Abstract

In the field of credit scoring, imbalanced data sets frequently occur as the number of defaulting loans in a portfolio is usually much lower than the number of observations that do not default. Under the scheme of misclassification costs, the state-of-the-art ensemble methods used for credit risk assessment, i.e. Random Forest and XGBoost etc., tend to misclassify plenty of 'good' instances as 'bad' when these instances are close to the decision boundary. This type of misclassification has an impact on the Internet banking industry because the negative impact of mouth of word from the disappointed rejected users may be substantially enlarged by the Internet.

Stacking is a widely-used algorithm to address this problem. However, empirical studies found that the performance of direct stacking is insignificant and unstable in the context of severe data imbalance. In early 2017 the 'Partially stacking blend based user credit assessment model' was proposed by a team from Sun Yat-sen University and it has been claimed that their model could mitigate the negative impacts of the weak base estimators in stacking. However, their research gives little details, and there is no further published work to justify their study yet.

To close the research gaps, this paper firstly provides instructions for practitioners to carry out this model. In addition, this research describes an empirical study of the predicted behaviour of this model based on four real-world and up-to-date Internet loan data sets. The experimental evaluation shows that the model has only slight potential to improve the prediction accuracy. Lastly this study proposes a practical approach to tuning the critical parameter of this model.

Keywords: stacking, XGBoost, credit risk assessment, imbalanced data

1 Introduction

Internet finance, a newborn industry, or a new direction of traditional finance has received considerable attention from all over the world (Zhang 2015). Third Party Payment, Crowdfunding, P2P Lending, Microloan and Consumer Finance etc. compose the main product matrix of Internet finance (Liu 2015). In contrast to conventional banking sector, it is relatively challenging for the Internet financial companies to strike a balance in growing rapidly and sustainably at the same time: Internet financial Start-ups are under considerable pressure to occupy the market in a short time; nonetheless, an slight increase in default rate may cause severe loss or even bankruptcy (Wang et al. 2015). In this context a high criteria for credit scoring is favored, nonetheless this is at the cost of slower growth.

This dilemma challenges the current state-of-the-art credit risk assessment algorithms. Empirical investigations (Malekipirbazari & Aksakalli 2015; Zheng zb et al. 2017) reveal that, in the context of imbalanced data, decision-tree based ensemble methods such as Random Forest (RF) and XGBoost (XGB) are superior in identifying best of the best and worst of worst borrowers. However, they exhibit a decrease in relative performance for tricky instances in between. In other words, such classifiers tend to misclassify a large body of 'good' instances as 'bad'. In spite of a high overall accuracy, companies may suffer from the negative impact of the electronic word of mouth (Barrutia & Echebarria 2005) from the disappointed customers, adverse selection of customers (Dell'Ariccia et al. 1999), growth stagnation or even bankruptcy (Wang et al. 2015).

In this context there is a strong demand for a mechanism that is able to improve the predictive accuracy of the tricky instances which are difficult for decision trees based ensemble methods to deal with, the marginal effect of such progress would be great to the Internet financial agencies. One direction to explore is heterogeneous ensemble, which exploits the diversity of skillful predictions from different models (Duan et al. 2007) by compensating each other so as to achieve an improved overall performance. Indeed, the success of heterogeneous ensemble has been demonstrated in many studies. However, given that a multi-model contains information from all participating models including the less skilful ones, the question remains as to why, and under what conditions, a multi-model can outperform the best participating single model (Weigel et al. 2008). In practice, it is argued by empirical works (Zheng zb et al. 2017) that a direct multi-model

ensemble may perform worse than the best participating individual classifier. In this sense, it makes sense to provide more transparency or human interventions in heterogeneous ensemble.

In 2017 a successful trial in this field is achieved by the team of Zheng et al. from Sun Yat-sen University: their study proposed a well-direct stacking model that could correct the errors made by XGB. Thanks to this new model, they claimed to improve their online AUC in a machine learning challenge for credit risk modeling by 0.003[1].

To be more specific, their model consists of the following steps: firstly, a number of base models are fitted. Secondly, the tricky samples that are supposed to be misclassified by XGB are forwarded to be fitted by another model at Level Two; hence, these samples obtain more accurate evaluations. Eventually, the predictions of samples fitted by both XGB and the different model are blended as the final prediction. In this way this novel model combines diverse models that take good care of different samples, consequently the overall performance is supposed to be superior to the ordinary stacking.

In spite of the excellent idea, until now the model has not yet received much attention: no evidence in this research has been formally discussed, nor have any further empirical studies been released. In this context, the current paper is proposed to close the research gaps. Put differently, the objective of this paper is to: firstly, elaborate the original model by providing more details for Zheng et al.'s article so that practitioners could put it into practice. Secondly, to evaluate the performance of this model by an empirical investigation into four real-world up-to-date and large-scale Internet loan data sets. Finally, to improve the model usability and performance by proposing a new parameter tuning approach.

