5.2 – Quantitative Methods of Anomaly Detection

Cluster-based methods are not the only types of UL anomaly detection methods.

- **Distance-based methods:** distance to all points, distance to k nearest neighbours (kNN), average distance to kNN, median distance to kNN, etc.
- **Density-based methods:** local outlier factor (LOF), isolation forest, HDBSCAN, etc.

5.2.1 – Distance-Based Methods

We find anomalous observations by comparing them to other observations (**anomalies are relative, not absolute**).

In the **distance-based context**, the natural way to compare observations is to consider their **distance from a subset of observations**: increasing distance being increasingly **suggestive** of anomalous status.

Requirement: a **distance function** or a **pre-computed table of pair-wise distances** (in discrete case).

The choice of subsets and distance functions distinguish the different distance-based algorithms.

Notation

- $D \subset \mathbb{R}^n$ is an n-dimensional (numerical) data set
- $\mathbf{p}, \mathbf{q} \in D$ are specific observations in D
- $P \subset D$ is a subset of D
- $d : D \times D \to \mathbb{R}$ is a distance function on $D \subset \mathbb{R}^n$
- the distance between $\mathbf{p}$ and $\mathbf{q}$ is written $d(\mathbf{p}, \mathbf{q})$

- the output of an anomaly detection algorithm is a function $a : D \to \mathbb{R}$
- $a(\mathbf{p})$ is a number that describes how anomalous $\mathbf{p}$ is
- if $a(\mathbf{p}) < a(\mathbf{q})$ for $\mathbf{p}, \mathbf{q} \in D$, then $\mathbf{p}$ is **less anomalous** than $\mathbf{q}$
- $\alpha \in \mathbb{R}$ is the **absolute anomaly threshold**
- any $\mathbf{p} \in D$ for which $a(\mathbf{p}) > \alpha$ is **absolutely anomalous**

Similarity Measures

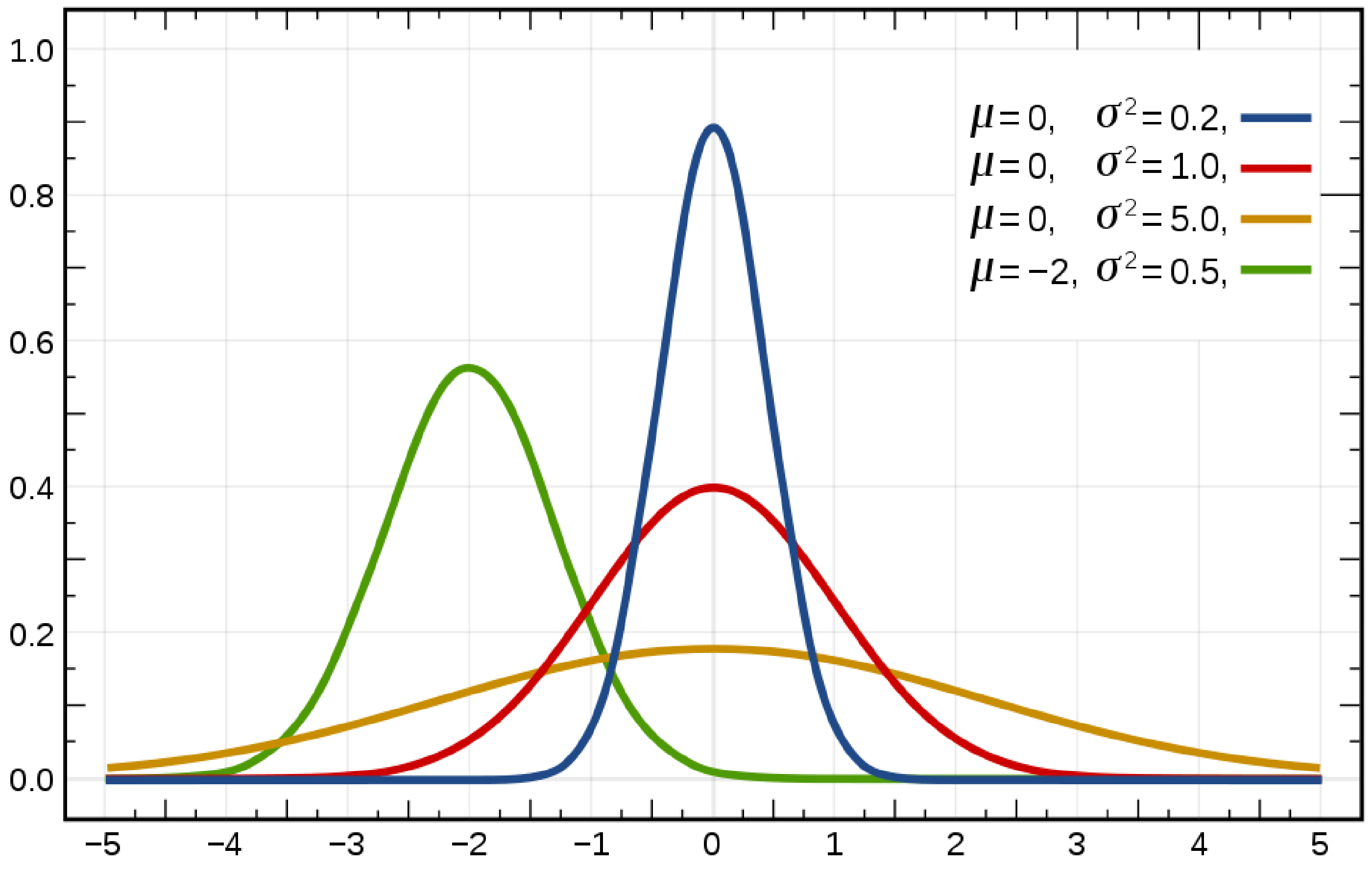
A **similarity measure** is a real-valued function that describes the **similarity between two objects**.

A common construction for the similarity w between two points $\mathbf{p}, \mathbf{q}$:

$$w(\mathbf{p}, \mathbf{q}) = \frac{1}{1 + d(\mathbf{p}, \mathbf{q})}, \quad \text{for some distance } d.$$

Note: $w \to 1$ as $d \to 0$, and $w \to 0$ as $d \to \infty$.

Similarity measures can also be constructed between **probability distributions** (see Hellinger distance).


$\mu=0, \quad \sigma^2=0.2,$
$\mu=0, \quad \sigma^2=1.0,$
$\mu=0, \quad \sigma^2=5.0,$
$\mu=-2, \quad \sigma^2=0.5,$
1.0
0.8
0.6
0.4
0.2
0.0
−5
−4
−3
−2
−1
0
1
2
3
4
5

We can think of a single point $\mathbf{p}$ as a probability distribution (with 0% chance of drawing another point).

The distance between that point and any other distribution with mean μ and covariance matrix $\mathbf{\Sigma}$ can be given using the **Mahalanobis framework**:

$$M(\mathbf{p}) = \sqrt{(\mathbf{p}-\mu)^{\top}\mathbf{\Sigma}^{-1}(\mathbf{p}-\mu)} \quad \text{(BACON)}.$$

Alternatively, if $\mathbf{p}$ and $\mathbf{q}$ are drawn from the same distribution with covariance Σ, then the Mahalanobis distance is a dissimilarity measure between $\mathbf{p}$ and $\mathbf{q}$:

$$d_M(\mathbf{p},\mathbf{q}) = \sqrt{(\mathbf{p}-\mathbf{q})^{\top}\mathbf{\Sigma}^{-1}(\mathbf{p}-\mathbf{q})}.$$

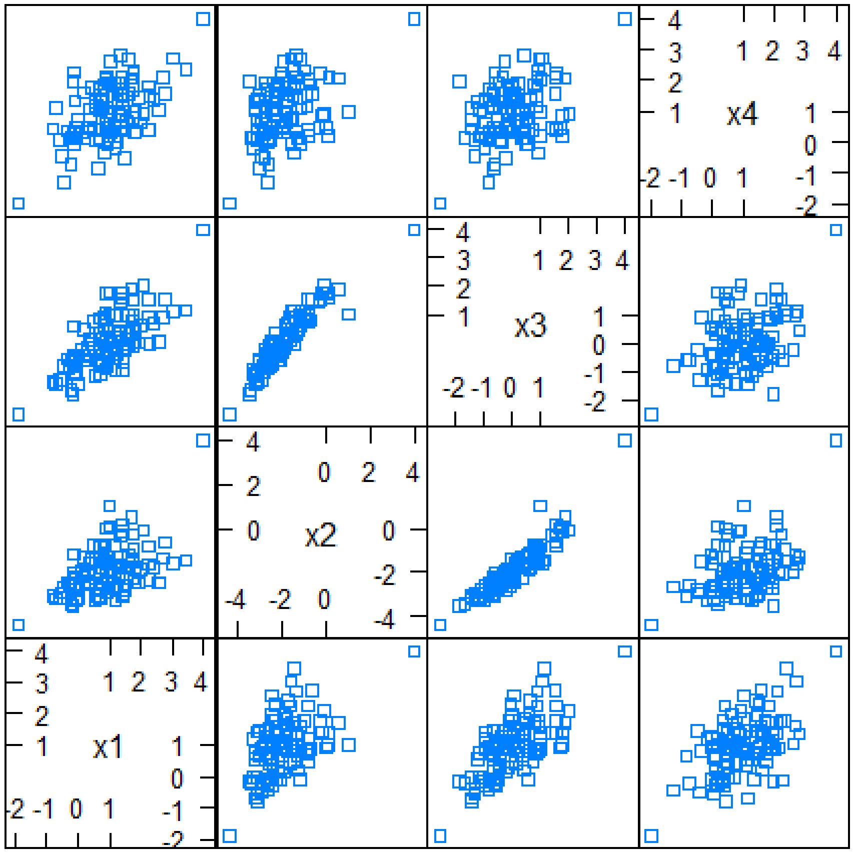
Example: consider a 4D-dataset drawn from a multivariate $\mathcal{N}(\mu, \mathbf{\Sigma})$ with

$$\mu = (1, -2, 0, 1), \quad \mathbf{\Sigma} = \begin{pmatrix} 1 & 0.5 & 0.7 & 0.5 \\ 0.5 & 1 & 0.95 & 0.3 \\ 0.7 & 0.95 & 1 & 0.3 \\ 0.5 & 0.3 & 0.3 & 1 \end{pmatrix}.$$

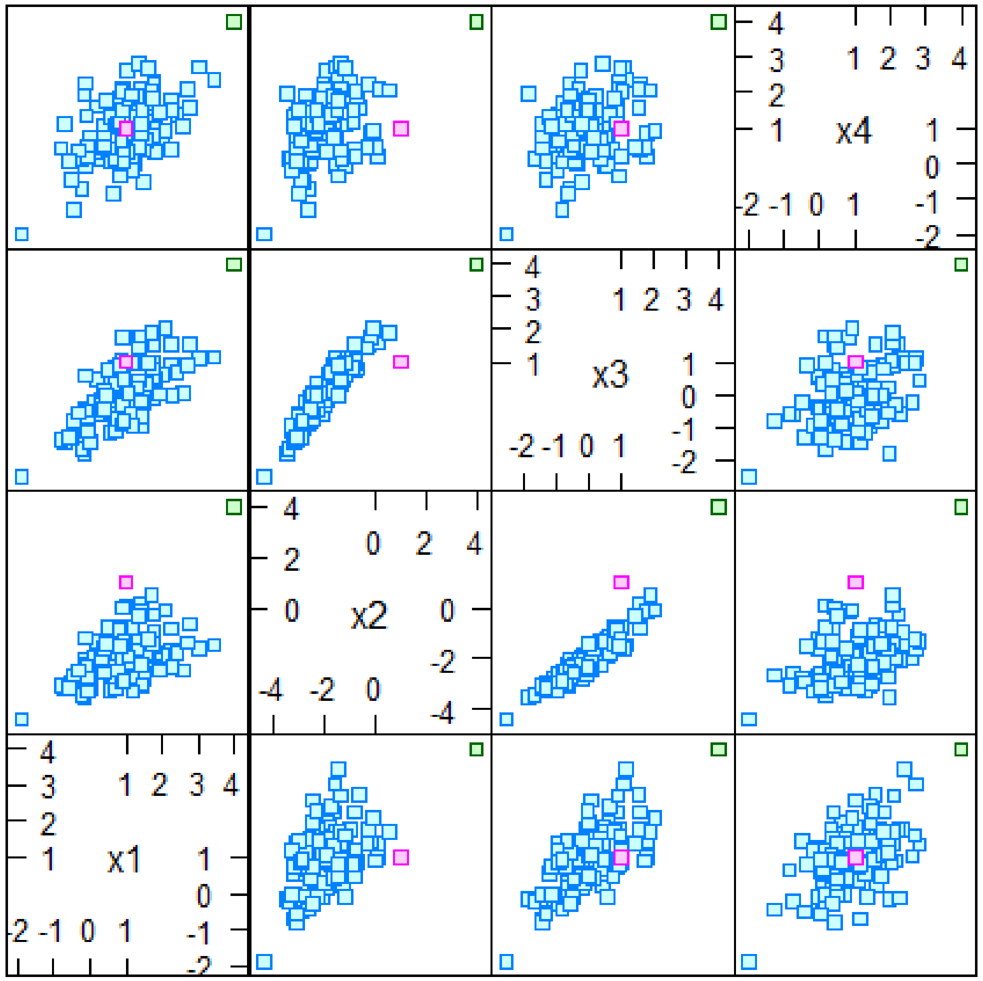
100 observations $\mathbf{p}_1$ to $\mathbf{p}_{100}$ are "normal":

stat	x_1	x_2	x_3	x_4
min	-1.9049	-4.4113	-2.5324	-1.9949
Q_1	0.3812	-2.6464	-0.6190	0.3361
med	0.9273	-2.0220	-0.0506	0.9381
avg	0.9374	-1.9788	0.0071	0.9438
Q_3	1.4615	-1.4002	0.6296	1.5906
max	3.4414	0.5223	2.0265	2.8073

2 observations are "anomalous": $\mathbf{z}_1 = (1, 1, 1, 1)$, $\mathbf{z}_4 = (4, 4, 4, 4)$.



Visually, it seems there might be 3 outliers.


x4
x3
x2
x1

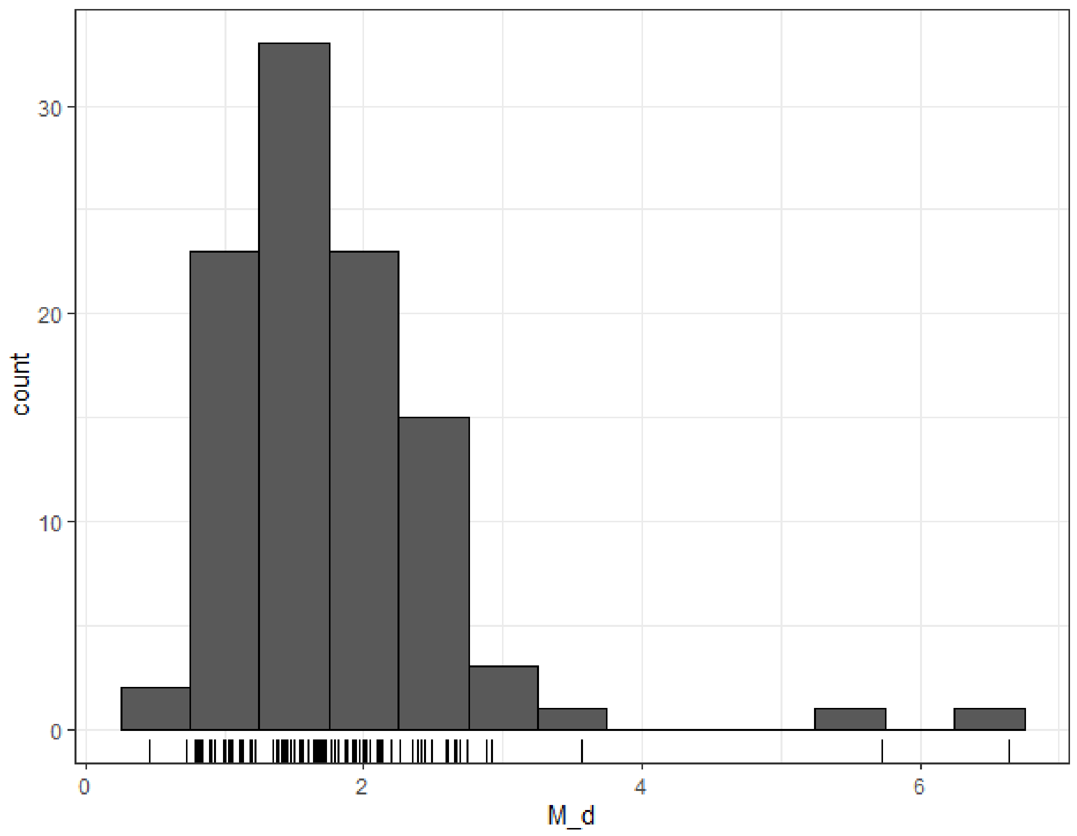
In general, the mean vector and the covariance structure must be estimated from the data:

$$\hat{\mu} = (0.968, -1.891, 0.056, 0.974), \quad \hat{\mathbf{\Sigma}} = \begin{pmatrix} 0.900 & 0.569 & 0.665 & 0.503 \\ 0.569 & 1.312 & 1.069 & 0.469 \\ 0.665 & 1.069 & 0.992 & 0.397 \\ 0.503 & 0.469 & 0.397 & 0.904 \end{pmatrix}.$$

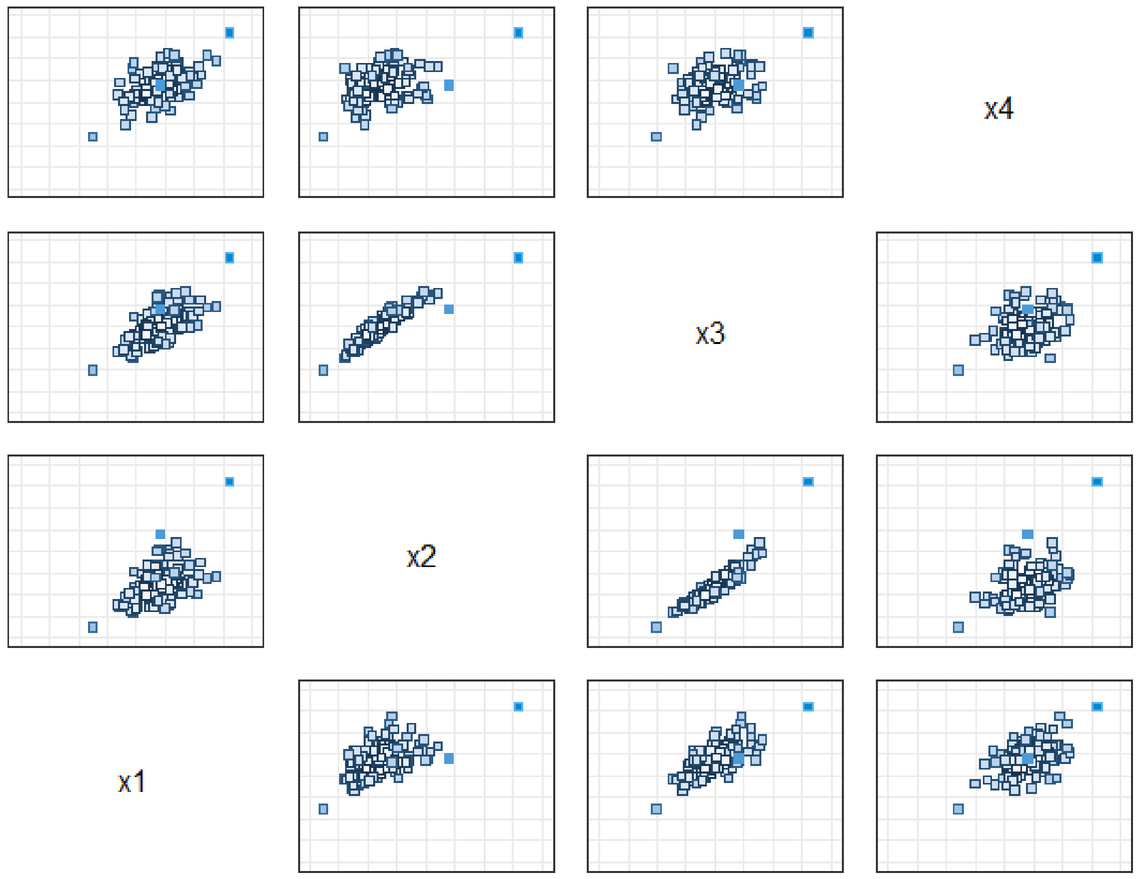
These are distinct from μ and $\mathbf{\Sigma}$, but close enough to be explained by

- sampling variation
- $\mathbf{z}_1, \mathbf{z}_4 \not\sim \mathcal{N}(\mu, \mathbf{\Sigma})$

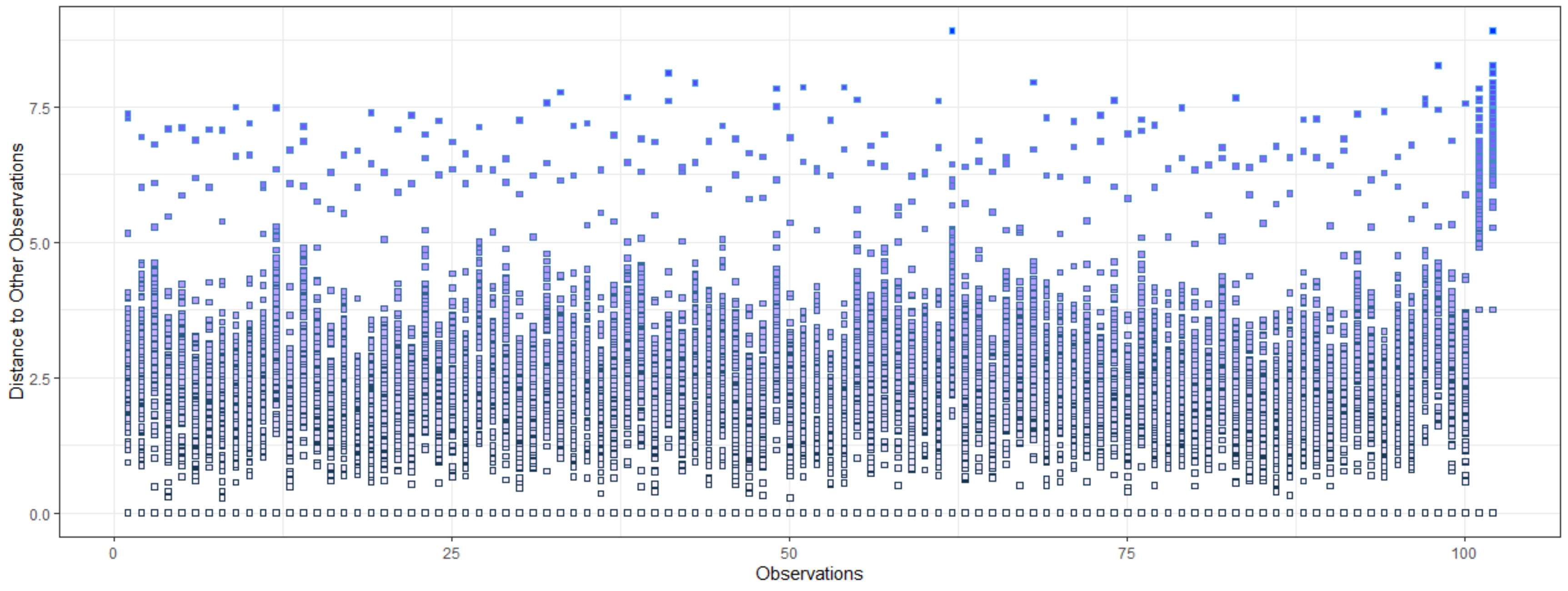
To identify anomalous observations, compute the Mahalanobis distance from all points to the empirical distribution, and between all pairs.



