

scorecal

Empirical score calibration under the microscope

Ross W. Gayler

2019-08-30 Credit Scoring & Credit Control XVI, Edinburgh, UK

<https://www.rossgayler.com> — <https://orcid.org/0000-0003-4679-585X>

Introduction

What is score calibration?

- *Calibration*: Answers the question “What do my scores mean?” by empirically determining function from score to expected value of some outcome statistic
 - Inherently about groups (cases with the same score)
 - Case outcome is binary (e.g. Good, Bad)
 - Outcome statistic is some function of binary outcomes of a group of cases (e.g. $Pr(\text{Bad}|\text{score})$ or $\text{logit}(Pr(\text{Good}|\text{score}))$)
 - Result of calibration is a function from score to outcome statistic
 - Fitting a function to the data (i.e. curve fitting)
 - Typically, the function is approximately linear from score to log-odds
- *Scaling*: Transform group outcome statistic to a desired scale
 - e.g. 1:1 odds \mapsto zero points; double odds $\mapsto \Delta +100$ points
 - Think of converting temperature from Fahrenheit to Celsius
 - Calibration is always on *some* scale, maybe not the one you want

Calibration parameters

Calibration depends on:

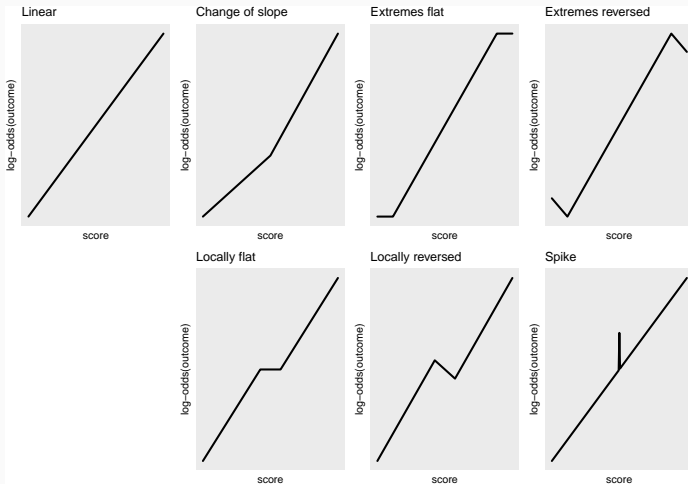
- Substantive parameters
 - Score definition (function from case attributes to a number)
 - Number is commonly integer, may be real
 - Population of cases
 - Outcome definition
- Technical parameters of calibration function estimation
 - Curve fitting technique
 - Fitting technique tuning parameters

How is the calibration function used?

- Operational process management
 - Set decision thresholds
 - Make loss predictions
- Technical diagnosis of the scoring model (my focus)
 - For a well-behaved scoring model, the score to log-odds function is generally quite linear (by definition)
 - Nonlinearity indicates there is possibly a problem
 - What is the problem? (shape of nonlinearity - not absolutely diagnostic)
 - Does the problem matter? (size of nonlinearity)
 - How to fix the problem? (“fix” may be a work-around)

Calibration function zoo

Some calibration function patterns that may be encountered:

