Individual Claims Reserving with Stan

August 29, 2016

The problem

The problem

- Desire for individual claim analysis - don't throw away data.
- We're all pretty comfortable with GLMs now. Let's go crazy with lots of variables.
- Risk management and regulatory oversight mean that second moment estimate becomes more critical
- Mama weer all Bayezee now!

More optimistically . . .

- Actuaries are seen as vital elements in steering the claims department. Must have a laser focus on individual claims.
- Actuaries are our go-to resource for fancy pants predictive models. Let's use this in our claims department.
- Managers have put up with the limitations of chain-ladder reserving for far too long. We need more technical solutions to old problems.

What I'll show you

- Detailed walk through of an example first proposed by Guszcza and Lommele.
- Comment on how this fits with aggregate methods
  - Bifurcated data
  - Hierarchical models
  - Bayesian
- Easy stan walk through
- Stan for individual claims

  What I won't talk about:

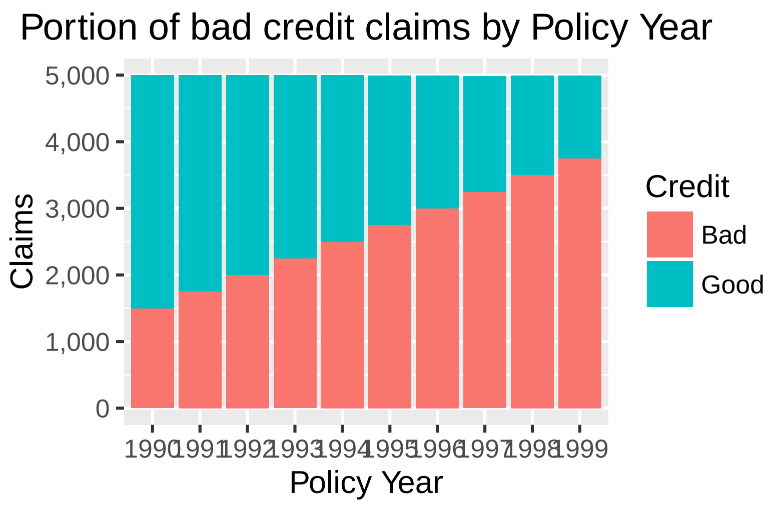
- The stochastic simulation assumptions
- IBNYR
- Diagnosing a Stan fit

Guszcza and Lommele

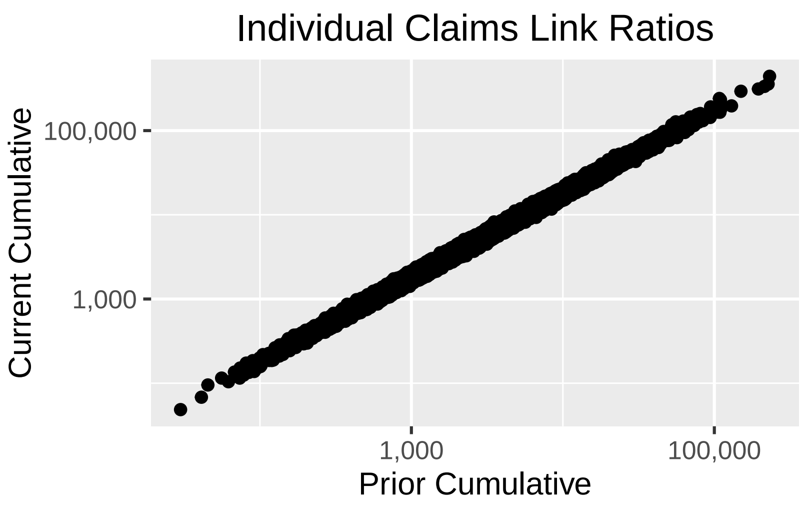
Guszcza and Lommele

Published way back in 2006, Guszcza and Lommele (2006) presented a model to develop reserves based on individual claim data.

