Data Mining In Action 2016

Илья Барышников (DMIA, 2016)
141/3055 (Top 4%)

Задача

- Предсказать сумму страхового возмещения (число)
- Данные анонимизированы
- 116 признаков – категориальные (буквы)
- 14 признаков – числа [0, 1]
- Таргет числовой
- Метрика

$$MAE = \frac{1}{n}\sum_{t=1}^{n}\left|x_t - \hat{x}_t\right|$$

Данные

Категориальные признаки

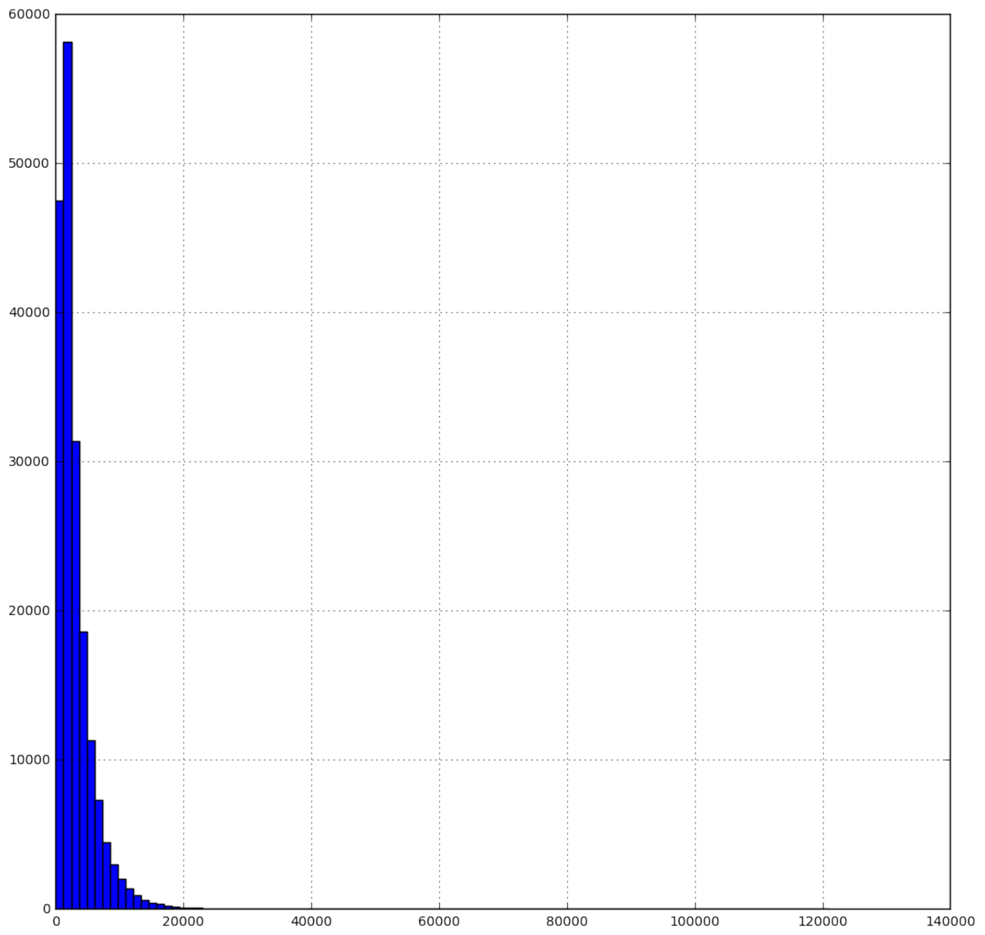
```
   id cat1 cat2 cat3 cat4 cat5 cat6 cat7 cat8 cat9 cat10 cat11    ... cat116
0   1    A    B    A    B    A    A    A    A    B     A     B     A      A
1   2    A    B    A    A    A    A    A    A    B     B     A     A      A
2   5    A    B    A    A    B    A    A    A    B     B     B     B      B
3  10    B    B    A    B    A    A    A    A    B     A     A     A      A
4  11    A    B    A    B    A    A    A    A    B     B     A     B      A
```

