Predictive Analytics Workshop

Penalized GLM

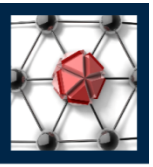
Missy Gordon, FSA, MAAA
Principal and Consulting Actuary
Milliman, Minneapolis

Joe Long
Assistant Actuary and Data Scientist
Milliman, Minneapolis

ILTCI

18th Annual Intercompany Long Term Care Insurance Conference

Limitations

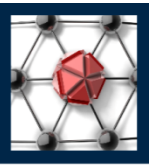


This presentation is intended for informational purposes only. It reflects the opinions of the presenter, and does not represent any formal views held by Milliman.

Milliman makes no representations or warranties regarding the contents of this presentation.

Milliman does not intend to benefit or create a legal duty to any recipient of this presentation.

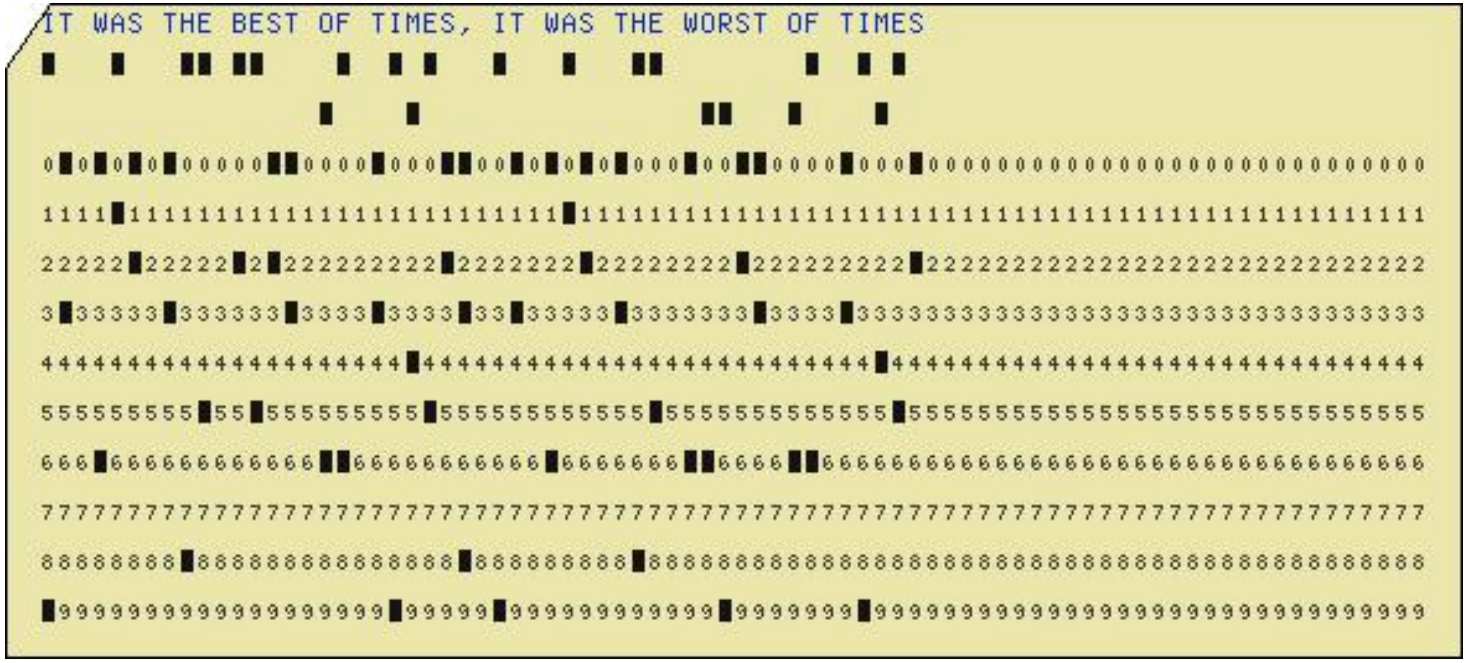
Classical GLM: challenges traversing BVT



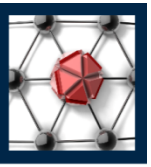
- Gives full credibility to data
- Violating underlying GLM assumptions (e.g., overdispersion) may produce misguided conclusions relative to variable selection (p-values)
- AIC/BIC used to balance model complexity, but still give full weight to the data

Is there a better way?

