FEATURE SELECTION

FEATURE SELECTION

- Objective: select a subset of variables that we keep
  - Discard useless or less informative ones

- A common preprocessing step in data with many variables
  - In ML, "more is better" is generally true for observations
    - Not for variables

WHY FEATURE SELECTION

- Too many variables
  - e.g., Health data: hundreds of variables
  - In theory, in ML, more is better
    - Not true in practice: many methods work less efficiently with too many variables
    - When the ratio between the number of variables and the number of observations is high, spurious correlations => Overfitting

- Redundant variables
  - High correlation
    - Hinder interpretability
    - Make learning harder (more variables=>harder task)

FEATURE SELECTION **VS** FEATURE EXTRACTION

- Feature extraction: PCA
  - Generate new features, combinations of the original ones
  - Advantages:
    - Eliminate linear correlation
    - Control the amount of information lost
  - Drawbacks:
    - Loss of interpretability
    - Linear combinations can erase important information

- Feature selection
  - Keep the original features unchanged

FEATURE SELECTION

- Embedded feature selection
  - Mechanism in an ML method performs feature selection during the learning process
  - =>Regularization
    - Lasso
  - => Tree-based methods: implicit feature selection
    - A feature which is never useful will never be selected, thus never be used
- Filter methods
  - Common preprocessing step
- Wrapper methods
  - More efficient but slow

FILTER METHODS

- Correlation filtering
  - Compute correlation between variables
  - Keep only one of each set of highly correlated variables
    - Remove redundancy
    - Example method: find sets of correlated variables by using community detection on a graph whose edges correspond to high correlation values
    - Which one to keep? Easier to interpret or more efficient single predictor of the target…

- In case of a huge number of variables, Relevance filtering
  - Compute correlation/similarity between each variable and the target
    - Correlation, Mutual Information, etc.
    - Keep top-k

WRAPPER METHODS

- RFE: Recursive Feature Elimination
- Given a model and a notion of feature importance
  - Ad hoc or external, e.g., SHAP values
- Loop:
  - 1)Train the model on (remaining) features
  - 2)Remove the feature(s) of lower importance

TO SUM-UP

- Feature selection is useful for interpretability
- But also for model performance!
  - Too many variables often hurt the training process
    - Noise
    - Algorithms struggle to reach a good solution due to the size of the solution space
    - Redundancy hurts many models
  - =>Most ML models do not find the global optimum, thus any unnecessary difficulty will result in reaching a poorer local maximum
  - =>With more "useless" variables, more chances of overfitting
    - Spurious correlations…

IMBALANCED DATASETS

CLASS IMBALANCE

- Context: classification
  - Some classes are more frequent than others
    - e.g., binary: credit card fraud detection, COVID detection, etc.
    - e.g., multiclass: (hyperthyroid, hypothyroid, normal)

- Problems
  - Most objective functions are of the form $\mathbf{min} \sum_i \ell(\hat{y}_i, y_i)$
    - =>cross-entropy, Gini/Entropy on decision trees, etc.
    - So each positive example counts as much as each negative example
    - =>Catastrophic errors (better to predict always the majority class)
  - Common evaluation scores are misleading
    - Accuracy, Precision, F1…

