Tree-based Cost Sensitive Methods for Fraud Detection in Imbalanced Data

Guillaume Metzler, X. Badiche, B. Belkasmi, E. Fromont, A. Habrard and M. Sebban

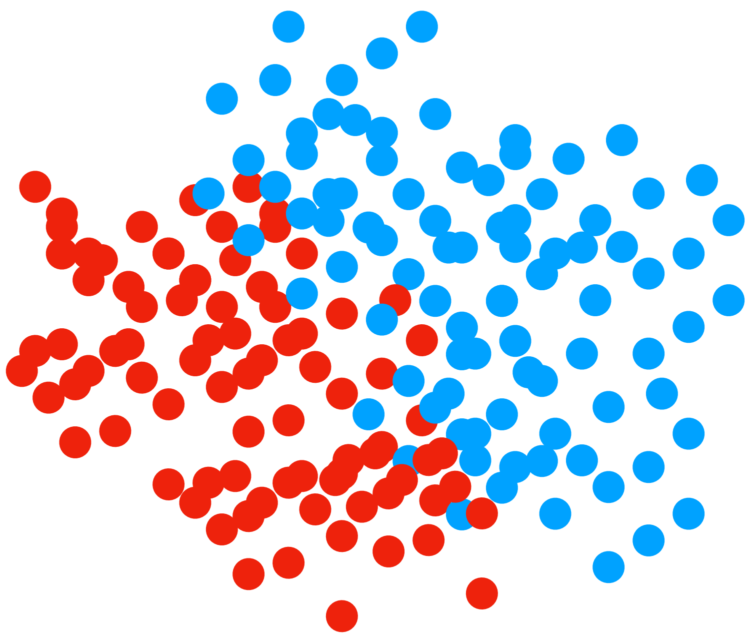
The Seventeenth International Symposium on Intelligent Data Analysis,
’s-Hertogenbocsh, 24-26 October

UNIVERSITÉ DE LYON

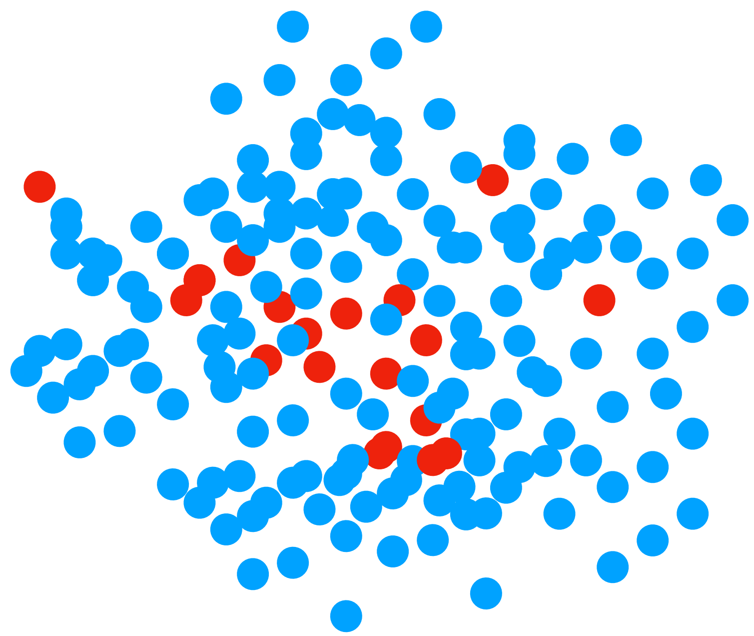
UNIVERSITÉ JEAN MONNET SAINT-ÉTIENNE

UNIVERSITÉ DE RENNES 1

Imbalanced learning



Balanced dataset (left)
VS.
Imbalanced one (right)



- Negative (N: common, majority, genuine)
- Positive (P: rare, minority, fraud)

In fraud detection (or imbalanced learning): few number of frauds or anomalies, P << N and IR = P/N < 0.5% in real cases.

Examples: spam detection, medical diagnosis, intrusion detection, bank fraud detection, …

Most of the classical Machine Learning techniques do not work well in such context

→ they focus on the majority class and predict all instances as negative

How to deal with imbalanced data?

