Fraud Detection Using Analytics

StatUa day 2020

February 7, 2020

Tim Verdonck



Universiteit Antwerpen

Joint work with Bart Baesens & Sebastiaan Höppner

Table of Contents

- Introduction
- Obtain fraud score using machine learning
- Dealing with imbalanced data
- Evaluate performance taking costs into account
- Design cost-sensitive fraud detection model
- Anomaly detection

Collaboration BNP PF & Allianz

- On December 13 2016, the *BNP Paribas Fortis Chair in Fraud Analytics* was created.
- Purpose of Chair is to investigate the use of predictive analytics in context of payment fraud.
- We have access to unique dataset containing all payments (and their information) of BNP Paribas Fortis in Belgium in last 14 months.
- We also have access to information of account holders of BNP Paribas Fortis.
- On October 18, 2018 the *Allianz Chair in Prescriptive Business Analytics* was created.





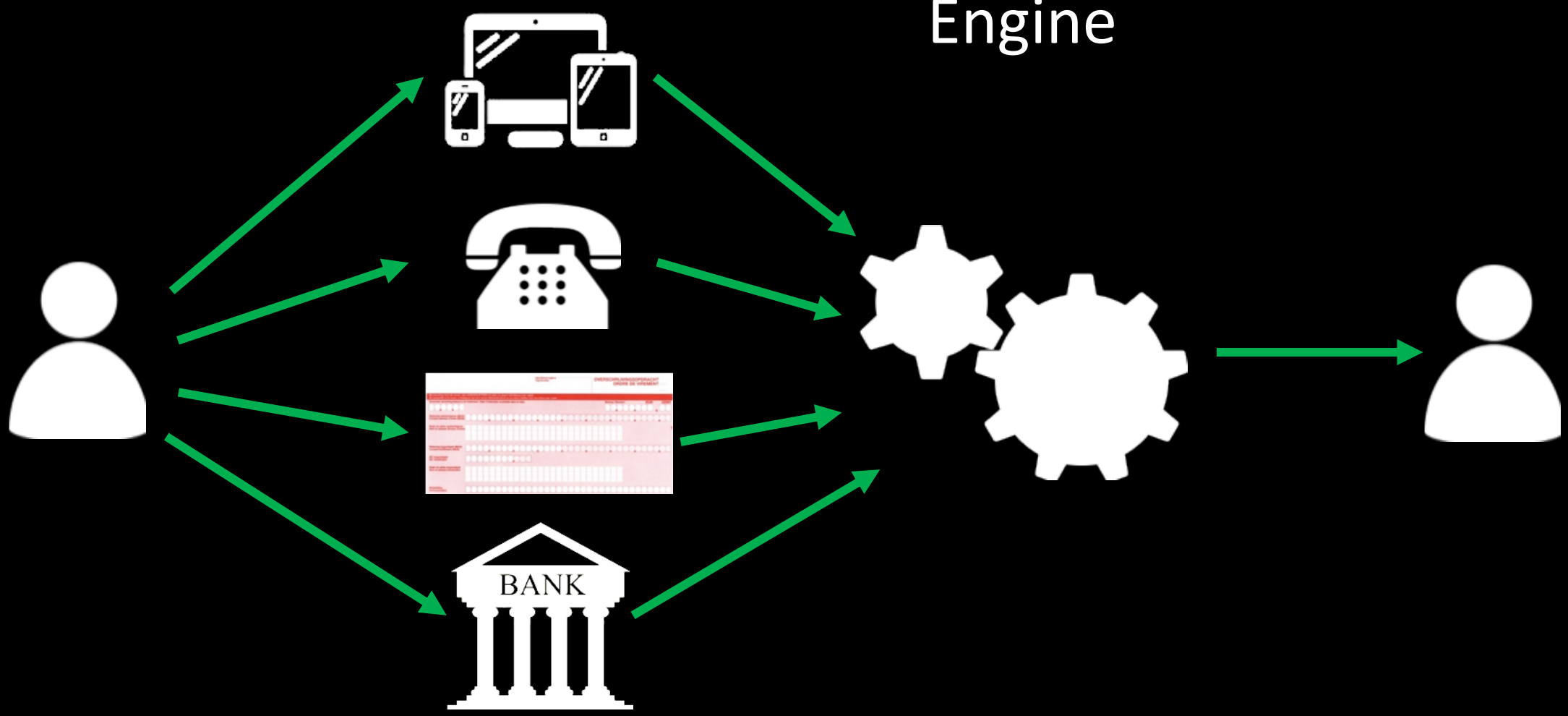
Credit transfers

Initiator

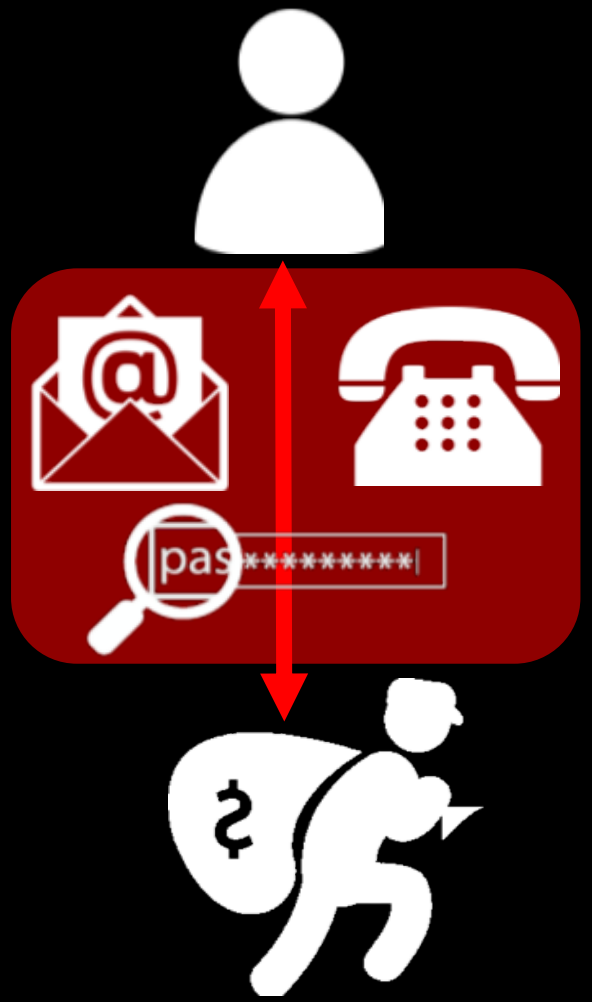
Channels

Payment Engine

Beneficiary

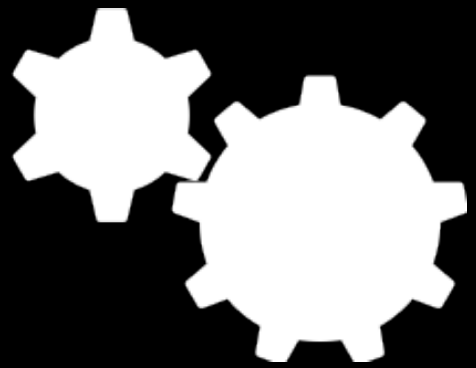


Phishing / vishing



Step 1: a fraudster tricks a customer into sharing their credentials

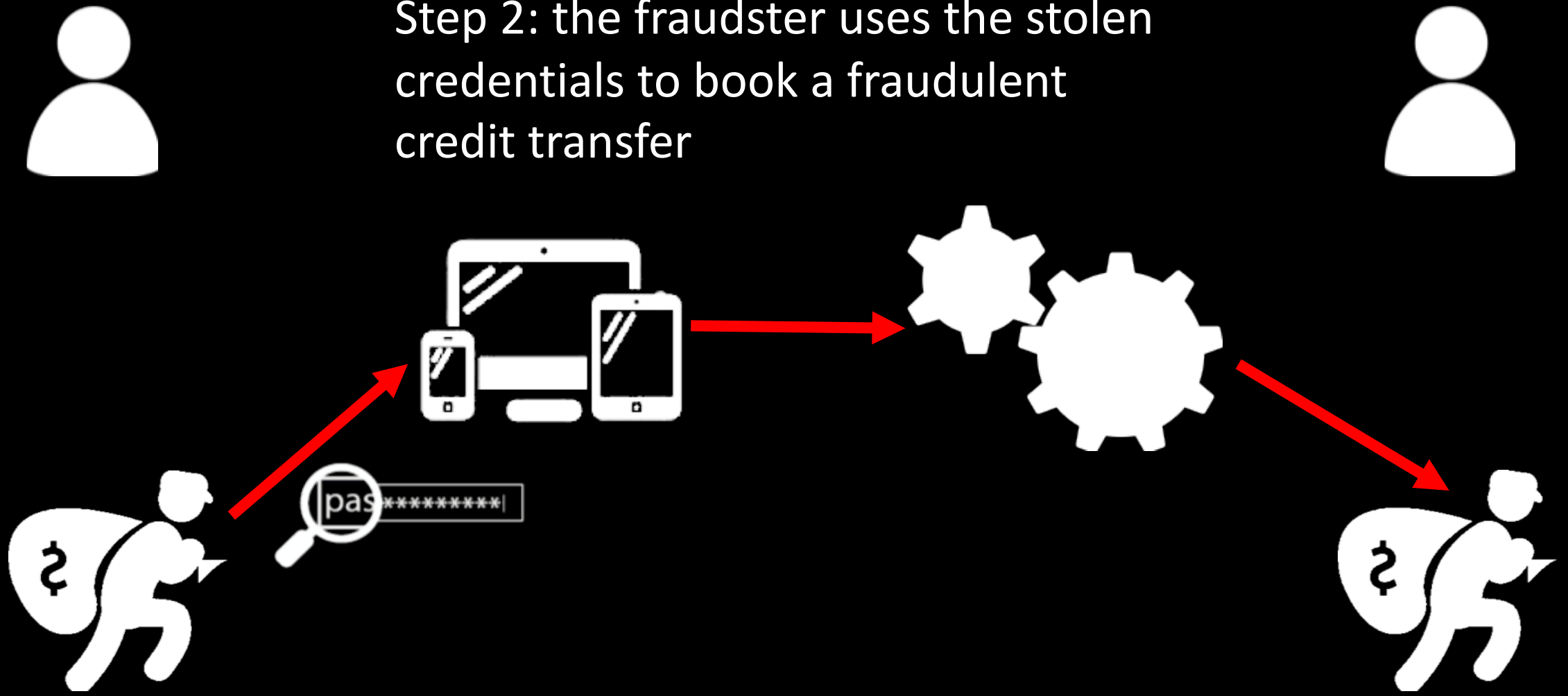






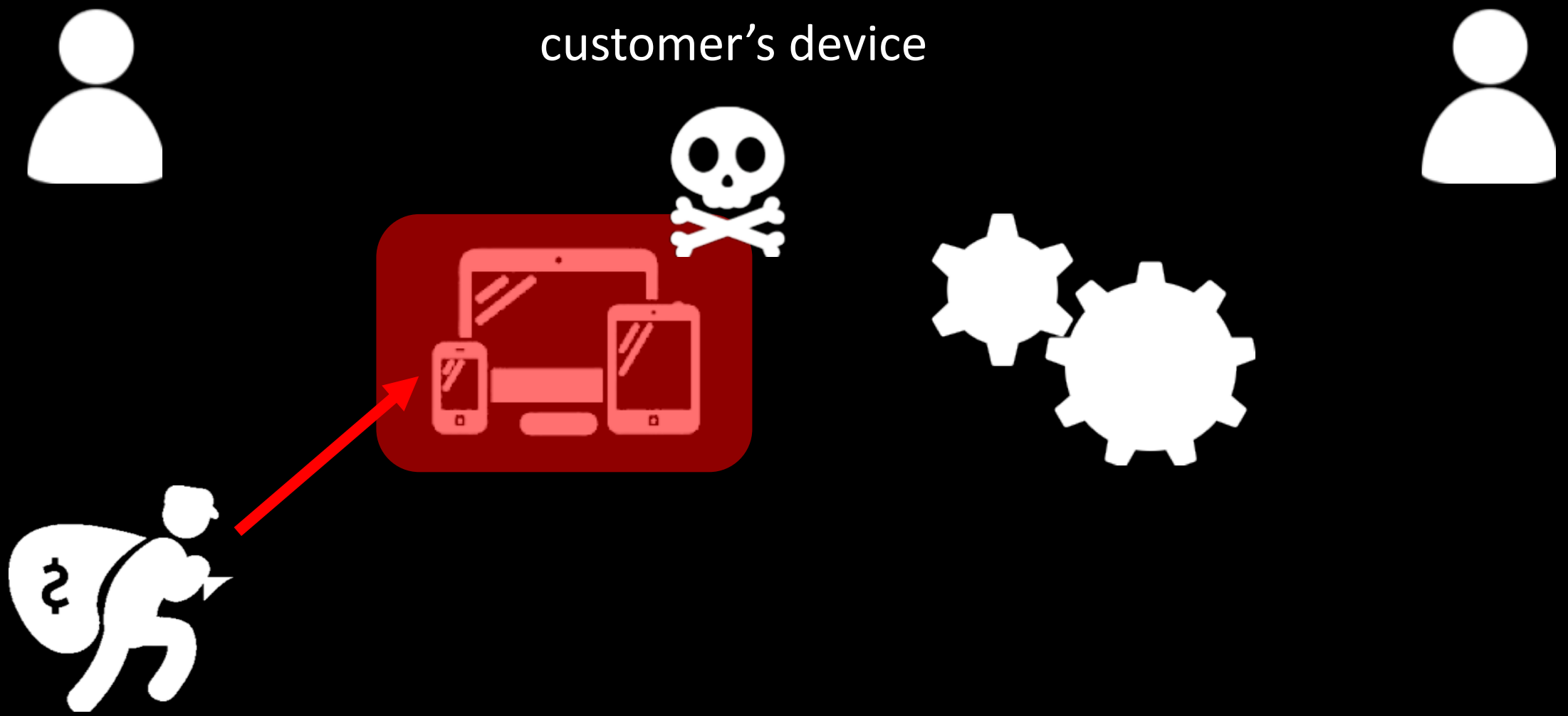
Phishing / vishing

Step 2: the fraudster uses the stolen credentials to book a fraudulent credit transfer



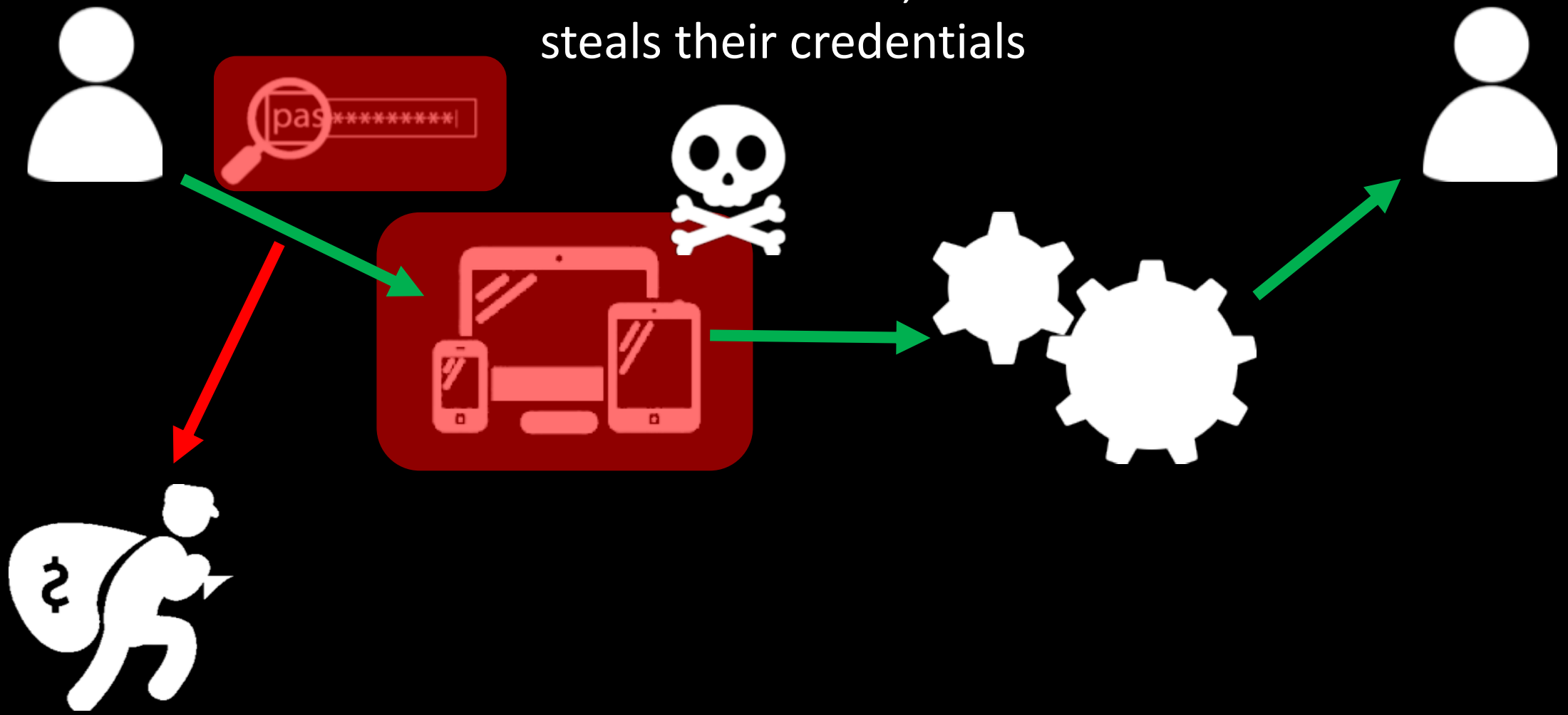
Hacking

Step 1: the fraudster installs malware on the customer's device



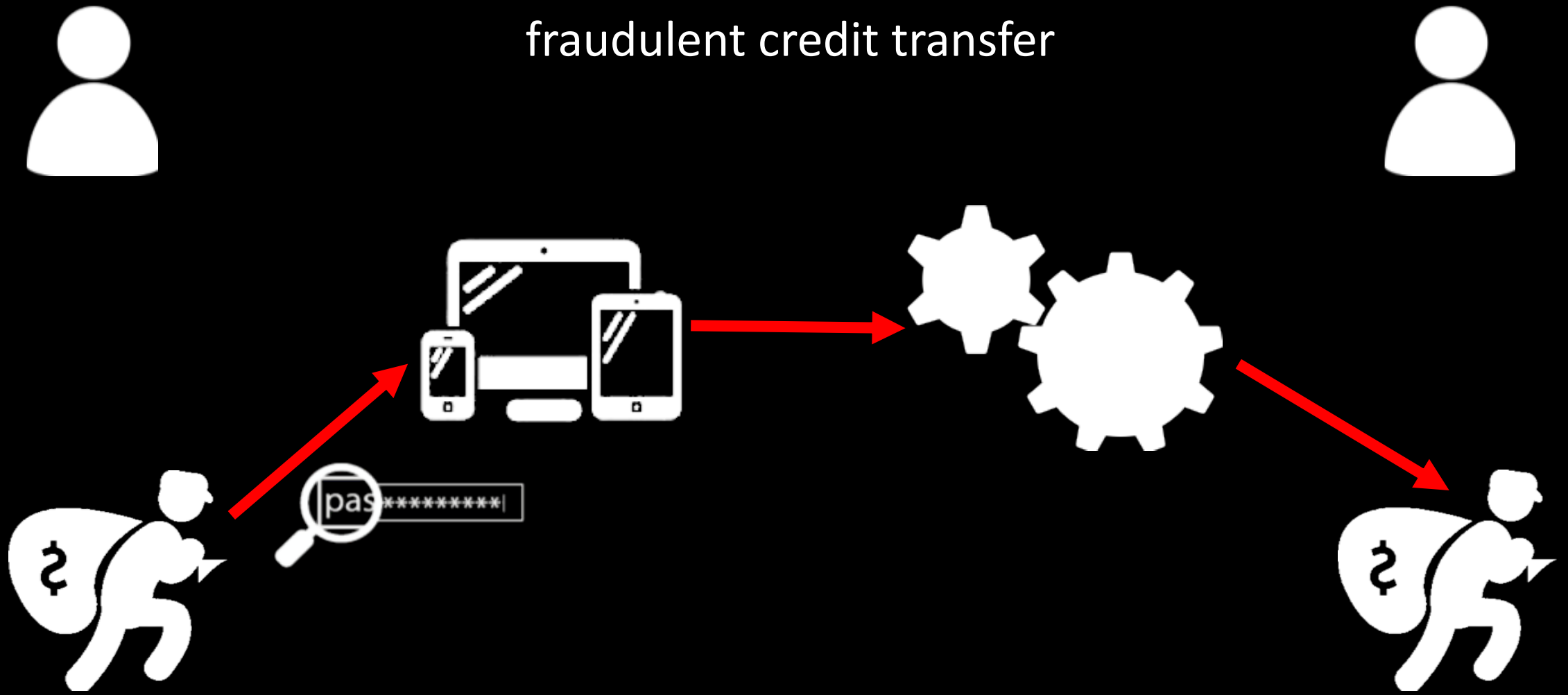
Hacking

Step 2: when the customer uses their device, the fraudster steals their credentials



Hacking

Step 3: the fraudster uses the stolen credentials to book a fraudulent credit transfer



Importance of Fraud Detection

- A typical organization loses **5% of its revenues** to fraud each year (www.acfe.com)
- The total cost of **insurance fraud** (non-health insurance) in the US is estimated to be more than **$40 billion per year** (www.fbi.gov)
- **Credit card** companies "lose approximately **seven cents per every hundred dollars** of transactions due to fraud" (Andrew Schrage, Money Crashers Personal Finance, 2012)
- "*The total value of fraudulent transactions conducted using **cards** issued within the Single Euro Payments Area (SEPA) and acquired worldwide amounted to Euro **1.33 billion in 2012**, which represented an increase of 14.8% from 2011.*" (European Central Bank, 2014)
- According to the European Insurance Committee, fraud takes up **5-10% of the claim amounts** paid for **non-life insurance**

➔ **Cost of missing fraud is huge!**

Fraud detection

Challenges

- **Few** fraudulent **cases** among torrent of data
- Keep operations **efficient** during **check**

Fraudsters

- Set-up **carefully organized** structures
- Try to **blend into norm**
- Continuously **adapt** and refine methods

Good clients

- **Avoid harassment** of e.g. blocked account

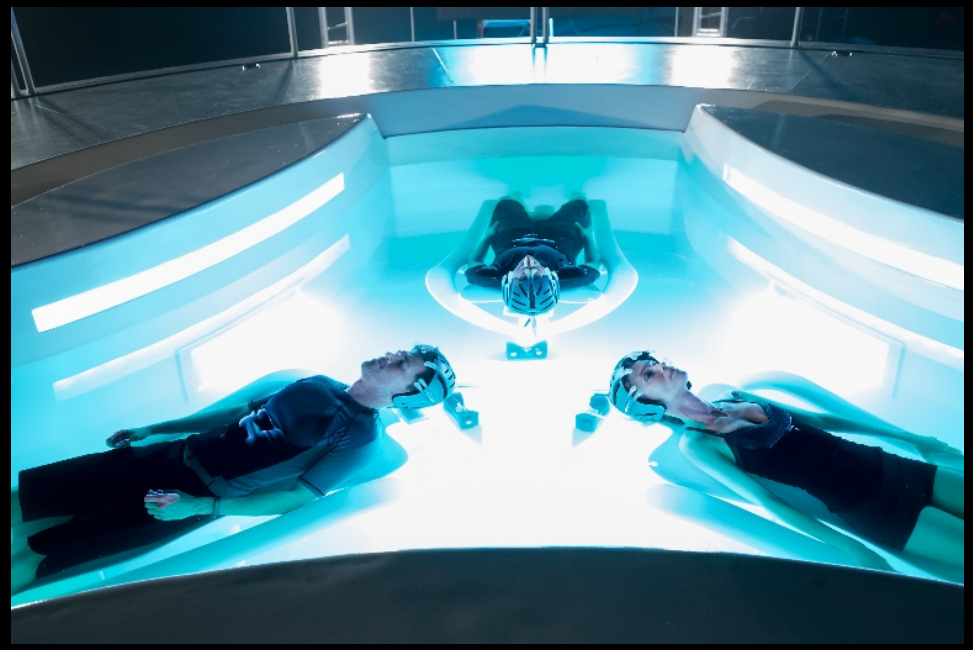


Fraud

- Uncommon
- Well-considered
- Concealed
- Time-evolving
- Many types

Key Characteristics of Successful Fraud Detection

- **Statistical accuracy**
  Hit rate, lift curves, …
- **Interpretability (?)**
  Understand the fraud patterns
- **Operational efficiency**
  Response time, online scoring, …
- **Economical cost**
  Model cost, software, data
- **Regulatory compliance**
  In line with regulation, privacy, …



Fraud detection

Rule-based

For example:

- Cash transactions > € 10.000 .
- To/from suspicious countries.
- To/from people on blacklist.
- Within hour same card different countries.

Rule-based vs ML-based Fraud Detection Systems

Rule-based fraud detection	ML-based fraud detection
Catching obvious fraudulent scenarios	Finding hidden and implicit correlations in data
Requires much manual work to enumerate all possible detection rules	Automatic detection of possible fraud scenarios
Multiple verification steps that harm user experience	The reduced number of verification measures
Long-term processing	Real-time processing

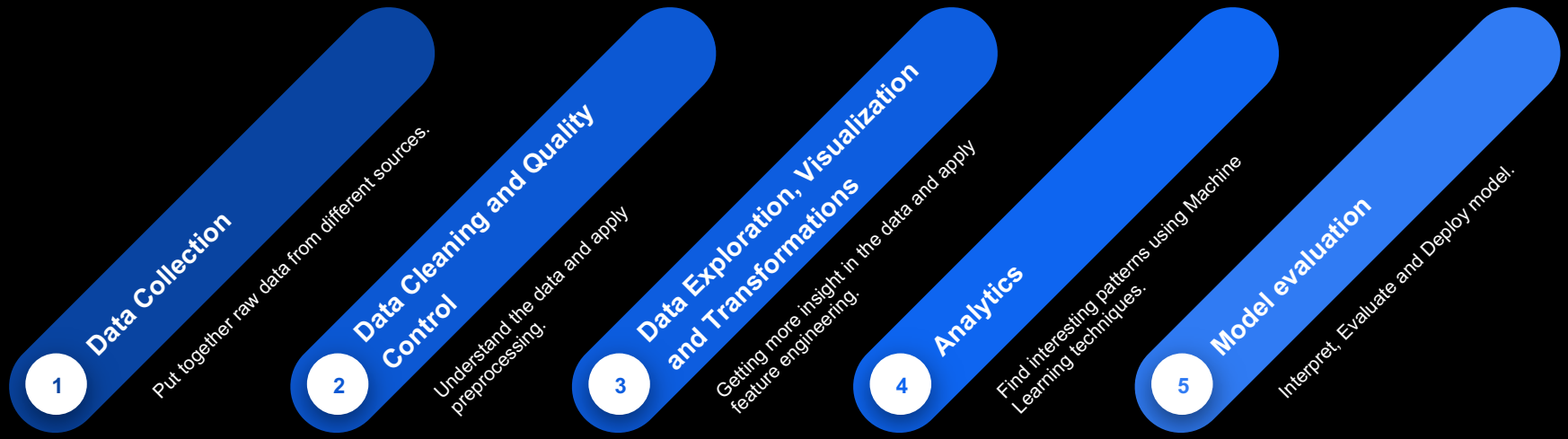
Source: AltexSoft

Expensive to build and difficult to maintain!