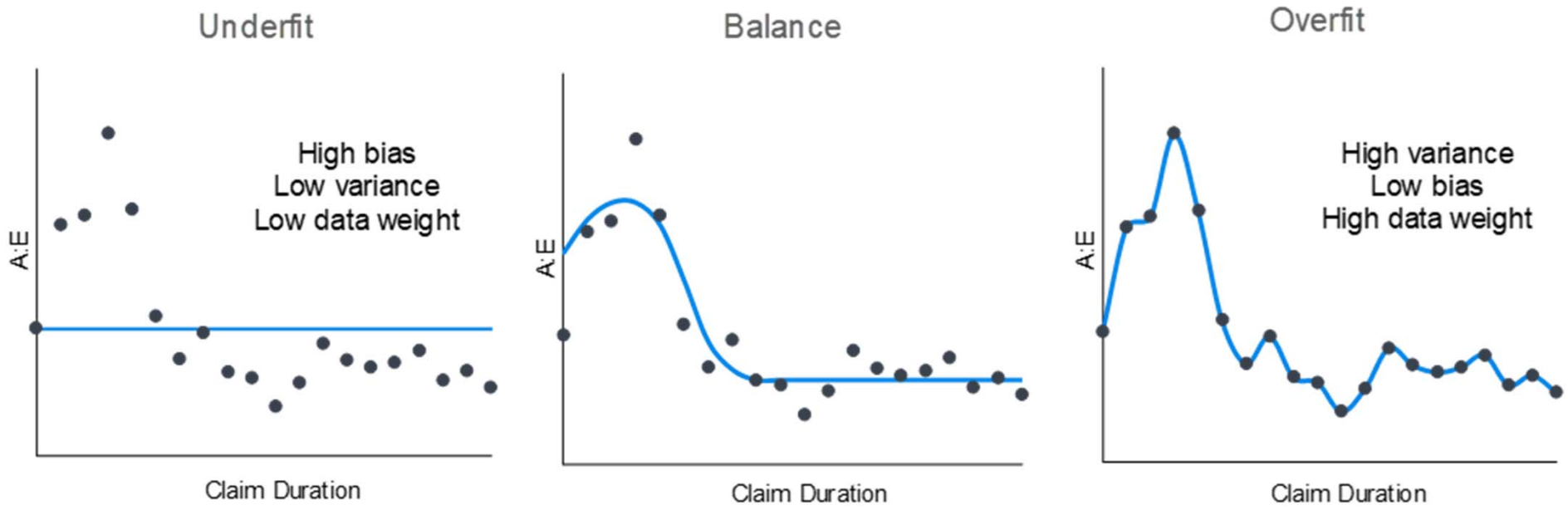
Classical GLM has been around for a long time and with advance computing power, is there a better way?



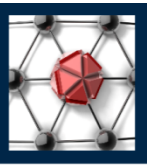
Penalized GLM



Use a penalized GLM to automatically traverse the bias-variance tradeoff

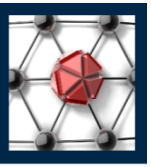


How penalization helps



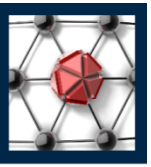
- Penalizes (shrinks) coefficients
- Penalty determines data credibility
  - No penalty = full credibility = Classical GLM
  - Full penalty = no credibility = benchmark
- Automatic feature selection and better handles multicollinearity
  - Penalty determined to minimize prediction error
  - Doesn't rely on tests with underlying assumptions that could be violated

In-sample vs. out-of-sample tests



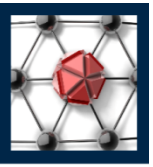
	In-sample tests	Out-of-sample tests
	AIC, BIC, Adjusted R^2 p-values to prune parameters	Separate train/test datasets k-fold cross-validation
Pros	- Model selection using all data - Fast to calculate	- No theoretical formulas - Compare across algorithms
Cons	- Relies on theoretical formulas - May misguide if assumptions violated - Harder (or not possible) to compare across algorithms	- Computationally expensive - Potential to misuse if not setup properly (information leak)

Modeling process overview



1. Data gathering and exploring
2. Pre-process dataset by standardizing/transforming necessary variables
3. Coerce dataset into a format desired by model
4. Use K-fold CV on training dataset to determine optimal hyperparameters (penalty in this case)
5. Predict using trained model on validation dataset
6. Review predictions for reasonableness and test of fit
7. Reiterate steps until happy with the modeling process
8. Final model check by predicting on testing dataset
9. Train model on full dataset

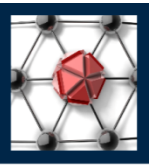
R 0410 Stepping Stone – No New Variables



