Boosting insights in insurance tariff plans with machine learning methods

Katrien Antonio
LRisk - KU Leuven and ASE - University of Amsterdam

December 3, 2020
My personal website: https://katrienantonio.github.io
This talk is based on joint work with

Marie-Pier Côté (Laval, Canada), Roel Henckaerts, and Roel Verbelen,

who work/have worked with me at KU Leuven in the framework of the Ageas research chair on insurance analytics.
This keynote’s mission is threefold

To discuss:

(1) specific considerations to keep in mind when using machine learning methods with frequency/severity data

(2) interpretation and comparison tools for machine learning methods, with a particular focus on pricing with frequency/severity data, including different types of risk factors

(3) maidrr, our strategy to construct a Model Agnostic Interpretable Data-driven suRRrogate.
Road to Explainable AI (XAI)

- An explainable AI (XAI) algorithm enables human users to understand, trust and manage its decisions.

- Matters in highly regulated industries, such as insurance and banking.

- Two roads or pathways to XAI:
  - after the event: use interpretation tools to (better) understand decision process in black box model
  - by design: develop and use transparent white box model.
A data driven binning strategy for the construction of insurance tariff classes by Henckaerts, Antonio, Clijsters and Verbelen (2018, Scandinavian Actuarial Journal), with GitHub repo.
THE MAIN TYPES OF MACHINE LEARNING

CLASSICAL ML
- Simple data
  - Clear features

ENSEMBLES
- When quality is a real problem
  - Complicated data
    - Unclear features
      - Belief in a miracle

NEURAL NETWORKS AND DEEP LEARNING
- No data, but we have an environment to interact with
  - Eternal competitors

REINFORCEMENT LEARNING

Taken from Machine learning for everyone. In simple words. With real-world examples. Yes, again.
Let’s dive into:

**Boosting insights in insurance tariff plans with tree-based machine learning methods**, by Roel Henckaerts, Marie-Pier Côté, Katrien Antonio and Roel Verbelen (2020, North American Actuarial Journal),

with reproducible examples in notebooks on GitHub:

- tree-based ML
- severity modeling.
GIVE A LOAN?

CREDIT HISTORY

bad

HAVE A PLEDGE

no

no

NOPE

yes

yes

GUARANTORS?

no

NOPE

yes

YES

HAVE A DEBT > $1000

no

YES

NOPE

yes

YES

DECISION TREE
Regression trees

- The process of building a regression tree with CART (Breiman et al., 1984):
  1. divide the predictor space into $J$ distinct, non-overlapping regions $R_1, R_2, \ldots, R_J$
    - top-down, greedy strategy with recursive binary splitting
  2. for every observation in region $R_j$ we make the same prediction:
    - the mean of the response values for the training observations in $R_j$.

- The prediction obtained with a regression tree:
  \[
  f_{\text{tree}}(X_1, \ldots, X_p) = \bar{y}_1 I_{\{X \in R_1\}} + \ldots + \bar{y}_J I_{\{X \in R_J\}},
  \]
  where $\bar{y}_j = \text{ave} (y_i | X_i \in R_j)$. 

Regression trees

Tree pruning

- **Prune** a large tree by minimizing:

\[
\sum_{j=1}^{J} \sum_{i: x_i \in R_j} L(y_i, \hat{y}_{R_j}) + J \cdot cp \cdot \sum_{i: x_i \in R} L(y_i, \hat{y}_R)
\]

- \(cp = 0\) gives biggest possible tree
- \(cp = 1\) gives root tree without splits.

- We employ a **tuning strategy and search grid** to find the optimal value for \(cp\), e.g. via cross-validation.
Regression trees
Example of a frequency tree

MTPL data set analyzed in Henckaerts et al. (2020, NAAJ).
Loss functions inspired by GLMs

Frequency

- classic: GLM with count distribution (e.g. Poisson or NegBin)
- ML: use Poisson deviance as loss function

\[
D(y, \hat{f}(x)) = 2 \sum_{i=1}^{n} \left( y_i \cdot \ln \frac{y_i}{\hat{f}(x_i)} - (y_i - \hat{f}(x_i)) \right)
\]

Severity

- classic: GLM with skewed distribution (e.g. Gamma or LogNorm)
- ML: use Gamma deviance as loss function

\[
D(y, \hat{f}(x)) = 2 \sum_{i=1}^{n} w_i \cdot \left( \frac{y_i - \hat{f}(x_i)}{\hat{f}(x_i)} - \ln \frac{y_i}{\hat{f}(x_i)} \right)
\]
MAKE DIFFERENT SETS OF DATA FROM INITIAL SET

SAME ALGORITHMS

BAGGING ON TREES || RANDOM FOREST

JUST AVERAGING ALL THE RESULTS

ANSWER

BAGGING
Bagging

- We generate $T$ different bootstrapped data sets $\{D_t\}_{t=1,...,T}$ from the training data $D$.

