Outlier Detection:

Density and Partition-Based

Mining Massive Datasets

Prof. Carlos Castillo — https://chato.cl/teach

Universitat Pompeu Fabra Barcelona

Sources

Liu, F. T., Ting, K. M., & Zhou, Z. H. Isolation forest. ICDM 2008.

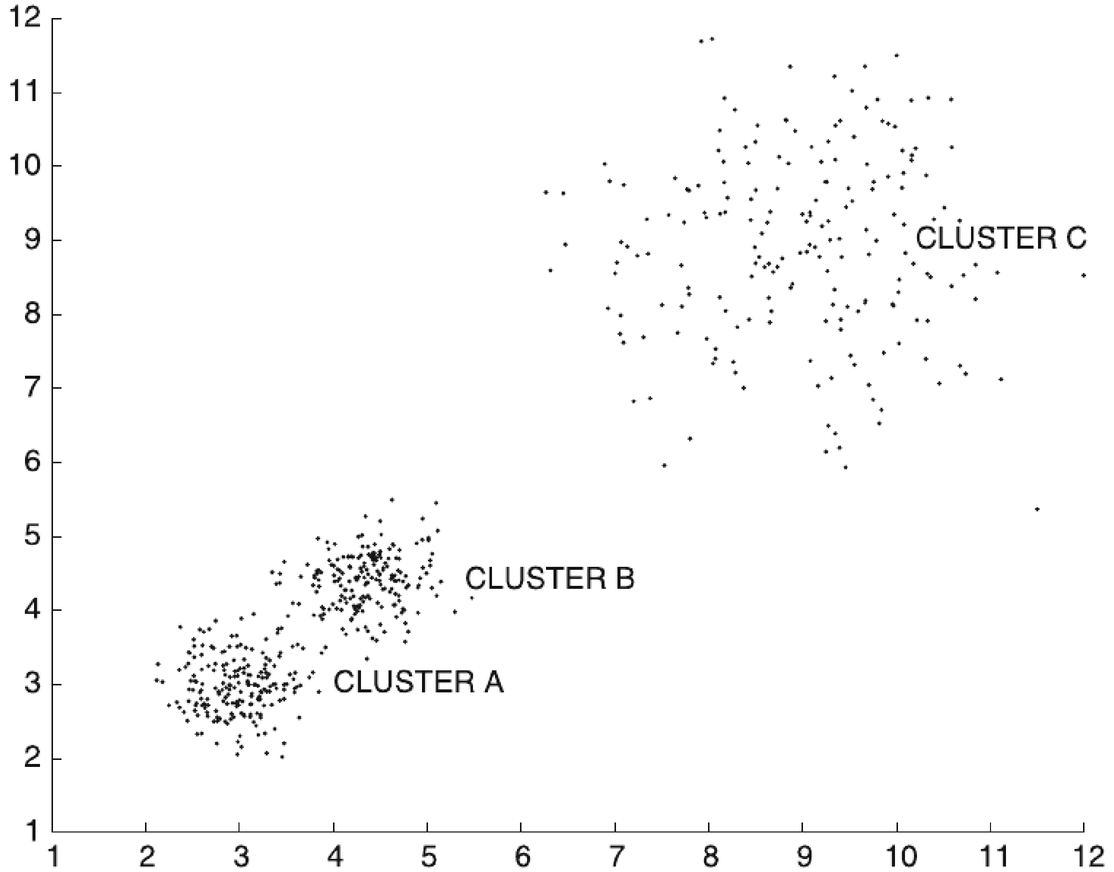
(1) Eryk Lewinson: Outlier detection with isolation forest (2018)

(2) Tobias Sterbak: Detecting network attacks with isolation forests (2018)

Density-based methods

Density-based methods

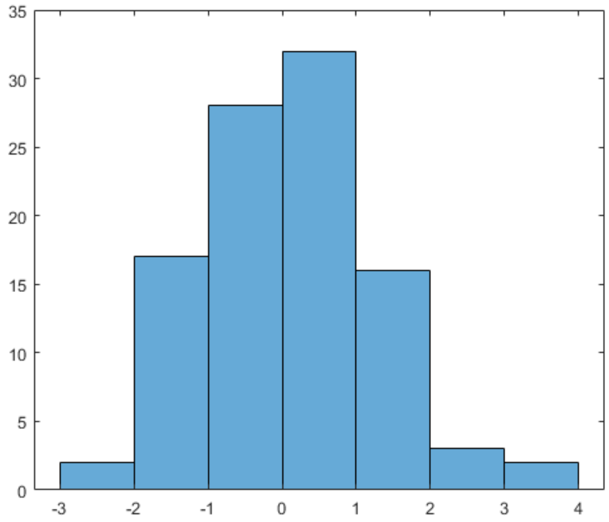
- Key idea:
  find sparse regions in the data
- Limitation:
  cannot handle variations of density



Histogram- and grid-based methods

Histogram-based method:

1. Put data into **bins**
2. Outlier score: $num - 1$, where num is the number of items in the same **bin**

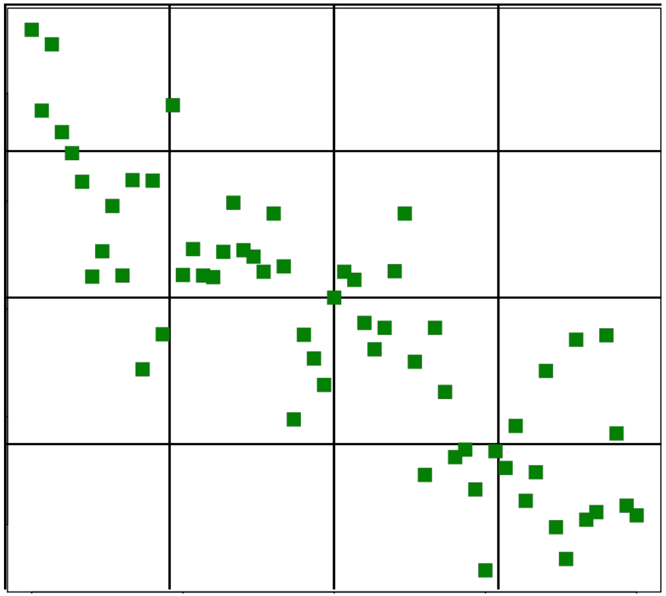


Clear outliers are alone or almost alone in a **bin**

Histogram- and grid-based methods

Grid-based method

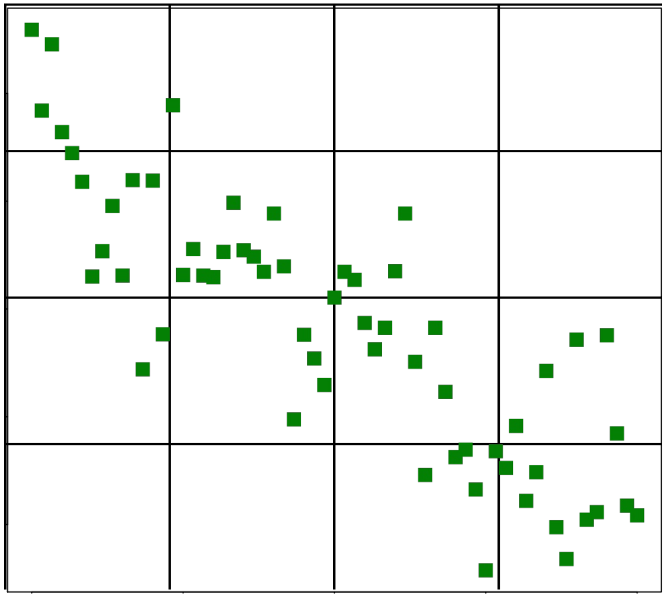
1. Put data into a **grid**
2. Outlier score: $num - 1$, where num is the number of items in the same **cell**



