KU LEUVEN

Outcome-Oriented Predictive Process Monitoring on Positive and Unlabelled Event Logs

Jari Peeperkorn[1], Carlos Ortega Vázquez[1], Alexander Stevens[1],

Johannes De Smedt[1] , Seppe vanden Broucke[2,1] , and Jochen De Weerdt[1]

ML4PM 2022

[1] Research Center for Information Systems Engineering (LIRIS), KU Leuven
[2] Department of Business Informatics and Operations Management, Ghent University

Introduction

- OOPPM → predicting future state ongoing processes, using historical cases (and often the prefix)

- Labels of the historical data might be incomplete
  - PU setting: we only have positive and unlabelled data
    → Unlabelled contains negative examples but also might contain positive examples
  - Use methods based on the Expected Risk Minimization (ERM)
    → Adapt loss function

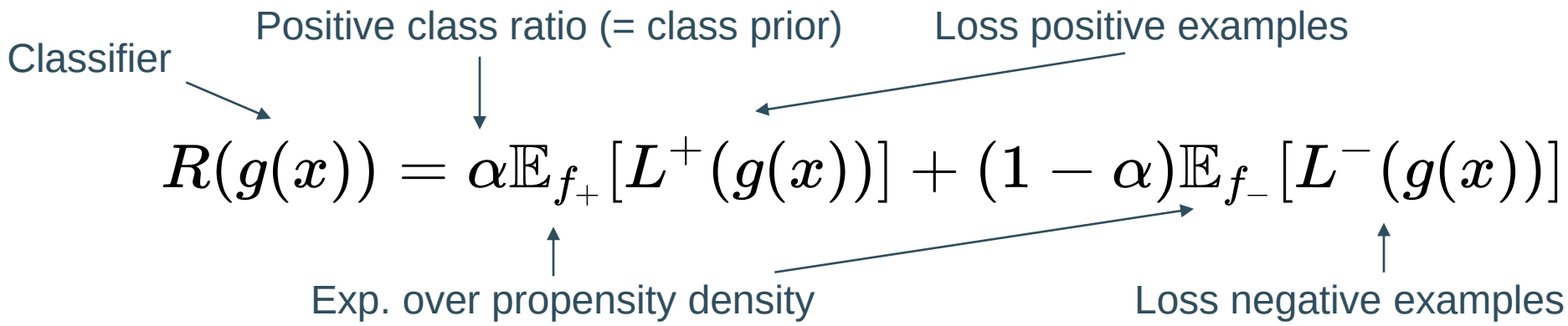
- Examples
  - When positive label = (hard to detect) outliers or fraud
  - Labels based on customer (dis)satisfaction
  - Medicine

Models

- LSTM Neural Network
  - Recurrent Neural Network
  - Long-term dependencies

- XGBoost
  - Gradient Boosting
  - Ensemble of Decision Trees (*weak learners*)
  - Each tree trained on the residual of the model

Empirical Risk Minimisation (ERM) framework (standard)