- We train our method on the $t$-th bootstrapped training set and get $\hat{f}_{\text{tree}}(x|D_t)$. Finally, we average all the predictions

$$\hat{f}_{\text{bagg}}(x) = \frac{1}{T} \sum_{t=1}^{T} \hat{f}_{\text{tree}}(x|D_t).$$

This is called bootstrap aggregating (or bagging) and goes back to Breiman (1996).

- With random forests each time a split in a tree is considered, a at random $m$ out of $p$ predictors are chosen as split candidates (Breiman, 2001).
INITIAL DATASET

TREE

NEW SET OF DATA
(priority for cases on which the previous algorithm was mistaken)

WOW, ANOTHER TREE!

COLLECT BAD CASES AGAIN

GUESS WHO IS HERE AGAIN?

REPEAT UNTIL IT FEELS GOOD

BOOSTING
Boosting trees

- Fitting trees in a forward stagewise procedure, solve:

\[
\hat{\Theta}_t = \arg \min_{\Theta_t} \sum_{i=1}^{n} L(y_i, f_{t-1}(x_i) + f_{\text{tree}}(x; \Theta_t)),
\]

where \( \Theta_t = \{R_{jt}, b_{jt}\}_{1}^{J_t} \), the regions and fitted values of the tree.

- With squared-error loss:

  fit a regression tree to the current residuals \( y_i - f_{t-1}(x_i) \).

- With other loss functions, this idea generalizes to pseudo-residuals.
The gradient boosting algorithm

- The gradient tree-boosting algorithm (Friedman, 2001):
  - initializes to the optimal constant model, which is just a single terminal node tree
  - fits a small tree of depth $d$ to the pseudo-residuals $\rho_{it} = -\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}$ evaluated at current model fit $f_{t-1}$ (more details in the paper)
  - a shrinkage parameter $\lambda$ controls the learning speed by shrinking updates
    \[ f_{\text{new}}(x) = f_{\text{old}}(x) + \lambda \cdot \text{update}. \]

- **Stochastic gradient boosting** injects randomness in the training process by subsampling the data at random without replacement in each iteration.
Tuning and comparison strategy

Stratified sampling on the MTPL data in Henckaerts et al. (2020)
Interpretation tools

- Classical statistical methods are highly interpretable:
  - coefficients in a GLM
  - smooth effects in a GAM.

- Not the case for machine learning methods:
  - ✓ regression trees
  - ✗ random forests
  - ✗ boosted trees.

- There is a need for interpretation tools: look under the hood!
Interpretation tools

PDPs

- (Univariate) Partial Dependence Plots (PDPs) to interpret the marginal effect of a feature on the outcome

\[
\bar{f}_\ell(x_\ell) = \frac{1}{n} \sum_{i=1}^{n} f_{\text{model}}(x_\ell, x^i_{-\ell}).
\]

- Global measure such that interaction effects can stay hidden.
Interpretation tools

PDPs
Interpretation tools

PDPs

Generalized linear model – Frequency

Regression tree – Frequency

Generalized additive model – Frequency

Gradient boosting machine – Frequency
Interpretation tools

ICEs

- Individual conditional expectation plots (ICEs)
  \[ \tilde{f}_{\ell,i}(x_{\ell}) = f_{\text{model}}(x_{\ell}, x^i_{-\ell}). \]

- ICEs show the effect of a variable on an individual level:
  - to picture the uncertainty of the effect of a variable on the prediction outcome
  - to detect interaction effects.
The gray lines are ICEs for 1000 random policyholders and the blue line shows the partial dependence curve.
Hunting for interaction effects

