A Submodular Optimization Approach to Accountable Loan Approval

Kyungsik Lee[1], **Hana Yoo**[1], **Sumin Shin**[1], **Wooyoung Kim**[1],
Yeonung Baek[1], **Hyunjin Kang**[1], **Jaehyun Kim**[1], **Kee-Eung Kim**[2]

[1] Hyundai Capital Services, Korea
[2] Kim Jaechul Graduate School of AI, KAIST, Korea
{kyungsik.lee, hana.yoo, sumin.shin, wooyoung.kim}@hcs.com, kekim@kaist.ac.kr

Abstract

In the field of finance, the underwriting process is an essential step in evaluating every loan application. During this stage, the borrowers' creditworthiness and ability to repay the loan are assessed to ultimately decide whether to approve the loan application. One of the core components of underwriting is credit scoring, in which the probability of default is estimated. As such, there has been significant progress in enhancing the predictive accuracy of credit scoring models through the use of machine learning, but there still exists a need to ultimately construct an approval rule that takes into consideration additional criteria beyond the score itself. This construction process is traditionally done manually to ensure that the approval rule remains interpretable to humans. In this paper, we outline an automated system for optimizing a rule-based system for approving loan applications, which has been deployed at Hyundai Capital Services (HCS). The main challenge lay in creating a high-quality rule base that is simultaneously simple enough to be interpretable by risk analysts as well as customers, since the approval decision should be accountable. We addressed this challenge through principled submodular optimization. The deployment of our system has led to a 14% annual growth in the volume of loan services at HCS, while maintaining the target bad rate, and has resulted in the approval of customers who might have otherwise been rejected.

Introduction

Finance companies process loan applications by assessing the customer's credit score and estimating the risk to decide whether to approve or decline the application. During this underwriting process, the credit is assessed via various estimation models such as internal risk scores or external credit bureau (CB) scores, commonly based on the estimated probability of default within the next 6 to 12 months. Then, a cut-off score would be set for approving or declining the loan application, such as "deny the loan if the score is under 700." The *loan approval rule* is usually comprised of such binary decision rules that take into account not only the estimate from the credit scoring model but also a wide range of other credit information.

Korean financial market is known to be relatively strong in the availability of credit information sources: there are several CBs which provide finance companies with not only the main CB scores for the general population but also various sub-CB scores for specific segments (e.g. customers under a debt workout program), as well as other detailed information sources such as credit summaries, alternative summaries, and income estimates. Thus, financial companies in Korea are thus highly motivated to leverage these sources of credit information to make more effective decisions on loan applications, which is especially true for those companies specialized in the credit finance business in order to gain a strong competitive edge.

The availability of various credit information has facilitated the development of advanced credit scoring models: while logistic regression has been the conventional model of choice, more sophisticated machine learning (ML) models such as XGBoost (Chen and Guestrin 2016) and LightGBM (Ke et al. 2017) are now being put to practical use. However, when it comes down to the final decision whether to approve or reject loan applications, the analyst would still prefer manually defining decision rules based on credit risk analysis for accountability. Thus the ultimate goal is to develop a systematic method for optimizing the loan approval rule, automating the conventional process involving manual construction and evaluation by analysts. Yet, the decision rules behind the loan approval rule should remain interpretable in order to comply with various regulations in the financial market, thus precluding most of the advanced deep learning models.

In this paper, we detail the development and deployment of a system at Hyundai Capital Services (HCS) for automatically constructing an interpretable loan approval rule. Named the HCS Automated Loan Approval Rule Engine, this system was deployed incrementally, beginning with a specific customer segment in August 2021 and ultimately expanding to encompass all of our customers by 2022. Compared to the previously operational approval rule, crafted by human expert analysts, the system has contributed to a yearly volume growth of 14% for credit loan service, equivalent to several tens of million USD. The system is now actively employed in shaping approval rules for credit loan services, and expanded to auto loan services. For the purpose of this paper, we will focus on the credit loan service as our case study.

rule #	atomic rule	# good cases	# bad cases	estimated bad rate
1	if (A <457) & (B >340K) then decline	220	37	14.4%
2	else if (C >= 2) & (D >= 0.9) & (F >= 3) then decline	225	29	11.4%
3	else if (G <513) & (H >= 2) then decline	254	26	9.3%
...				

Table 1: An example of loan approval rule. Features are anonymized and the numbers are not real.

Background

In the realm of credit loan services, risk management analysts construct new loan approval rules on a monthly basis to ascertain customer eligibility, as well as to establish personal credit limits for those deemed eligible. These credit limits are then presented to the customers in conjunction with the interest rate on the loan. It is important to note that the rules for determining both the credit limit and the interest rate are distinct processes, set up separately from one another. These rules fall outside the purview of this paper, which solely concentrates on binary decisions related to approval, based on the eligibility of the loan application.