- Attach expected benchmark for offset
  - Benchmark varies by duration and gender
- Start simple and limit variables to those underlying the benchmark (stepping stone)
- Explore data using A:E to feature engineer
  - Gradient boosting machine (GBM) is an advanced technique helpful with engineering
- Pre-processing setup for penalized GLM

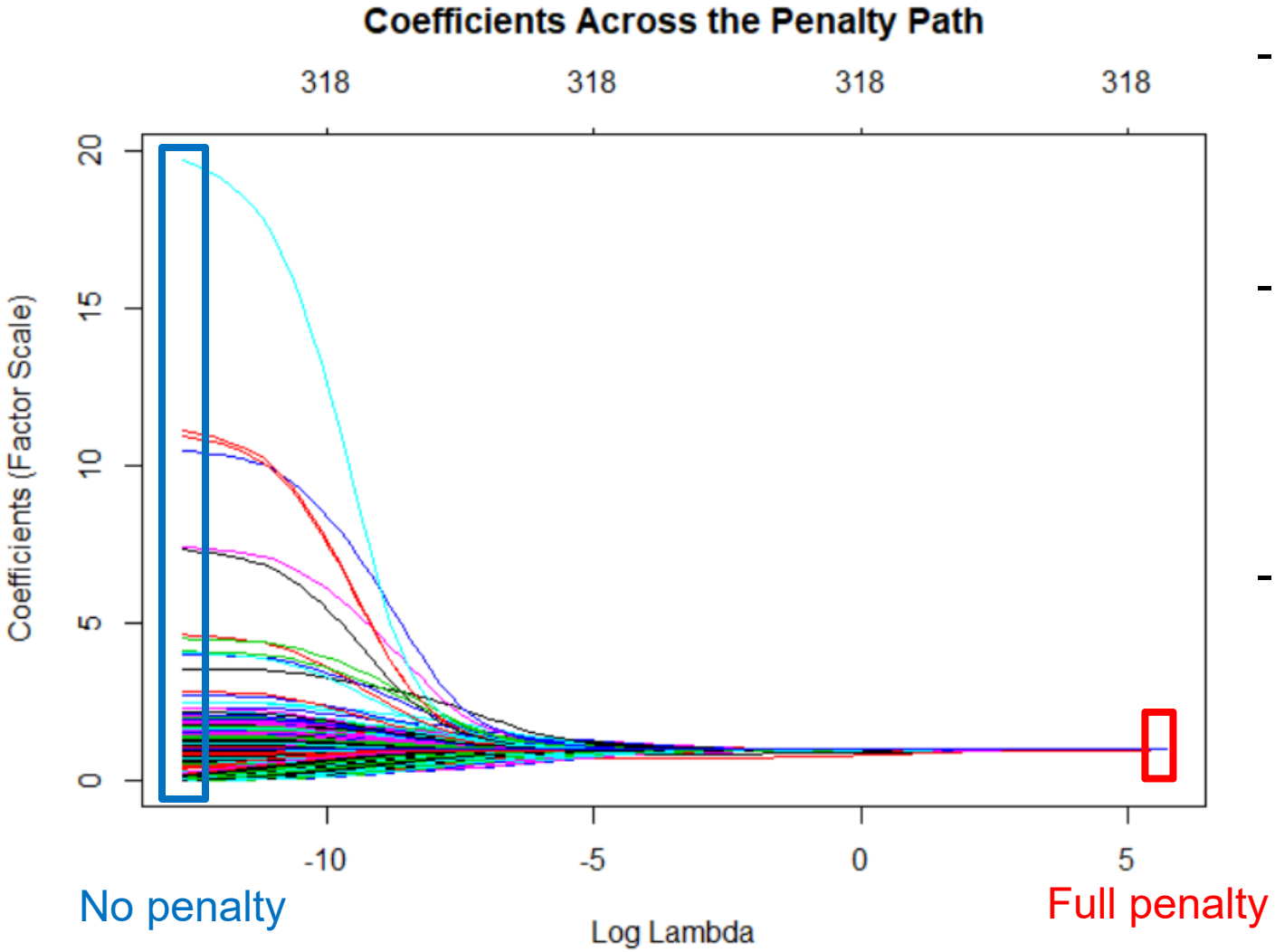
[Run R 0410: 4.1.0-4.1.3]

