(https://weclouddata.com)

BLOG

STUDENT BLOG

Credit Scoring Using Machine Learning

October 28, 2019

The credit score is a numeric expression measuring people's creditworthiness. The banking usually utilizes it as a method to support the decision-making about credit applications. In this blog, I will talk about how to develop a standard scorecard with Python (Pandas, Sklearn), which is the most popular and simplest form for credit scoring, to measure the creditworthiness of the customers.

Project Motivation:

Nowadays, creditworthiness is very important for everyone since it is regarded as an indicator of how dependable an individual is. In various situations, service suppliers need to evaluate customers' credit history first and then decide whether they will provide the service or not. However, it is time-consuming to check the entire personal portfolios and generate a credit report manually. Thus, the credit score is developed and applied for this purpose because it is time-saving and easily comprehensible.

The process of generating the credit score is called credit scoring. It is widely applied in many industries especially in the banking. The banks usually use it to determine who should get credit, how much credit they should receive, and which operational strategy can be taken to reduce the credit risk. Generally, it contains two main parts:

- Building the statistical model
- Applying a statistical model to assign a score to a credit application or an existing credit account

Here I will introduce the most popular credit scoring method called scorecard. There are two main reasons why the scorecard is the most common form of credit scoring. First, it is easy to interpret for people who have no related background and experience, such as the clients. Second, the development process of the scorecard is standard and widely understood, which means the companies don't have to spend much money on it. A sample scorecard is shown below. I will talk about how to use it later.

Characteristic	Attribute	Scorecard Points
AGE	<22	100
AGE	22<=AGE<26	120
AGE	26<=AGE<29	185
AGE	29<=AGE<32	200
AGE	32<=AGE<37	210
AGE	37<=AGE<42	225
AGE	>=42	250
HOME	OWN	225
HOME	RENT	110
INCOME	<10000	120
INCOME	10000<I=NCOME<17000	140
INCOME	17000<=INCOME<28000	180
INCOME	28000<=INCOME<35000	200
INCOME	35000<=INCOME<42000	225
INCOME	42000<=INCOME<58000	230
INCOME	>=58000	260

Data Exploration and Feature Engineering:

Now I'm going to give some details about how to develop a scorecard. The dataset I used here is from the Kaggle competition. The detailed information is listed in the Figure-2. The first variable is the target variable, which is a binary categorical variable. And the rest of the variables are the features.

Variable Name	Description	Type
SeriousDlqin2yrs	Person experienced 90 days past due delinquency or worse	Y/N
RevolvingUtilizationOfUnsecuredLines	Total balance on credit cards and personal lines of credit except real estate and no installment debt like car loans divided by the sum of credit limits	percentage
Age	Age of borrower in years	integer
NumberOfTime30-59DaysPastDueNotWorse	Number of times borrower has been 30-59 days past due but no worse in the last 2 years	integer
NumberOfTime60-89DaysPastDueNotWorse	Number of times borrower has been 60-89 days past due but no worse in the last 2 years	integer
NumberOfTime90DaysPastDueNotWorse	Number of times borrower has been 90 days or more past due	integer
DebtRatio	Monthly debt payments, alimony, living costs divided by monthly gross income	percentage
MonthlyIncome	Monthly income	real
NumberOfOpenCreditLinesAndLoans	Number of Open loans (installment like car loan or mortgage) and Lines of credit (e.g. credit cards)	integer
NumberRealEstateLoansOrLines	Number of mortgage and real estate loans including home equity lines of credit	integer
NumberOfDependents	Number of dependents in family excluding themselves (spouse, children etc.)	integer

After gaining an insight into the data set, I start to apply some feature engineering methods on it. First, I check each feature if it contains missing values, and then impute the missing values with median.

