Supervised individual loss reserving

Based on Antonio & Plat (2014)

Philipp Ratz

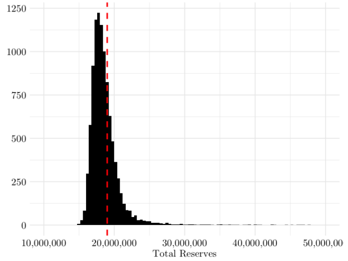
Université du Québec à Montréal (UQAM), Montréal (Québec), Canada
Séminaire en statistique

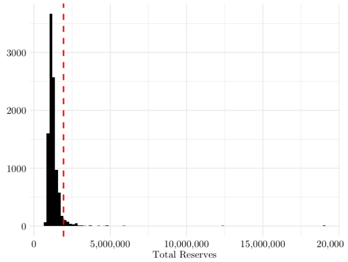
December 9, 2021

Table of Contents

(Not only my) Issues

- Quite clear that more data should improve the fit
- In the simulated data, the different business lines are not hugely dissimilar, their prediction accuracies are
- We are in the micro-level world but still use somewhat aggregated data. How do we actually forecast? How do we know we are on the right path?
- In practice the exercise is often done more frequently (eg. Switzerland has negative interest rates)





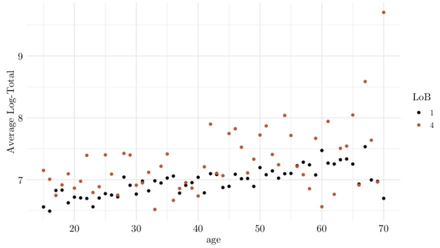
Data evidence

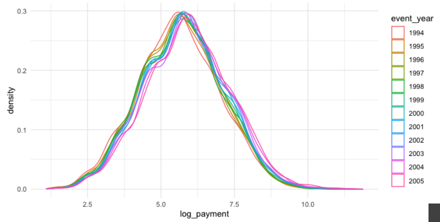
Research Question

Are claim developments individually predictable?

- If so, can we profit from some knowledge transfer between business lines?

- Recall from the last presentation that this is similar to the approach presented there, now just on the individual level
- The simulated data only have a few features, if it works here, it should work even better on more granular data





The Idea: Using the supervision

- The Microlevel approach is based on large amounts of data, but the original Model of Antonio & Plat restricts its usage
- Machine Learning helps to comb through the data. Instead of fully specifying the model, we can start with some loose intuition and let the data decide the final model. By evaluating its performance at each timestep we introduce *supervision*
- Instead of assuming that losses come from a specified distribution, we will try predict the settlement process directly
- The basemodel of Antonio & Plat contains several sub-models (ie. each part of the equation (5) in the paper that leaves the question

Where to apply?

- Reporting Delay?
- Occurence Process?
  - Additionally, time-series/stochastic processes are notoriously hard to predict with machine learning (see eg. [Makridakis et al., 2020]:
  - "[...] As was mentioned earlier, all ML methods submitted were less accurate than the Comb benchmark, and only one was more accurate than the Naïve 2 ")
  - "[...] the most accurate ML method was less accurate than the worst statistical ones."
- Development Process? Antonio & Plat use hazard rates to model *conditional* probabilities but do so in an aggregated fashion. Since our data is less precise we could change that slightly.
- Payments? Really one of the "classics" of Machine Learning

Problem statement

Instead of predicting the next event, we want to predict whether the next event is the last one.

$$\text{status}_{t+1} = f(\text{status}_t, \text{covariates}, \text{history}_t) \tag{1}$$

For the given data that brings an advantage, as it allows to take into account claims that are settled within eg. a year. The same goes for the payment amount:

$$\text{payment}_{t+1} = f(\text{status}_t, \text{covariates}, \text{history}_t) \tag{2}$$

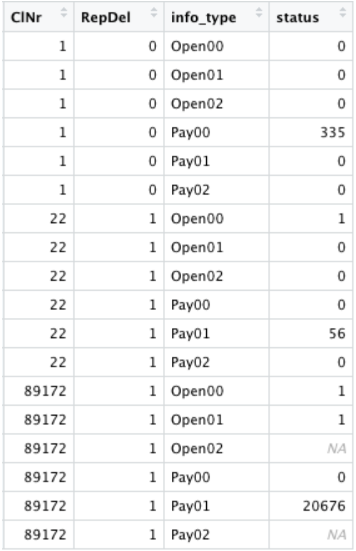
Note that a given payment can be zero. Also note that both predictions are conditional on the claim being reported, this will be treated later. Finally we get:

$$\mathcal{R}_{RBNS} = \sum_{i=1}^{N} \sum_{j=1}^{11} \text{payment}_{i,j} \mathbb{1}_{status_j = 1} \tag{3}$$

Note that here the RBNS is for *all* lines of business!

