the sum of the log likelihood column. Once Solver has done this, then the cells in the light yellow will display the confusion matrix as well as other stats to help us determine if these estimated coefficients have any predictive value. One thing to note is that the original data had 77.9% non-default or negatives. So if we just assume that everyone is a non-defaulter, then we would be 77.9% accurate. Thus any model with less accuracy than that is of no value. This does not take into consideration a cost associated with catching a defaulter. So if finding a defaulter early saves the company money, then a less accurate model could still be of value if the amount of money saved can compensate for the expenses needed to deal with the predicted defaulters.

**Five Attempts**

Based solely on my comparison of the datasets of all values vs those of just the defaults I made some attempts to determine which columns had the greatest predictive value. The first attempt (Solve1) uses only the PAY1-PAY6 columns. After running the calculations, the accuracy of the model is slightly better than just guessing at 78.4%. This model catches 5.7% of the defaulters (Recall) while not losing too many of the non-defaulters. However, the low catch rate of the defaulters renders this model very poor and the resulting F-score is a mere 10.6%. Solve 2 uses only my calculated values and does better than Solve1. The accuracy increases to 79.3% and the F-score almost doubles as does the rate of catching defaulters. But still, there are way too many False Negatives or predictions that a card holder will not default, when in fact they will. Solve3, really starts to show some improvement. Here I all 6 PAY columns as well as my calculated columns. The accuracy increases to 81% and the F-score doubles again to over 43.3%. It is now catching 32.2% of the defaulter. The prediction of defaulters (Precision) is at 65.8% meaning that if it predicts that a card holder is a defaulter, it is accurate 65.8% of the time. This measurement has decreased from the previous two attempts though. Solve4 is another attempt that rivals that of Solve3. The accuracy falls off a little, but the recall increases. It sacrifices heavily on the precision though, so many of those predicted to be defaulters are not. The F-score is similar and which model would be better would be determined by the costs and savings of finding a defaulter early. This model used only the previous 2 months PAY as well as a few of my calculated values. The final attempt added in the personal information of the card holder and did not do nearly as well as Solve3 or Solve4.



**Results**

Neither of these attempts did very well, but even a slight increase that saves some money that can be used over a large number of people could save substantial amounts of money. The final step was to see how these formulas did on the test set. The final two tabs in the Excel document are Test3 and Test4 where each formula was applied to the test data. The predicted Y value (Y hat) was compared to the actual just like in the train data and the results evaluated using the confusion matrix. Both formulas performed extremely poorly, with F-scores in the teens and accuracy in the low 70s. Precision and recall were both substantially lower than the training versions at about a third of their training values. This means that the formulas were over-fitted on the training data and have no real world value.

**Conclusion**

The analysis using Excel did not provide any meaningful insights. A better understanding of the issues (see II.B Issues) associated with this data set could possibly allow for better analysis. However, it is likely that other machine learning techniques would need to be employed in order to be able to predict the defaulters. A further analysis involving Python and machine learning techniques such as K Nearest Neighbor or Support Vector Machines would be great follow up techniques and may be explored in the future.