Minimizing prediction error



- Objective to minimize MSE or SSE
- Classical GLM: $\mathrm{SSE} = \sum(\boldsymbol{Y} - \boldsymbol{x\beta})^2$
- Ridge: $\mathrm{SSE} + \lambda * \sum \boldsymbol{\beta}^2$
  - Shrinks coefficients, but remains > 0
  - Helps with multicollinearity

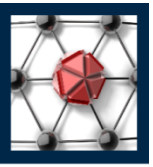
Coefficients after shrinking



- Each line is the coefficient (B_i) for variable x_i
- See how the coefficients shrink as the penalty increases
- Each point in a vertical line are the coefficients for a model with a given penalty

[Run R 0410: 4.1.4-4.1.6]

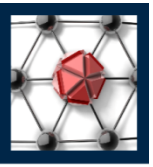
Minimizing prediction error



- LASSO: $\text{SSE} + \lambda * \sum|\boldsymbol{\beta}|$
  - Can shrink coefficients to 0
  - Provides automatic feature (variable) selection

- Elastic net: $\text{SSE} + \lambda * (\boldsymbol{\alpha} * \sum|\boldsymbol{\beta}| + (\mathbf{1} - \boldsymbol{\alpha}) * \sum \boldsymbol{\beta}^{\mathbf{2}})$
  - Blend of Ridge and LASSO
  - Helps with multicollinearity and provides feature selection
    - $\boldsymbol{\alpha}$ controls the blend
    - $\boldsymbol{\alpha}$ = 0 then Ridge, $\boldsymbol{\alpha} \in (0, 1)$ then Elastic net, $\boldsymbol{\alpha}$ = 1 then LASSO

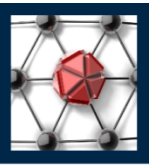
Choosing alpha



- Domain knowledge
  - Ridge ($\alpha = 0$) to keep all variables
  - Elastic net ($\alpha \in (0, 1)$) to capture some feature (variable) selection
- Performance based:
  - Useful for modeling competitions like Kaggle
  - Test a range of alphas from [0, 1]
  - Choose alpha that minimizes cross-validation error
- Exercise 1. As we change from Ridge to LASSO, what happens to the number of variables?

[Play R 0410: 4.1.5-4.1.6]

Cross-validation automates traversing BVT



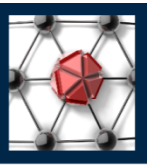
3-Fold	Test 1	Test 2	Test 3
1 33%	1 Holdout	1 Use	1 Use
2 33%	2 Use	2 Holdout	2 Use
3 33%	3 Use	3 Use	3 Holdout
Calibration data	100%	100%	100%

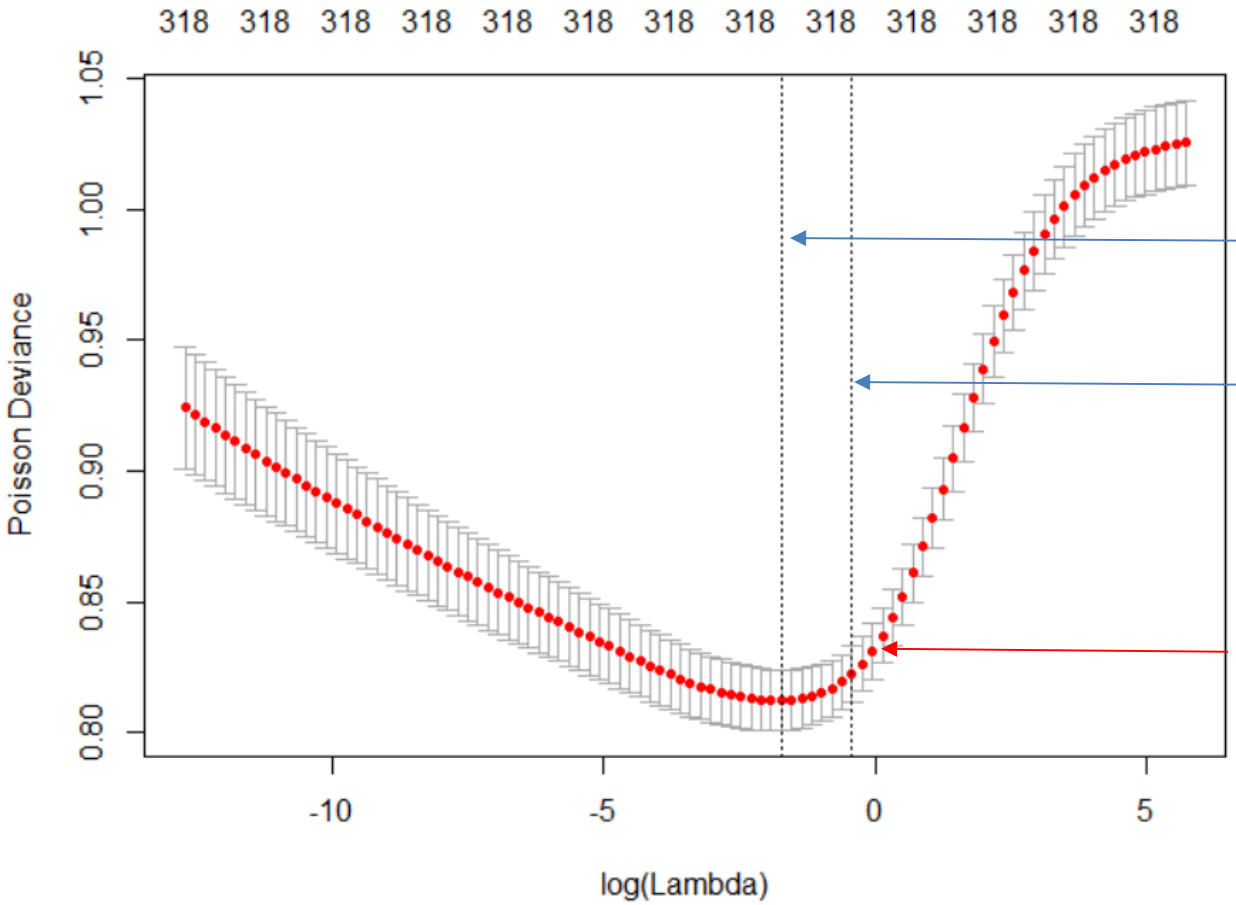
MSE on holdout data	Average
	Test 1
	Test 2
	Test 3

