Get Better Weight of Evidence for Scorecards Using a Genetic Algorithm

Xia Ke Shan, Peter Eberhardt, Matthew Kastin
Get Better Weight of Evidence for Scorecards Using a Genetic Algorithm

Xia Ke Shan, Peter Eberhardt, Matthew Kastin

ABSTRACT

• Scorecard is a very important risk management tool in credit card business. When a client starts to apply to a credit card in a bank, if his scorecard was lower than a cutoff score (e.g. 520), then the bank is going to reject his apply, vice versa.

• The WOE (a.k.a weight of evidence) plays the most important role in how to build a better scorecard to distinct the bad person and good person. This paper is trying to get better WOE by solving an optimization problem via genetic algorithm.

METHODS

• Use Genetic Algorithm to solve an optimal problem to make the continuous variable (e.g. age)'s WOE be monotonous increase or decrease, better is linear.

• Use Genetic Algorithm to solve an optimal problem to make the category variable's IV value be largest.

RESULTS

Here is the output for continuous variable. As you can see it is divided into four groups and its WOE is almost a line.
Get Better Weight of Evidence for Scorecards Using a Genetic Algorithm
Xia Ke Shan, Peter Eberhardt, Matthew Kastin

RESULTS CONTINUED
Here is the output for category variable. As you can see it is divided into six groups and its IV has maximal value 0.166.

<table>
<thead>
<tr>
<th>group</th>
<th>total number</th>
<th>bad number</th>
<th>good number</th>
<th>bad percent</th>
<th>good percent</th>
<th>woe</th>
<th>iv</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>84</td>
<td>40</td>
<td>44</td>
<td>0.48</td>
<td>0.116688</td>
<td>-0.51023</td>
<td>0.168117</td>
</tr>
<tr>
<td>5</td>
<td>234</td>
<td>145</td>
<td>89</td>
<td>0.267143</td>
<td>0.206887</td>
<td>-0.3502</td>
<td>0.168117</td>
</tr>
<tr>
<td>2</td>
<td>97</td>
<td>33</td>
<td>64</td>
<td>0.339535</td>
<td>0.113333</td>
<td>-0.23302</td>
<td>0.100117</td>
</tr>
<tr>
<td>0</td>
<td>193</td>
<td>131</td>
<td>62</td>
<td>0.367143</td>
<td>0.206887</td>
<td>-0.09923</td>
<td>0.100117</td>
</tr>
<tr>
<td>1</td>
<td>280</td>
<td>218</td>
<td>62</td>
<td>0.311429</td>
<td>0.206887</td>
<td>0.10033</td>
<td>0.168117</td>
</tr>
<tr>
<td>4</td>
<td>112</td>
<td>64</td>
<td>48</td>
<td>0.364286</td>
<td>0.06</td>
<td>0.505825</td>
<td>0.168117</td>
</tr>
</tbody>
</table>

CONCLUSIONS
We can get better WOE of credit scorecard by solving an optimal problem by Genetic Algorithm. This can get us better and powerful scorecard to control more risk and reduce the probability of default.

REFERENCES
SAS® GLOBAL FORUM 2018

April 8 - 11 | Denver, CO
Colorado Convention Center

Xia Ke Shan
iFRE Inc. Beijing, China
Phone: +8613521225927
E-mail: 12135835@qq.com

Peter Eberhardt
Fernwood Consulting Group Inc.
Toronto, Canada
E-mail: peter@fernwood.ca

Matthew Kastin
NORC at the University of Chicago
Hummelstown, Pennsylvania
E-mail: fried.egg@VERIZON.NET

#SASGF
Get Better WOE of Scorecard by Genetic Algorithm

Xia Ke Shan, iFRE Inc., Beijing, China
Peter Eberhardt, Fernwood Consulting Group Inc., Toronto, Canada
Matthew Kastin, NORC at the University of Chicago, Hummelstown, Pennsylvania

ABSTRACT

Scorecard is a very important risk management tool in credit card business. When a client start to apply to a credit card in a bank, if his scorecard was lower than a cutoff score (e.g. 520), then the bank is going to reject his apply, vice versa. The WOE (a.k.a weight of evidence) plays the most important role in how to build a better scorecard to distinct the bad person and good person. This paper is trying to get better WOE by solving an optimization problem via genetic algorithm.

INTRODUCTION

What is the WOE? What does the better WOE look like?

WOE measures the strength of grouped attributes of a variable in separating good and bad accounts. And it represents predictive power of each attribute. Here is an example to demonstrate what WOE is (it is taken from the second reference book). Suppose there are two variables. One flags the bad and good person, it has two values: bad and good. Another is age.

