2 Fundamentals

The first section of this chapter familiarizes the reader with the definitions of the basic notation and terminology used in this thesis. Concepts of graph theory are introduced in the next section. They give rise to a new concept of neighborhoods, which is utilized in several chapters. The last section explains a possible approach to knowledge discovery, which is applied in chapters 11 and 12.

2.1 Basic Definitions

Hilbert space
Let $\mathcal{H}$ be a vector space above a field K with the following properties for every pair of elements $(x, y, z) \in \mathcal{H}$ and $\alpha \in$ K:

1.) $\langle .,. \rangle_{\mathcal{H}} : \mathcal{H}$ x $\mathcal{H} \rightarrow$ K is a non-degenerate symmetric bilinear form:
   a. $\forall\, x \in \mathcal{H} : \langle x, x \rangle_{\mathcal{H}} \geq 0$
   b. $\langle x, y \rangle_{\mathcal{H}} = 0, \forall\, y \in \mathcal{H} \Rightarrow$ x=0
   c. $\langle x, y \rangle_{\mathcal{H}} = \overline{\langle y, x \rangle_{\mathcal{H}}}$ if $\mathrm{K} = \mathbb{C}$, and $\langle x, y \rangle_{\mathcal{H}} = \langle y, x \rangle_{\mathcal{H}}$ if $\mathrm{K} = \mathbb{R}$
   d. $\langle \alpha x, y \rangle_{\mathcal{H}} = \alpha \langle y, x \rangle_{\mathcal{H}}$
   e. $\langle x + y, z \rangle_{\mathcal{H}} = \langle x, z \rangle_{\mathcal{H}} + \langle y, z \rangle_{\mathcal{H}}$
2.) Each Cauchy sequence $\{x_i\}_{i \in \mathbb{N}}$ in $\mathcal{H}$ converges to an element of $\mathcal{H}$, i.e., the space is complete with respect to the norm induced by $\langle .,. \rangle_{\mathcal{H}}$.

Thus, $\mathcal{H}$ is a Hilbert space (for further details, see [Bronstein et al., 2005, pp. 635-636; Nolting, 2001, p. 22]).

Bra-ket notation
Bra-ket notation $\langle . | . \rangle$ is used in physics to describe functions or vectors in a Hilbert space when the coordinate system of the vectors is irrelevant. The left part is called the bra ($\langle . |$), and the right part is the ket ($| . \rangle$). This notation is used to describe physical states (it is also called Dirac notation, as described in [Dirac, 1981, pp. 15-22]; for a formal introduction, see [Nolting, 2001, pp. 147-148]).

Operator
An operator $\hat{A}$ is an unambiguous mapping of each element $|\alpha\rangle$ of the subset $D_\alpha \subseteq \mathcal{H}$ to an element $|\beta\rangle \in W_A \subseteq \mathcal{H}$ such that $|\beta\rangle = \hat{A}\,|\alpha\rangle = |\hat{A}\,\alpha\rangle$, where D_α is the definition range of $\hat{A}$ and the set of all $|\beta\rangle$ is the domain of $\hat{A}$, as defined in [Nolting, 2001, p. 153]; see also [Bronstein et al., 2005, pp. 49,639-640]. An "operator is considered to be completely defined when a result of its application to every ket vector [$|\alpha\rangle$] is given" [Dirac, 1981, p. 23].

Observation
An observation *f* is a set of measured values for the properties of a phenomenon. It is described in the bra-ket notation as the change from one physical state $\langle y|$ to another physical state $|x\rangle$ that results from the measurement of the operator $\hat{f}$, as denoted by $f = \langle y|\hat{f}|x\rangle$ (see [Feynman et al., 2006, pp. 145, 147]). Such an observation *f* is a measurement of a physical process.

M. C. Thrun, *Projection-Based Clustering through Self-Organization and Swarm Intelligence*, https://doi.org/10.1007/978-3-658-20540-9_2

Feature

Each individually measurable property *r* of a phenomenon being observed can be mapped to an operator $\hat{r}$ that can be applied to a physical state $|x\rangle$ [Stöcker et al., 2007, p. 744]. Such an individually measurable property is called a **feature**, **attribute** or **observable**. Here, an approximately continuous distribution of values in the vector space $\mathbb{R}^d$ is additionally assumed for a **variable** (see the definition of the **distribution of a variable**).

Data

A batch of data is defined as a matrix $\langle i|\hat{A}\,|j\rangle = A_{ij}$, in which **facts**[1] about a physical state are summarized based on observations of the form $\langle y|\hat{A}\,|x\rangle = \sum_{ij}\langle y|i\rangle\langle i|\hat{A}\,|j\rangle\langle j|x\rangle$ of a phenomenon in a Hilbert space, where $\langle i|$, $\langle j|$, $|i\rangle$ and $|j\rangle$ are the basic states relevant to the phenomenon (for further discussion, see [Feynman et al., 2006, pp. 147-150]).

Distribution of a variable

A formal distribution df is defined as the probability density of a feature r: $df(r) = \lim_{\Delta r \to 0} \frac{\langle x_{r,\Delta r}|\, x\rangle}{\sqrt{(\Delta r)}}$ [Nolting, 2001, p. 150]. If the feature *r* is continuous, then it is called a **variable** $z \in \mathbb{R}^d$, and *df* is called its probability density function (**pdf**) (see [Goodfellow et al., 2016, p. 58]). Here, when it describes how the relative probability of a variable z takes on a given value, such a distribution is a pdf that is assumed to be normalized as follows [Walck, 2007, p. 15]: $\int_{-\infty}^{\infty} pdf(z)dz = 1$.

"Statisticians often use the distribution function or as physicists more often call it the cumulative function which is defined as $cdf(z) = \int_{-\infty}^{x} pdf(z)dz$" [Walck, 2007, p. 15].

