SAS® GLOBAL FORUM 2017

April 2 – 5 | Orlando, FL

#SASGF

USERS PROGRAM

Presenter

Paul Edwards, Senior Manager, Risk Models, Scotiabank

Paul Edwards is a senior manager on the Canadian Retail Models & Analytics team. Paul has worked in the financial sector since 2013 holding roles in risk modeling and fraud analytics. Paul has used SAS for 3 years.

Real AdaBoost

Boosting for Credit Scorecards and Similarity to WOE Logistic Regression

Objectives

- The need for transparency in models
- The desire for machine learning
- Consumer risk models
  - Scorecards
  - Weight-of-evidence (WOE) Regression
- Boosting
  - How it works
  - Highlights of boosting
  - How it is similar to WOE techniques
- Real AdaBoost macro
  - Example

Transparency

- Modeling has undergone a renaissance
  - New machine learning algorithms
  - Powerful computers
  - Data-driven decision making has lead to large profits[1]

- Modeling departments at Financial Institutions are at a crossroads
  - Executives want some of the famed value of advanced methods
  - Others want models that are easy to understand & use
    - Regulators & auditors
    - Front line staff
    - Implementation teams (IT)

1 https://hbr.org/2016/05/how-companies-are-using-machine-learning-to-get-faster-and-more-efficient

Consumer Risk Models

Introduction

- Risk modelers have developed methodology that is easy to implement and effective
  - The methodology is based on decision trees and regression
- Characteristics are binned and each bin receives a score proportional to risk

Characteristic	Bin	Score points
Past loan delinquency	No past loan delinquency	21
	One past loan delinquency event	5
	More than one past loan delinquency event	0
Credit utilization	Low credit utilization (<30%)	25
	Medium credit utilization (30-80%)	10
	High credit utilization (>80%)	0

Consumer Risk Models

Scorecards

- This makes the models easy to understand, communicate and implement
- An applicant falls into just one bin per characteristic
  - The applicants gets one score from each characteristic. Total score is summed
  - Applicant proceeds down scorecard summing up a final score

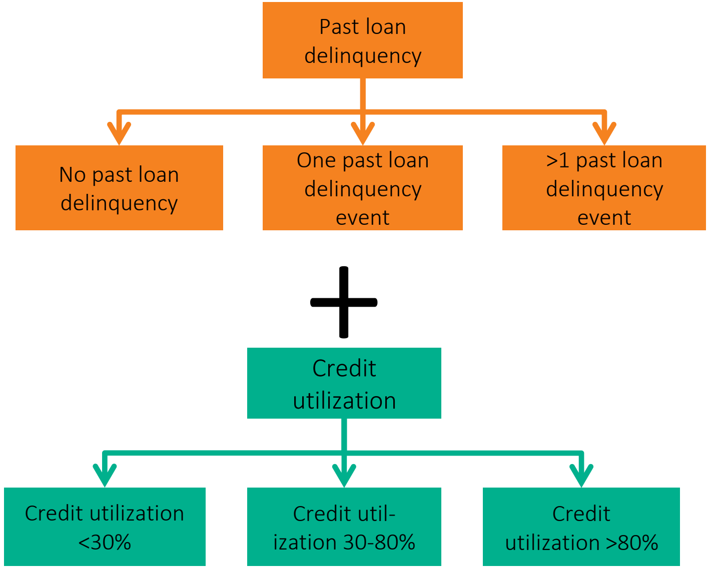
Characteristic	Bin	Score points
Past loan delinquency	No past loan delinquency	21
	One past loan delinquency event	5
	More than one past loan delinquency event	0
Credit utilization	Low credit utilization (<30%)	25
	Medium credit utilization (30-80%)	10
	High credit utilization (>80%)	0

Consumer Risk Models

Building Scorecards

- The bins for each characteristic are determined by a decision tree

Characteristic	Bin	Score points
Past loan delinquency	No past loan delinquency	21
	One past loan delinquency event	5
	More than one past loan delinquency event	0
Credit utilization	Low credit utilization (<30%)	25
	Medium credit utilization (30-80%)	10
	High credit utilization (>80%)	0