Data level: use sampling methods like under/over-sampling or SMOTE

Algorithms:
- use metric learning based algorithm (e.g. LMNN)
- cost-sensitive learning
- combine several models together (e.g. boosting and stacking)

Post-Process: tune the decision threshold (class probability estimator)

All strategies present advantages and drawbacks

Context and Outline

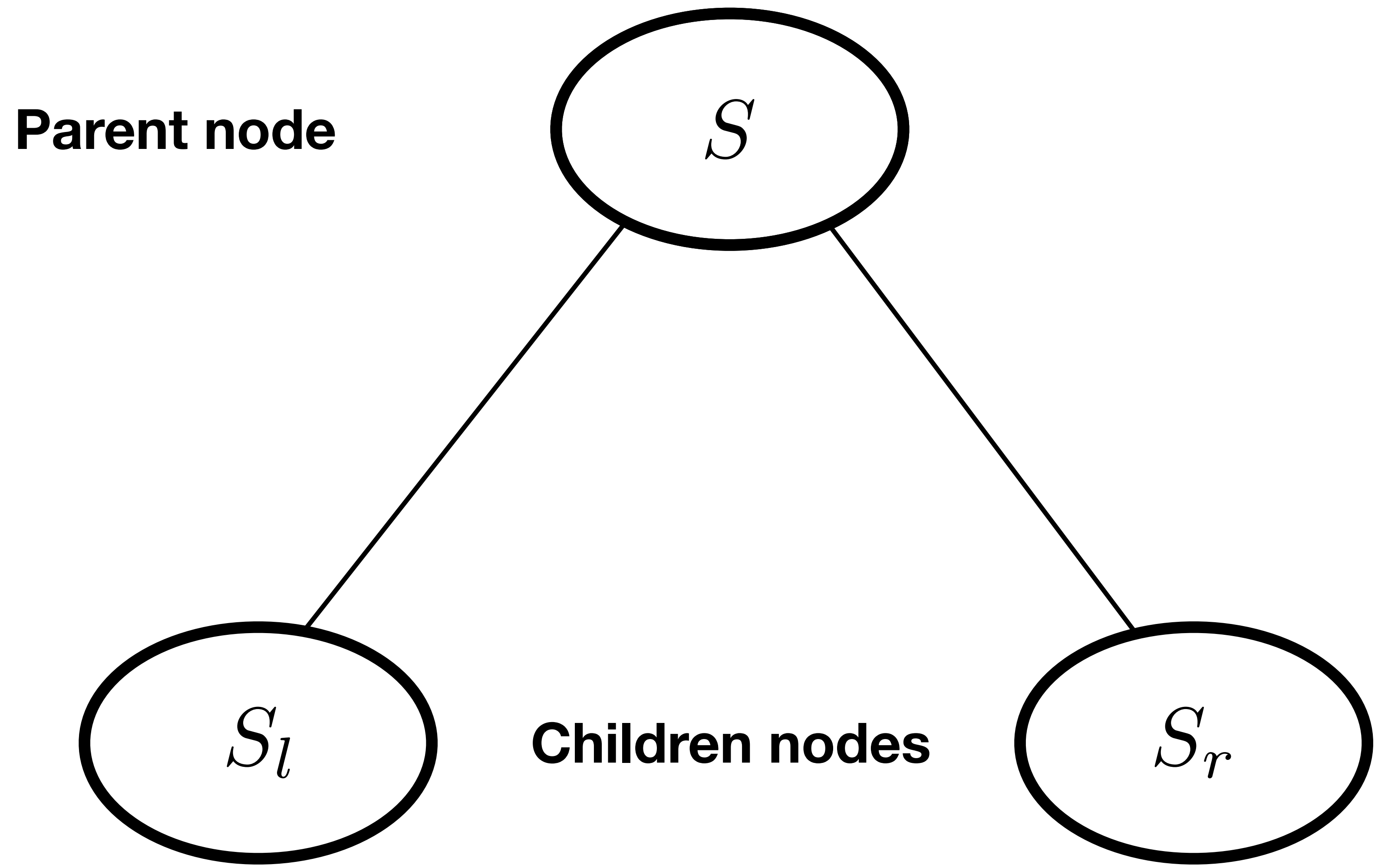
Context: Bank Fraud detection on check transactions, <0.5% of fraud
Retailers → maximize their profit and avoid frauds

Target: Build a model which focuses on retailers' desires → **cost-sensitive model**

Outline

1. Cost-sensitive decision trees
2. Ensemble of cost-sensitive decision trees
   1. Random forest
   2. Gradient boosting-based model
3. Experiments on a real dataset

Usual decision tree splitting criterion



Gini impurity of the node (binary case): $\Gamma = 1 - \sum_c p_c^2 = 1 - p_+^2 - p_-^2 = 2p_+p_-$

Split is made by maximizing: $\sum_{v \in \text{Children}} \Gamma_S - \alpha_v \Gamma_{S_v}$

[1] Breiman, L., Friedman, J. Olshen, R., Stone, C.: *Classification and Regression Trees. Wadsworth and Brooks*, CA (1984).