<table>
<thead>
<tr>
<th>Age</th>
<th>Count</th>
<th>Tot Distr</th>
<th>Goods</th>
<th>Distr Good</th>
<th>Bads</th>
<th>Distr Bad</th>
<th>Bad Rate</th>
<th>WOE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Missing</td>
<td>1,000</td>
<td>2.50%</td>
<td>860</td>
<td>2.38%</td>
<td>140</td>
<td>3.65%</td>
<td>14.00%</td>
<td>-42.719</td>
</tr>
<tr>
<td>18–22</td>
<td>4,000</td>
<td>10.00%</td>
<td>3,040</td>
<td>8.41%</td>
<td>960</td>
<td>25.00%</td>
<td>24.00%</td>
<td>-108.980</td>
</tr>
<tr>
<td>23–26</td>
<td>6,000</td>
<td>15.00%</td>
<td>4,920</td>
<td>13.61%</td>
<td>1,080</td>
<td>28.13%</td>
<td>18.00%</td>
<td>-72.613</td>
</tr>
<tr>
<td>27–29</td>
<td>9,000</td>
<td>22.50%</td>
<td>8,100</td>
<td>22.40%</td>
<td>900</td>
<td>23.44%</td>
<td>10.00%</td>
<td>-4,526</td>
</tr>
<tr>
<td>30–35</td>
<td>10,000</td>
<td>25.00%</td>
<td>9,500</td>
<td>26.27%</td>
<td>500</td>
<td>13.02%</td>
<td>5.00%</td>
<td>70.196</td>
</tr>
<tr>
<td>35–44</td>
<td>7,000</td>
<td>17.50%</td>
<td>6,800</td>
<td>18.81%</td>
<td>200</td>
<td>5.21%</td>
<td>2.86%</td>
<td>128.388</td>
</tr>
<tr>
<td>44+</td>
<td>3,000</td>
<td>7.50%</td>
<td>2,940</td>
<td>8.13%</td>
<td>60</td>
<td>1.56%</td>
<td>2.00%</td>
<td>164.934</td>
</tr>
<tr>
<td>Total</td>
<td>40,000</td>
<td>100%</td>
<td>36,160</td>
<td>100%</td>
<td>3,840</td>
<td>100%</td>
<td>9.60%</td>
<td></td>
</tr>
</tbody>
</table>

Here, age is divided into six groups. Column Goods stand for the good number in a group. For example, group 18-22 has 3040 good person. The same logic to column Bads, group 18-22 has 960 bad person. Column DistrGood=the good number in a group / all the good number. For example, for group 18-22, DistrGood= 3040/36160=8.41%. The same logic to column DistrBad. After that WOE=ln(DistrGood/DistrBad). For example, for group 18-22, WOE=ln(8.41%/25%)=-108.9.

What does the better WOE look like?

1) The difference between the WOE of groups should be as larger as it could be. The larger the difference between adjacent groups, the higher the predictive ability of this variable. And score would be most different from each group.

2) Column DistrGood and DistrBad generally should be greater than 5%. If it was too small, then there are not enough records to reflect the underlying rule. difference between adjacent groups, the higher the predictive ability of this variable. And score would be most different from each group.

3) For the continuous variable (e.g. age), WOE should be monotonous increase or decrease, better is linear.
4) For the category variable (e.g. sex), there is no requirement for WOE to be monotonous increase or
decrease, but better make IV(Information Value) is as bigger as it could be. IV measures the strength
of correlation between the variable and Y (odds for good or bad).

\[ IV = \sum (\text{DistrGoodi} - \text{DistrBadi}) \times \ln(\text{DistrGoodi} / \text{DistrBadi}) \]

5) Make as many groups as you can, more groups can keep more details of original variable, and make
the grouped variable has higher IV.

EXAMPLE

Now take the famous German Credit Card data as an example. The data can be downloaded from the url
https://onlinecourses.science.psu.edu/stat857/sites/onlinecourses.science.psu.edu.stat857/files/german_credit.csv

For the continuous variable, since its WOE must be monotonous increase or decrease, so I fit a linear
regression model, take WOE as x variable, group number (1 2 3 4 ...) as y variable, use genetic algorithm
to make the model's square sum of residual minimum. Once square sum of residual is near zero, I can
say WOE is monotonous increase or decrease. The following code is for the continuous variable, take
variable 'Credit Amount' as an example.

Note: if the continuous variable was a ratio which is between 0 and 1, then need to multiply 100 or 1000
to let this code work.