Clear outliers are alone or almost alone in a **cell**

Problems with grid-based methods

- How to choose the grid size?
- Grid size should be chosen considering data density, but density might vary across regions
- If dimensionality is high, then most cells will be empty



Kernel-based methods

- Given n points $\overline{X_1}, \overline{X_2}, \ldots, \overline{X_n}$

$$f(\overline{X}) = \frac{1}{n} \sum_{i=1}^{n} K_h(\overline{X} - \overline{X_i})$$

- K_h is a function peaking at $\overline{X}_i$ with *bandwidth* h

- For instance, a Gaussian kernel:

$$K_h(\overline{X} - \overline{X_i}) = \left(\frac{1}{\sqrt{2\pi} \cdot h} \right)^d \cdot e^{-\|\overline{X} - \overline{X_i}\|^2/(2h^2)}$$

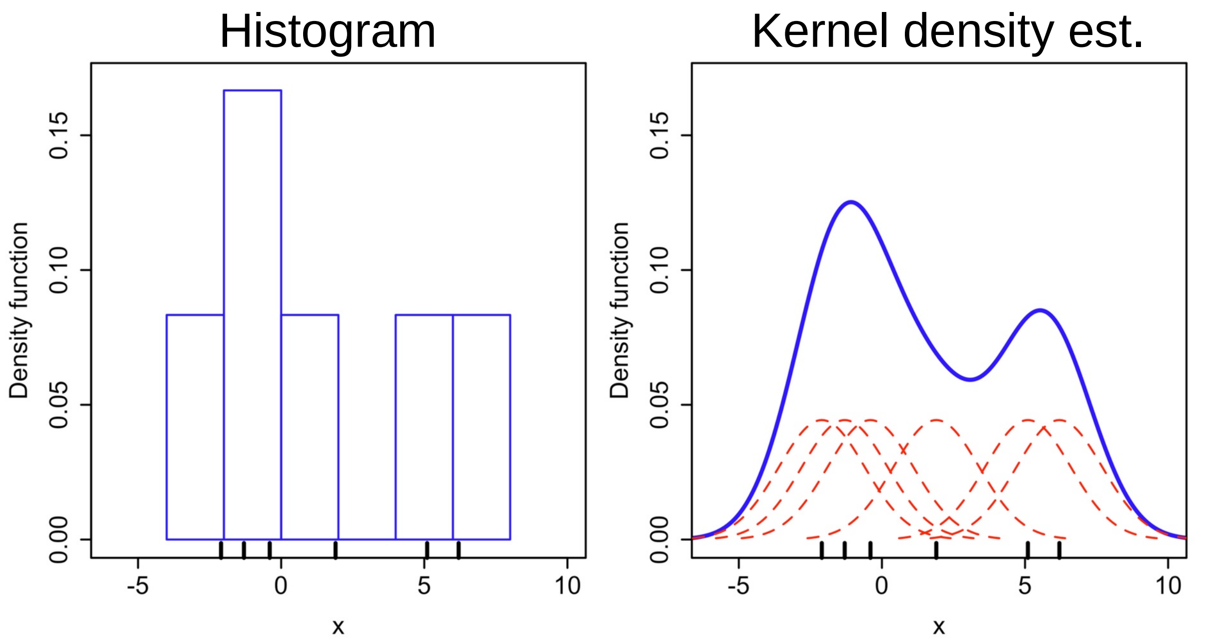
Kernel-based methods (cont.)

- Example with a Gaussian kernel

$$\overline{X} = < -2.1, -1.3, -0.4, 1.9, 5.1, 6.2 >$$

- Each K_h in red
- f = sum of K_h in blue

$$f(\overline{X}) = \frac{1}{n}\sum_{i=1}^{n} K_h(\overline{X} - \overline{X_i})$$



[Wikipedia: Kernel density estimation]

Partitioning-based method: isolation forest

Isolation forest method

- tree_build(X)
  - Pick a random dimension r of dataset X
  - Pick a random point p in $[\min_r(X), \max_r(X)]$
  - Divide the data into two pieces: $x_r < p$ and $x_r \geq p$
  - Recursively process each piece

