UNIVERSITÀ DEGLI STUDI DI PADOVA

Machine Learning 2024/2025

ARTIFICIAL INTELLIGENCE, MACHINE LEARNING AND CONTROL RESEARCH GROUP

Lecture #19 Unsupervised Learning & Anomaly Detection

Gian Antonio Susto

50% of the course is done!

Up until now we have seen:

- Preprocessing (statistics, visualizations)
- Supervised learning (regression, classification)
- 'Meta' concepts such as: optimization of a loss function, underfitting vs overfitting,
- ML programming in Python

What's next? Revised course outline

WEEK 07
2025-04-07 #19: Unsupervised Learning. Anomaly Detection
2025-04-10 #20: Clustering. K-Means Clustering. Evaluating clustering performance
2025-04-11 #21 (Lab 07): Anomaly Detection & Clustering

WEEK 08
2025-04-14 #22: Support Vector Machines (SVM), Linear and kernel-based approaches, ROC & AUC
2025-04-17 #23: Introduction to Neural Networks (NNs), Activation functions, Perceptrons

WEEK 09
2025-04-24 #24: Training of NNs #01

WEEK 10
2025-04-28 #25: Training of NNs #02

WEEK 11
2025-05-05 #26: Convolutional Neural Networks
2025-05-08 #27: Autoencoders & Recurrent NNs
2025-05-09 #28 (Lab 08): NN #01

WEEK 12
2025-05-13 #29: Fairness in ML
2025-05-15 Programming Mock Exam discussion
2025-05-16 #30 (Lab 09): NN #02

WEEK 13
2025-05-19 Recap session with TAs
2025-05-22 #31: Real-world Applications and MLOps (Industrial Guest Lecture)
2025-05-23 #32 (Lab 10): Recap LAB

WEEK 14
2025-05-26 #33: XAI #01
2025-05-29 #34: XAI #02
2025-05-30 #35 (Lab 10): XAI

WEEK 15
2025-06-05 #36: ML, what's next?
2025-06-06 Buffer slot

WEEK 16
2025-06-09 Buffer slot
2025-06-12 Buffer slot

What's next? Revised course outline

WEEK 07
2025-04-07 #19: Unsupervised Learning. Anomaly Detection
2025-04-10 #20: Clustering. K-Means Clustering. Evaluating clustering performance
2025-04-11 #21 (Lab 07): Anomaly Detection & Clustering

WEEK 08
2025-04-14 #22: Support Vector Machines (SVM), Linear and kernel-based approaches, ROC & AUC
2025-04-17 #23: Introduction to Neural Networks (NNs), Activation functions, Perceptrons

WEEK 09
2025-04-24 #24: Training of NNs #01

WEEK 10
2025-04-28 #25: Training of NNs #02

WEEK 11
2025-05-05 #26: Convolutional Neural Networks
2025-05-08 #27: Autoencoders & Recurrent NNs
2025-05-09 #28 (Lab 08): NN #01

WEEK 12
2025-05-13 #29: Fairness in ML
2025-05-15 Programming Mock Exam discussion
2025-05-16 #30 (Lab 09): NN #02

WEEK 13
2025-05-19 Recap session with TAs
2025-05-22 #31: Real-world Applications and MLOps (Industrial Guest Lecture)
2025-05-23 #32 (Lab 10): Recap LAB

WEEK 14
2025-05-26 #33: XAI #01
2025-05-29 #34: XAI #02
2025-05-30 #35 (Lab 10): XAI

WEEK 15
2025-06-05 #36: ML, what's next?
2025-06-06 Buffer slot

WEEK 16
2025-06-09 Buffer slot
2025-06-12 Buffer slot

By the end of this week, you'll have the basic tools to complete an end-to-end ML feasibility assessment!

Early next week we'll provide you with a programming 'mock' exam: solutions will be given and discussed in one month

What's next? Revised course outline

WEEK 07
2025-04-07 #19: Unsupervised Learning. Anomaly Detection
2025-04-10 #20: Clustering. K-Means Clustering. Evaluating clustering performance
2025-04-11 #21 (Lab 07): Anomaly Detection & Clustering

WEEK 08
2025-04-14 #22: Support Vector Machines (SVM), Linear and kernel-based approaches, ROC & AUC
2025-04-17 #23: Introduction to Neural Networks (NNs), Activation functions, Perceptrons

WEEK 09
2025-04-24 #24: Training of NNs #01

WEEK 10
2025-04-28 #25: Training of NNs #02

WEEK 11
2025-05-05 #26: Convolutional Neural Networks
2025-05-08 #27: Autoencoders & Recurrent NNs
2025-05-09 #28 (Lab 08): NN #01

WEEK 12
2025-05-13 #29: Fairness in ML
2025-05-15 Programming Mock Exam discussion
2025-05-16 #30 (Lab 09): NN #02

WEEK 13
2025-05-19 Recap session with TAs
2025-05-22 #31: Real-world Applications and MLOps (Industrial Guest Lecture)
2025-05-23 #32 (Lab 10): Recap LAB

WEEK 14
2025-05-26 #33: XAI #01
2025-05-29 #34: XAI #02
2025-05-30 #35 (Lab 10): XAI

WEEK 15
2025-06-05 #36: ML, what's next?
2025-06-06 Buffer slot

WEEK 16
2025-06-09 Buffer slot
2025-06-12 Buffer slot

We will go back to supervised learning soon: next week we'll talk about Support Vector Machines: this topic will be relevant only for the theoretic part of the exam!

What's next? Revised course outline

WEEK 07
2025-04-07 #19: Unsupervised Learning. Anomaly Detection
2025-04-10 #20: Clustering. K-Means Clustering. Evaluating clustering performance
2025-04-11 #21 (Lab 07): Anomaly Detection & Clustering

WEEK 08
2025-04-14 #22: Support Vector Machines (SVM), Linear and kernel-based approaches, ROC & AUC
2025-04-17 #23: Introduction to Neural Networks (NNs), Activation functions, Perceptrons

WEEK 09
2025-04-24 #24: Training of NNs #01

WEEK 10
2025-04-28 #25: Training of NNs #02

WEEK 11
2025-05-05 #26: Convolutional Neural Networks
2025-05-08 #27: Autoencoders & Recurrent NNs
2025-05-09 #28 (Lab 08): NN #01

WEEK 12
2025-05-13 #29: Fairness in ML
2025-05-15 Programming Mock Exam discussion
2025-05-16 #30 (Lab 09): NN #02

WEEK 13
2025-05-19 Recap session with TAs
2025-05-22 #31: Real-world Applications and MLOps (Industrial Guest Lecture)
2025-05-23 #32 (Lab 10): Recap LAB

WEEK 14
2025-05-26 #33: XAI #01
2025-05-29 #34: XAI #02
2025-05-30 #35 (Lab 10): XAI

WEEK 15
2025-06-05 #36: ML, what's next?
2025-06-06 Buffer slot

WEEK 16
2025-06-09 Buffer slot
2025-06-12 Buffer slot

Basics of Neural Networks / Deep Learning will be provided.

For the programming part of NNs we will mainly use libraries where the basic blocks of a NN are already implemented!

What's next? Revised course outline

WEEK 07
2025-04-07 #19: Unsupervised Learning. Anomaly Detection
2025-04-10 #20: Clustering. K-Means Clustering. Evaluating clustering performance
2025-04-11 #21 (Lab 07): Anomaly Detection & Clustering

WEEK 08
2025-04-14 #22: Support Vector Machines (SVM), Linear and kernel-based approaches, ROC & AUC
2025-04-17 #23: Introduction to Neural Networks (NNs), Activation functions, Perceptrons

WEEK 09
2025-04-24 #24: Training of NNs #01

WEEK 10
2025-04-28 #25: Training of NNs #02

WEEK 11
2025-05-05 #26: Convolutional Neural Networks
2025-05-08 #27: Autoencoders & Recurrent NNs
2025-05-09 #28 (Lab 08): NN #01

WEEK 12
2025-05-13 #29: Fairness in ML
2025-05-15 Programming Mock Exam discussion
2025-05-16 #30 (Lab 09): NN #02

WEEK 13
2025-05-19 Recap session with TAs
2025-05-22 #31: Real-world Applications and MLOps (Industrial Guest Lecture)
2025-05-23 #32 (Lab 10): Recap LAB

WEEK 14
2025-05-26 #33: XAI #01
2025-05-29 #34: XAI #02
2025-05-30 #35 (Lab 10): XAI

WEEK 15
2025-06-05 #36: ML, what's next?
2025-06-06 Buffer slot

WEEK 16
2025-06-09 Buffer slot
2025-06-12 Buffer slot

Explainability and Fairness will be covered!

What’s next? Revised course outline

WEEK 07
2025-04-07 #19: Unsupervised Learning. Anomaly Detection
2025-04-10 #20: Clustering. K-Means Clustering. Evaluating clustering performance
2025-04-11 #21 (Lab 07): Anomaly Detection & Clustering

WEEK 08
2025-04-14 #22: Support Vector Machines (SVM), Linear and kernel-based approaches, ROC & AUC
2025-04-17 #23: Introduction to Neural Networks (NNs), Activation functions, Perceptrons

WEEK 09
2025-04-24 #24: Training of NNs #01

WEEK 10
2025-04-28 #25: Training of NNs #02

WEEK 11
2025-05-05 #26: Convolutional Neural Networks
2025-05-08 #27: Autoencoders & Recurrent NNs
2025-05-09 #28 (Lab 08): NN #01

WEEK 12
2025-05-13 #29: Fairness in ML
2025-05-15 Programming Mock Exam discussion
2025-05-16 #30 (Lab 09): NN #02

WEEK 13
2025-05-19 Recap session with TAs
2025-05-22 #31: Real-world Applications and MLOps (Industrial Guest Lecture)
2025-05-23 #32 (Lab 10): Recap LAB

WEEK 14
2025-05-26 #33: XAI #01
2025-05-29 #34: XAI #02
2025-05-30 #35 (Lab 10): XAI

WEEK 15
2025-06-05 #36: ML, what’s next?
2025-06-06 Buffer slot

WEEK 16
2025-06-09 Buffer slot
2025-06-12 Buffer slot

We will have a recap for the theory and one for the programming part of the exam

What's next? Revised course outline

WEEK 07
2025-04-07 #19: Unsupervised Learning. Anomaly Detection
2025-04-10 #20: Clustering. K-Means Clustering. Evaluating clustering performance
2025-04-11 #21 (Lab 07): Anomaly Detection & Clustering

WEEK 08
2025-04-14 #22: Support Vector Machines (SVM), Linear and kernel-based approaches, ROC & AUC
2025-04-17 #23: Introduction to Neural Networks (NNs), Activation functions, Perceptrons

WEEK 09
2025-04-24 #24: Training of NNs #01

WEEK 10
2025-04-28 #25: Training of NNs #02

WEEK 11
2025-05-05 #26: Convolutional Neural Networks
2025-05-08 #27: Autoencoders & Recurrent NNs
2025-05-09 #28 (Lab 08): NN #01

WEEK 12
2025-05-13 #29: Fairness in ML
2025-05-15 Programming Mock Exam discussion
2025-05-16 #30 (Lab 09): NN #02

WEEK 13
2025-05-19 Recap session with TAs
2025-05-22 #31: Real-world Applications and MLOps (Industrial Guest Lecture)
2025-05-23 #32 (Lab 10): Recap LAB

WEEK 14
2025-05-26 #33: XAI #01
2025-05-29 #34: XAI #02
2025-05-30 #35 (Lab 10): XAI

WEEK 15
2025-06-05 #36: ML, what's next?
2025-06-06 Buffer slot

WEEK 16
2025-06-09 Buffer slot
2025-06-12 Buffer slot

Finally, two lectures on:
- Real world applications
- What's next in AI

You will be not evaluated on these two, but these lectures will provide you with a broaden view of the area!

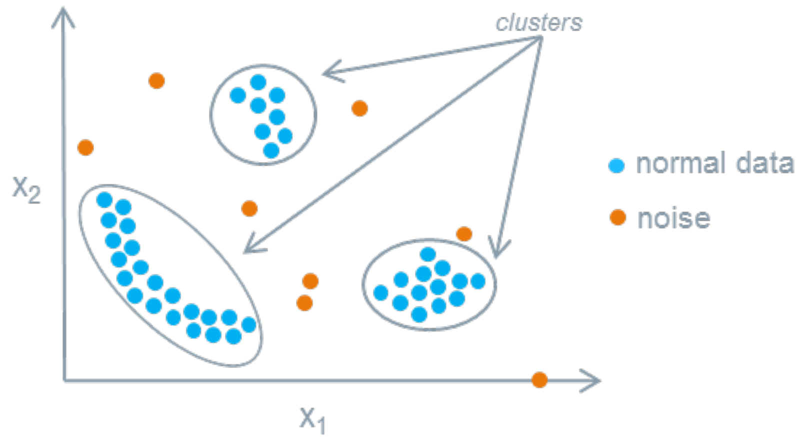
Supervised Learning

Setup: Observation of the environment

Data: (x,y)

Task: learn a map from inputs x to outputs y

Unsupervised Learning



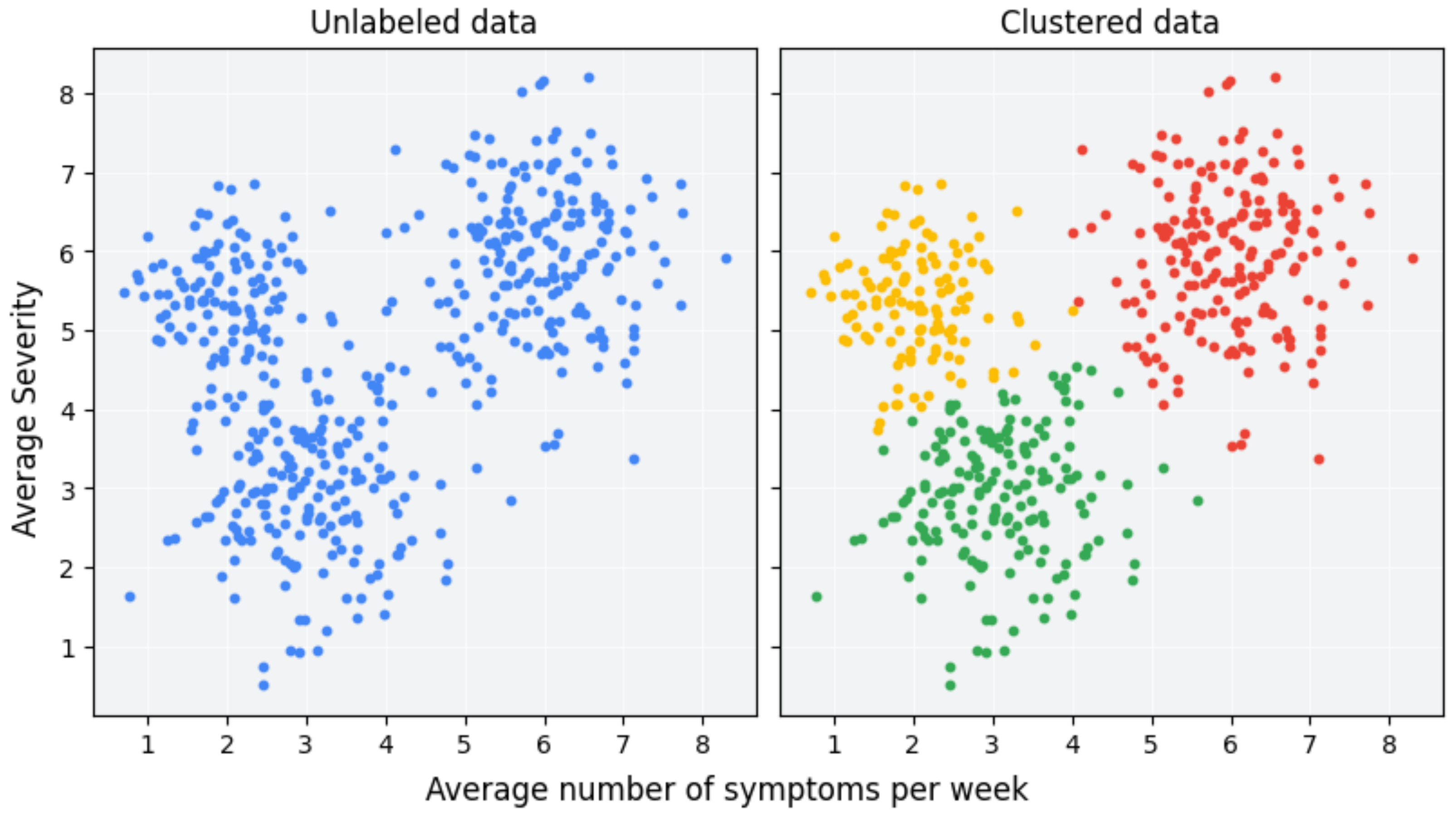
Setup: Observation of the environment

Data: x (no labels)

Task: Discover the underlying structure or distribution in data without labels

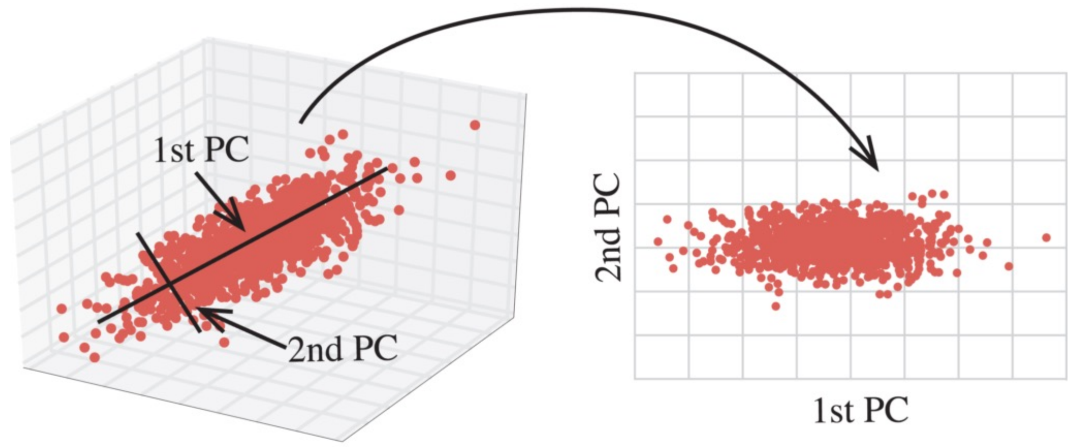
Some unsupervised learning tasks

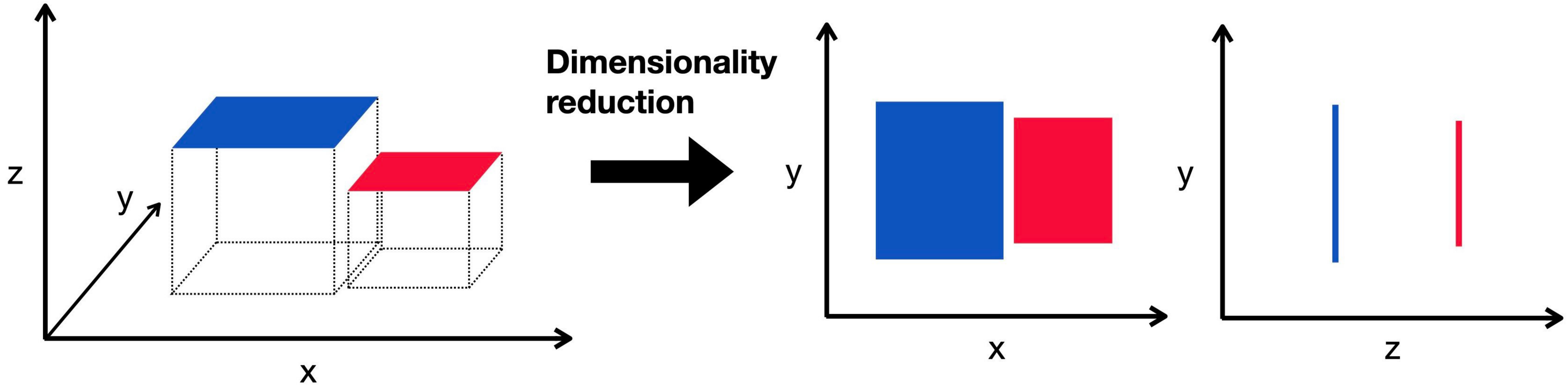
- Clustering: finding groups into data (lecture 20)



Some unsupervised learning tasks

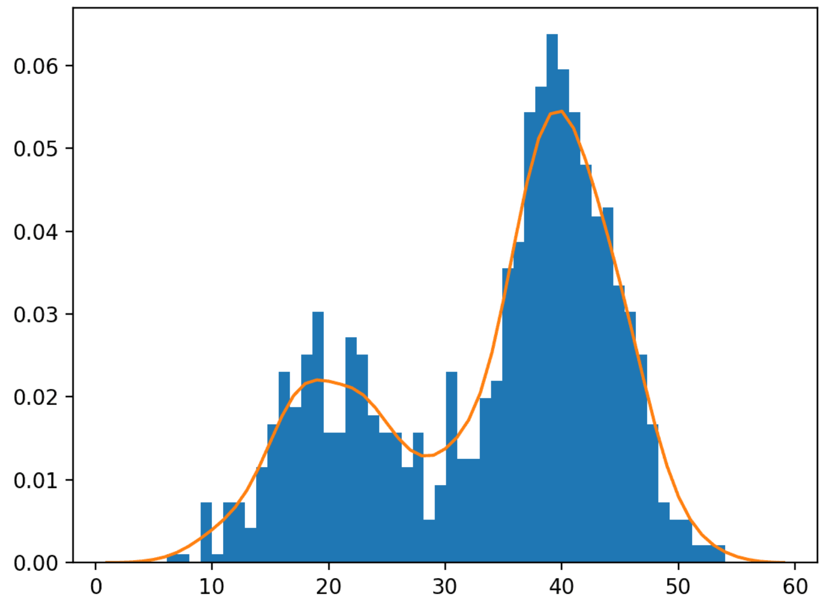
- Clustering: finding groups into data (lecture 20)
- Dimensionality reductions: reduce the number of features

