EXAMPLE-DEPENDENT COST-SENSITIVE CLASSIFICATION

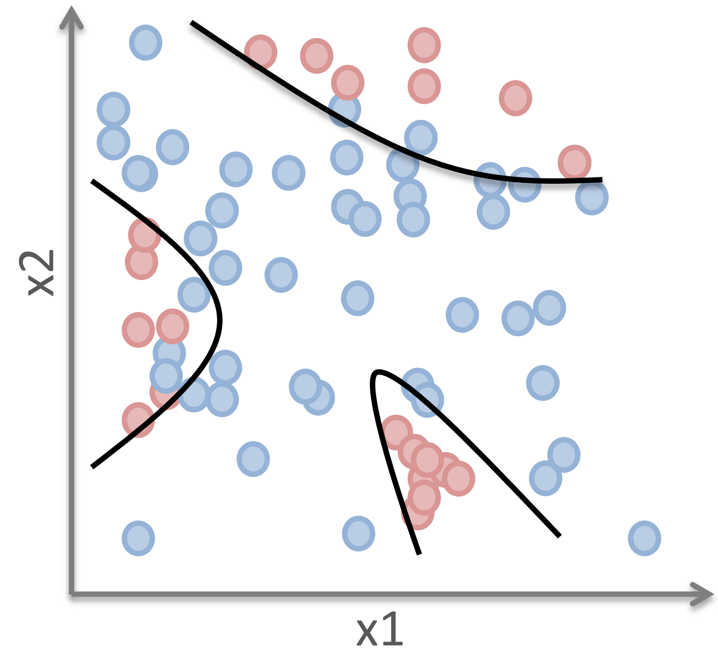
applications in financial risk modeling and marketing analytics

September 15, 2015

Alejandro Correa Bahnsen
with
Djamila Aouada, SnT
Björn Ottersten, SnT

Motivation

- Classification: **predicting the class of a set of examples given their features**.
- Standard classification methods aim at minimizing the errors
- Such a traditional framework assumes that all **misclassification errors carry the same cost**



- This is not the case in many real-world applications: **Credit card fraud detection, churn modeling, credit scoring, direct marketing**.

Motivation

SnT
securityandtrust.lu

- **Credit card fraud detection,** failing to detect a fraudulent transaction may have an economical impact from a few to thousands of Euros, depending on the particular transaction and card holder.
- **Credit scoring,** accepting loans from bad customers does not have the same economical loss, since customers have different credit lines, therefore, different profit.
- **Churn modeling,** misidentifying a profitable or unprofitable churner has a significant different economic result.
- **Direct marketing,** wrongly predicting that a customer will not accept an offer when in fact he will, may have different financial impact, as not all customers generate the same profit.

uni.lu
UNIVERSITÉ DU LUXEMBOURG

Agenda

- Motivation
- Cost-sensitive classification

  Background
- Real-world cost-sensitive applications

  Credit card fraud detection, churn modeling, credit scoring, direct marketing
- Proposed cost-sensitive algorithms

  Bayes minimum risk, cost-sensitive logistic regression, cost-sensitive decision trees, ensembles of cost-sensitive decision trees
- Experiments

  Experimental setup, results
- Conclusions

  Contributions, future work

Background - Binary classification

predict the class of set of examples given their features

$$f: S \rightarrow \{0,1\}$$

Where each element of S is composed by $X_i = [x_i^1, x_i^2, \ldots, x_i^k]$

It is usually evaluated using a traditional misclassification measures such as Accuracy, F1Score, AUC, among others.

However, these measures assume that different misclassification errors carry the **same cost**

Background - Cost-sensitive evaluation

securityandtrust.lu

We define a cost measure based on the **cost matrix** [Elkan 2001]

	Actual Positive $y_i = 1$	Actual Negative $y_i = 0$
Predicted Positive $c_i = 1$	C_{TP_i}	C_{FP_i}
Predicted Negative $c_i = 0$	C_{FN_i}	C_{TN_i}

From which we calculate the ***Cost*** of applying a classifier to a given set

$$Cost\big(f(S)\big) = \sum_{i=1}^{N} y_i\big(c_i C_{TP_i} + (1 - c_i) C_{FN_i}\big) + (1 - y_i)\big(c_i C_{FP_i} + (1 - c_i) C_{TN_i}\big)$$

Background - Cost-sensitive evaluation

However, the total cost may not be easy to interpret. Therefore, we propose a ***Savings*** measure as the cost vs. the cost of using no algorithm at all

$$Savings(f(S)) = \frac{Cost_l(f(S)) - Cost(f(S))}{Cost_l(f(S))}$$

Where $\boldsymbol{Cost_l(f(S))}$ is the cost of predicting the costless class

$$Cost_l(f(S)) = \min\{Cost(f_0(S)), Cost(f_1(S))\}$$

Background - State-of-the-art methods

securityandtrust.lu

Research in example-dependent cost-sensitive classification has been narrow, mostly because of the **lack of publicly available datasets** [Aodha and Brostow 2013].

Standard approaches consist in **re-weighting the training examples** based on their costs:

- Cost-proportionate rejection sampling [Zadrozny et al. 2003]
- Cost-proportionate oversampling [Elkan 2001]

UNIVERSITÉ DU LUXEMBOURG

Agenda

SnT
securityandtrust.lu

- Motivation
- Cost-sensitive classification
  Background
- Real-world cost-sensitive applications
  Credit card fraud detection, churn modeling, credit scoring, direct marketing
- Proposed cost-sensitive algorithms
  Bayes minimum risk, cost-sensitive logistic regression, cost-sensitive decision trees, ensembles of cost-sensitive decision trees
- Experiments
  Experimental setup, results
- Conclusions
  Contributions, future work

uni.lu UNIVERSITÉ DU LUXEMBOURG

Credit card fraud detection

Estimate the **probability** of a transaction being **fraud** based on analyzing customer patterns and recent fraudulent behavior

Issues when constructing a fraud detection system [Bolton et al., 2002]:

- Skewness of the data
- **Cost-sensitivity**
- Short time response of the system
- Dimensionality of the search space
- **Feature preprocessing**

Credit card fraud detection

SnT
securityandtrust.lu

Credit card fraud detection is a **cost-sensitive problem**. As the cost due to a false positive is different than the cost of a false negative.

- **False positives:** When predicting a transaction as fraudulent, when in fact it is not a fraud, there is an administrative cost that is incurred by the financial institution.
- **False negatives:** Failing to detect a fraud, the amount of that transaction is lost.

Moreover, it is not enough to assume a constant cost difference between false positives and false negatives, as the amount of the transactions **varies quite significantly**.

Credit card fraud detection

Cost matrix

	Actual Positive $y_i = 1$	Actual Negative $y_i = 0$
Predicted Positive $c_i = 1$	$C_{TP_i} = C_a$	$C_{FP_i} = C_a$
Predicted Negative $c_i = 0$	$C_{FN_i} = Amt_i$	$C_{TN_i} = 0$

A. Correa Bahnsen, A. Stojanovic, D. Aouada, and B. Ottersten, "Cost Sensitive Credit Card Fraud Detection Using Bayes Minimum Risk," in 2013 12th International Conference on Machine Learning and Applications. Miami, USA: IEEE, Dec. 2013, pp. 333–338.

