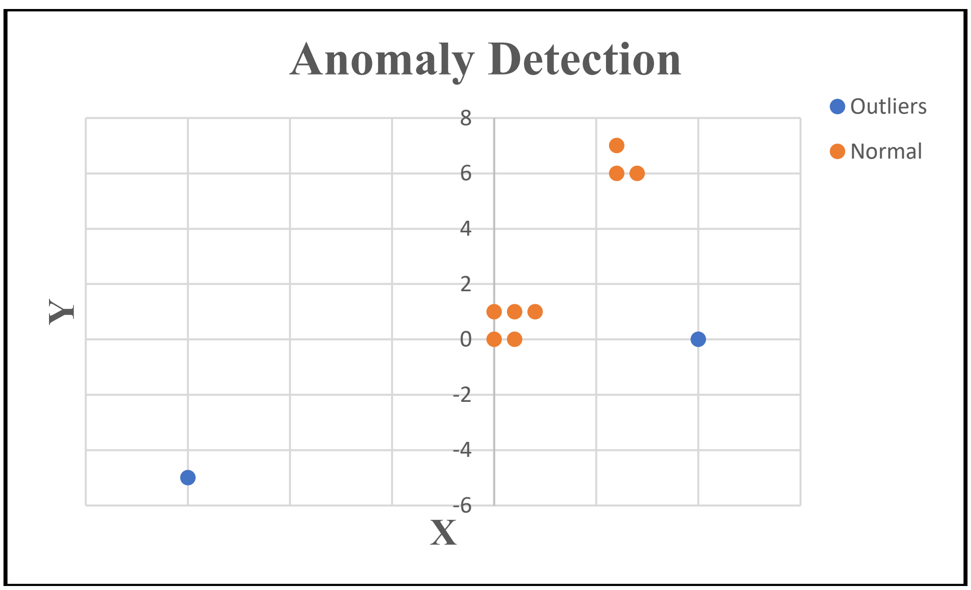
Anomaly detection

- **Introduction**
  - Identifying data points that deviate strongly from the norm: Outliers
  - Real-world examples:
    - Fraud detection: Credit-card fraud
    - Machine fault monitoring: High temp at night
    - Network attack spike
    - Medical outlier detection
  - Works well for large, high-dimensional datasets and few anomalies
  - Does not assume a normal distribution
  - Fast & scalable – based on random decision trees
  - Check this graph:



- **Z-score Anomaly Detection**
  - Z-score measures how far a data point is away from the mean as a signed multiple of the standard deviation. Large absolute values of the Z-score suggest an anomaly.

- **The z-score:**
  - A z-score measures how many standard deviations a data point is from the mean (μ).

$$Z = \frac{x - \mu}{\sigma}$$

    where x is the data point, μ is the mean, and σ is the standard deviation.
  - Z-score can be both positive and negative.
  - The farther away from 0, higher the chance of a given data point being an outlier.
  - A data point is considered **anomalous** if:

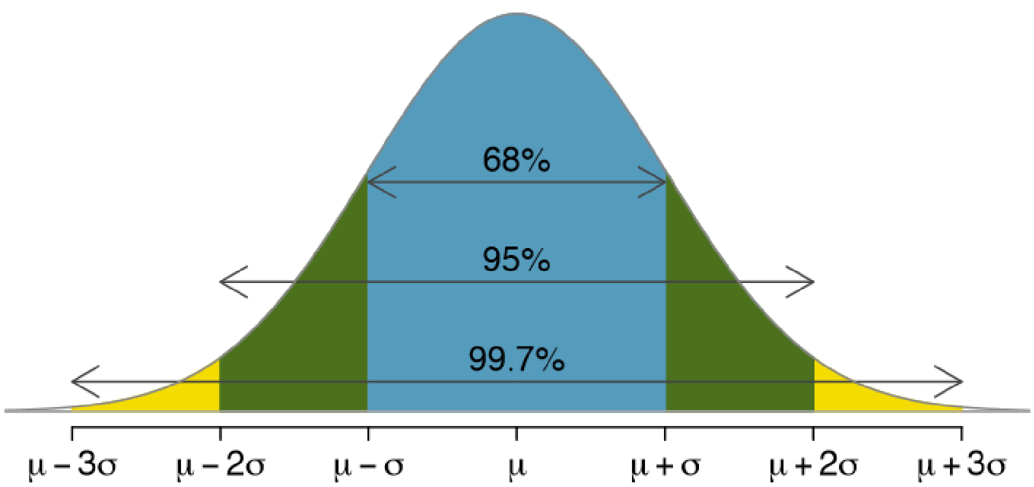
    |Zi| > threshold

    Typical thresholds:

    2.0 ➔ marks about the outer 5 % of data

    2.5 ➔ outer 2 % (Moderately strict)

    3.0 ➔ outer 0.3 % (very strict)



source: pinterest graphic

- **Advantages:**
    - Simple and easy to understand.
    - Works well for univariate data where the data is normally distributed.
- **Limitations:**
    - Highly sensitive to outliers, which can skew the mean and standard deviation.
    - Less effective for multivariate data or non-normally distributed data.
    - May not work well if anomalies are clustered or in complex patterns.
- **Example:**

Data Points	X	Y
P0	0	0
P1	0	1
P2	1	0
P3	1	1
P4	2	1
P5	6	6
P6	6	7
P7	7	6
P8	10	0
P9	-15	-5

- Compute Mean & Standard Deviation

Axis	Mean (μ)	Std (σ)
X	1.8	6.21
Y	2.3	3.68

- Calculate Z-Scores

Pt	X	Z-X	Y	Z-Y
P0	0	−0.29	0	−0.63
P1	0	−0.29	1	−0.35
P2	1	−0.13	0	−0.63
P3	1	−0.13	1	−0.35
P4	2	+0.03	1	−0.35
P5	6	+0.68	6	+1.00
P6	6	+0.68	7	+1.28
P7	7	+0.84	6	+1.00
P8	10	+1.32	0	−0.63
P9	−15	−2.71	−5	−1.96

- Check Anomalies: Threshold: 2.5
  - We'll use a threshold |Z| > 2.5 on **either axis**.

Data Points	|Z-X| > 2.5?	|Z-Y| > 2.5?	Anomaly ?
P0	x	x	No
P1	x	x	No
P2	x	x	No
P3	x	x	No
P4	x	x	No

P5	x	x	No
P6	x	x	No
P7	x	x	No
P8	x	x	No
P9	(2.71 > 2.5)	x	**Anomaly**

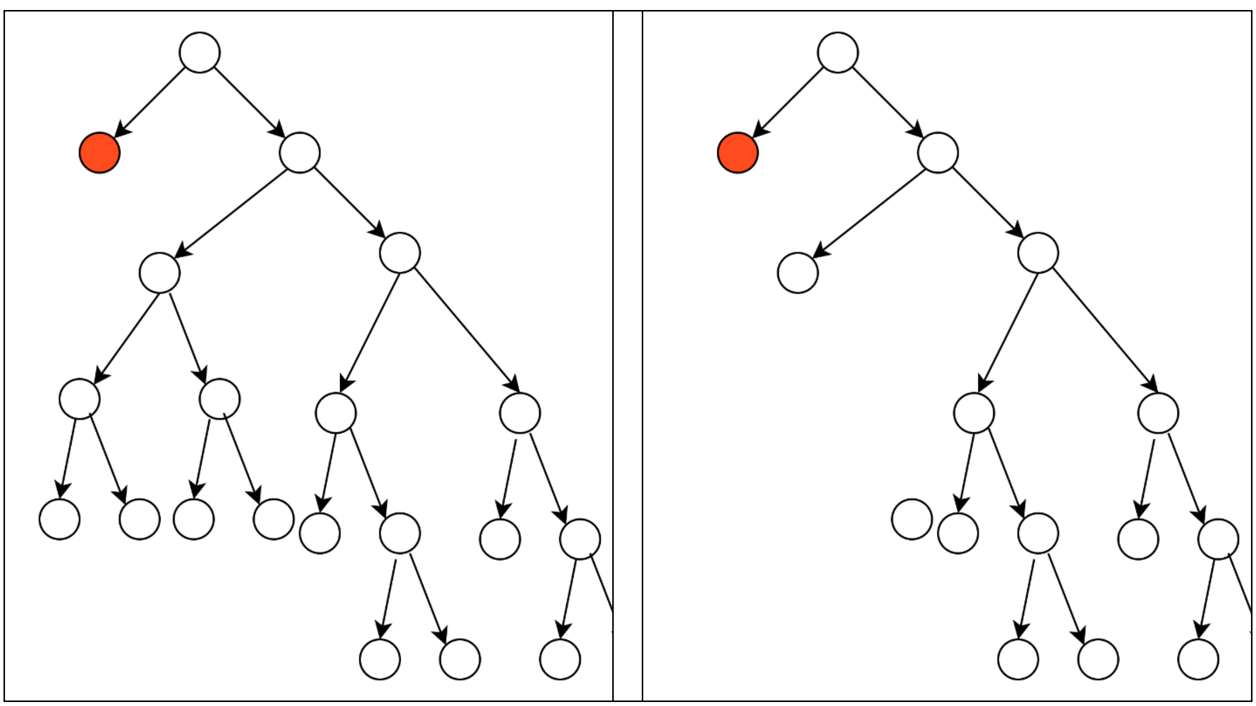
- Interpretation:
    - With **threshold = 2.5**, we only mark points that are unusually far from the mean.
    - **P9 (−15, −5)** is about 2.71 standard deviations from the mean X and 1.96 from Y.