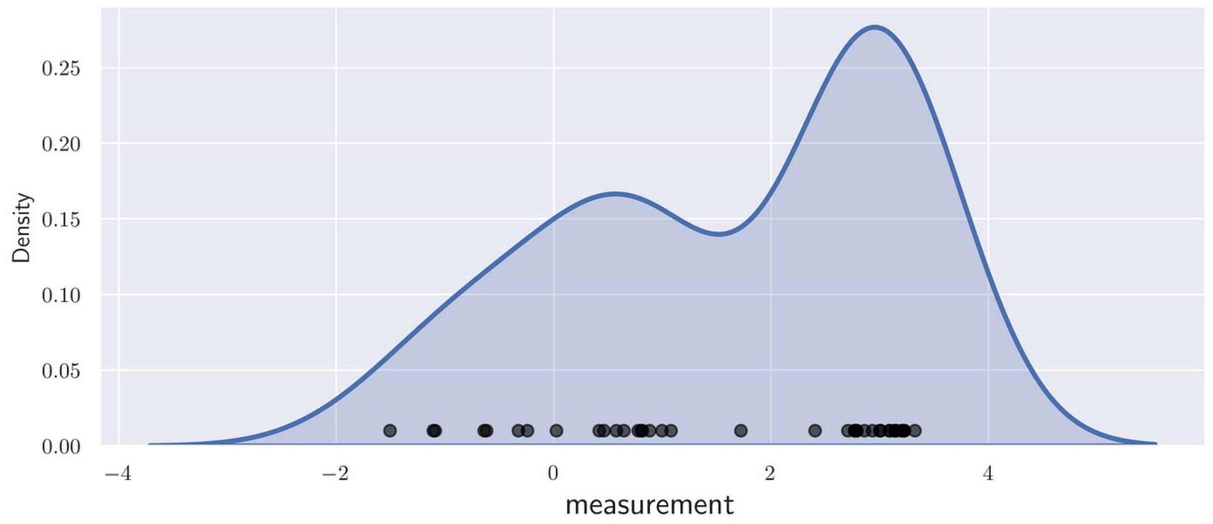




Some unsupervised learning tasks

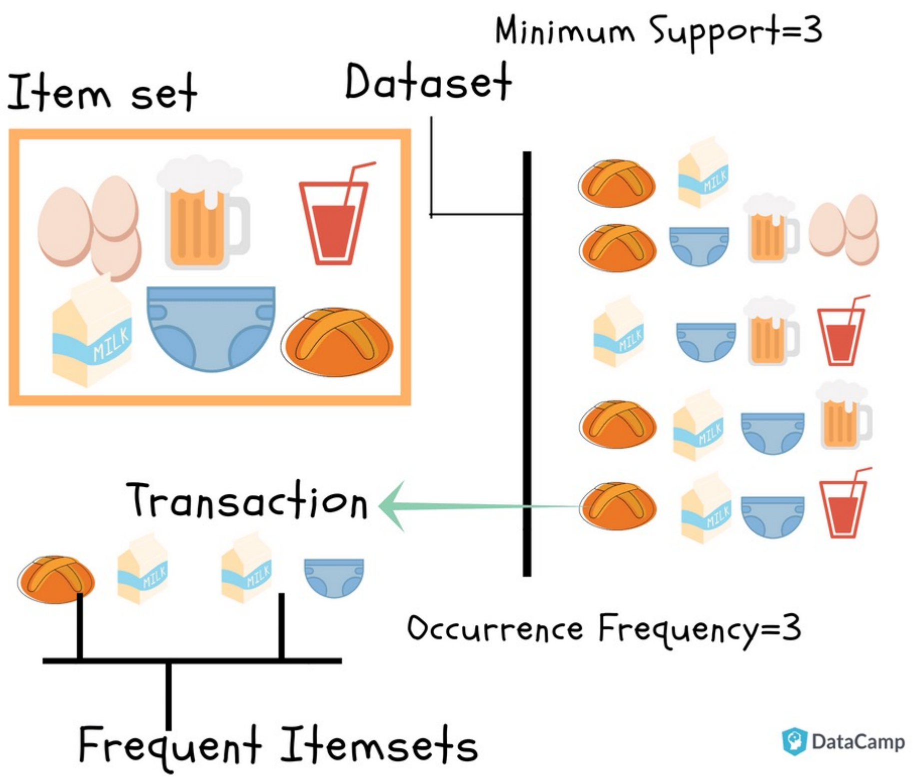
- Clustering: finding groups into data (lecture 20)
- Dimensionality reductions: reduce the number of features
- Density estimation: estimate the probability distribution that generates the data





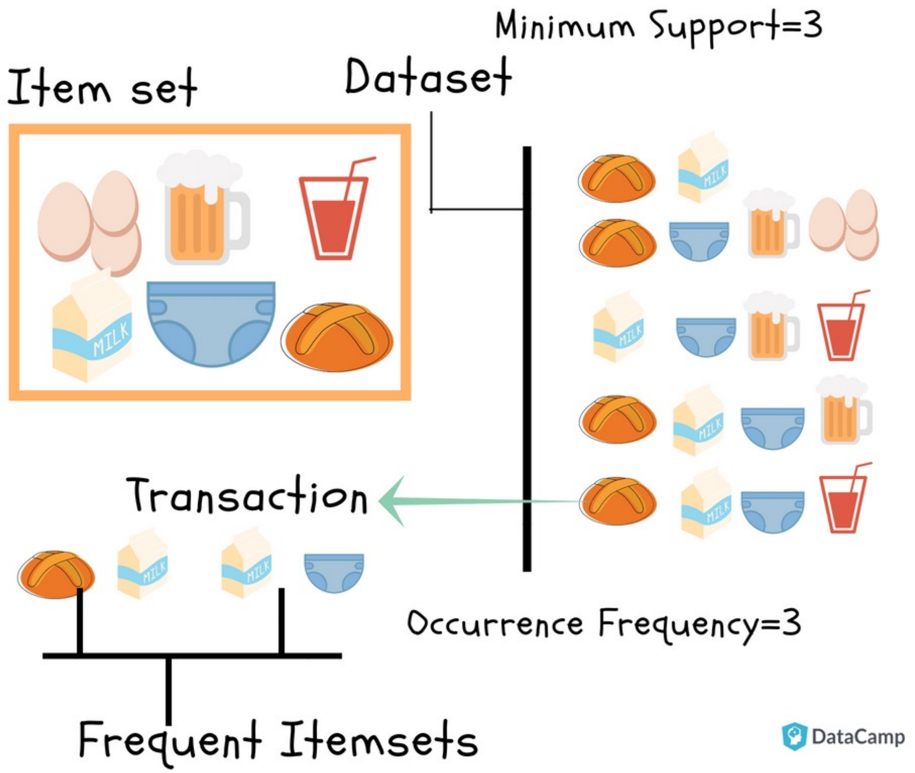
Some unsupervised learning tasks

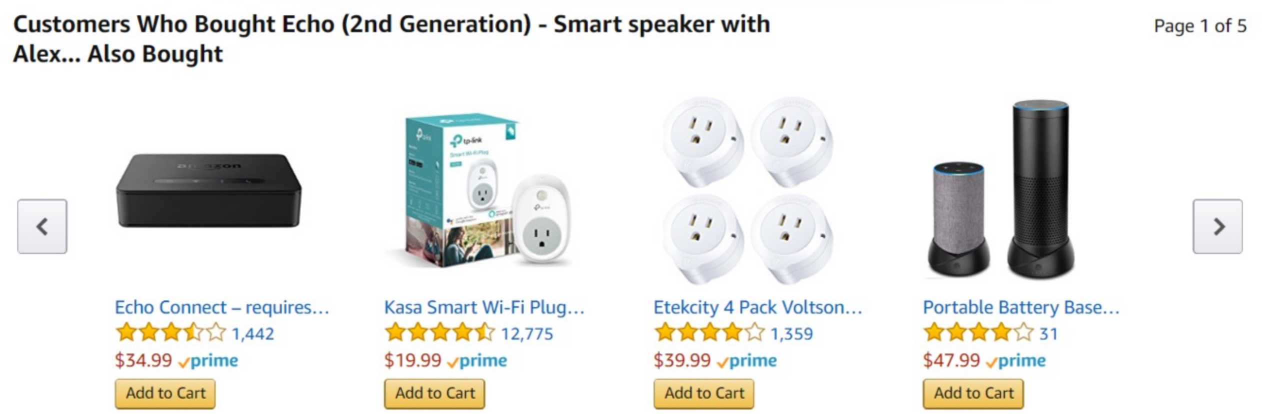
- Clustering: finding groups into data (lecture 20)
- Dimensionality reductions: reduce the number of features
- Density estimation: estimate the probability distribution that generates the data
- Association rule learning



Some unsupervised learning tasks

- Clustering: finding groups into data (lecture 20)
- Dimensionality reductions: reduce the number of features
- Density estimation: estimate the probability distribution that generates the data
- Association rule learning




Customers Who Bought Echo (2nd Generation) - Smart speaker with Alex... Also Bought

Page 1 of 5

Echo Connect – requires... 1,442 $34.99 prime Add to Cart

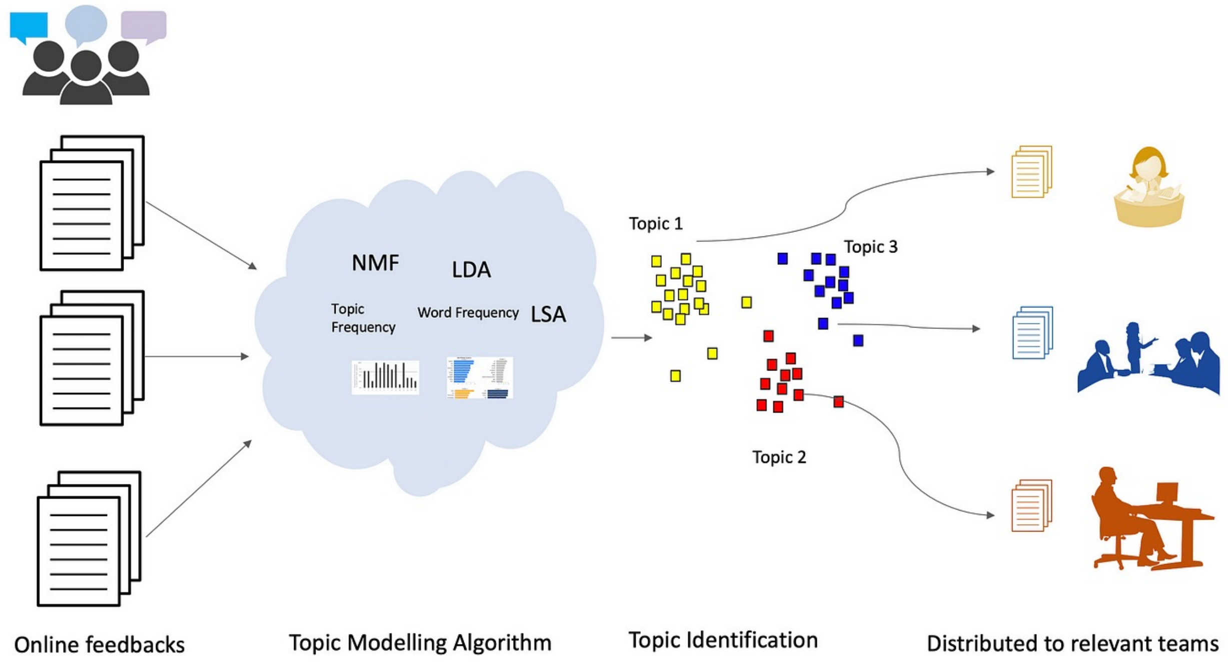
Kasa Smart Wi-Fi Plug... 12,775 $19.99 prime Add to Cart

Etekcity 4 Pack Voltson... 1,359 $39.99 prime Add to Cart

Portable Battery Base... 31 $47.99 prime Add to Cart

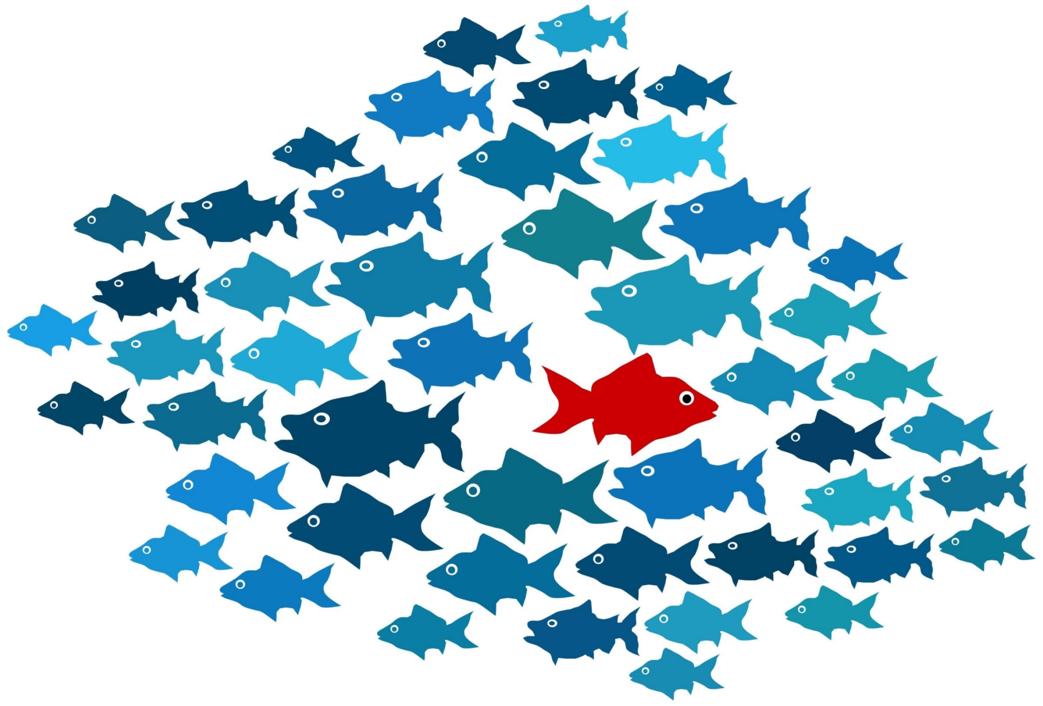
Some unsupervised learning tasks

- Clustering: finding groups into data (lecture 20)
- Dimensionality reductions: reduce the number of features
- Density estimation: estimate the probability distribution that generates the data
- Association rule learning
- Topic modelling (for text data)



Some unsupervised learning tasks

- Clustering: finding groups into data (lecture 20)
- Dimensionality reductions: reduce the number of features
- Density estimation: estimate the probability distribution that generates the data
- Association rule learning
- Topic modelling (for text data)
- Anomaly/Outlier detection (today!)

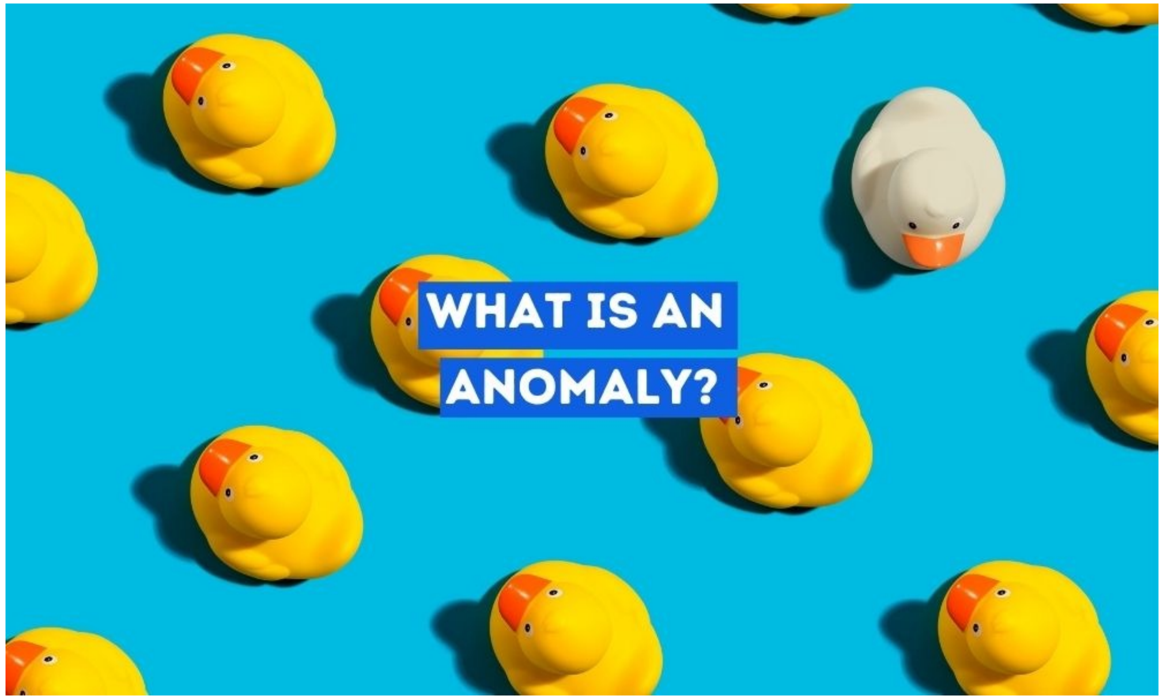


Anomaly/Outlier Detection

What is an anomaly/outlier?

‘An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism’ [1]

In a dataset – by definition – outliers should be few!

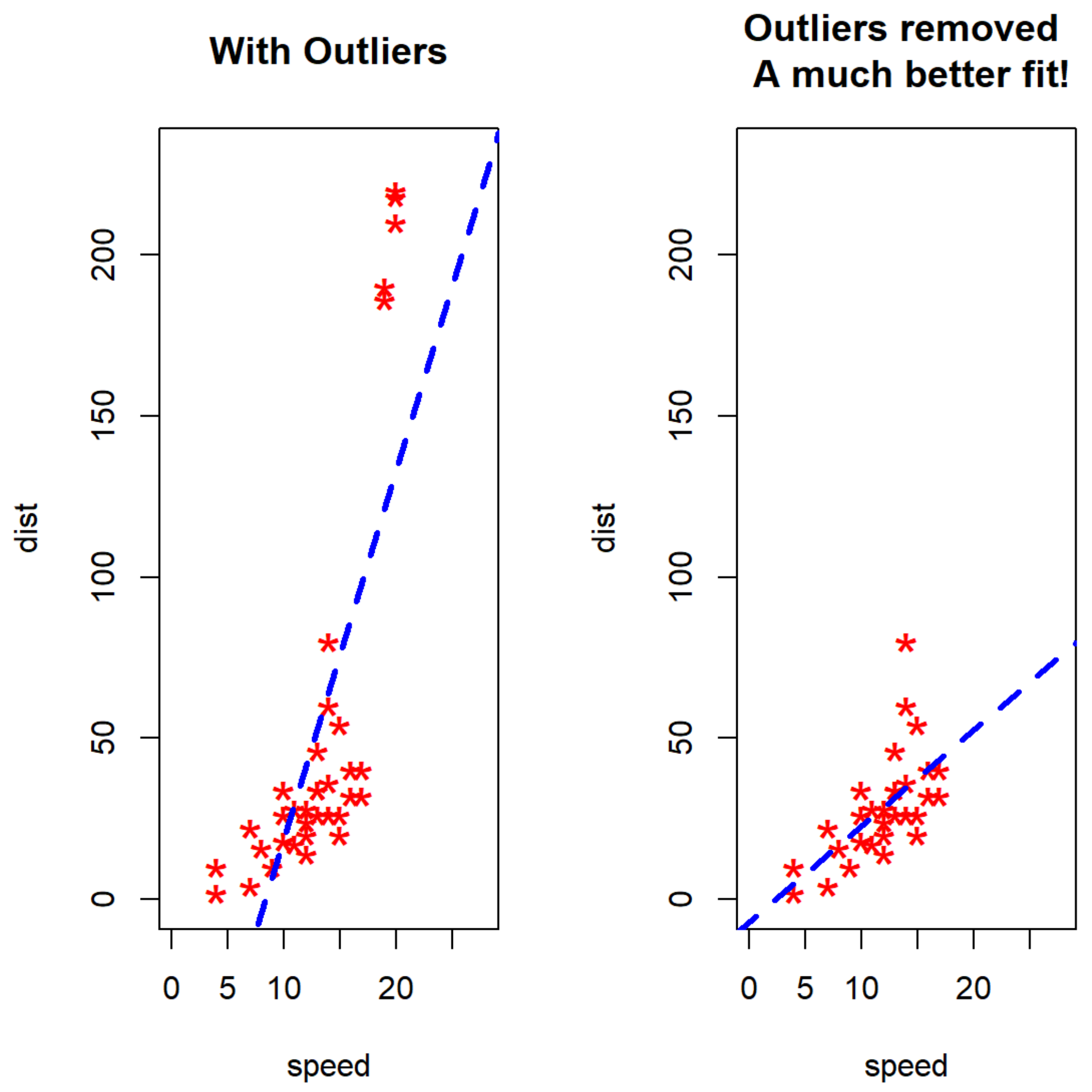


[1] D. M. Hawkins, Identification of outliers, vol. 11., Springer, 1980

Why detecting anomalies? Preprocessing (1/3)

Outliers can be data entry mistakes or measurement errors that can skew results

As a pre-processing step, anomaly detection is a data cleaning step that dramatically improve performances!

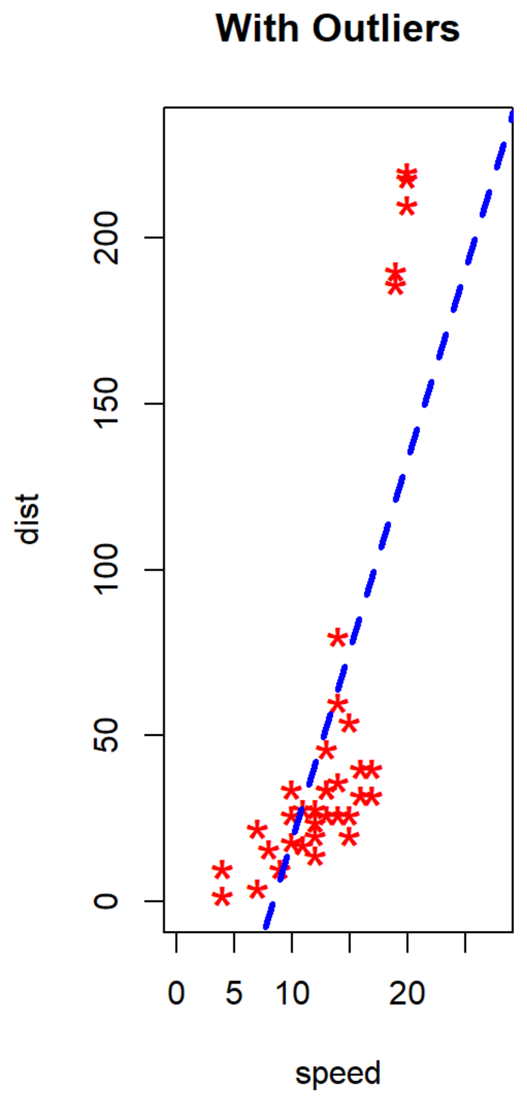


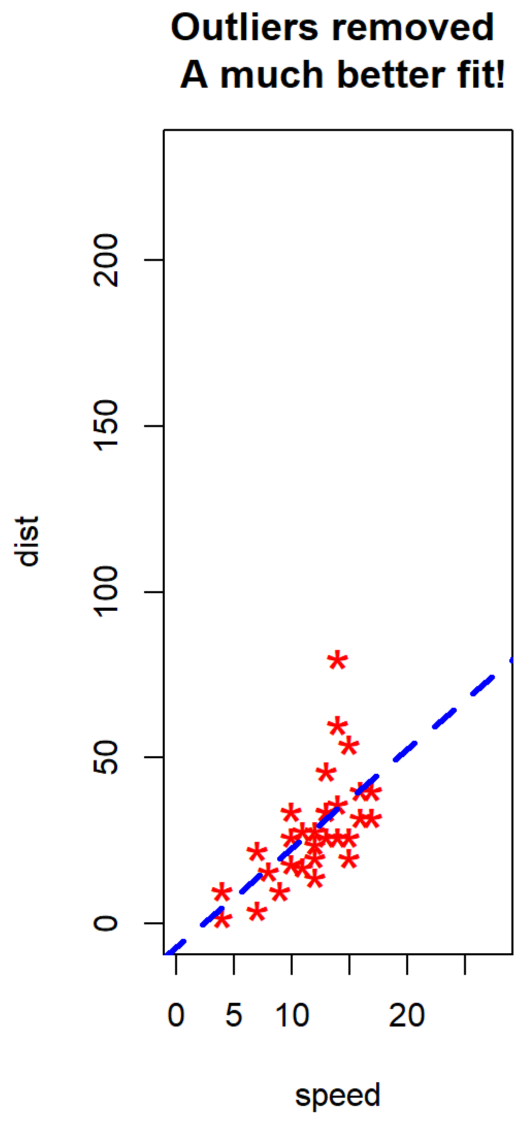
Why detecting anomalies? Preprocessing (1/3)

Outliers can be data entry mistakes or measurement errors that can skew results

As a pre-processing step, anomaly detection is a data cleaning step that dramatically improve performances!