K-fold cross-validation

- Use subset of data to develop coefficients
- Calculate error of predicted values on holdout data
- Average error across the k tests

Traversing the BVT = choosing the penalty





Judgement to choose between MIN and 1SE

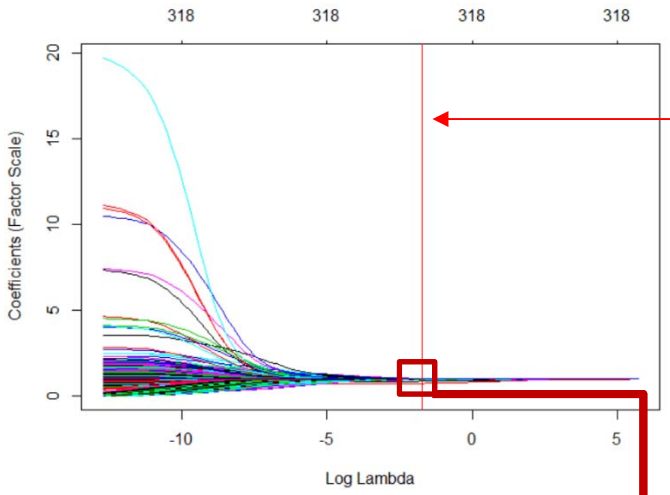
Minimum error

1 Standard error

Average error from cross-validation for given penalty

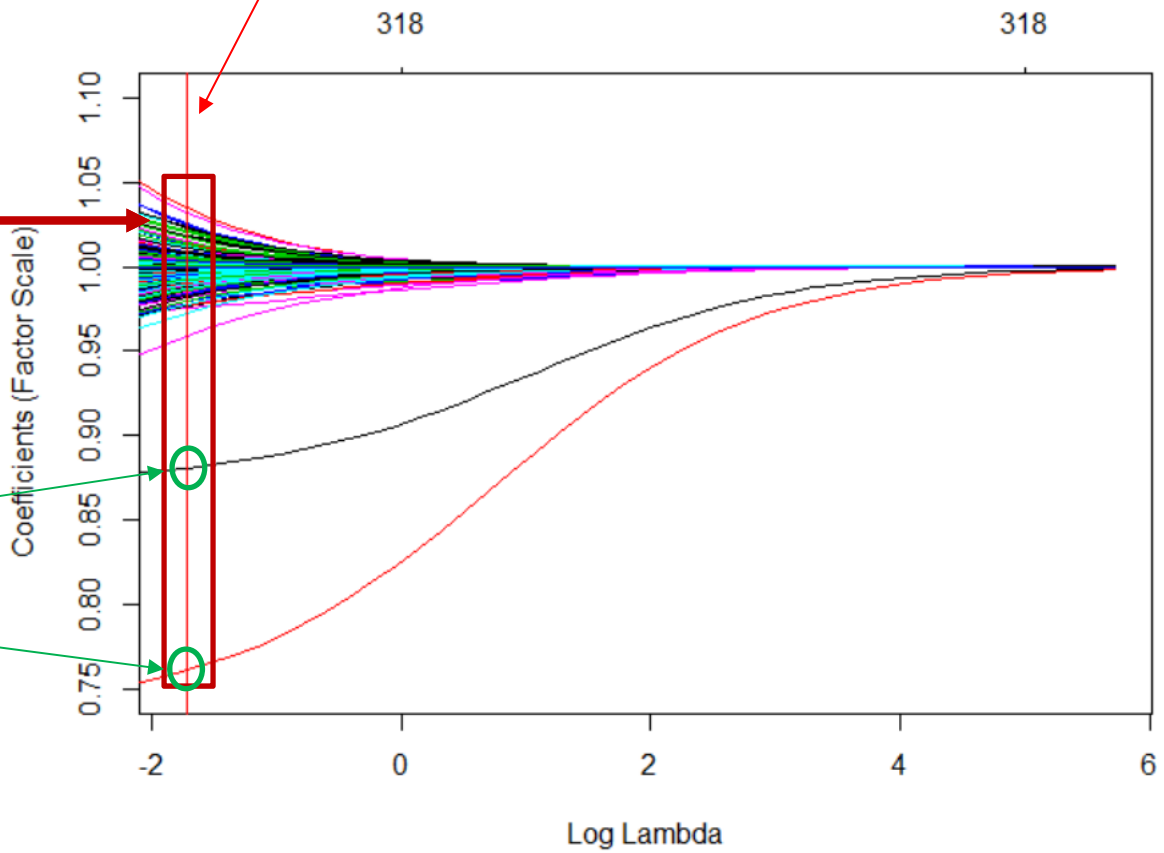
Overfitting	Balanced	Underfitting
No penalty	Minimize error	Full penalty
Fully trust data	Credibility of data	Don't trust data
(Classical GLM)		(Benchmark)

