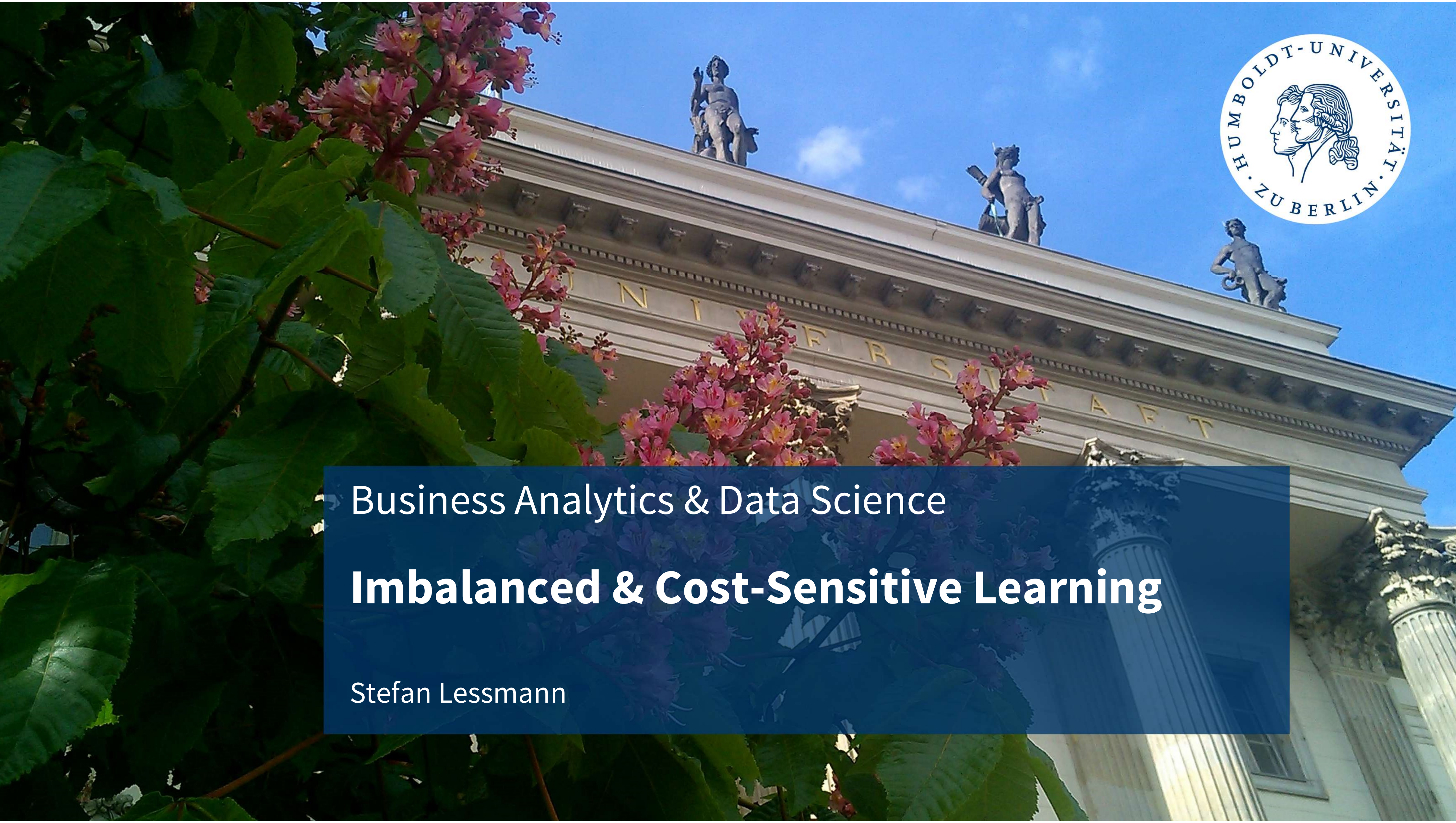


Business Analytics & Data Science

Imbalanced & Cost-Sensitive Learning

Stefan Lessmann

Agenda

Introduction

Imbalanced Learning

Model evaluation, class imbalance and model development

Cost Sensitive Learning

Expected value framework and Bayes decision theory, strategies for cost-sensitive model development, from class- to case-specific costs

