Machine Learning Project
Predicting the Defaulters of Credit Card users

Sudarsan Raju

Proposal objective vs results

In Credit Card Risk management perspective estimating the probability of defaulters will be more meaningful than classifying customers into the binary results – risky and non-risky. Therefore, whether or not the estimated probability of defaulters produced from the model can represent the ‘‘real” probability of defaulters is an important problem. So if we can predict the relationship between defaulter vs estimated probability of real defaulters can be real value for business to develop credit strategy.

Objective

The main objective of this project is to predict the accuracy of the estimated probability of credit card defaulters against the actual defaulters. The target variable is “default payment next month” is binary. Variables in the dataset are character types of **integer** (Amount of given credit, Age, History of past payment, Amount of bill statement, Amount of previous payment) and **real** (Marital status, education, Gender)

Methodology

Since that the target variable, the indicator of whether a customer can be defaulter “default payment next month, is binary, I used Logistic regression & decision tree modeling approaches to analyze the data.

Logistic regression – using this for model to understand the relationships among data elements, estimating the impact of a treatment on an outcome that can be used to assess whether the observations are due to chance alone. In this model, the logistic regression will generate a probability that a customer as defaulter and then the outcome will be compared against the estimated probability of defaulters.

Decision tree – using this model to split the data based on different categories in the dataset and the relationships among the features and the potential outcomes. In this model, whether a customer will be a defaulter is evaluated by different categories based on the attributes and evaluated against the actual defaulters.

Results

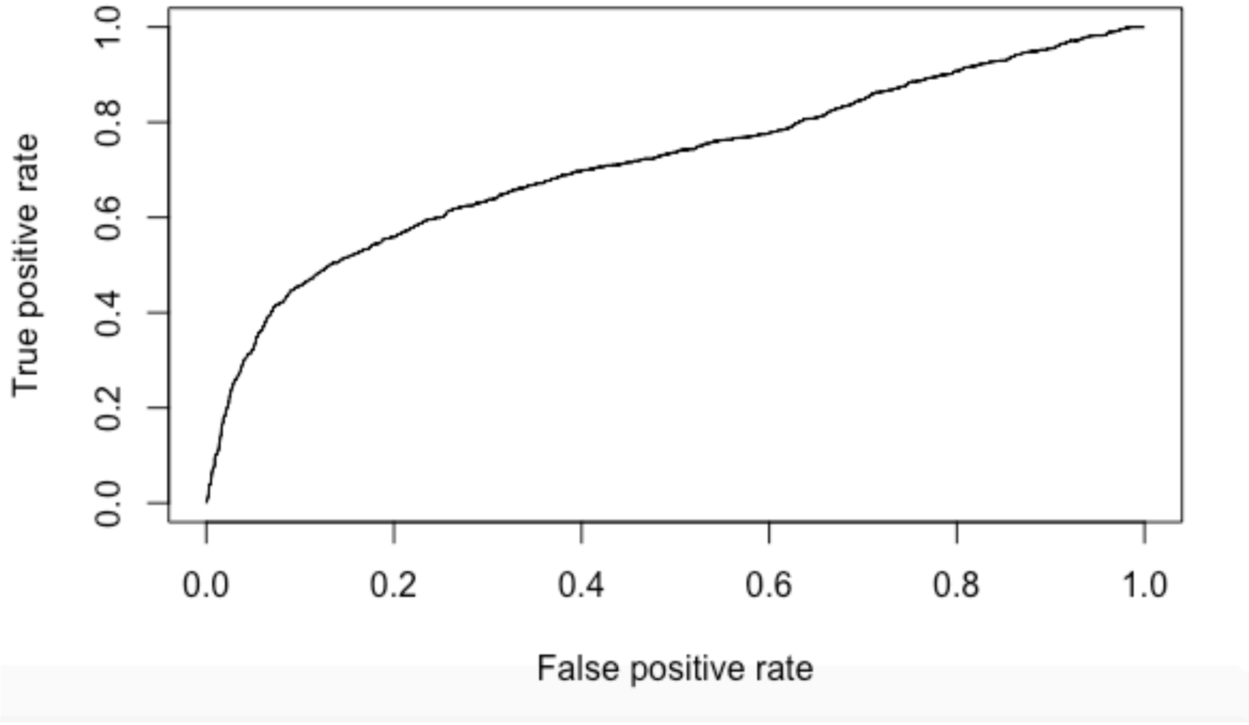
As part of this project using we predicted the accuracy of defaulters from the data using Logistic regression and Decision Tree. From our prediction both Logistic regression and Decision Tree has accuracy of 81**%.** Results will be discussed in detailed below.