Credit card fraud detection

Raw features

Attribute	name Description
Transaction ID	Transaction identification number
Time	Date and time of the transaction
Account number	Identification number of the customer
Card number	Identification of the credit card
Transaction type	ie. Internet, ATM, POS, ...
Entry mode	ie. Chip and pin, magnetic stripe, ...
Amount	Amount of the transaction in Euros
Merchant code	Identification of the merchant type
Merchant group	Merchant group identification
Country	Country of trx
Country 2	Country of residence
Type of card	ie. Visa debit, Mastercard, American Express...
Gender	Gender of the card holder
Age	Card holder age
Bank Issuer	bank of the card

Credit card fraud detection

SnT securityandtrust.lu

Transaction **aggregation** strategy [Whitrow, 2008]

Raw Features					Aggregated Features			
TrxId	Time	Type	Country	Amt	No Trx last 24h	Amt last 24h	No Trx last 24h same type and country	Amt last 24h same type and country
1	1/1 18:20	POS	Lux	250	0	0	0	0
2	1/1 20:35	POS	Lux	400	1	250	1	250
3	1/1 22:30	ATM	Lux	250	2	650	0	0
4	2/1 00:50	POS	Ger	50	3	900	0	0
5	2/1 19:18	POS	Ger	100	3	700	1	50
6	2/1 23:45	POS	Ger	150	2	150	2	150
7	3/1 06:00	POS	Lux	10	3	400	0	0

Credit card fraud detection

Proposed **periodic** features

When is a customer expected to make a new transaction?

Considering a **von Mises distribution** with a period of 24 hours such that

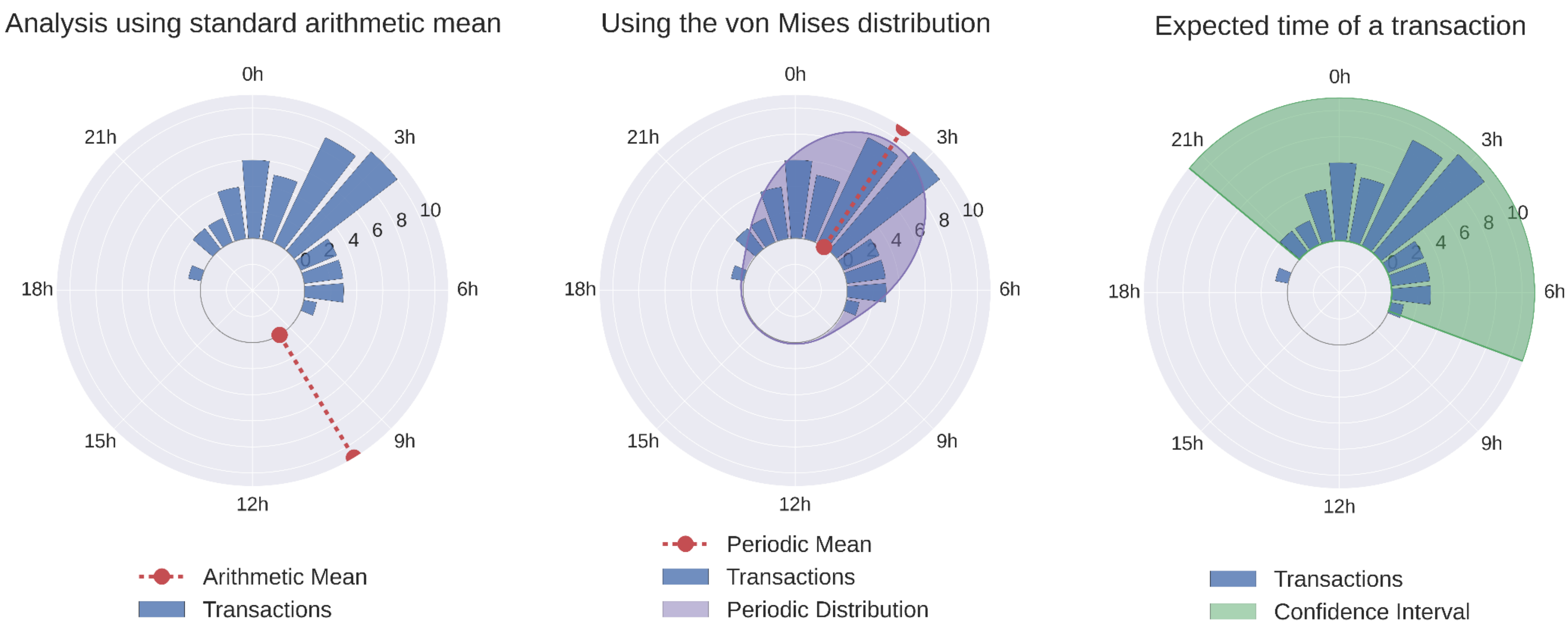
$$P(time) \sim vonmises(\mu, \sigma) = \frac{e^{(\sigma cos(time-\mu))}}{2\pi I_0(\sigma)}$$

where $\boldsymbol{\mu}$ is the mean, $\boldsymbol{\sigma}$ is the standard deviation, and $\boldsymbol{I_0}$ is the Bessel function

Credit card fraud detection

SnT
securityandtrust.lu

Proposed **periodic** features



A. Correa Bahnsen, A. Stojanovic, D. Aouada, and B. Ottersten, “Feature Engineering Strategies for Credit Card Fraud Detection” submitted to Expert Systems with Applications.

Credit scoring

Classify which potential customers are likely to **default** a contracted financial obligation based on the customer's **past financial experience**.

It is a cost-sensitive problem as the cost associated with approving a bad customer, i.e., **false negative**, is quite different from the cost associated with declining a good customer, i.e., **false positive**. Furthermore, the costs are **not constant** among customers. This is because loans have different credit line amounts, terms, and even interest rates.

Credit scoring

Cost matrix

	Actual Positive $\boldsymbol{y_i = 1}$	Actual Negative $\boldsymbol{y_i = 0}$
Predicted Positive $\boldsymbol{c_i = 1}$	$C_{TP_i} = 0$	$C_{FP_i} = r_i + C_{FP}^a$
Predicted Negative $\boldsymbol{c_i = 0}$	$C_{FN_i} = Cl_i * L_{gd}$	$C_{TN_i} = 0$

A. Correa Bahnsen, D. Aouada, and B. Ottersten, "Example-Dependent Cost-Sensitive Logistic Regression for Credit Scoring," in 2014 13th International Conference on Machine Learning and Applications. Detroit, USA: IEEE, 2014, pp. 263–269.

Churn modeling

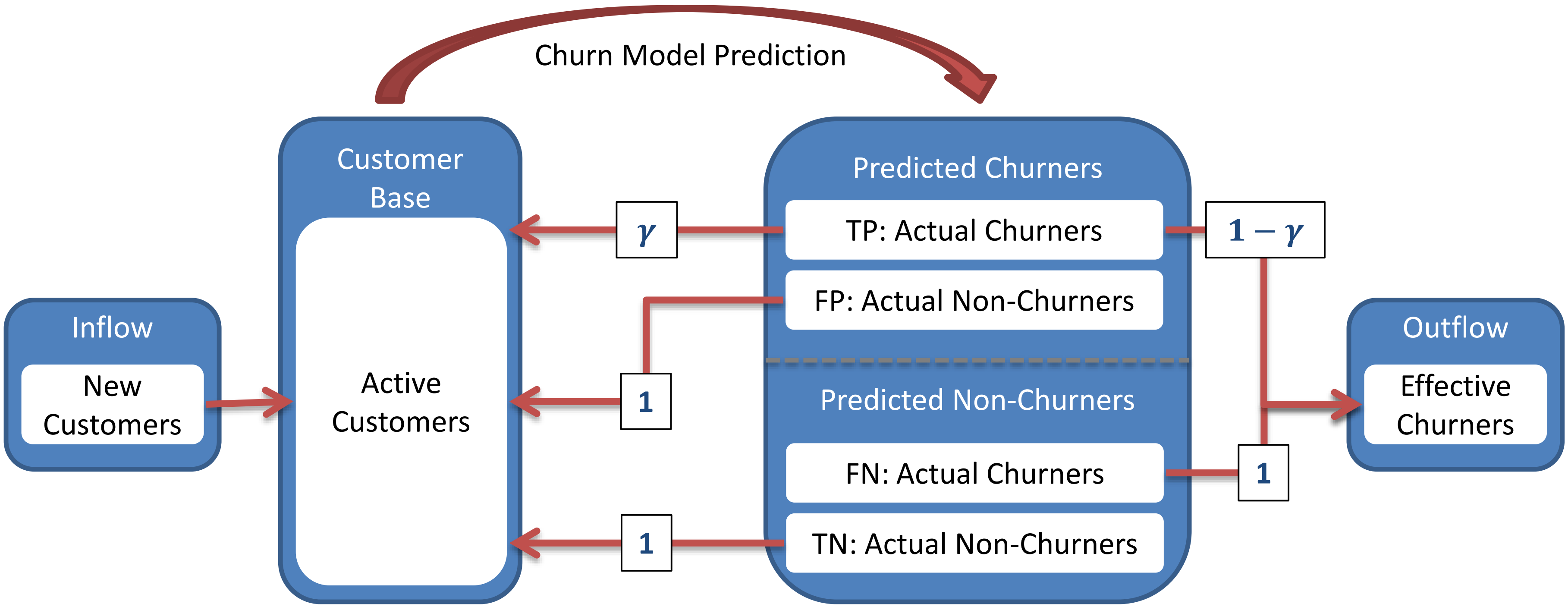
SnT
securityandtrust.lu

Predict the probability of a **customer defecting** using historical, behavioral and socioeconomical information.