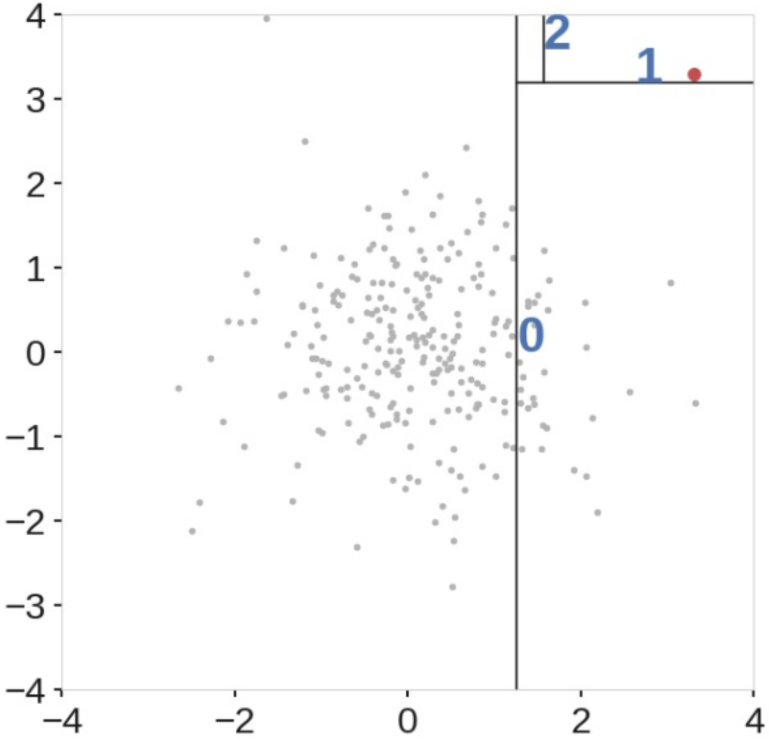
Stopping criteria for recursion

- Stop when a maximum depth has been reached

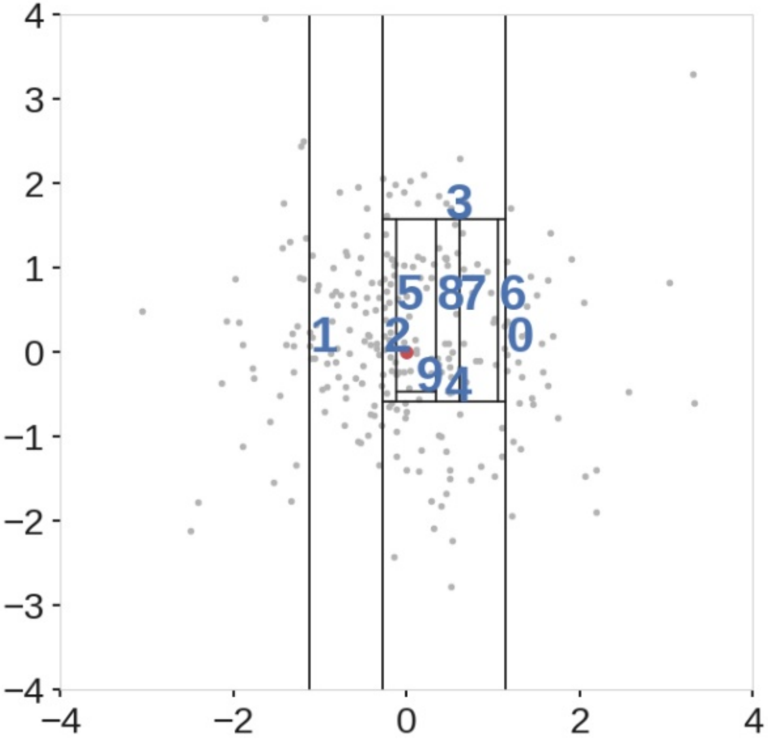
-or-

- Stop when each point is alone in one partition

Key: outliers lie at small depths



(a) Anomaly point



(b) Nominal point

https://towardsdatascience.com/outlier-detection-with-extended-isolation-forest-1e248a3fe97b

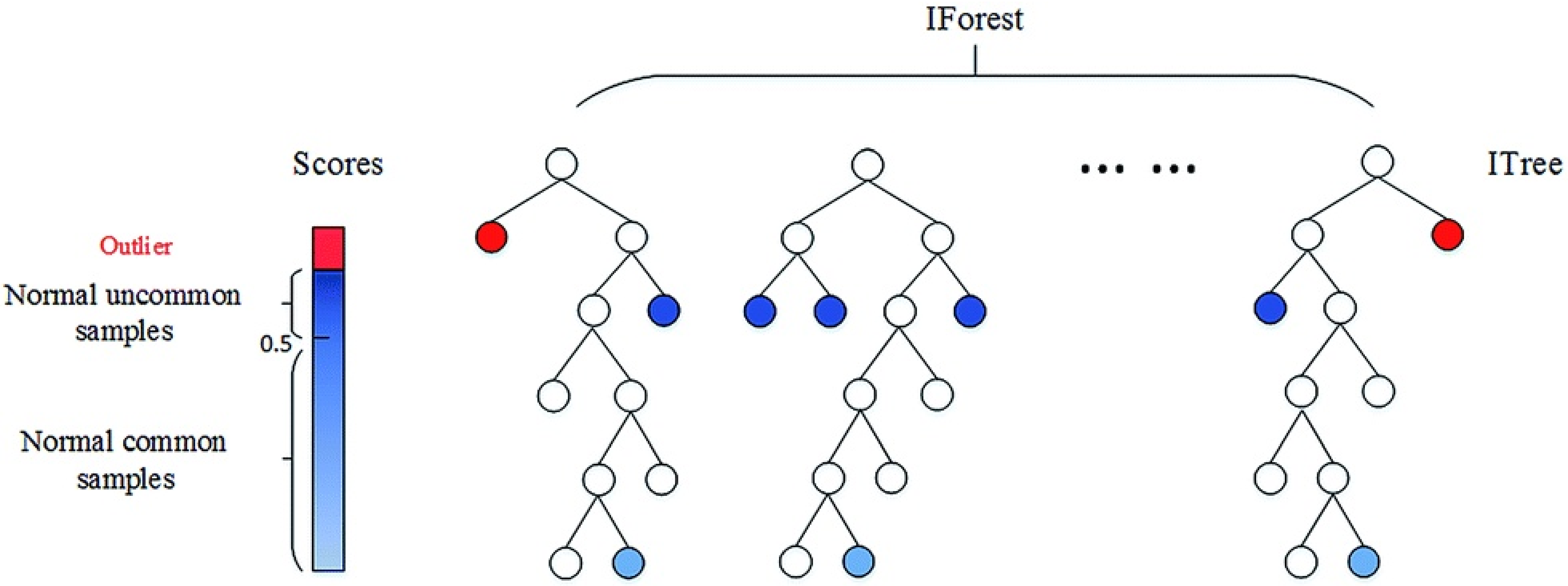
Outlier score

- Let c(n) be the average path length of an unsuccessful search in a binary tree of n items
$$c(n) = 2H(n-1) - (2(n-1)/n)$$
$$H(n) = \sum_{k=1}^{n} \frac{1}{k}$$

- h(x) is the depth at which x is found in tree
- Score: $\text{outlier}(x, n) = 2^{-\frac{E(h(x))}{c(n)}}$

Outlier scores in isolation forests

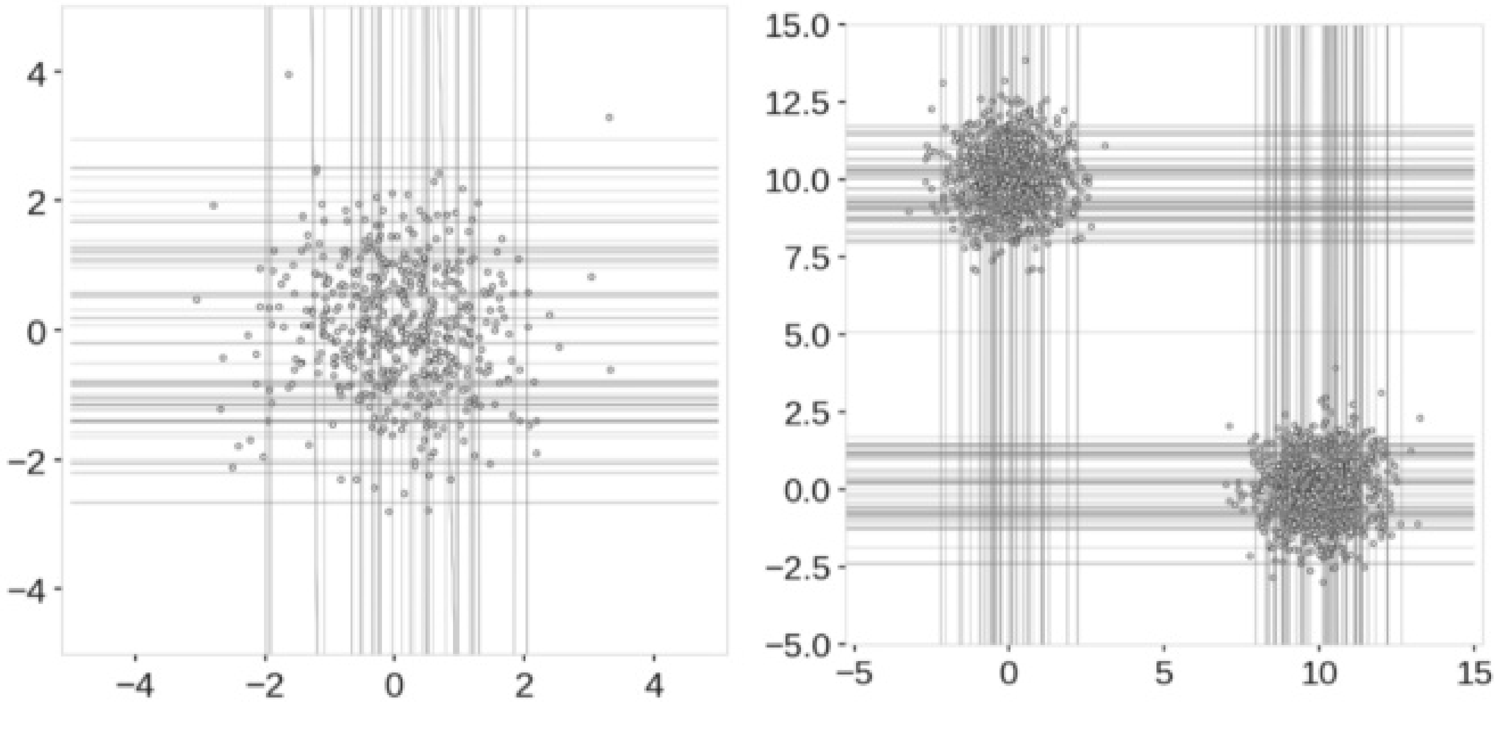
(each tree is built from a sub-sample of original data)



