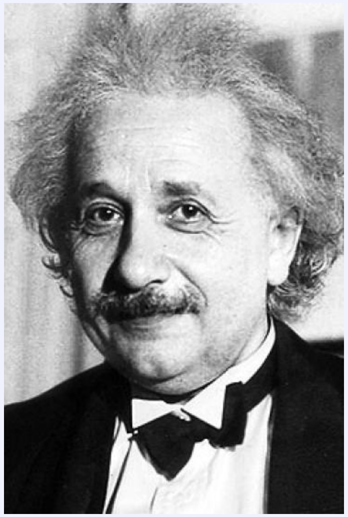
Efficiently Discovering Unexpected Pattern Co-occurrences

Roel Bertens
Jilles Vreeken
Arno Siebes

UNIVERSITÄT DES SAARLANDES
M2CI CLUSTER OF EXCELLENCE
max planck institut informatik

Beauty and Brains



Our world is filled with **beautiful** and **brainy** people,
but, how often does a **beauty pageant**
win a **Nobel prize**?

Question of the day

How can we
efficiently discover **unexpected co-occurrences of patterns**
in transaction data?

Anomalous Transactions

Definition 1. A transaction is anomalous when it deviates from our expectation considering the whole dataset

Classes of Anomalies

There are different
ways to express expectation.
Hence, there are different things
that can be regarded as anomalous.

We identify three classes of anomalies

Unexpected Transaction Lengths

A **class 0 anomaly** is a transaction with **significantly deviating** transaction length, with **unexpectedly high**

$$score_0(t) = -\log(P(|t|))$$

For example, transactions where people buy all items in the shop, instead of just one can of Coke[1], as most people.

[1] or Pepsi

Unexpected Transactions

A **class 1 anomaly** is a transaction that contains very little regularity

$$score_1(t) = -\log(P(t))$$

For example, transactions that cannot, or only badly be compressed by the optimal compressor for D

(see e.g. Smets & Vreeken 2011, Akoglu et al. 2013)

Unexpected Co-occurrences

A **class 2 anomaly** is a transaction
that contains two patterns that
occur much less often together
than expected from their supports

$$score_2(t) = \max_{\{X,Y \in S \mid X,Y \subseteq t\}} -\log\big(P(XY)\big) + \log\big(P(X)P(Y)\big)$$

For example, a mammal that lays eggs.
As, while there are many mammals,
and many egg-laying creatures,
the combination is very rare

Anomalous Anomalies

Perhaps surprisingly, but **class 2 anomalies**
are not detected by **class 1 detectors**

After all, they contain
many frequent itemsets,
fulfill key association rules,
and are easily compressed.

(eg. He et al. 2005, Narita et al 2008, Smets & Vreeken 2011, Akoglu et al 2013)

Describing Anomalies

Class 2 anomalies are
interpretable and **explainable:**

These two important patterns
almost never show up together,
yet here they are…

Who the heck buys
both Pepsi **and** Coca Cola?

Background Knowledge

Something can only be
anomalous with regard to
background knowledge.

For a **class 2 anomaly**,
such background knowledge
is a set of patterns and their supports.

Hm, which patterns?

Why not, rules?

Given the connection to $lift$, why don't we just mine association rules with low lift?

Well… to maximize $score_2$ the support of patterns X and Y should be as high as possible, while that of XY should be as low as possible.

That is, we will have to mine for **all rules** – including those with support 1 – to make sure we don't miss anything.

That's going to be infeasible.

All the patterns!

We take a set of patterns S, and
compute the score of each pair $X, Y \in S$,
identifying those transactions with a high score.

To maximize $score_2$ the support of
patterns X and Y should be as high as possible,
while that of XY should be as low as possible.

So S should be the set of all frequent patterns!
However, at a cost of $O(|D| \times |S|^2)$ this is infeasible,
while increasing $minsup$ leads to missed anomalies…

(Agrawal & Srikant 1994, etc, etc)

Sampling to the rescue?

Instead of mining all frequent patterns,
we could use a representative sample!

However, how many patterns should we sample?

If we choose too few,
we will miss anomalies, while
if our sample is too large it will be redundant
and we face runtime issues.

(Hasan & Zaki, 2009)

Descriptive Patterns

We choose to use
descriptive patterns.

That is, small sets of patterns, that
do not contain redundancy or noise,
and together describe the data well.

KRIMP and SLIM are two algorithms
to discover such sets efficiently.

(Siebes et al 2006, Smets & Vreeken 2012)

How to use our score?