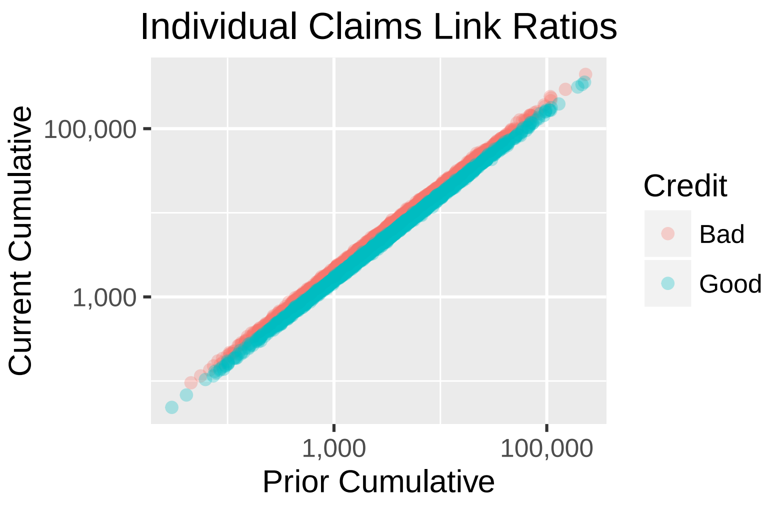
- 5,000 claims per year
- Value at first evaluation period is the same, lognormal
- Subsequent amounts are multiplicative chain ladder
  - Current period amount equals prior period times link ratio
  - Link ratios are random
  - Expected value of link ratio depends
- Fit the model using a Poisson GLM
- Aggregation of claims data misses the specific structure of the data



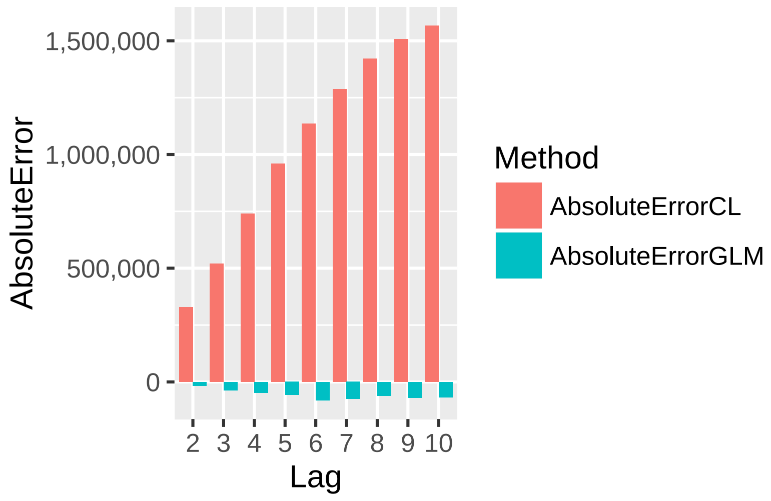
Regression based on individual claims looks pretty good. Axes are on a log scale.



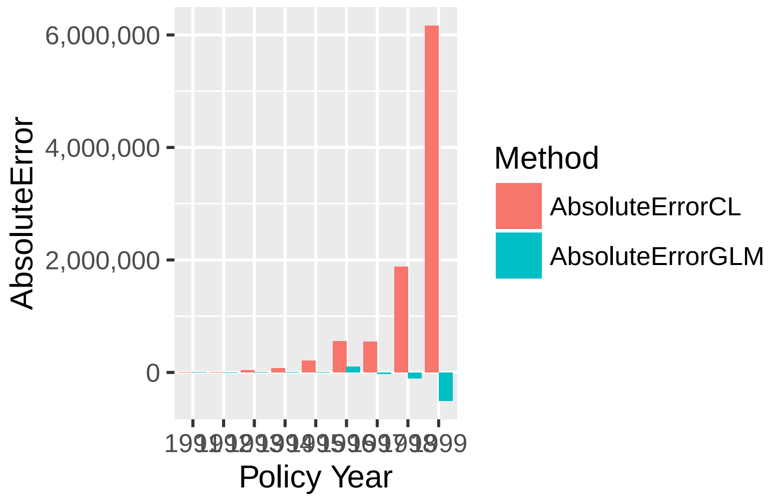
However, things look different when we differentiate based on credit grouping.



