3771-2019

Evaluating and Improving the Efficiency of Inference Rules for Segmentation in SAS

Alec Zhixiao Lin, Southern California Edison, Rosemead, CA

Abstract

Inference rules are a category of business rules expressed in a series of if-then statements. They are often used jointly with a modeling score for segmentation in many business decisions such as marketing, underwriting, pricing, attribution control, etc. Each inference rule identifies a sweet or sour spot in a population whose performance deviates greatly from the population average but whom a regression- or tree-based predictive model often fails to capture. A mature set of inference rules is a good compromise between maximizing business outcome and minimizing loss in business volume. Using decline rules in mortgage underwriting as an example, this paper introduces a step-wise process in SAS to evaluate and improve the efficiency of inference rules. The final product is a series of neural-network-look-like rules expressed in a white-box manner that are easily communicable and implementable.

Introduction

Inference rules are an important category of business rules expressed in a series of if-then statements used in parallel. The following is an example of decline rules used in mortgage underwriting:

```
if FICO < 680 and payment_to_balance_ratio < 0.3 then decline=1;
if total_pastdue_in_12_months > 0 then decline=1;
if delinquency_ever ge 3 and avg_month_on_file < 7 then decline=1;
if age_youngest_account < 3 and number_opened_loans_12months > 4 then decline=1;
```

If an applicant meets any of the above conditions, the application will be declined.

Inference rules are often used together with modeling score to improve risk control for underwriting. Both rules can be easily explained to meet the need for compliance review and to be cited as decline reasons, but each has its own advantage:

- A model score based on a regression model provides more flexibility in adjusting the threshold to meet changes in risk appetite. For example, we can lower the `risk_score` in the 4th rule bit to accommodate the need for expanding the origination volume at the price of increasing the overall risk of the new volume.
- Each inference rule identifies a segment of applicants that exhibit a much higher risk than the population average. These segments might not carry enough statistical influence to be captured by a regression model as a dummy variables. Many times a variable used in each inference rule does not show a monotonic or linear pattern to be used in a regression. Therefore, inference rules functions as an additional "safety net" for risk management.

For loan underwriting, a set of decline rules is ideally an optimal balance between risk reduction and portfolio maximization. How to find such a balance is a joint consideration of business needs and data analysis. This paper provides a process in SAS to create a "compact" set of inference rules in an attempt to arrive at such a balance. The process contains three consecutive steps as follows:

Step 1: Each rule should exhibit much higher business outcome than the population average and is evaluated in relation to the target outcome.

Step 2: Rules should have low overlaps with each other. Two rules are evaluated jointly. If they show a big overlap, a weaker rule will be deleted.

Step 3: Each rule should have a low overlap with all other rules combined. If the results of one rule has been largely captured by all other rules combined, the former will be discarded from the finalized set of inference rules.

We can start as many rules as we like in Step 1, but they are usually reduced to 4-7 rules. The reduced set of rules not only simplifies subsequent deployment, but also prevents potential valuable applicants from being denied of worthy opportunities and hence increase the volume.

Step 1 – Evaluating each rule separately

This step functions like a brain-storming session. We draw various insights from expert knowledge, data analysis and machine learning to assemble a set of rules. Each rule should be named separately as follows:

```
if FICO < 680 and payment_to_balance_ratio < 0.3 then decline_rule_1=1;
if FICO < 710 and total_debt_to_income > 0.57 then decline_rule_2=1;
if combined_loan_to_value < 0.85 and number_open_loans > 5 then decline_rule_3=1;
……
if total pastdue in 12 months > 0 then decline rule 25=1;
if inquiries_in_12months > 10 inquiries_in_3months_over_12months > 0.7  then decline_rule_26=1;
if delinquency_ever ge 3 and avg_month_on_file < 7 then decline_rule_27=1;
if risk_score ge 0.0775 then decline_rule_28=1;
```