If not elaborated further, here, the distribution of a variable *z* is regarded as an approximation of its pdf; for further details, see, for example, [Bock, 1974, p. 250; G. Ritter, 2014, p. 275 ff], and for types of pdfs, see [Walck, 2007].

Dirac delta function

The Dirac delta function δ is a function with the following properties [Jackson, 1999, p. 31]:

1.) $\delta(z-a) = 0$ iff $z \neq a$

2.) $\int \delta(z-a) = \begin{cases} 1, & \text{if } z = a \text{ lies in the integration area under the curve} \\ 0, & \text{otherwise} \end{cases}$

Density of data

Let *dn* be the number of observations in an **elementary volume** (see [Bronstein et al., 2005, p. 491]) $d^{\tilde{d}}v = dv_1 * dv_2 * .. dv_{\tilde{d}} = d\vec{v}$ of the Hilbert space $\mathbb{R}^{\tilde{d}}$ (henceforth, $\mathbb{R}^d$); then, the density of the data is defined as $\rho(\vec{v}\,) = \frac{dn}{d\vec{v}}$, where $\rho\colon \mathbb{R}^d \to \mathbb{R}$ is the density field function. Here, ρ is subject to the condition that N is the number of data points defined by $N = \int_{\mathbb{R}^d} \rho(\vec{v})\mathrm{d}\vec{v} = \int_{\mathbb{R}^d} \sum_{i=1}^{N} \delta(\vec{v} - \vec{v}_i)\mathrm{d}\vec{v}$, in analogy to [Jackson, 1999, p. 33], where δ is the Dirac delta function and $\rho(\vec{v}) = \sum_{i=1}^{N} q_i \delta(\vec{v} - \vec{v}_i)$ is the charge density of point charges. Then, the **homogeneity** of the data is defined as $N = \int_{\mathbb{R}^d} \rho(\vec{v})\mathrm{d}\vec{v} = \int_{\mathbb{R}^d} \rho_0 \mathrm{d}\vec{v} = \rho_0 \int_{\mathbb{R}^d} \mathrm{d}\vec{v}$, where $\rho_0 = \text{const.}$

[1] See [Fayyad et al., 1996, p. 6].

Pattern

A "[p]attern is an expression E in a language L describing facts [F] in a subset F_E of F. E is called a pattern if it is simpler than the enumeration of all facts in F_E" [Fayyad et al., 1996, p. 7]. Here, the expression E is "simpler" if it describes a group of similar (see the definitions of **metric space** and **distance** below) or homogeneous observations.

In graph theory, a pattern may be described by a neighborhood H (see the graph theory section for details). If the observations are not directly comprehensible, such a pattern is called a *hidden pattern*.

Discontinuity in data

A set of data can exhibit discontinuity if

$$\int_{\mathbb{R}^d} \rho(\vec{v}) \mathrm{d}\vec{v} \neq \rho_0 \int_{\mathbb{R}^d} \mathrm{d}\vec{v},$$

which means that the density of data ρ depends on its location $\vec{v}$ in the Hilbert space $\mathbb{R}^d$; Discontinuities can occur when interruptions or distortions exist in the homogeneity of the data, or in the continuity of the distribution of the data, in $\mathbb{R}^d$. Thus, there are elementary volumes $\mathrm{d}\vec{v}$ with high density and elementary volumes $\mathrm{d}\vec{v}$ with low density or even empty elementary volumes. In the one-dimensional case, such a discontinuity can be mathematically defined as an essential or jump discontinuity. In two or three dimensions, a discontinuity may manifest as a spatial separation (see, e.g., Figure 2.1 or chapter 5 and 9, the Hepta data set).

In a higher-dimensional case, a discontinuity represents a change in the characteristics of facts, resulting in multiple patterns (see, for example, the leukemia data set, chapter 3, Figure 3.7 and chapter 9).

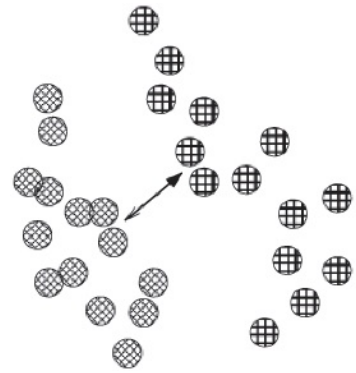

Figure 2.1: Spatial separation of data, after [Handl et al., 2005].

Metric space and distance

Let a metric space be represented by an ordered pair *(M, d)*, where *M* is an arbitrary set and *d* is a metric on *M*, i.e., a function

$$\mathrm{d} : \mathrm{M} \times \mathrm{M} \rightarrow \mathbb{R}$$

such that for any $l, j, m \in M$,

$$d(l,j) = d(j,l)$$
$$d(l,j) \geq 0$$
$$d(l,j) = 0, \text{iff } l = j$$

and the triangle inequality is satisfied as follows:

$$d(l,j) + d(j,m) \geq d(l,m)$$

Then, the metric *d* is also called a **distance** (see [Bronstein et al., 2005, pp. 624-625]). By contrast, for a **dissimilarity**, denoted by $\hat{d}$, the triangle inequality may not apply ([Bock, 1974, pp. 25-26]. The distance between two **similar** points $l, j \in M$ is small, whereas that between two **dissimilar** points $l, j \in M$ is large. Transformations exist between a dissimilarity $\hat{d}$ and a distance *d* (e.g., [Bock, 1974, pp. 77-79]).

If the distance is defined in an output space O, it is denoted by *d(l, j)*, whereas a distance defined in an input space *I* is denoted by *D(l, j).* An example of a metric space is a Hilbert space that is a real-numbered vector space $\mathbb{R}^{\mathrm{d}}$ of *d* dimensions. If the distances in a space are defined as Euclidean distances, then the corresponding space is called a Euclidean space.

