Algorithmic Data Analysis

Esther Galbrun

Spring 2024

UNIVERSITY OF
EASTERN FINLAND

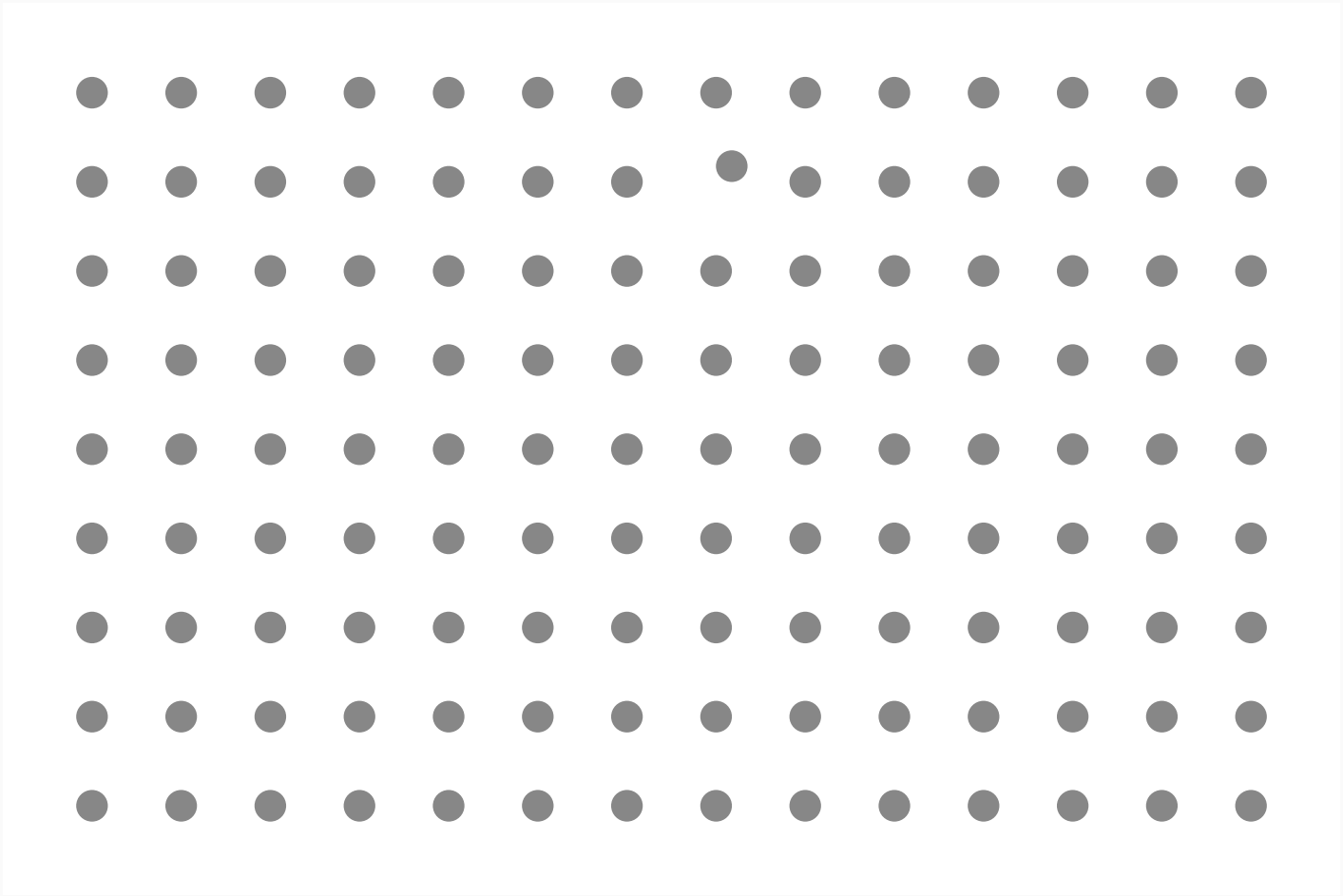
Last updated on 2023-12-28 11:31

Part VII

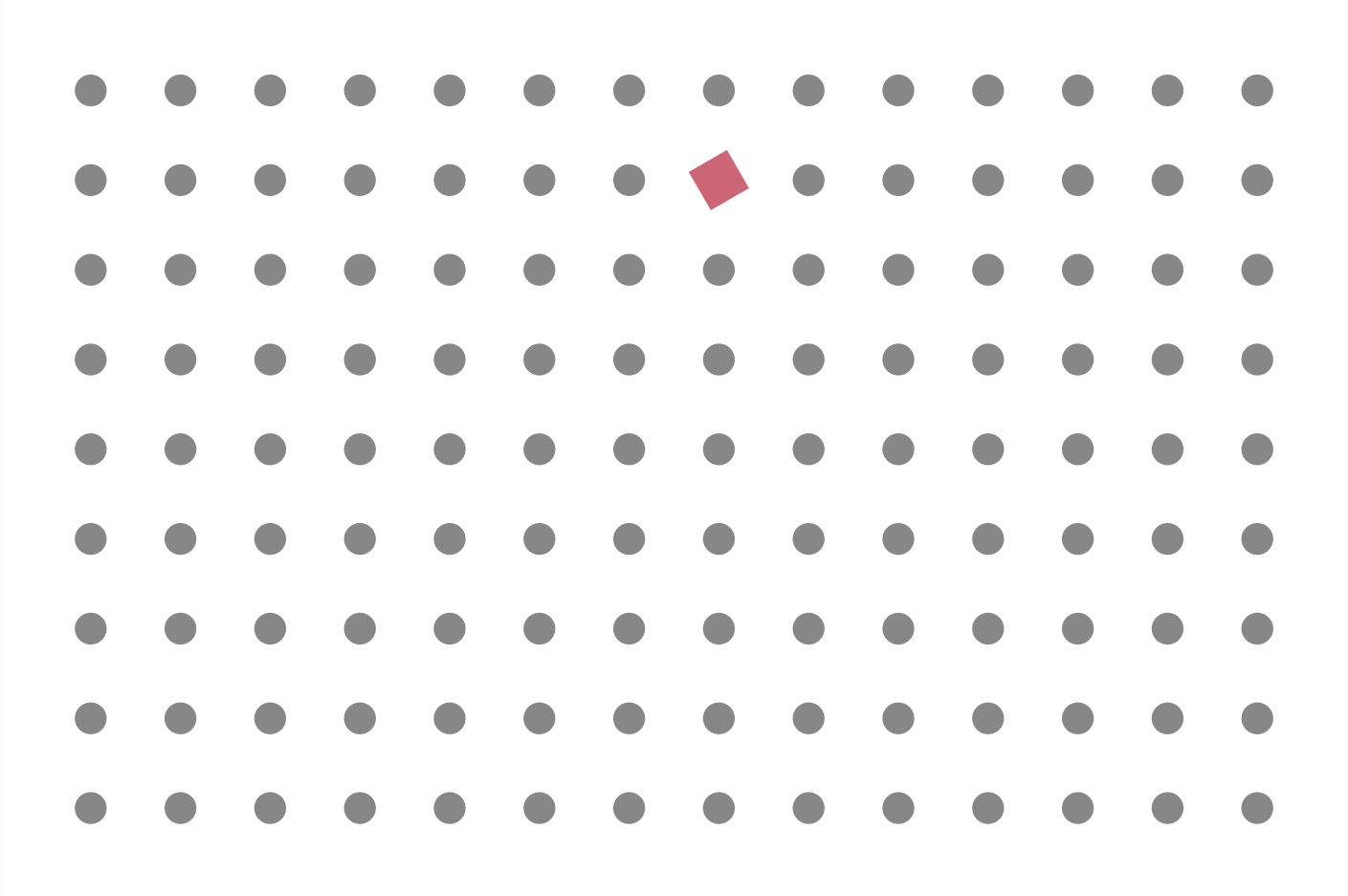
Outlier Analysis

Basics

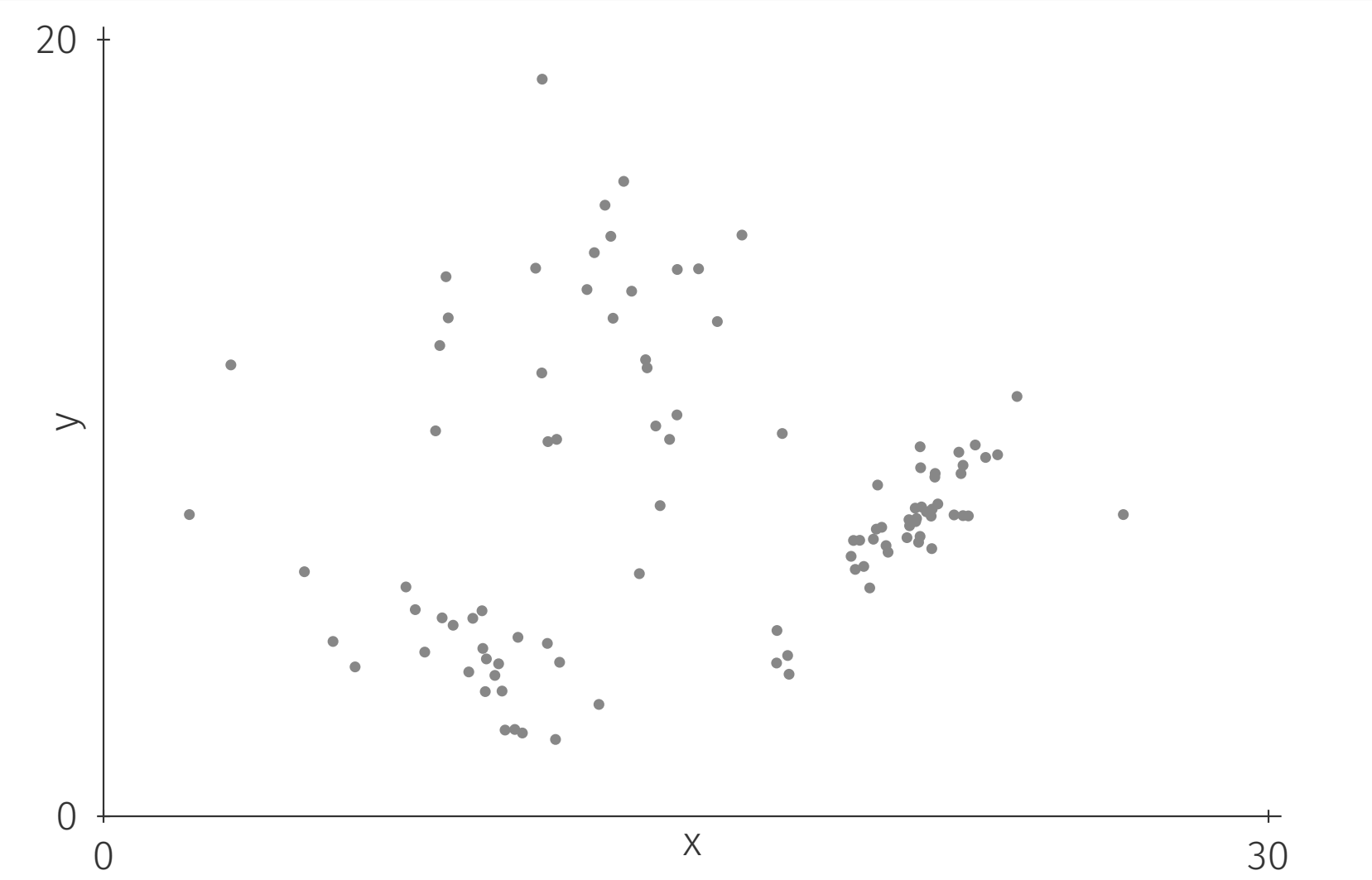
What is an outlier?



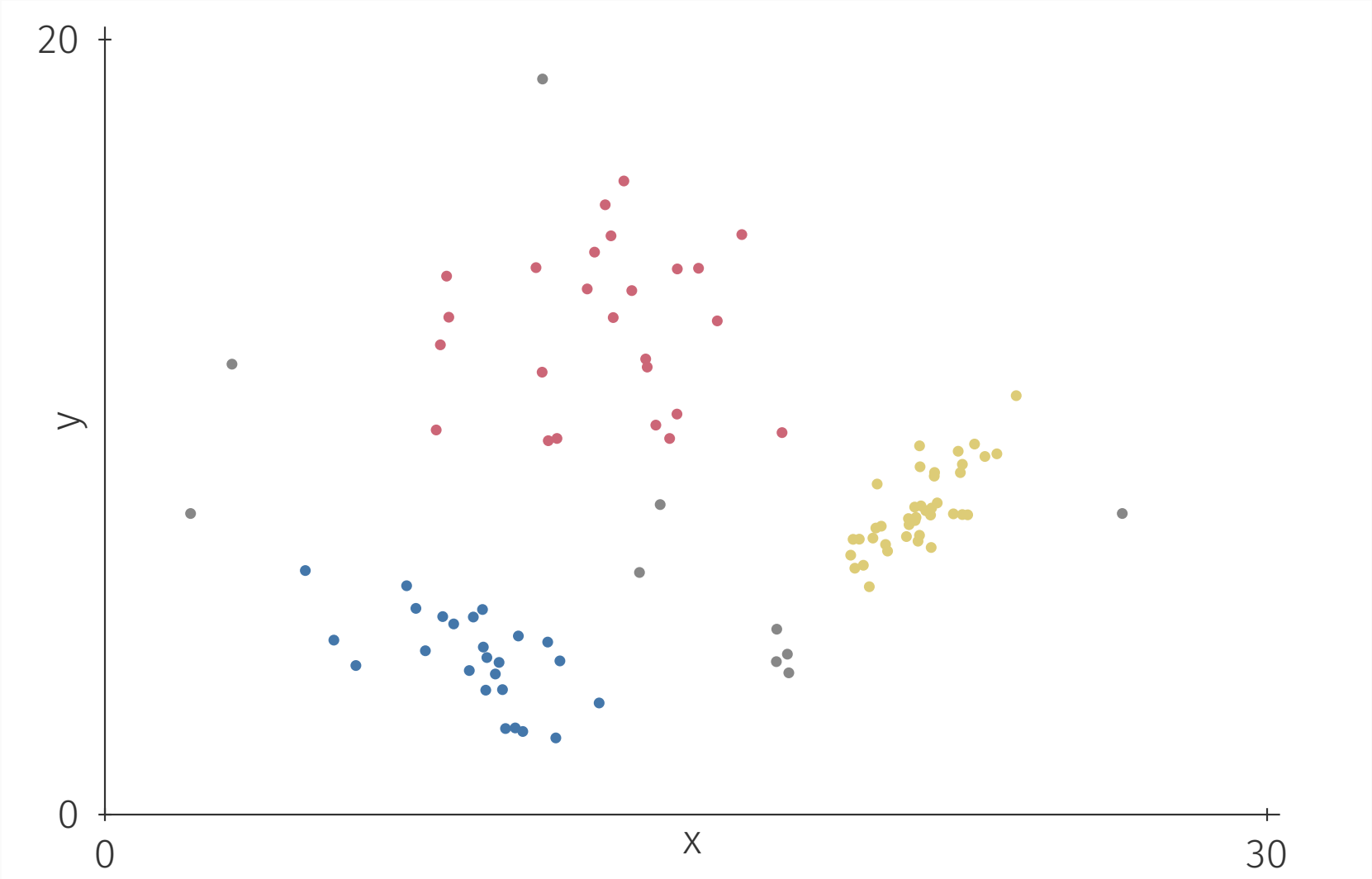
What is an outlier?



What is an outlier?



What is an outlier?



What is an outlier?

An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism.

D. M. Hawkins, 1980

What is an outlier?

Outliers can be seen as a complementary concept to clusters

Clusters are groups of data points that are similar
Outliers are individual data points that are not similar to the rest of the data

Outliers are also known as anomalies, abnormalities, discordants or deviants

Applications

The distinction between noise and interesting outliers can be difficult to make, in general

Outliers result from fluctuations during *data generation*

Noise are caused by artifacts of the *data collection* process

Detecting measurement errors

Outliers detection methods are sometimes used to identify measurement errors, seen as noise, that should be removed before further processing

> *One man's noise is another man's signal*
>
> E. Ng, 1990

Applications

Credit card fraud detection

Credit card companies maintain a record of transactions, including attributes such as user identifier, amount spent, timestamp, geographic location

Fraudulent transactions often show up as unusual combinations of attributes

Unusual patterns of credit card activity as a result of fraud
Much rarer than the normal patterns, can be detected as outliers

Applications

Quality control and fault detection

Track the number of defective units produced to detect anomalies in a manufacturing process

Continuous monitoring of production line robot, engine, built infrastructure

Typically involves tracking various parameters simultaneously

Early detection is desired, to organise preventive maintenance and avoid interruptions

Applications

Web log analytics and intrusion detection

Web sites, networks and computer systems generally automatically track agent behavior

Detect anomalous behavior from web logs or system logs
e.g. user trying to break into password protected website
Identify unusual sequences of actions

Applications

Medicine and public health

Unusual symptoms or test results may indicate health problem of a patient

Whether or not a result is abnormal often depends on characteristics of the patient, e.g. age, gender, etc.

Track occurrences of particular diseases across hospitals

Detect problems with, e.g. vaccination program

Applications

Sports statistics

Record various parameters about the performance of athletes and players

Identify outstanding players, detect cheating

Applications

Credit card fraud detection

Quality control and fault detection

Web log analytics and intrusion detection

Medicine and public health

Sports statistics

...

Swamping and masking

Swamping happens when the number of normal instances increases or they become scattered so that normal instances are wrongly identified as outliers

Masking happens when the number of outliers increases, forming dense clusters of anomalous data points and concealing their own presence

Both issues are consequences of too large amounts of data used for the detection of outliers
This can be solved by using subsampling

Analysis scenarios

Supervised scenario

Training data containing data points labelled as normal and abnormal is provided

There might be multiple normal and abnormal categories

This corresponds to a classification problem, often highly unbalanced

Semi-supervised scenario

Only partial labels are provided, e.g. data points only from the normal categories

Unsupervised scenario

In most cases, outlier detection is performed in an unsupervised manner, with no training data

Analysis scenarios

Unsupervised scenario

In most cases, outlier detection is performed in an unsupervised manner, with no training data

Unsupervised outlier detection is closely related to clustering

Many clustering algorithms do not assign all points to clusters to account for noisy data points

However, clustering algorithms are optimized to find clusters, not outliers
Multiple similar abnormal data points might be reported as a separate cluster

Analysis approaches

Reference set with respect to which normality is evaluated

Global vs. local approaches

Analysis approaches

Reference set with respect to which normality is evaluated

Global approaches

The reference set contains all other data points

Assumption: single normal generating mechanism

Drawback: other outliers in the reference set may falsify results

Local approaches

The reference set consists of a selected subset of data points

No *assumption* on number of normal generating mechanisms

Drawback: relies on appropriate choice of reference subset

Some approaches let the reference set vary from a single data point (local) to the entire dataset (global) automatically or depending on a user-defined parameter

Analysis approaches

Type of output

Labelling vs. scoring approaches

Analysis approaches

Type of output

Labelling approaches

Binary output, label data points as either normal or abnormal, inliers or outliers

Scoring approaches

Real-valued output, compute a score for each data point
e.g. probability of being an outlier
Allows to sort data points according to their scores

Scoring approaches typically focus on top-r outliers for user-defined parameter r
Choosing a threshold value turns scores into binary labels

Evaluation

Evaluating an outlier detection algorithm requires ground truth data, i.e. to know which points are true outliers

Outlier detection algorithms are typically evaluated on

- synthetic data with identified outliers or
- considering the rare class of labelled real-world data as ground truth

Rare classes do not always reflect all natural outliers in the data, but are generally representative enough when the evaluation is repeated over many datasets

Evaluation

Consider the (rare) class of outliers as the positive class and the rest of data points as the negative class

For algorithms that return an outlier score, this score is turned into a binary label using a threshold

A *strict threshold* will lead to reporting *fewer outliers*,
both true outliers as well as falsely detected ones,
i.e. *low* true positive rate (TPR) and false positive rate (FPR)

A *relaxed threshold* will lead to reporting *many outliers*,
i.e. *high* true positive rate (TPR) and false positive rate (FPR)

The curve showing the trade-off FPR vs. TPR is called the
receiver operating characteristic (ROC) curve

Evaluation

The curve showing the trade-off FPR vs. TPR is called the **receiver operating characteristic (ROC) curve**

Compare different algorithms by comparing their ROC curves

! All regions of the curve might not be equally important depending on the application

! Using the ROC curve and the area under the curve (AUC) to tune an algorithm can lead to drastic overestimation of the accuracy

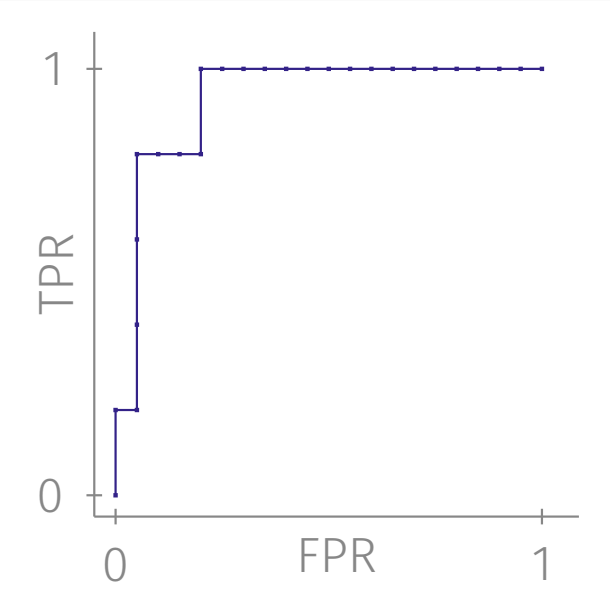
ROC curve

Dataset: 20 normal data points, 5 anomalies

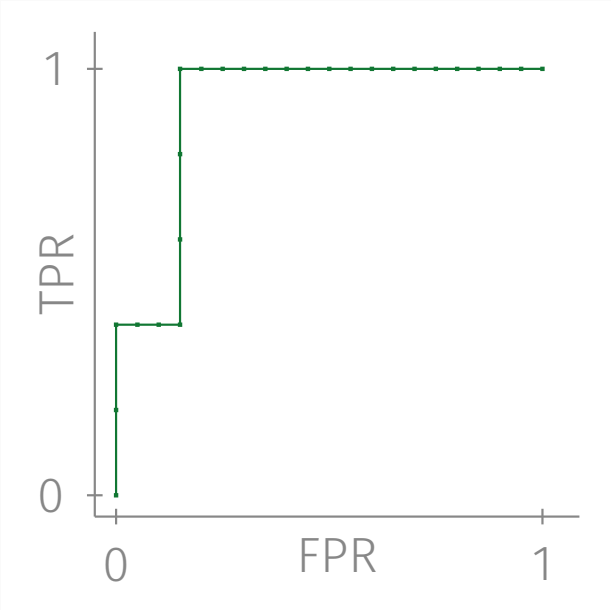
Consider three algorithms that rank data points by decreasing likelihood of being anomalous

Compare by looking at the positions of the anomalies in respective rankings

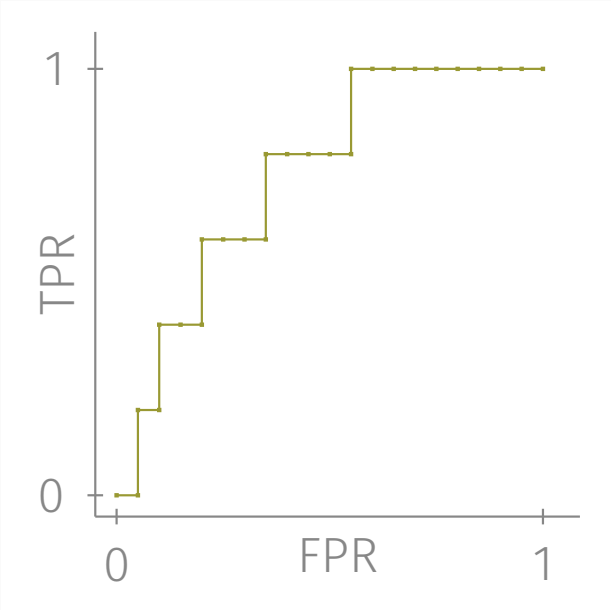
$\langle 1, 3, 4, 5, 9 \rangle$



$\langle 1, 2, 6, 7, 8 \rangle$



$\langle 2, 4, 7, 11, 16 \rangle$



Depth-based methods

Assumption: outliers lie at the border of the data space, whereas inliers lie in the center of the data space

Organize the data points into convex hull layers
i.e. peel the data layer by layer, like an onion
Depth of layer is used as score
Points on the ℓ outermost layers are declared outliers