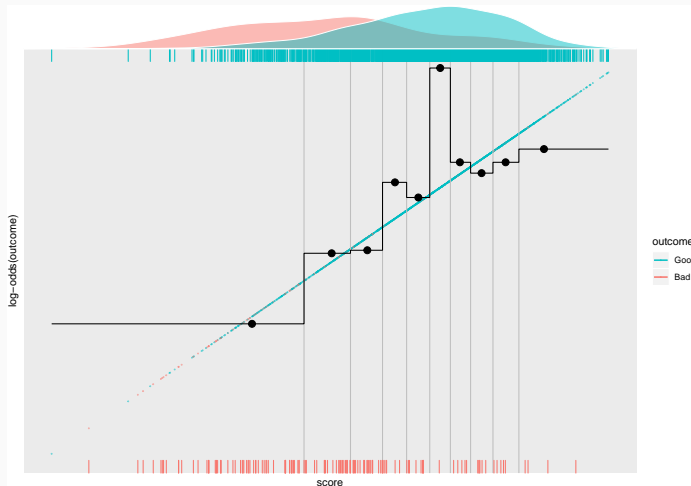


Typical approaches to calibration function estimation

- Logistic regression from score to outcome, over cases
 - `glm(outcome == "Good" ~ score, family=binomial)`
 - Estimated function *forced* to be linear
 - Unless you use `poly(score)` - but there are better ways
 - Blind to any nonlinearities
- Score bands
 - Group scores into bands; calculate outcome statistic for each band
 - Calibration function is a step-function
 - Doesn't assume *any* relationship between neighbouring bands
 - Can model any relationship (*coarsely* - because of band widths)
 - Local patterns may be hidden by bands (because of band widths)
 - Doesn't make efficient use of data (doesn't use score ordering)
 - Typically small number of observations per band
 - Large variance of estimates obscures patterns

Score band approach

Simulated data with linear score to log-odds relationship (n = 2,000; 7% Bad; 10 bands)



scorecal

scorecal objectives

scorecal: An R package for score calibration

Be a better microscope for examining deviations from linearity in calibration functions

Issues to be addressed:

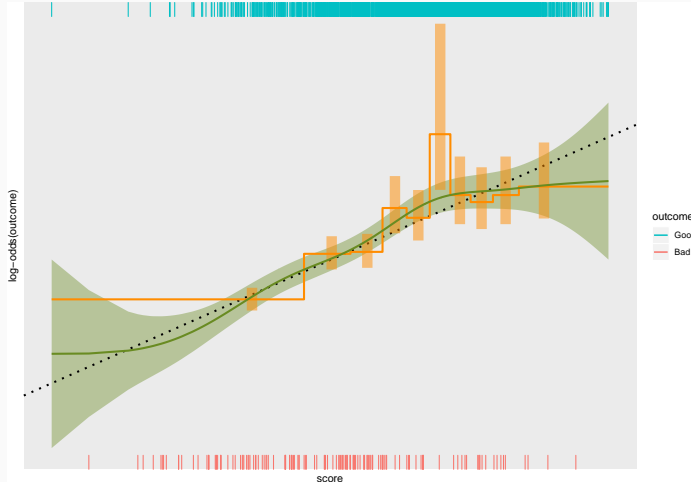
- Use data efficiently (assume continuity and smoothness)
- Relative magnitude of linear and nonlinear components
- Common scores
- Sparsity of cases in extreme tails
- Spike deviations

Use data efficiently - issue & approach

- Score band approach does not make efficient use of data because it assumes:
 - No relationship between neighbouring bands
 - No significance to ordering of scores within bands
- Expect neighbouring scores to have similar outcome statistics
(continuity of scores and smoothness of calibration curve)
 - Use smoothing spline or local regression models
 - Cases “borrow strength” from their neighbours (like having a moving-window estimator)
 - The effective number of cases used per score value is higher, giving narrower confidence intervals
 - But, outcome statistic estimates at neighbouring point values are correlated (which follows from assuming smoothness)

Use data efficiently - example

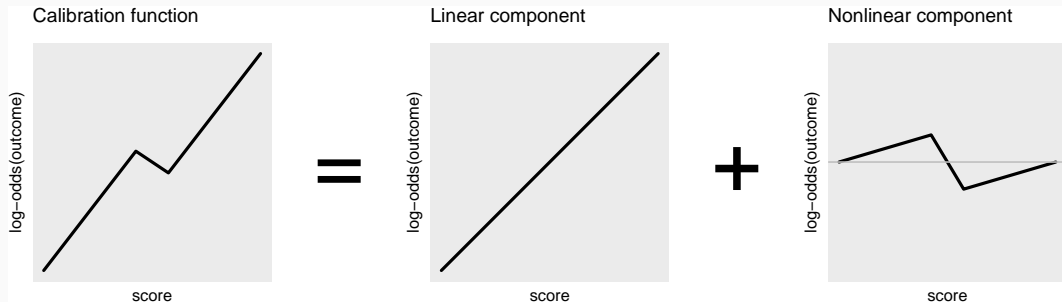
The same simulated data (95% confidence intervals)