Binary Cross-Entropy

Predicted probability $p \in [0, 1]$

Real label $y \in \{0, 1\}$

$$L^+(g(x_i)) = -log(p_i)$$

$$L^-(g(x_i)) = -log(1 - p_i)$$

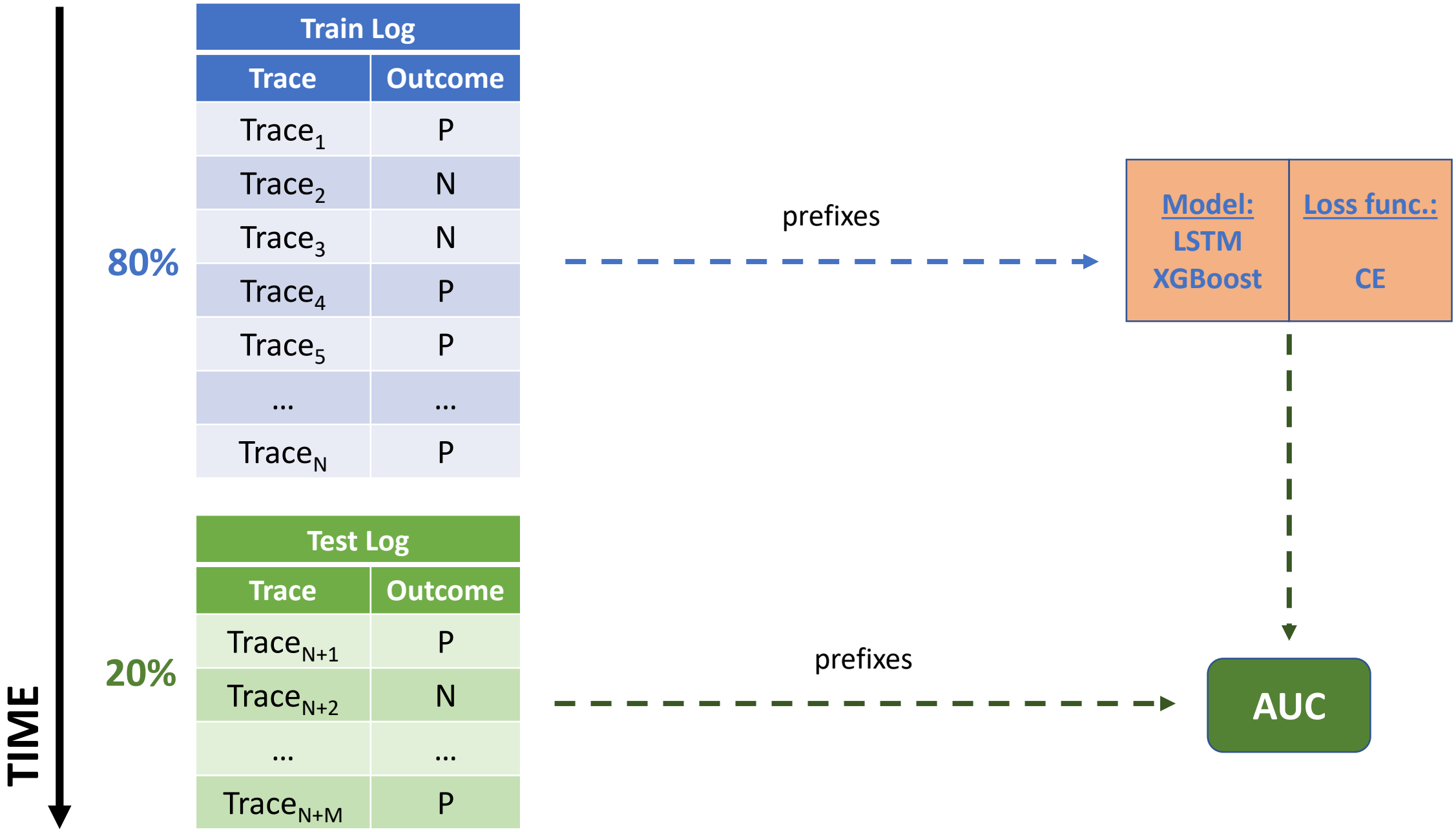
$$BCE = -\sum_{i=1}^{N} (y_i \log(p_i) + (1 - y_i) \log(1 - p_i))$$