Summary

Introduction

Introduction

Business apps often exhibit class imbalance & asymmetric error costs

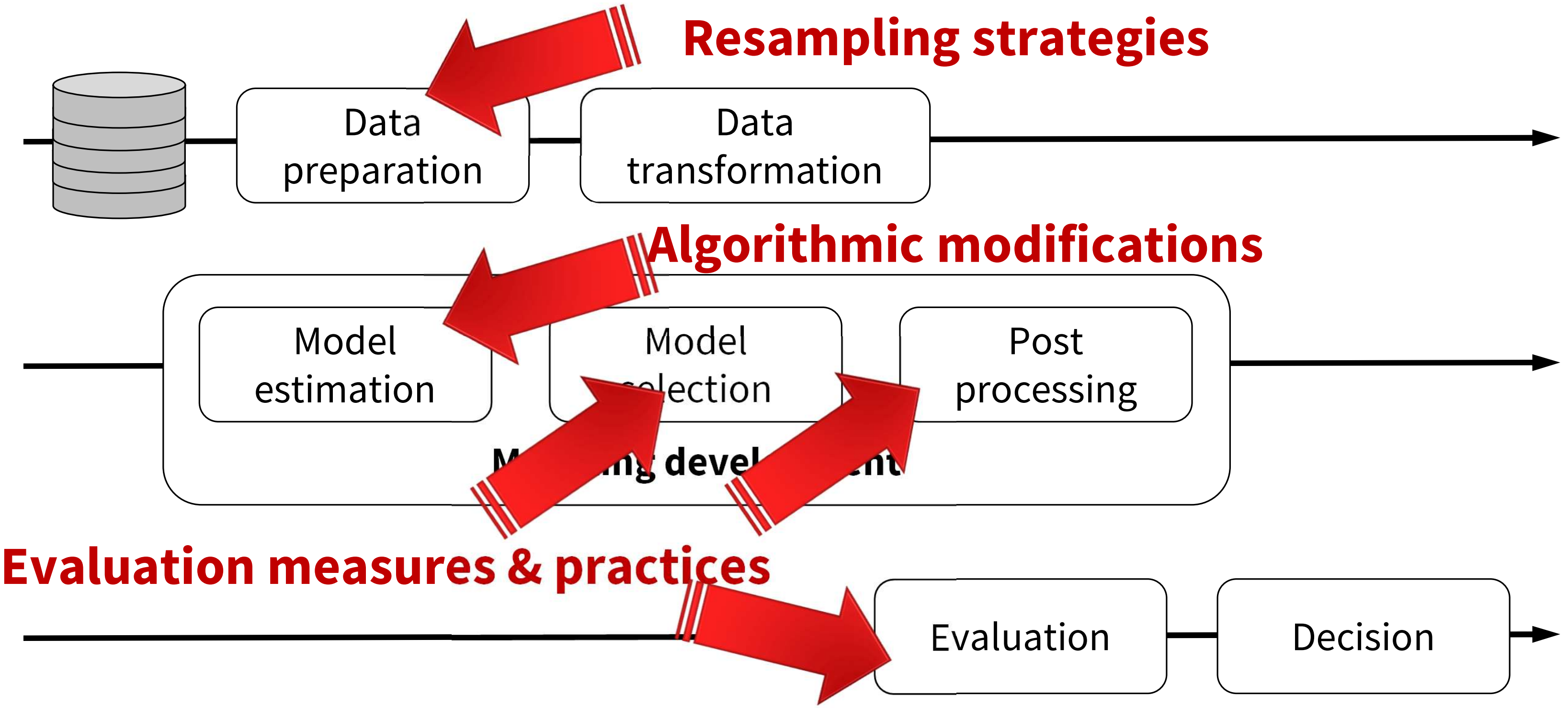
Example apps recurring in class	Common denominator
■ **Credit scoring** ■ **Customer churn** ■ **Direct marketing** ■ **Fraud discovery** ■ **…**	■ **Binary classification** □ One class of interest vs. rest □ One economically relevant target ■ **Relevant class is often a minority** ■ **The error costs are often asymmetric**

Introduction
Scope of this lecture

- **Imbalance learning**
  - ☐ Examples of one class heavily outnumbered by those of the other class
  - ☐ Modeling techniques to overcome detrimental effect of class skew
- **Cost sensitive learning**
  - ☐ Misclassification errors are associated with different costs
  - ☐ Error cost depend on class (or case → extension)
  - ☐ Modeling techniques to address asymmetry among error costs
- **Both are more developed in classification c.f. regression modeling**
  - ☐ For regression examples, see, e.g., Granger (1969), Crone (2010), Dress et al. (2018)
  - ☐ Use of asymmetric costs of error functions

Introduction
Predictive modeling process

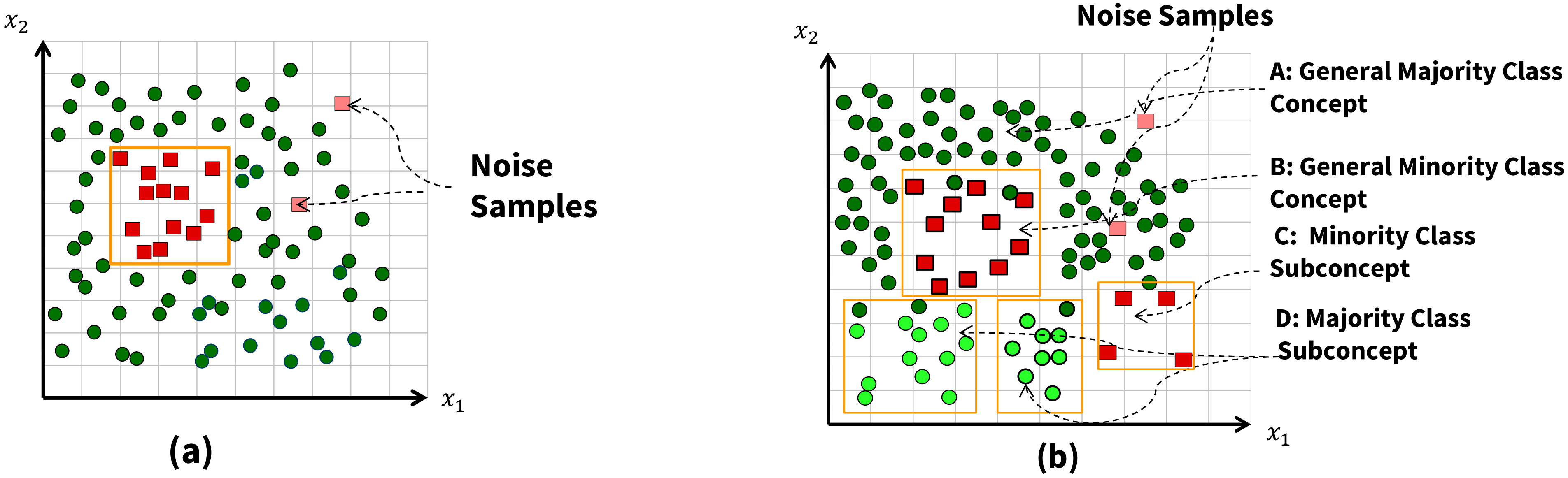
- **Multi-step approach to develop prediction model**
- **Address imbalance/cost-asymmetry at different points**



Imbalanced Learning

Model evaluation, class imbalance and model development

Nature of the Class Imbalance Problem



(a) A data set with a between-class imbalance.

