Credit Card Defaults

Zach Ballard

4/27/2022

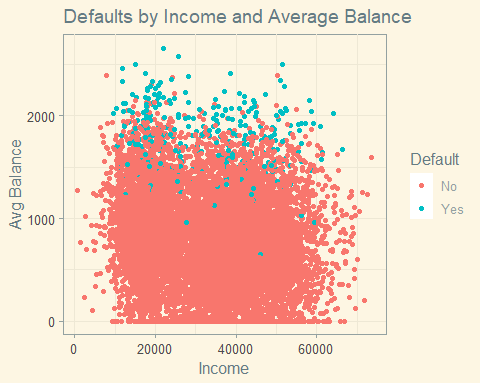
## Introduction

We will be processing the Default dataset from the Vincetarelbundock repository on Github. This dataset includes 10000 observations of Credit card customers. The data provided on these customers is their annual income, their average balance after each monthly payment, whether or not they are a student, and if they defaulted or did not default.  
In this report, we will be reviewing the data and analyzing it to try and best predict if a customer will default or not.

## Visualization

First we will take a look at all our default data in relation to the income and average monthly balances of our customers.

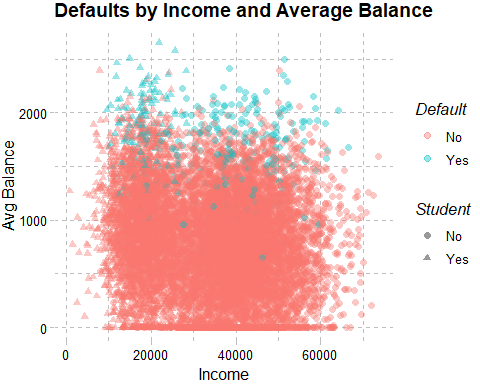
**Visualization 1**



We can see some logical distribution of the customers who defaulted.  
Those with higher balances seemed to default more often.

Now we take a look at the same data, but we introduce the student variable by turning our data points into shapes to represent whether the customer was a student or non student.

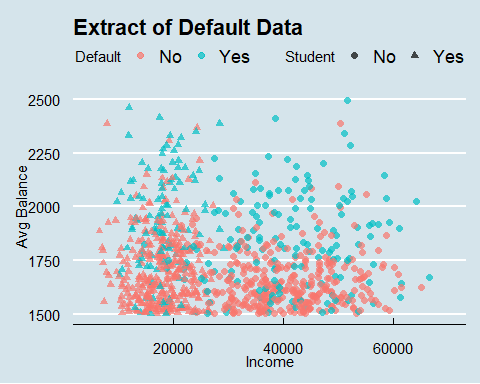
**Visualization 2**



This gives us a little better idea of how our student customers are spread out. We can clearly see that most of our student customers have low income.

Now to look at a zoomed in extract of the data. This should help us visualize the majority of the defaults better.

**Visualization 3**



This visualization more clearly shows us the majority of our customers in this data set that defaulted. Again, students are clearly seen by the triangles towards the left (lower income) side of the graph.

## Analysis

First, lets look at some simple statistics on our data.

Statistic Value  
1 Highest Income 73554.2335  
2 Lowest Income 771.9677  
3 Average Income 33516.9819  
4 Highest Monthly Balance 2654.3226  
5 Lowest Monthly Balance 0.0000

[1] "Percentage of student customers"

[1] "29.44%"

Now that we have a bit better idea about the overall characteristics of our data set, lets look at some percentage probabilities of defaulting by a few different criteria.

Probability\_Of\_Default\_if Value  
1 Income Less than 20k 4.3%  
2 Income between 20k and 40k 3.11%  
3 Income greater than 40k 3%  
4 Student 4.31%  
5 Non Student 2.92%  
6 Average Balance greater than 1500 28.32%  
7 Average Balance less than 500 0%

Now we have some probabilities to view, and we can see that as income increases, the odds of a customer defaulting goes down. We also see that a customers *average balance* has a huge impact on whether or not they will default. The higher balances (over 1500) have a 28% default rate, whereas the lower balances (under 500) have resulted in no customer defaults out of these ten thousand customers. We also see that our student customers default at a slightly higher rate than our non student customers.