      ➔ It is the **only anomaly**.

    - All other points fall well within ± 2.5 σ → **normal**.

- **Isolation Forest Algorithm:**
    - Isolation Forest builds multiple random binary trees (itrees) to isolate data points.
    - Randomly split the data using feature thresholds:
        - Unlike a decision tree used for prediction, an Isolation Forest's trees (**Isolation Trees or itrees**) are built *entirely at random*:
    - Normal data points:
        - It requires more splits to isolate.
    - Outliers:
        - It takes fewer splits.

- For each tree:
  - At each node, the algorithm picks one feature at random (in your data, either X or Y).
  - Chooses a **random split value** uniformly between the *minimum and maximum* of that feature among the points currently in the node.
  - Split the data into left/right subsets:
    - **Left child:** points with that feature ≤ threshold
    - **Right child:** points with that feature > threshold
  - Repeat recursively on each smaller subset until:
    - The subset has 1 data point, or
    - The maximum tree depth is reached (≈ $\log_2 n$).



- **Prediction**

- In an Isolation Forest, each data point *x* gets an **<u>anomaly score</u>** called **s(x,n) or s(x),** *Liu et al., "Isolation Forest," ICDM 2008*.
- It measures **how easily that point can be isolated** by the random trees.
- **Compute Path Length:**
  - The path length h(x) is the number of edges from the root node to the leaf where point x ends.
  - Each point has a different path length depending on how easy it was to isolate.
  - Interpretation:
    - Short path → easily isolated → likely an anomaly.
    - Long path → deep in the tree → likely normal.

- **Average Over All Trees:**
  - Repeat the process for many trees (e.g., 100).
  - For each point, compute the average path length across all trees:

    E[h(x)]=average path length across all trees

- **Anomaly Score:**

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

Where,

x: a data point
n: The number of data points used to build each tree
h(x)
E[h(x)]: The average path length of x across all trees, also called itrees.
c(n): It is the average value of h(x)

  - Interpretation:

    s(x)≈1 → highly likely anomaly

$s(x) \approx 0.5 \rightarrow$ normal
$s(x) < 0.5 \rightarrow$ strongly normal

- **Label the Outliers:**
  - Choose a threshold (often based on the contamination rate, e.g., 0.1 or 0.2) and label data points:

    If s(x)>threshold
    then
        label as anomaly (-1).
    Else
        label as normal (1).

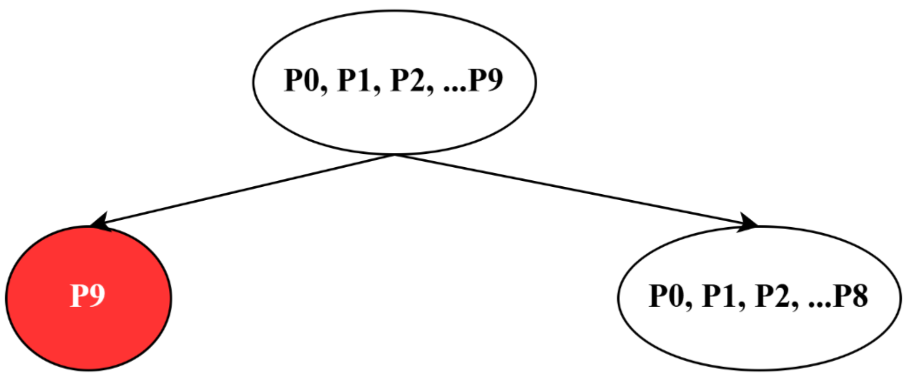
  - Outliers like P8 (10,0) and P9 (-15,-5) in our previous dataset will have:
    shorter paths → higher scores → anomalies.

- **Example**
  - Given the following dataset:

Data Points	X	Y
P0	0	0
P1	0	1
P2	1	0
P3	1	1
P4	2	1
P5	6	6
P6	6	7
P7	7	6
P8	10	0
P9	-15	-5

- **First Tree:**
  - First node, the algorithm chose:
    - Random feature = **X**
    - Range of X values in your dataset = [–15, 10]
    - Random threshold = **–0.592**

    - So, at this node:
      - **Left branch**:
        ➔Points with X ≤ –0.592 → only P9 (–15, –5)
      - **Right branch:**
        ➔points with X > –0.592 → all the others (P0–P8): P0,P1,P2,P3,P4,P5,P6,P7,P8



- **Subsequent Nodes:**
  - Repeat the process for the Right node.