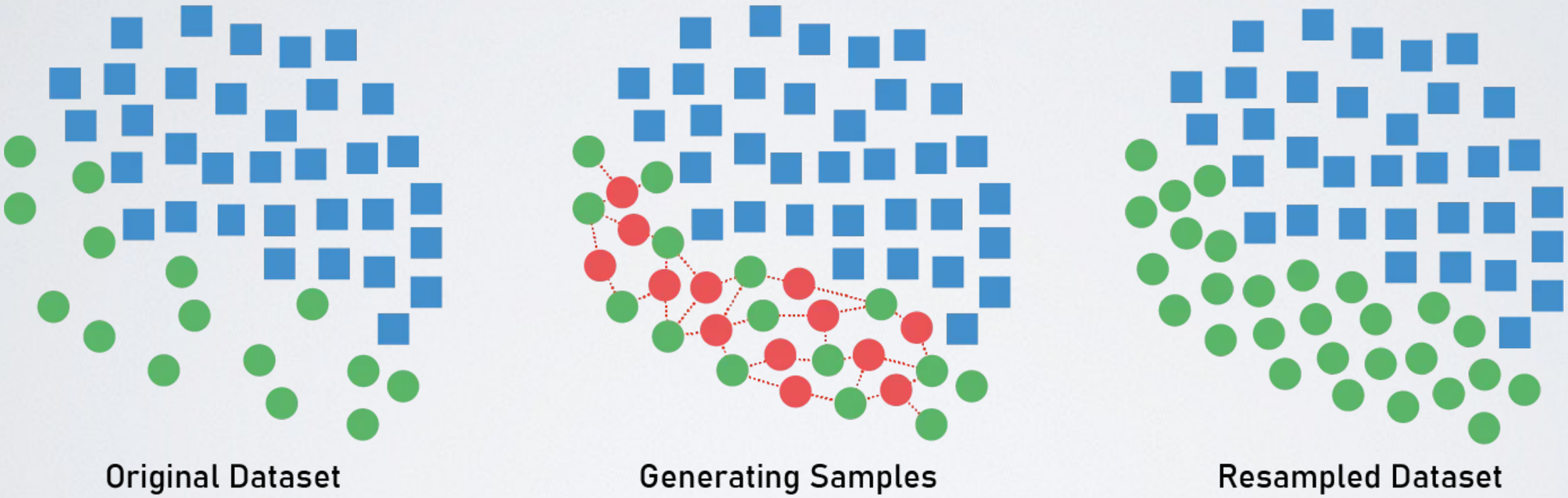
SOLUTIONS

- Undersampling
  - If there is enough data
    - Take a sample from the majority class of a similar size to the minority class
    - e.g., resemple to 70/30, or 50/50
    - =>Treat as a hyperparameter (validation set to evaluate the best undersampling ratio
    - 50/50 might remove too much data, or bias the model too much

- SMOTE (Synthetic Minority Over-Sampling Technique)
  - For cases in which data is scarce
  - Instead of removing instances, **create** synthetic ones for the minority class.
  - For tabular data: $x_{\text{new}} = x + \lambda(x_{nn} - x), \quad \lambda \in [0,1]$
    - With x an existing data point and x_{nn} a nearest neighbor of that point in the minority class
    - =>Create a new point in-between two existing points

SOLUTIONS

Synthetic Minority Oversampling Technique



https://medium.com/analytics-vidhya/bank-data-smote-b5cb01a5e0a2

SOLUTIONS

- Modifying the loss
  - =>Increase the weight of errors in the rare class
  - $L = \sum_{i=1}^{n} \ell(\hat{y}_i, y_i) \quad => \quad L = \sum_{i=1}^{n} w_{y_i} \ell(\hat{y}_i, y_i)$

- Typical approach: Inverse class frequency
  - $w_c = \frac{n}{k \cdot n_c}$
    - n: total number of samples
    - k: number of classes
    - n_c number of samples in class c

REGRESSION INBALANCE

- Imbalance is not only a problem for classification
  - =>It also exists for regression

- The distribution of target values is skewed, with some values being rare
  - Earthquake magnitude (large earthquakes are rare)
  - Apartment prices (very expensive apartments are rare)
  - Etc.

- For the same reason as before, errors made on rare values are mostly ignored

REGRESSION INBALANCE

- Most loss functions are global scores
  - $\min \sum_i \ell(\hat{y}_i, y_i)$
  - =>99% of data points in the "normal" range
    - =>The model optimizes mostly for this range

- Solutions:
  - Undersampling dense regions (if enough data)
  - SMOTER (SMOTE for Regression)
    - Create synthetic data only in sparse regions
  - Weighted loss