(b) A high complexity data set with both between-class and within-class imbalances, multiple concepts, overlapping, noise, and lack of representative data. (He & Garcia, 2009)

Model Evaluation in the Face of Class Imbalance

Recap: confusion matrix, classification accuracy, and cousins

		Actual Class	
		Positive $(Y = 1)$	**Negative** $(Y = 0)$
Predicted Class	**Positive** $(\hat{Y} = 1)$	True Positive (TP)	False Positive (FP)
	Negative $(\hat{Y} = 0)$	False Negative (FN)	True Negative (TN)

- **Classification accuracy / Percentage correctly classified** $$\frac{TP + TN}{TP + TN + FP + FN}$$
- **Classification error** $$\frac{FP + FN}{TP + TN + FP + FN}$$
- **Specificity** $$\frac{TN}{TN + FP}$$
- **Sensitivity / Recall** $$\frac{TP}{TP + FN}$$
- **Precision** $$\frac{TP}{TP + FP}$$

Model Evaluation in the Face of Class Imbalance

Shortcomings of classification accuracy / error

Confusion matrix for imbalanced data

		Actual Class	
		Positive $(Y = 1)$	**Negative** $(Y = 0)$
Predicted Class	**Positive** $(\hat{Y} = 1)$	2	0
	Negative $(\hat{Y} = 0)$	0	98

- **Example: 100 hundred cases; class ratio = 98:2**
- **Result**
  - Classifier achieves optimal performance
  - PCC = 100%
  - Error rate = 0%

Perfect classifier

Model Evaluation in the Face of Class Imbalance

Shortcomings of classification accuracy / error

Confusion matrix for imbalanced data

		Actual Class	
		Positive $(Y = 1)$	**Negative** $(Y = 0)$
Predicted Class	**Positive** $(\hat{Y} = 1)$	0	0
	Negative $(\hat{Y} = 0)$	2	98

- **Example: 100 hundred cases; class ratio = 98:2**
- **Result**
  - Classifier looks as if it achieves near-optimal performance
  - PCC = 98%
  - Error rate = 2%
  - But it does not…

Naïve classifier

Model Evaluation in the Face of Class Imbalance

Threshold metrics with higher robustness toward class skew

		Actual Class	
		Positive $(Y = 1)$	**Negative** $(Y = 0)$
Predicted Class	**Positive** $(\hat{Y} = 1)$	True Positive (TP)	False Positive (FP)
	Negative $(\hat{Y} = 0)$	False Negative (FN)	True Negative (TN)

- **G-Mean (geometric mean of sensitivity and specificity)**

$$\sqrt{\frac{TP}{TP + FN} \cdot \frac{TN}{TN + FP}}$$

- **F-Measure (weighted geometric mean of precision and recall:**

$$\frac{(\mathbf{1} + \boldsymbol{\beta})^{2} \cdot \boldsymbol{Recall} \cdot \boldsymbol{Precision}}{\boldsymbol{Recall} + \boldsymbol{Precision}}$$

with precision = $\frac{TP}{TP+FP}$

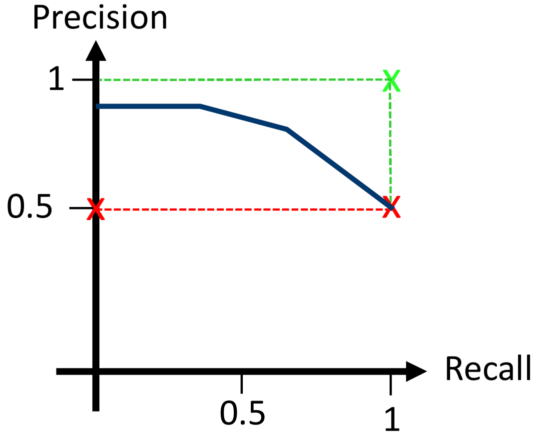
and recall = $\frac{TP}{TP+FN}$

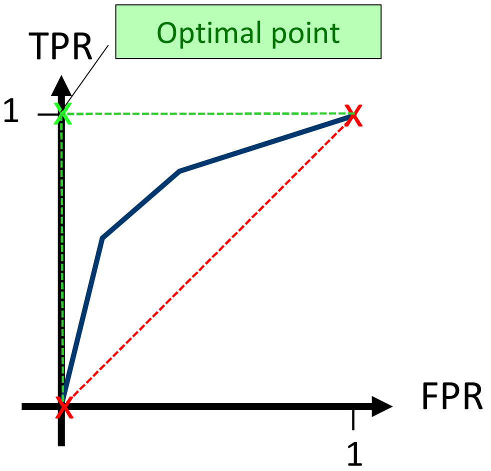
Model Evaluation in the Face of Class Imbalance

Graphical evaluation frameworks

- **Receiver Operating Characteristics Curves**
  - True positive rate (TPR) = sensitivity/recall= $\frac{TP}{TP+FN}$
  - False positive rate (FPR) = 1-specificity = $1 - \frac{TN}{TN+FP}$
- **Alternative charts**
  - Cost-Curves (Drummond/Holte, 2006)
  - Brier-Curves (Hernández-Orallo et al., 2011)
  - Precision-Recall Curves
    - Precision = $\frac{TP}{TP+FP}$
    - Random baseline depends on class ratio (e.g., 1:1)

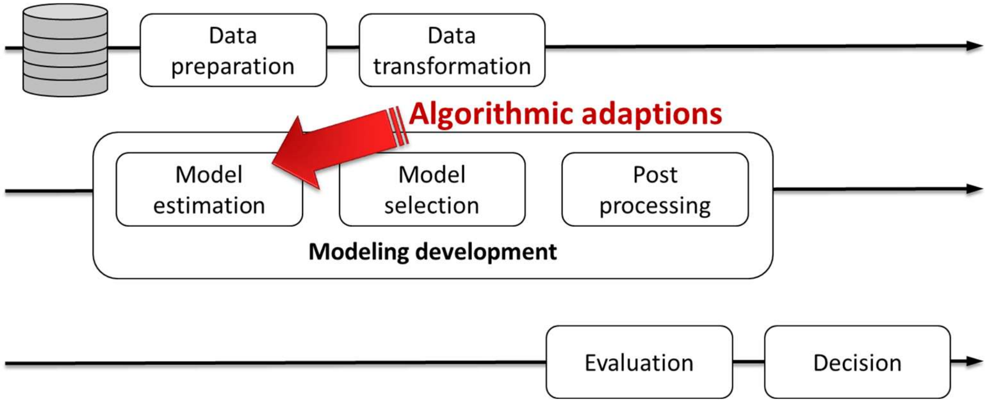
		Actual Class	
		Positive $(Y = 1)$	**Negative** $(Y = 0)$
Predicted Class	**Positive** $(\hat{Y} = 1)$	True Positive (TP)	False Positive (FP)
	Negative $(\hat{Y} = 0)$	False Negative (FN)	True Negative (TN)