https://donghwa-kim.github.io/iforest.html

Example

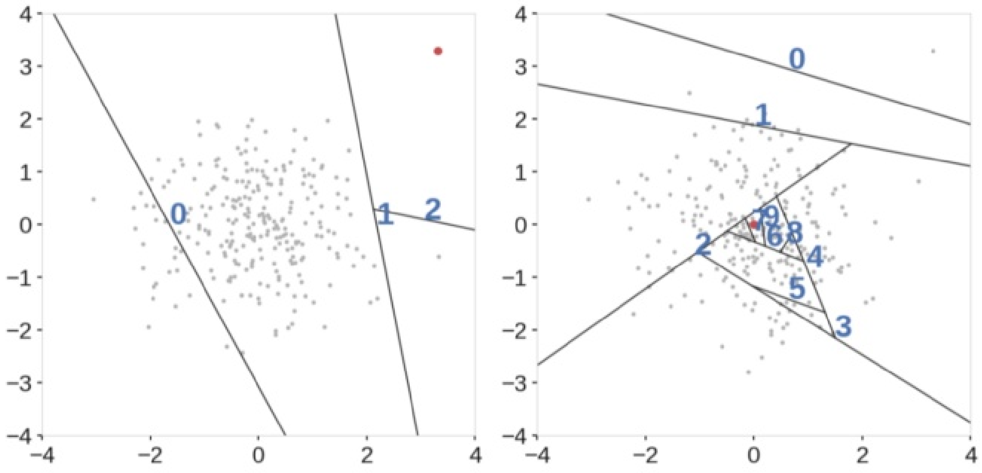
(Note: here lines cross each other: we do not cross lines)



(a) Single blob (b) Multiple Blobs

Extended Isolation Forest

- More freedom to partitioning by choosing a random slope and a random intercept

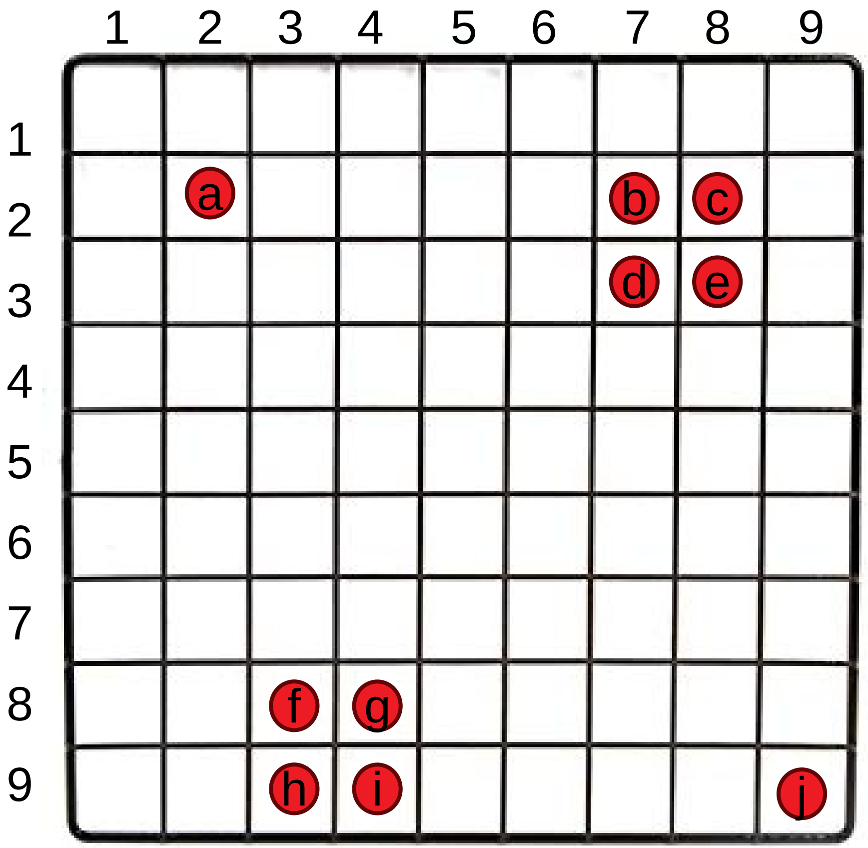


(a) Anomaly (b) Nominal

https://towardsdatascience.com/outlier-detection-with-extended-isolation-forest-1e248a3fe97b

Exercise: isolation forest

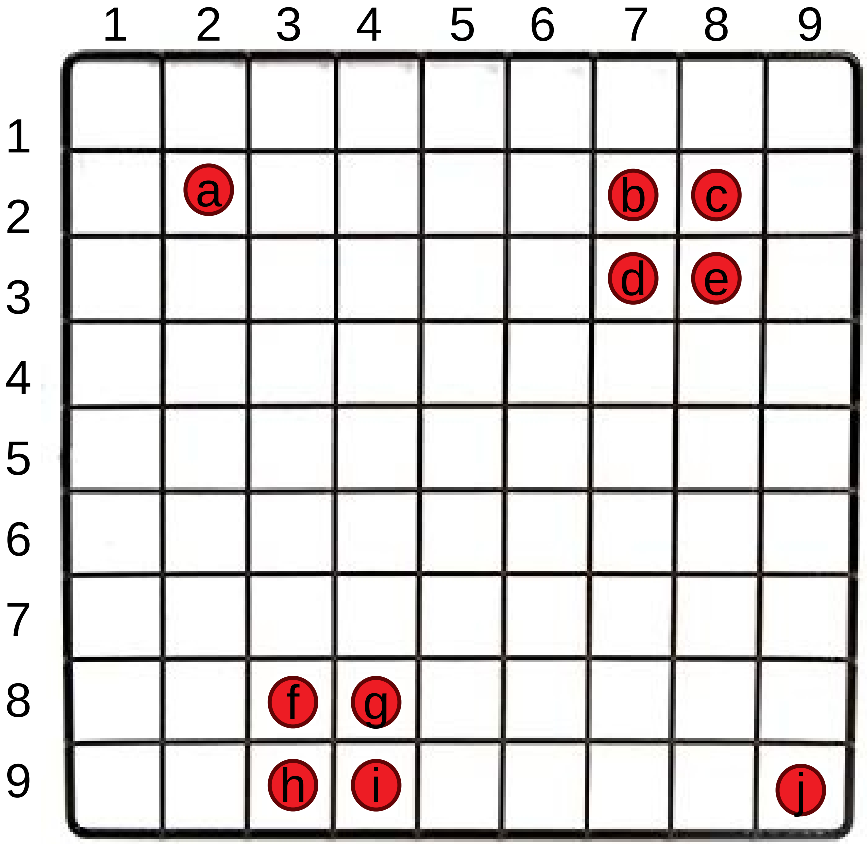
- Create one tree of the isolation forest by repeating 4 times:
  - Picking a sector containing >1 element
  - Picking a random dimension
  - Picking a random cut-off between min and max value along that dimension
  - Draw the line of your cut — do not cross lines, and label each line with a number 0, 1, 2, ...
- Stop when each point is isolated
- Label each point with its depth h(x)