This tool is of great benefit to **subscription based companies** allowing them to maximize the results of retention campaigns.

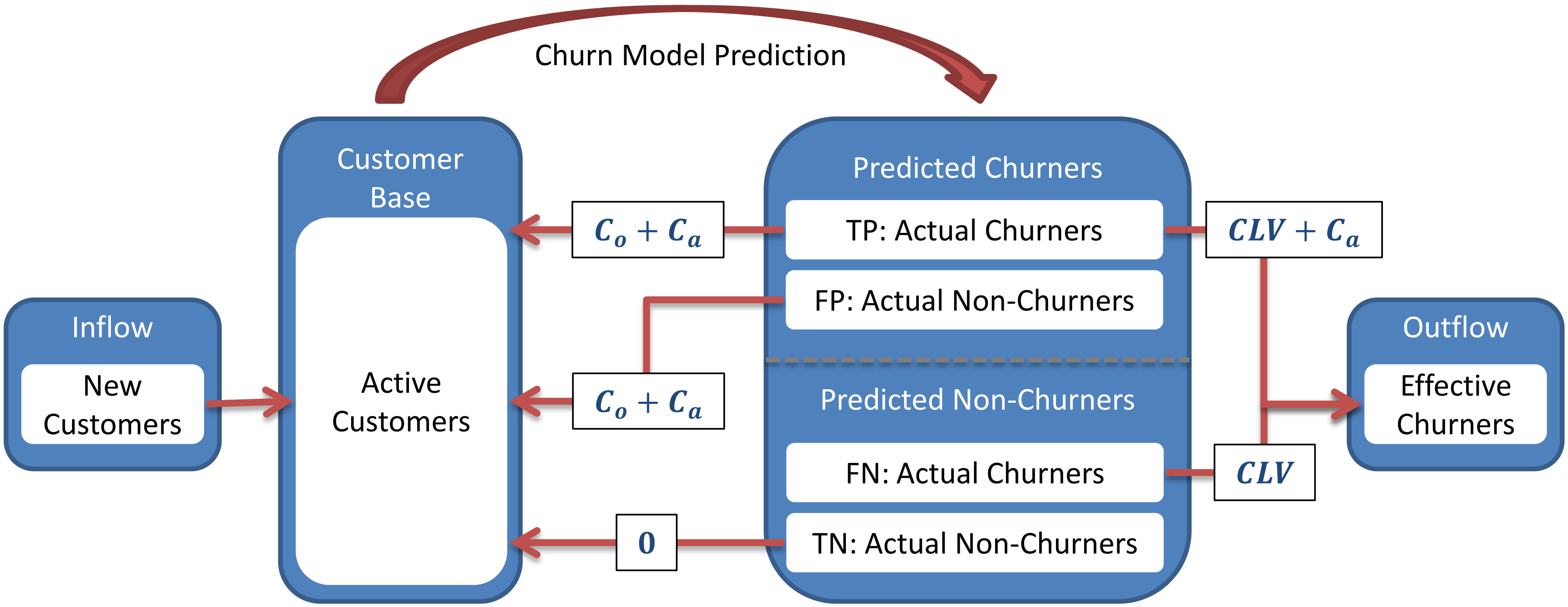
Churn modeling

Churn management campaign [Verbraken, 2013]



Churn modeling

Proposed financial evaluation of a churn campaign



A. Correa Bahnsen, A. Stojanovic, D. Aouada, and B. Ottersten, "A novel cost-sensitive framework for customer churn predictive modeling," Decision Analytics, vol. 2:5, 2015.

Churn modeling

Cost matrix

	Actual Positive $\boldsymbol{y_i = 1}$	Actual Negative $\boldsymbol{y_i = 0}$
Predicted Positive $\boldsymbol{c_i = 1}$	$C_{TP_i} = \gamma_i C_{o_i} + (1-\gamma_i)(CLV_i + C_a)$	$C_{FP_i} = C_{o_i} + C_a$
Predicted Negative $\boldsymbol{c_i = 0}$	$C_{FN_i} = CLV_i$	$C_{TN_i} = 0$

A. Correa Bahnsen, A. Stojanovic, D. Aouada, and B. Ottersten, "A novel cost-sensitive framework for customer churn predictive modeling," Decision Analytics, vol. 2:5, 2015.

Direct marketing

Classify those customers who are more likely to have a certain response to a marketing campaign.

This problem is example-dependent cost sensitive, as the **false positives** have the cost of contacting the client, and **false negatives** have the cost due to the **loss of income** by failing to making the an offer to the right customer.

Direct marketing

Cost matrix

	Actual Positive $y_i = 1$	Actual Negative $y_i = 0$
Predicted Positive $c_i = 1$	$C_{TP_i} = C_a$	$C_{FP_i} = C_a$
Predicted Negative $c_i = 0$	$C_{FN_i} = Int_i$	$C_{TN_i} = 0$

A. Correa Bahnsen, A. Stojanovic, D. Aouada, and B. Ottersten, "Improving Credit Card Fraud Detection with Calibrated Probabilities," in Proceedings of the fourteenth SIAM International Conference on Data Mining, Philadelphia, USA, 2014, pp. 677 – 685.

Agenda

securityandtrust.lu

- Motivation
- Cost-sensitive classification

  Background
- Real-world cost-sensitive applications

  Credit card fraud detection, churn modeling, credit scoring, direct marketing
- Proposed cost-sensitive algorithms

  Bayes minimum risk, cost-sensitive logistic regression, cost-sensitive decision trees, ensembles of cost-sensitive decision trees
- Experiments

  Experimental setup, results
- Conclusions

  Contributions, future work

UNIVERSITÉ DU LUXEMBOURG

Proposed cost-sensitive algorithms

securityandtrust.lu

- **Bayes minimum risk (BMR)**

A. Correa Bahnsen, A. Stojanovic, D. Aouada, and B. Ottersten, "Cost Sensitive Credit Card Fraud Detection Using Bayes Minimum Risk," in 2013 12th International Conference on Machine Learning and Applications. Miami, USA: IEEE, Dec. 2013, pp. 333–338.

A. Correa Bahnsen, Aouada, and B. Ottersten, "Improving Credit Card Fraud Detection with Calibrated Probabilities," in Proceedings of the fourteenth SIAM International Conference on Data Mining, Philadelphia, USA, 2014, pp. 677 – 685.

- **Cost-sensitive logistic regression (CSLR)**

A. Correa Bahnsen, D. Aouada, and B. Ottersten, "Example-Dependent Cost-Sensitive Logistic Regression for Credit Scoring," in 2014 13th International Conference on Machine Learning and Applications. Detroit, USA: IEEE, 2014, pp. 263–269.

- **Cost-sensitive decision trees (CSDT)**

A. Correa Bahnsen, D. Aouada, and B. Ottersten, "Example-Dependent Cost-Sensitive Decision Trees," Expert Systems with Applications, vol. 42:19, 2015.