Coefficients for optimal penalty



Coefficients on this line
use a model with a penalty
that minimized error

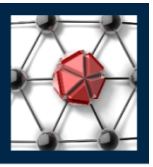
zooming in



0410_Coeff_lambda
_min_and_1se.csv

Female factor 0.88
Male factor 0.76

Review for reasonableness



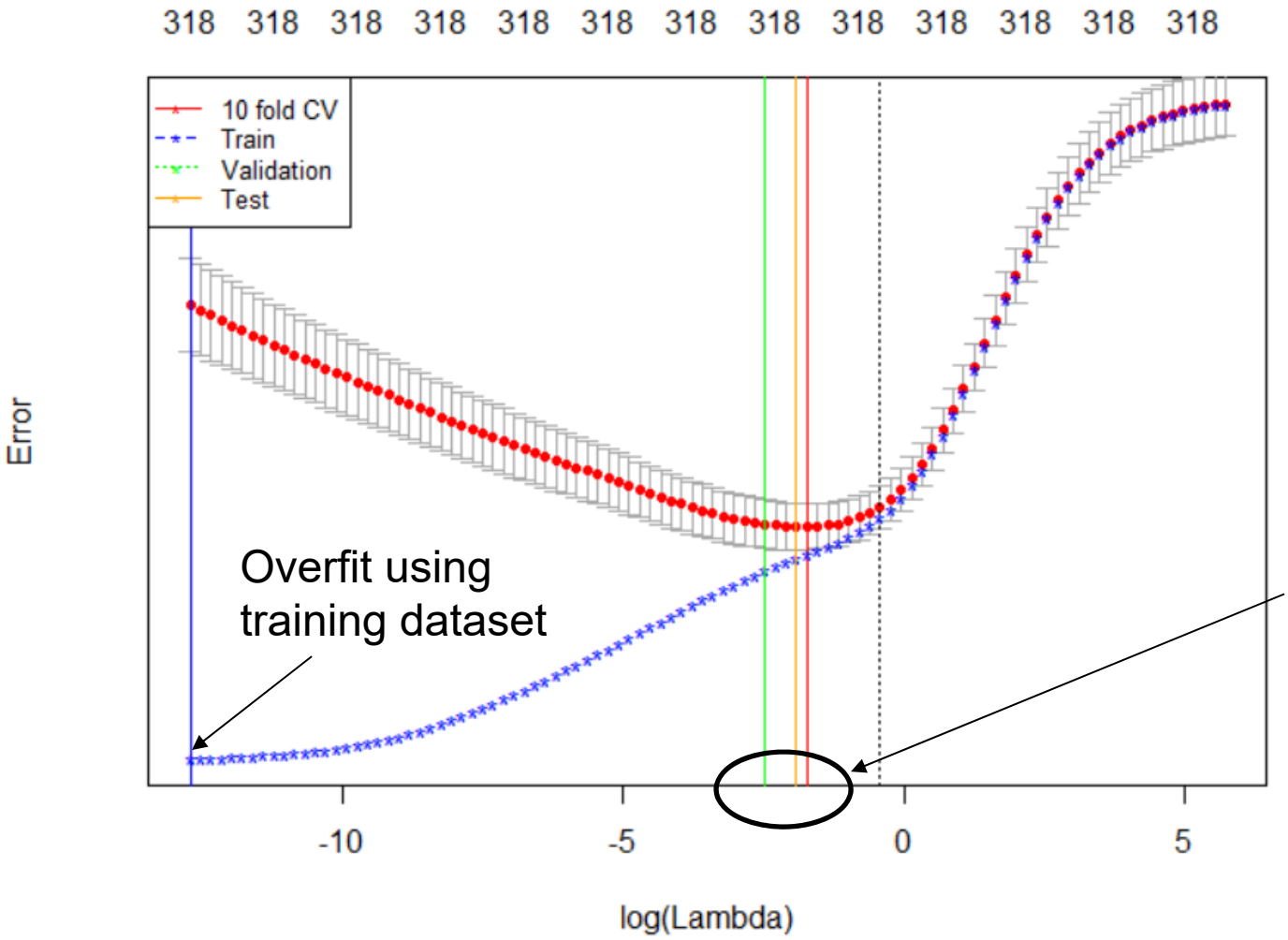
Export and review in Excel

- Factors – direction and magnitude
- Adjusted expected terminations
  - Graphs of shape and pattern
  - Relationships between variables