First of all, significance can be tested via the bootstrap

For example, with replacement,

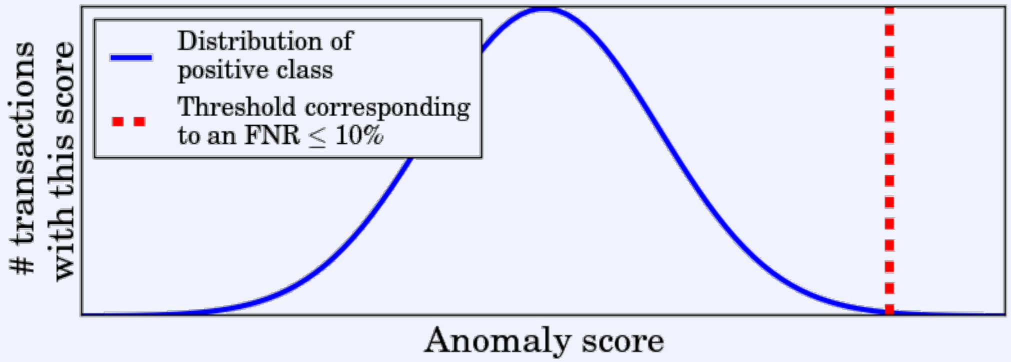
- sample 1000 datasets of size $|D|$ from D
  - store highest $score_2$ for each, and remember highest scoring transaction t^*
- sample 1000 datasets of size $|D|$ from $D \setminus t^*$
  - store highest $score_2$
- compare the two $score_2$ distributions

How to choose the threshold?

Which transactions to investigate?

To identify transactions that stand out
we can use Cantelli's inequality,

$$P(X - \mu_X \geq k\sigma_X) \leq \frac{1}{1+k^2}\,.$$

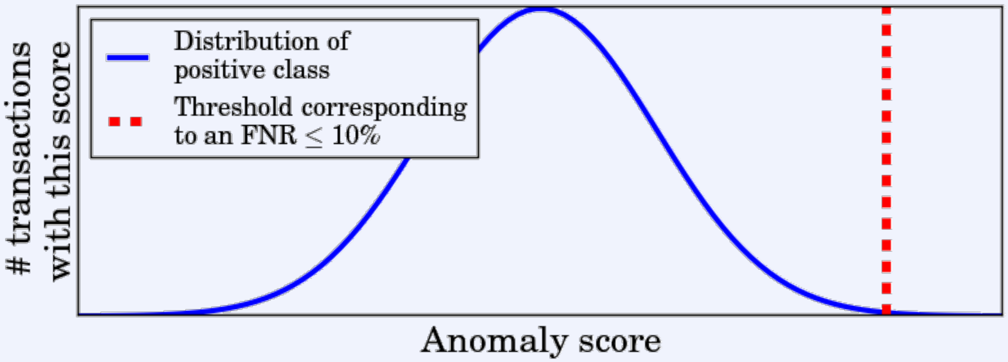


For example, for a confidence of 10%, threshold $\boldsymbol{\theta}$ should be at 3 standard deviations from the mean.

Which transactions to investigate?

To identify transactions that stand out
we can use Cantelli's inequality,

$$P(X - \mu_X \geq k\sigma_X) \leq \frac{1}{1+k^2}\,.$$



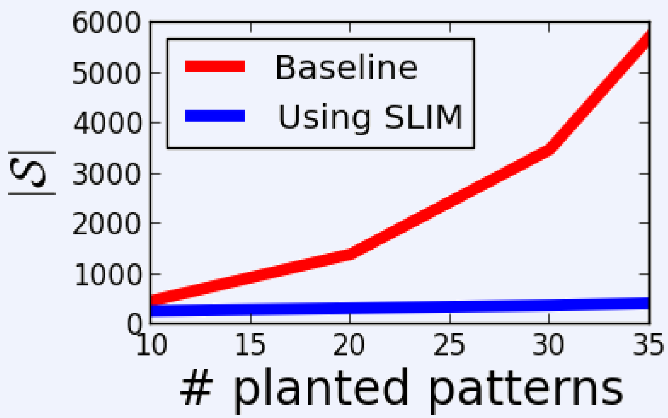
To find μ and σ we consider all $score_2$'s
over 1000 bootstrap samples.

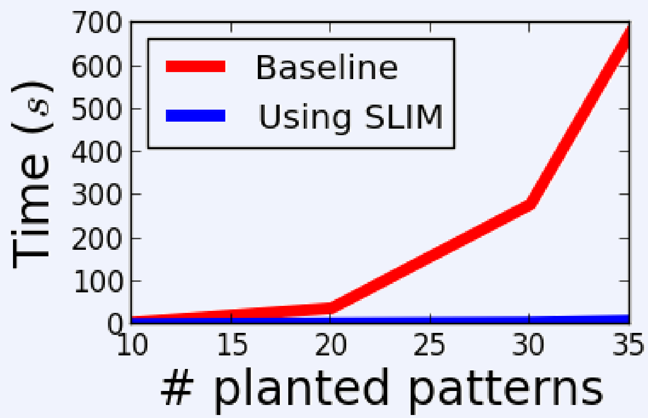
Does it work?

We generate random data, injected random patterns, and 2 anomaly generating patterns that only **co-occur once**.