- The scorecard add the contributions from each tree

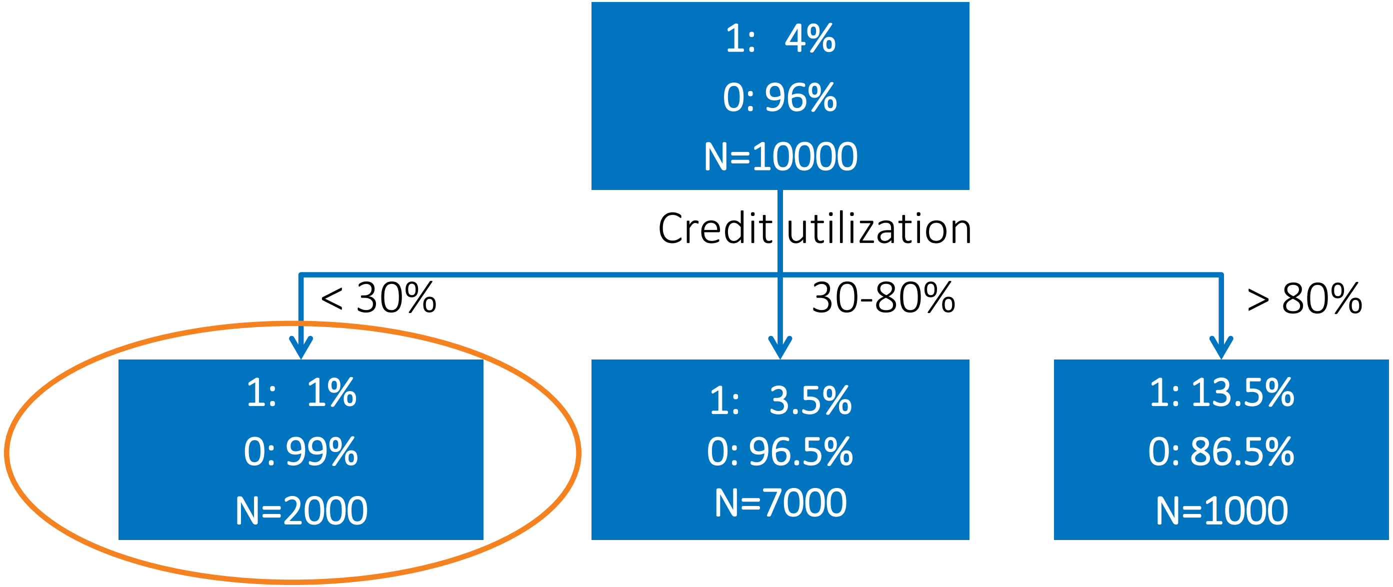
Building Trees for Scorecard

1. Gather (binary) training data
    - $Y \in \{0,1\}$: your target variable. In consumer risk, $Y = 1$ indicates an applicant will become delinquent
    - $\boldsymbol{x}: \{x_1, x_2, \dots, x_j\}$: predictor variables (characteristics; e.g. credit utilization)

Applicant	Y	x_1	x_2	…	x_j
111	0	0.1	A		.
112	1	0.9	A		1
113	0	0.0	B		6

Building Trees for Scorecard

2. Build a decision tree, splitting x_i into uniforms bins of Y
   - As an illustration, say x_1 is credit utilization



Building Trees for Scorecard

Weight-of-evidence

3. Standardize the avg(Y) in each bin using "weight-of-evidence" (WOE)

- WOE is measures the "purity" of Y in the bin. A bin with most Y=0 events has large value

Credit utilization <30%

1: 20 (1%)

0: 1980 (99%)