- **Ensembles of cost-sensitive decision trees (ECSDT)**

A. Correa Bahnsen, D. Aouada, and B. Ottersten, "Ensemble of Example-Dependent Cost-Sensitive Decision Trees," IEEE Transactions on Knowledge and Data Engineering, vol. under review, 2015.

Bayes Minimum Risk

Decision model based on **quantifying tradeoffs** between various decisions using probabilities and the costs that accompany such decisions

Risk of classification

$$R(c_i = 0|x_i) = C_{TN_i}(1 - \hat{p}_i) + C_{FN_i} \cdot \hat{p}_i$$

$$R(c_i = 1|x_i) = C_{FP_i}(1 - \hat{p}_i) + C_{TP_i} \cdot \hat{p}_i$$

Using the different risks the prediction is made based on the following condition:

$$c_i = \begin{cases} 0 & R(c_i = 0|x_i) \leq R(c_i = 1|x_i) \\ 1 & otherwise \end{cases}$$

Cost-Sensitive Logistic Regression

securityandtrust.lu

- Logistic Regression Model

$$\hat{p}_i = P(y_i = 0|x_i) = h_\theta(x_i) = g\left(\sum_{j=1}^{k} \theta_j x_i^j\right)$$

- **Cost Function**

$$J_i(\theta) = -y_i \log\big(h_\theta(x_i)\big) - (1 - y_i) \log\big(1 - h_\theta(x_i)\big)$$

- **Cost Analysis**

$$J_i(\theta) \approx \begin{cases} 0 & if \quad y_i \approx h_\theta(x_i) \\ inf & if \quad y_i \approx 1 - h_\theta(x_i) \end{cases} \Rightarrow \begin{aligned} C_{TP_i} &= C_{TN_i} \approx 0 \\ C_{FP_i} &= C_{FN_i} \approx \infty \end{aligned}$$

UNIVERSITÉ DU LUXEMBOURG

Cost-Sensitive Logistic Regression

- **Actual** Costs

$$J^c(\theta) = \begin{cases} C_{TP_i} & if \;\; y_i = 1 \; and \;\; h_\theta(x_i) \approx 1 \\ C_{TN_i} & if \;\; y_i = 0 \; and \;\; h_\theta(x_i) \approx 0 \\ C_{FP_i} & if \;\; y_i = 0 \; and \;\; h_\theta(x_i) \approx 1 \\ C_{FN_i} & if \;\; y_i = 1 \; and \;\; h_\theta(x_i) \approx 0 \end{cases}$$

- **Proposed Cost-Sensitive Function**

$$J^c(\theta) = \frac{1}{N}\sum_{i=1}^{N} y_i\left(h_\theta(x_i)C_{TP_i} + \left(1 - h_\theta(x_i)\right)C_{FN_i}\right) + (1 - y_i)\left(h_\theta(x_i)C_{FP_i} + \left(1 - h_\theta(x_i)\right)C_{TN_i}\right)$$

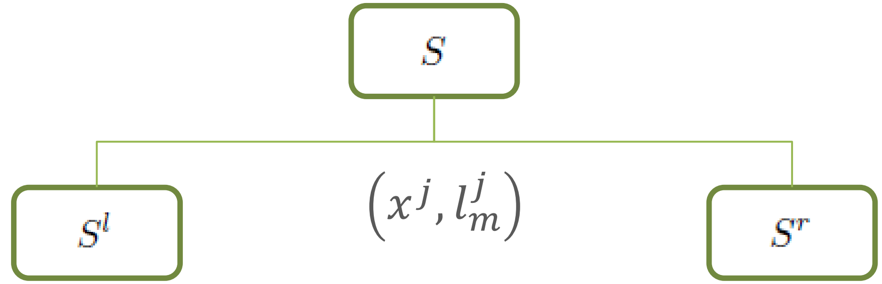
Cost-Sensitive Decision trees

- A decision tree is a classification model that iteratively creates **binary decision rules** $\left(x^j, l_m^j\right)$ that maximize certain criteria (gain, entropy, …). Where $\left(x^j, l_m^j\right)$ refers to making a rule using feature j on value m
- Maximize the accuracy is **different** than maximizing the cost.
- To solve this, some studies had been proposed method that aim to introduce the cost-sensitivity into the algorithms [Lomax 2013]. However, research have been focused on **class-dependent methods**

- We proposed:
  - **Example-dependent cost based impurity measure**
  - **Example-dependent cost based pruning criteria**

Cost-Sensitive Decision trees

Proposed Cost based impurity measure



$$S^l = \left\{x | x_i \in S \wedge x_i^j \leq l_m^j\right\} \qquad S^r = \left\{x | x_i \in S \wedge x_i^j > l_m^j\right\}$$

- The impurity of each leaf is calculated using:

$$I_c(S) = \min\left\{Cost\left(f_0(S)\right), Cost\left(f_1(S)\right)\right\}$$

$$f(S) = \begin{cases} 0 & if\ Cost\left(f_0(S)\right) \leq Cost\left(f_1(S)\right) \\ 1 & otherwise \end{cases}$$

- Afterwards the **gain** of applying a given rule to the set S is:

$$Gain_c\left(\left(x^j, l_m^j\right)\right) = I_c(\pi_1) - \left(I_c(\pi_1^l) + I_c(\pi_1^r)\right)$$

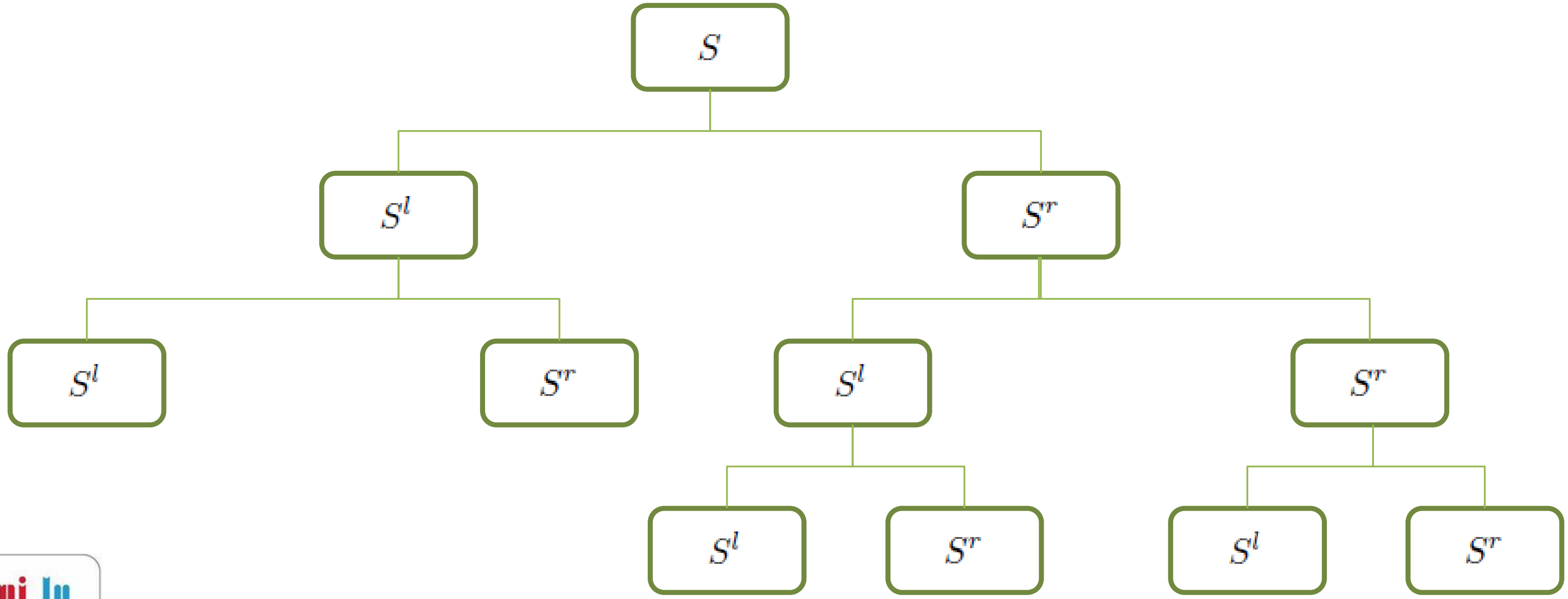
Cost-Sensitive Decision trees

Decision trees construction

- The rule that **maximizes the gain** is selected

$$(best_x, best_l) = arg \max_{(j,m)} \left(Gain\left((x^j, l_m^j) \right) \right)$$

