Agenda

1. Introduction

2. Basics
   2.1 Data Representation
   2.2 Data Reduction
   2.3 Visualization
   2.4 Privacy

3. Unsupervised Methods

4. Supervised Methods

5. Advanced Topics
Objects and Attributes

Entity-Relationship Diagram (ER)

- **Student**
  - **name**
  - **skills**
  - **semester**
  - **major**

UML Class Diagram

```
Student
- name
- semester
- major
- skills
```

Data Tables (Relational Model)

<table>
<thead>
<tr>
<th>name</th>
<th>sem</th>
<th>major</th>
<th>skills</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ann</td>
<td>3</td>
<td>CS</td>
<td>Java, C, R</td>
</tr>
<tr>
<td>Bob</td>
<td>1</td>
<td>CS</td>
<td>Java, PHP</td>
</tr>
<tr>
<td>Charly</td>
<td>4</td>
<td>History</td>
<td>Piano</td>
</tr>
<tr>
<td>Debra</td>
<td>2</td>
<td>Arts</td>
<td>Painting</td>
</tr>
</tbody>
</table>
### Overview of (Attribute) Data Types

#### Simple Data Types
- Numeric/metric, Categorical/nominal, ordinal

#### Composed Data Types
- Sets, sequences, vectors

#### Complex Data Types
- Multimedia: Images, videos, audio, text, documents, web pages, etc.
- Spatial, geometric: Shapes, molecules, geography, etc.
- Structures: Graphs, networks, trees, etc.
Simple Data Types: Numeric Data

Numeric Data

- Numbers: natural, integer, rational, real numbers
- Examples: age, income, shoe size, height, weight
- Comparison: difference
- Example: 3 is more similar to 30 than to 3,000
Simple Data Types: Categorical Data

- "Just identities"
- Examples:
  - occupation = \{ butcher, hairdresser, physicist, physician, ... \}
  - subjects = \{ physics, biology, math, music, literature, ... \}
- Comparison: How to compare values?
  - Trivial metric:
    \[
    d(p, q) = \begin{cases} 
      0 & \text{if } p = q \\
      1 & \text{else} 
    \end{cases}
    \]
  - Generalization hierarchy: Use path length

```
basics
  data representation
  november 2, 2018
```
**Metric Space**

Metric space \((O, d)\) consists of object set \(O\) and *metric distance* function \(d : O \times O \rightarrow \mathbb{R}^{\geq 0}\) which fulfills:

- **Symmetry:** \(\forall p, q \in O : d(p, q) = d(q, p)\)
- **Identity of Indiscernibles:** \(\forall p, q \in O : d(p, q) = 0 \iff p = q\)
- **Triangle Inequality:** \(\forall p, q, o \in O : d(p, q) \leq d(p, o) + d(o, q)\)

**Example:** Points in 2D space with Euclidean distance
Simple Data Types: Ordinal

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transitivity:</td>
<td>( \forall p, q, o \in O : p \leq q \land q \leq o \implies p \leq o )</td>
</tr>
<tr>
<td>Antisymmetry:</td>
<td>( \forall p, q \in O : p \leq q \land q \leq p \implies p = q )</td>
</tr>
<tr>
<td>Totality:</td>
<td>( \forall p, q \in O : p \leq q \lor q \leq p )</td>
</tr>
</tbody>
</table>

Examples

- Words & lexicographic ordering: \textit{high} \leq \textit{highschool} \leq \textit{highscore}
- (Vague) sizes: \textit{tiny} \leq \textit{small} \leq \textit{medium} \leq \textit{big} \leq \textit{huge}
- Frequencies: \textit{never} \leq \textit{seldom} \leq \textit{rarely} \leq \textit{occasionally} \leq \textit{sometimes} \leq \textit{often} \leq \textit{frequently} \leq \textit{regularly} \leq \textit{usually} \leq \textit{always}
Composed Data Types: Sets

**Characteristic**

Unordered collection of individual values

**Example**

- skills = { Java, C, Python }

**Comparison**

- Symmetric Set Difference:
  \[
  R \Delta S = (R - S) \cup (S - R) = (R \cup S) - (R \cap S)
  \]

- Jaccard Distance:  \( d(R, S) = \frac{|R \Delta S|}{|R \cup S|} \)
Composed Data Types: Sets

Bitvector Representation

▶ Given a set $S$, an ordered base set $B = (b_1, \ldots, b_n)$, create binary vector $r \in \{0, 1\}^n$ with $r_i = 1 \iff b_i \in S$.

▶ Hamming distance: Sum of different entries (equals cardinality of symmetric set difference)

Example

▶ Base: $B = \{\text{Math, Physics, Chemistry, Biology, Music, Arts, English}\}$
▶ $S = \{\text{Math, Music, English}\} = (1,0,0,0,1,0,1)$
▶ $R = \{\text{Math, Physics, Arts, English}\} = (1,1,0,0,0,1,1)$
▶ $\text{Hamming}(R, S) = 3$
Composed Data Types: Sequences, Vectors

### Characteristic

- Put $n$ values of a domain $D$ together
- Order does matter: $I_n \rightarrow D$ for an index set $I_n = \{1, \ldots, n\}$