Depth-based methods

Peel the data layer by layer, like an onion

Depth of layer is used as score

Points on the ℓ outermost layers are declared outliers

$\delta \leftarrow 1$

while $\mathcal{D} \neq \emptyset$ **do**

  $\mathcal{H}$ corners of the convex hull of $\mathcal{D}$

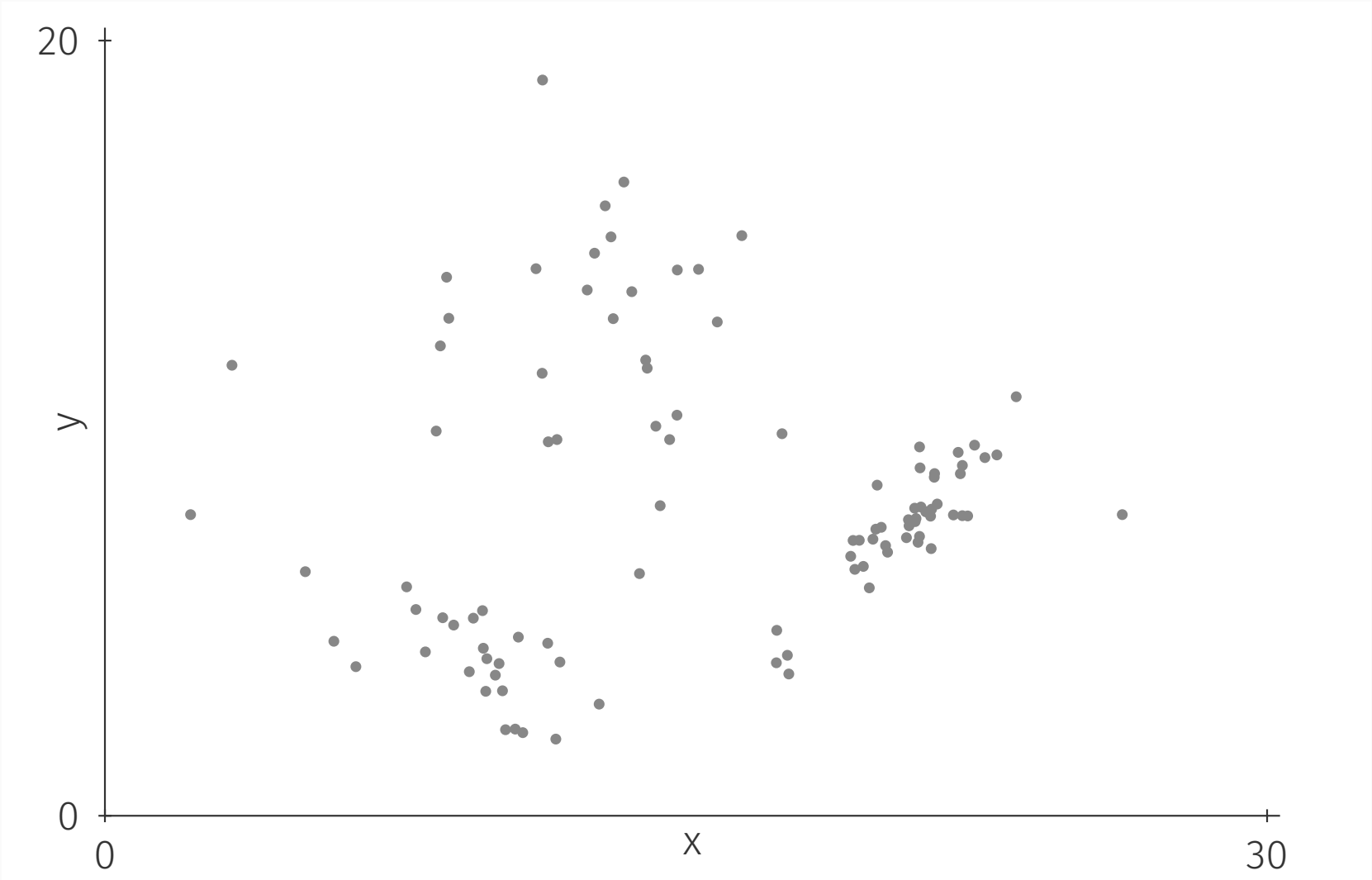
  $o_x \leftarrow \delta$, for $x \in \mathcal{H}$

  $\mathcal{D} \leftarrow \mathcal{D} \setminus \mathcal{H}$

  $\delta \leftarrow \delta + 1$

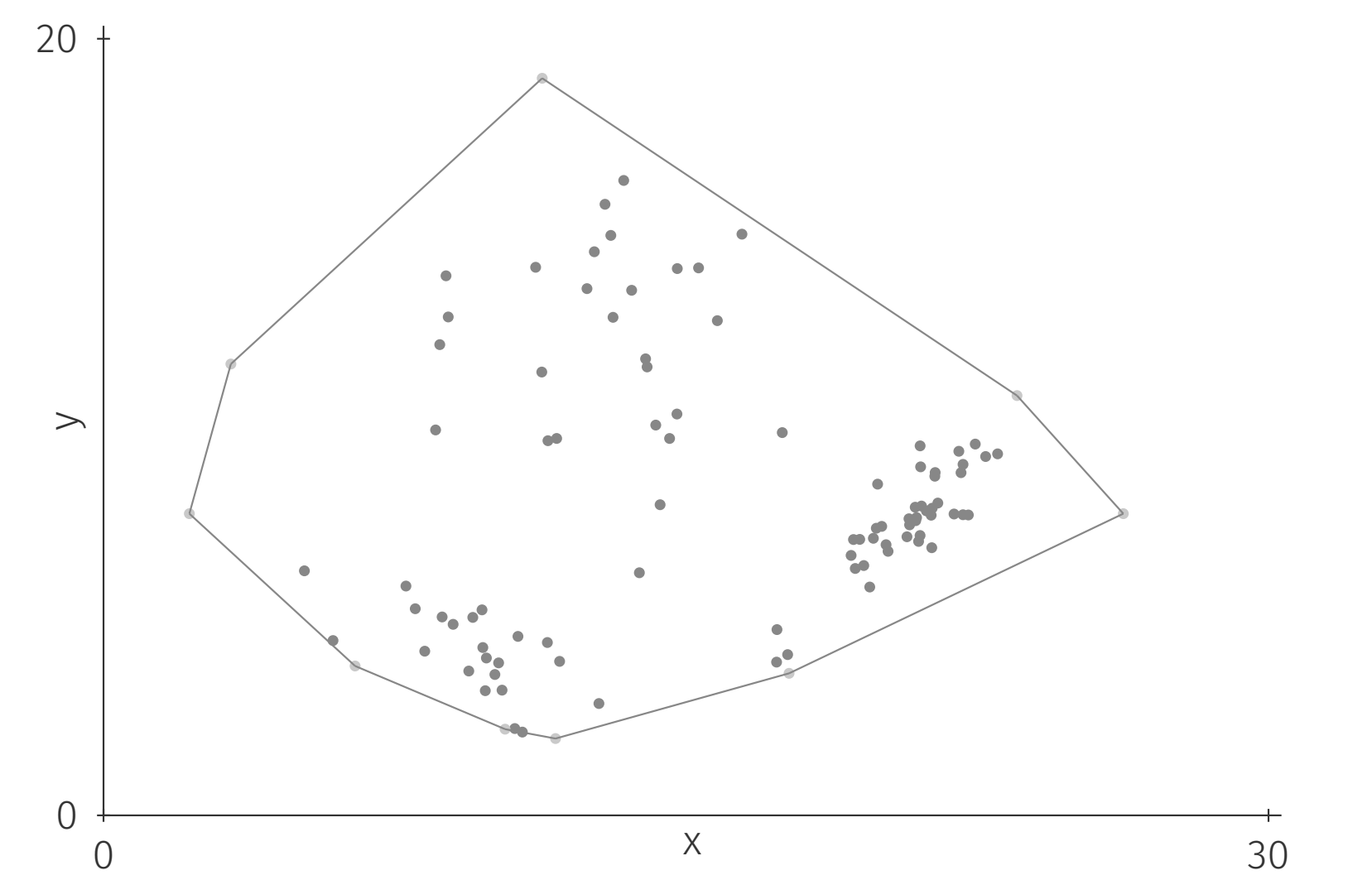
return $\{x \in \mathcal{D}, o_x \leq \ell\}$

Depth-based methods



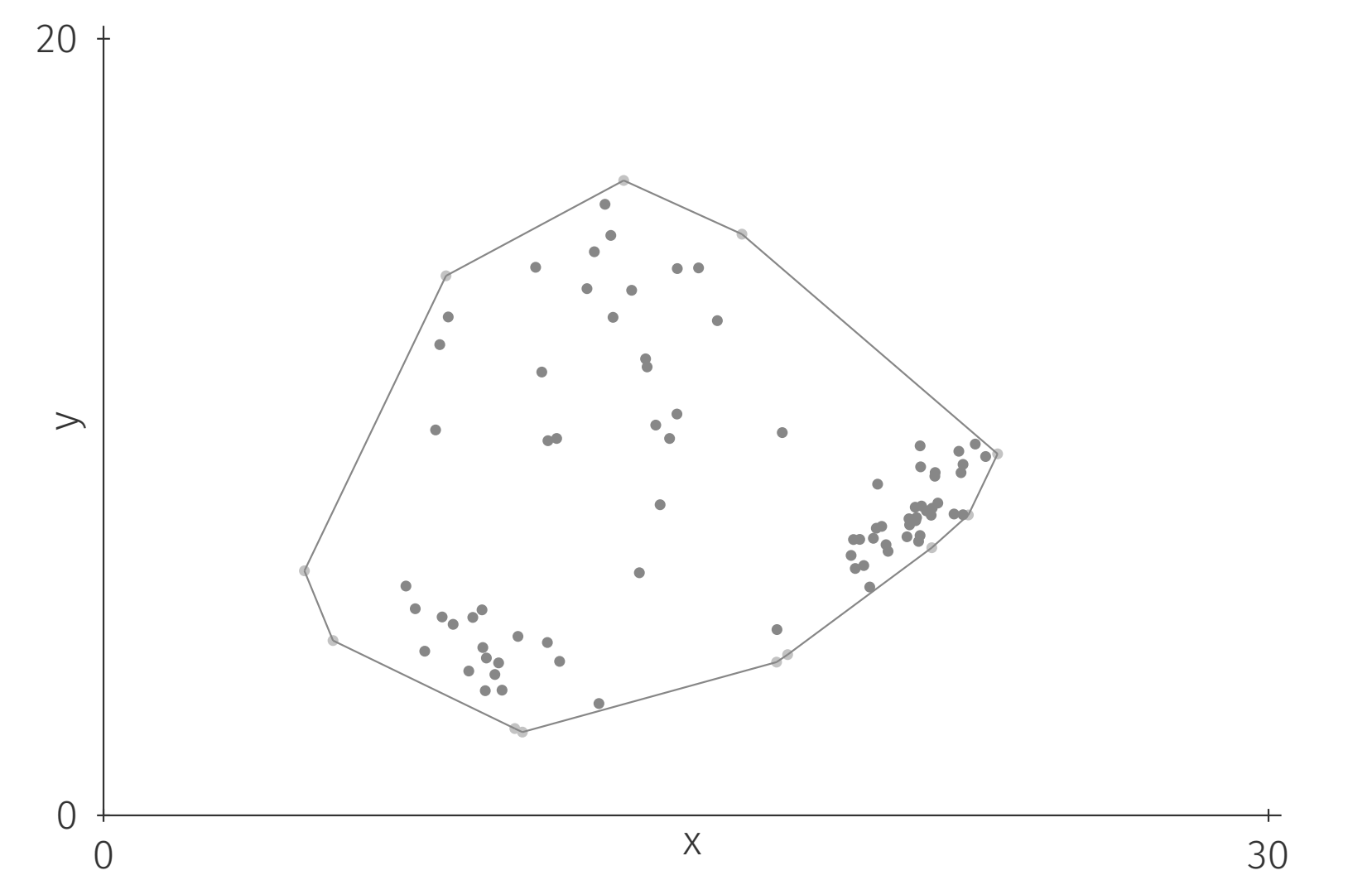
Depth-based methods

Peeling the outer layer of the data



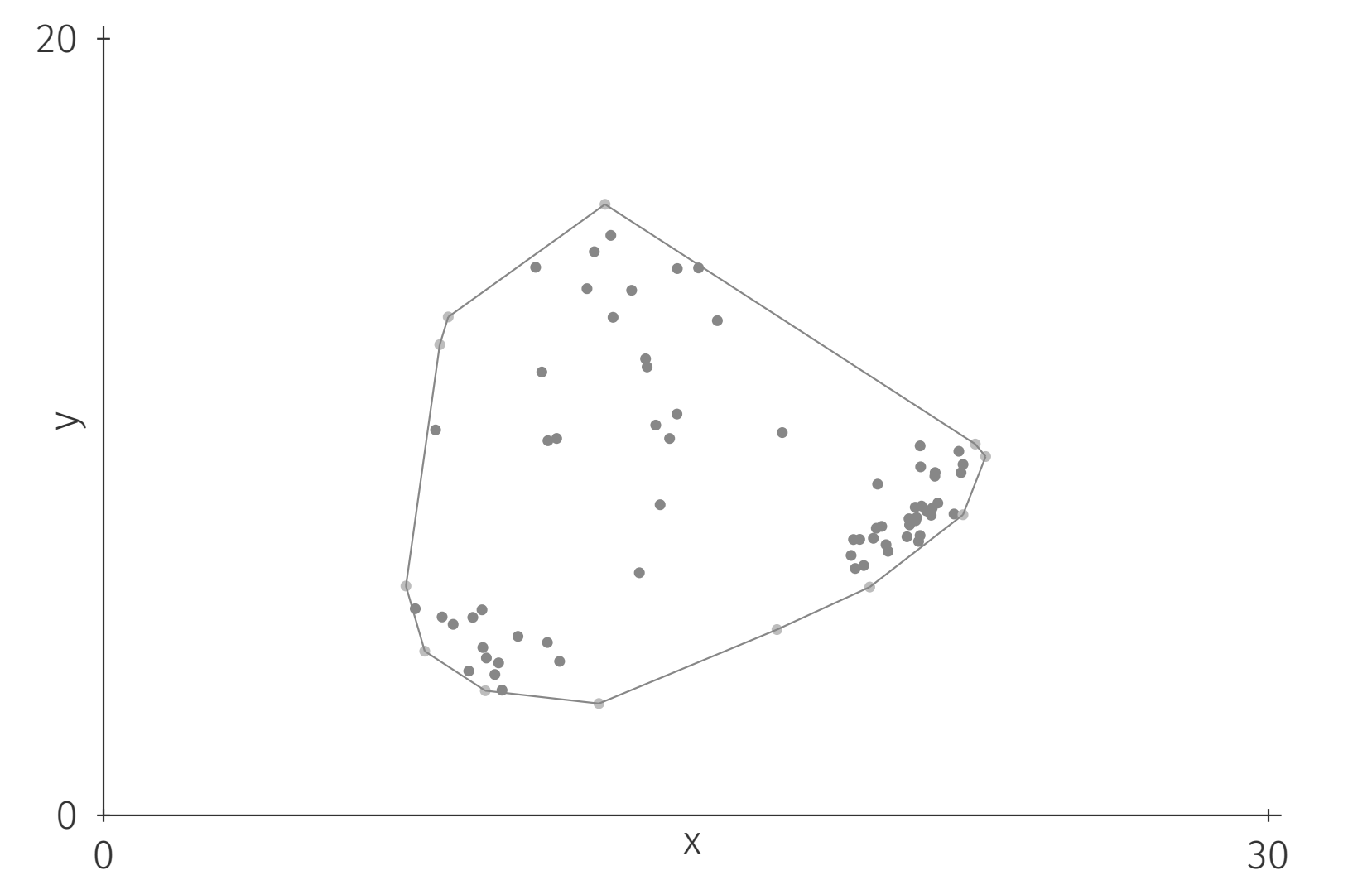
Depth-based methods

Peeling the outer layer of the data



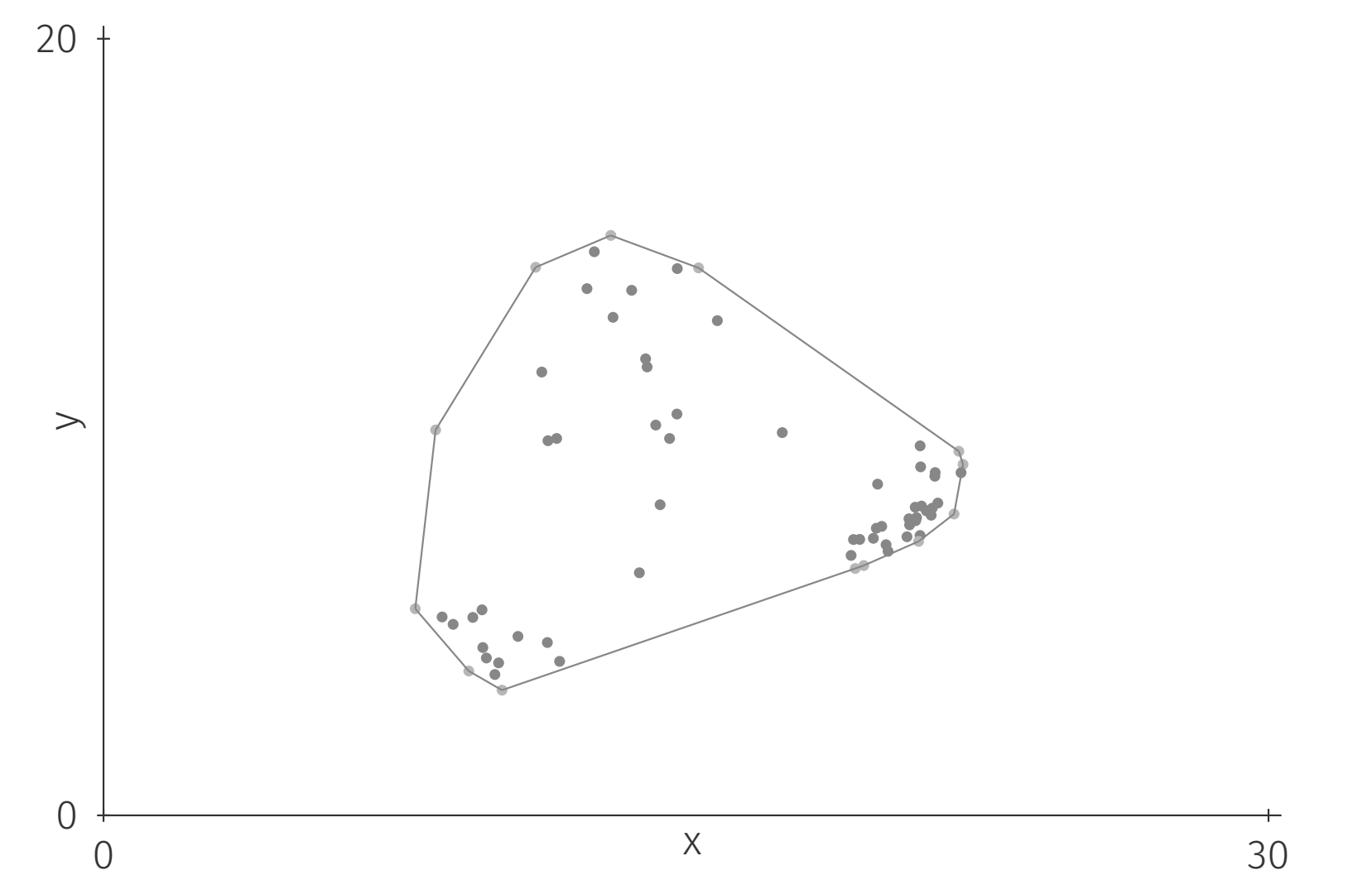
Depth-based methods

Peeling the outer layer of the data



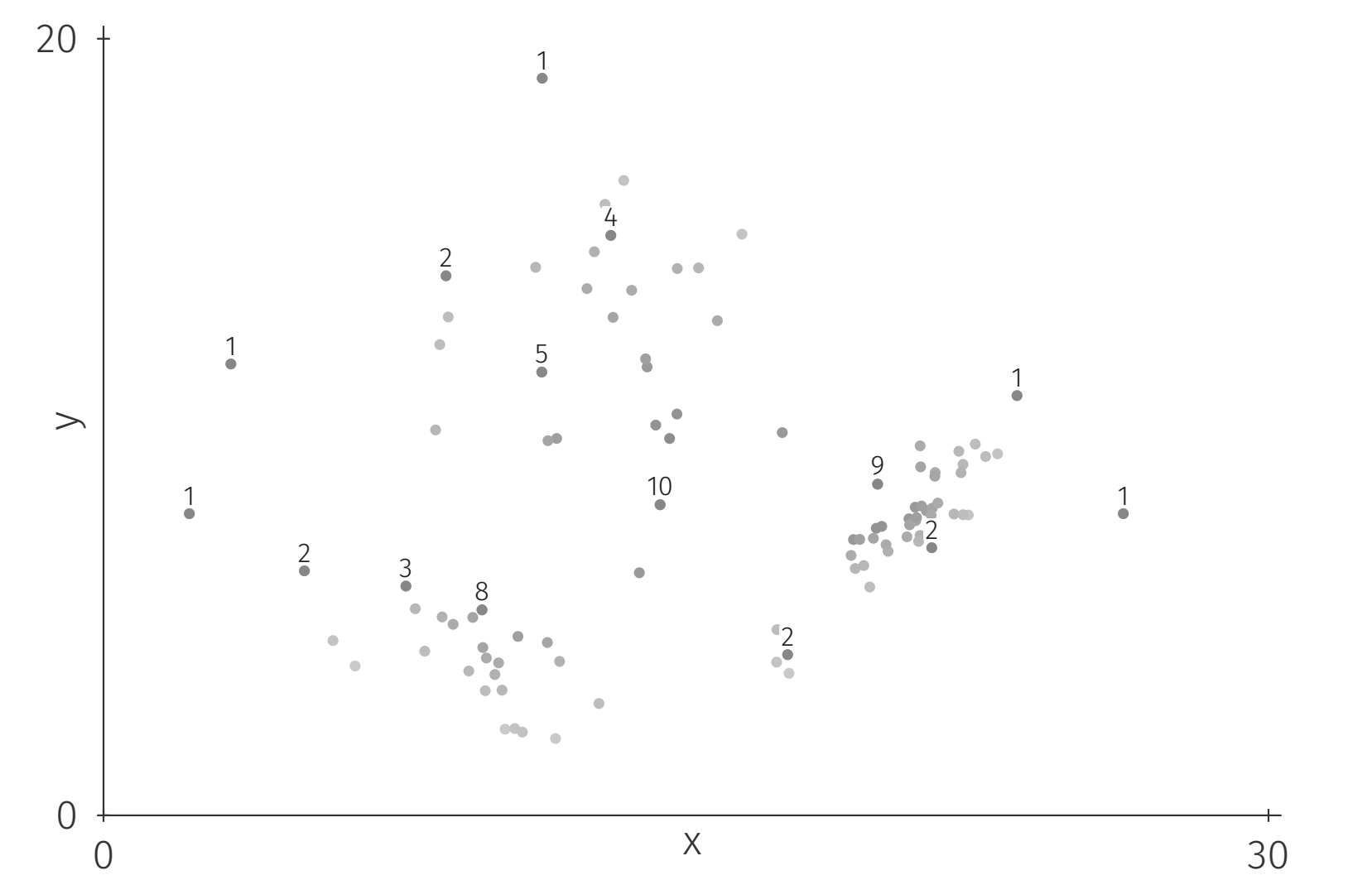
Depth-based methods

Peeling the outer layer of the data



Depth-based methods

Depth of layer as outlier score



Depth-based methods

Peel the data layer by layer, like an onion
Depth of layer is used as score
Points on the ℓ outermost layers are declared outliers

All points on the same layer are treated equally

Typically, increased dimensionality leads to increase of

- computational complexity of finding the convex hull
- fraction of points at corners of the convex hull
- number of undistinguishable points

Deviation-based methods

Assumption: outliers are the outermost points in the dataset

For a given set of points, the outliers are those points that do not fit the general characteristics of the set, the variance of the set is minimized when removing them

Information-theoretic methods

Assumption: outliers are not similar to the rest of the data

If we compress the data using *normal patterns*, outliers will increase the encoding length

Density-based methods

Assumption: outliers are not similar to the rest of the data

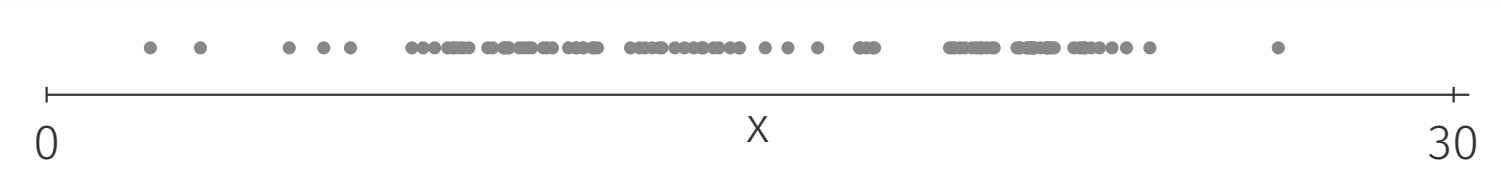
For univariate data, construct a histogram, i.e. discretize the data into bins of equal width, and compute the number of data points in each bin

Points lying in very low frequency bins are reported as outliers
Use the number of other points in the bin as outlier score

With smaller bins widths, more points are reported as outliers
With larger bins widths, anomalies and normal points might be merged, preventing detection
Choosing a suitable bins width is difficult

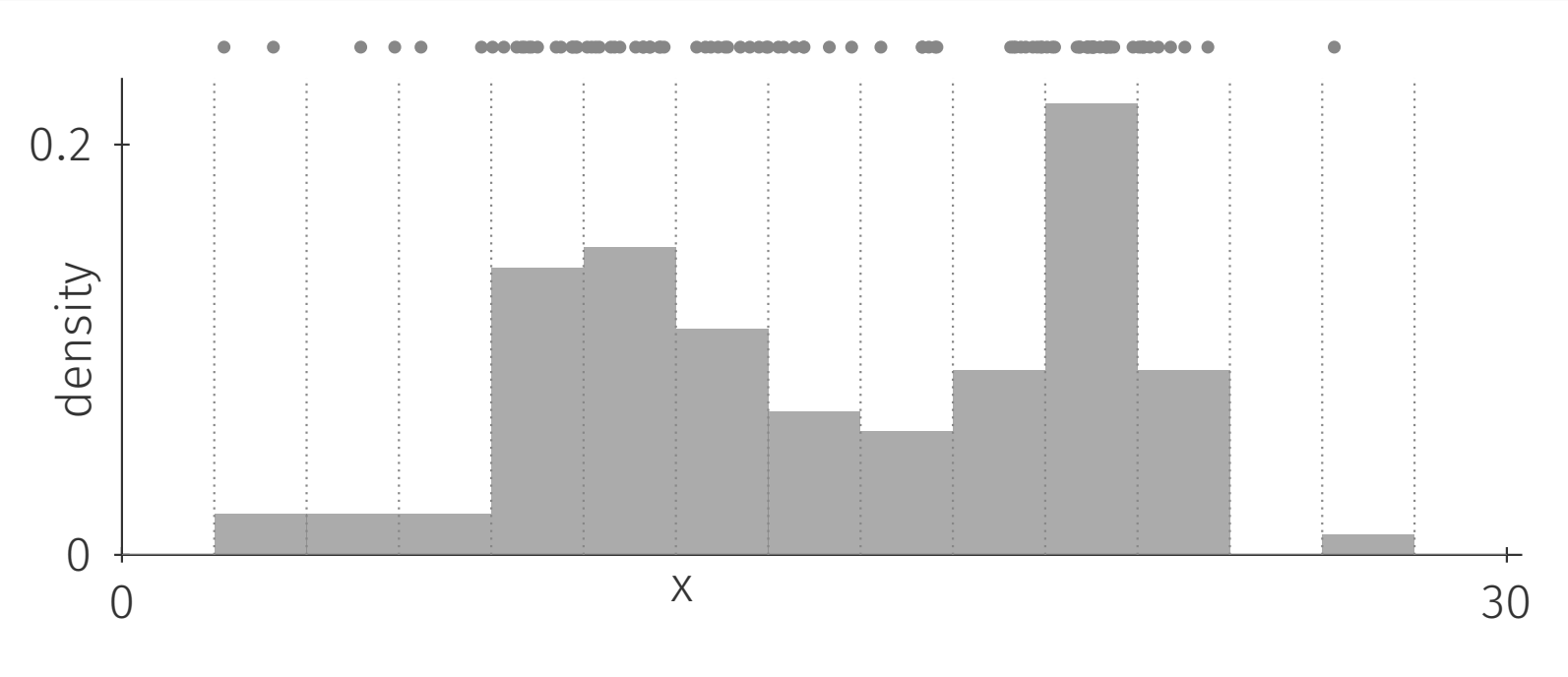
This approach is very local, when granularity is high, an isolated group of points may result in an artificially dense bin