The remainder of this paper is organized as follows. Firstly, some background knowledge on ensemble methods and their applications in credit scoring of Internet banking are given. Secondly, the instructions to implement this model are presented. Thirdly an empirical study provides evidence on the performance of this model. Finally, some conclusions are drawn.

[1] http://bbs.pkbigdata.com/static/417_detail.html

4 Experimental Design

This chapter firstly elaborates the data sets used for the experiment. Then it discusses the employed Individual Classifiers and corresponding libraries. Lastly data preparation is introduced.

4.1 Data Sets Characteristics

The characteristics of the data sets used in evaluating the performance of the improved model are given below in Table 4.1. Four real-world credit data sets from Internet finance industry are collected. They cover mainstream loan products from micro-credit to medium-term unsecured credit. As the data sets were published in recent two years via noted machine learning challenge platforms, the quality of the data is officially entrusted.

Table 4.1: Data Sets Characteristics

Original Resource	Resource	Loan Type	Region	Year	Shape	Target Variable	P/N
Cash Bus	DataCastle	micro-credit	China	2015	15,000*1,138	default	11:89
PaiPaiDai	Kesci	online credit	China	2016	30,000*231	delinquency	7:93
Rong360	DataCastle	short-term credit	China	2017	55,000*29	delinquency	7:93
QianHai	Kesci	unsecured loan	China	2017	40,000*491	default	13:87

The first dataset comes from Cash Bus, a micro-credit company that offers instant online loaning service to individuals.[2] Among all the Internet finance products, the micro-credit is seen as one of the most risky niche markets due to the profile of the target group (Anil K. Khandelwal 2007). This view is particularly justified by the relatively high bad loan ratio (P/N) in the data set. Following Cash Bus the second data set is from one of the leading Chinese P2P Lending companies PaiPaiDai and is published on Kesci, the major Big Data competition platform in China. However, the loan type is not clearly stated in the official document.[3] The third data set comes from Rong360[4], a credit product search engine that recommends appropriate credit products in accordance with the customers' profiles and preferences. As an intermediary it collects information such as customers'

[2] https://www.cashbus.com
[3] http://www.ppdai.com
[4] https://www.rong360.com

on-site behaviors as well as historical transaction records of different financial products from credit card, bank debit to micro-credit etc. Such third-party information is contained in the data set and hence distinguishes Rong360 from other data sets. Finally, the fourth one is provided by a big player in the rapidly rising individual credit reference industry in China, QianHai Credit Reference[5]. It belongs to Ping-An Insurance Group of China, one of the most powerful financial agencies there that covers business such as insurance, banking, asset management and Internet finance etc[6]. As one of the three most competitive non-official individual credit reference agencies in China, QianHai provides especially valuable information because its parent company gives it access to the customers' other financing records such as vehicle insurance, health insurance, bank deposit, credit card etc. Consequently the direct personal financial records are seen as a unique feature of the data from QianHai.[7]

Table 4.2: Benchmarking Credit Data Sets

Original Resource	Resource	Loan Type	Region	Year	Shape	Target Variable	P/N
Lending Club	Kaggle	P2P Lending	U.S.A	2016	887,383*75	delinquency	15:85 *
Give me some credit	Kaggle	bank loan	NA	2011	120,269*10	delinquency	6:94*
German Credit (Statlog)	UC Irvine*	bank loan	GER	1994	1,000*20	"bad loan"	30:70

Resource: Archange Giscard DESTINE, Forecasting P2P Credit Risk based on Lending Club data

Resource: Adam Pah, Kaggle "Give me some credit" challenge overview

Resource: UCI Machine Learning Repository

In addition to the general introduction, the attributes of the data sets are elaborated in the following section. Especially, the content below is organized by comparing the four data sets with the conventional popular credit scoring data sets that have already been extensively studied (West 2000; Eggermont et al. 2004; Emekter et al. 2015). Specifically, as demonstrated in Table 4.2, German Credit Data from the UCI Repository of Machine Learning Databases (Blake 1998) and another two from Kaggle are used as benchmarks. In overall, the novel Internet loan data sets are featured with high dimensions, more diverse information resources, more missing values and severe data bias.

