Overdue Rate Prediction based on Personal and Behavior Information of Clients

6202 Project

Professor: Amir Hossein Jafari
Jiaxi Jiang, Xueqi Zhou, Meijiao Wang

Due Date: 2020/06/30

Table of Contents

Introduction

Based on user's historical behavior data of the Chinese Credit Platform, we are aiming to predict the probability of users overdue in the next 6 months. In normal life, some people find it hard to obtain loans because of their no or less credit history. In order to increase the tolerance of loans for people with no bank account or credit history, credit institutions will use various alternative data: Telecommunications, transaction information, and other historical behavior data of customers can be a standard to predict customer repayment ability. Based on these data, various machine learning methods are used to make these predictions to calculate and reduce the expected loss level of credit.

There are several reasons why the proposal is worthwhile, cause Bank-like financial systems need to know their clients and try to avoid debts.

(1) The labor cost is relatively high

(2) the accuracy is difficult to guarantee by human

(3) it is easy to cause corruption if there are no good standards.

To put it simply, we want to establish fast screening algorithms to quickly determine whether to lend to this person or not, thereby improving the efficiency and profit of the financial institution. So, the important thing is to build a system, we need to extract the information from data, and find a good classifier.

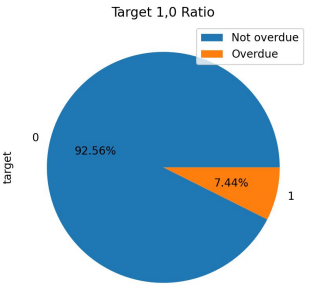
Description of dataset

We will use the database from Mirror Risk Control System, which is the first real risk control model based on big data in the industry, it includes 13.6 million online users and 12.86 billion pieces of data accumulated in 8 years. In the train dataset, we have 30 thousands observations and 19999 in the test

dataset. Originally, we had 228 features. In order to protect the privacy of clients, the organizers desensitized some features. It is large enough to train machine learning.

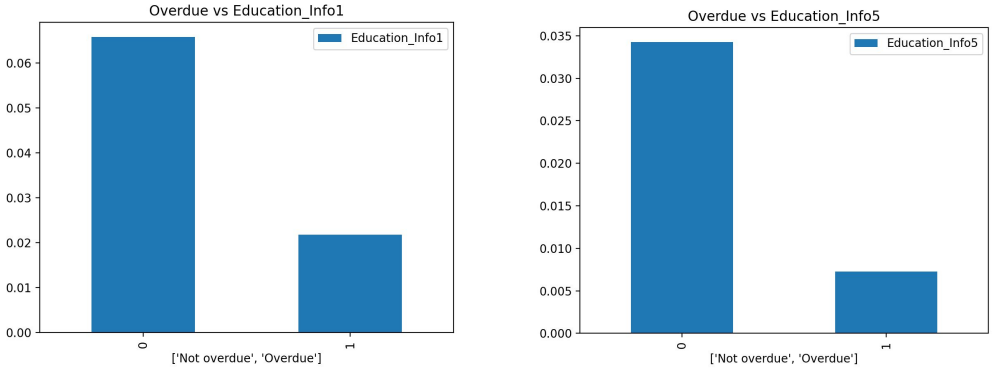
Experimental setup and Results

(1) data visualization



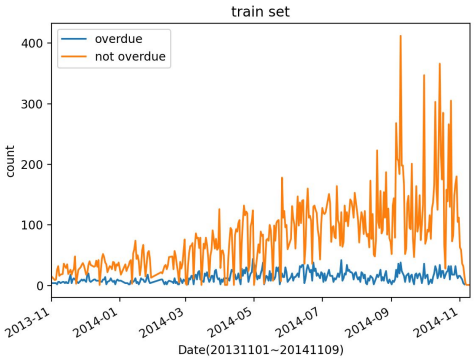
Graph 1

Graph 1 shows the comparison between the ratio of “overdue” and “not overdue” in the overall observations by pie graph. People can know that the overdue ratio is about 7% and the dataset has class-imbalanced problem.



Graph 2

Graph 2 shows the relationship between features-Education_Info1 and Education_Info5 and overdue. We are unable to tell much information from the graph, except the count of “not overdue” is larger than “overdue”.

