OptBinning: The Python optimal binning library

Guillermo Navas-Palencia

DBRS Morningstar

BBVA AI Factory - The Discussion Club

Madrid, 24th January 2023

Table of Contents

Types of binning algorithms

- Unsupervised binning techniques
  - Equal-width interval binning
  - Equal-size or equal-frequency interval binning
- Supervised binning techniques
  - Merging: Monotone Adjacent Pooling Algorithm
  - Decision trees: CART, Minimum Description Length Principle (MDLP), condition inference trees (CTREE)
- Fast and robust algorithms ✓
- Constraints ✗

Mathematical optimization framework (1/4)

Definition: The optimal binning is the optimal discretization of a variable into bins given a discrete or continuous numeric target.

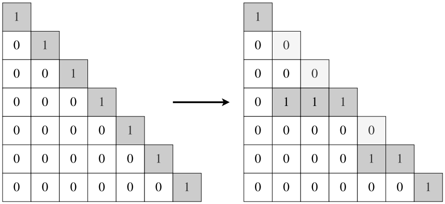
- Pre-binning
  - m split points, $s_1 < s_2 < \ldots < s_m$ to create $n = m + 1$ pre-bins. These pre-bins are defined by the intervals $(-\infty, s_1), [s_1, s_2), \ldots, [s_m, \infty)$.
- Optimization
  - Decision variable: $X_{ij} \in \{0, 1\}, \forall (i, j) \in \{1, \ldots, n : i \geq j\}$.

- Each column must contain exactly one 1. For $j = 1, \ldots, n$

$$\sum_{i=1}^{n} X_{ij} = 1.$$

- Continuity by rows, no $0 - 1$ gaps are allowed. For $i = 1, \ldots, n;\ j = 1, \ldots, i - 1$

$$X_{ij} \leq X_{ij+1}.$$



Mathematical optimization framework (2/4)

Objective function with binary and multi-class target, maximize a divergence measure (Jeffrey's divergence). Given non-event ($y = 0$) and event ($y = 1$)

$$J(P||Q) = IV = \sum_{i=1}^{n} (p_i - q_i) \log\left(\frac{p_i}{q_i}\right), \quad p_i = \frac{r_i^{NE}}{r_T^{NE}}, \quad q_i = \frac{r_i^E}{r_T^E}.$$

$$V_{ij} = \left(\sum_{z=j}^{i} \frac{r_z^{NE}}{r_T^{NE}} - \frac{r_z^E}{r_T^E}\right) \log\left(\frac{\sum_{z=j}^{i} r_z^{NE}/r_T^{NE}}{\sum_{z=j}^{i} r_z^E/r_T^E}\right), \quad i = 1, \ldots, n;\ j = 1, \ldots, i.$$

The IV can be computed using the described parameters and the decision variables X_{ij}, yielding

$$IV = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} \left(\frac{r_z^{NE}}{r_T^{NE}} - \frac{r_z^E}{r_T^E}\right) X_{ij}\right) \log\left(\frac{\sum_{j=1}^{i} r_j^{NE}/r_T^{NE} X_{ij}}{\sum_{j=1}^{i} r_j^E/r_T^E X_{ij}}\right).$$

Mathematical optimization framework (3/4)

Pre-compute all possible merges in linear-time

- Transform non-convex MINLP to IP
- Linearize objective function (exploit continuity constraint)

Optimal V_{ij} ($V_{i\cdot}$) by row, $i = 1, \ldots, n$:

$$V_{i\cdot} = V_{i1}X_{i1} + \sum_{j=2}^{i} V_{ij}(X_{ij} - X_{ij-1}) \Longleftrightarrow V_{ii}X_{ii} + \sum_{j=1}^{i-1}(V_{ij} - V_{ij+1})X_{ij}.$$

Objective function:

$$\sum_{i=1}^{n} V_{ii}X_{ii} + \sum_{j=1}^{i-1}(V_{ij} - V_{ij+1})X_{ij}.$$

Mathematical optimization framework (4/4)