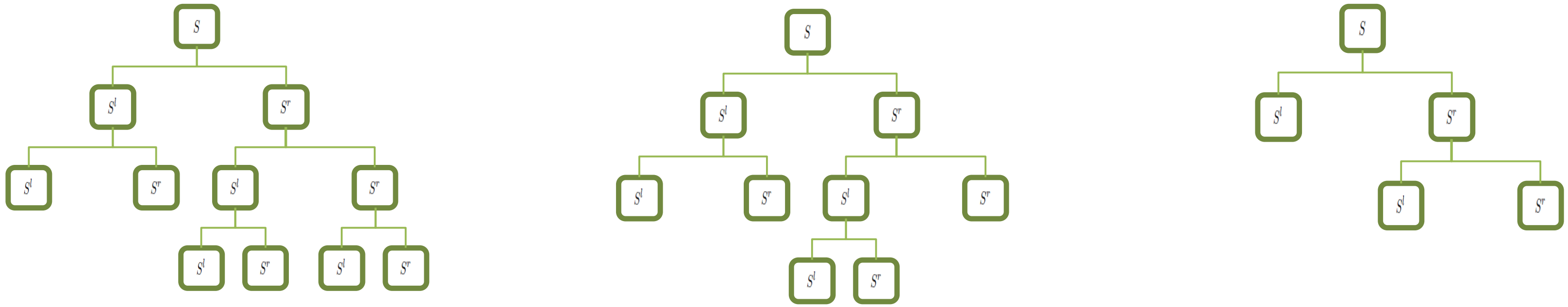
- The process is repeated until a stopping criteria is met:



Cost-Sensitive Decision trees

Proposed cost-sensitive pruning criteria

- Calculation of the **Tree savings** and pruned Tree savings



$$PC_c = \frac{Cost\big(f(S, Tree)\big) - Cost\Big(f\big(S, EB(Tree, branch)\big)\Big)}{|Tree| - |EB(Tree, branch)|}$$

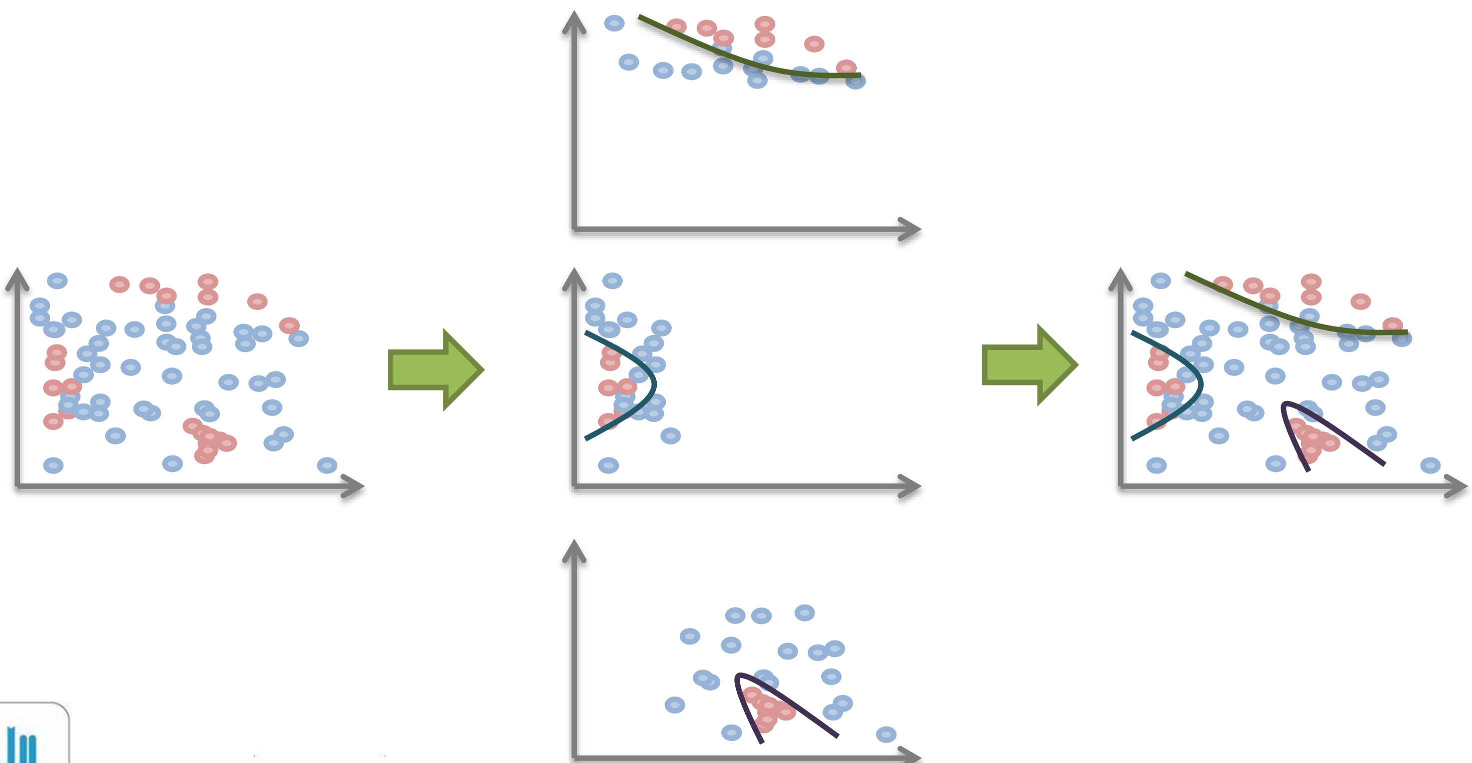
- After calculating the pruning criteria for all possible trees. The maximum improvement is selected and the Tree is pruned.
- Later the process is repeated until there is no further improvement.

Ensembles of Cost-Sensitive Decision trees

securityandtrust.lu

Typical ensemble is made by combining T different **base classifiers**. Each base classifiers is trained by applying algorithm M in a random subset