A cost sensitive model

Cost-sensitive matrix [2] with expert criteria

	Pred. fraud	Pred. genuine
Actual fraud	c_{TP_i}	c_{FN_i}
Actual genuine	c_{FP_i}	c_{TN_i}

where:

$$c_{TP_i} = 0 \qquad c_{FN_i} = (r - c) \cdot m$$

$$c_{FP_i} = \rho \cdot r \cdot m - \zeta \qquad c_{TN_i} = r \cdot m$$

m amount of the transaction

r profit rate

c loss rate (after insurance) of an unpaid transaction

ρ probability of finding another source of payment

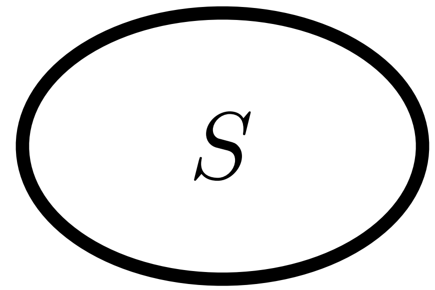
ζ customer dissatisfaction cost

→ Goal: maximize the overall profit of the retailer

[2] Bahnsen, A.C., Villegas, S., Aouada, D., Ottersten, B., Correa, A.M.: *Fraud Detection by Stacking Cost-sensitive Decision Trees*. DSCS (2017).

Cost sensitive decision trees

Parent node Γ_S



Use the « cost » matrix

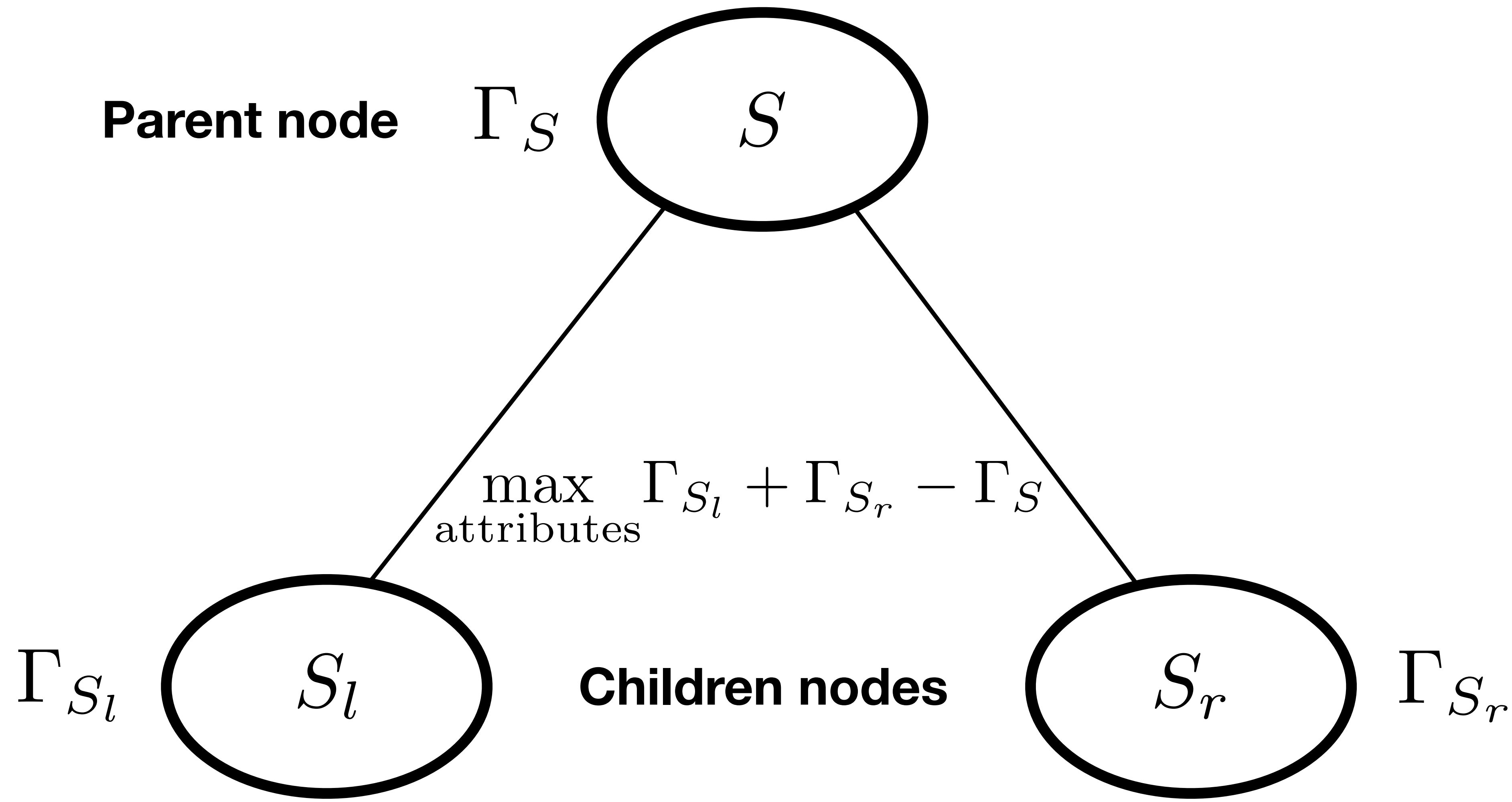
	Pred. fraud	Pred. genuine
Actual fraud	c_{TP_i}	c_{FN_i}
Actual genuine	c_{FP_i}	c_{TN_i}