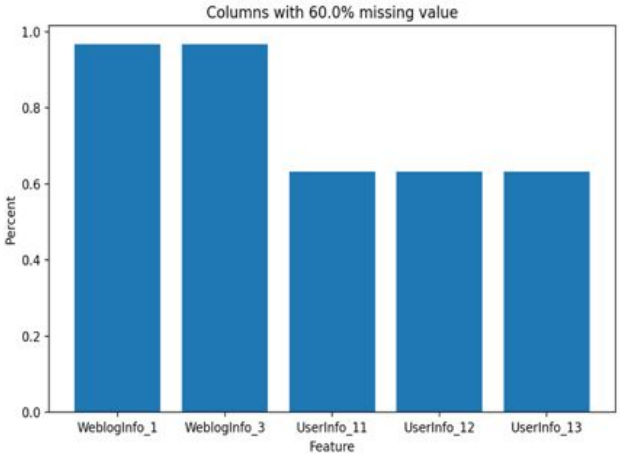


Graph 3

Graph 3 shows the fluctuation in counts of overdue and not overdue along dates. The business volume of Paipai Loan is generally increasing. The default rate is also slowly increasing at the beginning, and basically remains unchanged afterwards. In general, the default rate is stable.

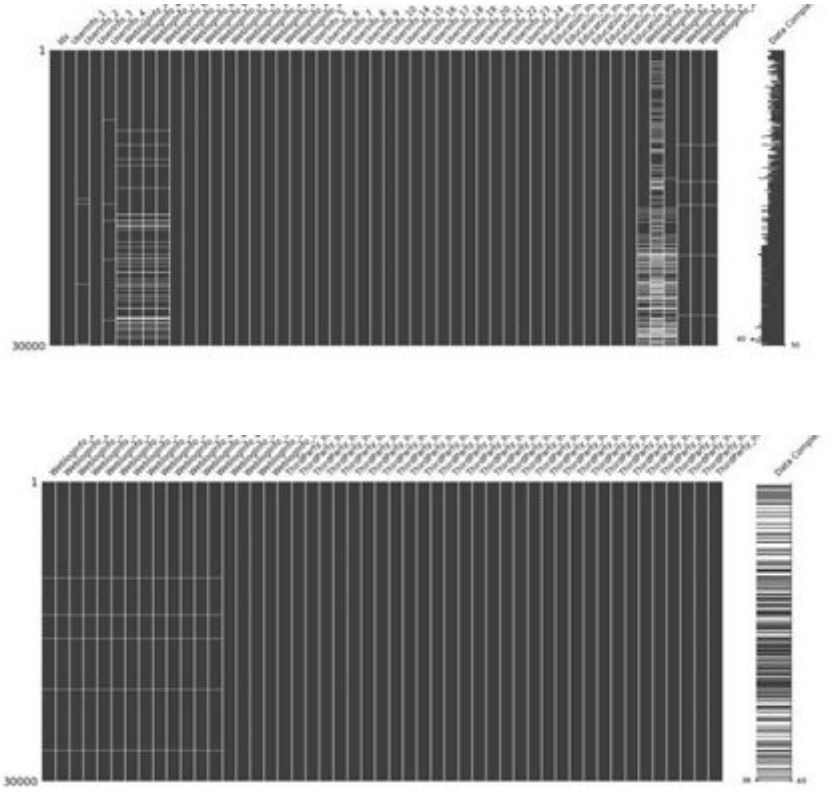
(2) data cleaning

We set 60% as our cutoff and delete the features which the ratio of missing value is larger than 60%. From the graph 2 below, we know that five features have a large ratio of missing value. The ratio of some features like WeblogInfo_1 and WeblogInfo_3 is even as high as 95%. We should definitely consider deleting these features.



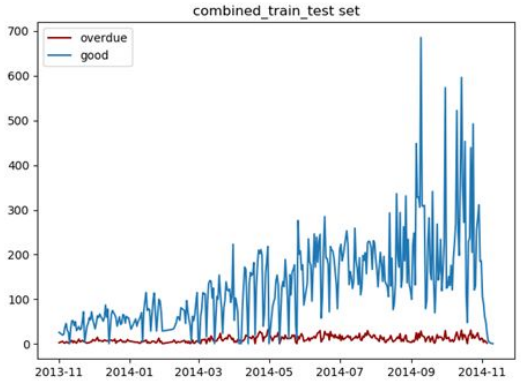
Graph 4

We used the ‘missingno’ package to detect the missing values. From graph 4, we are able to find some patterns. Instead of predicting the missing value, for convenience, we choose to change the “object” missing value with mode of its column, and change the “int” or “float” value with mean.