Basic MIP formulation:

$$\max_{X} \quad \sum_{i=1}^{n} V_{ii}X_{ii} + \sum_{j=1}^{i-1}(V_{ij} - V_{ij+1})X_{ij} \tag{1a}$$

$$\text{s.t.} \quad \sum_{i=j}^{n} X_{ij} = 1, \qquad j = 1, \dots, n \tag{1b}$$

$$X_{ij} - X_{ij+1} \leq 0, \qquad i = 1, \dots, n;\ j = 1, \dots, i-1 \tag{1c}$$

$$b_{\min} \leq \sum_{i=1}^{n} X_{ii} \leq b_{\max} \tag{1d}$$

$$r_{\min} X_{ii} \leq \sum_{j=1}^{i} r_j X_{ij} \leq r_{\max} X_{ii}, \qquad i = 1, \dots, n \tag{1e}$$

$$r_{\min}^{NE} X_{ii} \leq \sum_{j=1}^{i} r_j^{NE} X_{ij} \leq r_{\max}^{NE} X_{ii}, \qquad i = 1, \dots, n \tag{1f}$$

$$r_{\min}^{E} X_{ii} \leq \sum_{j=1}^{i} r_j^{E} X_{ij} \leq r_{\max}^{E} X_{ii}, \qquad i = 1, \dots, n \tag{1g}$$

$$X_{ij} \in \{0, 1\}, \qquad \forall (i,j) \in \{1, \dots, n : i \geq j\} \tag{1h}$$

Constraints: monotonicity (1/3)

Impose monotonic trends between consecutive bins

- 6 modes: Ascending/descending, peak/valley, convex/concave
- 2 auto modes: auto and auto ascending/descending

Auto modes: Auto monotonic trend using Machine Learning algorithm to determine the trend maximizing divergence.

Integer and constraint programming formulation techniques:

- Disjoint constraints
- Big-M constraints
- Half-reified linear constraints

Constraints: monotonicity (2/3)

For $i = 1, \ldots, n; \; j = 1, \ldots, i$

$$D_{ij} = \frac{\sum_{z=j}^{i} r_z^E}{\sum_{z=j}^{i} r_z}.$$

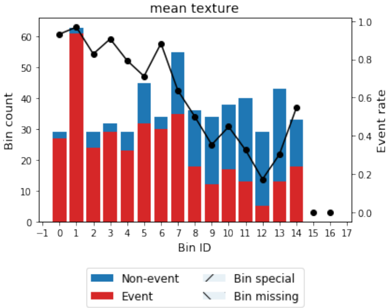
For $i = 1, \ldots, n.$

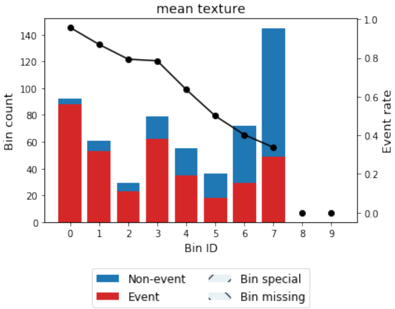
$$D_{i\cdot} = D_{ii}X_{ii} + \sum_{j=1}^{i-1}(D_{ij} - D_{ij+1})X_{ij}.$$

For $i = 2, \ldots, n; \; z = 1, \ldots i-1$

$$D_{ii}X_{ii} + \sum_{j=1}^{i-1}(D_{ij} - D_{ij+1})X_{ij} + \beta(X_{ii} + X_{zz} - 1)$$

$$\leq 1 + (D_{zz} - 1)X_{zz} + \sum_{j=1}^{z-1}(D_{zj} - D_{zj+1})X_{zj}.$$





Constraints: monotonicity (3/3)

The peak and valley trend define an event rate function exhibiting a single trend change or reversal.

