Insurance Data Science Conference
16-18 Jun 2021

A Novel Bayesian Pricing Model for Commercial Lines

Hierarchical expected loss-cost, frequency-severity decomposition with coupled covariance, running in Production

Jonathan Sedar - Director
Oreum Industries

Who am I?

Bayesian Data Science Consultant

Fifteen years consulting & learning from data:

- Design & deliver statistical models
- Build & lead data science teams
- Advise senior leaders & investors

Focus: Insurtech, Fintech, Startups, VC / PE

Location: Remote (Portugal < Korea < USA < UK)

Previous IDSC / R in Insurance attendee 2015, 2016, 2017, 2018

sedar.co

jonsedar

jonsedar

jonsedar

oreum.io

Overview

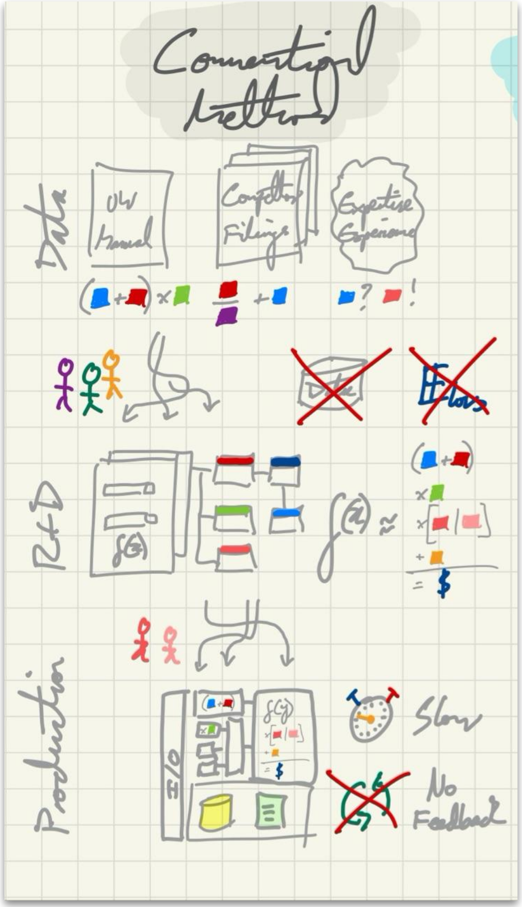
1. Problem - the need for better pricing
2. Solution - a novel Bayesian inferential model using leading software
3. Delivery - embedding a new approach into an old industry

Key technical point:

```
modern Bayesian statistical software +
domain knowledge +
proper engineering
= vital competitive advantage
```

The Problem

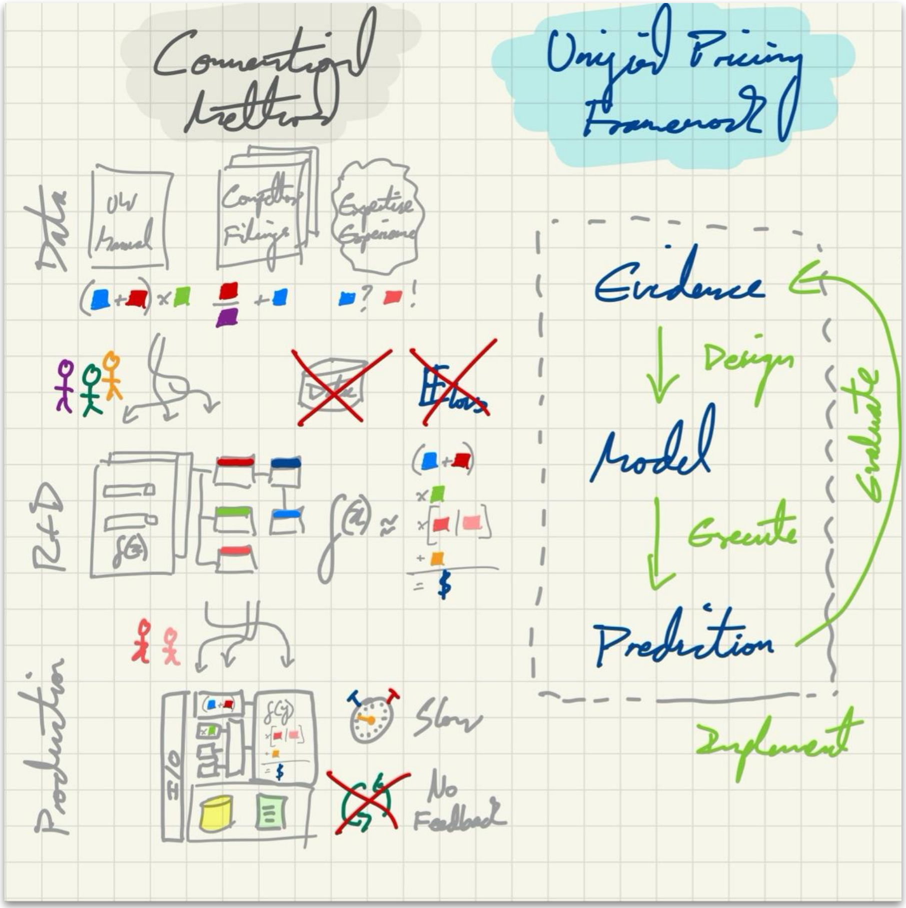
1. Tackling commercial lines with potential for extreme but infrequent losses
2. Admitted business, filed rates, can only tweak the edges
3. Pre-existing conventional model(s) based on loss ratios, unrelated to actual expected loss
4. In-house data often very relevant albeit limited, some values missing, affects the potential modelling approach
5. Historical development process conventional and convoluted --->



