Effective Automated Feature Derivation via Reinforcement Learning for Microcredit Default Prediction

Mengnan.Song
360 Financial
Beijing, China
songmengnan-jk@360jinrong.net

Jiasong.Wang
360 Financial
Beijing, China
wangjiasong-jk@360jinrong.net

Tongtong.Zhang
360 Financial
Beijing, China
zhangtongtong-jk@360jinrong.net

Guoguang.Zhang
360 Financial
Beijing, China
zhangguoguang-jk@360jinrong.net

Ruijun.Zhang
360 Financial
Beijing, China
zhangruijun-jk@360jinrong.net

Suisui.Su
360 Financial
Beijing, China
susuisui@360jinrong.net

***Abstract*—Microcredit is a new financial instrument serving the segment of population that typically lack collateral and are highly likely to be rejected by traditional financial institutions, by lending very small loans. For platforms that participate in such consumer finance activities, the key challenge lies in risk management and the popular credit scoring method predicting whether a borrower would default or not takes an important role in this field. However, the fact is that we are often facing mass of raw data and the traditional credit scoring is heavily depending on feature engineering involving domain expert knowledge, intuition and trial and error, which is often time consuming.**

It is very challenging to derive effective features from raw data as the searching space can be very large with noninformative features. In this paper, we propose a new performance-driven framework automated generating discriminating features from raw data via reinforcement learning to help improve the default prediction of the downstream classifier which may be a logistic regression or boosting tree. Specially, we first define a formal paradigm for the automated feature derivation framework which unifies the feature structure, its interpretation and the calculation logic together. For the particularity of the financial industry, the interpretation of the feature is often of high interest. Then we reformulate the feature generation problem as reinforcement learning by constructing a transformation link and regarding it as a sequential decision process. In addition, we carry out an effective practice on default prediction in consumer finance. Finally, experimental results on the data of user behavior log from 360 Financial show the significant improvement of the proposed method over our years of domain expert knowledge and the Genetic Programming. Moreover, this FDRL framework can be easily adapted to other applications due to its versatility.

***Index Terms*—Credit default, Feature derivation, Feature interpretation, Reinforcement learning**

I. Introduction

Microcredit is the extension of very small loans to impoverished borrowers who typically lack collateral, steady employment, or a verifiable credit history and are highly likely to be rejected by traditional financial institutions, relying on the vision of "even the poorest of the poor can work to bring about their own development". The essence of consumer financial behavior is that consumers pay certain financial costs to change the disposable capital flow within a specified period to match their consumption demand. After many years of initial development, China's consumer finance market, especially the online-lending business, has entered a period of market eruption and penetrated into all walks of life in the society. Many such platforms have provided basic financial service for massive number of users, like 360 Finance.

Given the relatively poor user quality as the characteristic of Microcredit, proper risk management lays the foundation of this Inclusive Finance with two basic types of decisions that are whether to grant credit to a new applicant and how to adjust the credit restrictions or the marketing effort directed at a current customer for creditors [1]. So for online-lending companies, one of the key challenges is to identify default borrowers. Credit scoring is a statistical method which is introduced in the 1950s and now widely used for consumer lending to predict the probability that a loan applicant or existing borrower would default or become delinquent [2] [3]. In general, the credit scoring is regarded as a binary classification problem using logistic regression or boosting tree which is more popular currently with adequate and heterogeneous features as input. Besides, coupled with advances in both computation power and machine learning theory, many other popular alternatives are attracting the interest of researchers. Jae and Lee [4] applies SVM, i.e. support vector machine, to the bankruptcy prediction problem using grid-search to find out the optimal parameter values of kernel function. Reference [5] gives a evaluation of the predictive performance of different fitness function used by genetic algorithms in credit scoring and proposes a bitmask version superior in both accuracy and sensitivity. A combination of SVM and genetic algorithm as a hybrid GA-SVM strategy in [6] can simultaneously perform feature selection task and