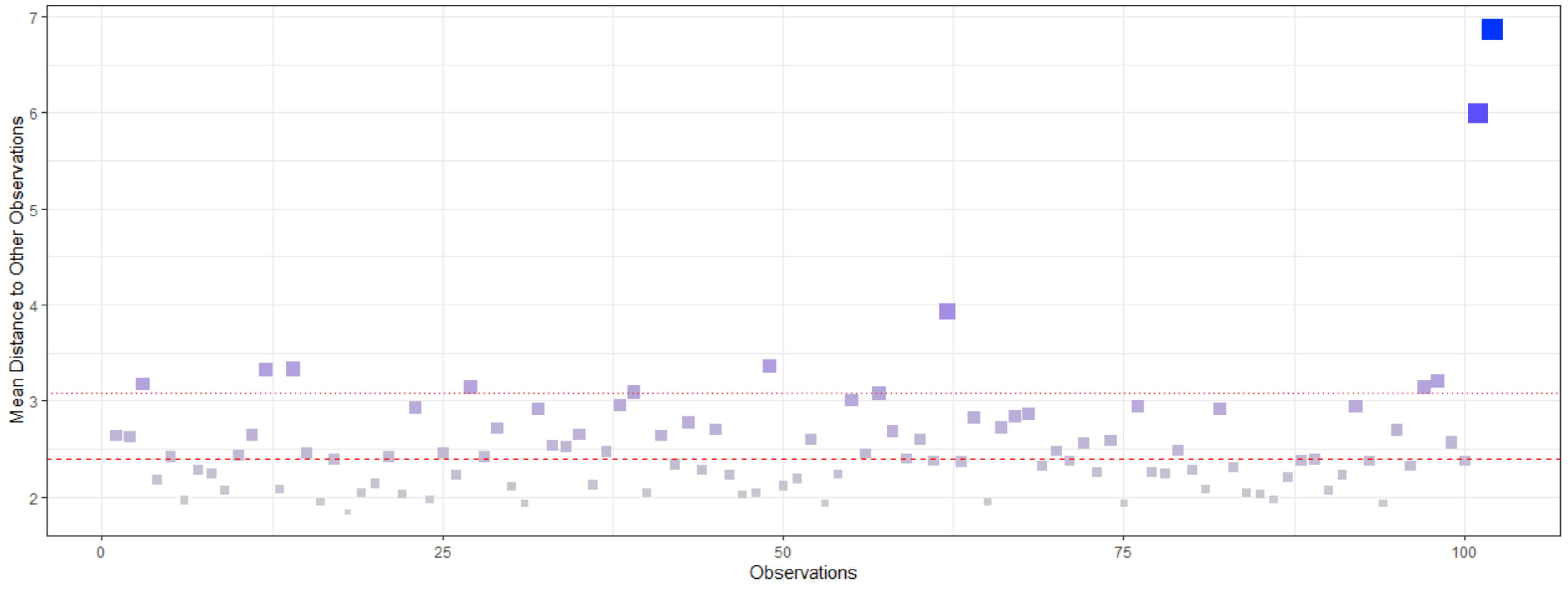
Histogram of Mahalanobis distances to empirical distribution.



Scatter plot of Mahalanobis distances to empirical distribution.



Mahalanobis distances between each pair (empirical distribution). Notice observations 101 and 102, as well as the diffuse cloud of points above the value 5.0.



Mean Mahalanobis distances between each pair (empirical distribution). Notice observations 101 and 102 again. The red lines represent the median mean distance, and 1 standard deviation the median mean distance. The Mahalanobis framework seems to identify 2 outliers.

If Σ is diagonal, then

$$d_M(\mathbf{p},\mathbf{q}) = \sqrt{\sum_{i=1}^{n} \frac{(p_i - q_i)^2}{\sigma_i^2}},$$

where σ_i^2 is the variance along the i-th dimension.

If Σ is the identity matrix, then we recover the **Euclidean distance**

$$d_2(\mathbf{p},\mathbf{q}) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}.$$

In an anomaly detection context, a **linear normalization** is usually applied to each dimension so that each entry lies in the hypercube $[-1,1]^n$.

The **Minkowski distance** of order p is a generalization of the Euclidean distance:

$$d_p(\mathbf{p},\mathbf{q}) = \left(\sum_{i=1}^{n} |p_i - q_i|^p\right)^{1/p}.$$

- For $p = 1$, we recover the **Manhattan distance**:

$$d_1(\mathbf{p},\mathbf{q}) = \sum_{i=1}^{n} |p_i - q_i|;$$

- for $p = \infty$, we recover the **supremum** (Chebychev) **distance**

$$d_\infty(\mathbf{p},\mathbf{q}) = \max_{i=1}^{n} \left\{|p_i - q_i|\right\}.$$

The Minkowski distance d_p is only an actual distance function (a **metric**) when $p \geq 1$, but an exception is made for

$$d_{-\infty}(\mathbf{p}, \mathbf{q}) = \min_{i=1}^{n} \{|p_i - q_i|\} .$$

When working with categorical data (such as in one-hot encoding of text), it can be useful to use distances for binary vectors.

Let $\mathbf{p}, \mathbf{q} \in \{0, 1\}^n$.

The **Hamming distance** between $\mathbf{p}$ and $\mathbf{q}$ counts the number of positions where they differ:

$$d_H(\mathbf{p}, \mathbf{q}) = \sum_{i=1}^{n} |p_i - q_i|.$$

The **Jaccard similarity** of two datasets P and Q, is defined as the size of their intersection divided by the size of their union

$$J(P,Q) = \frac{|P \cap Q|}{|P \cup Q|} = \frac{|P \cap Q|}{|P| + |Q| - |P \cap Q|}$$

Their **Jaccard distance** is $d_J(P,Q) = 1 - J(P,Q)$. This can be extended to binary vectors $\mathbf{p}$ and $\mathbf{q}$.

Consider an arbitrary set $D = \{x_1, x_2, \ldots, x_n\}$. We build P as follows: if $p_i = 1$ then $x_i \in P$; otherwise $x_i \notin P$. Similarly for Q.

Then $|P| = \sum p_i$, $|Q| = \sum q_i$, $|P \cap Q| = \sum p_i q_i = \mathbf{p} \cdot \mathbf{q}$ and

$$d_J(\mathbf{p},\mathbf{q}) = d_J(P,Q) = 1 - J(P,Q) = 1 - \frac{\mathbf{p} \cdot \mathbf{q}}{\sum (p_i + q_i) - \mathbf{p} \cdot \mathbf{q}}.$$

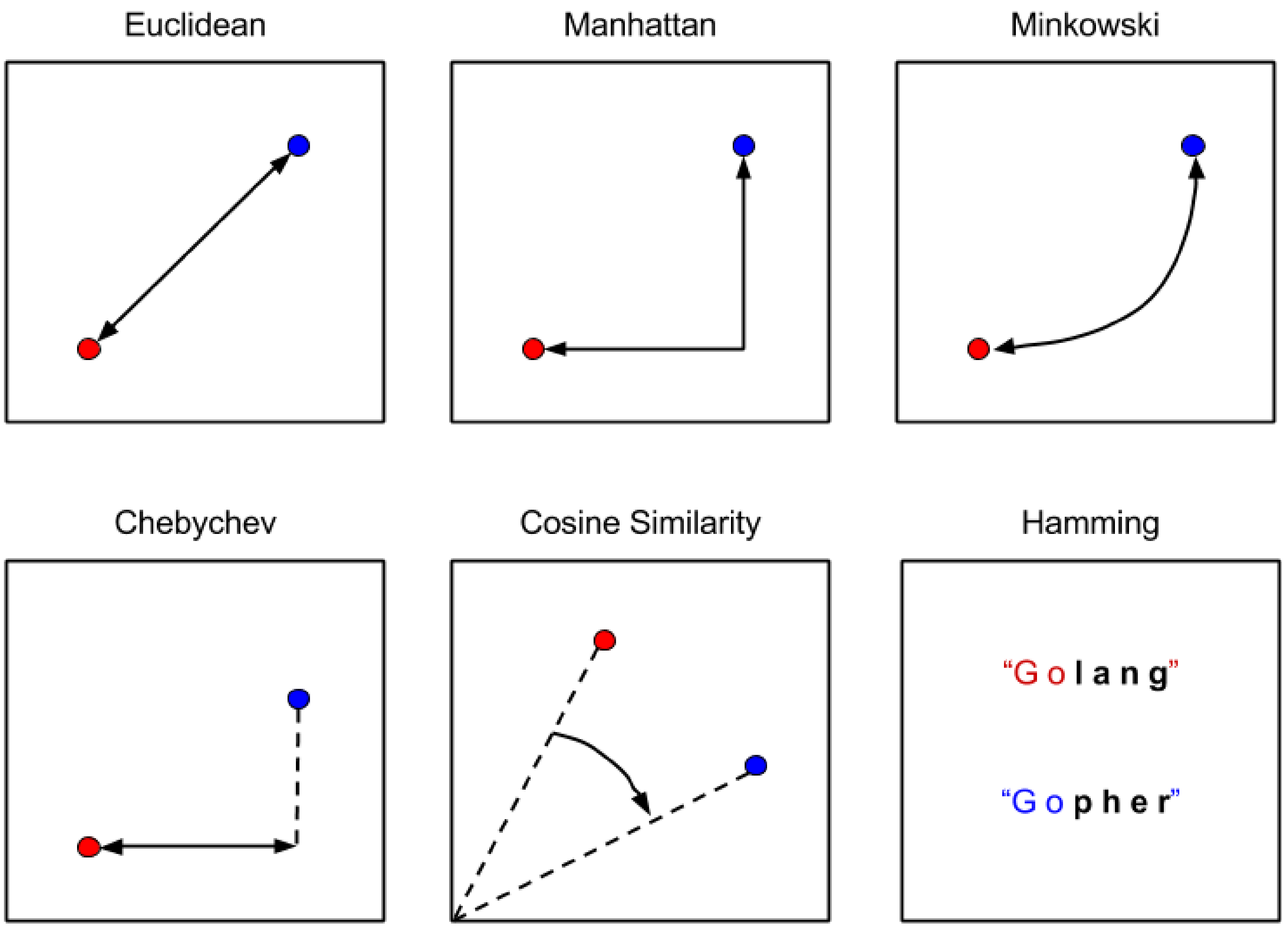
Finally, let $\mathbf{p}, \mathbf{q} \neq \mathbf{0}$. Recall that $\mathbf{p} \cdot \mathbf{q} = \|p\|\|q\| \cos\theta$, where θ is the angle between $\mathbf{p}$ and $\mathbf{q}$.

The **cosine similarity** between $\mathbf{p}$ and $\mathbf{q}$ is

$$\cos\theta = \frac{p \cdot q}{\|p\|\|q\|} = \frac{\sum_{i=1}^{n} p_i q_i}{\sqrt{\sum_{i=1}^{n} p_i^2}\sqrt{\sum_{i=1}^{n} q_i^2}}.$$

This also holds $\mathbf{p}, \mathbf{q}$ are non-binary. The value ranges between 1 and -1:

- $\cos\theta = 1$ when $\mathbf{p} = \mathbf{q}$;
- $\cos\theta = -1$ when $\mathbf{p} = -\mathbf{q}$, and
- $\cos\theta = 0$ when $\mathbf{p}$ and $\mathbf{q}$ are perpendicular.


Euclidean
Manhattan
Minkowski
Chebychev
Cosine Similarity
Hamming
"Golang"
"Gopher"

Distance-Based Approaches

Finding the right distance function to use for anomaly detection is **NOT AN EASY TASK** – contextual understanding and domain expertise are required.

Any such distance function can be used as the basis for anomaly detection algorithms (the ideas can also be extended to more complex algorithms).

Given some distance function d, dataset D, and integers $k, \nu \leq |D|$, the **distance to all points** (DTAP) anomaly detection algorithm considers each point $\mathbf{p}$ in D and adds the distance from $\mathbf{p}$ to every other point in D:

$$a(\mathbf{p}) = \sum_{\mathbf{q}\neq\mathbf{p}\in D} d(\mathbf{q}, \mathbf{p}).$$

The ν points with largest values for a are then said to be **anomalous according to** a.

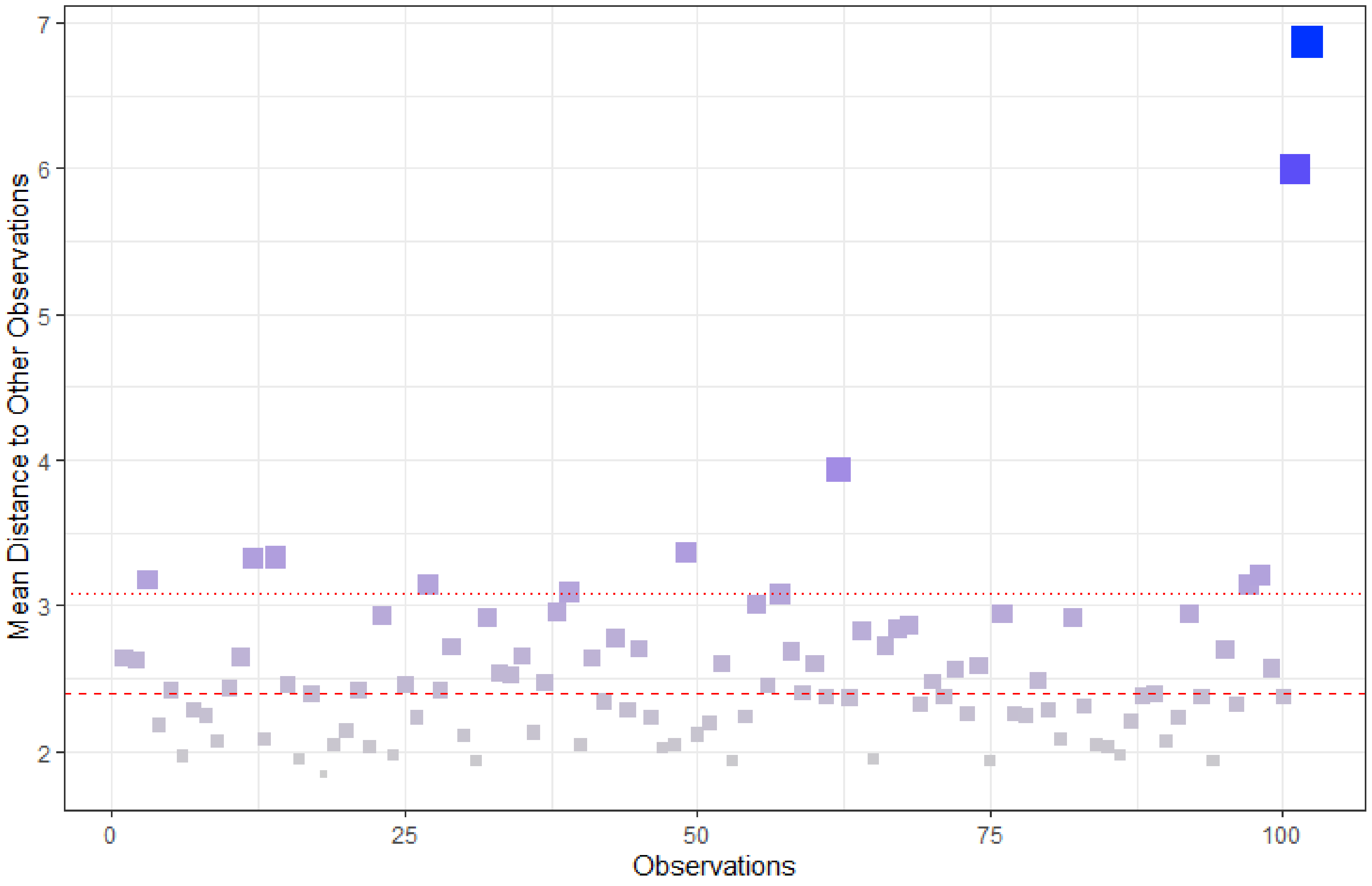
This approach often selects the most extreme observations as anomalous, which may be of limited use in practice.

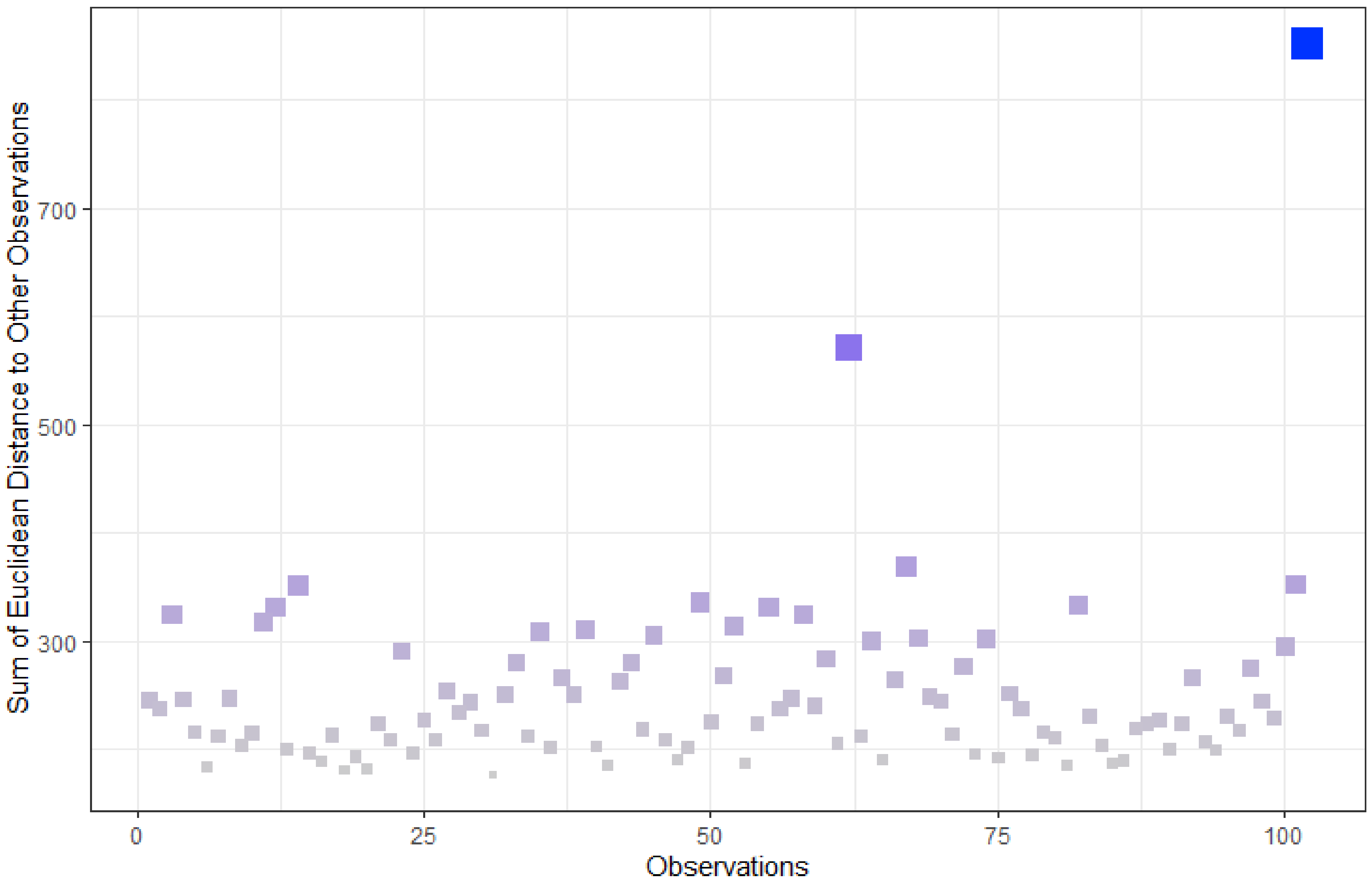
The **distance to nearest neighbour** (DTNN) algorithm uses

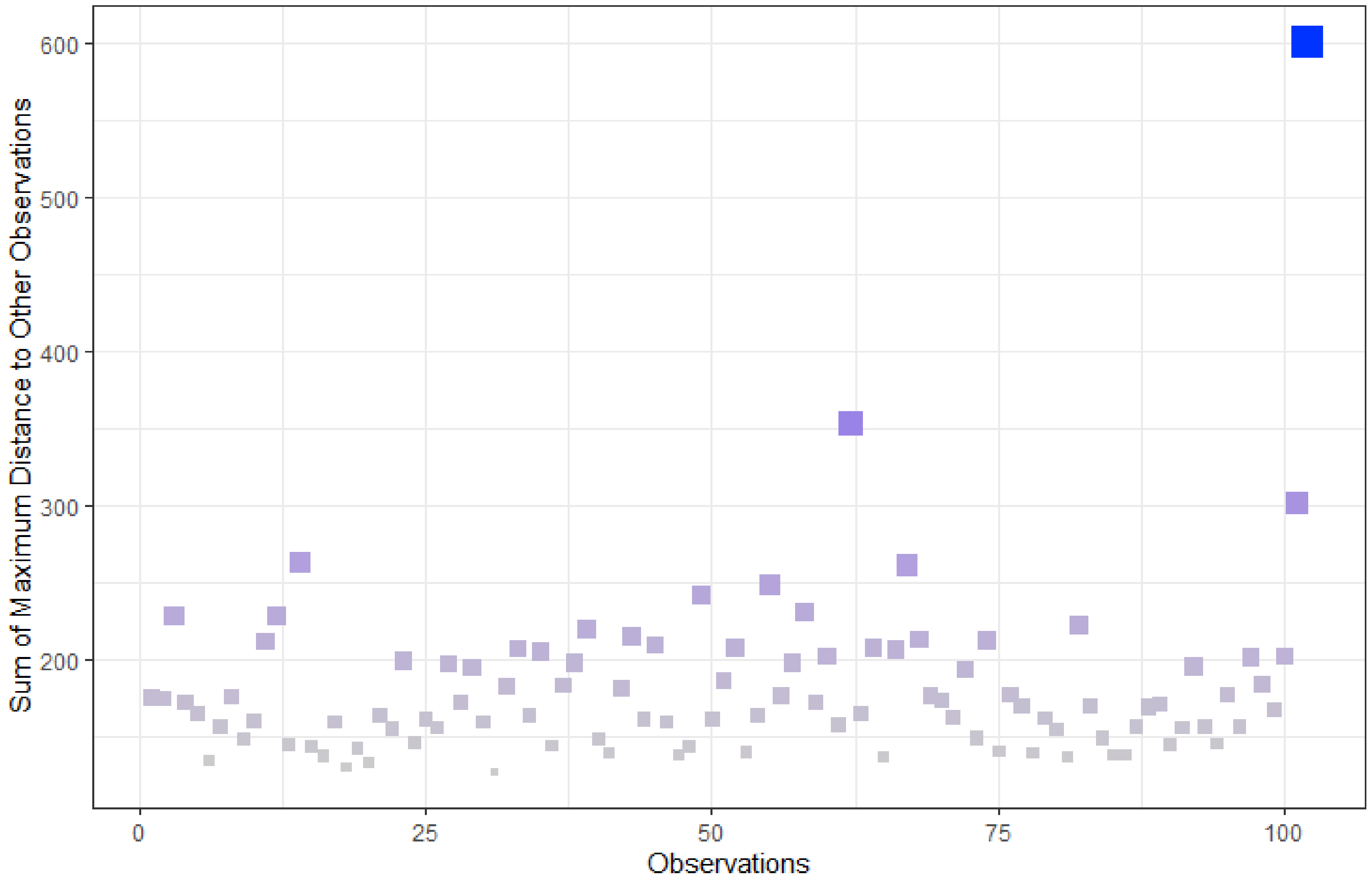
$$a(\mathbf{p}) = \min_{\mathbf{q}\neq\mathbf{p}\in D}\{d(\mathbf{q},\mathbf{p})\},$$