Числовые признаки

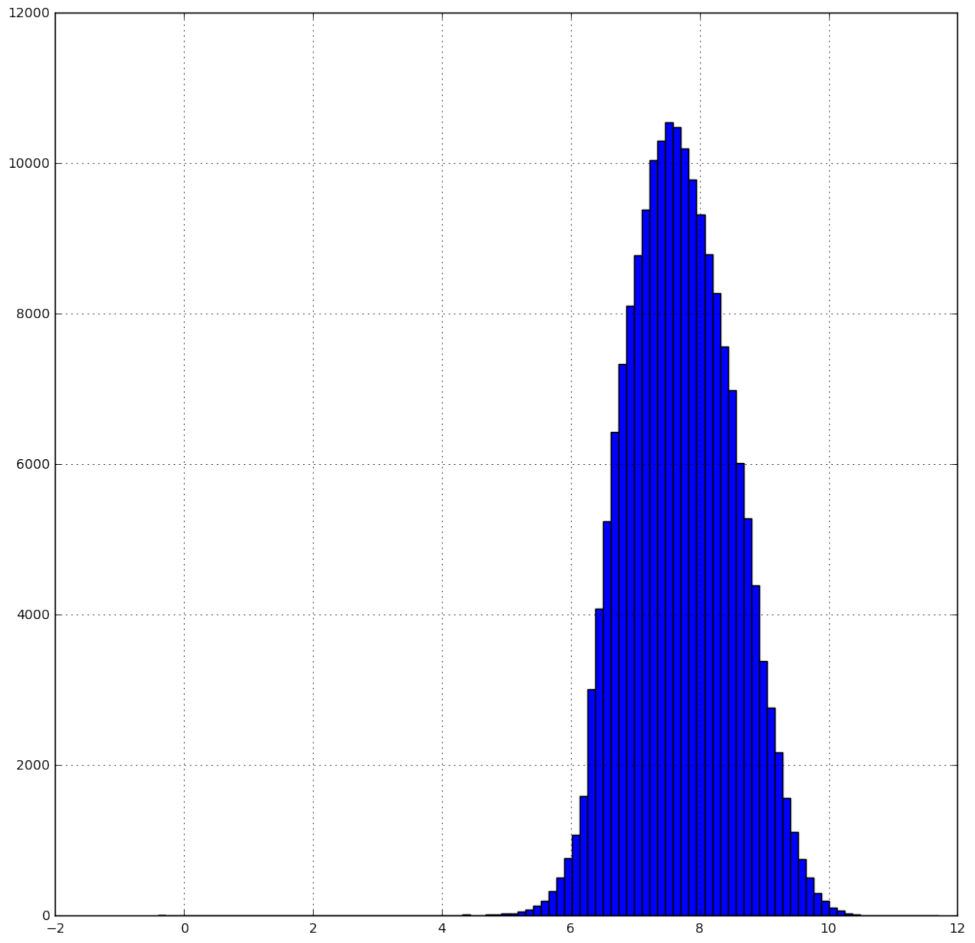
```
     cont1     cont12     cont3      ...      cont14    loss
0  0.83510  0.569745  0.594646  0.822493  0.714843  2213.18
1  0.43919  0.338312  0.366307  0.611431  0.304496  1283.60
2  0.32446  0.381398  0.373424  0.195709  0.774425  3005.09
3  0.44467  0.327915  0.321570  0.605077  0.602642   939.85
4  0.21230  0.204687  0.202213  0.246011  0.432606  2763.85
```

Данные

Таргет до трансформации



Таргет после логарифмирования



Для числовых признаков с большим skewness(скошеннсть) – Box-Cox transformation

Данные

Преобразование категориальных признаков: по формуле

$$\sum_{1}^{n}(char - 26) * 26^{n-i-1}$$

Пример:

$$A \to 1 * 26^0 = 1$$
$$Z \to 26 * 26^0 = 26$$

$$AA \to 1 * 26^1 + 1 * 26^0 = 27$$
$$AZ \to 1 * 26^1 + 26 * 26^0 = 52$$
$$BZ \to 2 * 26^1 + 26 * 26^0 = 78$$

Почему не one-hot?