ANOMALY DETECTION

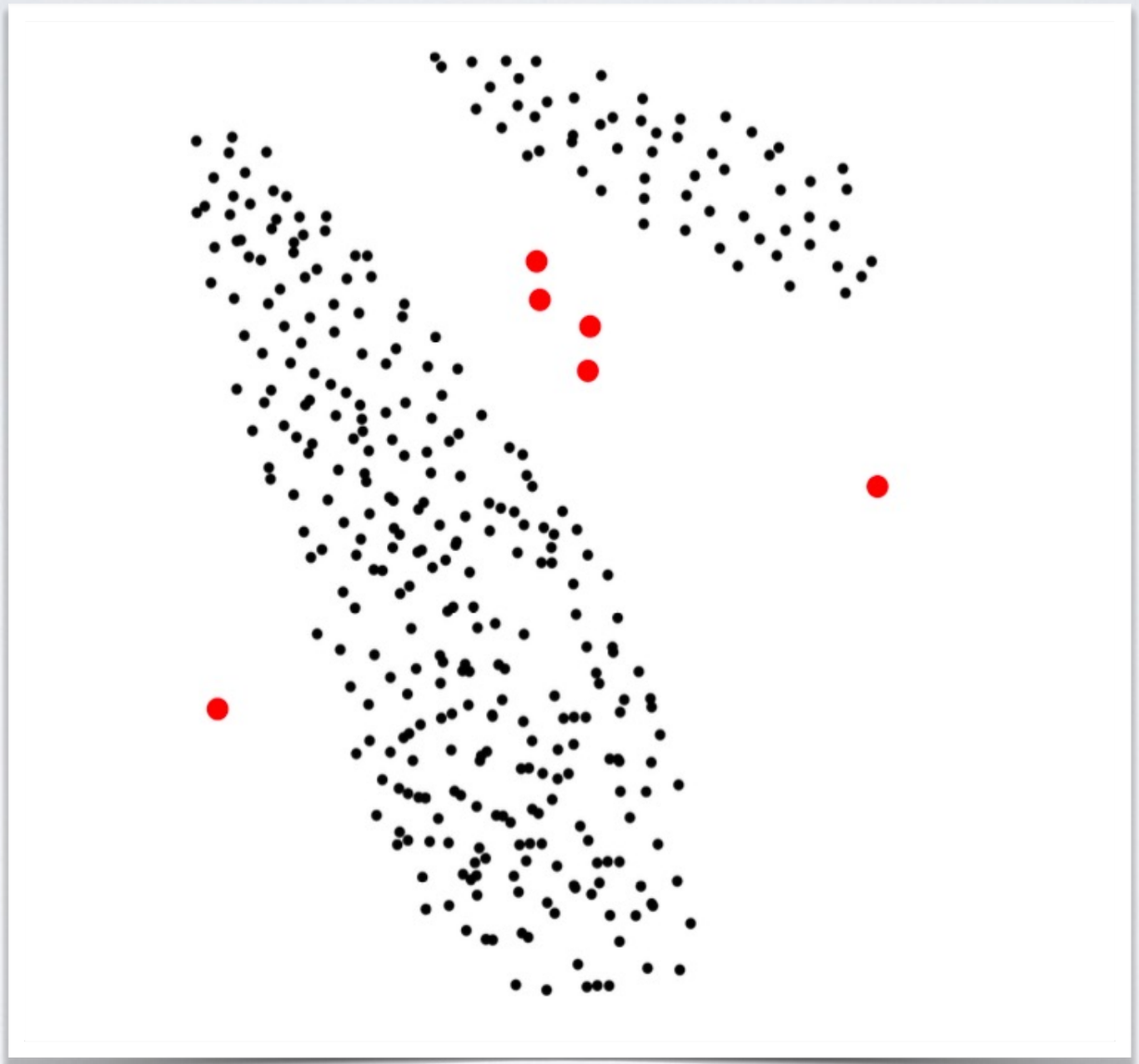
CONCEPT

- In a dataset, we want to discover **anomalies**

- An anomaly can be an error
  - Outliers

- More often, an anomaly can be what we are searching for
  - Fraud detection, Network intrusion, Equipment failure, Quality control, etc.

CONCEPT



ANOMALY VS CLASSIFICATION

- Anomaly detection can sometimes be treated as a classification problem
  - We have an extensive dataset of anomalies
    - Labelled anomalies
  - All possible types of anomalies have already been observed
    - Otherwise, we won't be able to recognize never-before-seen anomalies
  - =>Treat as highly unbalanced problem
- When it is not the case, we need to use unsupervised approaches

ANOMALY: CORE IDEA

- Intuition: An anomaly is what is not normal
  - We need first to define what is normal in our data, and what is not normal is an anomaly