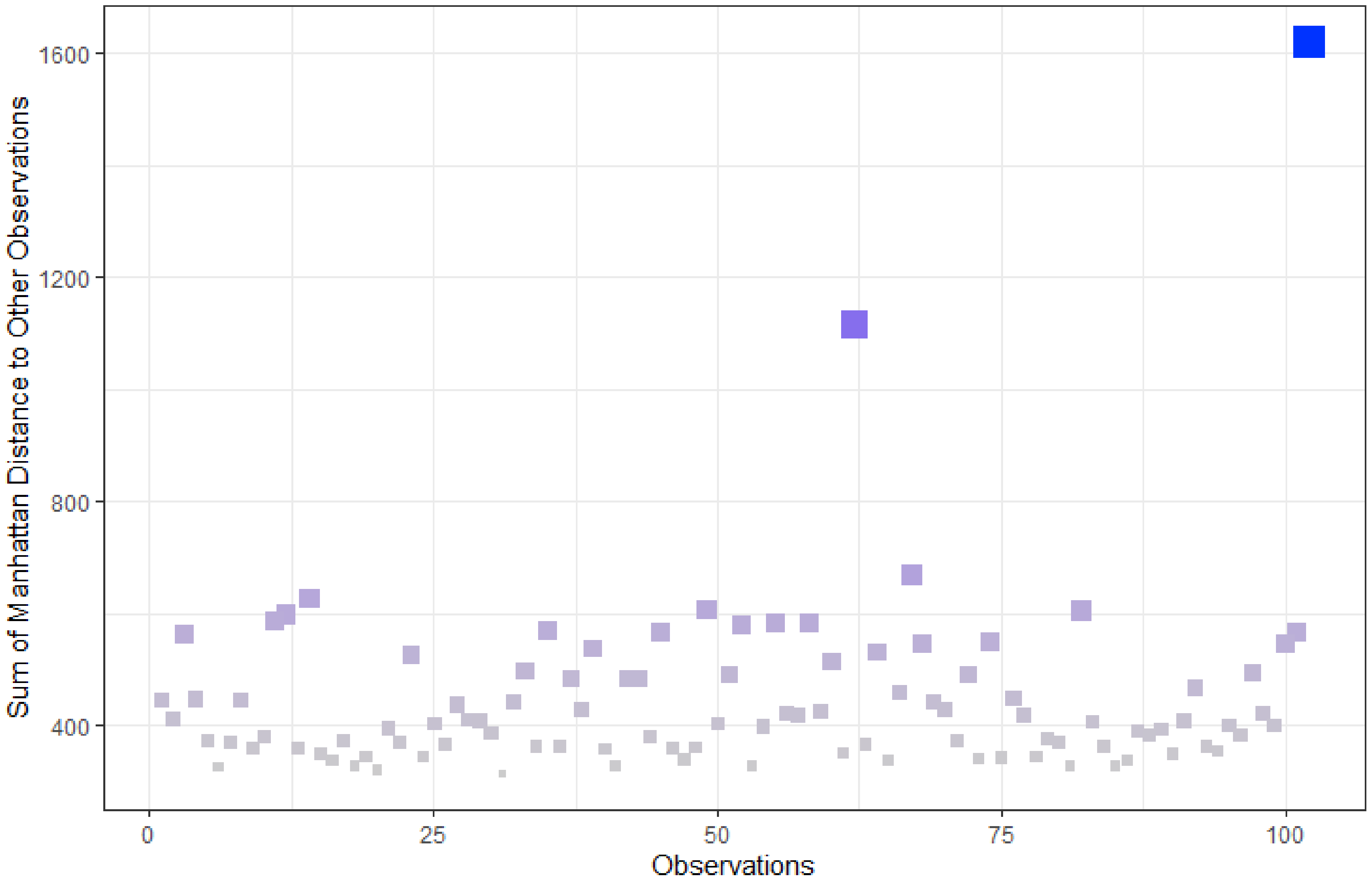
with a similar definition for the ν anomalous points.

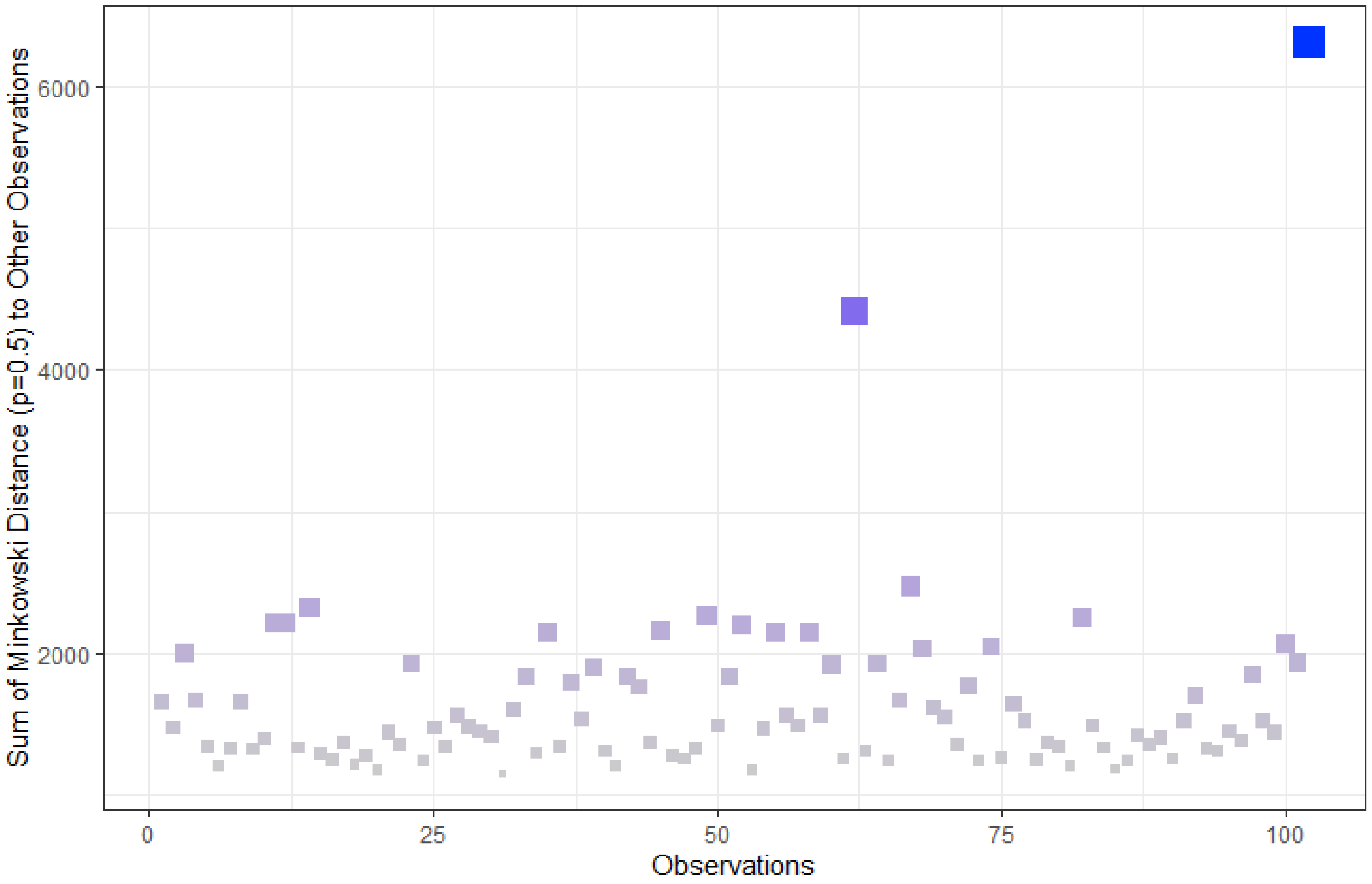
The **average distance to** k **nearest neighbours** and **median distance to** k **nearest neighbours** are defined similarly.

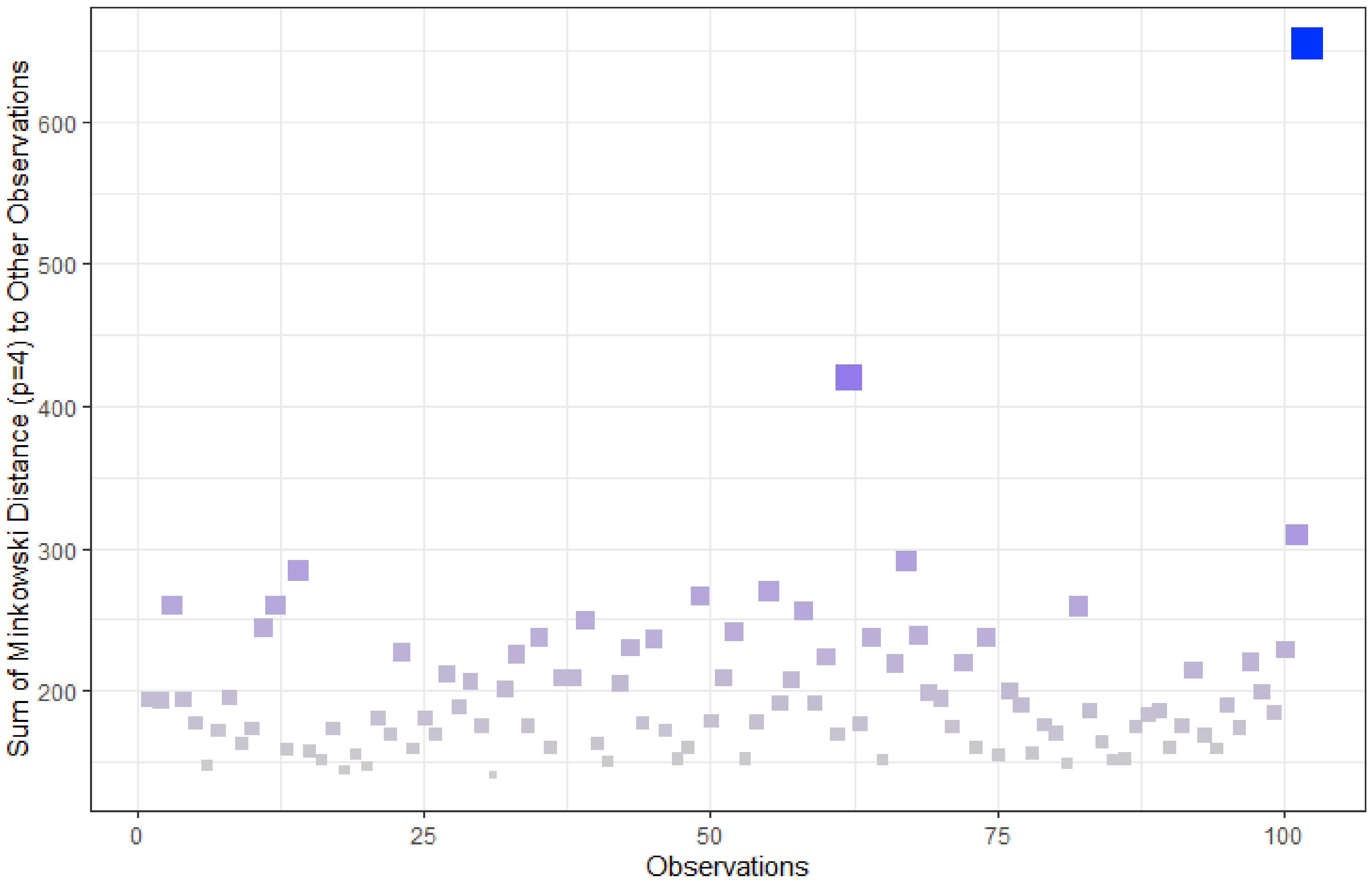

Mean Distance to Other Observations
Observations
7
6
5
4
3
2
0
25
50
75
100


Sum of Euclidean Distance to Other Observations
700
500
300
0
25
50
75
100
Observations


600
500
400
300
200
Sum of Maximum Distance to Other Observations
0
25
50
75
100
Observations


Sum of Manhattan Distance to Other Observations
1600
1200
800
400
0
25
50
75
100
Observations


Sum of Minkowski Distance (p=0.5) to Other Observations
6000
4000
2000
0
25
50
75
100
Observations


Sum of Minkowski Distance (p=4) to Other Observations
200
300
400
500
600
0
25
50
75
100
Observations

Mahalanobis	**Euclidean**	**Supremum**
102	102	102
101	62	62
67	67	101
14	101	14
12	14	67

Manhattan	**Minkowski** $(p = 0.5)$	**Minkowski** $(p = 4)$
102	102	102
62	62	62
67	67	101
14	14	67
49	49	14

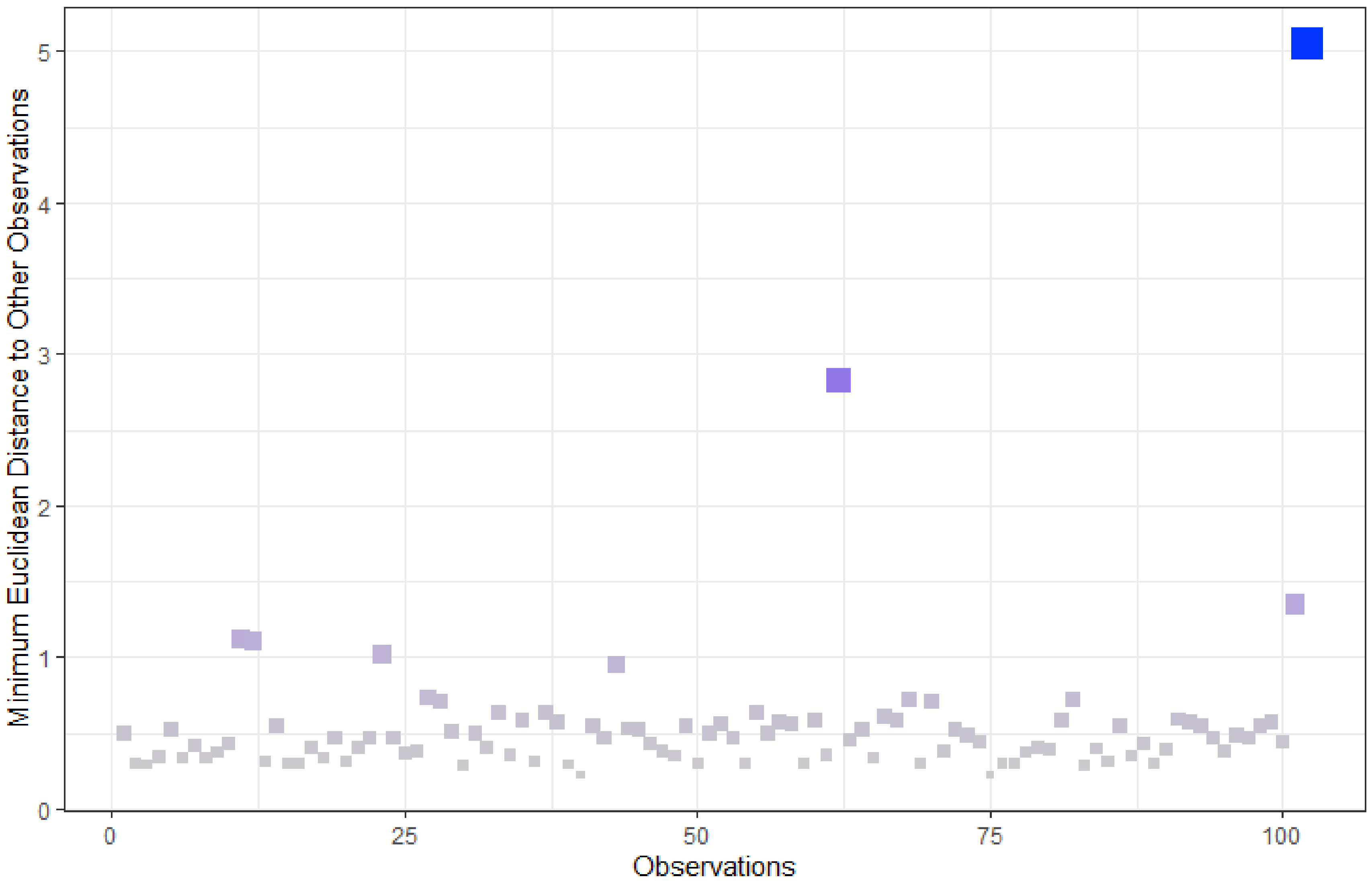
DTAP anomalies $(\nu = 5)$, for various distances; unscaled data.

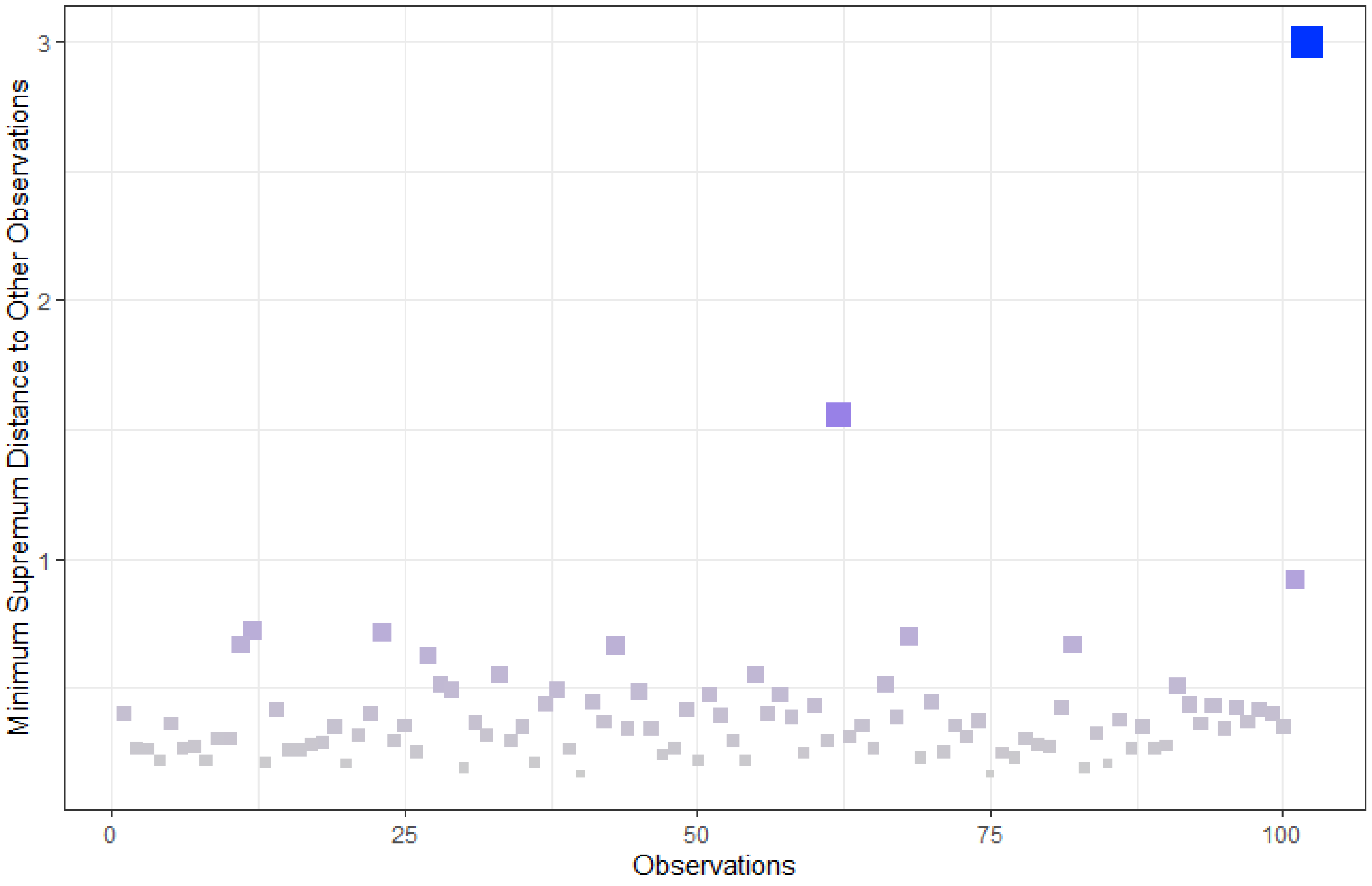
Euclidean	**Supremum**	**Manhattan**
102	102	102
61	62	62
67	14	14
101	55	67
14	49	49

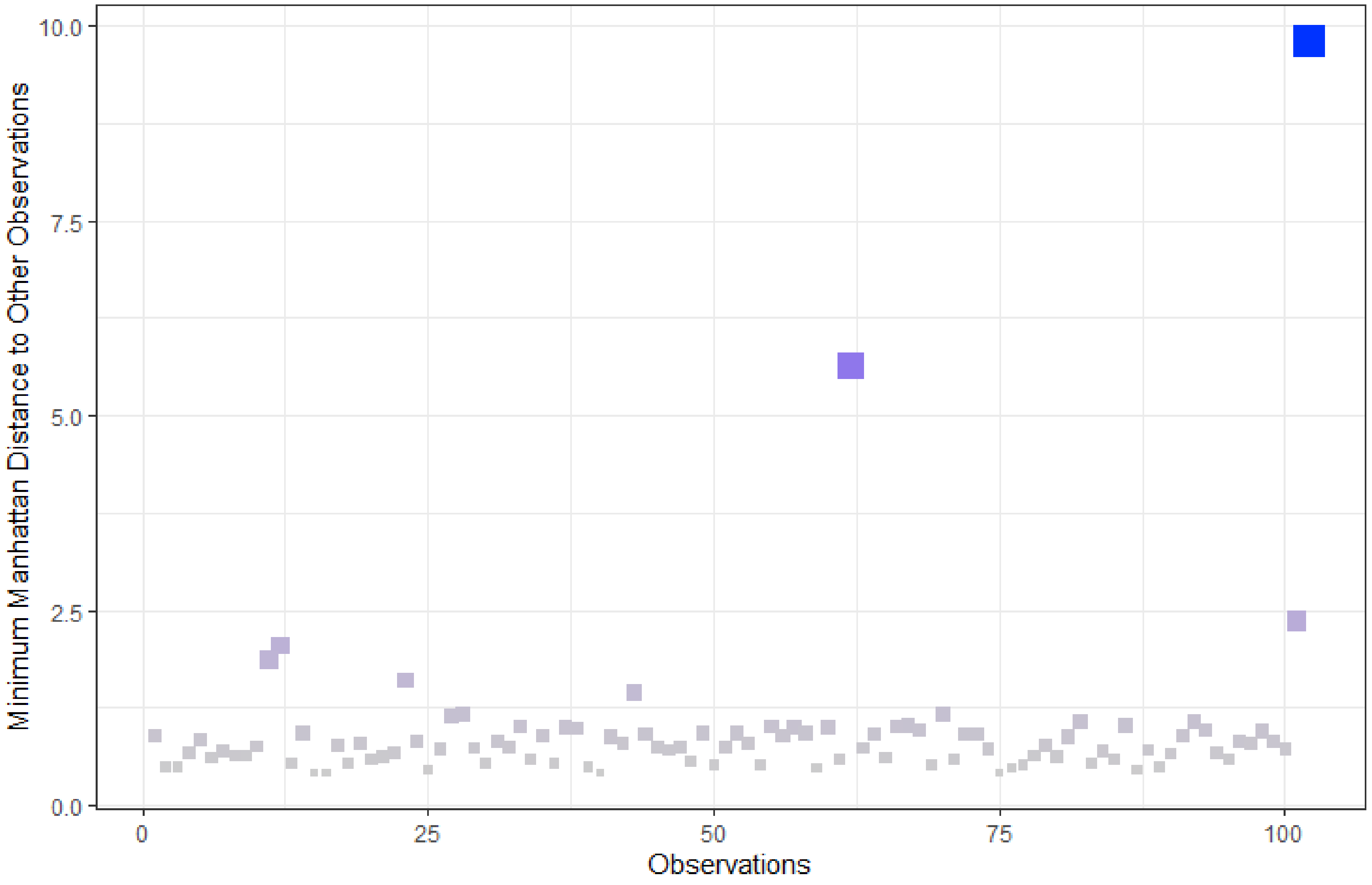
DTAP anomalies ($\nu = 5$), for various distances; scaled data.