Data set

A data set consists of a finite set of observations $f \in \mathrm{F} \subset \mathcal{H}^{\tilde{\mathrm{d}}}$ of $\tilde{d}$ observed features.

In this work, observations f are assumed to be vectors *l* in a metric space *M*, and features are assumed to be variables, if not stated otherwise.

Input space

An input space $I \subset \mathbb{R}^d$ is the d-dimensional space consisting of $d \leq \tilde{\mathrm{d}}$ variables in a data set that have been selected for a given task and contains *n* data points: $I = \{l_1, \ldots, l_n, n \in \mathbb{N}\}$. The properties of an input space are as follows (see [Lee/Verleysen, 2007, p. 243]):

I. The input space is considered to be *high dimensional* if it contains more than five variables, which makes direct visualization very difficult.
II. If the number of data points is greater than 2000, then the input space is considered to be *large*[2].
III. If the number of data points is fewer than 200, then the input space is considered to be *small*.

Data point

A data point $l \in I$ is a numeric vector consisting of one observation for each of the *d* variables in the input space, where a vector is an array of numbers arranged in a specific order defined with respect to the d variables.

[2] Note that, in general, the number of data points has greatly increased over time [Goodfellow et al., 2016, p. 21 , Fig. 1.8] and therefore the precise number may change with time

Object
When the data of interest are a set of facts F consisting of numerical, ordinal or nominal scaled entries, each fact $f \in F$, such that f $\notin \mathbb{R}^{\mathrm{d}}$, is called an object or **case**.
An object can be regarded as a generalization of a data point. If an object can be interpreted (has a meaning within itself), then it contains **information** ([Ultsch, 2016c]; see also [Ultsch, 1994, p. 2]).

Output space
An output space $O \subset \mathbb{R}^m$ is the m-dimensional space such that $m<d$ in which, for each point $j \in O$, a mapping to a data point l of the input space I$\subset \mathbb{R}^d$ exists.

Machine learning
The field of machine learning concerns computer programs that can imitate learning behavior [Natarajan, 2014] (see also [Goodfellow et al., 2016, p. 99]). Machine learning comes in two general forms[3] (see [Murphy, 2012, p. 2]). *Unsupervised learning* refers to the task of finding patterns in unlabeled data. Since the data are unlabeled, no reward function exists that can be used to evaluate potential results. If the data set is labeled, then *supervised learning* is possible. A typical supervised learning task is classification or regression. A typical unsupervised learning task is cluster analysis.

Label
A label is a tag $g \in \{1, \ldots, k\} \subset \mathbb{N}$ attached to an object $f \in F$ that identifies the object via a mapping $f: \{1, \ldots, k\} \to F$. The labels of such a set of objects range from *one* to *k* [Hennig et al., 2015, p. 2], where k is the number of groups of objects. Here, it is assumed that a label exists for every object.

Classification
A classification $C = \{G_1, G_2, \ldots\}$ is a system of subsets [Bock, 1974, p. 22] such that $C \subset \mathcal{H}^{\tilde{\mathrm{d}}}$. A subset $G_i = \{l_1, \ldots l_k\} i \in \mathbb{N}$, , is a set of k observations. In an exclusive classification, the subsets are disjunct, denoted by $G_1 \cap G_2 = \emptyset$; in a non-exclusive classification, elements that overlap between two subsets may exist, denoted by $G_j \cap G_k \neq \emptyset$. However, overlapping classification is not considered here (for various types of classification, see Figure 2.2 or [Hennig et al., 2015, p. 45]). Supervised and unsupervised classifications are defined as in the context of machine learning.

[3] Reinforcement learning is not considered in this context; semi-supervised learning (e.g. active learning) uses labeled data as well as unlabeled data.

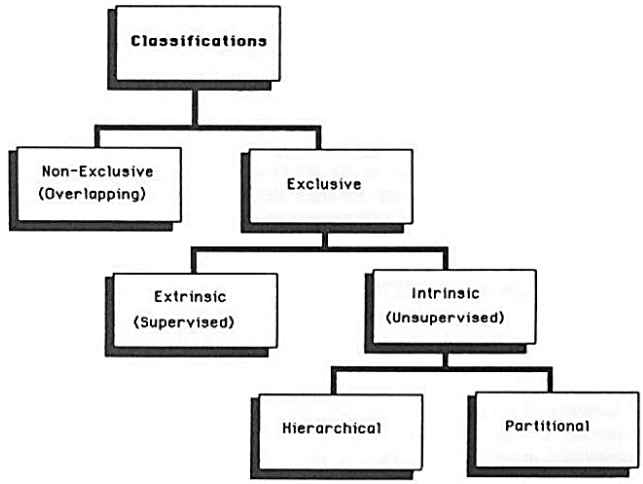

Figure 2.2: Tree of classification types, after [Jain/Dubes, 1988, p. 56]. This work concentrates on unsupervised classification (see unsupervised machine learning).

Classifier

A classifier is an algorithm that constructs a function $Cls: F \to \{1, \ldots, k\} \subset \mathbb{N}$ that maps objects $f \in F$ to class labels $g_i \in \mathbb{N}$.

In terms of understandability, a distinction can be drawn between symbolic and sub-symbolic classifiers [Ultsch/Korus, 1993]. Symbolic classifiers are able to acquire knowledge (for a detailed description, see the last section of this chapter). By contrast, sub-symbolic classifiers (e.g., KNN classifiers) are only able to integrate knowledge [Ultsch, 1994], because a characteristic property of a sub-symbolic representation of data is that a single object alone does not contain information (see [Ultsch, 1994, p. 2]).

Projected point

A projected point $j(x_1, .., x_m) = \vec{j}$ is a vector of m scalars x_i in the output space $O \subset \mathbb{R}^m$, where a vector is an array of numbers arranged in a specific order such that each individual number can be identified by its index.