This is normally repeated several times, in the end:

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

In this case c(10) = 2xH(9) – (2x9/10) ≃ 3.857 ≃ **4**


1 2 3 4 5 6 7 8 9
1
2
3
4
5
6
7
8
9
a
b
c
d
e
f
g
h
i
j

Summary

Things to remember

- Density-based methods
- Isolation forest

Exercises for TT19-TT21

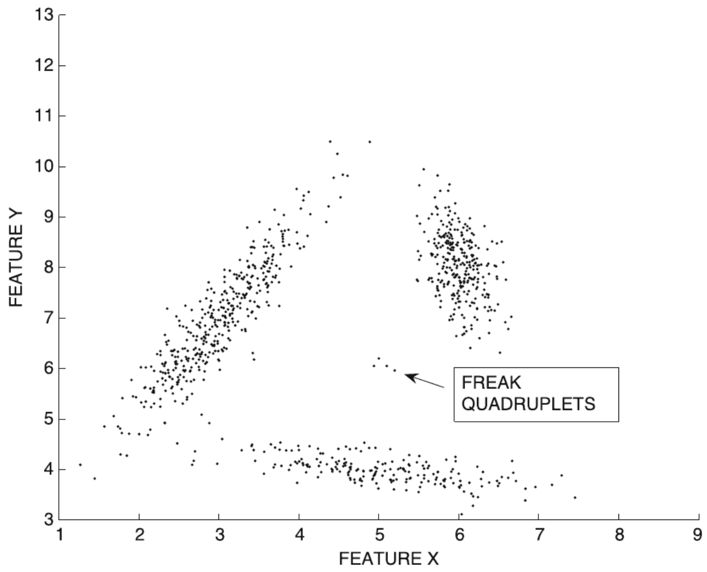
- Data Mining, The Textbook (2015) by Charu Aggarwal
  - Exercises 8.11 $\rightarrow$ all except 10, 15, 16, 17

Additional contents
(not included in exams)

Distance-based methods

Instance-specific definition

- The distance-based outlier score of an object x is its distance to its k^{th} nearest neighbor
- In this example of a small group of 4 outliers, we can set $k > 3$

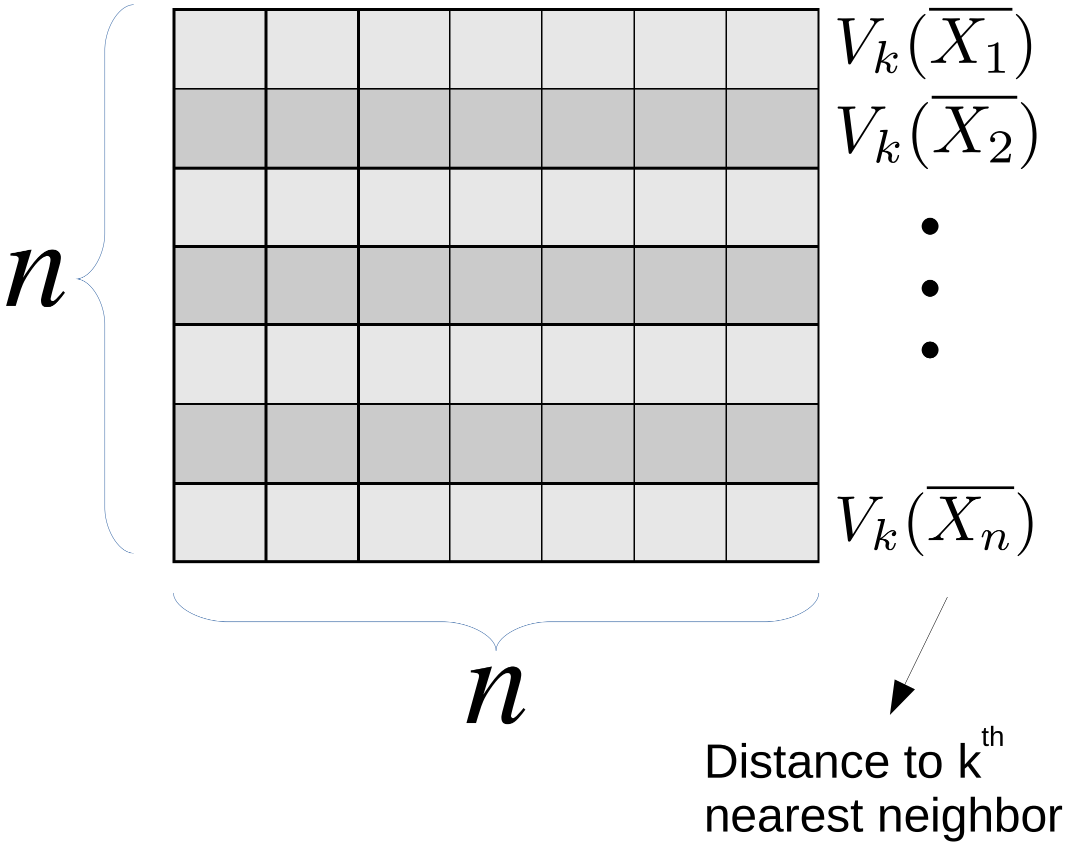


Problem: computational cost

- The distance-based outlier score of an object x is its distance to its k^{th} nearest neighbor
- In principle this requires $O(n^2)$ computations!
  - Index structure:
    useful only for cases of low data dimensionality
  - Pruning tricks:
    useful when only top-r outliers are needed