This procedure should be handled carefully: by taking out difficult data we always improve performances!





Why detecting anomalies? Preprocessing (1/3)

✅ **When removing outliers makes sense (not cheating):**

(i) They are errors or noise

1. Example: A sensor was faulty, or someone typed "2000" instead of "200".
2. Removing these helps your model focus on meaningful data.

(ii) You're preparing data for a model that assumes normality

1. Example: Linear regression can be badly influenced by extreme values.
2. Outlier removal improves robustness.

(iii) You're analyzing a population and want to avoid skewing

1. If you're looking at "typical behavior" (e.g., average customer purchase), outliers can distort the picture.

❌ **When removing outliers is cheating:**

(i) You remove them just to improve performance metrics

1. Example: Removing hard cases from a test set so your model looks better.
2. That's data leakage and cheating.

(ii) The outliers are the thing you care about!

1. Example: Fraud detection, disease diagnosis, equipment failure.
2. Removing them defeats the whole point — you'd miss the rare but critical events.

Why detecting anomalies? Preprocessing (1/3)

✅ **When removing outliers makes sense (not cheating):**

(i) They are errors or noise

1. Example: A sensor was faulty, or someone typed "2000" instead of "200".
2. Removing these helps your model focus on meaningful data.

(ii) You're preparing data for a model that assumes normality

1. Example: Linear regression can be badly influenced by extreme values.
2. Outlier removal improves robustness.

(iii) You're analyzing a population and want to avoid skewing

1. If you're looking at "typical behavior" (e.g., average customer purchase), outliers can distort the picture.

❌ **When removing outliers is cheating:**

(i) You remove them just to improve performance metrics

1. Example: Removing hard cases from a test set so your model looks better.
2. That's data leakage and cheating.

(ii) The outliers are the thing you care about!

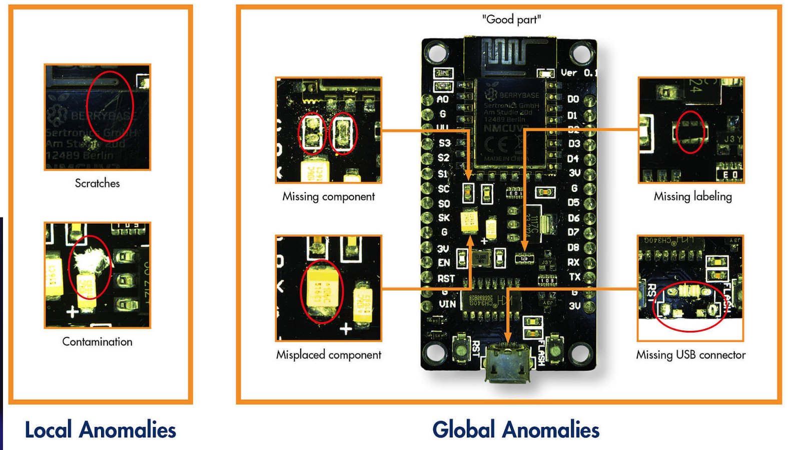
1. Example: Fraud detection, disease diagnosis, equipment failure.
2. Removing them defeats the whole point — you'd miss the rare but critical events.

Not always an easy decision:
- Domain expertise may be necessary!
- Keep track of such choices!

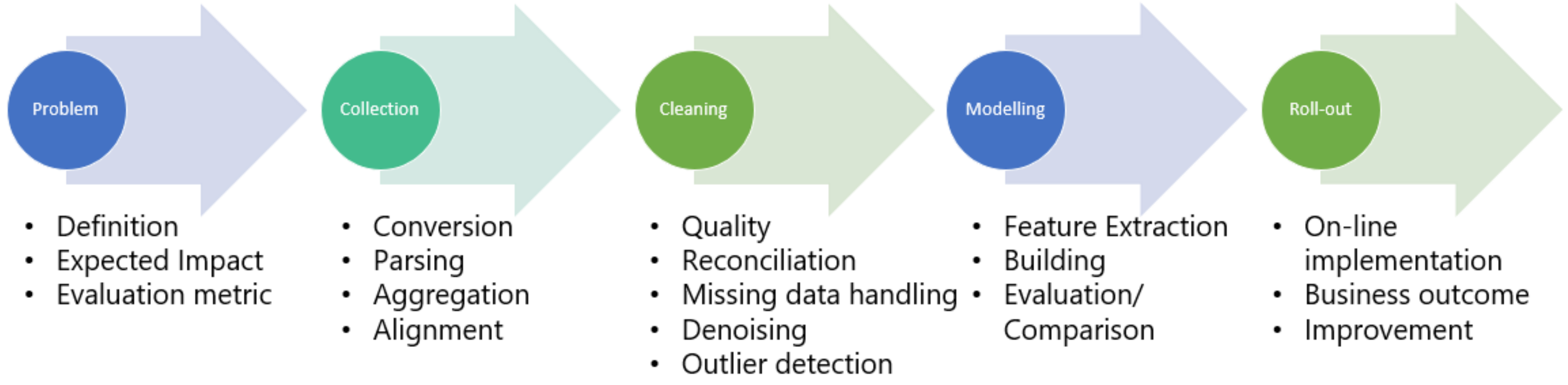
Why detecting anomalies? As the final objective (2/3)

In many domains, detecting anomalies is a fundamental task by itself:

- Fraud detection
- Diagnostic tools for manufacturing
- Cybersecurity
- Transactions on digital marketing



Unsupervised learning: either the final task or a preprocessing step!



- Definition
- Expected Impact
- Evaluation metric

- Conversion
- Parsing
- Aggregation
- Alignment

- Quality
- Reconciliation
- Missing data handling
- Denoising
- Outlier detection

- Feature Extraction
- Building
- Evaluation/ Comparison

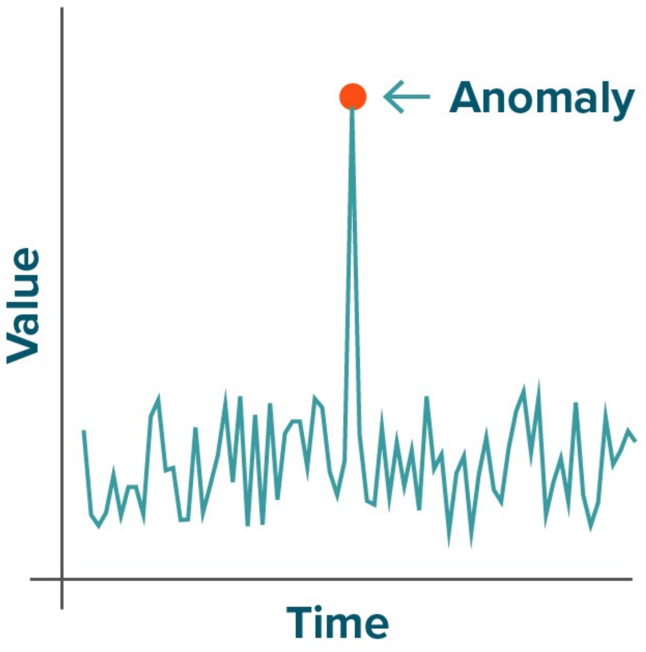
- On-line implementation
- Business outcome
- Improvement

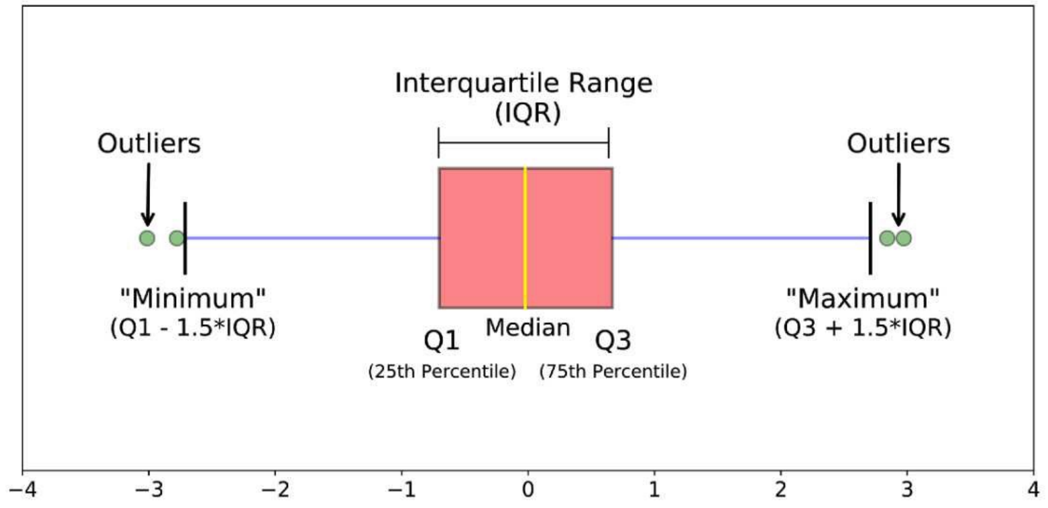
We will see, both with anomaly detection and clustering that unsupervised learning tasks can be part of a pre-processing pipeline and as the final goal of a Machine Learning project!

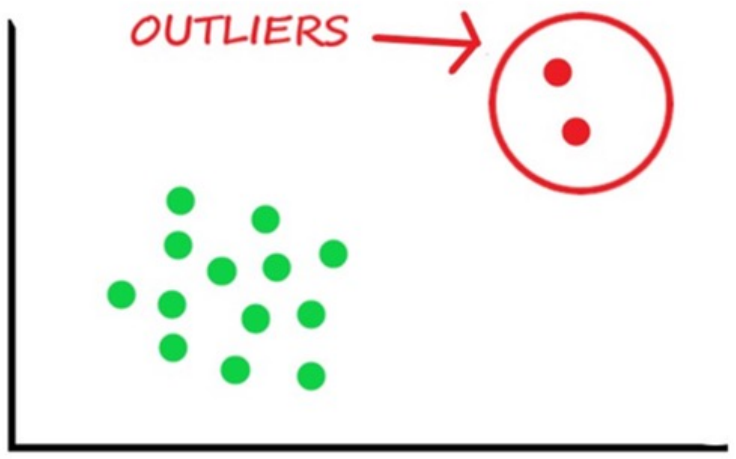
Anomaly/Outlier Detection: any ideas?

Anomaly/Outlier Detection: any ideas?

- Visualizations and statistics may be helpful!
- For example, in boxplot outliers are plotted as individual points beyond the "whiskers", aka 1.5 x IQR from Q1 and Q3







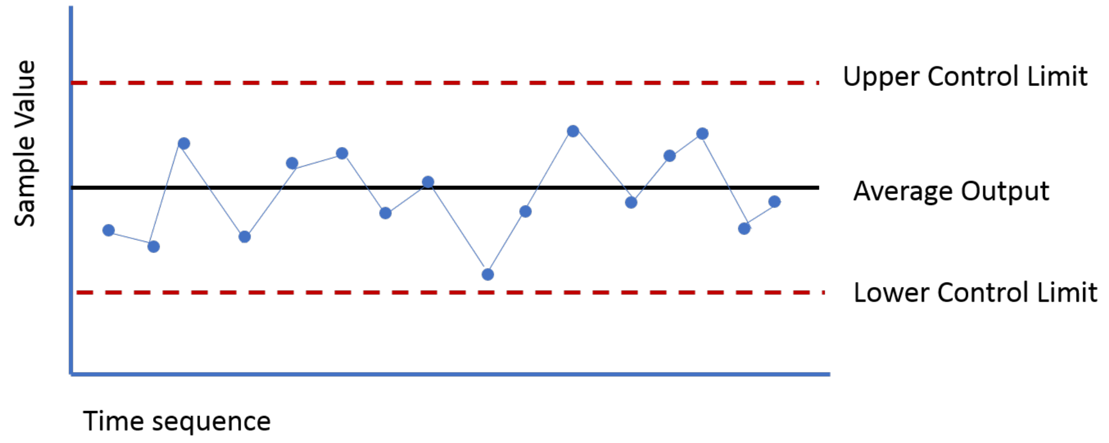
'On-line' anomaly detection: univariate control charts!

- An approach used in many industries are univariate control charts (CC)
- A control chart is a time series plot with:

(i) A center line, the expected process mean.

(ii) An upper control limit (UCL) and lower control limit (LCL), thresholds that define the "normal" range of variation.

- Typically done on variables and Key Process Indicators (KPIs), quantifiable measures describing a process 'goodness')



Computed statistically:

- Mean μ_x, STD σ_x
- Upper control limit $\mathrm{UCL} = \mu_x + A * \sigma_x$
- Lower control limit $\mathrm{LCL} = \mu_x - A * \sigma_x$

‘On-line’ anomaly detection: univariate control charts!

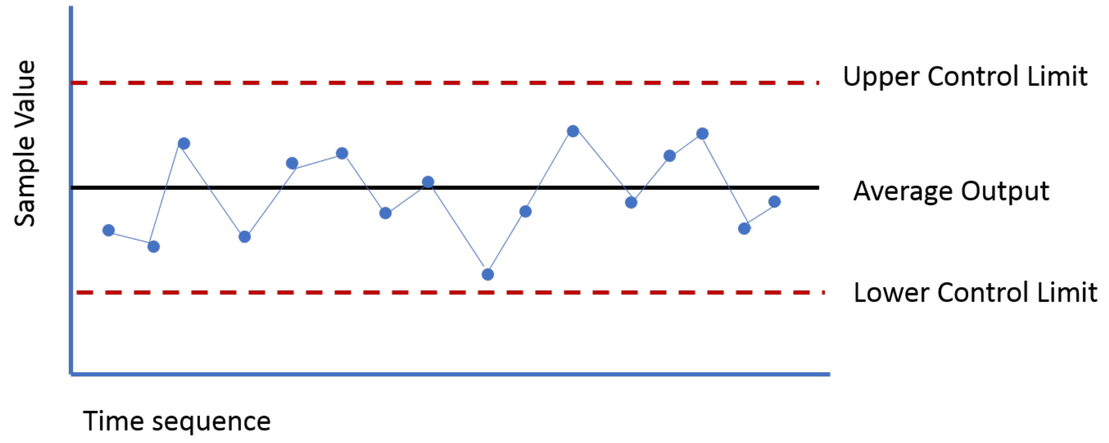
- An approach used in many industries are univariate control charts (CC)
- A control chart is a time series plot with:

(i) A center line, the expected process mean.

(ii) An upper control limit (UCL) and lower control limit (LCL), thresholds that define the "normal" range of variation.

- Typically done on variables and Key Process Indicators (KPIs), quantifiable meas[illegible] ’goo[illegible]

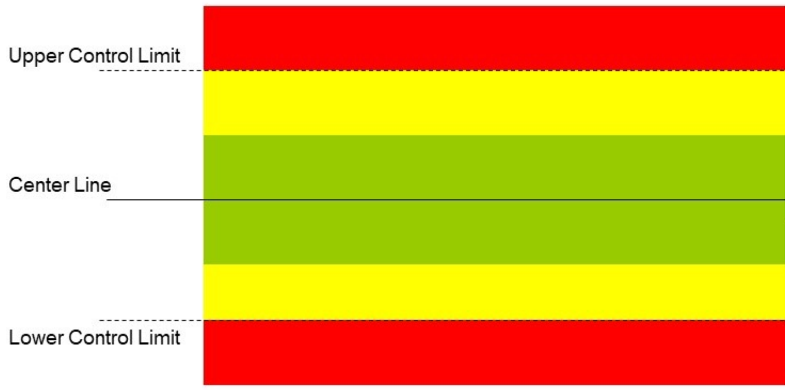
If we choose A = 3 (‘3 sigma’) and we have a gaussian variable, 99.73% of data will be ‘inlier’

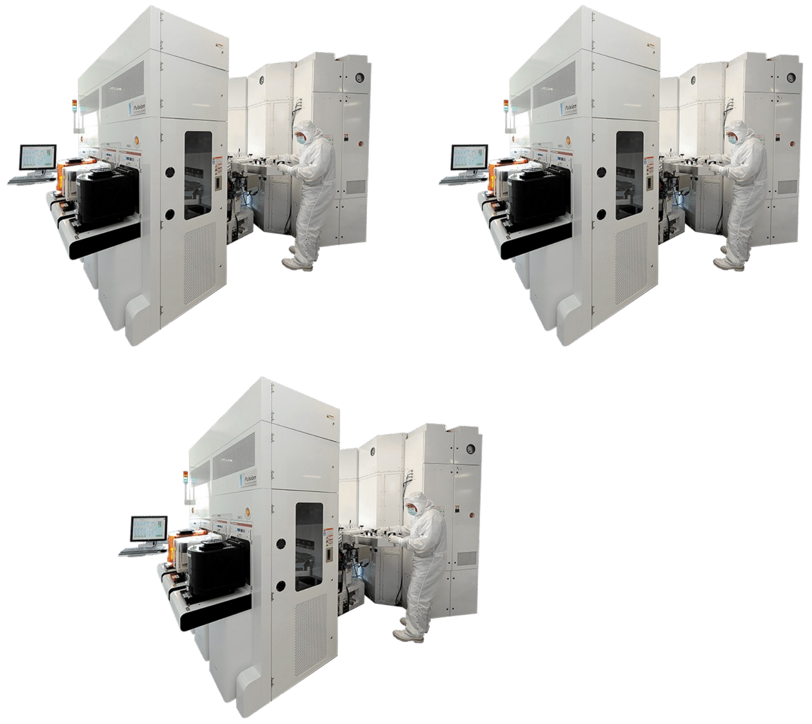


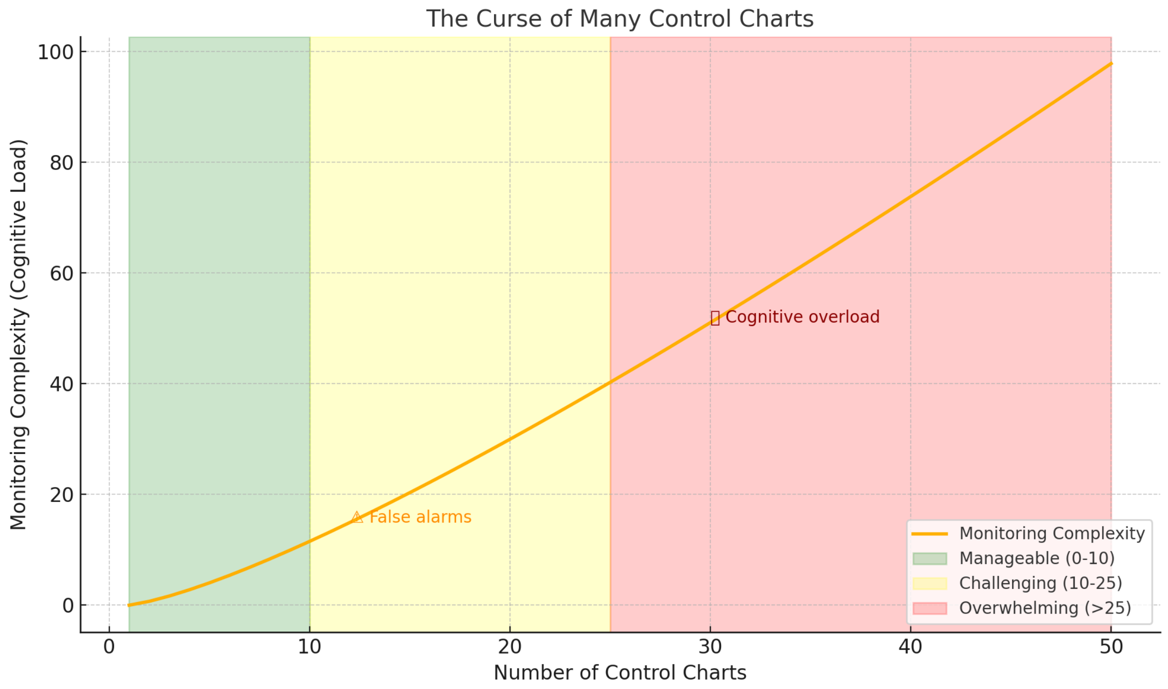
Computed statistically:

- Mean μ_x, STD σ_x
- Upper control limit $\mathrm{UCL} = \mu_x + A * \sigma_x$
- Lower control limit $\mathrm{LCL} = \mu_x - A * \sigma_x$

Problems with this approach? (1/2)