Density-based methods



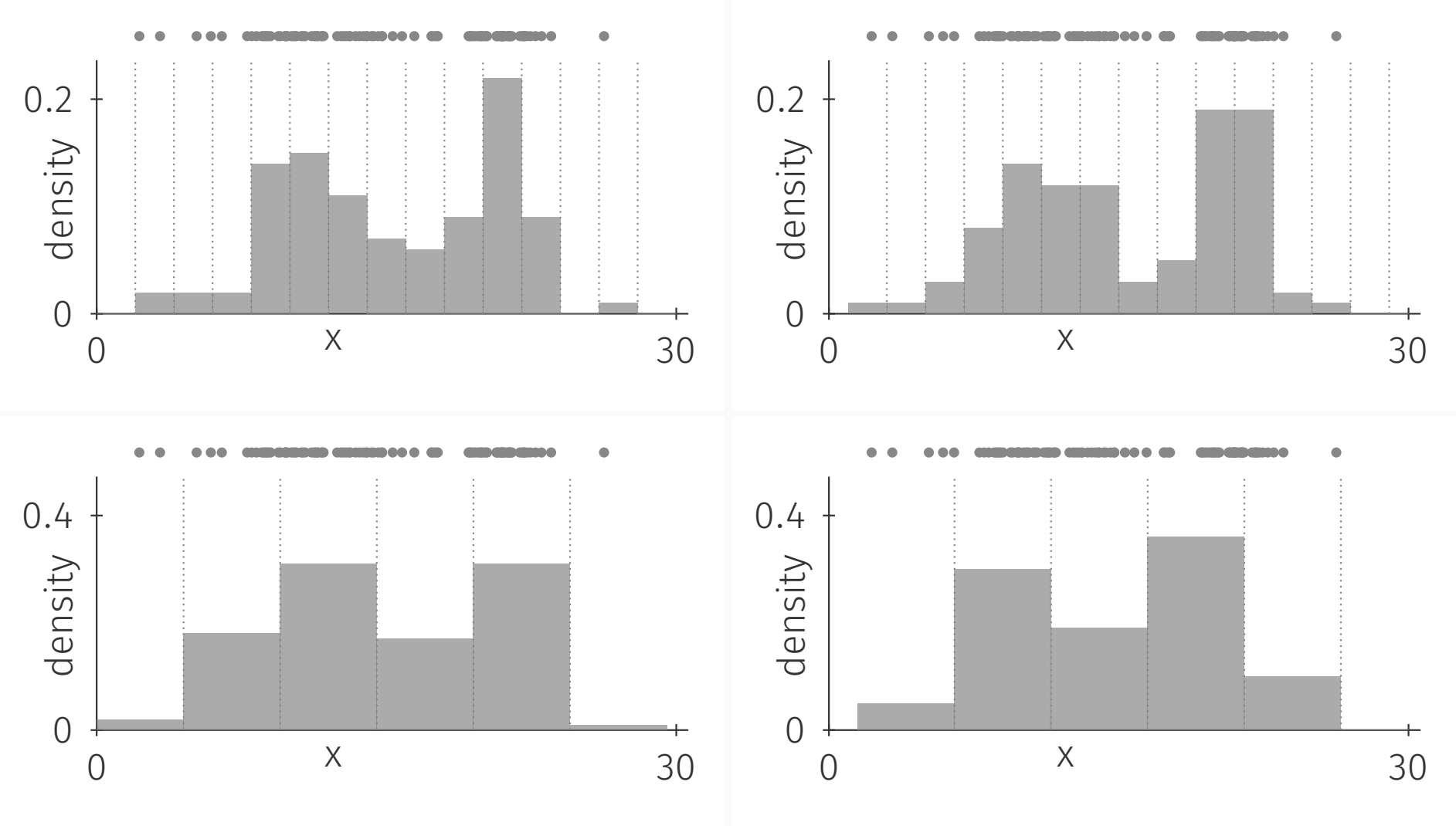
Density-based methods

Histogram for bin width 2 and anchor point 0



Density-based methods

Histograms for different bin widths and anchor points



Density-based methods

Assumption: outliers are not similar to the rest of the data

For multivariate data, construct a grid, i.e. partition each dimension into intervals of equal width, and compute the number of data points in each cell

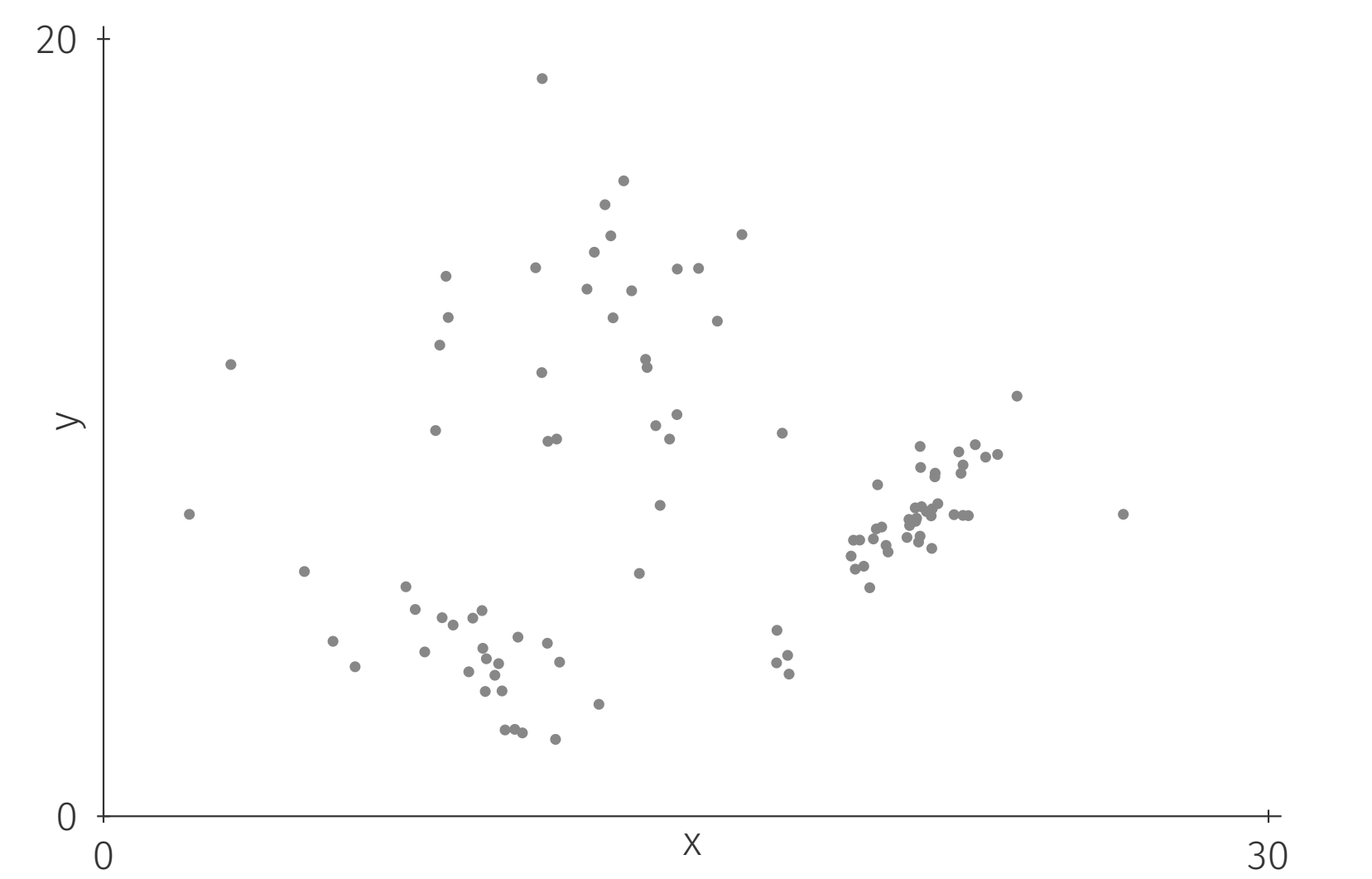
Points lying in very low frequency cells are reported as outliers
Use the number of other points in the cell as outlier score

Choosing a suitable cells widths is difficult

This approach is very local, when granularity is high, an isolated group of points may result in an artificially dense cell

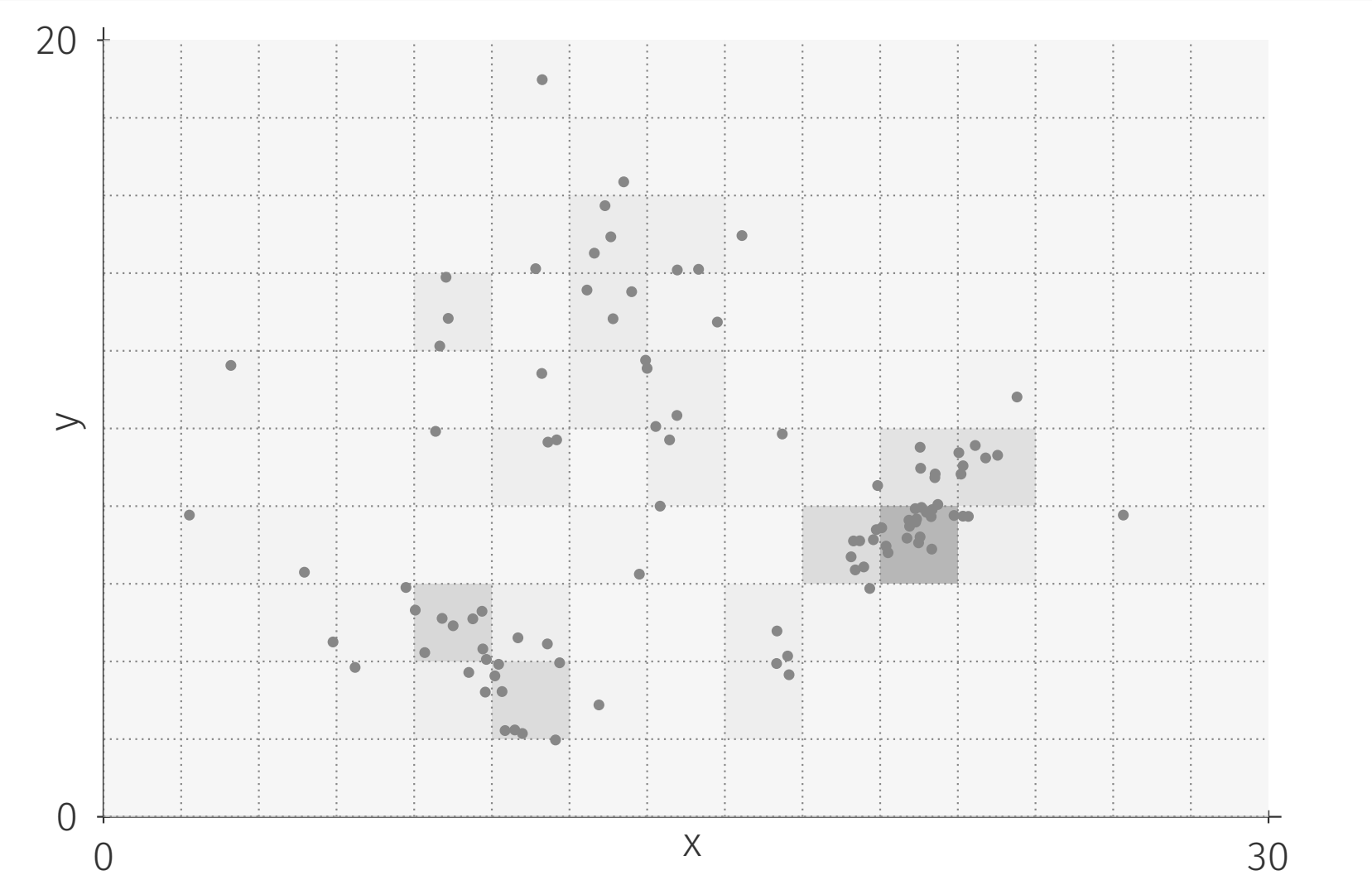
As dimensionality increases, the grid becomes sparser and the expected number of points per cell decreases exponentially

Density-based methods



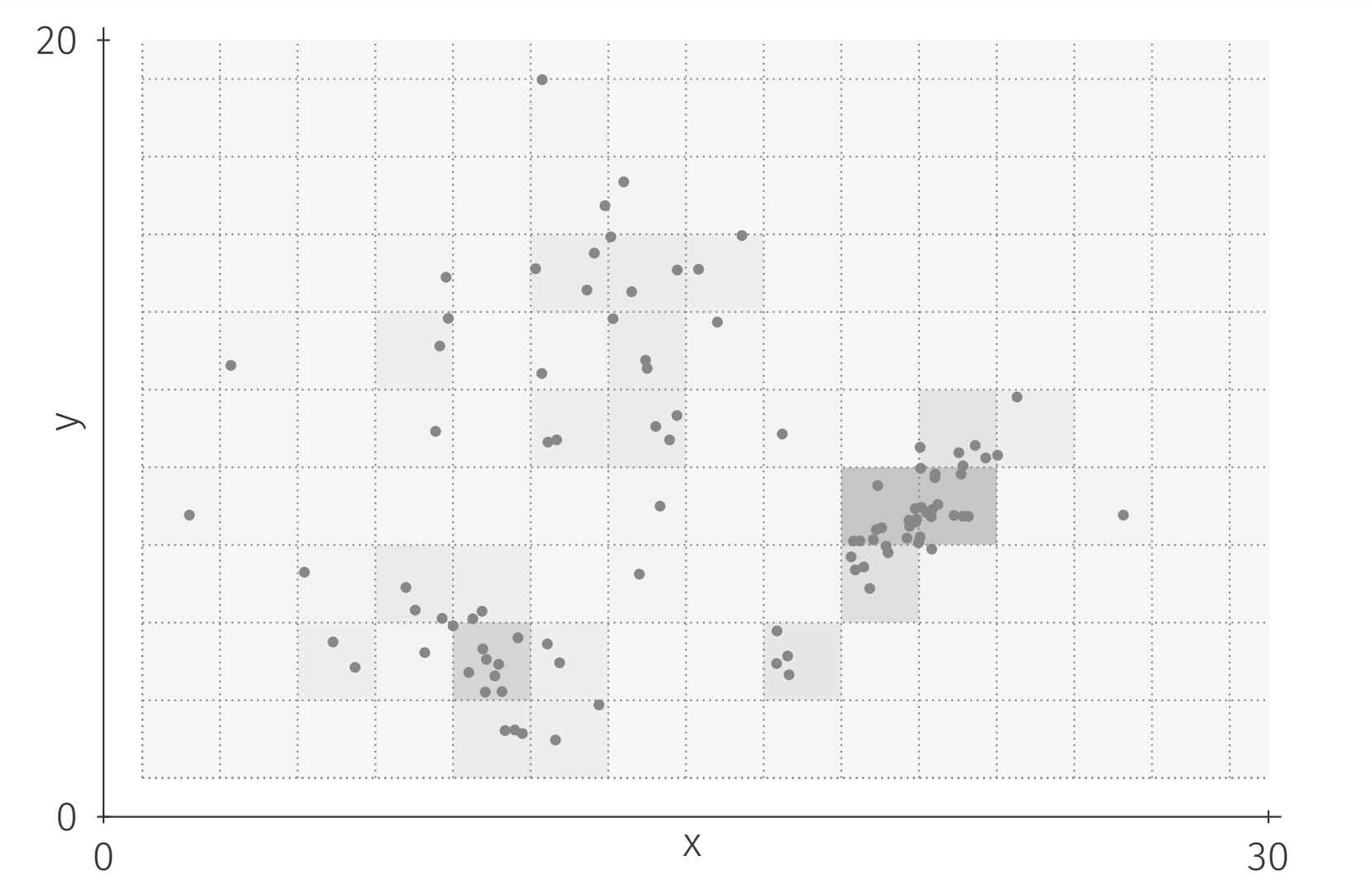
Density-based methods

Histogram for bin width 2 and anchor point $(0, 0)$



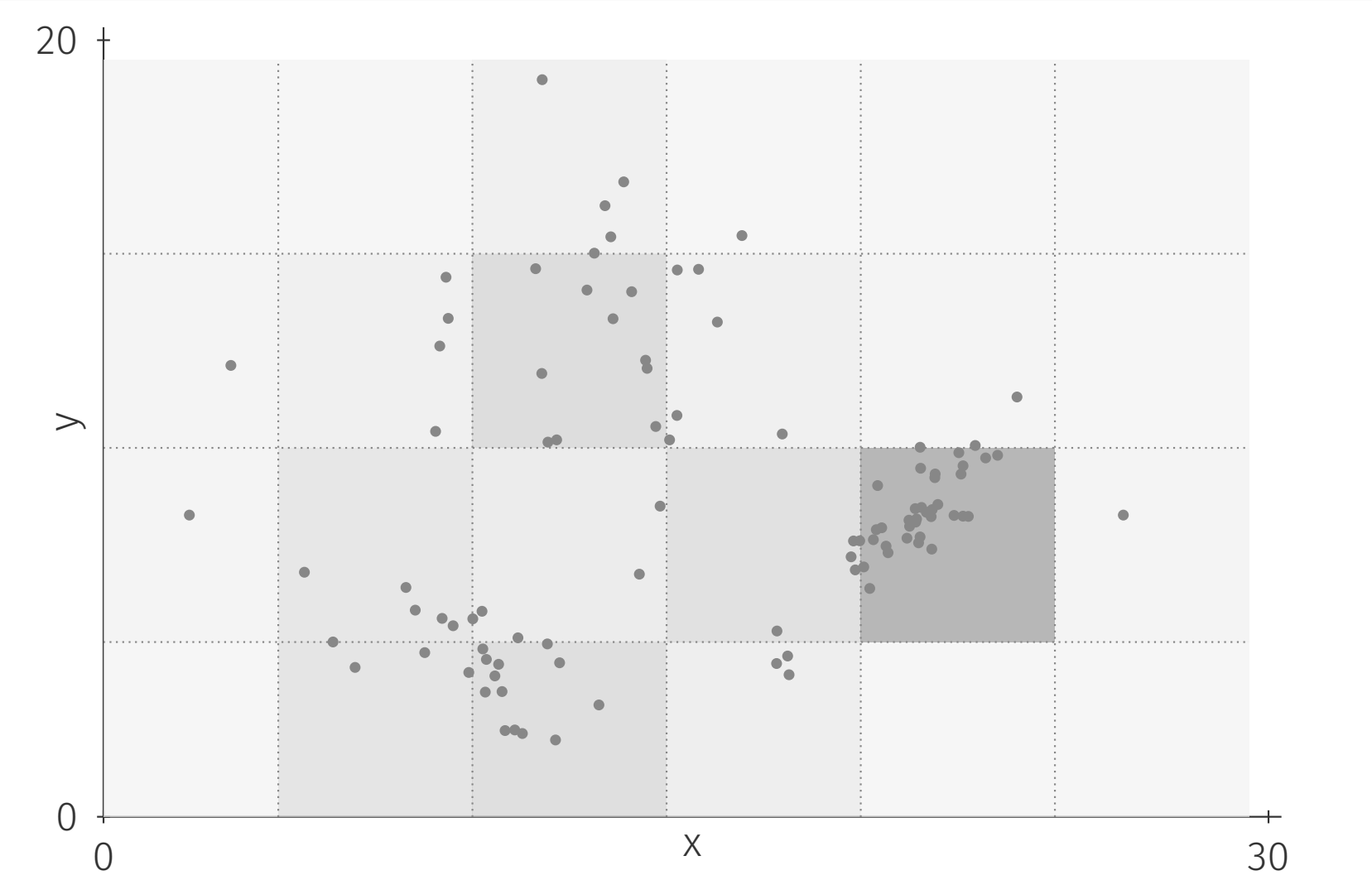
Density-based methods

Histogram for bin width 2 and anchor point (1, 1)



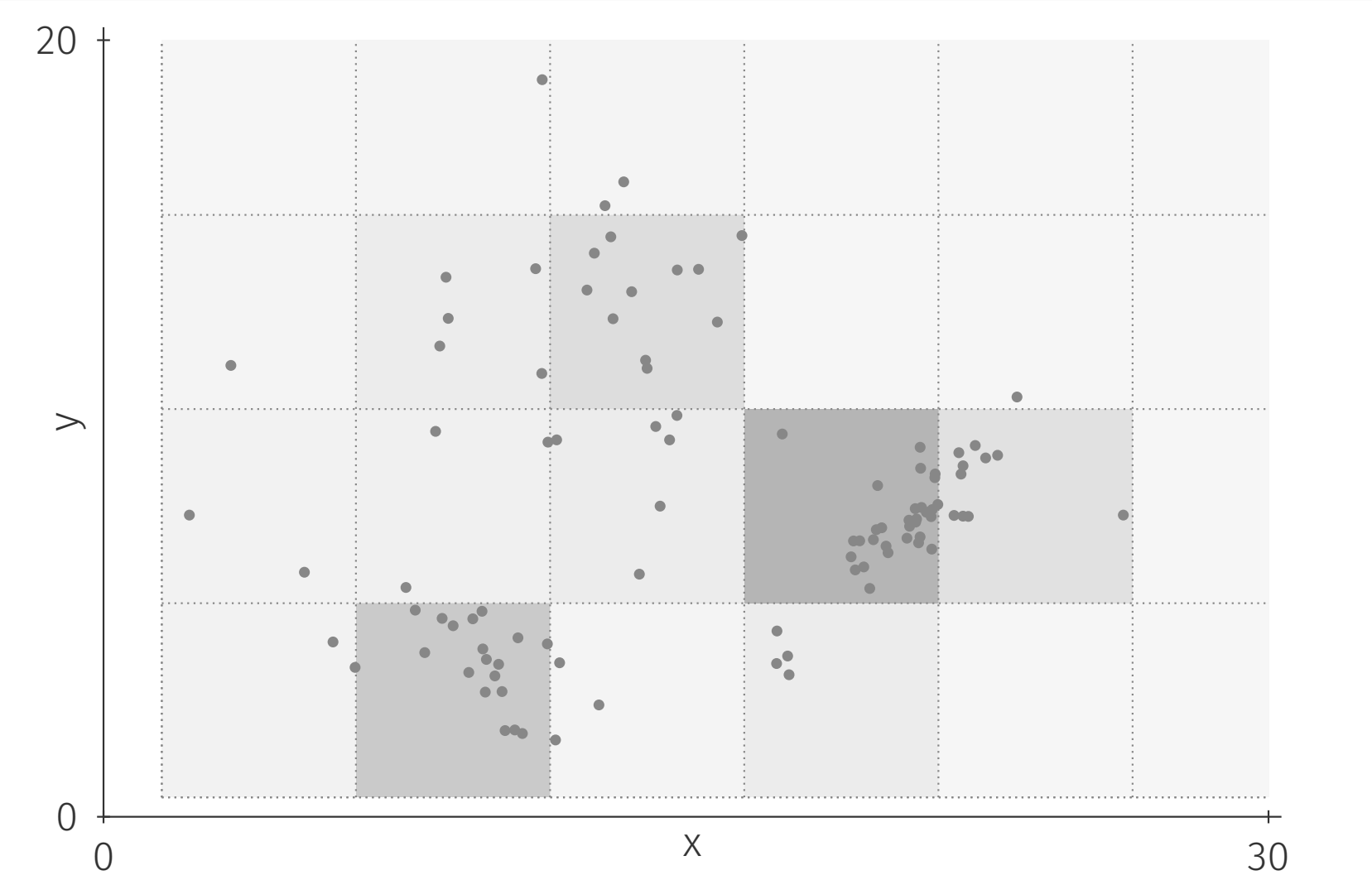
Density-based methods

Histogram for bin width 5 and anchor point $(0, 0)$



Density-based methods

Histogram for bin width 5 and anchor point $(2, 1)$



Density-based methods

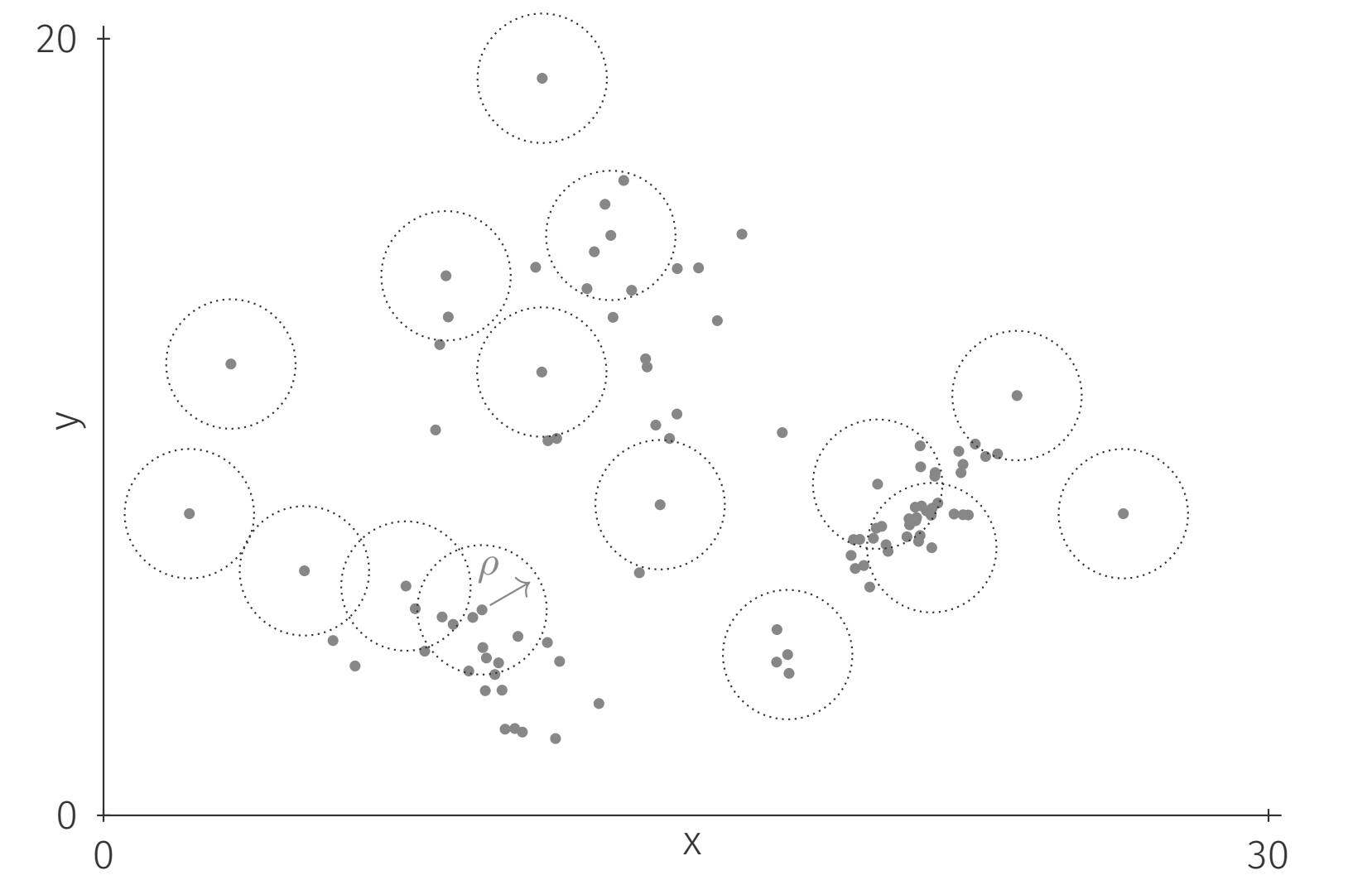
Assumption: outliers are not similar to the rest of the data