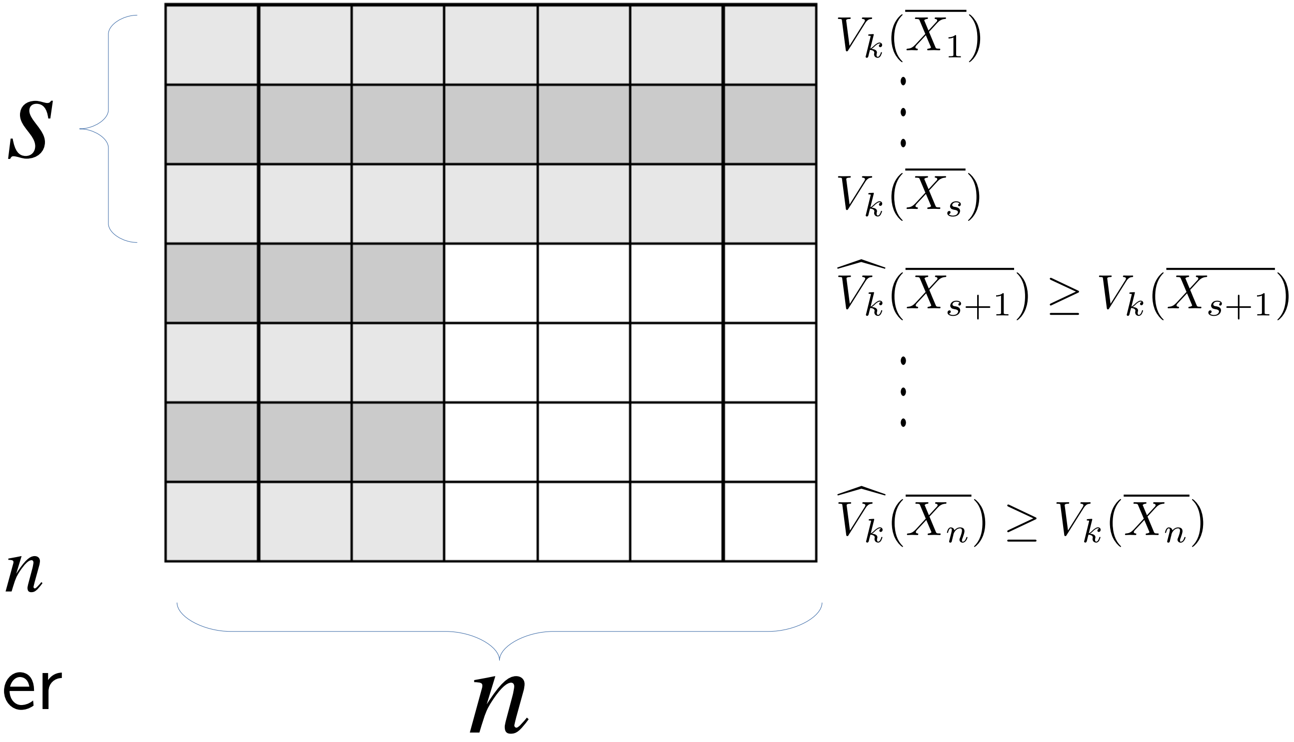
Problem: computational cost

- The distance-based outlier score of an item x is its distance to its k^{th} nearest neighbor
- In principle this requires:
  - $O(n^2)$ computations for evaluating the n x n distance matrix
  - $O(n^2)$ computations for finding the r smallest values on each row



Pruning method: sampling

- Evaluate s x n distances
- For points *1…s* we are OK
- For points *(s+1)…n* we know only upper bounds



Pruning method: sampling (cont.)

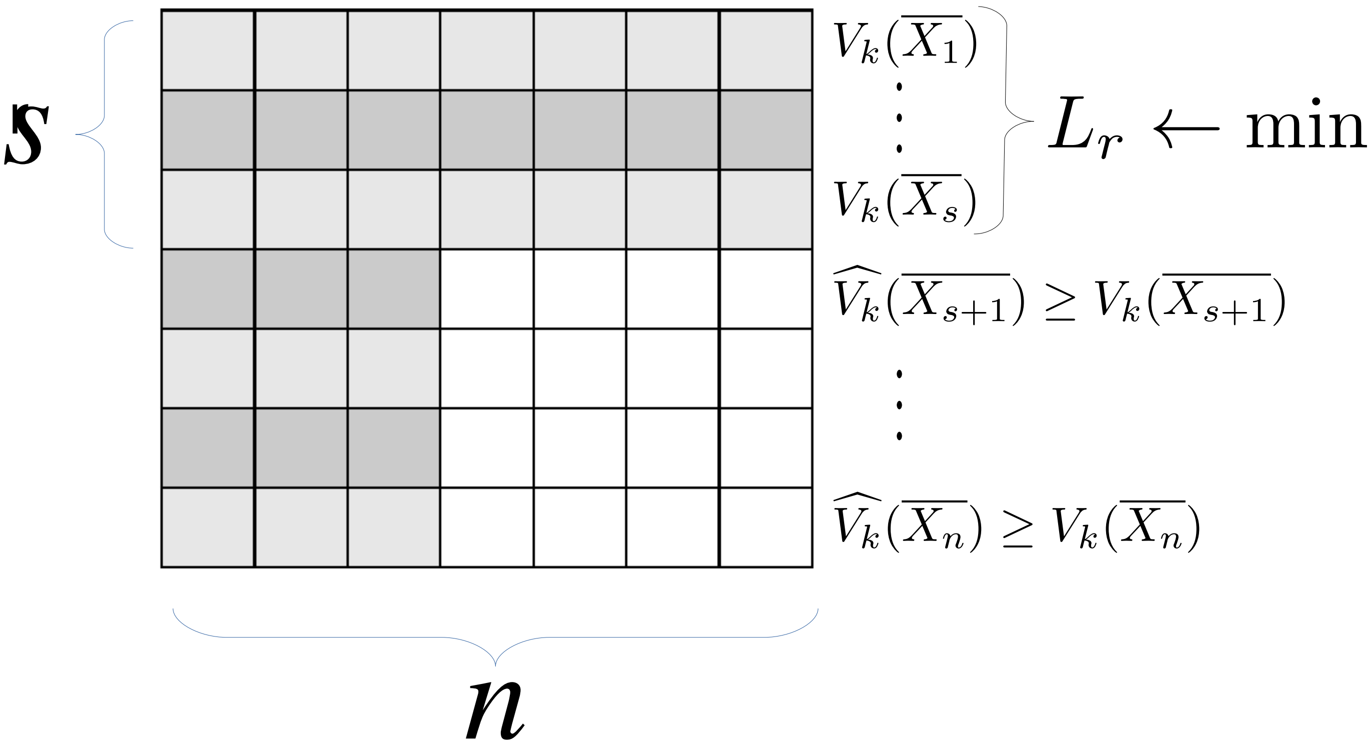
From points

1…s we already know the

"winners"

($r \leq s$ nodes with the larger distance to their

k^{th} nearest neighbor)



Any point having

$V_k < L_s$ cannot be among

the top r outliers

Pruning method: sampling (cont.)