Splitting criterion:

$$\Gamma_S = \frac{1}{|S|}\sum_{i\in S_-}\left(\frac{m_+}{m}c_{FP_i}(x_i) + \frac{m_-}{m}c_{TN_i}(x_i)\right) + \frac{1}{|S|}\sum_{i\in S_+}\left(\frac{m_+}{m}c_{TP_i}(x_i) + \frac{m_-}{m}c_{FN_i}(x_i)\right),$$

If $c_{TP_i} = c_{TN_i} = 0$ and $c_{FP_i} = c_{FN_i} = 1$, standard Gini Impurity.

Cost sensitive decision trees



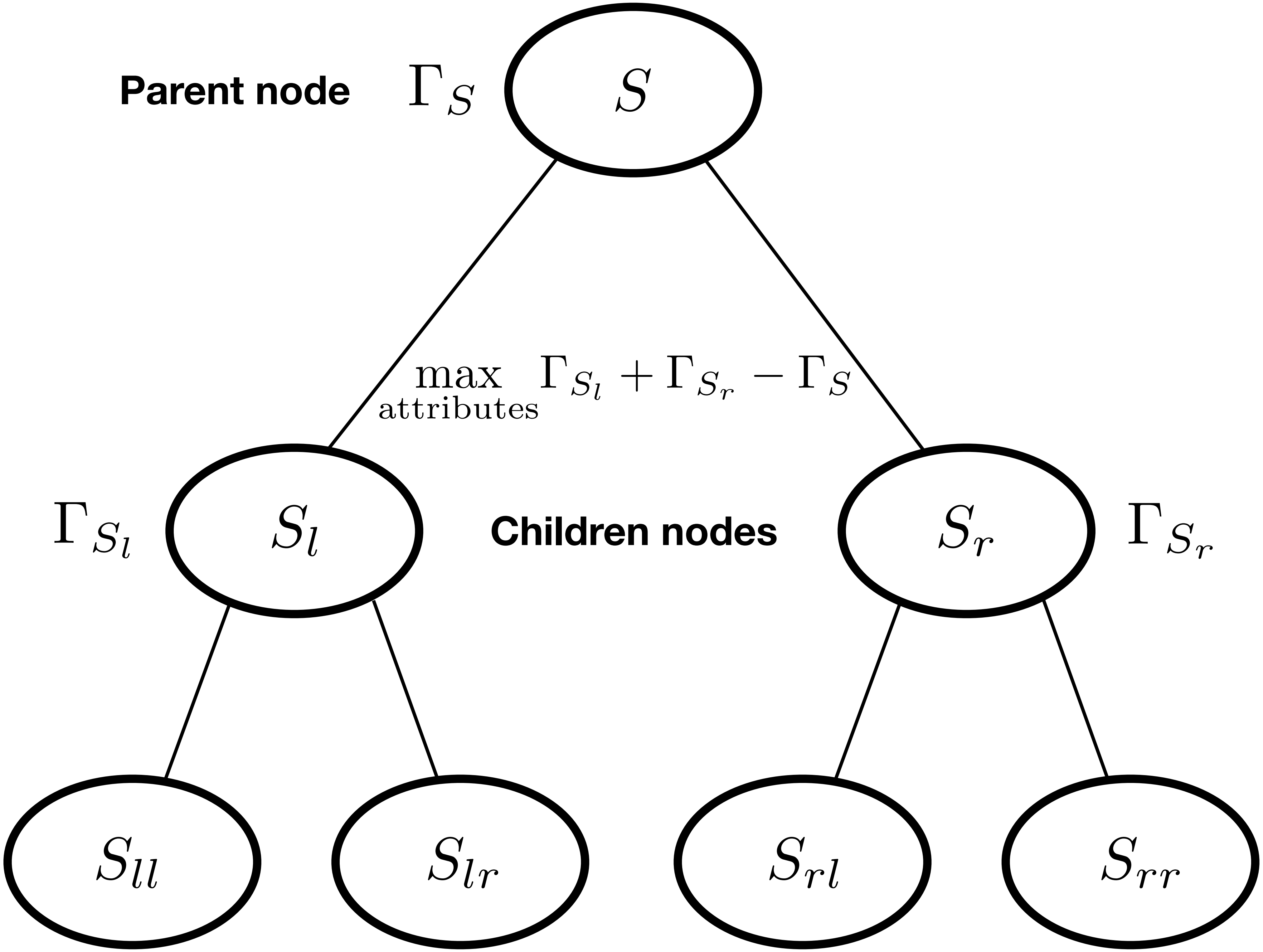
Compute the splitting criterion

$$\Gamma_S = \frac{1}{|S|} \sum_{i \in S_-} \left(\frac{m_+}{m} c_{FP_i}(x_i) + \frac{m_-}{m} c_{TN_i}(x_i) \right) + \frac{1}{|S|} \sum_{i \in S_+} \left(\frac{m_+}{m} c_{TP_i}(x_i) + \frac{m_-}{m} c_{FN_i}(x_i) \right),$$

Look for the best (attribute, value) which is solution of:

$$\max_{\text{attributes}} \Gamma_{S_l} + \Gamma_{S_r} - \Gamma_S$$

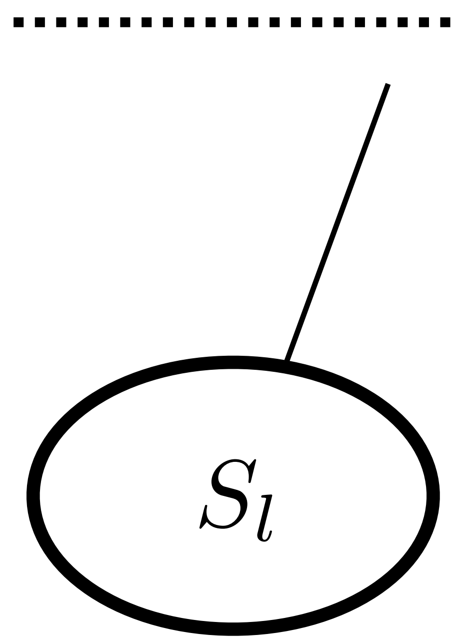
Cost sensitive decision trees



Until maximum depth or other stopping criterion

Cost sensitive decision trees

How to label the leaves ?



