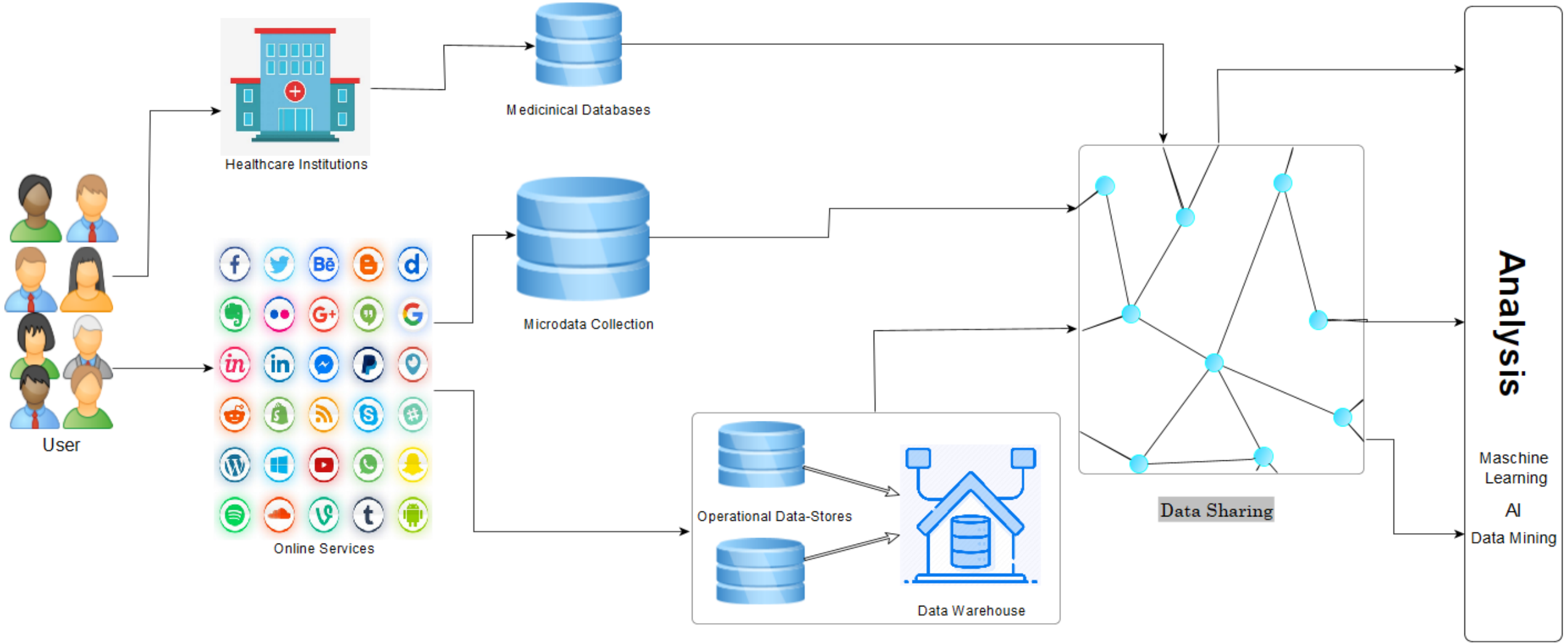
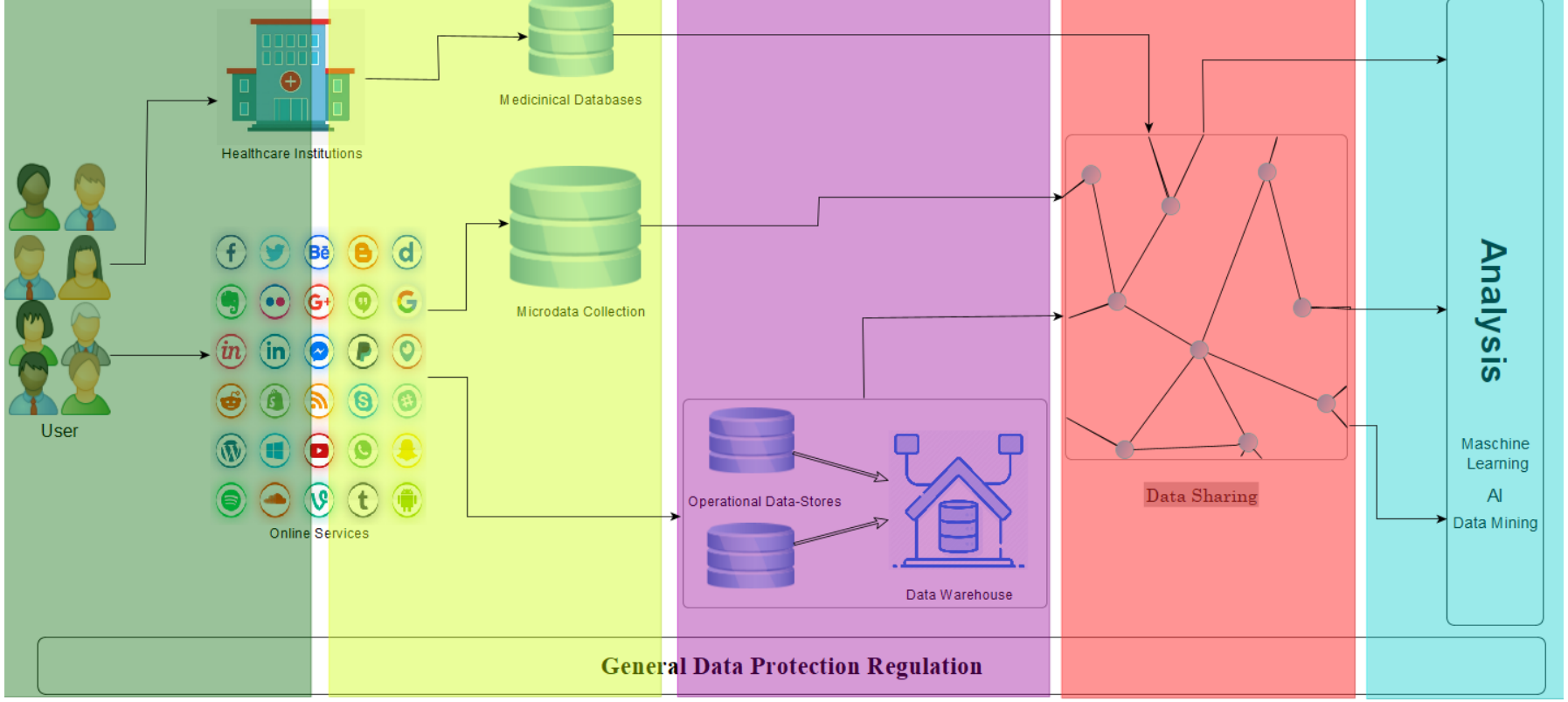
Chapter 4:
Privacy Risks and Anonymisation Techniques

The Phases of Data Processing





The data processing procedure can be roughly divided into 5 steps:

User Interaction:

Action:

- Browsing a website
- Using an App
- Visiting a doctor

Consequence:

- Implicit or explicit consent on privacy policy

Data collection:

Examples:

- Visited links/timestamps
- IP address/Port
- Screen/Font size
- Browser type/plugins
- Scroll speed/patterns
- Medical diagnosis/image