[5] https://qhzx.pingan.com/intro_aboutus.html
[6] http://www.pingan.cn/en/ir/summary.shtml
[7] http://www.zhengxinbao.com/3534.html

Firstly, it is not difficult to note that the data sets from these Chinese Internet companies normally have considerably more dimensions than the classical data sets. The primary reason is that China has not yet had a fully developed personal credit reporting system like FICO or Schufa (Huang et al. 2016). Hence, companies there have to collect much more information to increase the predictive power. Secondly, in terms of feature diversity, data sets from Chinese Internet financial agencies cover information such as personal information, online behaviors, online and offline transactions, financial products transactions, social network, locations etc. As shown below in Table 4.3, the overview of PaiPaiDai data provides an intuitive impression on how rich the dimensions are.

Table 4.3: PaiPaiDai Features

	Column Series Name	Column Series Description
Master Table	idx	unique key for each loan
	ListingInfo	loaning time
	UserInfo_*	user attributes
	WeblogInfo_*	online behavior information
	EducationInfo_*	user's education information
	ThirdParty_Info_PeriodN_*	third party information by different periods
	SocialNetwork_*	SNS information
	Target	1 = default, 0 = regular
Login Table	idx	unique key for each loan
	ListingInfo	loaning time
	LogInfo1	login index
	LogInfo2	login type
	LogInfo3	login time
Profile Table	idx	unique key for each loan
	ListingInfo	loaning time
	UserupdateInfo1	content of profile update
	UserupdateInfo2	time of profile update

Thirdly, the employed data sets are also suffering from high rate of absent records. Figure 4.1 plots the missing records per feature of QianHai data. Concerning the size of the data (40,000), except for User_Id and Target Variable, 330 of 500 features have around missing 15,000 of 40,000 records while the rest 168 columns have even higher missing value rates. This phenomenon is to some

extent similar to all novel data sets while this condition for conventional data sets is much more moderate. Lastly, as shown in the last column of Table 4.1 and Table 4.2, the data imbalance is also much more serious for novel data sets. These four unfavourable situations are calling for more advanced models, such as IPSBM proposed in this paper.

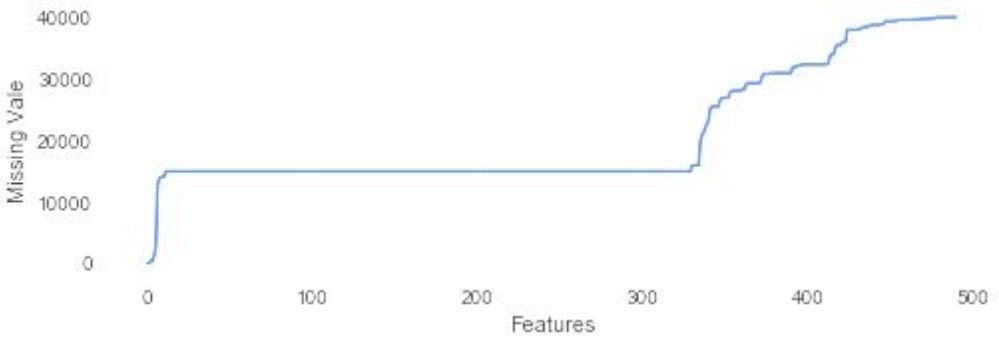

Figure 4.1: Frequency of Missing Values per Feature

4.2 Instantiation of Individual Classifier

In this chapter, the instantiation of individual classifiers is elaborated. All in all, it contains the following content: firstly, a table demonstrates the overview of employed classifiers. In the second part, the different versions of classifiers are disclosed. The last section explains how each classifier is instantiated.

The table below summarizes the employed BMs in this research. From LHS to RHS, it is structured by the type of classifier, name of classifier and the employed Python library. As can be seen in the first column, this research implements two types of BMs: LR and ensemble methods. Each type consists of several BMs: LR family includes Liblinear, SAG, Newton-cg and L-BFGS and the ensemble methods have five variations. Moreover, The last column demonstrates which specific package is used to instantiate these BMs.

Table 4.4: Individual Classifiers

Classifier Type	Classifier Name	Library	Details
Logistic Regression	Liblinear		(solver='liblinear')

	SAG Newton-cg lbfgs	sklearn.linear_model. LogisticRegression	(solver='sag') (solver='newton-cg') (solver='lbfgs')
Ensemble Methods	Bagging RF GBDT AdaBoost	sklearn.ensemble	BaggingClassifier RandomForestClassifier GradientBoostingClassifier AdaBoostClassifier
	XGB	xgboost	xgboost

Apparently, there are still plenty of alternative classifiers available for credit scoring and they are so different from each other that they have great potential to coordinate with other classifiers to make good model ensemble. For instance, in accordance with the research of Lessmann et al. (2015), there is room for estimators such as Extreme Learning Machine, SVM and Artificial Neural Network to function as BMs in this research. Nevertheless, this paper does not really put these or other classifiers that are outside the table into consideration. This is mainly because of their computational complexities and time complexities. Fortunately, this could be compensated in further studies.