Now we will try to predict whether or not a customer will default with a few different tests. For each test, we will display the three key metrics: Accuracy, Sensitivity, and Specificity.

First, we will use the nearest neighbors test with the five nearest neighbors.

**Nearest Neighbors**

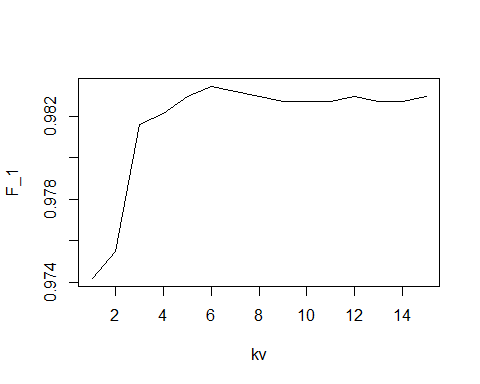
Metric Value  
1 Accuracy 0.967  
2 Sensitivity 0.075  
3 Specificity 0.997

Other than sensitivity, we get good accuracy and specificity metrics. Unfortunately, sensitivity is how well we predict defaults, so this line of testing provides really poor results. We are great at predicting no default (specificity). But considering how massive the proportion of no default clients is, this is no surprise.

We will try to tune this test and see if we can improve our results.

**Nearest Neighbors Tuned**

[1] 6



Based upon that tuning, we will use either the 6, 7, or 8 nearest neighbors to try and predict whether a customer will default.

Default\_predict  
 No Yes  
 No 1930 4  
 Yes 61 6

Metric Value  
1 Accuracy 0.968  
2 Sensitivity 0.090  
3 Specificity 0.998

Other than sensitivity, we get good accuracy and specificity metrics again. our accuracy ends up being the almost same, with trade offs for no/yes. our sensitivity improved slightly, while our specificity gets even closer to perfect. Tuning this data actually yields slightly better results. We tuned the k value in accordance with our highest F score.

Since our whole goal is to predict who will default, the nearest neighbors test really does not yield the results we want.

**Predicting with two variables**

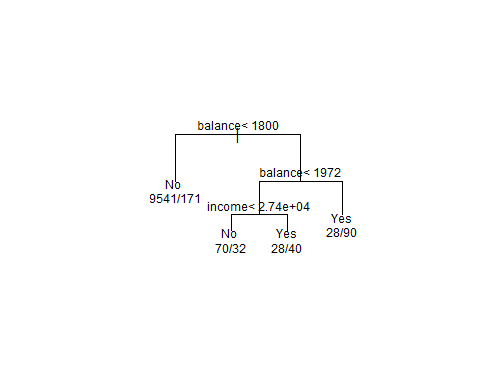
We will try to predict by two variables. The first variable will be balance, and we will state that if balance is above 1500, the customer will default. Our second variable will be income, and we will state that if the customers income is below 20k, they will default. These were selected by eyeballing the visualizations above.

Metric Value  
1 Accuracy 0.744  
2 Sensitivity 0.098  
3 Specificity 0.992

Again, not great results. We have around a 75% accuracy in predicting correctly, but we are mainly predicting those that did not default correctly (specificity). Our sensitivity is better than with the nearest neighbors by .008, but it still is not quite what we are looking for.

We will try to use the rpart function to get a better sensitivity value.

**Rpart**



Here is our rpart graph, showing us which variables to use to determine our prediction of a customers likelihood to default.

Metric Value  
1 Accuracy 0.974  
2 Sensitivity 0.699  
3 Specificity 0.979

As we can see from these results from the Rpart test, we have significantly improved our ability to predict a customers default (sensitivity).

## Conclusion

Based on this rpart test, we can conclude that a customer with a balance greater than 1,972 is likely to default, and if their balance is greater than 1,800, but less than 1,972, then they are likely to default if their income is greater than 27400.  
The Rpart test provides the greatest ability to predict whether or not a customer will default. The nearest neighbor test and using two predictors did not result in a good ability to predict.