MA-691 : Project
Loan Payment Prediction using Combined Regression Strategy (COBRA)

1. Kartikeya Kumar Gupta - 180123020
2. Naman Goyal - 180123029
3. Ayaz Anis - 180123007
4. Rathod Vijay Mahendra - 180123037
5. Kushal Jhanwar - 180123025

Instructor : Dr. Arabin Kumar Dey

Submission Date : 20-11-2021

---

1 Introduction

In the lending industry, investors provide loans to borrowers in exchange for the promise of repayment with interest. If the borrower repays the loan, then the lender profits from the interest. However, if the borrower is unable to repay the loan, then the lender loses money. Therefore, lenders face the problem of predicting the risk of a borrower being unable to repay a loan.

At the backend, the application uses Combined Regression Strategy (COBRA) [1] algorithm trained on Loan Payment Prediction Dataset from Kaggle [dataset]. The remaining part of report is organized as follows: In the next section we provide a brief outline of working of COBRA algorithm. Then in section 3, we explain the dataset that has been used and in section 4 we discuss the experimental results on the training and test data.

2 Combined Regression Strategy (COBRA)

Suppose that one has available M classifiers $C_{n,1}(\mathbf{x}), \ldots, C_{n,M}(\mathbf{x})$, where each data-based classifier $C_{n,m}$ is a map $C_{n,m} : \left\{\Re^d \times \{1, \ldots, K\}\right\}^n \times \Re^d \to \{1, \ldots, K\}$. The aim is then to combine

$C_{n,1}(\mathbf{x}), \ldots, C_{n,M}(\mathbf{x})$, as best as possible, to produce a new classifier for predicting Y. More generally for the $\{1, \ldots, K\}^M$-valued random vector $\left(C_{n,1}(\mathbf{x}), \ldots, C_{n,M}(\mathbf{x})\right)$, one would like to find a classifier $\psi : \{1, \ldots, K\}^M \to \{1, \ldots, K\}$ for which the error rate

$$\mathrm{err}_n(\psi) = P\left\{\psi\left(C_{n,1}(\mathbf{X}), \ldots, C_{n,M}(\mathbf{X})\right) \neq Y \mid T_n\right\}$$

is, somehow, as small as possible. Observe that in this combined classification setup, the map ψ acts on the vector $\left(C_{n,1}(\mathbf{X}), \ldots, C_{n,M}(\mathbf{X})\right)$ and not the feature vector $\mathbf{X}$. It is this feature that makes this problem different from the more traditional setup where ψ maps $\mathbf{X}$ to $\{1, \ldots, K\}$. Our proposed procedure, which is based on a class-majority vote, works as follows. Elect class k if for any other class $l \neq k$,

$$\begin{array}{l} \sum_{i:Y_i=k} I\left\{C_{n,1}\left(\mathbf{X}_i\right) = C_{n,1}(\mathbf{x}), \ldots, C_{n,M}\left(\mathbf{X}_i\right) = C_{n,M}(\mathbf{x})\right\} > \\ \sum_{i:Y_i=l} I\left\{C_{n,1}\left(\mathbf{X}_i\right) = C_{n,1}(\mathbf{x}), \ldots, C_{n,M}\left(\mathbf{X}_i\right) = C_{n,M}(\mathbf{x})\right\} \quad (1) \end{array}$$

That is, elect/choose class k if the number of class k data points for which all M classifiers predict (correctly or incorrectly) the same class, at both the data point and the new observation, is larger than for any of the other classes. This is a class-majority vote in the sense that one is choosing the class in which the number of observations (voters) satisfying (voting for) a certain condition (that their predicted class, by each classifier, be the same as the predicted class of the new observation) is the largest.

The motivation behind (1) is that if the individual classifiers $C_1, \ldots, C_M$ are all nonrandom (perhaps an unrealistic case), then (1) is simply a partitioning-type classification rule based on the iid discretized "data":

$$\left\{\left(C_1\left(\mathbf{X}_i\right), \ldots, C_M\left(\mathbf{X}_i\right)\right), Y_i\right\}, \quad \text{for} \quad i = 1, \ldots, n$$

Now the consistency properties of the general partitioning rules, as well as their simplicity, renders a procedure such as (1) intuitively appealing for nonrandom classifiers. Un-fortunately, in practice, the iid structure is lost in (1) as the $C_{n,j}$'s themselves depend on the data.

Equivalently, denoting our combined classifier by $\psi_n\left(C_{n,1}(\mathbf{x}), \ldots, C_{n,M}(\mathbf{x})\right)$, the foregoing classifier can be rewritten as

$$\begin{array}{l} \psi_n\left(C_{n,1}(\mathbf{x}), \ldots, C_{n,M}(\mathbf{x})\right) \\ = \mathrm{argmax}_{1 \leq k \leq K} \sum_{i=1}^{n} \prod_{j=1}^{M} I\left\{C_{n,j}\left(\mathbf{X}_i\right) = C_{n,j}(\mathbf{x})\right\} \times I\left\{Y_i = k\right\} \end{array} \quad (2)$$

In the case of ties, I take $\psi_n\left(C_{n,1}(\mathbf{x}), \ldots, C_{n,M}(\mathbf{x})\right)$ to be the smallest k for which the above equation holds. In the more popular two-class problem, with $Y = 0$ or 1, this simple rule reduces to

$$\psi_n\left(C_{n,1}(\mathbf{x}),\ldots,C_{n,M}(\mathbf{x})\right) = \begin{cases} 1 \text{ if } \sum_{i=1}^{n}\prod_{j=1}^{M} I\left\{C_{n,j}\left(\mathbf{X}_i\right)=C_{n,j}(\mathbf{x})\right\} I\left\{Y_i=1\right\} \\ > \sum_{i=1}^{n}\prod_{j=1}^{M} I\left\{C_{n,j}\left(\mathbf{X}_i\right)=C_{n,j}(\mathbf{x})\right\} I\left\{Y_i=0\right\} \\ 0 \text{ otherwise.} \end{cases} \tag{3}$$

Note that in the discretized "data" vectors $\left(C_{n,1}\left(\mathbf{X}_i\right),\ldots,,\ C_{n,M}\left(\mathbf{X}_i\right)\right), i=1,\ldots,n$ are not iid and in fact depend on T_n.