Loan approval rules are commonly structured as a list of if-then conditions, with each *atomic rule* defining specific criteria based on credit information. If a customer's profile meets any of these conditions, the loan application is declined; if no atomic rules are activated, the application is approved. An illustrative example of loan approval rule can be found in Table 1. The atomic rules within the list are organized by priority, usually determined by the estimated rate of 'bad' customers[1] that fall under each condition. As a result, atomic rules associated with higher estimated bad rates are placed higher in the list, with the exception of those associated with regulatory compliance (e.g., customers with a Debt-to-Savings Ratio higher than 100% are ineligible for a new loan, due to the regulatory requirement imposed by the Ministry of Finance and Economy), which are positioned at the top. This standard practice for arranging the order of atomic rules assists analysts in determining the cut-off point. Typically, they first construct a sufficiently extensive list and then select a prefix of atomic rules that aligns with the overall operational risk criteria concerning bad rates. The preference among risk analysts is often for a more succinct rule (i.e., a shorter list) to facilitate human inspection. Consequently, prioritizing atomic rules with higher estimated bad rates early in the list is generally helpful, as these rules tend to reduce the overall bad rate more effectively than others. The clarity of this model for loan approval also enables the provision of intuitive and convincing explanations to both customers and financial authorities, making it a valuable tool in the accountable lending process.

Our objective was to develop a framework to optimize the loan approval rule model, aiming to substantially enhance the underwriting process beyond what could be achieved through approval rules manually crafted by risk analysts. To meet this goal, we were tasked with addressing the following specific business requirements of the framework:

- **Accountability:** The resulting rule should be readily interpretable, necessitating the use of a white-box model. This model must comprise fully human-interpretable atomic rules, each formatted as logical combinations (using "and," "or") involving no more than four features with comparison operators (>, <, ==) to ensure simplicity.
- **Monotonicity:** The atomic rules must be constructed in a manner consistent with stakeholder's domain understanding, whether for customers raising complaints or regulatory auditors. For instance, a rule such as "recent delinquencies must be two or less" may not suffice in explaining a rejection if another customer with more delinquencies is approved. Therefore, the decision must adhere to *monotonicity* constraints on relevant feature subsets, like "2 *or more* months of delinquency cases in the past 3 years & recent credit card limit utilization rate 90% *or more* & having loan balances at 3 or more institutions."
- **Conformance:** Preferably, the estimated bad rates of the atomic rules should be arranged in decreasing order, maintaining alignment with traditional, manually constructed loan approval rule. A deviation from this pattern may render risk analysts uneasy, complicating the final human inspection phase.
- **Analysis Tools:** The system must offer user-friendly simulation toolkits and visualization interfaces, assisting analysts in fine-tuning the approval rule. Specifically, they should quantify the trade-off between the volume (total credit limit of approved loans) and the probability of bad loan. Lowering the target bad rate to minimize risk will naturally reduce volume by declining more applications, and vice versa. This trade-off analysis is vital for risk management-based business decisions. We have also developed models to predict customer responses and credit line utilization rates to forecast volume, although a detailed coverage of these additional models is outside the scope of this paper.
- **Automation:** The entire process, from data preparation to the final construction of loan approval rules, must be automated. This includes the potential for auto-retraining at regular intervals. Historically, this task depended on manual analysis by credit risk team members, limiting the frequency of rule updates, which hinders quickly responding to internal and external environmental changes.

In the subsequent section, we introduce our automated framework that substantially refines the procedure for op-

[1]Throughout this paper, we use the more generic term 'bad,' as the term 'default' carries a stricter definition under Basel (Wagner 2017)

Step	# features	Selection Criteria
Total	1656	- internal data (scores, loan history, weblog, legacy rules, etc.) - All data that can be obtained from credit bureaus
1^{st} Selection focus on predictive power	936	- Remove features with low information value (some critical credit history information is forced to remain even if the IV is low due to small coverage) - Among high correlated features, remove the one with low IV
2^{nd} Selection focus on explainability	358	- Remove features non-explainable for rejection (e.g. web/mobile channel usage) - Remove features that rely on specific circumstances (e.g. overseas credit card usage was not reliable during pandemic)

Table 2: Feature selection procedure

timizing and analyzing the loan approval rule, while concurrently satisfying the previously outlined requirements. Initially, we delineate the candidate set of atomic rules, ensuring that they possess both substantial support and adequate discriminative accuracy, defining this collection as the *search space*. This search space is constituted by the atomic rules corresponding to leaf nodes in a decision tree or an ensemble model trained on pertinent loan data. To maintain the search space at a manageable size, we employ a combinatorial enumeration of features or sample them utilizing domain expertise. Ultimately, the approval rule is derived by selecting an optimal subset of atomic rules within the search space, aligning with the operational objective, whether in terms of the overall bad rate (e.g., 5%) or the total volume (e.g., 100 billion KRW).

Methodology

Dataset

We compiled a dataset comprising active customers who availed the personal loan service with Hyundai Capital Services as of August 2020. Each customer was assigned a binary label, designating them as either ”good” or ”bad” depending on whether their loan was classified as bad within a 12-month period from the start date. Specifically, customers with a delinquent history of 60 or more days within this 12-month interval were defined as ”bad,” while the remaining customers were categorized as ”good.”

However, customers with a delinquency period ranging between 10 and 59 days presented a challenge as they did not neatly fall into either the ”good” or ”bad” categories, potentially weakening discrimination. Consequently, these customers were excluded as indeterminate. Data points that were already classified as ”bad” (i.e., having been delinquent for 60 or more days at the starting date) were likewise excluded.