$$i - n(1 - y_i) \leq t \leq i + ny_i, \qquad i = 1, \ldots, n \tag{2a}$$

$$t \in [0, n] \tag{2b}$$

$$y_i \in \{0, 1\}, \qquad i = 1, \ldots, n \tag{2c}$$

Peak trend constraints

$$y_i + y_z + 1 + (D_{zz} - 1)X_{zz} + \sum_{j=1}^{z-1}(D_{zj} - D_{zj+1})X_{zj}$$

$$\geq D_{ii}X_{ii} + \sum_{j=1}^{i-1}(D_{ij} - D_{ij+1})X_{ij}, \quad i = 2, \ldots, n;\ z = 1, \ldots, i-1,$$

$$2 - y_i - y_z + 1 + (D_{ii} - 1)X_{ii} + \sum_{j=1}^{i-1}(D_{ij} - D_{ij+1})X_{ij}$$

$$\geq D_{zz}X_{zz} + \sum_{j=1}^{z-1}(D_{zj} - D_{zj+1})X_{zj}, \quad i = 2, \ldots, n;\ z = 1, \ldots, i-1.$$

Constraints: Reduction of dominant bins

To prevent any particular bin from dominating the results and producing more homogeneous solutions.

$$\max_{X, p_{\min}, p_{\max}} \quad \sum_{i=1}^{n} V_{ii} X_{ii} + \sum_{j=1}^{i-1} (V_{ij} - V_{ij+1}) X_{ij} - \gamma (p_{\max} - p_{\min})$$

$$\text{s.t.} \quad (1b - 1h)$$

$$p_{\min} \leq r_T (1 - X_{ii}) + \sum_{j=1}^{i} r_j X_{ij}, \qquad i = 1, \ldots, n$$

$$p_{\max} \geq \sum_{j=1}^{i} r_j X_{ij}, \qquad i = 1, \ldots, n$$

$$p_{\min} \leq p_{\max}$$

$$p_{\min} \geq 0.$$

$$p_{\max} \geq 0.$$

Constraints: Maximum p-value

A necessary constraint to guarantee that event rates between consecutive bins are statistically different.

- Binary target: Z-test for proportions
- Continuous target: T-test

Use preprocessing procedure to detect pairs of pre-bins that do not satisfy the p-value constraints. Two cases are considered depending on j:

$$\begin{cases} X_{ij} + X_{kl} \leq 1 + X_{kl-1} & \text{if } j = 1, \\ X_{ij} + X_{kl} \leq 1 + X_{ij-1} + X_{kl-1} & \text{if } j > 1 \end{cases}, \quad \forall (i,j,k,l) \in \mathcal{I}.$$

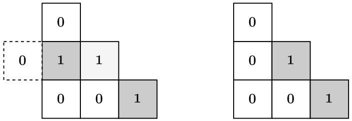
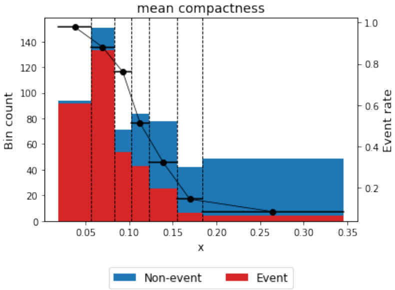


Figure: General violation constraint between two feasible solutions. Case $j = 1$ (left). Case $j > 1$ (right).

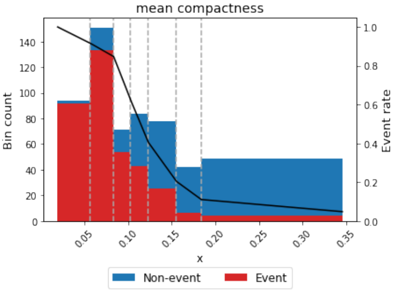
Optimal piecewise binning

- Classical binning algorithms: discontinuous piecewise constant function
- Constant value reduces the granularity of the transformation
- Discontinuities are a source of unfairness
- Good compromise between accuracy and explainability

Discontinuous piecewise constant



Continuous piecewise linear



Optimal piecewise binning: MIP formulation

Pre-binning produces mutually exclusive sets of indices satisfying

$$n = \sum_{j=1}^{B} |\mathcal{I}_j|$$

Basic MINLP formulation:

- Binary case $y = P[Y = 1|X = x]$ using ML estimator
- Polynomial degree K

$$\min \quad \sum_{j=1}^{B} \left(\sum_{i \in \mathcal{I}_j} \left| y_i - \sum_{k=1}^{K} c_{jk} x_i^{k-1} \right|^p \right)^{1/p} \tag{5a}$$

$$\text{s.t.} \quad \sum_{k=1}^{K} c_{jk} s_j^{k-1} = \sum_{k=1}^{K} c_{j+1k} s_j^{k-1}, \qquad j = 1, \ldots, B-1 \tag{5b}$$

$$l_j \leq \sum_{k=1}^{K} c_{jk} x_i^{k-1} \leq u_j, \qquad j = 1, \ldots, B, \ i \in \mathcal{I}_j \tag{5c}$$

$$c_{jk} \in \mathbb{R}, \qquad j = 1, \ldots, B, \ k = 1, \ldots, K \tag{5d}$$

Objective with $p = \{1, 2\}$ and suitable for quantile regression.