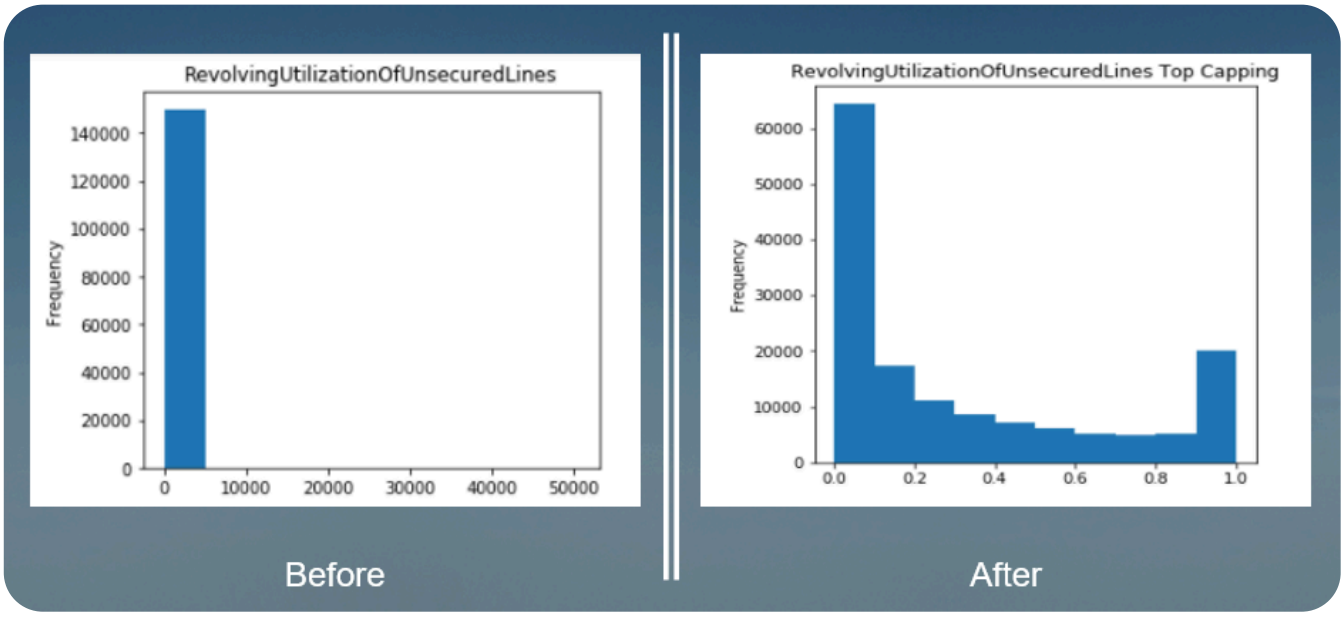
```
df_train.isnull().mean()
```

```
SeriousDlqin2yrs                      0.000000
RevolvingUtilizationOfUnsecuredLines  0.000000
age                                   0.000000
NumberOfTime30-59DaysPastDueNotWorse  0.000000
DebtRatio                             0.000000
MonthlyIncome                         0.198207
NumberOfOpenCreditLinesAndLoans       0.000000
NumberOfTimes90DaysLate               0.000000
NumberRealEstateLoansOrLines          0.000000
NumberOfTime60-89DaysPastDueNotWorse  0.000000
NumberOfDependents                    0.026160
dtype: float64
```

```
from sklearn.preprocessing import Imputer
imp_1 = Imputer(missing_values='NaN', strategy='median', axis=0)
imp_2 = Imputer(missing_values='NaN', strategy='median', axis=0)

imp_1.fit(df_train['MonthlyIncome'].values.reshape(-1, 1))
df_train['MonthlyIncome'] = imp_1.transform(df_train['MonthlyIncome'].values.reshape(-1, 1))
imp_2.fit(df_train['NumberOfDependents'].values.reshape(-1, 1))
df_train['NumberOfDependents'] = imp_2.transform(df_train['NumberOfDependents'].values.reshape(-1, 1))
```

Next, I do the outlier treatment. Generally, the methods used for outliers depends on the type of outliers. For example, if the outlier is due to mechanical error or problems during measurement, it can be treated as missing data. In this data set, there are some extremely large value, but they are all reasonable values. Thus, I apply top and bottom coding to deal with them. In Figure-3, you can see after applying the top coding, the distribution of the feature is more normal.



According to the sample scorecard shown in Figure-1, it is obvious that each feature should be grouped into various attributes (or groups). There are some reasons for grouping the features.

- Gain an insight into relationships attributes of a feature and performance.
- Apply linear models on nonlinear dependencies.
- Understand deeper on the behaviours of risk predictors, which can help in developing better strategies for portfolio management.

Binning is a proper method used for this purpose. After the treatment, I assign each value to the attribute in which it should be, which also means all numeric values are converted to categorical. Here is an example of the outcome of binning.

age_capped	age_capped_bin
45	A:40-50
40	A:30-40
38	A:30-40
30	A:0-30
49	A:40-50
74	A:70-80
57	A:50-60
39	A:30-40
27	A:0-30
57	A:50-60

After grouping all the features, the feature engineering is completed. Next step is to calculate the weight of evidence for each attribute and the information value for each characteristics (or feature). As mentioned before, I have used binning to convert all numeric value into categorical. However, we cannot fit model with these categorical values, so we have to assign some numeric values to these groups. The purpose of the Weight of Evidence (WoE) is exactly to assign a unique value to each group of categorical variables. The Information Value (IV) measures predictive power of the characteristic, which is used for feature selection. The formula of WoE and IV is given below. Here the "Good" means the customer won't have serious delinquency or target variable is equal to 0, and "Bad" means the customer will have serious delinquency or target variable is equal to 1.