N: 2000

WOE: 0.61

General equations

$$F_{G,j}(k) = \frac{N_{j,k}^{Y=0}}{N_k^{Y=0}}$$

$$F_{B,j}(k) = \frac{N_{j,k}^{Y=1}}{N_k^{Y=1}}$$

$$\text{WOE}_{j,k} = \log\left(\frac{F_{G,j}(k)}{F_{B,j}(k)}\right)$$

For credit utilization bin 1

$$F_{G,1}(1) = \frac{1980}{9600}$$

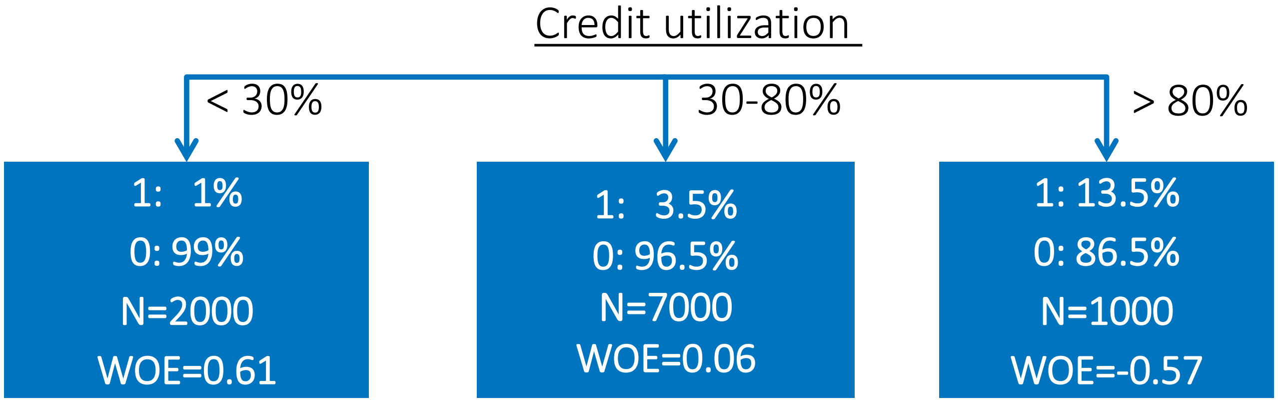
$$F_{B,1}(1) = \frac{20}{400}$$

$$\text{WOE}_{1,1} = \log\left(\frac{F_{G,1}(1)}{F_{B,1}(1)}\right) = 0.61$$

Building and Weighting Trees

Weight-of-evidence

- New function $W_j(x_j)$ - sorts characteristic *j* into appropriate bin and outputs the WOE value of that bin
- Examples
  - $W_1(x_1 = 40\%) = 0.06$
  - $W_1(x_1 = 85\%) = -0.57$
  - $W_1(x_1 = 90\%) = -0.57$



Weighting Trees

Logistic regression

- Logistic regression

$$\text{logit}\big(P(Y=1)\big) = \beta_0 + \sum_{j=1}^{M} \beta_j W_j(x_j)$$

- Recall $W_j(x_j)$ is a WOE tree: One term (one tree) per characteristic
- The β coefficients allow different contribution from each tree/characteristic
- Binning variables and standardizing with WOE allows
  - non-linear relationships to be modelled
  - categorical or missing data to be modelled naturally
- Non-linear version of logistic regression!

Link to Machine Learning

Weak learners

- The key to connecting WOE logistic regression with boosting methods is to understand that $W_j(x_j)$ is itself a predictive model of $P(Y = 1)$
  - A "weak learner" in ML parlance

Y	β_1	$W_1(x_1)$	x_1
?	0.55	**-0.57**	0.86
?	0.55	**0.61**	0.00
?	0.55	**0.61**	0.04

- A record with a negative WOE is more likely Y=1

Link to Machine Learning

Weak learners

- Our confidence grows as we add trees
- Record 1 looks even more likely to be Y=1

Y	β_1	$W_1(x_1)$	x_1	β_2	$W_2(x_2)$	x_2
?	0.55	**-0.57**	0.86	0.65	**-1.2**	5
?	0.55	**0.61**	0.00	0.65	**1.0**	1
?	0.55	**0.61**	0.04	0.65	**2.0**	0

Link to Machine Learning

Strong learner

- All three trees agree that the first record is Y=1
  - The probability P(Y=1) is proportional to $\beta_1 W_1(x_1)+\beta_2 W_2(x_2)+\beta_3 W_3(x_3)$

Y	β_1	$W_1(x_1)$	x_1	β_2	$W_2(x_2)$	x_2	β_3	$W_3(x_3)$	x_3
?	0.55	**-0.57**	0.86	0.65	**-1.2**	5	0.11	**-0.2**	5.5
?	0.55	**0.61**	0.00	0.65	**1.0**	1	0.11	**0.4**	-1.1
?	0.55	**0.61**	0.04	0.65	**2.0**	0	0.11	**0.4**	0.0