The Problem (general)

5. Historical development process conventional and convoluted --->

- ‘Model’ is a piecemeal recipe to construct a premium, not actually a model of loss
- Recipe inherited from unknown actuary(ies), weak provenance, can exist in multiple conflicting implementations
- Non-testable, non-reproducible, broken workflow / feedback loop
- Poor knowledge sharing, too many cooks
- Implementation slow and limited by old statistics *and* old technology



The Solution
(general)

Model workflow

Modern tech

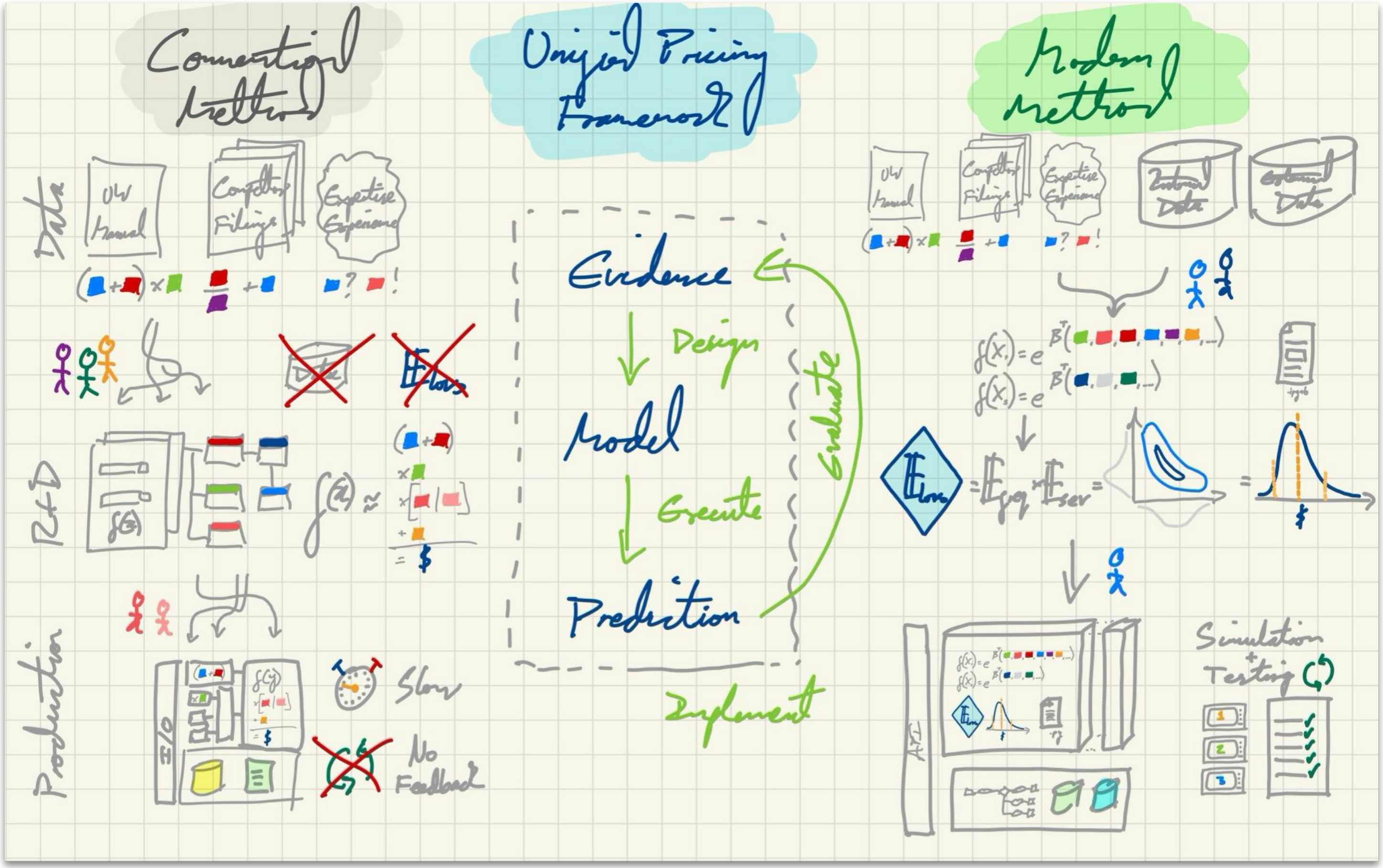
Holistic design

Testable

Reproducible

Productionisable

Multidisciplinary experts



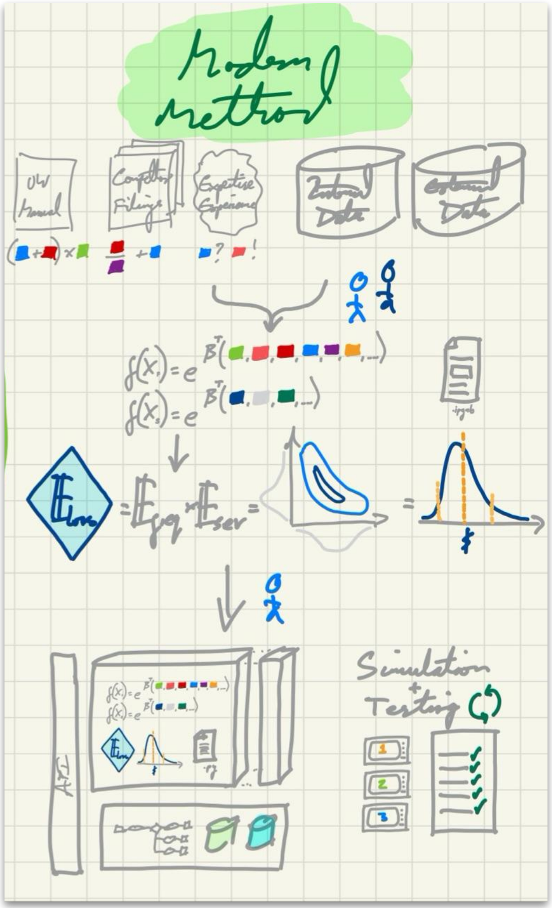
Solution is enabled by modern PPLs!

Bayesian Inference:

1. Specify a hypothesis of the data-generating process
2. Represent as a coherent model of parameterised probability distributions; incorporate prior knowledge with uncertainty; evidence with data; iterate
3. Evaluate model explanatory power with simulation and summary statistics

Probabilistic Programming Languages (PPLs):

- Coherent statistical & software framework
- Enable entire Bayesian Modelling Workflow
- Ready to integrate with Production