Projection

Let $j \in I$ denote data points in the input space $I \subset \mathbb{R}^d$, and let $l \in O$ denote projected points in the output space $O \subset \mathbb{R}^m$. Then, a mapping $\mathrm{proj}: I \to O,\ j \mapsto l$ is called a projection iff $m = const \wedge m \ll d$.

Note that unlike for a projection method, for a manifold learning method, the dimensionality of the output space m depends on the data set (see, e.g., [Lee/Verleysen, 2007, pp. 14-15]).

2.2 Concepts of Graph Theory Applied to Patterns

This section uses graph theory to describe patterns found in data.

Graph

"A graph [Γ] is a pair [$\Gamma = (V, E)$] consisting of a finite set $V \neq \emptyset$ and a set E of two-element subsets of V. The elements of V are called vertices. An element e = (a, b) of E is called an edge with end vertices a and b. […] [In such a case,] *a* and *b* are adjacent or neighbors of each other" [Jungnickel, 2013, p. 2].

A graph Γ is called undirected if, for every edge $e(a, b)$ in E, the edge $e(b, a)$ is also in E. A graph is called a weighted graph if a number (weight) is assigned to each edge.

Directed graph

A "directed graph or, for short, a *digraph* is a pair $\Gamma = (V, E)$ consisting of a finite set *V* and a set *E* of ordered pairs *(a, b),* where $a \neq b$ are elements of V" [Jungnickel, 2013, pp. 25-26].

Direct adjacency

Let Γ be a graph, and let j be a point in a metric space M; then,

$$\mathcal{H}(j, \Gamma, M) = \{l \in M \mid v_l \in V \wedge \exists\, e(v_l, v_j) \in E\}$$

is the set of points that are directly adjacent to j. The direct adjacency is defined by the specified graph.

Adjacency matrix

A digraph Γ with a vertex set $\{1, \ldots, n\}$ is specified by an $n \times n$ matrix $A = (a_{ij})$, where $a_{ij} = 1$ if and only if (i, j) is an edge of Γ, and $a_{ij} = 0$ otherwise. *A* is called the adjacency matrix of Γ [Jungnickel, 2013, p. 40].

Path

Let $(e_1, \ldots, e_n)$ be a sequence of edges in a graph Γ. If there exist vertices $v_0, \ldots, v_n$ such that $e_i = v_{i-1}v_i$ for $i = 1, \ldots, n$, then the sequence is called a walk; if $v_0 = v_n$, one speaks of a closed walk (Figure 2.3). A walk for which the e_i are distinct is called a trail (Figure 2.3), and a closed walk with distinct edges is a closed trail. If, in addition, the v_j are distinct, then the trail is a path [Jungnickel, 2013, p. 5].

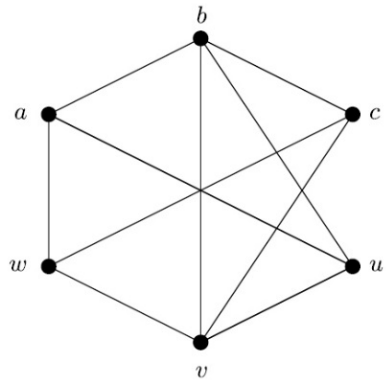

Figure 2.3: Examples of trails, walks and paths [Jungnickel, 2013, p. 6 Fig. 1.5]: (a, b, c, v, b, c) is a walk but not a trail, and (a, b, c, v, b, u) is a trail but not a path [Jungnickel, 2013, p. 5].

Connected Graph

Two vertices *a* and *b* of a graph *Γ* are called connected vertices if a walk exists with start vertex *a* and end vertex *b*. If all pairs of vertices of *Γ* are connected, then *Γ* itself is called a connected graph. For any vertex a, we consider *a* to be a trivial walk of length 0, such that any vertex is connected with itself. Thus, connectedness is an equivalence relation on the vertex set of *Γ*. The equivalence classes of this relation are called the connected components of *Γ*. Thus, *Γ* is connected if and only if its vertex set *V* is its unique connected component [Jungnickel, 2013, p. 6].

Lattice

A connected graph Γ with a particular well-defined two-dimensional tiling (tessellation) is defined as a lattice. A nxm lattice has n vertices on the x-axis and m vertices on the y- axis. If the tiling is rectangular (every vertex has exactly four perpendicular edges) it will be called a **lattice** (tiling) in this work, if the tiling is hexagonal (every vertex has exactly three edges) this will be called a **grid** (tiling) in this work.

Shortest path

For a connected graph Γ, there exists a distance *D(a, b)* between two vertices a and b that can be defined as the shortest path between these vertices [Jungnickel, 2013, pp. 65-66] as follows: For each path $P = (e_1, \dots, e_n)$, let the length of P be $p(P) := p(e_1) + \cdots + p(e_n)$; then, the distance between two vertices a and b in *(Γ, p)* is defined by

$$G(a, b, \Gamma) = \begin{cases} \infty, & \text{if } b \text{ is not accessible from } a \\ \min\{p(P): P \text{ is a path from } a \text{ to } b \text{ in } \Gamma\}, & \text{otherwise} \end{cases}$$

Let the vertices be denoted by points $l, j \in M$ in the metric space *M*; then, G(l, j, *Γ*) is the notation if the points l and j lie in the input space *I*, and $g(l, j, \Gamma)$ is the notation if they lie in the output space *O*.

Note that $d(a, a) = 0$ always holds because an empty sum is considered to have a value of 0, as usual. If no explicit length function is given, then the shortest paths and distances in a graph are defined using a length function that assigns a length of $p(e) = 1$ to each edge *e* [Jungnickel, 2013, p. 66]. An algorithm for calculating the shortest paths in a graph is described in [Jungnickel, 2013, pp. 83-87]. The authors Lee and Verleyson have claimed that graph distances outperform the traditional Euclidean metric in terms of dimensionality reduction [Lee/Verleysen, 2007, p. 227].