$$M_j \leftarrow M(S_j) \quad \forall_j \in \{1, \dots, T\}$$



UNIVERSITÉ DU LUXEMBOURG

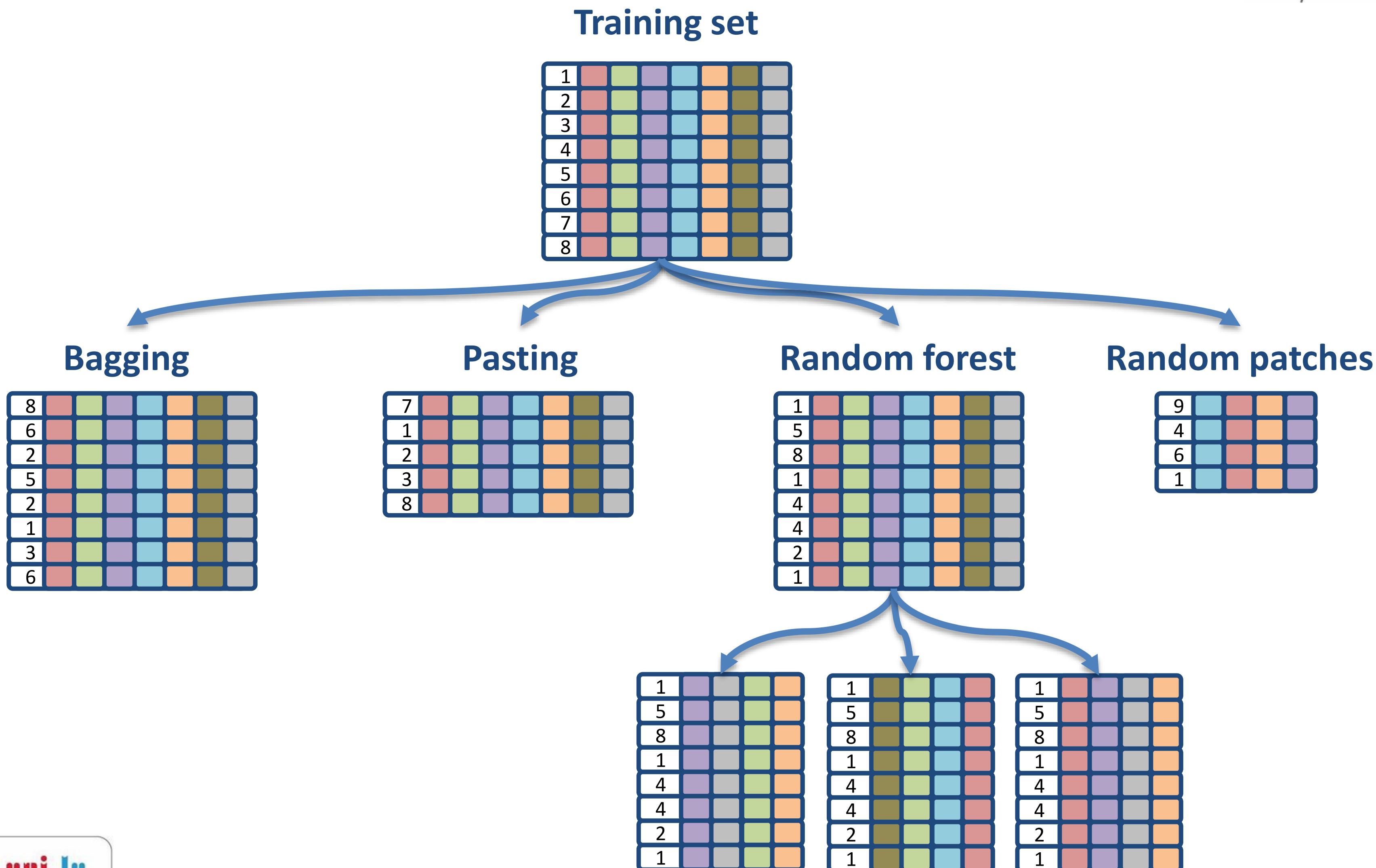
Ensembles of Cost-Sensitive Decision trees

securityandtrust.lu

The core principle in ensemble learning, is to **induce random perturbations** into the learning procedure in order to produce several different base classifiers from a single training set, then **combining** the base classifiers in order to make the final prediction.

UNIVERSITÉ DU LUXEMBOURG

Ensembles of Cost-Sensitive Decision trees



Ensembles of Cost-Sensitive Decision trees

After the base classifiers are constructed they are typically combined using one of the following methods:

- **Majority voting**

$$H(S) = f_{mv}(S,M) = arg \max_{c\in\{0,1\}} \sum_{j=1}^{T} 1_c\left(M_j(S)\right)$$

- **Proposed cost-sensitive weighted voting**

$$H(S) = f_{wv}(S,M,\alpha) = arg \max_{c\in\{0,1\}} \sum_{j=1}^{T} \alpha_j 1_c\left(M_j(S)\right)$$

$$\alpha_j = \frac{1-\varepsilon\left(M_j\left(S_j^{oob}\right)\right)}{\sum_{j_1=1}^{T} 1-\varepsilon\left(M_{j1}\left(S_{j1}^{oob}\right)\right)} \Rightarrow \alpha_j = \frac{Savings\left(M_j\left(S_j^{oob}\right)\right)}{\sum_{j_1=1}^{T} Savings\left(M_{j1}\left(S_{j1}^{oob}\right)\right)}$$

UNIVERSITÉ DU LUXEMBOURG

Ensembles of Cost-Sensitive Decision trees

securityandtrust.lu

- **Proposed cost-sensitive stacking**

$$H(S) = f_s(S, M, \beta) = \frac{1}{1 + e^{-\left(\sum_{j=1}^{T} \beta_j M_j(S)\right)}}$$

Using the cost-sensitive logistic regression [Correa et. al, 2014] model:

$$J(S, M, \beta) = \sum_{i=1}^{N} y_i \left(f_s(S, M, \beta)\left(C_{TP_i} - C_{FN_i}\right) + C_{FN_i}\right) + (1 - y_i)\left(f_s(S, M, \beta)\left(C_{FP_i} - C_{TN_i}\right) + C_{TN_i}\right)$$

Then the weights are estimated using

$$\hat{\beta} = arg \min_{\beta} J(S, M, \beta)$$

UNIVERSITÉ DU LUXEMBOURG

Agenda

- Motivation
- Cost-sensitive classification

  Background
- Real-world cost-sensitive applications

  Credit card fraud detection, churn modeling, credit scoring, direct marketing
- Proposed cost-sensitive algorithms

  Bayes minimum risk, cost-sensitive logistic regression, cost-sensitive decision trees, ensembles of cost-sensitive decision trees
- Experiments

  Experimental setup, results
- Conclusions

  Contributions, future work

Experimental setup - Datasets

Database	# Examples	% Positives	Cost (Euros)
Fraud	1,638,772	0.21%	860,448
Churn	9,410	4.83%	580,884
Kaggle Credit	112,915	6.74%	8,740,181
PAKDD09 Credit	38,969	19.88%	3,117,960
Direct Marketing	37,931	12.62%	59,507

Experimental setup - Methods

- **Cost-insensitive (CI):**
  - Decision trees (DT)
  - Logistic regression (LR)
  - Random forest (RF)
  - Under-sampling (u)
- **Cost-proportionate sampling (CPS):**
  - Cost-proportionate rejection-sampling (r)
  - Cost-proportionate over-sampling (o)
- **Bayes minimum risk (BMR)**
- **Cost-sensitive training (CST):**
  - Cost-sensitive logistic regression (CSLR)
  - Cost-sensitive decision trees (CSDT)

Experimental setup - Methods

- **Ensemble cost-sensitive decision trees (ECSDT):**

  Random inducers:
  - Bagging (CSB)
  - Pasting (CSP)
  - Random forest (CSRF)
  - Random patches (CSRP)

  Combination:
  - Majority voting (mv)
  - Cost-sensitive weighted voting (wv)
  - Cost-sensitive staking (s)

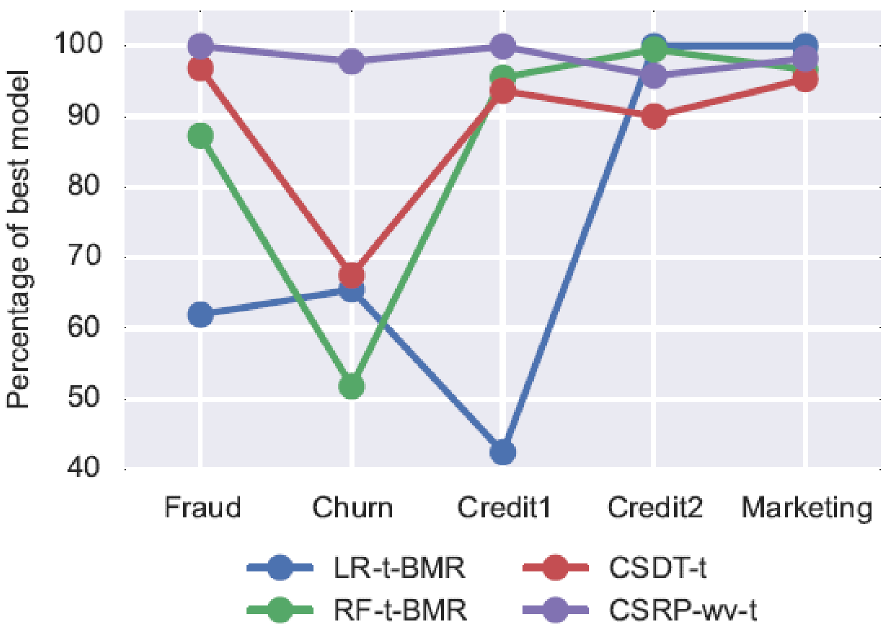
Experimental setup

- Each experiment was carried out **50 times**
- For the parameters of the algorithms a **grid search** was made
- Results are measured by **savings and F1Score**
- Then the **Friedman ranking** is calculated for each method