### Examples

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Formula</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Simple) sum</td>
<td>$d_1(o, q) = \sum_{i=1}^{n}</td>
<td>o_i - q_i</td>
</tr>
<tr>
<td>Root of sum of squares</td>
<td>$d_2(o, q) = \sqrt{\sum_{i=1}^{n} (o_i - q_i)^2}$</td>
<td>(Euclidean)</td>
</tr>
<tr>
<td>Maximum</td>
<td>$d_3(o, q) = \max_{i=1}^{n}</td>
<td>o_i - q_i</td>
</tr>
<tr>
<td>General formula</td>
<td>$d_4(o, q) = \sqrt[p]{\sum_{i=1}^{n}</td>
<td>o_i - q_i</td>
</tr>
<tr>
<td>Weighting of dimensions</td>
<td>$d_5(o, q) = \sqrt[p]{\sum_{i=1}^{n} w_i \cdot</td>
<td>o_i - q_i</td>
</tr>
</tbody>
</table>
Complex Data Types

Components

- Structure: graphs, networks, trees
- Geometry: shapes/contours, routes/trajectories
- Multimedia: images, audio, text, etc.

Similarity models: Approaches

- Direct measures – highly data type dependent
- Feature engineering – explicit vector space embedding with hand-crafted features
- Feature learning – explicit vector space embedding learned by machine learning model, e.g. neural network
- Kernel trick – implicit vector space embedding
## Complex Data Types

### Examples for similarity models

<table>
<thead>
<tr>
<th></th>
<th>Direct</th>
<th>Feature engineering</th>
<th>Feature learning</th>
<th>Kernel-based</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Graphs</strong></td>
<td>Structural Alignment</td>
<td>Degree Histograms</td>
<td>Node embeddings</td>
<td>Label Sequence Kernel</td>
</tr>
<tr>
<td><strong>Geometry</strong></td>
<td>Hausdorff Distance</td>
<td>Shape Histograms</td>
<td>Spectral Neural Network</td>
<td>Spatial Pyramid Kernel</td>
</tr>
<tr>
<td><strong>Sequences</strong></td>
<td>Edit Distance</td>
<td>Symbol Histograms</td>
<td>Recurrent neural network (RNN)</td>
<td>Cosine Distance</td>
</tr>
</tbody>
</table>
Feature Extraction

- Objects from database DB are mapped to feature vectors

Feature vector space
- Points represent objects
- Distance corresponds to (dis-)similarity
Similarity Queries

- Similarity queries are basic operations in (multimedia) databases
- Given: Universe \( O \), database \( DB \), distance function \( d \) and query object \( q \)

**Range query**

Range query for range parameter \( \epsilon \in \mathbb{R}_0^+ \):

\[
range(DB, q, d, \epsilon) = \{ o \in DB \mid d(o, q) \leq \epsilon \}
\]

**Nearest neighbor query**

\[
NN(DB, q, d) = \{ o \in DB \mid \forall o' \in DB : d(o, q) \leq d(o', q) \}
\]
Similarity Queries

**k-nearest neighbor query**

$k$-nearest neighbor query for parameter $k \in \mathbb{N}$:

$$NN(DB, q, d, k) \subset DB \text{ with } |NN(DB, q, d, k)| = k \text{ and }$$

$$\forall o \in NN(DB, q, d, k), o' \in DB - NN(DB, q, d, k) : d(o, q) \leq d(o', q)$$

**Ranking query**

Ranking query (partial sorting query): "get next" functionality for picking database objects in an increasing order w.r.t. their distance to $q$:

$$\forall i \leq j : d(q, rank_{DB,q,d}(i)) \leq d(q, rank_{DB,q,d}(j))$$
Similarity Search

▶ Example: Range query \( \text{range}(DB, q, d, \epsilon) = \{o \in DB \mid d(o, q) \leq \epsilon\} \)

▶ Naive search by sequential scan
  ▶ Fetch database objects from secondary storage (e.g. disk): \( O(n) \)
  ▶ Check distances individually: \( O(n) \)

▶ Fast search by applying database techniques
  ▶ Filter-refine architecture
    ▶ Filter: Boil database \( DB \) down to (small) candidate set \( C \subseteq DB \)
    ▶ Refine: Apply exact distance calculation to candidates from \( C \) only

▶ Indexing structures
  ▶ Avoid sequential scans by (hierarchical or other) indexing techniques
  ▶ Data access in (fast) \( O(n) \), \( O(\log n) \) or even \( O(1) \)
Filter-Refine Architecture

▶ Principle of multi-step search:
  1. Fast filter step produces candidate set $C \subset DB$ (by approximate distance function $d'$)
  2. Exact distance function $d$ is calculated on candidate set $C$ only.

▶ Example: Dimensionality reduction

▶ ICES$^b$ criteria for filter quality
  - Indexable – Index enabled
  - Complete – No false dismissals
  - Efficient – Fast individual calculation
  - Selective – Small candidate set

---

$^a$GEMINI: Faloutsos 1996; KNOP: Seidl & Kriegel 1998

$^b$Assent, Wenning, Seidl: ICDE 2006
Organize data in a way that allows for fast access to relevant objects, e.g. by heavy pruning.

R-Tree as an example for spatial index structure:

- Hierarchy of minimum bounding rectangles
- Disregard subtrees which are not relevant for the current query region
Indexing

- Example: Phone book
- Indexed using alphabetical order of participants
- Instead of sequential search:
  - Estimate region of query object (interlocutor)
  - Check for correct branch
  - Use next identifier of query object
  - Repeat until query is finished

![Indexing Diagram](Image source: hierher/flickr, Licence: CC BY 2.0)