Acyclic graph

Let $(M, \preccurlyeq)$ be a partially ordered set (a poset, for short), which consists of the set *M* together with a reflexive, antisymmetric and transitive relation $\preccurlyeq$, and let *M* correspond to a digraph *Γ* with the vertex set *M* and with edges defined by pairs *(a, b)* such that $a \prec b$; then, because of the transitive property, *Γ* is acyclic [Jungnickel, 2013, p. 49].

Tree

A tree is a graph *Γ* that satisfies the following three conditions [Jungnickel, 2013, pp. 7-8]:

I. *Γ* is connected.
II. *Γ* is acyclic.
III. *Γ* contains *n-1* edges and *n* vertices.

The vertices in a tree are often called nodes. If (a, b) is an edge in a tree, then a is called the parent of b, and b is a child of a. If a path exists from a to b ($a \neq b$), then a is a proper ancestor of b and b is a proper descendant of a [Safavian/ Landgrebe, 1990, p. 2]. If a node has no descendant, it is called a leaf; if a node has no ancestor, it is called a root.

Directed acyclic graph (DAG)
A DAG is a directed tree (see above) that contains no cycles and one vertex, defined as the root, into which no edges enter. There is a unique path from the root to every vertex [Safavian/Landgrebe, 1990, p. 3]. Every vertex has a descendant called a child, except for the leaf vertices, which do not.

Decision tree
Let G_i be a subset of a classification $C = \{G_1, \dots, G_i, \dots\} \subseteq \mathcal{H}^{\tilde{\mathrm{d}}}$; then, a decision tree is a tree with the following properties:

I. Each node that is not a leaf is mapped to a feature $f \in F \subset \mathcal{H}^{\tilde{\mathrm{d}}}$.
II. Every edge (a, b), where a is the parent and b is the child, is mapped to a condition that matches the feature mapped to the parent a (see I.).
III. Every leaf is mapped to a subset G_i.

Decision tree learning
Decision tree learning refers to a type of supervised machine learning in which decision trees are used (see [Safavian/Landgrebe, 1990]).

Binary tree
A binary tree is an ordered tree such that [Safavian/Landgrebe, 1990, p. 3] (see also the definition of a DAG)

I. each child of a vertex is designated as either a left child or as a right child, and
II. no vertex has more than one left child nor more than one right child.

Lemma 1
Let $\Gamma = (V, E)$ be a connected graph with a positive length function p. Then, *(V, D)* is a finite metric space, where the distance function is defined as $D = G(a, b)$ [Jungnickel, 2013, p. 68].

Proposition 1
Any finite metric space can be represented by a pair (Γ, p) (network) with a positive length function *p* [Jungnickel, 2013, p. 68].

Ultrametric space
Note that a metric space can be represented by a tree if and only if the following condition holds for any four vertices *x*, *y*, *z*, and *t* of the given metric space [Jungnickel, 2013, p. 69]:

$$d(x, y) + d(z, t) \leq max(d(x, z) + d(y, t), d(x, t) + d(y, z))$$

Changing the triangle inequality to this condition implies an ultrametric space.

2.2.1 Patterns Defined as a Generalization of Neighbourhoods

Here, it is argued that by using shortest paths and direct adjacency, the patterns that exist in data can be generalized to neighborhoods H of an extent *k*.

Let k ∈ ℕ, *k>0*, let Γ be a connected graph, let *j* be a point in a metric space M, and let $G(j, l, \Gamma)$ be the shortest path between $j \in M$ and an arbitrary point $l \in M$; then (1),

$$H_j(k, \Gamma, M) = \{l \in M | \, G(l, j, \Gamma) \leq k\} \qquad (1)$$

is the neighborhood set of the point *j* and *k* the neighborhood extent. The neighborhood *H* can define a pattern in the input space[4].

The easiest example is a neighborhood defined by distances in a Euclidean graph. In the context of graph theory, a Euclidean graph is an undirected weighted graph of the highest order with respect to all other graphs discussed here, because every vertex is connected to every other vertex. Note that the weights of the vertices in a Euclidean graph need not necessarily be defined by the Euclidean metric. Another representation of a neighborhood H is a Delaunay graph $\mathcal{D}(V, E)$, which is a subgraph of a Euclidean graph. A Delaunay graph $\mathcal{D}(V, E)$ is based on Voronoi cells [Toussaint, 1980]. Each cell is assigned to one data point, and the size of a cell is characterized in terms of the nearest data points surrounding the point assigned to that cell. Within the borders of one Voronoi cell, there is no position that is nearer to any outer data point than to the data point within the cell. Thus, a neighborhood of data points is defined in terms of direct links between borders of Voronoi cells that induce an edge E in the corresponding Delaunay graph [Delaunay, 1934]. In short, a Delaunay graph represents a graph for a neighborhood $H(1, \mathcal{D}, M)$. A neighborhood H can also be represented by a Gabriel graph $G(V, E)$ [Gabriel/Sokal, 1969], which is a subgraph of a Delaunay graph $\mathcal{D}(V, E)$ in which two points are connected if the line segment between the two points is the diameter of a closed disc that contains no other points within it (empty ball condition). A Gabriel graph represents a graph for a neighborhood $H(1, G, M)$. Another case that is often considered is that of a neighborhood H_j*(knn, K, M),* where the number of nearest neighbors of a point *j* is defined by the number of vertices connected to this point in the *K*-nearest-neighbor graph (KNN graph), e.g., [Brito et al., 1997]. Here, we will use the shorter notation *H(knn, M).*

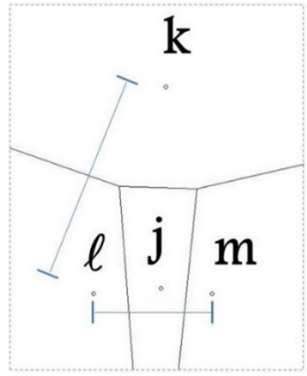

Figure 2.4: Four points and their Voronoi cells: *D(l, k)>D(l, m)* illustrate the different types of neighborhoods: unidirectional versus direction-based.