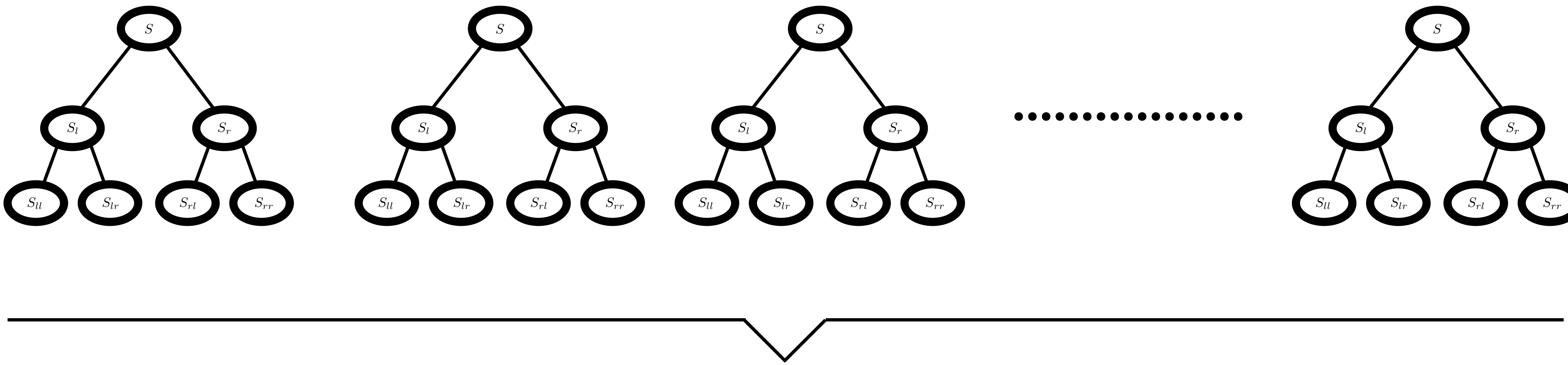
Compute the profits if all the examples are predicted positiveγ_+ and negative γ_-

$$\gamma_+(l) = \frac{1}{|l|}\left(\sum_{i:x_i \in l \cap S_-} c_{FP_i} + \sum_{i:x_i \in l \cap S_-} c_{TP_i}\right) \qquad \gamma_-(l) = \frac{1}{|l|}\left(\sum_{i:x_i \in l \cap S_-} c_{TN_i} + \sum_{i:x_i \in l \cap S_+} c_{FN_i}\right)$$

Choose the label j which is solution of: $\underset{j \in \{+,-\}}{max}\ \gamma_j$

Cost sensitive random forest

Build a collection of trees



Combine the output of each tree: average profit or predicted label
to build several model (see experimental setting)

Gradient Boosting model

Idea of boosting: combine several weak learners f_t into a single strong model F

$$F_T = \sum_{t=0}^{T} \alpha_t f_t$$

Usual loss function for boosting: exponential loss

$$L(x_i, y_i) = y_i \exp(-F(x_i)) + (1 - y_i) \exp(F(x_i))$$

Gradient boosting: work in the function space rather than the parameter space [3]

Using:

$$r_i = g_t = -\left[\frac{\partial L(y, F_{t-1}(x_i))}{\partial F_{t-1}(x_i)} \right] \qquad (f_t, \alpha_t) = \underset{\alpha, f}{\operatorname{argmin}} \sum_{i=1}^{m} (r_i - \alpha f(x_i))^2$$

Update rule: $F_t = F_{t-1} + \alpha_t f_t$

[3] Friedman, J.H.: *Greedy Function Approximation: A Gradient Boosting Machine.* Annals of Statistics **29** (2000).

Cost sensitive gradient boosting

Idea: include the cost matrix into the loss function L

	$\hat{y}_i = 1$	$\hat{y}_i = 0$
$y_i = 1$	c_{TP_i}	c_{FN_i}
$y_i = 0$	c_{FP_i}	c_{TN_i}

Our loss: $L(y,\hat{y}) = \sum_{i=1}^{m} \left[y_i(\hat{y}_i c_{TP_i} + (1-\hat{y}_i) c_{FN_i}) + (1-y_i)(\hat{y}_i c_{FP_i} + (1-\hat{y}_i) c_{TN_i}) \right]$

How to use the output of a gradient tree boosting model?

For a model which return a probability p_i, predict fraud ($\hat{y}_i = 1$) if [4]:

$$p_i > \frac{c_{TN_i} - c_{FP_i}}{c_{TP_i} - c_{FN_i} + c_{TN_i} - c_{FP_i}} = s_i$$

Rewrite loss as a minimization problem:

$$L(y,p) = -\sum_{i=1}^{m} (y_i c_{TP_i} + (1-y_i) c_{FP_i})\, \mathbb{I}_{p_i > s_i} + (y_i c_{FN_i} + (1-y_i) c_{TN_i}) \mathbb{I}_{p_i \leq s_i}$$

[4] Elkan, C.: *The Foundations of Cost-Sensitive Learning*. Proceedings of the 17th International Joint Conference on Artificial Intelligence (2001).

