Creditworthiness Assessment: is your Client Creditworthy (and how do you make them so)?

Keywords: #python #optimization #lp #linearprogramming #credit #creditworthiness #calculation #numpy #scipy #sympy

1. Abstract

In this white paper we discuss building a piece of software which performs Creditworthiness Assessment based on hundreds of parameters included in loan applications for a popular motorcycle brand. The process involves creating an algorithm which calculates the risk associated with each loan. The goal is to suggest modifications to the loan application parameters, which would adjust them so that they represent good risk for the expected return.

2. Problem statement

As a creditor issuing loans for a popular motorcycle brand, we want to understand, whether a particular loan application represents good risk to us and, if not, how we can revert to the applicant with recommendations on how they can modify their application to pass our creditworthiness criteria.

3. Background

Let’s imagine you issue loans for motorcycles of a popular brand and you have an elaborate spreadsheet which calculates, for every loan application, the risk and worth associated with the applicant. However, the spreadsheet only gives you a yes/no answers in terms of the creditworthiness, no additional feedback is produced. As it happens, the computations of the creditworthiness constitute an Optimization problem in general and a Linear Programming problem (an LP) in particular. This is an advanced computational problem, which would be tricky (although, admittedly, not impossible) to represent in a spreadsheet.

The approach to this problem involves creating dedicated computer code in Python, which solves the Optimization problem using the SciPy package and the SymPy package. The code not only answers whether a particular application "is green" (represents good risk to us), but also returns recommendations on how to aptly modify the the application, if it’s "not green".

4. Solution

Let’s think about a specific loan application. Here’s what it might consist of.

Loan application components

1. Bike price
2. Prior balance
3. Down payment
4. Trade-in value
5. Rebates
6. Service Plan
7. GAP
8. Tire & Wheel
9. Other

In order to qualify as good risk, the loan must pass all of the following tests.

Loan tests

A. LTV Test
B. Monthly payment test
C. Down payment test
D. New or used test
E. Min income test
F. Admin approval
G. Min FICO test

Each of these tests represents a linear combination of the above application components. The loan application, therefore, lives in a multidimensional space where the above values can be modified and for each application it can be stated, whether it is good risk to us or not.

However, if we decide an application is not good risk, what can we do besides simply rejecting it?

Here's where the computational part comes in. This Optimization problem can be aptly investigated using Python code, which looks into the application and searches for potential application improvement opportunities. This algorithm has internal knowledge of the Optimization problem and takes advantage of it by utilizing the capabilities of SciPy and SymPy.

This way we can easily revert to the loan applicant with recommendations, such as:

- reduce the bike price,
- increase down payment,
- resign from service plan,
- reduce insurance,
- etc.

Of course, for each of these a specific value of the change is suggested, so that the applicant can craft and adjust their application to their needs.

This algorithm was in fact developed at Jagan Solutions for a client based in the US, based on specs included in the calculation document, and subjected to elaborate automated testing. We used Linear Programming and Linear Optimization in this Python and NumPy/SciPy project.

5. Conclusion

Giving loans is all about risk so if you are a lender and want to profit in your business, you need to know which application represents good risk to you and which doesn't. This might sound trivial, but here's a question: what do you do with loan applicants whose applications are not good risk to you? It would be a waste to turn them all down and certainly there are high granularity recommendations on how to improve these applications.

In this white paper we have shown that the recommendations can be generated live for each of the applications and, therefore, we can assist the loan applicants in adjusting their applications so that they match the creditor's criteria. What's more, we can do that with acute precision driven by the numerical engines of Python libraries.

This optimization project entailed building software which performed creditworthiness calculations based on hundreds of parameters included in loan applications. In the process we created an algorithm, which calculated the risk associated with each loan. We developed this algorithm based on specs included in a calculation document and subjected it to elaborate automated testing. We used Linear Programming and Linear Optimization in this Python and NumPy/SciPy project for a client based in the US.

If you are a lender, you probably have an advanced method of calculating whether a given loan application represents good risk to you. However, if it doesn't, are you able to revert to your potential loan taker with feedback instructing them how to amend their application so that it passes all the required criteria? Our solutions enable a fully flexible approach to lending, where the potential loan taker is guided through the lending process so that he or she arrives at a deal attractive to both sides.

Do you process lots of loan applications and would like to improve their score instead of turning them down? Are you looking for a way to automate this process by utilizing advanced Optimization resources? We would be happy to harness the know-how associated with this white paper to help you do just that.

6. References

1. https://www.investopedia.com/terms/c/credit-worthiness.asp
2. https://en.wikipedia.org/wiki/Mathematical_optimization
3. https://en.wikipedia.org/wiki/Linear_programming
4. https://www.python.org/
5. https://www.sympy.org/
6. https://scipy.org/
7. https://docs.scipy.org/doc/scipy/reference/optimize.html
8. https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.linprog.html

Would you like to hear more? Please get in touch with us via https://www.jagansolutions.com/contact-us.