Relative magnitude of linear and nonlinear components - issue & approach

- Global linear trend is expected pattern
- Global linear trend generally much stronger than nonlinearities
 - Nonlinearities are harder to see when combined with the strong linear component
- Decompose calibration function into linear and nonlinear components
 - Fit linear model and use as offset in nonlinear models
 - Regularisation of nonlinear component makes the linear component the default pattern when data is sparse (similar effect to a Bayesian prior)
- Display nonlinear components separately

Relative magnitude of linear and nonlinear components - example



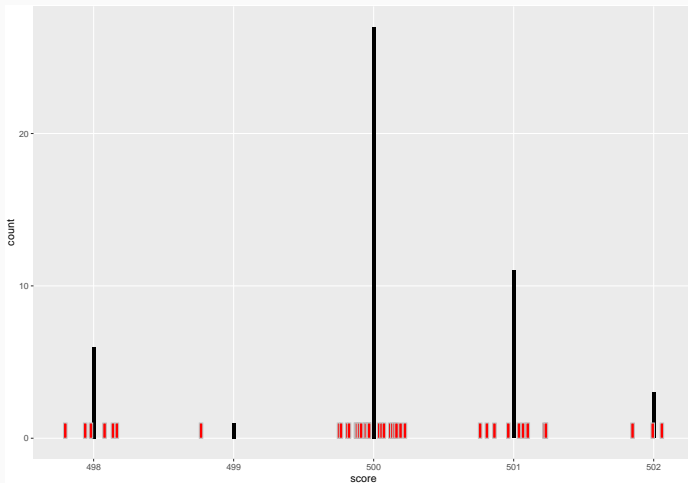
Common scores - issue & approach

For discrete scores, some score values are very common (occur on a large fraction of cases), e.g. bureau scores for New-to-Bureau cases

- For moving-window estimators with window width set at fixed fraction of cases, the fraction of cases on a common score may exceed the window width
 - No variance of the predictor (score) within the window; regression fails
- For smoothing-spline estimators, can reduce the effective number of score values
- Use jittering (add small random noise) to break tied scores
 - Jittering magnitude chosen to preserve order of scores
 - (Mostly) does no harm if using a smoothing-spline estimator
- Average the outcome estimates for all the jittered scores derived from the same unjittered score (i.e. transform the result back to the unjittered scores)

Common scores - example

Histograms of unjittered and jittered simulated data for a small range of scores



Sparsity of cases in extreme tails - issue

Distributions of scores tend to be skewed and heavy-tailed

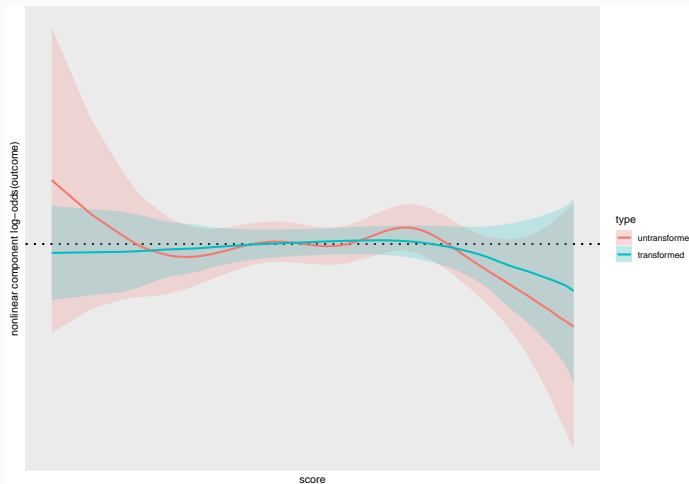
- Cases are sparse in the extreme tails
 - Confidence interval of fitted calibration curve may be *very* wide in tails
 - May include positive *and* negative slopes
 - Pattern is ill-defined in tails (needs stronger assumptions to extract the pattern)
 - Extreme tails have small fraction of cases
 - Generally not practically important
 - But, tend to be visually dominant
- Can cause technical problems
 - May be *very* few cases between smoothing spline knots
 - May be only one outcome class between smoothing spline knots
 - Case density may vary strongly within local regression window
 - Pattern at dense end of window may dominate pattern at sparse end

