MULTI-OBJECTIVE SCORING STRATEGIES TOOL (MOSST)

Student Team: Can Arkali, Jonathan Schniepp, Rebecca Turbish, Andrew Wollerstein

Faculty Advisor: Peter A. Beling
Department of Systems Engineering

Client Advisor: Zvi Covaliu
Analytics Director
Analytic Technology Development
Fair, Isaac and Company, Inc.
8010 Corporate Drive, Suite G
Baltimore MD, 21236
zvicovaliu@fairisaac.com

KEYWORDS: Credit Scoring, Cutoff Score, M-Files, Population Odds, Fitted Odds

ABSTRACT

This project involves the development of an interactive graphing tool to aid lenders in the credit scoring process. Credit scoring is a system in which lenders assess the risk of individual credit applicants to determine whether or not to grant them credit. Our client, Fair, Isaac & Company (FICO), is the world's leading credit scoring consulting company. They aid in evaluating the current credit lending strategies for several financial institutions. These institutions constantly face tradeoffs such as maximizing profit while still maintaining a competitive market share when determining the optimal cutoff score. A cutoff score is represented by a specific credit score number in which all applicants above this number are accepted and applicants that fall below are denied. Our Capstone team designed a software package to help FICO visualize these tradeoffs incurred when managing credit portfolios. We developed an interactive tool to display the possible risks and/or benefits associated with credit portfolios.

INTRODUCTION

This project created and implemented a software program that was designed to aid FICO in examining the tradeoffs associated with a credit card portfolio. Choosing a lending strategy is a crucial decision for a credit card company since it dramatically affects their resulting profit, risk, and market share. For example, when market share increases, the amount of risk also increases. Typically, lending strategies are analyzed using a series of specialized graphs. Prior to our project, FICO did not have software dedicated specifically to creating and analyzing these graphs. The program created in this project, which we call the Multi-Objective Scoring Strategies Tool (MOSST), allows portfolio managers to quickly generate these graphs and visualize the effects of each lending strategy.

MOSST was developed using MATLAB 6.1. Although this software application represents a significant improvement over existing tools, there is appreciable room for improvement in terms of testing and refinement of the system.

This paper further details the need for credit scoring. It also discusses the requirements of our project and the class architecture we designed that encompassed these requirements. In addition, our interface design is included before our conclusions on the project and future recommendations.

CREDIT SCORING

Credit scoring is the process of using a statistical model, known as a scorecard, to transform an individual's credit history and financial standing into a single number indicating the probability that the applicant will repay credit debt. A scorecard is comprised of mutually exclusive categorical attributes, are taken from a credit application and a credit bureau report, with pre-assigned weights. Figure 1 displays a sample scorecard.

Age				
18-25	26-35	35-50	50-62	62 or older
15	**12**	**20**	**24**	**31**
Occupation				
Professional	Clerical	Retired	Sales	Other
35	**24**	**32**	**17**	**15**
Income				
Less than $25,000	$25,000-$49,999	$50,000-$68,000	$67,999-$79,999	over $80,000
12	**18**	**27**	**34**	**38**
Credit History				
Major Derogatory	Minor Derogatory	No Record	Satisfactory	Good
-17	**-9**	**-2**	**10**	**18**

Figure 1: Sample Scorecard

Each attribute is assigned a numerical value to represent the applicant's individual data. For example, the applicant's age, occupation, income, and credit history serve as basic criteria when evaluating credit scores. Thus, an applicant who is 24 years old, works in sales, has an annual income of $40,000, along with a major derogatory credit reference would receive a credit score of: 15 + 17 + 18 – 17 = 33. This score is used to ascertain the applicant's credit risk when compared to a predetermined cutoff score from the institution's lending strategy. If the cutoff score were set at 40, the applicant would be denied credit.

In today's market, credit lending institutions face several tradeoffs when setting a cutoff score. Not all the departments of a lending institution agree on the degree of risk to undertake. The main business objective of the company focuses on raising profit. The marketing department wants to extend credit to as many applicants as possible. However, accepting too many customers will increase the number of bad creditors, a situation that raises concern with the risk management division. A model can help lenders distinguish between these three objectives.

A common decision rule is a cutoff score where individuals with scores greater than or equal to the specified score, s_c, receive credit and those below the score do not. Cutoff policies consider tradeoffs between account risk and portfolio size.

A random risk variable for the population of applicants yields either Good (accounts that pay on time, do not default, or do not prepay) or Bad (accounts that default or pay late) outcomes. Credit services companies assess risk variables and predict the proportion of Good/Bad outcomes. They create a *strategy curve* that quantifies the positive tradeoff between the expected numbers of Bad outcomes that result as you decrease the cutoff and increase the expected number of accepted accounts.