Сохраняем отношение порядка в отдельных фичах и при их конкатенации

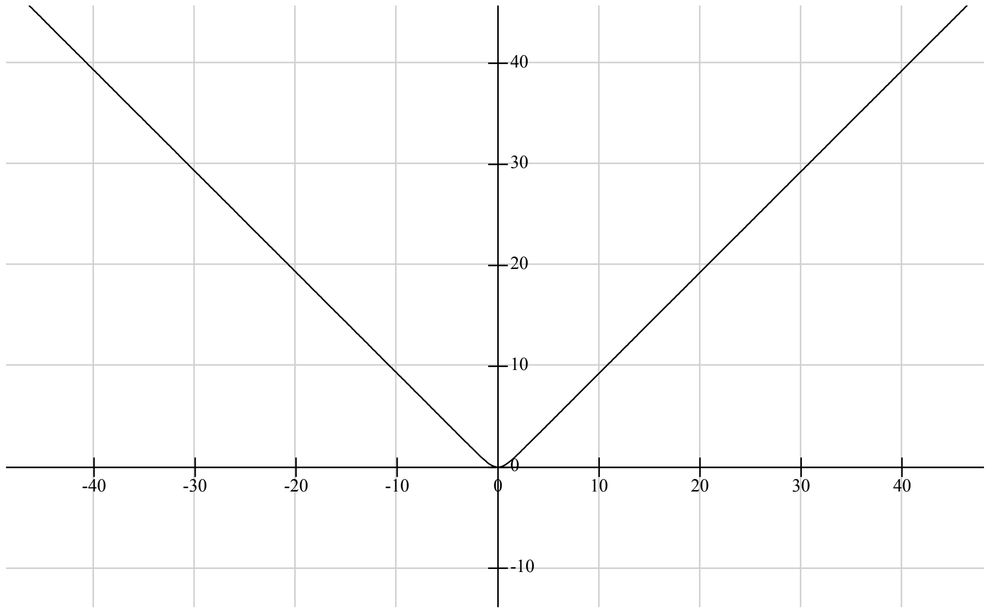
Objective

- Проблема выбора функции, чтобы минимизировать MAE (не дифференцируема в нуле)

$$\log(\cosh(x))$$

Log-Cosh Loss:

- ADVANTAGE: "Best of Both Worlds" of Squared and Absolute Loss
- Twice-differentiable
- Takes on behavior of Squared-Loss when loss is small, and Absolute Loss when loss is large.

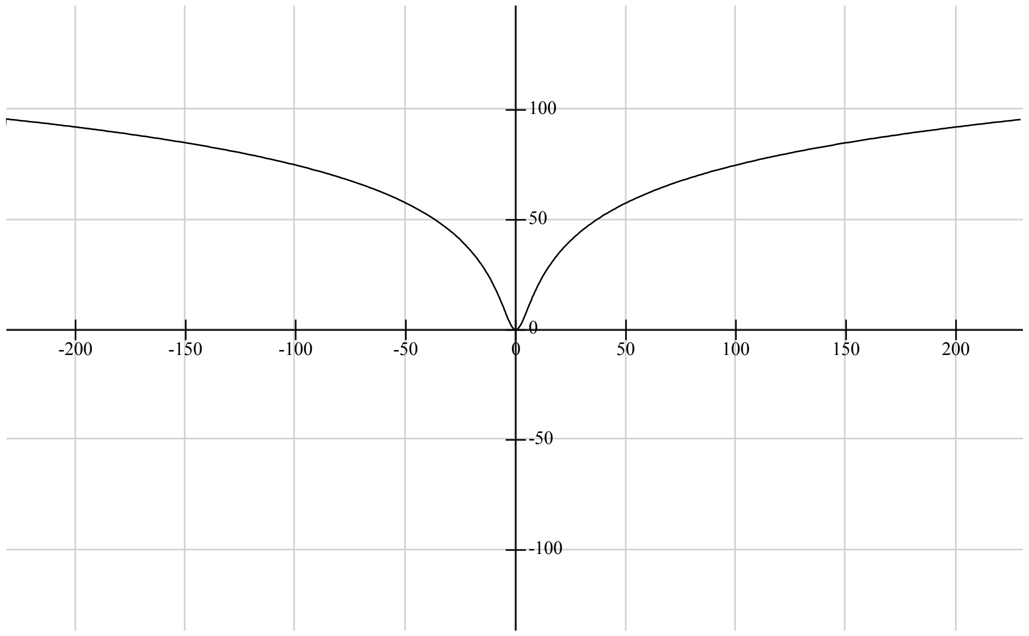


1. http://www.cs.cornell.edu/courses/cs4780/2015fa/web/lecturenotes/lecturenote10.html