Friedman’s $H$-statistic:

$$H_{k\ell}^2 = \frac{\sum_{i=1}^{n} \left\{ \overline{f}_{kl}(x_k^{(i)}, x_\ell^{(i)}) - \overline{f}_k(x_k^{(i)}) - \overline{f}_\ell(x_\ell^{(i)}) \right\}^2}{\sum_{i=1}^{n} \overline{f}_{kl}^2(x_k^{(i)}, x_\ell^{(i)})}.$$ 

<table>
<thead>
<tr>
<th>Variables</th>
<th>$H$-statistic</th>
<th>Variables</th>
<th>$H$-statistic</th>
<th>Variables</th>
<th>$H$-statistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>(lat, long)</td>
<td>0.2687</td>
<td>(agec, coverage)</td>
<td>0.1185</td>
<td>(bm, power)</td>
<td>0.0800</td>
</tr>
<tr>
<td>(fuel, power)</td>
<td>0.1666</td>
<td>(ageph, power)</td>
<td>0.1062</td>
<td>(ageph, lat)</td>
<td>0.0799</td>
</tr>
<tr>
<td>(agec, power)</td>
<td>0.1319</td>
<td>(ageph, bm)</td>
<td>0.0961</td>
<td>(agec, ageph)</td>
<td>0.0785</td>
</tr>
<tr>
<td>(ageph, sex)</td>
<td>0.1293</td>
<td>(power, sex)</td>
<td>0.0829</td>
<td>(long, sex)</td>
<td>0.0732</td>
</tr>
<tr>
<td>(coverage, long)</td>
<td>0.1203</td>
<td>(fuel, long)</td>
<td>0.0828</td>
<td>(agec, bm)</td>
<td>0.0678</td>
</tr>
</tbody>
</table>
PDPs to picture interaction effects

K. Antonio, KU Leuven & UvA
Conclusion:

- Poisson deviance for frequency: GBM > GLM > RF > CART
- gamma deviance for severity: GBM ≈ GLM ≈ RF ≈ CART.
Model comparison tools

Findings: model lift

- We combine frequency and severity into a (technical) tariff or risk premium.
- We compare the GLM, GAM, decision tree, random forest and GBM constructed tariffs.
- Managerial tools:
  - total loss vs. total premiums
  - loss ratio lift, double lift, Gini index.
- Conclusion: GBM > GLM > RF > CART.
How about using the GBM to inform feature engineering for a GLM?

In fact, the GBM could be replaced by any ML method (RF, NN, etc.).

We developed maidrr, see the working paper on arxiv by Henckaerts, Antonio and Côté.
maidrr: Model-Agnostic Interpretable Data-driven suRRogate

1. Group values of feature $x_j$ based on univariate PD $\bar{f}_j(x_j)$ ⇒ use dynamic programming (DP) algo for clustering

2. Find relevant interactions $x_a$ and $x_b$ based on $H$-statistic

3. Cluster similar $\bar{f}_{a,b}(x_a, x_b)$ ⇒ use DP

4. Fit GLM on segmented features.
Values of feature $x_j$ with a similar PD show a similar relation to the prediction target:

- group values/levels of $x_j$ based on the univariate PD $\bar{f}_j(x_j)$

- let $m_j$ denote the unique number of observed values for $x_j$

- let $x_{j,q}$ denote its $q$th value for $q \in \{1, \ldots, m_j\}$ and define $z_{j,q} = \bar{f}_j(x_{j,q})$

- allocate elements of $m_j$ dimensional input vector to $k_j$ clusters by minimizing within-cluster sum of squares.

Adjacency constraints can be imposed (e.g. for ordinal variables).
We choose the number of groups \((k_j)\) for feature \(x_j\) via a penalized loss function:

\[
\frac{1}{m_j} \sum_{q=1}^{m_j} (z_{j,q} - \tilde{z}_{j,q})^2 + \lambda_{\text{marg}} \cdot \log(k_j)
\]

with \(\tilde{z}_{j,q}\) the average PD effect for the group to which value/level \(x_{j,q}\) belongs.

\(\lambda_{\text{marg}}\) is a tuning parameter, independent of \(j\).
The interaction between features $x_a$ and $x_b$ is captured by subtracting both one-dimensional PDs from the two-dimensional PD:

$$\bar{f}_{a,b}(x_a, x_b) = \frac{1}{n} \sum_{i=1}^{n} f_{\text{pred}}(x_a, x_b, x_i^{a,-b}) - \frac{1}{n} \sum_{i=1}^{n} \sum_{\ell \in \{a, b\}} f_{\text{pred}}(x_{\ell}, x_i^{\ell}).$$

DP algorithm without adjacency constraint allows to cluster similar $\bar{f}_{a,b}(x_a, x_b)$ values.