Economic curves that describe the relationships between expected profit, loss and volume help lending institutions to visualize their operation dimensions and evaluate their business objectives. Figure 2 displays a sample Expected Profit versus Expected Loss curve. One would clearly prefer to operate at A* rather than at A, since one would have lower expected losses for the same expected profit. The set of all dominant points is known as the efficient frontier and is shown in pink.

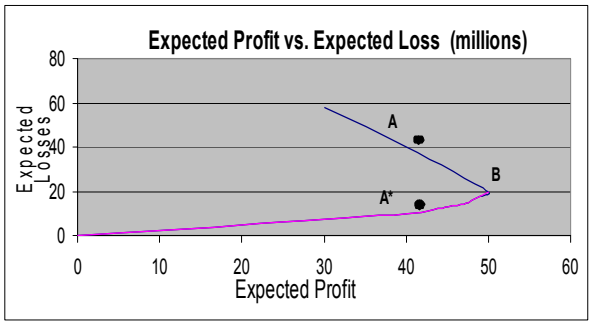

Figure 2: Sample Expected Profit versus Expected Loss curve

REQUIREMENTS

The first and most important was designing the class structure to be modular. FICO wanted to easily update the code for additional graphs in the future.

Secondly, our tool must be capable of calculating and displaying the following credit scoring graphs:

1. ROC Curve- the Receiver Operating Characteristic Curve plots the percent bad rejected versus the percent good rejected.
2. Strategy Curve- plots the expected acceptance rate versus the expected bad rate.
3. E[Profit] v E[Loss]- This plot will indicate the tradeoffs between expected losses and profits for different cutoff scores.
4. E[Profit] v E[Volume]- This plot will indicate the tradeoffs between expected profits and market share for different cutoff scores.
5. Std Deviation of Profit v E[Profit]- This plot will show the distribution of the standard deviation of profit versus the expected profit, indicating the distribution of profits.

6. 3D Curve- plots the expected volume versus the expected loss versus the expected profit. The 3-D graph makes it possible to analyze all three objectives simultaneously. The use of these graphs can determine a more optimal operating point.

For each of the graphs, with the exception of the 3D Curve, the user will be able to display one scorecard or compare two different scorecards. In addition, the user can perform sensitivity analysis for each scorecard, which measures the change in the decision making process due to modification of associated parameters.

Furthermore, Fair, Isaac requires additional features for all graphs. To aid in comparing lending strategies, each curve (or set of curves) can be customized by the user with the following features:

1. *Tracking Points:* By left clicking, a tracking point can be added to display the coordinates of a selected point on a curve. The tracking point will allow the user to move on the associated curve to examine corresponding axis values.
2. *Pop-Up Box:* By right-clicking on a point in curve, users can display information about the selected point. This will allow the user to capture information contained in the selected point to examine risks and tradeoffs at that particular point.
3. *Tangent Line:* For economic curves, the user can draw a tangent line by right-clicking on a curve at the desired point and display the slope of the tangent. For economic curves the slope of the tangent represent the trade-offs between the x and y axis elements.
4. *Markers:* User can add vertical and/or horizontal markers that can be dragged across the range of the graph to facilitate visual comparisons.
5. *Efficient Frontier Overlay:* For certain curve types the user can choose to highlight the efficient frontier. The efficient frontier will be indicated by a color overlay on the existing curve.
6. *Point-specific Uncertainty Intervals*: By right clicking, the user can display uncertainty intervals for the selected point on a curve, as "whiskers" in one, two or three dimensions. The length of the "whiskers" will depend on user-specified confidence level.
7. *Uncertainty Interval Bands:* Smoothed (interpolated) uncertainty bands around the entire curve will be available for display as bands two or three dimensions as appropriate, for a preset level of confidence.
8. *3D Rotations:* For three dimensional curves, the user will be able to rotate the display to view the curve from different angles.

SOFTWARE DESIGN

We built the class architecture around the input our client provided, a text file containing numeric credit scoring information from a population of individuals whose credit payment behavior was observed for several years, starting from the time of application for credit. Figure 3 displays five rows of typical input file. This input file categorizes individual applicants into score bins. For each bin, we assume an average dollar loss associated with Bad applicants in that score range and an average gain associated with Good applicants in that score range. The number of actual Good and Bad customers in a portfolio can be determined using a distribution of Good and Bad applicants, taken from the tailgood and tailbad columns, and the response rate. The response rate indicates the actual fraction of applicants who accept the offered credit.

Score	Tail good	Tail bad	Loss	Gain	Response rate
176	1	1	2979	239	0.024
179	0.997	0.978	5099	193	0.023
181	0.981	0.959	4822	210	0.018
182	0.963	0.926	3765	200	0.019
185	0.959	0.915	6263	219	0.017

Figure 3: Sample Input Data

Our modular, or object-oriented, design consists of several classes. In order to accomplish this, the class architecture must display an object-orientated design. An object-orientated design encompasses individual classes that collectively interact to form the system. Each feature of a class acts completely independent of all others, therefore it is possible to simply add a new curve without altering the existing code.