Objective

- Проблема выбора функции, чтобы минимизировать MAE (не дифференцируема в нуле)

$$\frac{c^2 \ln\left(\frac{x^2}{c^2}+1\right)}{2}$$



Cauchy's function (Lorentzian function)

- Robust
- Does not guarantee a unique solution.
- The 95% asymptotic efficiency on the standard normal distribution is obtained with the tuning constant c=2.3849.

1. http://research.microsoft.com/en-us/um/people/zhang/INRIA/Publis/Tutorial-Estim/node24.htm

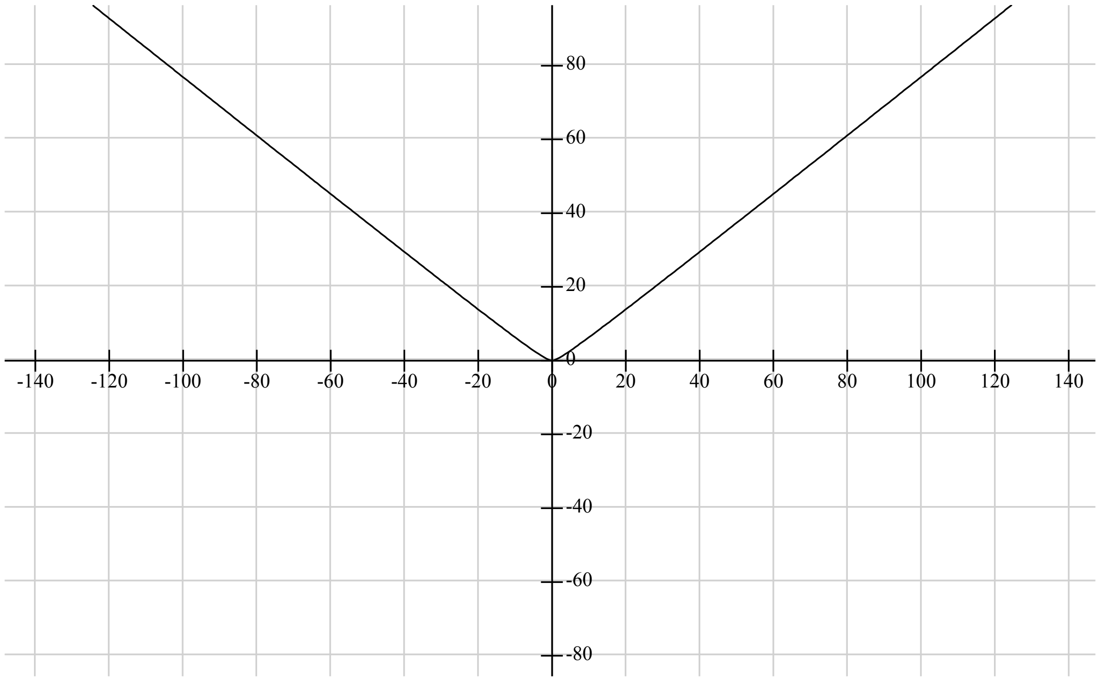
Objective (итоговая)