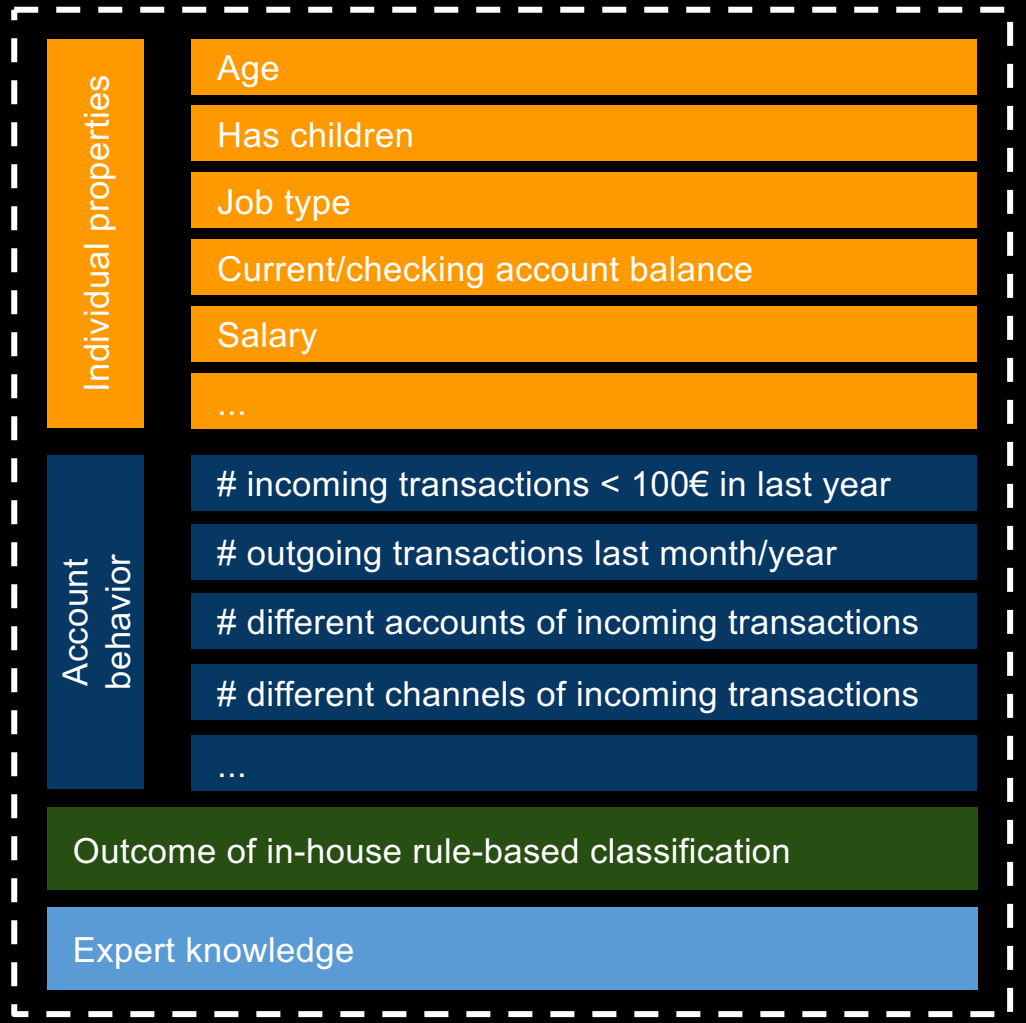
Table of Contents

- Introduction
- Obtain fraud score using machine learning
- Dealing with imbalanced data
- Evaluate performance taking costs into account
- Design cost-sensitive fraud detection model
- Anomaly detection

Fraud analytics process model

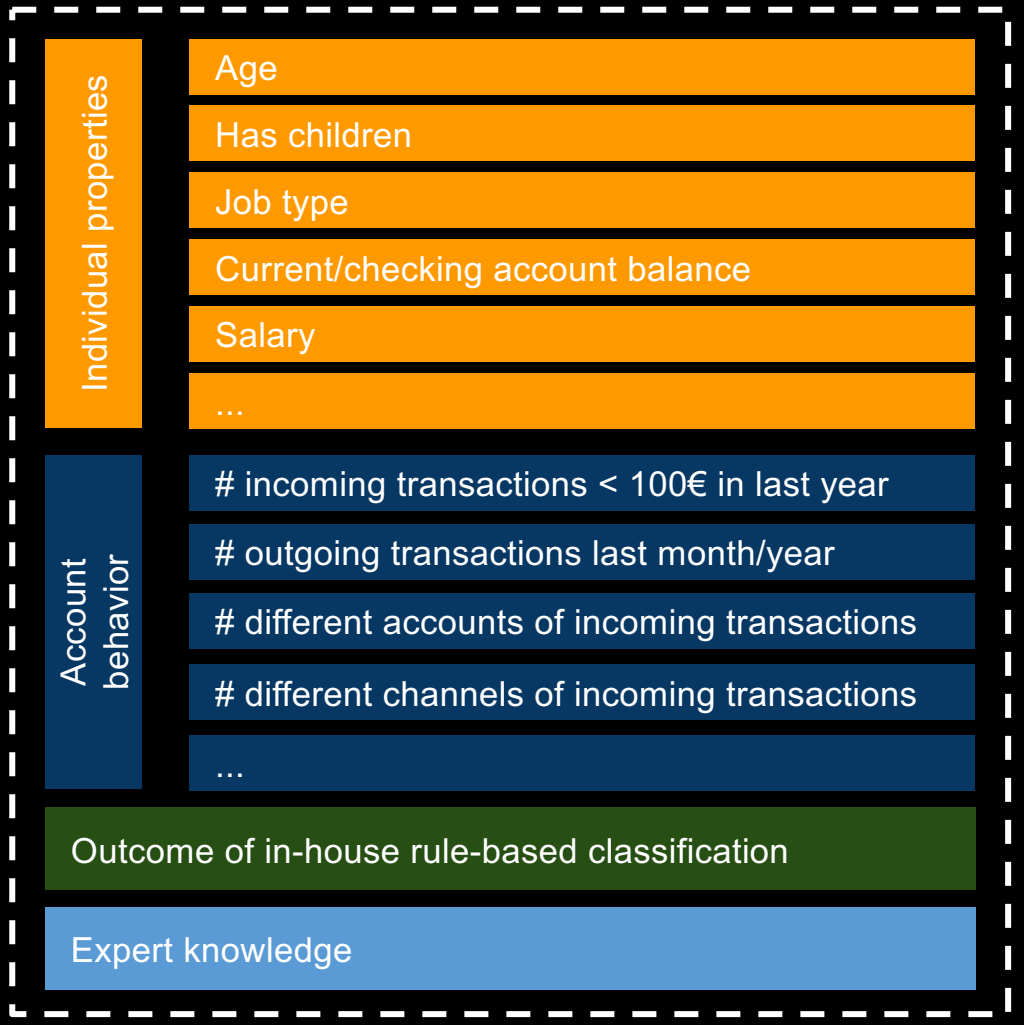


Fraud score = f(



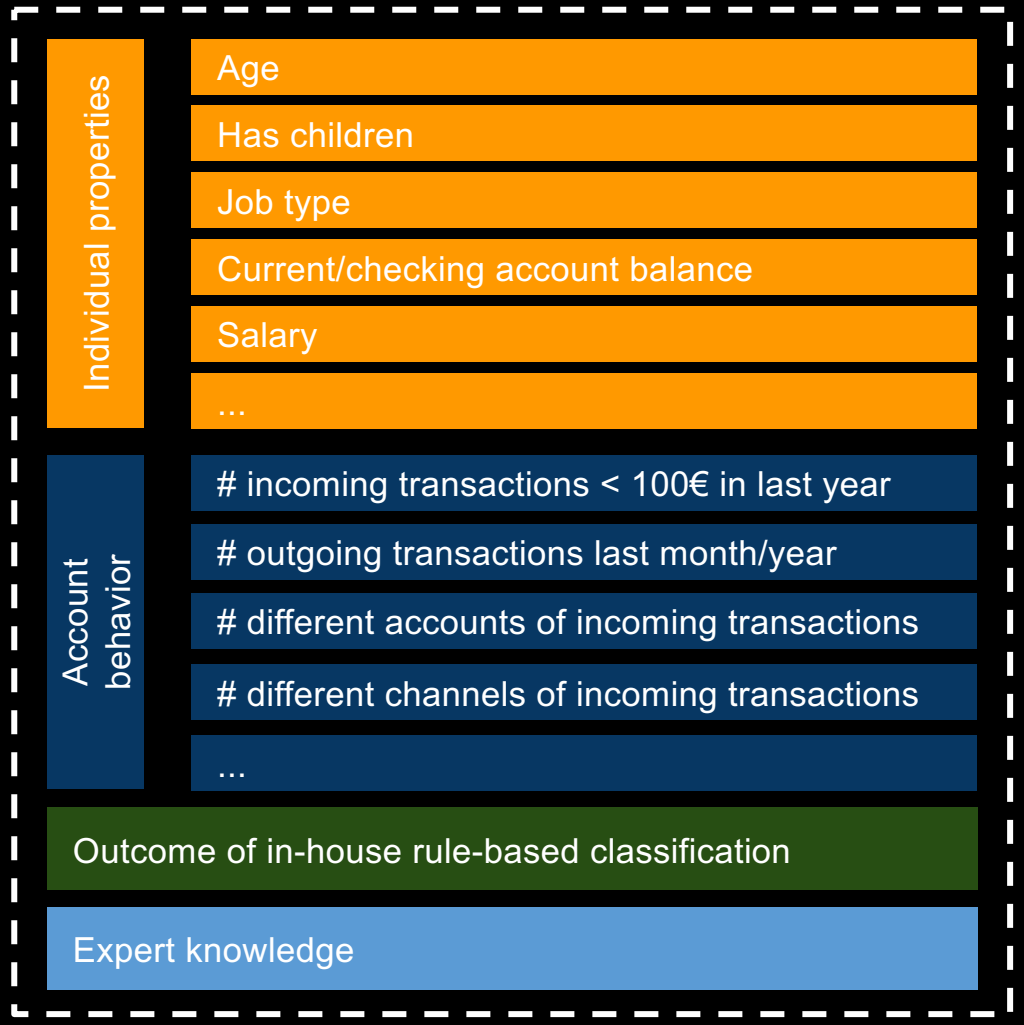
)

Fraud score = f( )



- Select best predictive model

Fraud score = f( )

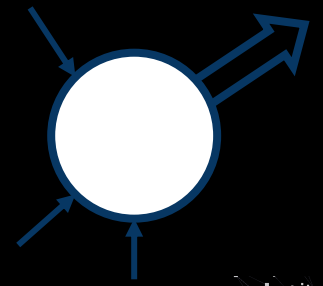


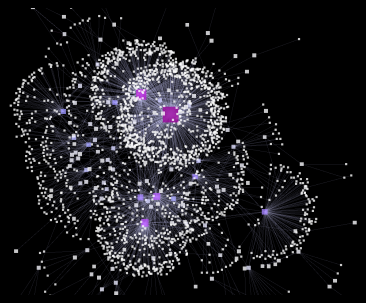
- Select best predictive model
- Feature engineering

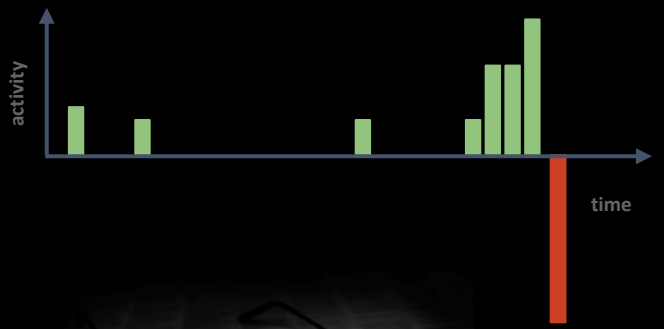
Fraud score = f(

- **Individual**
  - Feature 1
  - Feature 2
  - ...
- **Account behavior**
  - Feature 1
  - Feature 2
  - ...
- **Network features**
  - Feature 1
  - Feature 2
  - ...
- **Anomaly detection**
  - Feature 1
  - Feature 2
  - ...
- **External source**
  - Feature 1
  - Feature 2
  - ...
- **NLP (comm.)**
  - Feature 1
  - Feature 2
  - ...
- **In-house predictions**
- **Expert knowledge**

)









"I tried calling several times"

Examples in payment fraud

It is important to engineer additional features that will help detecting fraud in your dataset.

Let us focus on some examples for payment fraud.

Time is an important aspect in fraud detection:

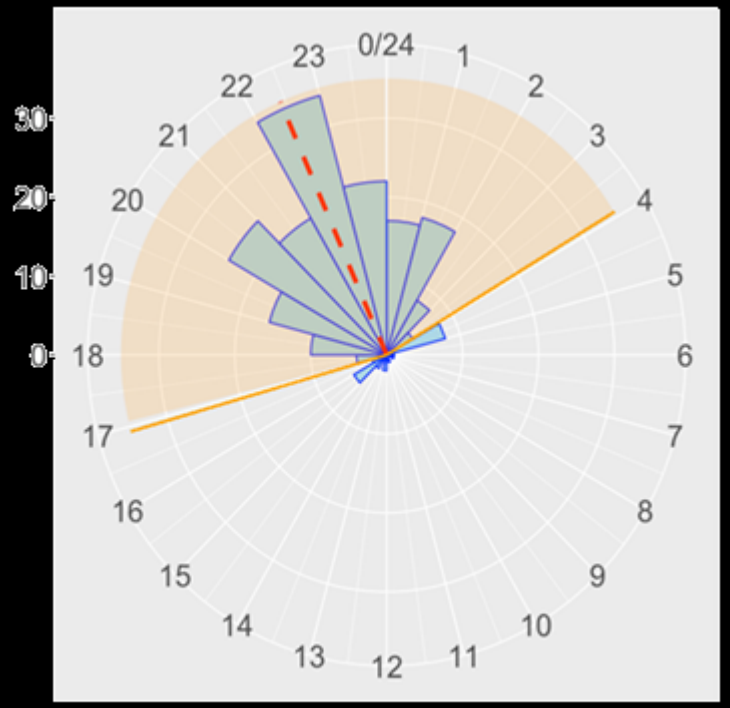
- There exist tools to analyze timestamps and create features that indicate suspicious behaviour.
- Recency features capture the dimension of time in such a way that anomalous behavior of fraud cases are highlighted.
- Frequency counts the number of previous transfers with the same authentication method as the current one





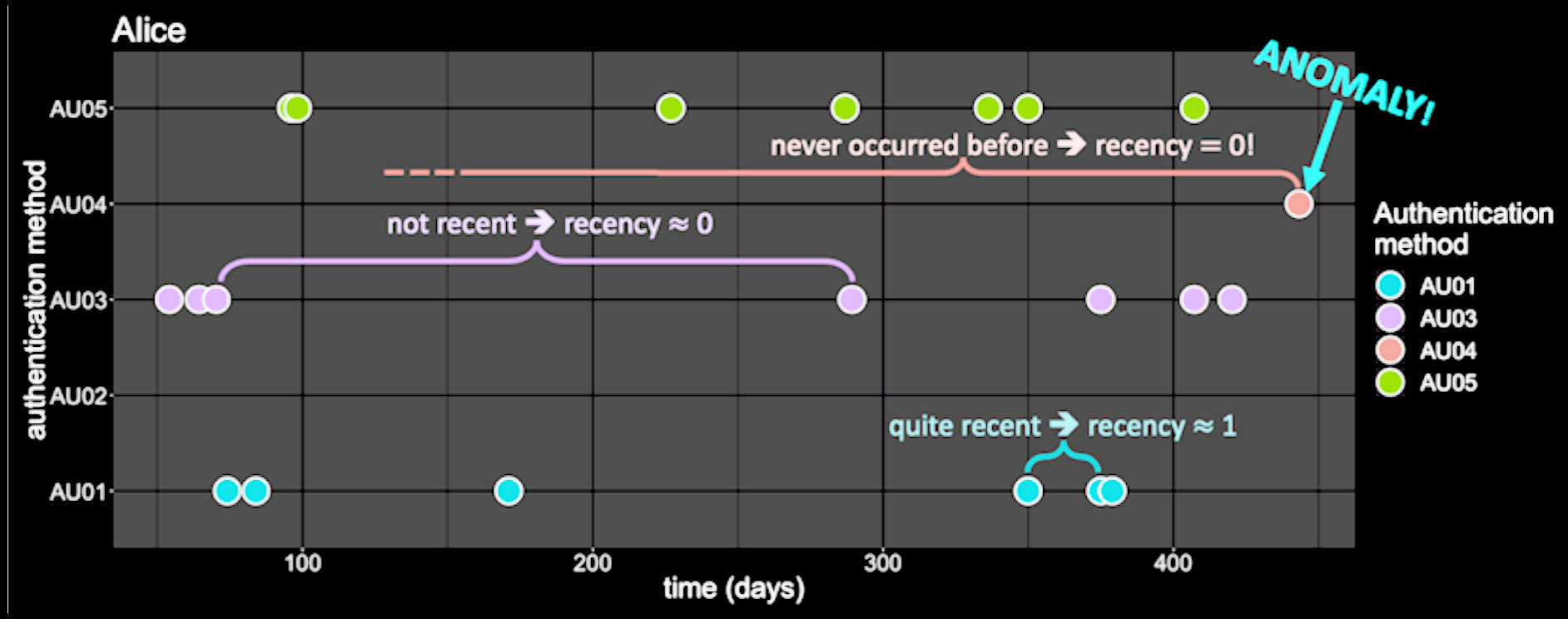
Example: time aspect

- Certain events are expected to occur at similar moments in time
  - customer making transactions at similar hours
- **Aim**: capture information about the time aspect by meaningful features
  - Model time as a periodic variable using von Mises distribution
  - Periodic normal distribution = normal distribution wrapped around a circle
- Dealing with time can be tricky
  - 00:00 = 24:00
  - no natural ordering: 23:00 < , > 01:00?
  - Do not use arithmetic mean to compute an average timestamp!



Example: recency of authentication method

Recency features capture the dimension of time in such a way that anomalous behavior of fraud cases are highlighted



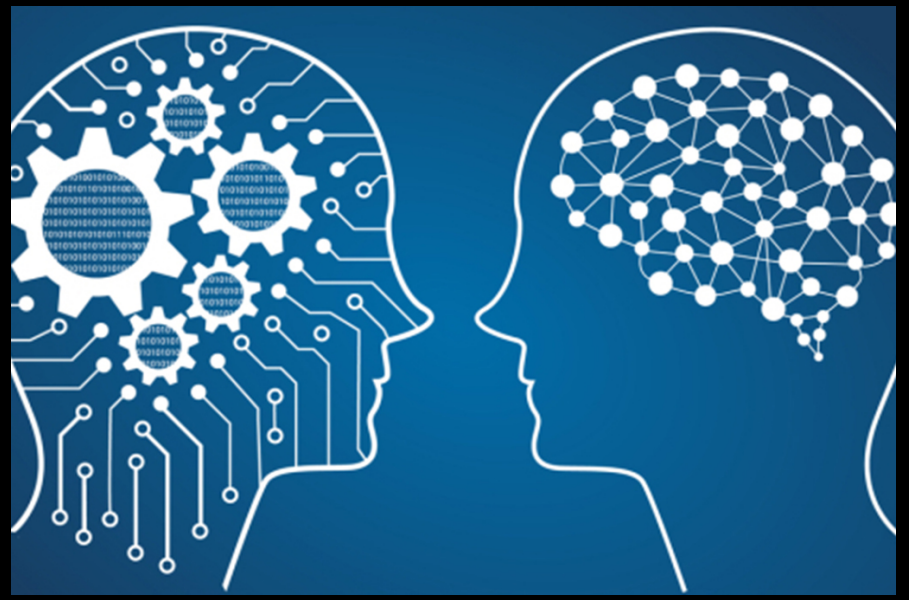
Network analysis

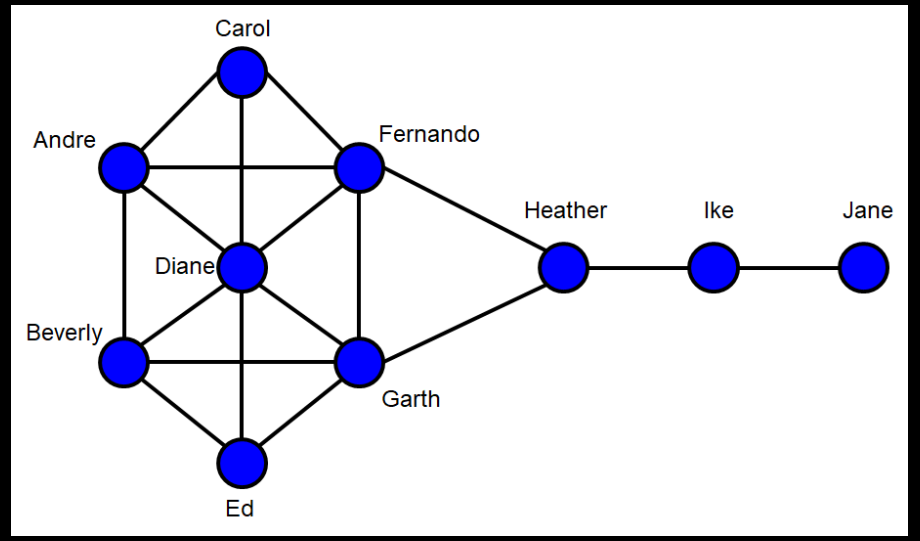
Networks are everywhere

Capture influence of peers/environment
Effect of entity on rest of networs

Find most important/influential entities!
Determine node importance

- Person with most connections?
  - Degree centrality
- Person on average closest to everyone?
  - Closeness centrality
- Person connecting communities?
  - Betweenness centrality





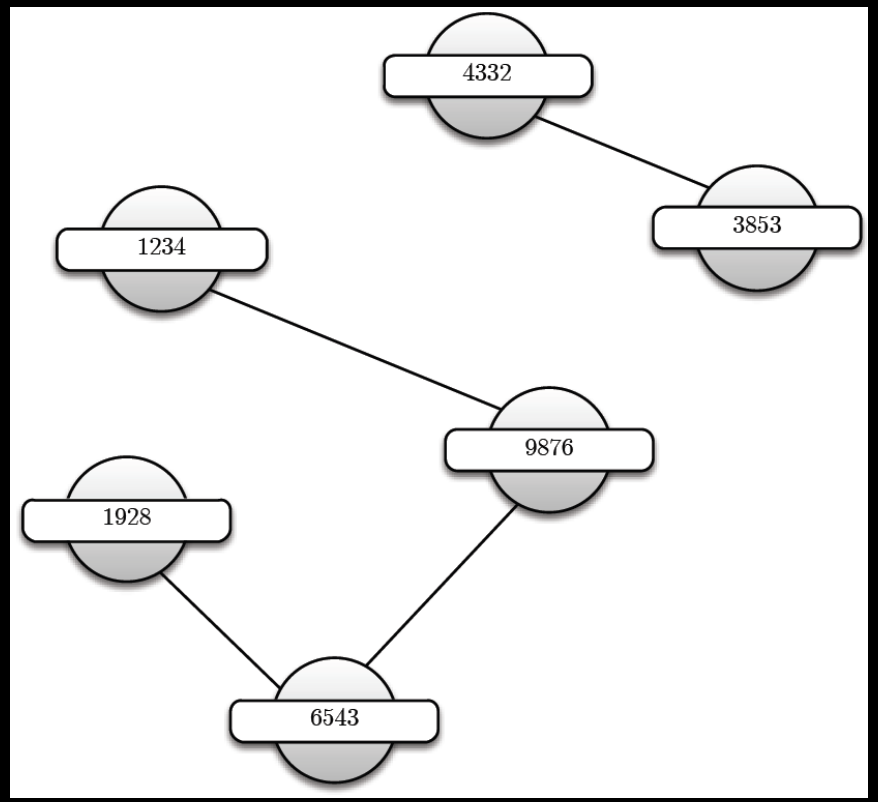
Social Network Components

From a transactional data source…

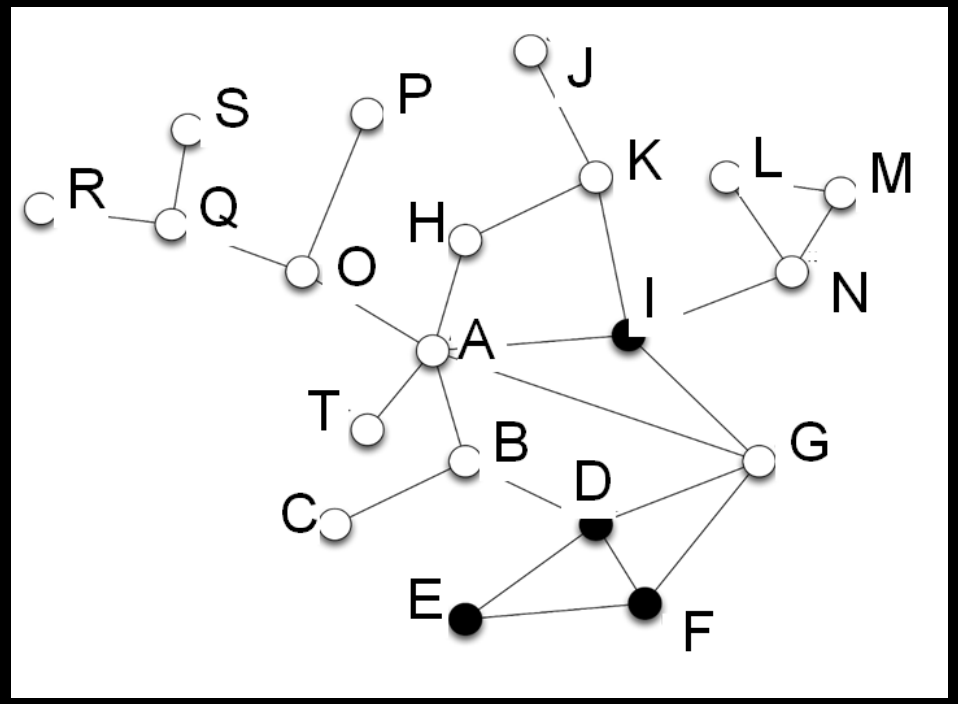
From	To	Duration	Location (from)	Location (to)	Provider (from)	Provider (to)
1234	9876	5:40	Brussels	Leuven	A	B
4567	6543	1:05	Leuven	Leuven	A	C
1234	9876	2:34	Brussels	Leuven	A	B
6543	1928	2:02	Leuven	Ghent	C	A
4332	3853	0:46	Bruges	Antwerp	A	A
9876	1234	3:51	Antwerp	Brussels	B	A
9876	6543	6:57	Leuven	Ghent	B	C

Social Network Components

… toward a network



Add features based on network



Node	A	B	C	D	E	F	G	H	I	J
Total Degree	6	3	1	4	2	3	4	2	4	1
Fraud Degree	1	1	0	2	2	2	3	0	0	0
Legit Degree	5	2	1	2	0	1	1	2	4	1
Total Triangles	1	0	0	2	1	2	2	0	1	0
Fraud Triangles	0	0	0	1	1	1	1	0	0	0
Legit Triangles	0	0	0	0	0	0	0	0	1	0
Semi-Fraud Triangles	1	0	0	1	0	1	1	0	0	0
Geodesic Path	1	1	2	0	0	0	1	2	0	2
# 1-hop Paths	1	1	0	2	2	2	3	0	0	0
# 2-hop Paths	4	3	1	8	4	7	5	2	6	1
# 3-hop Paths	18	13	3	19	15	17	25	4	9	0
Closeness	2	2.57	3.53	2.89	3.79	3.00	2.32	2.74	2.21	3.89
Betweenness	104	24.67	0	13.83	0	6.167	35.33	9	65	0

Table of Contents

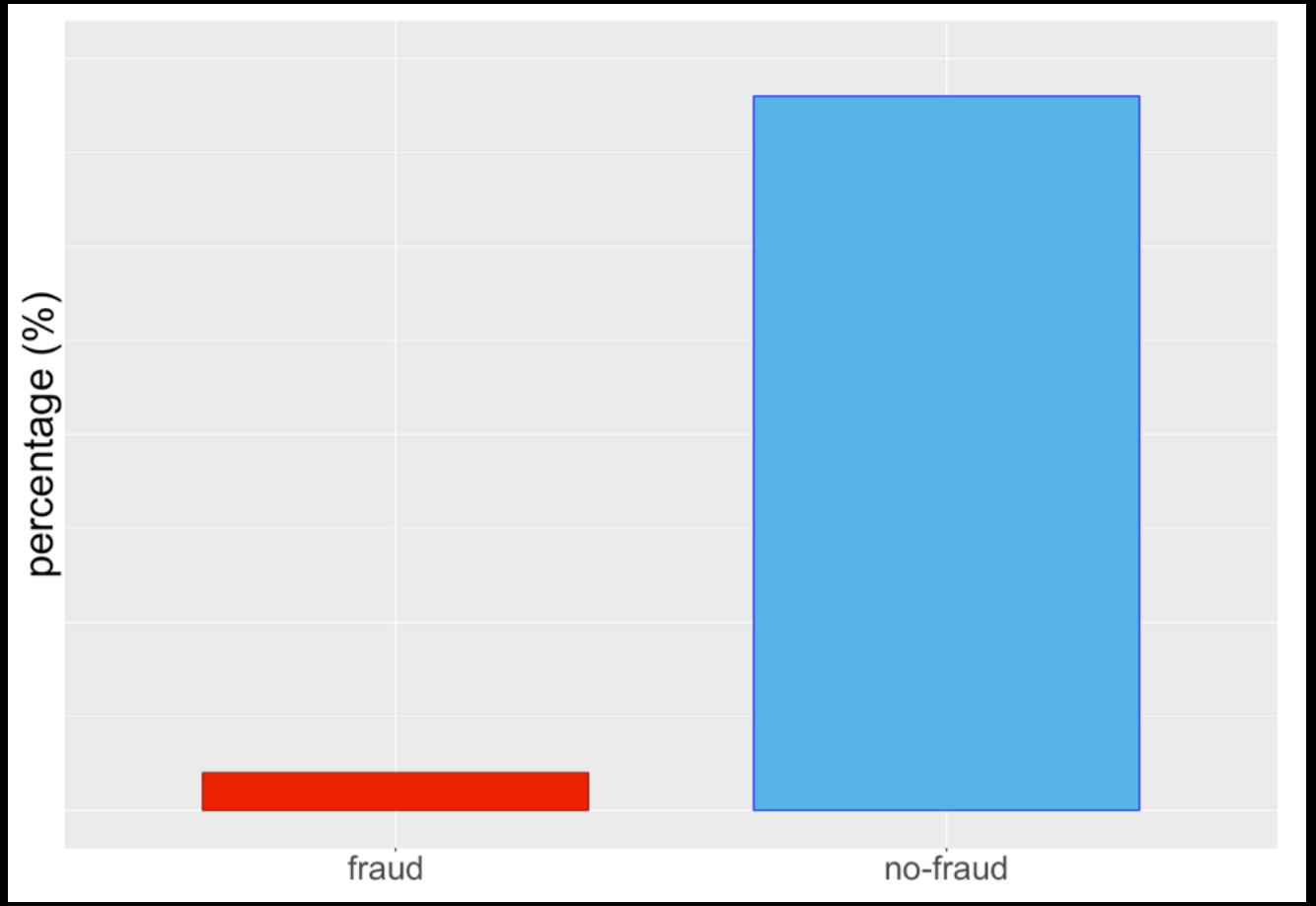
- Introduction
- Obtain fraud score using machine learning
- Dealing with imbalanced data
- Evaluate performance taking costs into account
- Design cost-sensitive fraud detection model
- Anomaly detection