Data used - **http://archive.ics.uci.edu/ml/machine-learning-databases/00350/**. Dataset contains about 30000 records. No data was omitted and used all the records as part of this project.

Comparing model 1 and Model 2 results:

a) Results from Logistic regression – Model 1

Using Logistic Regression, the predictability of the model the accuracy is [1] 0.81. The 0.81 accuracy on the test set is quite a good result and we can say that outcome of the defaulter is accurate about 0.81 from the dataset. Also plotted the ROC curve and calculated the AUC, which are typical performance measurements for a binary classifier. AUC is 0.8647186. which is closer to 1, We can say model has Good predictive ability.



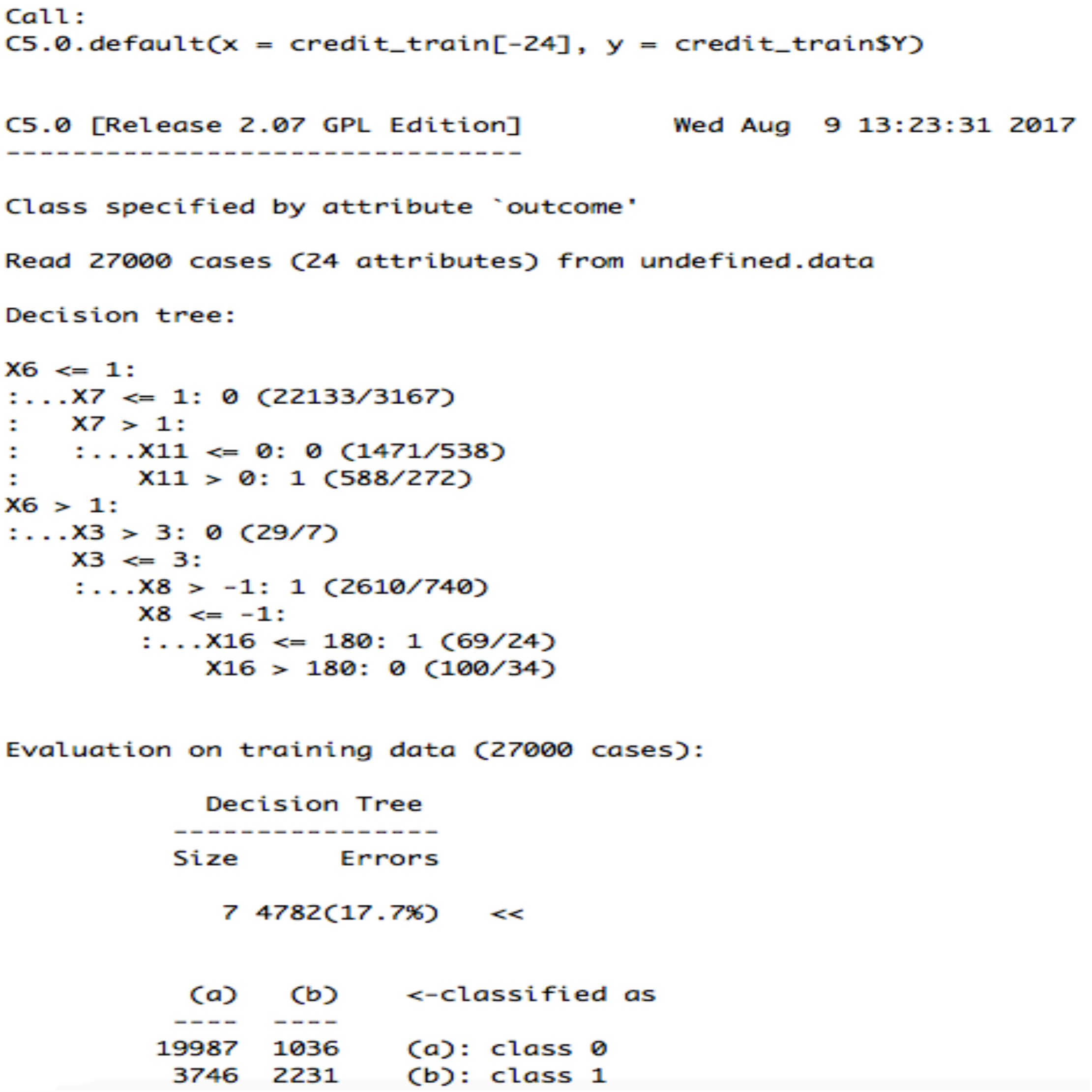
Data was selected randomly using set.seed(123) command and divided between training and test data. Here's the summary of the model