Everything is saved internally by collecting entity

Warehousing/Mining:

- Digital fingerprinting
- User preferences
- User habits
- Classifying/grouping

Information Exchange:

- Information business
- Sharing knowledge about costumer groups
- Reprocessing foreign with internal data
- Transfering medical data between specialists

Analysis:

- Targeted advertising
- Design optimizations
- Corporate decisions
- Product creation
- Research
- Treatment decisions in medicin

Privacy Concerns

- Most steps mentioned are regulated by the EUs General Data Protection Regulation with resulting rights for users and obligations for companies (e.g. fingerprinting is only allowed when a service is provided that uses the collected data)
- For data collectors, serving a wide variety of local regulations and policies while providing a uniform global service and sharing data with international buisness partners is a delicate balancing act for companies
- The treatment of medical data is subject to further directives by local legislation and ethic commitiees, due to the longer history and independence from cyberspace, common practice in data privacy is already established
- Resulting privacy concerns in the data processing procedure are:

User Interaction:
- Transparent information on data processing
- Subsequent consent on processing policy
- Right to receive personal data collected

Data collection:
- Collection must be in harmony with:
  - User consent
  - Minimality demand
- Metadata on collection for later reconstruction

Warehousing/Mining:
- Combining different data sources with different privacy policies
- Securing historical data against breaches
- Keeping track of data lineage

Information Exchange:
- Need for anonymisation of shared/published sets
- International companies need to handle national privacy policies simultaneously

Analysis:
- Respecting privacy policies of target groups/users

4.1
Privacy Models

Privacy-Preservation Technologies in Information Systems

Dr. Armin Gerl

WS 2021/2022

Why Privacy Models?

- We have now seen various basic methods and transformations on microdata that can be used to anonymise microdata
- Each method has its own strengths and weaknesses depending on the types of data they are used on
- Anonymization Methods alone are not sufficient to protect data -> Privacy Models needed
- Privacy Models are the tools used to achieve certain privacy goals that an anonymised data set has to meet
- In order to messure the effectiveness of the performed transformations on a data set with respect to privacy, researchers developed a wide variety of models to solidify these goals
- The most common ones are based on the attribute classifications (EI, QI, SA, NSA) that were mentioned throughout the lecture, even though other approaches exist
- In the following chapter, we will take a look at both families of privacy models, starting with the classification based ones

Overview of Privacy Models

- A wide variety of transformations to the records of the data set provides protection against privacy disclosure

Non-perturbative Masking

- Detail reductions and partial suppressions without altering the original records
- Suitable where the „truthfulness“ of single records/attributes is required, e.g., health-care

Perturbative Masking

- Distortions on the original records that mostly preserve the derived statistics of the data set
- Suitable if truthfulness data-set properties is required, e.g., analysis of shopping behaviour

- Perturbative and non-perturbative methods can be combined in order to achieve an optimal balance between disclosure risk minimization and accuracy

Literature: „Database Anonymisation” by Josep Domingo-Ferrer, David Sánchez, Jordi Soria-Comas, 2016, ISBN:9781627058438

4.2
Non-perturbative Masking

Privacy-Preservation Technologies in Information Systems

Dr. Armin Gerl

WS 2021/2022

Non-perturbative Masking

- Goal of Non-pertubative Masking:
  - Preserving the truthfullness of the original entries but making them less specific

- Used Methods to reduce information level:
  - Generalization
  - Suppression
  - Deletion
  - Typically applied on EI, QI, optional SA

- We start with basic concepts/approaches and continue with detailing privacy models

Sampling

- A simple approach that can be applied to any data set, but in practice is only suitable for categorical attributes
- Instead of the original data set X, a subset of records $S \subset \mathrm{X}$ is selected and published
- Since the released records are in its original form, special care must be given to the selection process
- If external matching is deemed unlikely, the non-interference of the method with the records that are published might be a desired property
- When only a low fraction of records are sufficient to carry the point of the released microdata, sampling can be considered for protection
- Although the records themselves are released unperturbed, the selection process allows some control over what information is disclosed about the specific attribute values (inferencial disclosure)

Sampling Requirements

- Before releasing the chosen sample, the availability of external records for each attribute must be taken into consideration
- Chosen sample must carry over the statistical properties of the original if used as source data, seperate computations of those must be taken on *X* and *S* in addition to measuring information loss

! Needless to say, sampling to adjust statistics infered from the source data is deception

- This might still happen unconsciously on selection if not checked properly
- Example cases of sampling might include:
  - If the microdata contains respondents who agreed to publish their information and no statistical properties need to be infered
  - If the subject of the microdata is specific enough that a model of the attackers background knowledge can be assumed with enough certainty

Sampling Limitations

- For categorical data, a sample unique is unlikely to be a population unique leading to a lower external matching risk but is not sufficient to guarantee confidentiality
- Sampling could also be applied to continuous microdata, though a combination with secondary anonymisation methods is highly recommended to avoid the following problem:
  - The probability of a continuous attribute taking the same value for different records is a lot lower than for a categorical one
  - Therefor, with unaltered values of the original attributes are published, it is substantially more likely that a unique match with an external file occurs

Sideline:

There is some confusion in literature when it comes to the attribute classification „continuous“: Striktly speaking, any attribute persisted in digital form is subject to truncation/rounding and isn‘t truly continuous. Yet in our case, most properties connected to continuous attributes also apply to numerical attributes with a sufficiently large value range

Sampling Use Case: Census Data

- Gouvernment bureaus conducting state-wide surveys are a special case when it comes to disclosure risk assessment:
  - Certain risk measures like „Uniqueness“ that use sample weights and are based on the uncertainty of a correct match are not applicable, since a sample unique automatically is a population unique
  - That means any matching attempts by an attacker that are successful, are without a doubt correct
  - In fact, census data often provides the basis for those types of disclosure risk assessments
- Due to the large scale of the collected data, disclosure of the released microdata would not only affect a lot of people but also destroy their trust in the gouvernment
- To limit the possible damage a successful attack on the anonymized microdata could do, only a very small portion of the census is released to the public
- Sampling is used as a first step with additional masking methods applied on top

Sampling Practices

- In general, sampling as a standalone masking technique does not imply confidentiality of the released data under most privacy standards
- In practice, it is invoked pior to other anonymisation methods when:
  - Lowering disclosure risk is mandatory at any cost
  - Disclosure risk assessment is insufficient
  - The microdata set covers a large enough fraction of the population to still be useful enough when sampled
- When it comes to census data, institutions will often release samples of <5% to the public and increasing fractions with additional attributes to registered and verified researchers
- For Example: The British Office for National Statistics (ONS) 2011 Census:
  - A 1% sample with limited attributes was released for educational porposes
  - Two 5% samples are available for researchers who accept the terms and use conditions
  - Two 10% samples reside in controlled research data centers with approved reserchers and research goals
  - All files have undergone additional anonymisation after sampling

Sampling Evaluation

Pros:

- ✓ Doesn’t perturb any records
- ✓ Offers some control over inferential disclosure risk
- ✓ Offers beforehand damage control

Cons:

- ○ Unsuitable as a standalone method
- ○ Limited disclosure control in general
- ○ Difficult preservation of statistical properties if dataset is small
- ○ Attack model and subject knowledge necessary