Problem: imbalanced data

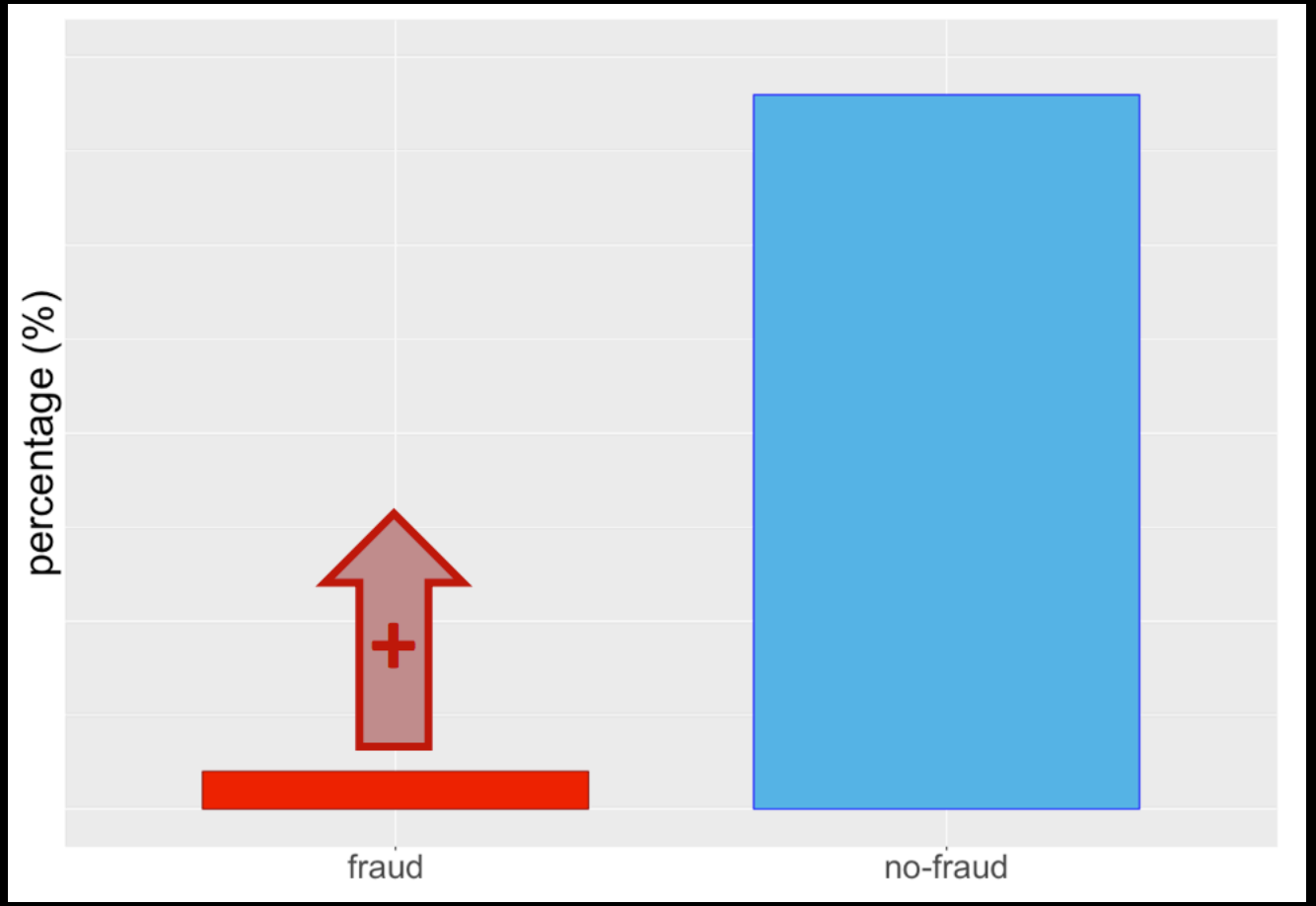
- ***Binary classification***
  legitimate vs fraud
- ***Imbalanced data***
  (very) large difference in number of observations of both groups
- ***Credit card fraud***
  less than 1 out 10m transactions (< 0.00001%)
- Typically very few “cases of interest” compared to legitimate observations (20% - 0.01%)



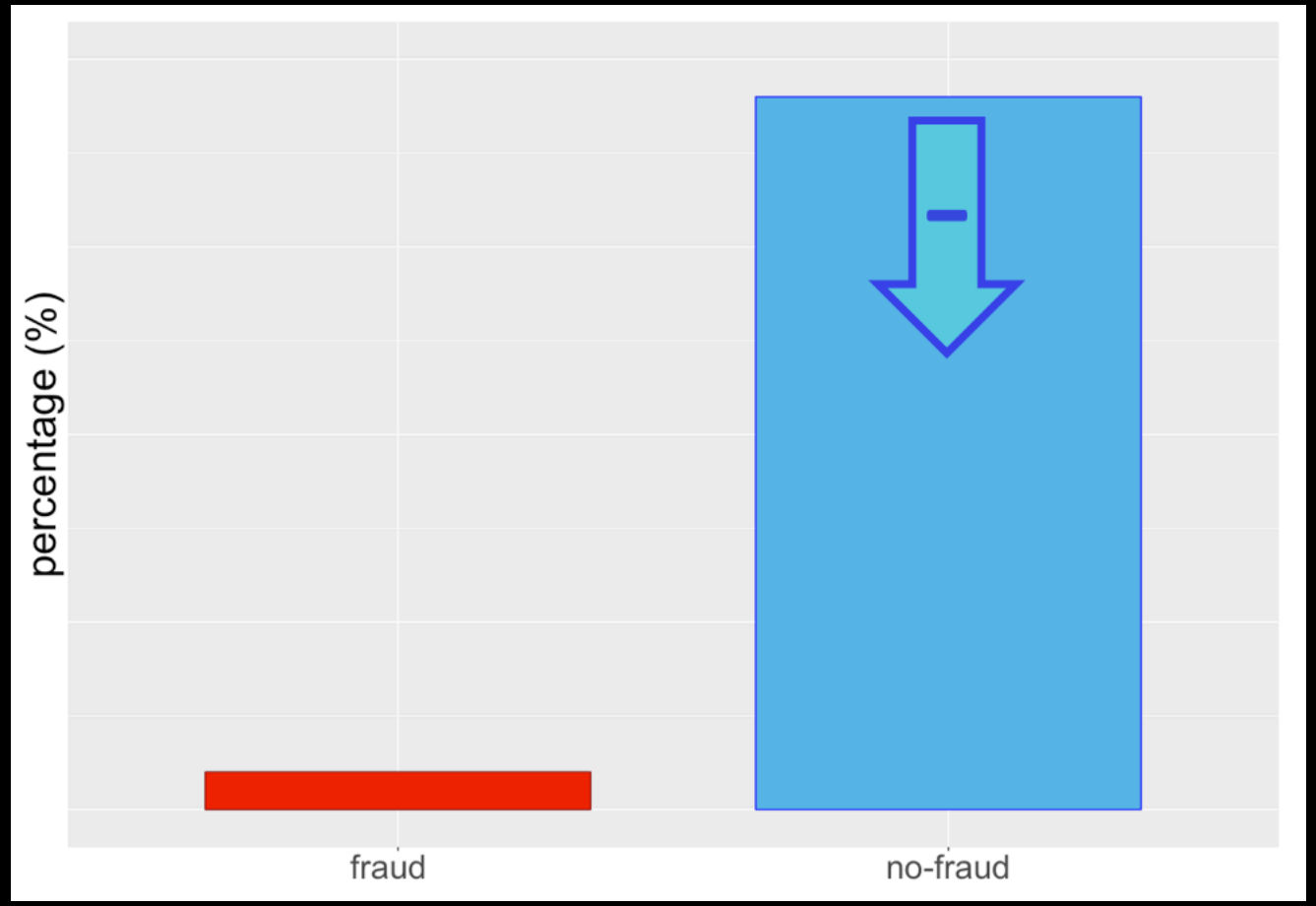
Imbalanced data sets



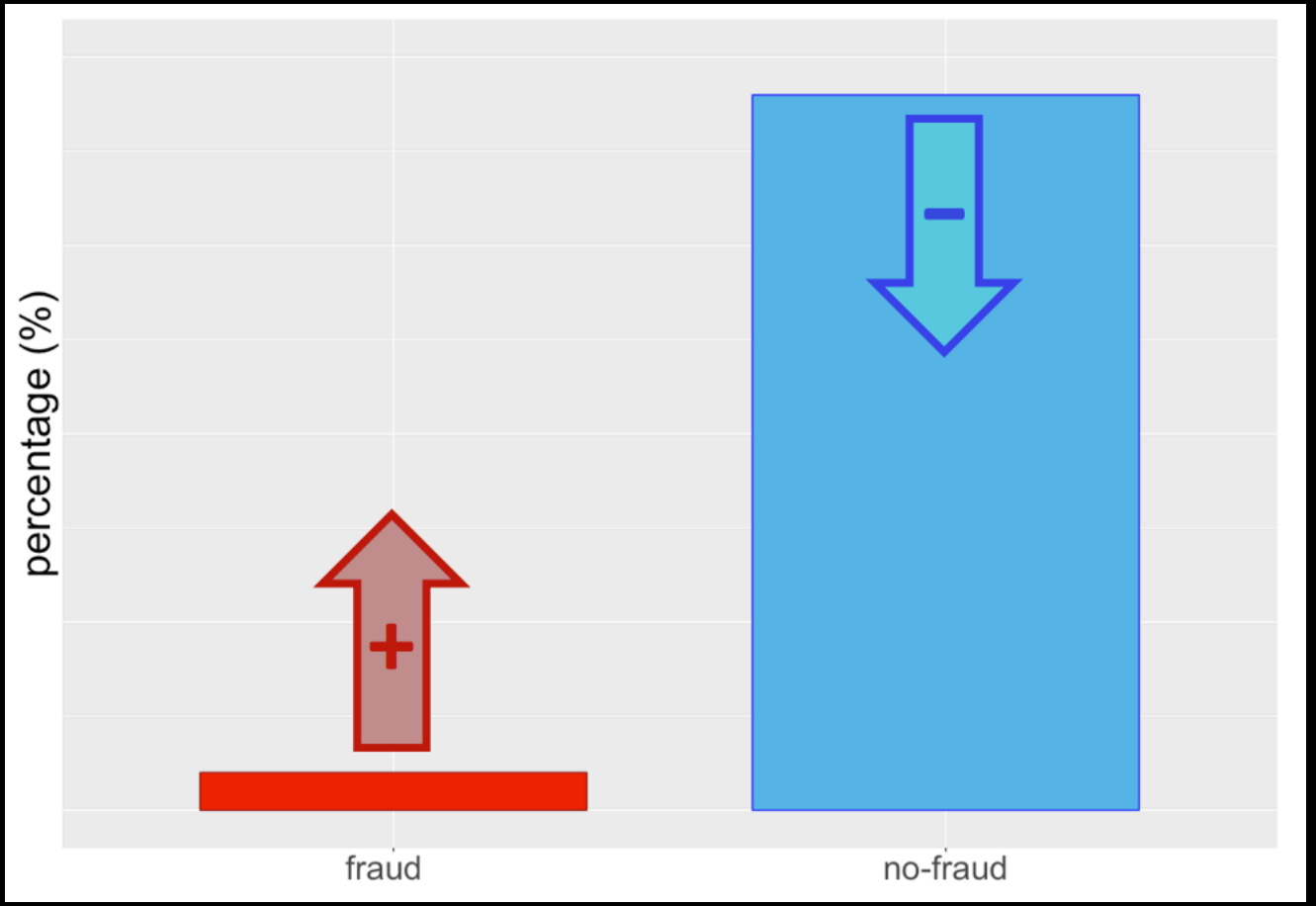
Over-sampling minority class...



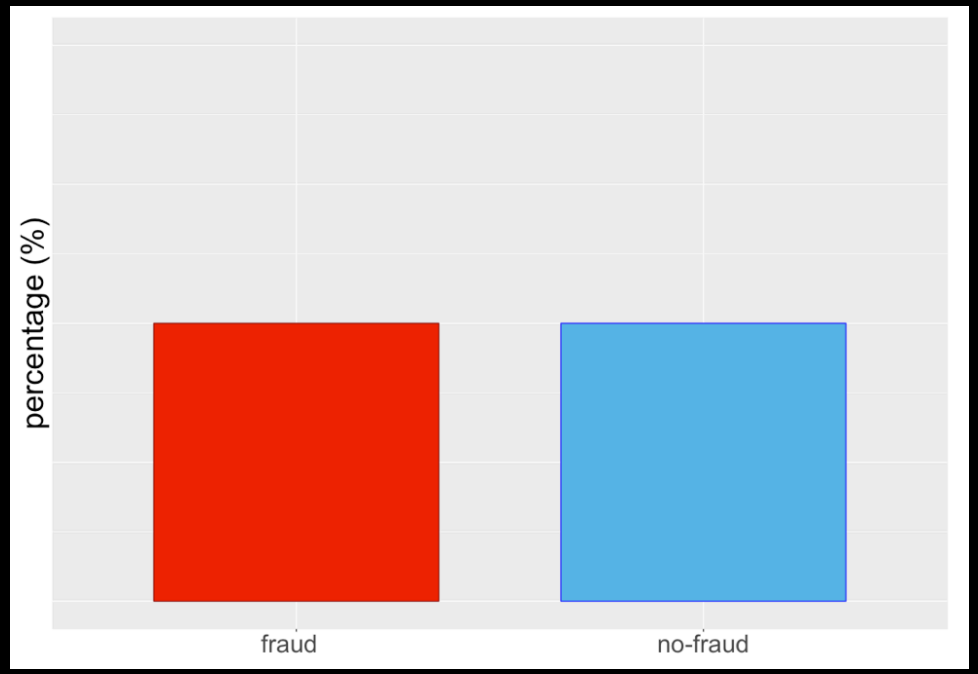
… or under-sampling majority class …

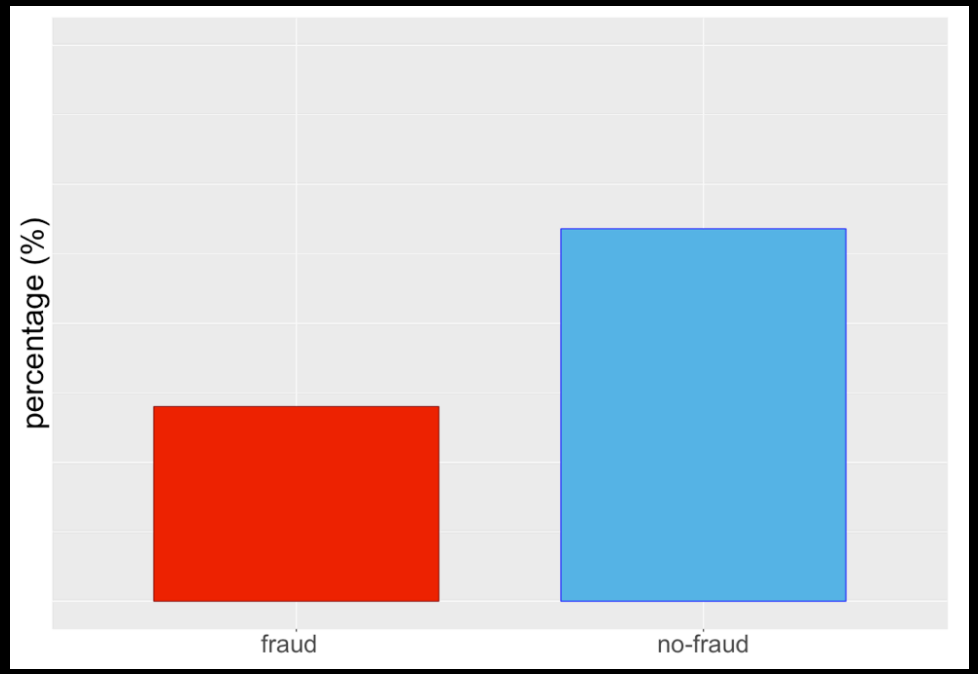


Or both…



Result after sampling...





"How to reduce imbalance in training data?"

Reduce # legitimate cases

- Random under-sampling: randomly sub-sample legitimate cases

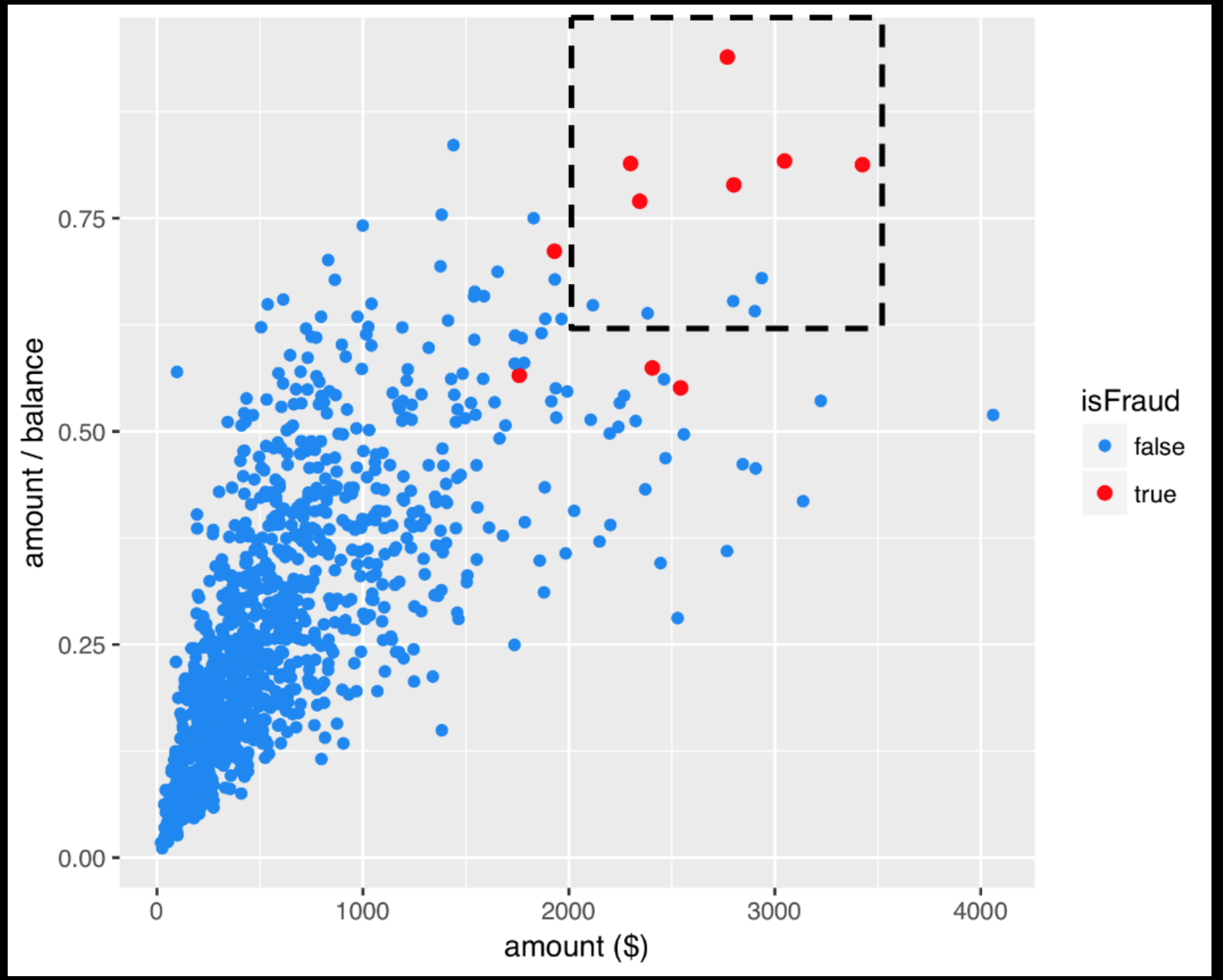
Increase # fraud cases

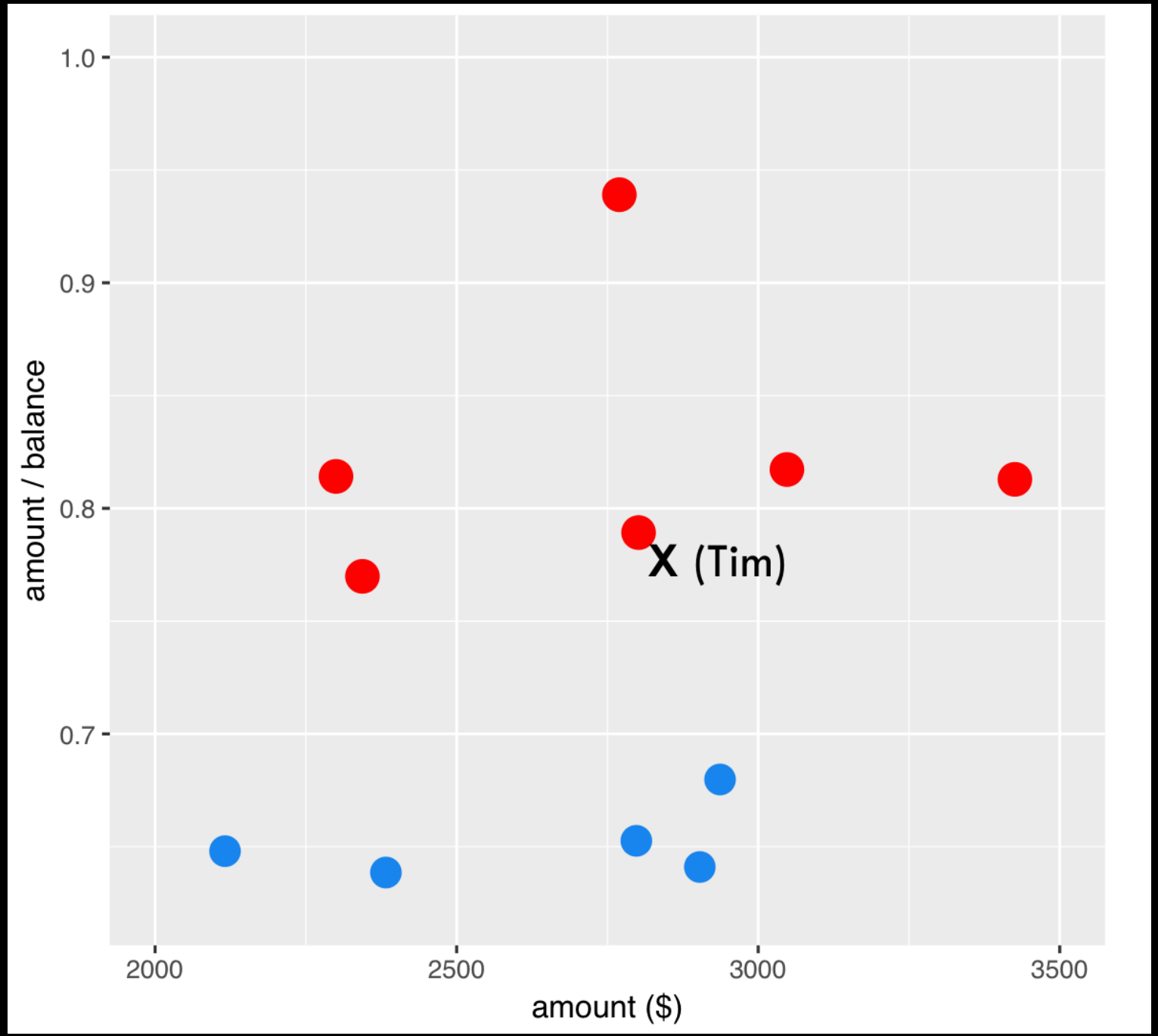
- Random over-sampling: sampling with replacement of fraud samples
- ***Generate synthetic minority/fraud cases***

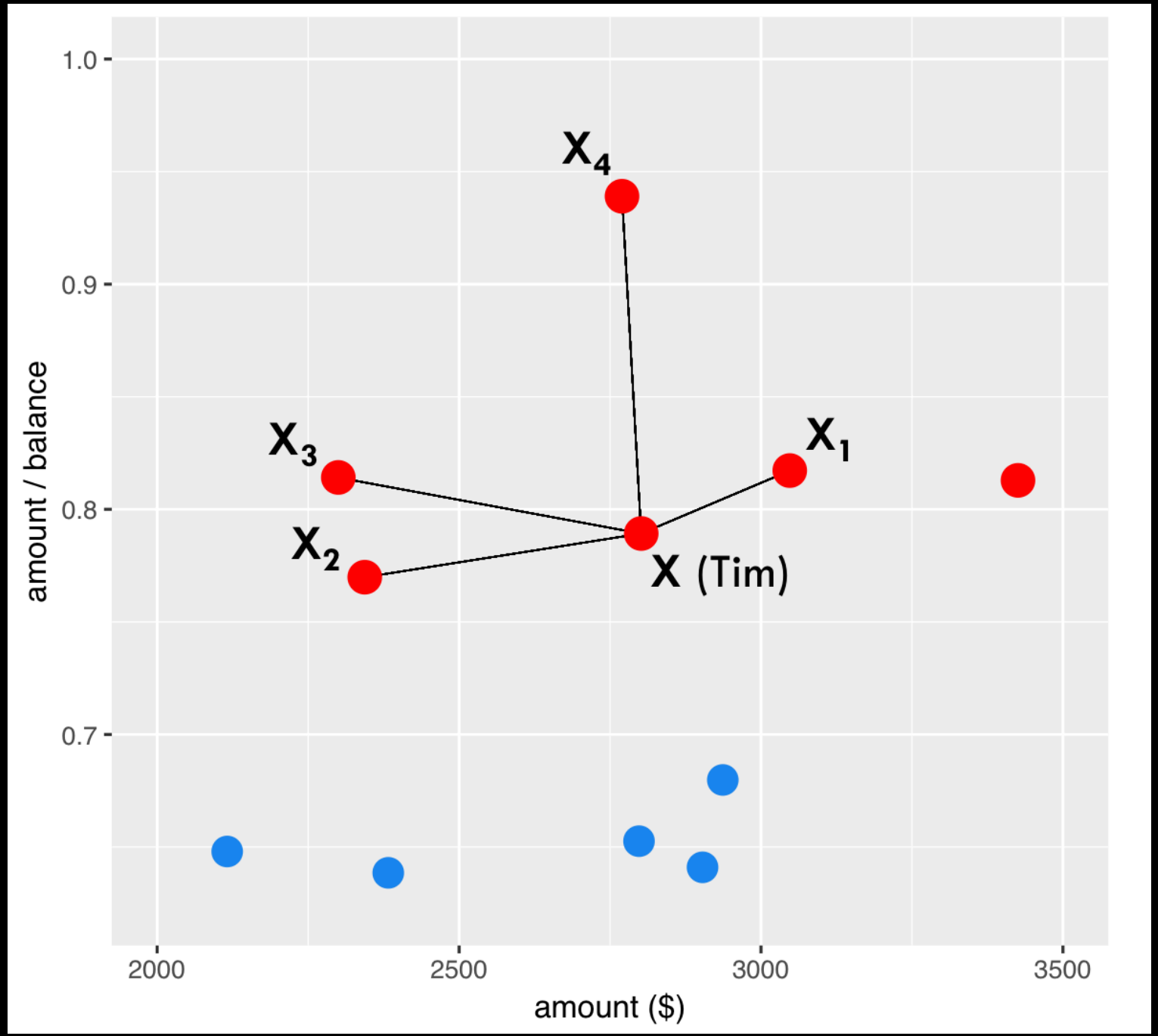
Evaluate model on imbalanced "original" test data !!!