Given a radius ρ and a threshold $\tau \in [0, 1]$, a data point $\boldsymbol{x}$ is reported as outlier if at most a fraction τ of the other points are at distance at most ρ from it, i.e. if

$$\left|\{\boldsymbol{x}' \in \mathcal{D} \setminus \{\boldsymbol{x}\}, \mathrm{d}(\boldsymbol{x}, \boldsymbol{x}') \leq \rho\}\right| \leq \tau(n-1)$$

Density-based methods

Looking at points in radius ρ



Statistical tests

Assumption: probability distribution underlying the data generation process
Normal data points occur in high probability regions whereas outliers occur in low probability regions

The parameters of the chosen statistical distribution are estimated assuming all data points were generated by the distribution

Points that have a low probability under the estimated distribution are declared outliers

Data points lying in the low probability regions of the distribution constitute **extreme values**

Univariate extreme values

Assumption: probability distribution underlying the data generation process
Normal data points occur in high probability regions whereas outliers occur in low probability regions

Considering a univariate probability density function $f_{\mathcal{D}}(x)$ the tails of the distribution are the two extreme regions where $f_{\mathcal{D}}(x) \leq \theta$ for some user-defined threshold θ

For distributions that are not symmetric, lower and upper tails may not have the same probability
Some distributions, e.g. exponential, have a tail only at one end

Data points lying in the tails of the distribution constitute **extreme values**

Univariate extreme values

Assuming a univariate Gaussian distribution, the parameters are estimated as the mean μ and standard deviation σ over all data points in $\mathcal{D}$

The probability density function of the Gaussian distribution is

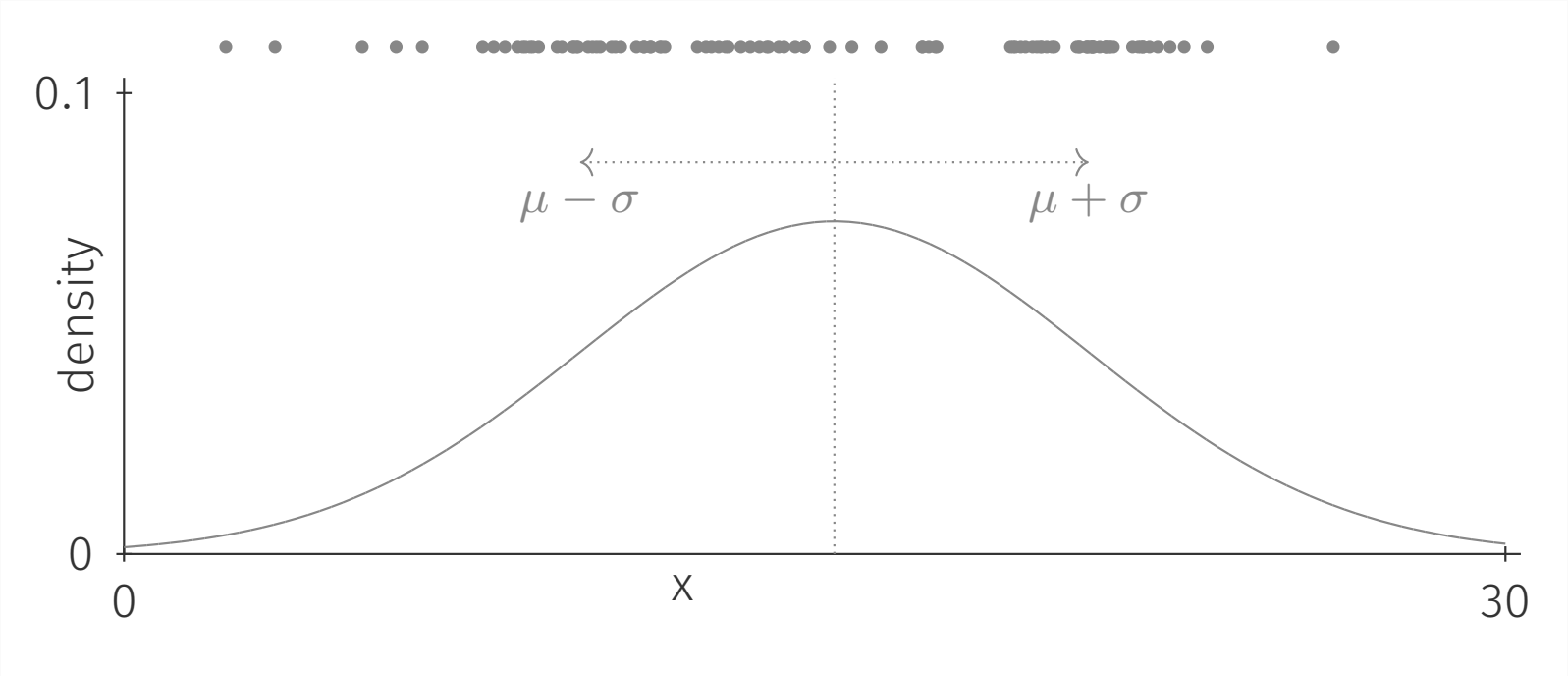
$$f_{\mathcal{D}}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

For a data point x the *standardized* value $z = (x - \mu)/\sigma$ is called its z-**number**

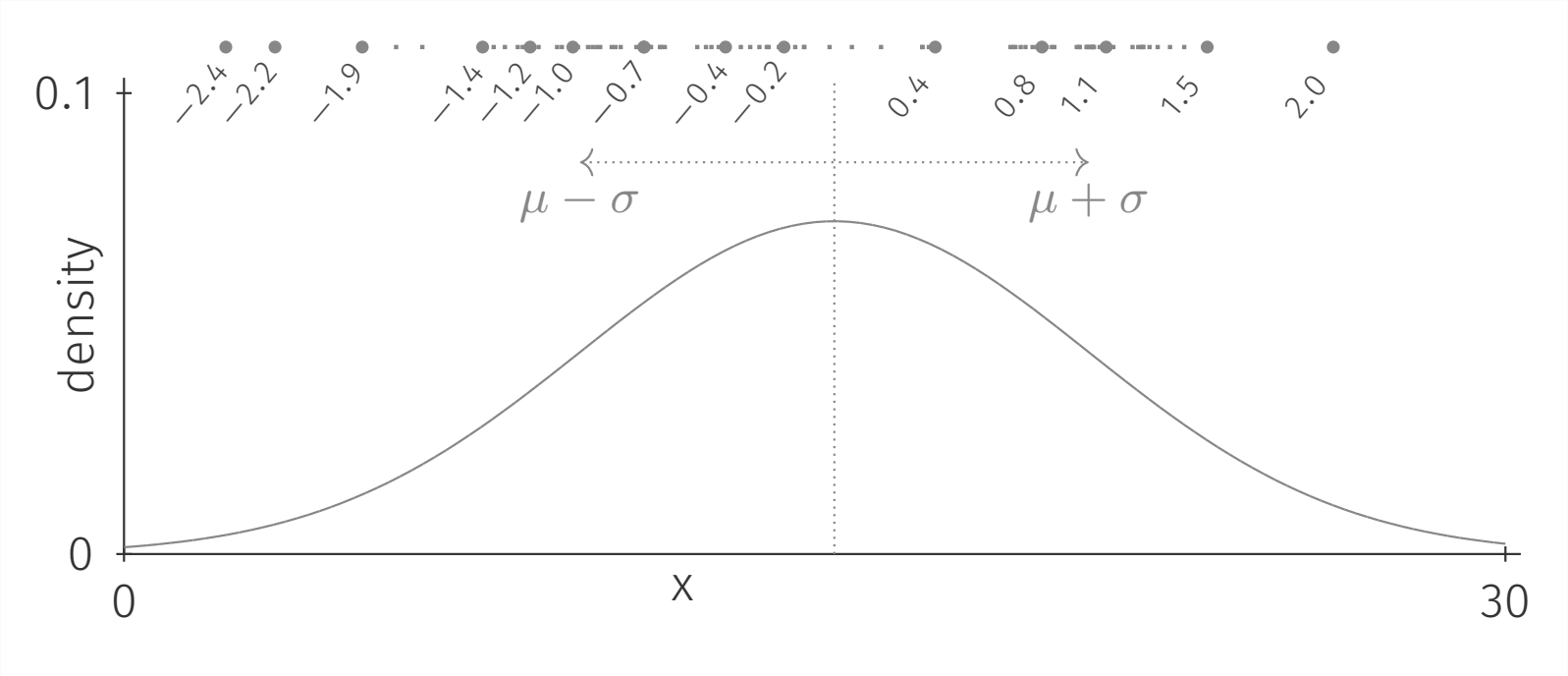
Points in the lower tail correspond to large negative z-numbers
Points in the upper tail correspond to large positive z-numbers

Univariate extreme values

Normal distribution estimated from all univariate data points



Computing z-numbers



Univariate extreme values

Assuming a univariate Gaussian distribution, the parameters are estimated as the mean μ and standard deviation σ over all data points in $\mathcal{D}$

The probability density function of the Gaussian distribution is

$$f_{\mathcal{D}}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

For a data point x the *standardized* value $z = (x - \mu)/\sigma$ is called its z-**number**

The probability density function can be written in terms of the z-number

$$f_{\mathcal{D}}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-z^2}{2}}$$

Univariate extreme values

The probability density function can be written in terms of the z-number

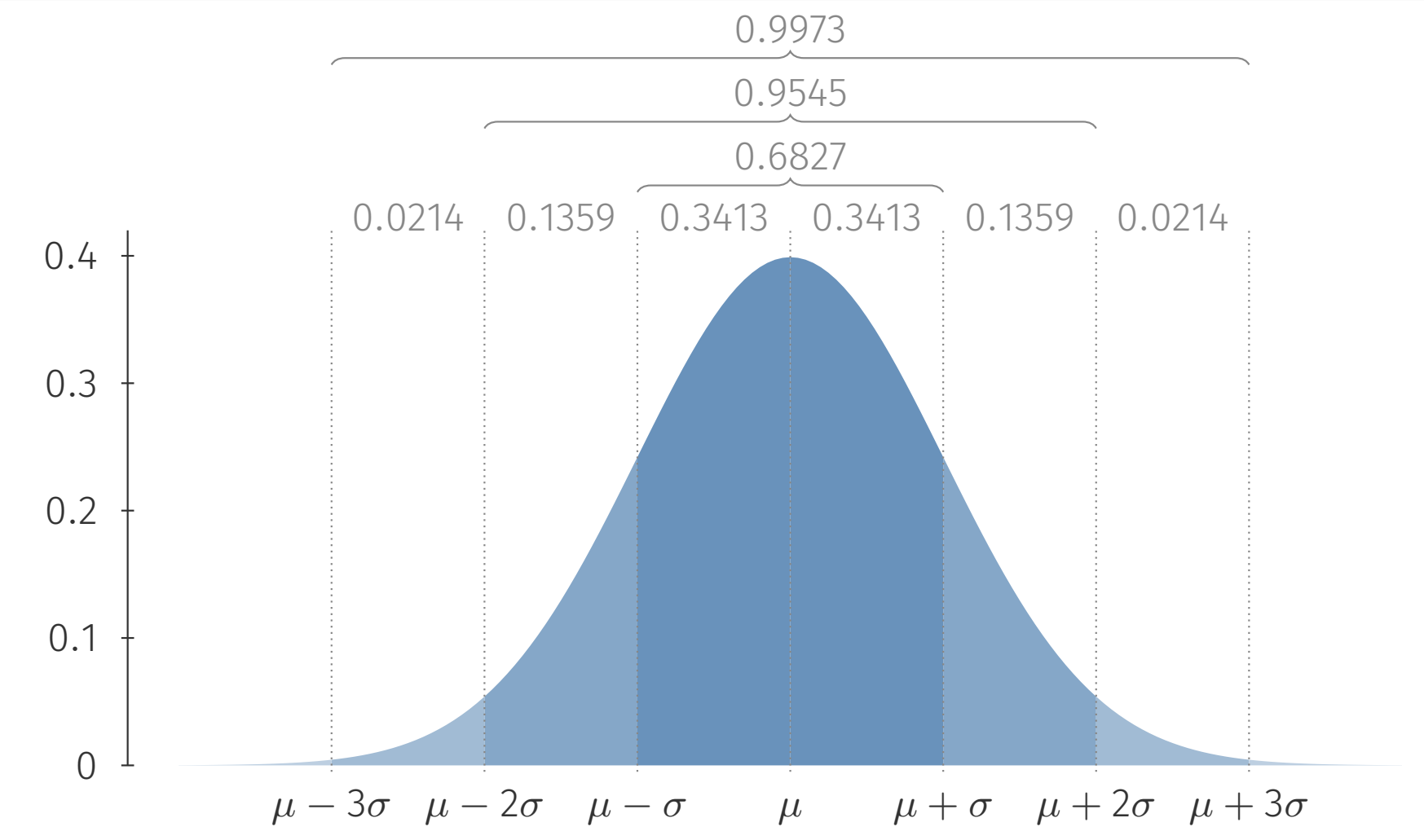
$$f_{\mathcal{D}}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-z^2}{2}}$$

Hence, the cumulative Gaussian distribution can be used to determine the area of the tail that is more extreme than z
When the number of available data points n is limited, Student t-distribution with n degrees of freedom is used instead

Points are typically declared outliers if the absolute value of their z-number is greater than 3
i.e. if they deviate more than 3 times the standard deviation from the mean

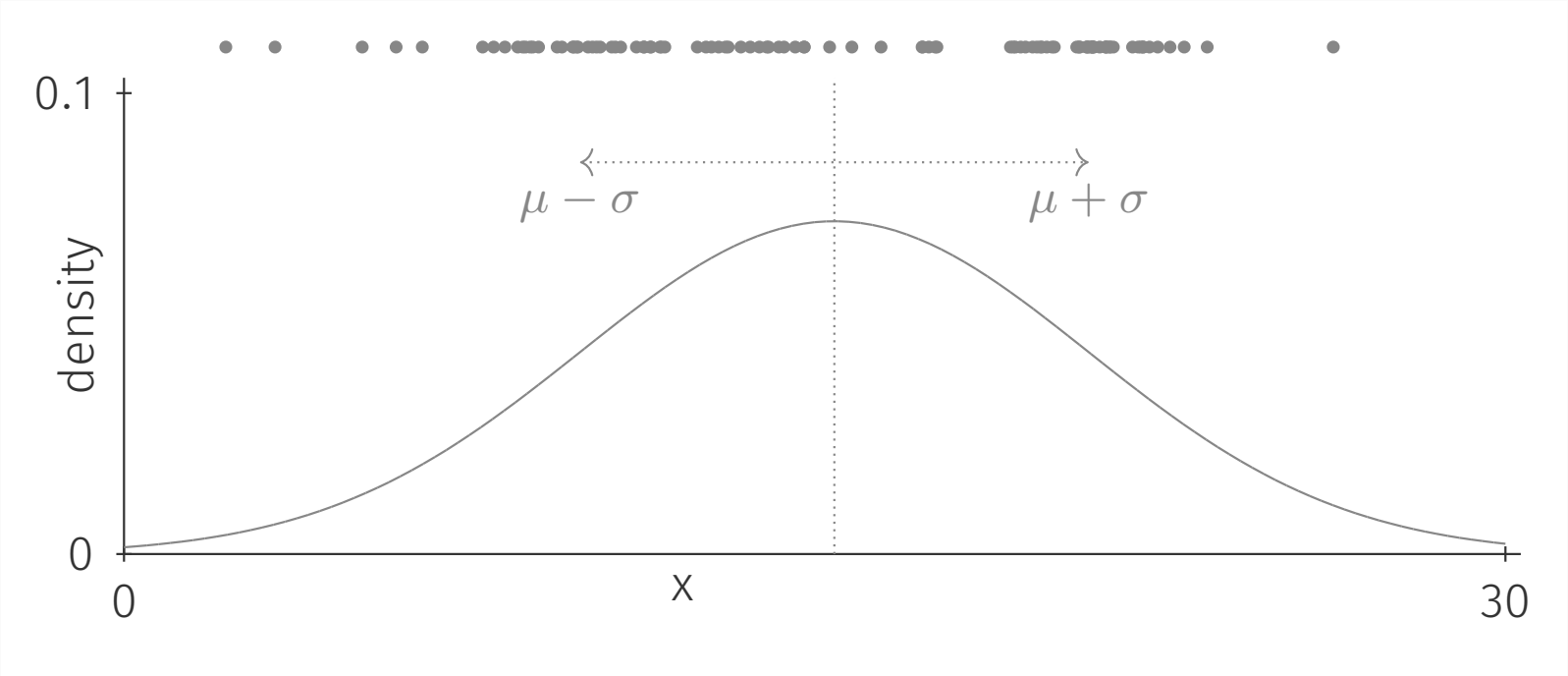
Univariate extreme values

...more than 3 times the standard deviation from the mean



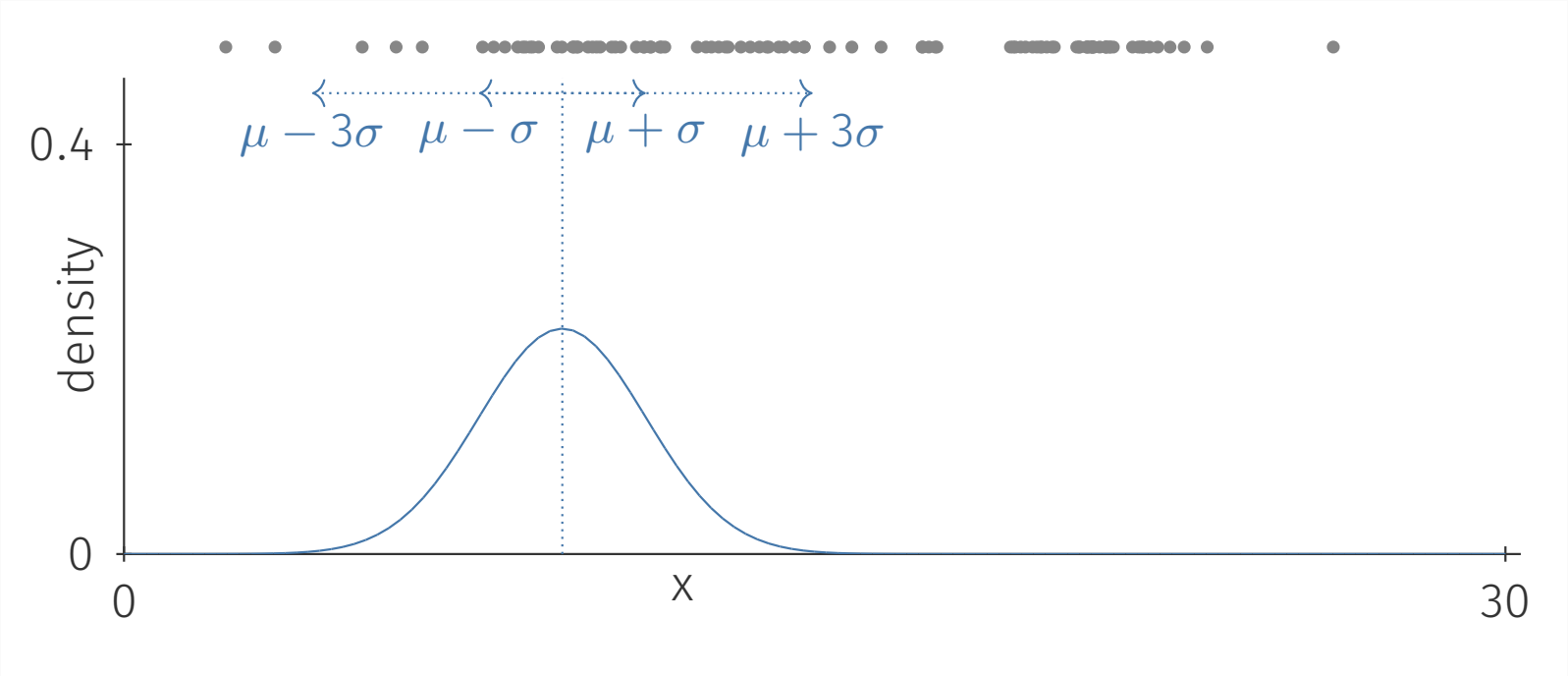
Univariate extreme values

Normal distribution estimated from all data points



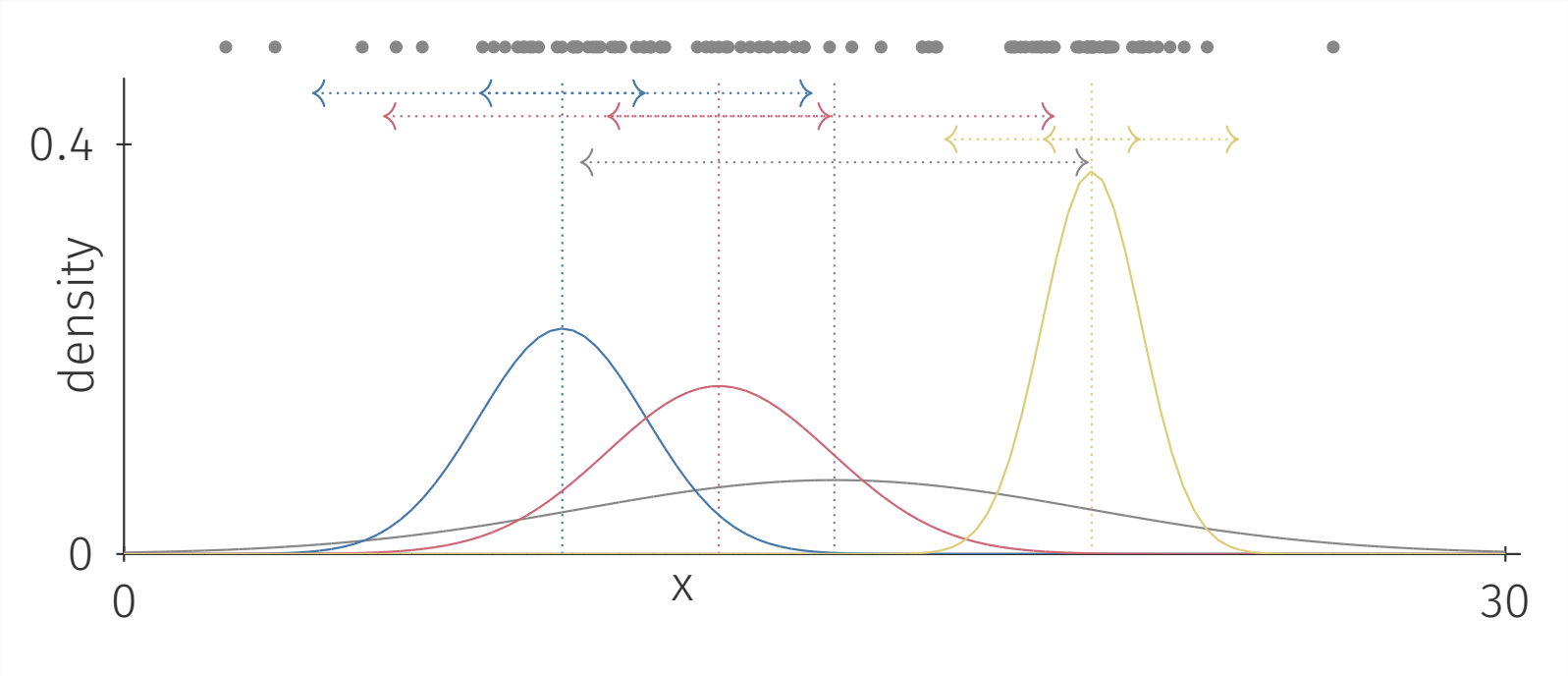
Univariate extreme values

Normal distribution estimated from cluster of data points



Univariate extreme values

Normal distributions estimated from cluster vs. all data points



Multivariate extreme values

The same ideas can be extended to multidimensional data, i.e. m-dimensional data points