Call:glm(formula = Y ~ ., family = binomial(link = "logit"), data = credit_train)

Deviance Residuals: Min   1Q   Median   3Q   Max -3.1672 -0.7013 -0.5462 -0.2895   3.8034

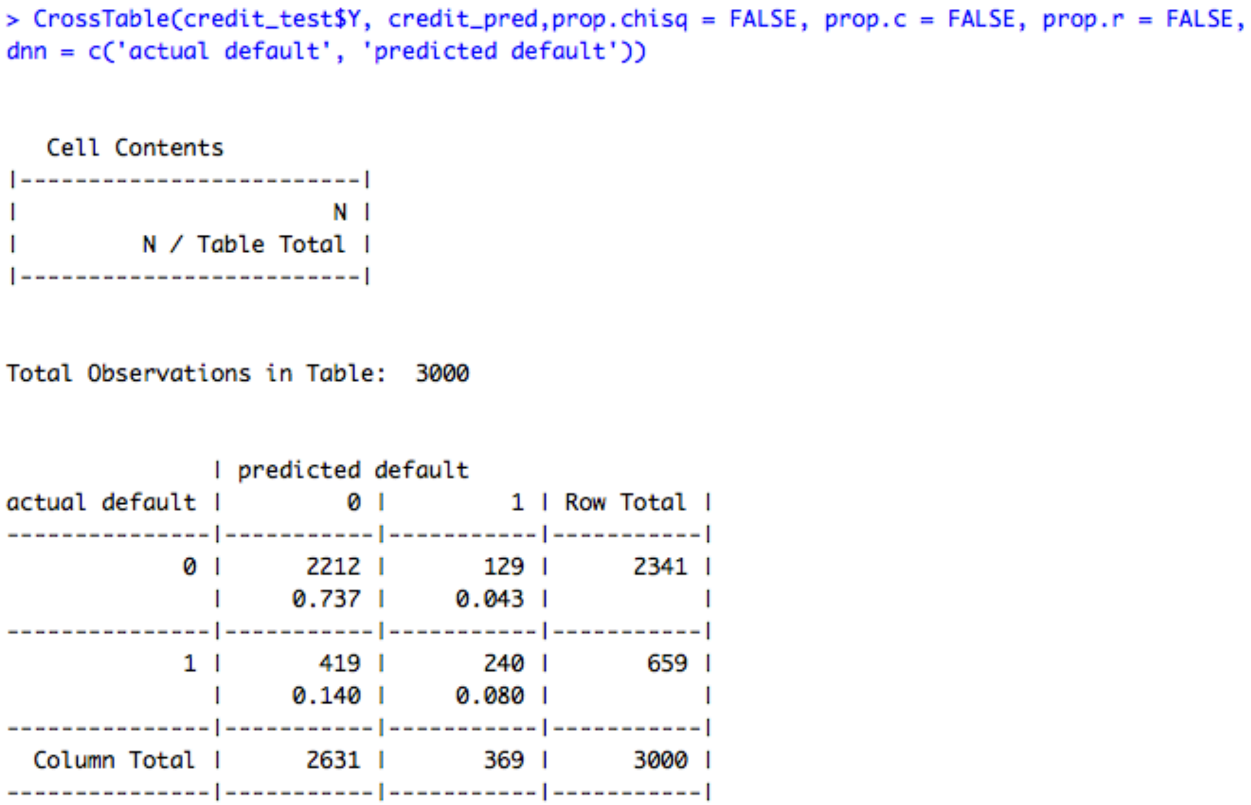
b) Results from Decision Tree – Model 2

Using decision tree we have predicted the accuracy of probability of defaulters is 81%. The Errors output notes that the model correctly classified all but 4782 of the 2700 training instances for an error rate of 17.7 percent. A total of 1036 actual no values were incorrectly classified as yes (false positives), while 3746 yes values were misclassified as no (false negatives).



```
Call:
C5.0.default(x = credit_train[-24], y = credit_train$Y)


C5.0 [Release 2.07 GPL Edition]          Wed Aug  9 13:23:31 2017
-------------------------------

Class specified by attribute `outcome'

Read 27000 cases (24 attributes) from undefined.data

Decision tree:

X6 <= 1:
:...X7 <= 1: 0 (22133/3167)
:   X7 > 1:
:   :...X11 <= 0: 0 (1471/538)
:       X11 > 0: 1 (588/272)
X6 > 1:
:...X3 > 3: 0 (29/7)
    X3 <= 3:
    :...X8 > -1: 1 (2610/740)
        X8 <= -1:
        :...X16 <= 180: 1 (69/24)
            X16 > 180: 0 (100/34)


Evaluation on training data (27000 cases):

            Decision Tree
          ----------------
          Size      Errors

             7 4782(17.7%)   <<


           (a)   (b)    <-classified as
          ----  ----
         19987  1036    (a): class 0
          3746  2231    (b): class 1
```