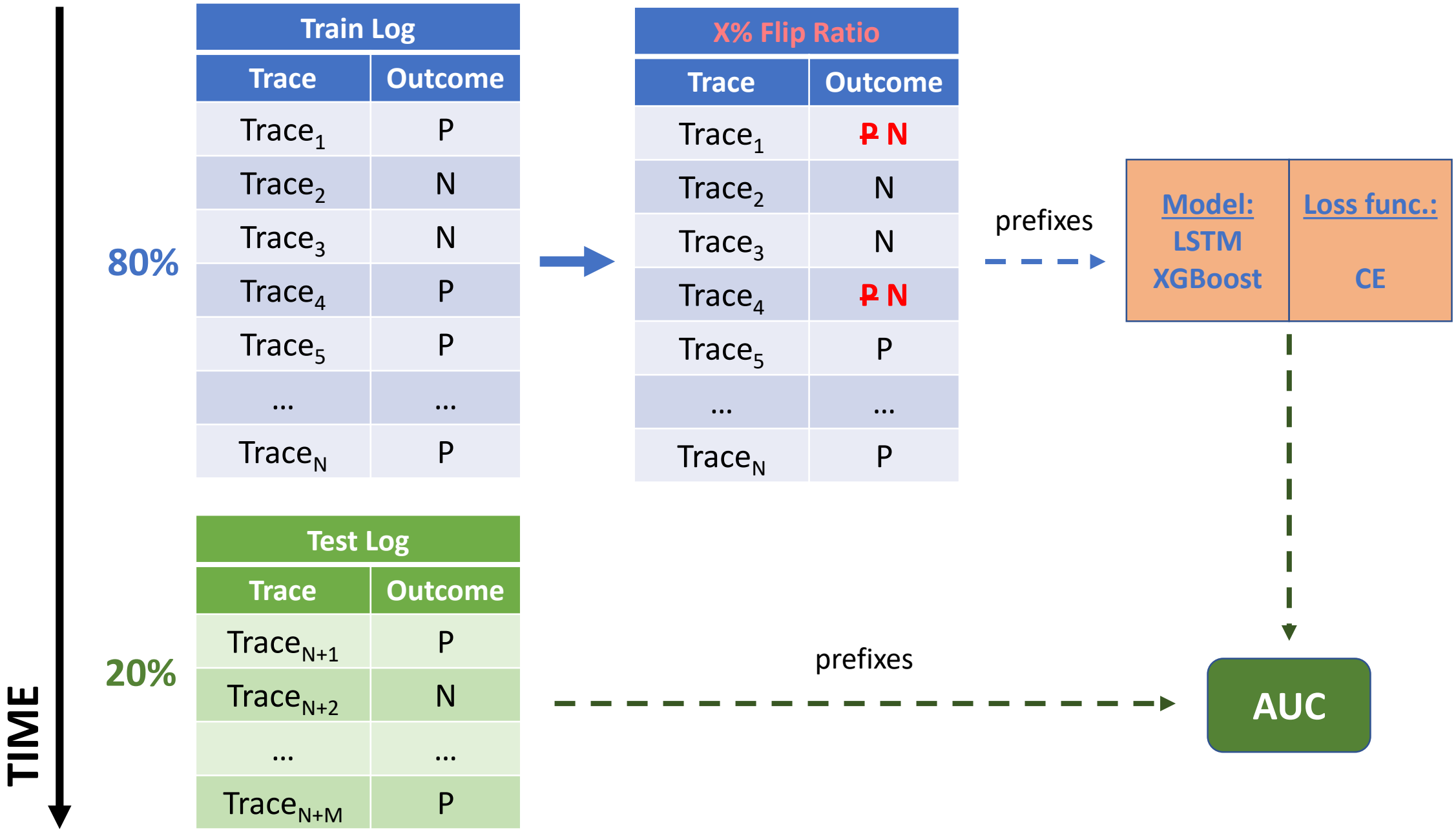
→ For LSTM: the loss function that is used for backpropagation (through time)
→ For XGBoost: the loss to be optimized when adding new trees (not within a tree)

Experiment 1

“Incorrectly labelling deviant (positive) behaviour as normal (negative), can have an important impact on the (future) performance of a predictive model.”

- Test two models (LSTM NN + XGBoost) with cross-entropy loss
- Start with **correctly labelled** event log
- Train different models on different Train Logs
  - Original Log
  - Logs where we flip different % of positive cases to negative
- Test on independent Test Log (correctly labelled)


TIME
80%
Train Log
Trace
Outcome
Trace1
P
Trace2
N
Trace3
N
Trace4
P
Trace5
P
...
...
TraceN
P
prefixes
Model:
LSTM
XGBoost
Loss func.:
CE
20%
Test Log
Trace
Outcome
TraceN+1
P
TraceN+2
N
...
...
TraceN+M
P
prefixes
AUC