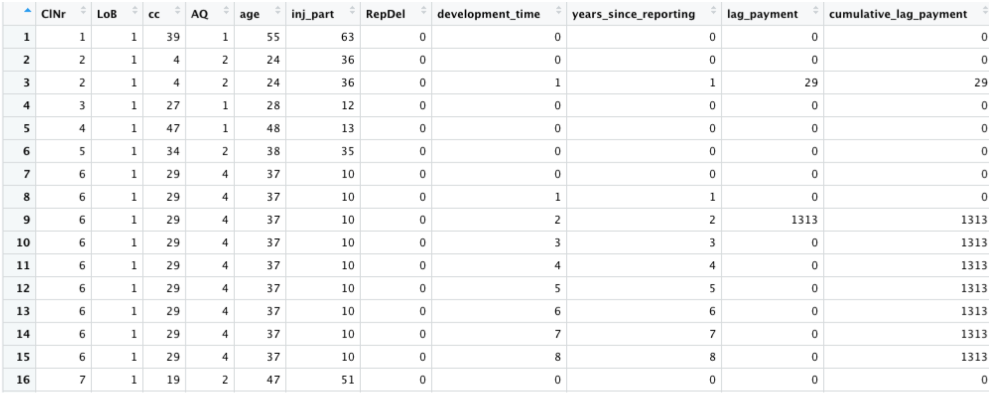
Data Formatting - I

- Standard Panel approaches will most likely fail, because we have different individuals, different (nonrandom) observation times, different time-periods and to top it all of also (nonrandom) censoring
- Instead we try to implement the temporal dependence by hand (which is summarised in the "history" parameter of equations 2 and 1



ClNr	RepDel	info_type	status
1	0	Open00	0
1	0	Open01	0
1	0	Open02	0
1	0	Pay00	335
1	0	Pay01	0
1	0	Pay02	0
22	1	Open00	1
22	1	Open01	0
22	1	Open02	0
22	1	Pay00	0
22	1	Pay01	56
22	1	Pay02	0
89172	1	Open00	1
89172	1	Open01	1
89172	1	Open02	NA
89172	1	Pay00	0
89172	1	Pay01	20676
89172	1	Pay02	NA

Data Formatting - II



	ClNr	LoB	cc	AQ	age	inj_part	RepDel	development_time	years_since_reporting	lag_payment	cumulative_lag_payment
1	1	1	39	1	55	63	0	0	0	0	0
2	2	1	4	2	24	36	0	0	0	0	0
3	2	1	4	2	24	36	0	1	1	29	29
4	3	1	27	1	28	12	0	0	0	0	0
5	4	1	47	1	48	13	0	0	0	0	0
6	5	1	34	2	38	35	0	0	0	0	0
7	6	1	29	4	37	10	0	0	0	0	0
8	6	1	29	4	37	10	0	1	1	0	0
9	6	1	29	4	37	10	0	2	2	1313	1313
10	6	1	29	4	37	10	0	3	3	0	1313
11	6	1	29	4	37	10	0	4	4	0	1313
12	6	1	29	4	37	10	0	5	5	0	1313
13	6	1	29	4	37	10	0	6	6	0	1313
14	6	1	29	4	37	10	0	7	7	0	1313
15	6	1	29	4	37	10	0	8	8	0	1313
16	7	1	19	2	47	51	0	0	0	0	0

Gradient Boosting in 2'