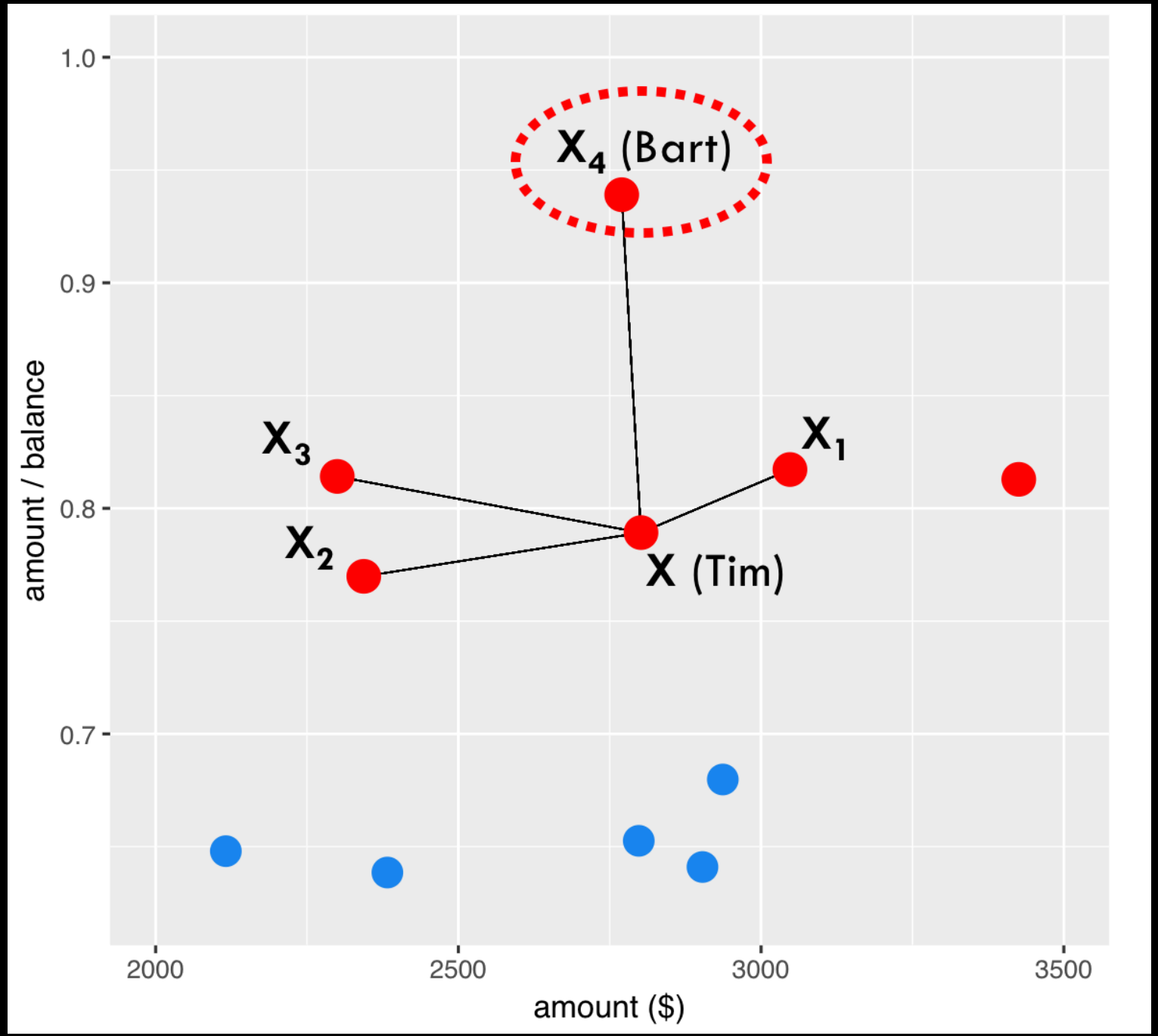
SMOTE -
Synthetic Minority Over-sampling Technique

Chawla, Bowyer, Hall & Kegelmeyer (2002)


amount / balance
0.75
0.50
0.25
0.00
0
1000
2000
3000
4000
amount ($)
isFraud
false
true


1.0
0.9
0.8
0.7
amount / balance
X (Tim)
2000
2500
3000
3500
amount ($)


1.0
0.9
0.8
0.7
amount / balance
2000
2500
3000
3500
amount ($)
X4
X3
X1
X2
X (Tim)


1.0
0.9
0.8
0.7
amount / balance
2000
2500
3000
3500
amount ($)
X4 (Bart)
X3
X1
X2
X (Tim)

SMOTE

X (Tim)

Amount	Ratio
2800	0.79

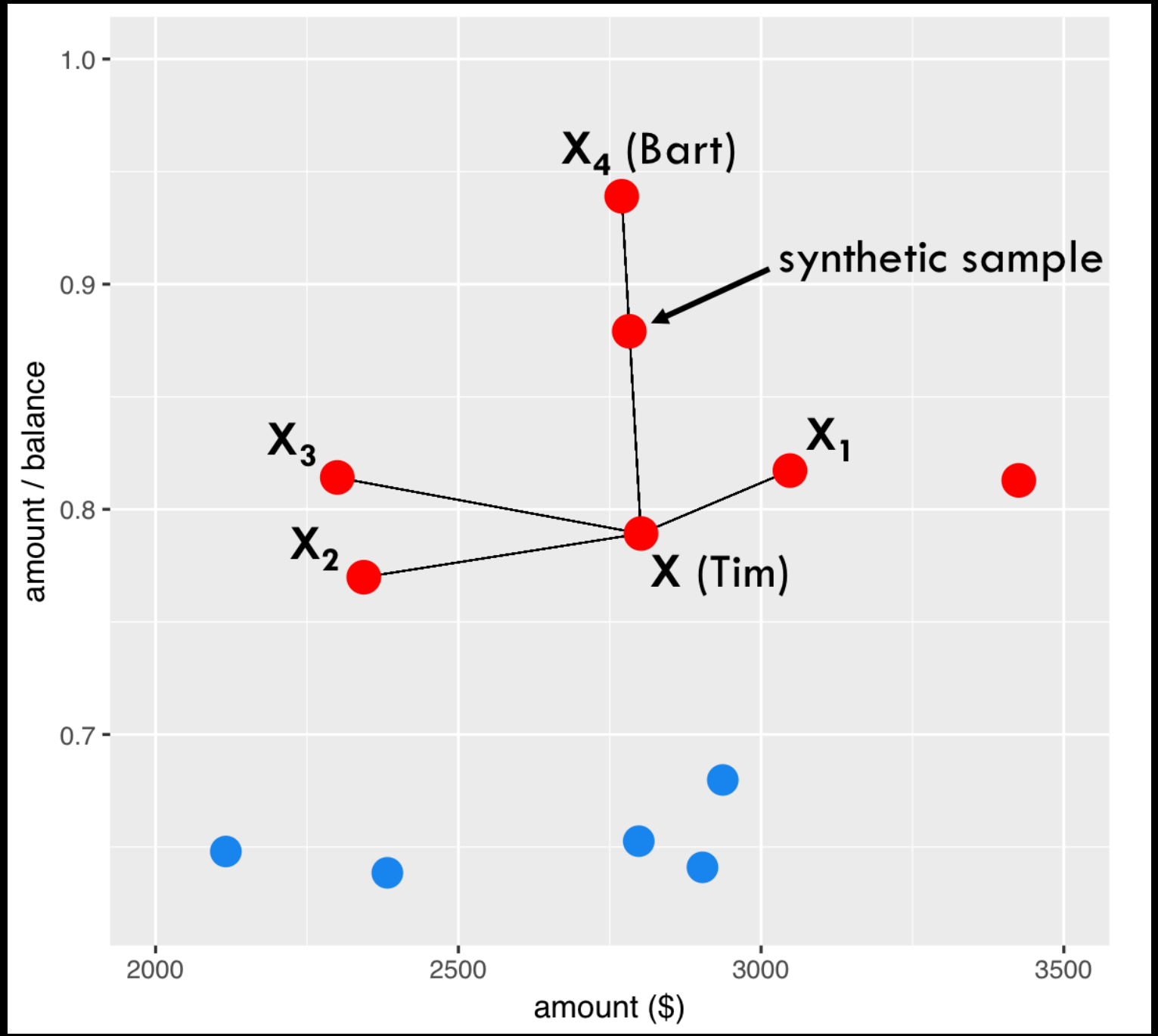
X_4 (Bart)

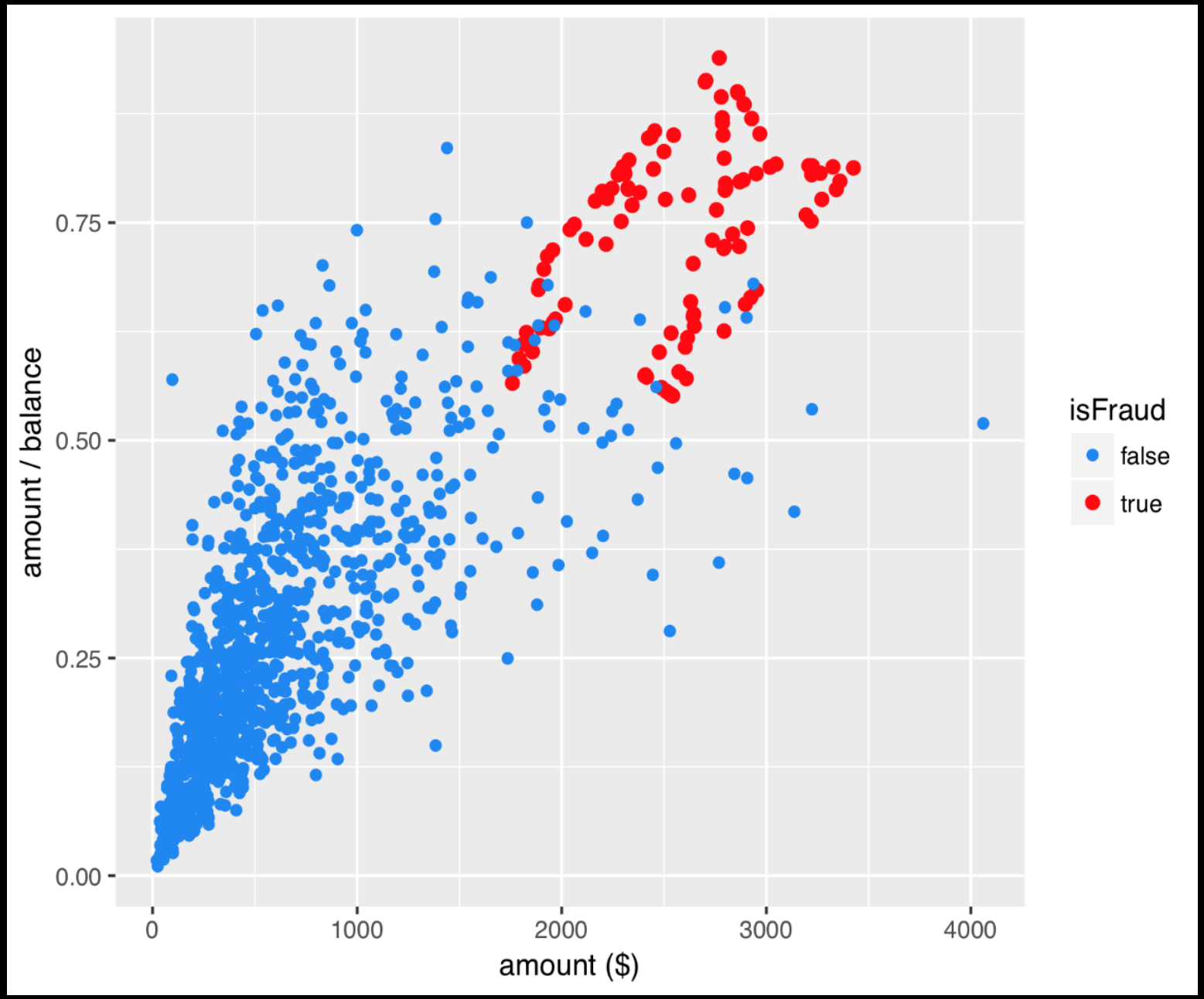
Amount	Ratio
2770	0.94

Choose random number between 0 and 1, e.g. **0.6**

Synthetic sample

Amount	Ratio
2800 + **0.6** * (2770 - 2800) = **2782**	0.79 + **0.6** * (0.94 – 0.79) = **0.88**


1.0
0.9
0.8
0.7
amount / balance
2000
2500
3000
3500
amount ($)
X4 (Bart)
synthetic sample
X3
X1
X2
X (Tim)

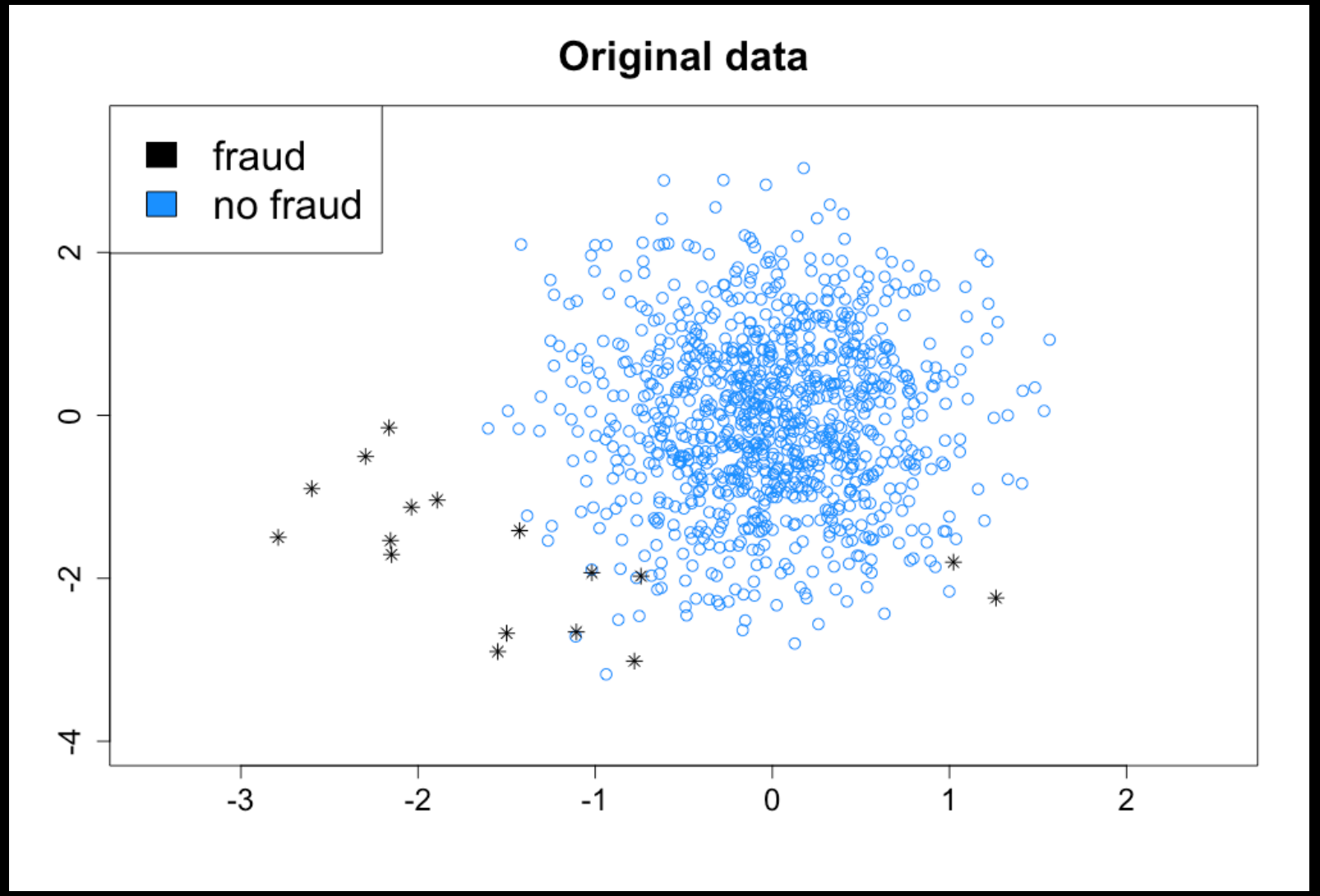

0.75
0.50
0.25
0.00
amount / balance
0
1000
2000
3000
4000
amount ($)
isFraud
false
true

Some variants/alternatives of SMOTE

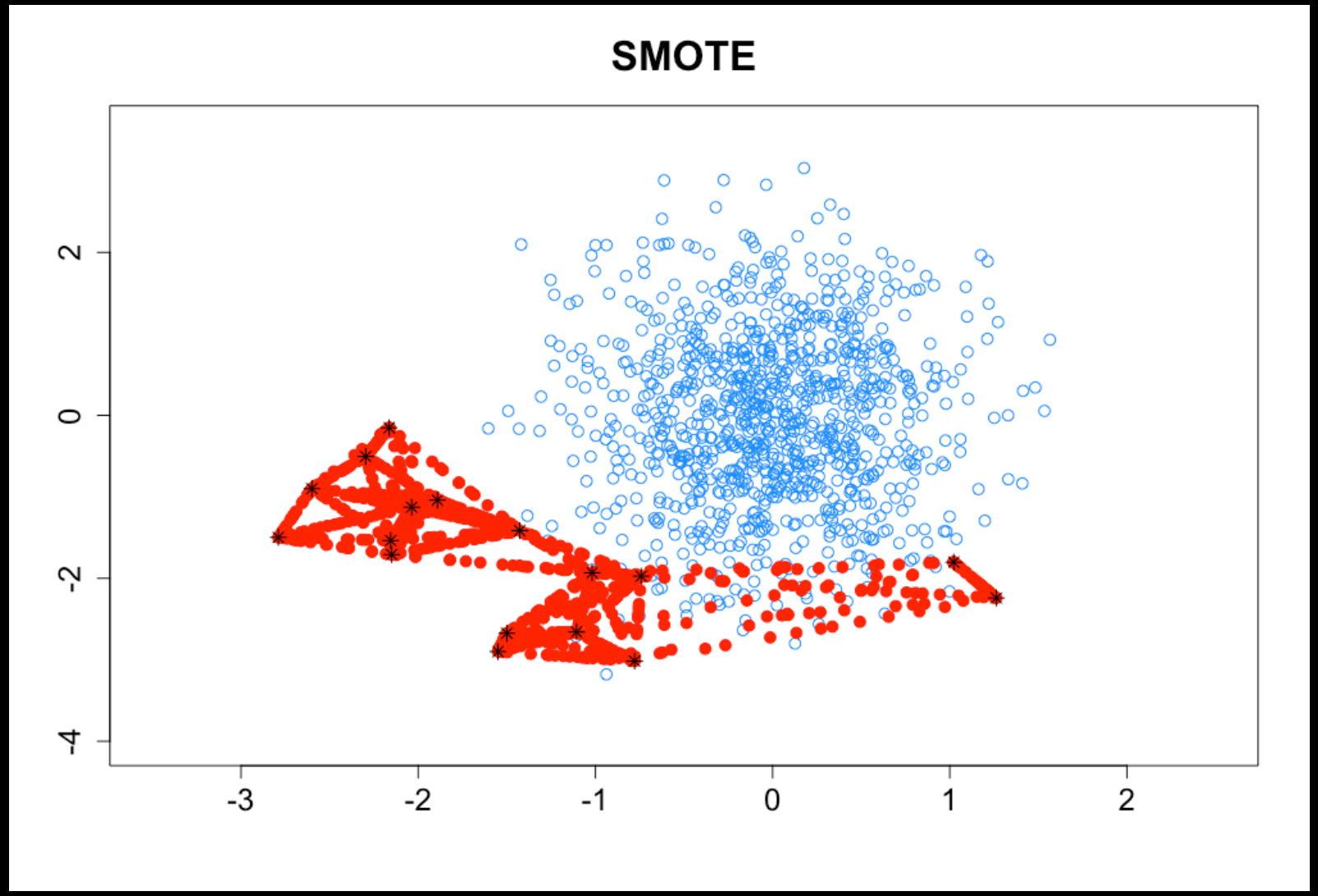
- Borderline-SMOTE (Han et al., 2005)
- ADASYN (He et al., 2008)
- Safe-level SMOTE (Bunkhumpornpat et al., 2009)
- Density-based SMOTE (Bunkhumpornpat et al., 2011)
- MWMOTE (Barua et al., 2014)
- ROSE (Menardi & Torelli, 2014)
- RobROSE (Baesens, Höppner, Ortner, Verdonck, 2020)

Zhu, B., Gao, Z., Zhao, J., vanden Broucke, S. (2019). IRIC: An R library for binary imbalanced classification. *SoftwareX*, *10*. doi: 10.1016/j.softx.2019.100341