Assuming a multivariate Gaussian distribution, the parameters are estimated as the mean $\boldsymbol{\mu}$ and $m \times m$ covariance matrix Σ over all data points in $\mathcal{D}$

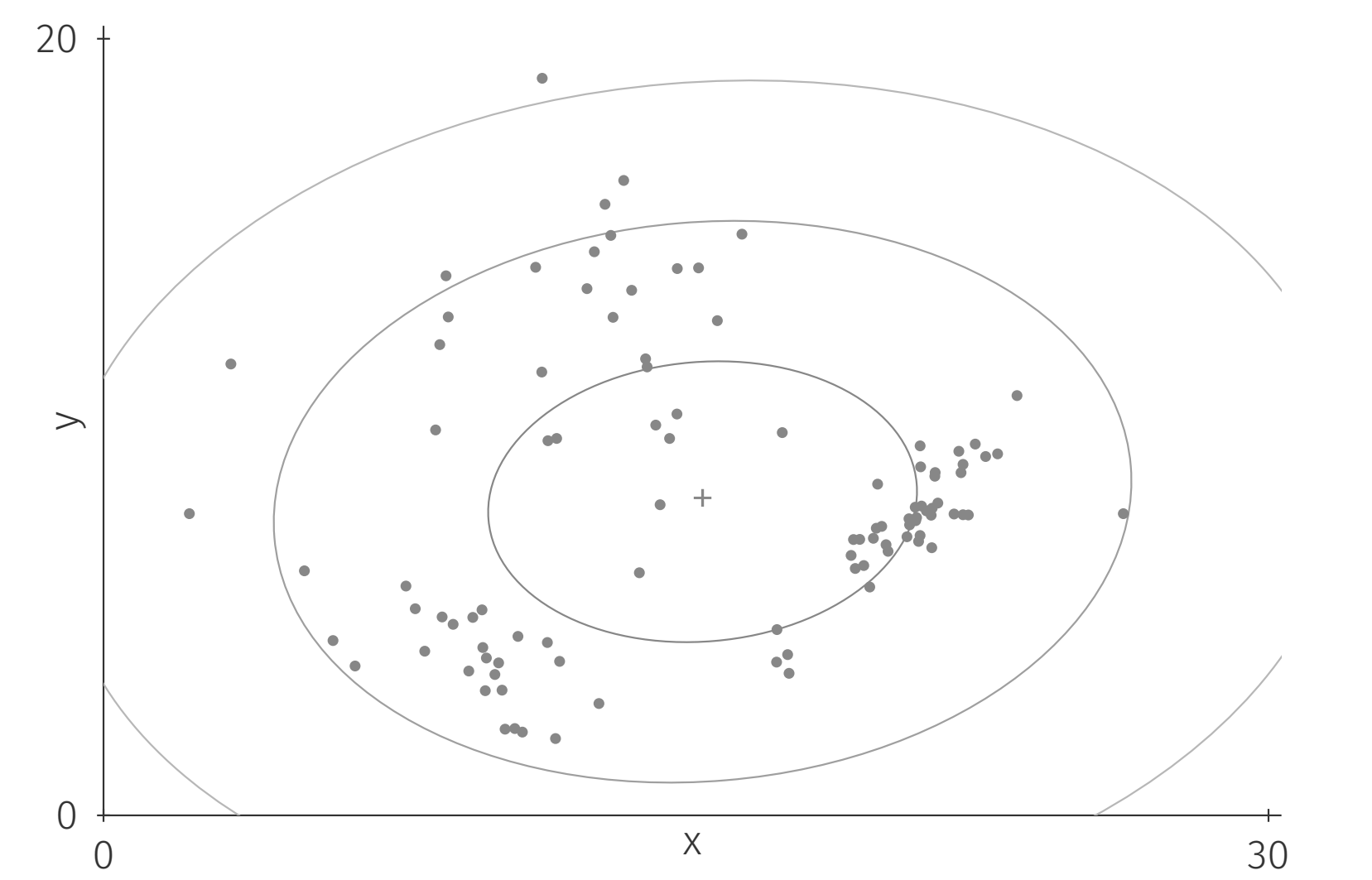
The probability density function of the Gaussian distribution is

$$f_{\mathcal{D}}(\boldsymbol{x}) = \frac{1}{\sqrt{\det(\Sigma) \cdot (2\pi)^m}} e^{-(\boldsymbol{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{\mu})/2}$$

where $\det(\Sigma)$ is the determinant of the covariance matrix

Multivariate extreme values

Probability density function estimated from all data points



Mahalanobis distance

The **Mahalanobis distance** from data point $\boldsymbol{x}$ to a distribution with mean $\boldsymbol{\mu}$ and covariance Σ is

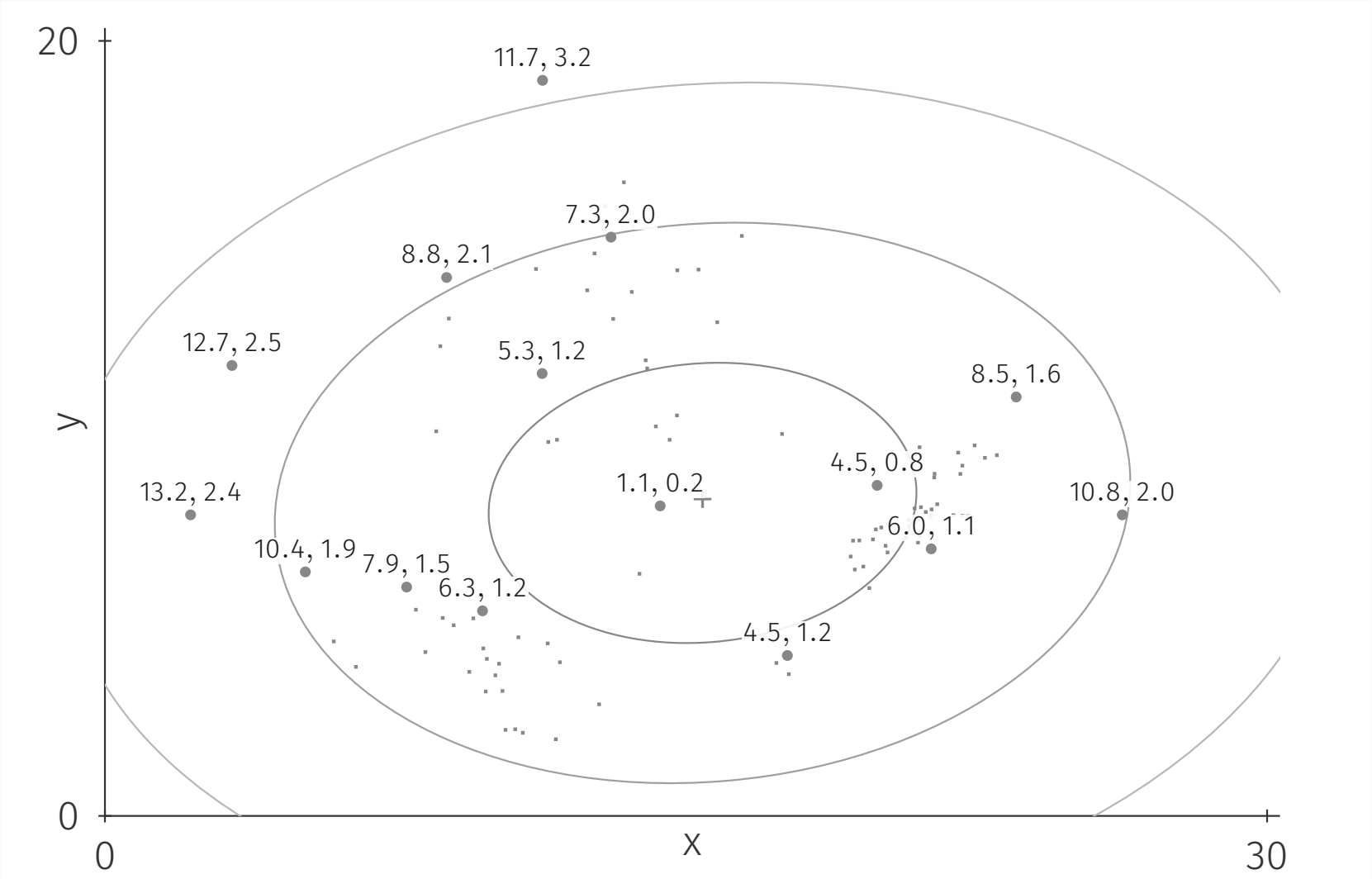
$$D_{\Sigma}(x, \boldsymbol{\mu}) = \sqrt{(x - \boldsymbol{\mu})^T \Sigma^{-1} (x - \boldsymbol{\mu})}$$

Can be seen as a multidimensional extension of the z-number, measuring the number of standard deviations by which the data point differs from the mean of the distribution

Computing the Mahalanobis distance is equivalent to computing the Euclidean distance after rotating the data to the principal directions and dividing each of the transformed coordinate by the corresponding standard deviation

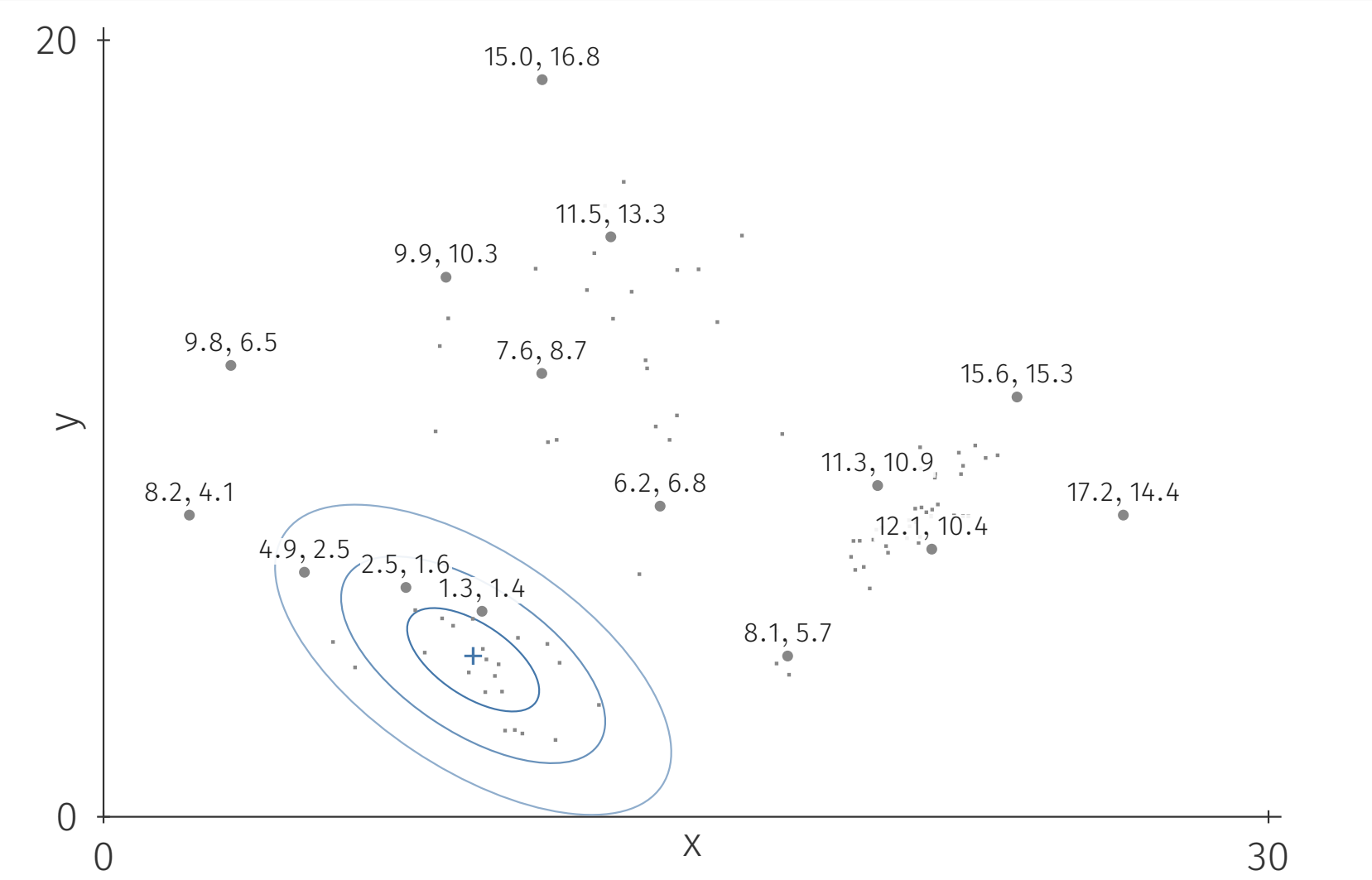
Mahalanobis distance

Comparing $\|x - \mu\|$ and $D_\Sigma(x, \mu)$



Mahalanobis distance

Comparing $\|x - \mu\|$ and $D_\Sigma(x, \mu)$



Multivariate extreme values

The probability density function can be written in terms of the Mahalanobis distance

$$f_{\mathcal{D}}(\boldsymbol{x}) = \frac{1}{\sqrt{\det(\Sigma) \cdot (2\pi)^m}} e^{-(D_\Sigma(\boldsymbol{x},\boldsymbol{\mu}))^2/2}$$

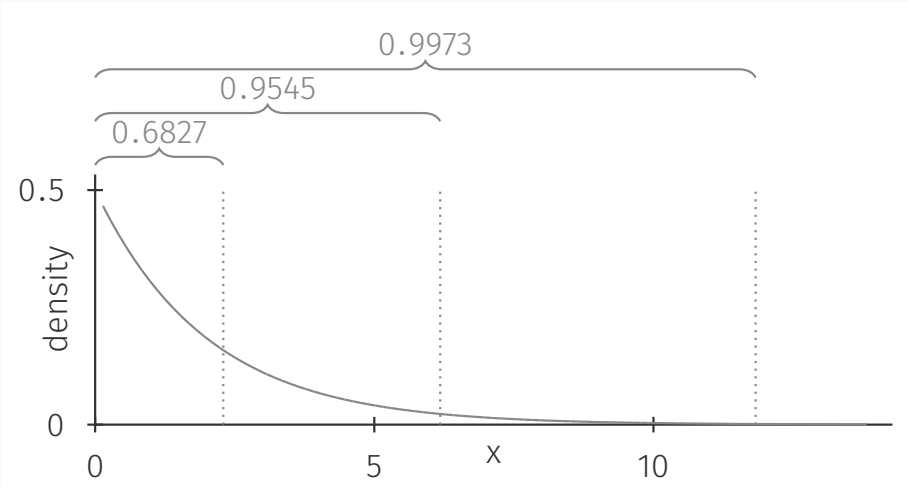
Each of the independent component of the Mahalanobis distance can be modeled as a one-dimensional standard normal distribution $\mathcal{N}(0, 1)$

The sum of squares of m such variables follows a χ^2 distribution with m degrees of freedom

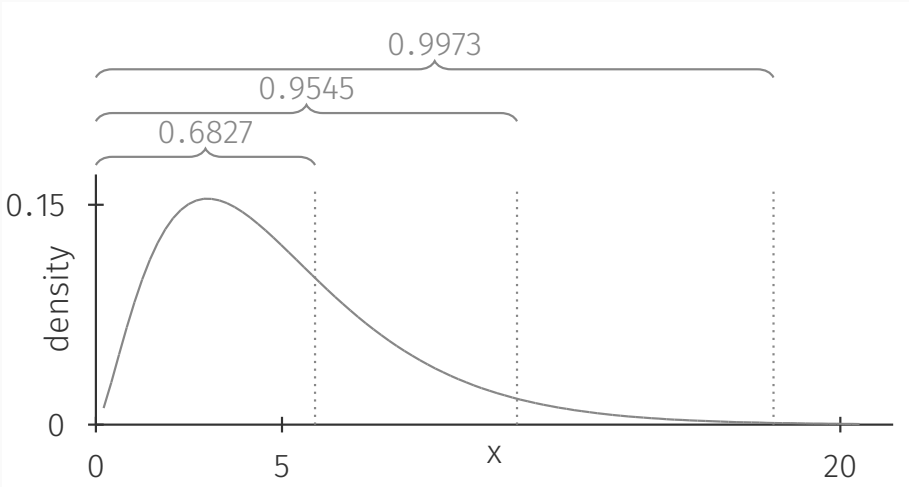
The cumulative probability of the region of the χ^2 distribution with m degrees of freedom for which the value is greater than $D_\Sigma(\boldsymbol{x}, \boldsymbol{\mu})$ can be reported as the extreme value probability of $\boldsymbol{x}$

χ^2 distribution

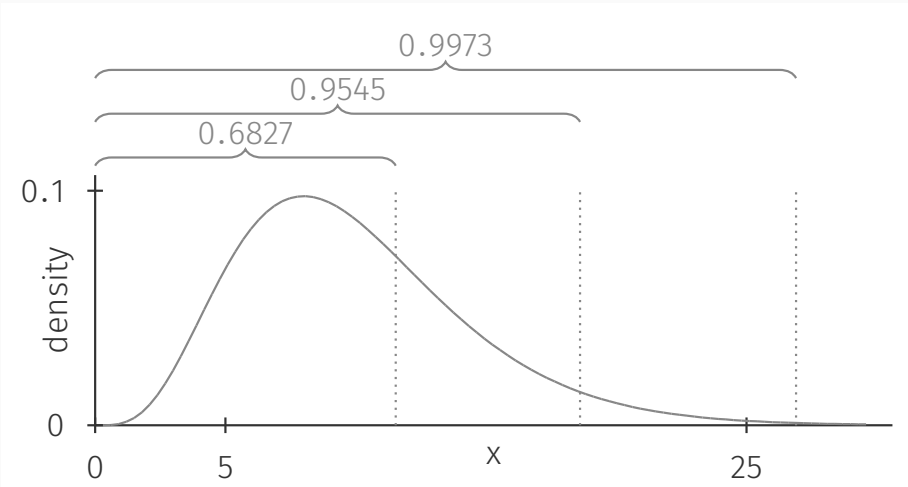
2 degrees of freedom



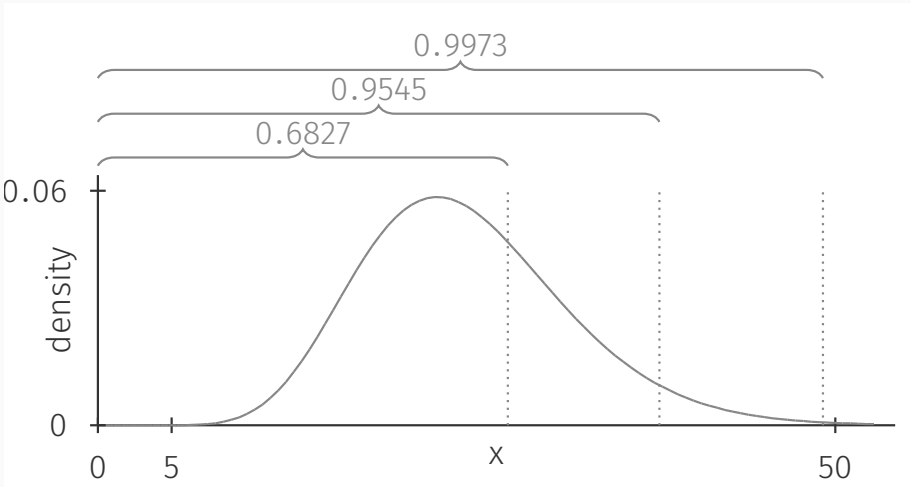
5 degrees of freedom



10 degrees of freedom



25 degrees of freedom



Multivariate extreme values

! **Curse of dimensionality**

As dimensionality increases the Mahalanobis distances of all points become more similar

! **Robustness**

The estimation of parameters is sensitive to outliers

Clustering models

Assumption: clustering aims at finding groups of similar points, whereas outliers are not similar to the rest of the data

Cluster the data and report as outliers points that have a large raw distance to the closest cluster centroid

The raw distance is not well suited if the clusters are elongated and have varying densities
Use the Mahalanobis distance with respect to the clusters, i.e. *local* Mahalanobis distances

Clustering models

Assumption: clustering aims at finding groups of similar points, whereas outliers are not similar to the rest of the data

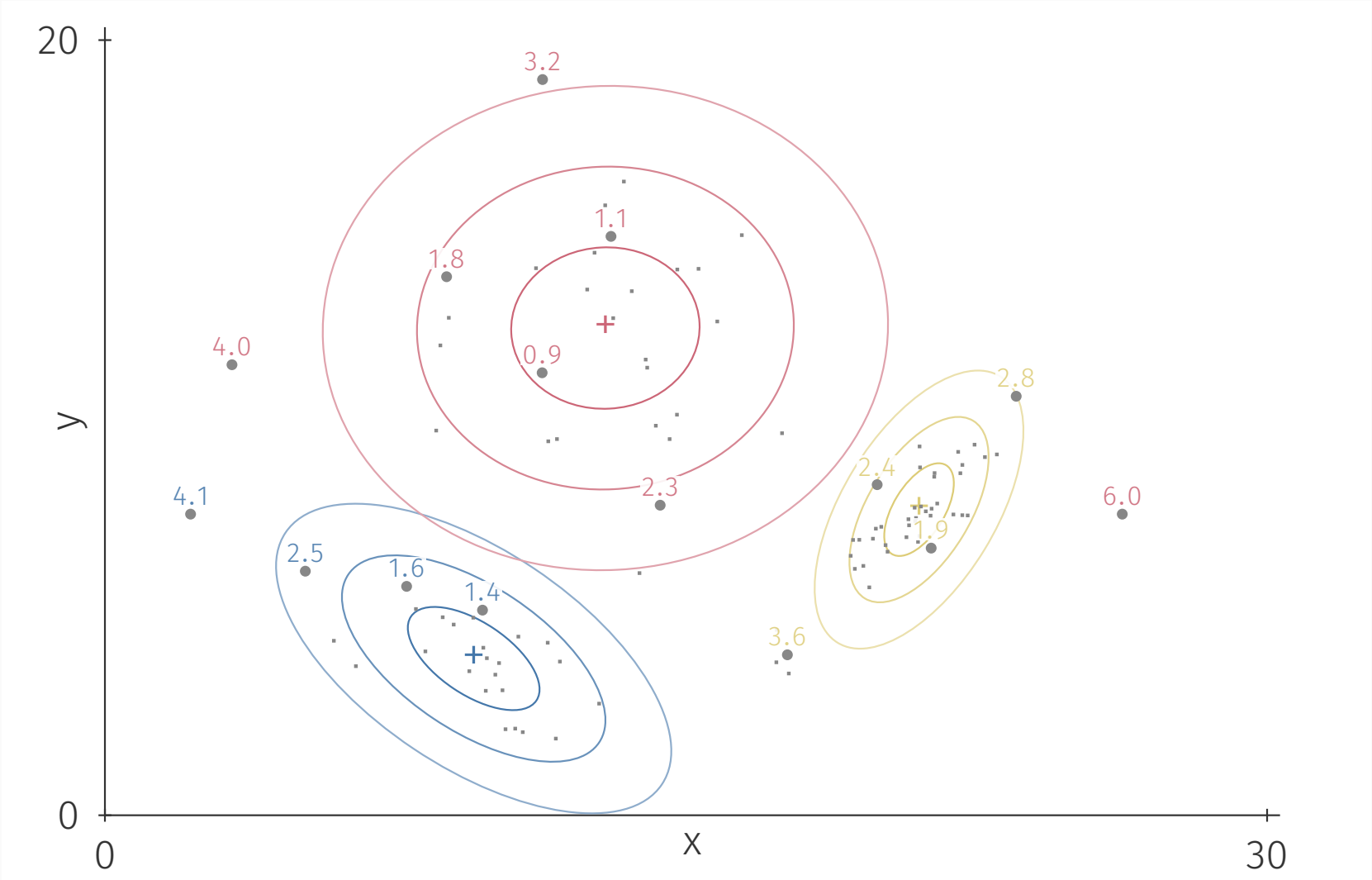
Assuming that k clusters have been detected
The Mahalanobis distance from point $\boldsymbol{x}$ to the j^{th} cluster, having mean μ_j and covariance matrix Σ_j, is

$$D_{\Sigma_j}(\boldsymbol{x}, \boldsymbol{\mu}_j) = (\boldsymbol{x} - \boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_j)$$

Report $\mathbf{min}_{j=1,\ldots,k}\, D_{\Sigma_j}(\boldsymbol{x}, \boldsymbol{\mu}_j)$ as outlier score of point $\boldsymbol{x}$

Clustering models

$\min\left(D_{\Sigma}(x,\mu), D_{\Sigma}(x,\mu), D_{\Sigma}(x,\mu)\right)$ as outlier score



Clustering models

In the case of EM clustering with Gaussian mixture model, each cluster C_i is modelled as a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$ with probability density function f_i and associated to a prior probability α_i

The probability that data point $\boldsymbol{x}$ is generated by the model is

$$\sum_i \alpha_i f_i(\boldsymbol{x})$$

Points that are highly unlikely to be generated by the model, i.e. have very low fit, are reported as outliers