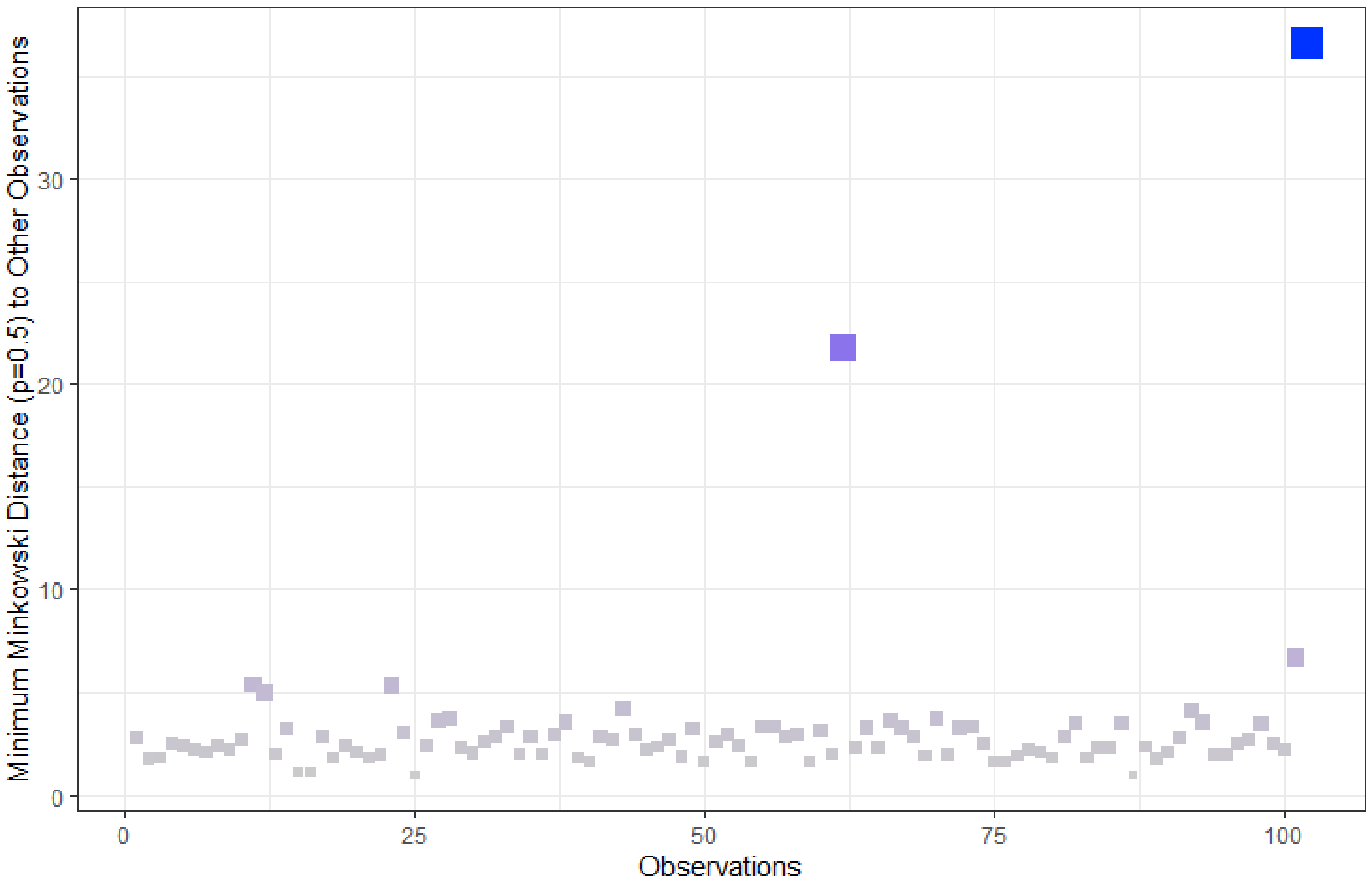
The rankings change according to the selected distance function, the data scaling, and the choice of algorithm (see following slides).

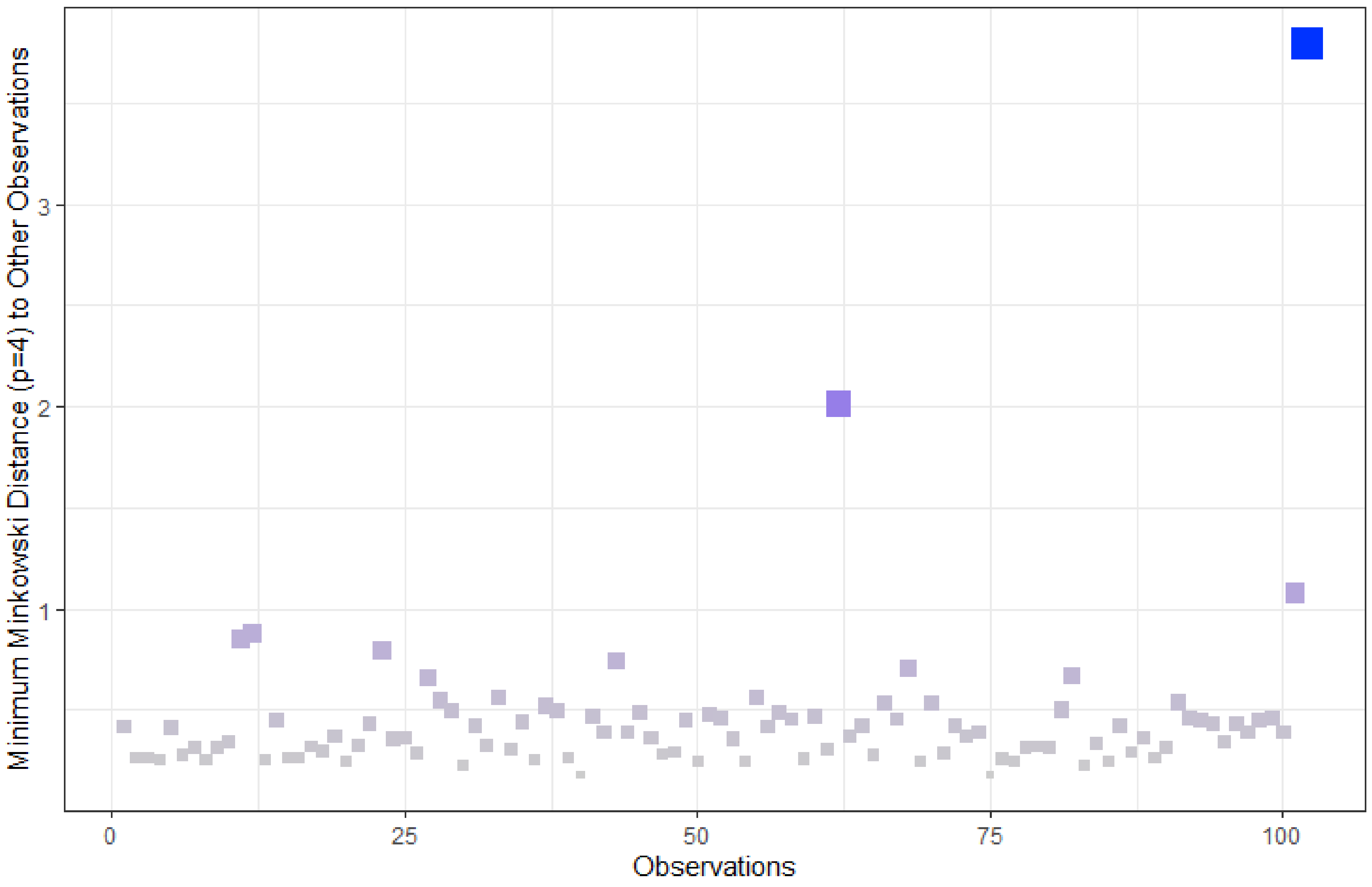
How do we make these decisions, then?


Minimum Euclidean Distance to Other Observations
0
1
2
3
4
5
0
25
50
75
100
Observations


Minimum Supremum Distance to Other Observations
3
2
1
0
25
50
75
100
Observations


Minimum Manhattan Distance to Other Observations
10.0
7.5
5.0
2.5
0.0
0
25
50
75
100
Observations


Minimum Minkowski Distance (p=0.5) to Other Observations
0
10
20
30
0
25
50
75
100
Observations


Minimum Minkowski Distance (p=4) to Other Observations
3
2
1
0
25
50
75
100
Observations

Mahalanobis	**Euclidean**	**Supremum**
102	102	102
101	62	62
67	101	101
14	11	12
12	12	23

Manhattan	**Minkowski** $(p = 0.5)$	**Minkowski** $(p = 4)$
102	102	102
62	62	62
101	101	101
12	11	12
11	23	11

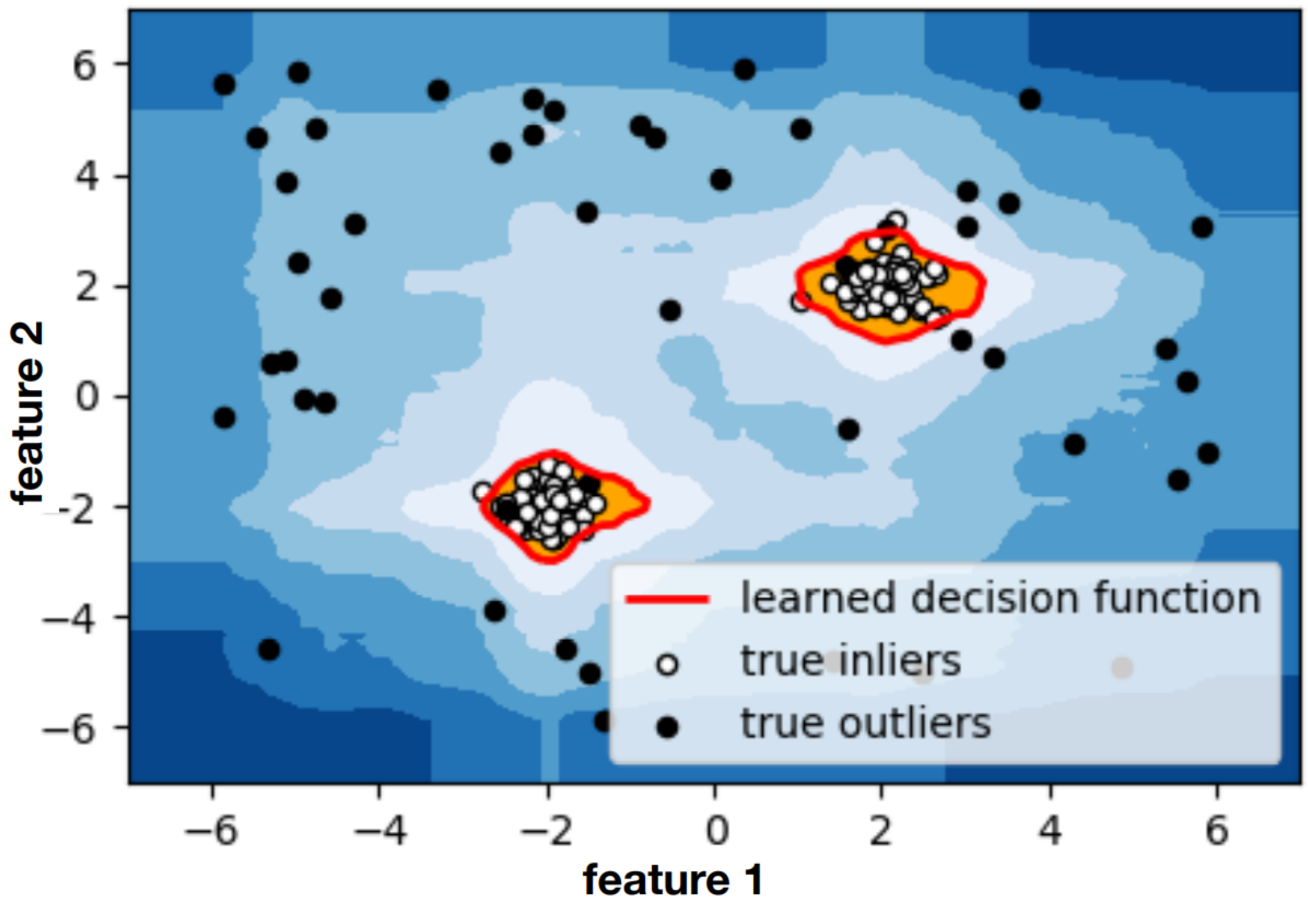
DTNN anomalies $(\nu = 5)$, for various distances; unscaled data.

5.2.2 – Density-Based Methods

The flexibility in the choice of distance functions, scaling, and distance-based anomaly detection algorithm gives rise to different **anomaly rankings**. This is par for the course in the anomaly detection context.

Density-based approaches view points as anomalous if they occur in **low density regions**. Methods include:

- **local outlier factors**;
- **DBSCAN**, and
- **isolation forests**.



Low-density areas as outlier nurseries [Baron].

Local Outlier Factor

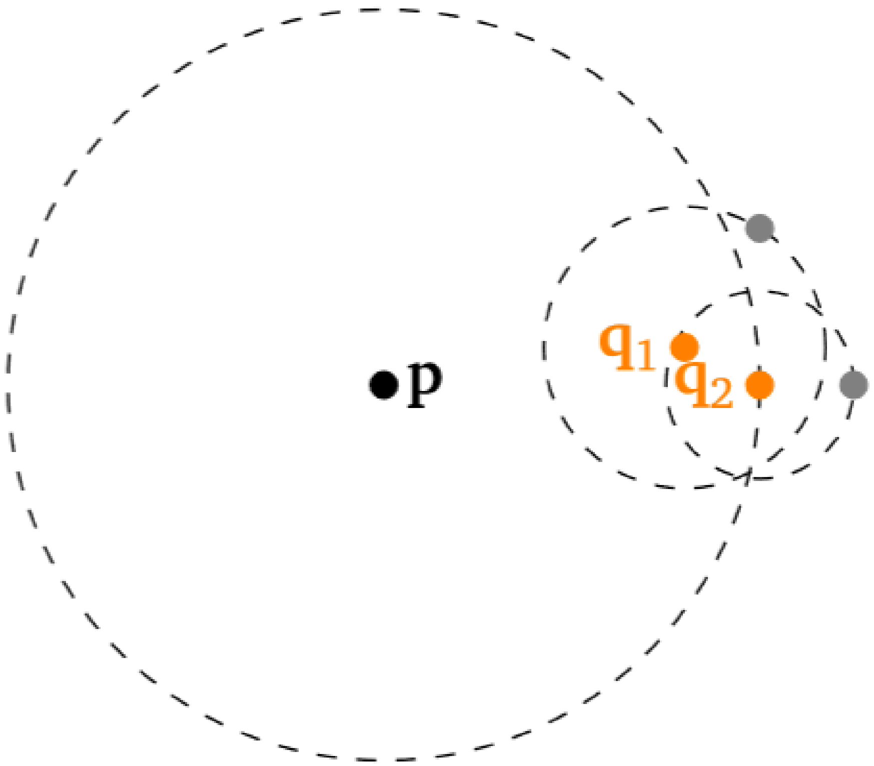
The **Local Outlier Factor** (LOF) algorithm works by measuring the **local deviation** of each observation **from its** k **nearest neighbours**.

An observation is said to be anomalous if the **deviation is large**.

A **local** $k-$**region** around a point $\mathbf{p}$ is simply the set $N_k(\mathbf{p})$ of the k nearest neighbours of $\mathbf{p} \Longrightarrow$ need to select a distance measure.

Local $k-$regions have different extent from one observation to the next $\Longrightarrow$ each $\mathbf{p}$ has a **local density**.

Observations with anomalously small local density compared to its $k-$neighbours are identified as **outliers**.



For $k = 2$, $\mathbf{p}$ has lower density than its $k-$neighbours $\mathbf{q}_1, \mathbf{q}_2$. The formal procedure is provided in Algorithm 1.

Algorithm 1: Local Outlier Factor (LOF)

```
1 Input: dataset $D$, point $\mathbf{p} \in D$, integer $k$ for
    number of nearest neighbours to consider,
    distance function $d$
2 Compute the distance between all points in $D$
3 for $\mathbf{p} \in D$ do
4     for $\mathbf{q} \in D \setminus \{\mathbf{p}\}$ do
5         Compute $d(\mathbf{p}, \mathbf{q})$
6     end
7     Order $D$ by increasing distance from $\mathbf{p}$
8     Set $d_k(\mathbf{p}) = d(\mathbf{p}, \mathbf{q}_k)$
9 end
```

```
10 Find the $k$ nearest neighbours of $\mathbf{p}$
11 Set $N_k(\mathbf{p}) = \{\mathbf{q} \in D \setminus \{\mathbf{p}\} : d(\mathbf{p},\mathbf{q}) \leq d_k(\mathbf{p})\}$
12 Define the reachability distance
       $d_{\text{reach}}(\mathbf{p},\mathbf{q}) = \max\{d_k(\mathbf{q}), d(\mathbf{p},\mathbf{q})\}$
13 Define the average reachability distance
       $\overline{d_{\text{reach}}}(\mathbf{p}) = \frac{\sum_{\mathbf{q} \in N_k(\mathbf{p})} d_{\text{reach}}(\mathbf{p},\mathbf{q})}{|N_k(\mathbf{p})|}$
14 Define the local reachability density
       $\ell_k(\mathbf{p}) = \left(\overline{d_{\text{reach}}}(\mathbf{p})\right)^{-1}$
15 Compute the local outlier factor $a_k(\mathbf{p}) = \frac{\sum_{\mathbf{q} \in N_k(\mathbf{p})} \frac{\ell_k(\mathbf{q})}{\ell_k(\mathbf{p})}}{|N_k(\mathbf{p})|}$
16 **Output:** LOF $a_k(\mathbf{p})$
```

Any point with a local outlier factor $a_k(\mathbf{p})$ above some threshold τ is a **local outliers**.