TIME
80%
Train Log
Trace
Outcome
Trace1 P
Trace2 N
Trace3 N
Trace4 P
Trace5 P
... ...
TraceN P
X% Flip Ratio
Trace
Outcome
Trace1 P N
Trace2 N
Trace3 N
Trace4 P N
Trace5 P
... ...
TraceN P
prefixes
Model:
LSTM
XGBoost
Loss func.:
CE
20%
Test Log
Trace
Outcome
TraceN+1 P
TraceN+2 N
... ...
TraceN+M P
prefixes
AUC

Data sets

Dataset	Min Len	Med Len	Max Len	Trunc. Len	#Train	#Test	$R(+)$ Train	$R(+)$ Test
2011_1	1	25	1814	36	912	228	0.38	0.48
2011_2	1	54.5	1814	40	912	228	0.81	0.66
2011_3	1	21	1368	31	896	225	0.20	0.36
2011_4	1	44	1432	40	912	228	0.25	0.39
2015_1	2	42	101	40	555	140	0.22	0.26
2015_2	1	55	132	40	602	151	0.20	0.17
2015_3	3	42	124	40	1062	266	0.17	0.25
2015_4	1	42	82	40	460	116	0.17	0.13
2015_5	5	50	134	40	840	211	0.32	0.26

→ Setup + labels based on:

Irene Teinemaa, Marlon Dumas, Marcello La Rosa, and Fabrizio Maria Maggi. 2019. Outcome-Oriented Predictive Process Monitoring: Review and Benchmark. ACM Trans. Knowl. Discov. Data 13, 2, Article 17 (April 2019), 57 pages. https://doi.org/10.1145/3301300

Results

Dataset	Method	Flip Ratio 0%	25%	50%	75%
bpic2011_1	LSTM	0.891	0.514	0.667	0.509
bpic2011_1	XGB	0.944	0.867	0.805	0.751
bpic2011_2	LSTM	0.882	0.520	0.783	0.494
bpic2011_2	XGB	0.972	0.962	0.905	0.785
bpic2011_3	LSTM	0.680	0.863	0.755	0.831
bpic2011_3	XGB	0.989	0.982	0.803	0.905
bpic2011_4	LSTM	0.873	0.680	0.680	0.736
bpic2011_4	XGB	0.865	0.855	0.813	0.720
bpic2015_1	LSTM	0.885	0.706	0.712	0.579
bpic2015_1	XGB	0.917	0.919	0.904	0.761
bpic2015_2	LSTM	0.937	0.854	0.803	0.807
bpic2015_2	XGB	0.947	0.952	0.914	0.909
bpic2015_3	LSTM	0.878	0.673	0.694	0.624
bpic2015_3	XGB	0.962	0.941	0.942	0.930
bpic2015_4	LSTM	0.858	0.784	0.715	0.465
bpic2015_4	XGB	0.917	0.898	0.837	0.847
bpic2015_5	LSTM	0.916	0.757	0.759	0.667
bpic2015_5	XGB	0.944	0.939	0.907	0.813