[4] Such neighborhoods *H* will prove useful for various evaluation steps, as summarized in Fig. 2.5.

Neighborhoods of points can be divided into two types, namely, *unidirectional* and *direction-based* neighborhoods. Consider the four points shown in Figure 2.4. The points *l*, *k*, *j*, and *l* are in the same neighborhood $H_l(1, \mathcal{D}, M)$ in the corresponding Delaunay graph, but the points *l* and *m* are never neighbors in this graph, even if the distance *D(l, m)* is smaller than *D(l, k)*. Thus, in this neighborhood definition, the direction information is more important than the real arrangement of the points in space as characterized by the distances *D*.
However, if a neighborhood is defined in terms of a KNN graph, then the points *l* and *m* could be in the same neighborhood $H_l(knn, K, M)$, and the points *l* and *k* could be in different neighborhoods, depending on the value of knn and on the ranking of the distances between these points. Therefore, this type of neighborhood is called unidirectional. In other words, it can be said that the points *l*, *j*, and *m* are more *dense* with respect to each other than they are with respect to *k*. Thus, unidirectional neighborhoods defined in terms of KNN graphs or unit disk graphs [Clark et al., 1990] can be used to define neighborhoods based on density.

2.3 Overview of Knowledge Discovery

> *"The term knowledge discovery in databases [...] was coined in 1989 to refer to the general process of finding knowledge in data and to emphasize the 'high-level' application of particular data mining methods" [Fayyad et al., 1996, p. 3].*

In 1996, Fayyad et al. used this term in his introduction to "From Data Mining to Knowledge Discovery" as follows:

> *"Knowledge discovery in databases is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data" [Fayyad et al., 1996, p. 6].*

Dropping the suffix *in databases*, the term *knowledge discovery* was extensively discussed in [Mörchen, 2006, pp. 6-7]. According to the definition used in that work, *knowledge discovery* is "data mining with the goal of finding knowledge, i.e., novel useful, interesting, understandable, and automatically interpretable patterns" [Mörchen, 2006, p. 7]. The definition of *data mining* as given in [Mörchen, 2006, p. 7] is

> *"The process of finding hidden information or structure in a data [...] [set.] This includes extraction, selection, preprocessing, and transformation of features describing different aspects of the data".*

The following overview in Figure 2.5 presents a possible approach to knowledge discovery, as applied in chapters 11 and 12. It is not claimed here that this view is the only approach available in this research field. The remainder of this chapter will describe the various tasks involved in knowledge discovery which are shown in Figure 2.5.

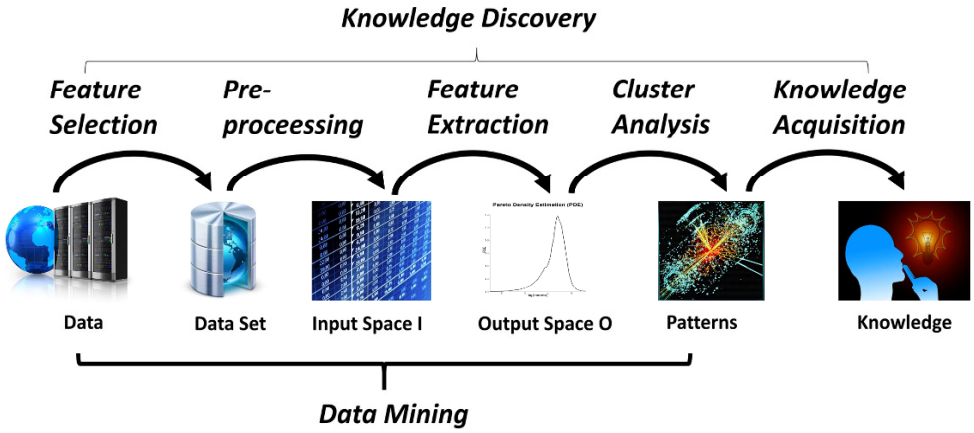

Figure 2.5: The step-wise process of knowledge discovery, as inspired by [Fayyad et al., 1996, p. 10; Ultsch, 2000b]. The systematic process may contain loops between any steps [Behnisch/Ultsch, 2015, p. 52]. This work focuses on Clustering analysis which will be separately discussed in the next chapter, but in general applying Machine learning algorithms would be the 4th step.

2.3.1 Feature Selection

In the first step, the "features must be properly selected so as to encode as much information as possible concerning the task of interest. […] minimum information redundancy among the features is a major goal" [Theodoridis/Koutroumbas, 2009, pp. 596-597] (see also [Lee/Verleysen, 2007, p. 230]). Redundancy refers to a case in which certain features of a data set are not independent of each other [Lee/Verleysen, 2007, pp. 1-2]. For example, if the two variables l and j are correlated, then $D(l,j) = \sqrt{\sum_i l_i - j_i}$ is no longer a Euclidean distance [Cormack, 1971, p. 326].