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

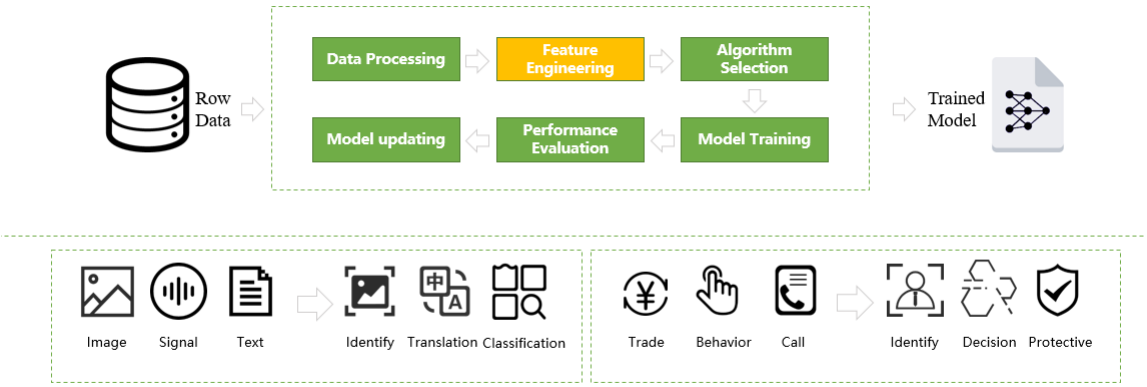


Fig. 1: A typical data science endeavor.

model parameters optimization. D.West [7] tests the neural network credit scoring model and concludes that both the mixture of experts and radial basis function neural network models should be considered for credit scoring applications. However, despite the breakthrough of AI-based approaches, simple models, like logistic regression, still remain to be popular because of its good interpretability and the feature engineering making a decisive contribution.

As shown in Fig. 1, the common credit modeling process can be described as follows:

a) Data Processing: The data scientist collects mass of data including user information which can be seen directly as model feature and log data that are used to create new features

b) Feature Engineering: We divide the concept of feature engineering into two categories. One is to create new feature by a algebraic equation with existing features. A common method is symbolic regression. The other is deriving new features from a source raw data through some complex strategy under a given budget. Unlike most existing researches mainly tackling the first type of problem, here in this paper, we are focusing on the second category which makes the most contribution of the default prediction and takes more than 80% of the time cost in financial credit modeling

c) Model Training: With a good job of feature engineering, it is relatively easy to get a reliable model both in accuracy and generalization. Logistic regression used to be the most popular method with a few number of features as input depending on the carefully designing and selecting features by data scientist. More recently, boosting tree gradually takes its place due to the fact that credit data in Chinese market now are usually high dimensional and extremely heterogeneous.

From Fig. 1, we can also see that most researches about AutoML are focusing on image process, speech recognition and language translation. While, 360 Financial is dedicated to the practice in financial sector with its unique characteristic. In this paper, we develop this FDRL framework to derive discriminative features from raw data automatically aiming to make these endeavors more efficient, enjoyable and successful. Reinforcement learning can interact with environment, learn from action rewards, balance exploitation and exploration and search for long term optimal decisions [8]. These traits provide great potential to automated feature construction exploration. To the best of our knowledge, this is the first work that learns a performance-driven strategy for deriving effective new features form a source of raw data like user behavior log. Our contributions through this paper are as follows:

- A formal paradigm for the automated feature derivation framework is defined to unify the feature structure, its interpretation and the calculation logic together
- The feature generation problem is reformulated as reinforcement learning by constructing a transformation link and regarding it as a sequential decision process
- We carry out an effective practice on default prediction in consumer finance and achieve a efficient and automated system exceed our expert experience

The remainder of this paper is organized as follows. Literatures of related works are described in section 2. Section 3 introduces the FDRL framework including the paradigm definition and searching strategy. In section 4, groups of experiments are implemented to evaluate the efficiency of the proposed method. Conclusions and future works are summarized in section 5.

II. Related Works

In recent years, AutoML has emerged as a new sub-area in machine learning and been successfully applied in many fields such as computer vision, speech analysis and natural language processing. Taking the application into consideration, AutoML can be grouped into three directions which are automated model selection, neural architecture search and automated feature engineering with respect to Auto-sklearn, Google Cloud and Feature Labs as typical applications.