Concerning the implementation, by and large, the entire experiment is conducted in Python 3.6. Except XGB that uses the original package "xgboost", all classifiers are implemented by the library sklearn, which is a widely used machine learning library for the Python programming language. As an integrated environment, it provides plenty of features to facilitate model training.

Before introducing the instantiations, it is important to notice that, each single BM in the table above actually has more than one employed son models. These son models all belong to the same BM but they differ from each other in terms of hyper parameter values or samples they are trained on. For instance, XGB has more than 6 derived models: xgb_1000 and xgb_2000 are different from rounds of iterations while xgb_2500 and xgb_2500_2 execute the same rounds of iterations but the latter is more extreme in penalizing the misclassified minority class. In order to keep the table more readable, the details of these son models are eliminated.

Concerning the implementation of LR with different regularizers and optimizers, sklearn library provides five types of solvers: "liblinear", "sag", "saga", "newton-cg" and "lbfgs" (Pedregosa et al. 2011). Firstly, Liblinear, a popular linear model that once won KDD Cup 2010 (Fan et al. 2008), is

a library for Large Linear Classification that supports both L1 and L2 penalty. Secondly, SAG stands for Stochastic Average Gradient, and this model minimizes finite sums with the Stochastic Average Gradient (Schmidt et al. 2017). Following SAG, Newton-cg is a modified Newton's method and uses a conjugate gradient algorithm to approximately invert the local Hessian (Shewchuk & Others 1994). Newton's method is based on fitting the function locally to a quadratic form. Lastly, L-BFGS, whose full name is Limited Memory BFGS, is an optimization algorithm in the family of quasi-Newton methods that approximates the Broyden Fletcher Goldfarb Shanno (BFGS) algorithm (Andrew & Gao 2007) using a limited amount of computer memory. It is a popular algorithm for parameter estimation in machine learning (Malouf 2002).

Regarding the instantiation of ensemble methods, Bagging has two editions in the research: both of them are instantiated by BaggingClassifier but one takes RandomForestClassifier as base_estimator while the other takes DecisionTreeClassifier. Similarly, RF, GBDT and AdaBoost are instantiated by class RandomForestClassifier, GradientBoostingClassifier and AdaBoostClassifier respectively.

4.3 Data Preparation

Data preparation of this research consists of four steps: data preprocessing, feature engineering, feature selection and data partitioning. By doing so, the original data is prepared for modeling. This paper follows a standard operation in processing all the data sets, the strategy is highlighted in the figure below. It is necessary to point out that, the idea and methodology of ranking are mainly employed in the entire data preparation procedure. Specifically, ranking in this context means sorting each observation (user) from a variety of perspectives so that the payback ability and payback willingness of each borrower is presented in a way that makes more sense.