Cost sensitive gradient boosting

The loss can be rewritten (using the analytical value of s_i):

$$L(x_i, y_i) = (1 - s_i)y_i \mathbb{I}_{p_i < s_i} + s_i(1 - y_i)\mathbb{I}_{p_i > s_i}$$

We show, because $\mathbb{I}_{p_i > s_i} \leq \exp(F(x_i))$, it is enough to minimize

$$L(x_i, y_i) = (1 - s_i)y_i e^{-F(x_i)} + s_i(1 - y_i)e^{F(x_i)}$$

The gradient boosting model is computed using a specific solver, XGboost which only needs the first and second derivative of the loss function.

Experimental Setting

Baseline: standard random forest (**RF**) with 24 trees.

$\mathbf{RF_{lab-maj}}$: each leaf is labeled according to the majority class of the examples that fall into the leaf + use of a majority vote.

$\mathbf{RF_{lab-pro}}$: each leaf is labeled to maximize the profit over the set of all examples in the leaf + use of a majority vote.

$\mathbf{RF_{mean-pro}}$: same as before but vote is done with the concept of profits.

$\mathbf{GB_{tune-\ldots}}$:gradient boosting model with a logistic loss, threshold is tuned for … criterium.

$\mathbf{GB_{profits}}$: gradient boosting model with the « profit loss »

Data: 10 months of transactions (6 training/validation set and 4 test set)
~ 2.7M of transactions and only 0.33% of frauds.