In terms of feature selection, we began with an extensive list of 1,656 candidate features. These encompassed predictions from credit scoring models (such as CB scores and internal scores), external credit information (including delinquency history, loan/repayment history, and credit card usage history), and internal behavior information (e.g., loan history with HCS, delinquency history, weblogs). This initial set was subsequently refined down to a total of 358 features through a rigorous selection process. Features were initially chosen for their top predictive power, as determined by information value (IV) with auto-binning, and further filtered to remove those with high correlation. This was done to ensure appropriate data coverage and to consider the correlative relationships among the variables (as detailed in Table 2).

Search Space Builder

The search space comprises a set of useful atomic rules that serve as candidates for forming the optimized loan approval rules; these are equivalent to the rules present in each line of Table 1. These atomic rules are identified by uncovering frequent patterns within the data, a well-established field of study within data mining (Agarwal, Srikant et al. 1994; Han and Pei 2000; Han, Pei, and Yin 2000). In the present study, we utilized XGBoost (Chen and Guestrin 2016) to populate the search space with pertinent rules. XGBoost is a widely-used data mining algorithm that builds an ensemble of decision trees.

The generation of candidate rules involved training the ensemble of decision trees repeatedly to predict the binary labels. We then extracted the logical conditions corresponding to each path from the root node to the leaf nodes. A column sampling rate constraint was applied as a hyperparameter, enabling the production of diverse trees through repetition. In alignment with the accountability requirement, we capped the maximum tree depth at four to prevent the formulation of overly complex rules that might hinder human interpretability.

Below, we describe some noteworthy settings for running the XGBoost for constructing the search space:

1. **Monotonicity:** The atomic rules within the search space must adhere to the monotonicity requirement concerning relevant features. For instance, an increase in the number of delinquencies is indicative of a negative trait. Therefore, rules involving this feature must employ the greater-than condition ($>$) with a specific value. Conversely, the number of days since the last delinquent date is preferably large, requiring the application of the less-than condition ($<$) with a particular value. While XGBoost has the capability to manage monotonicity constraints, it doesn’t entirely satisfy our requirement: it merely enforces the monotonic relationship between pertinent features and the predicted probability of the label. As such, we continue to verify the monotonicity constraint of each rule during extraction, omitting any that are violated.
2. **Minimum Precision:** The predictive efficacy of a candidate atomic rule can be depicted through its precision, which is the percentage of actual bad instances among those predicted as bad by a specific leaf node within a

node	condition	% total	estimated bad rate	monotonicity	support	precision	insert into search space?
1	(A>= 3) & (C>= 17) & (B>= 1.5)	1.3%	79.6%	O	O	O	O
2	(A>= 3) & (C<17) & (D>= 640)	0.6%	55.8%	X	O	O	X
...							
5	(A>= 3) & (C<17) & (D<640)	1.1%	28.8%	O	O	X	X
6	(A<3) & (D>= 640) & (D<15130)	20.3%	21.4%	X	X	O	X
7	(A<3) & (D<640) & (A>= 1)	5.1%	15.3%	X	O	O	X
8	(A<3) & (D<640) & (A<1)	73.9%	5.5%	X	X	X	X

Table 3: Extracting atomic rules corresponding to leaf nodes from a decision tree of depth 3, and checking the condition for their validity. Features are anonymized.

tree. Our objective here is to ascertain a sufficiently extensive set of rules that put forth high bad rates for inclusion in the final approval rule. Accordingly, the candidate atomic rules are limited to those leaf nodes that demonstrate precision at or above a designated threshold.

3. **Adequate Support:** An atomic rule's significance is contingent on the number of customers it filters falling within a particular range. For example, an atomic rule that excludes 0.2 million out of 1 million customers is unsuitable, as it would result in an excessive rejection rate due to one sole condition. Conversely, a rule that filters merely five individuals from 1 million customers would contribute to erratic rule coverage, fluctuating based on the time of deployment. Such inconsistency could undermine operational stability and complicate the explanation of the approval rule to stakeholders. Therefore, we have eliminated from consideration any leaf nodes not meeting the specified minimum/maximum support criteria.

Table 3 provides an illustration of the process by which rules are extracted from a decision tree, in accordance with the aforementioned settings. Furthermore, to enhance the diversity of the rules within the search space and to augment their predictive accuracies, we implemented the subsequent techniques that leverage domain-specific knowledge:

4. **Segmentation:** In instances where the company has a strategically targeted customer segment, it is possible to derive a more specialized approval rule by training the tree-based model on that particular subset of customers. For instance, when targeting customers who already possess loan balances with watch-listed financial companies, the tree may set a lower threshold on the number of past delinquencies in comparison to a tree trained on the entire customer data set. We then incorporate this specialized rule into the search space, conjoined with the condition "Having loan balance at watch-listed financial companies == True," which defines the particular customer segment.
5. **Feature Dropping:** Certain features with pronounced predictive power (e.g., CB scores and previous delinquencies) are prone to being chosen as a node in the majority of the trees, leading to the possibility that analogous rules may be repeatedly added to the search space. To mitigate this concern, we selectively excluded some features that are too highly correlated with the label before constructing the trees, in order to extract additional and diversified rules. Moreover, we endeavored to further diversify the search space by building trees after excluding all features except those most frequently absent from the rules.