- Adding weak learners to form a strong one is a motivating principle in ML
  - This is possibly why WOE regression works

Real AdaBoost

- Real AdaBoost[1] add weak learner trees: $H_j(x_j)$ just like $W_j(x_j)$
- But Real AdaBoost builds trees stage wise,
    1. Build $H_1(x_1)$ (i.e., bin x_1 using a tree)
    2. Estimate residual $w = Y - H_1(x_1)$
    3. Build $H_2(x_2)$ weighted by residuals. Two (equivalent) ways to think about this:
        - Resample your training data, proportional to w, then build $H_2(x_2)$
        - The second tree tries hard to predict the difficult cases about which the previous tree was wrong
    4. Repeat
- H returns the weighted log odds of the bin, rather than the WOE of the bin

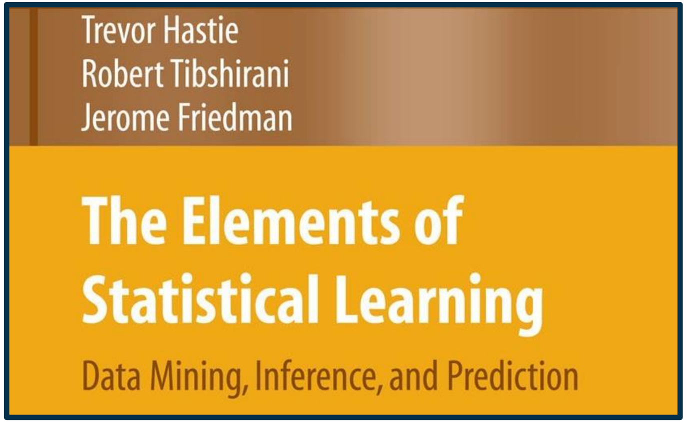
$$G(P(Y=1)) = \sum_{j=1}^{M} H_j(x_j); \qquad H_j(x_j) = \frac{1}{2} log\left(\frac{P_w(Y=1|x_j)}{P_w(Y=0|x_j)}\right)$$

1: Friedman, H. Hastie, T., and Tibshirani, R. 2000. "Additive logistic regression: a statistical view of boosting." The Annals of Statistics, 28(2):337-407.

Real AdaBoost

Highlights

- Adaptive binning “wrings out” any variance left in the model
  - SAS EM credit scoring add-on builds all WOE trees first, then does regression.
  - Minimizes multicolinearity & remove need for variable reduction
- Automatic, but modifiable
  - Real AdaBoost can automatically fit a model even automatically detecting variable interactions
  - A business partner may insist on a certain variable, which could be added at from of AdaBoost series
- Established technique
- No fitted Coefficients
  - No regression step. The authors prove that a β=1 coefficient will always minimizes error
- Scorecards
  - A Real AdaBoost model is a sum of a series of trees. The model can be expressed as a scorecard
- Extensible
  - Boosting (though not Real AdaBoost) can be done on non-binary targets



They wrote the book on machine learning!

Real AdaBoost

Macro

- A brief example of macro usage (synthetic data)

Original input data						
ID	COL1	COL2	COL3	COL4	COL5	DF
1	1.241	1.617	-0.808	-1.286	-2.463	0
2	-0.535	1.200	-0.969	-2.597	2.085	1
3	-1.014	0.356	1.063	0.444	-0.006	1
4	0.690	-0.357	0.708	-0.605	0.821	0

```
%adaboost(data=fakepd_t, target=df, var=col1 col2 col3 col4 col5,
          scoreme=fakepd_v fakepd_o,seed=1234, ntree=10, interaction=0,
          treedepth=2, outada=outadaboost);
```

Real AdaBoost

Macro outputs

The scored data set

Original input data							New columns							
ID	COL1	COL2	COL3	COL4	COL5	DF	f1	…	f10	adascore	p_df1	p_df0	ada-predict_df
1	1.241	1.617	-0.808	-1.286	-2.463	0	0.143		-0.085	0.350	0.587	0.413	1
2	-0.535	1.200	-0.969	-2.597	2.085	1	0.143		0.038	0.495	0.621	0.379	1
3	-1.014	0.356	1.063	0.444	-0.006	1	0.024		0.038	0.431	0.606	0.394	1