It is a common practice to try different classification methods on different scenarios as different classifiers are applicable to different learning problems [9] and get best prediction as the final result. However it is not a easy work to set up classifiers and fine turn their hyper-parameters. Auto-sklearn [10] [11] is

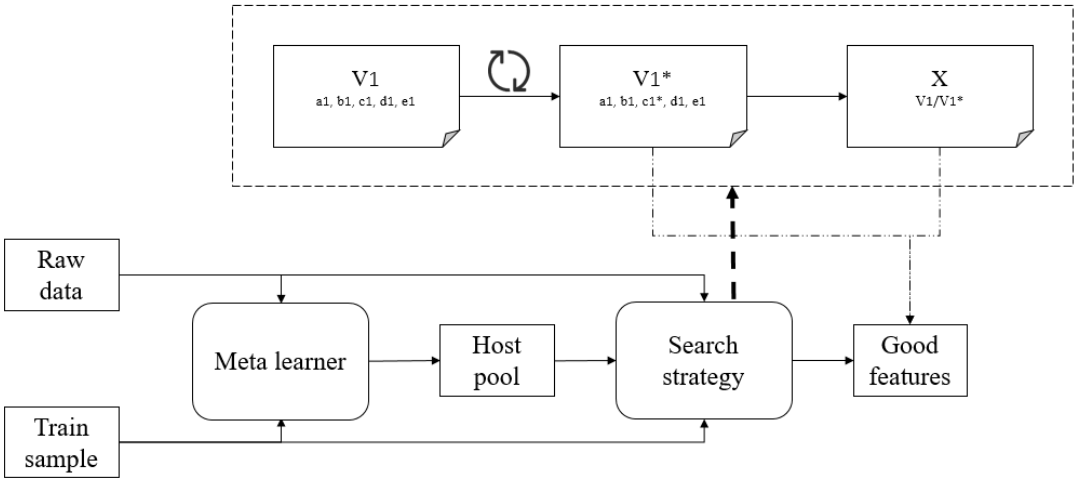


Fig. 2: The FDRL framework.

a kindly tool to help search for proper models and optimize their corresponding hyper-parameters.

Deep learning has enabled remarkable progress over the last ten years on a variety of tasks, such as image, signal and text [12] [13] [14]. One crucial aspect for this progress is novel neural architectures through a path from VGGNet, GoogleNet, ResNet to DenseNet. Hence, for the task in hand, automated design of neural architectures is of great importance to good learning performance and has attracted great attention in researchers [15]. Reference [16] provides an overview of existing works in this field and categorize them according to three dimensions: search space, search strategy and performance estimation strategy. Besides, NAS has been used in Googles Cloud AutoML, which frees customers from the difficult and time-consuming architecture design process.

While recent developments in deep learning and automated processing of images, signal and text have enabled significant automation in feature engineering for those data types, feature engineering for human behavioral data remains iterative, human intuition driven and challenging and hence, time consuming [17]. We divide the feature engineering into two categories.

One is to create new features like $3a + 2b * log(c)$ by a algebraic equation with the existing features a, b and c. We call this category feature composition aiming to transform the original input space to a new space with better separability. GPMFC [18], a multiple feature construction system using genetic programming, evolves functions by a entropy-based fitness function that maximizes the purity of class intervals applying to the original input features to get multiple new features. Domain experts are more likely to deploy autonomously learned controllers if they are understandable and convenient to evaluate. The algebraic equations are supposed to meet the requirements as long as they are restricted to an adequate complexity. GPRL [19] is a genetic programming for reinforcement learning approach which autonomously learns policy equations from pre-existing default state-action trajectory samples. Experiments on three reinforcement learning benchmarks demonstrate the superiority of GPRL compared to the symbolic regression. Reinforcement learning is gradually becoming a popular solution path. Ref. [20] proposes a novel framework based also on reinforcement learning. It involves training an agent on feature engineering examples to learn an effective strategy of exploring available choices under a given budget. The learning and application of the exploration strategy is performed on a transformation graph, a directed acyclic graph representing relationships between different transformed versions of the data.