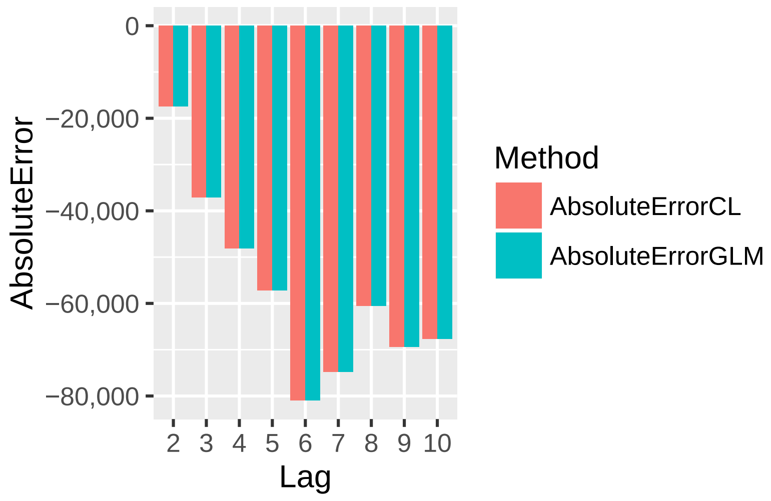
GLM vs Chain Ladder



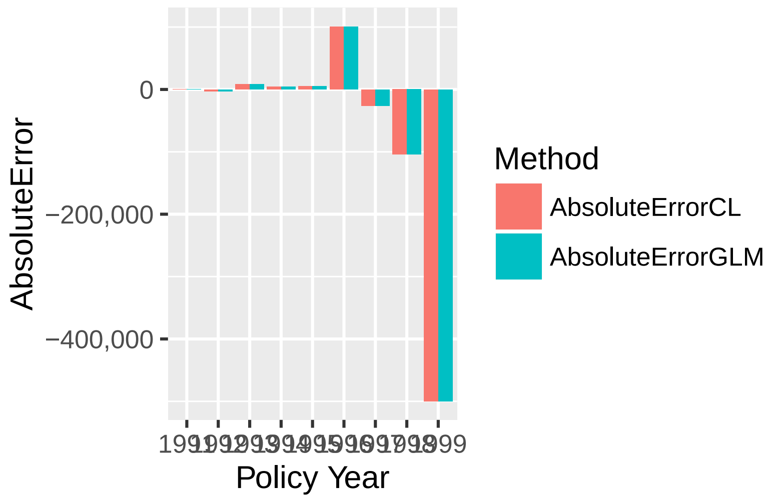
GLM vs Chain Ladder



What if we had split our claims data?



What if we had split our claims data?



What if we had split our claims data?

- We would have been fine.
- Not surprising. The Poisson likelihood function aggregates naturally.

$$L = \prod_{i=1}^{N} \frac{\lambda^{x_i} e^{-\lambda}}{x_i!}$$

- The subscripts don't really matter. We can add individual claims into accident years and then fit a model, or just apply the model

However

- When we look at individual claims, we may construct a training and test data set. That's not nearly as effective when we aggregate into accident years before fitting.
- Issues like heteroskedasticity are easier to spot with more data.
- Confidence intervals around link ratios are tighter with more sample points.
- This won't necessarily work for other distributional forms.

So we're done?

So we can split our data into subsets and get results that are just about as good as when fitting individual data without quite so much fuss.