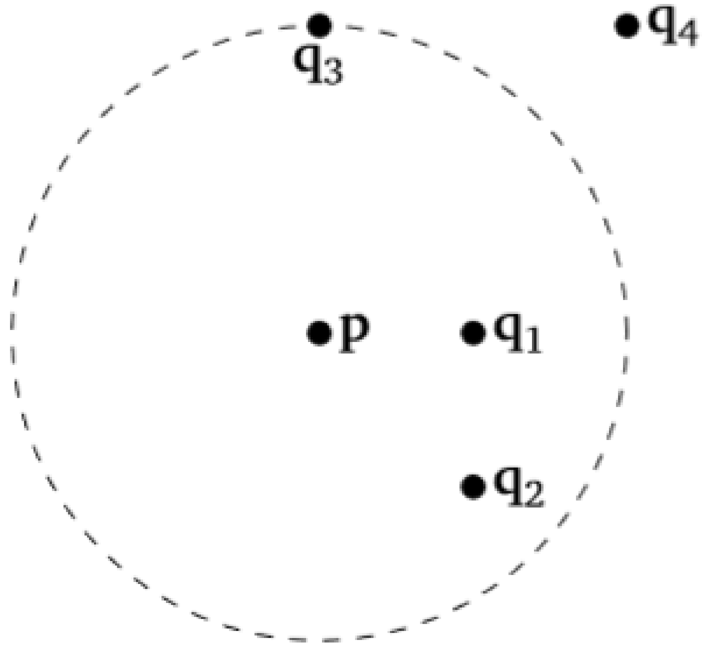
⚠ **Selecting appropriate k and threshold τ is not simple.**

Using a derived **reachability distance** improves the stability of the algorithm results: within $N_k(\mathbf{p})$,

$$d_{\text{reach}}(\mathbf{p},\mathbf{q}) = \max_{\ell}\{d(\mathbf{p},\mathbf{q}_\ell); \mathbf{q}_\ell \in N_k(\mathbf{p})\};$$

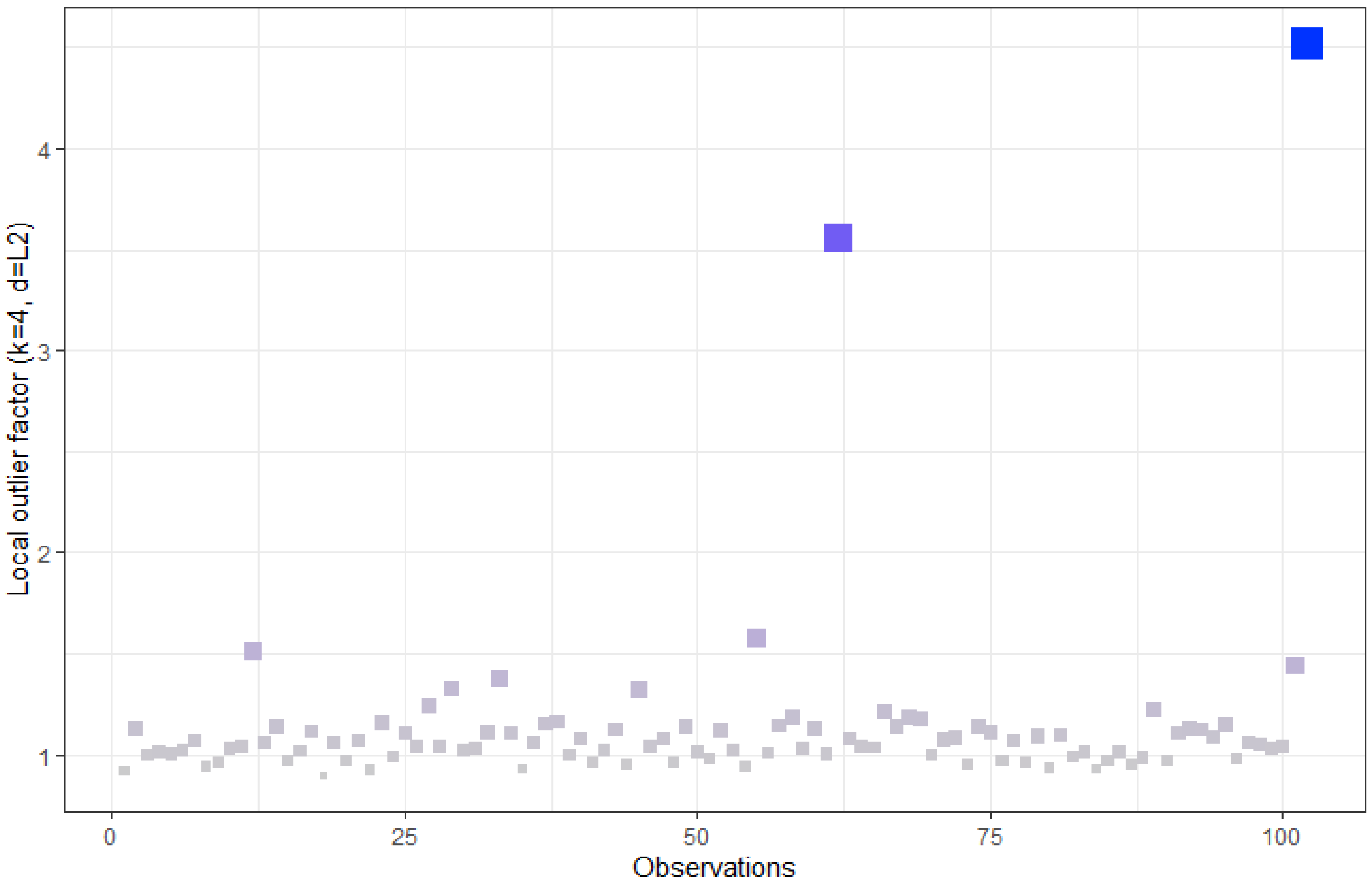
outside of $N_k(\mathbf{p})$,

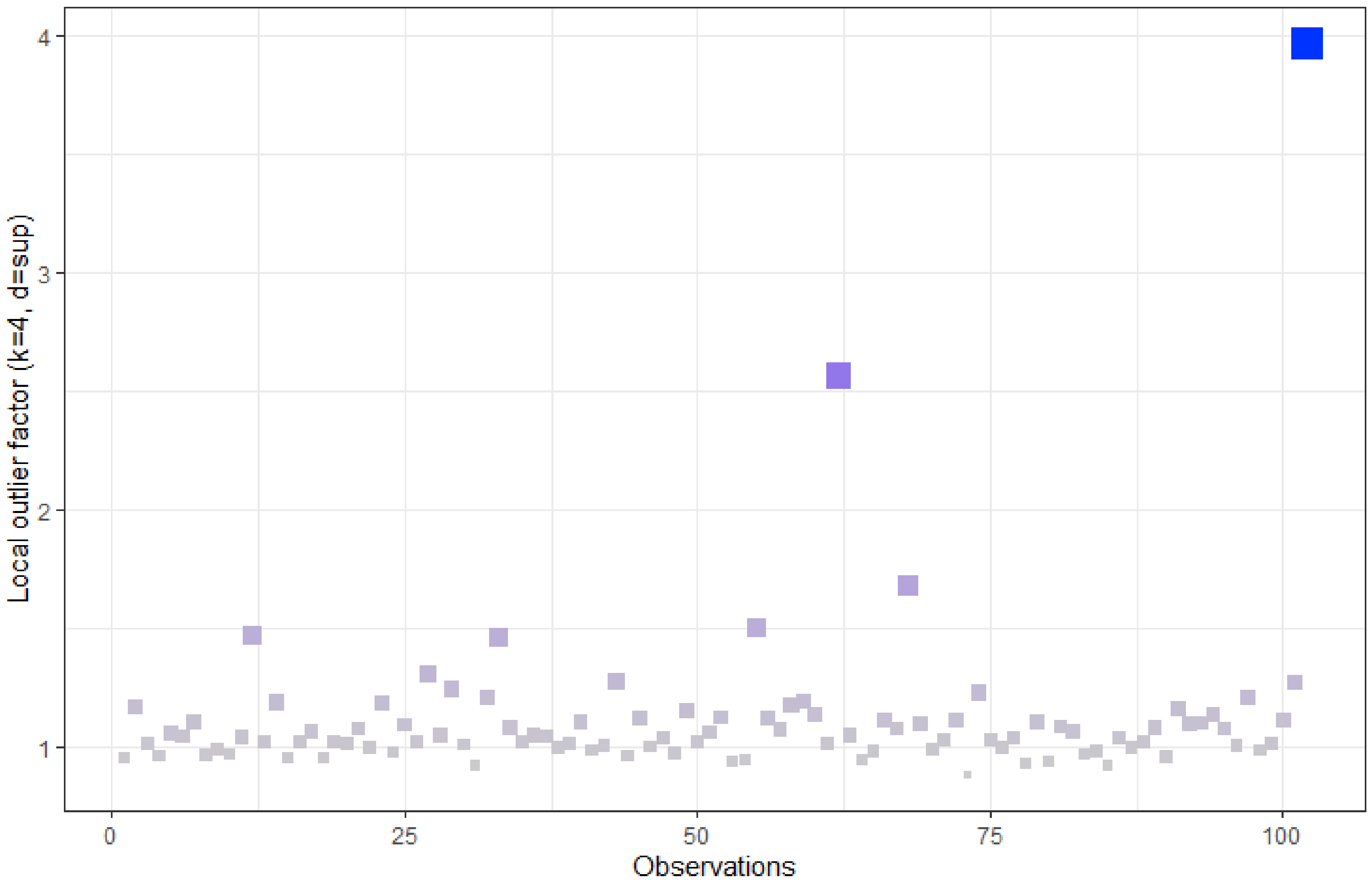
$$d_{\text{reach}}(\mathbf{p},\mathbf{q}) = d(\mathbf{p},\mathbf{q}).$$



The region of uniform reachability distance around $\mathbf{p}$ for $k = 3$:

$d_{\text{reach}}(\mathbf{p}, \mathbf{q}_1) = d_{\text{reach}}(\mathbf{p}, \mathbf{q}_2) = d_{\text{reach}}(\mathbf{p}, \mathbf{q}_3) = d(\mathbf{p}, \mathbf{q}_3)$; $d_{\text{reach}}(\mathbf{p}, \mathbf{q}_4) = d(\mathbf{p}, \mathbf{q}_4)$.


Local outlier factor (k=4, d=L2)
4
3
2
1
0
25
50
75
100
Observations


Local outlier factor (k=4, d=sup)
4
3
2
1
0
25
50
75
100
Observations

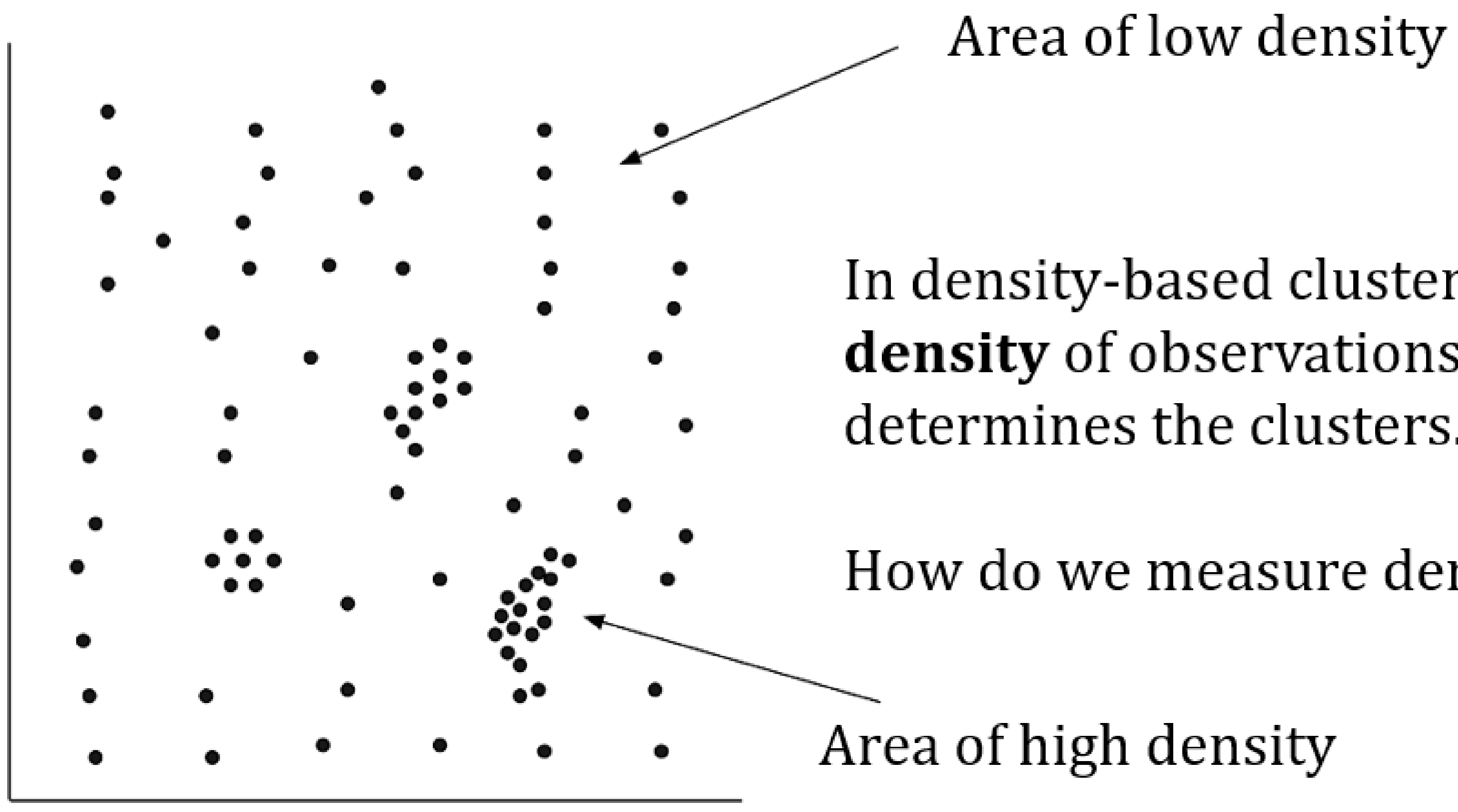
DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a density-based clustering algorithm that groups nearby points together.

Points that do not fall in the clusters are labeled as (potential) **anomalies**.

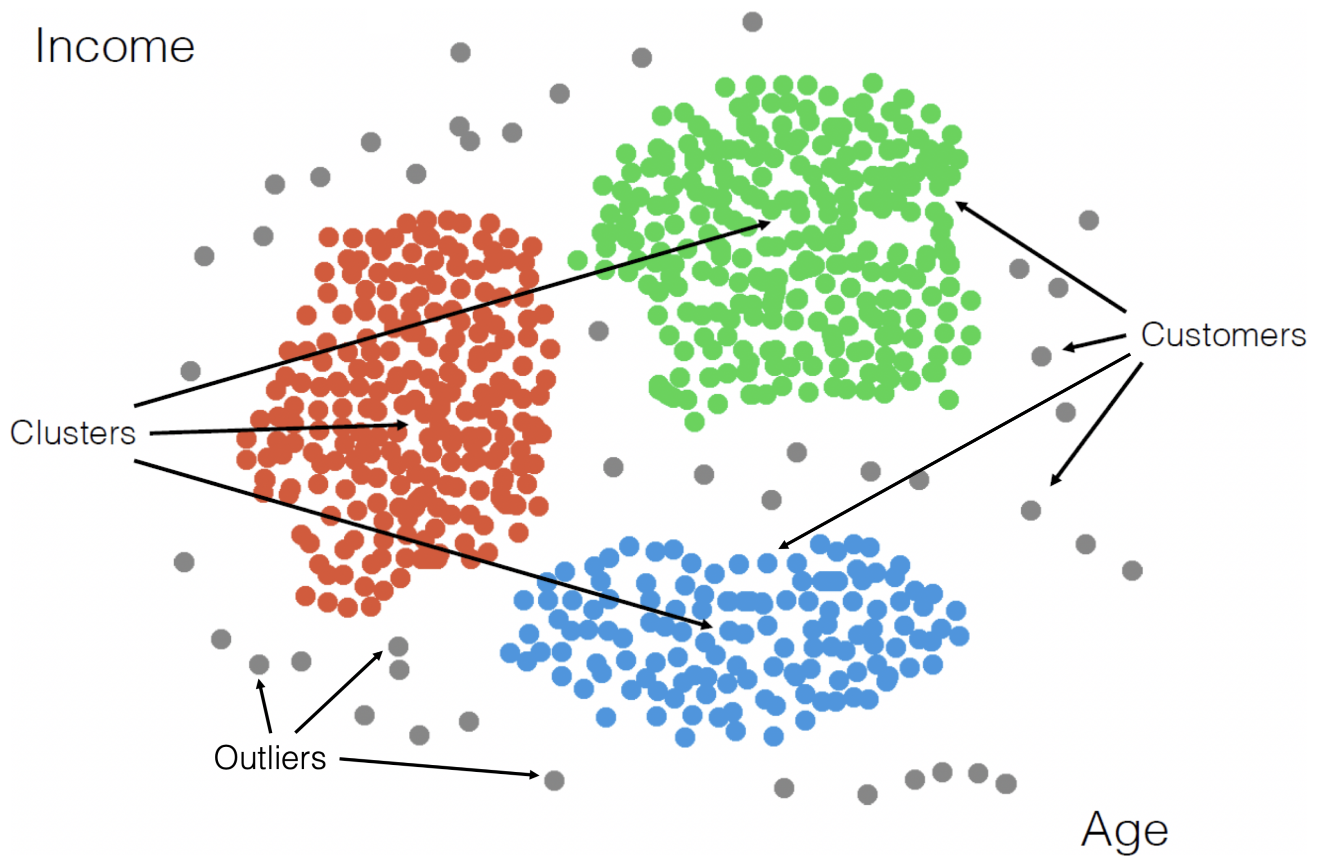
Hierarchical DBSCAN (HDBSCAN), which removes the problem of choosing one of DBSCAN's parameters (the radius of neighbourhoods).

⚠ We won't be talking about **clustering** in a general sense – it could form the basis of $2+$ courses – but it would be a good idea to take some time to read up on the topic if you are not familiar with it.



In density-based clustering, the **density** of observations determines the clusters.

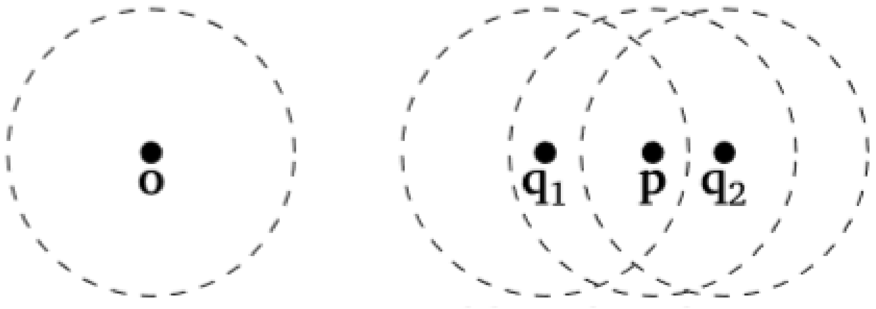
How do we measure density?



Clusters of regular customers (red, green, blue) and potential anomalies/outliers (grey) in an artificial dataset.

In DBSCAN,

- $\mathbf{p}$ is a **core point** if there are $m+$ points within distance r of $\mathbf{p}$;
- $\mathbf{q}$ (non-core) is a **border point** if it is within distance r of a core point;
- $\mathbf{o}$ is an **outlier** if it is neither a core nor a border point.



For minimum neighbourhood size $m = 2$ and the fixed radius r above, $\mathbf{o}$ is an outlier, $\mathbf{p}$ is a core point, and $\mathbf{q}_1, \mathbf{q}_2$ are border points.

DBSCAN considers each point in the dataset individually:

- if a point is not a core point or a border point, it is considered an outlier;
- if it is a border point, it is not considered an outlier, but it does not form the basis of a new cluster of regular observations;
- if it is a core point, then its r-neighbourhood forms the beginning of a new cluster;
- each point in this r-neighbourhood is then considered in turn, with the r-neighbourhoods of other core points contained in the neighbourhood being added to the cluster (regular observations).

This expansion repeats until all points have been examined.

During the expansion, points that were previously labelled as outliers may be updated as they become border points in a new cluster.

This process continues until every point has either been assigned to a cluster or labelled as an outlier.

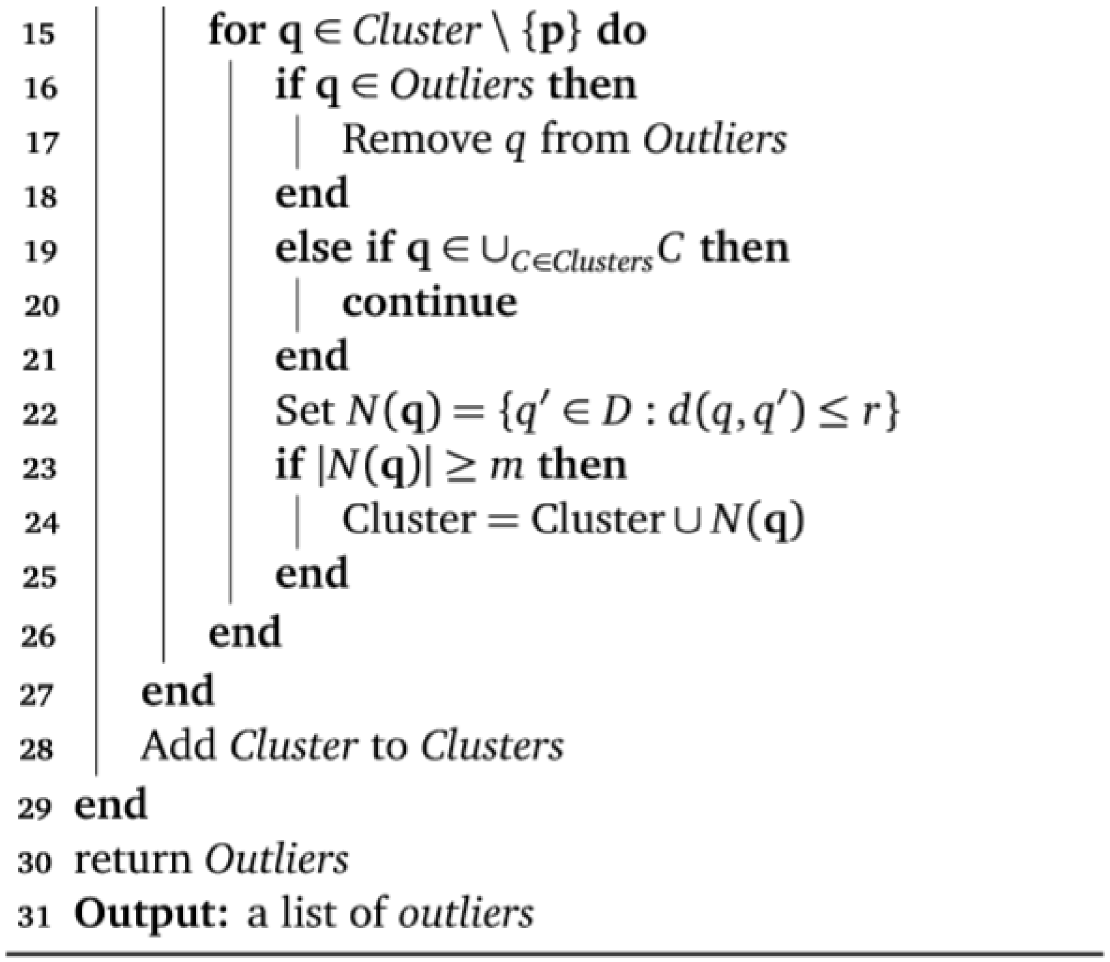
DBSCAN's dual use as a clustering algorithm may seem irrelevant in the outlier detection setting, but its ability to succesfully identify clusters is crucial (the remaining points are outliers).

The formal procedure is provided in Algorithm 2.