Solution is enabled by modern PPLs and frameworks

PyMC3

PYRO

TensorFlow

ArviZ

PyTorch

turing.jl julia

NumPy

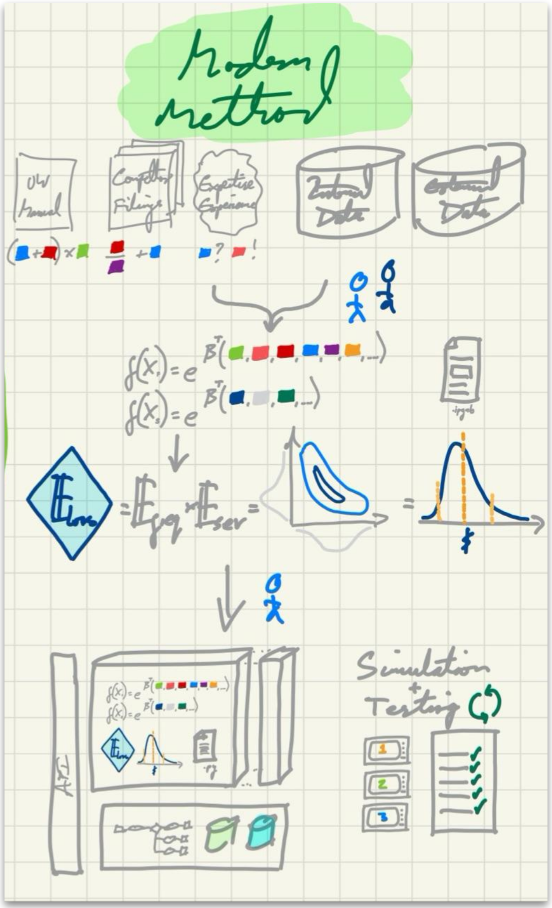
jupyter

FastAPI

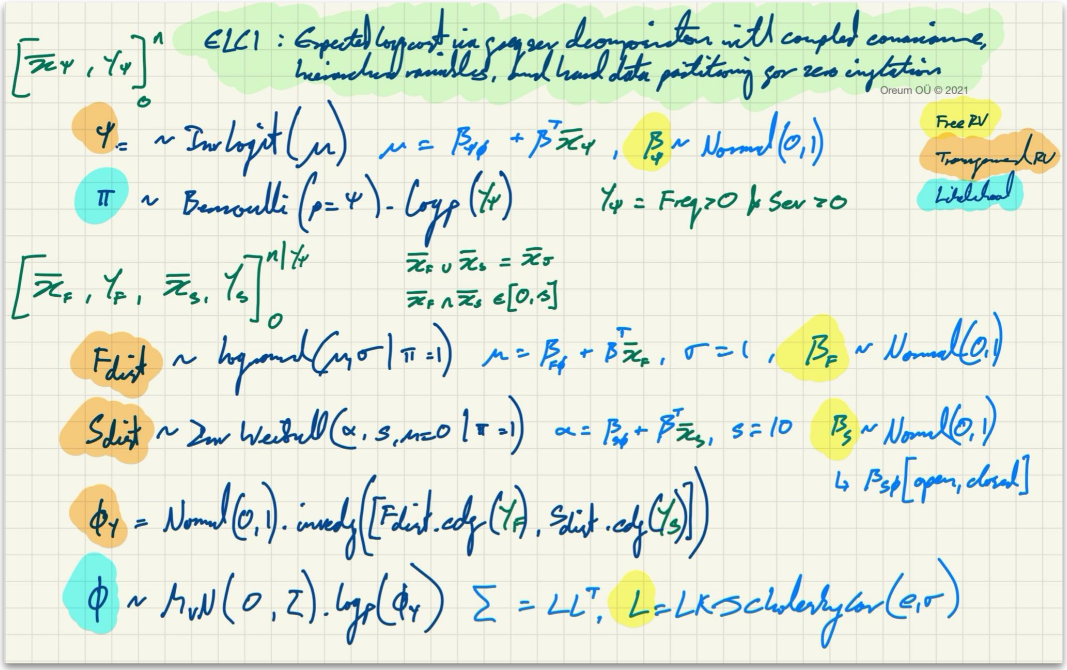
NUMFOCUS
OPEN CODE = BETTER SCIENCE

Modelling Approach

1. **Design:**
   a. Expected loss for each policy via freq-sev decomposition
   b. Choose distributional forms for the likelihoods and priors
   c. Parameterise to the available features
2. **Implement & Evaluate:**
   a. Coherent `pymc3` model and workflow
   b. Deal with zero-inflation, missing values, hierarchical priors (mixed random effects), covariance, recency-bias
   c. Refine and narrow initial uncertainty evidenced on the data to make predictions with quantified uncertainty
3. **Productionise:**
   a. Modular design, API embedding, extensibility
   b. Offline training, online prediction (`numpy`)



Model Design: Expected Loss Cost as Freq x Sev



$$freq_i = \frac{n\_claims_i}{\$\_exposure_i}, \text{ for } i \in n\_policies$$

$$sev_i = \frac{\$\_loss_i}{n\_claims_i}, \text{ for } i \in n\_policies$$