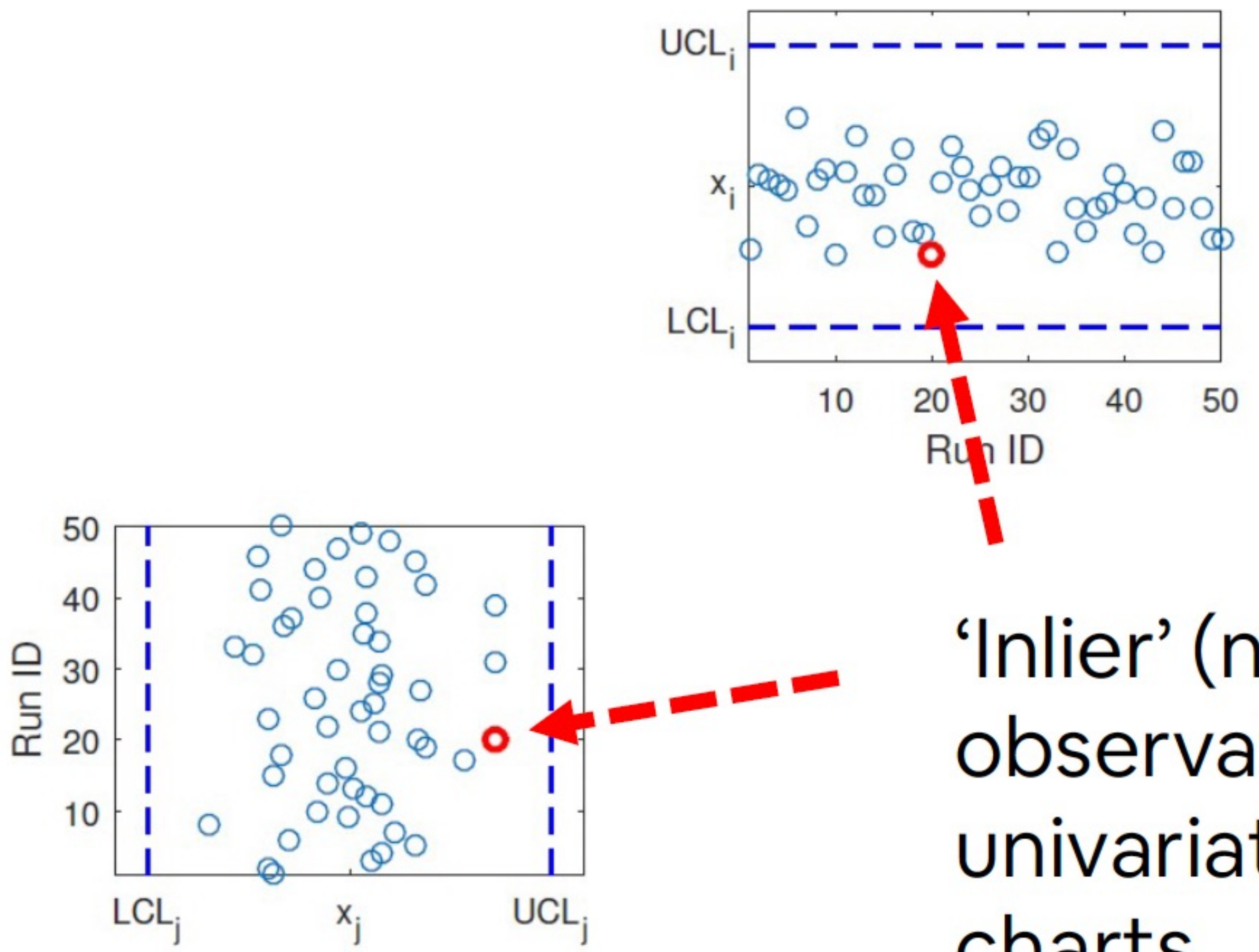






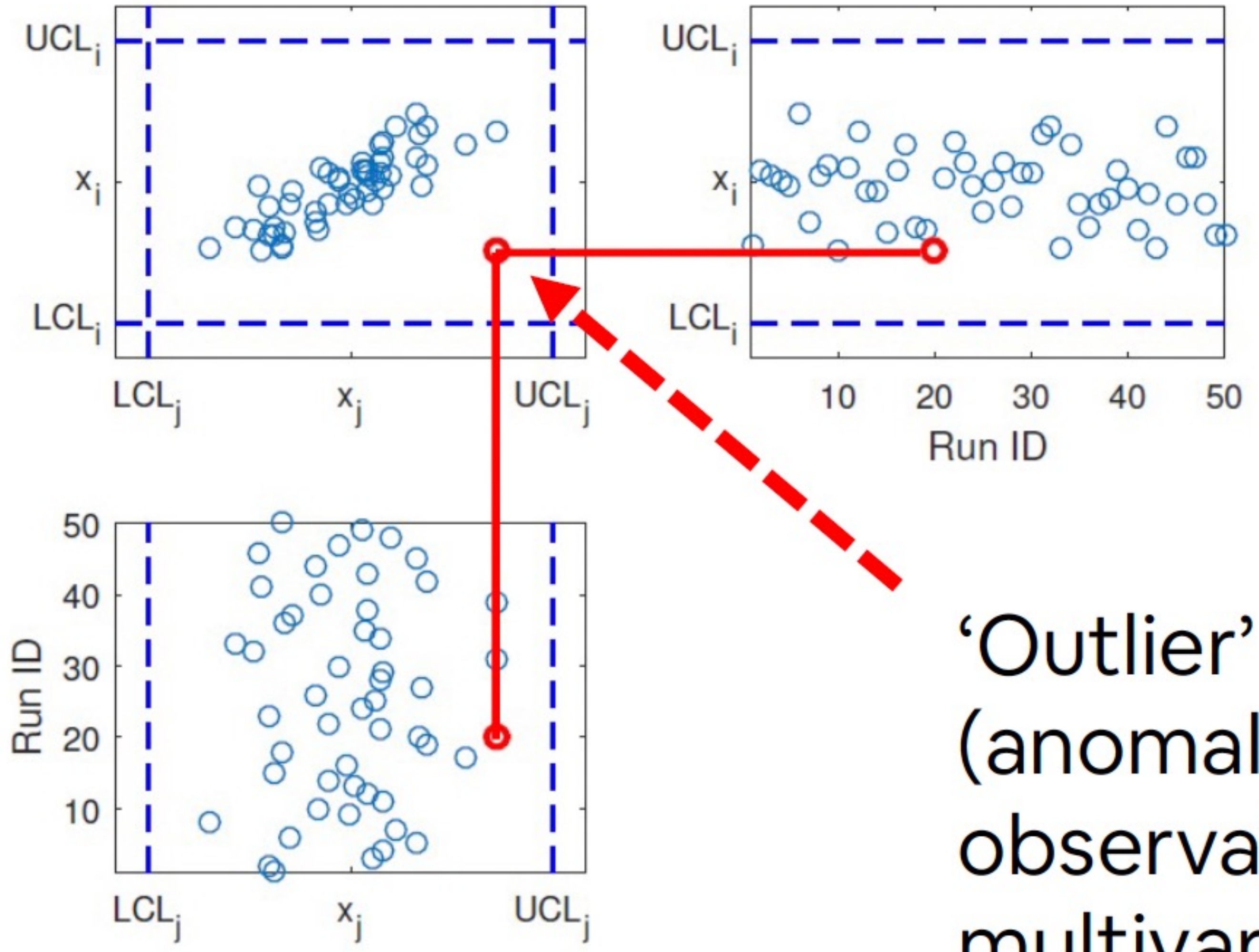
- Monitoring complexity increases with the number of control charts!
- Shift from a manageable to an overwhelming situation: increasing risks of cognitive overload and false alarms

Problems with this approach? (2/2)



'Inlier' (normal) observation in univariate control charts

Problems with this approach? (2/2)



Univariate approaches are unable to capture multivariate anomalies!

‘Outlier’ (anomalous) observation in the multivariate control chart

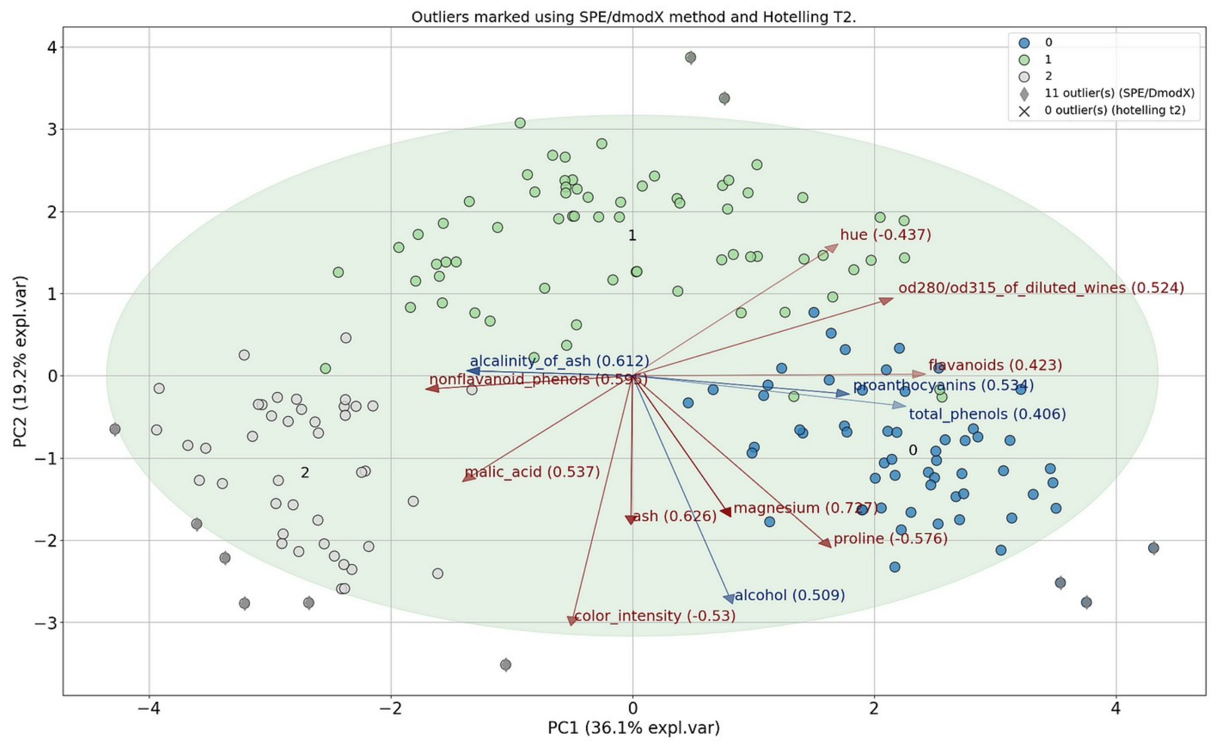
Multivariate control charts: Hotelling's T^2

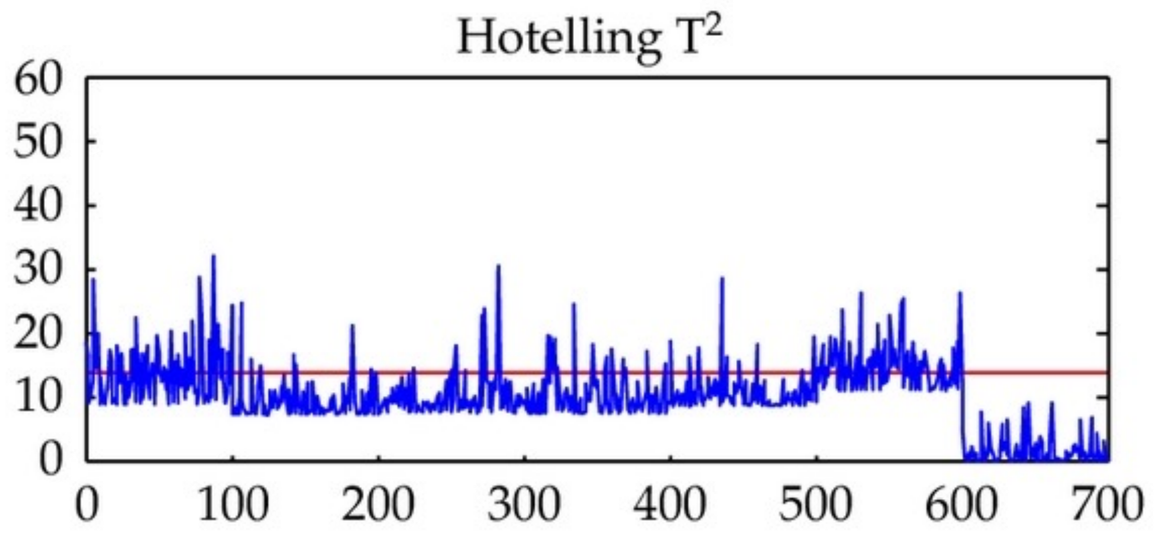
$$\boxed{T_i^2 = (\mathbf{x}_i - \bar{\mathbf{x}})^T \mathbf{S}^{-1} (\mathbf{x}_i - \bar{\mathbf{x}})}$$

$$\mathbf{S}_{p \times p} = \mathrm{cov}(\mathbf{X}) = \frac{1}{n-1} (\mathbf{X} - \bar{\mathbf{X}})^T (\mathbf{X} - \bar{\mathbf{X}})$$

T^2 is like a multivariate z-score: it measures how far the observation is from the mean, considering correlations

We monitor if a data point is a multivariate outlier if: T^2 > UCL





Multivariate control charts: Hotelling's T^2

$$\boxed{T_i^2 = (\mathbf{x}_i - \bar{\mathbf{x}})^T \mathbf{S}^{-1} (\mathbf{x}_i - \bar{\mathbf{x}})}$$

$$\mathbf{S}_{p\times p} = \mathrm{cov}(\mathbf{X}) = \frac{1}{n-1}(\mathbf{X} - \bar{\mathbf{X}})^T (\mathbf{X} - \bar{\mathbf{X}})$$

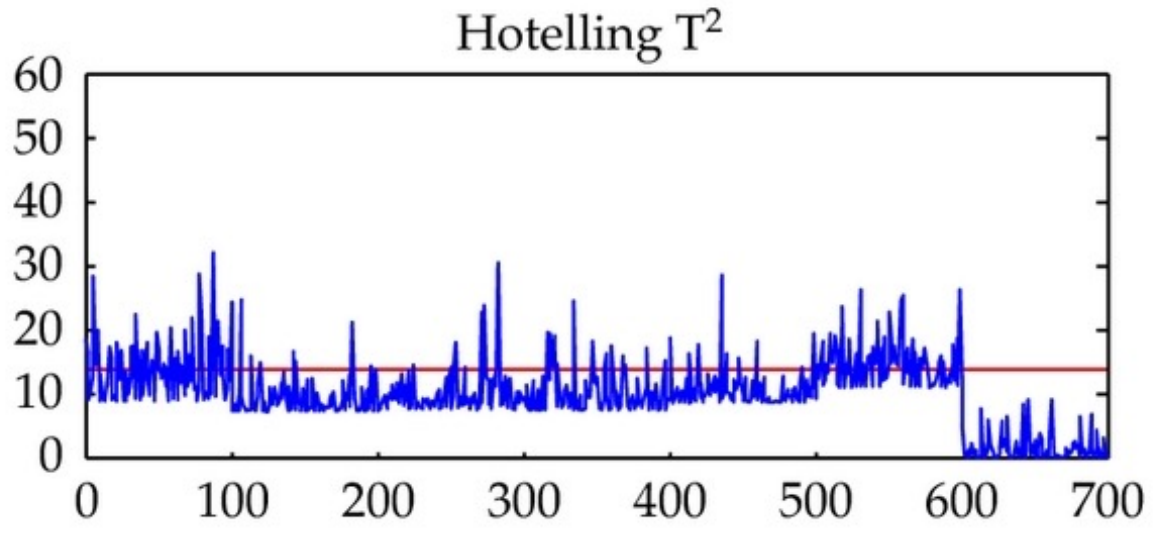
T^2 is like a multivariate z-score: it measures how far the observation is from the mean, considering correlations

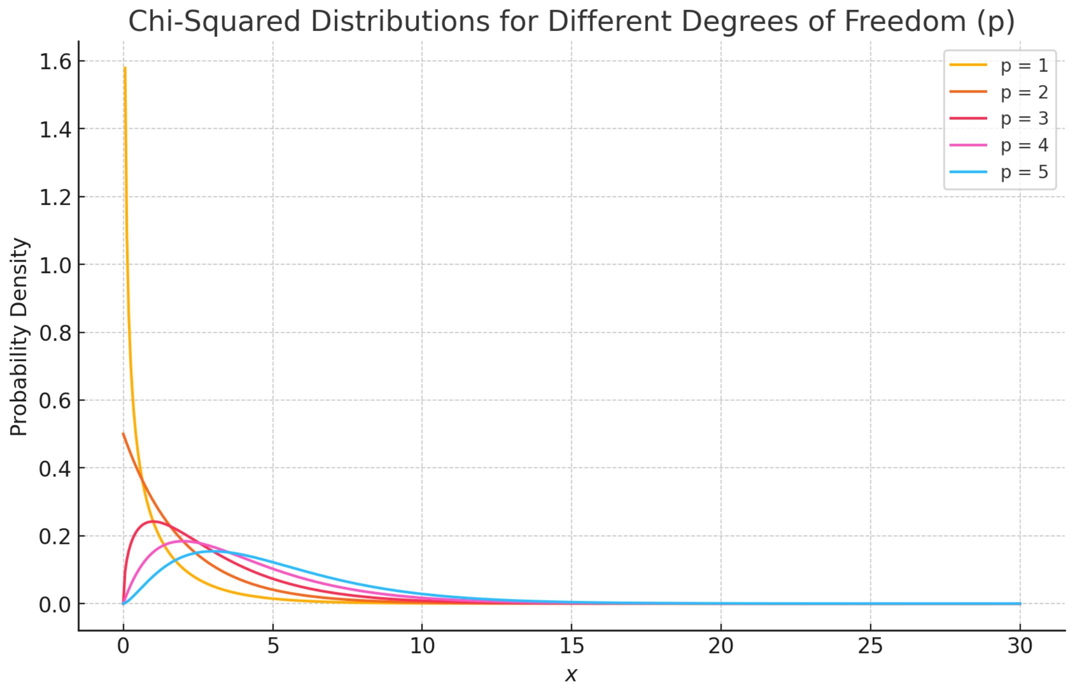
We monitor if a data point is a multivariate outlier if: T^2 > UCL

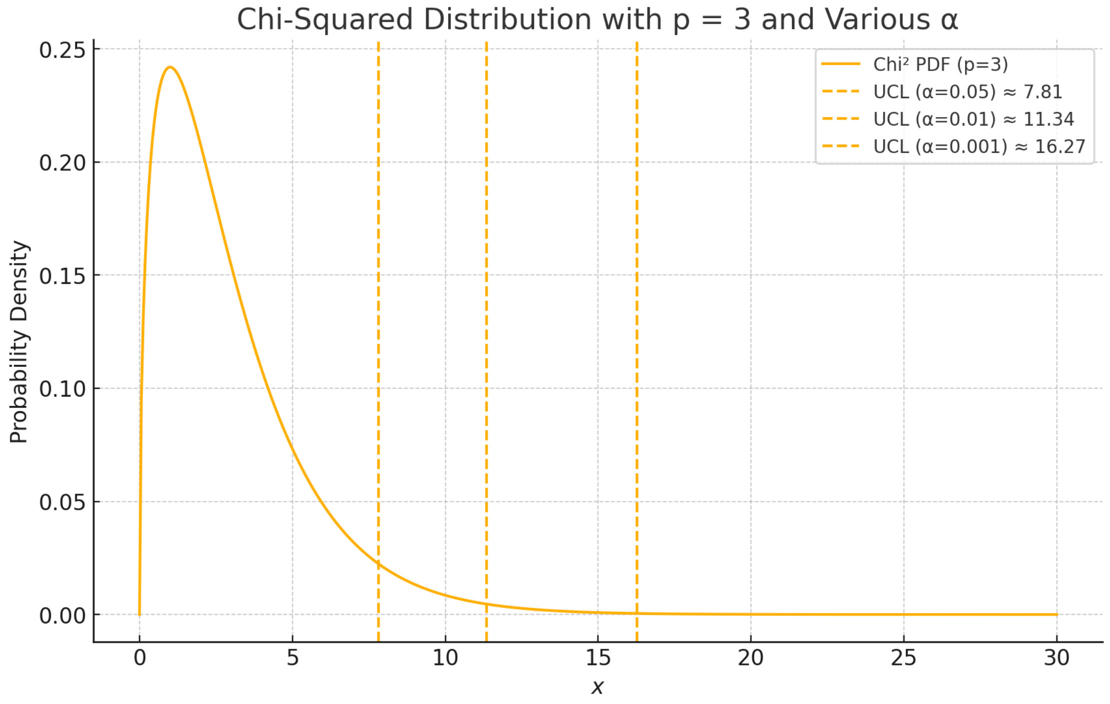
With Gaussian data, UCL is typically computed:

$$\mathrm{UCL} = \chi^2_{p,1-\alpha}$$

- *p* number of variables in the dataset
- α how strict you want to be (common values are 0.05 or 0.01)
- *χ2* is the chi-squared distribution (a value you look up on 'tables')







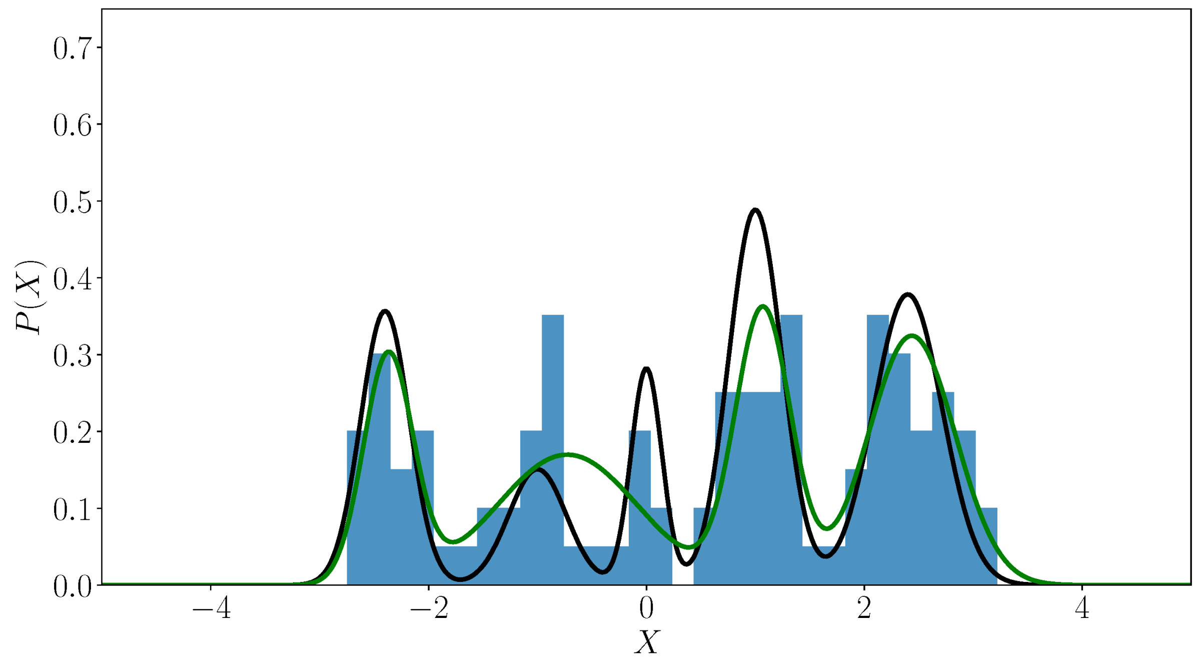
With Gaussian data, UCL is typically computed:

$$\mathrm{UCL} = \chi^2_{p,1-\alpha}$$

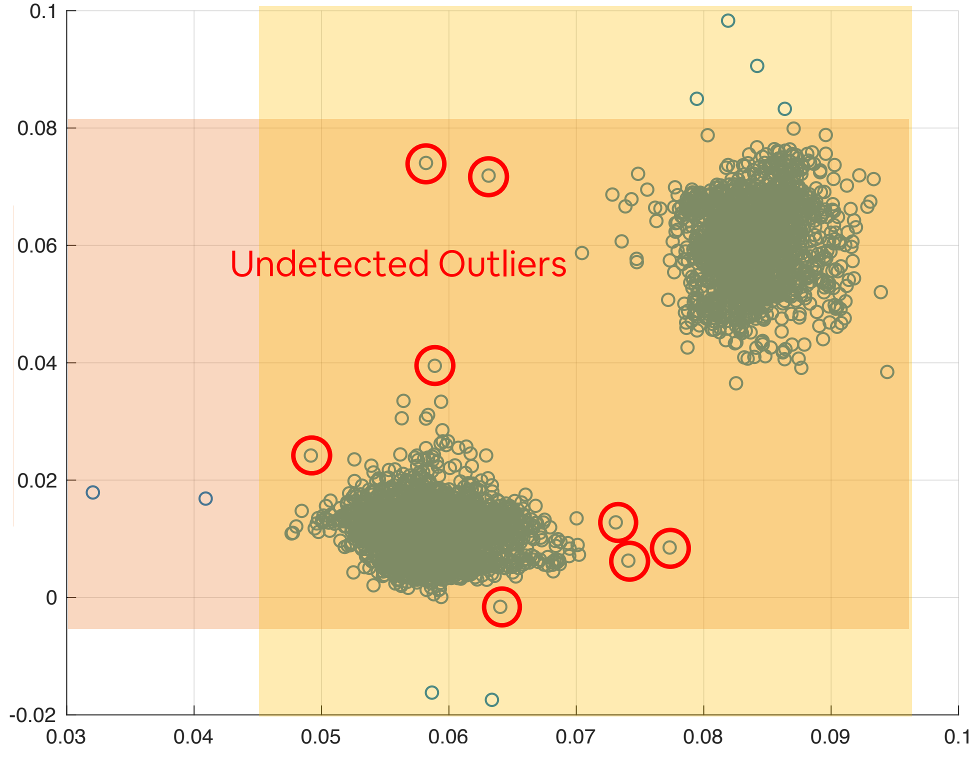
- *p* number of variables in the dataset
- α how strict you want to be (common values are 0.05 or 0.01)
- *χ2* is the chi-squared distribution (a value you look up on ‘tables’)

Limitations of Hotelling's T^2

Variables in real world problems are hardly Gaussian and hardly unimodal (has only one peak or one mode)



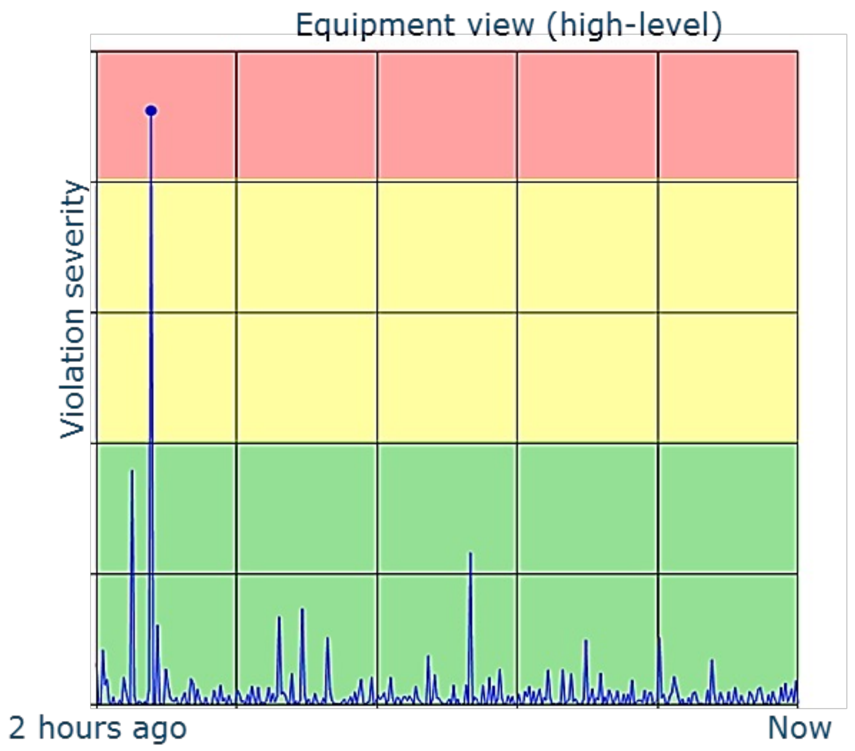
Real manufacturing data!