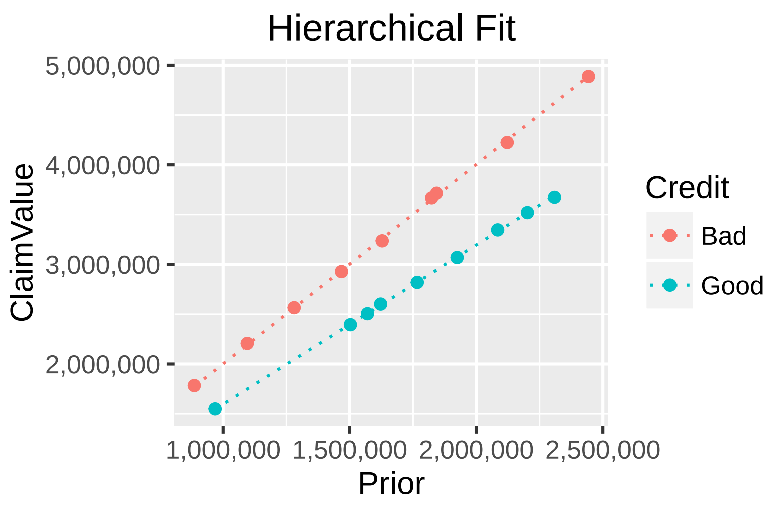
- What if we're worried about credibility of the resulting segments?
- Well, we can just use hierarchical models, right?

Hierarchical methods

Hierarchical models

- Also referred to as fixed/random effects models
- Common in social science research to control for "house effects" (classrooms in schools, schools in districts)
- Presume that a model parameter (like a link ratio) is the result of a random draw from another distribution which has its own hyperparameters
- Example: link ratios by state ought to be fairly similar. We may consider them to be random draws from a nationwide distribution
- Very similar to the actuarial concept of credibility
- Read more in (Guszcza 2008)

Fit a hierarchical model



Hierarchical Models

- Note that this is a bit different than simply splitting the data. We're looking at all of the data at once.
- In our example, we know that credit score affects loss development *by construction*. In the real world, that assumption is harder to make.
- Good news: we're using all of our data.
- Bad(ish) news: estimates are based only on the data we have to hand. No exogeneous information.

Is there an alternative?

What if we don't want to look at all of our data at the same time? What if we don't have data?

Bayesian estimation

Bayesian estimation

Bayesian estimation allows us to replace data with prior judgment.

	Hierarchical models	Bayesian
Fit method	Maximum likelihood	Closed form if you're lucky, numerical methods (like MCMC) if you're not

	Hierarchical models	Bayesian
Complementary data	Part of the fitting process	Use a prior distribution
Objectivity	Objective	Subjective when the prior swamps the data

So lets all be Bayesian!

- But Bayesian estimation is hard!!
- There are integrals!
- Complicated integrals!
- Metropolis? Bugs? Jags? Shrinkage? What on earth do these things even mean?

Enter Stan

- Named for Stanislav Ulam, father of Monte Carlo methods (and, um, made lots of improvements to nuclear weapons)
- Project w/Andrew Gelman and many others
- Predecessors were BUGS, Jags (GS = Gibbs sampler)
- Available for a number of platforms and languages, including R and Python
- Uses MCMC as the estimation engine
- Very simple syntax to describe models

Stan doco

- Loads of examples ("Example Models" 2015) here: `https://github.com/stan-dev/example-models/wiki`
- Lots of examples from Gelman and Hill (2006). If you haven't yet purchased this book, set that right immediately!
- Stan users guide (2015) is > 500 pages. Full disclosure: I haven't every page of it.

Simple STAN example

Imagine I've flipped a coin ten times and came up with two heads.

- What is a distribution around p?
- What is my prediction for the next five flips?
- How does my answer change if I'm *pretty sure* that the coin is fair?

What do I need for this?

I need the following:

- Data
- Some prior distribution
- Model
- Predictions

All of this information is stored in a block of data for Stan. Gather the data with R, pass it to Stan and let it go to work. I store my Stan information in a separate file.

Data

```
data {
  int<lower=0> sampleN;
  int<lower=0, upper=1> heads[sampleN];
  int<lower=0> predN;
  int<lower=0> betaA;
  int<lower=0> betaB;
}
```

Prior distribution and model

```
parameters {
  real<lower=0,upper=1> theta;
}

model {
  theta ~ beta(betaA, betaB);
  heads ~ bernoulli(theta);
}
```

In this simple example, there's just the one parameter. θ is the beta variable that serves as our range around the binomial coefficient, p.