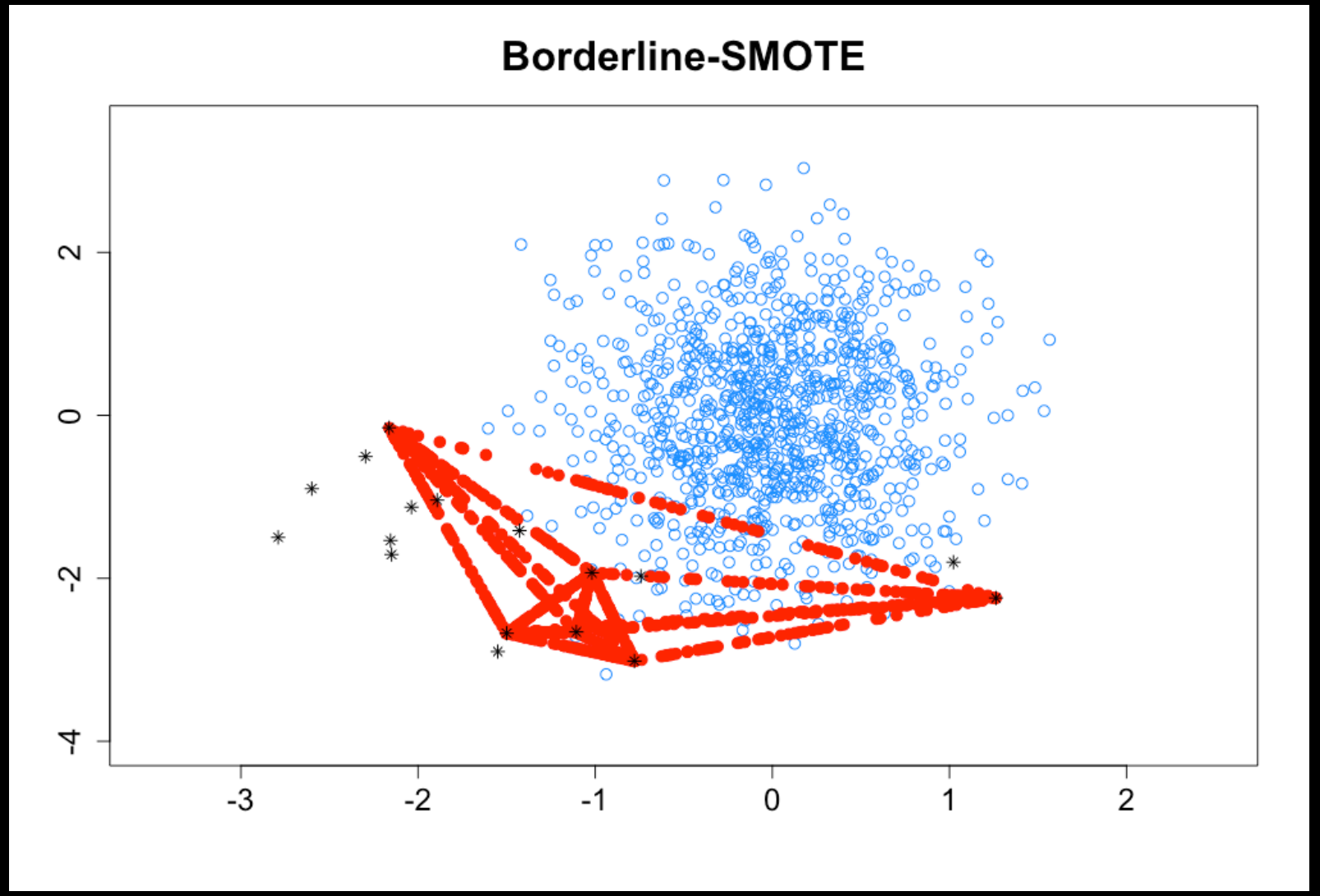
Original data



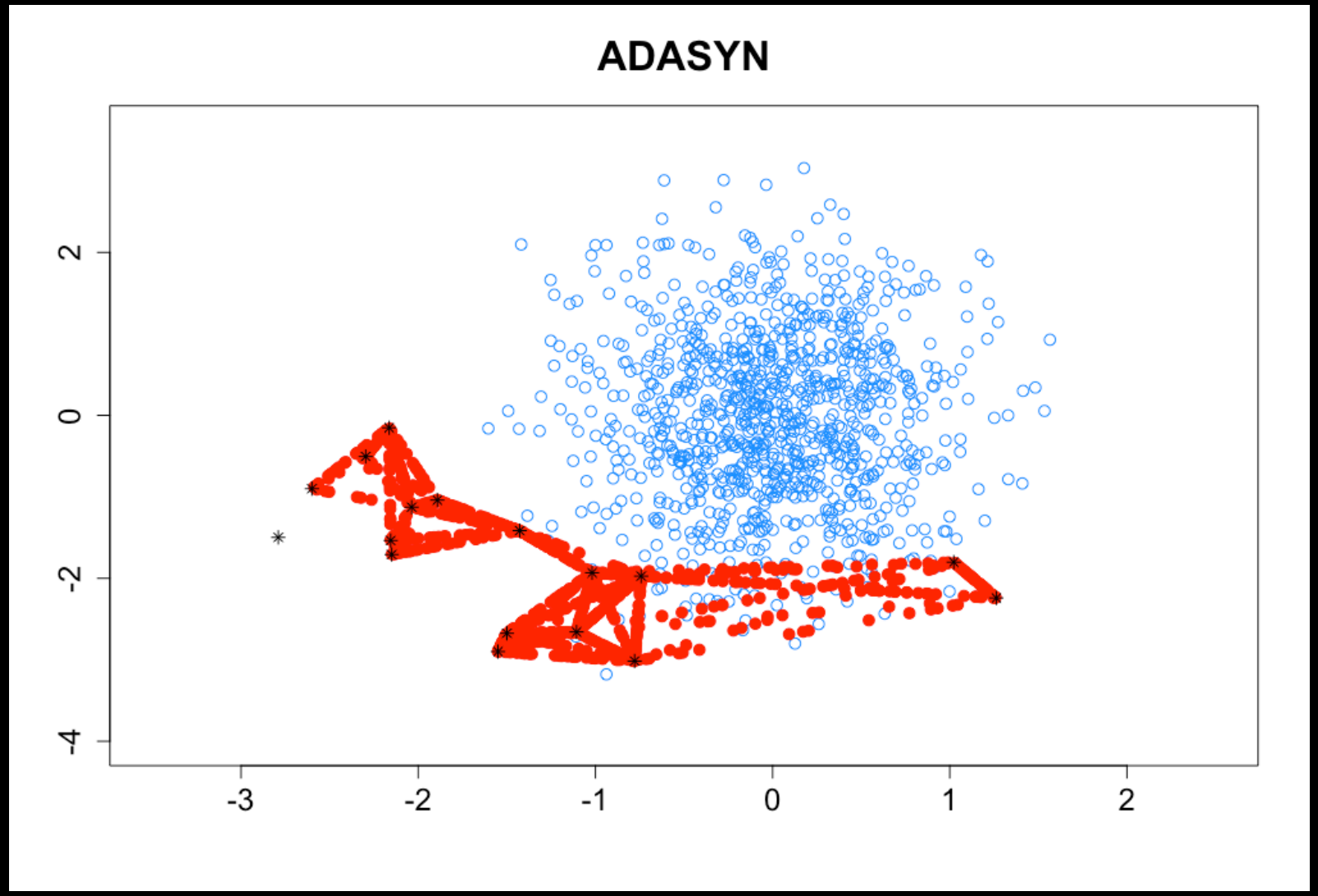
SMOTE



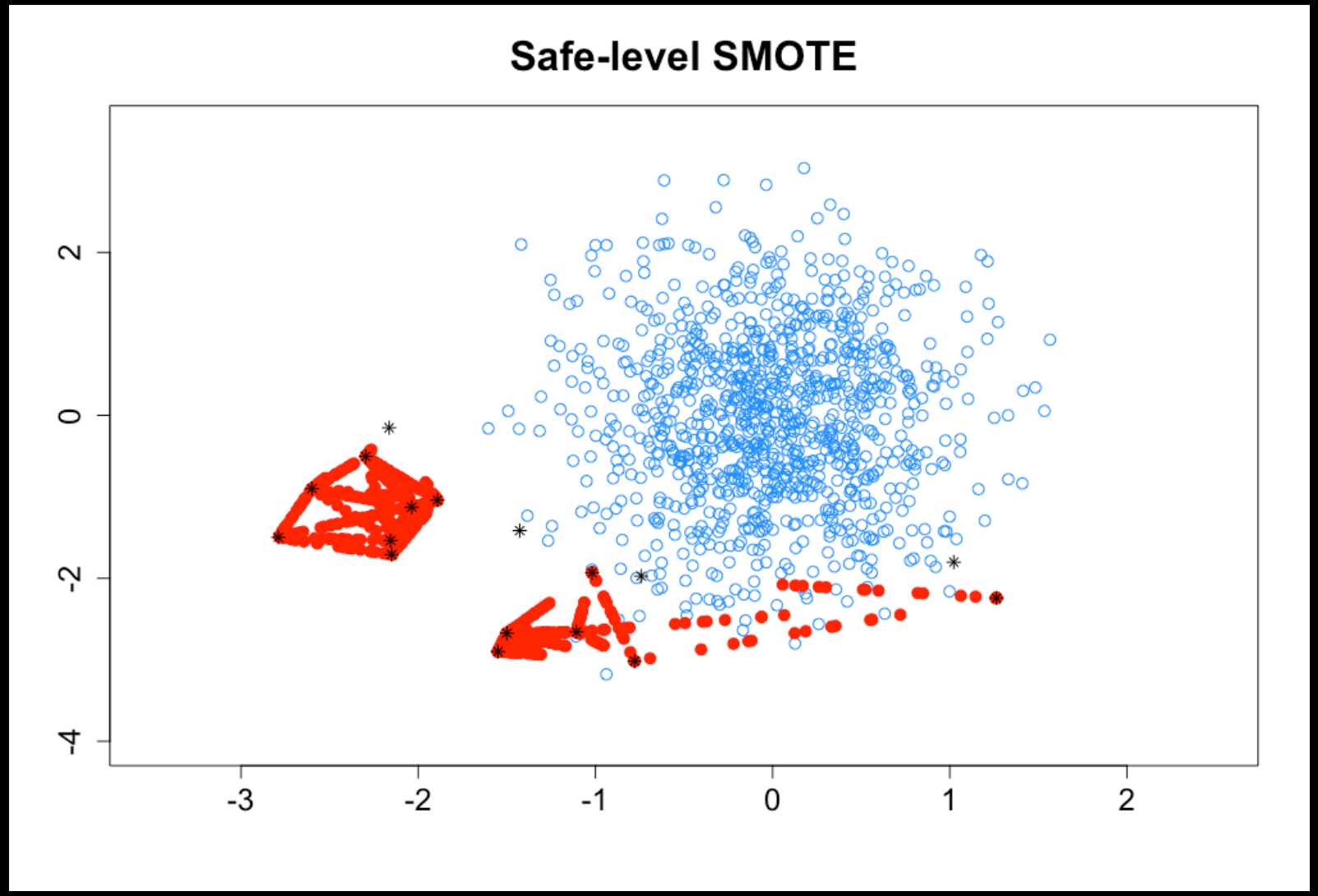
Borderline-SMOTE



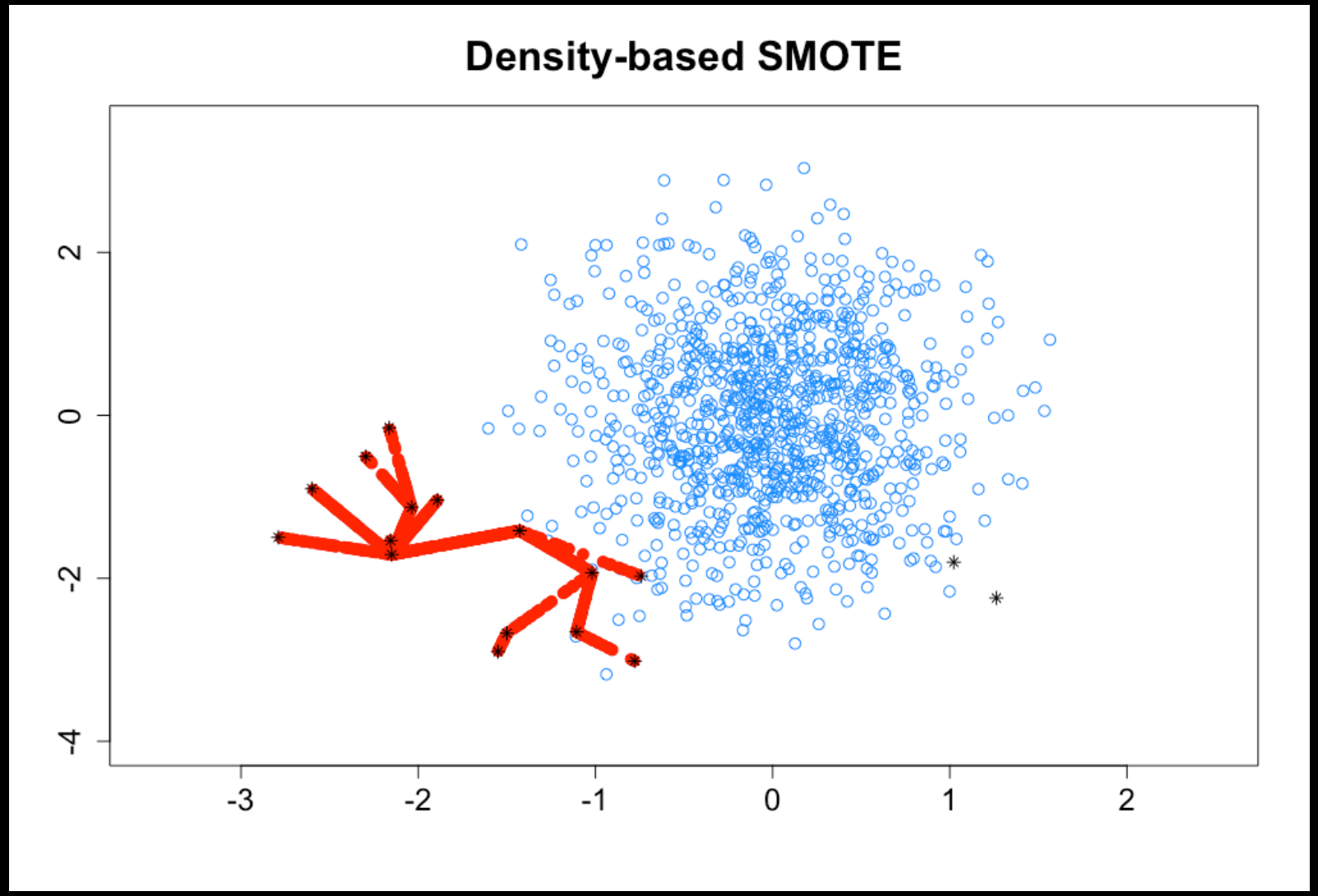
ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning



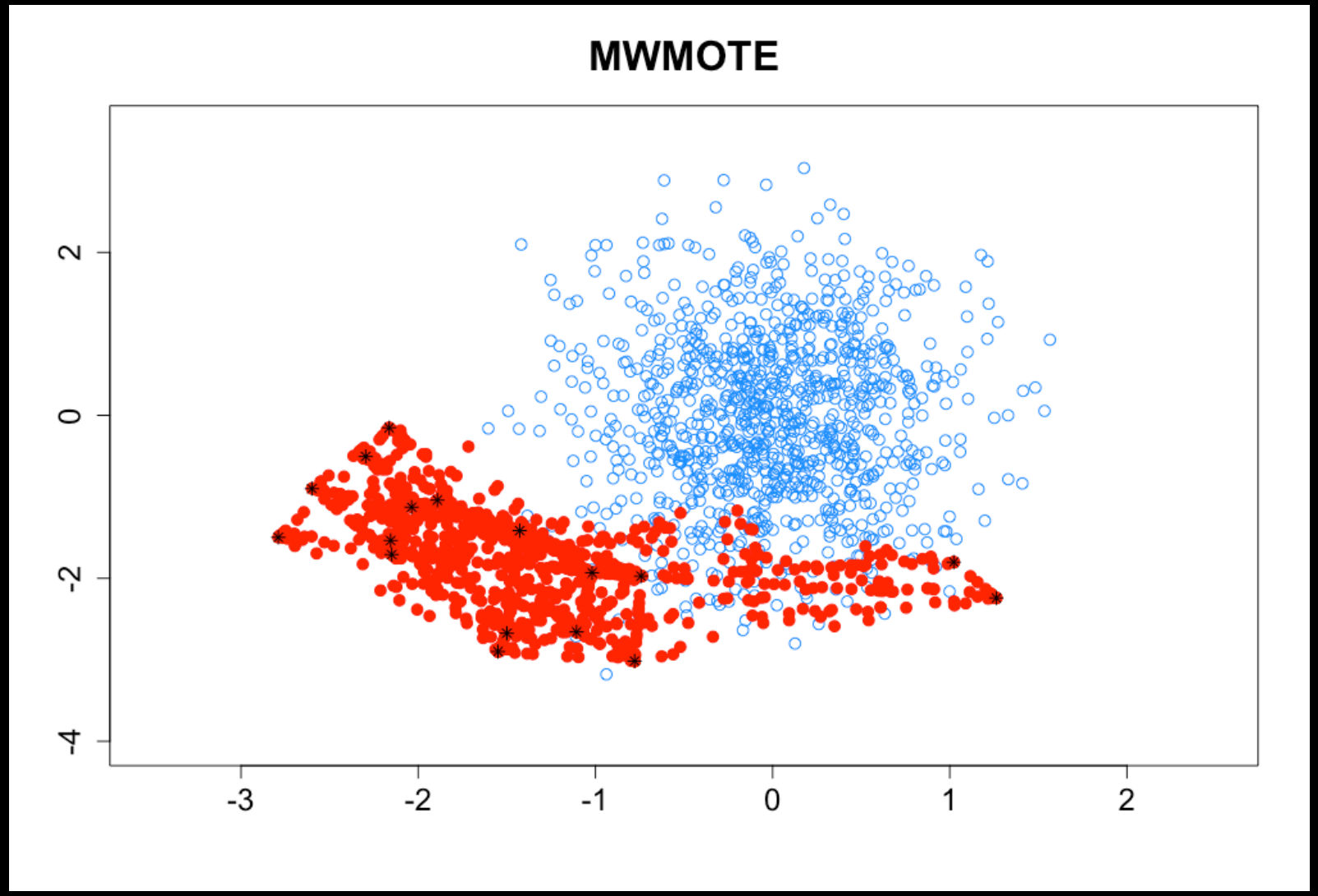
Safe-level SMOTE



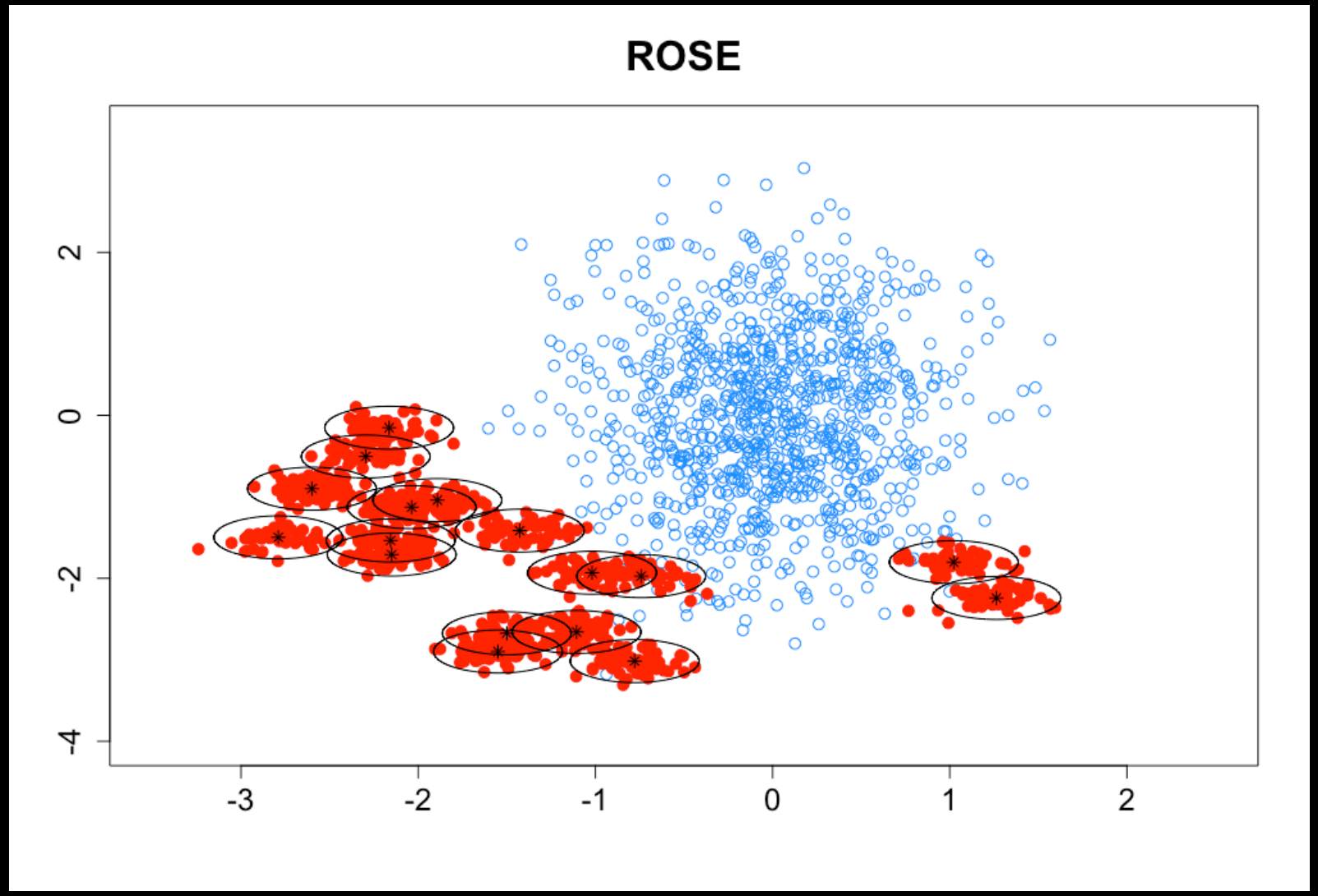
Density-based SMOTE



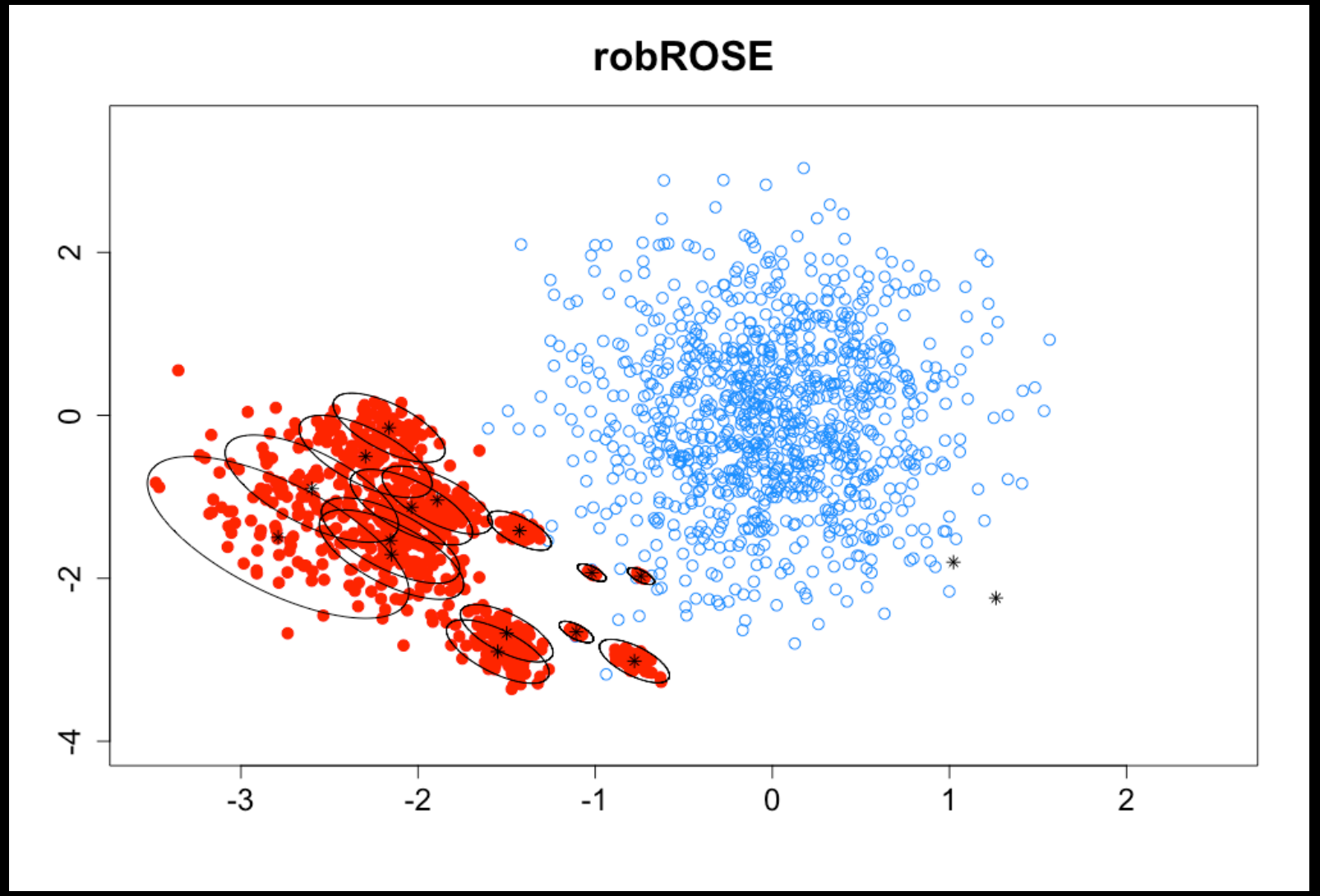
MWMOTE: Majority weighted minority oversampling technique for imbalance dataset learning

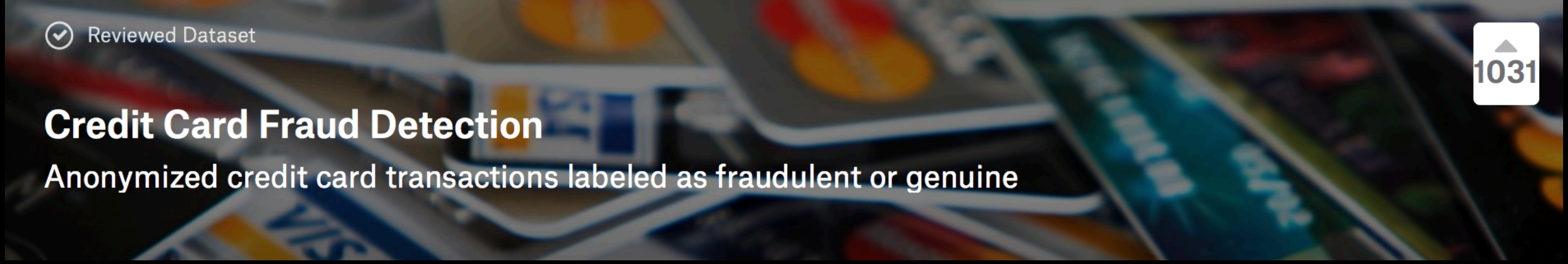


ROSE: Randomly Over-Sampling Examples



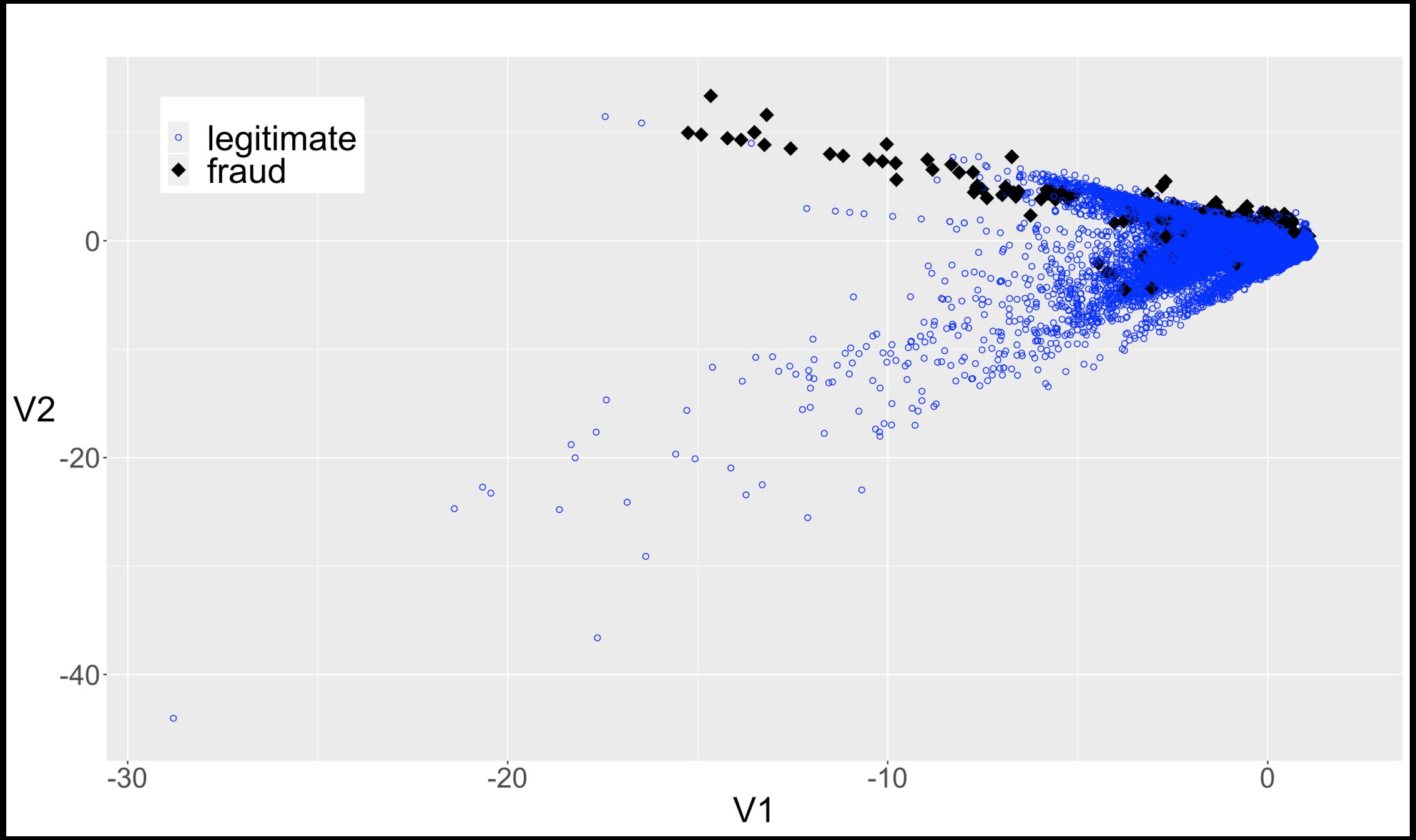
Robust ROSE





source: kaggle.com, made available by Andrea Dal Pozzolo et al., Calibrating Probability with Undersampling for Unbalanced Classification. In Symposium on Computational Intelligence and Data Mining (CIDM), IEEE, 2015

- Transactions made in two days by credit cards in September 2013 by European cardholders
- 497 frauds out of 284,807 transactions ⇒ 0.172% is fraud


legitimate
fraud
0
-20
-40
V2
-30
-20
-10
0
V1

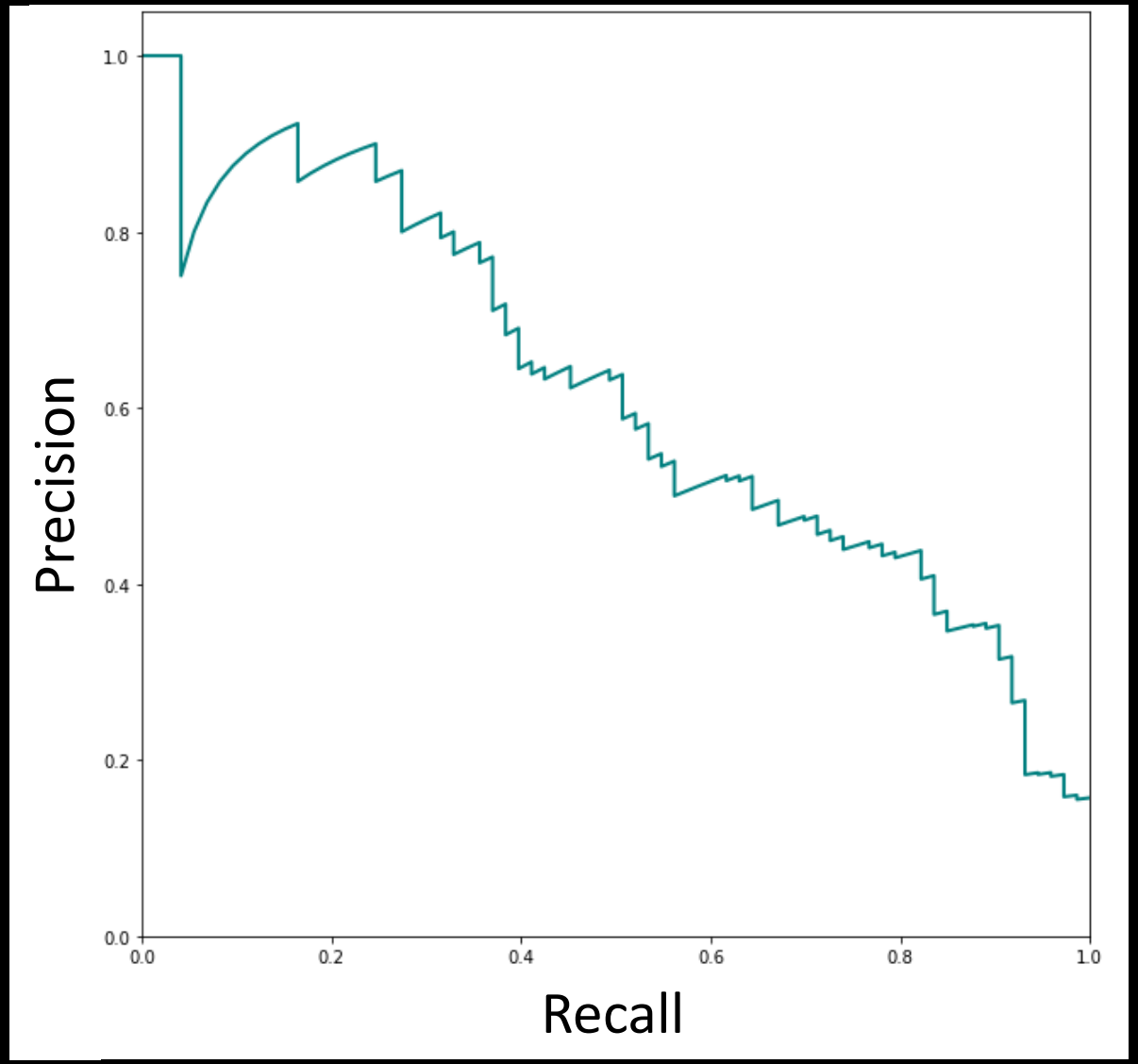
- Precision = $\frac{TP}{TP+FP}$

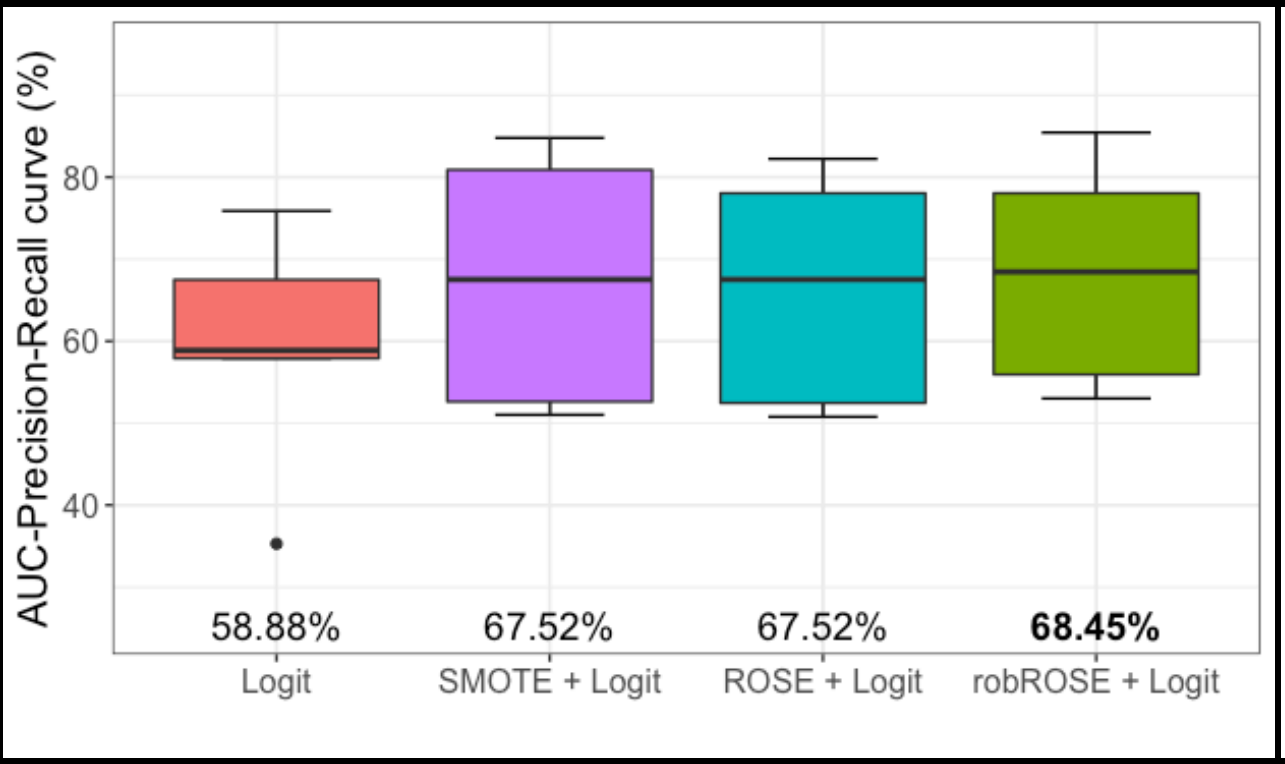
  Out of all cases classified as fraud, how many are actually fraud?