Multivariate Unsupervised Anomaly Detection

Multivariate Unsupervised AD approaches provide ‘anomaly scores’: unique quantitative indicators able to represent the degree of ‘outlierness’ of complex systems with many variables

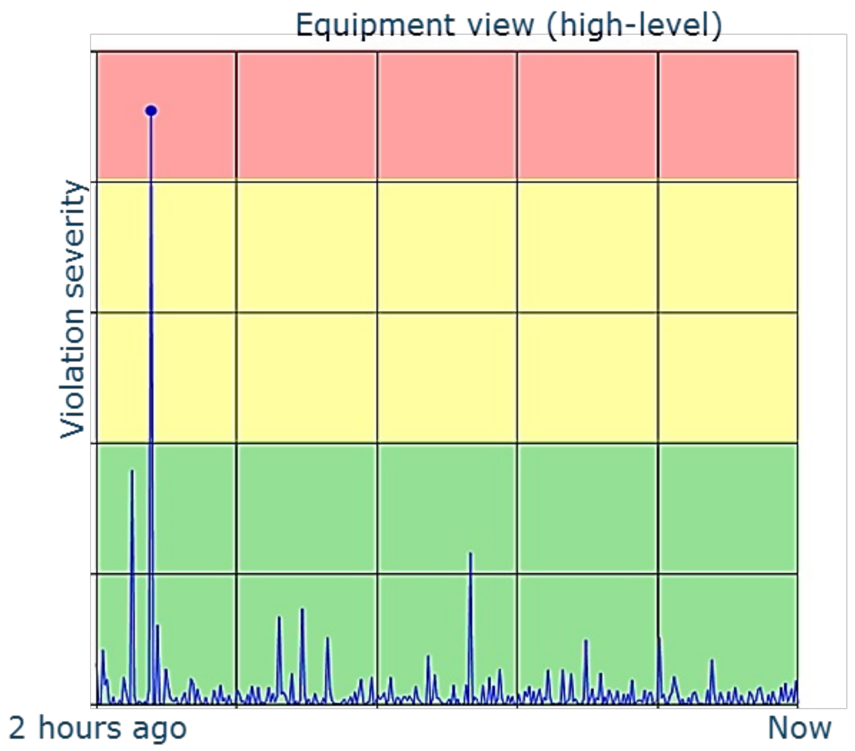
1. No labelled data are required
2. Dozens/Hundreds of sensors variables can be considered at the same time
3. No need for gaussian/unimodal distributions



Multivariate Unsupervised Anomaly Detection

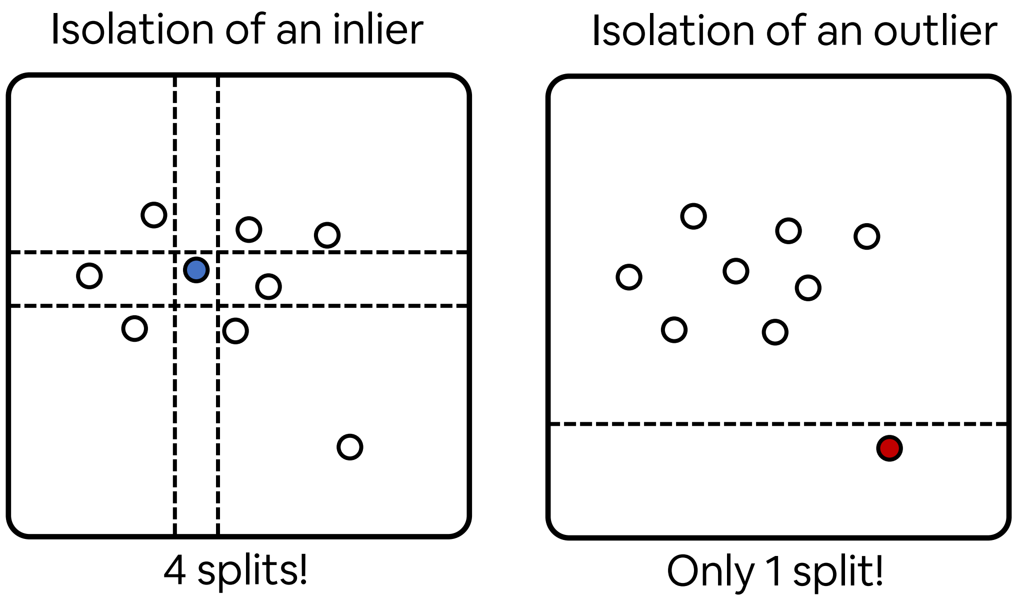
Many approaches for tabular data (data where rows are observations and columns are variables) [2]:

- Density-based methods (e.g. LOF, DBSCAN)
- Distance-based methods (e.g. kNN)
- Clustering-based methods (e.g. CBLOF)
- Neural Networks (e.g. Autoencoder)
- Isolation Forest
- ...



[2] PyOD (Python library for detecting outlying objects) https://pyod.readthedocs.io/en/latest/

Isolation Forest [3]

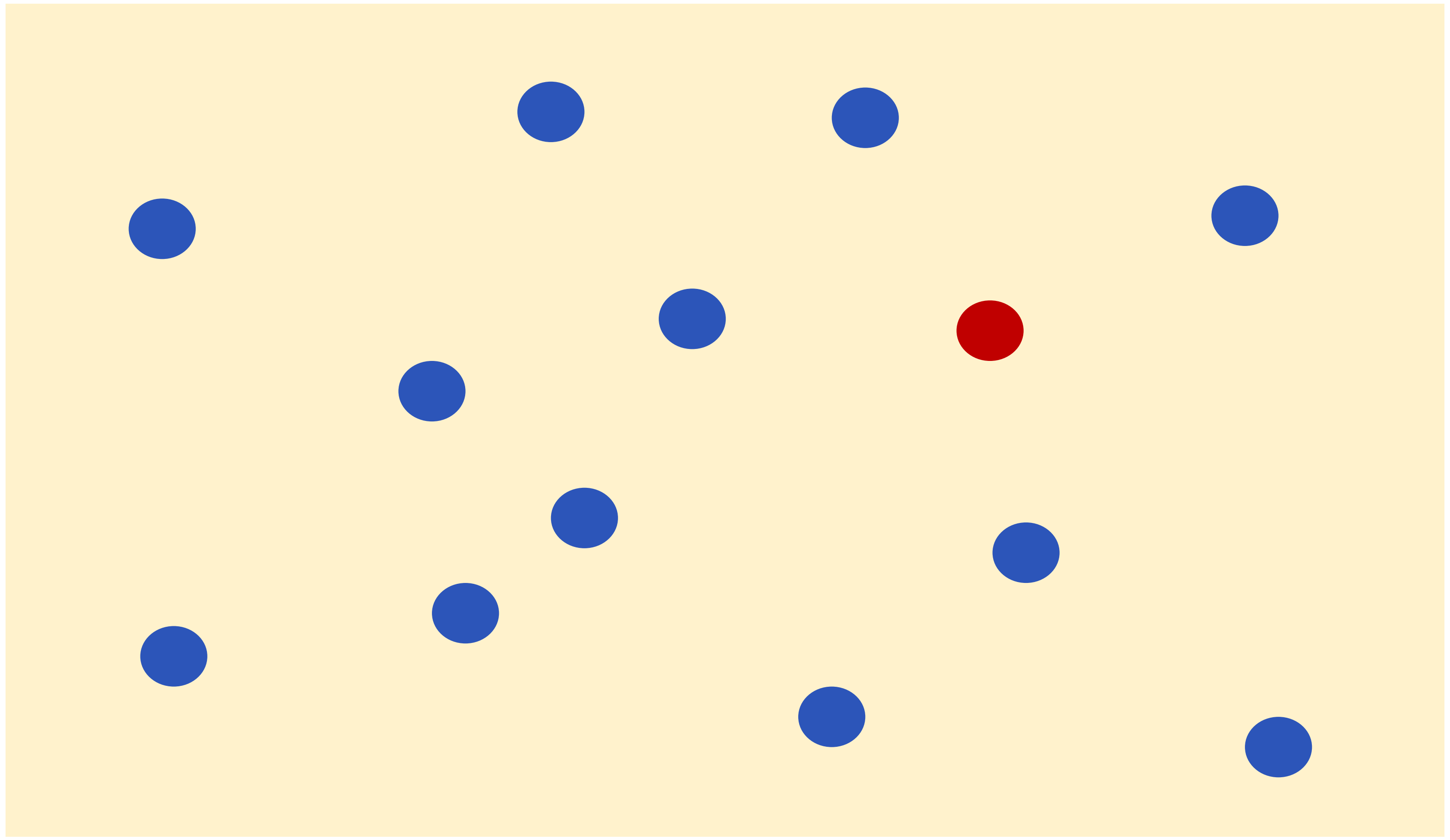


- Efficient algorithm that outperforms other AD methods in several domains [4]
- Based on a partitioning procedure (that creates isolation trees) and on the idea that outlier and inlier are differently affected by such procedure

[3] Liu et al. (2012). Isolation-based anomaly detection. *ACM Trans. on Knowledge Discovery from Data 6*(1), 1-39.
[4] Ma et al. (2023) The need for unsupervised outlier model selection: A review and evaluation of internal evaluation strategies.