$$E\_loss_i = freq_i * sev_i = \frac{\$\_loss_i}{\$\_exposure_i}$$

Model Design: Choose Reasonable Marginals

Estimate Freq NonZero

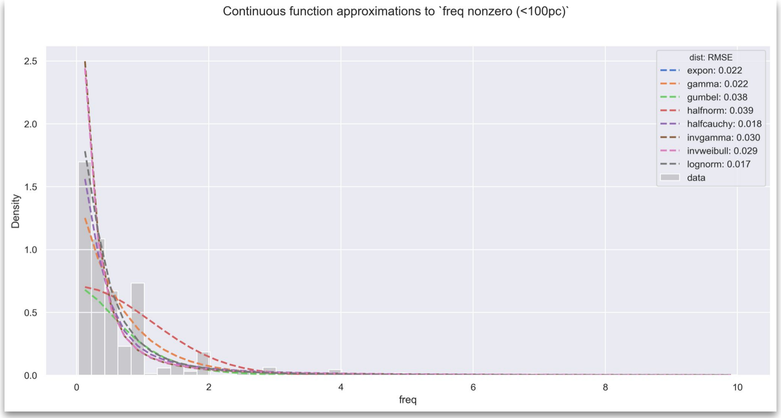
$$\mu_f \sim \mathrm{Normal}(\mu = \beta^T \vec{x}_y^{jf}, \sigma)$$
$$\sigma_f \sim \mathrm{InverseGamma}(\alpha, \beta)$$
$$\mathfrak{freq}_y \sim \mathrm{Lognormal}(\mu = \mu_f, \sigma = \sigma_f)$$

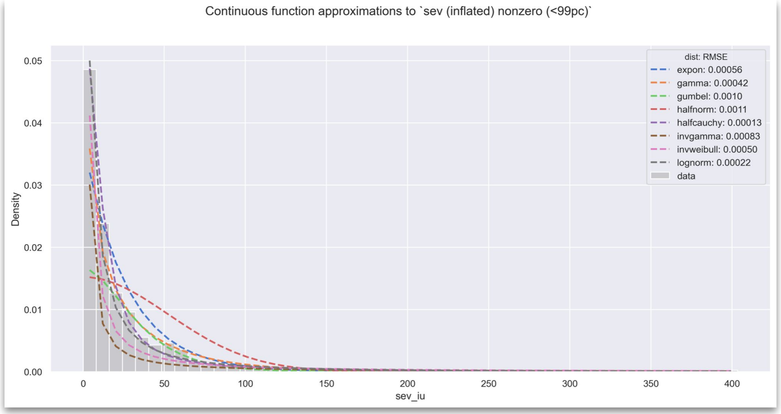
Estimate Sev NonZero

$$\mu_s \sim \mathrm{Normal}(\mu = \beta^T \vec{x}_y^{js}, \sigma)$$
$$\sigma_s \sim \mathrm{InverseGamma}(\alpha, \beta)$$
$$\mathfrak{sev}_y \sim \mathrm{Lognormal}(\mu = \mu_s, \sigma = \sigma_s)$$

Alternative Estimate Sev NonZero (extreme values)

$$\log \alpha_s \sim \mathrm{Normal}(\mu = \beta^T \vec{x}_y^{js}, \sigma)$$
$$s_s \sim \mathrm{InverseGamma}(\alpha, \beta)$$
$$\mathfrak{sev}_y \sim \mathrm{InverseWeibull}(\alpha = \alpha_s, s = s_s, m = 0)$$





Model Design: Estimate Covariance via Copula

1. Create covariance:

$$L \sim \text{LKJCholesky}(2),\ \ R \sim \text{LKJCorr}(2)$$
$$\sigma \sim \text{InverseGamma}(\alpha, \beta)$$
$$\Sigma \sim LL^T = diag(\sigma) * R * diag(\sigma)$$