- Generally, performance deteriorates
- Not everywhere
- XGBoost > LSTM

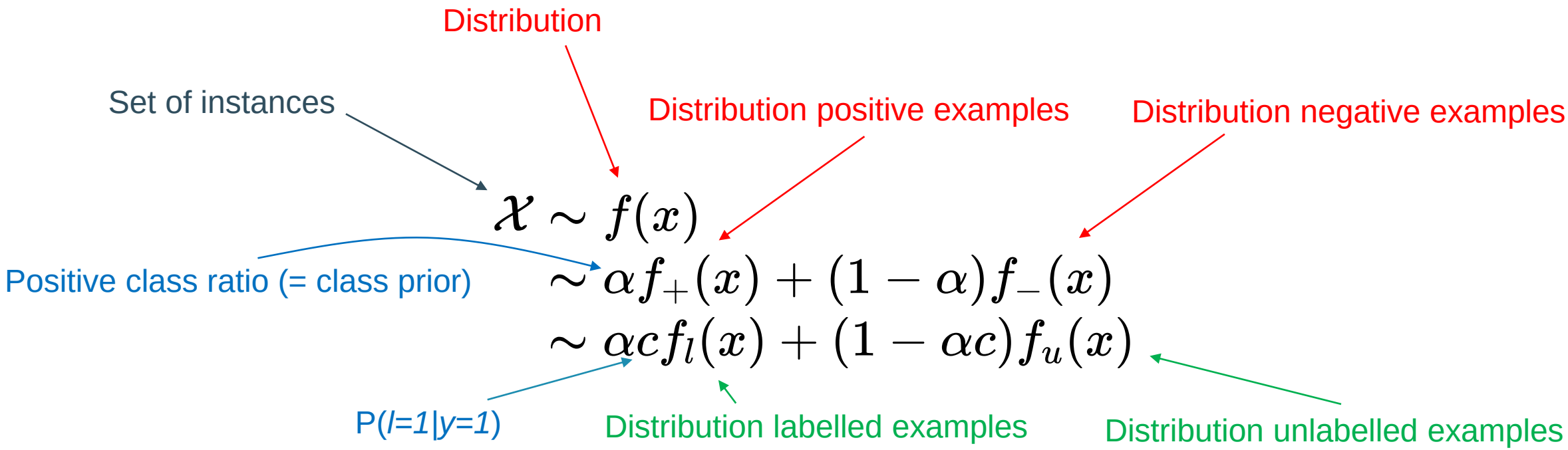
Unbiased PU Learning

- Ground truth label *y* not available

  → label status *l* ∈ {0, 1}

- Idea: reweight positive examples with the class prior to correct the learning
  - ~~Correcting the input~~
  - Correcting the risk: increase the emphasis on the (few) labelled positives

  → decision boundary covers a larger region around the few positives



Unbiased PU Learning

Unlabelled examples are considered negative with a weight of 1

$$R_{upu}(g(x)) = \alpha c \mathbb{E}_{f_l}\left[\frac{1}{c}L^+(g(x)) + (1 - \frac{1}{c})L^-(g(x))\right] + (1 - \alpha c)\mathbb{E}_{f_u}[L^-(g(x))]$$

Labelled examples added as both positive with weight $\frac{1}{c}$ and negative with weight $1 - \frac{1}{c}$

Loss functions

$$L^+(g(x_i)) = -log(p_i) \qquad L^-(g(x_i)) = -log(1-p_i)$$

You might assume we add a term here, for the unlabelled, but actually positives which rewards the model for detecting these as positives

We add a term for the labelled positives: for both instances predicted to be positive and negative by the model → push decision boundary

$$uPU_{BCE} = -\sum_{i=1}^{N}\left(l_i\left[\frac{1}{c}\log(p_i) + (1-\frac{1}{c})\log(1-p_i)\right] + (1-l_i)\left[\log(1-p_i)\right]\right)$$

non-negative PU risk estimator

- Problem with uPU risk estimator → can provide negative empirical risks
  - Problem for flexible (powerful) techniques that can easily overfit (XGBoost, deep mearning)