Optimal piecewise binning: additional constraints

Ascending/descending monotonicity

$$\sum_{k=1}^{K} c_{jk}(k-1)x_j^{k-2} \geq 0, \quad j = 1, \ldots, B, \ i \in \mathcal{I}_j$$
$$\sum_{k=1}^{K} c_{jk}(k-1)x_j^{k-2} \leq 0, \quad j = 1, \ldots, B, \ i \in \mathcal{I}_j$$

Regularization: avoid coefficients for large K when unnecessary.

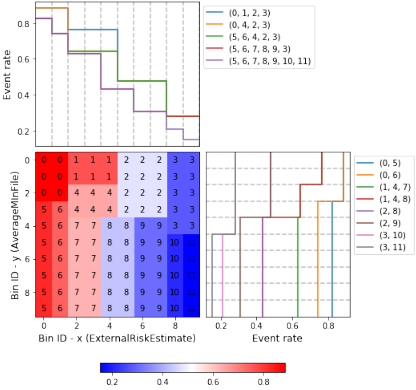
$$\min \sum_{j=1}^{B} \left(\sum_{i \in \mathcal{I}_j} \left| y_i - \sum_{k=1}^{K} c_{jk} x_i^{k-1} \right|^p \right)^{1/p} + \|c\|_{p'}$$

Implementation using SOCP solver in the RoPWR library [NP20c].

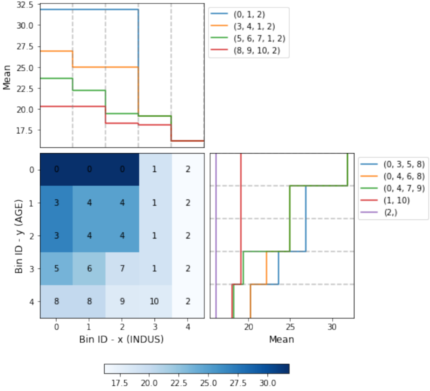
Optimal Binning 2D

Extension to 2D binning (two features x_1, x_2 and target y). Many of the 1D binning constraints are applicable, including monotonicity planes.

Binary target



Continuous target



The OptBinning library

- Python library, open source (Apache-2.0)
  - 15,231 cloc (v0.17.2) + 3,205 cloc (173 unit tests)
- GitHub: `https://github.com/guillermo-navas-palencia/optbinning`
  - Current version: 0.17.2
  - GitHub stars: 301
  - Downloads: $>$ 5.3M ($\sim$90k/month)
- Documentation: `http://gnpalencia.org/optbinning/`
- Dependencies
  - Data: NumPy, Pandas
  - Stats: SciPy, scikit-learn, RoPWR
  - Optimization: Google OR-Tools / LocalSolver (optional - commercial)
  - Plots: Matplotlib
- Functionalities
  - Optimal binning for binary, continuous and multi-class target [NP20b]
  - Optimal binning over data streams [NP20a]
  - Optimal binning with uncertainty
  - Scorecard development
  - Counterfactual explanations for scorecard models [NP21]

Bibliography

G. Navas-Palencia.
Optimal binning for streaming data.
http://gnpalencia.org/blog/2020/binning_data_streams/, 2020.

G. Navas-Palencia.
Optimal binning: mathematical programming formulation.
abs/2001.08025, 2020.

G. Navas-Palencia.
Robust piecewise regression.
https://github.com/guillermo-navas-palencia/ropwr, 2020.

G. Navas-Palencia.
Optimal counterfactual explanations for scorecard modelling.
abs/2104.08619, 2021.

Gracias!