2. Transform marginals via their CDFs:

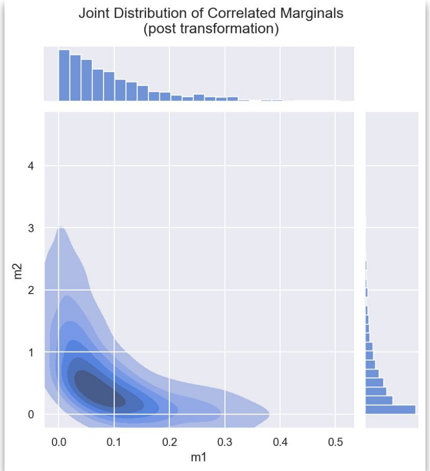
$$\mathbf{FreqU}_y = \mathfrak{freq}_y \Phi(\mathbf{Freq}_y)$$
$$\mathbf{SevU}_y = \mathfrak{Sev}_y \Phi(\mathbf{Sev}_y)$$

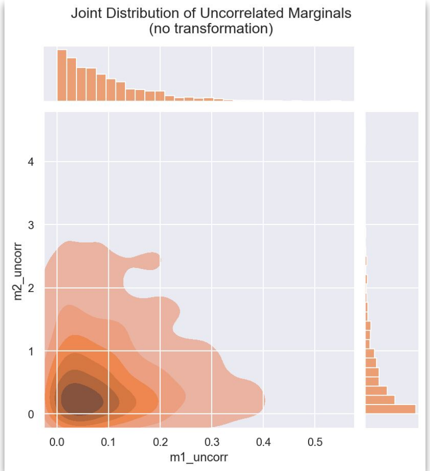
3. Transform the uniform marginals via a Normal InvCDF:

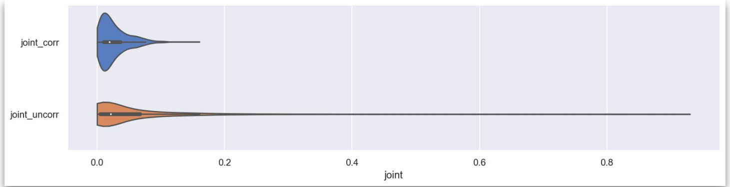
$$(\mathbf{FreqN}, \mathbf{SevN})_y = \text{MvNormal}(\mu = 0, \sigma = 1)\Phi^{-1}([\mathbf{FreqU}_y, \mathbf{SevU}_y])$$

4. Evaluate likelihood at the copula:

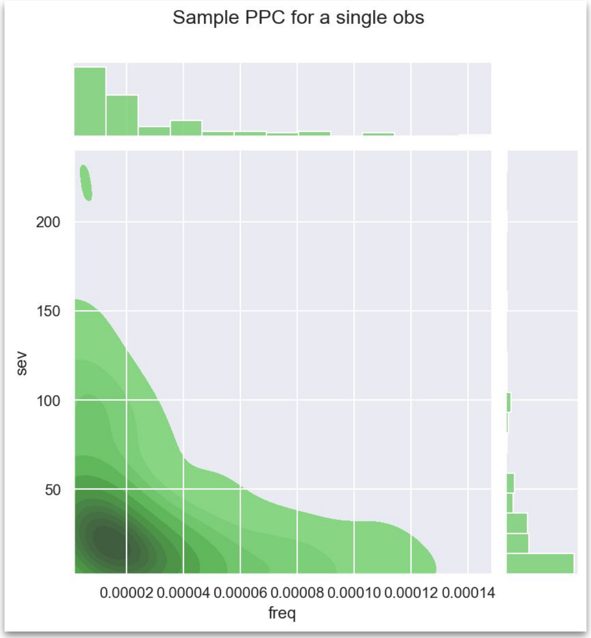
$$copula \sim \text{MvNormal}(\mu = 0, \Sigma, observed = (\mathbf{FreqN}, \mathbf{SevN})_y$$