Consider some value to be predicted y given some input x and a loss $\mathcal{L}$. *Gradient boosting* estimates the function f such that:

$$\hat{f} = \arg\min_{f} \mathbb{E}_{x,y}[\mathcal{L}(y, f(x))], \quad \text{for example by: } \hat{f}(x) = \sum_{i=1}^{M} \gamma_i h_i(x)$$

where h are so-called *weak learners*. It is usually done in a greedy fashion such that:

$$f(x)_m = f(x)_{m-1} + \gamma h_m(x)$$

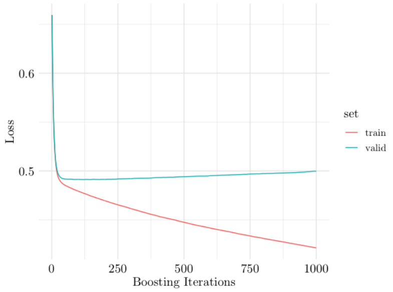
Intuitively, $f(x)_{m-1}$, our prediction at iteration m can be improved if we change its residual slightly into the right direction. The most popular way is to use a "too small" CART tree to do that[1].

[1]There is a slightly more formal explanation in the appendix

Modelling & Tuning - I

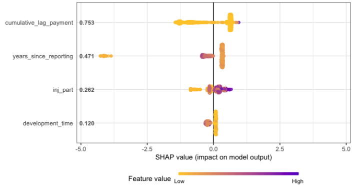
Choosing the right number of boosting iterations *M* then becomes a bias-variance trade-off choice, to which there is usually no a-priori solution

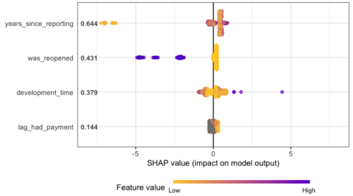
- [Bühlmann and Yu, 2003] show that for specific forms the trade-off is flatter than for most other estimators
- Given our large dataset, we can tune the boosting iterations to ensure a good problem specific fit (this is what ML is all about)
- Here I used LightGBM (My favourite Microsoft product:), usually a lot faster than XGBoost and the defacto (Swiss) Industry Standard)

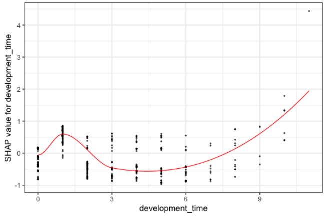


Modelling & Tuning - II

- Besides fitting well to the data, we can also gather some understanding of the data. This is done via SHAP-values (see eg. [Lundberg et al., 2019]). Intuitively, it is the average marginal contribution of a feature across all feature coalitions
- In summary there are two models, one to predict whether the claim will be settled in the next period, one predicting the payment amount (which can be zero) in the next period

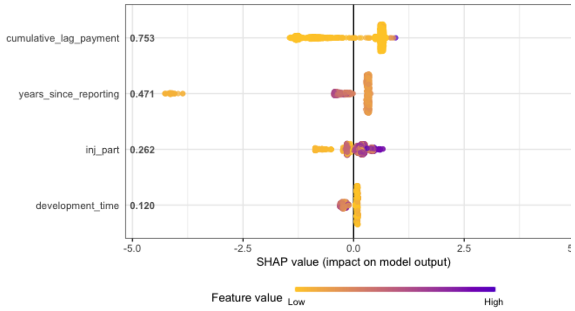




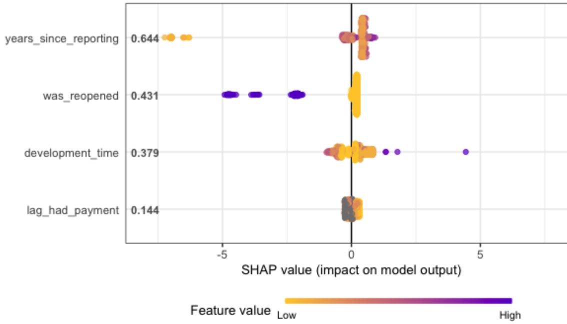


Modelling & Tuning - II (Backup)