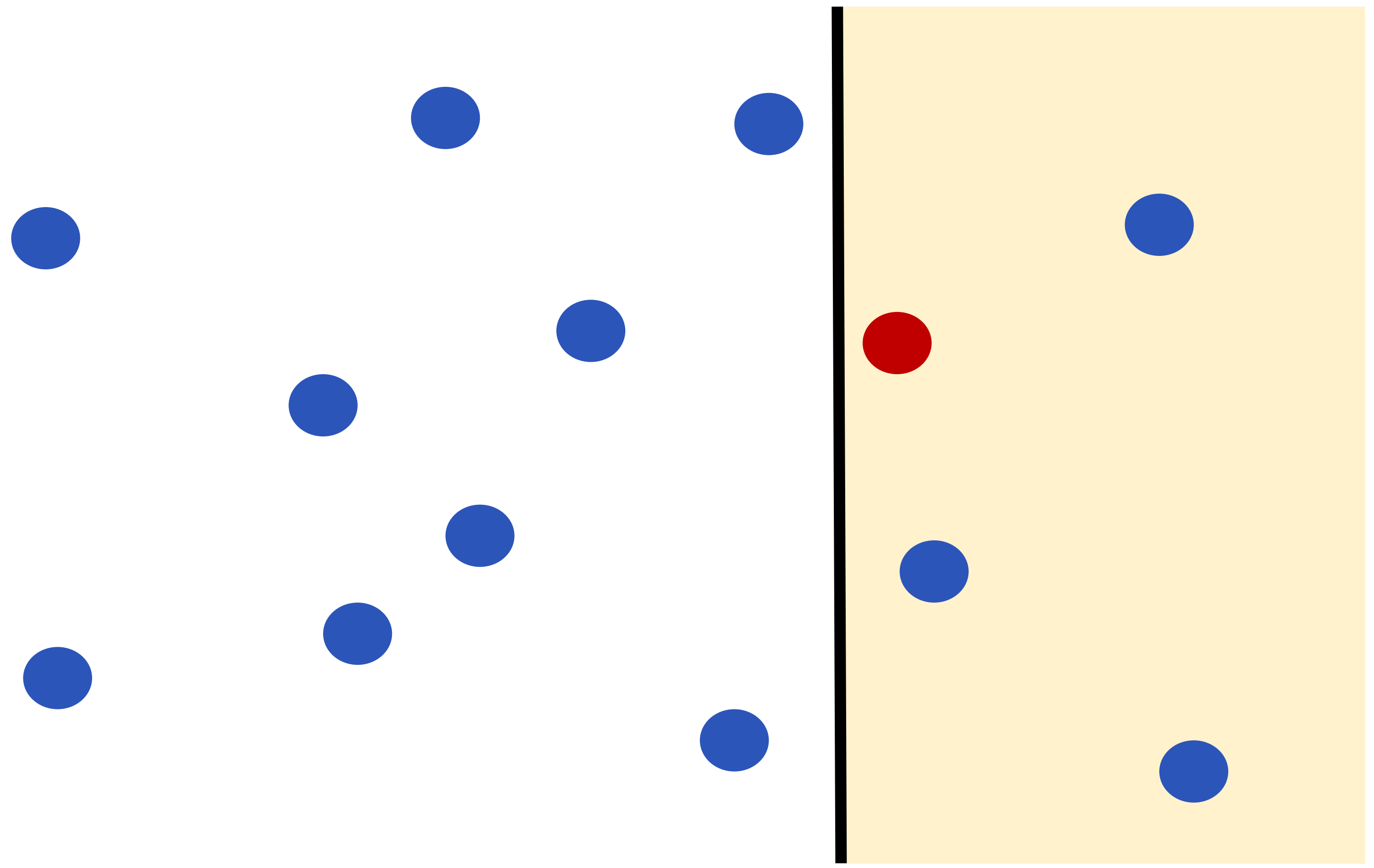
The Isolation Tree

Isolation Forest, is an Anomaly Detection method, based on partitioning



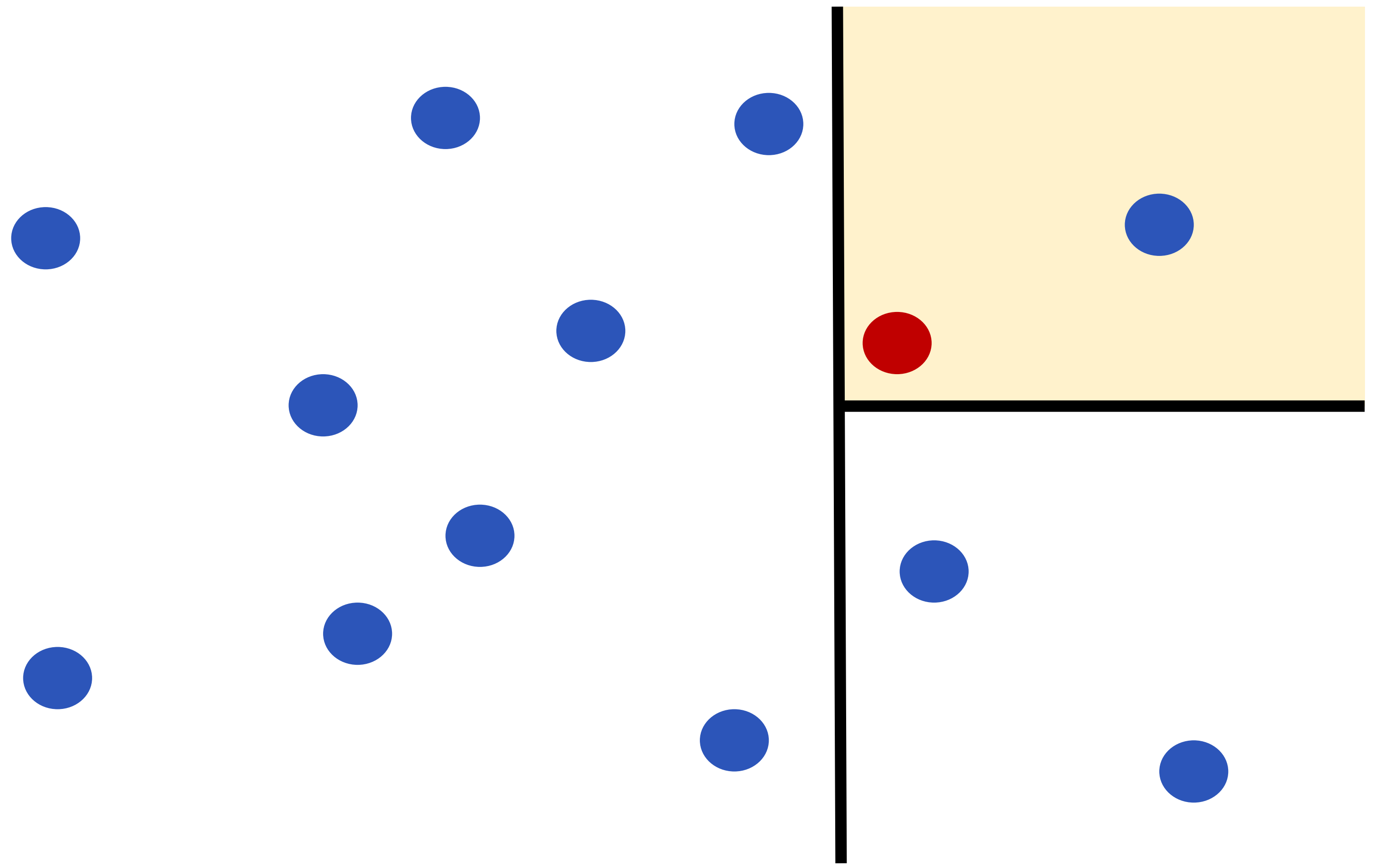
The Isolation Tree

Isolation Forest, is an Anomaly Detection method, based on partitioning



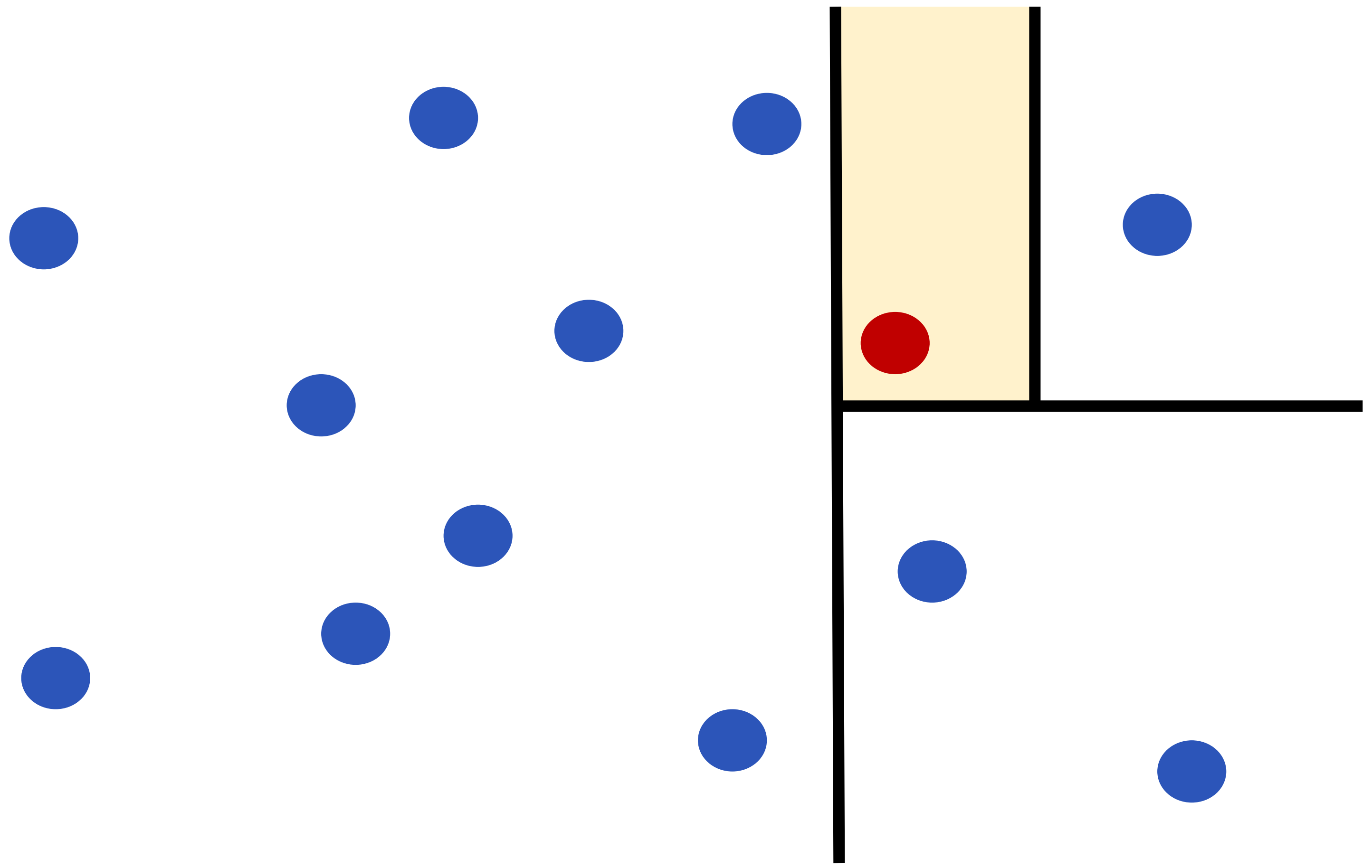
The Isolation Tree

Isolation Forest, is an Anomaly Detection method, based on partitioning

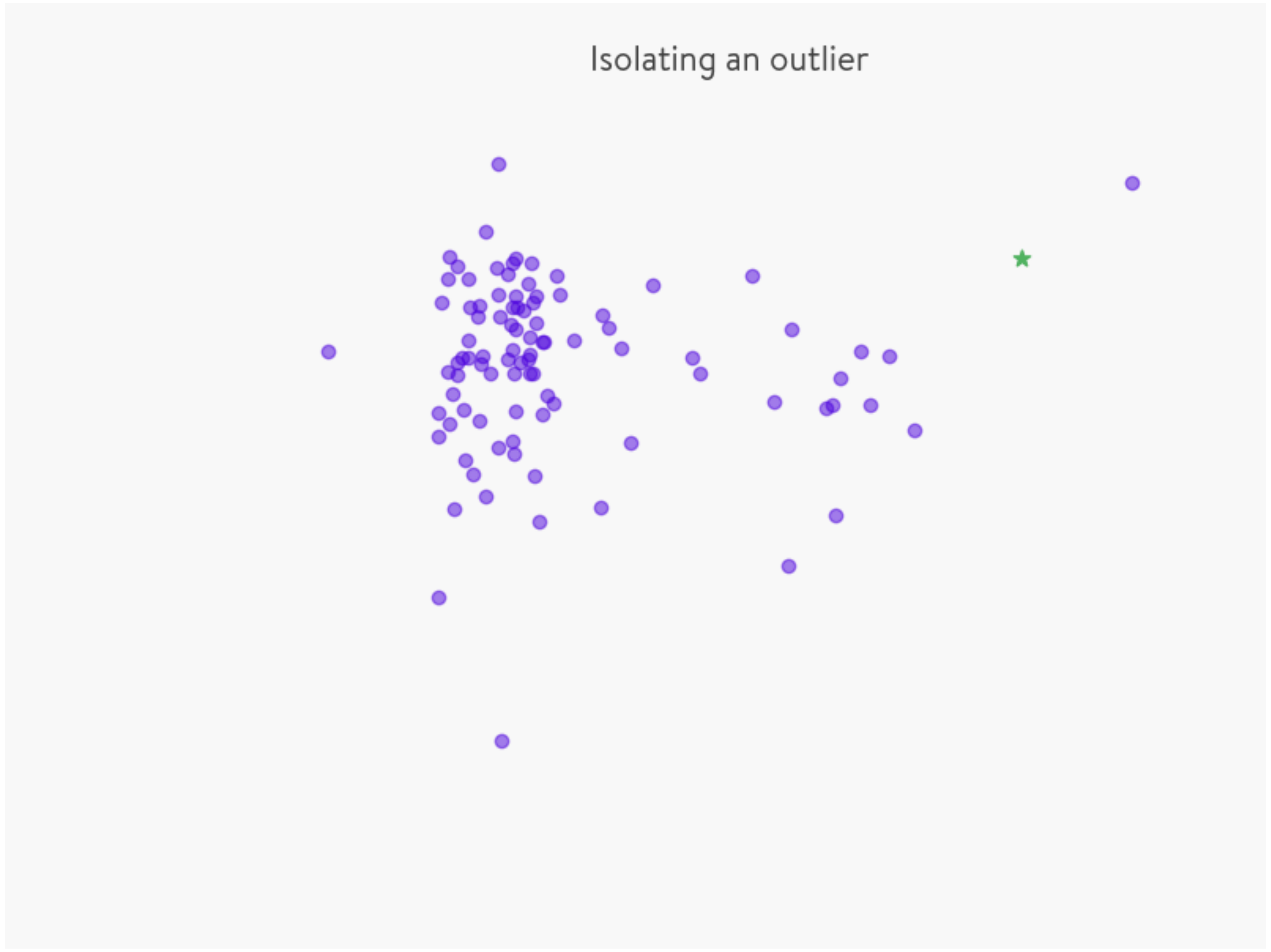


The Isolation Tree

Isolation Forest, is an Anomaly Detection method, based on partitioning



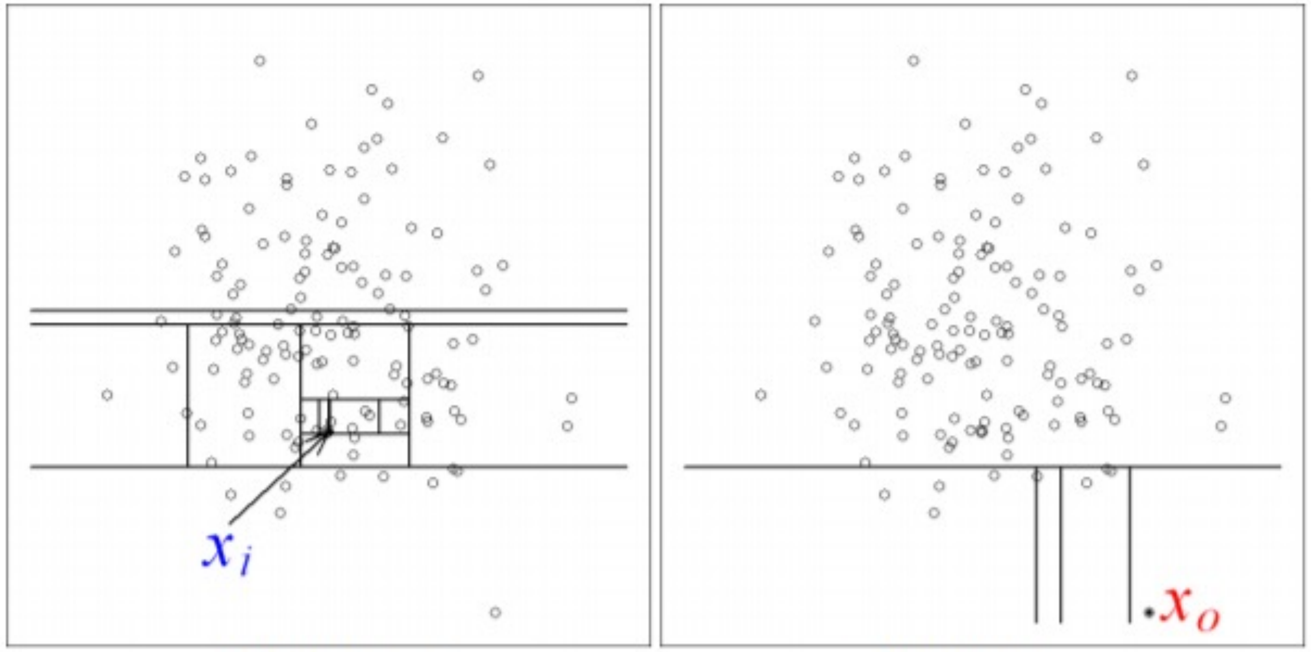
The Isolation Tree



- Efficient algorithm that outperforms other AD methods in several domains [4]
- Based on a partitioning procedure (that creates isolation trees) and on the idea that outlier and inlier are differently affected by such procedure

[3] Liu et al. (2012). Isolation-based anomaly detection. *ACM Trans. on Knowledge Discovery from Data 6*(1), 1-39.
[4] Ma et al. (2023) The need for unsupervised outlier model selection: A review and evaluation of internal evaluation strategies.

The Isolation Tree

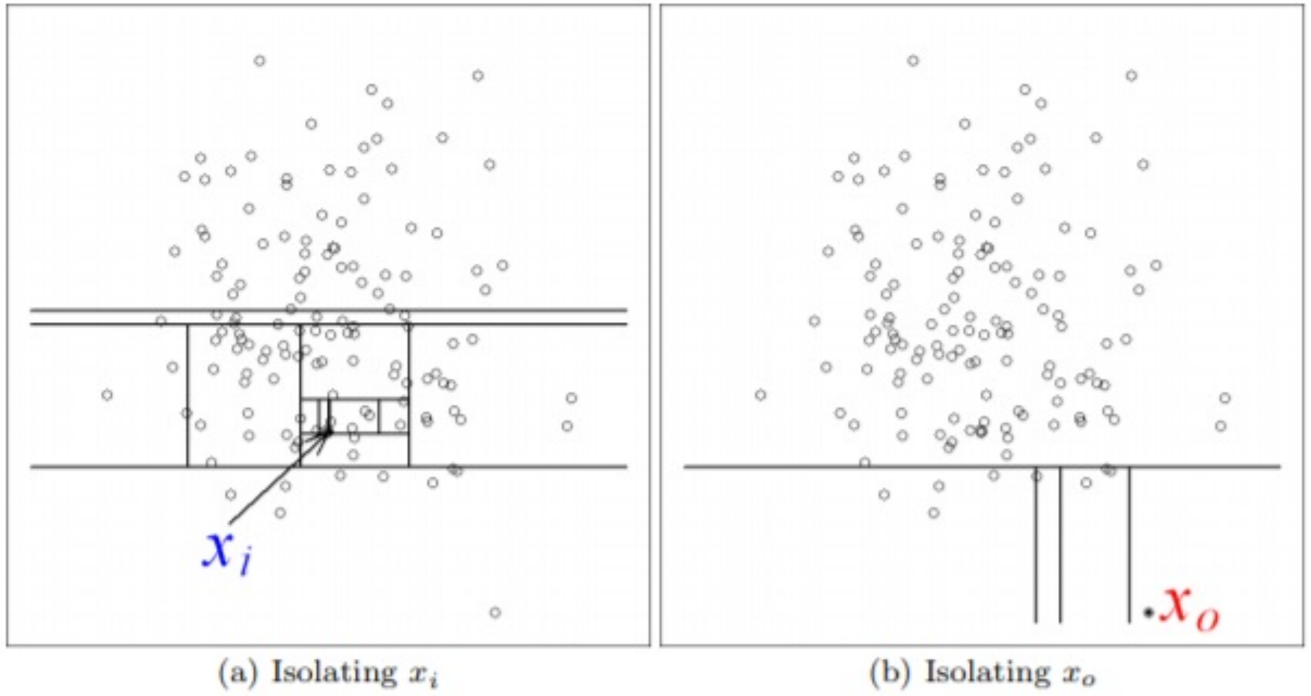


(a) Isolating x_i (b) Isolating x_o

- Efficient algorithm that outperforms other AD methods in several domains [4]
- Based on a partitioning procedure (that creates isolation trees) and on the idea that outlier and inlier are differently affected by such procedure

[3] Liu et al. (2012). Isolation-based anomaly detection. *ACM Trans. on Knowledge Discovery from Data 6*(1), 1-39.
[4] Ma et al. (2023) The need for unsupervised outlier model selection: A review and evaluation of internal evaluation strategies.

The Isolation Tree



- Efficient algorithm that outperforms other AD methods in several domains [4]
- Based on a partitioning procedure (that creates isolation trees) and on the idea that outlier and inlier are differently affected by such procedure

At each iteration, both the choice of the variable and the choice of the split value is random!

[3] Liu et al. (2012). Isolation-based anomaly detection. *ACM Trans. on Knowledge Discovery from Data 6*(1), 1-39.
[4] Ma et al. (2023) The need for unsupervised outlier model selection: A review and evaluation of internal evaluation strategies.

The Isolation Tree: Why So Much Randomness?

Because the goal is not to learn a precise model — it's to separate anomalies from the rest quickly. Here's how randomness helps:

1. Anomalies Are Easier to Isolate

Outliers are rare and very different from normal data → so random splits are more likely to isolate them early in the tree (i.e., shallow depth).

2. No Need to Find "Good" Splits

Unlike decision trees, which aim for optimal splits (e.g., Gini or entropy), iForest is only concerned with how many splits it takes to isolate a point. So random is fine — even ideal.

Aspect	Why Random?
Feature selection	To avoid bias and explore all dimensions
Split value selection	To simplify and speed up isolation

The Isolation Tree: Why So Much Randomness?

Because the goal is not to learn a precise model — it's to separate anomalies from the rest quickly. Here's how randomness helps:

1. Anomalies Are Easier to Isolate

Outliers are rare and very different from normal data → so random splits are more likely to isolate them early in the tree (i.e., shallow depth).

2. No Need to Find "Good" Splits

Unlike decision trees, which aim for optimal splits (e.g., Gini or entropy), iForest is only concerned with how many splits it takes to isolate a point. So random is fine — even ideal.

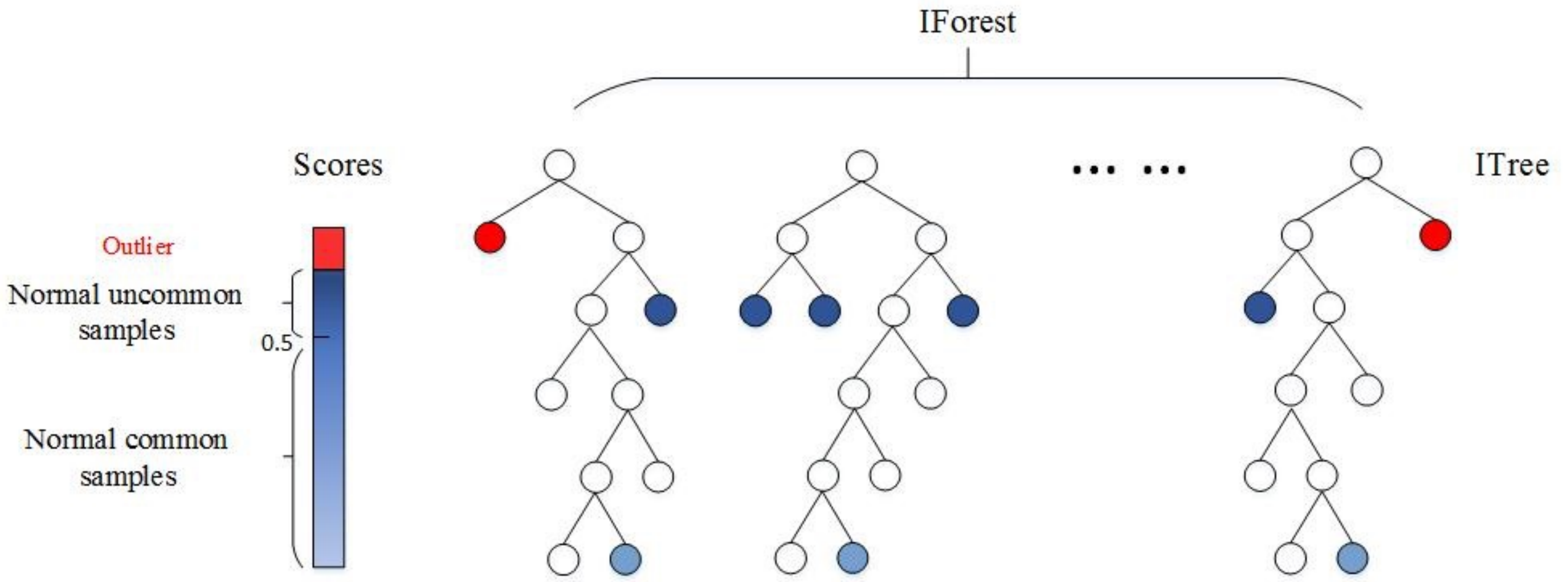
3. Ensemble Averaging Smooths Noise

By using many trees with different random splits, we average out the randomness and get a robust anomaly score.

Aspect	Why Random?
Feature selection	To avoid bias and explore all dimensions
Split value selection	To simplify and speed up isolation
Multiple trees	To make randomness statistically meaningful

Isolation Forest

An ensemble approach: anomaly score computed as mean of the depth over the various isolation trees



Isolation Forest: step-by-step

1. Subsampling the Data

- From the full dataset, randomly sample a subset of data points (e.g. 256 points)
- This is done to reduce computation and improve generalization.

2. Build Isolation Trees (iTrees)

For each sampled subset, build a binary tree recursively with these rules:

✅ At each node:

- Randomly select a feature (column)
- Randomly select a split value between the min and max of that feature in the current data.

🚫 Stop splitting when:

- The node has only one data point, or
- A max depth is reached (usually $\lceil \log_2(n) \rceil$, where n is sample size)

➡️ Result: Each path from root to leaf isolates a data point.

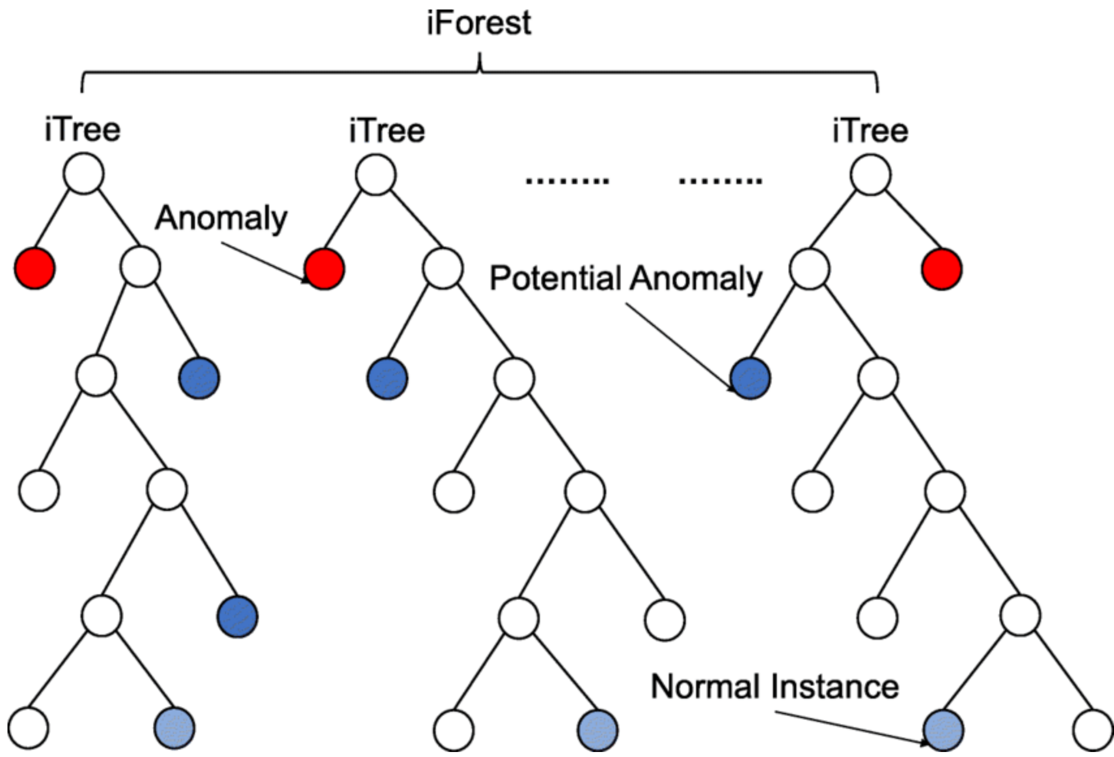
Isolation Forest: step-by-step

<u>3. Repeat and Build a Forest</u>

- Build many trees (typically 100), each on a different random sample
- The randomness ensures diverse ways to isolate points

<u>4. Compute Path Lengths</u>

- For each data point in the original dataset:
- Send it down every tree.
- Record the number of splits needed to isolate it (i.e., how deep it ends up in each tree).



Isolation Forest: step-by-step

5. Calculate Anomaly Score

- For each point, compute the average path length across all trees.
- Convert that to an anomaly score using:

$$\boxed{s(x,n) = 2^{-\frac{E(h(x))}{c(n)}}}$$

- $E(h(x))$ is the average path length for point x,
- c(n) is the average path length of an unsuccessful search in a Binary Search Tree (a normalization factor).

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n} \qquad H(n) = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$$

Isolation Forest: step-by-step

5. Calculate Anomaly Score

- For each point, compute the avera
trees.
- Convert that to an anomaly score

$$\boxed{s(x,n) = 2^{-\frac{E(h(x))}{c(n)}}}$$

We need to compare path lengths across datasets of different sizes.
A short path in a small dataset is not the same as a short path in a big one.
So, we normalize each path length using c(n) to make the anomaly score comparable.

- $E(h(x))$ is the average path length for point x,
- c(n) is the average path length of an unsuccessful search in a Binary Search Tree (a normalization factor).

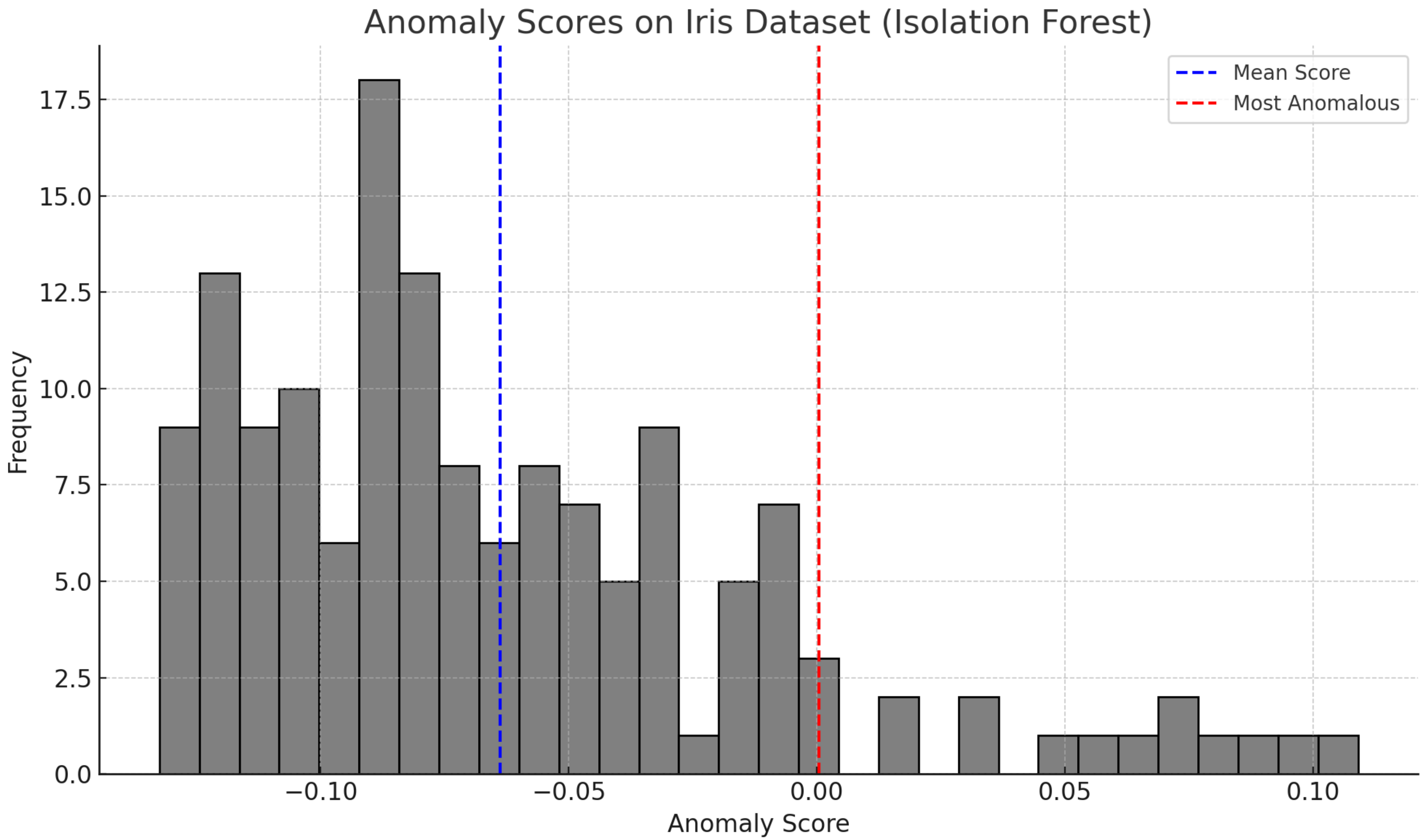
$$c(n) = 2H(n-1) - \frac{2(n-1)}{n} \qquad H(n) = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$$