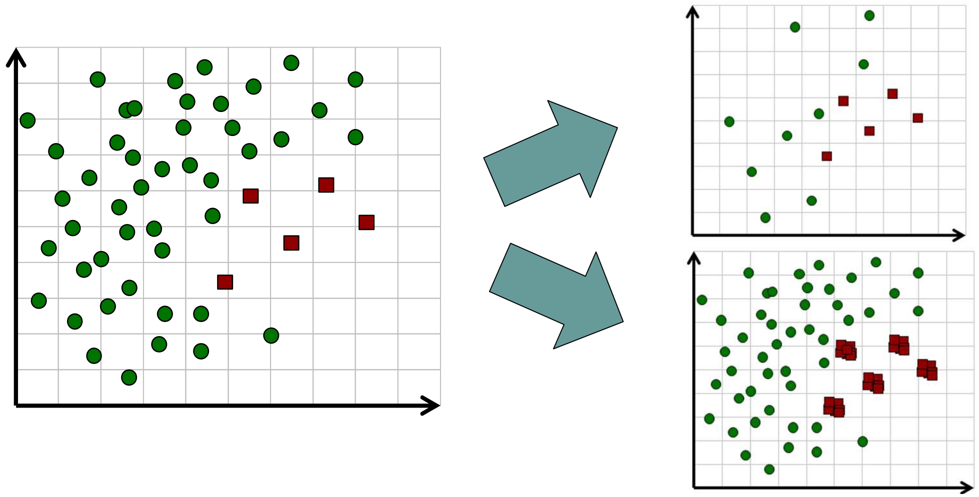
Class Imbalance and Model Development

- **Class imbalance affects model development**
  - **Model selection**: evaluation of candidate models requires suitable performance indicators (see above)
  - **Model estimation**: optimization of some functional that captures how well the model fits the data (e.g., training error)
- **Empirical risk minimization principle**
  - Entropy: $I(N) = -\sum_j p(c_j|N) \cdot \log_2\left(p(c_j|N)\right)$
  - Max. Likelihood: $\sum_{i=1}^{n} y_i \log\left(p(y_i = 1|\boldsymbol{x}_i)\right) + (1 - y_i)\log(1 - p(y_i = 1|\boldsymbol{x}_i))$
  - See King & Langche (2001) for formal analysis of the logit model in rare event settings
- **Fit calculation overemphasizes majority class examples**
  - Similar problem as in evaluation
  - Performance indicators like AUC less suitable for model fitting



Resampling Strategies
Random over- and undersampling

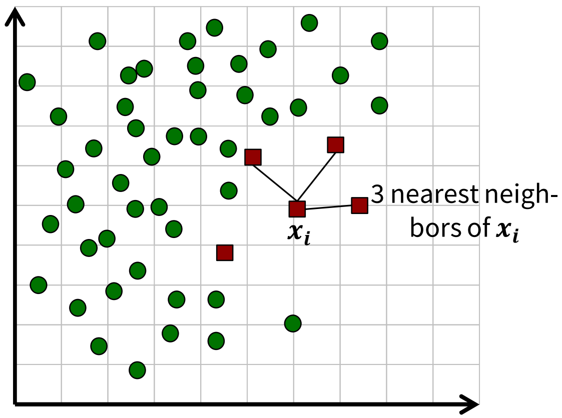
Undersampling	Oversampling
■ Random deletion of majority class instances ■ Reduces training time ■ May discard useful information	■ Random duplication of minority class instances ■ No information loss ■ Increased training time

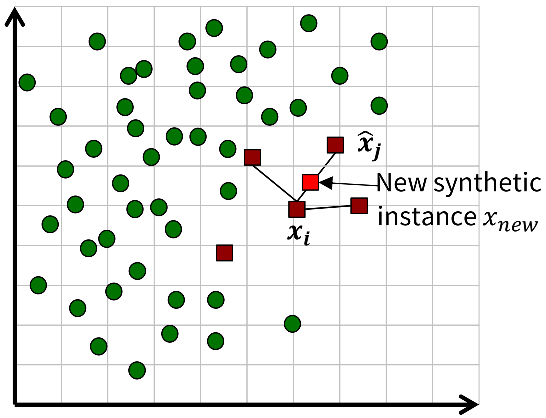


Literature disagrees on which approach works better.

Resampling Strategies
Synthetic sampling with data generation

- **Synthetic Minority Class Oversampling (SMOTE)**
- **For each minority class case $\boldsymbol{x_i}$**
  - Identify *K* nearest neighbors of minority class
  - Randomly select one of these neighbors $\hat{\boldsymbol{x}}_j$
    - Calculate feature vector difference
    - Multiply with random number $\delta \in [0,1]$
    - Add result to $\boldsymbol{x_i}$
  - $\boldsymbol{x_{new}} = \boldsymbol{x_i} + \left(\boldsymbol{x_i} - \hat{\boldsymbol{x}}_j\right) \cdot \delta$
  - New instance is a point along the line segment connecting minority class case $\boldsymbol{x_i}$ and its randomly selected nearest neighbor $\hat{\boldsymbol{x}}_j$
- **For details see Chawla et al. (2002)**





Cost-Sensitive Learning

Bayes decision theory, strategies for cost-sensitive model development, from class- to case-specific costs