A simple calculation of combinatorial enumerations of all the features yields the search space size of $\binom{358}{2} + \binom{358}{3} + \binom{358}{4} \approx 680$ million atomic rules, but using the aforementioned settings, we ended up with the search space size of approximately 50,000 atomic rules.

Rule Construction Engine

An ideal loan approval rule should minimize the overall bad rate of the approved customers while maximizing the volume (i.e. aggregate credit limit of all approved customers) which determines sales revenue. Note that these two objectives are conflicting with each other: in order to reduce the bad rate, the loan approval needs to be conservative by declining more customers, but this would end up reducing the volume. On the other hand, in order to increase the volume, we need to approve more customers, but this would increase the bad rate. We address this trade-off by framing the problem as a constrained optimization problem: find the set of rules that would minimize the overall bad rate while making the volume above a certain threshold.

We address the problem as submodular optimization (Fujishige 2005) by defining submodular functions on the set of rules that reflect the two objectives. Formally, given a finite set X, a set function $f : 2^X \to \Re$ is called *submodular* if for all subsets $A \subset B \subseteq X$ and $x \notin A$, it holds that $f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B)$, i.e. it satisfies the "diminishing returns". Although finding the optimal set that maximizes f is known to be computationally intractable, there are many algorithms with simple heuristics that achieve strong approximation bounds (Krause and Guestrin 2005; Nemhauser, Wolsey, and Fisher 1978; Sviridenko 2004).

Given a set R of all atomic rules that comprise the search space, we first define function $f : 2^R \to \Re$ that counts the number of bad customers in the dataset, correctly filtered out by the set of atomic rules chosen by the loan approval rule. Since each atomic rule filters out a potentially overlapping subset of bad customers in the dataset, f is naturally submodular. By maximizing f, we obtain the optimal set of rules that filters out as many bad customers as possible. On the other hand, we can notice that there is a possibility that we end up incorrectly filtering out too many good customers since f only counts bad customers. In practice, this was not a big problem since the search space was constructed in such a way that all the rules are of high precision.

At the same time, the loan approval rule should not lose too much in volume. For this objective, we define function $g : 2^R \rightarrow \Re$ that calculates the loss in volume from the filtered-out customers. Note that customers have different credit limits, and thus some rules will incur a higher loss in volume and vice versa, even if they declined the same number of customers. Again, since each rule filters out a potentially overlapping subset of customers, g is naturally submodular.

Using f and g, we obtain submodular cost submodular cover (SCSC) and submodular cost submodular knapsack (SCSK) formulations for constructing the optimal rule (Iyer and Bilmes 2013):

$$\text{SCSC: } \min g(X) \text{ s.t. } f(X) \geq a$$
$$\text{SCSK: } \max f(X) \text{ s.t. } g(X) \leq b,$$

where the SCSC formulation states that we want to maximize the volume while making sure that we don't approve too many bad customers, and the SCSK formulation states that we want to minimize the number of bad customers being approved while making sure that the volume is above a certain threshold. Although these two problems are computationally intractable, there are a number of practical algorithms with strong worst-case approximation guarantees (Narasimhan and Bilmes 2005; Iyer and Bilmes 2012, 2013).

In this work, we adopted a greedy algorithm that worked reasonably well on our dataset. The final rule is obtained by iterating the following steps that explore the search space and greedily select rules toward the target volume:

1. If there are any atomic rules that must be included to comply with regulatory requirements, include them as the initial set of atomic rules.
2. Identify the remaining pool of customers in the training data that are not filtered out by the current set of rules, and calculate the support and the precision on the remaining pool for each of the atomic rules in the search space.
3. Select the atomic rule with the highest precision (i.e. identify the maximum number of bad customers in the remaining pool) while satisfying the minimum size condition and not exceeding the volume loss constraint. For robustness, simulate the atomic rule on the validation set, and if the filtering size and the precision on the validation set differ more than some threshold, move on to the second-best atomic rule. Repeat this until an atomic rule is found, and remove the rule from the search space as it has now been selected.
4. Terminate if no atomic rule could be found. If not, repeat the search loop by continuing to step 2.

This algorithm is essentially the greedy algorithm for SCSK (Iyer and Bilmes 2013), except we use the validation set for the robust selection of rules. Although this algorithm has a very loose worst-case approximate optimality guarantee, we found it working reasonably well on our dataset, significantly surpassing the previous operating approval rule developed by human-expert risk analysts. Table 4 shows an example of the loan approval rule found by the greedy algorithm. Interestingly, when we use this greedy algorithm, rules with higher bad rates appeared at the head of the list in all of our experiments, thus coinciding with the traditional practice in manually crafting the loan approval rule where the atomic rules are ordered by their estimated bad rates.

rule#	rule	decision
1	Reject criteria due to regulatory policy (e.g. Debt-to-Service-Ratio >1.0)	reject
2	Reject criteria due to internal policy (e.g. currently delinquent)	reject
...		reject
99	(X <3) & (D >1) & (Y <300)	reject
100	(E >= 1) & (F >= 90) & (Z >= 7.5)	reject (cut-off point)
101	True	approve

Table 4: An examplar rule found by our rule construction engine. Features are anonymized.