And the second category of feature engineering is feature derivation that creates new features directly from raw data. Taking a hypothetical user behavior data set for a online lending platform into consideration, our goal is to calculate features describing customer's behaviors based on all available data. One kind of feature that aggregates values related to a customer within a time period under some constrains plays a important role in credit risk management, for instance, "how often does this customer submit an application?" or "how long since this customer's last login?". We may also look at entities related to a customer, for instance, "how much does the draw amount vary for this customer?" or "how many customers come from the same place of residence with this customer?" We are expecting to generate these types of features from the raw behavior records. The Deep Feature Synthesis [17] follows relationships in the data to a base field, and then sequentially applies mathematical functions along that path to create the final features. By stacking calculations sequentially, it defines each new feature as having a certain depth which usually takes 1 or 2 for the interpretability and complexity of the generated features. The One Button Machine [21] adapts nearly the same idea from Feature Tools that joins the related tables together, identifies data types, applies corresponding pre-defined feature engineering techniques on the given types and follows a feature selection process. This kind of approach takes the strategy of expansion-reduction that

they firstly collect all features as exhaustive as possible without a supervisor guidance and then perform a feature selection which will suffer performance and scalability bottleneck due to the calculation and selection of massive non-informative features.

III. FDRL Framework

In this section, we first introduce the raw data of user behavior from 360 Financial online system. Then, we give a description of the formal paradigm for the proposed framework from the aspects of feature structure, interpretation and calculation logic by answering the questions of "what kind of feature to derive", "how to introduce guidance signal" and "how to calculate the feature value". Finally, the reinforcement learning based searching strategy is elaborated and the whole process will be an end to end automated feature engineering without human effort. The whole framework is illustrated in Fig. 2 with 2 core parts that are meta learner and search strategy. The meta learner is a traditionally trained model taking a preprocessing role to accelerate the calculation and reduce the search space for the downstream search strategy.

A. Dataset

Our data set is provided by 360 Financial, one of the leading online consumer finance marketplace in China, who provides installment loans as credit product with the period of 3, 6 or 12 months.

User behavior data record the interactions between user and platform together with its related properties shown in the following figure. Due to data privacy, Fig. 3 demonstrates only a part of the data filled with some fictional values. Event id is a global and unique index of these records but not used for retrieval. With millions of active users, the amount of the behavior data is huge and it is not smart to locate one specific line. The column of datetime keeps the time stamp when this event happens, i.e. the time when user takes this action. Event types are stored in the event name column and gender column stands for the user gender. Besides these columns, we have constructed many other meta-fields to provide a detailed description of different kinds of events and users. The amount of the raw data is too large to use directly and we usually sample a part of it base on the supervisor from rows and expert domain knowledge from columns.

event_id	user_id	datetime	event_name	gender
1	001	2019/1/1 0:00	login	female
2	001	2019/1/2 0:00	apply	female
3	002	2019/1/3 0:00	approve	male
4	003	2019/1/4 0:00	apply	female
5	003	2019/1/5 0:00	reject	female
6	004	2019/1/6 0:00	logout	female

Fig. 3: Part of the user behavior data.

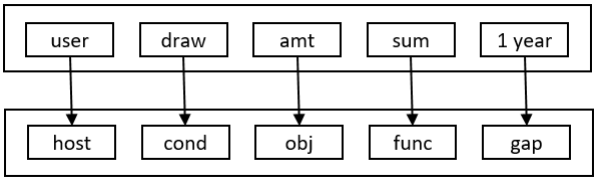


Fig. 4: Decomposition of a velocity feature

id	datetime	label	user_id	gender
001	2019/1/1 0:00	1	001	female
002	2019/1/2 0:00	1	002	female
003	2019/1/3 0:00	0	003	male
004	2019/1/4 0:00	0	004	female

Fig. 5: A sample of train set

B. Paradigm Definition

As described in Feature Tools [17], it accomplishes feature engineering from a relational database with three main steps that are data collection, data transformation and feature selection. Its main work is to effectively organize the relational data tables and then calculate all potential features. While in our application, we collect and preprocess the user behavior records together with all available information that we are concerning about or do not know whether they are useful in one table by pre-defined relations directly on the online system. Unlike the Feature Tools, the relationship of tables is not our focus as we are trying to gather as much data as possible in advance and then perform a novel search strategy to generate discriminative features.