2.3.2 Preprocessing

> *"Preprocessing the data to be mined is utterly important for a successful outcome of the analysis. If the data is not cleansed and normalized, there is a high danger of getting spurious and meaningless results. Cleansing includes the removal of outliers, i.e., data objects with extreme values, replacement of missing values, or the removal of erroneous corresponding data sets" [Mörchen, 2006, pp. 7-8].*

Sometimes, this first step is already referred to as feature extraction [Bishop, 2006, p. 2]. Many data mining methods rely on the concept of (dis-)similarity between pieces of information encoded in data. For example, for Euclidean distances, "normalization of the data needs to be considered to avoid undesired emphasis of features with large ranges and variances" [Mörchen, 2006, p. 8] (see also [Jain/Dubes, 1988, p. 38]). This process of creating such "syntetic" data features that retain the most important information of a pattern in question is here called feature extraction (consistent with [Mirkin, 2005, p. 208]).

2.3.3 Feature Extraction

The first step of feature extraction is to determine the distribution of each individual variable.

> *"Important tools for this inspection are the quantile-quantile plot (QQ-plot) and kernel estimators for the probability density function (pdf). Here we use the PDE method for pdf estimation [Ultsch, 2003b] as it is specially designed to uncover subsets in the variables" [Behnisch/Ultsch, 2015, p. 54].*

A QQ-plot makes it possible to compare the given distribution of a variable to standard distributions. Additionally, box-whisker diagrams (boxplots) may be used to visualize the quartiles of a variable.

2.3.3.1 Transformations

> *"Real valued data often comes from domains where variables have greatly varying variances because of different scales. Variables with large variances are likely to dominate the obtained distance structure, e.g. when using Minkowski metrics. To overcome this problem, each variable is linearly transformed (standardized) such that the estimated variance is the same on all variables. The Z-score scheme transforms a variable's values* $x \leftarrow (x - m)/\sigma$ *with mean m and standard deviation* σ*"* [Herrmann, 2011, p. 28].

If a variable can be non-linearly transformed to a normal distribution, the Box-Cox algorithm (see [Asar et al., 2014]) is often used to estimate the factor of the transformation. With an approximation of the factor obtained from the ladder of powers [Tukey, 1977], an "understandable" transformation, e.g., "log" or "sqrt," can be applied that is as near as possible to the factor of the Box-Cox algorithm. "These allow for hypotheses on why the distribution is shaped in a particular way" [Behnisch/Ultsch, 2015, p. 56].
For non-normally distributed variables (e.g., a variable with a multimodal distribution), a meaningful variance σ^2 may be difficult to estimate. "Instead, a (robust) min/max-standardization transforms a variable's values $x \leftarrow \frac{x-min(x)}{max(x)-min(x)}$ with robust estimates $min(x)$, $max(x)$ for minimum and maximum values. There is empirical evidence by Milligan and Cooper [Milligan/Cooper, 1988] that min/max standardization is to be preferred over Z-score, especially if variances of underlying distributions is [sic] hard to estimate" [Herrmann, 2011, p. 28]. In this context, $max(x)$ and $min(x)$ are estimated as the 95th and 5th percentiles, respectively, of the distribution [Herrmann, 2011, p. 127].

2.3.3.2 Dimensionality Reduction

A common approach to feature extraction is dimensionality reduction (DR). To cope with the "curse of high dimensionality" (for further details, see [Verleysen et al., 2003]), dimensionality reduction reduces an input space I$\subset \mathbb{R}^d$ to an output space O$\subset \mathbb{R}^m$ such that $m < d$ [Lee/Verleysen, 2007].

> *"All difficulties that occur when dealing with high-dimensional data are often referred to as the 'curse of dimensionality'. When data dimensionality grows, the good and well-known properties of the usual 2D or 3D Euclidean spaces make way for strange and annoying phenomena" [Lee/Verleysen, 2007, p. 3].*

The various phenomena related to this concept are explained in [Lee/Verleysen, 2007, pp. 4-9] (see also [Bellman, 1957]). A DR method is usually either a manifold learning method or a projection method. DR methods such as autoencoders [Hinton/Salakhutdinov, 2006], Isomap [Tenenbaum et al., 2000] or local linear embedding (LLE) [Roweis/Saul, 2000] that are designed to find a manifold[5] that represents a given set of high-dimensional data[6] are called *manifold learning* methods. Such methods are disregarded here because these manifolds usually have more than two dimensions. DR methods of the type known as projection methods are

[5] "A manifold is a connected region. Mathematically, it is a set manifold of points, associated with a neighborhood around each point. From any given point, the manifold locally appears to be a Euclidean space." [Goodfellow et al., 2016, p. 160]

[6] Often described using the term *intrinsic dimension* (e.g., [Lee/Verleysen, 2007, pp. 18-24, 41, 47ff]).

separately introduced in chapter 4. There, the focus is placed on methods that attempt to visualize information by means of projections that are restricted to visualizing high-dimensional data in a two-dimensional space while preserving their structure (for details, see chapter 5). The quality of a projection critically depends on the concept of dissimilarity that is chosen to be applied to the input space *I*. This concept could be a definition based on either distance or local proximity. An index used to evaluate the quality of a projection is called a quality measure (QM), and 19 QMs are introduced in chapter 6.

2.3.4 Cluster Analysis

Many data mining methods rely on some concept of the dissimilarity between pieces of information encoded in the data of interest. These methods are used for cluster analysis, and common approaches will be described in the next chapter. Cluster analysis is the task of unsupervised classification that results in a clustering. Given a data set *I* that contains *n* data points, the objective of cluster analysis is to group the data points into *K* disjoint subsets of *I*, denoted by $c_1, \dots, c_K$ [Hennig et al., 2015, p. 2]. "A clustering is […] the partition obtained" with $\mathrm{K} = \{c_1, \dots c_K\}$. If a data point *l* belongs to a cluster c_g, then it has the class label $g \in \mathbb{N}$. In the literature, this process is often called hard clustering to distinguish it from methods such as fuzzy clustering, in which a fractional degree of membership is assigned to each $l \in I$ [Jain et al., 1999].