/* Download data from
https://onlinecourses.science.psu.edu/stat857/sites/onlinecourses.science.psu.edu.stat857/files/german_credit.csv *
*/

options validvarname=v7;
filename x '/folders/myfolders/german_credit.csv';
proc import datafile=x out=have dbms=csv replace;
run;

%let var=Credit_Amount;
%let group=4;
%let n_iter=100;
data temp;
set have;
good_bad=ifc(Creditability=1,'bad ','good');
keep &var good_bad ;
run;

proc sql noprint;
select sum(good_bad='bad'),sum(good_bad='good'),
floor(min(&var)),ceil(max(&var)) into : n_bad,: n_good,: min,: max
from temp;
quit;
%put &n_bad &n_good &min &max;
proc sort data=temp;by &var ;run;
proc iml;
use temp(where=(&var is not missing));
read all var {&var good_bad};
close;
start function(x) global(bin,&var ,good_bad,group,woe);
if countunique(x)=group-1 then do;
col_x=t(x);
call sort(col_x,1);
cutpoints= .M//col_x//.I ;
b=bin(&var ,cutpoints,'right');
if countunique(b)=group then do;
do i=1 to group;
idx=loc(b=i);
temp=good_bad[idx];
n_bad=sum(temp='bad');
n_good=sum(temp='good');
bad_dist=n_bad/&n_bad ;
good_dist=n_good/&n_good ;
if Bad_Dist>0.05 & Good_Dist>0.05 then woe[i]=log(Bad_Dist/Good_Dist);
else woe[i]=.;
end;
if countmiss(woe)=0 then do;
xx=j(group,1,1)||woe;
beta=solve(xx`*xx,xx`*bin);
yhat=xx*beta;
sse=ssq(bin-yhat);
end;
else sse=999999;
end;
else sse=999999;
end;
else sse=999999;
return (sse);
finish;
group=&group ;
bin=t(1:group);
woe=j(group,1,.);
encoding=j(2,group-1,&min );
encoding[2,]=&max ;
id=gasetup(2,group-1,123456789);
call gasetobj(id,0,"function");
call gasetsel(id,10,1,1);
call gainit(id,1000,encoding);
niter = &n_iter;
do i = 1 to niter;
call garegen(id);
call gagetval(value, id);
end;
call gagetmem(mem, value, id, 1);
col_mem=t(mem);
call sort(col_mem,1);
cutpoints= .M//col_mem//.I ;
b=bin(&var ,cutpoints,'right');
create cutpoints var {cutpoints};
append;
close;
create group var {b};
append;
/* Missing value handling */

close;
print value[l = "Min Value:"] ;
call gaend(id);
quit;

data all_group;
  set temp(keep=&var rename=(&var=b) where=(b is missing)) group;
run;
data all;
  merge all_group temp;
  rename b=group;
run;

proc sql;
create table woe_&var as
  select group label=' ',count(*) as total label='total number',
  min(&var) as min value',max(&var) as max label='max value',
  sum(good_bad='bad') as n_bad label='bad number',sum(good_bad='good') as
  n_good label='good number',
  sum(good_bad='bad')/(select sum(good_bad='bad') from all ) as bad_dist
  label='bad percent',
  sum(good_bad='good')/(select sum(good_bad='good') from all ) as good_dist
  label='good percent',
  log(calculated Bad_Dist/calculated Good_Dist) as woe
  from all
  group by group
  order by woe;
select *,sum( (Bad_Dist-Good_Dist)*woe ) as iv
  from woe_&var ;
quit;

proc sgplot data=woe_&var noautolegend;
vbar group/response=woe nostatlabel missing;
vline group/response=woe nostatlabel missing markers
    MARKERATTRS=(symbol=circlefilled size=12)
    MARKERFILLATTRS=(color=white)
    MARKEROUTLINEATTRS=graphdata1
    FILLEDOUTLINEDMARKERS;
run;
ods select fitplot;
proc reg data=woe_&var;
   model group=woe/ cli clm ;
quit;
<table>
<thead>
<tr>
<th>group</th>
<th>total number</th>
<th>min value</th>
<th>max value</th>
<th>bad number</th>
<th>good number</th>
<th>bad percent</th>
<th>good percent</th>
<th>woe</th>
<th>iv</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>102</td>
<td>7106</td>
<td>18424</td>
<td>54</td>
<td>46</td>
<td>0.077143</td>
<td>0.10</td>
<td>-0.72951</td>
<td>0.382445</td>
</tr>
<tr>
<td>3</td>
<td>60</td>
<td>5771</td>
<td>7127</td>
<td>37</td>
<td>23</td>
<td>0.052867</td>
<td>0.076667</td>
<td>-0.37187</td>
<td>0.382445</td>
</tr>
<tr>
<td>2</td>
<td>139</td>
<td>3590</td>
<td>5743</td>
<td>96</td>
<td>43</td>
<td>0.137143</td>
<td>0.143333</td>
<td>-0.04415</td>
<td>0.382445</td>
</tr>
<tr>
<td>1</td>
<td>669</td>
<td>250</td>
<td>3576</td>
<td>513</td>
<td>186</td>
<td>0.732657</td>
<td>0.62</td>
<td>0.167231</td>
<td>0.382445</td>
</tr>
</tbody>
</table>

**Min Value:**
0.0820377

---

![Graph showing WOE and IV values for different groups.](image-url)
Output 1. Output from Code

The minimal value is near zero. And from the graph, there is apparently monotonous decrease relationship.