Optimal number of groups is again chosen using a penalized loss, with separate tuning parameter $\lambda_{\text{intr}}$ for the interaction effects.
Comparison tools
Other surrogates, accuracy, local interpretations

The paper reports our findings on a benchmark study with 6 insurance data sets.

Decision tree (DT) surrogate

- original data as features and the GBM predictions as target
- maximum tree depth restricted to four.

Linear model (LM) surrogate

- original data as features and the GBM predictions as target.
Comparison tools
Other surrogates, accuracy, local interpretations

<table>
<thead>
<tr>
<th></th>
<th>ausprivauto</th>
<th>bemtpl</th>
<th>frempl</th>
<th>fremtpl</th>
<th>norauto</th>
<th>pricingame</th>
<th>avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLM</td>
<td>0.10</td>
<td>0.49</td>
<td>1.80</td>
<td>0.92</td>
<td>0.03</td>
<td>0.48</td>
<td>0.64</td>
</tr>
<tr>
<td>LM</td>
<td>0.22</td>
<td>1.15</td>
<td>18.39</td>
<td>6.35</td>
<td>0.07</td>
<td>2.53</td>
<td>4.79</td>
</tr>
<tr>
<td>DT</td>
<td>0.25</td>
<td>1.68</td>
<td>4.82</td>
<td>2.66</td>
<td>0.28</td>
<td>2.13</td>
<td>1.97</td>
</tr>
</tbody>
</table>

\[ \Delta D^{Poi} = 100 \times \left( \frac{D^{Poi} \{ y, f_{surro}(x) \}}{D^{Poi} \{ y, f_{gbm}(x) \}} - 1 \right). \]
Comparison tools
Other surrogates, accuracy, local interpretations

<table>
<thead>
<tr>
<th></th>
<th>ausprivauto</th>
<th>bemtpl</th>
<th>frempl</th>
<th>fremtpl</th>
<th>norauto</th>
<th>pricingame</th>
<th>avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLM</td>
<td>0.86</td>
<td>0.94</td>
<td>0.91</td>
<td>0.78</td>
<td>0.99</td>
<td>0.93</td>
<td>0.90</td>
</tr>
<tr>
<td>LM</td>
<td><strong>0.89</strong></td>
<td>0.83</td>
<td>0.62</td>
<td>0.30</td>
<td>0.95</td>
<td>0.88</td>
<td>0.75</td>
</tr>
<tr>
<td>DT</td>
<td>0.75</td>
<td>0.74</td>
<td>0.88</td>
<td>0.75</td>
<td>0.84</td>
<td>0.76</td>
<td>0.78</td>
</tr>
</tbody>
</table>

\[
R^2 = 1 - \frac{\sum_{i=1}^{n} \{ f_{\text{surro}}(x_i) - f_{\text{gbm}}(x_i) \}^2}{\sum_{i=1}^{n} \{ f_{\text{gbm}}(x_i) - \mu_{\text{gbm}} \}^2}.
\]
Comparison tools
Other surrogates, accuracy, local interpretations

(a) GBM: high risk

(b) GBM: medium risk

(c) GBM: low risk

(d) GLM: high risk

(e) GLM: medium risk

(f) GLM: low risk
The mindset of the actuary - research ambition

The narrative must be that actuaries are entering the data science world not entirely to compete but also to bring the element of the actuarial profession where we build integrity and transparency into any work that we do, and how documentation of that is possible.

Quote from What data science means for the future of the actuarial profession, British Actuarial Journal, June 2018.
R packages

R packages developed by Roel Henckaerts (as part of his PhD):

- for distRforest see https://henckr.github.io/distRforest/
- for maidrr see https://henckr.github.io/maidrr
References and acknowledgements

- For detailed list of references, please consult the papers.

- Visuals are from

- More interpretation tools available in the (online) book by Christophe Molnar, see https://christophm.github.io/interpretable-ml-book/.