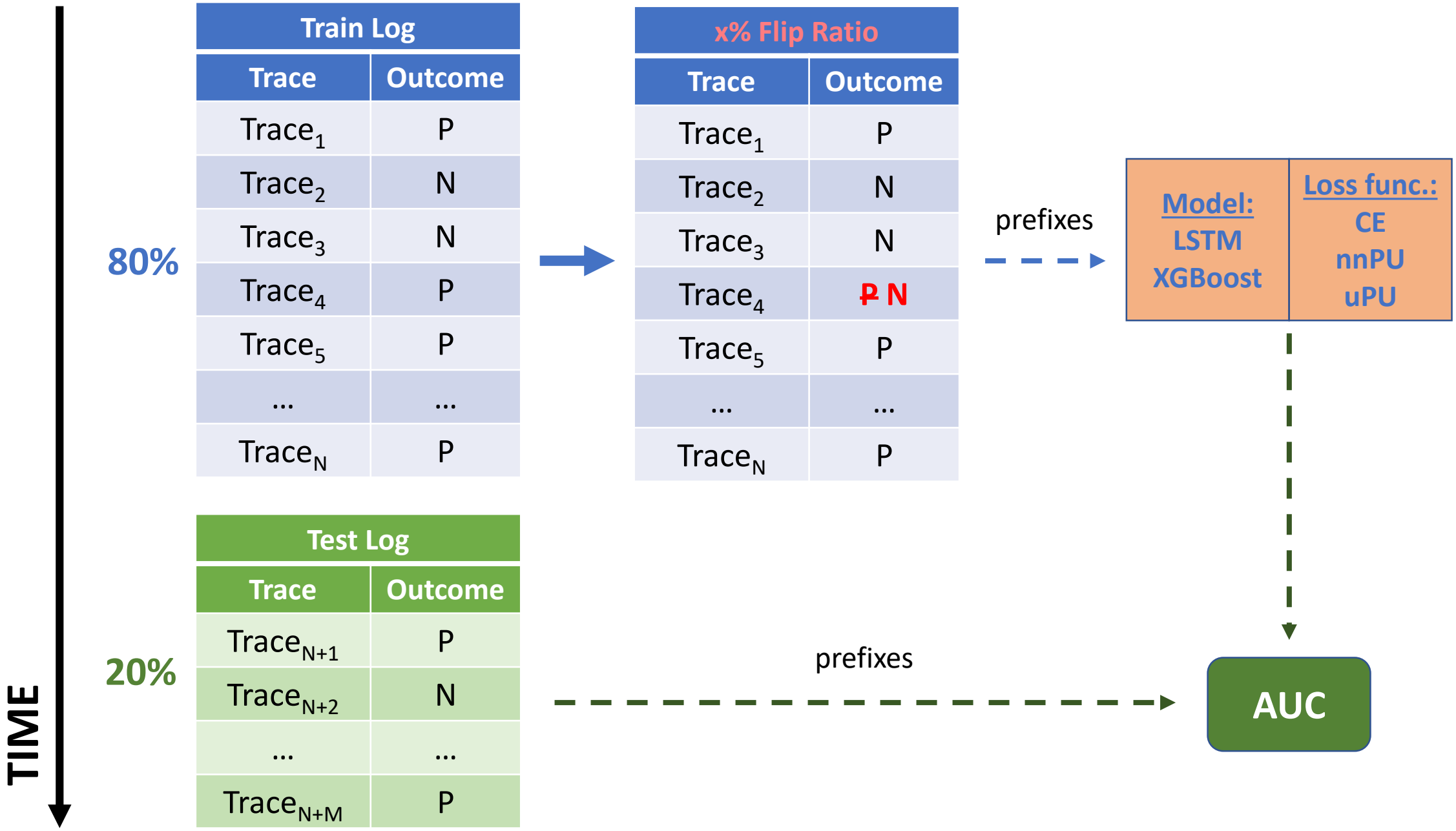
$$R_{nnpu}(g(x)) = \alpha c \mathbb{E}_{f_l}\left[\frac{1}{c} L^+(g(x))\right] + \max\left(0, (1-\alpha c)\mathbb{E}_{f_u}[L^-(g(x))] + \alpha c \mathbb{E}_{f_l}\left[(1-\frac{1}{c})L^-(g(x))\right]\right)$$

$$nnPU_{BCE} = -\sum_{i=1}^{N}\left(l_i\left[\frac{1}{c}\log(p_i)\right] + \max\left(0, (1-l_i)\left[\log(1-p_i)\right] + l_i(1-\frac{1}{c})\log(1-p_i)\right)\right)$$

Experiment 2

“Using loss functions from PU-learning, the problem above can be (partially) mitigated.”