For the category variable, there is no requirement for WOE to be monotonous increase or decrease, but should get better IV after grouping variable. So I am solving an optimization problem (i.e. Make the IV largest) by genetic algorithm. The following code is for the category variable, take variable ‘Purpose’ as an example.

```/* Download data from
https://onlinecourses.science.psu.edu/stat857/sites/onlinecourses.science.psu.edu.stat857/files/german_credit.csv */
```

```options validvarname=v7;
filename x '/folders/myfolders/german_credit.csv';
proc import datafile=x out=have dbms=csv replace;
run;```
%let var=Purpose;
%let group=6;
%let n_iter=100;
data temp;
   set have;
   good_bad=ifc(Creditability=1,'bad ','good');
   keep &var good_bad;
run;

proc sql noprint;
   select sum(good_bad='bad'),sum(good_bad='good') into : n_bad,: n_good
   from temp;
   create index &var on temp;
quit;
%put &n_bad &n_good;
proc iml;
   use temp;
   read all var {&var good_bad};
   close;
   start function(x) global(bin,&var ,good_bad,group,level,woe);
   if countunique(x)=group then do;
      do i=1 to group;
         idx=loc(x=i);
         levels=level[idx];
         temp=loc(element(&var,levels));
         n_bad=sum(temp='bad');
         n_good=sum(temp='good');
         bad_dist=n_bad/&n_bad;
         good_dist=n_good/&n_good;
         if Bad_Dist>0.05 & Good_Dist>0.05 then woe[i]=(Bad_Dist-
            Good_Dist)#log(Bad_Dist/Good_Dist);
      else woe[i]=.;
      end;
      if countmiss(woe)=0 then iv=sum(woe); 
   else iv=-999999;
   end;
   return (iv);
   finish;
   group=&group;
   bin=t(1:group);
   woe=j(group,1,.);
   level=unique(&var);
   n_level=countunique(&var);
   encoding=j(2,n_level,1);
   encoding[2,:]=group;
   id=gasetup(2,n_level,123456789);
   call gasetobj(id,1,"function");
   call gasetsel(id,10,1,1);
   call gainit(id,1000,encoding);
   niter = &n_iter;
   do i = 1 to niter;
call garegen(id);
call gagetval(value, id);
end;
call gagetmem(mem, value, id, 1);
col_mem=t(mem);
create group var{col_mem level};
append;
print value[l = "IV Max Value:"] ;
call gaend(id);
quit;
data all;
merge temp group(rename=(level=&var col_mem=group)) ;
by &var;
run;

proc sql;
create table woe_&var as
  select group,count(*) as total label='total number',
  sum(good_bad='bad') as n_bad label='bad number',sum(good_bad='good') as n_good label='good number',
  sum(good_bad='bad')/(select sum(good_bad='bad') from all ) as bad_dist label='bad percent',
  sum(good_bad='good')/(select sum(good_bad='good') from all ) as good_dist label='good percent',
  log(calculated Bad_Dist/calculated Good_Dist) as woe
  from all
  group by group
  order by woe;
create index group on woe_&var;
create index col_mem on group;
select *,sum( (Bad_Dist-Good_Dist)*woe ) as iv
  from woe_&var ;
quit;

\begin{tabular}{|c|c|c|c|c|c|c|c|}
\hline
  group & total number & bad number & good number & bad percent & good percent & wce & iv \\
\hline
  3 & 84 & 49 & 35 & 0.07 & 0.16667 & -0.51063 & 0.166117 \\
  5 & 234 & 145 & 89 & 0.207143 & 0.26667 & -0.3592 & 0.166117 \\
  2 & 97 & 53 & 34 & 0.09 & 0.113333 & -0.23052 & 0.166117 \\
  5 & 193 & 131 & 62 & 0.187143 & 0.20667 & -0.09923 & 0.166117 \\
  1 & 280 & 218 & 62 & 0.311429 & 0.20667 & 0.410063 & 0.166117 \\
  4 & 112 & 94 & 38 & 0.134286 & 0.06 & 0.805625 & 0.166117 \\
\hline
\end{tabular}

\textbf{Output 2. Output from Code}
REFERENCES

CONTACT INFORMATION
Your comments and questions are valued and encouraged. Contact the author at:

Xia Ke Shan
iFRE Inc.
Beijing, China
Phone: +8613521225927
E-mail: 12135835@qq.com

Matthew Kastin
NORC at the University of Chicago
Hummelstown, Pennsylvania
E-mail: fried.egg@VERIZON.NET

Peter Eberhardt
Fernwood Consulting Group Inc.
Toronto, Canada
E-mail: peter@fernwood.ca

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.