A bivariate analysis is run to one by one in order to screen out some rules that do not meet our expectations. These rules usually exhibit business outcomes that are not so much distinctly different enough from the population average. Starting with 28 decline rules, we apply the following SAS code:

```
%let rulecnt=28;
** evaluate each decline rule by itself;
proc sql noprint; select count(*) into :total_cnt from cs.mortgage_data2; quit;

%macro checkrule;
%do i=1 %to &rulecnt;
proc sql;
create table check rule &i as select
put(&i, 4.) as decline_rule_num,
avg(bad) as avg_bad,
count(*) as record_count,
count(*)/&total_cnt as record_pct
from cs.mortgage_data2
where decline_rule_&i=1; quit;
%end;
%mend;
%checkrule;

** check all decline rules combined;
data cs.mortgage_data2; set cs.mortgage_data2;
if sum(of decline_rule_1-decline_rule_&rulecnt) > 0 then decline_rule=1;
run;

proc sql;
create table check_rule_100 as select
"all_rules_combined" as decline_rule_num,
avg(bad) as avg_bad,
count(*) as record_count,
count(*)/&total_cnt as record_pct
from cs.mortgage_data2
where decline_rule=1; quit;

** get population average;
proc sql;
create table check_rule_1000 as select
"population average" as decline_rule_num,
avg(bad) as avg_bad,
count(*) as record_count,
1 as record_pct
from cs.mortgage_data2; quit;

** puts results together;
data all_rules_results;
retain decline_rule_num avg_bad record_count record_pct;
format decline_rule_num $20.;
format record pct percent7.2;
set check_rule_:; run;
proc sort data=all_rules_results; by decline_rule_num; run;
proc print data=all_rules_results noobs; run;
```

SAS gives the following output:

decline_rule_num	avg_bad	record_count	record_pct
1	0.024013	4,456	2.82%
2	0.010234	8,734	5.53%
3	0.033301	1,021	0.65%
4	0.026223	5,110	3.23%
5	0.027433	2,661	1.68%
6	0.024859	2,655	1.68%
7	0.029301	1,058	0.67%
8	0.028969	863	0.55%
.....	.....	.....	.....
25	0.00823	16,346	10.34%
26	0.015342	3,424	2.17%
27	0.00734	1,256	0.79%
28	0.0243	2,435	1.54%
All Rules Combined	0.00626	65,056	41.17%
Population Average	0.004745992	158,028	100.00%

This decline rule does not perform well:
- Bad rate is not significantly higher than population average
- It will exclude too many records.

Table 1 – SAS Output for Step 1

Please note that for record_count for "All Rules Combined" is not the additive sum of individual rules because of overlaps of records between rules.

The output shows several problems:

- It is clear that decline_rule_2, decline_rule_25 and decline_rule_26 do not meet our expectation because their performances are considerably weaker than other rules.

- It is possible that decline_rule_2 and decline_rule_25 will exclude a lot of potentially valuable applications.

- Even though applications flagged for decline show almost twice as high in risk, a further analysis shows that 54% of risk can be avoided by declining 41% of applications. This treatment by decline rules are too broad-brushed and less than efficient.

decline_rule	avg_bad	record_count	record_pct	# Bad accts captured	% Bad Accts Captured
1	0.00626	65,056	41.2%	407	54.2%
0	0.00370	92,972	58.8%	344	45.8%
Total	0.00475	158,028	100.0%	751	100.0%

Table 2 – Risk control by decline rules

For removing three rules mentioned above, we apply the following changes to the SAS code:

- The retained 25 decline rules should be numbers from 1 to 25 as the following:

```
if FICO < 680 and payment_to_balance_ratio < 0.3 then decline_rule_1=1;
** if FICO < 710 and total_debt_to_income > 0.57 then decline_rule_2=1;
if combined_loan_to_value < 0.85 and number_open_loans > 5 then decline_rule_2=1;
……
** if total_pastdue_in_12_months > 0 then decline_rule_25=1;
** if inquiries 12months > 10 inquiries 3months over 12months > 0.7  then decline_rule_26=1;
if delinquency_ever ge 3 and avg_month_on_file < 7 then decline_rule_24=1;
if risk_score ge 0.0775 then decline_rule_25=1;
```

- Change the macro value `%let rulecnt=28` to `%let rulecnt=25.`

After similar output have been generated, we review the results once again. Step 1 usually takes multiple rounds of such iteration to arrive at a relatively satisfactory set of decline rules, with each rule exhibiting much better performance than the population average. The following output shows some good improvement in efficiency:

decline_rule	avg_bad	record_count	record_pct	# Bad accts captured	% Bad Accts Captured
1	0.00956	40,020	25.3%	383	50.9%
0	0.00313	118,008	74.7%	369	49.1%
Total	0.00475	158,028	100.0%	751	100.0%

Table 3 – Improvement in Inference Rules

Now by excluding 25.3% of applicants, we can reduce the risk by 50.9%. The declined applicants show a potential risk three times as high as approved applications. However, this is not the end of the process yet due to the concern that some rules could have a large overlap with one another. The overlap not only suggests redundancy in rules but could also exclude potentially valuable customers. Step 2 addresses to this concern.