Bayes Decision Theory
Predict the class with minimum risk

- **Multiple possible decisions $\mathcal{D} = \{1, \dots, D\}$**
  - In our context, decisions equate to classes
  - Classifying an instance into a class c implies that we take a certain action for that instance
  - So we assume a multi-class classification setting with $|\mathcal{D}|$ different classes
- **Calculation of risk R follows the expected value principle**

$$R(c|\boldsymbol{x}) = \sum_{d \in \mathcal{D}} p(d|\boldsymbol{x}) C(c, d)$$

  - c denotes the predicted class
  - d denotes the actual class
  - $C(c, d)$ is a cost matrix that gives the cost of taking decision c if the true class is d
  - $p(d|\boldsymbol{x})$ is the conditional probability that instance $\boldsymbol{x}$ belongs to class d

Bayes Decision Theory

Two class credit scoring setting

- **Assume two classes representing GOOD and BAD credit applicants**
  - Action implied by classification as BAD is to reject the loan application
  - Action implied by classification as GOOD is to approve the loan application
- **Risk minimization illustrated**

$$R(c|\boldsymbol{x}) = \sum_{d \in \mathcal{D}} p(d|\boldsymbol{x}) C(c,d)$$

$$R(c = \mathrm{GOOD}|\boldsymbol{x}) = p(d = \mathrm{GOOD}|\boldsymbol{x}) C(c = \mathrm{GOOD}, d = \mathrm{GOOD}) + p(d = \mathrm{BAD}|\boldsymbol{x}) C(c = \mathrm{GOOD}, d = \mathrm{BAD})$$

$$R(c = \mathrm{BAD}|\boldsymbol{x}) = p(d = \mathrm{GOOD}|\boldsymbol{x}) C(c = \mathrm{BAD}, d = \mathrm{GOOD}) + p(d = \mathrm{BAD}|\boldsymbol{x}) C(c = \mathrm{BAD}, d = \mathrm{BAD})$$

- **Risk minimization rule: predict BAD, iif:** $R(c = \mathrm{BAD}|\boldsymbol{x}) < R(c = \mathrm{GOOD}|\boldsymbol{x})$

Bayes Decision Theory
Bayes optimal classification cut-off

- **Simplification of notation**
  - Let G and B represent GOOD and BAD credit applicants
  - Let g and b denote the corresponding model classifications (i.e., predictions) of class GOOD and BAD
- **Recall the risk minimization rule: predict BAD, iif:** $R(b|\boldsymbol{x}) < R(g|\boldsymbol{x})$

$$p(B|\boldsymbol{x})C(b,B) + p(G|\boldsymbol{x})C(b,G) < p(G|\boldsymbol{x})C(g,G) + p(B|\boldsymbol{x})C(g,B)$$

- **After rearranging terms, we obtain:**

$$p(B|\boldsymbol{x})\big(C(b,B) - C(g,B)\big) < p(G|\boldsymbol{x})\big(C(g,G) - C(b,G)\big)$$

- **Recall that** $p(G|\boldsymbol{x}) = 1 - p(B|\boldsymbol{x})$
- **Optimal threshold to classify x as BAD, iif:**

$$p(B|\boldsymbol{x}) \geq \tau^* = \frac{\big(C(b,G) - C(g,G)\big)}{\big(C(b,G) + C(g,B) - C(b,B) - C(g,G)\big)}$$

Cost-benefit matrix		Actual Class: GOOD ($Y=1$)	Actual Class: BAD ($Y=0$)
Predicted Class	good ($\hat{Y}=1$)	$C(g,G)$	$C(g,B)$
Predicted Class	bad ($\hat{Y}=0$)	$C(b,G)$	$C(b,B)$

Bayes Decision Theory

Risk minimization rule revisited

- **Risk minimization rule for class BAD after rearranging terms (repeated)**

$$p(B|\boldsymbol{x})\big(C(b,B) - C(g,B)\big) < p(G|\boldsymbol{x})\big(C(g,G) - C(b,G)\big)$$

- **Implication: classification decision does not change if we add a constant to a column of the cost-benefit matrix**
  - Allows re-scaling the cost-benefit matrix to obtain a proper cost-matrix
  - Standard form for cost-sensitive learning

Cost matrix		**Actual Class**	
		GOOD $(Y=1)$	**BAD** $(Y=0)$
Predicted Class	**good** $(\hat{Y}=1)$	0	$C(g,B) - C(b,B)$
	bad $(\hat{Y}=0)$	$C(b,G) - C(g,G)$	0

Cost-benefit matrix		**Actual Class**	
		GOOD $(Y=1)$	**BAD** $(Y=0)$
Predicted Class	**good** $(\hat{Y}=1)$	$C(g,G)$	$C(g,B)$
	bad $(\hat{Y}=0)$	$C(b,G)$	$C(b,B)$

Bayes Decision Theory
Not the actual costs but their ratio determines classification decision

- **Recall Bayes optimal threshold: classify x as BAD, iif:**

$$p(B|\boldsymbol{x}) \geq \tau^* = \frac{\left(C(b,G) - C(g,G)\right)}{\left(C(b,G) + C(g,B) - C(b,B) - C(g,G)\right)}$$

Cost-benefit matrix		Actual Class	
		GOOD ($Y = 1$)	BAD ($Y = 0$)
Predicted Class	good ($\hat{Y} = 1$)	$C(g,G)$	$C(g,B)$
	bad ($\hat{Y} = 0$)	$C(b,G)$	$C(b,B)$