From points

$1 \ldots s$ we already know the

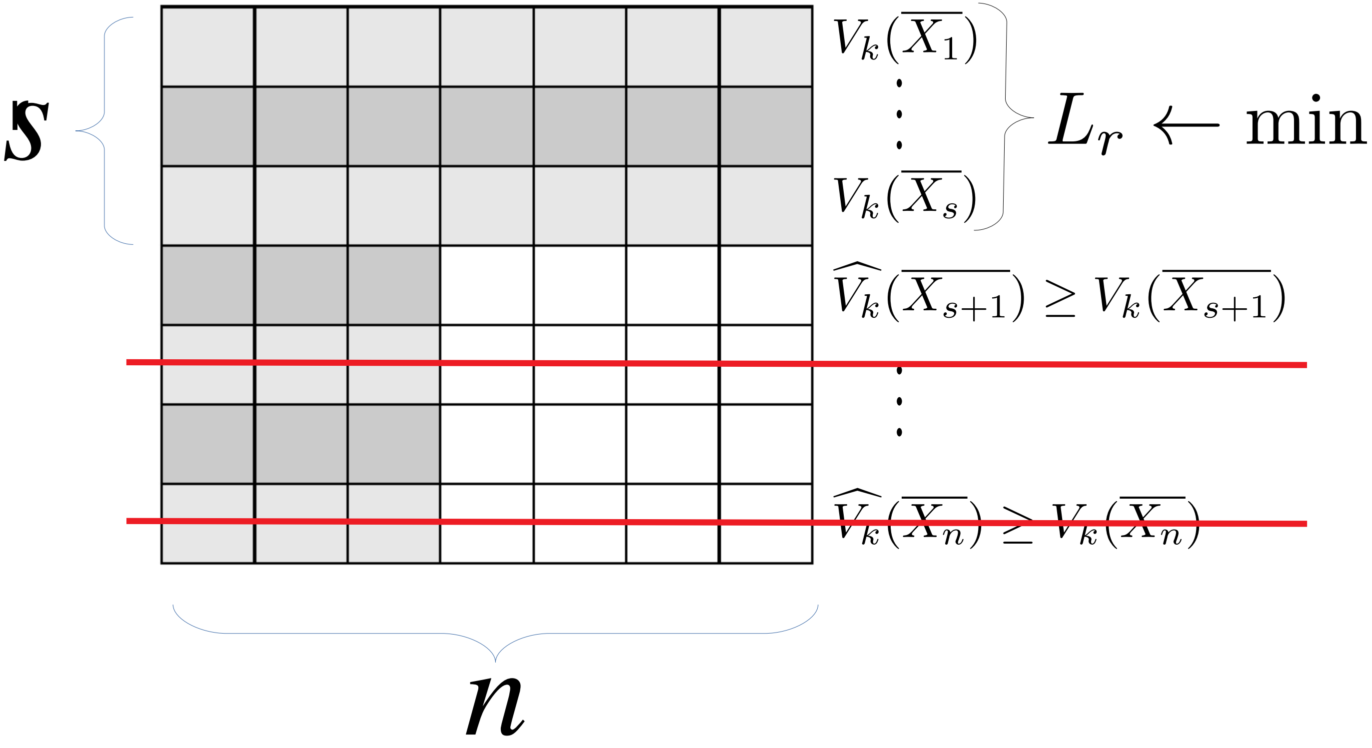
"winners"

($r \leq s$ nodes with the larger distance to their k^{th} nearest neighbor)

Any point having

$V_k < L_s$ cannot be among

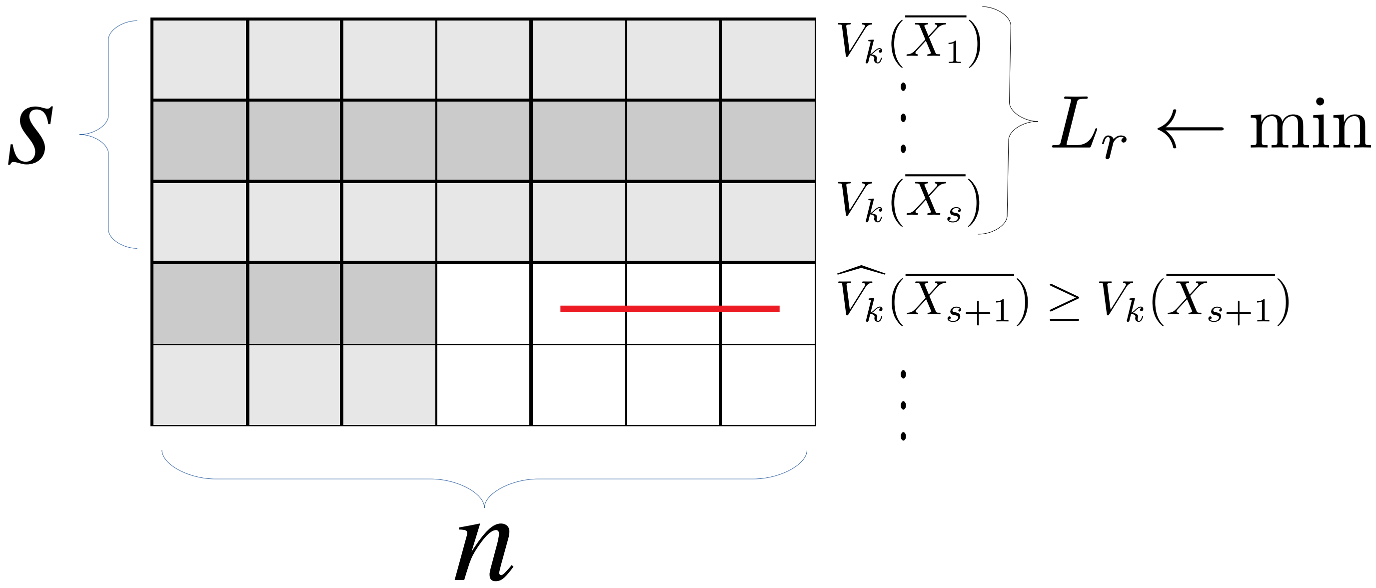
the top r outliers



Pruning method: sampling (cont.)

Remove points

having $\widehat{V_k} \leq L_r$



Update L_r keeping

r largest values, and

stop computing for a

row if one already finds

k nearest neighbors in that row that

are all below distance L_r

Local outlier factor

Local Outlier Factor (LOF)

- Let $V_k(\overline{X})$ be the distance of $\overline{X}$ to its k-nearest neighbor
- Reachability distance

$$R_k(\overline{X}, \overline{Y}) = \max\{\mathrm{Dist}(\overline{X}, \overline{Y}), V_k(\overline{Y})\}$$

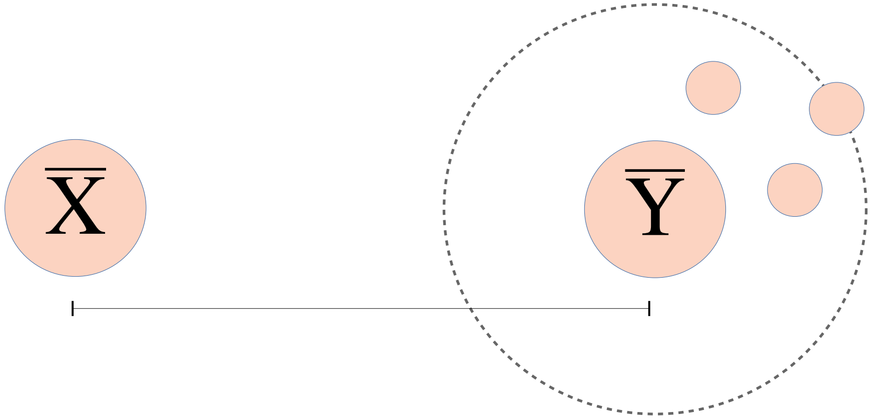
Local Outlier Factor (LOF) (cont.)