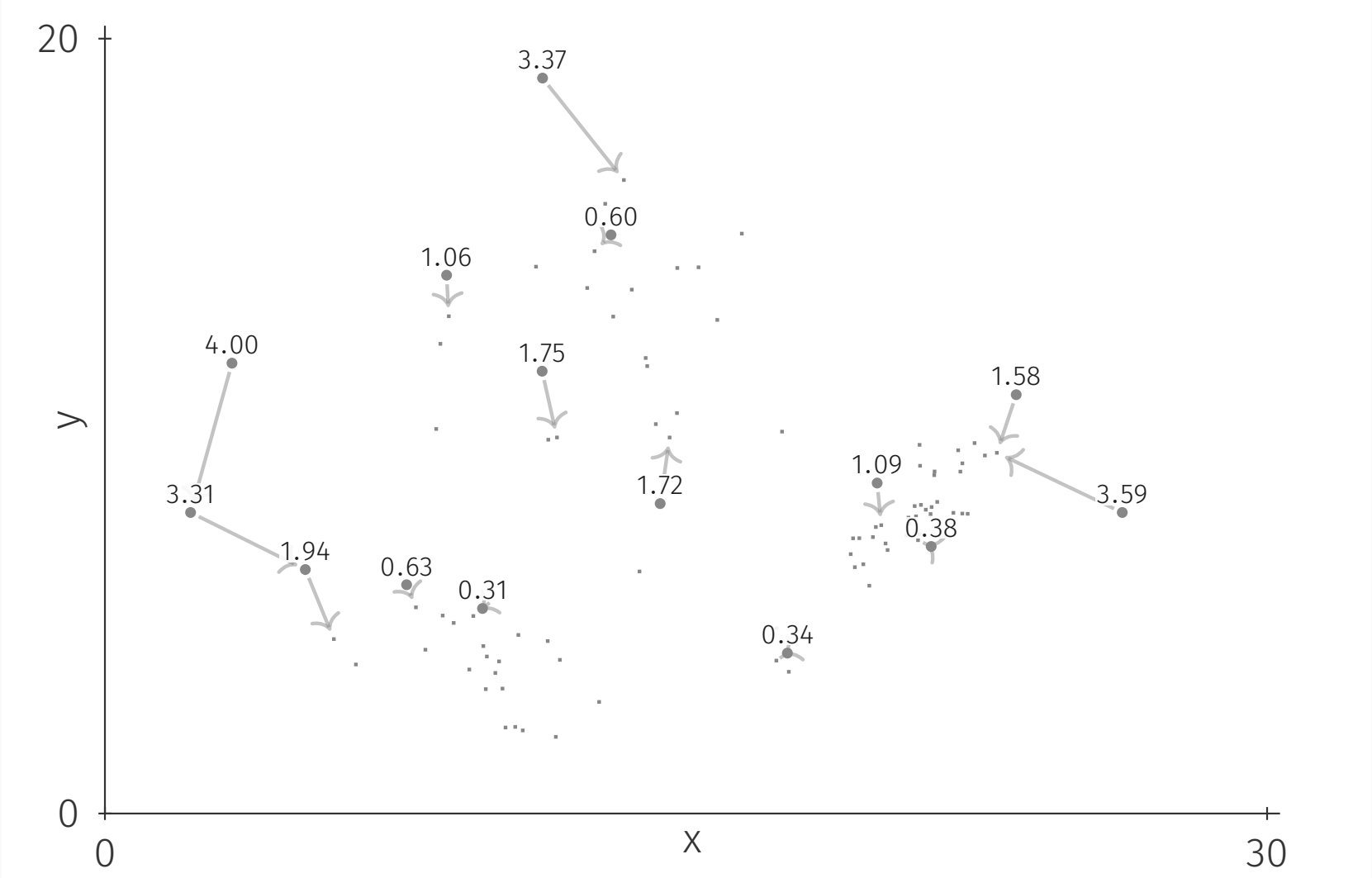
Distance-based models: k-NN distance

Assumption: outliers are not similar to the rest of the data, i.e. they are far apart from their neighbors

Report the distance from a point to its k-nearest neighbor as the outlier score

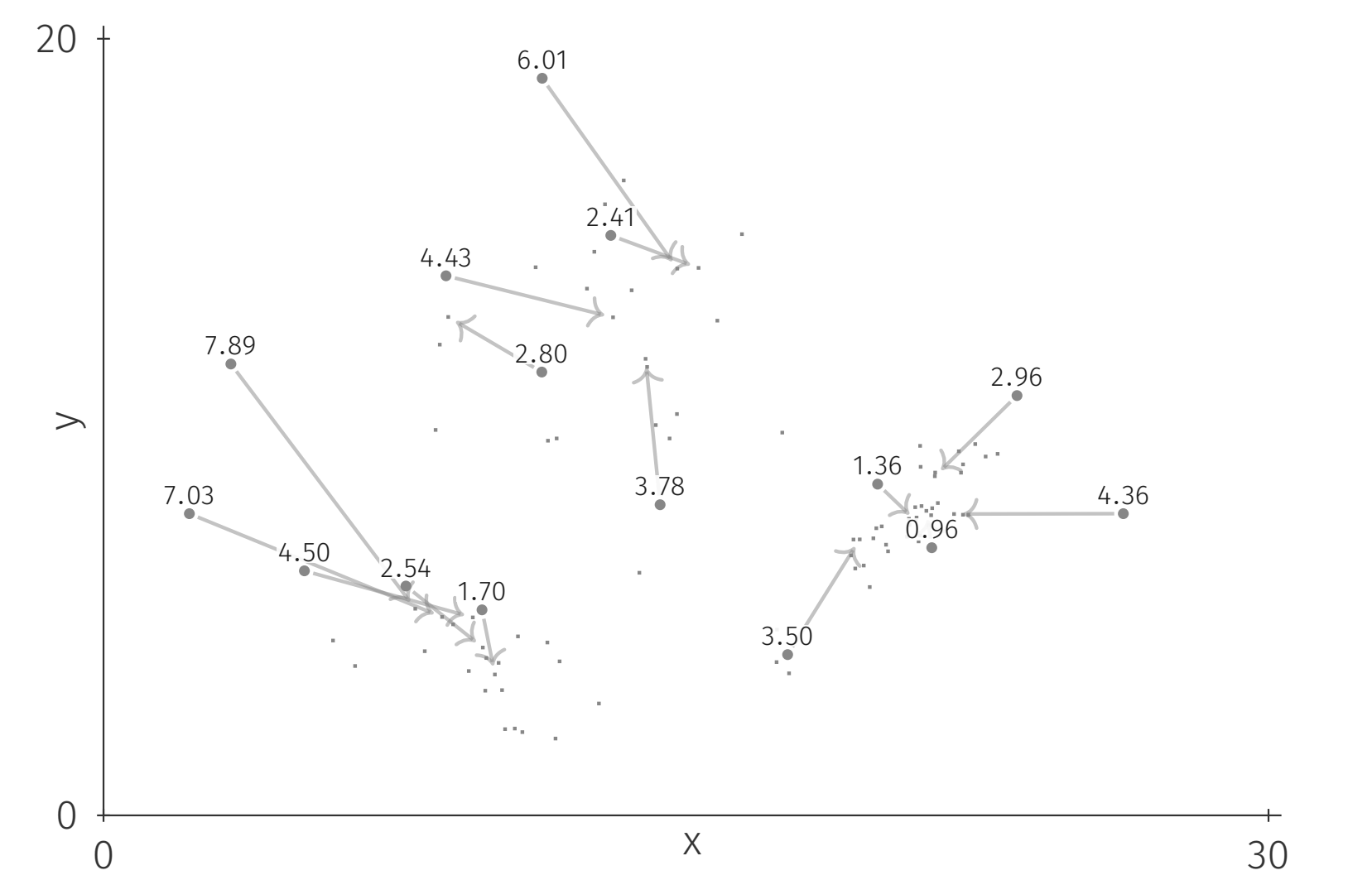
Distance-based models: k-NN distance

Distance to k-nearest neighbor, $k = 1$



Distance-based models: k-NN distance

Distance to k-nearest neighbor, $k = 9$



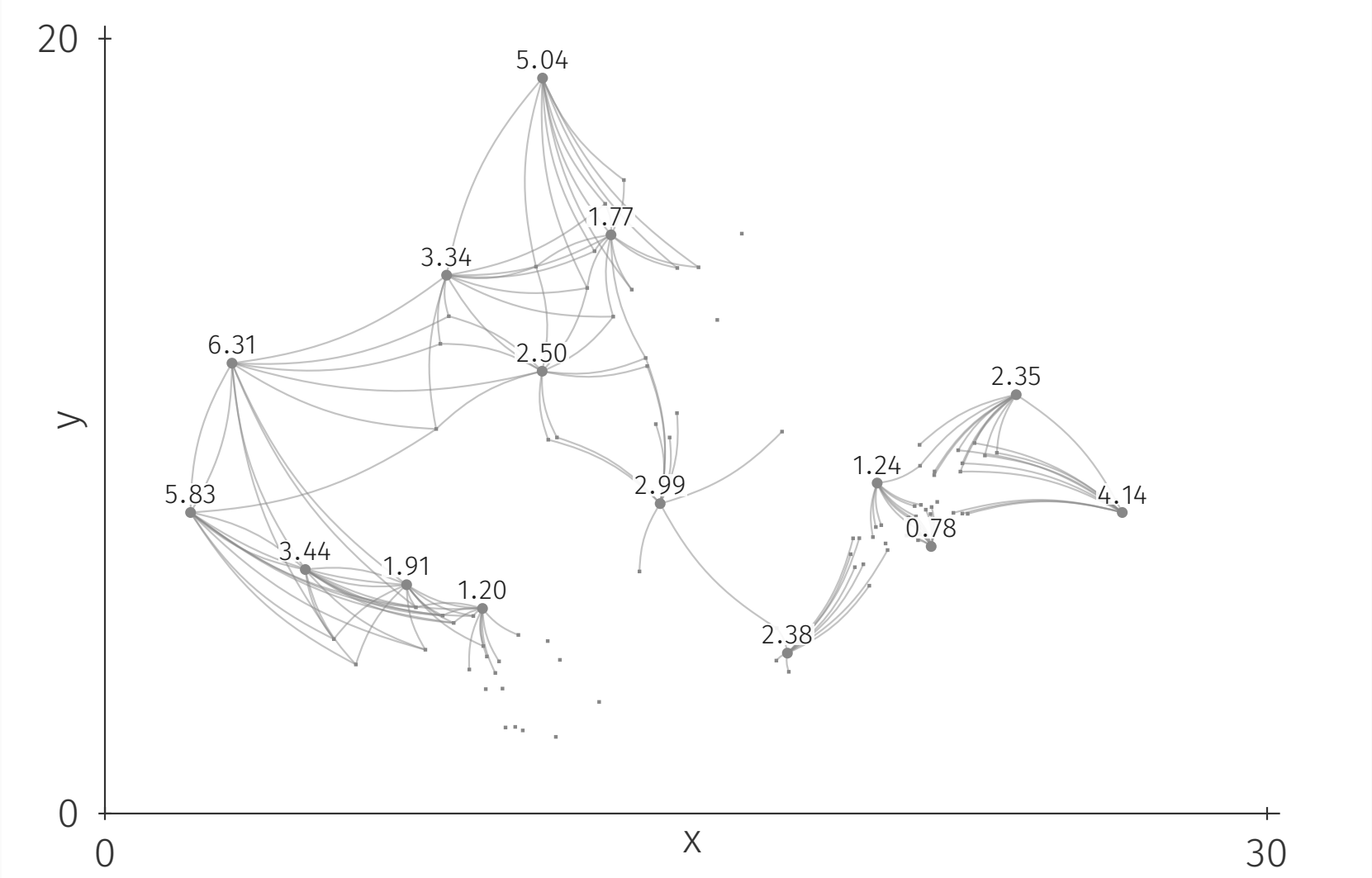
Distance-based models: k-NN distance

Assumption: outliers are not similar to the rest of the data, i.e. they are far apart from their neighbors

Report the *average* distance from a point to its k-nearest neighbors as the outlier score

Distance-based models: k-NN distance

Average distance to k-nearest neighbors, $k = 10$



Distance-based models: k-NN distance

Assumption: outliers are not similar to the rest of the data, i.e. they are far apart from their neighbors

Construct the k-nearest neighbor graph for the dataset, where each data point $\boldsymbol{x}$ is represented by a vertex v_x and there is a directed edge from v_x to $v_{x'}$ if $\boldsymbol{x'}$ is among the k-nearest neighbors of $\boldsymbol{x}$
Report point $\boldsymbol{x}$ as outlier if the in-degree of v_x is less than user-defined threshold τ

Distance-based models: k-NN distance

$O \leftarrow \langle\rangle$ *r top outliers*
$\lambda \leftarrow 0$ *k-NN distance of top r outlier*
for each $x \in \mathcal{D}$ **do**
    $N \leftarrow \langle\rangle$ *k nearest neighbors of* x
    $\delta \leftarrow \infty$ *distance to k nearest neighbor of* x
    for each $x' \in \mathcal{D} \setminus \{x\}$ **do**
        if $\mathrm{d}(x, x') < \delta$ **then**
            insert x' into N and update δ accordingly
    if $\delta > \lambda$ **then**
        insert x into O and update λ accordingly
return O

Distance-based models: k-NN distance

Distance-based models have a finer granularity than clustering models, but it comes at the cost of higher computational complexity

Computing the k-nearest neighbor distance requires $O(n)$ time for each data point when a sequential scan is used,
i.e. $O(n^2)$ time for the entire dataset, which is not scalable

Early termination

In most cases the scores of all data points are not required, only the top r outliers

The scan for a data point can be terminated if the *upper bound estimate* on its k-nearest neighbor distance falls below the current r^{th} top outlier score

Distance-based models: k-NN distance

$O \leftarrow \langle\rangle$ *r top outliers*
$\lambda \leftarrow 0$ *k-NN distance of top r outlier*
for each $x \in \mathcal{D}$ **do**
    $N \leftarrow \langle\rangle$ *k nearest neighbors of* x
    $\delta \leftarrow \infty$ *distance to k nearest neighbor of* x
    for each $x' \in \mathcal{D} \setminus \{x\}$ **do**
        if $d(x, x') < \delta$ **then**
            insert x' into N and update δ accordingly
            if $\delta < \lambda$ **then** drop x *early termination*
    if $\delta > \lambda$ **then**
        insert x into O and update λ accordingly
return O

Distance-based models: k-NN distance

Distance-based models have a finer granularity than clustering models, but it comes at the cost of higher computational complexity

Computing the k-nearest neighbor distance requires $O(n)$ time for each data point when a sequential scan is used,
i.e. $O(n^2)$ time for the entire dataset, which is not scalable

Early termination

Two steps method with sample

Compute distances exhaustively for a small sample of points
Compute distances for other points that are potential outliers

Distance-based models: k-NN distance

Compute distances exhaustively for a small sample of points

$S \leftarrow \{s$ data points sampled randomly from $\mathcal{D}, s \ll n\}$

compute $\mathsf{d}(x, x')$ for $s \cdot n$ pairs $(x, x') \in S \times \mathcal{D}$

$\Delta_{S,k}(x)$ denotes the distance from x to its k nearest neighbor in S

$O \leftarrow \langle r$ top outliers from $S\rangle$ *i.e. r points $x \in S$ with highest $\Delta_{S,k}(x)$*

$\lambda \leftarrow k$-NN distance of top r outlier from S *i.e. $\Delta_{S,k}(O[r])$*

Compute distances for other points that are potential outliers

for each $x \in \mathcal{D} \setminus S$ **do**

  if $\Delta_{S,k}(x) < \lambda$ **then** drop x *early termination*

  $N_k \leftarrow k$ nearest neighbors of x in S; $\delta \leftarrow \Delta_{S,k}(x)$

  for each $x' \in \mathcal{D} \setminus S$ **do**

    if $\mathsf{d}(x, x') < \delta$ **then**

      insert x' into N_k and update δ accordingly

      if $\delta < \lambda$ **then** drop x *early termination*

  if $\delta > \lambda$ **then** insert x into O and update λ accordingly

return O

Distance-based models: k-NN distance

Compute distances exhaustively for a small sample of points

$S \leftarrow \{s$ data points sampled randomly from $\mathcal{D}, s \ll n\}$

compute $\mathsf{d}(x, x')$ for $s \cdot n$ pairs $(x, x') \in S \times \mathcal{D}$

$\Delta_{S,k}(x)$ denotes the distance from x to its k nearest neighbor in S

$O \leftarrow \langle r$ top outliers from $S\rangle$ *i.e. r points $x \in S$ with highest $\Delta_{S,k}(x)$*

$\lambda \leftarrow k$-NN distance of top r outlier from S *i.e. $\Delta_{S,k}(O[r])$*

Compute distances for other points that are potential outliers

for each $x \in \mathcal{D} \setminus S$ ordered by decreasing $\Delta_{S,k}(x)$ **do**

  if $\Delta_{S,k}(x) < \lambda$ **then** drop x *early termination*

  $N_k \leftarrow k$ nearest neighbors of x in S; $\delta \leftarrow \Delta_{S,k}(x)$

  for each $x' \in \mathcal{D} \setminus S$ ordered by increasing $\Delta_{S,k}(x')$ **do**

    if $\mathsf{d}(x, x') < \delta$ **then**

      insert x' into N_k and update δ accordingly

      if $\delta < \lambda$ **then** drop x *early termination*

  if $\delta > \lambda$ **then** insert x into O and update λ accordingly

return O

Distance-based models: k-NN distance

The k-NN distance is sensitive to the neighborhood density
Need for corrections to account for local variations in density

Local outlier factor (LOF)
Normalizes distances with average local density
Sometimes seen as a density-based method
Sometimes as a distance-based method
Both types of methods rely on proximity

Local outlier factor

Let $\Delta_k(x)$ denote the distance from x to its k nearest neighbor
Let $N_k(x)$ denote the points within distance $\Delta_k(x)$ of x

$$N_k(x) = \{x' \in \mathcal{D} \setminus \{x\}, \mathrm{d}(x, x') \leq \Delta_k(x)\}$$

Due to ties, $N_k(x)$ might contain more than k points

The reachability distance of point x with respect to x' is

$$R_k(x, x') = \max(\mathrm{d}(x, x'), \Delta_k(x'))$$

The reachability distance is not symmetric
Intuitively, when x' is in a dense region and the distance between x and x' is large $R_k(x, x')$ equals the true distance, whereas when the distance between x and x' is small $R_k(x, x')$ is smoothed out by the k-NN distance of x'

Local outlier factor

Let $\Delta_k(x)$ denote the distance from x to its k nearest neighbor
Let $N_k(x)$ denote the points within distance $\Delta_k(x)$ of x
The reachability distance of point x with respect to x' is

$$R_k(x, x') = \max(\mathrm{d}(x, x'), \Delta_k(x'))$$

The average reachability distance of point x with respect to its neighborhood is

$$AR_k(x) = \frac{1}{|N_k(x)|} \sum_{x' \in N_k(x)} R_k(x, x')$$

The local outlying factor of point x is

$$LOF_k(x) = \frac{1}{|N_k(x)|} \sum_{x' \in N_k(x)} \frac{AR_k(x)}{AR_k(x')}$$

Local outlier factor

Let $\Delta_k(x)$ denote the distance from x to its k nearest neighbor
Let $N_k(x)$ denote the points within distance $\Delta_k(x)$ of x

$$R_k(x, x') = \max(\mathrm{d}(x, x'), \Delta_k(x'))$$

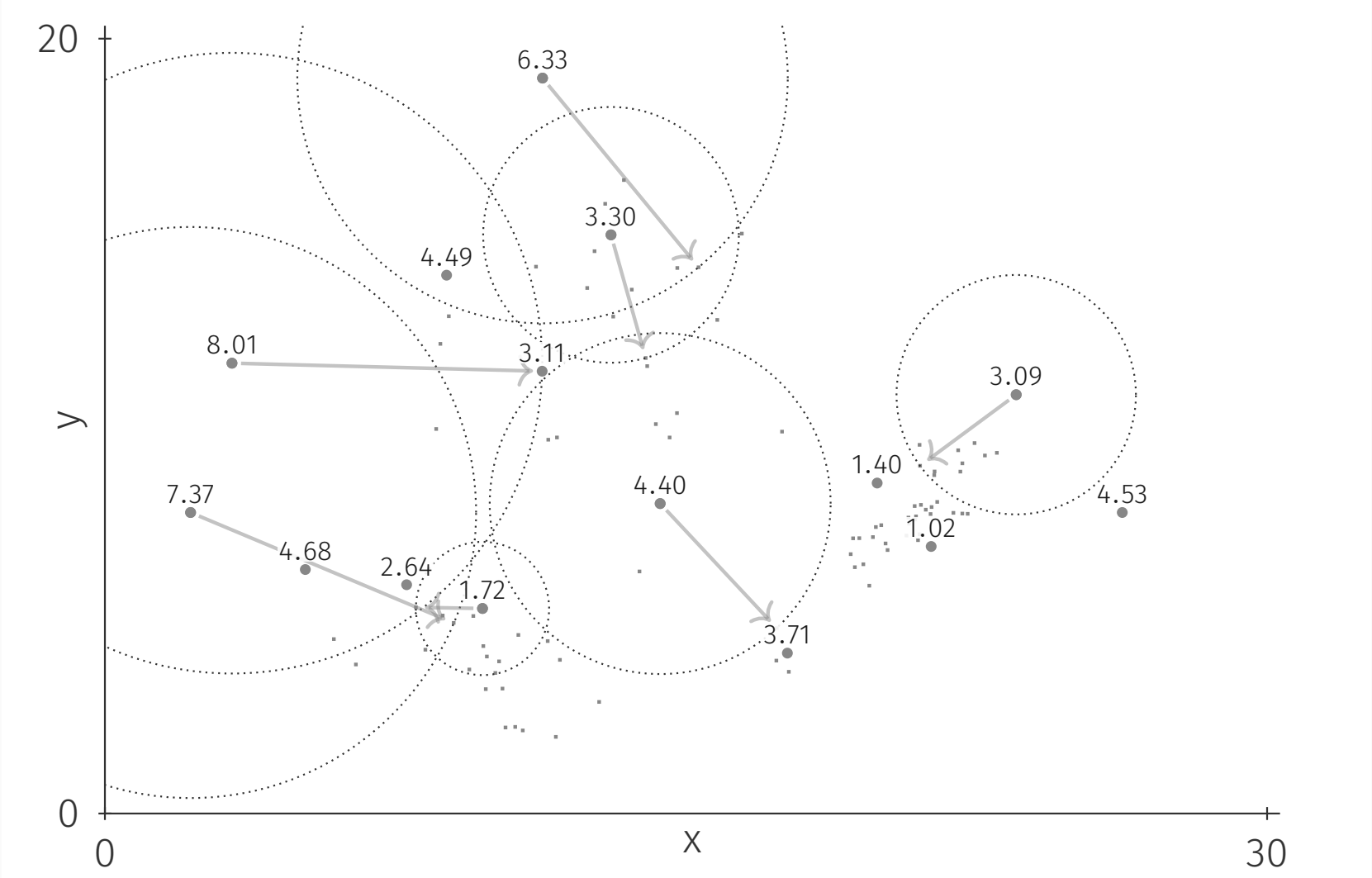
$$AR_k(x) = \frac{1}{|N_k(x)|} \sum_{x' \in N_k(x)} R_k(x, x') \quad LOF_k(x) = \frac{1}{|N_k(x)|} \sum_{x' \in N_k(x)} \frac{AR_k(x)}{AR_k(x')}$$

Typically, LOF_k values for points in a cluster are close to 1 if the points are distributed homogeneously
Points with $LOF_k \gg 1$ are reported as outliers

In practice, determine the best neighborhood size k by taking the maximum LOF_k over a range of values