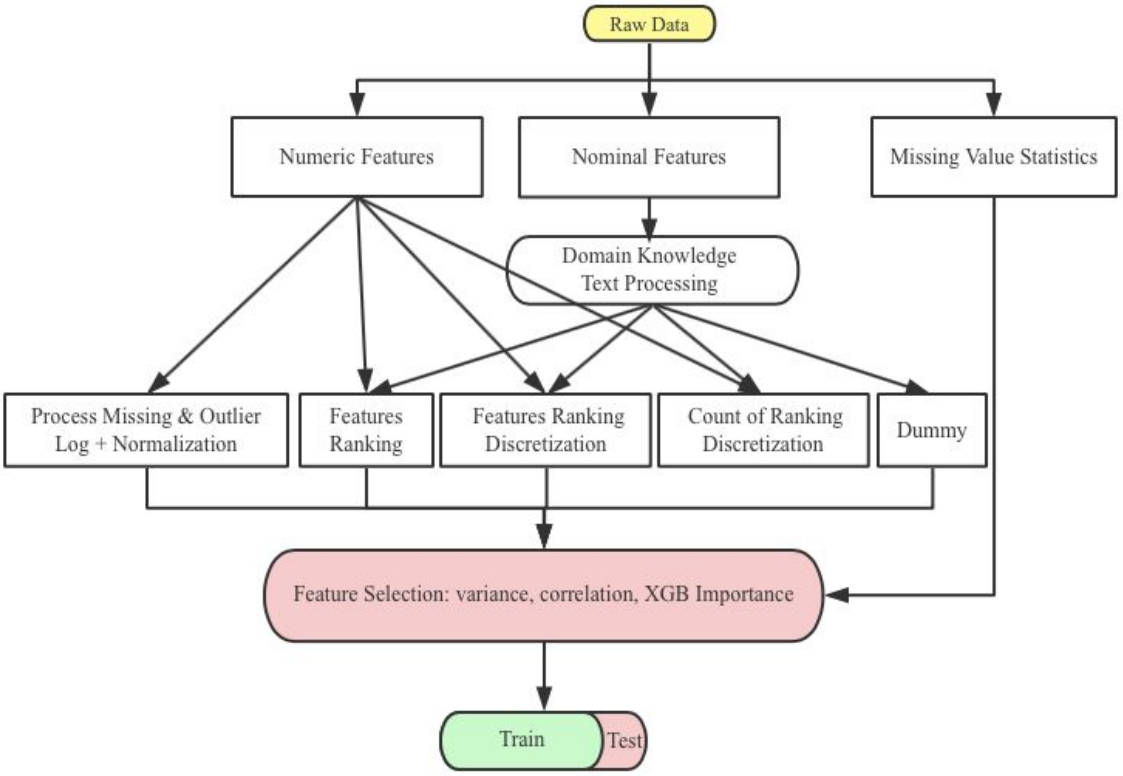

Figure 4.2: Data Preparation Strategy

4.3.1 Missing Value Statistics

Visualized by the rectangular on the RHS corner of the chart, missing value statistics is an indispensable part of Internet loan credit scoring. This is because missing value itself is critical information: in the case of information is intentionally hided by borrowers, missing value accentuates the group of people that are riskier, since some cheating borrowers or professional frauds want to cover their tracks. Meanwhile, the incomplete information may convey the development of employed credit risk controlling system which delivers useful information. In this way, recording and processing missing value contribute to modeling.

In practice, frequency of missing value per observation is firstly recorded. Presumably there are ten features in the data set and user No.1 has absent records (Na, blank, space, “Unknown” etc.) in five of the ten features, it will achieve value of 5 in the new feature “missing_count”. Furthermore, this value is sorted across all instances and discretized in an equal frequency manner. In this way three columns are generated: missing_count, missing_count_rank and missing_count_rank_dis. This is exactly the information group “Missing Value Statistics” in the chart above. Lastly, missing value

for numeric variables is replaced by median of the column, depending on if the customer paid back (0) or not (1). In the case of nominal variables, missing value is treated as an individual category.

4.3.2 Numeric Variables Processing

Numerical variable describes a borrower in a continuous manner, for instance, the borrower's age, annual income and frequency of profile updating etc. Abstracted as the rectangular on the LHS corner, the raw numeric variables are firstly sorted ascendingly and then every individual's position in this ranking column is recorded as its rank. It is notable that same values receive the same rank but the presence of duplicate values affects the ranks of subsequent values. Furthermore, equal frequency binning is undertaken to the ranking feature, then the ranking is discretized into 10 categories. Afterwards, the frequency of each observation's numerical feature discretization will also be recorded, and 10 columns called n1, n2 … and n10 are created to record the frequency of each observation's in each level.

On the whole, if there are n columns of numerical features in a data set, there would be additionally n ranking features, n discretization features and 10 count of discretization features generated. Thanks to the 2n+10 derived ranking features, the borrower's profile is more vividly presented, this is because the ranking features tell the loan agency much more information than plain numeric information. For instance, the ranking of borrower's annual income, ranking of month expense and ranking of car insurance cost reveal that whether the user is rich or poor while an absolute value could not 'hit the point'. Additionally, ranking makes the model more robust against outliers since the variance of feature is significantly reduced; therefore, the risk of overfitting is alleviated as well. Eventually, the raw numerical features are log transformed and normalized (Yang et al. 2016).

Log Transformation :

$$LogAll(x_k) = \log(x_k - \min + 1)$$

Normalization:

$$Normalize(x_k) = \frac{x_k - mean}{std}$$

4.3.3 Nominal Variables Processing

In comparison to numeric features that can describe a borrower as rich or poor, nominal features are in general not that informative in telling the borrower's ability and willingness in paying back.

However, one can learn if an individual belongs to majority or minority group from nominal features, for instance the ethnic groups, education level and employment situation etc. In order to fully extract this information, each categorical variable is sorted by the frequency of every single category of that categorical variable. In detail, for every nominal variable, the frequency of each category in the column is calculated and the variable is sorted by the frequency ascendingly. Afterwards, the equal-frequency binning and count of binning are recorded. By doing so, the popularity of each category is quantitatively presented so that the model explainability is improved.

After extracting the statistical information, the original nominal variables are converted into dummies. Furthermore, note that the nominal variables should be processed with domain knowledge first before conducting the statistical operations above. For example the PaiPaiDai data contains borrower's residence information, such as ‘Hunan Road, Nanjing City, Jiangsu Province’, for this reason the Chinese Word Segmentation is involved.