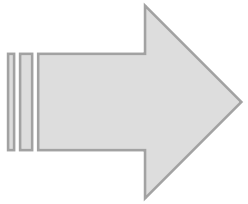
Recoding (Generalisation)

- Applicable to categorical and continuous attributes, generalisation provides a precision tool for protecting records with low frequency attribute values (and value ranges) against disclosure
- When applied to categorical attributes, values with low representation across all records get merged into other categories:
  - Example record: Unusual attribute combination of „Martial Status = Widow/er“ and „Age = 18“ yields high reidentification risk
  - Generalizing „Martial Status“ category „Widow/er“ to „Widow/er or divorced“ merges the two, creating a broader category and reducing reidentification risk by lowering the probability of the record being unique
- When applied to a continuous (or numerical) attribute, the attribute is discretized, meaning singular data points are generalized to intervals, transforming the attribute into a categorical one:
  - Example record: High precision measurements „Weight = 121,2345kg“ and „Body fat = 2,789%“, where attackers can easily infere the respondant is a professional body builder and match with competition data
  - This inference is made almost impossible if intervals „Weight = 100-130kg“ and „Body fat = 0-10%“ are used

Top and Bottom Coding

- A special case of global recoding, which can be used on ordinal attributes (continuous and categorical)
- After ranking the attribute values, those above (and/or below) a certain threshold are lumped together to form a new category
- Low effort with decent reward, since extreme values are scarce but tend to cluster at the edges in many cases

Sex	Age	Education
f	20	secondary
f	20	tertiary
m	18	secondary
f	23	postgrad.
m	18	secondary
f	21	tertiary
m	25	postgrad.
m	19	secondary
f	23	tertiary



Sex	Age	Education
f	20	secondary
f	20	tertiary and above
m	18	secondary
f	21+	tertiary and above
m	18	secondary
f	21+	tertiary and above
m	21+	tertiary and above
m	19	secondary
f	21+	tertiary and above

Top coding of ages ≥ 20 and tertiary with postgraduate education levels

Top and Bottom Coding II

- Since global recoding often cuts too deep into the analytical power of the microdata set, top and bottom coding takes advantage of typical distributions often present in real world microdata
- When the bulk of the values lies in the center of the attribute distribution with the peripheral categories/values being only scarcly represented, this generalisation approach protects the few outliers by increasing their representation frequency while preserving statistical information in the center
- With the majority of values left unperturbed by this method, additional disclosure control might be needed to ensure confidentiality
- Top and bottom coding is often just one step in the anonymisation process

Global Recoding

- Global Recoding: Mapping domains of quasi-identifiers to generalized or altered values using a single function
- Merging of categories or discretization into intervals both seek to reduce the possible number of values an attribute can hold
- Provides decrease in external matching risk due to broader categories

Formal Definition

- Notation D_{xi} is the domain of attribute Xi in table T
- Single Dimensional
  - ϕ_i : Dxi -> D' for each attribute Xi of the quasi-id
  - ϕ_i applied to values of Xi in tuple of T
- Multi-Dimensional
  - Recode domain of value vectors from a set of quasi-identifier attributes
  - ϕ : Dx1 x … x Dxn -> D'
  - ϕ applied to vector of quasi-identifier attributes in each tuple in T

Global Recoding Example

Patient Data

Quasi Identifiers: Age, Sex, Zicode

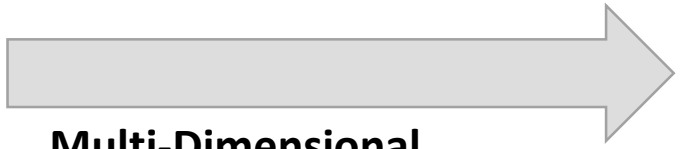
Age	Sex	Zipcode	Disease
25	M	53711	Flu
25	F	53712	Hepatitis
26	M	53711	Bronchitis
27	M	53710	Broken Arm
27	F	53712	AIDS
28	M	53711	Hang Nail

Single-Dimension Partitions:

Age: {[25-28]}

Sex: {M, F}

Zipcode: {[53710-53711], 53712}



Multi-Dimensional Partitions:

{Age: [25-26],Sex: M, Zip: 53711}

{Age: [25-27],Sex: F, Zip: 53712}

{Age: [27-28],Sex: M, Zip: [53710-53711]}

Age	Sex	Zipcode	Disease
[25-28]	M	[53710-53711]	Flu
[25-28]	F	53712	Hepatitis
[25-28]	M	[53710-53711]	Bronchitis
[25-28]	M	[53710-53711]	Broken Arm
[25-28]	F	53712	AIDS
[25-28]	M	[53710-53711]	Hang Nail

Age	Sex	Zipcode	Disease
[25-26]	M	53711	Flu
[25-27]	F	53712	Hepatitis
[25-26]	M	53711	Bronchitis
[27-28]	M	[53710-53711]	Broken Arm
[25-27]	F	53712	AIDS
[27-28]	M	[53710-53711]	Hang Nail

Recoding Evaluation

Pros:

- ✓ Solid identity disclosure safety for affected records
- ✓ Hides rare record combinations “in plain sight”

Cons:

- ○ High information loss for continuous/numerical values
- ○ Requires subject knowledge to use effectively
- ○ Only the global version provides some standalone safety

Local Suppression

- In some sense, the inverse principle to sampling, but more flexible as attributes are targeted
- Mostly effective for categorical data
- Certain values of attributes are suppressed
- Goal is to increase the number of records that agree on a combination of values

- Which records yield high reidentification risks is decided mostly by hand, although toolsets exists that ease the blackening process (e.g. sdcMicroGUI)
- Recoding and local suppression often target the same records
- Like recoding, local suppression is a precision tool that can be invoked in combination with other anonymisation techniques

Properties of Local Suppression

- Despite the similar records that spark the decision to use recoding or suppression, the 2 methods differ in granularity of their effects on the data set
- “Global” versus “Local”:
  - Although merging categories might seem like a less invasive measure than suppression, global recording affects all records that hold any of the 2 attribute values to be merged, greatly increasing information loss
  - Local suppression can be applied independently to different attributes across different records
- A combination of values in the data set that yields a high re-identification risk (referred to QI* or QI-groups), can be modified by local suppression by crossing out different components of the group in different records where it occurs, minimizing lost information
- Which parts of the key are deemed “safe” is subject dependent

Local Suppression Example

- The value combination of {“female”, “rural”, “higher”} in record 1 is a sample unique and therefor at risk of disclosure
- Recoding/generalizing one or more attributes would make these already broad categories almost universal
- Local suppression is used to precisely suppress one of the values of the unsafe combination, leaving the other entries unperturbed
- By suppressing either the value “female” or “higher”, the respondent cannot be distinguished from the other respondents anymore
- When all possible targets are equally important to the subject, usually the attribute that has the largest value range is replaced by the “NA/missing” placeholder, maximizing the number of possible combinations

Variable	Before local suppression			After local suppression		
ID	Gender	Region	Education	Gender	Region	Education
1	female	rural	higher	female	rural	NA/missing
2	male	rural	higher	male	rural	higher
3	male	rural	higher	male	rural	higher
4	male	rural	higher	male	rural	higher
5	female	rural	lower	female	rural	lower
6	female	rural	lower	female	rural	lower
7	female	rural	lower	female	rural	lower