Step 2 - Minimizing overlap between two rules

In Step 2, we cross tabulate each pair of decline rules to examine their overlap. If 15 declines are inherited from Step 1, we will run 105 pairs of comparison (15x14/2=105) by applying the following SAS code.

```
** number of rules inherited from Step 1;_
%let maxrule=15;

%macro drheloc;
if FICO < 680 and payment_to_balance_ratio < 0.3 then decline_rule_1=1;
** if FICO < 710 and total_debt_to_income > 0.57 then decline_rule_2=1;
if combined_loan_to_value < 0.85 and number_open_loans > 5 then decline_rule_2=1;
……
** if total_pastdue_in_12_months > 0 then decline_rule_25=1;
** if inquiries 12months > 10 inquiries 3months over 12months > 0.7  then decline_rule_26=1;
if delinquency_ever ge 3 and avg_month_on_file < 7 then decline_rule_24=1;
if risk_score ge 0.0775 then decline_rule_25=1;


if sum(of decline_rule_1-decline_rule_&maxrule) > 0 then decline_rule=1; else decline_rule=0;

if decline_rule_1=1 then do;
   if sum(of decline_rule_2-decline_rule_&maxrule) ge 1 then decline_rule_otherfor_1=1;
end;

if decline_rule_2=1 then do;
   if sum(decline_rule_1,  sum(of decline_rule_3-decline_rule_&maxrule)) ge 1 then
decline_rule_otherfor_2=1;
end;

%macro loopsum;
%do i=3 %to %eval(&maxrule-2);

if decline_rule_&i=1 then do;
   if sum(sum(of decline_rule_1-decline_rule_%eval(&i-1)),  sum(of decline_rule_%eval(&i+1)-
decline_rule_&maxrule)) ge 1 then decline_rule_otherfor_&i=1;
end;
%end;
%mend;
%loopsum;

if decline_rule_%eval(&maxrule-1)=1 then do;
   if sum(sum(of decline_rule_1-decline_rule_%eval(&maxrule-2)),  decline_rule_&maxrule) ge 1
then decline_rule_otherfor_%eval(&maxrule-1)=1;
end;

if decline_rule_&maxrule=1 then do;
   if sum(of decline_rule_1-decline_rule_%eval(&maxrule-1)) ge 1 then
decline_rule_otherfor_&maxrule=1;
end;

bad=CO_ind;
%mend;
```

```
data dr.check_dr; set dr.decline_rule_cleaned;
%drheloc;
run;


%let inset=dr.check_dr;

%macro chkrule;
%do i=1 %to &maxrule;
proc sql;
create table checkbad&i as select
&i as decline_rule,
avg(bad) as avg_bad,
count(*) as decline_record_cnt
from &inset
where decline_rule_&i=1; quit;

data for_confusion_matrix;
set &inset;

if decline_rule_&i=1 and bad=1 then true_positive=1;
if decline_rule_&i=1 and bad=0 then false_positive=1;
if decline_rule_&i=. and bad=1 then false_negative=1;
if decline_rule_&i=. and bad=0 then true_negative=1;
run;

proc means data=for_confusion_matrix sum nway noprint;
var true_positive false_positive false_negative true_negative;
output out=confusion_matrix_&i(drop=_type_ _freq_)
       sum=true_positive false_positive false_negative true_negative; run;

data confusion_matrix_&i;
set confusion_matrix_&i;
decline_rule=&i;
sensitivity=true_positive/sum(true_positive, false_negative);
specificity=true_negative/sum(true_negative, false_positive);
positive_predictive_value=true_positive/sum(true_positive, false_positive);
run;
%end;
%mend;

%chkrule;
```

The SAS output is a series of 2×2 cross-tabulations for us to examine their overlaps.