To the best of our knowledge, employing submodular optimization to build rules for loan approvals has not appeared in the literature, which highlights the novelty as well as the practicality of our approach.

Simulation Toolkit

Finally, it was very important that the results from the rule construction algorithm were clearly communicated to the risk analysts and the business decision makers. For this, we needed to conform to the conventional way of consolidating the final loan approval rule by human experts. Traditionally, the risk analysts manually crafted the list of atomic rules ordered by their importance (i.e. precision), and determine the cut-off point where the rules below are removed from the list. By moving the cut-off point up or down, the analysts simulate the trade-off between the overall bad rate and the volume, and the final cut-off point is determined based on the nature of the financial product or the situation surrounding the company or the market.

We thus developed an interactive simulation toolkit that visualizes the trade-off, and let the human expert determine the cut-off point, shown in Figure 1. To create the simulation plot, we first constructed a sufficiently large rule list by running the greedy algorithm without the volume constraint and took each position as a potential cut-off point to generate data for the graph. Once the human expert determines the final cut-off point, the greedy algorithm for SCSK in the previous subsection was run once again to further optimize on the overall bad rate.

Results

In order to compare the performance of the loan approval rule from our system against that of the operational base-

	% Ineligible	Precision	Recall	Accuracy	F1-score	Explainability
SUBMOD	10.5	3.5	66.2	90.2	6.6	Y
BASE	10.4	2.7	52.2	89.9	5.2	Y
LR	10.5	3.1	58.6	90.0	5.8	Y
XGB	10.3	3.5	65.7	90.4	6.7	N
DNN	10.3	3.6	66.3	90.5	6.8	N

(unit for numbers : %)

Table 5: Performance comparison of loan approval rules on our personal loan services data.

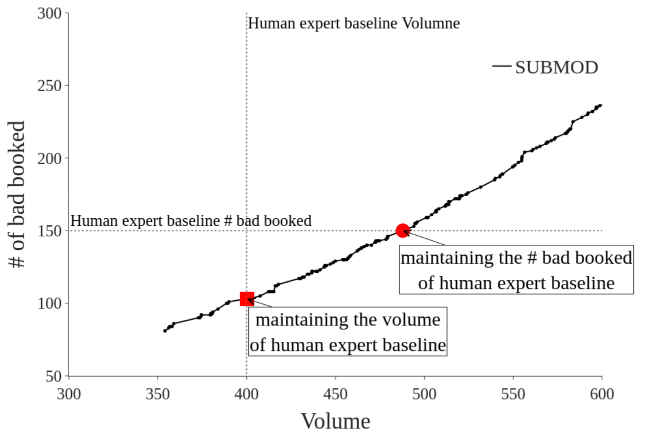

Figure 1: The simulation plot of the # bad booked vs. the volume

line rule built by an expert analyst, we compare these two rules on the real personal loan service data: we set up both rules to reject 10% of applicants as being ineligible, resulting to approve the remaining 90% of them as eligible for financing. To compare the performances of the previously operational rule (BASE) vs. the optimized rule (SUBMOD), we applied them to samples not included in the population used for training (out-of-time test data). In addition to these two strategies, we also tested three credit scoring models, namely logistic regression (LR), XGBoost (XGB), and deep neural network (DNN)[2] with the same conditions. In order to assess how well the model was able to filter out (i.e. reject) customers who are likely to go into delinquency, we calculated the precision, recall, accuracy, and F1-score metrics commonly used for evaluation in binary classification tasks using the ground-truth label indicating whether the customer actually went into bad over the following 12 months. As shown in Table 5, the SUBMOD showed significantly better performance than other strategies across all of the metrics, and on a comparable level with the much more complex ensemble model from XGB. Its precision jumped approx. 29% vs. the baseline strategy (2.7% to 3.5%) while the ineligible rates were set to be similar at 10%, suggesting that the SUBMOD was better at rejecting bad customers, thus achieving higher efficiency in loan approvals. This performance gain over BASE is mostly due to the fact that our rule engine automatically generates diverse atomic rules in the search space builder, going beyond the limitation of manual preparation by human experts. Thus, we were able to find the optimal point at much higher precision. This directly implies higher profitability for the company which can now reject individuals more likely to go into delinquency and instead approve better quality customers. On the other hand, the ensemble model from XGB showed slightly higher performance but cannot provide a transparent explanation to customers, and is therefore unsuitable for deployment. The SUBMOD rule, on the other hand, is clearly explainable yet shows just as high performance as XGB. Based on this result, the management at the company decided to replace the previously operational rule (BASE) built by risk analysts with the optimized rule (SUBMOD) constructed by our rule engine. In addition, we can obtain a plot of the F1-score versus the percentage of customers rejected by moving the cut-off point in the list, thus providing a comparative analysis against the human baseline strategy and other models (Figure 2). This is similar to the Receiver-Operating Characteristic (ROC) curve analysis, but provides a more intuitive picture for business decision makers since the control parameter is given in terms of operational condition (i.e. how many customers are we willing to reject, and thus reduce the volume?)