- Shape of MIN vs 1SE adjusted expected
  - Smoothness
  - Magnitude of deviations from expected

[Run R 0410: 4.1.7-4.1.8]

Test of fit – Prediction Error

Compare the optimal penalty selected based on the error calculated on the following datasets: cross-validation, training, validation, and test



CV does a good job picking the penalty that also works well for the validation and test datasets

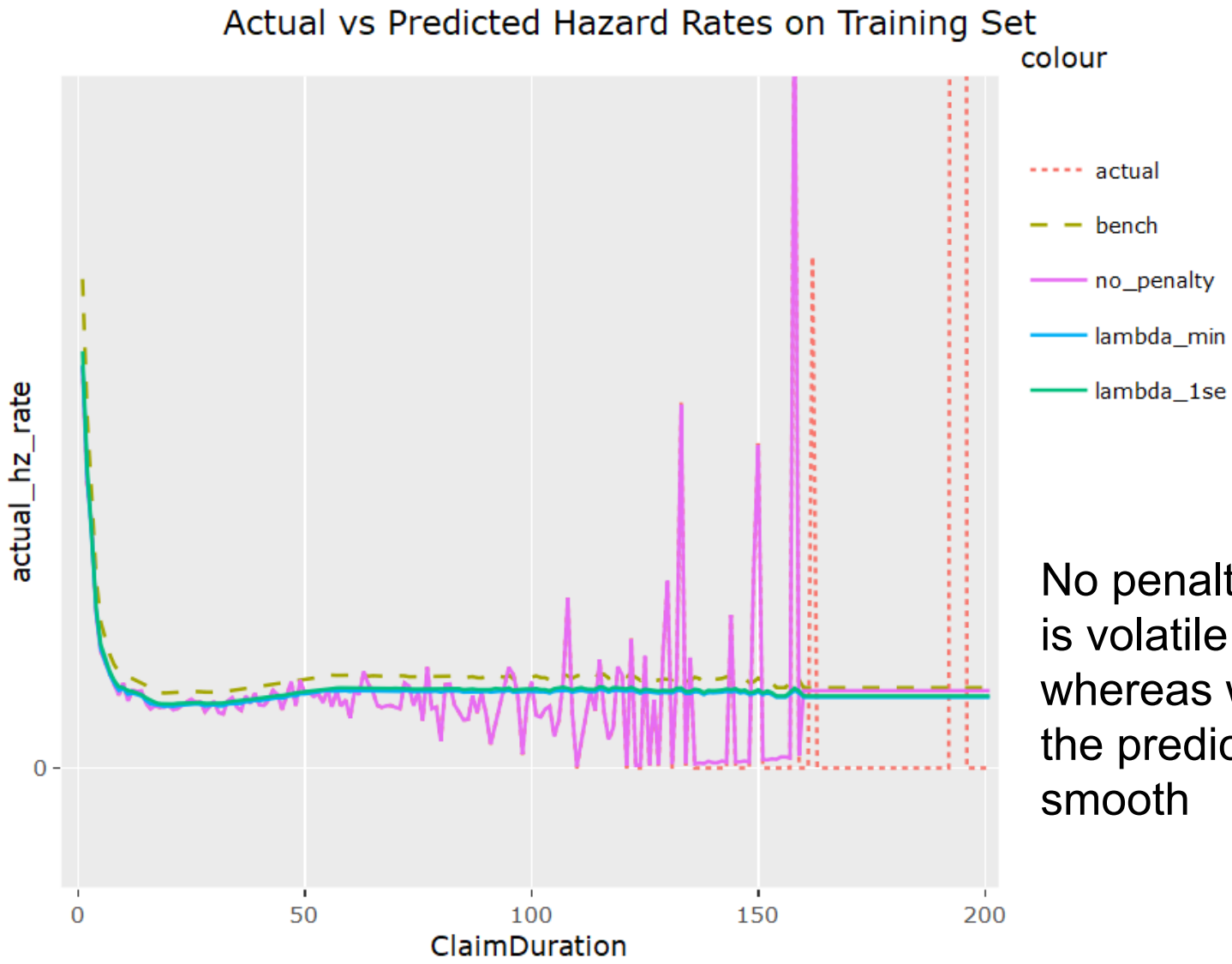
[Run R 0410: 4.1.9]