In our analysis we have used the following classifiers :

- Stochastic Gradient Descent (SGD) Classifier
- Support Vector Classification
- K Neighbors Classifier
- Decision Tree Classifier

3 Dataset

For repayment of loans, we are using publicly available data from LendingClub.com, a website that connects borrowers and investors over the Internet. We are using Loan Repayment Prediction Dataset from Kaggle [dataset] to train our model. The dataset contains 3078 observations (rows) and 15 features (columns) per row.
Here are what the columns represent:

1. sl.no: Indicates the serial number.
2. credit.policy: It's value is 1 if the customer meets the credit underwriting criteria of LendingClub.com, and 0 otherwise.
3. int.rate: Indicates the interest rate of the loan, as a proportion (for example, a rate of 11% would be stored as 0.11). Borrowers who are judged by LendingClub.com to be more risky are assigned higher interest rates.
4. installment: Indicates the monthly installments owed by the borrower if the loan is funded.
5. log.annual.inc: Indicates the natural log of the self-reported annual income of the borrower.

6. dti: Indicates the debt-to-income ratio of the borrower, i.e amount of debt divided by annual income.

7. fico: Indicates the FICO credit score of the borrower.

8. days.with.cr.line: Indicates the number of days the borrower has had a credit line.

9. revol.bal: Indicates the borrower's revolving balance i.e amount which is left unpaid at the end of the credit card billing cycle.

10. revol.util: Indicates the borrower's revolving line utilization rate i.e the amount of the credit line used relative to total credit available.

11. inq.last.6mths: Indicates the borrower's number of inquiries by creditors in the last 6 months.

12. delinq.2yrs: Indicates the number of times the borrower had been 30+ days past due on a payment in the past 2 years.

13. pub.rec: Indicates the borrower's number of derogatory public records (bankruptcy filings, tax liens, or judgments).

14. not.fully.paid: It's value is 1 if the amount is not fully paid before maturity, and 0 otherwise.

The dataset is divided into training and testing sets and it used for training machine learning model to predict loan payment repayment. Looking at the dataset the following conclusions can be drawn:-

1. More no. of people prefer lower interest rates and after a point, no. of people decreases with increasing interest rates.

2. On a general trend, no. of people decreases with increasing monthly installment.

3. Curve of log.annual.inc is higher in the middle which indicates more no. of people tend to have average annual income.

4. The no. of people increases with increasing debt-to-income ratio initially then it decreases. Same pattern is followed for the FICO credit score.

5. The no. of people increases rapidly with increasing days.with.cr.line initially then it decreases slowly to being almost zero at high days.with.cr.line.

4 Training and Testing the Model

We divided the data into training set and testing set with a 7:3 ratio and trained COBRA Model on the above introduced dataset. Then we explicitly tested it on the test set to evaluate it's performance. Click here to go to our Github repository.

There are four ways to check if the predictions are right or wrong:

- TN / True Negative: the case was negative and predicted negative
- TP / True Positive: the case was positive and predicted positive
- FN / False Negative: the case was positive but predicted negative
- FP / False Positive: the case was negative but predicted positive

Precision is the ability of a classifier not to label an instance positive that is actually negative. Precision reflects the accuracy of positive predictions.

$$Precision = TP/(TP + FP)$$

Recall is the ability of a classifier to find all positive instances. Recall reflects the fraction of positives that were correctly identified.

$$Recall = TP/(TP + FN)$$

The **F1 score** is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0. F1 scores are lower than accuracy measures as they embed precision and recall into their computation. As a rule of thumb, the weighted average of F1 should be used to compare classifier models, not global accuracy.

$$F1Score = 2 * (Recall * Precision)/(Recall + Precision)$$

Support is the number of actual occurrences of the class in the specified dataset. Support must be balanced in the training set to ensure a structural stability in the reported scores of the classifier. Support doesn't change between models but instead diagnoses the evaluation process.

The below figure shows the classification report generated.

	precision	recall	f1-score	support
0	0.67	0.69	0.68	608
1	0.69	0.67	0.68	624
accuracy			0.68	1232
macro avg	0.68	0.68	0.68	1232
weighted avg	0.68	0.68	0.68	1232

Figure 1: Classification Report

5 Conclusion

- We have combined 4 classifiers (Stochastic Gradient Descent Classifier, Support Vector Classification, K Neighbors Classifier, Decision Tree Classifier) to make a Combined Classifier.
- We have obtained a loan repayment dataset and divided it into training set and testing set with a 7:3 ratio and trained our Combined Classifier model on it.
- We observed that the accuracy increases in case of the Combined classifier (which is about 70%) from about 50% in case of an individual classifier.

References

[1] Majid Mojirsheibani. Combining classifiers via discretization. *Journal of the American Statistical Association , Jun., 1999, Vol. 94, pp. 600-609*, Jun., 1999.