Lastly, using our rule engine, we uncovered insights about alternative features not previously considered by risk analysts. For example, certain features like consistent payment of insurance fees or a recent car purchase were overlooked by analysts due to their lack of intuitive explanation in isolation. Yet, when paired with commonly known negative credit features (e.g., delinquency history and multi-debt information), these attributes proved valuable in identifying good customers who might otherwise be rejected.

Application Use

Once we observed promising performance of the optimized rule by back-testing with the initial dataset gathered for development, we tested the optimized rule for 3-month period in Aug-Oct, 2021 by deploying it as an override. Here, the override refers to a policy of maintaining the current baseline rule for operation, while providing financing opportunities to customers who had been rejected by the baseline rule but

[2]Note that DNN cannot be employed in practice due to internal and external strict regulatory requirements on interpretability. Similarly, XGB, although an interpretable white-box model, cannot be used as well since it tends to build very complex unreadable rules from ensembles. On the other hand, the Decision Tree with an equivalent maximum depth of 4 would inevitably show very poor performance due to limited representational capacity.

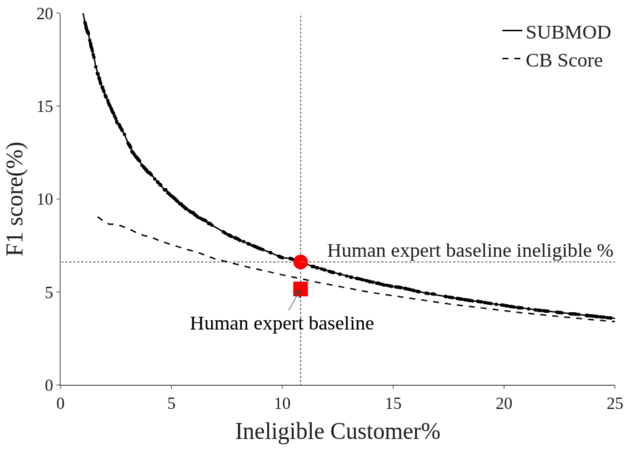

Figure 2: Performance comparison of SUBMOD (solid curve) against human expert baseline strategy (square dot) and logistic regression (CB score, dotted curve)

	predicted bad rate	actual bad rate
BASE	3.86%	3.83%
SUBMOD override	2.54%	2.58%
Overall	3.71%	3.70%

Table 6: The live-test performance comparison of predicted and actual bad rates.

found to be eligible by the optimized rule. We then followed up this entire active customer base to see if the approved loan actually became bad or not (bad within 12 months) . The actual bad rate for that customers was similar to the estimated bad rate, shown in Table 6, which was the objective for the override test, while volume grew by 14%, live-validating the effectiveness of the optimized rule.

In addition to the performance upsides such as volume growth and reduced risk for the company, our system also offered substantial benefits in terms of internal operational structure efficiency. The company was able to move much more nimbly to address the rapidly changing market, by replacing the manual rule construction pipeline and updating the loan approval rule as often as needed. Also, it was easy to incorporate new source of information on credit score due to the automatic rule construction engine. The clarity of the rules eased the burden of cross-checking and reviewing to identify potential risks that would have been difficult to address pre-emptively with a black-box model, and allows risk analysts who are unfamiliar with machine learning to have a full control on the rule. It also offered a significant advantage in terms of consumer rights protection, since the decision can be explained in a transparent manner to customers.

We have also broadened the use of our rule construction engine to refine debt collection strategies, classifying delinquent customers for specific recovery actions. Our primary objective is to enhance collection rates by adjusting action intensity to each customer's recovery likelihood. Employing the same submodular optimization approach, our engine crafts allocation rules. These rules prioritize in-person visits for those deemed highest risk, human calls for intermediate risk, and call-bot interactions for those at the lowest risk. Over the past 9 months, this methodology has proven effective, safeguarding delinquent balances of several million USD.

Deployment and Maintenance

Since initial deployment in Aug 2021, the optimized loan approval rule has now been applied to across all customer segments in Nov 2022, so pre-approval is being solely conducted by the optimized rule. The performance results are gathered on a monthly basis, which runs through fully automated processes on the internal MLOps system (Figure 3). The debt collection allocation uses the same rule optimization pipeline since Oct 2022.

Since its initial deployment, the loan approval rule has been regularly revised to account for the evolving profiles of incoming customers due to economic shifts. To achieve this, we first implemented a soft-retraining process every 3 months: while retaining the existing search space, we update the training data to identify an optimized rule. This entails automatically collecting training data via the MLOps system, running the rule construction engine, and subsequently back-testing and simulating the performance of the new rule set. This allows us to determine if the current approval rule requires replacement. When fresh insights emerged from exploring features beyond credit information, we conducted full re-training either semi-annually or annually. This involved updating the search space with the most recent data to incorporate these new features.