We compare closed itemsets at minsup 5% to SLIM.

The true anomaly is top-ranked with both pattern sets.

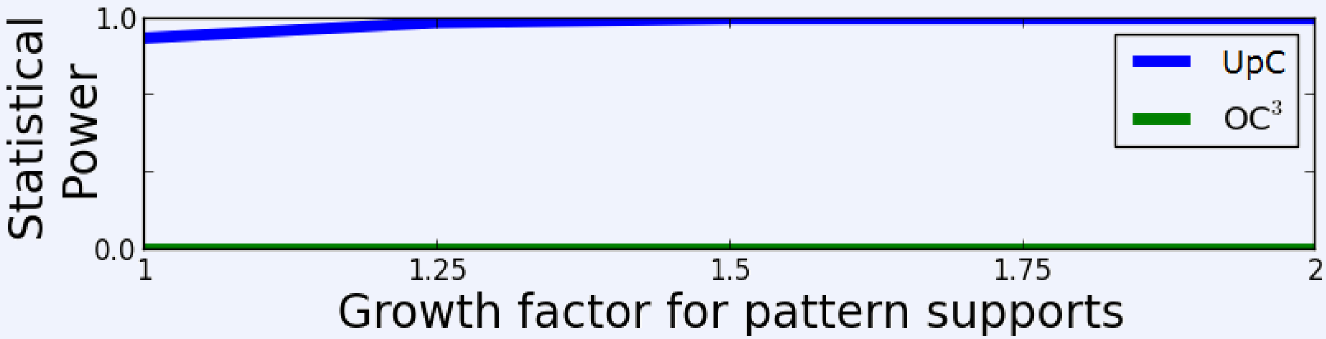




How does it compare?

UpC consistenly ranks the true anomaly at rank 1, whereas OC^3 and CompreX rank it between 2028-8281th.

UpC obtains very high statistical power.

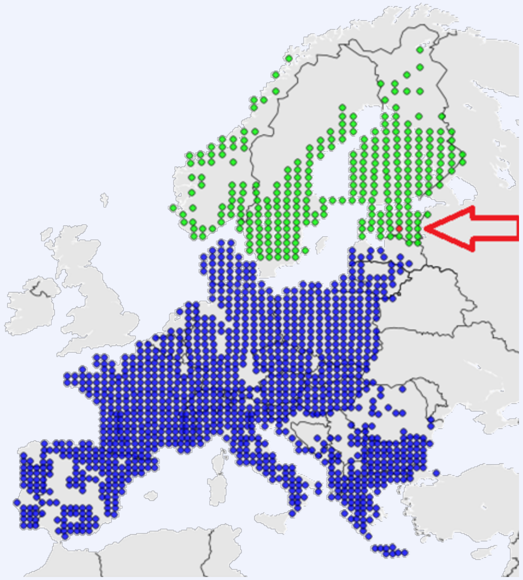


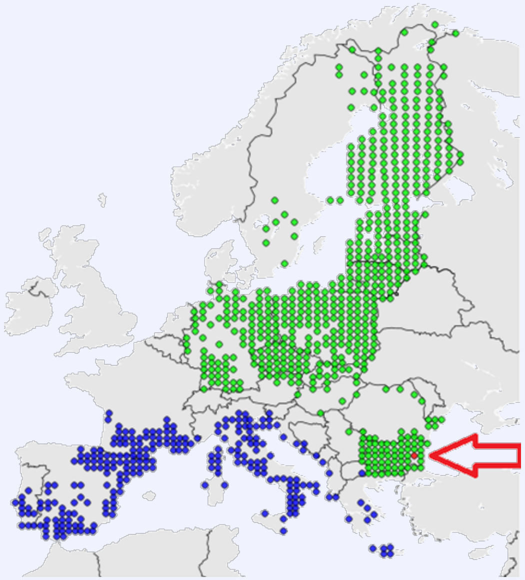
(UpC stands for Unexpected Pattern Co-occurrences)

What does it find?

On real data, we identify

- sex = female **and** relationship = husband (Census)
- platypus, and scorpion (Zoo)
- pattern mining **and** training, (Abstracts)
- frequent pattern mining **and** compare (Abstracts)





Conclusions

We identified a new class of anomalies in transaction data.

In short, UPC

- detects unexpected pattern co-occurrences
- efficient, non-parametric, easy to use
- scales favourably in both data size and dimensionality
- detects true anomalies missed by existing methods

Future work

- extend to continuous, and, or, sequential data

(source code available at: eda.mmci.uni-saarland.de/upc)

Thank you!

We identified a new class of anomalies in transaction data.

In short, UPC

- detects unexpected pattern co-occurrences
- efficient, non-parametric, easy to use
- scales favourably in both data size and dimensionality
- detects true anomalies missed by existing methods

Future work

- extend to continuous, and, or, sequential data

(source code available at: eda.mmci.uni-saarland.de/upc)