- automatic feature selection and binning of risk factors via regularization (i.e. lasso and friends)
- R package smurf on CRAN
- end product is a GLM!
Appendix
GLMs and GAMs for telematics insurance pricing

Unravelling the predictive power of telematics data in car insurance pricing by Verbelen, Antonio & Claeskens (2018, JRSS C)

- black box collected data on group of young drivers
- compositional data (‘parts of a whole’) on kilometers driven across road types, time slots
- GAMs for claim frequencies, with specific attention to effects of compositional data and interpretation.
initialize fit to the optimal constant model: $f_0(x) = \arg \min_b \sum_{i=1}^n \mathcal{L}(y_i, b)$;

for $t = 1, \ldots, T$ do

randomly subsample data of size $\delta \cdot n$ without replacement from data $\mathcal{D}$;

for $i = 1, \ldots, \delta \cdot n$ do

$\rho_{i,t} = -\left[ \frac{\partial \mathcal{L}[y_i, f(x_i)]}{\partial f(x_i)} \right]_{f=f_{t-1}}$

fit a tree of depth $d$ to the pseudo-residuals $\rho_{i,t}$ resulting in regions $R_{j,t}$ for $j = 1, \ldots, J_t$;

for $j = 1, \ldots, J_t$ do

$\hat{b}_{j,t} = \arg \min_b \sum_{i : x_i \in R_{j,t}} \mathcal{L}[y_i, f_{t-1}(x_i) + b]$

update $f_t(x) = f_{t-1}(x) + \lambda \sum_{j=1}^{J_t} \hat{b}_{j,t} 1(x \in R_{j,t})$;

$\hat{f}_{\text{gbm}}(x) = f_T(x)$;

**Algorithm 2**: Procedure to build a (stochastic) gradient boosting machine.
## Appendix

### Tuning and hyper-parameters

<table>
<thead>
<tr>
<th>Model</th>
<th>Tuning parameters</th>
<th>Hyper-parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression tree</td>
<td>complexity parameter $cp$</td>
<td>$\kappa = 0.01$</td>
</tr>
<tr>
<td></td>
<td>coefficient of variation gamma prior $\gamma$</td>
<td>$\kappa = 0.01$</td>
</tr>
<tr>
<td>Random forest</td>
<td>number of trees $T$</td>
<td>$cp = 0$, $\gamma = 0.25$</td>
</tr>
<tr>
<td></td>
<td>number of split candidates $m$</td>
<td>$\kappa = 0.01$, $\delta = 0.75$</td>
</tr>
<tr>
<td>Gradient boosting machine</td>
<td>number of trees $T$</td>
<td>$\lambda = 0.01$</td>
</tr>
<tr>
<td></td>
<td>tree depth $d$</td>
<td>$\kappa = 0.01$, $\delta = 0.75$</td>
</tr>
</tbody>
</table>
Appendix

maidrr algorithm in pseudo code

Algorithm 1 maidrr

Input: data, $f_{\text{pred}}$, $\lambda_{marg}$, $\lambda_{intr}$, $k$ and $h$

for $j = 1$ to $p$ do

    calculate the PD effect $\tilde{f}_j$ via Eq. (1)

    apply the DP algorithm to feature $x_j$ with $k_j^* = \arg\min_{k_j \in \{1, \ldots, k\}}$ Eq. (2) for $\lambda = \lambda_{marg}$

    $x_j^c$ represents the grouped version of $x_j$ in categorical format with $k_j^*$ groups

end for

feature selection: $F = \{j \mid k_j^* > 1\}$

upfront interaction selection: $I = \{(l, m) \mid l \in F$ and $m \in F$ and $H(x_l, x_m) \geq h\}$

for all $(a, b)$ in $I$ do

    calculate the PD effect $\tilde{f}_{a,b}$ via Eq. (3)

    apply the DP algorithm to interaction $(x_a, x_b)$ with $k_{ab}^* = \arg\min_{k_{ab} \in \{1, \ldots, k\}}$ Eq. (2) for $\lambda = \lambda_{intr}$

    $x_{a:b}^c$ represents the grouped version of $x_{a:b}$ in categorical format with $k_{ab}^*$ groups

end for

interaction selection: $I = I \setminus \{(l, m) \mid k_{lm}^* = 1\}$

fit a GLM to the target with features $x_j^c$ for $j \in F$ and interactions $x_{a:b}^c$ for $(a, b) \in I$

Output: surrogate GLM