Example from the sdcMicro Practice Guide: https://sdcpractice.readthedocs.io/en/latest/anon_methods.html#local-suppression

Local Suppression Practices

- Though the idea of crossing out targeted attribute values could also be done with continuous data, its effecteness is generally limited:
  - Given the value range, every instance of a continuous variable could be classified as unique or rare
  - When key combinations contain categorical and continuous attributes, it almost always makes more sense to suppress an identifying category to lower the risk
- The presence of a missing value will alert the eye of statistical spies/attackers, the inferences drawn from this are entirely subject dependent and difficult to model beforehand
- Local suppression is most commonly used as a last step on already transformed datasets, in order to satisfy a certain privacy model with minimal additional modification
- For example: The µ-Argus SDC package combines initial global recoding with subsequent local suppression to achieve *k*-anonymity (a Privacy Model)

Local Suppression Evaluation

Pros:

- ✓ Information loss kept to a minimum
- ✓ Useable as last step to complement other methods for precision adjustments

Cons:

- ○ Generally unfit for continuous data
- ○ Subject knowledge is required
- ○ The missing attribute might allow some inferential disclosure in some cases
- ○ Relies on modelled background knowledge of the attacker

Privacy Model Motivation

- How can we ensure that the individuals‘ privacy is protected?
- Consider the following scenario: An attackers gains access to a data set, that consists of only sensitive and quasi-identifiers
- Using background information, he might be able to link the sensitive information through an external matching attempt with the quasi-identifiers to certain individuals
  - For example: Postal code and Age might be sufficient to uniquely identify all inhabitants of a small village without knowing any names or other information
- **Record Linkage Attack: The task of finding records in data set(s) that refer to the same real world entity**
- The trivial case of matching by comparing identifiers is usually excluded from the problem definition, this is solved by matching algorithms
- Matching quasi-identifiers onto a small group doesn’t lead to unique identification anymore, as soon as two entries posses identical values for the attribute in question
- This makes it impossible to distinguish between the two

Age	Sex	Zipcode	Disease
25	M	53711	Flu
25	F	53712	Hepatitis
26	M	53711	Bronchitis
27	M	53710	Broken Arm
27	F	53712	AIDS
28	M	53711	Hang Nail

k-anonymity (cont.)

- This desired state of multiple values for an attribute being present within a data set is the core of k-anonymity
- Based on the classification of attributes introduced at the beginning:
  - Explicit identifiers are deleted
  - Quasi-identifiers are retained, but clustered into hierarchies
  - Sensitive attributes are left in original form
  - Non-sensitive attributes pose no risk and are unperturbed
- If, in our village example, every individual would have a duplicate value for each quasi-identifier in the data set, a state of 2-anonymity is reached
- The more identical entries exist for each individual, the lower the risk for identification of an individual
- In order to obtain these attribute states within the data, generalization is often performed on the quasi-identifiers:
  - Transform dates into time-intervals
  - Key codes (e.g. postal code or IP-address) are blurred step-by-step (192.168.1.127 -> 192.168.1.* or 94032 –> 9403*)

k-anonymity

- k = 2
- Quasi-identifiers
  - Zipcode
  - Age

Each 2 entries with

[25-26]	M	53711
[25-27]	F	53712
[27-28]	M	[53710-53711]

Age	Sex	Zipcode	Disease
25	M	53711	Flu
25	F	53712	Hepatitis
26	M	53711	Bronchitis
27	M	53710	Broken Arm
27	F	53712	AIDS
28	M	53711	Hang Nail

What to anonymize and in which way?

Age	Sex	Zipcode	Disease
[25-26]	M	53711	Flu
[25-27]	F	53712	Hepatitis
[25-26]	M	53711	Bronchitis
[27-28]	M	[53710-53711]	Broken Arm
[25-27]	F	53712	AIDS
[27-28]	M	[53710-53711]	Hang Nail

Greedy Partitioning Algorithm

- Problem
  - Need an algorithm to find multi-dimensional partitions
  - Optimal k-anonymous strict multi-dimensional partitioning is **NP-hard**
- Solution
  - Use a greedy algorithm
  - Based on k-d trees
  - Complexity O(n logn)

Mondrian - Greedy Partitioning Algorithm

Anonymize($partition$)
  if (no allowable multidimensional cut for $partition$)
    return $\phi : partition \rightarrow summary$
  else
    $dim \leftarrow$ choose_dimension()
    $fs \leftarrow$ frequency_set($partition$, dim)
    $splitVal \leftarrow$ find_median(fs)
    $lhs \leftarrow \{t \in partition : t.dim \leq splitVal\}$
    $rhs \leftarrow \{t \in partition : t.dim > splitVal\}$
    return Anonymize(rhs) $\cup$ Anonymize(lhs)

Example: Iteration #1 (Full table)

Partition

Age	Sex	Zipcode	Disease
25	M	53711	Flu
25	F	53712	Hepatitis
26	M	53711	Bronchitis
27	M	53710	Broken Arm
27	F	53712	AIDS
28	M	53711	Hang Nail

1) dim = Zipcode

2) Calculate Frequency Set fs (dim = Zipcode)

Zipcode	Count
53710	1
53711	3
53712	2

3) Find Median of fs
splitVal = 53711

4) LHS

Age	Sex	Zipcode	Disease
25	M	53711	Flu
26	M	53711	Bronchitis
27	M	53710	Broken Arm
28	M	53711	Hang Nail

4) RHS

Age	Sex	Zipcode	Disease
25	F	53712	Hepatitis
27	F	53712	AIDS

Example: Iteration #2 (LHS from Iteration #1)

Partition

Age	Sex	Zipcode	Disease
25	M	53711	Flu
26	M	53711	Bronchitis
27	M	53710	Broken Arm
28	M	53711	Hang Nail

1) dim = Age

2) Calculate Frequency Set fs (dim = Age)

Zipcode	Count
25	1
26	1
27	1
28	1

3) Find Median of fs
splitVal = 26

4) LHS

Age	Sex	Zipcode	Disease
25	M	53711	Flu
26	M	53711	Bronchitis

4) RHS

Age	Sex	Zipcode	Disease
27	M	53710	Broken Arm
28	M	53711	Hang Nail

Example: Iteration #3 and #4

#3: LHS from Iteration #2
Partition

Age	Sex	Zipcode	Disease
25	M	53711	Flu
26	M	53711	Bronchitis

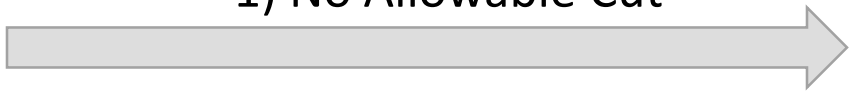
1) No Allowable Cut

Summary:
Age = [25-26] Zip = [53711]

#4: RHS from Iteration #2
Partition

Age	Sex	Zipcode	Disease
27	M	53710	Broken Arm
28	M	53711	Hang Nail

1) No Allowable Cut



Summary:
Age = [27-28] Zip = [53710 - 53711]

Example: Iteration #1 (Full table)