We created classes for a point, a scorecard, and each individual curve. This class structure offers several benefits to FICO when they choose to upgrade the application. Programmers can easily add, change, or delete items from our program without modifying every object in the system. For example, adding a new curve

would not require any modifications outside the creation of that new curve class. In addition, our object-oriented structure protects the initial data from users. Each object uses data encapsulation to store its data members.

Figure 4 illustrates the modular design of our program. The point and scorecard classes serve as data containers. The point class stores all of the data needed to construct curves and the display data for each point. The point class contains several functions that calculate different credit scoring statistics including the fraction of good customers accepted and the loss per bad customer. The scorecard class serves as a point container. Each scorecard constructs an array of point objects from an input file. This array of point objects contains all of the data from that input file.

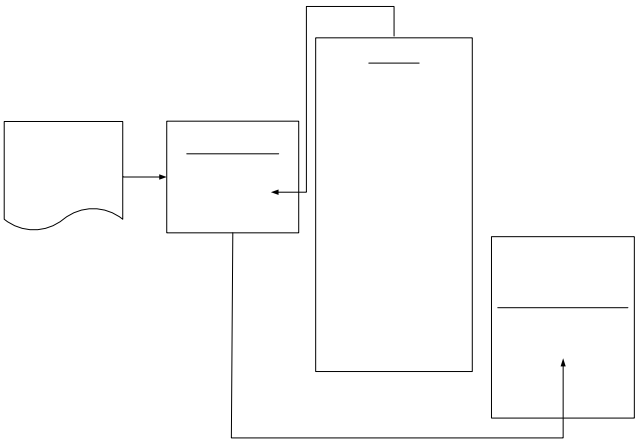

Figure 4: Class Architecture Design

The set of curve classes in our hierarchy allow for the displaying and creation of different curves. We created several curve classes that have the same structure (data members), but different functions associated with displaying that curve in the interface. For example, the class *roccurve* contains functions which calculate the x and y values for a point on a ROC Curve, while the class *strategycurve* contains functions which calculate the x and y values for a point on a Strategy Curve.

Point Class

The *point* class serves as a container for all of the input data. The point class constructor contains several data members which include: index, score, fitted odds, tailgood, tailbad, loss, gain, response rate, good accepted, bad accepted, total accepted, response rate good, response rate bad, and cutoff odds. Each point data member stores a different credit scoring statistic. In addition to these statistics holders, each point contains an index that stores the input file row number associated with that point's data. For example, if a point's index = 8, that point's fitted odds data member stores the fitted odds value provided in the eighth row of the input file.

The *point* class also has several functions that calculate values for data members not included in the input file. These data members, or atomic elements, include: good accepted, bad accepted, total accepted, response rate good, response rate bad, and cutoff odds. In addition, the point class stores the x, y, and z coordinates (if necessary) for all curve types.

Scorecard Class

The *scorecard* class serves as a large container for point objects created from the input file. A scorecard contains a name that identifies the scorecard and an array of point objects, pointsarray. The scorecard class's constructor allows for the creation of points from several input file formats. For example, if the input file does not contain loss, and gain values for each score, the *scorecard* class can provide the data to fill those categories. This data must be input by the user.

All curves have a scorecard, so they can access the necessary data for each point. Users have the option of viewing graphical analysis from multiple scorecards.

Curve Classes

The curve classes transform data objects into graphical figures. Each curve FICO requested has its own class containing all of the data members and functions required for constructing that specific curve. The curve classes we developed include *roccurve, strategycurve, lossvsprofitcurve, profitvsvolumecurve, stdprofitvsprofitcurve*, and *volumevslossvsprofitcurve*.

Each curve class stores a scorecard. The curve classes make all x, y, and z coordinate (if necessary) calculations from the data supplied in the scorecard. They also plot each point that constructs the curve, after determining their coordinates.

INTERFACE

MATLAB has the capability to serve a dual purpose: a graphing utility and a development environment. Our design can be described as two parts that interact, the interface and the system architecture. The interface serves as the output to and input from the user; it also controls the actions the user can perform in each situation. Our system architecture provides objects and functions that serve as building blocks for lending strategy analysis.

Our interface design is displayed in Figure 5. The user must first select the type of curve he/she wishes to plot. Next, the user must browse through folders to select the scorecard to be displayed.