Model Design: Result: Probabilistic Predictions of Eloss

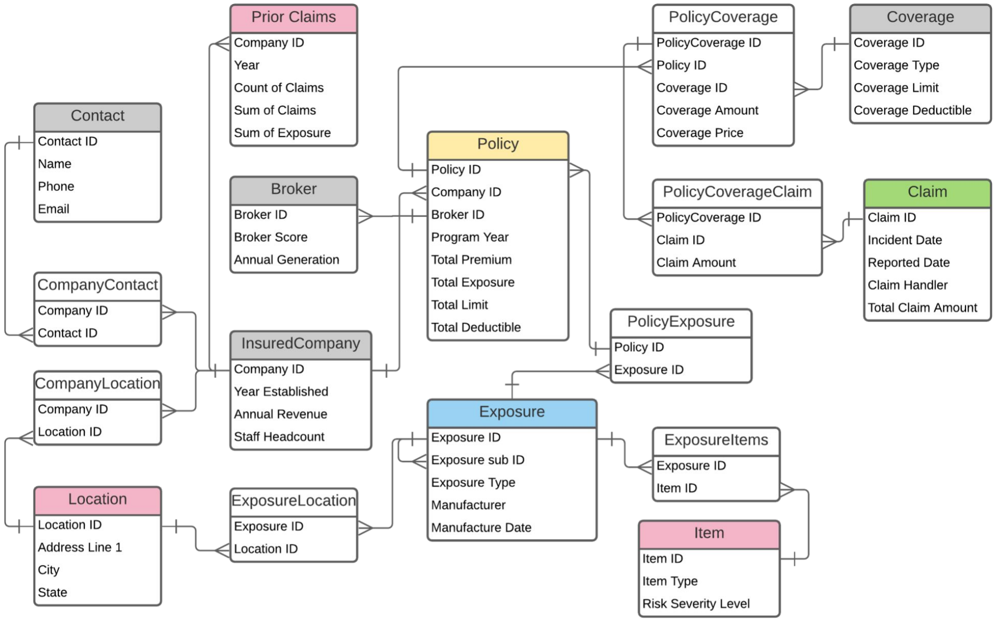


$$freq_i = \frac{n\_claims_i}{\$\_exposure_i}, \text{ for } i \in n\_policies$$

$$sev_i = \frac{\$\_loss_i}{n\_claims_i}, \text{ for } i \in n\_policies$$

$$E\_loss_i = freq_i * sev_i = \frac{\$\_loss_i}{\$\_exposure_i}$$

Model Design: Parameterise to Available Features



Example Entity Relationship Diagram (ERD)

Create features for the model:

- Data Engineering / Databasing
- Data Processing
- Feature Engineering
- Exploratory Data Analysis (EDA)
- Redesign data capture systems and question sets
- Incorporate external data

Implement: Implementation Challenges & Solutions

Zero-Inflation (freq and sev, independently)	Introduce 3rd component Psi (“is claim”) and parameterise this too $\log \frac{\psi_f}{1-\psi_f} = \mathrm{Normal}(\mu = \beta^T \vec{x}_i^{j\psi}, \sigma)$ $\pi = \mathrm{Bernoulli}(p = \psi_f, \mathrm{observed} = \mathbf{Pi})$

Missing Values
(systematic or human error)
	Use `pymc3` auto-imputation of missing values with hierarchical priors for stability
	`## missing value fts (note hierarchy on missing vals not beta coefs)`
	`freq_b_mv = pm.Normal('freq_b_mv', mu=0., sigma=1., dims='nm_j_freq_mv')`
	`x_freq_mv_mu = pm.Normal('x_freq_mv_mu', mu=0., sigma=1., dims='nm_j_freq_mv')`
	`x_freq_mv = pm.Normal('x_freq_mv', mu=x_freq_mv_mu, sigma=1.,`
	`observed=xma_freq_mv, dims=['obs_id', 'nm_j_freq_mv'])`

High Cardinality
(locations, goods types, etc)
	Use hierarchical priors (mixed random effects) or carefully aggregate factors
	`## hierarchical intercept on location,`
	`psi_loc_mu = pm.Normal('psi_loc_mu', mu=-1., sigma=1)`
	`psi_loc_sigma = pm.InverseGamma('psi_loc_sigma', alpha=101., beta=100.)`
	`# centered parameterisation (non-centered yields lower ESS)`
	`psi_loc = pm.Normal('psi_loc', mu=psi_loc_mu, sigma=psi_loc_sigma, dims='nm_loc')`

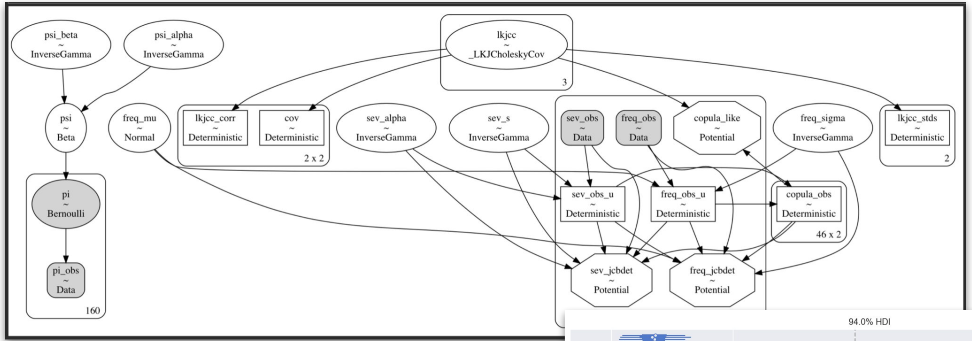
Claims Inflation Drift
(historical guesswork accumulates errors)
	Introduce an in-model parameter for recency bias, to de-weight older program years
	`## bernoulli likelihood inc recency bias`
	`pi_dist = pm.Bernoulli.dist(p=psi)`
	`pi_like = pm.Potential('pi_like',`
	`pi_dist.logp(y_pi) *`
	`x_psi_recency_bias)`

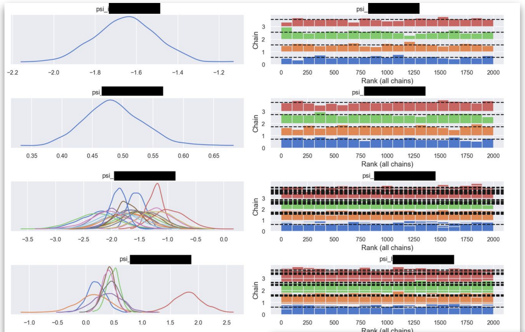
Consistent data transformation
(Production same as R&D)
	Use `patsy` formula transforms with custom mods for factors, and use self-contained reusable Python package for data curation, eda, model framework & handling: `oreum_core`
	`## patsy linear model desc:`
	`1 + F(location) + is_renewal + np.log(veh_age) ...`