Isolation Forest: step-by-step

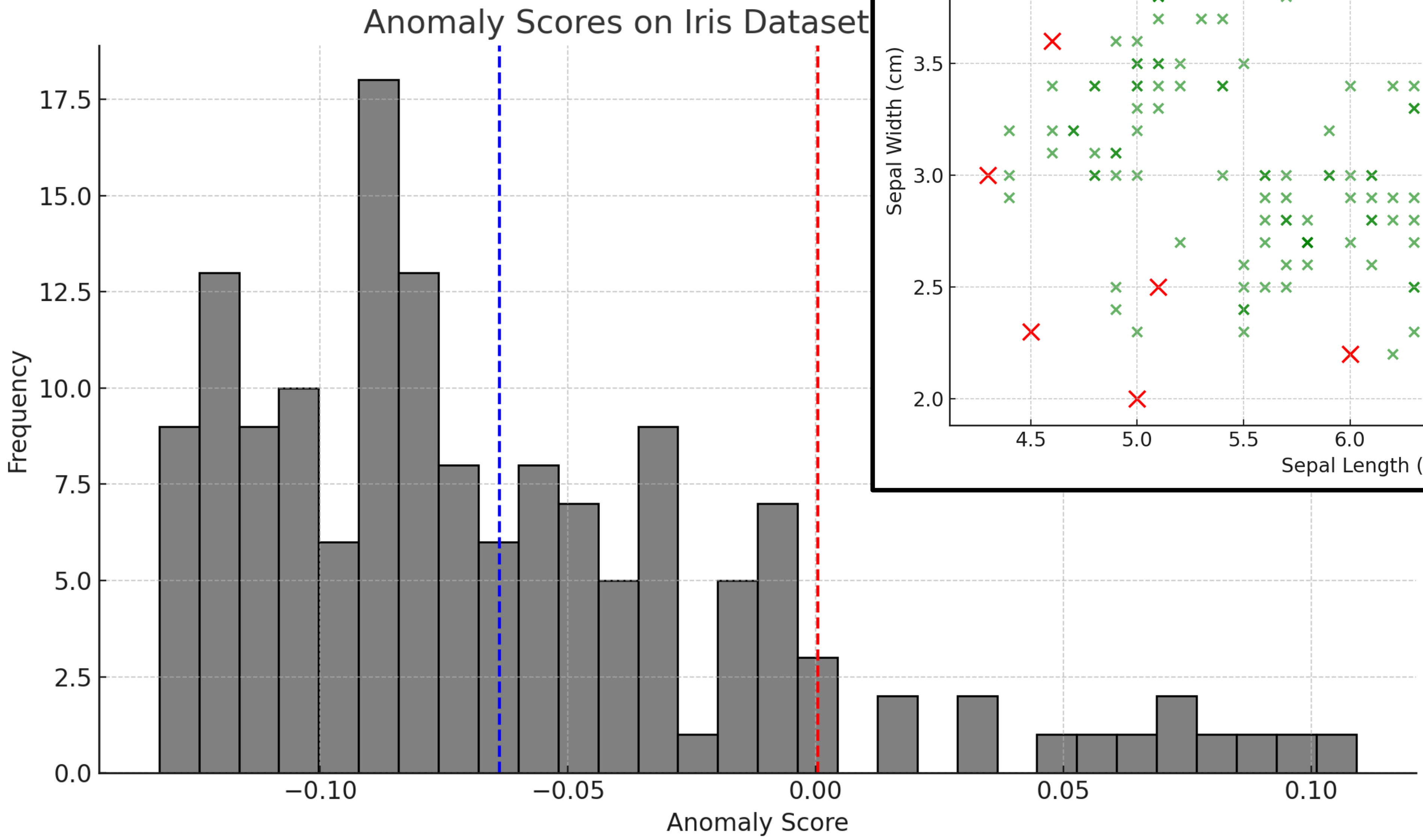
6. Compare the anomaly score with a threshold τ

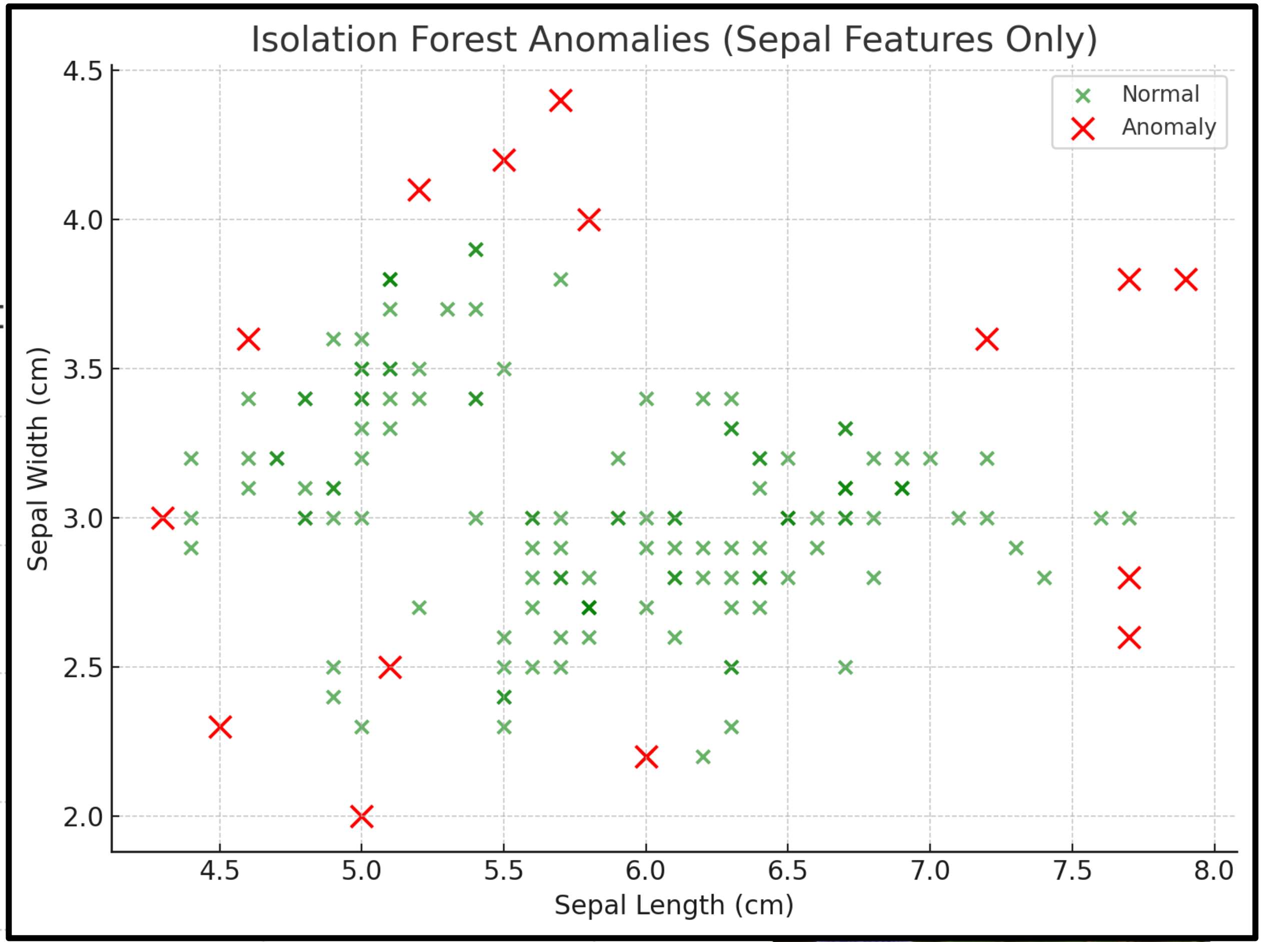
- If the anomaly score is above a given threshold, we can declare a data point anomalous
- A way to derive the threshold is to consider the contamination: a parameter that is a proportion (a float between 0 and 0.5, usually) that indicates the expected fraction of outliers in the dataset
- Contamination can be considered a tuning knob between false positive (declared anomalies that are ‘normal’ data) and false negatives (not declared anomalies that should have been segnalated)

Iris dataset

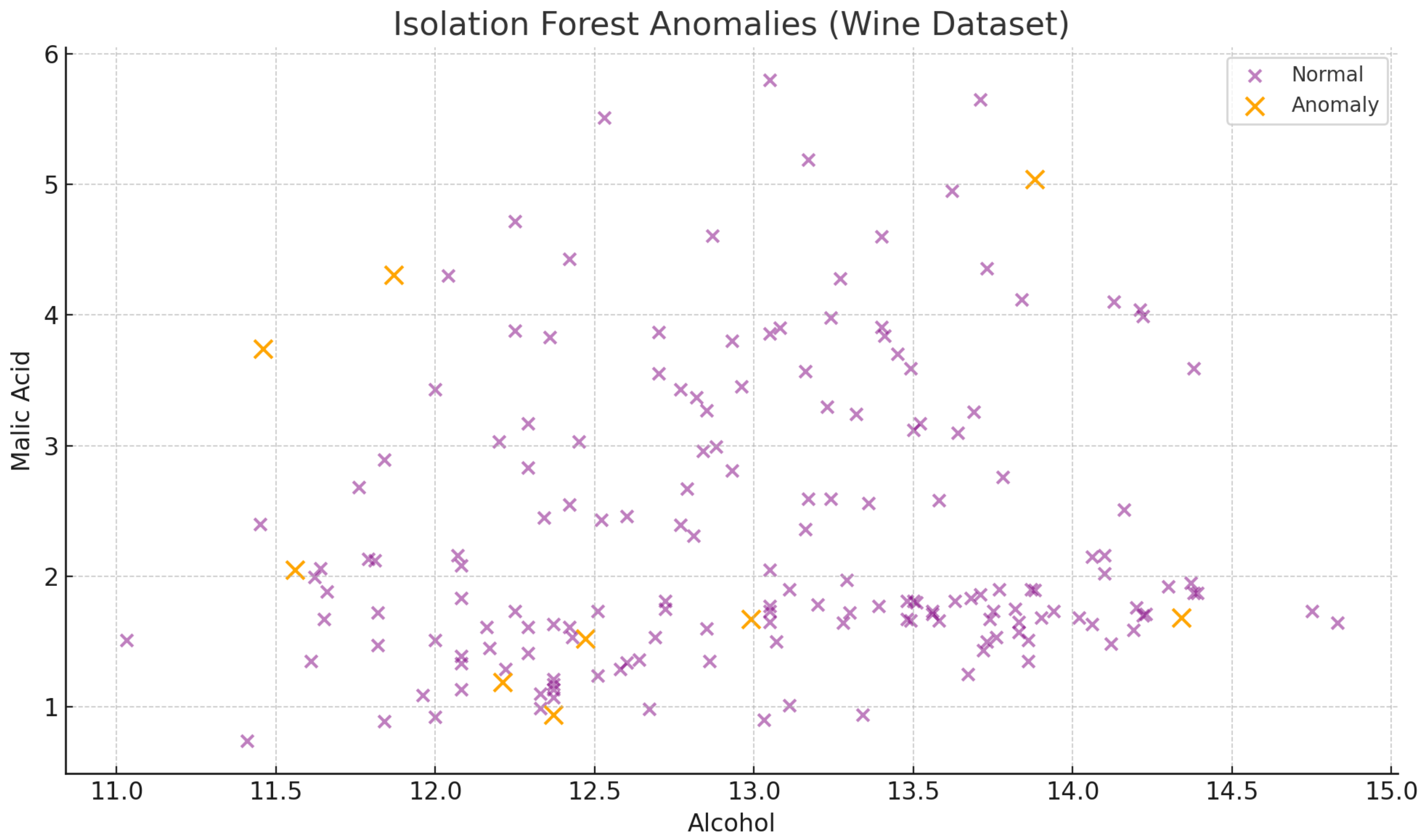


Iris dataset

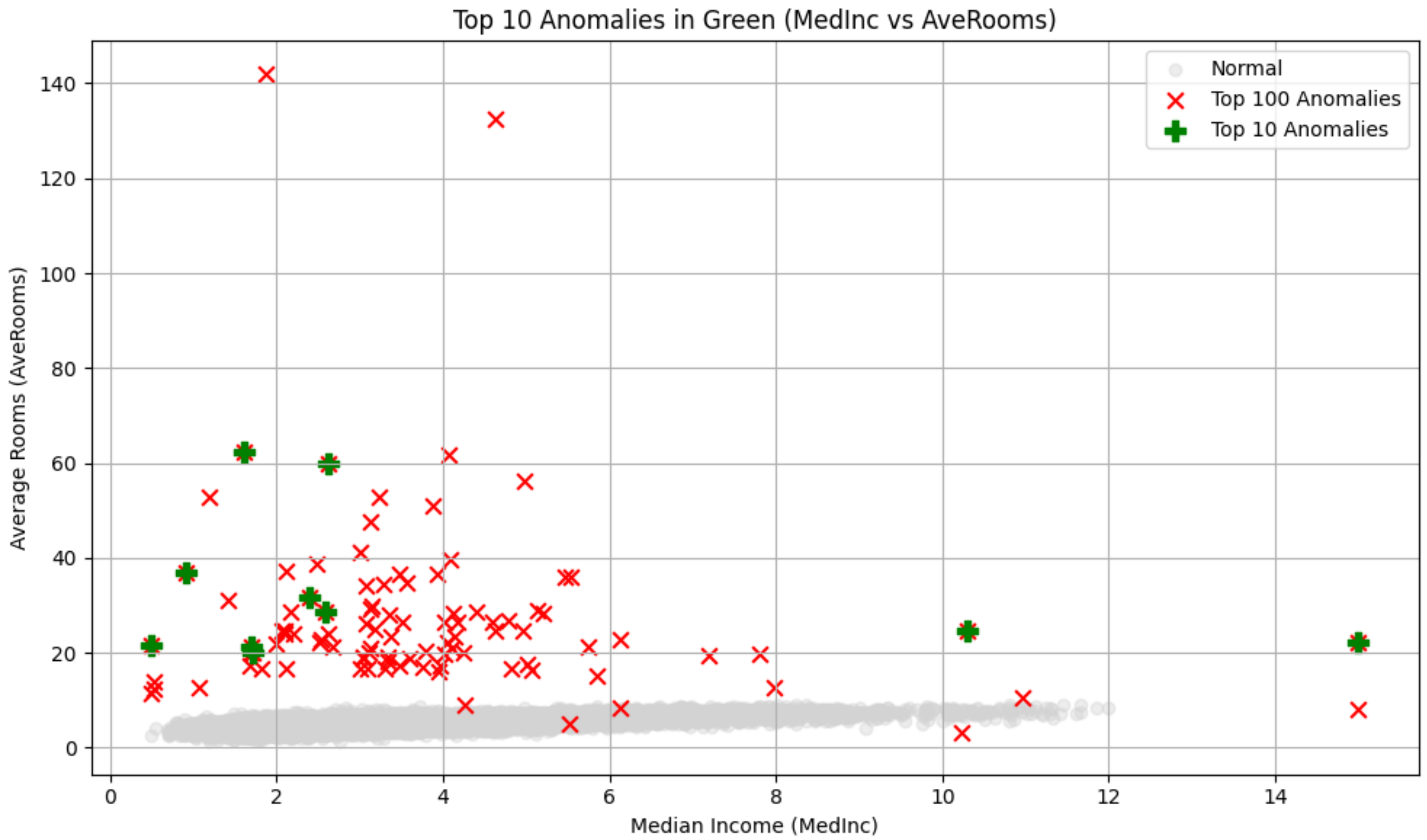




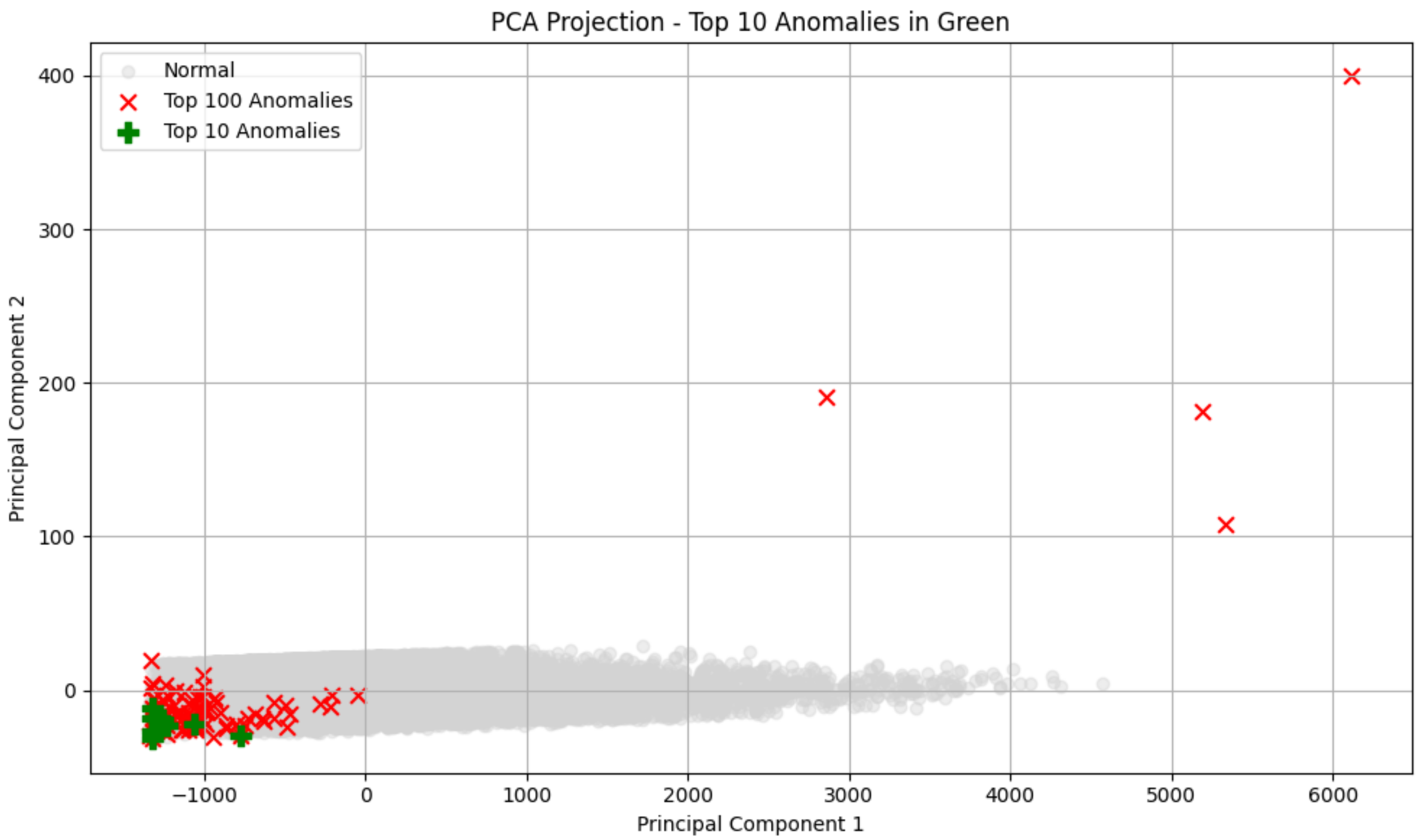
Wine dataset



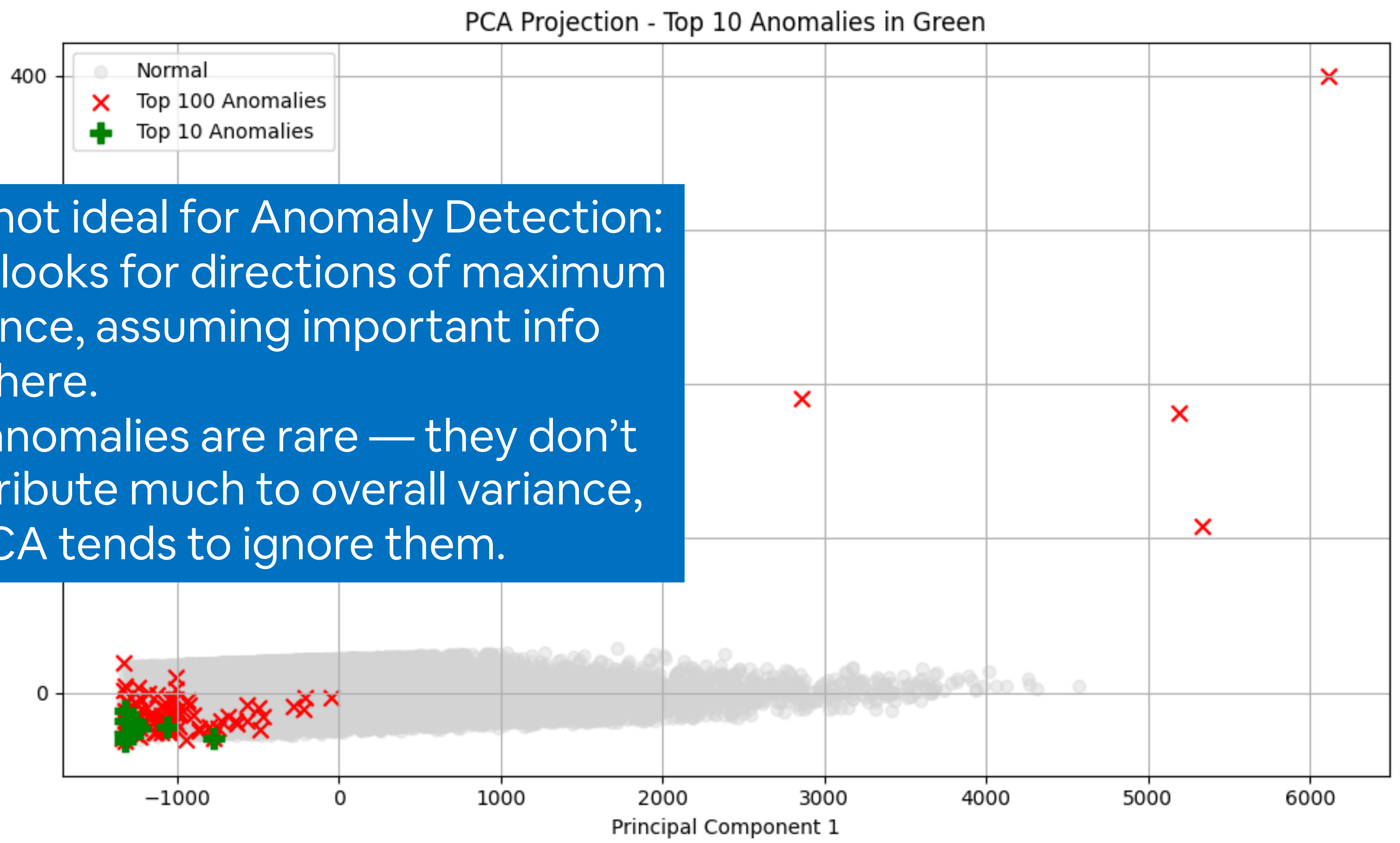
California housing dataset



California housing dataset



California housing dataset



PCA is not ideal for Anomaly Detection:

- PCA looks for directions of maximum variance, assuming important info lies there.
- But anomalies are rare — they don't contribute much to overall variance, so PCA tends to ignore them.