- Проблема выбора функции, чтобы минимизировать MAE (не дифференцируема в нуле)

$$c^2\left(\frac{|x|}{c}-\ln\left(\frac{|x|}{c}+1\right)\right)$$

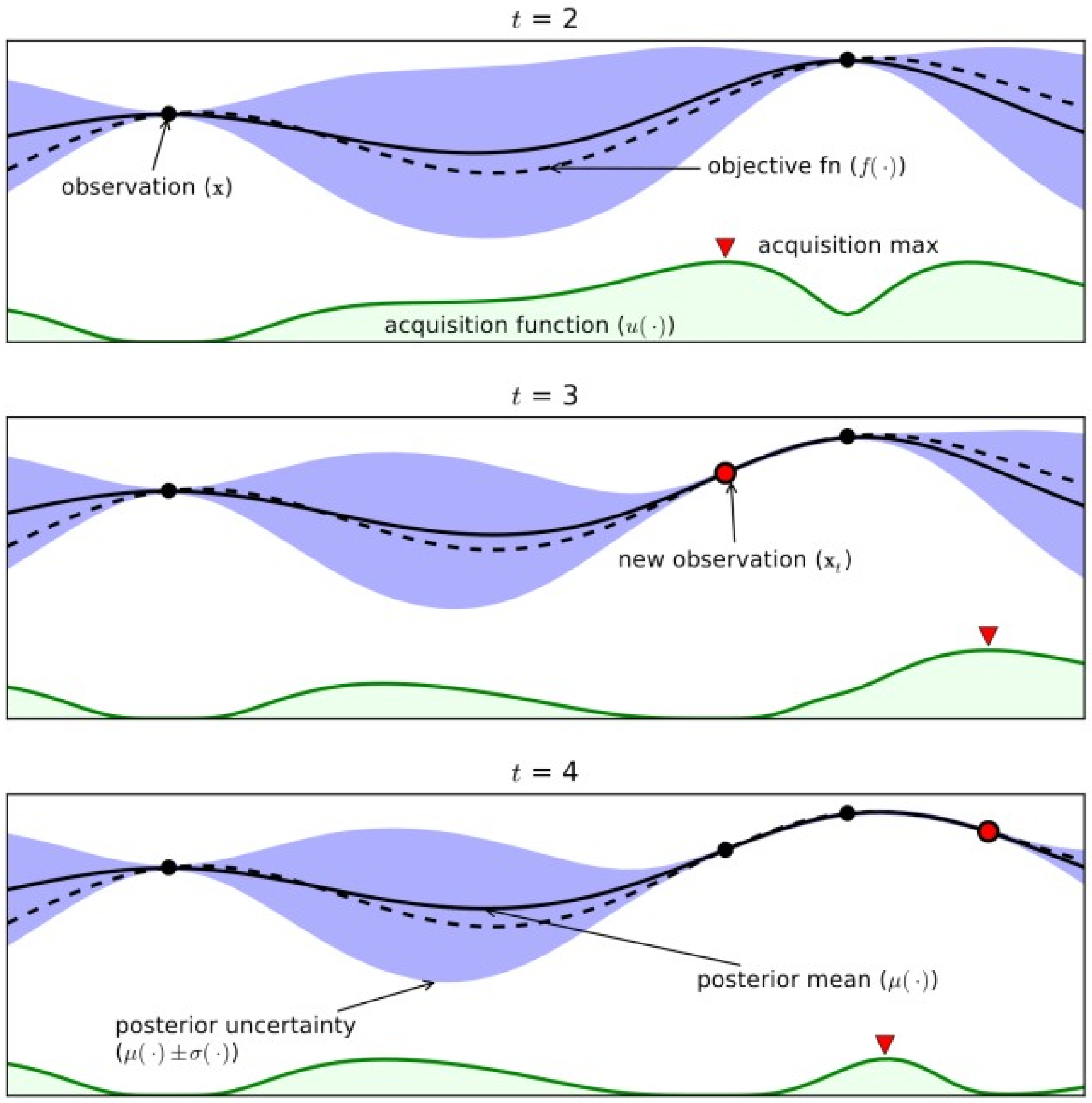
“Fair” function

- Robust
- It has everywhere defined continuous derivatives of first three orders
- Guarantee a unique solution.
- The 95% asymptotic efficiency on the standard normal distribution is obtained with the tuning constant c=1.3998.