Prediction

```
generated quantities{
  real heads_pred;
  heads_pred <- binomial_rng(predN, theta);
}
```

The sample size doesn't need to be the same as the predicted results. I could use five years of data to predict the next two, or whatever.

Run the code within R

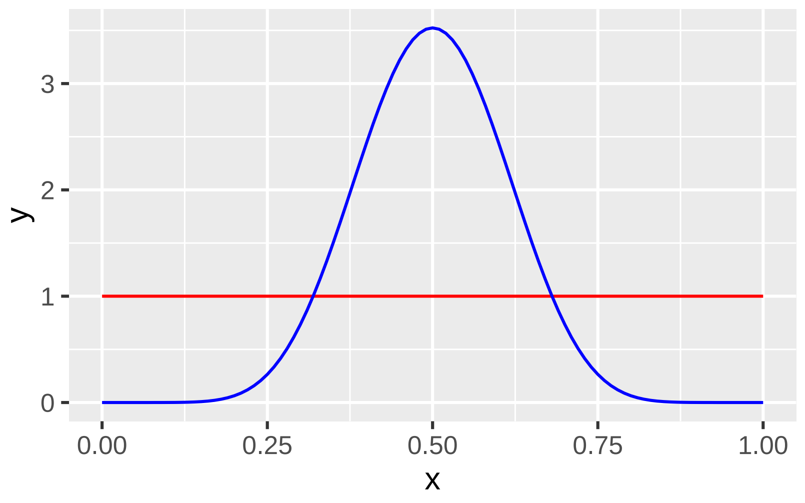
```
sampleN <- 10
```

```
heads <- c(1, 1, 0, 0, 0, 0, 0, 0, 0, 0)
predN <- 5
fit1 <- stan(file = './stan/bernoulli.stan'
            , data = list(sampleN
                          , heads
                          , predN
                          , betaA = 1
                          , betaB = 1)
            , iter = 1000
            , seed = 1234)
```