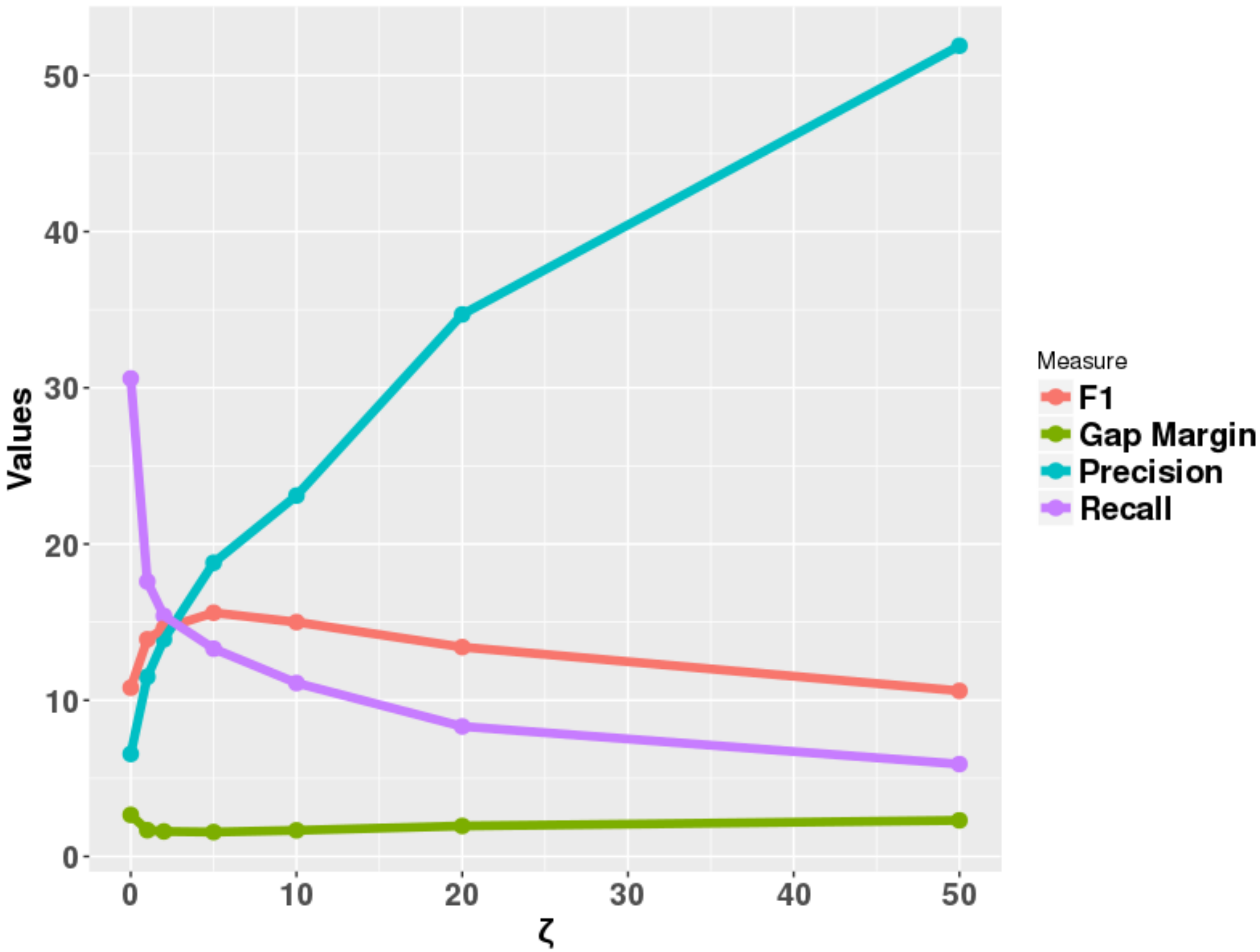
Experimental Results

Compare different procedures with the current model of the company

Experiments	**Rate loss max profits**	**Precision**	**Recall**	$\mathbf{F_1}$
$\mathbf{RF}$	2.99%	68.1%	5.66%	10.5%
$\mathbf{RF_{maj}}$	2.88%	73.8%	4.71%	8.86%
$\mathbf{RF_{maj-mar}}$	1.81%	30.2%	10.6%	15.7%
$\mathbf{RF_{mean-margin}}$	1.87%	30.3%	9.52%	14.5%
$\mathbf{GB_{tune-Pre}}$	3.01%	**61.0%**	6.49%	11.7%
$\mathbf{GB_{tune-mar}}$	2.26%	19.1%	**16.6%**	**17.8%**
$\mathbf{GB_{tune-F1}}$	2.70%	45.4%	9.24%	15.4%
$\mathbf{GB_{margin}}$	**1.56%**	18.8%	13.3%	15.6%

- Improve the benefit of the retailers with both models.
- GB-based model gives better results.
- Reduce the loss $\longrightarrow$ having a lower precision (< 30%) but higher recall.

Optimal value of ζ for the retailers



Evolution of Precision, Recall, F-Measure and the gap to the maximal profits with respect to the parameter ζ

NB: gap computed with $\zeta = 5$

- ζ: customer dissatisfaction cost
- when ζ increases the cost of a False Positive is decreasing

↓

The Precision is increasing and the Recall is decreasing

Conclusion

- Provide an understandable model for our customer
- Reduce the gap of the maximal benefits from 2.99 % to 1.56 % (represents a gain of 60 k euros per 4 months