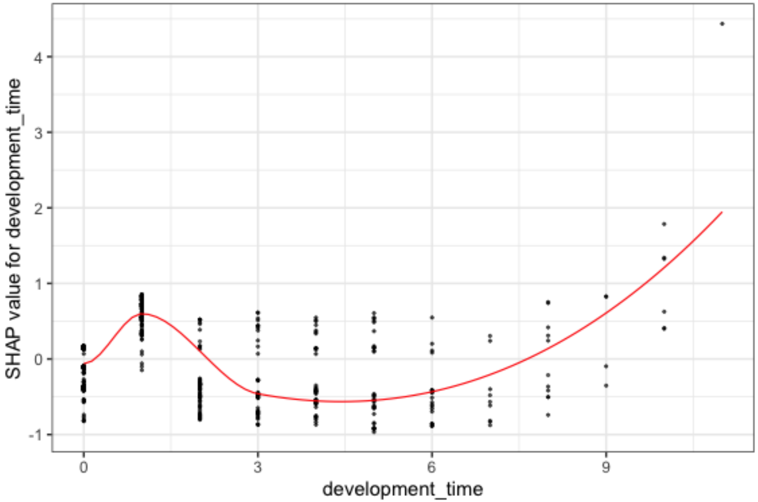
Payment regression



Status classification

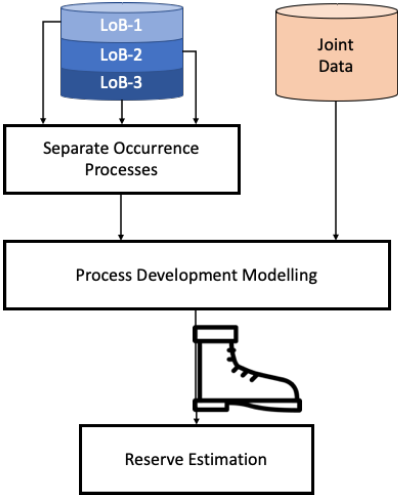


Modelling & Tuning - II (Backup)



Combination with Antonio & Plat

- Basically, the "drawing from a distribution" part of when building up reserves is replaced by individual predictions
- The approach works for RBNS reserves of any type, the IBNR will still need to be simulated. This will still be done according to their article

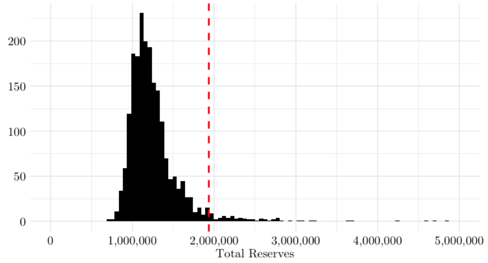


The individual reserving procedure

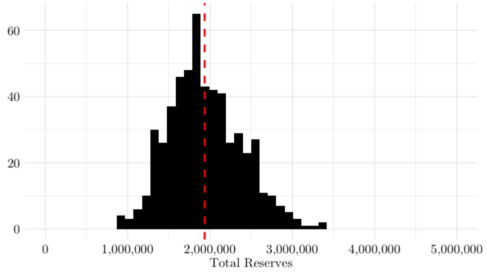
Algorithm 1 Pseudocode for individual reserves

```
1: Initialize empty payment vector, Threshold and EvalPeriod, EndPeriod, DataSet
2: while CurrentPeriod < EndPeriod do
3:     for $i = 1$ to $N_k$ do
4:         Predict closure probability $\varphi_i$ and payment $p_{k,i}$ for EvalPeriod $k$, save $p_{k,i}$ in payment vector
5:         if $\varphi$ > threshold then
               Remove observation $i$ from DataSet
6:         end if
7:     end for
8:     Update DataSet, Update CurrentPeriod += EvalPeriod
9: end while
```