We expect to elaborate the paradigm through answering the above 3 questions.

a) What kind of feature to derive: In the field of credit risk management, a kind of feature that can be described as aggregating information related to an user within a time period is very effective in terms of risk. We call this kind of feature velocity feature due to its time partition characteristic. For instance, "the draw amount of user within a year" captures the level of loan demand of users. Here we name the item that the specific feature is analysing as host which could be user as we have mentioned above or user's incidental attributes. A typical example is "the number of applications from the region where the current user from" representing the status of hometown of user taking into consideration that people from the same place may have something in common. Another kind feature we are interested in is composed of two other velocity features by dividing one by another, like "the draw amount within a week divided by the amount with in a month of user" revealing the sudden changes in user habits. We name the compound feature velocity$^+$ feature for being composed of two other velocity features. These two kinds of features are the most commonly used features in practice for financial risk management and what we are expecting to derive from raw data directly. The

velocity feature can be diverse with sufficient raw materials. To give a formal paradigm, we decompose a velocity feature into 5 components which are host, object, function, time gap and condition as shown in Fig. 4 with a real example. Host is what we are trying to portray like user or some incidental attributes of user. Object is the evidence used to describe the host. Function and gap involve an aggregator and a time period set respectively with some pre-defined options. Condition can be very flexible with usually categorical columns, such as "event name equals to draw", "application region equals to Beijing", or "age is greater than 40". This paradigm unifies the feature structure, interpretability and calculation logic together. According to the above designed paradigm, one can create a new feature by filling the 5 components with corresponding enumerate options, which can be seen as a typical searching problem.

b) How to introduce the guidance signal: Since each component of the feature may have a tremendous amount of candidates, it is impossible to traverse them all under a reasonable resource limitation. For example, if each component has 10 potential enumerate options, then the total number of features will be 10^5. So we consider that the searching strategy can be adaptively adjusted with a feedback given a evaluation mechanism. Then a train set with samples to be analyzed and their labels is introduced based on which we are able to evaluate the calculated features through information value, in order to find better features. The information value is a popular index to evaluate the performance of variables for binary classification. In this way, it is avoided to train a model and makes the searching strategy be in a model independent manner. As to the train set, shown in Fig. 5, it must has at least 4 columns. Id is the sample indicator. For example, we are trying to model user and then use user id as the sample indicator. The datetime column stands for the decision time as a time line basis that any information beyond this line must not be used. In consumer finance, a most common problem is to predict whether a borrower would default or not which can be seen as a binary classification task. So we take 1 as normal borrower and 0 representing cheating user. The rest columns are hosts that are pre-screened by a meta-learner from the columns of raw data to improve the efficiency of feature calculation process, such as user and user gender. It is obvious that there must be at least 1 host.

c) How to calculate the feature value: Ref. [17] also defines a depth to quantify the complexity of a generated feature concerning the computation and interpretability. Although, this kind of velocity feature can go deeper by repeating the aggregation operation, 1 depth is of totally enough effectiveness in real application according to our expert experience. Further more, due to the difficulty of our online system backtracking historical data at any time range in real time, we force the right side of the time period as the decision time, which means that for the time gap part, we only need to consider the length of the time period. Under the above restrictions, let us see how can we calculate the feature value efficiently. First, because there can be a plenty of columns in the raw data set, we use a traditional tree based model to train a meta-learner with meta features like the rate of null value, the number of unique value, the relevance to labels from supervisor train samples and variance to assign the selectable columns to the component of host and initialize a potential host candidates pool. For example, the expected candidates of host are user, sex, region and even the combination of user and region for people moving in different areas. This do not mean other components can not use these options in host candidates pool. Then, by joining the train samples and raw data on the host columns respectively, the raw data can be sampled as multiple subsets and valid aggregators can be applied to these subsets, which will greatly reduce the calculation for there being no need calculate features for all user but the user samples in train set on each subset. Meanwhile, the aggregations of different subsets can be paralleled.

C. Methodology

Through the above designed paradigm, the feature engineering job has been transformed into a searching problem. Then we reformulate it as reinforcement learning. In order to form a Markov chain, a transformation link is constructed as a sequential decision process, in which each node represents a feature obtained by applying some action on its parent node. Each of the transformation links is a candidate solution for the feature engineering problem. Start from a random feature, by going through the transformation link, we are expecting to get a feature with high information value and the policy gradient method is adopted to achieve this goal.