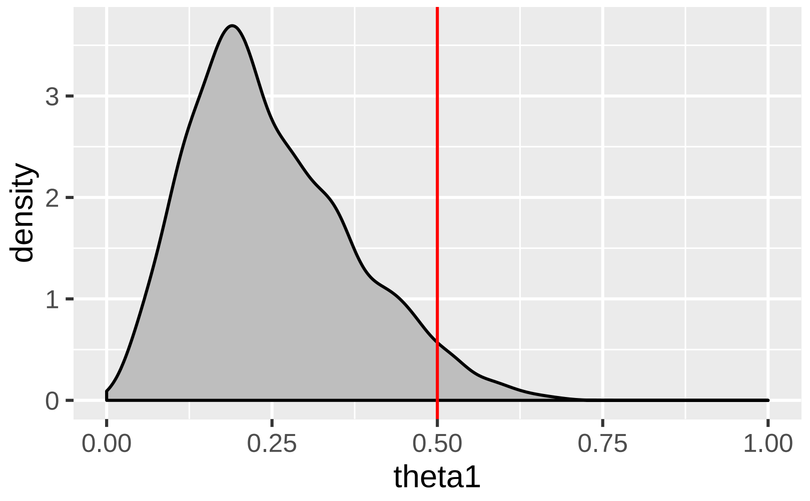
Two priors

- No idea if the coin is fair
- Pretty sure the coin is fair

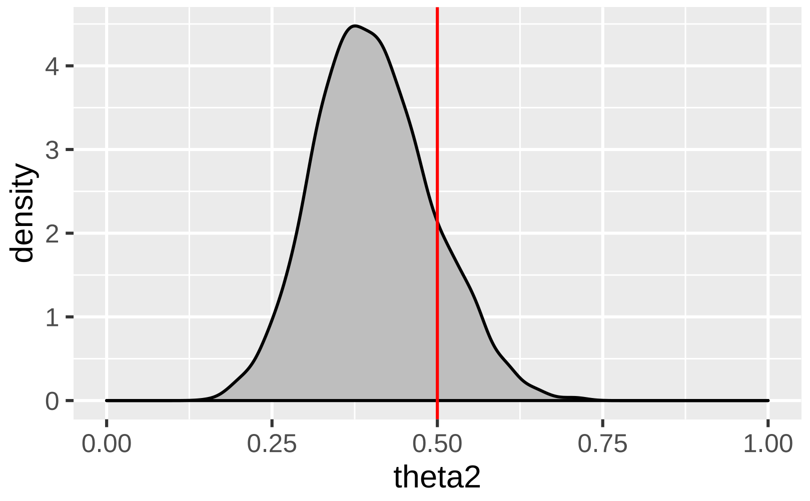
Priors



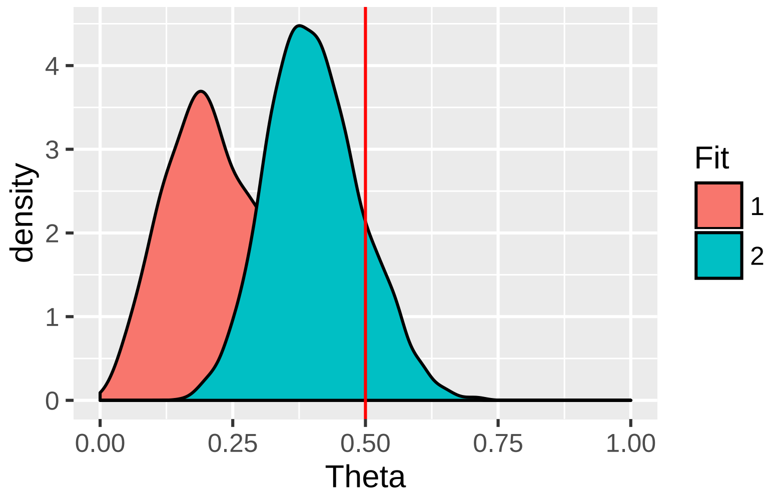
No idea if the coin is fair



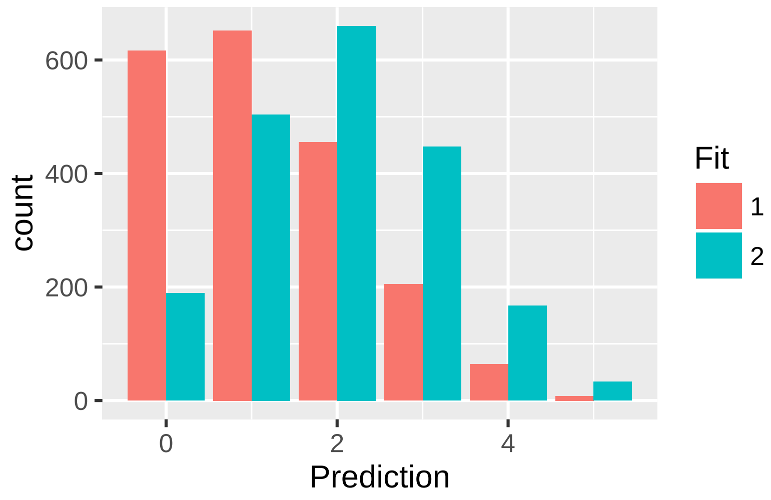
Pretty sure *the coin is fair*



Both assumptions



Future predictions



Notice that the mean of the generated thetas is always between the sample mean of 0.2 and our prior belief of 0.5. In the first case, it's 0.25. In the second it's 0.4.

Simple, but potentially practical example

Is this a contrived scenario?

Consider:

- Ten years of a high excess treaty
- One year of ten comparable treaties

- For late run-off claims, probability that a claim will close in the next year

Bayesian Estimation of Individual Claims

Note

This model was based on an example first described in (Gelman and Hill 2006). The stan model code may be found here: `https://github.com/stan-dev/example-models/blob/master/ARM/Ch.8/roaches.stan`

Data

```
data {
  // sample data
  int<lower=0> numClaims;
  vector[numClaims] Prior;
  int<lower=0, upper=1> BadCredit[numClaims];
  int Current[numClaims];

  // prior parameters
  real shape;
  real rate;

  // New predicted quantities
  int<lower=0> numNewClaims;
  vector[numNewClaims] NewPrior;
  int<lower=0, upper=1> NewBadCredit[numNewClaims];
}
```

Transformed Data

```
transformed data {
  vector[numClaims] logPrior;
  vector[numNewClaims] logNewPrior;

  logPrior <- log(Prior);
  logNewPrior <- log(NewPrior);
}
```

Because this is a Poisson GLM, we'll take the log of the predictor.