Results

Database	Algorithm	Savings	Savings (Euros)	% Pos.
Fraud	CSRP-wv-t	0.73	628,127	0.21
Churn	CSRP-s-t	0.17	98,750	4.83
Credit1	CSRP-mv-t	0.52	4,544,894	6.74
Credit2	LR-t-BMR	0.31	966,568	19.88
Marketing	LR-t-BMR	0.51	30,349	12.62

Percentage of the **highest savings**



Results

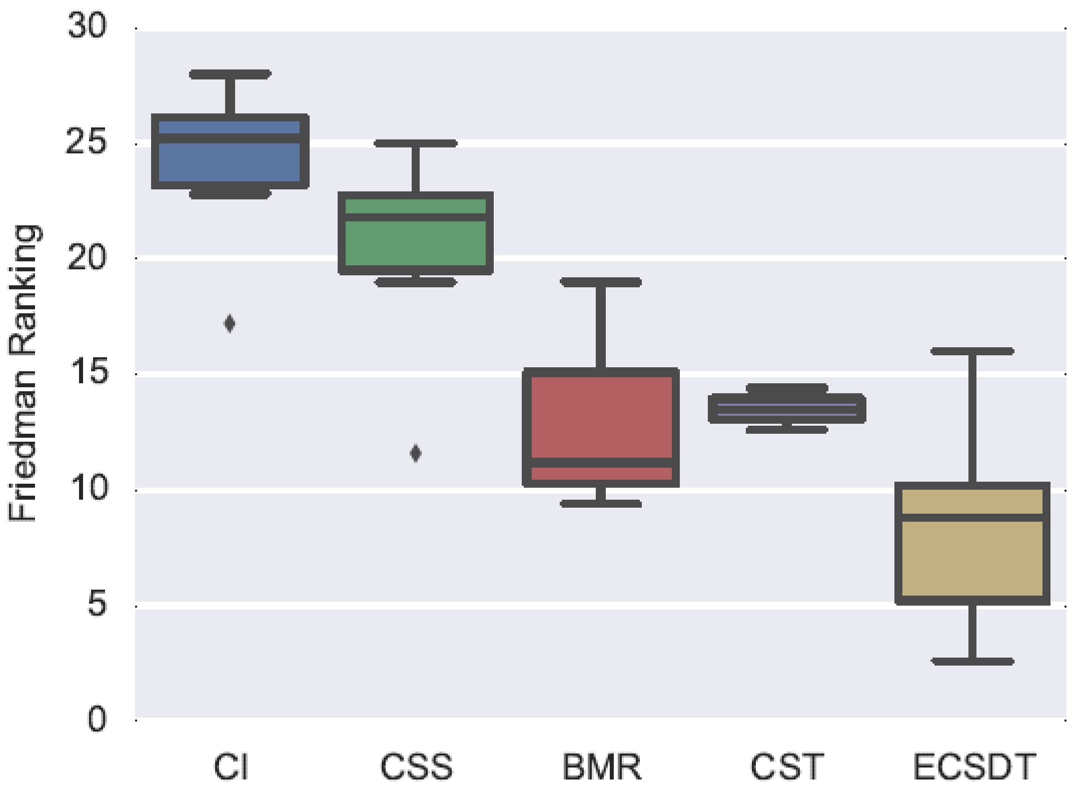
Results of the **Friedman rank** of the savings (1=best, 28=worst)

Family	Algorithm	Rank
ECSDT	**CSRP-wv-t**	**2.6**
ECSDT	CSRP-s-t	3.4
ECSDT	CSRP-mv-t	4
ECSDT	CSB-wv-t	5.6
ECSDT	CSP-wv-t	7.4
ECSDT	CSB-mv-t	8.2
ECSDT	CSRF-wv-t	9.4
BMR	**RF-t-BMR**	**9.4**
ECSDT	CSP-s-t	9.6
ECSDT	CSP-mv-t	10.2
ECSDT	CSB-s-t	10.2
BMR	**LR-t-BMR**	**11.2**
CPS	**RF-r**	**11.6**
CST	**CSDT-t**	**12.6**

Family	Algorithm	Rank
CST	CSLR-t	14.4
ECSDT	CSRF-mv-t	15.2
ECSDT	CSRF-s-t	16
CI	**RF-u**	**17.2**
CPS	LR-r	19
BMR	DT-t-BMR	19
CPS	LR-o	21
CPS	DT-r	22.6
CI	LR-u	22.8
CPS	RF-o	22.8
CI	DT-u	24.4
CPS	DT-o	25
CI	DT-t	26
CI	RF-t	26.2

Results

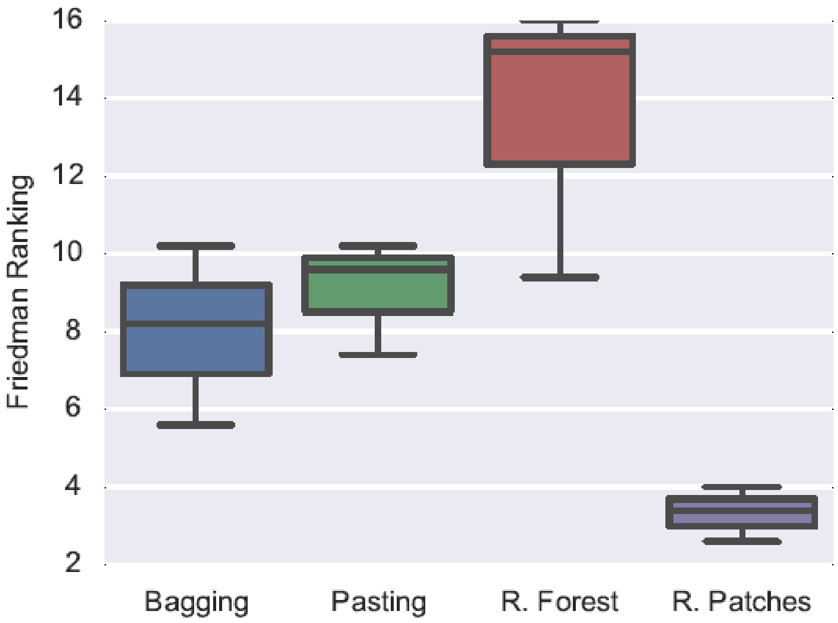
Results of the **Friedman rank** of the savings organized by family