The components of reinforcement learning are defined as follows.

- **State**: Given a random velocity feature which has been decomposed into 5 components, it can be represented by the corresponding value of each component. We use a tuple like (a, b, c, d, e) to represent a velocity feature. While for the velocity$^+$ feature consisting of 2 other velocity features with one divided by another, we use $(a1, b1, c1, d1, e1)$ standing for molecule and $(a2, b2, c2, d2, e2)$ standing for denominator. Then the whole feature can be described as $(a1, b1, c1, d1, e1, a2, b2, c2, d2, e2)$
- **Action**: For velocity feature, at each step, the agent chooses one component of a parent node and change its value to another option to produce a new feature as the child node. For example if the agent chooses to change value c to c^* with parent feature state (a, b, c, d, e), the child feature will be (a, b, c^*, d, e). While, as to the velocity$^+$ feature, the action will be restricted to apply only on the denominator part. A similar example is $(a1, b1, c1, d1, e1, a2, b2, c2, d2, e2)$ being transformed into $(a1, b1, c1, d1, e1, a2, b2, c2, d2^*, e2)$ by changing value $d2$ to $d2^*$
- **Reward**: After applying an arbitrary action on the parent node, we know exactly what the child node will be and receive a difference between the two feature in the information value $\Delta iv = (iv_{child} - iv_{parent})$ as reward

Under the above definition, the agent interacts with the environment based on the current state for the purpose of achieving more rewards. Without any constrains on the agent, the number of possible actions can be unbounded which is difficult for reinforcement learning to deal with. In our proposed method, we restrict that for a parent node, only one component can be changed by one action. A possible valid state transformation is like $(a1, b1, c1, d1, e1) \rightarrow (a1, b1, c1, d1^*, e1)$ with $d1$ replaced by $d1^*$. This restriction has several benefits as it forces the agent exploring the whole space with a small step size which is helpful for convergence, restricts the action space to an appropriate size and relatively maintains the interpretability of the child node.

The policy of choosing an action given a current state can be parameterized as:

$$\pi(a|s, \theta) = P_r(A_t = a|S_t = s, \theta_t = \theta) \tag{1}$$

where t is the current step, s is the state of parent node, the model parameter is θ and P is the probability of taking a as the current action. For each step, we do not take the action with the highest probability directly, but sample according to its corresponding probability. By repeating this decision making process, we can get a trajectory like $\tau = (s_1, a_1, ...s_T, a_T)$ consisting of a series of nodes and their corresponding actions. The probability of τ based on the policy π can be represented as:

$$P(\tau|\pi) = \prod_{t=0}^{T} P(s_{t+1}|s_t, a_t)\pi(a_t|s_t) \tag{2}$$

where s_0 is randomly initialized.

As we use $r(t) = \Delta iv$, i.e. $(iv_{child} - iv_{parent})$ as the instant reward from environment after taking an action. It is obvious that the reward is positively correlated with feature quality. If child node is better than its parent, the environment will return a positive feedback and vice versa. So within this strategy and along the exploration, child nodes are expected always better than parent nodes leading to the "best" node finally. The total reward of a complete trajectory can be defined as:

$$R(\tau) = \sum_{t=1}^{T} r(t) = \sum_{t=1}^{T} (iv_{node_t} - iv_{node_{t-1}}) \tag{3}$$

where a stop criteria could be set, for example, the longest steps. With the observed trajectories τ, our goal is to maximize the expectation of long term reward:

$$maximize \quad J(\pi_\theta) = E_{\tau \sim \pi_\theta}[R(\tau)] \tag{4}$$

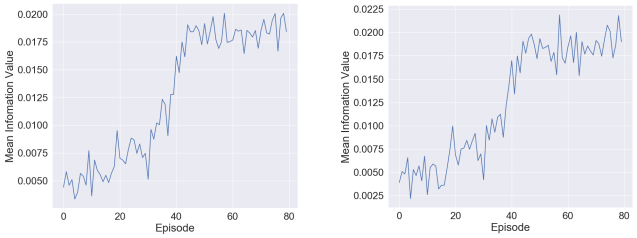
Based on the gradient algorithm, the parameter update can be described as:

$$\Delta_\theta J(\pi_\theta) = E[\Delta_\theta log \pi_\theta(s, a)v] \tag{5}$$

As the training process goes on, the agent learns to choose proper action to transform a ordinary feature to a good feature through a plenty of trying. We summarize the learning method in Algorithm 1. After the training process, a set of features is randomly initialized and set as the start of the transformation link. By exploring the transformation link, "best" features are generated at the final state.