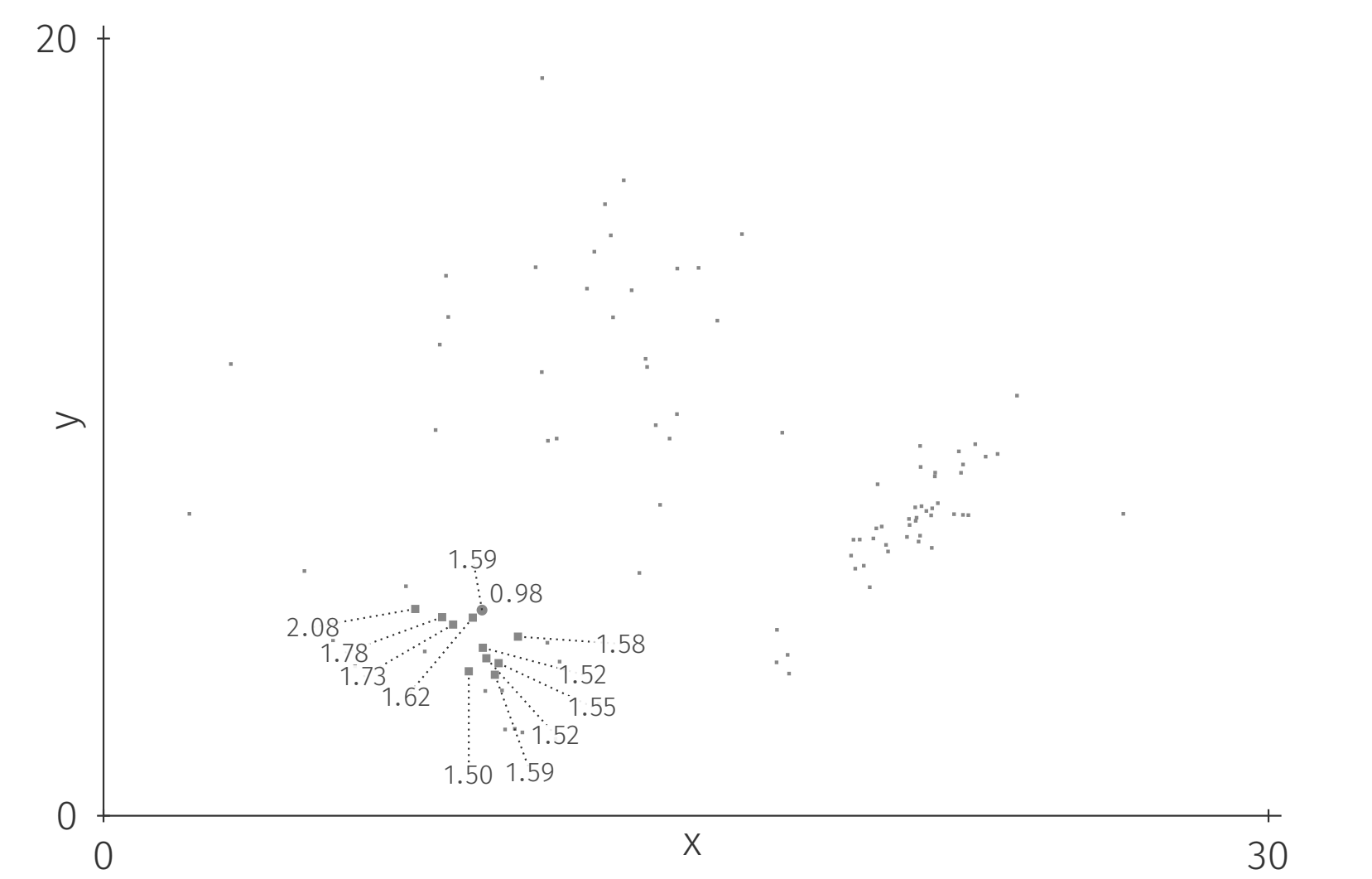
Local outlier factor

Distance to tenth-nearest neighbor Δ_{10}



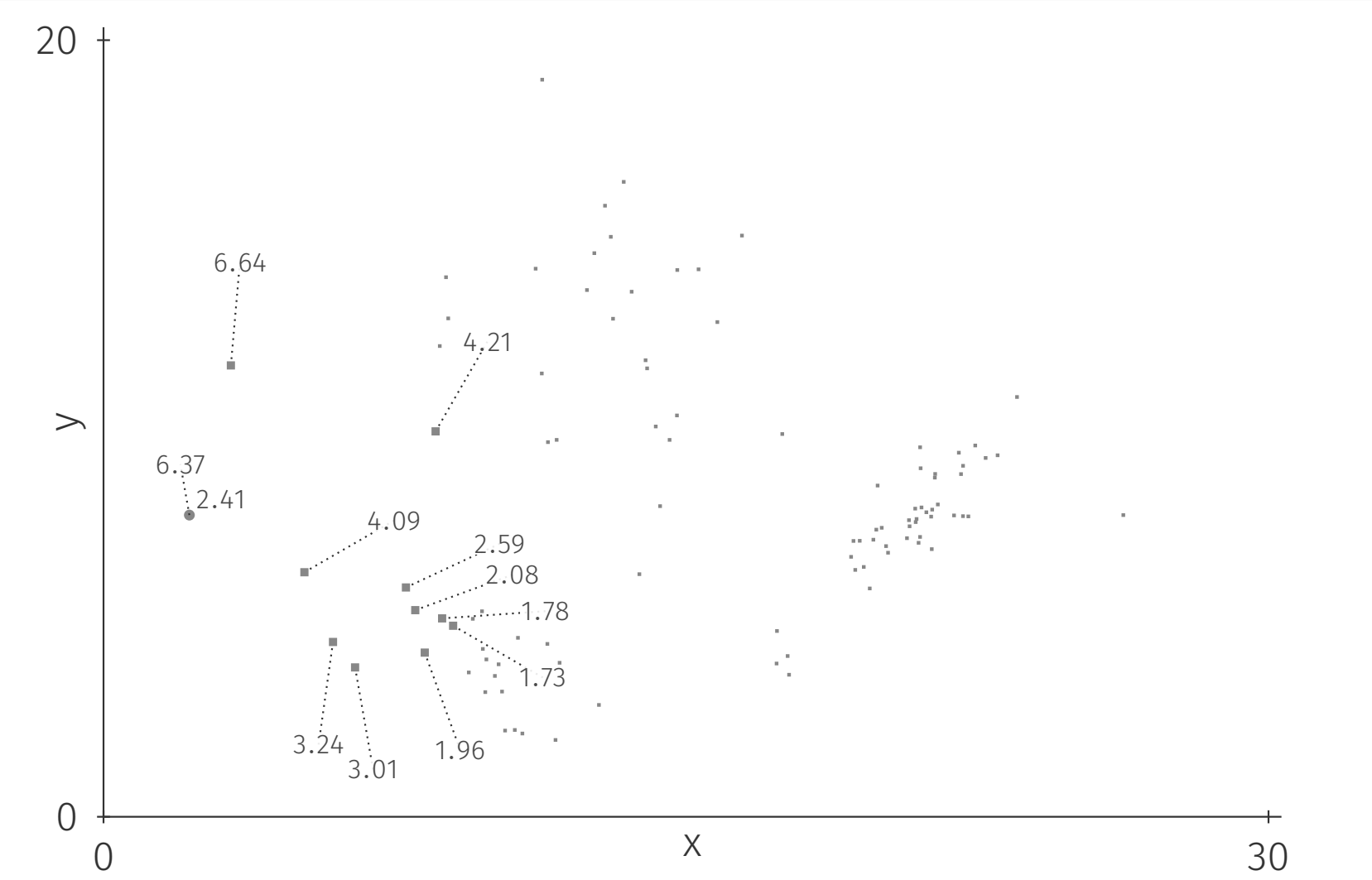
Local outlier factor

$AR_{10}(x)$ vs. $AR_{10}(x'), x' \in N_{10}(x)$, computing $LOF_{10}(x)$



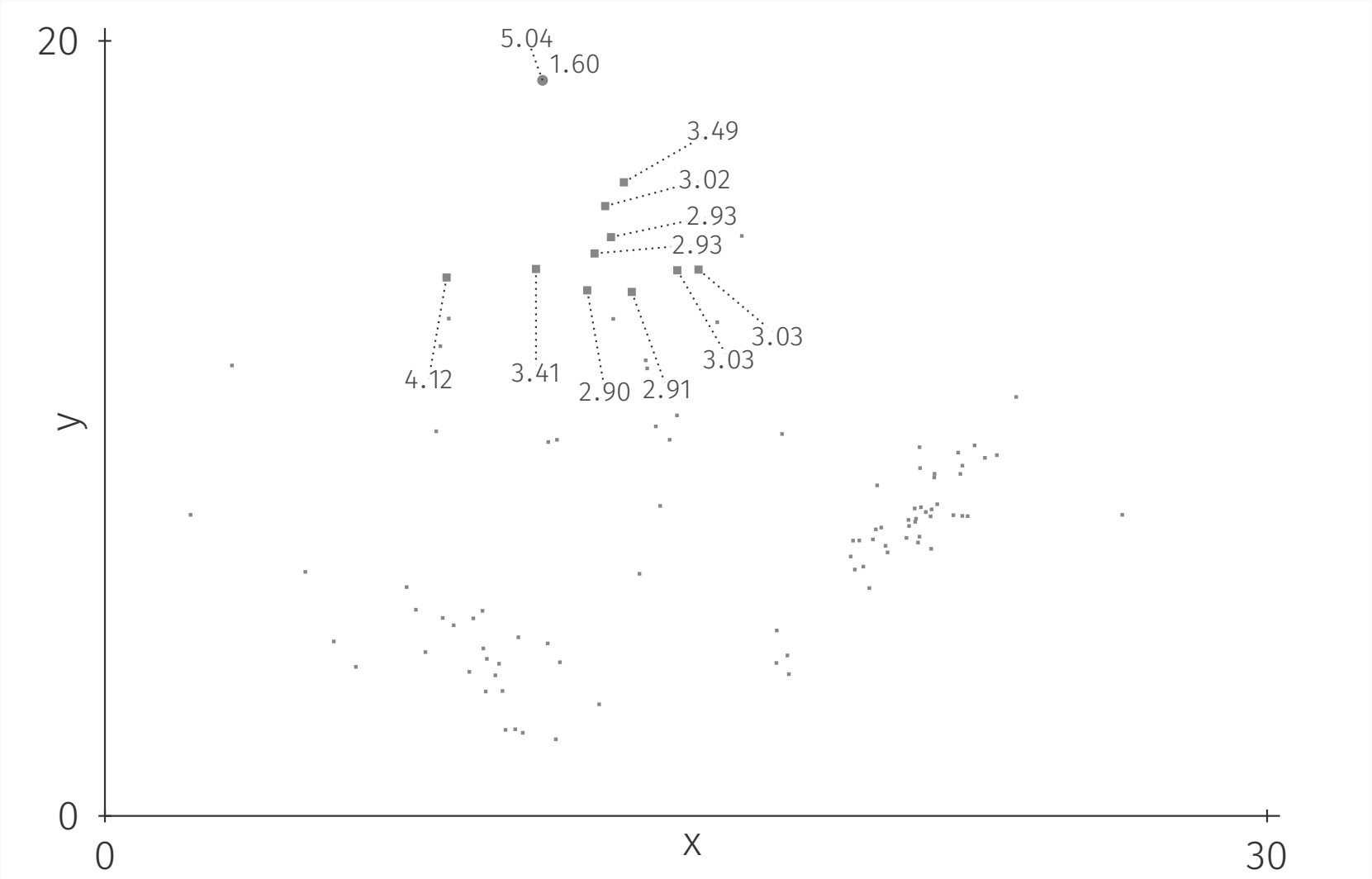
Local outlier factor

$AR_{10}(x)$ vs. $AR_{10}(x'), x' \in N_{10}(x)$, computing $LOF_{10}(x)$



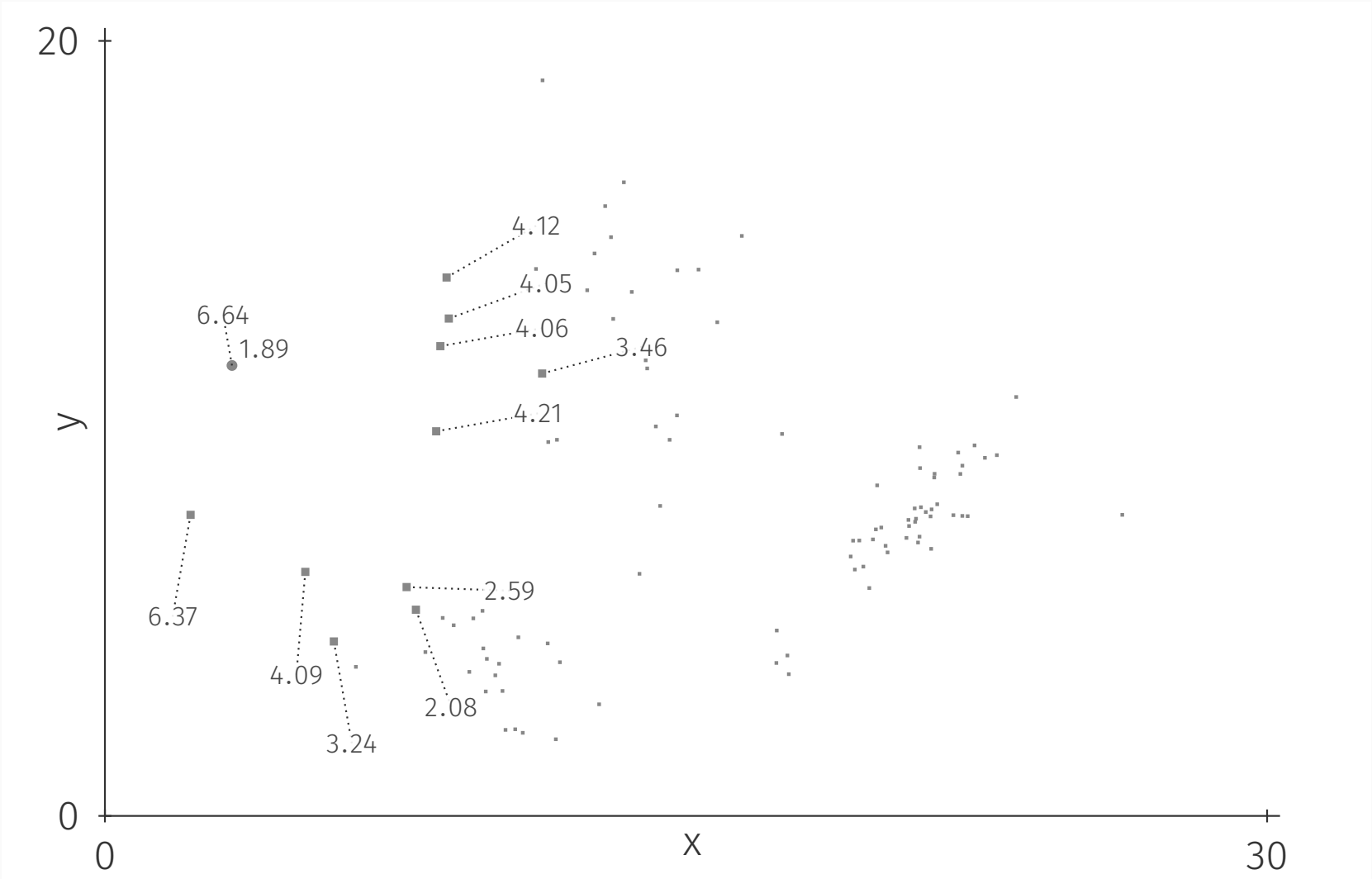
Local outlier factor

$AR_{10}(x)$ vs. $AR_{10}(x'), x' \in N_{10}(x)$, computing $LOF_{10}(x)$



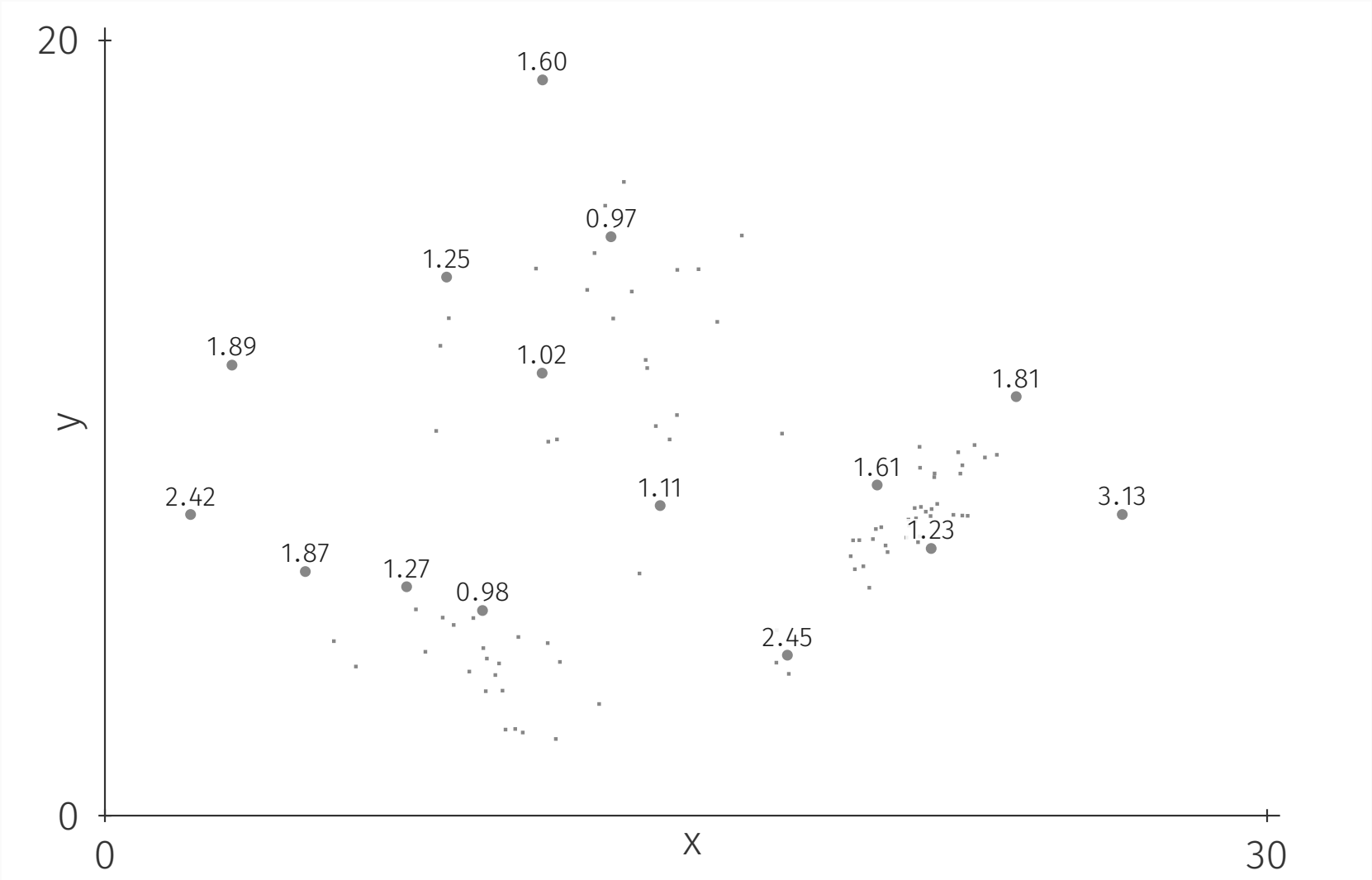
Local outlier factor

$AR_{10}(x)$ vs. $AR_{10}(x'), x' \in N_{10}(x)$, computing $LOF_{10}(x)$



Local outlier factor

Local outlier factors LOF_{10}



Distance-based models: k-NN distance

The k-NN distance is sensitive to the neighborhood shape
Need for corrections to account for local variations in shape

Instance-specific Mahalanobis distance
Compute Mahalanobis distance that accounts for the local covariance structure

Instance-specific Mahalanobis distance

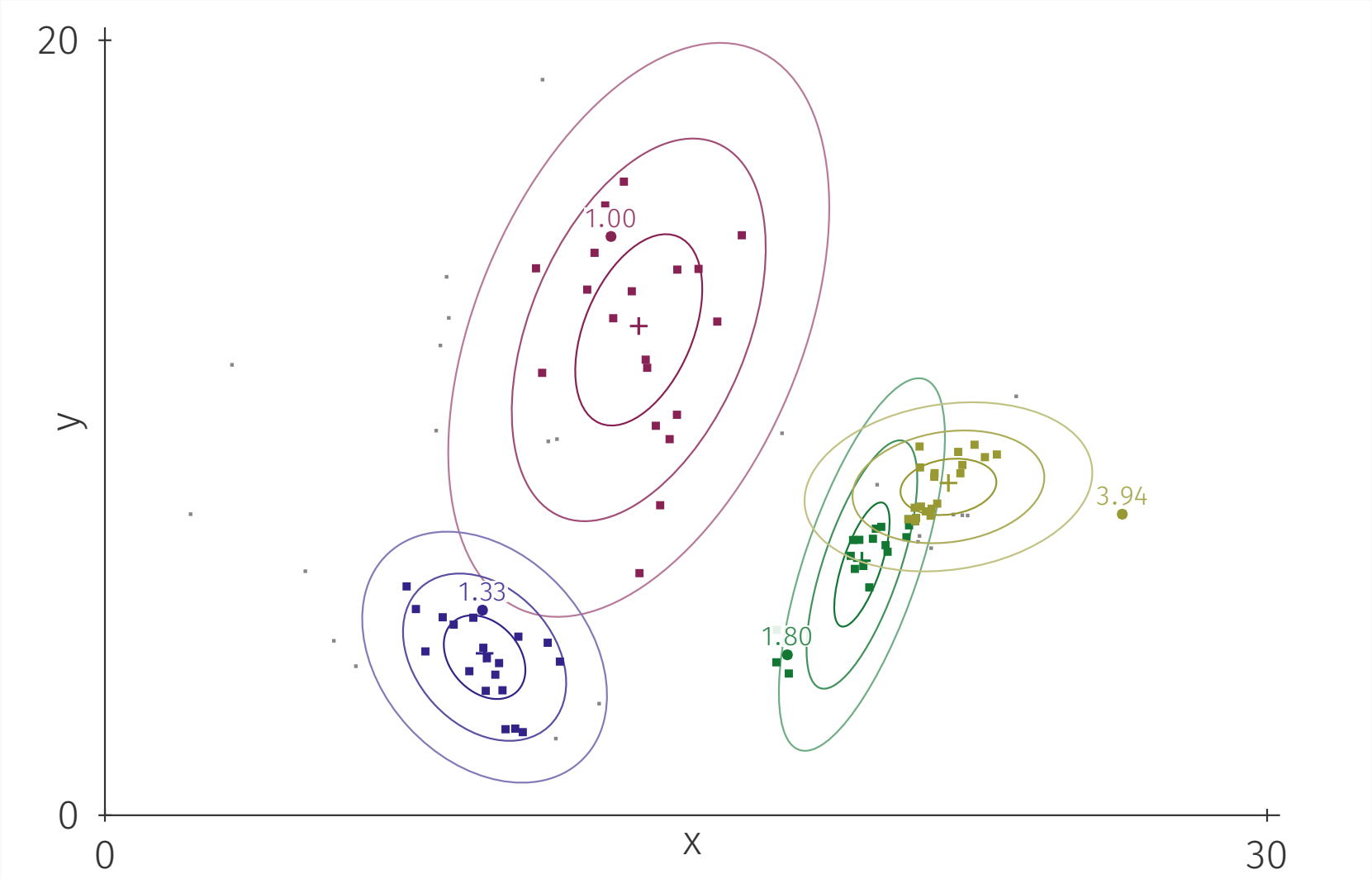
Determine the k-neighborhood of point $\boldsymbol{x}$ following an agglomerative approach

$N \leftarrow \{\boldsymbol{x}\}$
for $i = 1, \ldots, k$ **do**
    $N \leftarrow N \cup \{\arg\min_{\boldsymbol{x}' \in \mathcal{D} \setminus N} \min_{u \in N} \mathrm{d}(\boldsymbol{x}', u)\}$
return N

Use $D_{\Sigma_N}(\boldsymbol{x}, \boldsymbol{\mu}_N)$ as outlier score for point $\boldsymbol{x}$, with $\boldsymbol{\mu}_N$ and Σ_N respectively the mean and covariance matrix of the k-neighborhood N of $\boldsymbol{x}$, i.e. the Mahalanobis distance that accounts for the local covariance structure

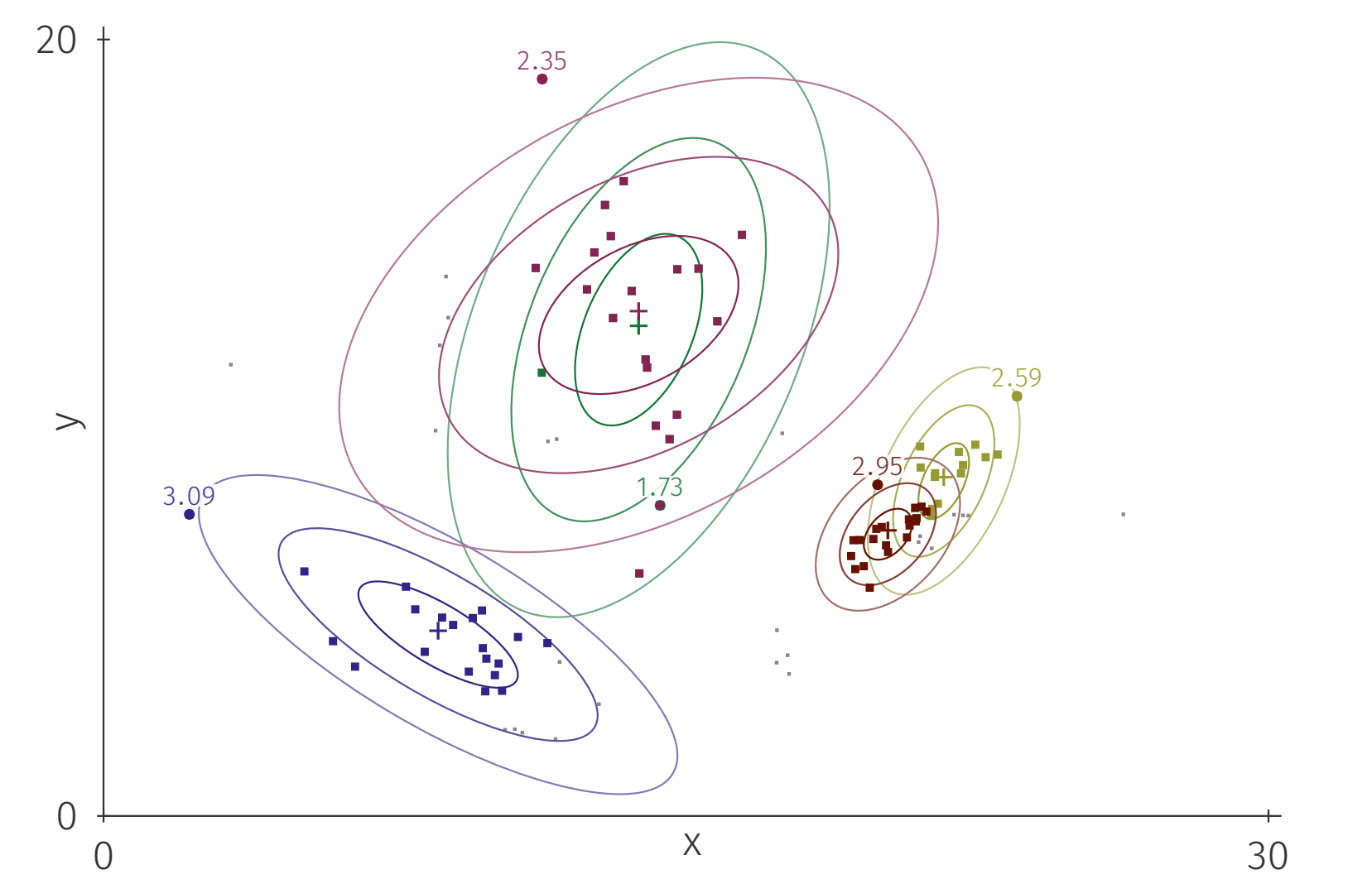
Instance-specific Mahalanobis distance

$D_{\Sigma_N}(\boldsymbol{x}, \boldsymbol{\mu}_N)$ for the k-neighborhood N of $\boldsymbol{x}$, $k = 19$



Instance-specific Mahalanobis distance

$D_{\Sigma_N}(\boldsymbol{x}, \boldsymbol{\mu}_N)$ for the k-neighborhood N of $\boldsymbol{x}$, $k = 19$



High-dimensional data

High-dimensional approaches

As dimensionality increases the distances between pairs of points become more similar, outliers become increasingly more difficult to tell apart from normal points

Angle-based method

More stable than distances in high-dimensional spaces
e.g. cosine based similarity measure for text data

Assumption: outliers lie at the border of the data space, whereas inliers lie in the center of the data space

The rest of the data is in a similar direction from an outlier, in varying directions from an inlier

High-dimensional approaches

Outliers typically present anomalous behavior only in a small subset of attributes while other dimensions are irrelevant to the anomaly detection process

Subspace outlier detection

An outlier is defined in association with one or more subspaces that are specific to it

Consider projections into lower dimensional subspaces to detect associated outliers

Subspace outlier detection

There is an analogy between *subspace clustering* and *subspace outlier detection* but the levels of difficulty are not similar