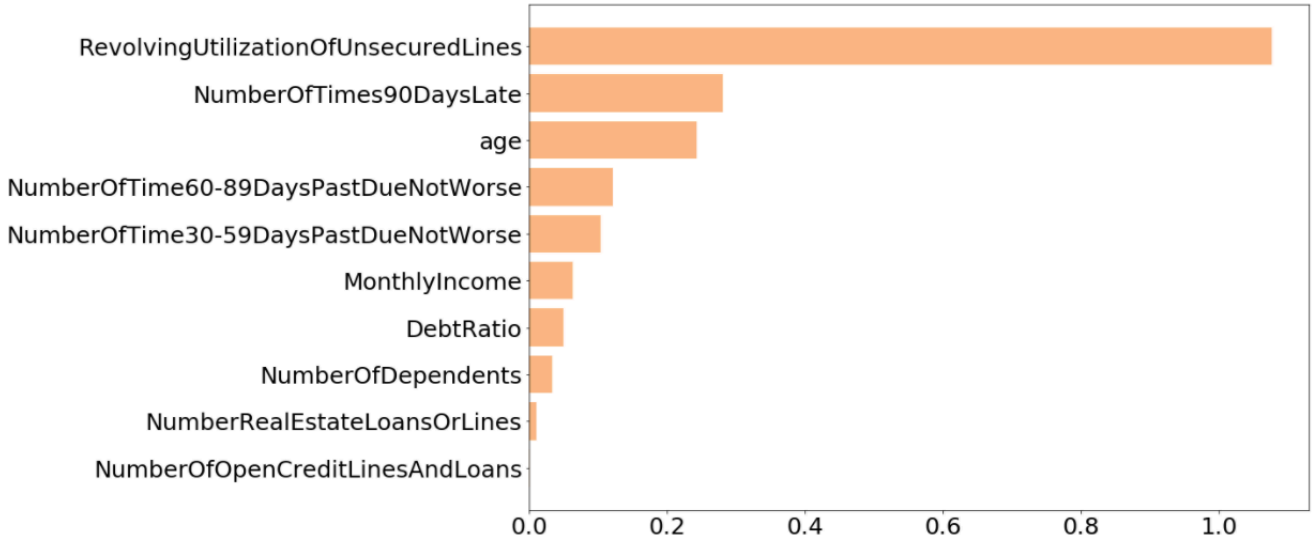
$$\text{WoE:} \qquad [\, ln\left(\frac{\text{Distr Good}}{\text{Distr Bad}}\right)] \times 100.$$

$$\text{IV:} \qquad \sum_{i=1}^{n} (\text{Distr Good}_i - \text{Distr Bad}_i) * ln\left(\frac{\text{Distr Good}_i}{\text{Distr Bad}_i}\right)$$

Usually, characteristics analysis reports are produced to get WoE and IV. Here I define a function in Python to generate the reports automatically. As an example, the characteristics analysis report for "Age" is shown in Figure-5.

Age								IV = 0.242747
Attribute	All	Bad	Good	Total Distribution	Bad Rate	Distribution Good	Distribution Bad	WoE
A:0-30	10758	1244	9514	0.07172	0.115635	0.06797	0.124077	-60.1843
A:30-40	24339	2390	21949	0.16226	0.098196	0.156808	0.23838	-41.8847
A:40-50	35037	2893	32144	0.23358	0.08257	0.229643	0.28855	-22.8343
A:50-60	34806	2149	32657	0.23204	0.061742	0.233308	0.214343	8.478164
A:60-70	27424	952	26472	0.182827	0.034714	0.189121	0.094953	68.90028
A:70-80	12700	298	12402	0.084667	0.023465	0.088602	0.029723	109.2245
A:80-90	4447	89	4358	0.029647	0.020013	0.031134	0.008877	125.4857
A:90-130	489	11	478	0.00326	0.022495	0.003415	0.001097	113.544

Then I make a bar chart to compare the IV of all the features. In the bar chart, you can see the last two features “NumberOfOpenCreditLinesAndLoans” and “NumberRealEstateLoansOrLines” have pretty low IV, so here I choose other eight feature for model fitting.



Model Fitting and Scorecard Point Calculation:

After the feature selection, I replace the attributes with the corresponding WoE. Until now, I get the proper data set for the model training. The model used for developing scorecard is a logistic regression, which is a popular model for binary classification. I apply cross-validation and grid search to tune the parameters. Then, I use the test data set to check the prediction accuracy of the model. Since the Kaggle won’t give the values for target variable, I have to submit my result online to obtain the accuracy. To show the effect of data processing, I train the model with raw data and the processed data. Based on the result given by the Kaggle, the accuracy is improved from 0.693956 to 0.800946 after the data processing.