Our real-world application faced certain challenges since the deployment. While the overall delinquency rates improved, we noticed a 2-percentage point increase among customers with credit scores below 700. This metric has traditionally been used to monitor customer profile portfolios. Such outcomes made it challenging to secure full support from internal risk analysts and organizational stakeholders. To address this, we've adopted a hybrid operational approach. This model merges optimized rules from distinct search processes, some of which include primary score determinants from the traditional human-driven strategy. This combined approach not only simplified internal communication but also ensured a seamless transition.

Related Work

Optimizing loan approval rules comprises two primary tasks: estimating the bad rate, often referred to as credit scoring models, and constructing a decision rule to finalize loan application outcomes as approved or declined.

Historically, the bulk of research has concentrated on leveraging machine learning to refine credit scoring models, while simpler decision rules, such as thresholding based on predicted scores, were commonly employed. Extensive literature exists on this subject (see, for example (Dastile, Celik, and Potsane 2020)). It encompasses a diverse array of machine learning models, ranging from logistic regression (Cramer 2002) and decision trees (Quinlan 1986;

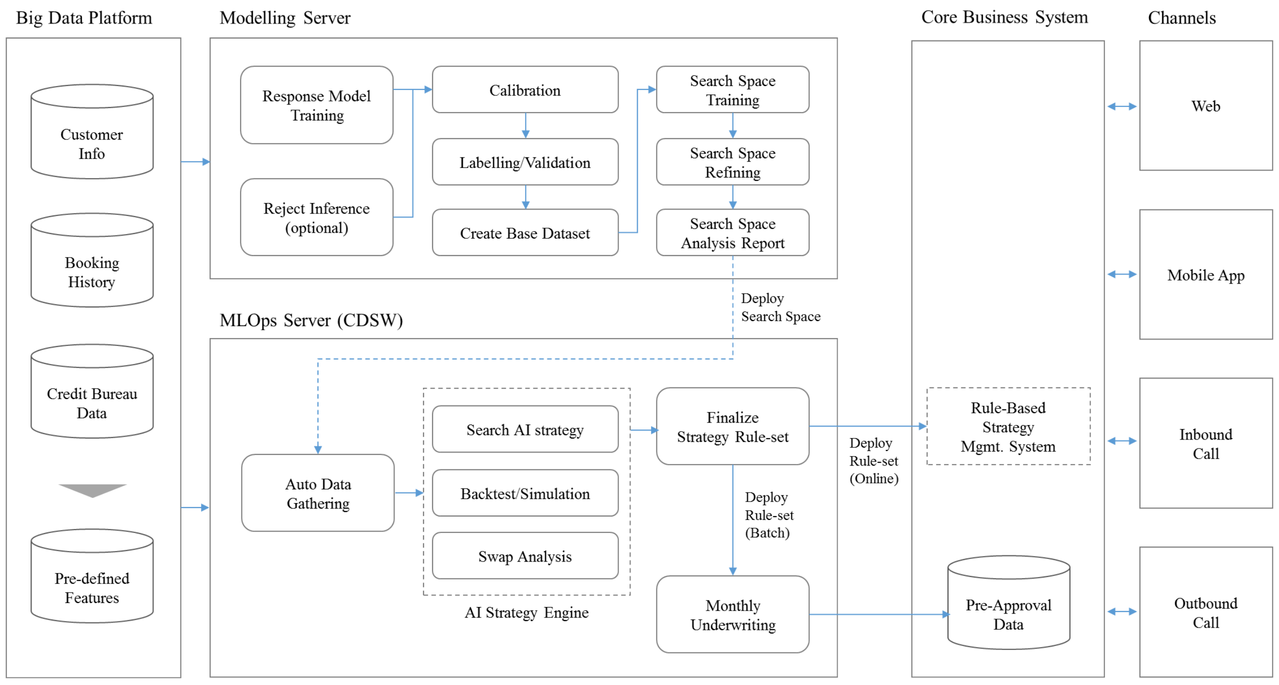

Figure 3: A schematic overview of the HCS loan approval rule engine.

Kruppa et al. 2013), to nsemble models like extreme gradient boosting (XGBoost) (Chen and Guestrin 2016), and even deep neural networks (Babaev et al. 2019).

When it comes to constructing decision rules, Dumitrescu et al. (2022) introduced the penalized logistic tree regression (PLTR). This method generates covariates for the logistic regression model by extracting logical formulas from leaf nodes in decision trees. While this facilitates the creation of a sophisticated scorecard model from data, its primary utility lies in predicting credit scores. A nuanced challenge is its limited application in overall underwriting, especially when balancing the trade-off between bad rate and volume. Chen et al. (Chen et al. 2018) put forward a 2-layer neural network reminiscent of traditional subscale models. Yet again, its strength is primarily in forecasting credit scores. Of significant relevance to our work is the falling rule list (Wang and Rudin 2015; Chen and Rudin 2018). This model shares similarities with our underwriting strategy, presenting decision rules ordered by the estimated probability of a specific outcome. However, while the falling rule list emphasizes finding optimal solutions via a Monte-Carlo search, it struggles with scalability on datasets of our magnitude. Moreover, its learning objective requires expansion to cater to multiple objectives, i.e. bad rate and volume. Despite these challenges, the falling rule list remains a promising framework. In future endeavors, we plan to consider extending and integrating this model for constructing loan approval rules.