It is much easier to determine *frequent* characteristics of a dataset than *rare* characteristics
Dense subspaces can be determined by *aggregate analysis* of the data points whereas detecting outliers requires to explore subspaces in a way that is *specific to individual points*

For a d-dimensional dataset, there are 2^d subspaces
Only a small fraction of them will expose the anomalous behavior of an individual point

Grid-based sparsity coefficient

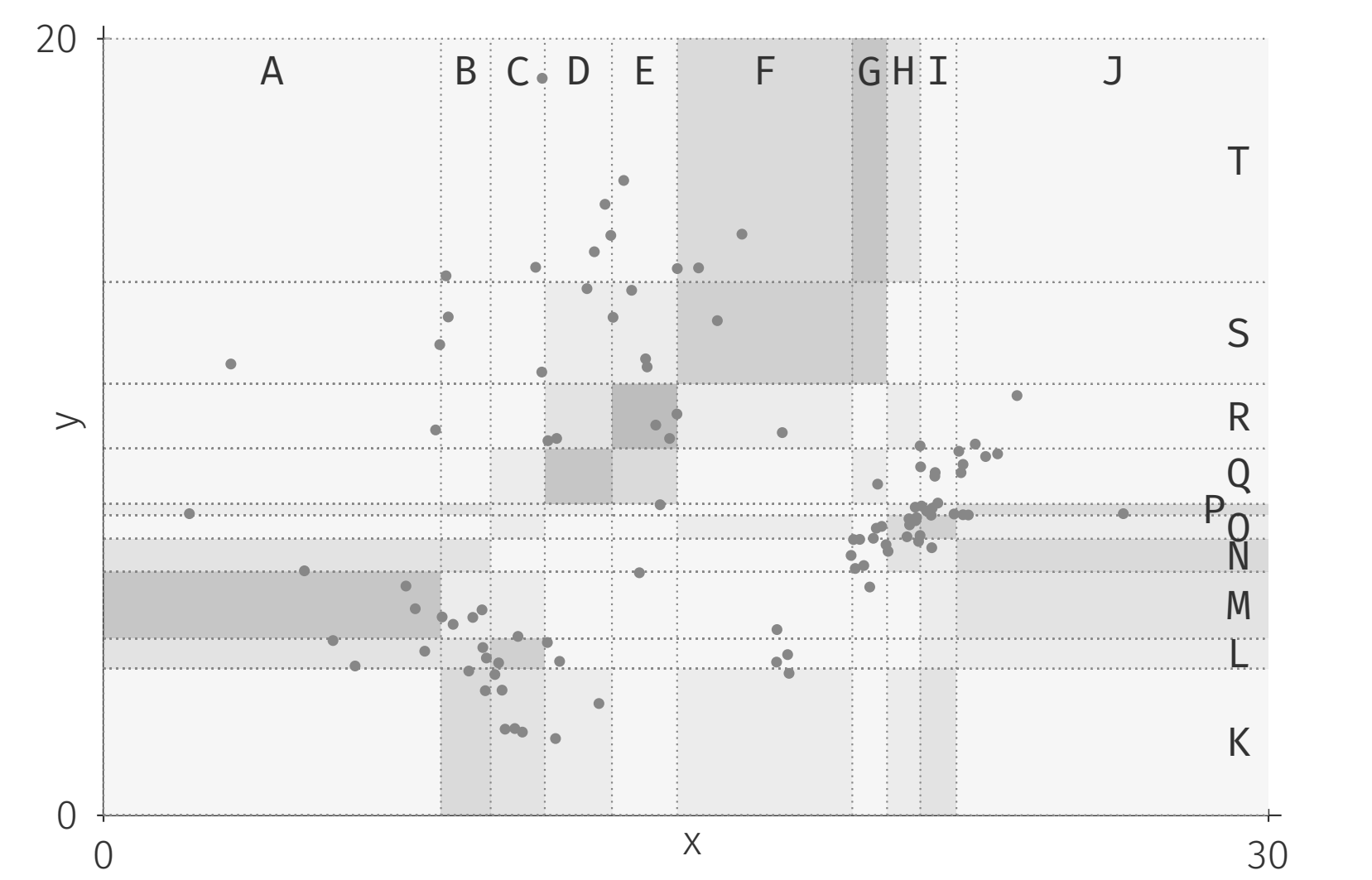
Partition each attribute into p bins containing each an equal fraction $f = 1/p$ of data points

Selecting k attributes and one bin from each defines a k-dimensional grid cell or cube

Under the independence assumption, the presence or absence of an individual point in such a cube is a Bernoulli random variable with success probability f^k
When the total number of points n is large, the number of points in the cube follows a normal distribution with $\mu = n \cdot f^k$ and $\sigma^2 = n \cdot f^k \cdot (1 - f^k)$

Grid-based sparsity

Histogram for bins containing 10% of data points



Grid-based sparsity coefficient

Partition each attribute into p bins containing each an equal fraction $f = 1/p$ of data points

Selecting k attributes and one bin from each defines a k-dimensional grid cell or cube

The sparsity coefficient for cube $\mathcal{R}$ containing $n_{\mathcal{R}}$ data points is

$$S(\mathcal{R}) = \frac{n_{\mathcal{R}} - n \cdot f^2}{\sqrt{n \cdot f^k \cdot (1 - f^k)}}$$

A negative sparsity coefficient indicates that the number of points in the cube is significantly lower than expected

Grid search for subspace outliers

Individual dimensions provide no information about the combination of dimensions
Level-wise algorithms are not practical
Consider an evolutionary (genetic) algorithm

Genetic algorithms mimic the process of biological evolution to solve optimization problems

Genetic algorithms

Candidate solutions to the optimization problem are represented by a *population of individuals*
Each feasible solution has a string *encoding*, akin to its chromosome

The *fitness* of an individual is the objective value of the corresponding solution

The *selection operator* accounts for the fact that fitter individuals are more likely to survive and multiply
The *crossover* and *mutation operators* allow individuals to evolve

Genetic algorithms

The selection operator replicates individuals in the population with a bias towards fitter individuals

The crossover operator exchanges the segments of two encodings to the right of a randomly chosen position
An optimized crossover operator the outcomes of possible recombinations and select the best one

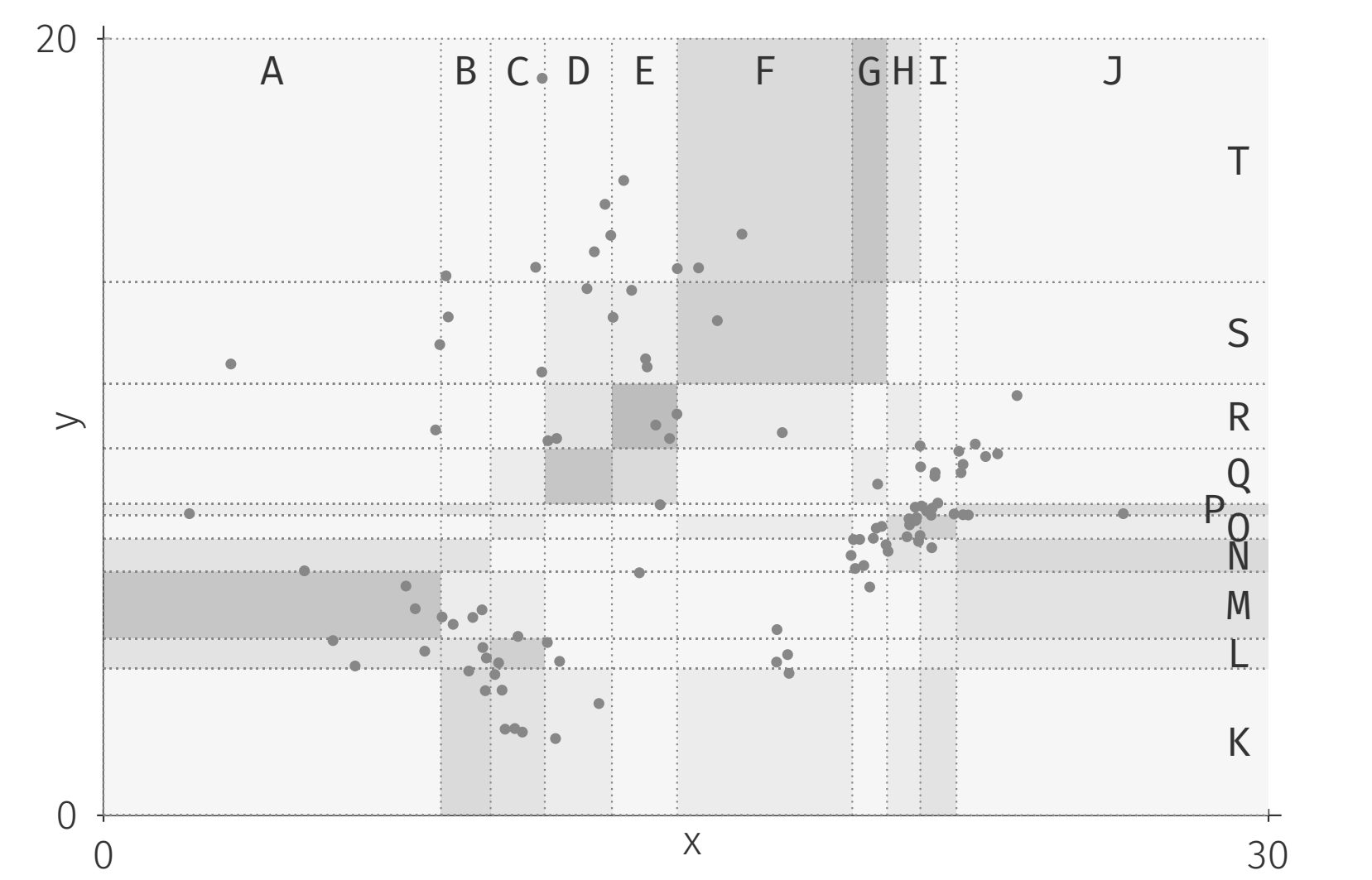
The mutation operator flips positions in the encoding with a predefined probability

Genetic algorithm for subspace outliers

In the grid search for subspace outliers encodings are strings of length d
Each position specifies a bin for the corresponding attribute or that the dimension is not included, $\{1, \ldots, p, *\}$
Each encoding corresponds to a cube

The fitness of an individual is the sparsity coefficient of the cube associated to its encoding

Grid-based sparsity



Genetic algorithm for subspace outliers

The process starts with a population of q random individuals and iteratively repeats the process of selection, crossover, mutation

Individuals in the population progressively improve in fitness and become more similar

A position in the encoding has converged when a predefined fraction of the population has the same value for that position

The population has converged when all positions in the encoding have converged

Keep track of the best solutions encountered, i.e. cubes with most negative sparsity coefficients

Data points contained in those cubes are reported as outliers

Isolation-based methods

Assumption: outliers are few, not similar to the rest of the data and located in sparse regions, hence suceptible to isolation

Grow binary decision trees at random until all distinct data points are in a node of there own

Data points that are reached via short paths are reported as outliers

Isolation trees: training

Build an isolation tree with recursive algorithm `iTree`

`iTree`(S):
if S cannot be divided **then**
    return leaf node
else
    $a \leftarrow$ select attribute of S at random
    $v \leftarrow$ select value in $[\min_{x \in S} x_a, \max_{x \in S} x_a]$ at random
    return node with test $x_a \geq v$,
        left child `iTree`$(\{x \in S, x_a < v\})$ and
        right child `iTree`$(\{x \in S, x_a \geq v\})$

Isolation trees: training

Build an isolation tree with recursive algorithm `iTree`
Collect trees built on θ different random data samples of size κ to form a decision forest

$\mathcal{F} \leftarrow \emptyset$
for $i = 1 \ldots \theta$ **do**
    $S \leftarrow$ sample κ data points from $\mathcal{D}$ at random
    $\mathcal{F} \leftarrow \mathcal{F} \cup \{\texttt{iTree}(S)\}$
return $\mathcal{F}$

Isolation trees: evaluation

The outlier score of a data point x is the average length of paths from root to leaf in trees of the forest

Run x through each tree in the forest until reaching a leaf
Return the average length of the path from root to leaf node over the different trees

In practice, the depth of trees is limited during training
Path lengths are normalized to account for this limit and for the sample size

Temporal data

Outliers in temporal data

In the context of temporal data, *outlier detection* is also known as *event detection*, especially when performed in real-time

A sudden change at a given timestamp of a time-series or sequence is referred to as **contextual outlier** or **point outlier**

An anomalous pattern of consecutive data points is referred to as **collective outlier**, as well as **shape outlier** in the context of time-series and **combination outlier** in the context of discrete sequences

Point outliers

The detection of point outliers is closely related to forecasting

A data point is considered an outlier if it deviates significantly from its forecasted, i.e. expected, value

Point outliers in discrete sequences

Build a probabilistic suffix tree from historical data, capturing the typical behavior of the sequence

The probability of observing a specific value at a given position, in the context of the values occurring at the previous postion(s) can be retrieved from the tree
Positions where this probability is very low are reported as anomalies

Point outliers in multivariate time-series

Given a multivariate time-series $\mathcal{S}_X = \langle \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)} \rangle$, with $\boldsymbol{x}^{(i)} \in \mathbb{R}^m$, the aim is to detected unexpected events

Point outliers in multivariate time-series

Given a multivariate time-series $\mathcal{S}_X = \langle \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)} \rangle$, with $\boldsymbol{x}^{(i)} \in \mathbb{R}^m$, the aim is to detected unexpected events

1. Predict the values at each timestamp using some time-series modelling approach

Let $\boldsymbol{y}^{(i)}$ be the m-dimensional vector of forecasted values at timestamp i

Point outliers in multivariate time-series

Given a multivariate time-series $\mathcal{S}_X = \langle \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)} \rangle$, with $\boldsymbol{x}^{(i)} \in \mathbb{R}^m$, the aim is to detected unexpected events

2. Compute the multivariate time-series of deviations between the forecasted and the actual values

Let $\boldsymbol{\delta}^{(i)} = \boldsymbol{y}^{(i)} - \boldsymbol{x}^{(i)}$ be the m-dimensional vector of deviations at timestamp i

Point outliers in multivariate time-series

Given a multivariate time-series $\mathcal{S}_X = \langle \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)} \rangle$, with $\boldsymbol{x}^{(i)} \in \mathbb{R}^m$, the aim is to detected unexpected events

3. Compute the normalized deviation, i.e. z-number, for each timestamp and each variable

Let μ_j and σ_j^2 be the mean and variance of the deviations for variable j across the forecasted timestamps, $\delta_j^{(1)}, \delta_j^{(2)}, \ldots, \delta_j^{(n)}$

$$z_j^{(i)} = \frac{\delta_j^{(i)} - \mu_j}{\sigma_j}$$

Point outliers in multivariate time-series

Given a multivariate time-series $\mathcal{S}_X = \langle \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)} \rangle$, with $\boldsymbol{x}^{(i)} \in \mathbb{R}^m$, the aim is to detected unexpected events

4. Report as anomalies the pairs of timestamps and variables for which the z-number exceeds a chosen threshold value, typically 3

Timestamp–variable pair (i, j) is reported as outlier if $z_j^{(i)} > 3$

Depending on the application, one might aggregate the deviations at a given timestamp, taking for instance the maximum or average over the different variables, i.e. report timestamp i as outlier if $\mathbf{max}_{j=1\ldots m}\, z_j^{(i)}$ or $\mathbf{mean}_{j=1\ldots m}\, z_j^{(i)}$, respectively, exceed the chosen threshold

Point outliers in multivariate time-series

Given a multivariate time-series $\mathcal{S}_X = \langle \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)} \rangle$, with $\boldsymbol{x}^{(i)} \in \mathbb{R}^m$, the aim is to detected unexpected events

1. Predict the values at each timestamp using some time-series modelling approach
2. Compute the multivariate time-series of deviations between the forecasted and the actual values
3. Compute the normalized deviation, i.e. z-number, for each timestamp and each variable
4. Report as anomalies the pairs of timestamps and variables for which the z-number exceeds a chosen threshold value, typically 3

Combination outliers

The aim is to identify unusual combinations of values appearing in a sequence

Small windows of a chosen size, referred to as *comparison units*, are extracted from the sequence
Distances between comparison units can be computed using e.g. dynamic time warping (DTW) distance, edit distance, etc.

The k-nearest neighbor distance can be used as outlier score

Shape outliers

Shape outliers are defined over windows of the time-series
Distance to k-nearest neighbors is used as outlier score

1. Extract all candidates by sliding a window of length w over the time-series
2. Compute the Euclidean distance from each candidate to all other non-overlapping windows
3. Report candidates with highest k-nearest neighbor distance as outliers

Use non-overlapping windows to prevent trivial matches
Pruning and early termination are used to improve efficiency

Shape outliers: HOTSAX

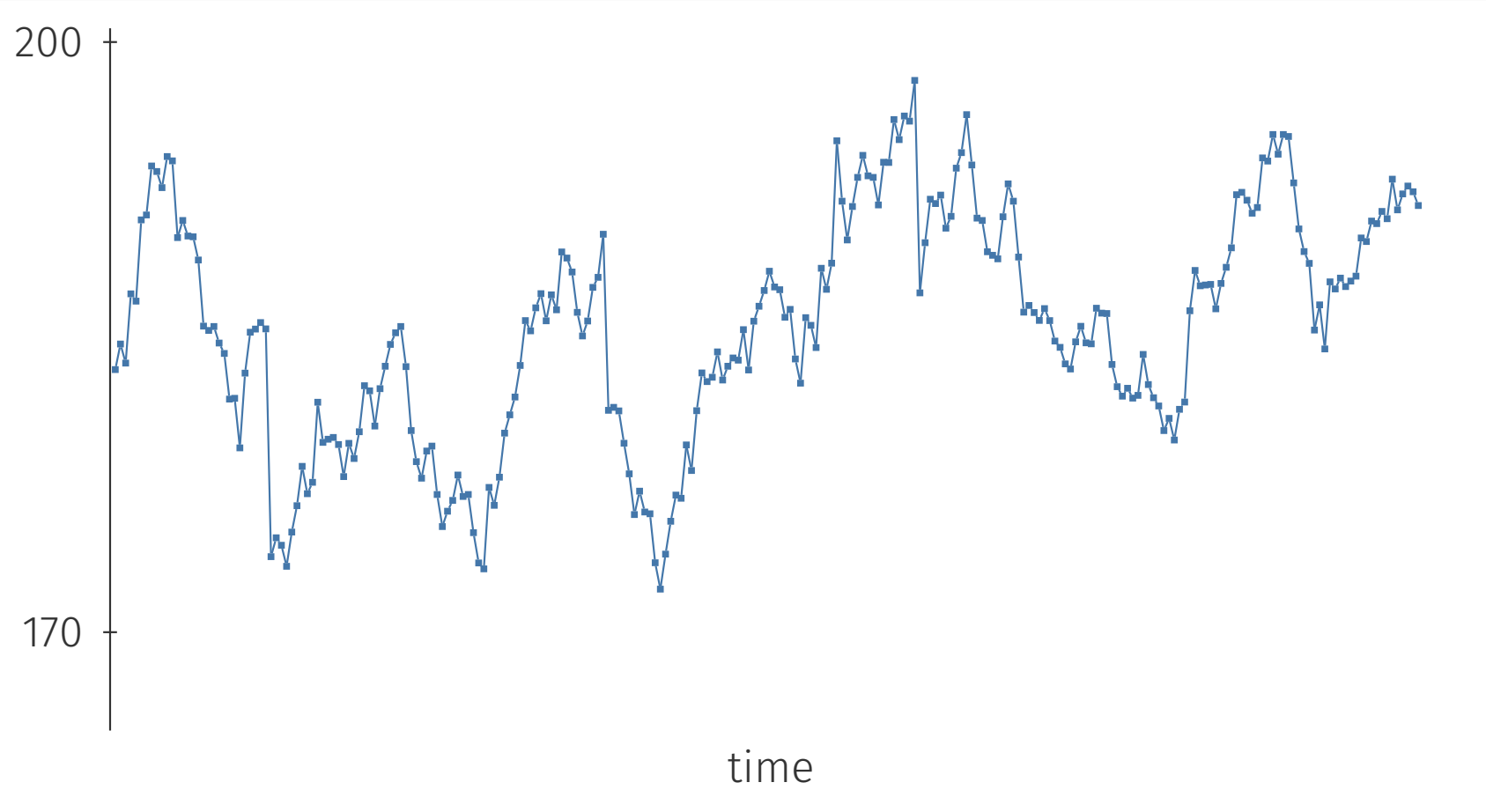
Pruning and early termination are used to improve efficiency
It works best if true outliers are found early, i.e. more promising candidates need to be processed first

The clustering behavior of candidates informs about how promising they are
Process candidates from clusters having fewest members first
Other candidates from the same clusters are considered first when computing the nearest neighbor distances

Use **symbolic aggregate approximation (SAX)** representation to map candidate windows to clusters, one cluster for each distinct SAX word
Piecewise aggregation approximation is done with intervals of size $k < w$ resulting in SAX words of length w/k

Shape outliers: HOTSAX

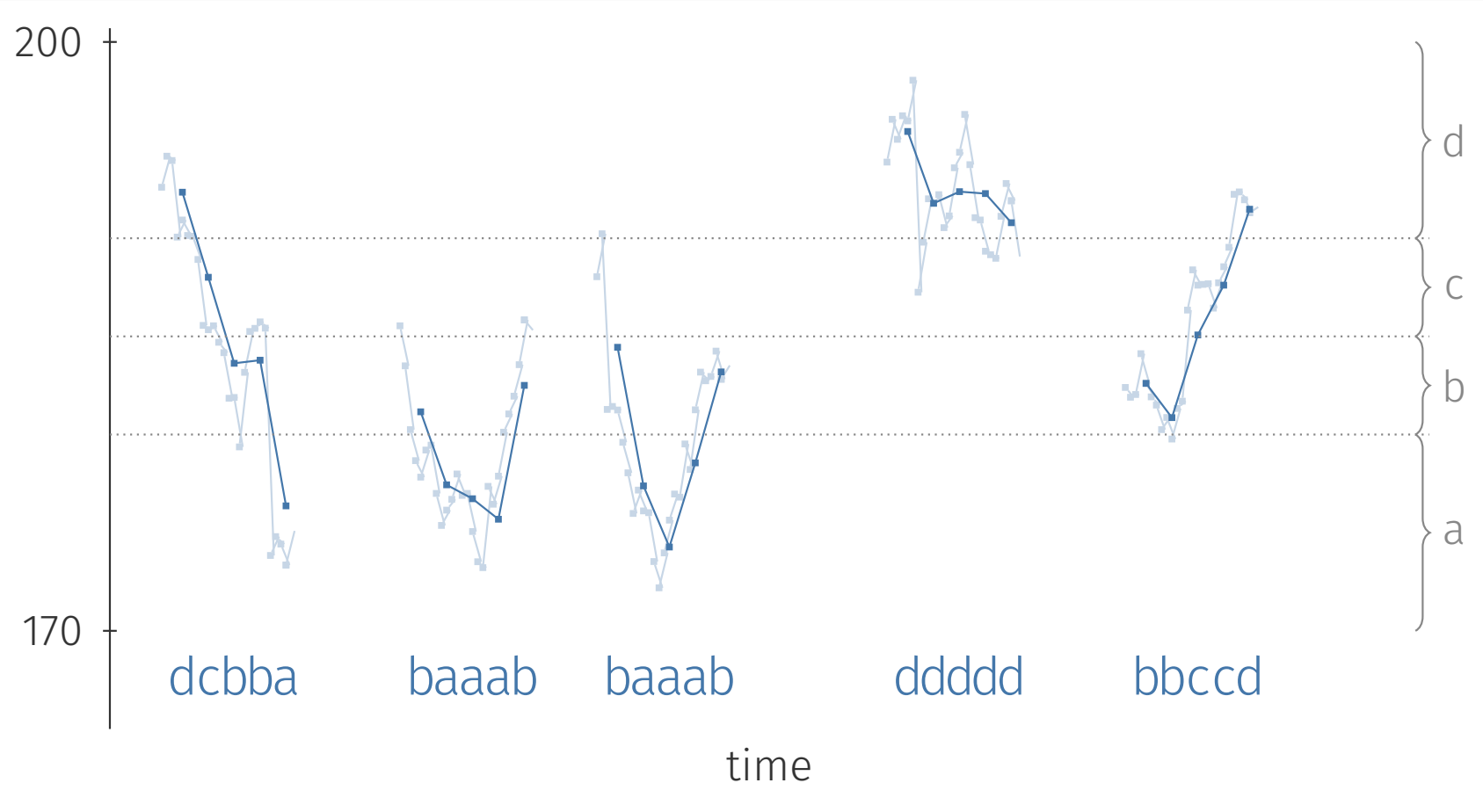
IBM stock prices from Sept. 2013 to Sept. 2014



Shape outliers: HOTSAX

IBM stock prices from Sept. 2013 to Sept. 2014

SAX of candidate window maps to cluster



Rule-based supervised detection of shape outliers

Training

Given a database of time-series of length n, some labelled as anomalous, the aim is to train a classifier that can identify anomalous series

1. Use the discrete wavelet transform to convert each time-series into a vector of coefficients
2. Discretize the wavelet representation, i.e. turn each dimension of the numerical wavelet representation into a categorical attribute by partitioning the range of values into intervals
3. Extract a set of rules by applying a rule-based classifier

Rule-based classifiers

Rule-based classifiers make predictions using a collection of rules of the form "`if` *condition* `then` *conclusion*", $Q \implies c$

The *condition* Q (a.k.a. *antecedent*) typically consists of a conjunction of tests on the data attributes

The *conclusion* c (a.k.a. *consequent*) typically consists of a class label

If an instance satisfies the conditions of a rule, we say that the rule *covers* the instance and that the instance *triggers* the rule

A set of exhaustive and mutually exclusive rules can be generated from decisions trees

An ordered list of rules can be extracted by growing them one by one using a sequential covering algorithm

Rule-based supervised detection of shape outliers

Predicting

To make a prediction for a given time-series, its discrete wavelet transform is computed then discretized in the same way as the training instances

The collection of rules is scanned, evaluating their conditions on the categorical transformed representation of the time-series

The time-series is reported as an outlier if it triggers some rule having the minority outlier class as its conclusion