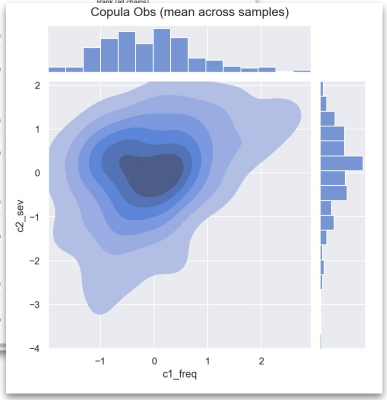
Implement: Iteratively Build & Evaluate Model Architecture

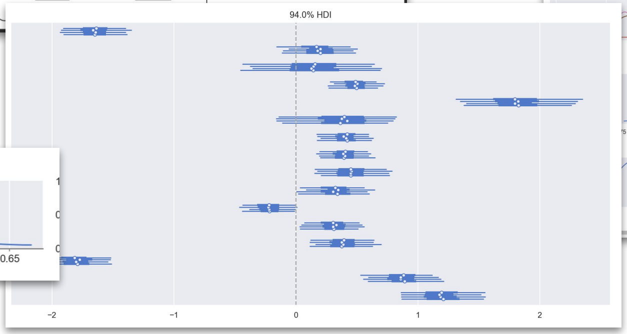
Examples from a Bayesian Workflow:

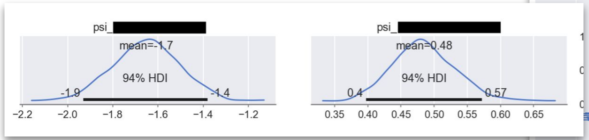
- *Plate Notation, Posterior Traceplots, HDI, Forestplots, etc*







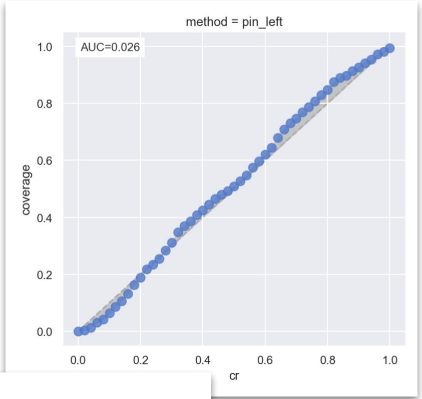


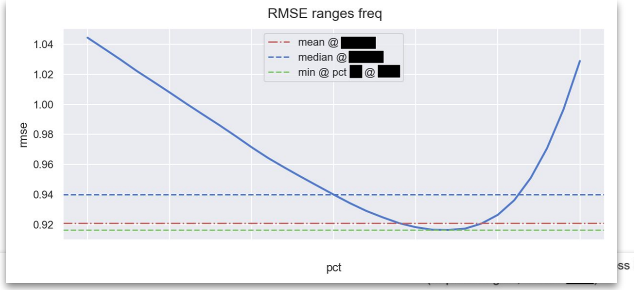


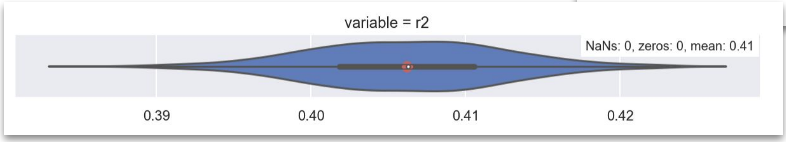
Implement: Evaluate Model Performance

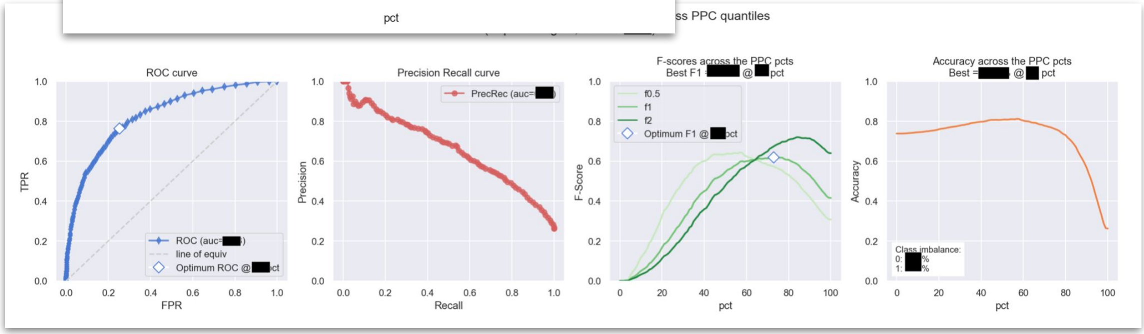
Example Posterior Predictive Perf, measures from a Bayesian Workflow:

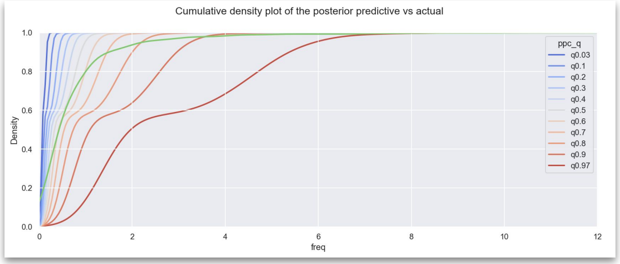
- *Binary: ROC, Prec Red, F-Score, Accuracy*
- *Continuous: Bayesian R^2, posterior density, coverage, RMSE, etc*