- **Given that we can convert a cost-benefit matrix into a cost matrix, we can, without loss of generality assume zero costs for correct classifications**
- **Bayes optimal, cost-minimal threshold for a given cost matrix**

$$p(b|\boldsymbol{x}) \geq \tau^* = \frac{C'(b,G)}{C'(b,G) + C'(g,B)}$$

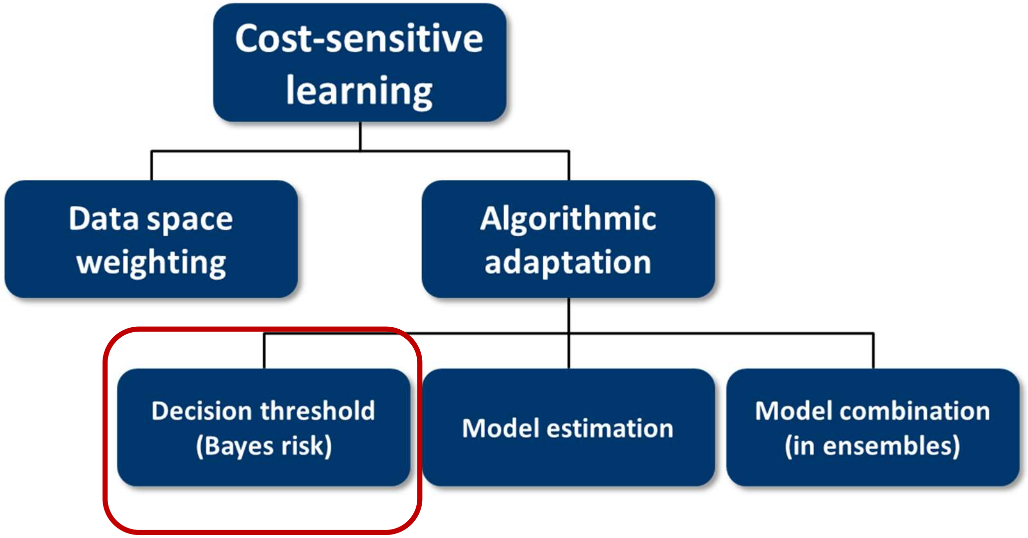
Cost matrix		Actual Class	
		GOOD ($Y = 1$)	BAD ($Y = 0$)
Predicted Class	good ($\hat{Y} = 1$)	0	$C(g,B) - C(b,B)$
	bad ($\hat{Y} = 0$)	$C(b,G) - C(g,G)$	0

  - Where $C'(\cdot,\cdot)$ denotes an entry of the cost matrix,
  - which we obtain by adding suitable constants to the columns of the cost-benefit matrix
  - So $C'(b,G) = C(b,G) - C(g,G)$ and $C'(g,B) = C(g,B) - C(b,B)$

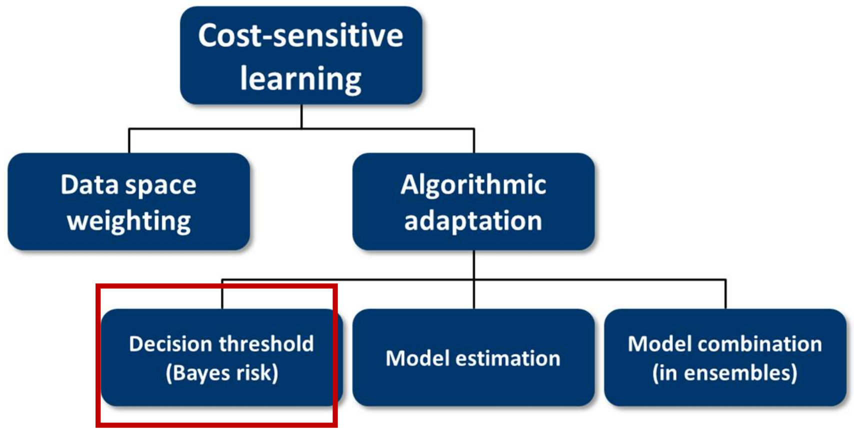
Bayes Decision Theory

Cost-minimal classification cut-off facilitates cost-sensitive classification

- **Way to make any classifier cost-sensitive**
  - Obtain probability forecast $\hat{p}(Y = 1|\boldsymbol{x})$
  - Apply optimal cut-off τ^* to predict class
- **Foundation of many CSL approaches**
- **Caveat**
  - Classifier must produce ‘good’ probability estimates (i.e., be well calibrated)
  - May be difficult to achieve in imbalanced settings



Bayes Decision Theory
Empirical thresholding



- **Alternative to cost-optimal threshold**
  - Use cross-validation to tune the cut-off
  - No accurate estimate of probability required
  - Accurate classifier ranking suffices
  - See, e.g., Sheng & Ling (2006)
- **Additional meta-parameter**
  - Rule of thumb:
    - Rank order predictions
    - Set cut-off such that the share of predicted positives equals the prior probability of the positive class in the training set
  - Empirical tuning (supported by e.g., `sklearn`)