- Recall = $\frac{TP}{TP+FN}$

  Out of all fraud cases, how many are detected?

- Evaluation measure:
  area under precision-recall curve




AUC-Precision-Recall curve (%)
80
60
40
58.88%
67.52%
67.52%
68.45%
Logit
SMOTE + Logit
ROSE + Logit
robROSE + Logit

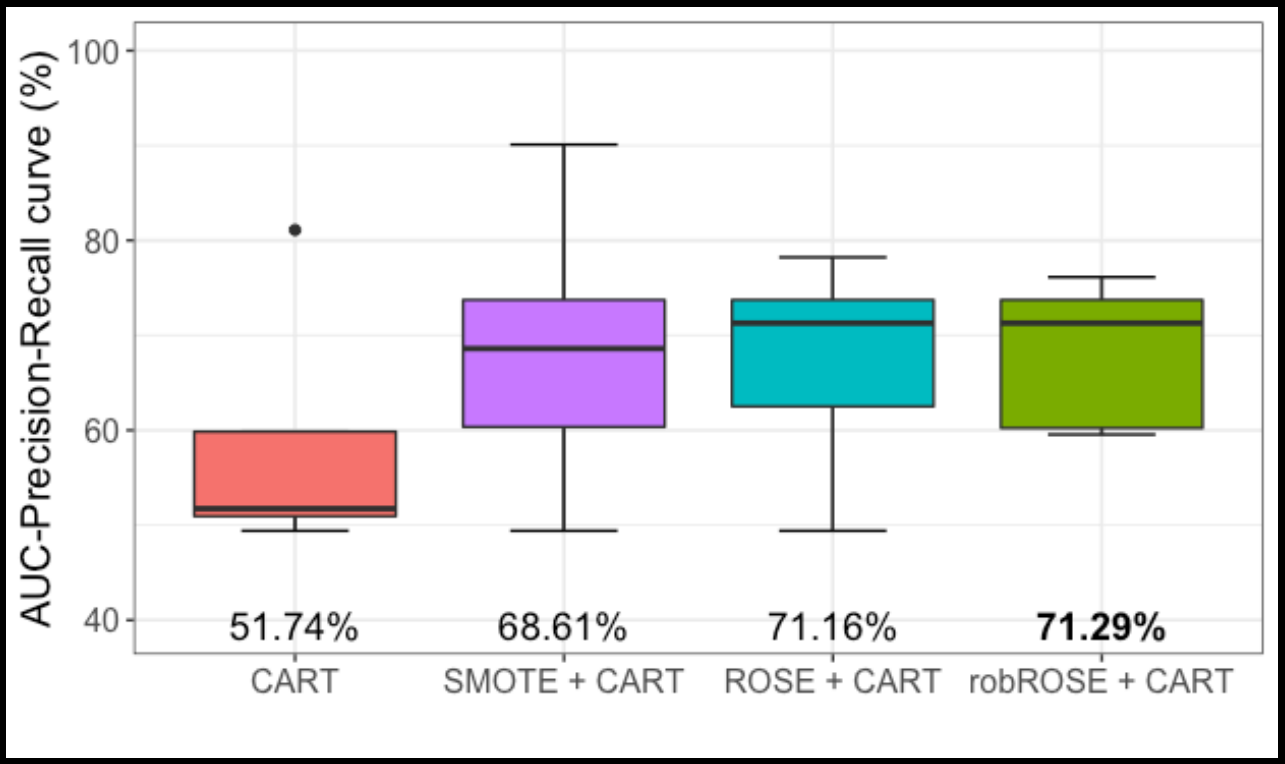
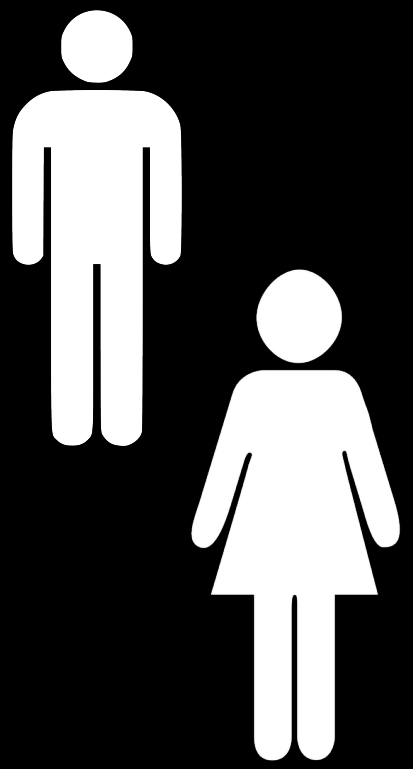

AUC-Precision-Recall curve (%)
100
80
60
40
51.74%
68.61%
71.16%
71.29%
CART
SMOTE + CART
ROSE + CART
robROSE + CART

Table of Contents

- Introduction
- Obtain fraud score using machine learning
- Dealing with imbalanced data
- Evaluate performance taking costs into account
- Design cost-sensitive fraud detection model
- Anomaly detection



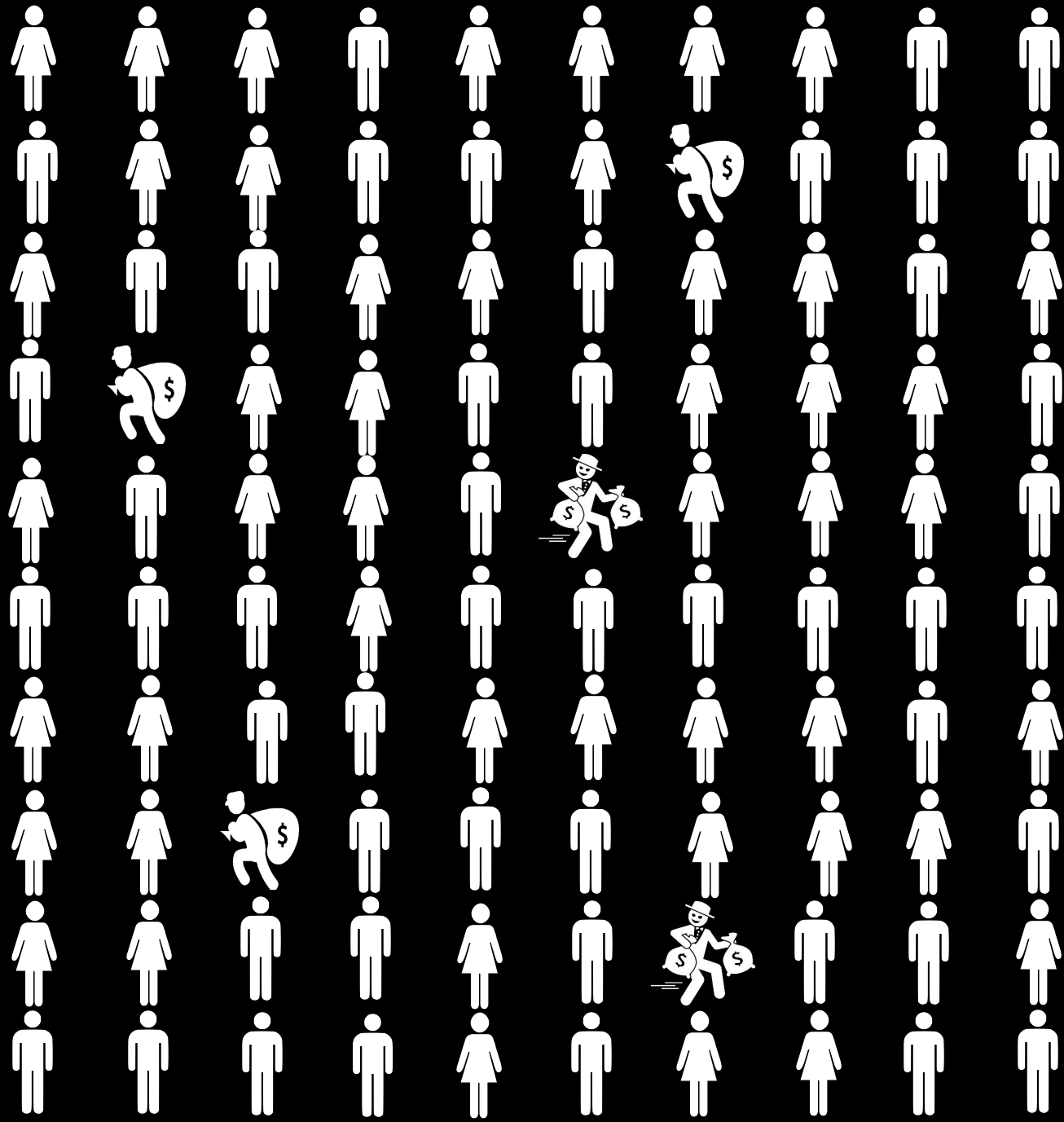


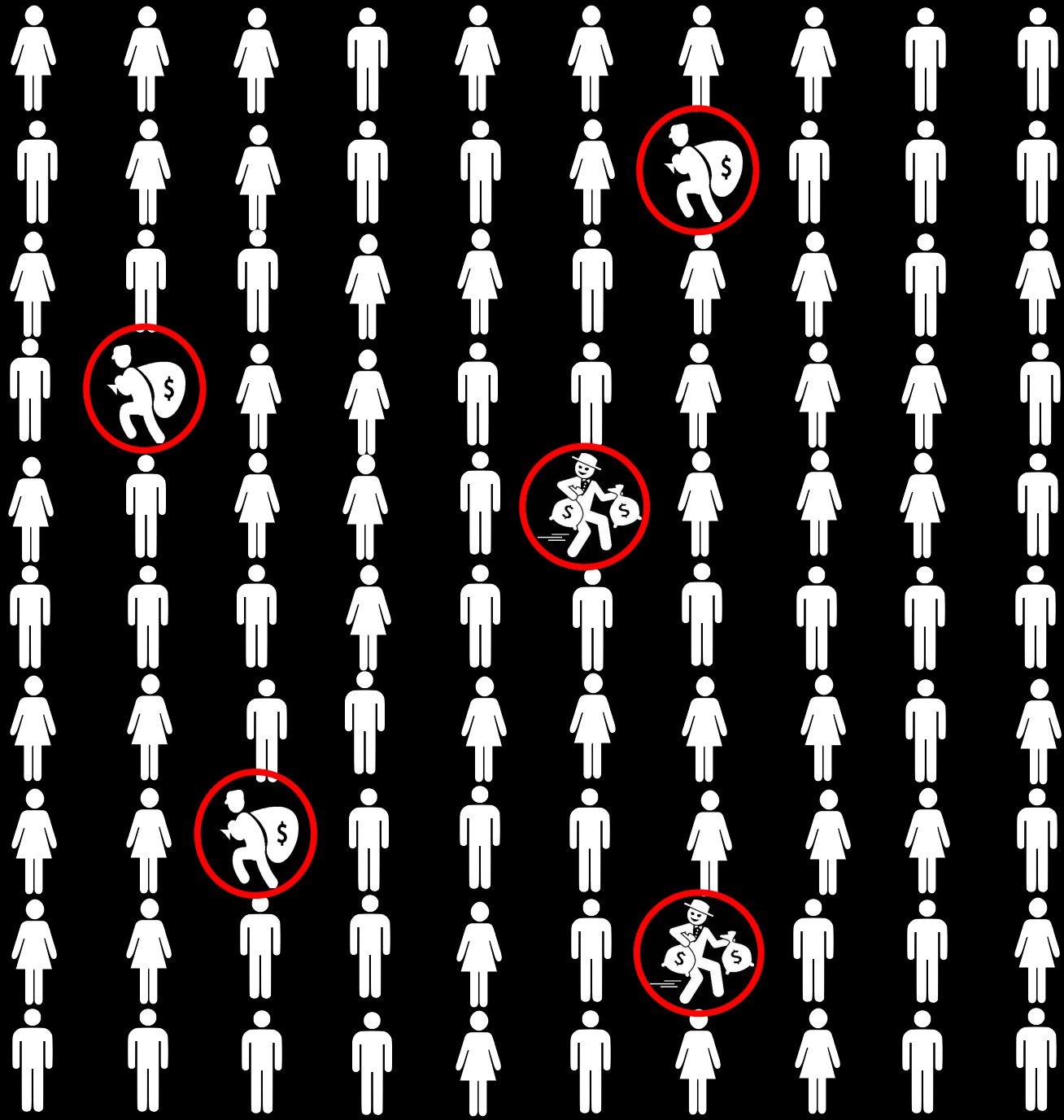
LEGITIMATE

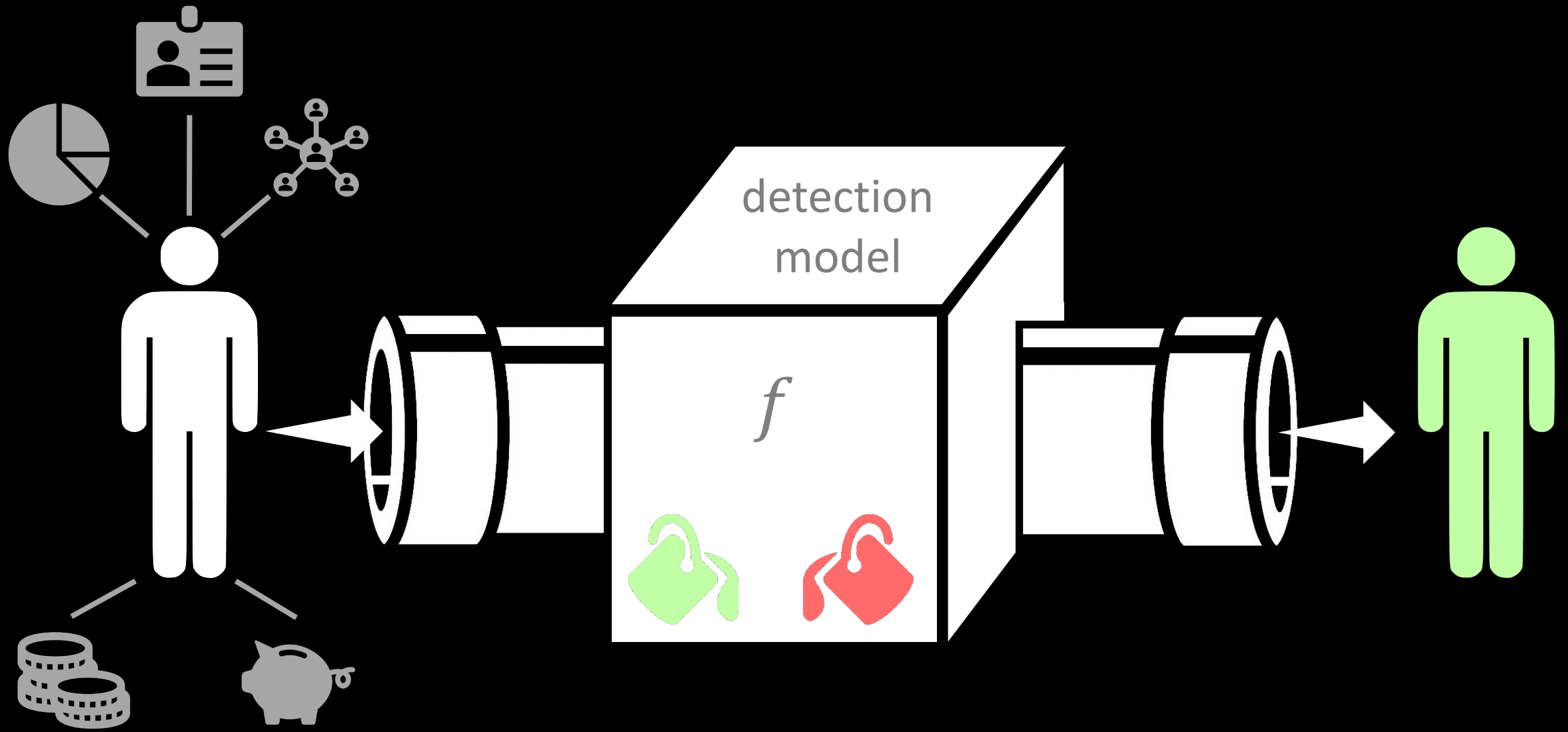
"negative" ($y = 0$)

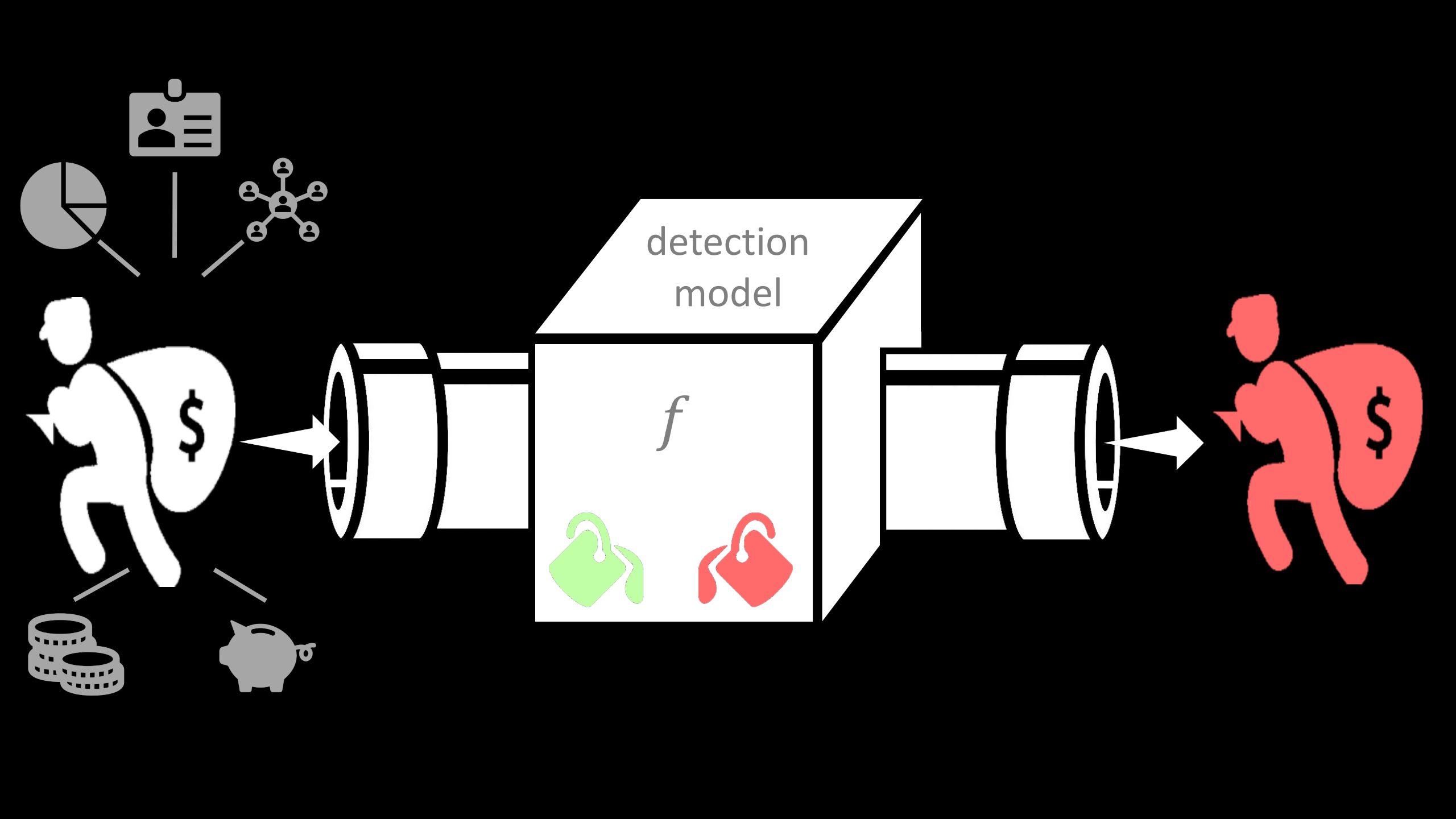
FRAUD

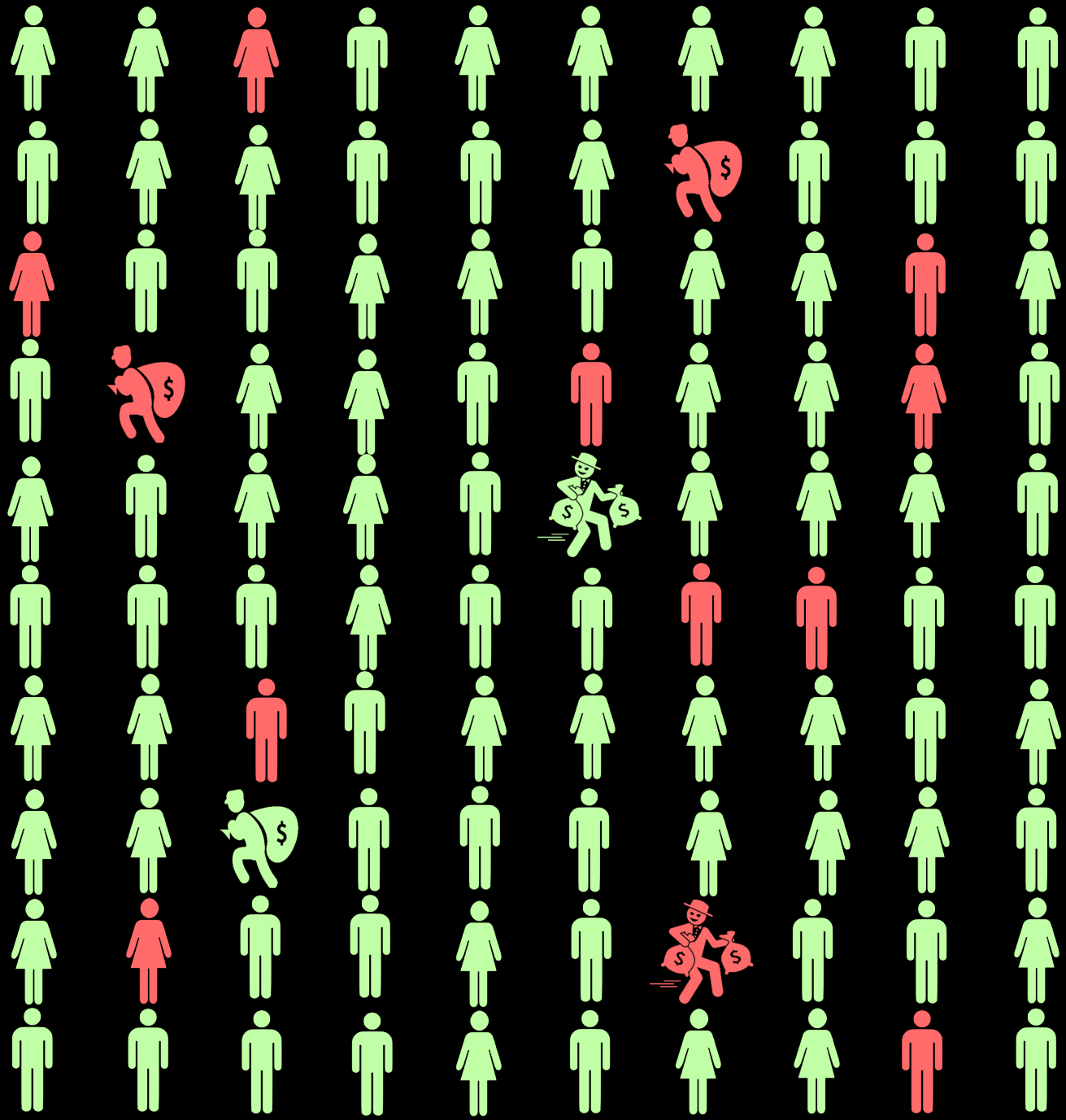
"positive" ($y = 1$)