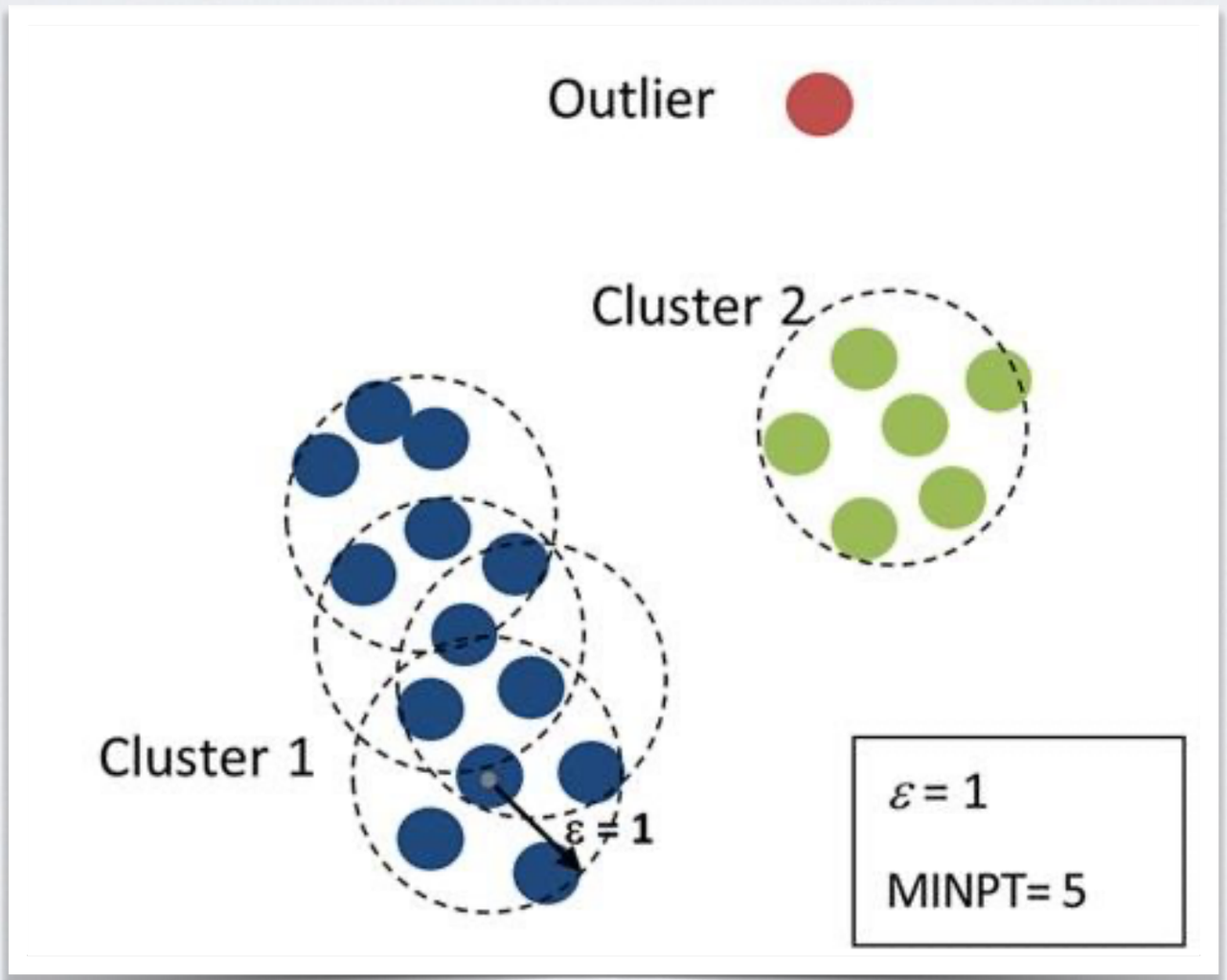
- Implementations:
  - Density-based => Anomalies are in low-density regions
    - => e.g., noise obtained using DBSCAN clustering method
  - Isolation/distance => Anomalies are far from any normal point
    - =>Isolation forest (see later)
  - Model-based => An anomaly is what a statistical model of the data cannot predict well
    - Likelihood of points in a Gaussian mixture clustering
  - Reconstruction-based => Anomalies are what is no

ANOMALY: CORE IDEA

- Intuition: An anomaly is what is not normal
  - We need first to define what is normal in our data, and what is not normal is an anomaly
- Implementations:
  - Density-based
  - Model-based
  - Reconstruction-based
  - Isolation/distance

DENSITY-BASED

- Density-based => Anomalies are in low-density regions
  - => e.g., noise obtained using DBSCAN clustering method



MODEL-BASED

- Model-based => An anomaly is what a statistical model of the data cannot predict well
  - Likelihood of points in a Gaussian mixture clustering

RECONSTRUCTION-BASED

- 1) Use a method that *compresses* the data in a low-dimensional space and then allows for reconstruction

- 2) Anomalies are what is not well reconstructed

- e.g., PCA, NMF…
  - 1)Tranform into a lower dimensional data
  - 2)transform back into the original space
  - 3) measure reconstruction error

- e.g., autoencoders
  - Measure reconstruction error

ISOLATION/DISTANCE

- Isolation/distance => Anomalies are far from any normal point
  - kNN distance outliers
    - Compute distance to neighrest points
    - Large distances => Anomaly
    - => Limits: scalability, curse of dimensionality
  - Isolation Forest (iForest)
    - More scalable
    - More robust (noise, curse of dimensionality…)