#5: RHS from Iteration #1
Partition

Age	Sex	Zipcode	Disease
25	F	53712	Hepatitis
27	F	53712	AIDS

1) No Allowable Cut

Summary:
Age = [25-27] Zip = [53712]

Age	Sex	Zipcode	Disease
25	M	53711	Flu
25	F	53712	Hepatitis
26	M	53711	Bronchitis
27	M	53710	Broken Arm
27	F	53712	AIDS
28	M	53711	Hang Nail

Age	Sex	Zipcode	Disease
[25-26]	M	53711	Flu
[25-27]	F	53712	Hepatitis
[25-26]	M	53711	Bronchitis
[27-28]	M	[53710-53711]	Broken Arm
[25-27]	F	53712	AIDS
[27-28]	M	[53710-53711]	Hang Nail

Taxonomy of Generalization Algorithms

- Complete (optimal) vs. Greedy (approximate)
- Top-down specialization vs. Bottom-up generalization
- Global (single dimensional) vs. Local (multi-dimensional)
- Hierarchy-based (user defined) vs. Partition-based (automatic)

Examples for Generalization Algorithms

- Early systems
  - µ-Argus, Hundpool, 1996 - Global, bottom-up, greedy
  - Datafly, Sweeney, 1997 - Global, bottom-up, greedy
- k-Anonymity algorithms
  - AllMin, Samarati, 2001 - Global, bottom-up, complete, impractical
  - MinGen, Sweeney, 2002 - Global, bottom-up, complete, impractical
  - Bottom-up generalization, Wang, 2004 – Global, bottom-up, greedy
  - TDS (Top-Down Specialization), Fung, 2005 - Global, top-down, greedy
  - K-OPTIMIZE, Bayardo, 2005 – Global, top-down, partition-based, complete
  - Incognito, LeFevre, 2005 – Global, bottom-up, hierarchy-based, complete
  - **Mondrian**, LeFevre, 2006 – Local, top-down, partition-based, greedy

Breaking k-anonymity

- **Homogeneity Attack:**
  - When there is inadequate heterogeneity in the sensitive attributes, this can generate clusters that expose information.
  - Example “Heart Disease”

- **Background Knowledge Attack:**
  - Attacker has a known knowledge about the individual and with additional logical reasoning, individual’s sensitive attributes can be leaked.
  - Example: Alice, Age 36, Zipcode 47673; Probability of 66%

➢ Identification of person or private information is possible

Extension of k-anonymity required!

3-anonymous patient table

Age	Zipcode	Disease
2*	476**	Heart Disease
2*	476**	Heart Disease
2*	476**	Heart Disease
>=40	4790*	Flu
>=40	4790*	Heart Disease
>=40	4790*	Cancer
3*	476**	Heart Disease
3*	476**	Cancer
3*	476**	Cancer

ℓ-diversity

ℓ-diversity principle:

A q-block is l-diverse if contains at least ℓ “well represented” values for the sensitive attribute S. A table is ℓ-diverse if every q-block is ℓ-diverse

Distinct ℓ-diversity

- Each equivalence class has at least l well-represented sensitive values

- Limitation:
  - Doesn't prevent the probabilistic inference attacks
  - Example:
    - In one equivalent class, there are ten tuples. In the "Disease" area, one of them is "Cancer", one is "Heart Disease" and the remaining eight are "Flu". This satisfies 3-diversity, but the attacker can still affirm that the target person's disease is "Flu" with the accuracy of 80%.

Disease
Cancer
Heart Disease
Flu
Flu
Flu
Flu
Flu
Flu
Flu
Flu

Entropy ℓ-diversity

- Each equivalence class not only must have enough different sensitive values, but also the different sensitive values must be distributed evenly enough.

- “Evenly enough”: It means the entropy of the distribution of sensitive values in each equivalence class is at least log(l)

- Limitation: Sometimes too restrictive: When some values are very common, the entropy of the entire table may be very low.

Recursive (c,ℓ)-diversity

- A table is said to agree to this principle if the sensitive values in each equivalence class do not occur either too frequently or too rarely.
- In a given q-block, let r_i denote the number of times the i-th most-frequent sensitive value appears in that q-block.
- Given a constant c, the q-block satisfies recursive (c, l)-diversity if

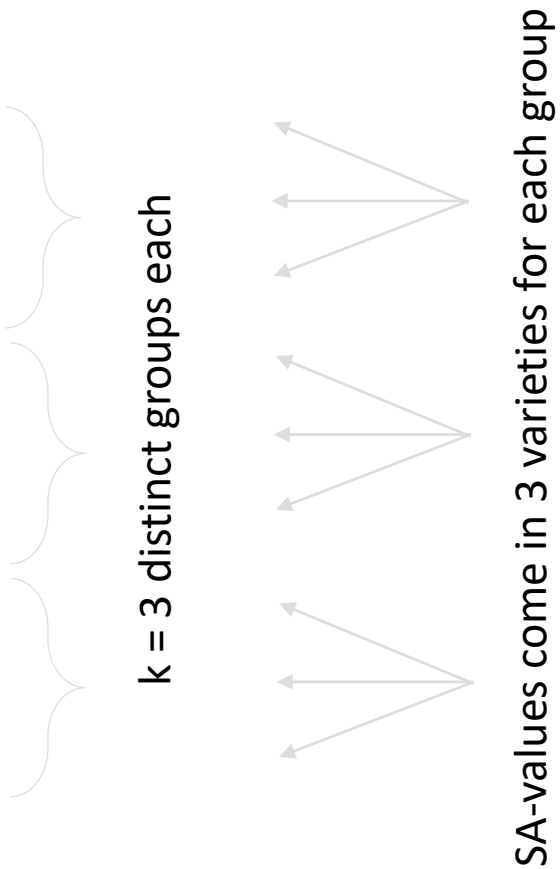
$$r1 > c(r_l + r_{l+1} + ... + r_m)$$

- A table T satisfies recursive (c, l)-diversity, if every q-block satisfies recursive l-diversity. 1-diversity is always satisfied
- This notion is stronger than the previous two notions mentioned above

ℓ-diversity Example 1

- 3-diverse table

Postal Code	Age	Diagnose
940**	<30	HIV
940**	<30	Heart disease
940**	<30	Cancer
940**	3*	Cancer
940**	3*	Lung odeama
940**	3*	Palvic fracture
940**	>39	HIV
940**	>39	Hepatits
940**	>39	Common cold



ℓ-diversity Example 2

- Another Example 3-diverse Table

	Non-Sensitive			Sensitive
	Zip Code	Age	Nationality	Condition
1	1305*	≤ 40	*	Heart Disease
4	1305*	≤ 40	*	Viral Infection
9	1305*	≤ 40	*	Cancer
10	1305*	≤ 40	*	Cancer
5	1485*	> 40	*	Cancer
6	1485*	> 40	*	Heart Disease
7	1485*	> 40	*	Viral Infection
8	1485*	> 40	*	Viral Infection
2	1306*	≤ 40	*	Heart Disease
3	1306*	≤ 40	*	Viral Infection
11	1306*	≤ 40	*	Cancer
12	1306*	≤ 40	*	Cancer

Limitations of ℓ-diversity

- Difficult to achieve due to a single sensitive attribute
  - Two values: HIV positive (1%) and HIV negative (99%)
  - Very different degrees of sensitivity