Cluster

No generally accepted definition of clusters exists in the literature [Hennig et al., 2015, p. 705]. When describing clusters, the term *pattern* is often used (e.g., [Theodoridis/Koutroumbas, 2009]).

Here, consistent with Bouveyron et al., it is assumed that a cluster is a group of similar objects [Bouveyron et al., 2012]. Chapter 3 will elaborate on this statement while presenting the definition of *natural* clusters.

Intracluster Distance

Let $c_p \subset I$ be a cluster such that $\forall c_q \subset I$, where $p, q \in \{1, \dots, k\}$ and $p \neq q$, $c_p \cap c_q = \{\}$; then, the distance $Intra(c_p) \coloneqq D(l, j)$ between two data points $j, l \in c_{p,}$ is called an intracluster distance.

Intercluster Distance

Let $c_p \subset I$ and $c_q \subset I$ be two clusters such that $p, q \in \{1, \dots, k\}$, $c_p \cap c_q = \{\}$, and $p \neq q$; then, the distance $Inter(c_p, c_q) = \mathrm{D}(j, l)$ between two data points j and l in the two clusters, $j \in c_p$ and $l \in c_q$, is called an intercluster distance.

Compact Structures

Compact structures in a data set are mainly defined by distances *d* if discontinuity in data exist such that the intracluster distances are small and the intercluster distances are large. Note, that the distance distribution is often bimodal if the data structures are compact. This type of structures leads to natural clusters (see chapter 3).

Connected Structures

Connected structures in a data set are mainly defined by density $\rho(\vec{v})$ if discontinuity in data exist. If a connected graph Γ is chosen appropriately regarding the data set, these data structures are based on neighborhoods $H_j\ (k, \Gamma, M)$. This type of structures leads to natural clusters (see chapter 3).

2.3.5 An Approach to Knowledge Acquisition

If, for a given data set, there exist labels defined by a clustering or a domain expert, the next step may be to determine what each cluster means [Behnisch/Ultsch, 2015, p. 65] or what kind of knowledge can be acquired from it[7].

> *"Under knowledge we understand a symbolic representation of objects, facts and rules for an interpreter with symbol processing capability, e.g. a human[8]. In particular, knowledge is communicable by word or writing" [Ultsch, 1994, p. 1] (see also [Ultsch, 1987, p. 22]).*

Knowledge has the properties of being valid, comprehensible, nontrivial, potentially innovative and useful in practice [Behnisch/ Ultsch, 2015, p. 52]. It can be stored in a knowledge base, which "is an organized collection of knowledge together with operations for accessing and manipulating knowledge" [Ultsch, 1987, p. 22]. One example of a representation of knowledge is a rule [Ultsch, 2016c], which is defined as a prescription regarding how to generate, interpret and manipulate facts [Ultsch, 1987, p. 22].

In the context of knowledge discovery, knowledge acquisition can be defined "as the encoding of knowledge into the formal representation scheme of a knowledge-based system [KBS]" [Ultsch, 1987, p. 23]); here, a KBS is defined as "a computer program that contains an explicit, formal representation of knowledge in a knowledge base and is capable of [drawing conclusions[9]]" [Ultsch, 1987, p. 23]. In another context, researchers may interview domain experts "to become educated about the domain and to elicit the required knowledge, in a process called knowledge acquisition" [Russell et al., 2003, p. 217]. In short, knowledge acquisition can be described as a process that leads to a formal representation of knowledge (see [Aikins, 1983]), for example, a process leading to the generation of rules required for a computer program, e.g., DENDRAL [Russell et al., 2003, p. 22] or MYCIN [Aikins, 1983]. One possible approach to knowledge acquisition is to use machine learning [Russell et al., 2003, p. 687]. With regard to understandability, the machine learning methods used for this purpose can be classified as either symbolic or sub-symbolic methods [Ultsch/Korus, 1993].

> *"Sub-symbolic methods model the structure of data using many numerical parameters. They are usually aimed at prediction or classification. The output of sub-symbolic methods often depends on the values and interactions of most or all model parameters. They fail to explain the prediction or classification. There are certainly areas of data mining where it is sufficient to build such black-box models that can approximately reproduce a classification or predict future data. An important requirement for knowledge discovery is the interpretability of the results. In many domains the expert wants to know why a decision was made or what a [...] pattern describes. Comprehensible descriptions of the models are crucial for success in this case"* [Mörchen, 2006, p. 120].

For the acquisition of knowledge through cluster analysis, symbolic methods are preferable, as described in chapters 11 and 12 (see also [Ultsch, 1994]). In chapter 12, decision tree learning

---

[7] In another context one would like to explain a prediction done by a machine learning algorithm.

[8] For humans 7±2 rules appear to be the optimum [Miller 1956].

[9] Formally defined as *inference* in [Ultsch, 1987, p. 22].

is used in a knowledge acquisition approach called Classification And Regression Tree (CART) analysis [Breiman et al., 1984]). This method relies on a binary tree in which the splitting criteria (decisions) for the vertices are expressed in terms of the Gini index (for further details, see [Safavian/Landgrebe, 1990, p. 15]).

"A class is described by a number of conditions" [Ultsch/Korus, 1993, p. 3] that lead to the generation of a subset $G_i \subset C$ defined by a previously identified clustering. Additionally, for each class, a unique class label $g \in \mathbb{N}$ exists for all $o \in G_i$. Every observation $o \in G_i$ can be unambiguously described by one or more properties that are shared among all observations of G_i. Here, the conclusion that an observation can be correctly assigned to a class G_i is reached based on the conditions defining a path (rule) from the corresponding leaf to the root of the binary tree, and this conclusion is called the decision to place o in G_i. Therefore, the class G_i has a semantic characterization because it is characterized by the rules governing the decision tree, which allow this class to be distinguished from other classes. Here, it is assumed that the last step in the evaluation of a clustering is to ask domain experts to validate the identified classes.