Test of fit – A:E vs MSE

- Review metrics on training, validation, and testing datasets
- A:E results can mask offsetting errors
- Errors can't hide within MSE, they accumulate
- Classical GLM performs best on training dataset because it focuses on minimizing bias (overfit)
- Penalized GLM performs better on validation and testing datasets because it traverses BVT

[Run R 0410: 4.1.10-4.1.11]

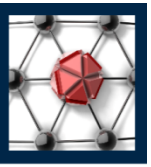
Test of fit – Actual vs Predicted



No penalty (Classical GLM) is volatile (overfits the data), whereas with penalization the prediction is relatively smooth

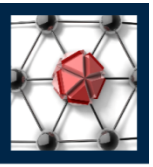
[Run R 0410: 4.1.12]

Challenges that remain



- Extrapolation
- Trend
- Navigating complex interactions
- Review of results for reasonableness
- Sharp knives… beware of the dangers

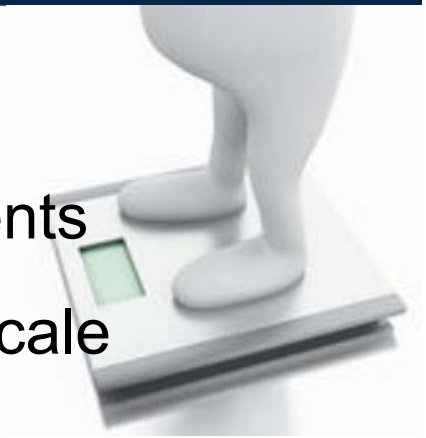
R 0420 Adding New Variables



- Simple example adding two new variables not underlying the benchmark
  - Calendar year
  - Claim type (situs)
- Similar steps and process as 0410
- Standardize continuous variables (calendar year)

[Run R 0410: 4.1.13]
[Run R 0420: 4.2.0-4.2.1.3]

Standardizing variables



- Scale is important
  - Because penalty is applied to sum of coefficients
  - Penalized coefficients will change based on scale
- Goal of standardization
  - Resulting coefficients come from the same distribution
  - Penalization applied evenly to all the coefficients
- Standardized to mean of 0 and variance of 1
  - Note that sometimes other techniques may be applicable
  - For example, leave binary variables as is

[Run R 0420: 4.2.1.4-4.2.13]

Exercise

2. Add additional variable(s) to 0420
   - View the ctr_data dataset and choose variables to add
   - Walk through the 0420 and look for where you need to add/analyze new variable
     - Hint to update sections: 4.2.1.1, 4.2.2.2, 4.2.3, and 4.2.4
3. Home work: After confirming your modeling process works well, the final step is to re-run the program using all data (not just training) to develop final coefficients and again review results for reasonableness before using