Out of the 3000 test dataset records, our model correctly predicted that 2212 did not default and 240 did default, resulting in an accuracy of 81 percent and an error rate of 19 percent. This is somewhat worse than its performance on the training data, but not unexpected, given that a model's performance is often worse on unseen data. Also note that the model only correctly predicted 240 of the 659 defaulters in the test data. Below picture represents **Confusion matrix.**

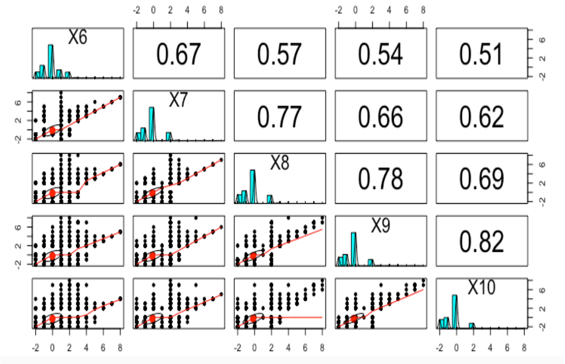


```
> CrossTable(credit_test$Y, credit_pred,prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
dnn = c('actual default', 'predicted default'))


   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  3000


               | predicted default
actual default |         0 |         1 | Row Total |
---------------|-----------|-----------|-----------|
             0 |      2212 |       129 |      2341 |
               |     0.737 |     0.043 |           |
---------------|-----------|-----------|-----------|
             1 |       419 |       240 |       659 |
               |     0.140 |     0.080 |           |
---------------|-----------|-----------|-----------|
  Column Total |      2631 |       369 |      3000 |
---------------|-----------|-----------|-----------|
```

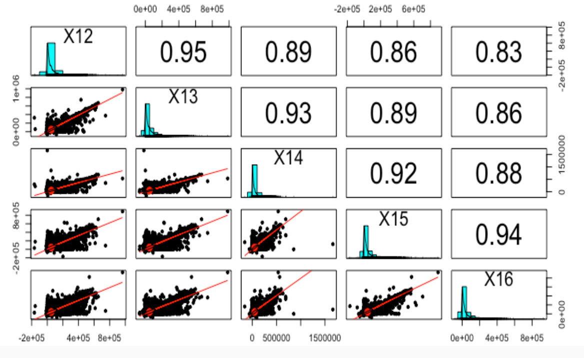
Comparing the results obtained from both the models. We can see 81% accuracy from both the models. Though the 81% accuracy of defaulters is good number from model perspective this may not be good for business to devise a credit strategy. We will discuss advantage and disadvantage about both models and then discuss about the improvement of the models.

Advantages and disadvantages of Logistic Regression & Decision Tree.

Advantage of Logistic regression is it provides estimates of both the strength and size of the relationships among features and the outcome and predicted the accuracy. As we can see some strong co-relation between history of payments and between amount paid over a period of time.



A) Correlation of past payment



B) Amount Paid.

Disadvantage of Logistic regression it Makes strong assumptions about the data and it doesn't handle missing data. It works only for numeric values. So pre-processing of data may be required.

Advantage of Decision Tree is Highly automatic learning process, which can handle numeric or nominal features and Excludes unimportant features. Results in a model that can be interpreted without a mathematical background. Also it Can be used on both small and large datasets

Disadvantage of Decision Tree **is** Decision tree models are often biased toward splits on features having a large number of levels. Can have trouble modeling some relationships due to reliance on axis-parallel splits. Small changes in the training data can result in large changes to decision logic. But we have we have split the randomly which would avoid the disadvantage.

Improving the two models.

To increase the **predictive** ability of Logistic regression, I would need to divide the data set in half, fit models to one half of the data, and then use them to predict the defaulters of credit card in the other half of the data set. Note that this describes the simplest case of validation of a model using a single data set. I would also employ n-fold cross validation (for example, using the rmspackage in R) to make the most efficient use the data.

To increase the **predictive** ability of Decision tree, I would use The C5.0() function to boost the accuracy by giving additional trails The trials parameter would set an upper limit; then the algorithm will stop adding trees if it recognizes that additional trials do not seem to be improving the accuracy.