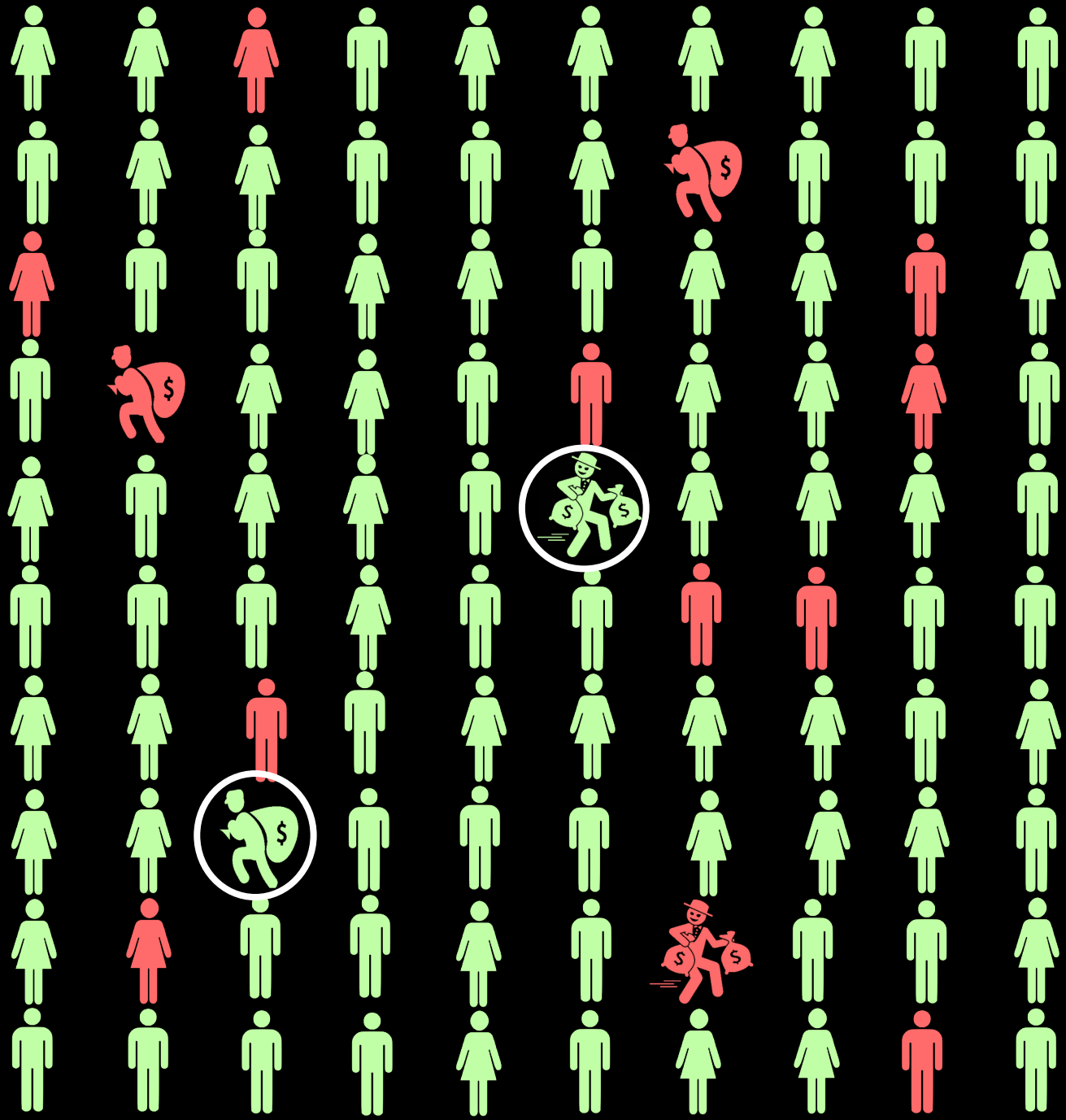


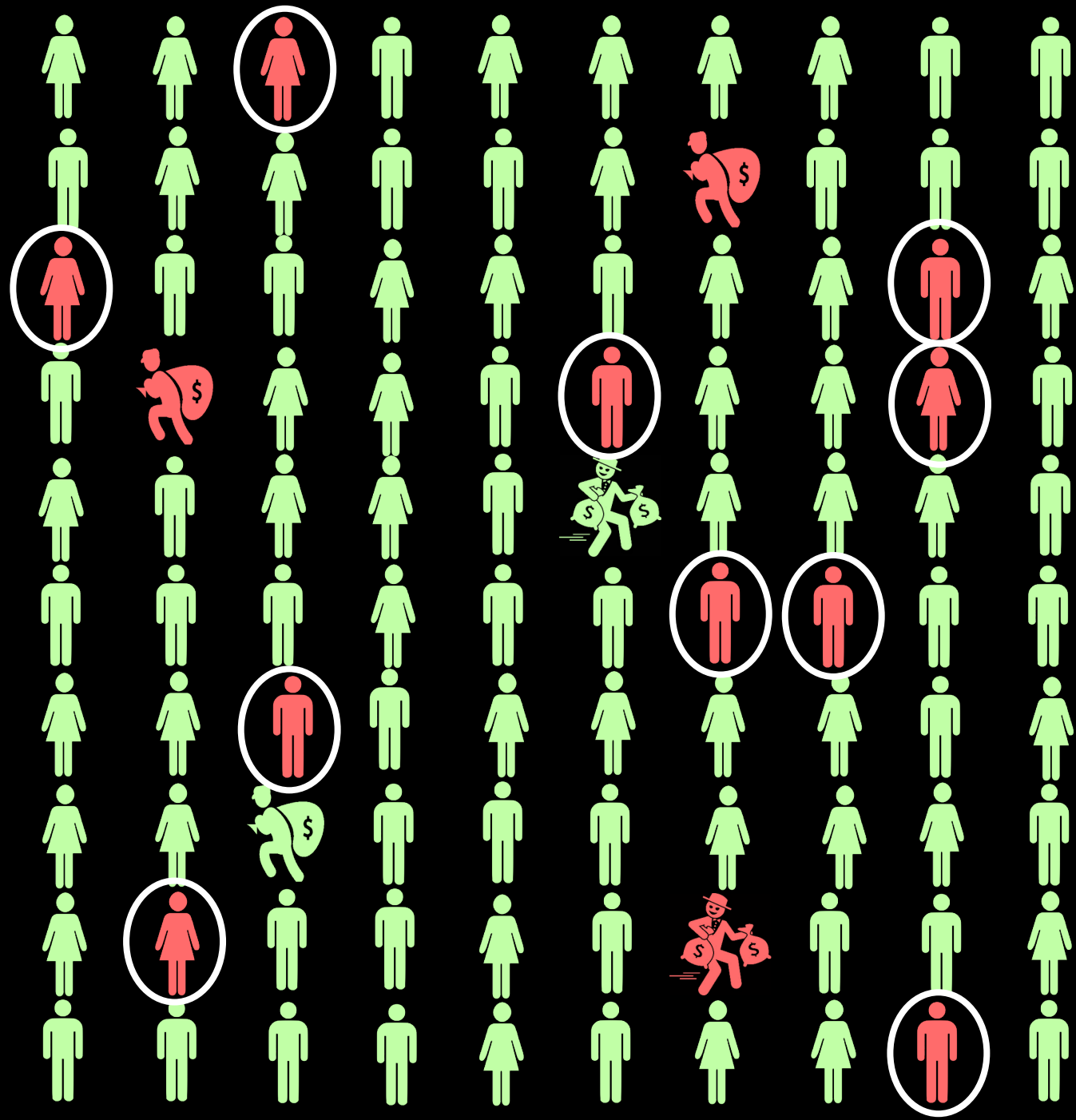


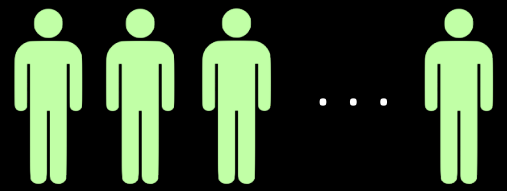
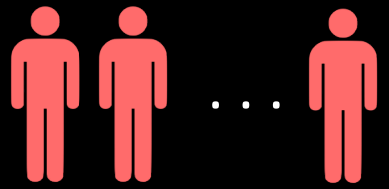

detection
model
f

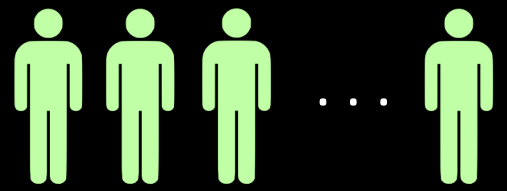
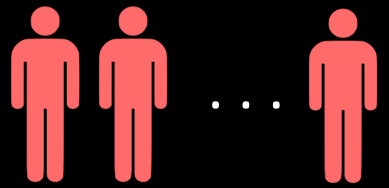

detection
model
f







	Actual negative (legit) $y = 0$	Actual positive (fraud) $y = 1$
Predicted negative (legit) $c = 0$	 85 true negatives (TN)	 2 false negatives (FN)
Predicted positive (fraud) $c = 1$	 10 false positives (FP)	 3 true positives (TP)

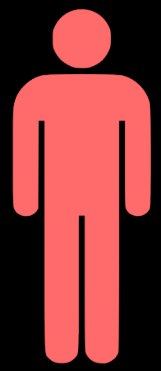
	Actual negative (legit) $y = 0$	Actual positive (fraud) $y = 1$
Predicted negative (legit) $c = 0$	 85 true negatives (TN)	 2 false negatives (FN)
Predicted positive (fraud) $c = 1$	 10 false positives (FP)	 3 true positives (TP)

$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN} = \frac{85 + 3}{100} = 88\%$$

$$Precision = \frac{TP}{TP + FP} = \frac{3}{13} = 23\%$$

$$Recall = \frac{TP}{TP + FN} = \frac{3}{5} = 60\%$$

Cost-sensitive approach



- €



- €€



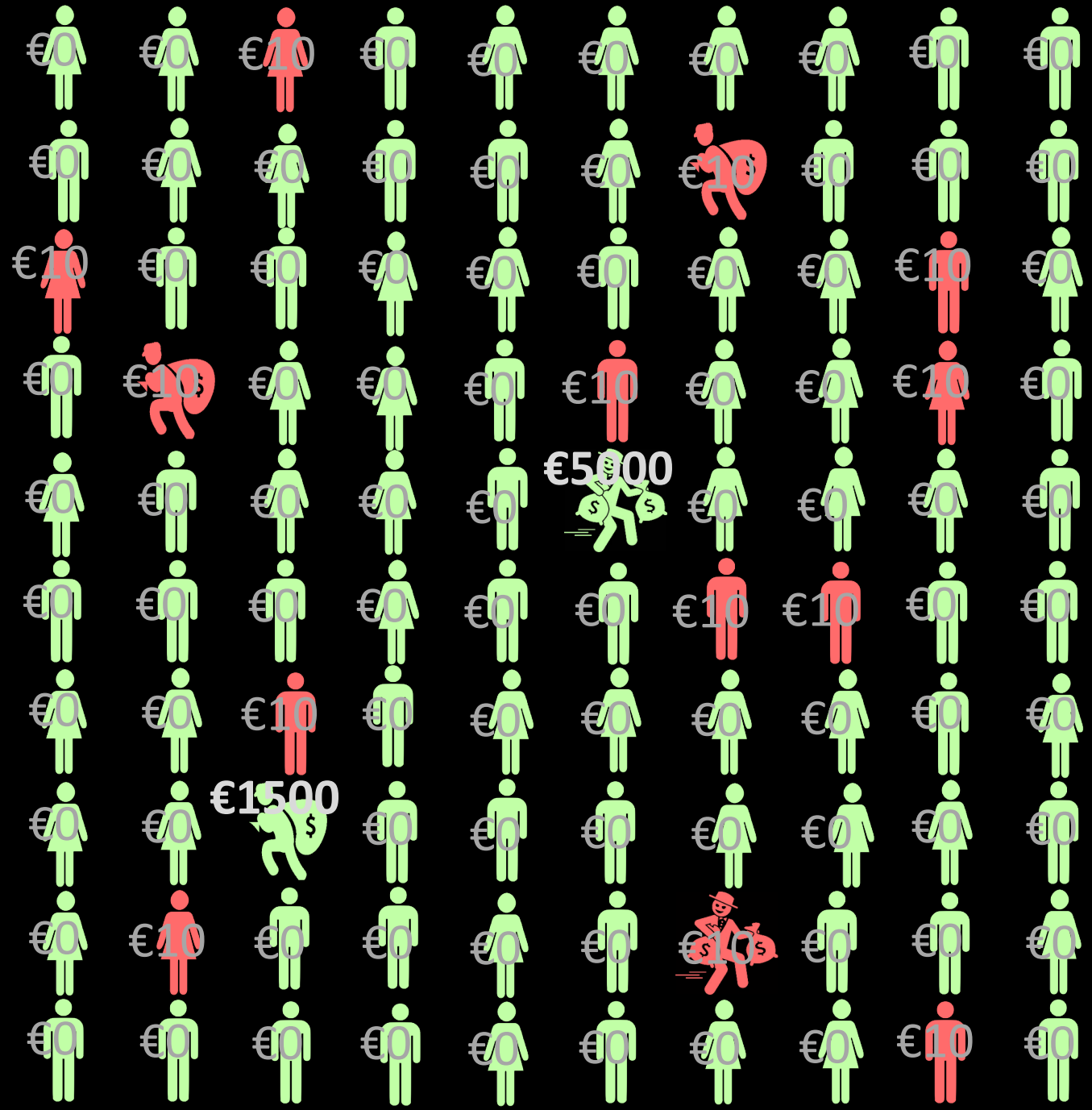
- €€€

Different ***costs*** associated with different misclassification errors!

Cost-sensitive approach

	Actual negative (legit) $y = 0$	Actual positive (fraud) $y = 1$
Predicted negative (legit) $c = 0$	 $C_{TN} = 0$	 $C_{FN} = Amount\ (A)$
Predicted positive (fraud) $c = 1$	 $C_{FP} = C_a$	 $C_{TP} = C_a$

C_a = fixed administrative cost for investigating transaction and contacting customer



$C_a = €10$

Total cost = €6630

Cost of fraud detection model

- $S = set\ of\ N\ transactions$
- $f = model\ trained\ on\ set\ S$
- $f\ generates\ predicted\ label\ c_i\ for\ each\ transaction\ i\ (i = 1, \dots, N)$

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

$$= \sum_{i=1}^{N} y_i(1 - c_i)A_i + c_i C_a$$

Correa Bahnsen et al. (2014)

$f_0 = model\ that\ predicts\ all\ transactions\ in\ S\ as\ legit\ (i.e.class\ 0)$

$Cost\big(f_0(S)\big) = cost\ of\ using\ no\ detection\ model$

$$= total\ amount\ of\ stolen\ money = \sum_{i=1}^{N} y_i A_i$$

$Savings\big(f(S)\big) =$ $cost\ savings\ as\ compared\ with\ Cost\big(f_0(S)\big)$

$$Savings\big(f(S)\big) = \frac{Cost\big(f_0(S)\big) - Cost\big(f(S)\big)}{Cost\big(f_0(S)\big)} = \frac{\sum_{i=1}^{N} y_i c_i A_i - c_i C_a}{\sum_{i=1}^{N} y_i A_i}$$

Table of Contents

- Introduction
- Obtain fraud score using machine learning
- Dealing with imbalanced data
- Evaluate performance taking costs into account
- Design cost-sensitive fraud detection model
- Anomaly detection

Fraud score

Consider following information about a transaction:

- amount (x_1)
- timestamp (x_2)
- info on originator (x_3)
- info on beneficiary (x_4)

} data on the transfer $\boldsymbol{x} = (x_1, x_2, x_3, x_4)$

Fraud score

Consider following information about a transaction:

- amount (x_1)
- timestamp (x_2)
- info on originator (x_3)
- info on beneficiary (x_4)

} data on the transfer $\boldsymbol{x} = (x_1, x_2, x_3, x_4)$

Given this data on the transaction, then
the ***probability*** that the transaction is fraudulent ($y = 1$) equals

$$s = fraud\ score$$
$$= P[\, y = 1 \mid x_1, x_2, x_3, x_4 \,]$$

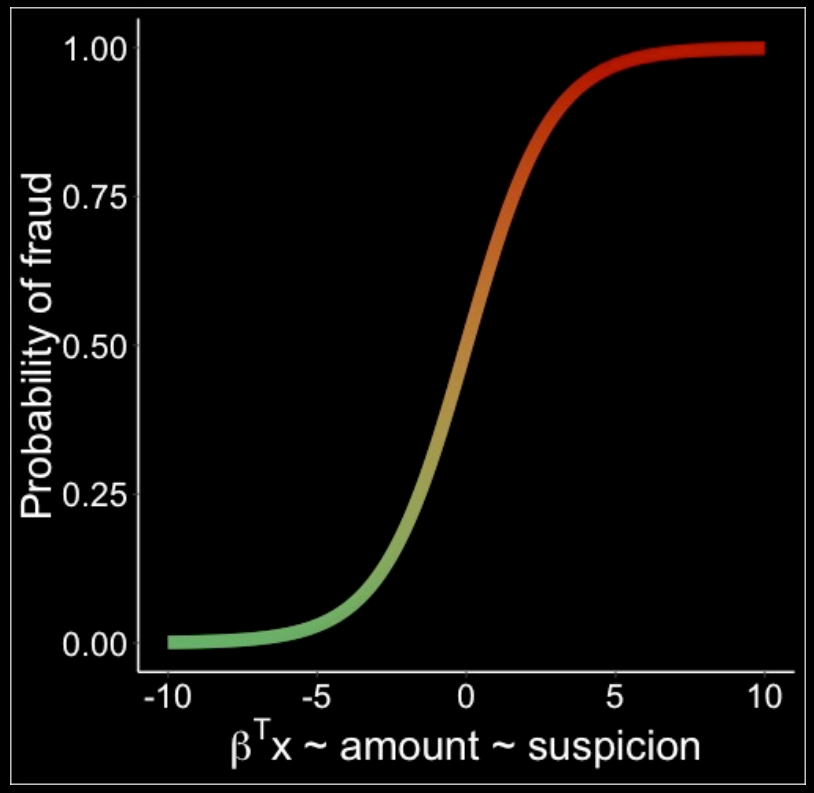
Model fraud score using logistic regression (or logit model)

$$s = P[\, y = 1 \mid x_1, x_2, x_3, x_4 \,]$$

$$= sigmoid(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4)$$

$$= sigmoid(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x})$$

$$= \frac{1}{1 + e^{-(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x})}}$$



Average expected cost

Train model on a dataset S = ***find optimal parameters*** $(\beta_0, \boldsymbol{\beta})$ w.r.t. set S by minimizing objective function:

Average expected cost

Train model on a dataset S = ***find optimal parameters*** $(\beta_0, \boldsymbol{\beta})$ w.r.t. set S by minimizing objective function:

$$minimize\ Average\ Expected\ Cost\ (AEC) + ``complexity''(lasso\ penalty)$$

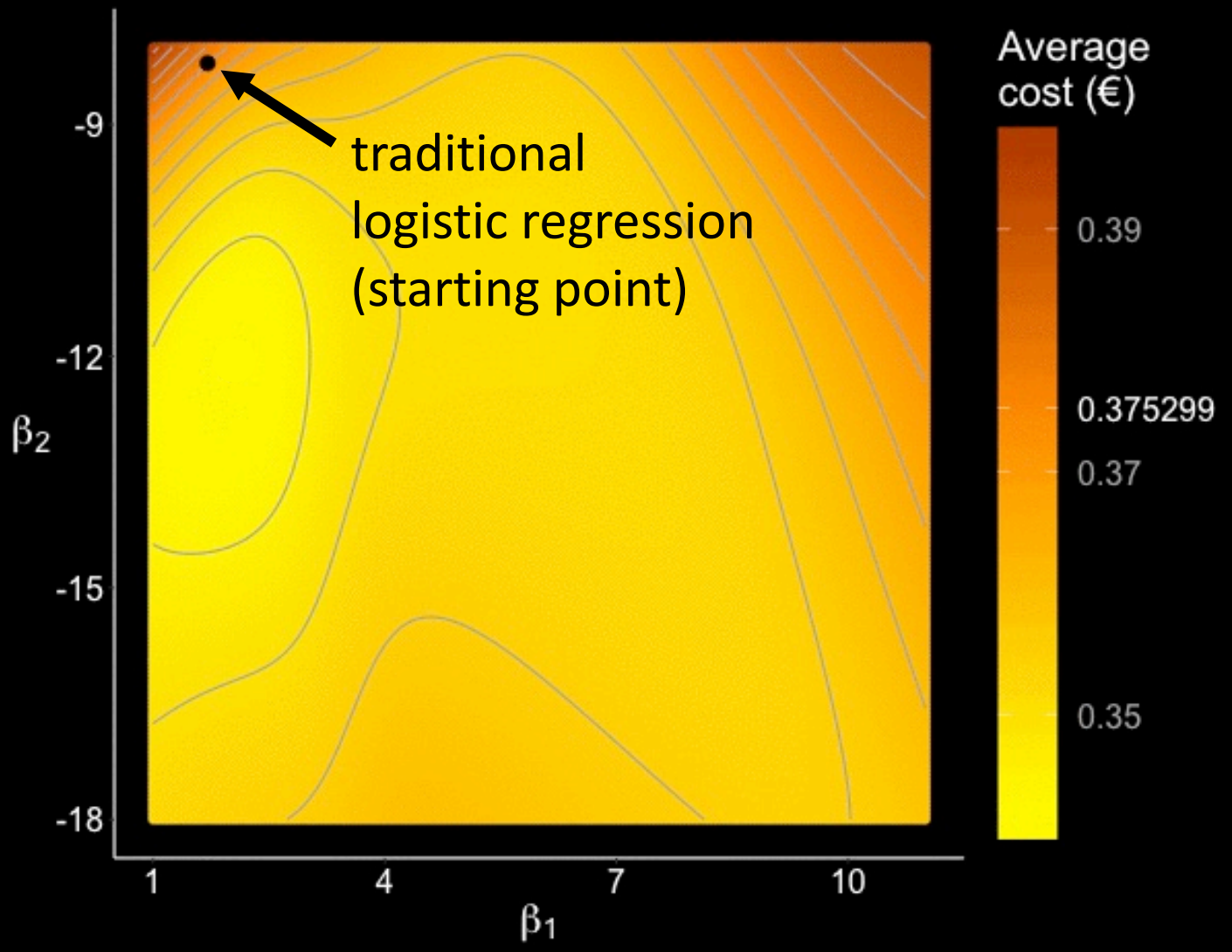
$$\Leftrightarrow minimize\ \ AEC(\beta_0, \boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_1$$

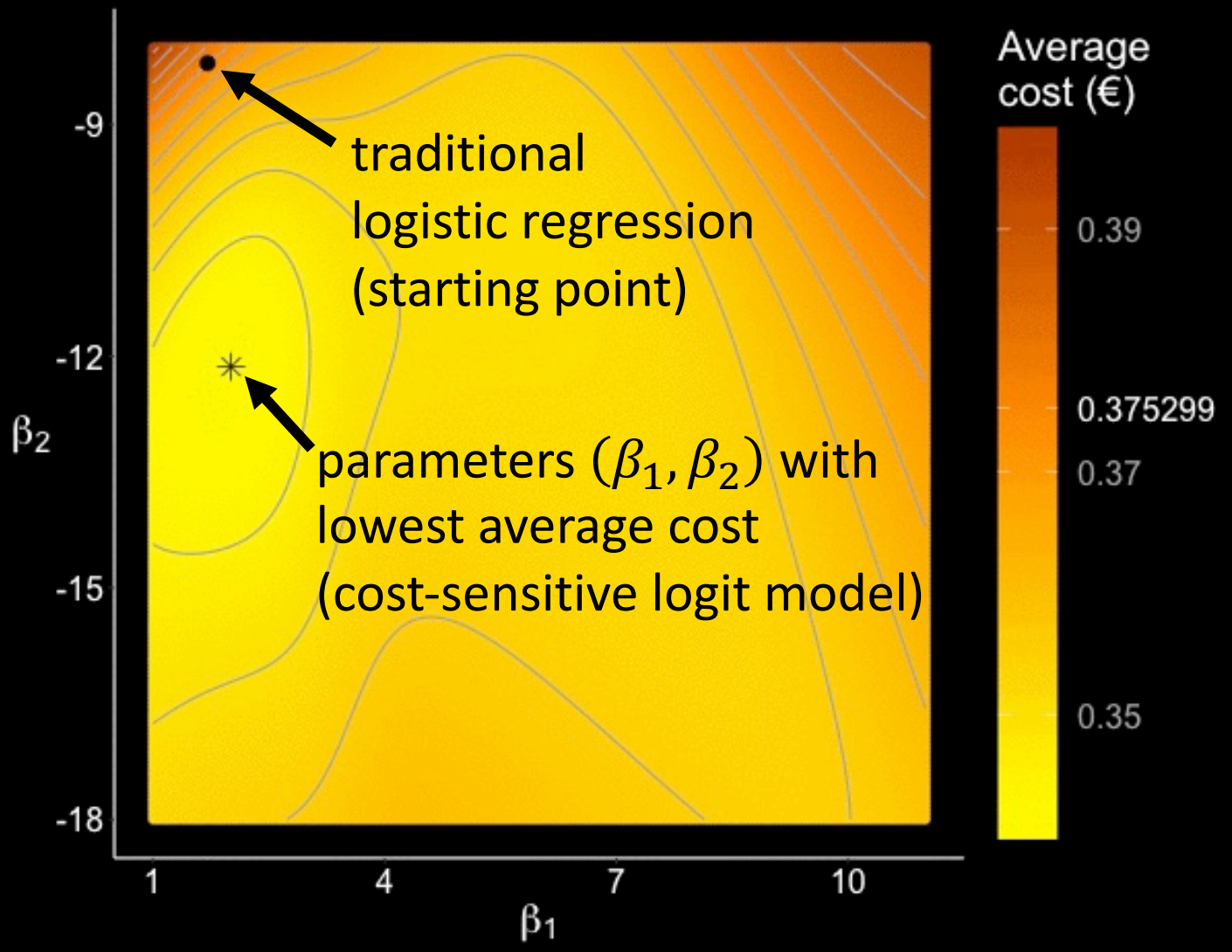
$$\Leftrightarrow\ minimize\ \frac{1}{N}\sum_{i=1}^{N}\left[y_i\left(1 - s_i(\beta_0, \boldsymbol{\beta})\right)A_i + s_i(\beta_0, \boldsymbol{\beta})C_a\right] + \lambda\sum_{j=1}^{d}|\beta_j|$$

Finding the coefficients

- Objective function is minimized using a gradient-based optimization method.
- Components of gradient are given by following partial derivatives for $j = 0,1, \ldots, d$:

$$\frac{1}{N}\sum_{i=1}^{N}\left[-y_i A_i + c_f\right] s_i(\beta_0, \boldsymbol{\beta})\left[1 - s_i(\beta_0, \boldsymbol{\beta})\right] x_i + \lambda \frac{\partial \|\beta\|_1}{\partial \beta_j}$$


traditional
logistic regression
(starting point)
-9
-12
-15
-18
β2
1
4
7
10
β1
Average
cost (€)
0.39
0.375299
0.37
0.35


traditional
logistic regression
(starting point)
parameters (β_1, β_2) with
lowest average cost
(cost-sensitive logit model)
-9
-12
-15
-18
β_2
1
4
7
10
β_1
Average
cost (€)
0.39
0.375299
0.37
0.35

transactions = 1 068 781
fraud cases = 631 (0.06%)

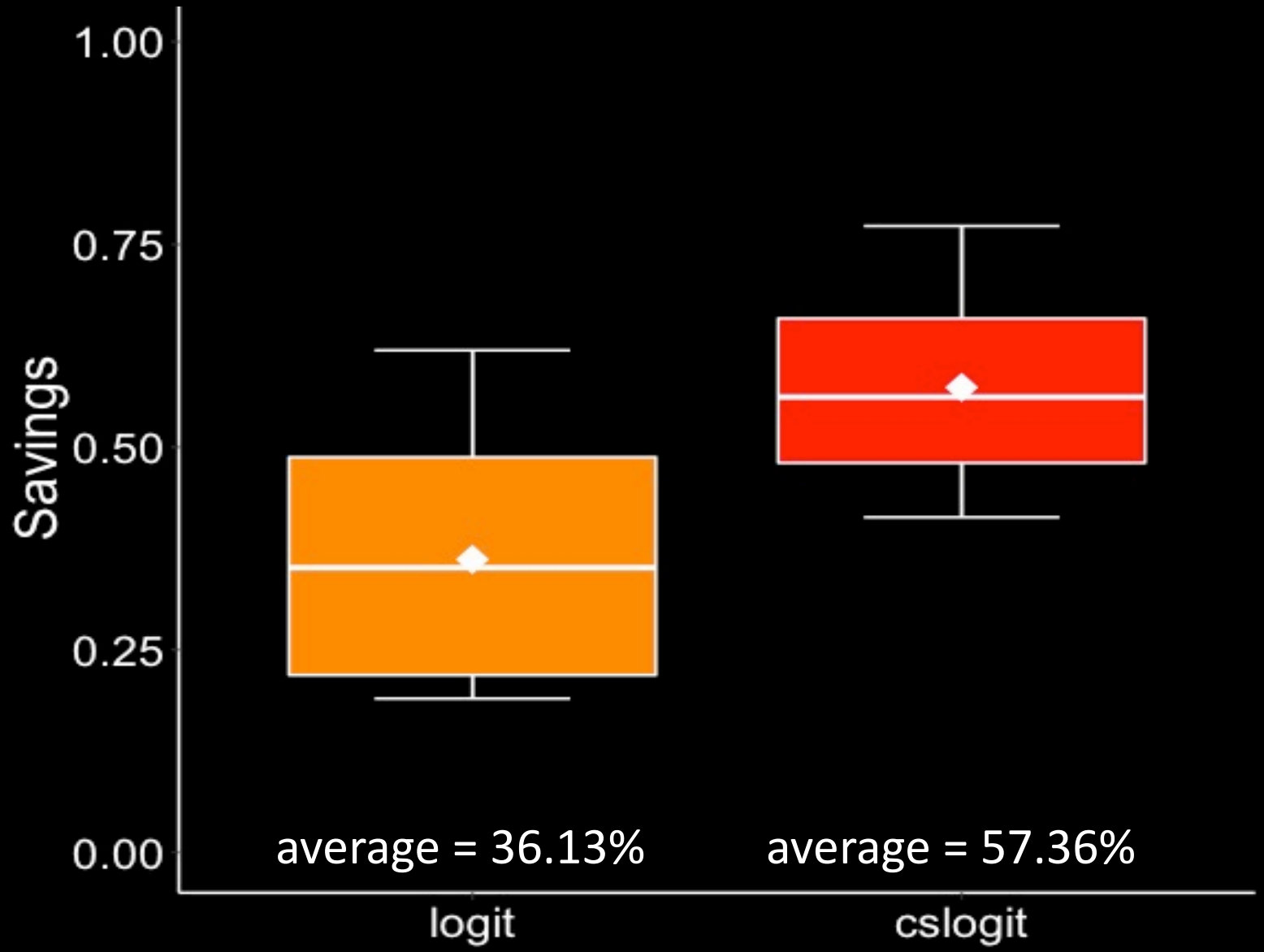
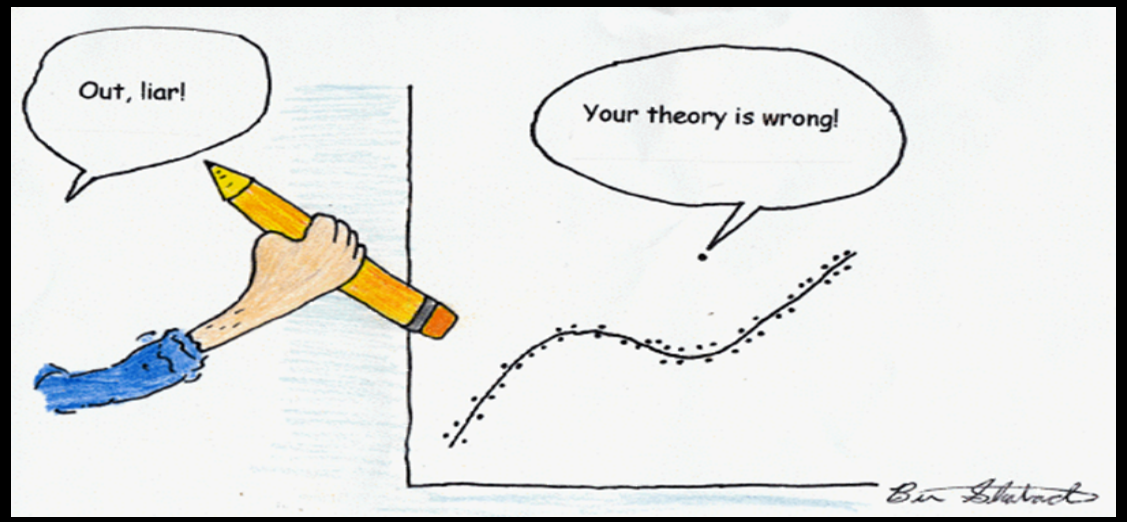


Table of Contents

- Introduction
- Obtain fraud score using machine learning
- Dealing with imbalanced data
- Evaluate performance taking costs into account
- Design cost-sensitive fraud detection model
- Anomaly detection

Anomaly or outlier

- Fraudsters may be detected by fact that their behavior is deviant (slightly and in a complex way) from nonfraudsters,
- Outlier or anomaly detection is very important tool for fraud detection.