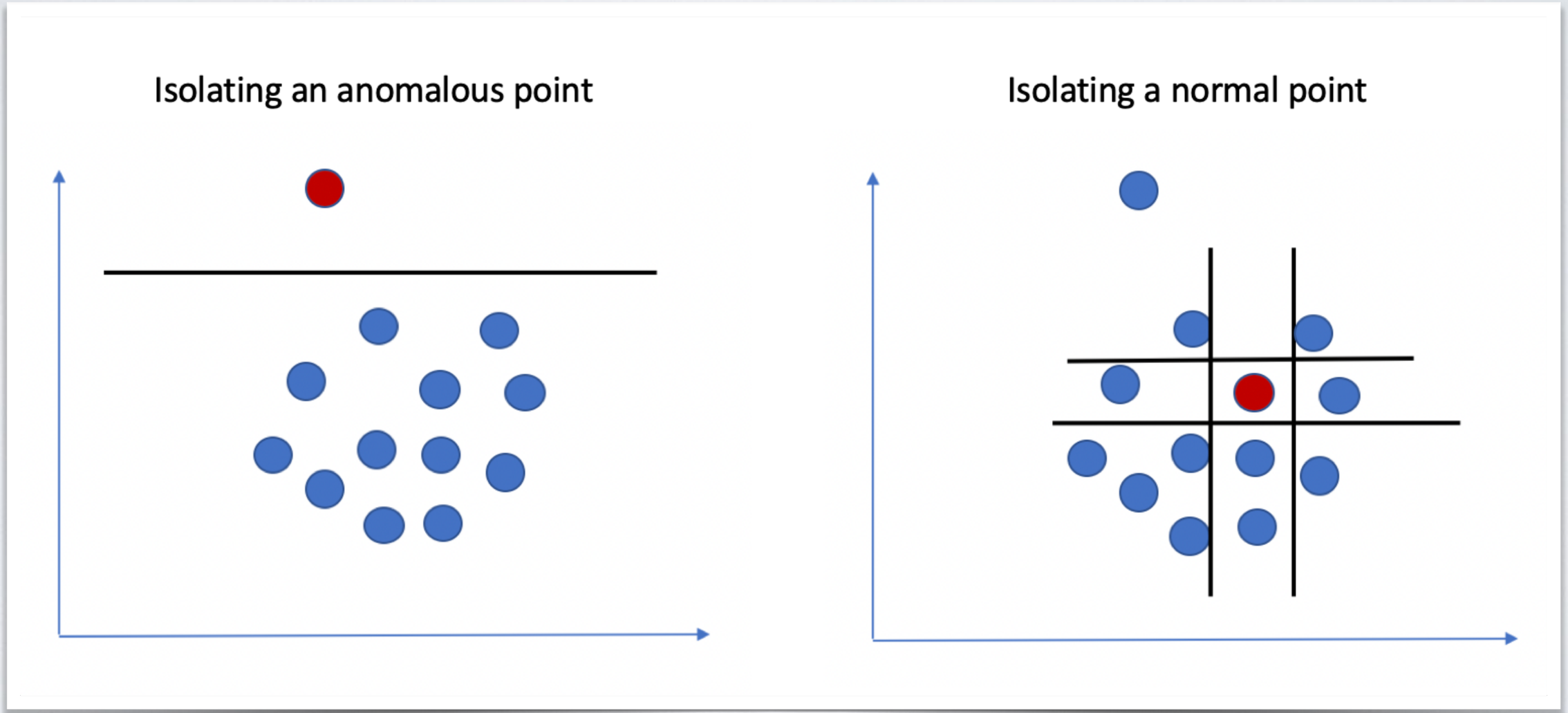
ISOLATION FOREST

- Based on many small classification trees, like the *random forest* algorithm

- Build **random** trees by
  - Choosing features at random
  - Splitting the feature at random (unlike normal trees)
    - Numeric variable: split on a random value in the range
    - Categorical variable: split by choosing a random category

- Intuition: anomalies are easier to **isolate** than normal values

ISOLATION FOREST



ISOLATION FOREST

- Parameters:
  - Subsample size
    - Each tree is built on a small subsample (e.g., 256)
  - Stop criteria
    - e.g., each instance in its own leaf, or max_depth
  - Number of trees
    - Typically 200-300

- Algorithm (for each tree)
  - While *Stop criteria* not reached
    - Pick a variable at random (among those present in the sample)
    - Split at random (among limits present in the sample)

- Anomaly score: average depth in the trees
    - Note that each tree sees only few samples but is used to evaluate each sample

ISOLATION FOREST