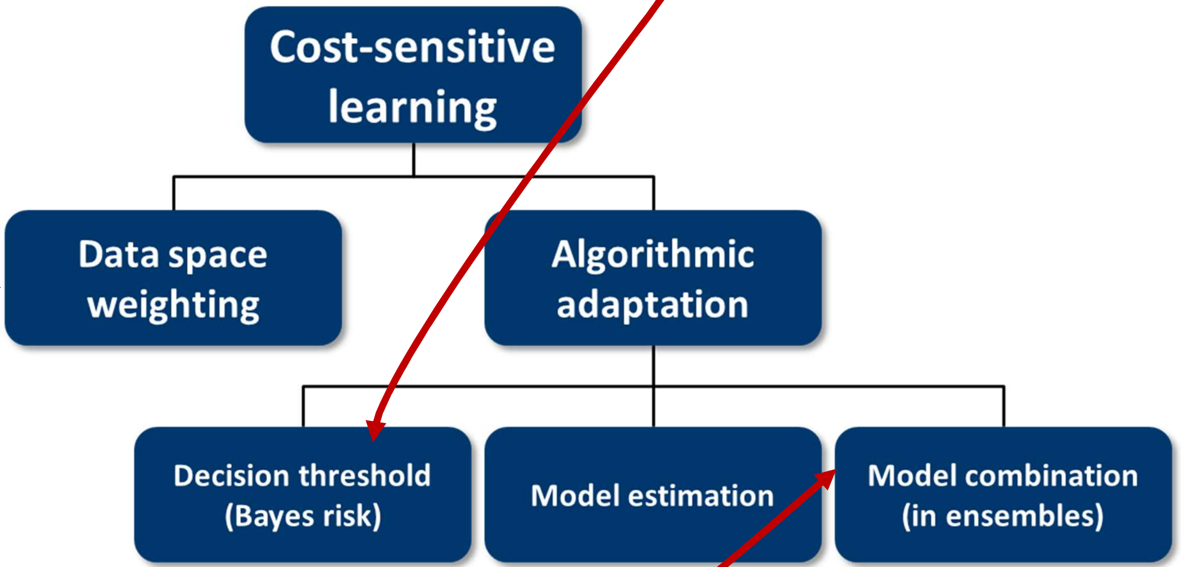
Say p(BAD)=20%

$\tau = 0.71$

ID	p(BAD|*x*)
1	0.9
2	0.7
3	0.6
4	0.6
5	0.2

Cost-Sensitive Learning
Interim conclusion

- **Bayesian decision theory**
  - Generic approach for cost-sensitive learning
  - Algorithm level adaptation
- **Up next**
  - Data space weighting
    - MetaCost
    - Adaptive boosting
  - More algo-level adaptation
    - Decision Trees
    - Neural networks
    - Others



Strategies for Cost-Sensitive Model Development
MetaCost (Domingos, 1999)

- **Generic approach to make classifiers cost sensitive**
- **Starts from the Bayes risk (see above)** $R(c|\boldsymbol{x}) = \sum_{d \in \mathcal{D}} p(d|\boldsymbol{x}) C(c, d)$
- **Relabels training cases according to** $R(c|\boldsymbol{x})$
- **Three-step procedure**
  - Develop classification model
    - Needed to estimate $p(c|\boldsymbol{x})$
    - Domingos (1999) recommends bagging
  - Relabel training data
    - Assign each case to the cost-minimal $c_i = \operatorname{argmin}_{c_i} \sum_{d \in \mathcal{D}} p(d|\boldsymbol{x}_i) C(c_i, d)$
    - Based on classifier and cost-matrix
  - Develop final classifier from relabeled training set

Strategies for Cost-Sensitive Model Development

Adaptive Boosting (Sun et al., 2007)

- **Introduces costs into weight-updating of Adaboost**
- **Recall Adaboost idea**
  - Construct weight distribution over the training data
  - Update equation: $D_{t+1}(i) = D_t(i)exp(-\alpha_t h_t(\boldsymbol{x}_i)y_i)/Z_t$
  - Where *t*=iteration, *i*=case index, *h*(***x***)=classifier, *y*=class label, α=error dependent classifier weight, *Z*=normalization factor
- **Alternative ways to incorporate costs**
  - In exponential: $D_{t+1}(i) = D_t(i)exp(-\alpha_t C_i h_t(\boldsymbol{x}_i)y_i)/Z_t$
  - Out of exponential: $D_{t+1}(i) = C_i D_t(i)exp(-\alpha_t h_t(\boldsymbol{x}_i)y_i)/Z_t$
  - Both ways: $D_{t+1}(i) = C_i D_t(i)exp(-\alpha_t C_i h_t(\boldsymbol{x}_i)y_i)/Z_t$
  - With C_i being the cost of misclassifying $\boldsymbol{x}_i$

From Class- to Case-Specific Costs

- **Previous approaches use class-dependent error costs**
  - Recall cost matrix and cost-benefit matrix
  - Common simplification in the literature
- **Many real-world applications exhibit case-specific costs**

CREDIT SCORING		Actual Class	
		BAD	GOOD
Predicted Class	bad	0	$r_i - C^a$
	good	$Cl_i \cdot LGD_i$	0

CHURN PREDICTION		Actual Class	
		CHURN	LOYAL

OTHERS	Actual Class

Note index ***i***: cost depend on case/customer

From Class to Example Dependent Costs

Decision tree learning with example dependent costs

- **Tree induction principle:** recursively partition data set through (node) splitting
  - Split aims at reducing impurity
  - Merit of a split given by information gain $IG(N) = I(N) - p_{N_1} I(N_1) - p_{N_2} I(N_2)$
- **Cost-sensitive impurity measure (Bahnsen et al. 2015)**
  - Consider a naïve classification in which all cases in a node are classified as positive or negative
  - Let f_1 and f_0 denote the corresponding naïve classifiers
  - Given training data with example dependent cost info, we can calculate the costs of f_1 and f_0
  - A node in a tree is just a set of examples, so calculate node impurity as

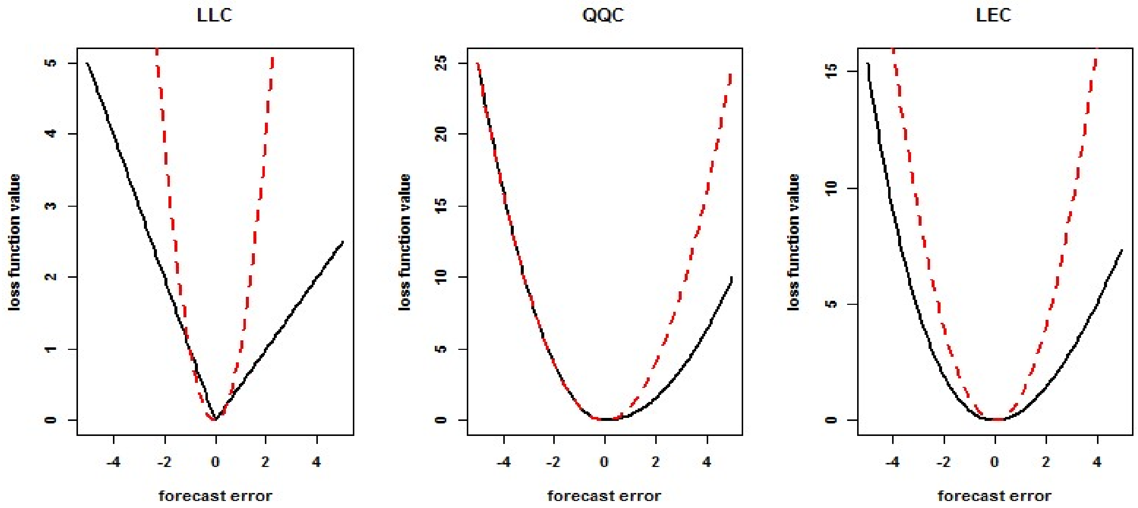
$$I_{cost}(N) = min\{Cost(f_0(N)), Cost(f_1(N))\}$$