The final step is calculating the scorecard point for each attribute and produce the final scorecard. The score for each attribute can be calculated with the formula:

Score = (β×WoE+ α/n)×Factor + Offset/n

Where:
β — logistic regression coefficient for characteristics that contains the given attribute
α — logistic regression intercept
WoE — Weight of Evidence value for the given attribute
n — the number of characteristics included in the model
Factor, Offset — scaling parameter

The first four parameters have already been calculated is the previous part. The following formulas are used for calculating factor and offset.

- Factor = pdo/Ln(2)
- Offset = Score — (Factor × ln(Odds))

Here, pdo means points to double the odds and the bad rate has been already calculated in the characteristics analysis reports above. If a scorecard has the base odds of 50:1 at 600 points and the pdo of 20 (odds to double every 20 points), the factor and offset would be:
Factor = 20/Ln(2) = 28.85
Offset = 600- 28.85 × Ln (50) = 487.14

When finishing all the calculation, the process of developing the scorecard is done. Part of the scorecard is shown in Figure-7.

Characteristics	Attribute	Scorecard Point
Age	A:0-30	49
Age	A:30-40	51
Age	A:40-50	53
Age	A:50-60	57
Age	A:60-70	65
Age	A:70-80	70
Age	A:80-90	72
Age	A:90-130	71
DebtRatio	DR:0-0.2	57
DebtRatio	DR:0.2-0.4	59
DebtRatio	DR:0.4-0.6	55
DebtRatio	DR:0.6-0.8	52
DebtRatio	DR:0.8-1.0	50
DebtRatio	DR:1.0-1.2	49
DebtRatio	DR:1.2-1.4	48
DebtRatio	DR:1.4-1.6	53
MonthlyIncome	MI:0-2000	53
MonthlyIncome	MI:2000-4000	52
MonthlyIncome	MI:4000-6000	57
MonthlyIncome	MI:6000-8000	58
MonthlyIncome	MI:8000-10000	60
MonthlyIncome	MI:10000-12000	63
MonthlyIncome	MI:12000-14000	63
MonthlyIncome	MI:14000-16000	61
MonthlyIncome	MI:16000+	61

When you have new customers coming, you just need to find the correct attribute in each characteristic according to the data and get the score. The final credit score can be calculated as the sum of the score of each characteristic. For instance, the bank has a new

applicant for a credit card with age of 45, debt ratio of 0.5 and monthly income of 5000 dollars. The credit score should be: 53 + 55 + 57 = 165.

To develop a more accurate scorecard, people usually have to consider more situations. For example, there are some individuals identified as “Bad” in the population but their application is approved, while there will be some “Good” persons that have been declined. Thus, reject inference is supposed to be involved in the development process. I don’t do this part because it requires the data set of rejected cases which I don’t have in my data. If you want to know more about this part, I highly recommend you to read *Credit Risk Scorecards — Developing and Implementing Intelligent Credit Scoring* written by Naeem Siddiqi (https://support.sas.com/content/dam/SAS/support/en/books/cedit-risk-scorecards/59376_excerpt.pdf).

This blog is posted by WeCloudData’s Data Science Immersive Bootcamp student Hongri Jia (Linkedin (https://www.linkedin.com/in/hongrijia/))

To see Hongri’s original blog post please click here (https://medium.com/henry-jia/how-to-score-your-credit-1c08dd73e2ed). To follow and see Hongri’s latest blog posts, please click here (https://medium.com/@hongri208).

To find out more about the courses our students have taken to complete these projects and what you can learn from WeCloudData, click here (https://weclouddata.com/learning-paths/) to see our upcoming course schedule.

Other blogs you might like

Kick start your career transformation

Explore Courses (/courses/)

WeCloudData is the leading data science and AI academy. Our blended learning courses have helped thousands of learners and many enterprises make successful leaps in their data journeys.

Sign up for newsletter

"*" indicates required fields

Email

Subscribe

Programs

Data Science(/learning-paths/data-science/)

Business Intelligence(/learning-paths/business-intelligence/)

Data Engineering(/learning-paths/data-engineering/)

ML Engineering(/learning-paths/ml-engineering/)

DevOps(/learning-paths/devops/)

Corporate Services

Corporate Training(/corporate-training/)

AI Consulting Services(/consulting-services)

Talent Partnership(/talent-partnership/)

Resources

Blogs(/blogs/)

Career Guides(/career-guides/)

WeCloudOpen(/wecloudopen)

Company

About Us(/about-us)

Join our team(/join-our-team/)

WeCareer(http://wecareer.ai/)

Learning Portal(https://learn.wecloudata.com/)

KPI(/kpi)

Terms & Conditions(https://wecloudata.com/terms-and-conditions/)

Contact Us(/contact-us/)

Lewes, DE 19958, USA

info@weclouddata.com (mailto:info@weclouddata.com)