- l-diversity is unnecessary to achieve
  - 2-diversity is unnecessary for an equivalence class that contains only negative records (e.g. patient data)

- l-diversity is difficult to achieve
  - Suppose there are 10000 records in total
  - To have distinct 2-diversity, there can be at most 10000*1%=100 equivalence classes

Breaking ℓ-diversity

- **Similarity Attack**
  - Bob:

ZIP	Age
47678	27

- Conclusion:
  - Bob has a salary between 20k and 40k
  - Bob has some kind of stomach-related disease

- **Skewness Attack:**
  - Knowledge about attributes can be used to make good a good guess
  - Gastritis is very common -> Bob probably has Gastritis
- Semantic meanings of sensitive values are not considered

Zipcode	Age	Salary	Disease
476**	2*	20k	Gastric Ulcer
476**	2*	30k	Gastritis
476**	2*	40k	Stomach Cancer
4790*	>=40	50k	Gastritis
4790*	>=40	100k	Flu
4790*	>=40	70k	Bronchitis
476**	3*	60k	Bronchitis
476**	3*	80k	Pneumonia
476**	3*	90k	Stomach Cancer

Breaking ℓ-diversity

- With only knowledge of Johns age and that he's visiting both doctors, we can disclose him in the released data sets
- ➢ **Protecting Privacy becomes harder if more Data-Sets are published, even if they are anonymized**

Postal Code	Age	Diagnose
940**	<30	HIV
940**	<30	Heart disease
940**	<30	Cancer
940**	3*	Cancer
940**	3*	Lung odeama
940**	3*	Palvic fracture
940**	>39	HIV
940**	>39	Hepatits
940**	>39	Common cold

Postal code	Age	Diagnose
940**	<40	HIV
940**	<40	Common cold
940**	<40	Lung odeama
940**	<40	Common cold
940**	<40	Lung odeama
940**	4*	Irritation
940**	4*	Palvic fracture
940**	4*	HIV
940**	4*	Common cold
940**	4*	Common cold

Beyond ℓ-diversity

- As we‘ve seen in the previous example, we can easily extract the fact that John has HIV from the data sets
- This was possible, despite the privacy models that applied to both sets:
  - The first set was 3-anonymous and 3 –diverse
  - The second set was 5-anonymous and 4-diverse
- To counteract these matching attempts, another extension to the k-anonymity-family comes into play: t-closeness

t-closeness

- Docking on top of ℓ-diversity, the distribution of SA-values within the quasi-identifier groups is examined
- To satisfy t-closeness, the distribution of SA-values within a given QI group is only allowed to differ by a maximum of t from the distribution of the set as a whole
- The distribution across the entire data set is marked by Q
- t is calculated on the basis of a distance measurement between the SA-values
- This distance is computed using 3 different algorithms that will be discussed later:
  - Ordered distance
  - Equal distance
  - Hierarchical distance
- All 3 are formed using a sum over the participating distributions

t-closeness

- Equal Distance t-Closeness

$$D[P,Q] = \frac{1}{2}\sum_{i=1}^{m}|p_i - q_i| = \sum_{p_i \geq q_i}(p_i - q_i) = -\sum_{p_i < q_i}(p_i - q_i)$$

- Ordered Distance t-Closeness

$$D[P,Q] = \frac{1}{m-1}(|r_1| + |r_1 + r_2| + ... + |r_1 + r_2 + ...r_{m-1}|) = \frac{1}{m-1}\sum_{i=1}^{i=m}\left|\sum_{j=1}^{j=i} r_j\right|$$

t-closeness Example

Ordered Distance for Numerical Attribute “Salary”

The distance between two values is based on the number of values between them in the total order

Q = {3k, 4k, 5k, 6k, 7k, 8k, 9k, 10k, 11k)
P1 = {3k, 4k, 5k}
P2 = {6k, 8k, 11k}

D[P1, Q] Calculation

(5k→11k) = 6, (5k→10k) = 5, etc. for (5k→9k), (4k→8k), (4k→7k), (4k→6k), (3k→5k), (3k→4k)

D[P1, Q] = 1/9 × (6 + 5 + 4 + 4 + 3 + 2 + 2 + 1)/8 = 27/72 = 3/8 = 0.375
D[P2, Q] = 0.167

	ZIP Code	Age	Salary	Disease
1	47677	29	3K	gastric ulcer
2	47602	22	4K	gastritis
3	47678	27	5K	stomach cancer
4	47905	43	6K	gastritis
5	47909	52	11K	flu
6	47906	47	8K	bronchitis
7	47605	30	7K	bronchitis
8	47673	36	9K	pneumonia
9	47607	32	10K	stomach cancer

original

	ZIP Code	Age	Salary	Disease
1	476**	2*	3K	gastric ulcer
2	476**	2*	4K	gastritis
3	476**	2*	5K	stomach cancer
4	4790*	≥ 40	6K	gastritis
5	4790*	≥ 40	11K	flu
6	4790*	≥ 40	8K	bronchitis
7	476**	3*	7K	bronchitis
8	476**	3*	9K	pneumonia
9	476**	3*	10K	stomach cancer

0.375-closeness for Salary

t-closeness Example

Ordered Distance for Hierarchy Attribute “Disease”

The distance between two values is based on the number of values between them in the total order

Distance defined for Hierarchy:

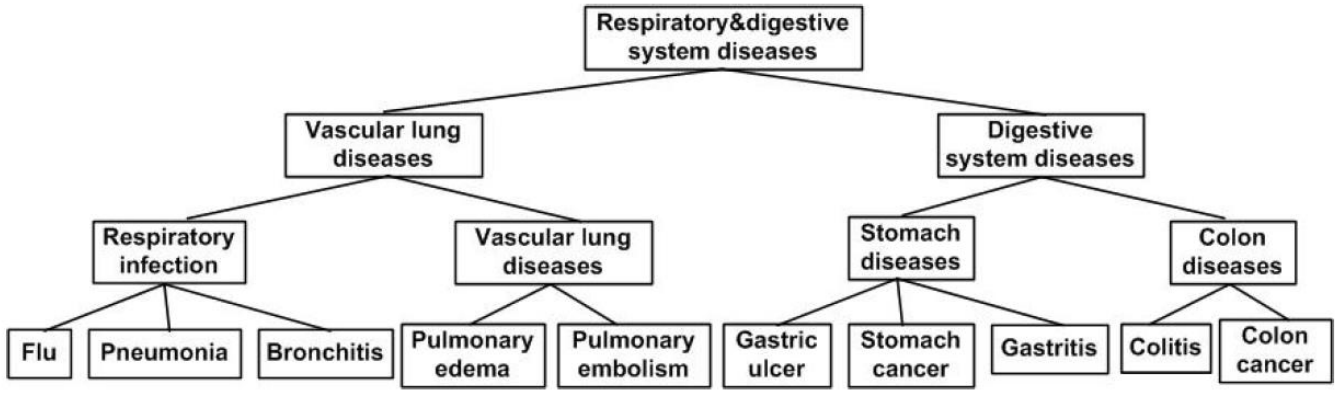
- Flu -> Bronchitis = 1/3
- Flu -> Pulmonary embolism = 2/3
- Flu -> Stomach Cancer = 3/3

Q = {Flu, Pneumonia, Bronchitis, ….}
P1 = {gastric ulcer, stomach cancer, pneumonia}