- Test two models (LSTM NN + XGBoost)
  - With cross-entropy, uPU and nnPU loss functions
- Start with **correctly labelled** event log
- Train different models on different Train Logs
  - Logs where we flip different % of positive cases to negative
- Test on independent Test Log (correctly labelled)
- Compare models with different loss functions


TIME
80%
Train Log
Trace
Outcome
Trace1 P
Trace2 N
Trace3 N
Trace4 P
Trace5 P
... ...
TraceN P
x% Flip Ratio
Trace
Outcome
Trace1 P
Trace2 N
Trace3 N
Trace4 P N
Trace5 P
... ...
TraceN P
prefixes
Model:
LSTM
XGBoost
Loss func.:
CE
nnPU
uPU
20%
Test Log
Trace
Outcome
TraceN+1 P
TraceN+2 N
... ...
TraceN+M P
prefixes
AUC

Results

Dataset	Flip	LSTM CE	LSTM nnPU	LSTM uPU	XGB CE	XGB nnPU	XGB uPU
bpic2011_1	25%	0.514	**0.818**	**0.818**	0.867	**0.910**	0.897
bpic2011_1	50%	0.667	**0.736**	0.565	0.805	**0.889**	0.800
bpic2011_1	75%	0.509	0.505	**0.727**	0.751	**0.801**	0.684
bpic2011_2	25%	0.520	**0.752**	0.723	0.962	**0.963**	0.921
bpic2011_2	50%	0.783	**0.820**	0.662	0.905	0.922	**0.942**
bpic2011_2	75%	0.494	0.530	**0.612**	0.785	**0.827**	0.545
bpic2011_3	25%	**0.863**	0.838	0.750	0.982	0.975	**0.987**
bpic2011_3	50%	0.755	**0.773**	0.687	0.803	**0.925**	0.831
bpic2011_3	75%	**0.831**	0.779	0.707	0.905	**0.931**	0.911
bpic2011_4	25%	0.680	0.773	**0.775**	0.855	**0.868**	0.861
bpic2011_4	50%	0.680	**0.784**	0.734	**0.813**	0.812	0.718
bpic2011_4	75%	0.736	0.694	**0.840**	0.720	**0.797**	0.729

Dataset	Flip	LSTM CE	LSTM nnPU	LSTM uPU	XGB CE	XGB nnPU	XGB uPU
bpic2015_1	25%	0.706	0.804	**0.817**	**0.919**	0.916	0.917
bpic2015_1	50%	0.712	**0.803**	0.663	0.904	**0.918**	0.865
bpic2015_1	75%	0.579	0.609	**0.638**	0.761	0.631	**0.774**
bpic2015_2	25%	**0.854**	0.486	0.839	**0.952**	0.949	0.945
bpic2015_2	50%	0.803	0.594	**0.855**	**0.914**	0.902	0.867
bpic2015_2	75%	**0.807**	0.742	0.653	**0.909**	0.858	0.821
bpic2015_3	25%	0.673	**0.777**	0.592	0.941	**0.955**	0.947
bpic2015_3	50%	0.694	**0.715**	0.628	**0.942**	**0.942**	0.934
bpic2015_3	75%	0.624	**0.835**	0.583	**0.930**	0.904	**0.930**
bpic2015_4	25%	0.784	**0.821**	0.801	0.898	0.898	**0.923**
bpic2015_4	50%	**0.715**	0.615	0.678	0.837	**0.886**	0.844
bpic2015_4	75%	0.465	**0.664**	0.598	**0.847**	0.835	0.839
bpic2015_5	25%	**0.757**	0.710	0.684	**0.939**	0.937	0.924
bpic2015_5	50%	**0.759**	0.755	0.693	0.907	**0.921**	0.912
bpic2015_5	75%	0.667	**0.680**	0.576	0.813	**0.837**	0.777

Limitations

- Need a (good estimate for) class prior
  - We used the flip ratio

- We more or less assume a smooth transition between positives and negatives: other unlabelled positives are in the neighbourhood of the positives.