Last but not least, the combination of discretized ranking feature of numerical and categorical variables would also reveal important information. In detail, by taking the products of these two kinds of derived columns, the more specific information on the borrower’s identity is disclosed: if this user is a heavily-leveraged minority or this user belongs to ordinary people in lower economic class. Nonetheless, owing to the complexity, this paper could not take these arithmetical operations into practice.

4.3.4 Feature Selection and Data Partitioning

Some final details are added to wrap up data preparation. Firstly, variables have variance lower than 0.001 are removed. Afterwards, features are added into a collection one by one, a new variable could join in only if it has correlation under 0.8 with any column in the set already. Subsequently, an XGB training on the collection of features is carried out, and only Top 80% of important features are left. Lastly, the data is partitioned to train (80%) and hold-out (20%).

5 Empirical Results

5.1 Comparison of Classifiers

This section presents comparison of the classifiers on the four data sets as evaluated via hold-out test. The results are shown in Table 5.1. On the whole, it is to observe that IPSBM is inferior to normal stacking model though it makes slight progress on the single best individual classifier. To note, the best result is highlighted by bold face.

Table 5.1: Model Performance Overview

Data Set	Model	Model Details	AUC
Cash Bus	BBM	XGB_2500	0.6803
	IPSBM	L=14, no error, 13 'good' saved	0.6815
	Stacking	stacking of 36 BMs, combined by LR_Liblinear	**0.6884**
PaiPaiDai	BBM	XGB_2500_6	0.7437
	IPSBM	L=32, no error, 29 'good' saved	0.7455
	Stacking	stacking of 36 BMs, combined by Adboost_20	**0.7458**
Rong360	BBM	XGB_2000_6	0.7801
	IPSBM	L=170, no error, 27 'good' saved	0.7805
	Stacking	stacking of 36 BMs, combined by GBDT_20	**0.7816**
QianHai	BBM	XGB_2000	0.7536
	IPSBM	L=58, no error, 58 'good' saved	0.7544
	Stacking	stacking of 36 BMs, combined by XGB_2000	**0.7607**

More specifically, based on BBM, IPSBM makes only moderate progress in four hold-out data sets. For example, in PaiPaiDai, AUC grows from 0.7437 to 0.7455 only. By contrast, Stacking, a simple stacking model that combines the prediction outputs of each base model, achieves much more remarkable increase in AUC than IPSBM. In the case of Rong360, AUC witnesses an increase by 0.0015 via Stacking while via IPSBM it grows by 0.0004 only. This is because, considering the average size (7,000) of the four hold-out data sets, in average only 32 ‘good’ instances could be modified when they were misclassified as ‘bad’. Such a small change is too low for a dramatic rising in AUC. In conclusion, comparing the workload with the payoff, IPSBM seems not to be an economical choice.

5.2 Comparison of Tuning Approaches

In Chapter 3.3.3, two strategies of tuning the critical parameter L for IPSBM are discussed. In this section, the performance of the two approaches are quantitatively evaluated. Note that, QianHai data is taken as example in this section because other data sets follow basically the same pattern.

In terms of experimental setting up, it is slightly different from the setting of model comparison in Chapter 5.1: similarly, the training data contains 4/5 of all observations while the hold-out takes 1/5. To note, there is additional 20% of the training used as local validation to tune the optimal value of L for Strategy One as well as the optimal intervals and ranking difference for Strategy Two. Put differently, the hold-out is to test the optimal parameters tuned in local validation. Take QianHai data as example, the original data that has 40,000 observations, is partitioned into training that has 32,000 instances and hold-out which is made of 8,000 instances. In addition, 6,400 of the 32,000 observations in training are used as local validation.

Firstly, the hypothesis of consistent distribution of ranking difference is evaluated. As can be seen from Figure 5.1, the shape of ranking difference remains almost the same in hold-out data, which is in line with the hypothesis of the original paper.