- Able to control the precision
- RF (with our decision rule) and GB-based model can give similar results [6], but GB-based models are trained one order of magnitude faster

Perspectives

- Improve the fraud detection models with currently unused informations:
  - loyalty cards
  - customers' baskets
  - historical purchase information

[5] Nikolaou, N., Edakunni, N., Kull, M., Flach, P., Brown, G.: *Cost-Sensitive Boosting Algorithms. Do we really need them ?* Machine Learning **104** (2016).

Thank you for your attention !

Bibliography

[1] Breiman, L., Friedman, J. Olshen, R., Stone, C.: *Classification and Regression Trees. Wadsworth and Brooks*, CA (1984).

[2] Bahnsen, A.C., Villegas, S., Aouada, D., Ottersten, B., Correa, A.M.: *Fraud Detection by Stacking Cost-sensitive Decision Trees*. DSCS (2017).

[3] Friedman, J.H.: *Greedy Function Approximation: A Gradient Boosting Machine.* Annals of Statistics **29** (2000).

[4] Elkan, C.: *The Foundations of Cost-Sensitive Learning*. Proceedings of the 17th International Joint Conference on Artificial Intelligence (2001).

[5] Nikolaou, N., Edakunni, N., Kull, M., Flach, P., Brown, G.: *Cost-Sensitive Boosting Algorithms. Do we really need them ?* Machine Learning **104** (2016).