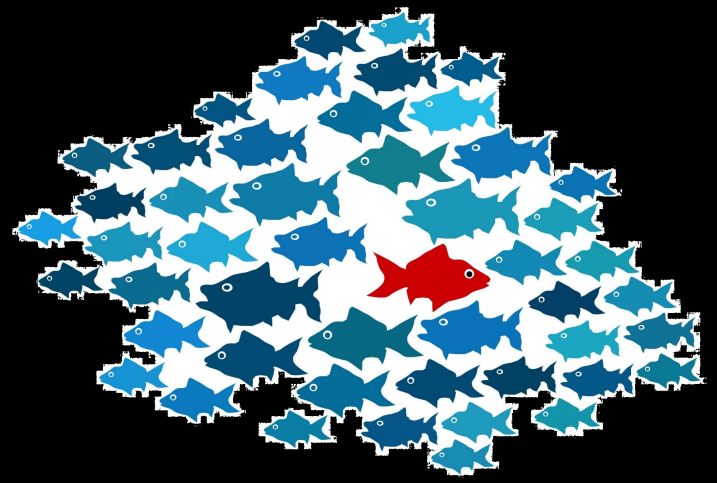
- Outliers do not necessarily represent fraudulent observations, so **follow-up and validation is needed**.

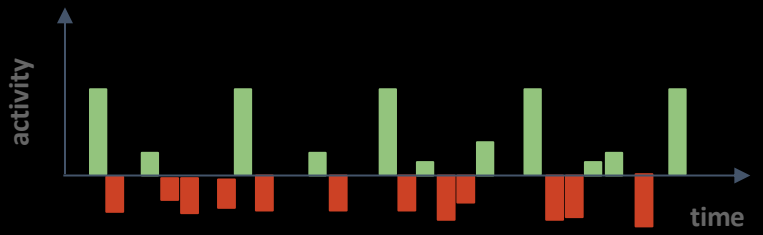
Anomaly detection

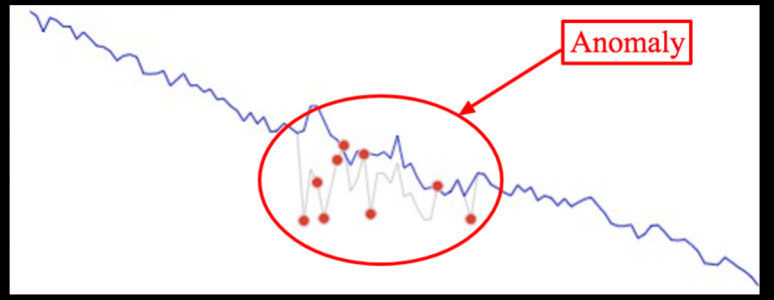
Preferably trained without labels

- Describe ‘normal’ behaviour.
- Find suspicious observations

- Detect new types of fraud!







Outlier Detection Methods: z-score

- z-score measures how many standard deviations an observation lies away from the mean for a specific variable as follows:

$$z_i = \frac{x_i - \hat{\mu}}{\hat{\sigma}}$$

- μ is mean of variable x_i and σ standard deviation.
- Typically one uses classical mean $\bar{x}$ and standard deviation s.
- Outliers are defined when $|z_i| > 3$.
- Toy example: X = natural logarithms of annual incomes.
- Income of 13th person (Jim) is much larger than other values.

Name	X
John	10.78
Phil	10.09
…	…
Marie	9.79
Jim	15.21

Toy example: data characteristics

	12 regular observations	All 13 observations
Mean	10.1088	10.3697
Median	10.0858	10.0858
Standard deviation	0.4836	1.0483
IQR	0.3787	0.3787
Skewness	0.2006	2.1595
Kurtosis	-0.4859	6.1047

$$z_{13} = 2.9 \text{ versus } z_{13}^{*} = \frac{x_{13} - Med(x)}{IQR(x)} = 9.0 > 3$$

- Classical estimators are highly influenced by the outlier.
- Robust estimators are less influenced by the outlier.
- Robust estimate computed from all observations is comparable with the classical estimate applied to the non-outlying data.

Detecting outliers using boxplot

Boxplot is popular visual aid for detecting outliers.

- A box is drawn from first to third quartile

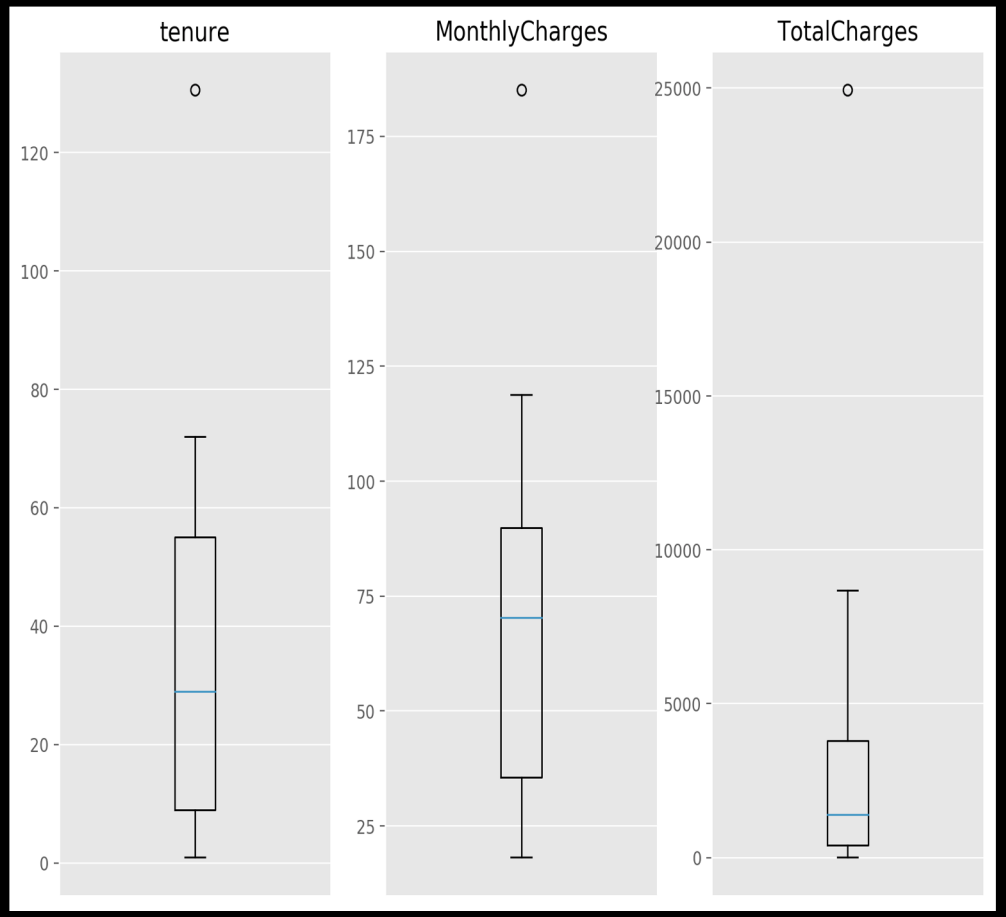
$$P(X \leq Q_1) = 0.25 \text{ and } P(X \leq Q_3) = 0.75$$

- Median or second quartile is added

$$P(X \leq Q_2) = 0.5$$

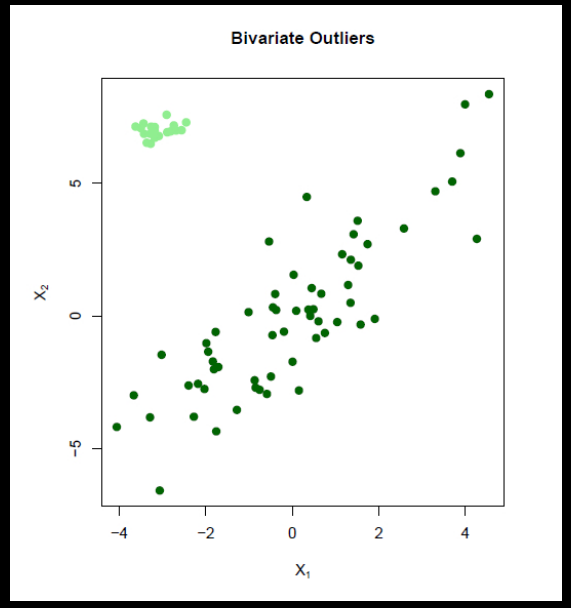
- Whiskers are drawn to minimum and maximum values that are not outlying.
- Outliers are defined as observations that lie more than 1.5 times the interquartile range ($IQR = Q_3 - Q_1$) away from Q_1 or Q_3. Hence points that do not belong to the interval:

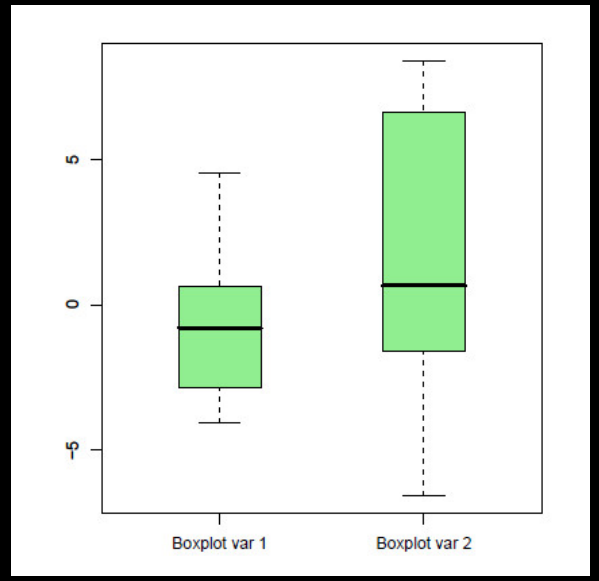
$$[Q_1 - 1.5\ IQR; Q_3 + 1.5\ IQR]$$



Univariate vs. multivariate outlier

- Former methods all focus on univariate outliers
- In the multivariate setting, outliers can NOT always be detected by simply applying outlier detection rules to each variable separately.





- We can only detect such outliers by correctly estimating the **covariance structure**.

Multivariate outlier: detection

- Calculate **Mahalanobis distance** or generalized distance to the mean $\widehat{\boldsymbol{\mu}} = \overline{\boldsymbol{x}}$ taking covariance matrix $\widehat{\boldsymbol{\Sigma}} = \boldsymbol{S_X}$ into account

$$MD(\boldsymbol{x}_i) = \sqrt{(\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}})' \widehat{\boldsymbol{\Sigma}}^{-1} (\boldsymbol{x}_i - \widehat{\boldsymbol{\mu}})}.$$

- Classical 97.5% **tolerance ellipsoid** is given by

$$\left\{ x_i; MD(x_i) \leq \sqrt{\chi^2_{p,0.975}} \right\}.$$

  where $\chi^2_{p,0.975}$ is 0.975 quantile of chi-squared distribution with p df.
- Observation is flagged as outlying if it does not belong to tolerance ellipsoid.
- Automatically detect **outliers** by looking at distance-distance plot or obtained weights.

Example: (Land) Animals data set

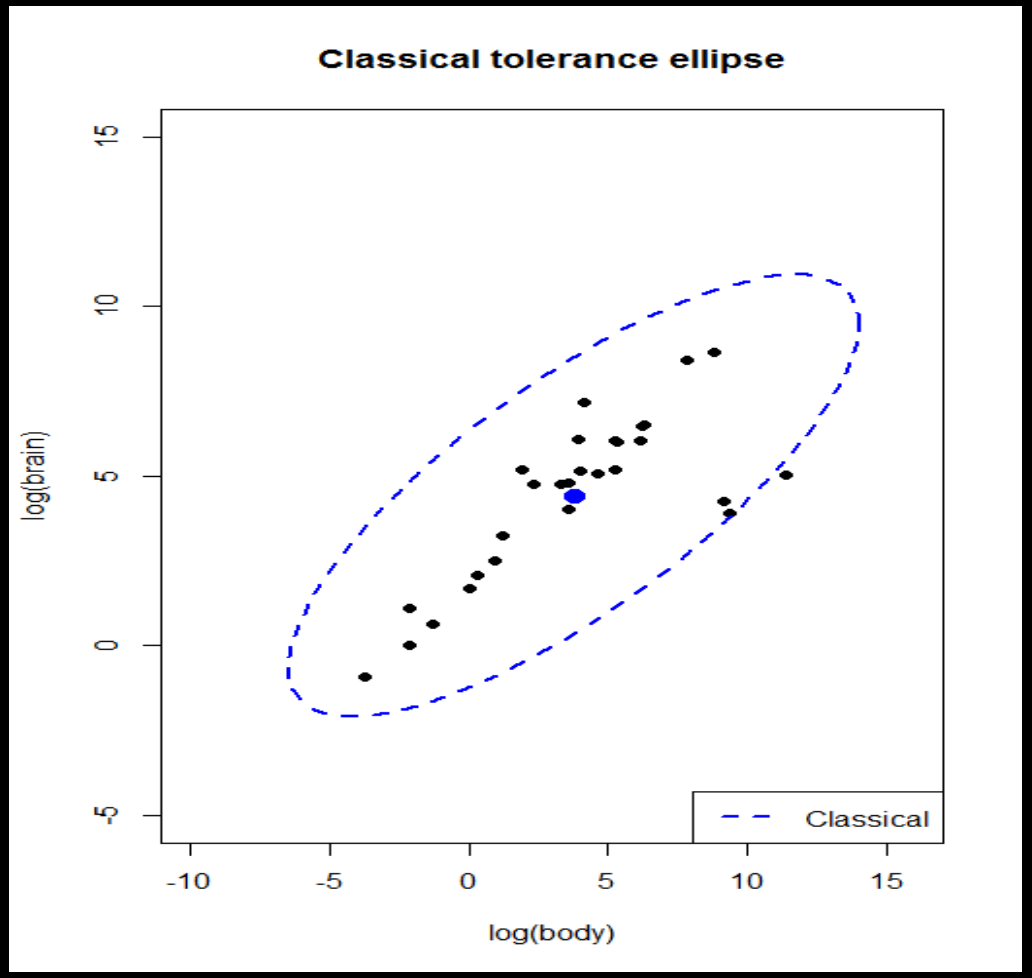


Table 7. Body and Brain Weight for 28 Animals

Index (i)	Species	Body Weight[a] (x_i)	Brain Weight[b] (y_i)
1	Mountain beaver	1.350	8.100
2	Cow	465.000	423.000
3	Gray wolf	36.330	119.500
4	Goat	27.660	115.000
5	Guinea pig	1.040	5.500
6	Diplodocus	11700.000	50.000
7	Asian elephant	2547.000	4603.000
8	Donkey	187.100	419.000
9	Horse	521.000	655.000
10	Potar monkey	10.000	115.000
11	Cat	3.300	25.600
12	Giraffe	529.000	680.000
13	Gorilla	207.000	406.000
14	Human	62.000	1320.000
15	African elephant	6654.000	5712.000
16	Triceratops	9400.000	70.000
17	Rhesus monkey	6.800	179.000
18	Kangaroo	35.000	56.000
19	Hamster	0.120	1.000
20	Mouse	0.023	0.400
21	Rabbit	2.500	12.100
22	Sheep	55.500	175.000
23	Jaguar	100.000	157.000
24	Chimpanzee	52.160	440.000
25	Brachiosaurus	87000.000	154.500
26	Rat	0.280	1.900
27	Mole	0.122	3.000
28	Pig	192.000	180.000

[a]In kilograms.
[b]In grams.
Source: Weisberg (1980) and Jerison (1973).

Minimum Covariance Determinant (MCD)

- Better: use **robust estimators for locations and scatter**
- MCD estimator (Rousseeuw, 1984) is a popular high-breakdown estimator of location and scatter.

For fixed h, with $\frac{[n+p+1]}{2} \leq h \leq n$,

1. $\hat{\mu}_0$ is the mean of the h observations for which the determinant of the sample covariance is minimal;
2. $\hat{\Sigma}_0$ is that covariance matrix (multiplied by consistency factor).

- Calculate Robust distance to the mean

$$RD(x_i) = \sqrt{(x_i - \hat{\mu}_0)' \hat{\Sigma}_0^{-1} (x_i - \hat{\mu}_0)}.$$

- Flag observation as outlier if $RD(x_i) \leq \sqrt{\chi^2_{p,0.975}}$

Tolerance ellipsoid

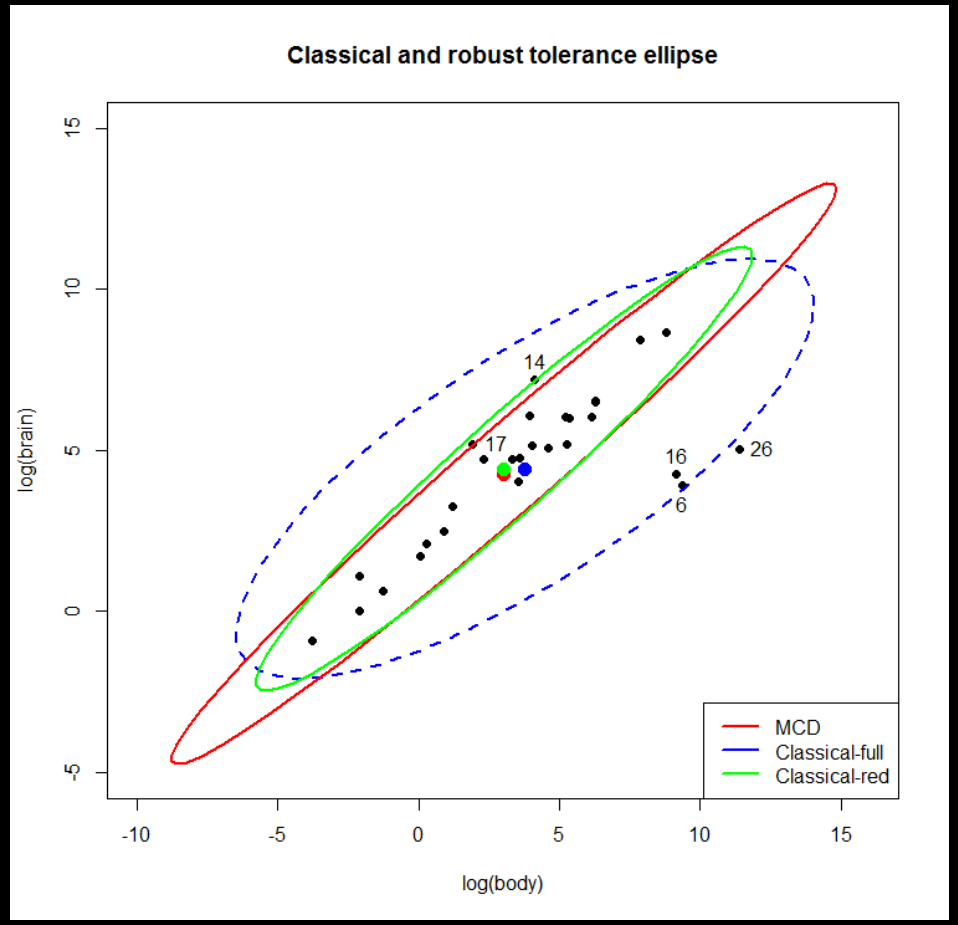


Table 7. Body and Brain Weight for 28 Animals

Index (i)	Species	Body Weight[a] (x_i)	Brain Weight[b] (y_i)
1	Mountain beaver	1.350	8.100
2	Cow	465.000	423.000
3	Gray wolf	36.330	119.500
4	Goat	27.660	115.000
5	Guinea pig	1.040	5.500
6	Diplodocus	11700.000	50.000
7	Asian elephant	2547.000	4603.000
8	Donkey	187.100	419.000
9	Horse	521.000	655.000
10	Potar monkey	10.000	115.000
11	Cat	3.300	25.600
12	Giraffe	529.000	680.000
13	Gorilla	207.000	406.000
14	Human	62.000	1320.000
15	African elephant	6654.000	5712.000
16	Triceratops	9400.000	70.000
17	Rhesus monkey	6.800	179.000
18	Kangaroo	35.000	56.000
19	Hamster	0.120	1.000
20	Mouse	0.023	0.400
21	Rabbit	2.500	12.100
22	Sheep	55.500	175.000
23	Jaguar	100.000	157.000
24	Chimpanzee	52.160	440.000
25	Brachiosaurus	87000.000	154.500
26	Rat	0.280	1.900
27	Mole	0.122	3.000
28	Pig	192.000	180.000

[a] In kilograms.
[b] In grams.
Source: Weisberg (1980) and Jerison (1973).

Distance-distance plot

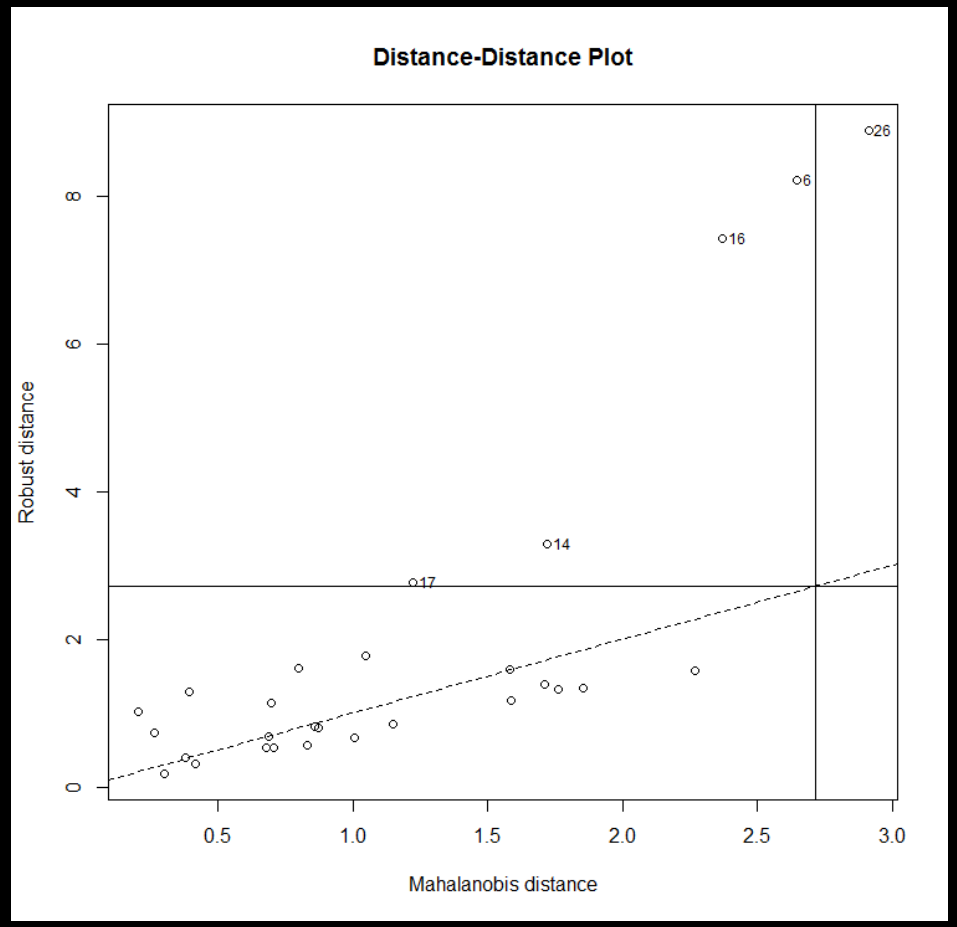


Table 7. Body and Brain Weight for 28 Animals

Index (i)	Species	Body Weight[a] (x_i)	Brain Weight[b] (y_i)
1	Mountain beaver	1.350	8.100
2	Cow	465.000	423.000
3	Gray wolf	36.330	119.500
4	Goat	27.660	115.000
5	Guinea pig	1.040	5.500
6	Diplodocus	11700.000	50.000
7	Asian elephant	2547.000	4603.000
8	Donkey	187.100	419.000
9	Horse	521.000	655.000
10	Potar monkey	10.000	115.000
11	Cat	3.300	25.600
12	Giraffe	529.000	680.000
13	Gorilla	207.000	406.000
14	Human	62.000	1320.000
15	African elephant	6654.000	5712.000
16	Triceratops	9400.000	70.000
17	Rhesus monkey	6.800	179.000
18	Kangaroo	35.000	56.000
19	Hamster	0.120	1.000
20	Mouse	0.023	0.400
21	Rabbit	2.500	12.100
22	Sheep	55.500	175.000
23	Jaguar	100.000	157.000
24	Chimpanzee	52.160	440.000
25	Brachiosaurus	87000.000	154.500
26	Rat	0.280	1.900
27	Mole	0.122	3.000
28	Pig	192.000	180.000

[a] In kilograms.
[b] In grams.
Source: Weisberg (1980) and Jerison (1973).

Fraudsters in animal data set?

- Outliers (body weight in kg; brain weight in g)
    - 6: Diplodocus (11700;50)
    - 16: Triceratops (9400;70)
    - 26: Brachiosaurus (87000; 154.5)
    - 14: Human (62; 1320)
    - 17: Rhesus monkey (6.8;179)

Conclusions

- Financial institutions are forced to continuously improve their fraud detection systems and they do so by increasingly relying on statistical models.
- Feature engineering is very important to improve performance.
- In Fraud detection we have to deal with imbalanced datasets.
- We present method that takes into account the actual financial costs associated with the fraud detection process.
- Anomaly detection techniques help detecting suspicious behavior.

Some references

- Bahnsen, A. C., Aouada, D., Stojanovic, A., & Ottersten, B. (2016). Feature engineering strategies for credit card fraud detection. *Expert Systems with Applications*, *51*, 134-142.
- Bahnsen, A. C., Aouada, D., & Ottersten, B. (2014, December). Example-dependent cost-sensitive logistic regression for credit scoring. In *2014 13th International Conference on Machine Learning and Applications* (pp. 263-269). IEEE.
- Hubert, M., Rousseeuw, P. J., & Verdonck, T. (2012). A deterministic algorithm for robust location and scatter. *Journal of Computational and Graphical Statistics*, *21*(3), 618-637.
- Hubert, M., & Vandervieren, E. (2008). An adjusted boxplot for skewed distributions. *Computational statistics & data analysis*, *52*(12), 5186-5201.
- Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008, December). Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining* (pp. 413-422). IEEE.
- Nigrini, M. J. (2012). *Benford's Law: Applications for forensic accounting, auditing, and fraud detection* (Vol. 586). John Wiley & Sons.
- Rousseeuw, P. J., & Hubert, M. (2018). Anomaly detection by robust statistics. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *8*(2), e1236.
- DataCamp courses *Fraud Detection in R* and *Anomaly Detection in R*.

Some references

- 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.
- Han, H., Wang, W. Y., & Mao, B. H. (2005, August). Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing* (pp. 878-887). Springer, Berlin, Heidelberg.
- He, H., Bai, Y., Garcia, E. A., & Li, S. (2008, June). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)* (pp. 1322-1328). IEEE.
- Bunkhumpornpat, C., Sinapiromsaran, K., & Lursinsap, C. (2009, April). Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Pacific-Asia conference on knowledge discovery and data mining* (pp. 475-482). Springer, Berlin, Heidelberg.
- Bunkhumpornpat, C., Sinapiromsaran, K., & Lursinsap, C. (2012). DBSMOTE: density-based synthetic minority over-sampling technique. *Applied Intelligence*, *36*(3), 664-684.
- Barua, S., Islam, M. M., Yao, X., & Murase, K. (2012). MWMOTE--majority weighted minority oversampling technique for imbalanced data set learning. *IEEE Transactions on Knowledge and Data Engineering*, *26*(2), 405-425.
- Menardi, G., & Torelli, N. (2014). Training and assessing classification rules with imbalanced data. *Data Mining and Knowledge Discovery*, *28*(1), 92-122.

Thank you

Questions?

tim.verdonck@uantwerpen.be

Add me on linkedin for news about research, events, …