1. http://research.microsoft.com/en-us/um/people/zhang/INRIA/Publis/Tutorial-Estim/node24.html

Bayesian optimization (hyperparameter tuning)



1. https://advancedoptimizationatharvard.wordpress.com/2014/03/30/bayesian-optimization-part-i/
2. https://startup.ml/blog/hyperparam
3. https://github.com/fmfn/BayesianOptimization

Модели

- XGBoost
  - Fair objective, C=0.7 (CV)
  - Bayesian optimized hyperparams
- LightGBM
  - Fair objective, C=0.7 (CV)
  - Bayesian optimized hyperparams
- Neural Network
  - 3 Dense Layers with Batch normalisation, Dropout=0.2, optimizer=AdaDelta
  - 55 epochs, 10Folds, 5bags

Кросс-валидация

- **Кросс-валидация по холдауту**
  - Нужно много данных
  - Проблема выбора репрезентативной выборки

- **10-Fold валидация (итоговая)**
  - Не забыть зафиксировать SEED!
  - Учить следующие уровни только по OOF-predictions
  - Использовать выше такие же фолды
  - Придумать хорошее разбиение (Stratified…)
  - Выбрать количество фолдов согласно CV и вашим мощностям

MCMC weights

- Markov Chain Monte Carlo methods
  - http://statweb.stanford.edu/~owen/mc/
  - https://github.com/pymc-devs/pymc3
  - https://habrahabr.ru/company/wunderfund/blog/279545/

Submissions

- XGBoost ~ 1108
- XGBoost + LGBM ~ 1106.40977
- XGBoost +NN + LGBM ~ 1103.06469
- 3-Layer stack:
  - 1 Layer: XGBoost + LGBM
  - 2 Layer: MCMC weights, MCMC powers, raw NN
  - 3 Layer: MCMC weights

~1101.33446   [113 Public/ 141 Private]

Обзор решений

7 Place Solution (Gilberto Titeritz) (1-Public/7-Private)

- XGB:
  - 2-way features interactios
  - different target transformations like: t^1, t^(1/2), t^(1/4), t^(1/8), log(t), log(t+100), log(t+200), log(t+400), 10/t.
  - bagging 10 time on each fold
- NN:
  - Same as XGB
  - different target transformations like: t^1, t^(1/2), t^(1/4), t^(1/8), log(t), log(t+100), log(t+200), log(t+400), 10/t.
- SVR, Factorization Machines
- Weighted average with Scipy Optimize
  - optimized the models weights for different predictions ranges

Обзор решений

3 Place Solution (FARON) (3-Public/3-Private)

Categorials:

- one hot encoding
- label encoding (in lexicographical order)
- creation of interactions (up to 4-way)
- target loss encoding
- tf-idf encoding
- XGB-embedding (leaf indices of single tree with depth >75 and min_child_weight > 100 used as feature for Neural Networks)
- Keras-embedding via embedding layers
- removal of rare values
- count of values

Numericals:

- quantile-based binning
- for NN: standard scaling, min-max scaling, Box-Cox transformation, percentile transformation
- target loss encoding
- count of values

Used target transformations

- y
- log(y + n)
- sqrt(y)
- boxcox(y)
- y => [0,1] (for logistic regression)

1 Layer: 100 base models (70% XGB & 30% Keras models)
2 Layer: 20-times bagged Keras model with one hidden layer as stacker at the 2nd level.

Обзор решений

2 Place Solution (Alexey Noskov) (9-Public/2-Private)

First-level models

70 models on top of different data transformations, target transformations and different parameters.

Second level

XGB and Keras NN models, with different params, linear regression with different target transformations, random forests and sklearn gradient boosting

Third level

For my third level I used quantile regression from statsmodels package. It doesn't have any regularization and seems to produce noisy results, so in last couple of days I tried to reduce model noise:

I've trained L3 model with 8 bags in each fold and used fold-average predictions for submission.

I've groupped L2 models by their similariity and averaged predictions in each group, so producing 10 features for L3 level. Also I've applied power correction to some of groups, like X^1.03.

1 Layer: 100 base models (70% XGB & 30% Keras models)
2 Layer: 20-times bagged Keras model with one hidden layer as stacker at the 2nd level.

Обзор решений

1 Place Solution (Bishwarup B) (4-Public/1-Private)

1 Layer: 81 models
2 Layer: XGB and two NN
3 Layer: w1*NN1^w2 + w3*NN2^w4 + w5*XGB1^w6 + w7
weights optimized by using optim (Nelder-Mead)