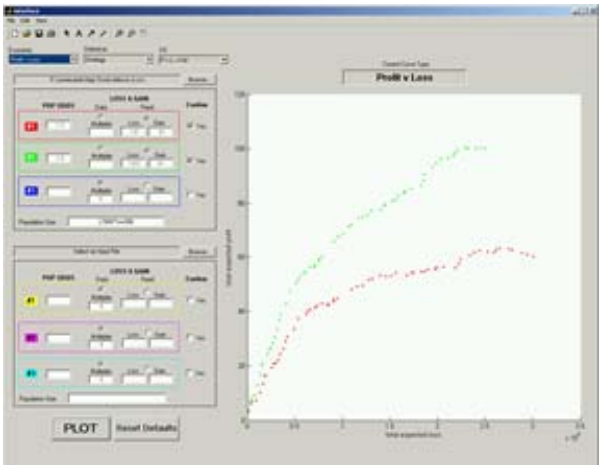

Figure 5: Interface Design

Sensitivity Analysis

MOSST enables the user to perform sensitivity analysis for each curve. The user may alter the value of the population odds (the odds of a given customer being "bad" cardholder) and either the fixed loss to gain ratio (which is a fixed value for all score bins in a portfolio) or the score-dependent loss to gain ratio (which is different for each score bin in a portfolio). The user is able to display each curve based on a specified combination of values for:

Population odds; *and*

1. User specified loss to gain ratio (fixed); *or*
2. The score-dependent loss to gain ratio, which can be multiplied (or not) by a user-specified factor.

Interactions with the Interface

A scorecard is created when the user selects the check off box under Confirm. This conformation process checks the validity of the user-supplied parameters and disables all input boxes, so that approved user entries cannot be changed, unless the confirm box is unchecked. When the user clicks on the Plot button, our software application invokes the array of points for that scorecard, retrieves all the information necessary to display individual points for the chosen curve type and finalizes the graph.

CONCLUSIONS

We built an intelligent software application, MOSST that can be utilized by FICO analysts to assess risks and tradeoffs inherent to credit portfolio management. Producing statistical, economic and 3D graphs, our application provides a flexible and interactive environment to manipulate available curve types by realizing the conversion process of client data into actual components of any given credit scoring system.

We used MATLAB 6.1 to develop the application. Several function M-files in MATLAB directories compose the system architecture. A function M-file is an external text file that communicates with MATLAB workspace trough input and output variables. Our software application consists of several function M-files to instantiate point, scorecard and curve objects, whereas a single function M-file manages the application's interface. If compiled as a stand-alone, the application does not require a MATLAB 6.1 license and FICO can distribute it freely.

Future Expansion

The software application is characterized by two essential attributes extracted from requirements and specifications posed by FICO. The Object-oriented class architecture creates and sustains the main components of analysis tools, whereas the interface determines the flow of the software application and coordinates the interrelationships between graphical user interface capabilities and functionality of the class architecture. The modular class structure provides FICO programmers with a convenient framework for expanding the functionality of the application. The interface provides FICO analysts with an interactive environment that captures several scenarios associated with lending strategies.

REFERENCES

Ballard, Geoff. "Algorithms and Software for Credit Portfolio Optimization." Thesis. U of Virginia, 2000.

Beling, Peter, Covaliu, Zvi, and Oliver, Robert M. "Dominant Score Policies And Efficient Frontiers." Presentation at the Edinburgh Credit Scoring Conference. September 2001.

Leonard, Kevin. "The Development of Credit Scoring Quality Measures for Consumer Credit Applications." *International Journal of Quality & Reliability Management* n4 (April 1995): 79-85.

Oliver, Robert M., and Wells, Eric. "Efficient Frontier Cutoff Policies in Credit Portfolios." Presentation at the Edinburgh Credit Scoring Conference. September 1999.

Thomas, Lyn C. "A Survey of Credit and Behavioral Scoring; Forecasting Financial Risk of Lending to Consumers." International Journal of Forecasting Vol.16 (2000):149-186.

BIOGRAPHIES

Can Arkali is a fourth-year Systems and Information Engineering major, concentrating in Computer and Information Systems from Istanbul, Turkey. His principal contribution to the project involved design of point and scorecard classes and implementation of individual curve type classes. Mr. Arkali has accepted a position in Sprint's Associate Engineer Program in Reston, VA.

Jonathan Schniepp is a fourth-year Systems and Information Engineering student with a concentration in Management Systems from Hershey, PA. He designed and developed the interface of the application. Mr. Schniepp will work for Booz, Allen & Hamilton next year in Great Falls, VA.

Rebecca Turbish is a fourth-year Systems and Information Engineering student with a concentration in Management Systems from Poquoson, VA. Her principle contribution to the project involved the development of the class architecture and writing a user's guide for MOSST. Currently, Miss Turbish has no specific plans for next year.

Andrew Wollerstein is a fourth-year Systems and Information Engineering major, concentrating in Economics, from Roslyn Heights, NY. His principal contribution to the project involved designing and coding the class architecture that handles all mathematical calculations and object creations within the interface. Mr. Wollerstein will put his IT skills to work next year at Eveready Insurance Company in New York City.