Algorithm 1

Initialise θ $arbitrarily$
for each episode $\{s_1, a_1, r_1, ..., s_{T-1}, a_{T-1}, r_T\} \sim \pi_\theta$ **do**
  for $t = 1$ TO $T - 1$ **do**
    $\theta := \theta + \alpha \Delta_\theta log \pi_\theta(s_t, a_t)v_t$
  end for
end for
return θ



(a) Velocity (b) Velocity$^+$

Fig. 6: Learning effectiveness

IV. Experiments

The proposed method is evaluated with a real default prediction task and compared with our expert experience and the traditional Genetic Programming.

A. Train data

We sample 100000 users from 360 Financial online lending system with the registration time distributed in 3 months. All of these users have one or multiple loan records which depends on the number of successful loans. Each record can be further composed of loan time, loan amount and repayment time. The loan history is used to label default users. More specifically, we define a user who has 30 days overdue repayment as default borrower, while leave others as normal users.

B. Learning effectiveness

Fig. 6 shows the learning effectiveness of the proposed method for velocity with the left figure and velocity$^+$ features with the right one. We use the information value of the last state of each transformation link to evaluate the learning effectiveness. At the beginning stage of the training process, the feature generate process can be seen as random with the mean information value around 0.005 for both kinds of features. The mean information value of the final states gradually increase and converge as the training goes on. For the velocity feature, the final average information value rises up to 0.018 and the information value for velocity$^+$ is near 0.02. It is reasonable that the prediction ability of velocity$^+$ feature is a little higher than velocity feature from the aspect of both interpretation and structure. For both kinds of features, our proposed method brings a improvement by nearly 4 times compared to the random policy.

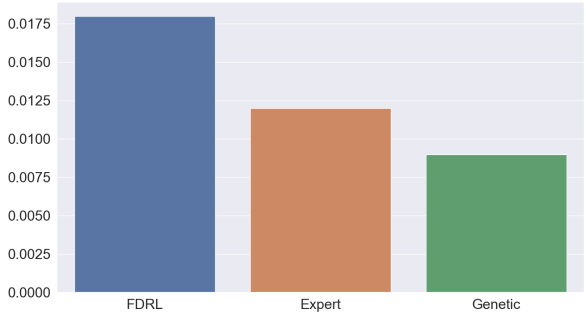


Fig. 7: Mean information value

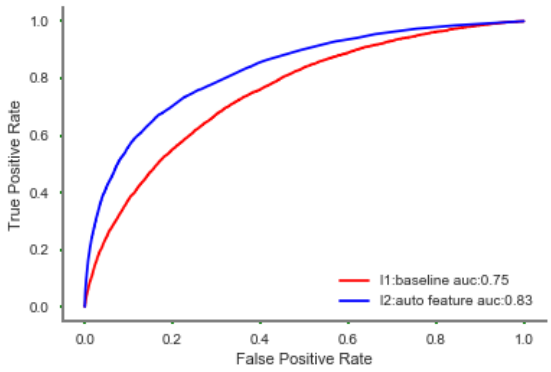


Fig. 8: Model performance

C. Performance comparison

Over the years, the credit risk management depended heavily on domain experts and our risk managers accumulated a plenty of business experience through years of data analysis. So we adopt the top artificially designed features as one baseline. As shown in Fig. 8, the default prediction model with auto features gets 8% gain than the model with artificially designed features in terms of AUC score which is widely used to evaluate model performance. Besides, as we have transformed the feature derivation to a search problem, we also take the Generic Algorithm which is a widely used global search algorithm as another baseline. Fig. 7 shows the mean information value of velocity features for different methods. We can see that the mean information value of artificially designed features is 0.011 while our proposed method can reach 0.018. Another advantage is that by adjusting the length of the transformation link to control its searching ability, the final states of transformation links will be different with different initial states. So the proposed method can yield relatively rich features with good prediction ability. Through further case by case analysis, we also find that these top artificially designed features are basically covered by the features derived by our proposed method. The mean information value of genetic algorithm is 0.09 which is a typical local optimum.