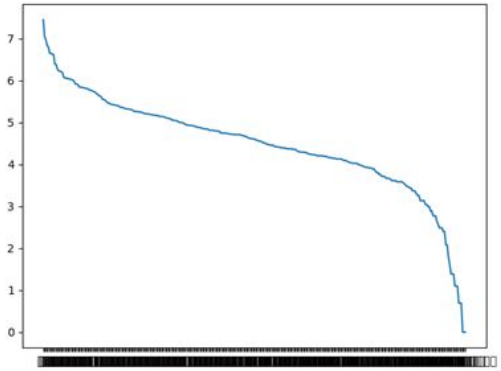
Graph 5

Then, we delete features containing ‘Telecom operators’ which is like the AT&T operator in the US, since this value has nothing to do with the result action. Next, we plot the overdue with date and group by the overdue, to see whether there are some patterns between the date and overdue(which is our target). We can see that there did have some relationships between the time and overdue rate. So we decided to cut the date into different groups to change the object into numbers. We break the time into subinterval with 10 as space length, for example we set 1~10 as 1, 11~20 as 2, ……and 29991~30000 as 3000.



Graph 6

In order to detect the potential relationship between the repeat times of cities and the overdue, we change it into log data. From graph 6, indeed we find the pattern, and cut it into 3 groups.



Graph 7

Then, we dealt with finding the standard deviation of every feature and delete the features which have standard deviation near 0.

Finally, we find that in some features, the city's names are the same but in different format, such as "重庆" equals to "重庆市". Therefore, we renamed some cities and maintained the same format for the same cities.

(3) feature engineering

(i) We calculate the overdue rate according to different provinces, and focus on the provinces which have the overdue ratio higher than 85%. Then, create new features to determine whether the provinces' names in features are these provinces. 1 means it is among the high overdue ratio provinces and 0 means not. We find 7 provinces satisfying the condition and create 7 new features based on them.

(ii) There are four columns showing the city's information. They are UserInfo_2, UserInfo_4, UserInfo_8, and UserInfo_20. I tried to figure out whether there are differences in the city's name between every two columns and create new features-"diff_". For example, diff_24 means difference between UserInfo_2 and UserInfo_4. 1 means two city's names are the same and 0 means not.

(iii) For the features containing cities' names, we give them rank based on their prosperity. 1 means first-tier cities, 2 means second-tier cities, and 3 means third-tier cities.

(iv) Except the features we dealt with above, for the rest of the features, we made it one hot encoder.

(4) feature selection

We are trying to select the first 200 most important features by the combination of five models. The models are Pearson correlation selector, chi-squared selector, random forest selector, logistic selector, and RFE. For every selector, it will create a new column showing whether to select the feature. We rank the feature importance based on the sum of the five new columns.

(5) Model Building

1. Choose the number of features that we have selected before

   n_samples:total number of samples

   n_informative: Number of multi-information features

   n_redundant: Redundant information, random linear combination of informative features

   n_repeated: Repeat information, randomly extract n_informative and n_redundant features

   n_features = n_informative + n_redundant+ n_repeated

2. Get the stacking model

   What is the stacking? Let us use an example to explain that.

   The first half is to use a basic model for 5-fold cross-validation, such as: using XGBoost as the basic model Model 1, 5-fold cross-validation is to take out four folds as training

data (training set), the other one as validation data (validation set). In stacking, this part of data will use the entire training data. For example: suppose our entire training data contains 10000 rows of data, and test data contains 2500 rows, then each cross-validation is actually to divide the training data from the training set and the verification set. In each cross-validation, the training data will be 8000 rows, validation data is 2000 rows.

Each cross-validation consists of two processes: 1. Train the model based on training data; 2. Predict validation data based on the model generated by training data training. After the entire first cross-validation is completed, we will get the predicted value of the current validation data, which will be a 2000-row, 1-column data, denoted as a1. After this part of the operation is completed, we also need to make predictions on the original test data of the data set. This process will generate 2500 predicted values. This part of the predicted values will be used as part of the next layer of model test data, denoted as b1. Because we are performing 5-fold cross-validation, the process we mentioned above will be carried out five times, and finally will generate 2000 rows and 5 columns of data a1, a2, a3, a4, and a5, predicted for validation data. The forecast will be 2500 rows and 5 columns of data b1, b2, b3, b4, b5.

After completing the entire steps for Model 1, we can find that a1, a2, a3, a4, and a5 are actually the predicted values of the original training data. Putting them together, a matrix of 10,000 rows and one column will be formed, denoted as A1 . For the data of b1, b2,

b3, b4, and b5, we add the average of each part to obtain a matrix with 2500 rows and 1 column, which is denoted as B1.