D[P1, Q] = 0.278

	ZIP Code	Age	Salary	Disease
1	4767*	≤ 40	3K	gastric ulcer
3	4767*	≤ 40	5K	stomach cancer
8	4767*	≤ 40	9K	pneumonia
4	4790*	≥ 40	6K	gastritis
5	4790*	≥ 40	11K	flu
6	4790*	≥ 40	8K	bronchitis
2	4760*	≤ 40	4K	gastritis
7	4760*	≤ 40	7K	bronchitis
9	4760*	≤ 40	10K	stomach cancer

0.167-closeness for Salary and
0.278-closeness for Disease



δ-disclosure

- k-anonymity, ℓ-diversity and t-closeness are model to primarily solve disclosure problems and prevent matching attacks
- So far, we only looked at the privacy aspect of those models, but you can‘t have privacy without loosing utility
- There is always a trade-off between the utility of a data set and the privacy of the individuals it contains
- The t-closeness model suffers from an unnecessary large loss of utility, leading to the adoption of a second approach that solves the same privacy problem:

δ-disclosure

- The main difference lies in the computation of the maximum allowed distribution divergence t (or b in this case), which is done logarithmically with a product instead of a sum
- The core principle stays the same:
  - „a table is δ-disclosure private if the distribution of sensitive attribute values within each quasi-identifier class is roughly the same as their distribution in the entire table“

Basic β-likeness

- t-closeness and δ-disclosure mostly differ in the amount of utility retained, but both fail to incorporate the impact of SA-values, that only occur in very low frequency distributions, into their respective models
- To counteract this, the measurements of β-likeness connects the relative distance between SA-values within a quasi-identifier group and the distance these values have with respect to the whole data set
- A new parameter β results from this connection with 0 as an exclusive lower limit value
- The „basic“ of β-likeness points to an inherent flaw in the system
- The better incorporation of low frequency values is really a shift of modelling priorities, the cost being a worse incorporation of very common SA-values
- Luckily, enhanced β-likeness introduces a solution

Enhanced β-likeness

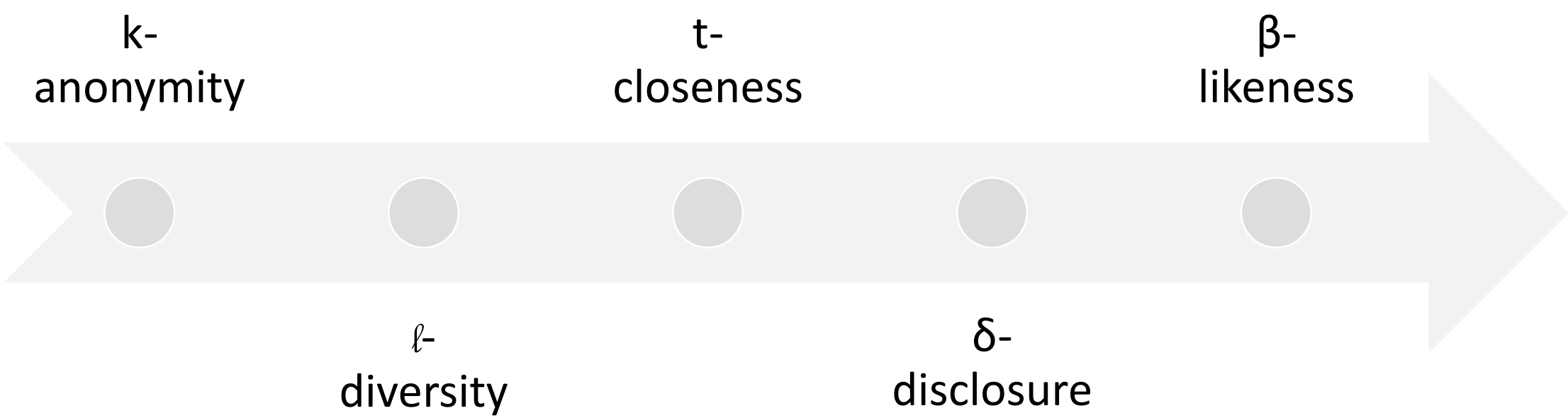
- The model was modified, so that the distance measurement D() has to be either smaller than β or smaller than the negative ln() taken over the global distribution of the attribute that‘s being measured
- Mathematically:

$$\forall q_i, D(p_i, q_i) = \frac{q_i - p_i}{p_i} \leq \min\{\beta, -\ln p_i\}$$

q_i: SA−value distribution across a quasi-identifier group
p_i: SA−value distribution across the whole data set

Looking back

- The privacy models we discussed so far were largely based upon one another, starting with k-anonymity



- The following models live outside this k-anonymity bubble

4.3
Perturbative Masking

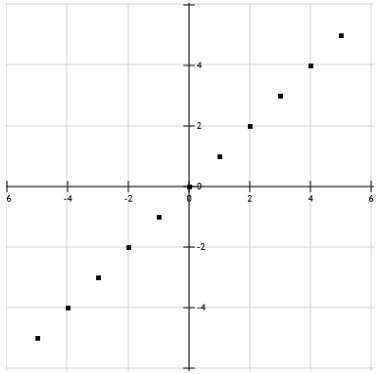
Privacy-Preservation Technologies in Information Systems

Dr. Armin Gerl

WS 2021/2022

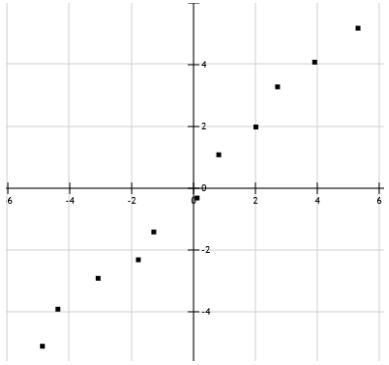
Perturbative Masking

- When determining which anonymisation method fits best is, the expense of the method, its effectiveness in reducing disclosure risk and the information loss it comes with should be taken into consideration
- Perturbative masking methods alter the entries of the data set to prevent re-identification, while seeking to keep the effect of the alterations on the **statistics** of the entire set as low as possible
- When the truthfulness of singular entries of the data set is of minor concern to the publishers, perturbative masking methods provide effective measures for microdata anonymisation
- For example: When providing source data for statistical analysis, the derived calculations from the entire dataset should be reproduceable, while the fact, that the j-th attribute of the i-th respondent x_i^j doesn't carry the originally collected data point anymore, is negligible, as long as the overall statistical properties of the set are preserved