V. Conclusion and Future Work

In this paper, we propose a new performance-driven framework to automated generate discriminating features from raw data via reinforcement learning to help improve the default prediction of the downstream classifier. Specially, we first define a formal paradigm for the automated feature derivation framework which unifies the feature structure, its interpretation and the calculation logic together. Then we reformulate the feature generation problem as reinforcement learning by constructing a transformation link and regarding it as a sequential decision process. In addition, an effective practice is carried out on default prediction in consumer finance. Experiments demonstrate that the proposed method can not only cover and improve the human effort but also avoid the local optimum which traditional genetic programming suffers.

In order to constrain the action space to an appropriate size, we restrict that for a parent node, only one component can be changed by one action. But the possible number of actions which is equal to the total number of enumerate options of all component can be still large. In the future, we will try to set each component of a feature an agent to arrange its corresponding options and modify the learning policy a multi agent reinforcement learning.

References

[1] L. C. Thomas, D. B. Edelman, and J. N. Crook, *Credit scoring and its applications*. SIAM, 2002.

[2] L. J. Mester *et al.*, "Whats the point of credit scoring?" *Business review*, vol. 3, no. Sep/Oct, pp. 3–16, 1997.

[3] H. C. Koh, W. C. Tan, and G. C. Peng, "Credit scoring using data mining techniques," *Singapore Management Review*, vol. 26, no. 2, p. 25, 2004.

[4] J. H. Min and Y.-C. Lee, "Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters," *Expert systems with applications*, vol. 28, no. 4, pp. 603–614, 2005.

[5] V. Kozeny, "Genetic algorithms for credit scoring: Alternative fitness function performance comparison," *Expert Systems with Applications*, vol. 42, no. 6, pp. 2998–3004, 2015.

[6] C.-L. Huang, M.-C. Chen, and C.-J. Wang, "Credit scoring with a data mining approach based on support vector machines," *Expert systems with applications*, vol. 33, no. 4, pp. 847–856, 2007.

[7] D. West, "Neural network credit scoring models," *Computers & Operations Research*, vol. 27, no. 11-12, pp. 1131–1152, 2000.

[8] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," 2011.

[9] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics New York, 2001, vol. 1, no. 10.

[10] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," in *Advances in neural information processing systems*, 2015, pp. 2962–2970.

[11] L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown, "Auto-weka 2.0: Automatic model selection and hyperparameter optimization in weka," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 826–830, 2017.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[13] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*, 2016, pp. 173–182.

[14] J. D. M.-W. C. Kenton and L. K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *CoRR abs/1810.04805*, 2018.

[15] Y. Quanming, W. Mengshuo, J. E. Hugo, G. Isabelle, H. Yi-Qi, L. Yu-Feng, T. Wei-Wei, Y. Qiang, and Y. Yang, "Taking human out of learning applications: A survey on automated machine learning," *arXiv preprint arXiv:1810.13306*, 2018.

[16] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *arXiv preprint arXiv:1808.05377*, 2018.

[17] J. M. Kanter and K. Veeramachaneni, "Deep feature synthesis: Towards automating data science endeavors," in *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 2015, pp. 1–10.

[18] K. Neshatian, M. Zhang, and P. Andreae, "A filter approach to multiple feature construction for symbolic learning classifiers using genetic programming," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 5, pp. 645–661, 2012.

[19] D. Hein, S. Udluft, and T. A. Runkler, "Interpretable policies for reinforcement learning by genetic programming," *Engineering Applications of Artificial Intelligence*, vol. 76, pp. 158–169, 2018.

[20] U. Khurana, H. Samulowitz, and D. Turaga, "Feature engineering for predictive modeling using reinforcement learning," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[21] H. T. Lam, J.-M. Thiebaut, M. Sinn, B. Chen, T. Mai, and O. Alkan, "One button machine for automating feature engineering in relational databases," *arXiv preprint arXiv:1706.00327*, 2017.