The above description is the complete process of a model in stacking. The same layer in stacking usually contains multiple models. Suppose there are Model 2: LR, Model3: RF, Model4: GBDT, and Model 5: SVM. For these four models, we can repeat the above steps, after the end of the whole process, we can get the new A2, A3, A4, A5, B2, B3, B4, B5 matrix.

After this, we merged A1, A2, A3, A4, A5 side by side to get a 10000 rows and 5 columns matrix as training data, B1, B2, B3, B4, B5 side by side merged and got a 2500 rows and 5 columns matrix as test data. Let the next layer of models be further trained based on them.

In the next layer of model training, the matrix of 10000 rows and 5 columns is used as the feature matrix of the model, and the True Labels of the most original training data is used as the output matrix of the model for model training. After the model training is completed, the processed 2500 rows and 5 columns of test data are used as the feature matrix, and the model is used to output 2500 rows and 1 column as the result (Hussain, 2020).

3. Get the individual models

Like knn, LogisticRegression, DecisionTree, svm, nn, xgb, also the stacking

Logistic regression

Logistic regression is one of the generalized linear regression analysis models, and its essence belongs to the classification problem, so it is mainly used when the explanatory variable is a categorical (discrete, such as 0, 1) variable. In the classification problem, logistic regression is better than linear regression, because linear regression will have a negative probability when fitting the explanatory variable to be discrete, which will lead to wrong sample classification. Logistic regression uses a logarithmic function to compress the prediction range between 0 and 1, thereby improving the prediction accuracy ([Machine Learning] Logistic regression, 2020).

From the original data, we select a sample dataset of 3000 rows and 50-dimensional features for model training. The features are processed dynamically according to the cross-validation feedback of the model. Throughout the entire modeling process, the single-factor analysis and the numerical features are discretized for the output of the important feature after the xgboost model is trained. Construct and add combined features. Because linear models lack accurate characterization of non-linear relationships, feature combinations can add non-linear expressions to enhance the expressive power of the model. When the model has too many parameters, it is easy to encounter the problem of overfitting.

It is necessary to control the complexity of the model. The typical approach is to add regular items to the optimization target to prevent overfitting by penalizing too large parameters. L1 regularization tends to make the parameter become 0, and feature selection can be performed for high latitudes. The data has a good effect, so we choose L1 regularity.

KNN:

Compared with other algorithms, the KNN algorithm is a particularly well-implemented and easy-to-understand classification algorithm, which is mainly classified according to the distance between different features. The general classification algorithm must first train a model, and then use the test set to test the model, but the KNN algorithm does not need to train the model, directly using the distance between the sample to be tested and the training sample to achieve classification.

1st. Step: Calculate the distance between test data and each row of training data

2nd. Step: Now, based on the distance value, sort them in ascending order.

3rd. Step: Next, it will choose the top K rows from the sorted array.

4th. Step: Final, it will assign a class to the test point based on the most frequent class of these rows (KNN Algorithm - Finding Nearest Neighbors).

DecisionTree-CART

Classification and regression trees are a kind of decision tree algorithm. It is the same as other decision tree algorithms. It is also composed of feature selection, tree generation, and

pruning. CART is widely used and is used in the integrated model of trees. This type of decision tree can solve both classification problems and regression problems.

Therefore, for classification trees and regression trees, CART has two generation methods. For any machine learning model, the training process is similar, the purpose is to make the training set data as good as possible to classify and fit. Whether CART is a classification tree or a regression tree, it is necessary to minimize certain errors in the training process and arrange the training data most appropriately (A Beginner's Guide to Classification and Regression Trees).

SVM

SVM refers to a series of machine learning methods. The basis of such methods is the support vector algorithm. The basic principle of the SVM algorithm is to find a hyperplane that can distinguish between the two types, so that the margin is maximized (An Introduction to Support Vector Machines (SVM), 2017). All points located on the marginal hyperplanes on both sides are called Support Vectors. We use the SVM algorithm to solve the classification problem, which is SVC.

The goal of SVC is to return a "best match" hyperplane to divide or classify your data. After the hyperplane is obtained, the above feature values are provided to the classifier to view the prediction. This algorithm is very suitable for our needs

MLP

MLP can implement nonlinear discriminants, and if used for regression, can approximate the input nonlinear function. In fact, MLP can be used for "universal approximation", that is, it can be proved that any function with continuous input and output can be approximated by MLP. It has been shown that MLP with a hidden layer can learn any nonlinear function of the input(Jeshwanth, 2015).

In the neural network, each node is a perceptron, the basic function of the neurons in the model biological neural network: the electrical signals from the outside are transmitted to the neurons through synapses. When the sum of the signals received by a cell exceeds a certain threshold, The cell is activated and sends electrical signals to the next cell through the axon to complete the processing of outside information.