In the following example, we see a very limited overlap between decline_rule_2 and decline_rule_4. Although the overlapped area show a bad rate more than twice as high, it contains for too few records (594 out of 6523)[1].

	decline_rule_4		decline_rule_4		All
			0	1	
	0	1	bad	bad	bad
	N	N	Mean	Mean	Mean
decline_rule_1					
0	.	2067	.	0.0218	0.0218
1	3862	594	0.0205	0.0471	0.024
All	3862	2661	0.0205	0.0274	0.0233

Table 4 – Lack of overlap between two rules

The following is an example of using overlap to remove one of the rules. Initially decline_rule_2 shows a much higher bad rate at 0.04955 than decline_rule_7 at 0.029. However, the cross tabulation reveals that much of the risk for the former has been absorbed by the latter. In fact, those applicants originally declined by decline_rule_2 only but not by decline_rule_7 show a much lower risk even than the population average.

[1] If the overlapped areas absorbs a majority of the records from the two rules, we can potentially combine the two rules into one.

	decline_rule_7		decline_rule_7		All
			0	1	
	0	1	bad	bad	bad
	N	N	Mean	Mean	Mean
decline_rule_2					
0	.	799	.	0.0263	0.0263
1	234	854	0.0023	0.0625	0.049553
All	234	863	0.0023	0.029	0.023305

Table 4 – Use overlap to remove a rule

Therefore, removing decline_rule_2 not only will improve risk prediction, but also increase the origination volume because 234 potentially valuable customers can be added.

After repeating the same exercise above for multiple rounds, we can improve the compactness of rules even more as follows:

decline_rule	avg_bad	record_count	record_pct	# Bad accts captured	% Bad Accts Captured
1	0.01504	20,020	12.7%	301	40.1%
0	0.00326	138,008	87.3%	450	59.9%
Total	0.00475	158,028	100.0%	751	100.0%

Table 5 – Further improvement in Inference Rules

Now we can reduce risk by 44.5% when including only 15.8% of applicants. The declined applicants have a potential bad rate more than four times as high as the approved applicants. This is a clear improvement than what we got in table 3.

However, this is still not the end of the process. What if the risk contained in one of the decline rules has been mostly accounted for by all other decline rules combined? In this case, the former can be dropped to further improve the compactness of the rules. We will accomplish this in Step 3.

Step 3 - Minimizing overlap between one rule and all other rules combined

The SAS code for Step 3 can be directly appended to that for Step 2 as follows:

```
** one rule vs. other rules combined;
%macro chkvlay2;
%do i=1 %to &maxrule;

data check_crossrule;
set &inset;

if decline_rule_&i=1 or decline_rule_otherfor_&i=1;

if decline_rule_&i=. then decline_rule_&i=0;
if decline_rule_otherfor_&i=. then decline_rule_otherfor_&i=0;
run;

proc tabulate data=check_crossrule format=6.4;
class decline_rule_&i decline_rule_otherfor_&i;
var bad;
tables decline_rule_&i, decline_rule_otherfor_&i*n decline_rule_otherfor_&i*bad*mean; run;
%end;
%mend;
%chkvlay2;

proc tabulate data=dr.check_dr format=10.5;
class new_rule_decline decline_rule;
var bad;
tables (new_rule_decline all), decline_rule*n decline_rule*bad*mean; run;

** check on the validation sample;
```

```
data dr.check_dr_val;
set dr.heloc_dr_val;
%drheloc;

if sum(of decline_rule_1-decline_rule_8) > 0 then old_rule=1; else old_rule=0;
if decline_rule_9 > 0 then new_rule=1; else new_rule=0;
run;

proc means data=dr.check_dr_val;
class decline_rule;
var bad; run;
```

The following is an example in which one rule does not overlap much with all other rules combined because the bad rates are very similar. In this case, we will do nothing.

	decline_rule_otherfor_2		decline_rule_otherfor_2	
			0	1
	0	1	bad	bad
	N	N	Mean	Mean
decline_rule_2				
1	515	506	0.0311	0.0356

Table 6 – Lack of overlap between one rule and all other rules combined

In contrast, the following example shows that the risk contained in decline_rule_6 has been mostly accounted for by all other rules combined. In this case, decline_rule_6 can be dropped due to its redundancy in the family of all rules.

	decline_rule_otherfor_6		decline_rule_otherfor_6	
			0	1
	0	1	bad	bad
	N	N	Mean	Mean
decline_rule_6				
1	459	2459	0.0064	0.0371