Results

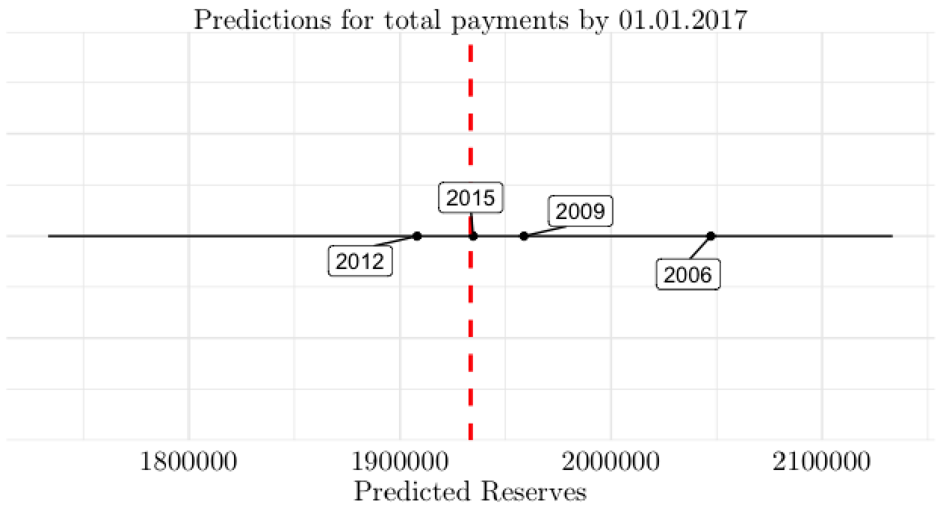


200
150
100
50
0
0
1,000,000
2,000,000
3,000,000
4,000,000
5,000,000
Total Reserves



60
40
20
0
0
1,000,000
2,000,000
3,000,000
4,000,000
5,000,000
Total Reserves

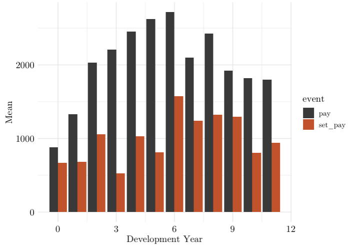
Practical Interest

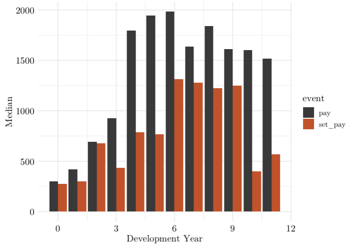


Joint estimation

- At least some evidence that the payment amount and the probability of closing a claim are linked
- Maybe we can exploit this property?
- The literature of Econometrics uses the so-called "simultaneous equation" models (Unfortunately a discussion goes beyond the scope of the presentation, but I would recommend [Kuan and White, 1994])

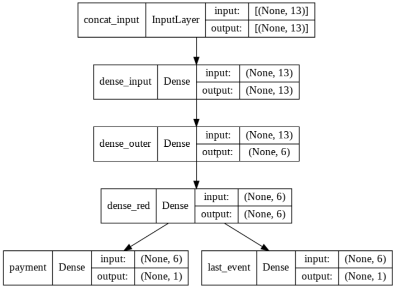
Enter Neural Networks (NN)...





Coupling in Neural Networks

- Besides (non-linear) modelling, deeplearning frameworks offer the possibilities of subtask specialisation. Modelling is not restricted to a single form, we just need something gradient based (this includes boosting!) see eg. [Finn et al., 2017]
- We can then validate on total payments
- Fitting the model is nontrival though..



concat_input | InputLayer | input: [(None, 13)] | output: [(None, 13)]
dense_input | Dense | input: (None, 13) | output: (None, 13)
dense_outer | Dense | input: (None, 13) | output: (None, 6)
dense_red | Dense | input: (None, 6) | output: (None, 6)
payment | Dense | input: (None, 6) | output: (None, 1)
last_event | Dense | input: (None, 6) | output: (None, 1)

Summary & Limitations

- The basic idea is to transform the micro into the "individual"
- Even with very limited data promising results
- Use of Deeplearning Frameworks allows joint tuning of models, (almost) no limitations as long as it is gradient based, see eg. [Finn et al., 2017]
  BUT..
- Model complexity increases drastically, especially when fitting NN. Goes against principle "which reduces accessibility of the model and transparency towards practicing actuaries" [Antonio and Plat, 2014]
- NN are known for their occasional hickups

Further Research

- Estimation is computationally rather heavy (but should not pose limitations). Frequentist approaches need further (resource hungry) boostrapping. Probabilistic approaches exists..
- Here it is a stepwise procedure, but we are not limited to that. Could imagine modelling survival directly (Kaplan-Meier) or further integrating submodels
- Risk has a special Issue on Machine Learning... :-)

References I

Antonio, K. and Plat, R. (2014).
Micro-level stochastic loss reserving for general insurance.
Scandinavian Actuarial Journal, 2014(7):649–669.