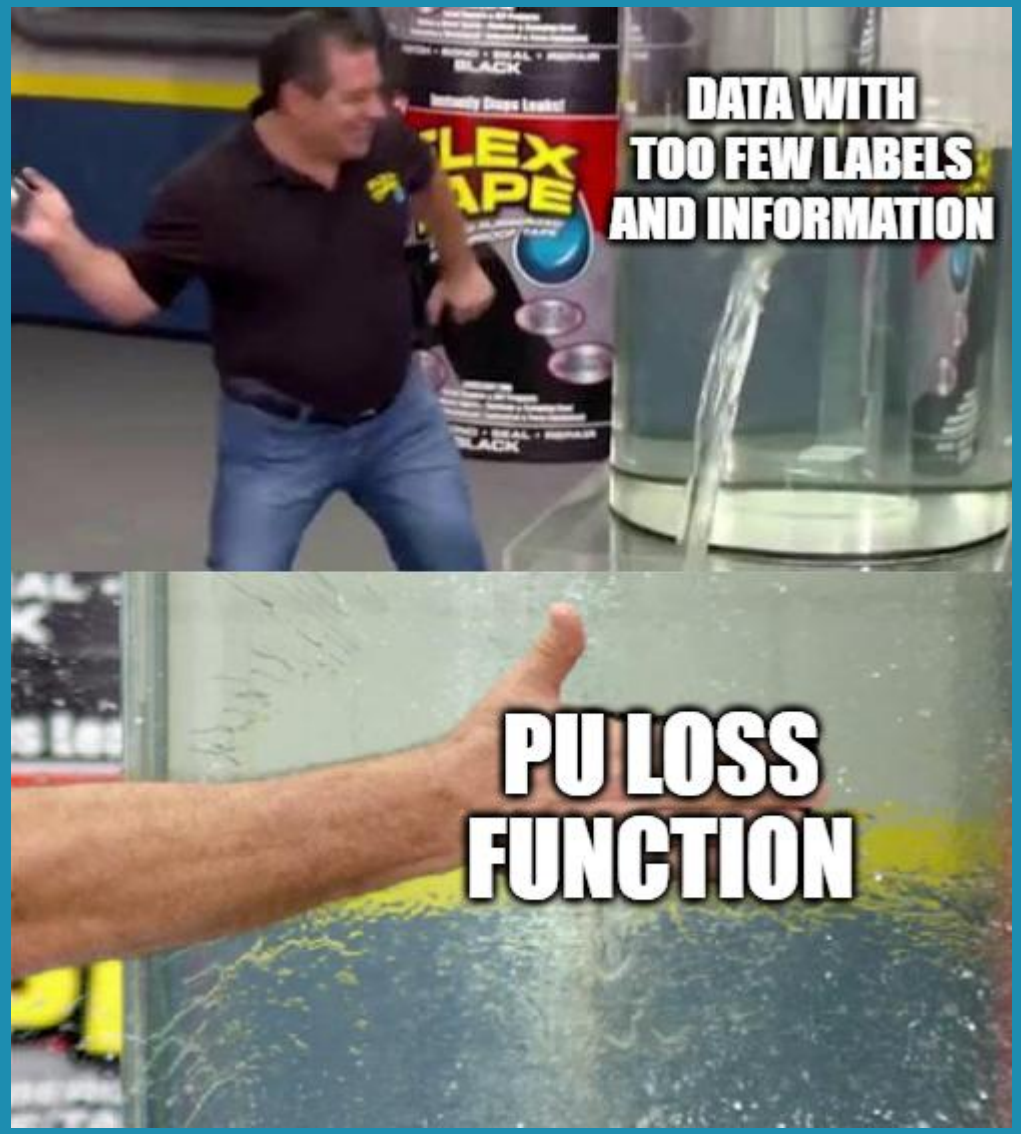
Conclusions

- Introduced OOPPM to a PU setting
- We generally see a drop in models' performance when more *positive* examples are *unlabelled* (*negative*)
- In most cases using a "PU loss function" helps
  - Not for all event logs

Future Work

- Extending experiment
  - Different event logs
  - Multiple iterations (flips)
  - More suitable data
- Experiments on sensitivity class prior
- Investigating options besides altering the loss function
  - Unsupervised learning
  - Process behavior

Questions?



Appendix

Why nnPU?

- When p approaches 1 → when you overfit

- E.g. $c = 0.1$, $p = 0.99$