Insurance Pricing With GLMs

Alan Chalk

2025-01-22

Table of contents

Acknowledgements

To Serban Dragne:

You've taught me so much about R, Python, SQL and ML in general.

You are a constant reminder to me about how little I know.

Thanks.

I think.

1 Introduction

Dear Reader,

Can we fit good GLMs?

Can we rely mostly on the data to choose which features to select?

Can we fit those features accurately, with manual smoothing left as close to the end of the process as possible?

Some techniques to start us on this journey are discussed in the CAS monograph: From GLMs to Comprehensive Insurance Pricing: Techniques and Challenges.

We include here R code to help the reader try out those techniques.

Alan Chalk May 2025

2 Data preparation

It is very tempting once a dataset is available, to jump right in to the modelling stage - because that is interesting and data preparation is boring. Or because you are being chased by management for a new and improved rating plan which they in turn have promised their boss to have ready by yesterday. In this regard they will often quote the 80-20 rule to you. Meaning that you can get 80% of the benefit of a data cleansing exercise by just 20% of the effort. (Or they might use the phrase "perfect is the enemy of good".) This magical rule means that if you need a month to do data cleansing, your boss can reasonably expect it of you in just under a week, and that whatever has been left undone will not matter to the final result. There is no simple answer to such time pressures. But realistic planning at the start of a project can help decide what should and should not be attempted, given the time available.

A typical outcome of poor data preparation is getting to the end of a few days (or months) of modelling and then realizing that you included a feature which should not be included. Or you treated a feature as numeric which should been treated as a character / factor. An example of the latter is the number of doors of a vehicle (in auto insurance). Treating this as a numeric feature will often lead to the assumption that the risk of accident decreases (or increases) linearly as the number of doors increases. You probably did not want to assume this. The net result of such errors can vary from some minor changes needing to be made post-hoc, to having to redo the whole exercise.

This chapter will discuss and provide code examples for various aspects of data preparation, including:

- Naming conventions for our features
- Character and numeric features
- Feature screening
- Missing values and general sense-checking

3 Finding and fitting interactions

When we train decision trees, random forests, or gradient boosting trees, or neural networks, they automatically find interactions for us (even if it is not totally straightforward to actually find out what they are). In fact, some time back, a certain type of decision tree was known as CHAID - a CHi-squared Automatic Interaction Detector.

When we train GLMs they do not automatically find interactions for us.

Thus, a price for the transparency of the GLM is that we need to find (and code up) interactions (if there are any).

The approach we take here is to first find and propose candidate interaction pairs and then, for each candidate, to code it up as a feature for our GLM and finally to fit it. Interactions which improve CV performance and seem sensible for the domain we are working in, are likely to end up in our final model.

- Domain knowledge. policyjolder age x vehicle accelaration (0-60)
- Learning from black-box models
- MARS

4 Summary