Scorecard

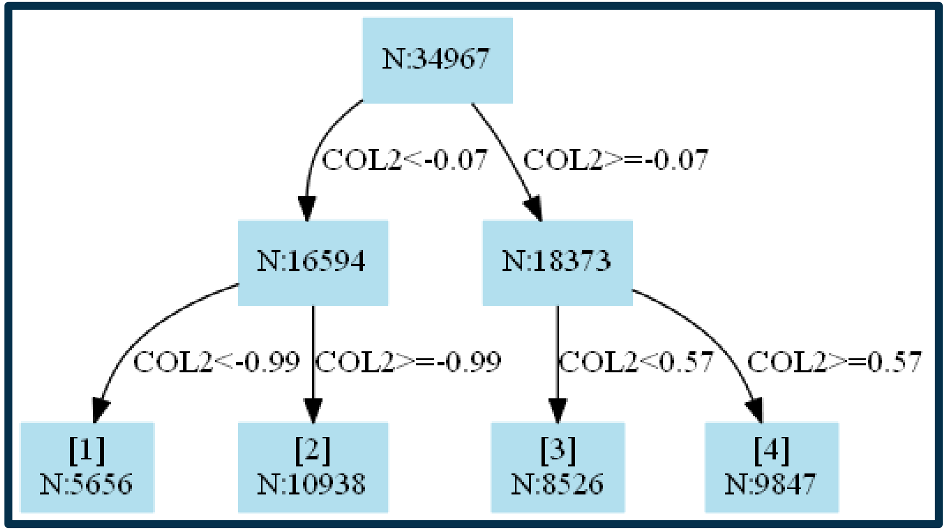
LEAF	rule	score	ADATREENUMBER
1	;COL2<-0.99	-0.183	1
2	;COL2>=-0.99;COL2<-0.07	-0.059	1
3	;COL2>=-0.07;COL2<0.57	0.024	1
4	;COL2>=0.57	0.143	1

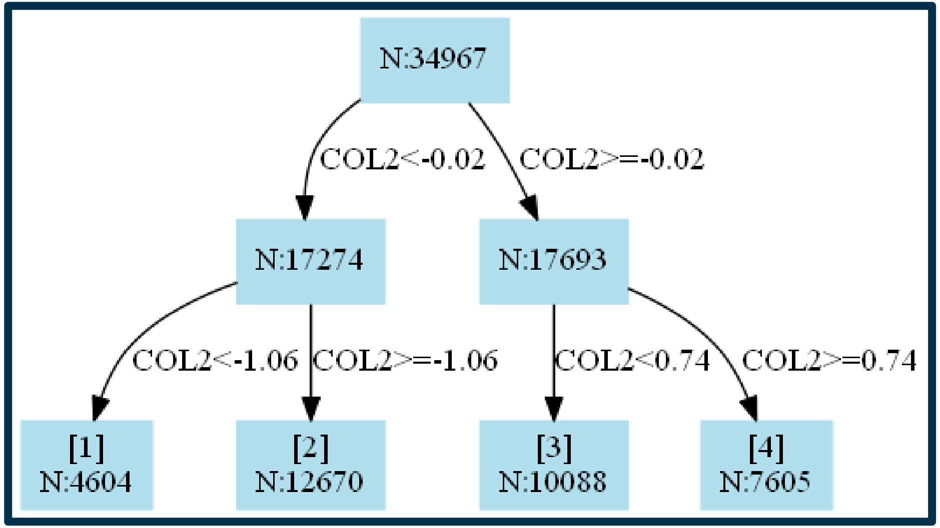
Real AdaBoost

Macro outputs

- Graphical trees
  - A helper program included in macro can generate graphical trees

Tree #2 in Real AdaBoost model





Tree #1 in Real AdaBoost model

Questions

- Thanks for your attention!

Contact	Try the macro
paul.edwards2@scotiabank.com Questions & comments welcome	• The most up-to-date macro will always be on github* • https://github.com/pedwardsada/real_adaboost

* Pull requests are welcome! Submit your bugs and patches