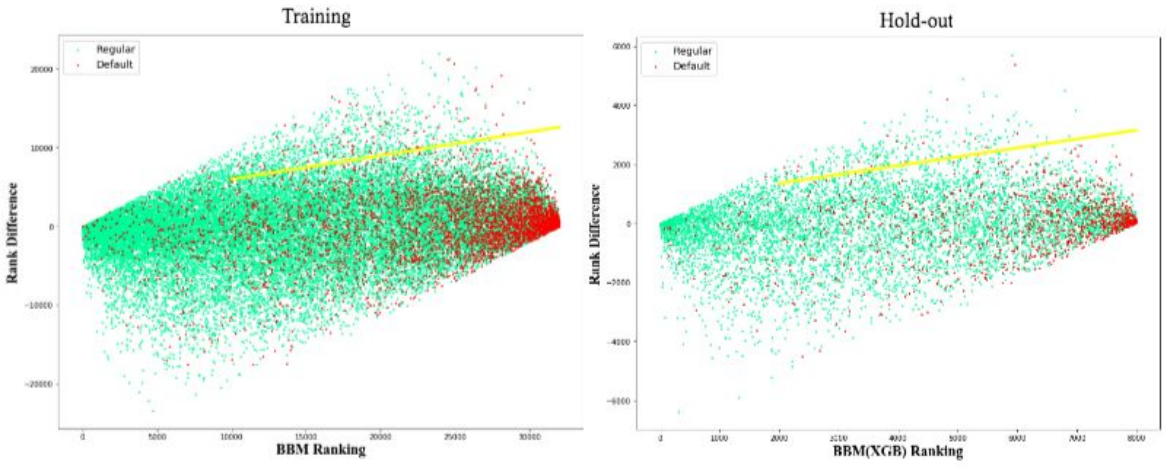

Figure 5.1: Comparison of Observed and Hold-out Data

Secondly, the robustness of Strategy One is tested. Table 5.2 compares the performance of the optimal L in the hold-out test. In the local validation, the optimal value of L is 455. In this scenario, there are 58 instances that are above the line Y = 0.3X + 3000/5. Important to note, the intercept of

this line is scaled down according to data sizes ratio. The predictions of these 58 cases are modified as zero and none of them is misclassified, therefore it witnesses an increase in AUC by 7.58E-04 (See Table 5.2).

Table: 5.2: Robustness of Strategy One

	Rows	Numbers of Top L	Numbers of Instances Set to Zero	Correct	Wrong	Δ AUC
Local Validation	6400	200	22	22	0	2.66E-04
		455	58	58	0	**7.58E-04**
		460	60	59	1	4.35E-04
		1000	138	130	8	-8.28E-03
		5000	261	238	23	-1.25E-03
Hold-out	8000	100	10	10	0	1.15E-04
		170	19	19	0	**2.00E-04**
		175	20	19	1	-3.60E-05
		570	67	62	5	-6.16E-04
		5000	261	238	23	-8.28E-03

Considering the data set size ratio of local validation to hold-out, the optimal value of L in hold-out is scaled up to 570. As shown in the table above, this value causes a loss of AUC by -6.16E-04. The reason for this drop is that 67 of the chosen 570 instances are located above the line. However, 5 of them are ‘bad’, but are misclassified as ‘good’. By contrast, the actual optimal value in hold-out turns out to be 170, which is the number before the first ‘bad’ (L equals to 175) comes out.

In accordance with the experiment results, the given strategy from the original paper is over sensitive to the change of distribution, the phenomenon that the optimal value in local validation is inconsistent with the value in hold-out, is witnessed by other data sets as well. Specifically, the order that the first ‘bad’ occurs on the list of top L ‘good’ instances makes a great impact on the performance of IPSBM. As long as the the first ‘bad’ comes out earlier than it is supposed to, AUC would suffer from a loss. In this sense, the given strategy is hardly practical.

Subsequently, the performance of the proposed strategy is examined. In contrast to Strategy One that determines L by re-fitting of specific samples, Strategy Two decides the value of L by drawing lines by different intervals (See Figure 5.2). This strategy takes the hypothesis that the relative

positions of where the ‘bad-free’ region occurs are consistent from observed to unseen data. In this sense, the chosen intervals in observed data (LHS of Figure 5.2) would create a ‘bad-free’ space in the corresponding positions in unseen data as well.