Parameters

```
parameters {
  real credit;
  real linkRatio;
}

transformed parameters {
  real logLink;
  logLink <- log(linkRatio);
}
```

We're using Poisson with an offset, so we need to transform the parameters.

Model

```
model {
  for (i in 1:numClaims) {
    linkRatio ~ gamma(shape, rate);
    Current[i] ~ poisson_log(logPrior[i]
                              + logLink + credit * BadCredit[i]);
  }
}
```

Predictions

```
generated quantities{
  int newCurrent[numNewClaims];
  for (i in 1:numNewClaims){
    newCurrent[i] <- poisson_log_rng(logNewPrior[i]
                                    + logLink + credit * NewBadCredit[i]);
  }
}
```

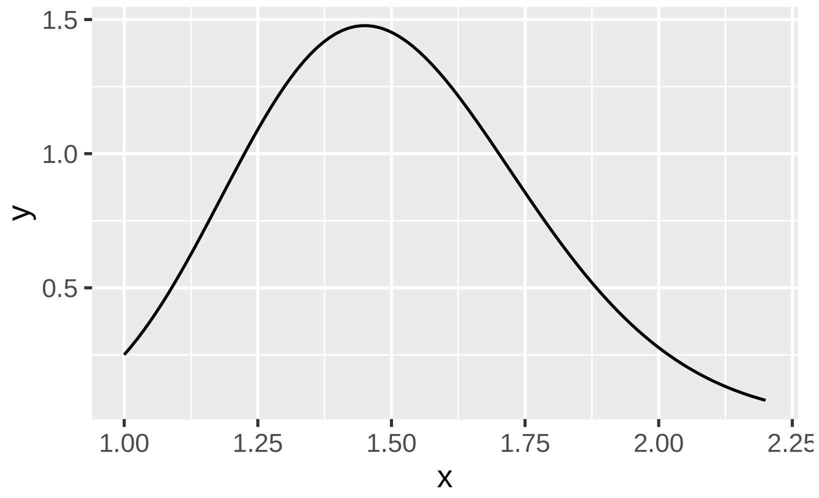
What link ratio do I expect?

I'm free to do pretty much whatever I want with the model.

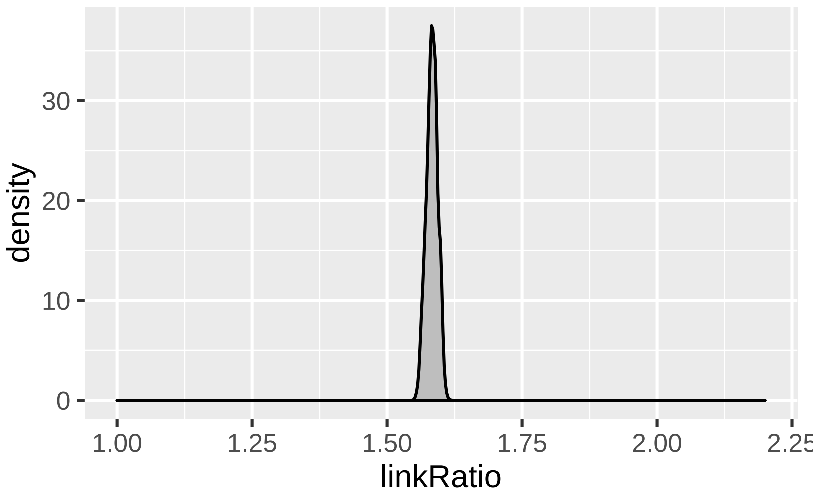
That's a feature, not a bug!

Here, I'll assume that the link ratio is something like 1.5 and I have a modest confidence interval around that guess.