Table 7 – Use overlap to remove one rule

After examining such overlaps between each rule and all other rules combined, we can possibly remove a few more rules to make the final set of inference rules even more compact and intense as follows:

decline_rule	avg_bad	record_cnt	true_positive	false_positive	false_negative	true_negative	sensitivity	specificity
1	0.024013	4456	107	4349	643	152929	0.14267	0.97235
2	0.033301	1021	34	987	716	156291	0.04533	0.99372
3	0.026223	5110	134	4976	616	152302	0.17867	0.96836
4	0.027433	2661	73	2588	677	154690	0.09733	0.98355
5	0.024859	2655	66	2589	684	154689	0.088	0.98354
6	0.029301	1058	31	1027	719	156251	0.04133	0.99347
7	0.028969	863	25	838	725	156440	0.03333	0.99467
All Rules Combined	0.021266	12508	266	12242	484	145036	0.35467	0.92216

Table 8 – Finalized set of inference rules

The above finalized rules are a mature set of inference rules that are ready for deployment for guide business decisions. The efficiency of the finalized set of inference rules is as follows:

decline_rule	avg_bad	record_count	record_pct	# Bad accts captured	% Bad Accts Captured
1	0.02127	12,508	7.9%	266	35.4%
0	0.00333	145,520	92.1%	485	64.6%
Total	0.00475	158,028	100.0%	751	100.0%

Table 9 – Efficiency of finalized inference rules

Compared to Table 2, we can now reduce 35.4% of the total risk by excluding only 7.9% of applicants.

Case Study: Underwriting for Home Equity Line of Credit (HELOC)

Our study also includes an out-of-time validation sample. As the loans in the validation sample has considerably shorter performance period, the bad rate is considerably lower. However, the performance power of the inference rules remains quite stable.

	Modeling Sample		Validation Sample		
Decline Rule	% Loans Declined	% Delinquecy	% Loans Declined	% Delinquecy	Rule Description
1	2.82%	2.40%	3.17%	0.63%	FICO < 710 and ratio of actual payment to minimum payment for credit cards ≤ 10
2	0.65%	3.33%	0.62%	1.40%	total pastdue for all trades in last 12 months > 0
3	3.23%	2.62%	3.29%	0.71%	LD proprietary risk-score > 0.018
4	1.68%	2.74%	1.34%	0.87%	FICO < 730 and inquiries in last 6 months => 3
5	1.68%	2.49%	1.59%	0.60%	# delq 30+ ever ≥ 3 and DTI ≥ 35
6	0.67%	2.93%	0.65%	1.00%	# delq 30+ ever ≥ 3 and avg months on file for all trades ≤ 7 years
7	0.55%	2.90%	0.71%	0.61%	# delq 30+ ever ≥ 3 and # open installment loans ≥ 3
Total	**7.92%**	**2.13%**	**8.00%**	**0.59%**	

Table 10 – finalized inference rules

We present the performance of decline rules in three views.

1) The declined loans have a delinquency rate 6-7 times as high as the pro forma loans. By excluding 8% of applicants, the decline rules can reduce risk by about 30%.

	Modeling Sample (loans originated in 2009-2013)		Validation Sample (loans originated in 2014-2015)	
	% Loans	% DQ	% Loans	% DQ
Approved	92.08%	0.33%	92.00%	0.08%
Declined	7.92%	2.13%	8.00%	0.59%
Total	100.00%	0.47%	100.00%	0.12%
Risk Reduced		29.9%		33.6%
		6.39		7.31

Table 11 – Results of modeling and validation samples

2) Decline rules will weed out loans of potentially high risk for FICO < 740.

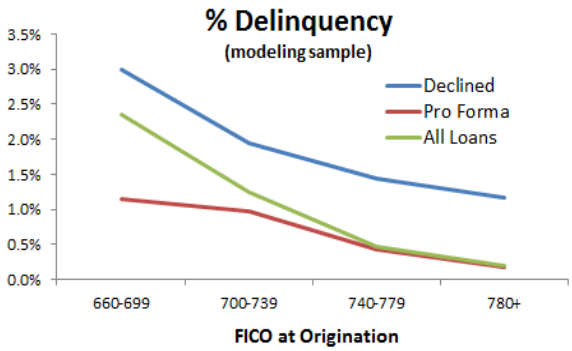

Figure 1 – Risk reduction (modeling sample)

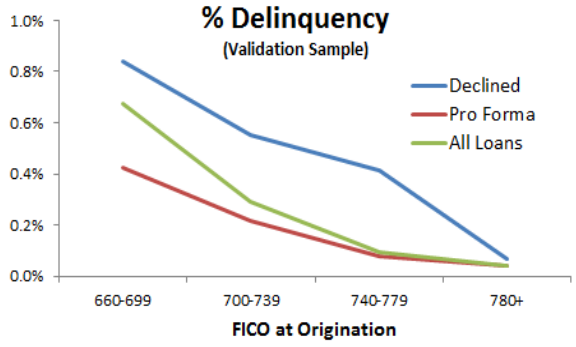

Figure 2 – Risk reduction (validation sample)

3) Decline rules perform consistently well in risk reduction on loans originated in different periods.