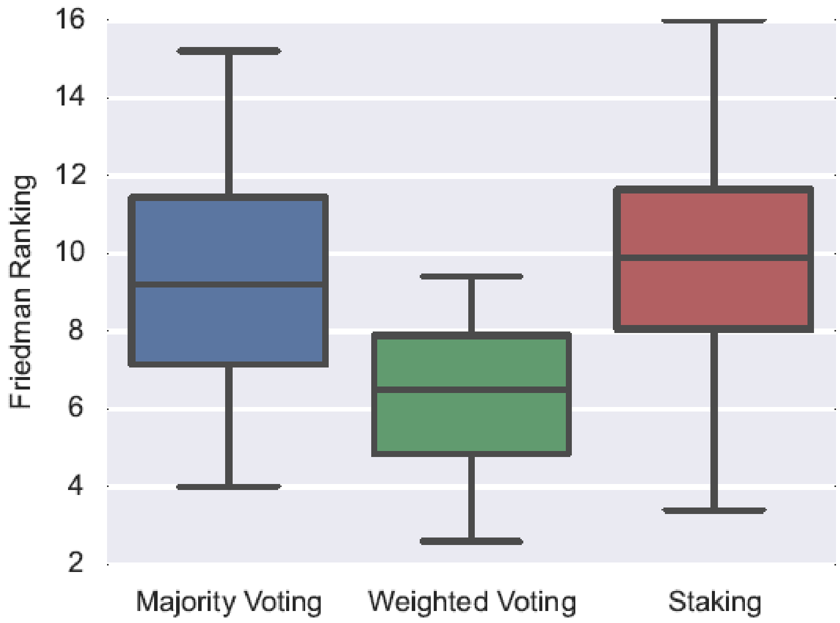
Results within the ECSDT family

securityandtrust.lu

By random inducer



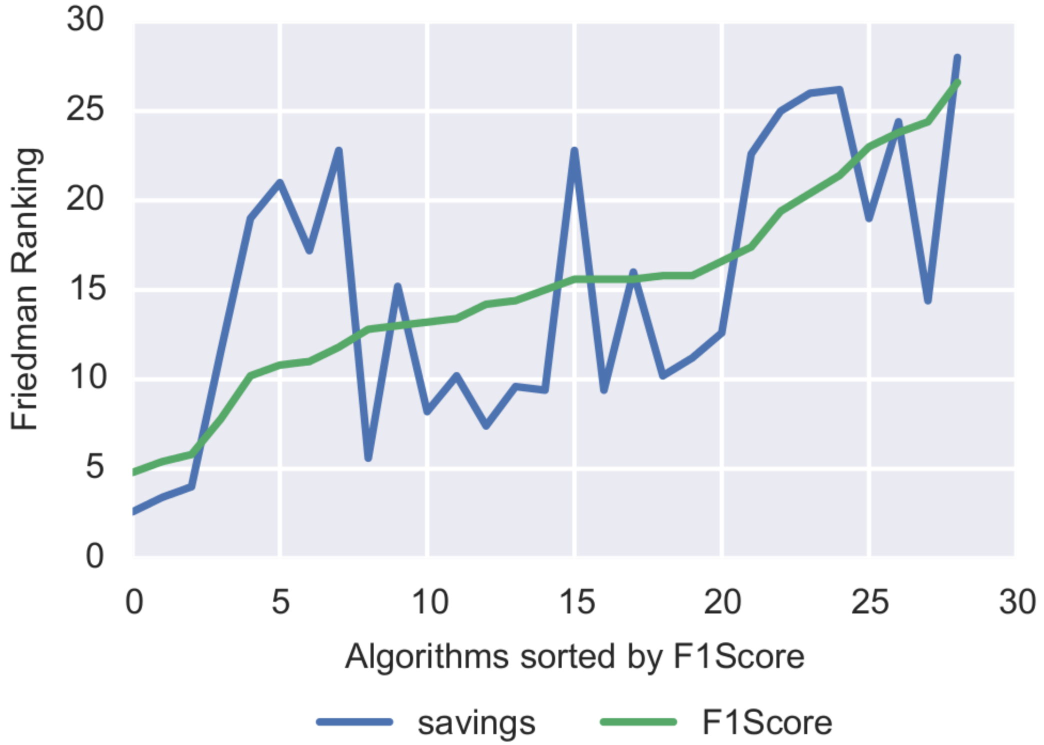
By combination method



UNIVERSITÉ DU LUXEMBOURG

Results

Comparison of the Friedman ranking of the **savings** and **F1Score** sorted by F1Score ranking



Agenda

- Motivation
- Cost-sensitive classification

  Background
- Real-world cost-sensitive applications

  Credit card fraud detection, churn modeling, credit scoring, direct marketing
- Proposed cost-sensitive algorithms

  Bayes minimum risk, cost-sensitive logistic regression, cost-sensitive decision trees, ensembles of cost-sensitive decision trees
- Experiments

  Experimental setup, results
- Conclusions

  Contributions, future work

Conclusions

- New framework of **example dependent cost-sensitive classification**
- Using five databases, from four real-world applications: credit card fraud detection, churn modeling, credit scoring and direct marketing, we show that the **proposed algorithms significantly outperforms** the state-of-the-art cost-insensitive and example-dependent cost-sensitive algorithms
- Highlight the importance of using the **real example-dependent financial costs** associated with the real-world applications

Future research directions

- Multi-class example-dependent cost-sensitive classification
- Cost-sensitive calibration
- Staking cost-sensitive decision trees
- Example-dependent cost-sensitive boosting
- Online example-dependent cost-sensitive classification

Contributions - Papers

Date	Name	Conference / Journal	Status
July 2013	Cost Sensitive Credit Card Fraud Detection using Bayes Minimum Risk	IEEE International Conference on Machine Learning and Applications	Published
October 2013	Improving Credit Card Fraud Detection with Calibrated Probabilities	SIAM International Conference on Data Mining	Published
June 2014	Credit Scoring using Cost-Sensitive Logistic Regression	IEEE International Conference on Machine Learning and Applications	Published
October 2014	Example-Dependent Cost-Sensitive Decision Trees	Expert Systems with Applications	Published
January 2015	A novel cost-sensitive framework for customer churn predictive modeling	Decision Analytics	Published
March 2015	Ensemble of Example-Dependent Cost-Sensitive Decision Trees	IEEE Transactions on Knowledge and Data Engineering	Under review
March 2015	Feature Engineering Strategies for Credit Card Fraud Detection	Expert Systems with Applications	Under review
June 2015	Detecting Credit Card Fraud using Periodic Features	IEEE International Conference on Machine Learning and Applications	In Press

Contributions - Costcla

costcla is a Python module for **cost-sensitive classification** built on top of Scikit-Learn, SciPy and distributed under the 3-Clause BSD license.

In particular, it provides:

- A set of example-dependent cost-sensitive algorithms
- Different real-world example-dependent cost-sensitive datasets.

Installation

pip install costcla

Documentation: https://pythonhosted.org/costcla/

Prepare dataset and load libraries

In [38]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.cross_validation import train_test_split
from costcla.metrics import savings_score
from costcla.datasets import load_creditscoring2
from costcla.sampling import cost_sampling
from costcla import models
data = load_creditscoring2()
X_train, X_test, y_train, y_test,
cost_mat_train, cost_mat_test = \
train_test_split(data.data, data.target, data.cost_mat)
```

Random forest

In [19]:
```
f_RF = RandomForestClassifier()
y_pred = f_RF.fit(X_train, y_train).predict(X_test)
print savings_score(y_test, y_pred, cost_mat_test)
```

```
0.042197359989
```

cost-sensitive decision tree

In [2]:
```
f_CSDT = models.CSDecisionTreeClassifier()
f_CSDT.fit(data.data, data.target, data.cost_mat)
y_pred = f_CSDT.predict(data.data)
print savings_score(data.target, y_pred, data.cost_mat)
```

```
0.289489571352
```

cost-sensitive random patches

In [33]:
```
f_CSRP = costcla.models.CSRandomPatchesClassifier()
f_CSRP.fit(data.data, data.target,  data.cost_mat)
y_pred = f_CSRP.predict(data.data)
print savings_score(data.target, y_pred, data.cost_mat)
```

```
0.306607400467
```

Thank You!!

Alejandro Correa Bahnsen

Contact information

Alejandro Correa Bahnsen

University of Luxembourg

albahnsen.com

al.bahnsen@gmail.com

http://www.linkedin.com/in/albahnsen

https://github.com/albahnsen/CostSensitiveClassification