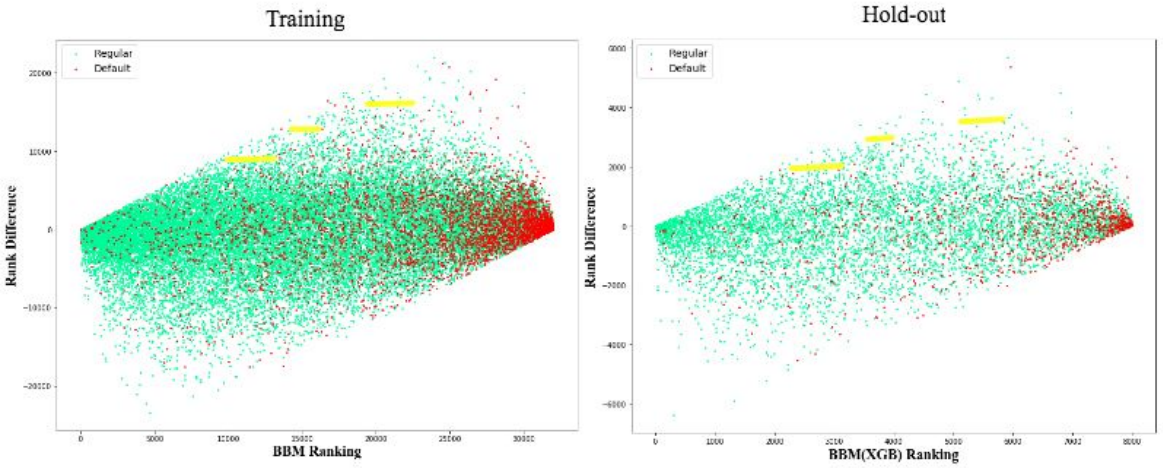

Figure 5.2: Comparison of Observed and Hold-out Data

The main purpose of this strategy is to avoid the mistakes of Strategy One. In particular, the red dots which come out in the top corner of the RHS in the chart are instances that are correctly classified by BBM as ‘bad’ but DBM makes an error. For this reason, these samples have high positive number in ranking difference. Therefore, these cases should be excluded from L though they own high ranking difference values. Ignoring these cases is exactly the mistakes made by Strategy One.

In order to test this strategy, firstly, a local validation is conducted so the values of two parameters are determined: the optimal intervals and the corresponding lowest ranking difference, which is the value of Y axis. Secondly, the corresponding optimal values are tested on the hold-out, as shown in Table 5.3, the tuned L makes an increase in AUC but it is even smaller than the achievement of Strategy One. Even worse, this phenomenon could not repeatedly show up in other data sets, which means the verified ‘bad-free’ spaces contain several ‘bad’ in hold-out, this triggers a drop in AUC. In this context, Strategy Two is proved to be subjected to distribution as well. In overall it performs almost as badly as Strategy One, but it is much more straightforward to implement.

Table: 5.3: Robustness of Strategy 2

Rows	Range	Minimum	Numbers of Instances	Correct	Wrong	Δ AUC

			Difference	Set to Zero			
Training	32000	9000-14000	9000	170	136	0	Na
		14000-16000	12000		12	0	
		20000-23000	15000		22	0	
Local Validation	6400	1800-2800	1800	27	22	0	**6.01E-04**
		2800-3200	2400		1	0	
		4000-4600	3000		4	0	
Test	8000	2250-3500	2250	36	27	0	8.20E-06
		3500-4000	3000		3	0	
		5000-5750	3750		6	0	

In conclusion, compared with Stacking, IPSBM is much more complex and is unable to provide more accurate predictions. In addition, the original strategy to tune the key parameter is fairly complicated and highly subjected to data set. Therefore, an easy-to-use approach is proposed but is proved to be unstable and inaccurate.

6 Conclusion

In the field of online banking, ensemble methods such as Random Forest and XGBoost have become the state-of-the-art approaches to identifying true creditworthiness of a potential loan borrower. These methods are proven to be powerful in predicting extremely trustworthy and considerably risky borrowers. Nonetheless, this comes at the cost of misclassifying some of the good borrowers as bad: these borrowers are not on the top or at the bottom when they are ranked based on scores of such classification rules. The industry has been exploring approaches to reducing this kind of errors in response to the negative impact from frustrated customers who are being rejected and the concern of losing market.

The ‘Partially stacking blend based user credit assessment model’ was proposed in this context. It claims to be able to provide promising improvement in saving the innocent good borrowers. In order to promote and prove this model, in this study, the details to implement this model are firstly provided. Additionally, a more practical approach to tuning the critical parameter is suggested. Lastly, a comparative assessment of three ensemble methods is conducted. In particular, the individual XGBoost, stacking, and the improved version of the model in interest are applied to four real world Internet credit data sets, i.e. Cash Bus, PaiPaiDai, Rong360 and QianHai Credit Reference. Experimental results reveal that the application of this model has brought rather small improvement for the best individual base learner, and the suggested tuning strategy is also ineffective. In conclusion, this model is barely able to make impressive progress in prediction accuracy.

Nonetheless, this algorithm provides an intuitive and efficient suggestion on which borrowers are likely to be inappropriately classified. With this information an improvement of prediction could be achieved by employing different algorithms or even experts’ domain knowledge instead of a plain LR. Therefore, there is a considerable amount of room for further work. Firstly, further analyses are encouraged to explore different classifiers. Secondly, another interesting extension to the research would be to apply these techniques on even more biased data sets. Further, it would also be of interest to look into other methods to tune the value of L. Lastly, it is highly suggested to strive to correct the bad observations that are misclassified as good, though it is even more challenging.