Bühlmann, P. and Yu, B. (2003).
Boosting with the l 2 loss: regression and classification.
Journal of the American Statistical Association, 98(462):324–339.

Finn, C., Abbeel, P., and Levine, S. (2017).
Model-agnostic meta-learning for fast adaptation of deep networks.
CoRR, abs/1703.03400.

Kuan, C.-M. and White, H. (1994).
Artificial neural networks: An econometric perspective.
Econometric reviews, 13(1):1–91.

Lundberg, S. M., Erion, G. G., and Lee, S.-I. (2019).
Consistent individualized feature attribution for tree ensembles.

References II

Makridakis, S., Spiliotis, E., and Assimakopoulos, V. (2020).
The m4 competition: 100,000 time series and 61 forecasting methods.
International Journal of Forecasting, 36(1):54–74.

Boosting in 2'

First, recall that given some locally differentiable multi-variable function $f(x)$ in the neighbourhood θ, $f(x)$ decreases fastest in the direction of the negative gradient. For greedy optimization:

$$\theta_m = \theta_{m-1} - \gamma \nabla f(\theta_{m-1})$$

Now, if $f_m(x)$ represents the prediction of y at iteration m we can rewrite:

$$\hat{y}_m = f_m(x) = \hat{y}_{m-1} - \gamma h_m(x)$$

the relationship now becomes clear. If we want to minimize the global loss we can replace $h_m(x)$ with the gradient of out loss at the iteration. This yields the gradient boosting equation which can then be replaced with its sample equivalent:

$$f_m(x) = f_{m-1}(x) - \gamma \nabla_{f_{m-1}(x)} \mathcal{L}(y, f_{m-1}(x))$$

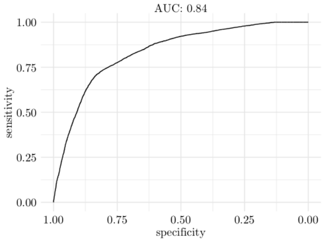
Autoregressive Nature in payments process

Table: Simple linear regression of payments

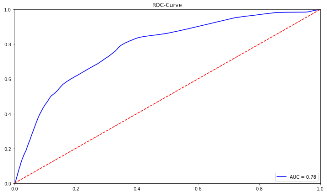
	Dependent variable:
	payment
lag_payment	0.566***
	(0.001)
Constant	$-$20.151***
	(0.862)
Observations	512,656
R^2	0.501
Adjusted R^2	0.501
Residual Std. Error	609.302 (df = 512654)
F Statistic	514,232.200*** (df = 1; 512654)
Note:	*p$<$0.1; **p$<$0.05; ***p$<$0.01

Performance indicators

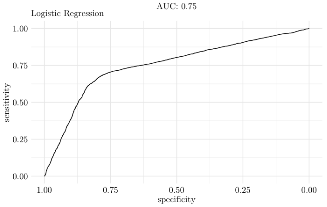
LightGBM



Tensorflow-NN

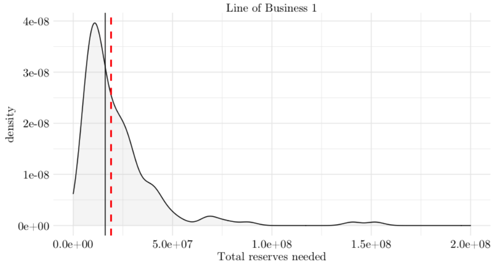


Logistic Regression

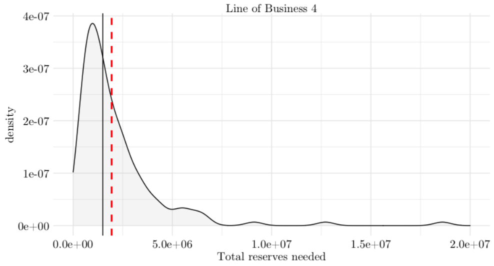


Real distribution function

LoB-1



LoB-4


Line of Business 4
density
4e-07
3e-07
2e-07
1e-07
0e+00
0.0e+00
5.0e+06
1.0e+07
1.5e+07
2.0e+07
Total reserves needed