- $V_k(\overline{X})$: distance of $\overline{X}$ to its k-nearest neighbor

- Reachability distance

$$R_k(\overline{X}, \overline{Y}) = \max\{\mathrm{Dist}(\overline{X}, \overline{Y}), V_k(\overline{Y})\}$$

- Not symmetric
- Equal to simple distance for long distances
- Smoothed by $V_k(\overline{X})$ for short distances



Local Outlier Factor (LOF) (cont.)

- Reachability distance

$$R_k(\overline{X}, \overline{Y}) = \max\{\mathrm{Dist}(\overline{X}, \overline{Y}), V_k(\overline{Y})\}$$

- Average reachability distance

$$AR_k(\overline{X}) = \underset{\overline{Y} \in L_k(\overline{X})}{E}\left[R_k(\overline{X}, \overline{Y})\right]$$

$L_k(\overline{X})$ is the set of points within distance $V_k(\overline{X})$ of $\overline{X}$

(might be more than k due to ties)

Local Outlier Factor (LOF) (cont.)

$$R_k(\overline{X}, \overline{Y}) = \max\{\mathrm{Dist}(\overline{X}, \overline{Y}), V_k(\overline{Y})\}$$

$$AR_k(\overline{X}) = \underset{\overline{Y} \in L_k(\overline{X})}{E} \left[R_k(\overline{X}, \overline{Y})\right]$$

- Local outlier factor

$$\mathrm{LOF}_k(\overline{X}) = \underset{\overline{Y} \in L_k(\overline{X})}{E} \frac{AR_k(\overline{X})}{AR_k(\overline{Y})}$$

Outlier score

$$\max_k \mathrm{LOF}_k(\overline{X})$$

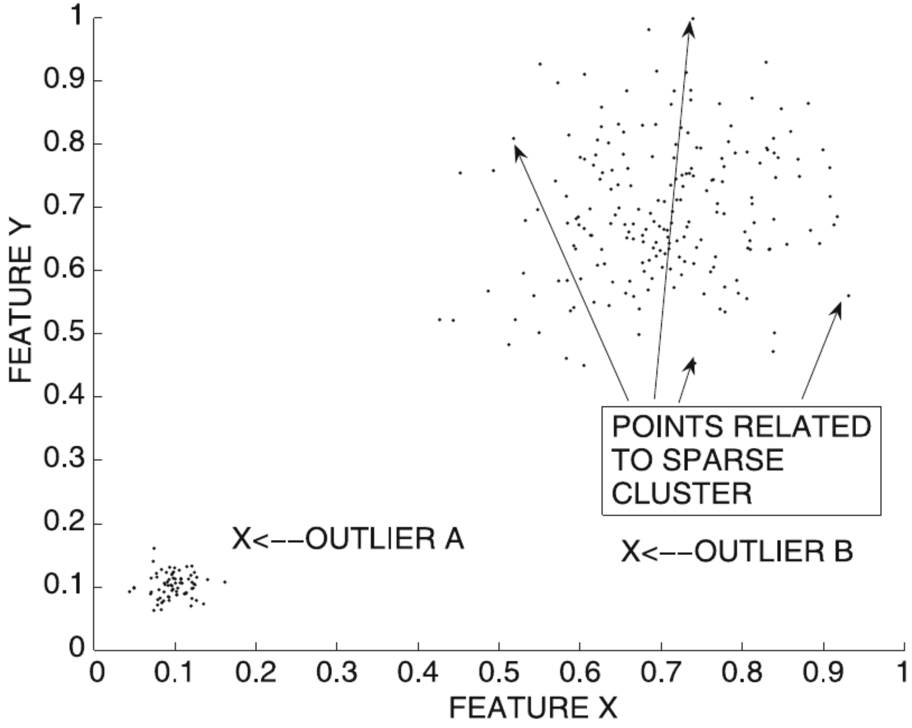
- Large for outliers, close to 1 for others

Local Outlier Factor (LOF) (cont.)

- Local outlier factor

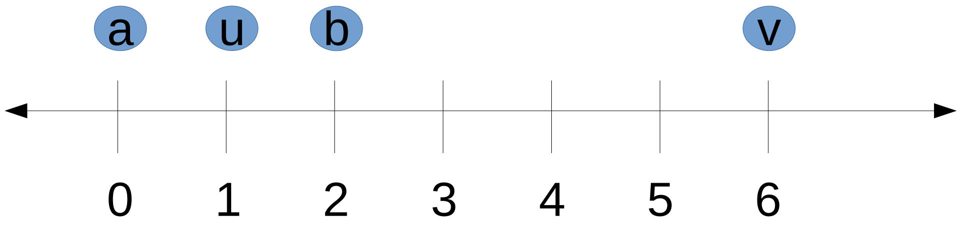
$$\mathrm{LOF}_k(\overline{X}) = \underset{\overline{Y} \in L_k(\overline{X})}{E} \frac{AR_k(\overline{X})}{AR_k(\overline{Y})}$$

- LOF values for points inside a cluster are close to one if cluster is homogeneous
- LOF values much higher for outliers: they are computed in terms of average distances of near-by clusters



Exercise

compare outlier score LOF(u), LOF(v)



- Let k=2
- $LOF_2(u) = E[\ \{AR_2(u)\ /AR_2(a),\ AR_2(u)/AR_2(b)\}] =$ ________
- $LOF_2(v) = E[\ \{AR_2(v)\ /AR_2(b),\ AR_2(v)/AR_2(u)\}] =$ ________
- $AR_2(u) = E[\ \{R_k(u,a),\ R_k(u,b)\ \}] =$ _______
- $AR_2(v) = E[\ \{R_k(v,b),\ R_k(v,u)\ \}] =$ _______
- $AR_2(a) = E[\ \{R_k(a,u),\ R_k(a,b)\ \}] =$ ______
- $AR_2(b) = E[\ \{R_k(b,u),\ R_k(b,a)\ \}] =$ ______
- **$R_k(a,u)$ = ____;** $R_k(a,b) =$ _____; $R_k(b,u) =$ ______; $R_k(b,a) =$ _______
- $R_k(u,a) =$ _____; $R_k(u,b) =$ ______; $R_k(v,b) =$ _______; $R_k(v,u) =$ ______
- V_2 = distance to 2nd nearest neighbor: $V_2(u) =$ _____ ; $V_2(v) =$ _____; $V_2(a) =$ _____ ; $V_2(b) =$ _____

$$\mathrm{LOF}_k(\overline{X}) = \underset{\overline{Y} \in L_k(\overline{X})}{E} \frac{AR_k(\overline{X})}{AR_k(\overline{Y})}$$

$$AR_k(\overline{X}) = \underset{\overline{Y} \in L_k(\overline{X})}{E} \left[R_k(\overline{X}, \overline{Y}) \right]$$

$$R_k(\overline{X}, \overline{Y}) = \max\{\mathrm{Dist}(\overline{X}, \overline{Y}), V_k(\overline{Y})\}$$