However, the perceptron learning algorithm cannot be directly applied to the parameter learning of the multi-layer perceptron model. The originally proposed learning scheme is: In addition to the last neuron, the weights of all other neurons are fixed in advance. The learning process is only to use the perceptron learning algorithm to learn the weight coefficients of the last neuron.

In fact, this is equivalent to transforming the original feature space into a new feature space through the first layer of neurons. Each neuron in the first layer constitutes one

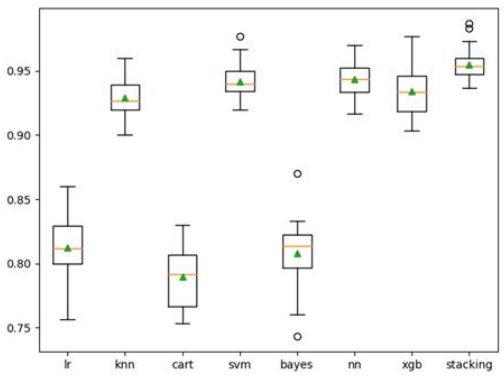
dimension of the new space, and then learns with a perceptron in the new feature space. The algorithm constructs a linear classifier (Introduction to Artificial Neural Networks(ANN), 2019).

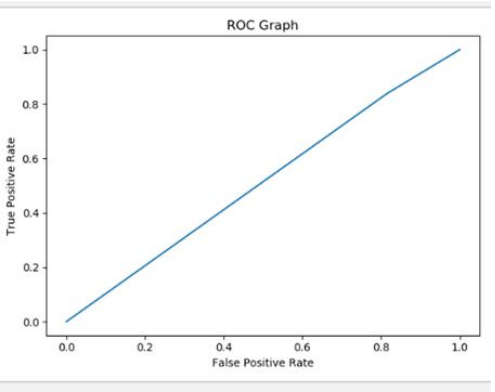
Obviously, since the weights of the first neuron layer need to be given manually, the performance of the model depends greatly on whether an appropriate first layer neuron model can be designed, and this depends on the problems and data faced. Understand, there is no way to solve the first layer of neuron parameters for any problem. The core idea is the gradient descent method, that is, the degree to which the training samples are misclassified as the target function, and each time an error occurs during training, the weight coefficient is updated in the direction of the negative gradient of the target function relative to the weight coefficient, knowing that the target has not been until the wrong sample.

4. Kfold, 10-fold

5. Got the mean absolute error (MAE). The scikit-learn library inverts the sign on this error to make it maximizing, from negative infinity to 0 for the best score.

(6) Accuracy





```
[0 1 1 ... 1 0 1]
[[ 3481 15407]
 [  179   933]]
0.057099143206854344
0.8390287769784173
0.1069218427687371
0.5116628425341049
```

Graph 8

The first row is our prediction. Then, it's a confusion matrix. And the next four rows represent precision score, recall-score, f1-score and AUC respectively.

Conclusion

From the accuracy calculation, we are able to know that the prediction is not accurate. In order to improve the model, we should definitely put more efforts on feature engineering and building models. For features which are simply converted into one hot encoder, we definitely have better ways of dealing with it. We simply use the pd.get_dummies function to change the rest of objects into dummy variables. In fact, what we should do is to use the pd.factorize, and pd.qcut to treat different types of variables, maybe we will do that in the next Kaggle game.

References

Hussain, M. (2020, May 30). *Ensemble learning with Stacking and Blending*.

Available from: https://www.mygreatlearning.com/blog/ensemble-learning/ [20 June 2020].

[Machine Learning] Logistic regression. (2020, June 13).

Available from: https://zhuanlan.zhihu.com/p/74874291

KNN Algorithm - Finding Nearest Neighbors.

Available from: https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_knn_algorithm_finding_nearest_neighbors.htm

A Beginner's Guide to Classification and Regression Trees.

Available from: https://www.digitalvidya.com/blog/classification-and-regression-trees/ [23 January 2019]

An Introduction to Support Vector Machines (SVM). (2017, June 22).

Available from: https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/#:~:text=A%20support%20vector%20machine%20(SVM,on%20a%20text%20classification%20problem.

Jeshwanth C.(2015) Adaptive Multiple Optimal Learning Factors for Neural Network Training, The University of TEXAS at Arlington

Introduction to Artificial Neural Networks(ANN). (2019, October 10).

Available from: https://towardsdatascience.com/introduction-to-artificial-neural-networks-ann-1aea15775ef9