Sparsity of cases in extreme tails - approach

- For nonlinear smooth fit, transform jittered score to *normal* density first
 - Compresses heavy tails; expands light tails
 - Estimate calibration curve then inverse transform back to original score scale
 - Transform is to normal density rather than uniform, because uniform is too aggressive
- Effect of density transform is to increase smoothing where tails are heavy and decrease smoothing where tails are light
 - Smoothing is effectively low-pass filtering
 - Compression of tails by transformation shifts frequencies of patterns up
 - Higher frequencies are attenuated more by the smoothing
 - Inverse transformation back to original scores shifts frequencies down again
 - Expansion of tails by transformation does the converse

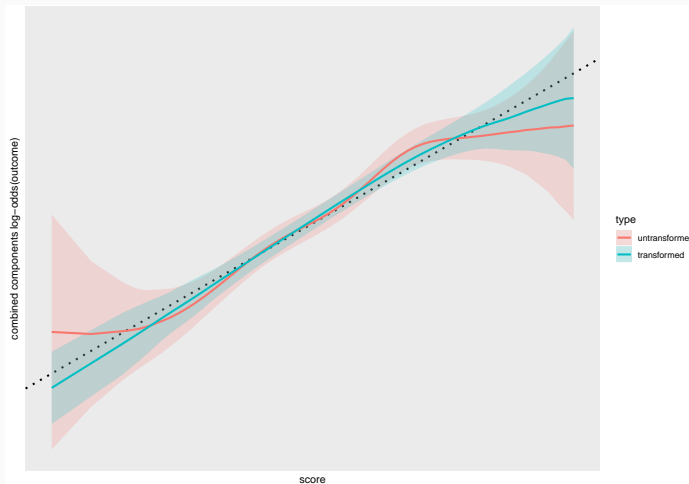
Sparsity of cases in extreme tails - example - nonlinear smooth

The same simulated data (nonlinear components; 95% confidence intervals)



Sparsity of cases in extreme tails - example - total curve

The same simulated data (combined components; 95% confidence intervals)

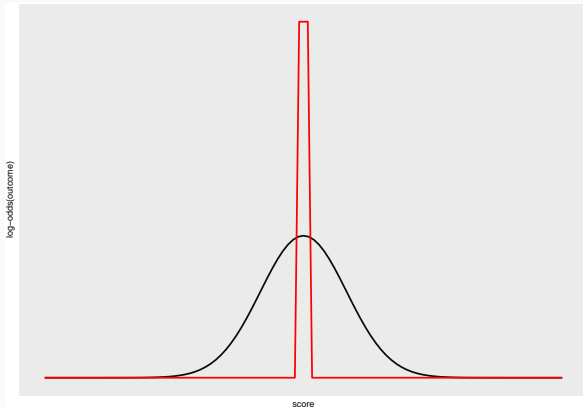


Spike deviations - issue

- A specific score can have an outcome probability very different from neighbours
 - Interpretable as the cases in the spike having the wrong score
 - Possibly due to score calculation error
 - Possibly due to applying scorecard to a different population
- Difficult to detect unless the score is a common score
- Difficult to detect in a continuous scorecard because spikes are spread