While enhancing the predictive accuracy of credit scoring models using advanced machine learning and extensive data is feasible, model explainability remains crucial. In theory, techniques like LIME (Ribeiro, Singh, and Guestrin 2016) and SHAP (Lundberg and Lee 2017) can elucidate intricate machine learning models. However, in our view, these methods fall short in highly regulated industries like finance (Rudin 2019). Their explanations often failed to provide an accurate representation of the model's internal processes.

Conclusion

In this paper, we introduced the Automated Loan Approval Rule Engine, a system deployed at HCS. This practical framework streamlines the creation of loan approval rules for financial services. While advanced machine learning models have substantially improved credit scoring, they pose challenges in interpretability and accountability when used directly for underwriting. As a result, many have not been adopted, leaving risk analysts to manually design loan approval rules. Our system overcomes this limitation, automating the construction of optimized, yet comprehensible rules through machine learning (specifically, XGBoost) and principled optimization techniques by submodular optimization. Since its initial live test with a subset of personal loan customers in August 2021, our system has supplanted the manual rule-making process. By 2022, it managed all customer applications and has now been expanded to tasks such as debt collection allocation.

References

Agarwal, R.; Srikant, R.; et al. 1994. Fast algorithms for mining association rules. In *Proc. of the 20th VLDB Conference*, volume 487, 499.

Babaev, D.; Savchenko, M.; Tuzhilin, A.; and Umerenkov, D. 2019. Et-rnn: Applying deep learning to credit loan applications. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2183–2190.

Chen, C.; Lin, K.; Rudin, C.; Shaposhnik, Y.; Wang, S.; and Wang, T. 2018. An interpretable model with globally consistent explanations for credit risk. In *NIPS 2018 Workshop on Challenges and Opportunities for AI in Financial Services: the Impact of Fairness, Explainability, Accuracy, and Privacy*.

Chen, C.; and Rudin, C. 2018. An optimization approach to learning falling rule lists. In *International conference on artificial intelligence and statistics*, 604–612. PMLR.

Chen, T.; and Guestrin, C. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, 785–794. New York, NY, USA: Association for Computing Machinery. ISBN 9781450342322.

Cramer, J. S. 2002. The origins of logistic regression.

Dastile, X.; Celik, T.; and Potsane, M. 2020. Statistical and machine learning models in credit scoring: A systematic literature survey. *Applied Soft Computing*, 91.

Dumitrescu, E.; Hué, S.; Hurlin, C.; and Tokpavi, S. 2022. Machine learning for credit scoring: Improving logistic regression with non-linear decision-tree effects. *European Journal of Operational Research*, 297(3): 1178–1192.

Fujishige, S., ed. 2005. *Submodular functions and optimization*. Reading, Mass.: Elsevier.

Han, J.; and Pei, J. 2000. Mining frequent patterns by pattern-growth: methodology and implications. *ACM SIGKDD explorations newsletter*, 2(2): 14–20.

Han, J.; Pei, J.; and Yin, Y. 2000. Mining frequent patterns without candidate generation. *ACM sigmod record*, 29(2): 1–12.

Iyer, R.; and Bilmes, J. 2012. Algorithms for approximate minimization of the difference between submodular functions, with applications. In *Proc. of the 28th Ann. Conf. on Uncertainty in Artificial Intelligence (UAI'12)*, 407–417.

Iyer, R. K.; and Bilmes, J. A. 2013. Submodular optimization with submodular cover and submodular knapsack constraints. *Advances in neural information processing systems*, 26.

Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; and Liu, T.-Y. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30.

Krause, A.; and Guestrin, C. 2005. A note on the budgeted maximization of submodular functions. Technical report, Carnegie Mellon University. Center for Automated Learning and Discovery.

Kruppa, J.; Schwarz, A.; Arminger, G.; and Ziegler, A. 2013. Consumer credit risk: Individual probability estimates using machine learning. *Expert systems with applications*, 40(13): 5125–5131.

Lundberg, S. M.; and Lee, S.-I. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.

Narasimhan, M.; and Bilmes, J. A. 2005. Submodular-supermodular procedure with applications to discriminative structure learning. In *Proc. of the 21st Annual Conf. on Uncertainty in Artificial Intelligence (UAI 2005)*, 404–410.

Nemhauser, G. L.; Wolsey, L. A.; and Fisher, M. L. 1978. An analysis of approximations for maximizing submodular set functions. *Mathematical programming*, 14: 265–294.

Quinlan, J. R. 1986. Induction of decision trees. *Machine learning*, 1: 81–106.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 1135–1144.

Rudin, C. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5): 206–215.

Sviridenko, M. 2004. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters*, 32(1): 41–43.

Wagner, H. 2017. Default Definition under Basel. In *Intelligent Credit Scoring*, chapter 7, 119–130. John Wiley & Sons, Ltd. ISBN 9781119282396.

Wang, F.; and Rudin, C. 2015. Falling rule lists. In *Artificial intelligence and statistics*, 1013–1022. PMLR.