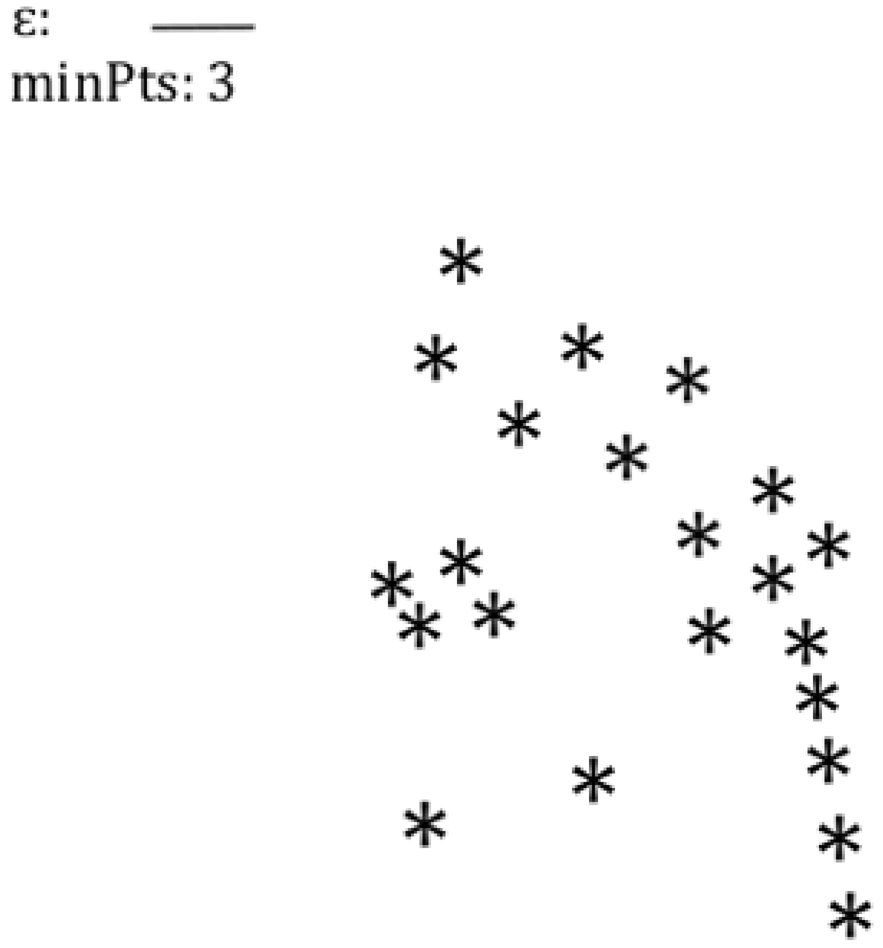
(But why use DBSCAN instead of any other of the $50+$ cluster algorithms?)

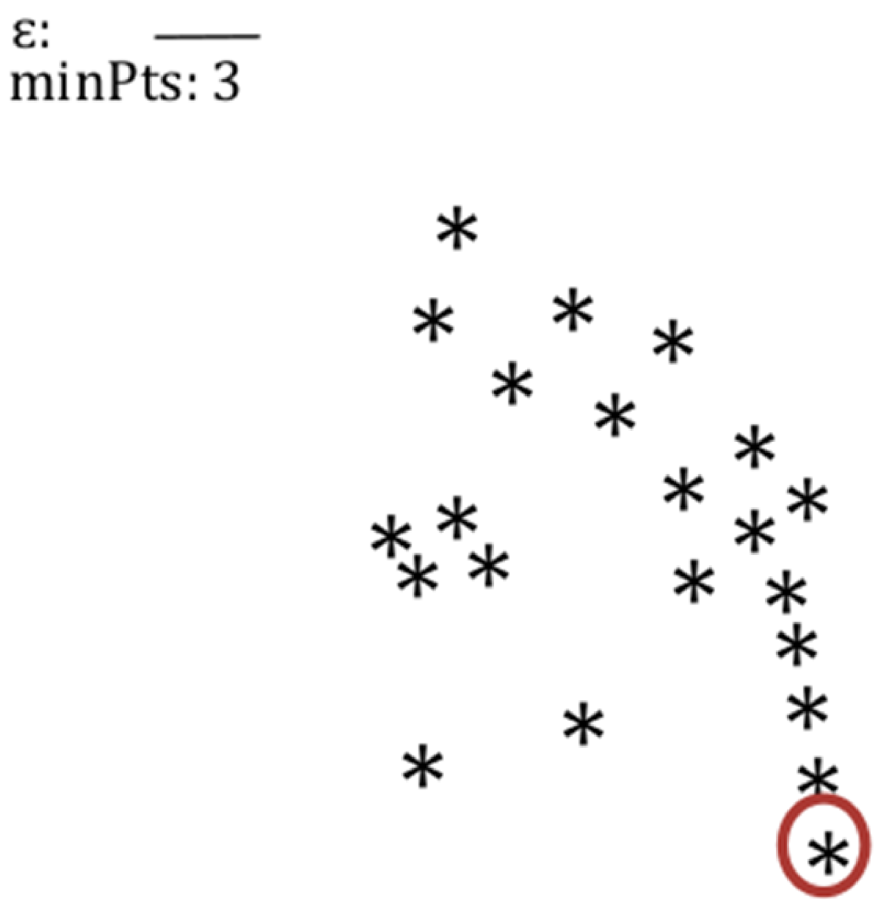
Algorithm 2: DBSCAN

```
1  Input: dataset $D$, distance function $d$,
     neighbourhood radius $r > 0$, minimum number of
     points to be considered a cluster $m \in \mathbb{N}$
2  $Clusters = \{\}$
3  $Outliers = \{\}$
4  for $\mathbf{p} \in D$ do
5      if $\mathbf{p} \in Outliers \cup (\cup_{C \in Clusters} C)$ then
6          continue
7      end
8      Set $N(\mathbf{p}) = \{\mathbf{q} \in D : d(\mathbf{p}, \mathbf{q}) \leq r\}$
9      if $|N(\mathbf{p})| < m$ then
10         Add $\mathbf{p}$ to $Outliers$
11         continue
12     end
13     else
14         $Cluster = N(\mathbf{p})$
```

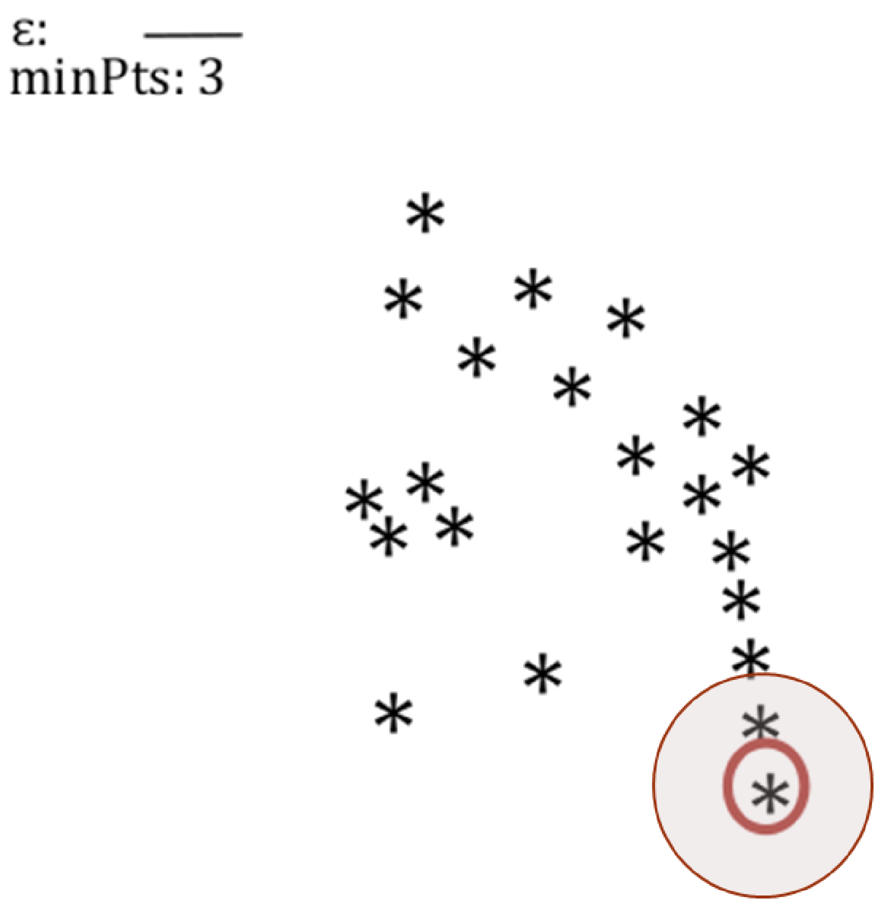


```
        for $\mathbf{q} \in Cluster \setminus \{\mathbf{p}\}$ do
            if $\mathbf{q} \in Outliers$ then
                Remove $q$ from $Outliers$
            end
            else if $\mathbf{q} \in \cup_{C \in Clusters} C$ then
                continue
            end
            Set $N(\mathbf{q}) = \{q' \in D : d(q, q') \leq r\}$
            if $|N(\mathbf{q})| \geq m$ then
                Cluster $=$ Cluster $\cup N(\mathbf{q})$
            end
        end
    end
    Add $Cluster$ to $Clusters$
end
return $Outliers$
Output: a list of $outliers$
```