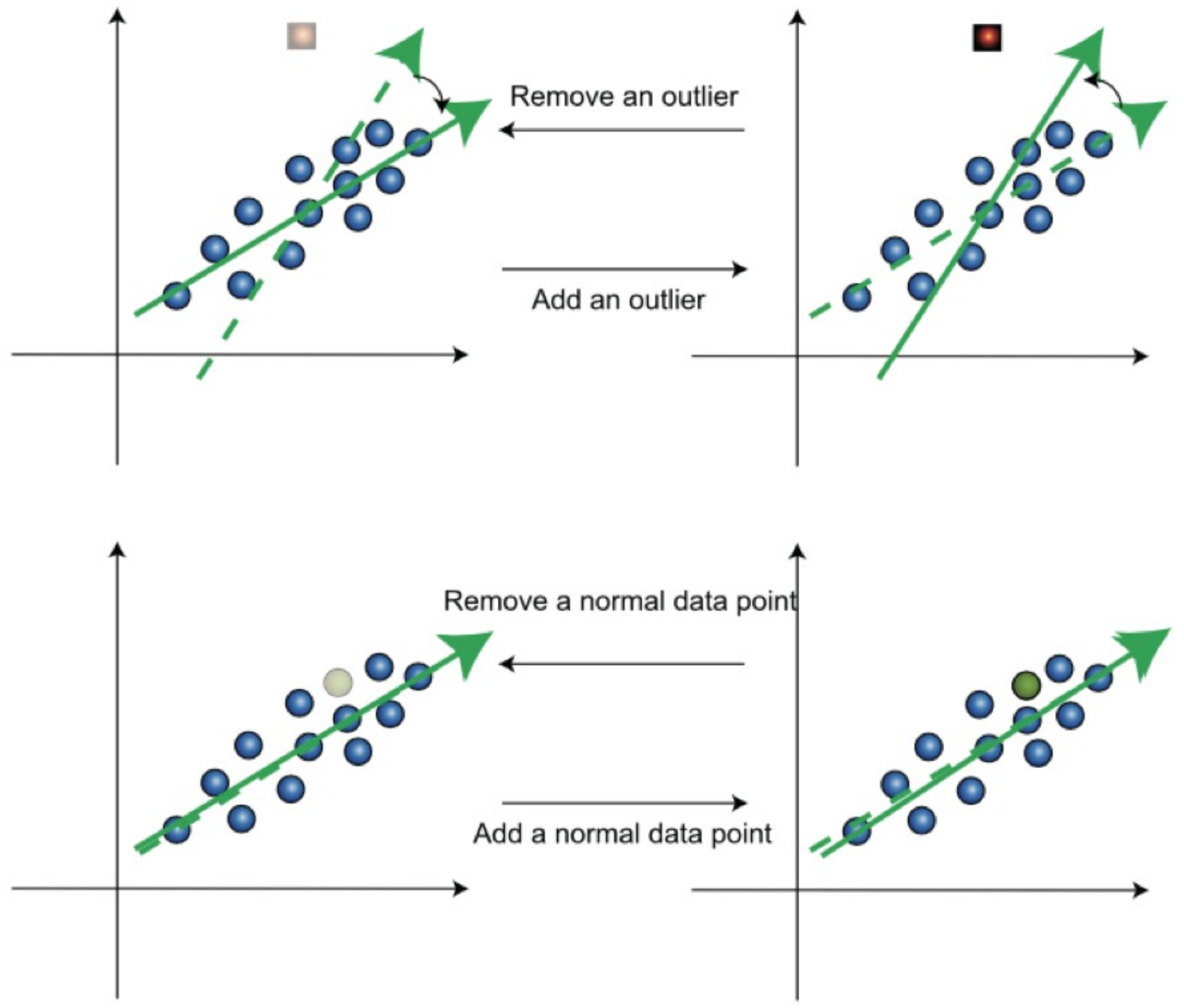
Oversampling PCA: osPCA

Idea:

- principal directions represent "normal" attributes
- outliers change directions of principal components

To make it effective, each data sample, when checked if it changed the direction of the first PC, is oversampled (replicated many times)

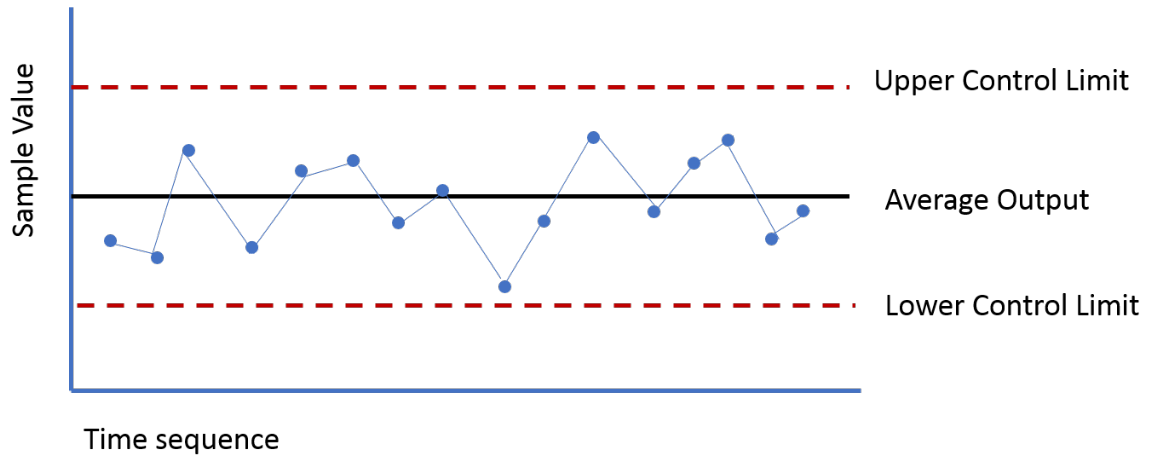
Lee, Yuh-Jye, Yi-Ren Yeh, and Yu-Chiang Frank Wang. "Anomaly detection via online oversampling principal component analysis.", 2013.

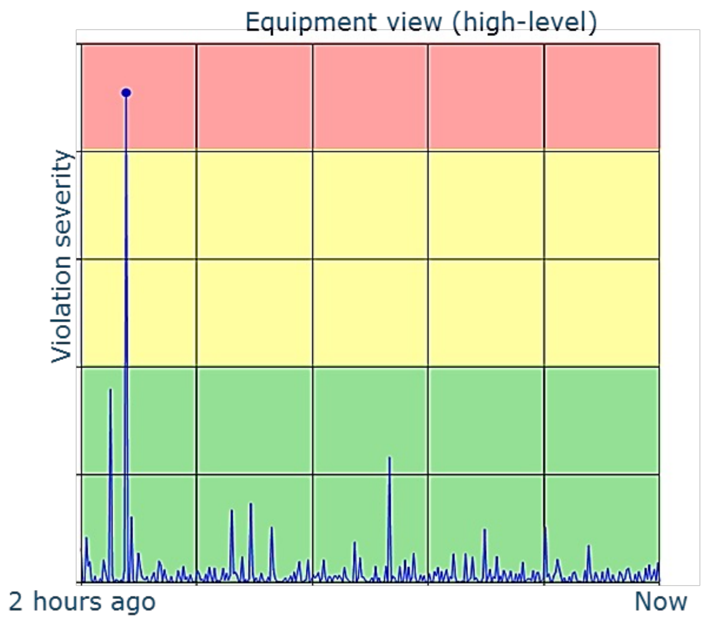


Univariate Control Chart vs Unsupervised AD

To enable Decision Making information should be:

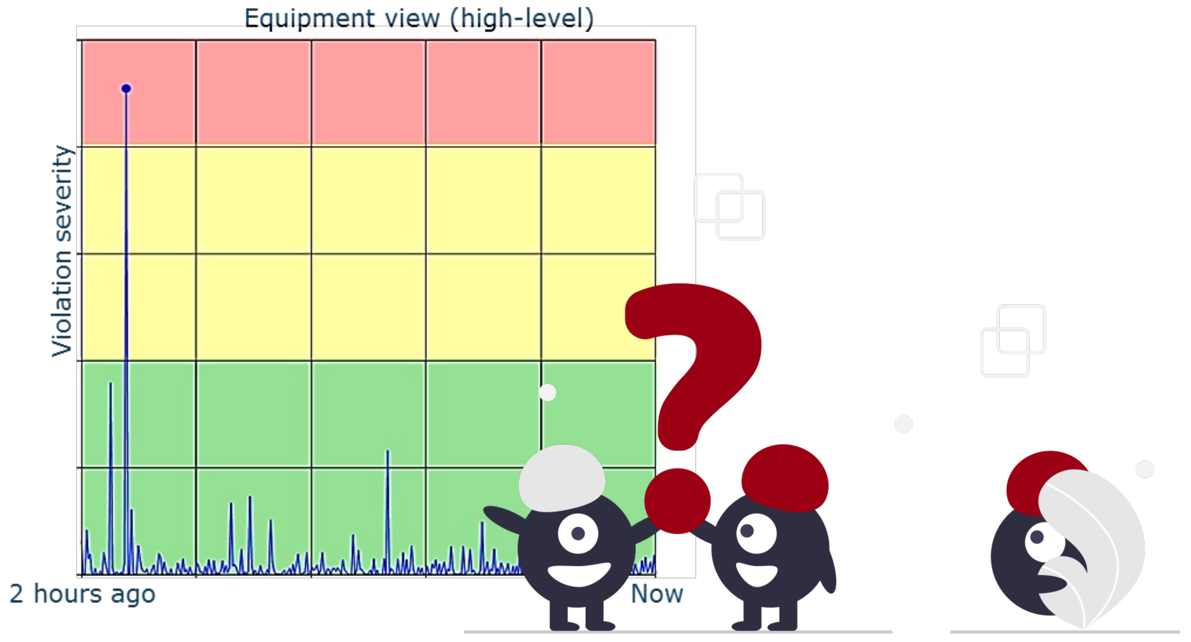
✗	complete	✓
✗	concise	✓
✓	interpretable	✗

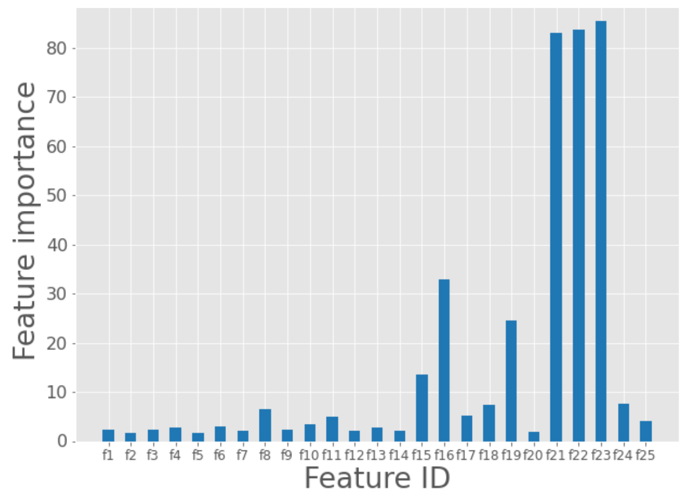




I’ve got the Anomaly Score: now, what?

- Thanks to the Anomaly score users are alerted of potential anomalous situation, however it is up to them to discover potential troubles
- It would be nice to ease the Root Cause Analysis to provide additional information, like feature rankings...



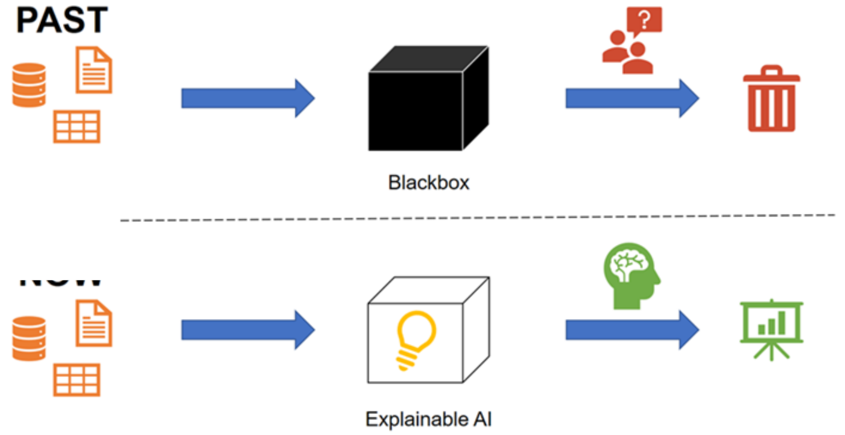


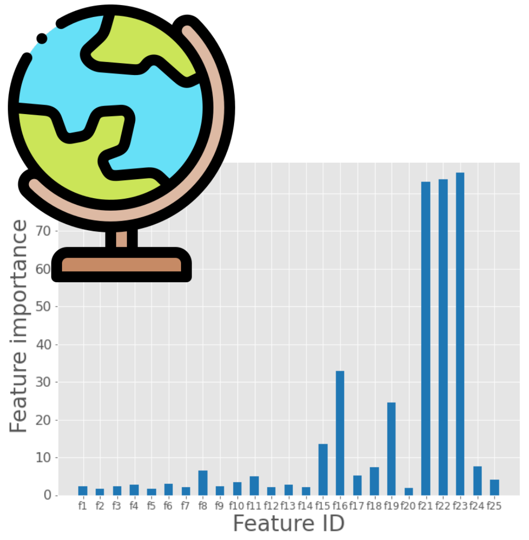
Depth-based Isolation Forest Feature Importance (DIFFI) [5]

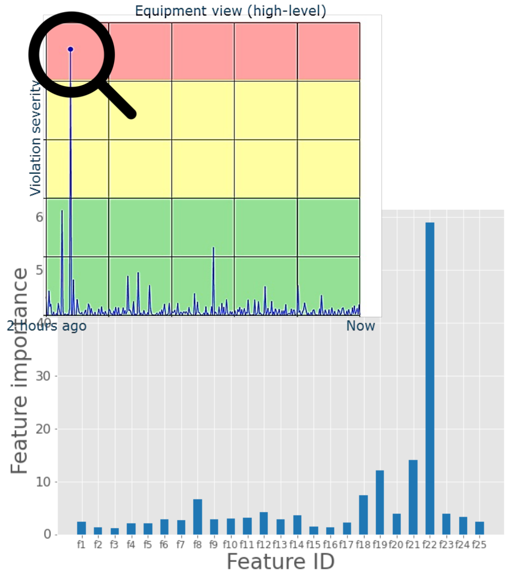
DIFFI is an eXplainable Artificial Intelligente (XAI) approach designed for the Isolation Forest

DIFFI provides a variable ranking for:

- Global Explainability (ie. what variables are important for the whole Isolation Foreset model)
- Local Explainability (ie. what variables are important for a particular prediction)

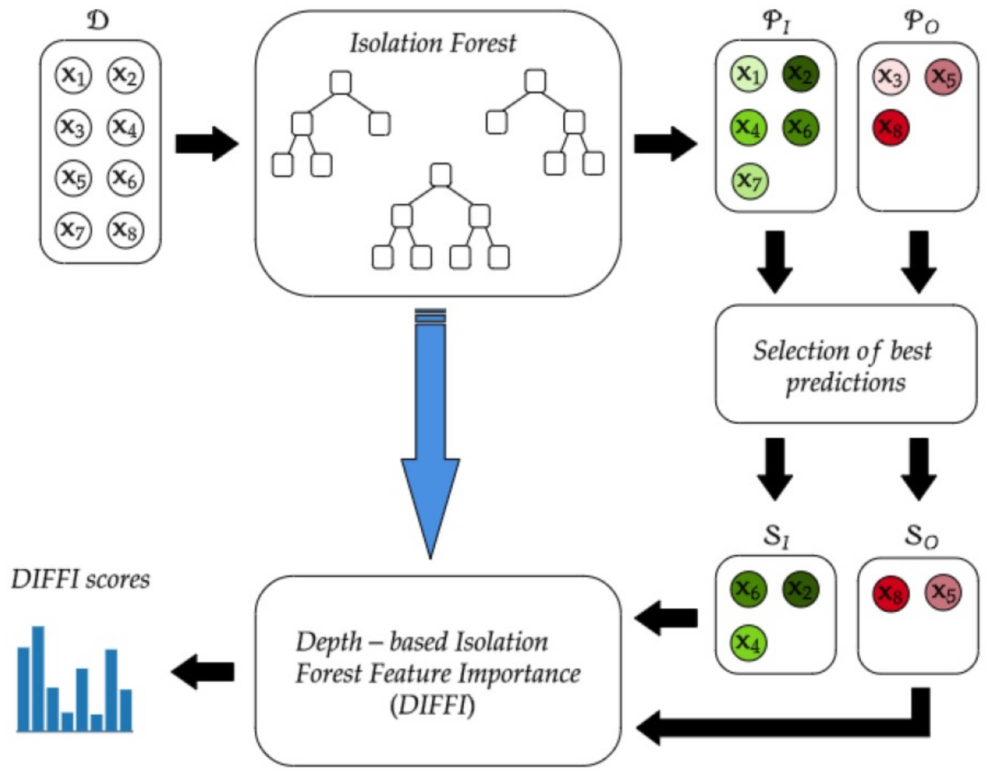






[5] Carletti, M., Terzi, M., & Susto, G. A. (2023). Interpretable Anomaly Detection with DIFFI: Depth-based feature importance of Isolation Forest. *Engineering Applications of Artificial Intelligence*, *119*, 105730.

Depth-based Isolation Forest Feature Importance (DIFFI) [5]



$\mathcal{D}$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8
Isolation Forest
$\mathcal{P}_I$
$\mathcal{P}_O$
Selection of best predictions
$\mathcal{S}_I$
$\mathcal{S}_O$
DIFFI scores
Depth – based Isolation Forest Feature Importance (DIFFI)

DIFFI provides a variable ranking that:

- Does not require true labels (other XAI approaches do!)
- Low computational cost
- No tuning

IDEA: mark a feature as "important" if

- it induces isolation of outliers at small depths (i.e. near the root)
- At the same time, does not contribute to the isolation of inliers

[5] Carletti, M., Terzi, M., & Susto, G. A. (2023). Interpretable Anomaly Detection with DIFFI: Depth-based feature importance of Isolation Forest. *Engineering Applications of Artificial Intelligence*, *119*, 105730.

For technical details

Dashboards Ride Monitoring with Anomaly Detection

Boss Statwolf

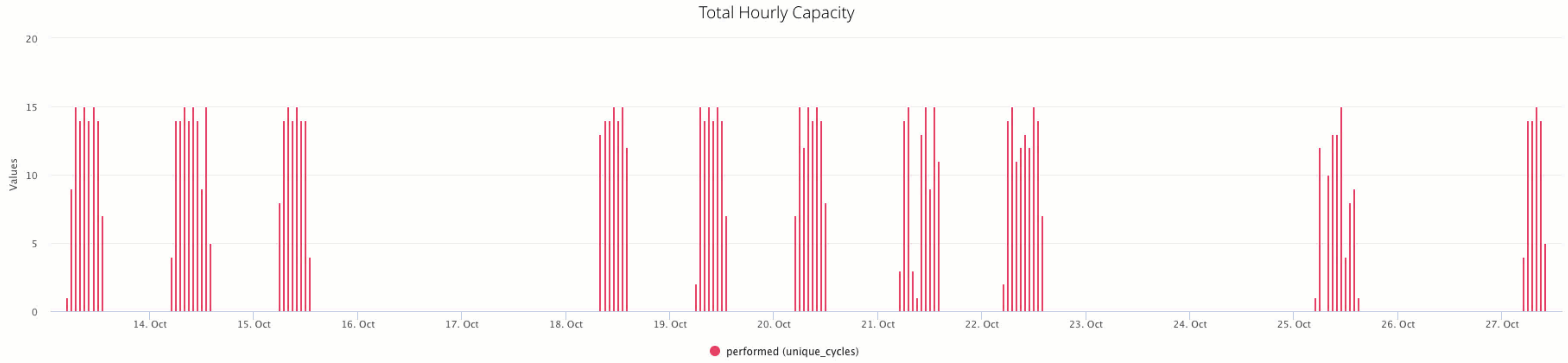
Timeframe

All Time

Capacity

Capacity is measured in number of cycles.

980	980	0	0
Total Cycles	**Performed Cycles**	**Stopped Cycles**	**Estop Cycles**

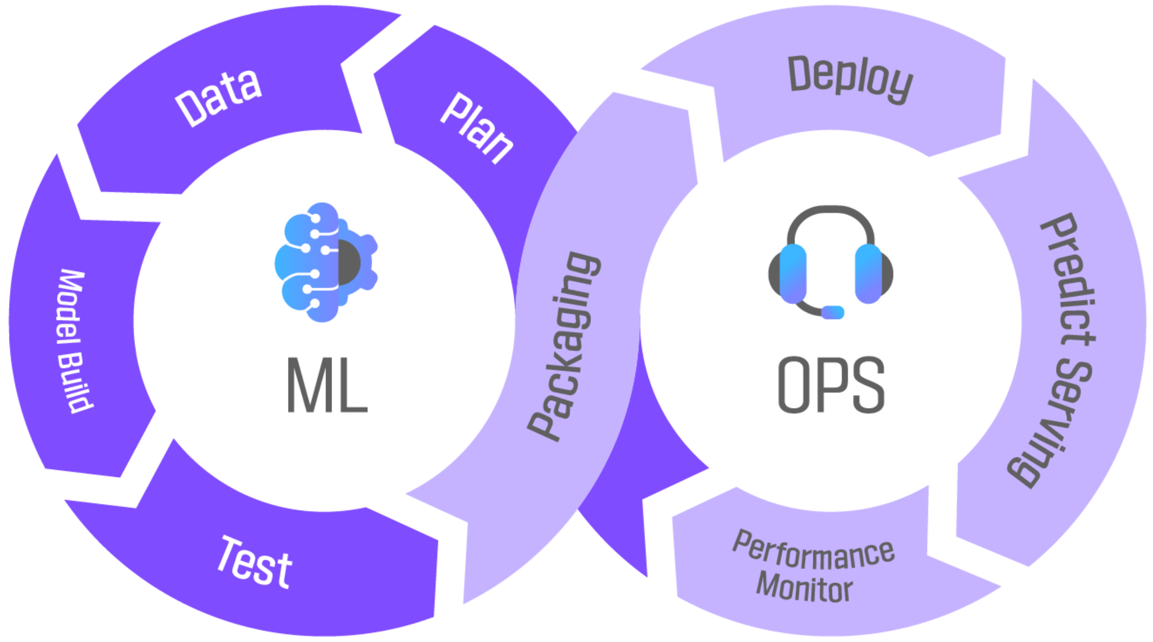


Duration

Duration is the sum of Working Time, Stopped Time, Downtime and Waiting Time.

Why detecting anomalies? For monitoring of a productive solution (3/3)

- Once we deploy a Machine Learning model in production, is not over!
- For example, we should monitor if the underlying data are consistent with what we had in training
- Anomaly Detection can be useful for monitoring! If we have outliers, we can avoid trusting a single prediction or we can start with re-train
- This is part of MLOps (Machine Learning operations) principles



How to evaluate an Unsupervised Anomaly Detection system if we are in unsupervised settings?

- That’s the true drawback of unsupervised approaches!
- If you can't get ground truth labels, try to get approximate labels or insights:

(i) Domain experts: Ask them to review top *N* anomalies flagged by your model. Are they meaningful?

(ii) Known anomaly cases: Use events like system failures, alerts, or log anomalies as partial labels.

(iii) Synthetic injection: Add known anomalies to the dataset (e.g. noise, out-of-distribution points) and check if your model finds them.

Università degli Studi di Padova

Machine Learning 2024/2025

AMCO Artificial Intelligence, Machine Learning and Control Research Group

Thank you!

Gian Antonio Susto