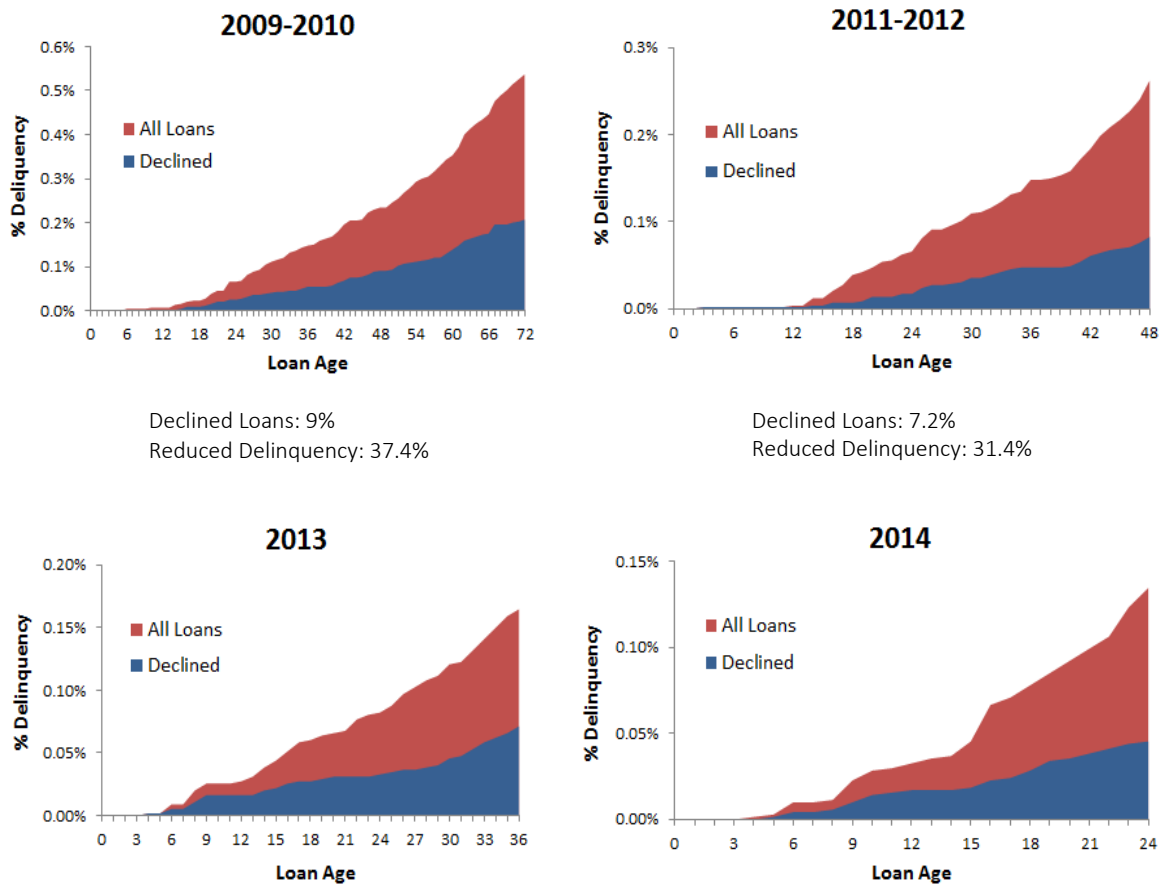

Figure 3 – Risk reduction by HELOC loans originated in different years

Conclusion

This paper uses mortgage underwriting as an example to introduce an approach to evaluate and improve the efficiency of inference rules. The same approach can also be applied to such business decisions as marketing, pricing, retention and collection to maximize intended business outcomes.

Contact

Alec Zhixiao Lin
alecindc@gmail.com
https://www.linkedin.com/in/alec-zhixiao-lin-26170825/