Spike deviations - issue with smoothing approach

- Spike deviations break assumption of smoothness
- Analysis developed so far *hides* spikes



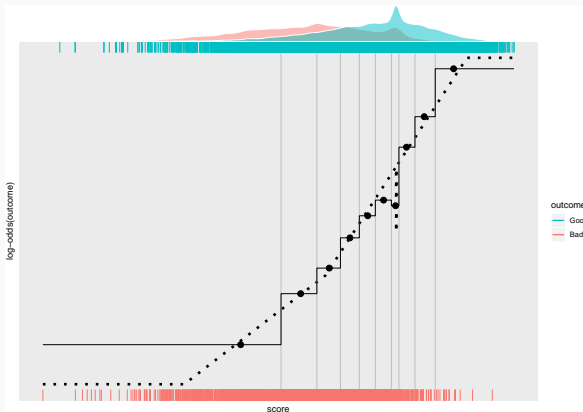
Spike deviations - approach

Approach: Model spikes with an indicator variable for each spike score

- Issue: To find the spikes we need an indicator variable for each unique score
 - Ideally, fit smooth and select spikes simultaneously with regularised regression
 - This is possible, but I haven't done it yet
- Current approach:
 - Pre-filter potential spikes by frequency (say $> 1\%$ cases)
 - Use lasso regression to select spikes with smooth as offset
 - Re-estimate smooth with selected spikes as added predictors

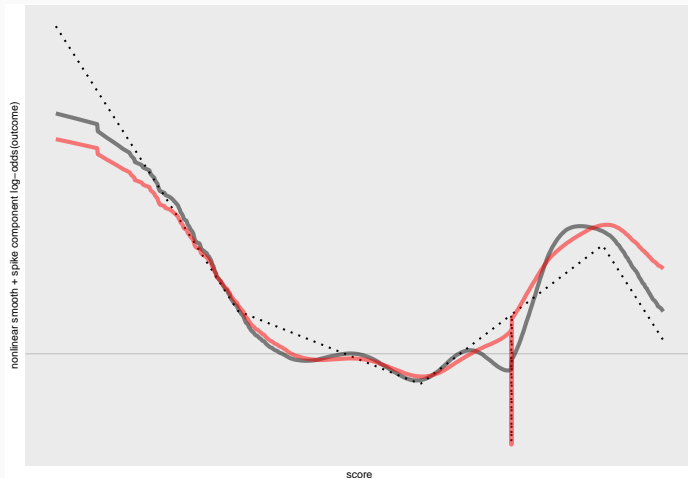
Spike deviations - example data

New simulated data with nonlinear score to log-odds relationship and a spike deviation
($n = 20,000$; $n_{\text{spike}} = 1,000$; 10% Bad; 10 bands)



Spike deviations - example results

Compare smoothed nonlinear components estimated with and without spike term



Conclusions

- Calibration curves can be usefully decomposed into linear, smooth nonlinear, and spike components
- The decomposition can be automated reasonably well
 - Everything breaks under some circumstances
- The method is a work in progress
- All the R code for this presentation is publicly available
 - The R package will soon be publicly available (very alpha)

Meta conclusions

Content of analysis - simple stuff can be interesting

- Useful inferences can be drawn from comparatively restricted evidence
- Apparently simple problems can be full of subtleties

Tools for analysis - openness and reproducibility are important

- Reproducible computational research
 - Open source tools
 - Open source research
 - Tools to simplify reproducibility
 - Workflows for reproducibility

Resources

This presentation is implemented as an executable R notebook, which is publicly accessible on GitHub at: github.com/rgayler/scorecal_CSCC_2019

<https://doi.org/10.5281/zenodo.3381641>



This presentation is licensed under a Creative Commons Attribution 4.0 International License

The scorecal R package will be publicly accessible on GitHub at:
github.com/rgayler/scorecal