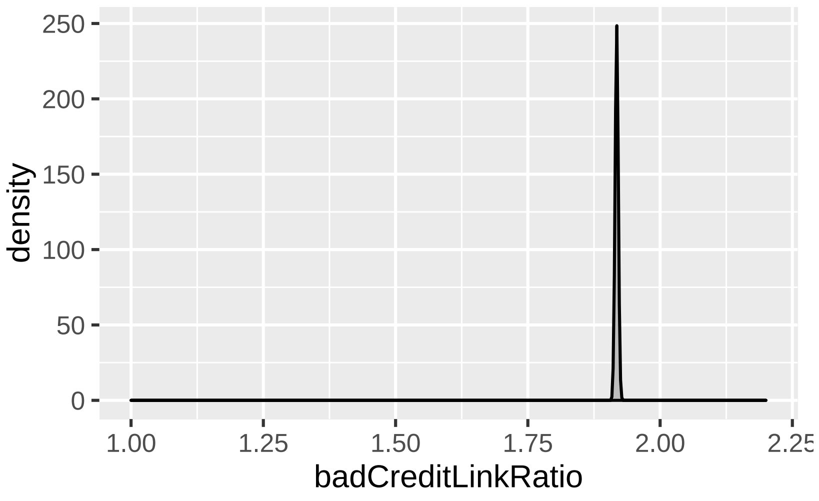
Here it is with $\alpha = 30$ and $\beta = 20$



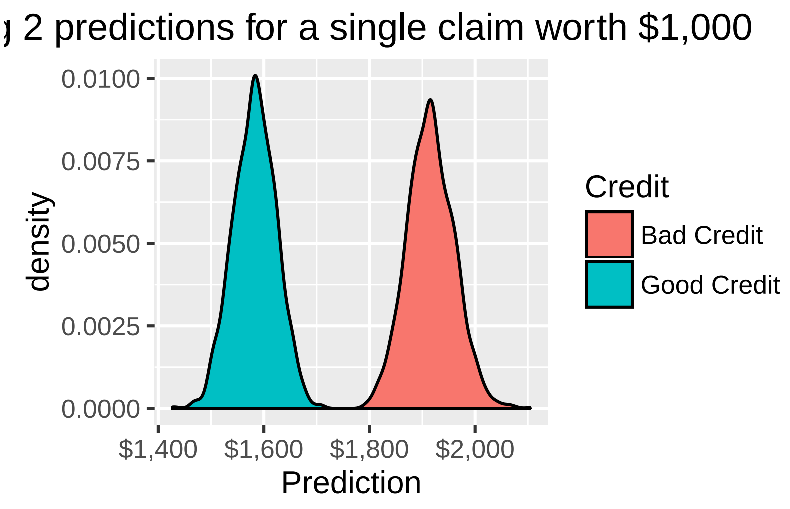
The posterior for good credit



The posterior for bad credit



Predictions for individual claims!



Summary

Summary

- Individual claims analysis presumes a model. In that respect, not so much different from aggregate techniques.
- If categories are the modelling problem, simply divide and conquer. But we lose data!
- Hierarchical models can help as segments get small, but they require complementary data.

- Bayesian techniques allow us to bring judgment to bear on small samples.

Read everything here:

`http://pirategrunt.com/CLRS2016`

References

"Example Models." 2015. `https://github.com/stan-dev/example-models/wiki`.

Gelman, Andrew, and Jennifer Hill. 2006. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. `http://www.stat.columbia.edu/~gelman/arm/`.

Guszcza, James. 2008. "Hierarchical Growth Curve Models for Loss Reserving." *Forum* Fall 2008. Casualty Actuarial Society. `https://www.casact.org/pubs/forum/08fforum/7Guszcza.pdf`.

Guszcza, James, and Jan Lommele. 2006. "Loss Reserving Using Claim-Level Data." *Forum* Spring 2006. Casualty Actuarial Society. `https://www.casact.org/pubs/forum/06fforum/115.pdf`.

Stan Development Team. 2015. *Stan Modeling Language User's Guide and Reference Manual, Version 2.10.0*. `http://mc-stan.org/`.