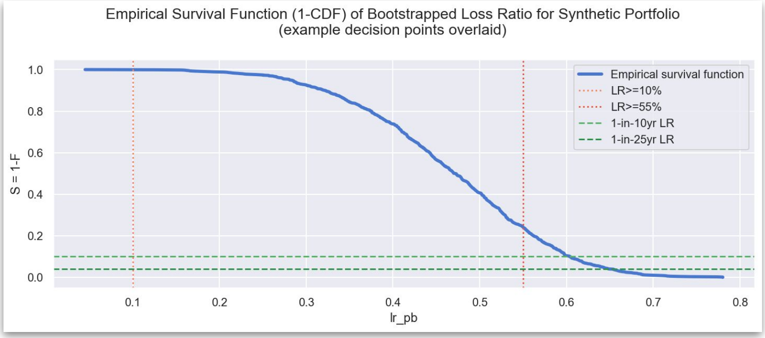


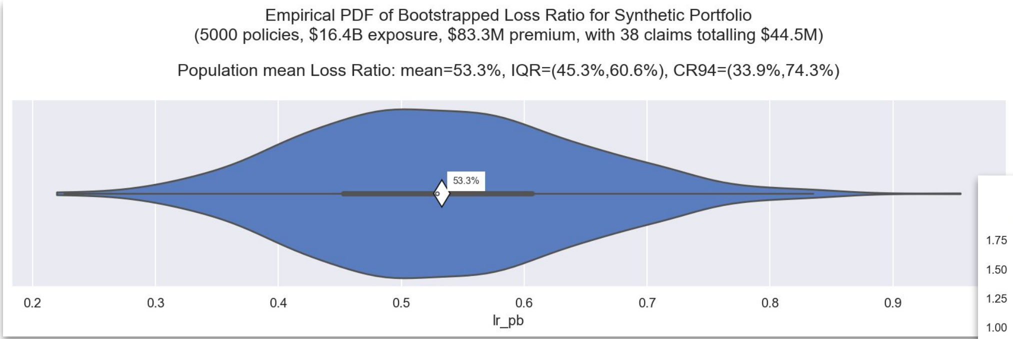


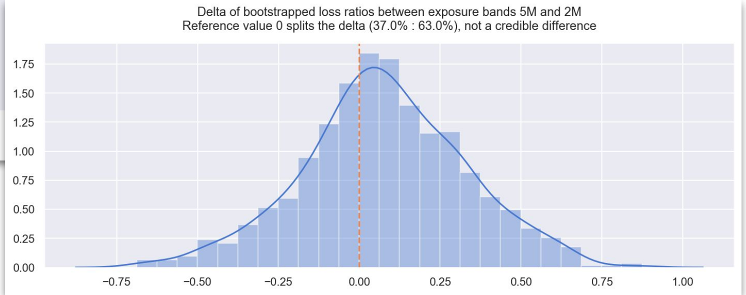
Model Evaluation: Evaluate Holistic / Conventional Measures

Example Holistic / conventional performance measures

- *Portfolio loss ratios (bootstrapped)*
- *Exceedance curve*
- *2-sample bootstrap hypothesis testing*

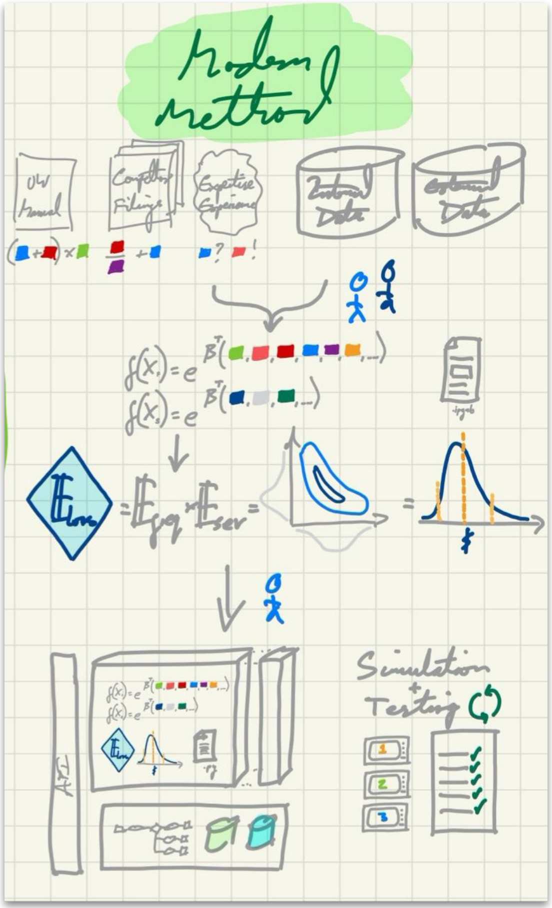






Production Implementation

1. Offline training, online prediction
2. Modular construction: data transformers, vectorised model forward-pass, PPC calcs
3. Online stage re-written in `numpy` for practicality
4. Integrate into a hosted API service (e.g. `FastAPI`)
5. Monitor model performance and undertake regular structured evaluation, ensure model delivers value



PyMC3 ArviZ NumPy

jupyter FastAPI

Recap

1. Problem - the need for better pricing
2. Solution - a novel Bayesian inferential model using leading software
3. Delivery - embedding a new approach into an old industry

Key technical point:

```
modern Bayesian statistical software +
domain knowledge +
proper engineering
= vital competitive advantage
```

oreum

INDUSTRIES