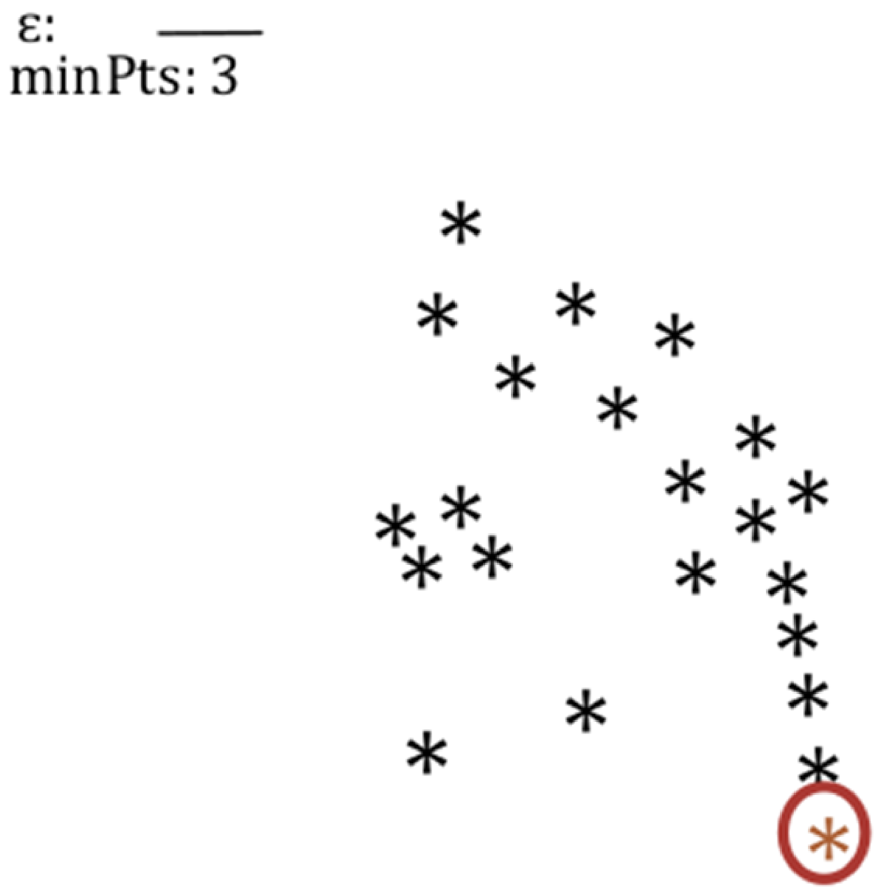

ε:
minPts: 3



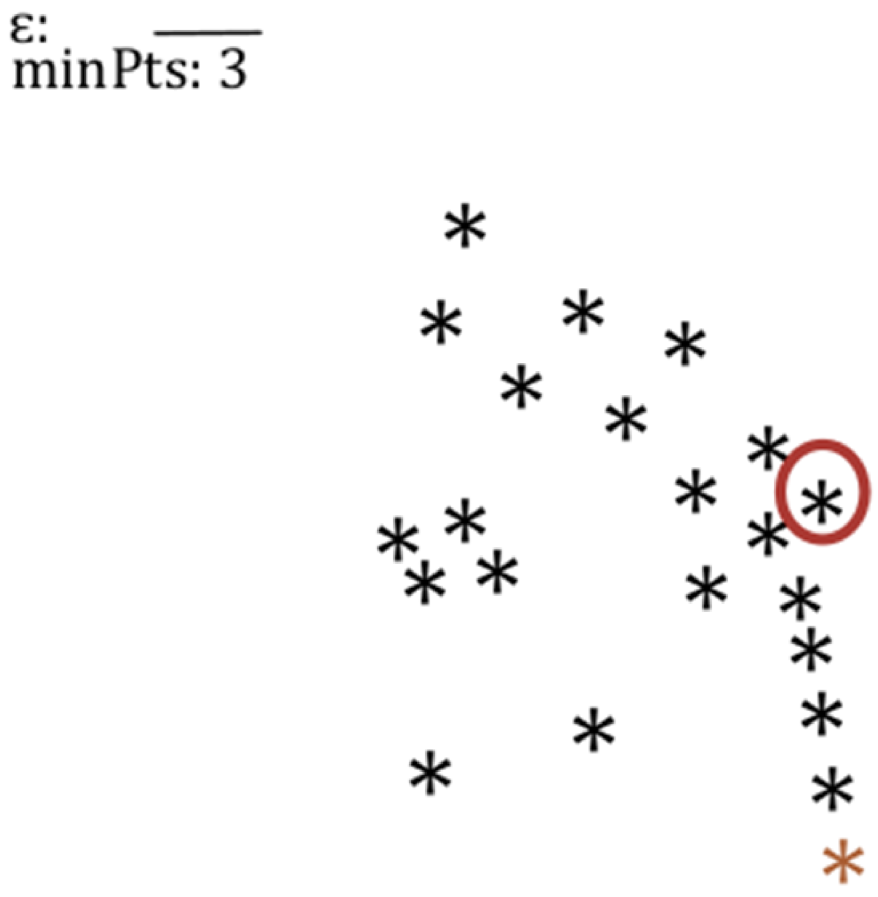
Point picked at random



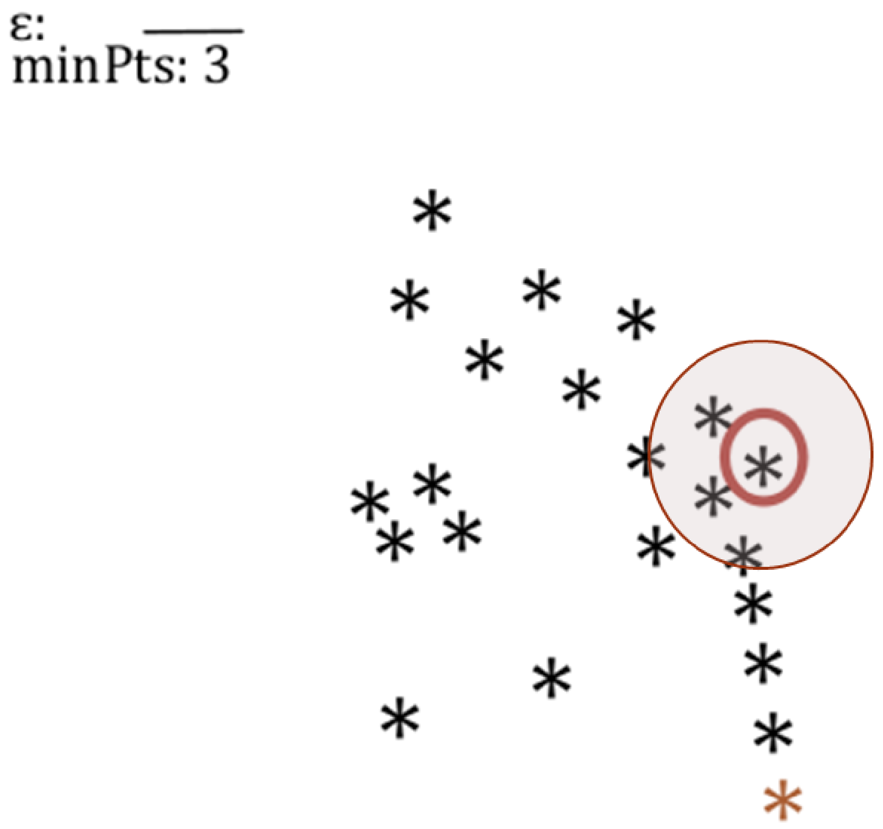
Point identified as non-core point



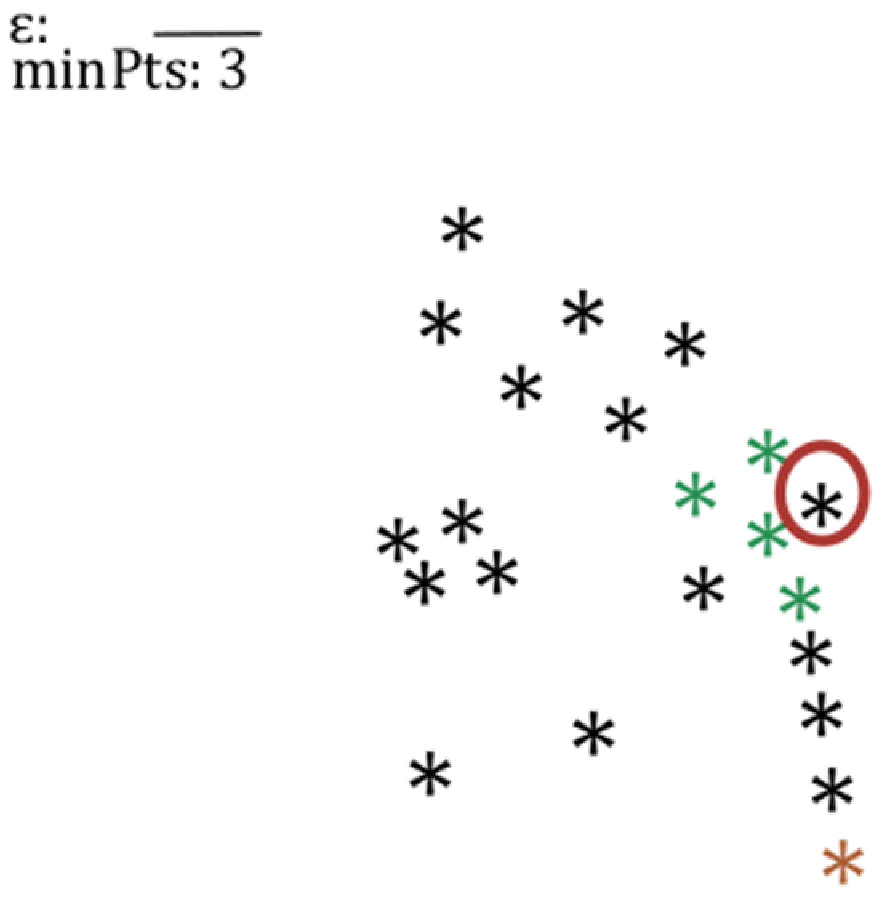
Point identified as non-core point



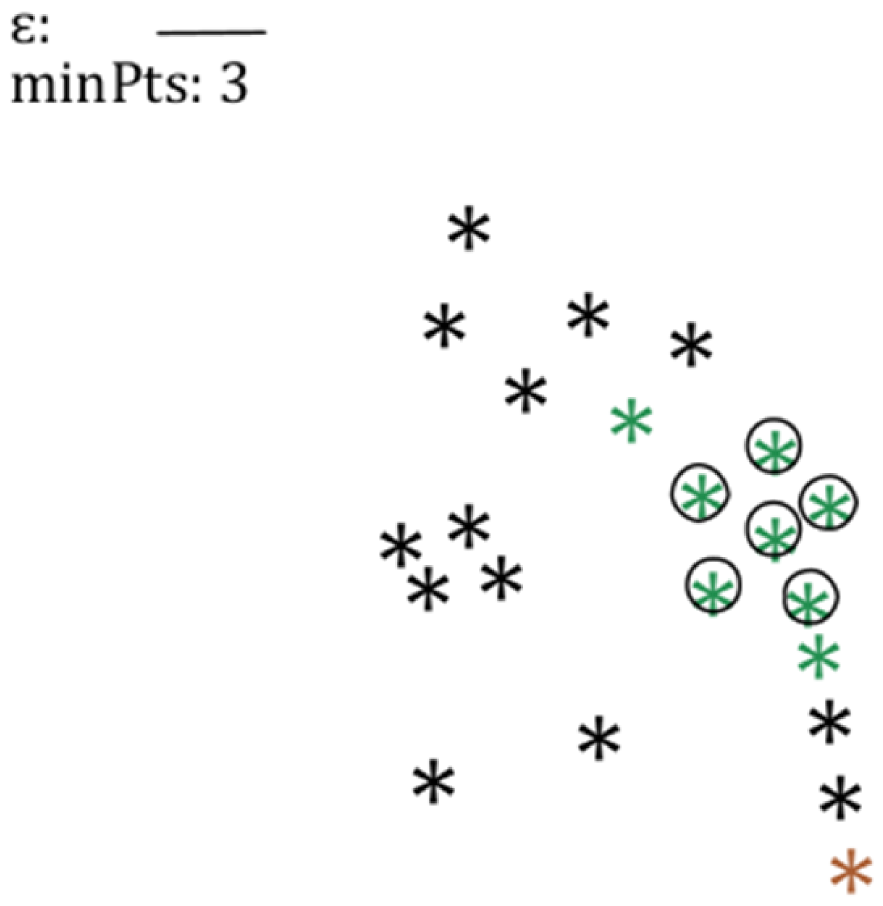
Another point picked at random



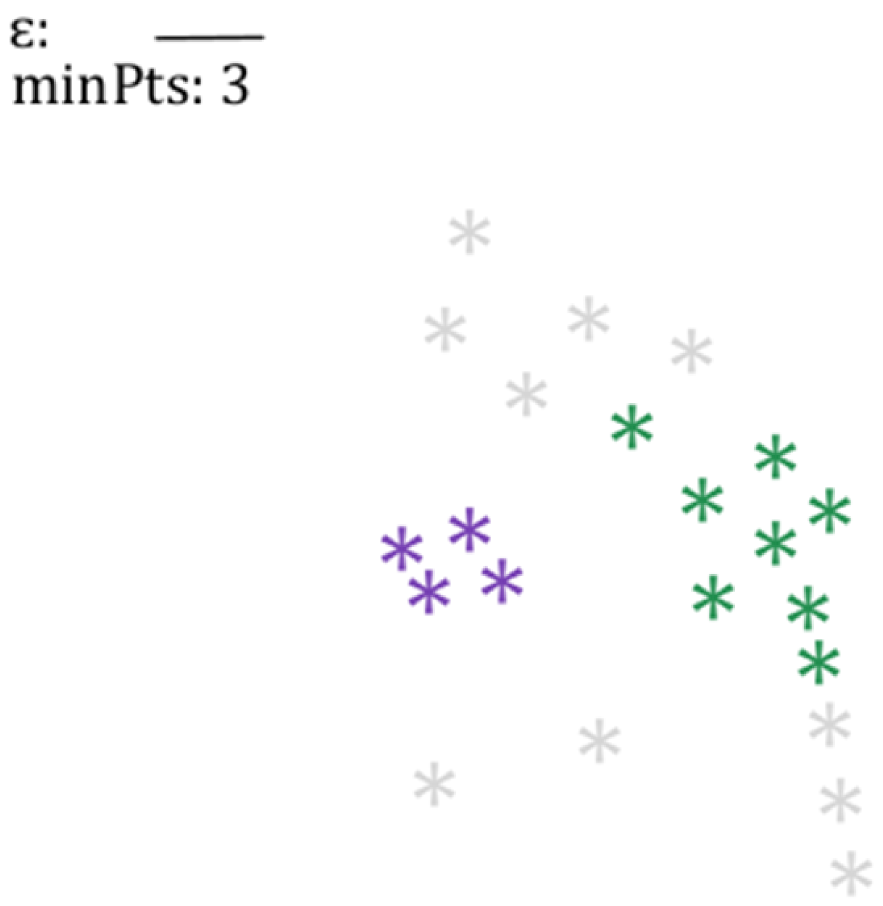
Point identified as a core point



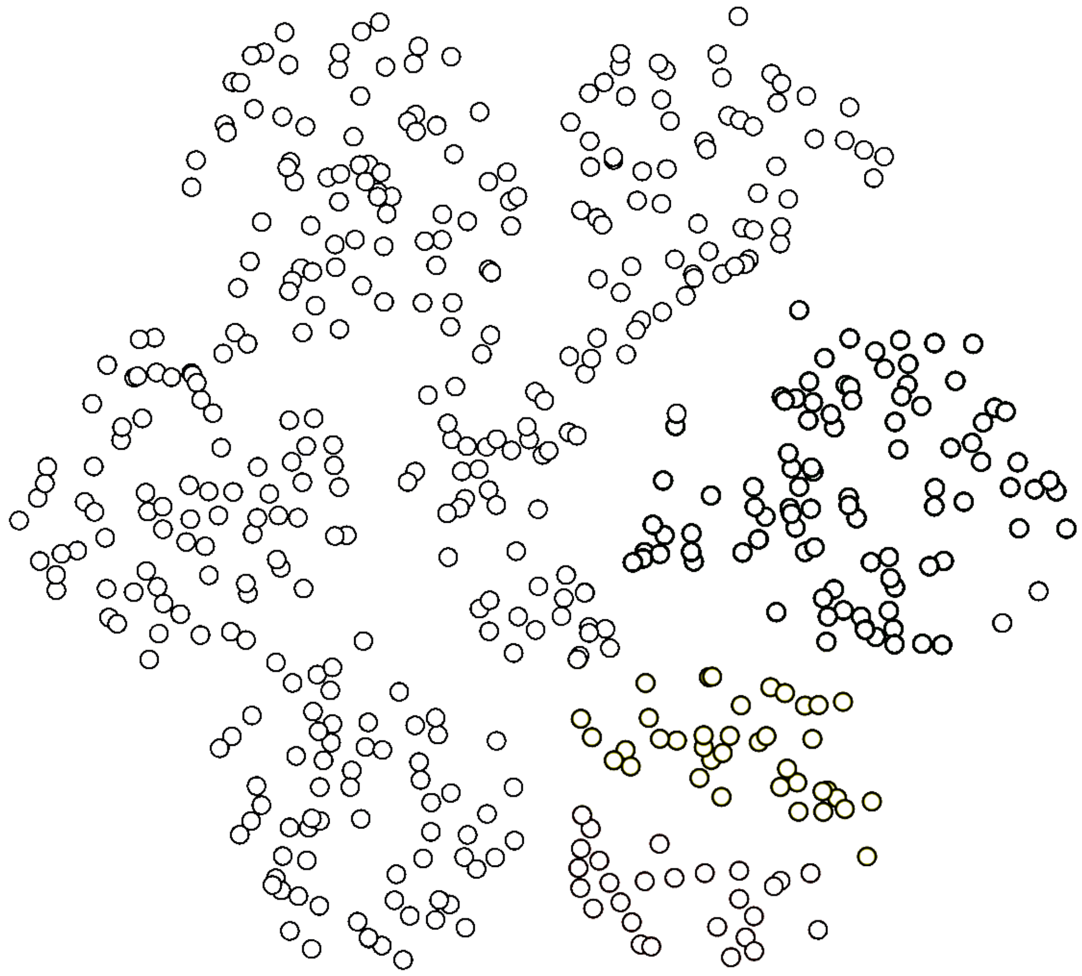
Points in the $\varepsilon-$neighbourhood

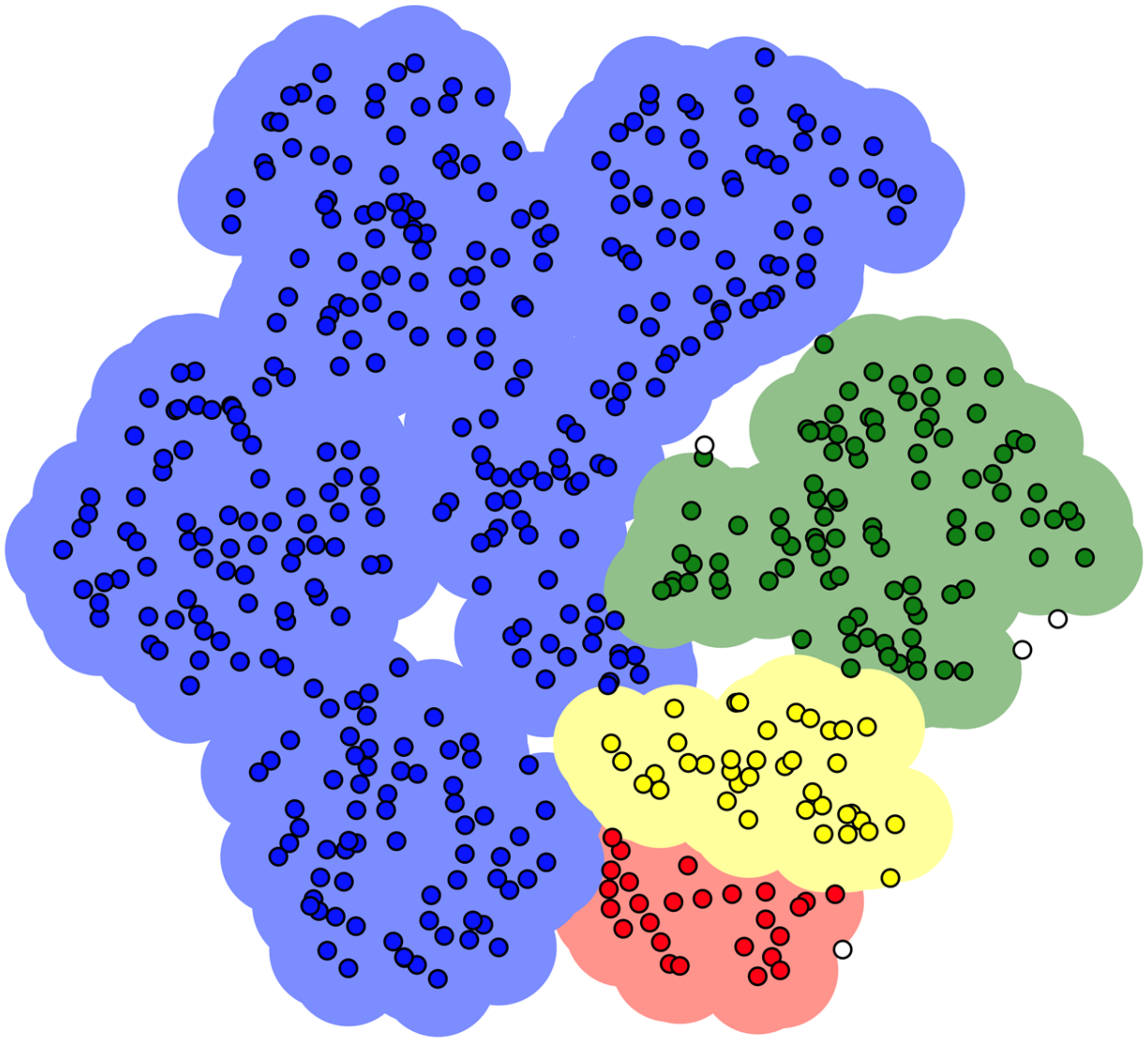


Resulting cluster



Clusters and outliers





Strengths:

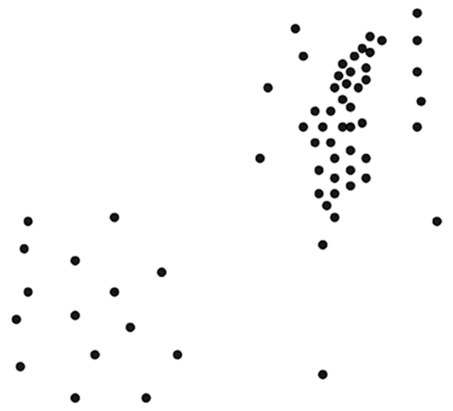
- the number of clusters does not need to be known beforehand;
- clusters of arbitrary shape can be detected;
- with HDBSCAN, only the parameter for the minimum cluster size $m \geq n+1$ is required (larger values of m allow for better noise identification).

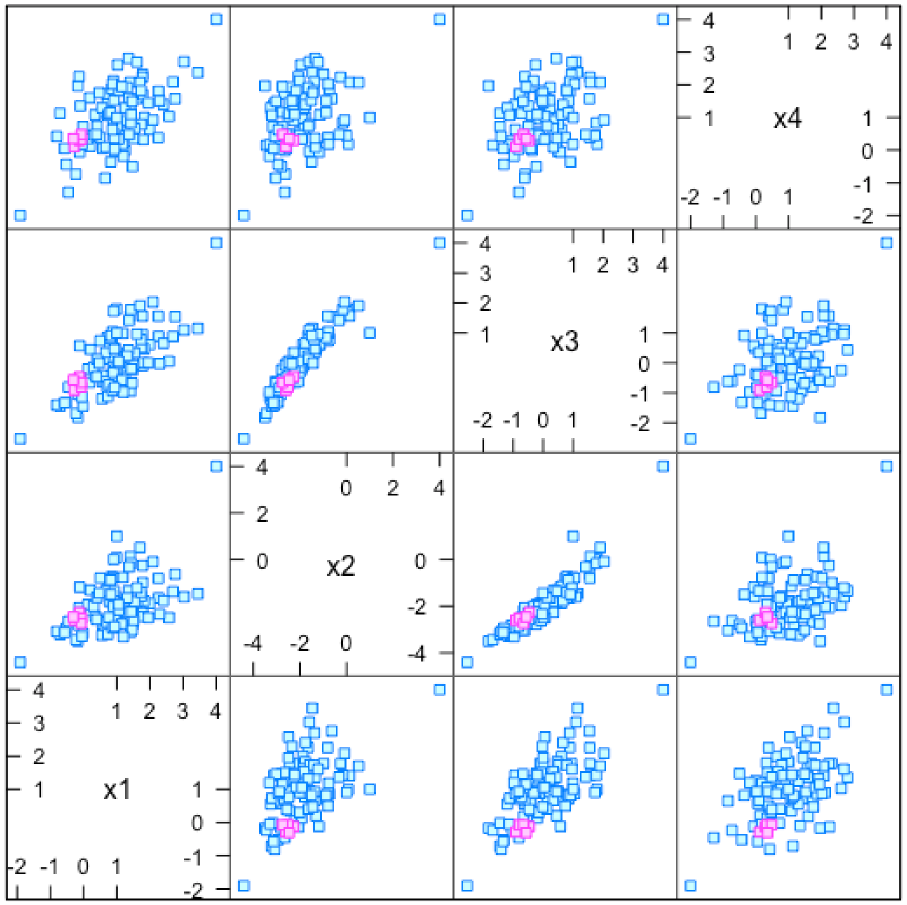
Limitations:

- not entirely deterministic, as border points can be assigned to different clusters depending on the order in which core points are considered (does not affect its use as an anomaly detection algorithm).

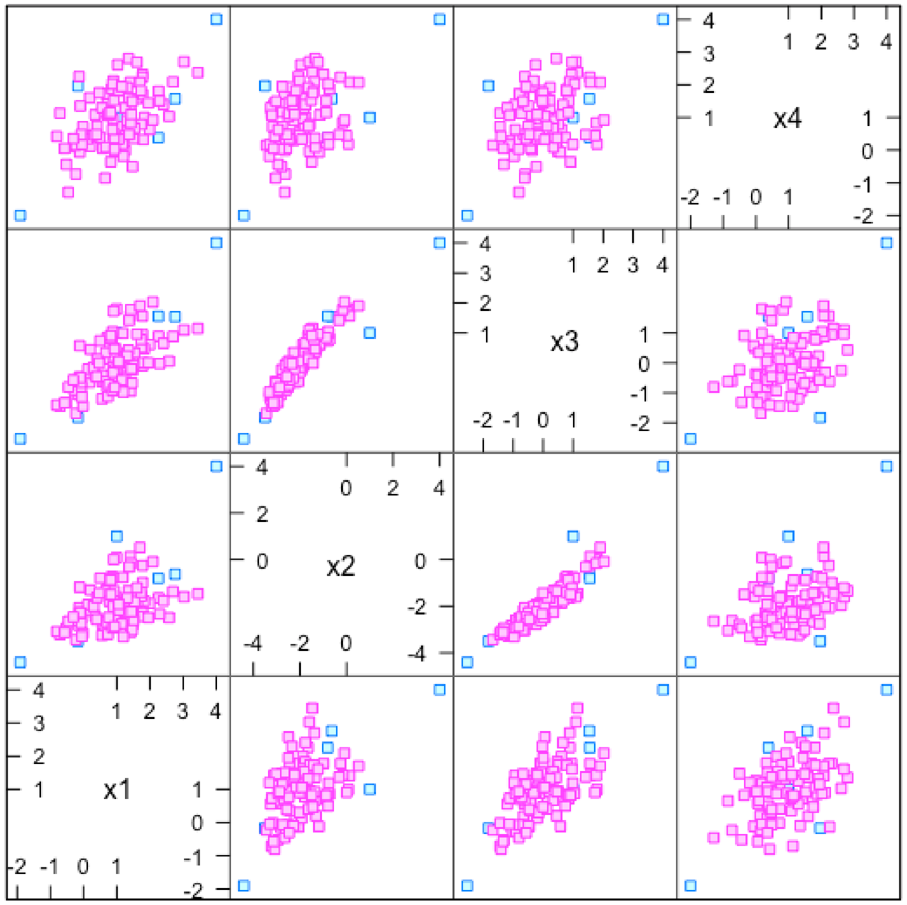
- suffers from the **Curse of Dimensionality** – in high-dimensional spaces, Euclidean-based distances have a difficult time distinguish **near** observations from **distant** ones;

- cannot handle differences in local densities when the radius r of a neighbourhood is fixed $\Longrightarrow$ sparser clusters could be labelled as outliers, or outliers surrounding a denser cluster could be included in the cluster.

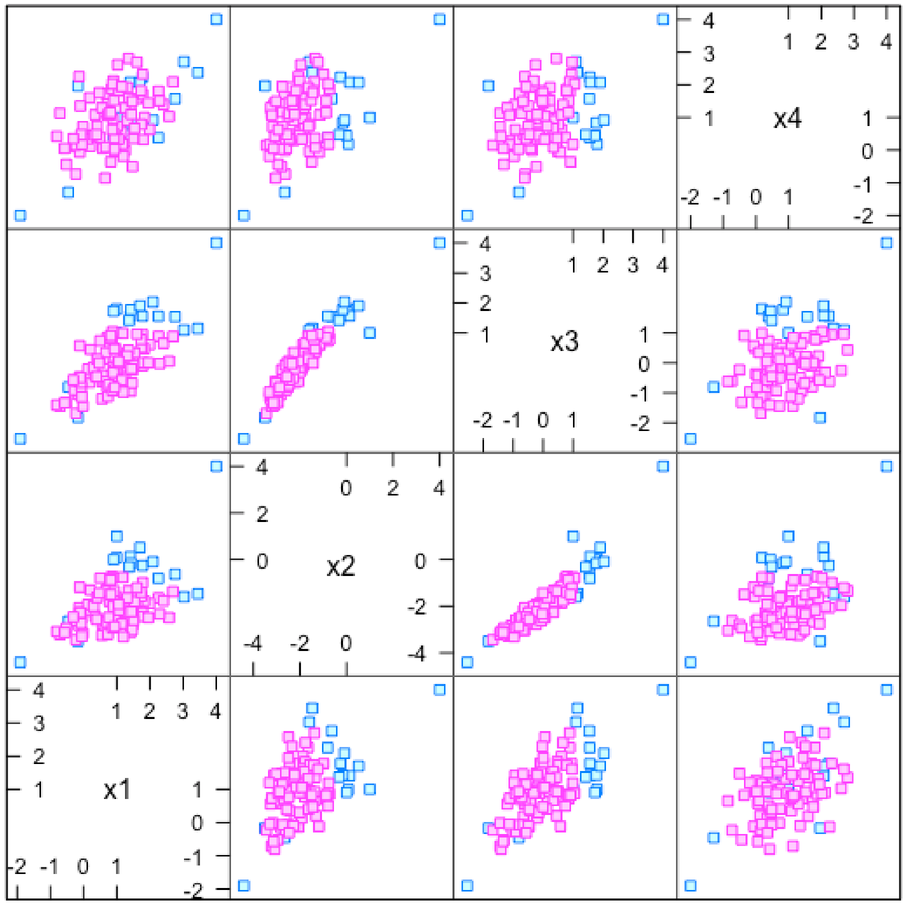




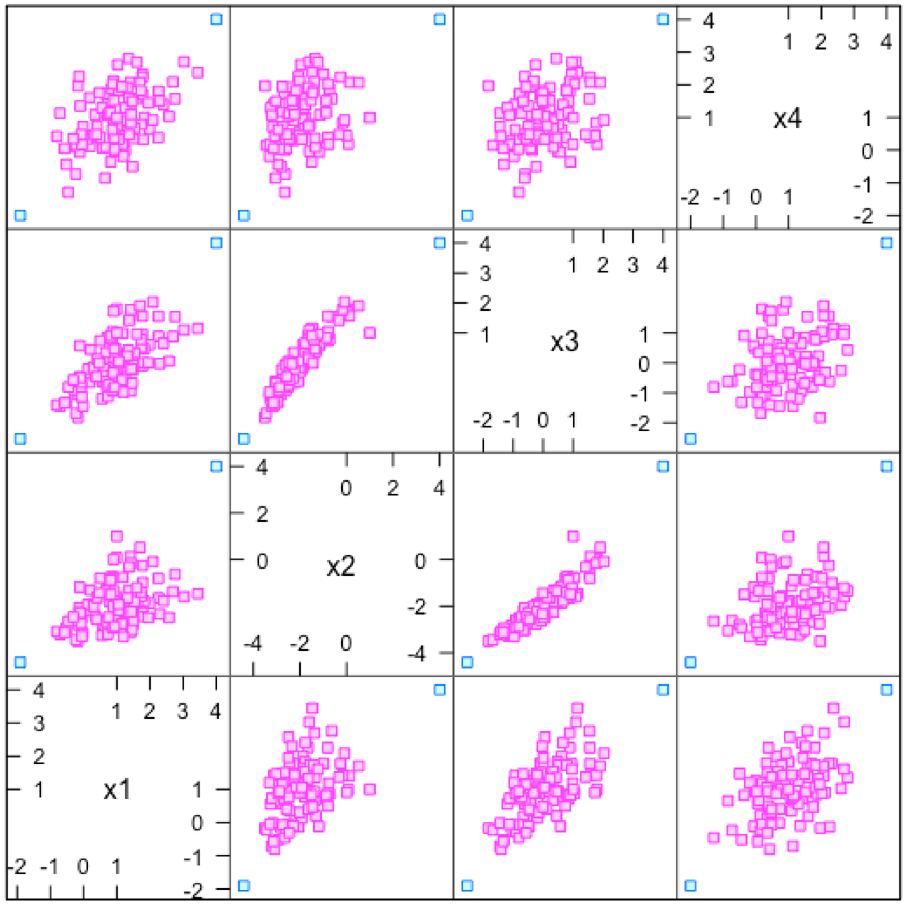
DBSCAN, d_2, unscaled data, $\varepsilon = 0.4$, minPts $= 4$



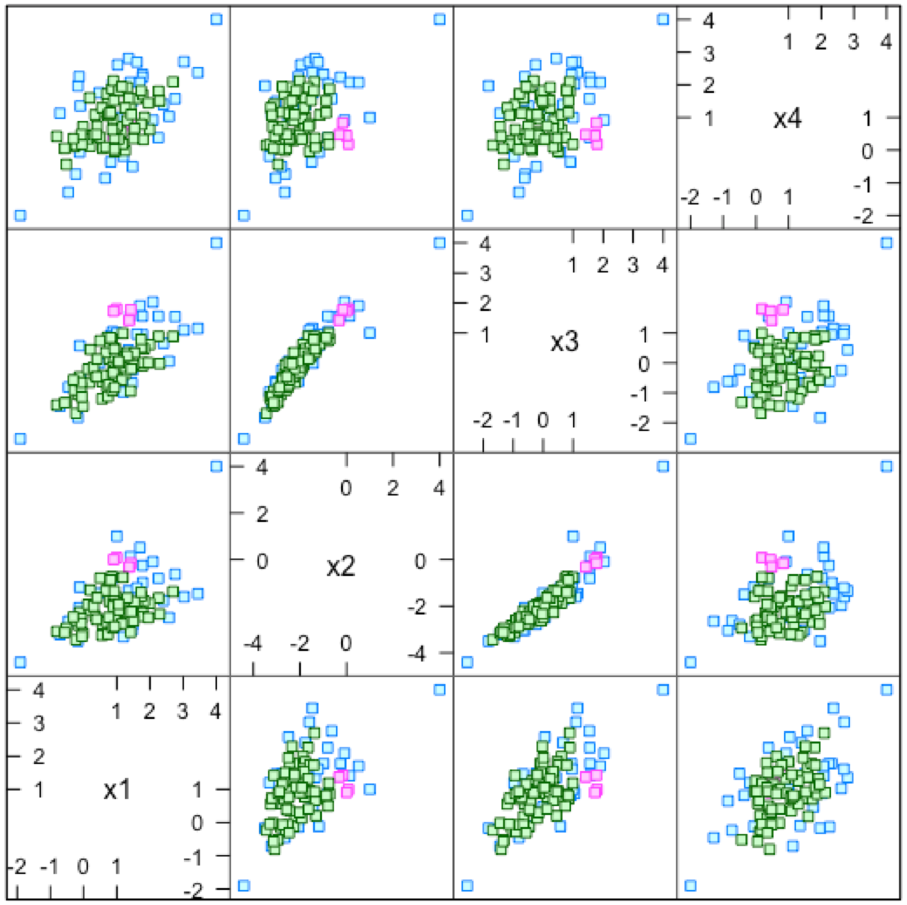
DBSCAN, d_2, unscaled data, $\varepsilon = 1$, minPts $= 4$