- **Build tree to maximize IG using I_{cost}**
- **Can use same logic to prune the tree**

From class to Example Dependent Costs

Asymmetric Cost-of-Error-Functions in Regression

- **Regression error measures imply symmetric costs**
  - Consider squared-error loss
  - Same penalty for positive / negative residuals
- **Asymmetric cost of error functions (Granger, 1969)**
- **Real-world applications**
  - Crime forecasting (Berk, 2011)
  - Demand-planning (Crone, 2010)
  - Car leasing (Dress et al., 2018)
  - Many others



Summary

Summary

Learning goals

- Implications of class skew and error costs
- Strategies for model development and evaluation

Findings

- Need robust, suitable accuracy indicators
- Three options in the predictive modeling process
  - Data (i.e., pre-processing): resampling & weighting
  - Algorithmic adjustments
  - Post-processing of predictions (e.g., cut-off)
- Bayes decision theory
- Selected modeling approaches

What next

- Marketing decision support
- Model evaluation and development

Literature

- Bahnsen, A. C., Aouada, D., & Ottersten, B. (2015). Example-dependent cost-sensitive decision trees. *Expert Systems with Applications, 42*, 6609-6619.
- Bahnsen, A. C., Aouada, D., & Ottersten, B. (2014). Example-Dependent Cost-Sensitive Logistic Regression for Credit Scoring. In *Proceedings of the 13th IEEE 13th International Conference on Machine Learning and Applications (ICMLA)* (pp. 263-269).
- Batista, G., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter, 6*, 20-29.
- Berk, R. (2011). Asymmetric loss functions for forecasting in criminal justice settings. Journal of Quantitative Criminology, 27, 107-123.
- Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research, 16*, 321-357.
- Crone, S. F. (2010). *Neuronale Netze zur Prognose und Disposition im Handel*. Wiesbaden: Gabler.
- Dress, K., Lessmann, S., & von Mettenheim, H.-J. (2018). Residual value forecasting using asymmetric cost functions. *International Journal of Forecasting,* 34(4), 551-565.
- Domingos, P. (1999). MetaCost: A General Method for Making Classifiers Cost-Sensitive. In U. M. Fayyad, S. Chaudhuri & D. Madigan (Eds.), *Proc. of the 5th Intern. Conf. on Knowledge Discovery and Data Mining* (pp. 155-164). San Diego, CA, USA: ACM Press.
- Drummond, C., & Holte, R. C. (2006). Cost curves: An improved method for visualizing classifier performance *Machine Learning, 65*, 95-130.
- Fawcett, T. (2006). ROC graphs with instance-varying costs. *Pattern Recognition Letters, 27*, 882-891
- Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters, 27*, 861-874
- Galar, M., Fernández, A., Barrenechea, E., Bustince, H., & Herrera, F. (2016). Ordering-based pruning for improving the performance of ensembles of classifiers in the framework of imbalanced datasets. *Information Sciences, 354*, 178-196.
- Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., & Herrera, F. (2012). A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42*, 463-484.
- Granger, C. W. J. (1969). Prediction with a generalized cost of error function. *Operational Research Quarterly, 20*, 199-207.

Literature (cont.)

- Haibo, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering, 21*, 1263-1284.
- Hand, D. J., & Anagnostopoulos, C. (2014). A better Beta for the H measure of classification performance. *Pattern Recognition Letters, 40*, 41-46.
- Hand, D. J., & Anagnostopoulos, C. (2013). When is the area under the receiver operating characteristic curve an appropriate measure of classifier performance? *Pattern Recognition Letters, 34*, 492-495.
- Hand, D. J. (2009). Measuring classifier performance: A coherent alternative to the area under the ROC curve. *Machine Learning, 77*, 103-123.
- Hernández-Orallo, J., Flach, P. A., & Ramirez, C. F. (2011). Brier Curves: A New Cost-Based Visualisation of Classifier Performance. In L. Getoor & T. Scheffer (Eds.), *Proc. of the 28th Intern. Conf. on Machine Learning* (pp. 585-592). Bellevue, WA, USA: Omnipress.
- King, G. & Zeng, L (2001). Logistic Regression in Rare Events Data. *Political Analysis*, 9, 137–163. Copy at http://j.mp/2oSEnmf
- Lessmann, S. (2013). Modelling Mismatch in Predictive Analytics. In *Proceedings of the 21st European Conference on Information Systems (ECIS'2013)*. Utrecht, Netherlands: AIS.
- Lomax, S., & Vadera, S. (2013). A survey of cost-sensitive decision tree induction algorithms. *ACM Comput. Surv., 45*, 1-35.
- Jo, T., & Japkowicz, N. (2004). Class imbalances versus small disjuncts. *ACM SIGKDD Explorations Newsletter, 6*, 40-49.
- Paleologo, G., Elisseeff, A., & Antonini, G. (2010). Subagging for credit scoring models. *European Journal of Operational Research, 201*, 490-499.
- Saito, T., & Rehmsmeier, M. (2015). The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. *PLoS One, 10*, e011843.
- Sheng, V.S. & Ling, C.X. (2006). Thresholding for Making Classifiers Cost-sensitive. In: Proceedings of the 21st National Conference on Artificial Intelligence, 476-481. July 16–20, 2006, Boston, MA, USA.
- Sun, Y., Kamel, M. S., Wong, A. K. C., & Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition, 40*, 3358-3378.
- Zhou, Z.-H., & Liu, X.-Y. (2006). Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering, 18*, 63-77.

Thank you for your attention!

Stefan Lessmann

Chair of Information Systems
School of Business and Economics
Humboldt-University of Berlin, Germany

Tel. +49.30.2093.5742
Fax. +49.30.2093.5741

stefan.lessmann@hu-berlin.de
http://bit.ly/hu-wi

www.hu-berlin.de