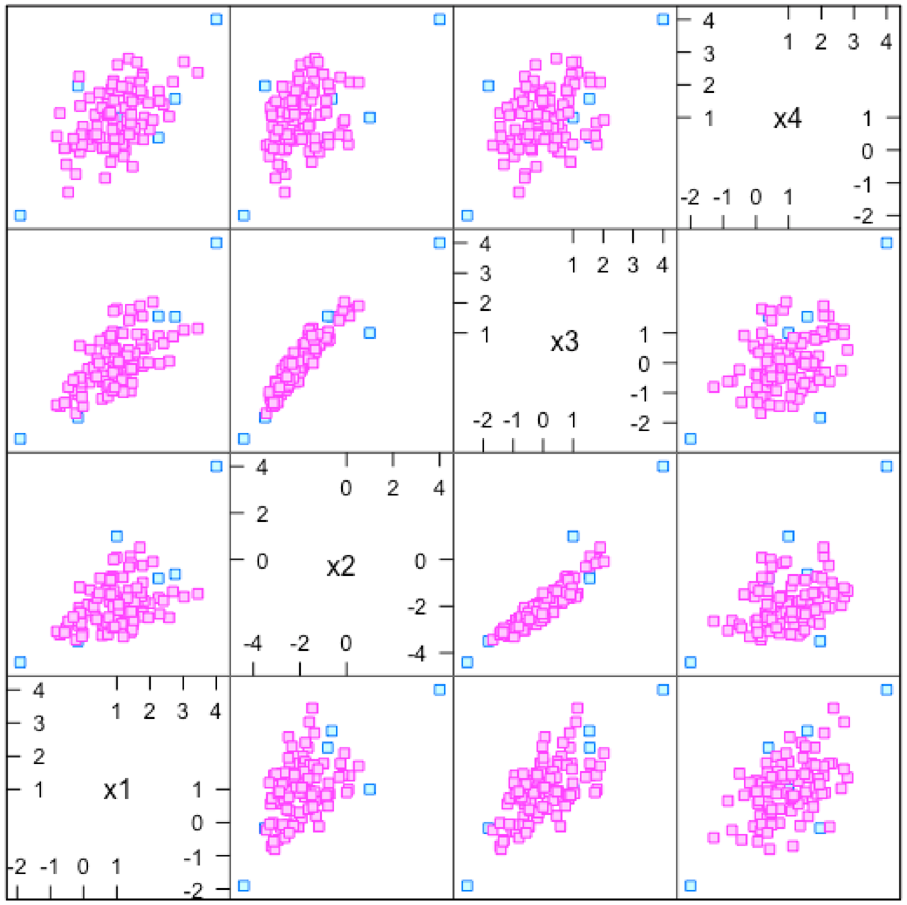
DBSCAN, d_2, unscaled data, $\varepsilon = 1$, minPts $= 8$



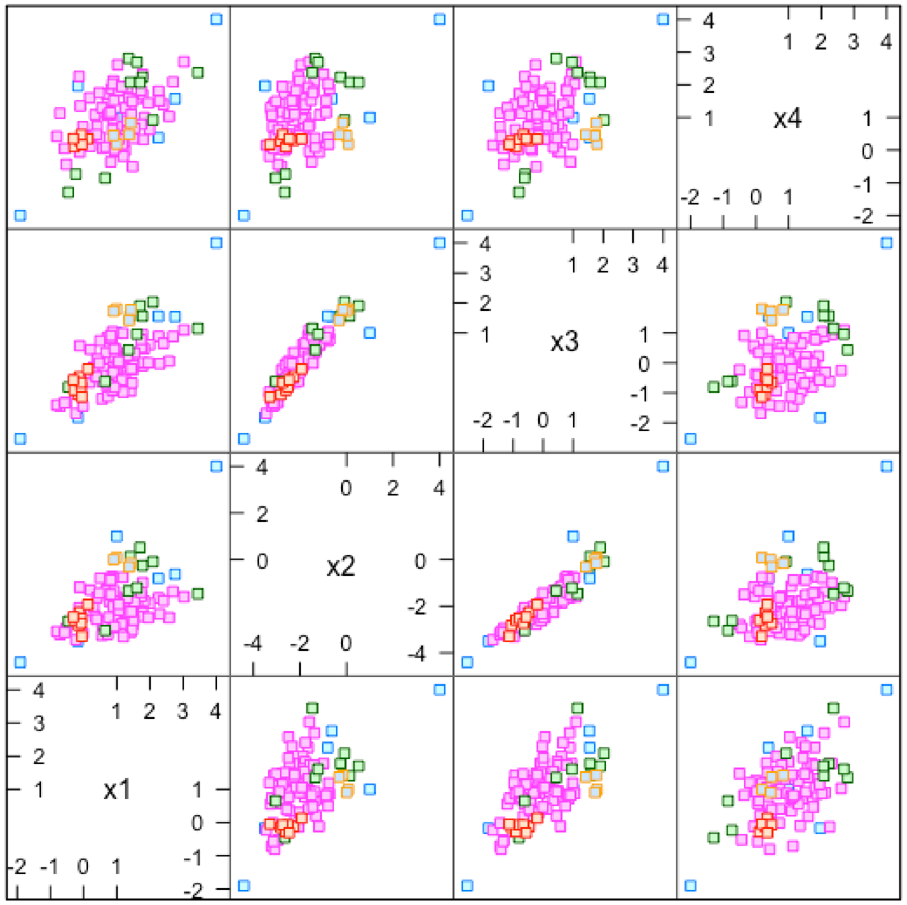
DBSCAN, d_2, unscaled data, $\varepsilon = 2$, minPts $= 8$



HDBSCAN, d_2, unscaled data, minPts $= 4$



OPTICS, d_2, unscaled data, $\varepsilon = 1$, minPts $= 4$, $\varepsilon_{\text{cl}} = 1$



OPTICS, d_2, unscaled data, $\varepsilon = 1$, minPts $= 4$, $\varepsilon_{\text{cl}} = 1$, $\xi = 0.05$

Isolation Forest

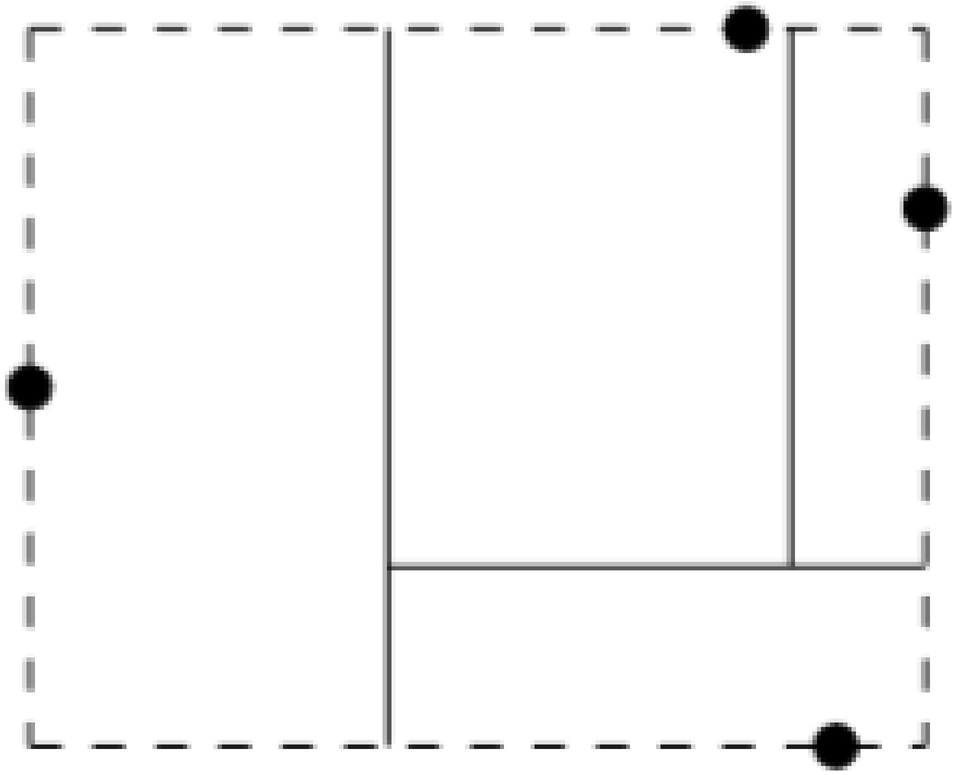
Both the LOF and the DBSCAN approach first construct models of what normal points look like, and then identify points that do not fit this model.

The **isolation forest** (IsoForest) algorithm tries instead to explicitly identify outliers under the assumptions that:

- there are **few** outliers, and
- that these outliers have **very different attributes** compared to normal observations.

IsoForest uses sampling techniques that increase algorithmic speed while decreasing memory requirements.

IsoForest tries to **isolate** anomalous points by **randomly** selecting an attribute and a split value between that attribute's min/max values, continuing until every point is alone in its component.



This recursive partitioning yields an **Isolation Tree** (IsoTree):

- the **root** of this tree is the entire dataset;
- each **node** is a subset of the observations;
- each **branch** corresponds to one of the generated partitions, and
- the **leaves** are sets containing a single isolated point.

Each point is then assigned a score derived from **how deep in the tree** its singleton partition appears.

Points that are **shallower** in the tree are easier to separate from the rest $\Longrightarrow$ likely **outliers**?

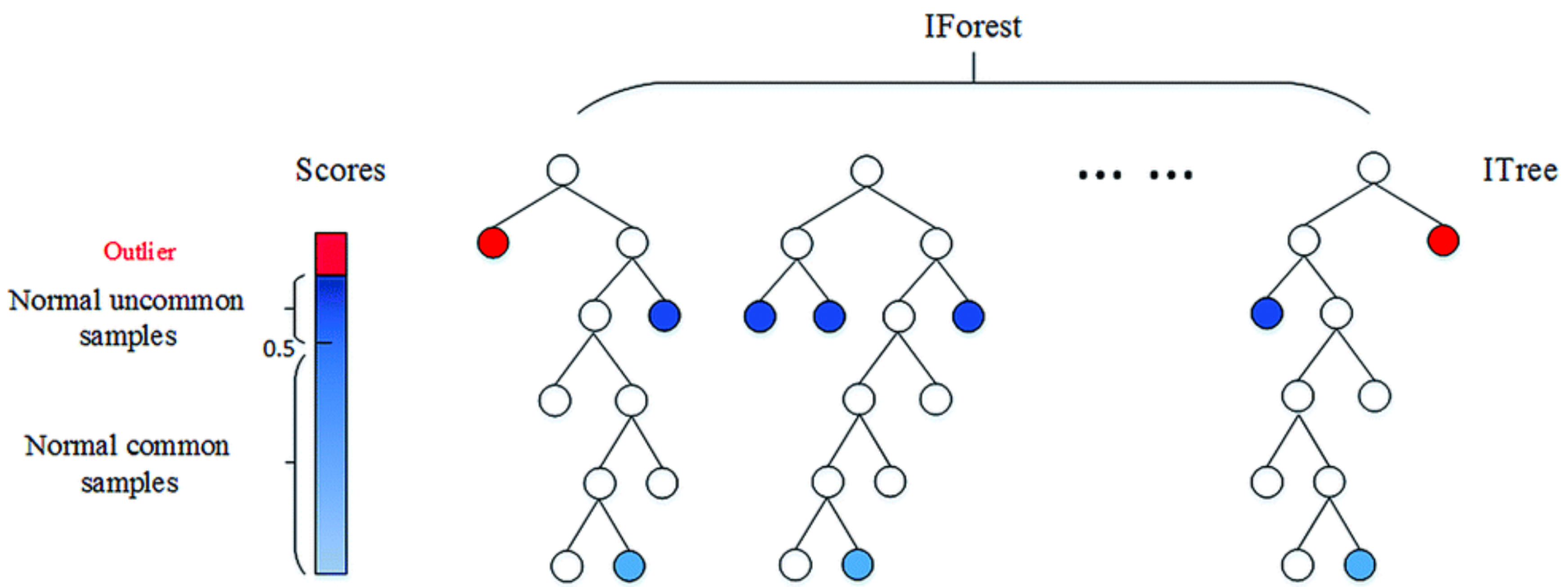
The points are of interest are shallow: once the height of the tree has reached a **given threshold** (expected height of a random binary tree?), stop growing the tree (reduces computational cost).

IsoTrees can be constructed from subsets: the location of any point within this smaller tree can be estimated (reduces computational cost).

A collection of IsoTrees forms an **Isolation Forest**.

An IsoForest **score** can be computed for each point: search each tree for the point's location and record the path length required to reach it. The score is simply the average path length (it can be **normalized** to make it independent of the dataset's size); low scores $\Longrightarrow$ outliers.

The formal procedure is provided in Algorithms 3 and 4.



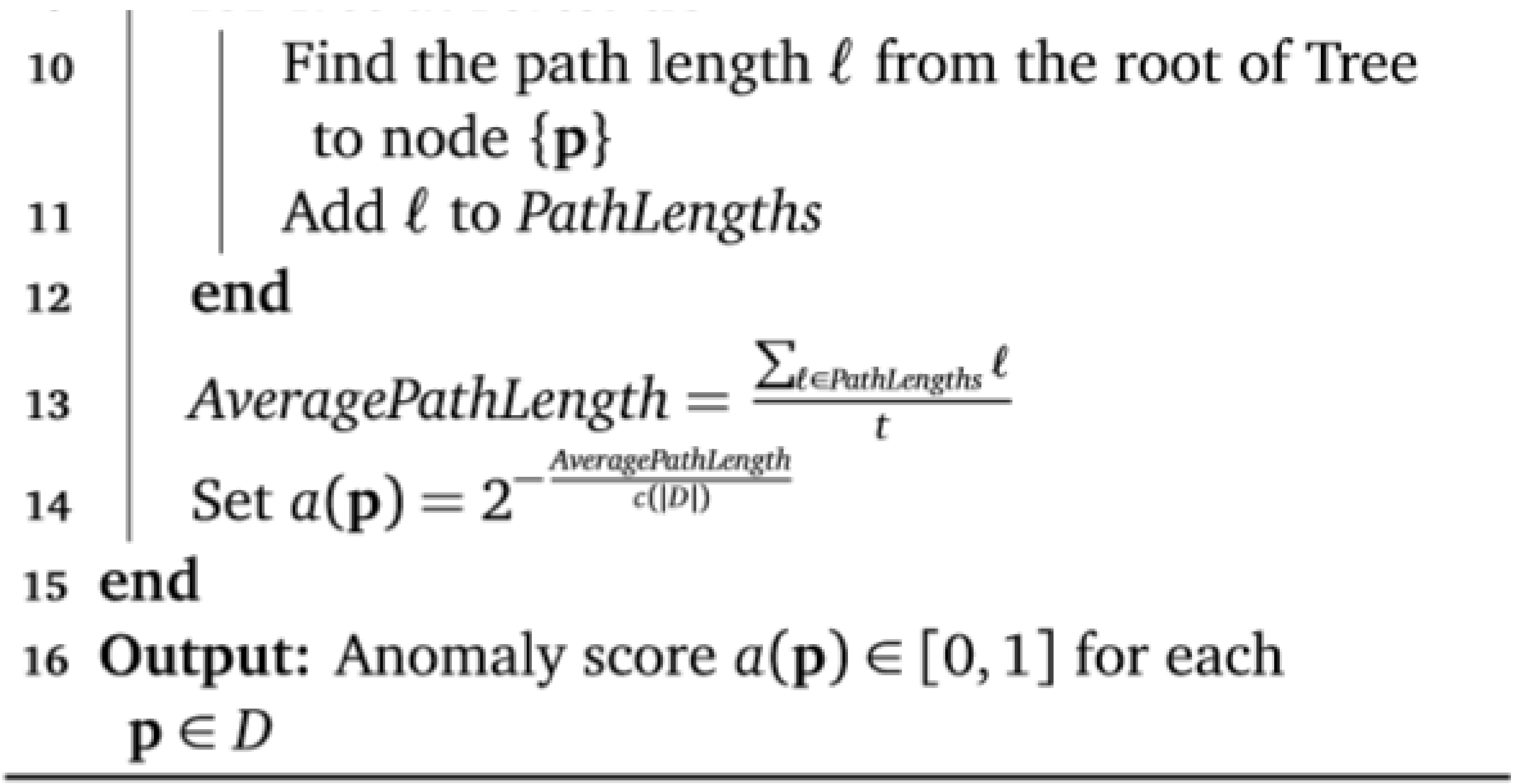
Isolation Forest schematics [Baron].

Algorithm 3: Recursive Isolation Tree Construction: iTree(D)

```
Input: dataset $D$
if $|D| \leq 1$ then
    return {}
end
else
    Let $\overline{A}$ be a list of attributes in $D$
    Randomly select an attribute $A \in \overline{A}$
    Randomly sample a point $s$ from
      $[\min_{\mathbf{q} \in D} A(\mathbf{q}), \max_{\mathbf{q} \in D} A(\mathbf{q})]$
    Return
      $\text{Node} \begin{cases} \text{LeftChild} & = \text{iTree}(\{\mathbf{q} \in D : A(\mathbf{q}) \leq s\}) \\ \text{RightChild} & = \text{iTree}(\{\mathbf{q} \in D : A(\mathbf{q}) > s\}) \\ \text{NodeValue} & = D \end{cases}$
end
Output: Binary tree with node values that are
  subsets of $D$
```

Algorithm 4: Isolation Forest

```
Input: dataset $D$, integer $t$ number of Isolation
  Trees
$Forest = \{\}$
for $i = 1$ to $t$ do
    $Tree = \text{iTree}(D)$
    Add $Tree$ to $Forest$
end
for $\mathbf{p} \in D$ do
    $PathLengths = \{\}$
    for $Tree$ in $Forest$ do
```



```
10        Find the path length $\ell$ from the root of Tree
            to node $\{\mathbf{p}\}$
11        Add $\ell$ to $PathLengths$
12      end
13      $AveragePathLength = \frac{\sum_{\ell \in PathLengths} \ell}{t}$
14      Set $a(\mathbf{p}) = 2^{-\frac{AveragePathLength}{c(|D|)}}$
15  end
16  Output: Anomaly score $a(\mathbf{p}) \in [0, 1]$ for each
      $\mathbf{p} \in D$
```

With $|D| = n$, it can be shown (not obvious!) that the expected length to a random point in an IsoTree is

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n},$$

where $H(n)$ is the $(n)^{\text{th}}$ harmonic number: $H(n) \approx \ln n + 0.577$.

The normalized anomaly score of $\mathbf{p}$ in the IsoForest, $a(\mathbf{p})$, is

$$\log_2 a(\mathbf{p}) = -\frac{\text{average path length to } \mathbf{p} \text{ in the Isolation Trees}}{c(n)}.$$

If $a(\mathbf{p}) \approx 1$, we label $\mathbf{p}$ an **anomaly**; if $a(\mathbf{p}) \leq 0.5$, a **regular observation**. If every point receives a score around 0.5, there are no outright outlier.

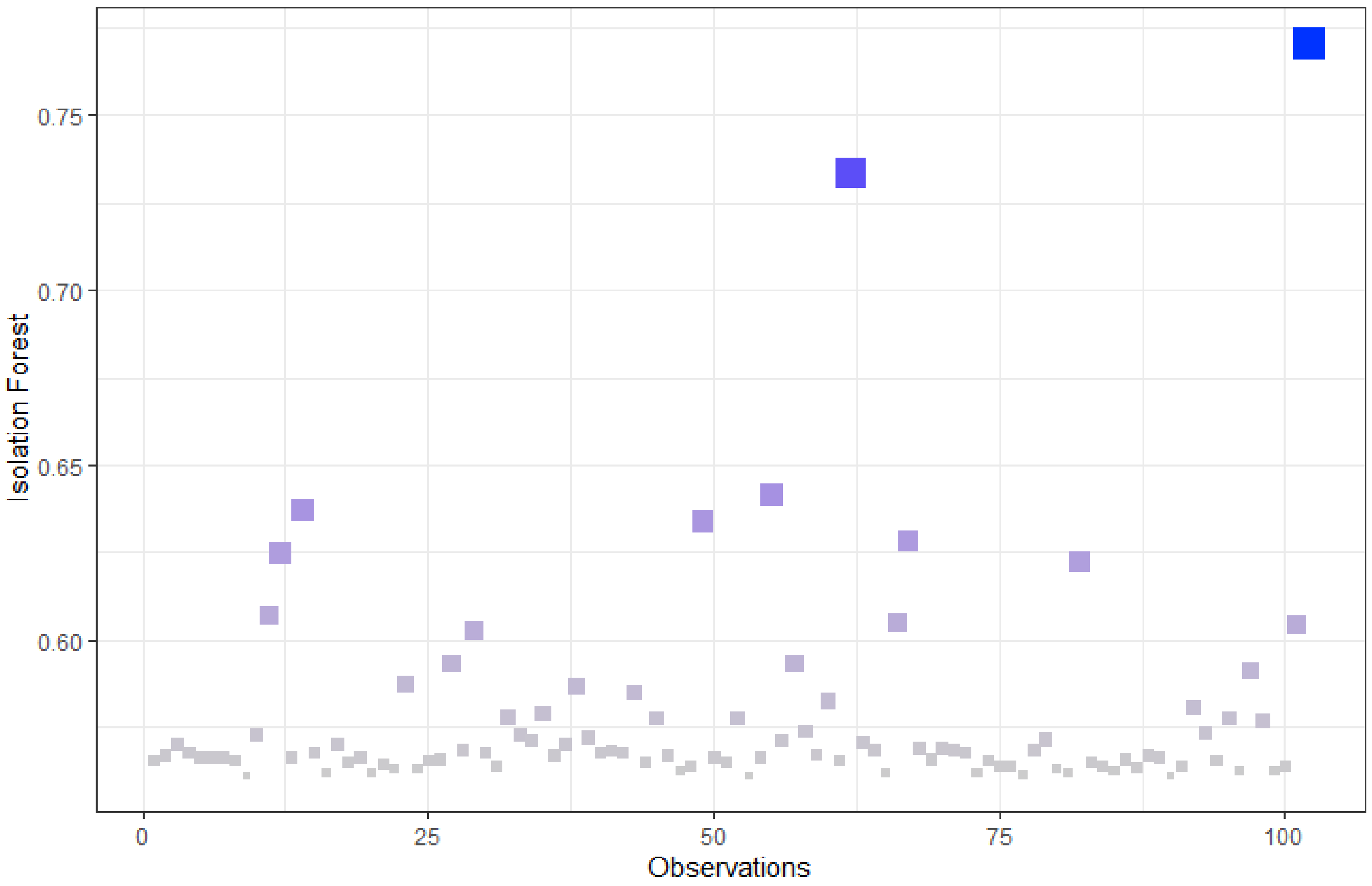
Strengths:

- small time and memory requirements;
- can handle high dimensional data;
- do not need labeled anomalies in the training set.

Main Limitation:

- anomaly score can have high variance over multiple runs.

In general, density-based schemes are more powerful than distance-based schemes when a dataset contains patterns with diverse characteristics, but less effective when the patterns are of comparable densities with the outliers.


Isolation Forest
0.75
0.70
0.65
0.60
0
25
50
75
100
Observations