The original data *X* displays a simple linear distribution

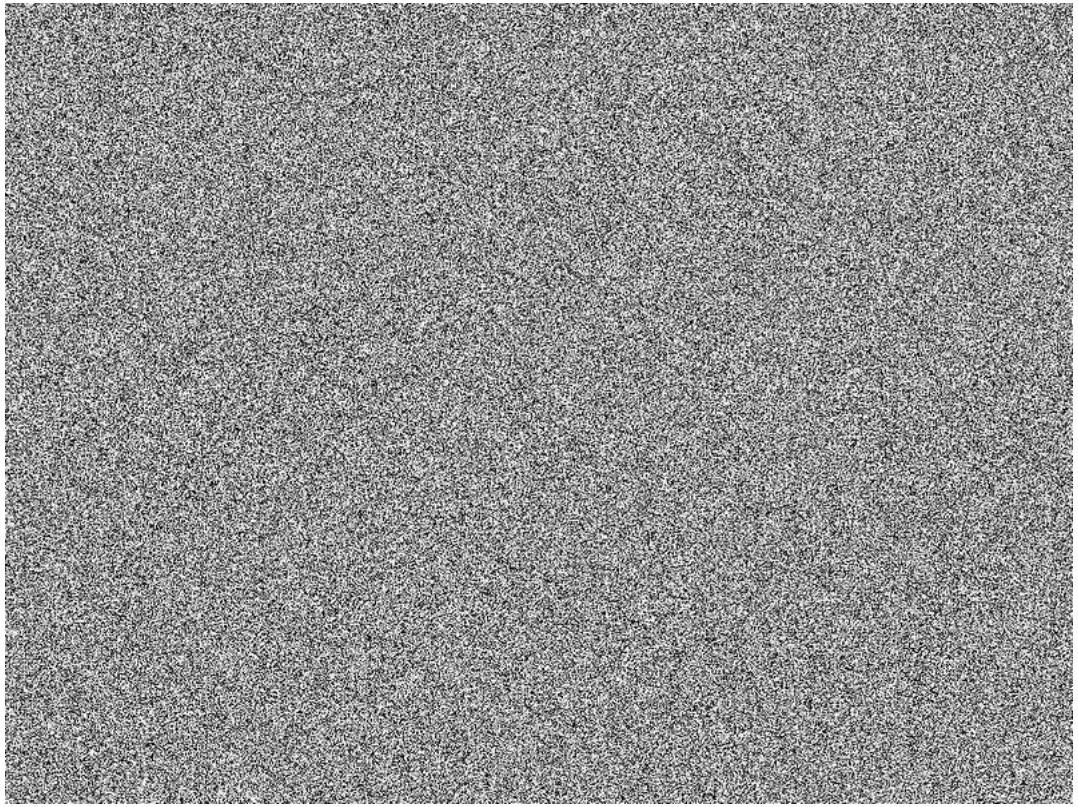
Noise is added to every data point



In *Y* the overall distribution is preserved, despite every point being altered

Noise Addition

- Noise addition methods try to mask the original data points in *X* by adding randomly generated noise in various ways
- How the noise is generated and the statistical properties of its apposition to *X* determine the effects on the resulting data set *Y*
- We will discuss the generation from a Gaussian or normal distribution
- Often used prior to non-perturbative methods to blur sensitive attributes



Uncorrelated Noise Addition

- To transform the i-th attribute original data set X^{i}, the column is replaced by the vector $y^i = x^i + e^i$ with e^i being a vector of normally distributed errors
- The respective components of the vector e^i are independently drawn from the distribution
- Uncorrelated means, that there is no correlation between the noise added to different attributes
- Let e_k^i and e_j^i be components of the vector e^i with $k \neq j$

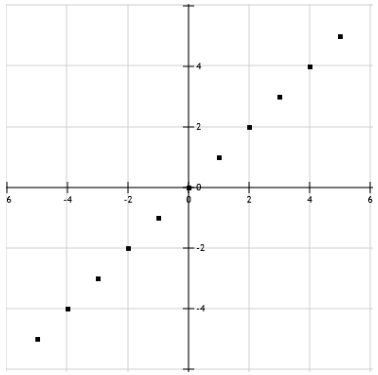
  $\rightarrow\ e_k^i \sim N(0, s_i^2)$; $e_j^i \sim N(0, s_i^2)$ with e_k^i and e_j^i independent variables
- The variance of the the noise added to attribute X^{i} is proportional to the variance of X^{i} itsself

  $\rightarrow\ s_i^2 = \alpha Var(x^i)$ with α being the proportionality constant

Simplified Example

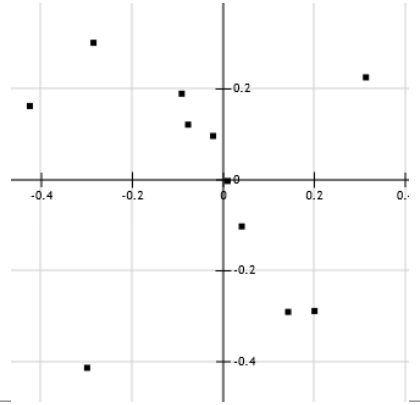
x1	x2
-5	-5
-4	-4
-3	-3
-2	-2
-1	-1
0	0
1	1
2	2
3	3
4	4
5	5

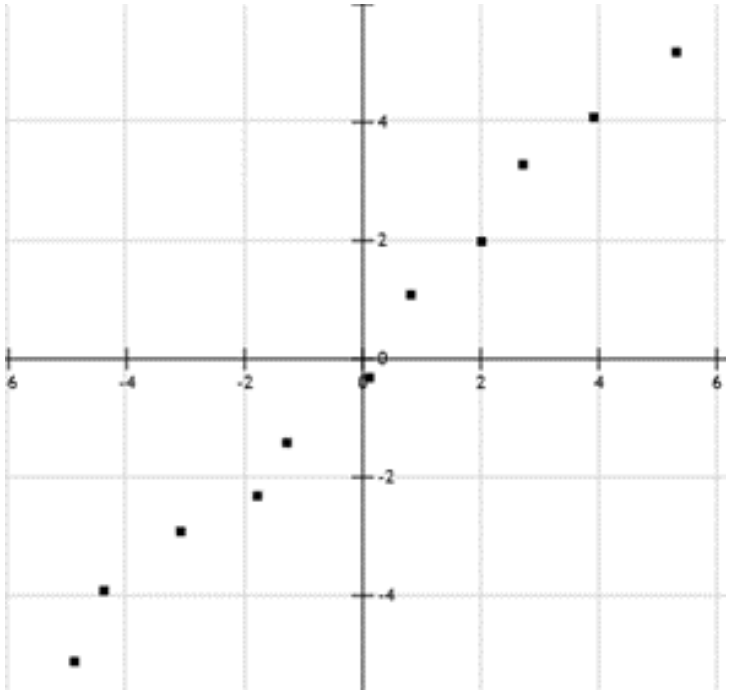
The original data *X*



e1	e2
0,312	0,227
-0,093	0,191
-0,287	0,303
0,009	-0,001
-0,024	0,098
0,141	-0,289
-0,301	-0,412
0,199	-0,287
-0,079	0,123
-0,427	0,164
0,039	-0,101

The random errors for x1 and x2





Statistical Properties of Uncorrelated Noise

- Whilst relatively easy to compute, this transformation only preserves some of the original data‘s statistical properties
- The expected value of a random variable drawn from a normal distribution $N(\mu, \sigma^2)$ is μ
- With that and the linearity of the expected value follows its conservation under the vector addition transformation:

$$\rightarrow E(y^i) = E(x^i + e^i) = E(x^i) + E(e^i) = E(x^i)$$

- When applied to different columns x^i and x^j covariances are also preserved

$$\rightarrow Cov(y^i, y^j) = Cov(x^i + e^i, x^j + e^j) = Cov(x^i, x^j) + Cov(x^i, e^j) + Cov(e^i, e^j) + Cov(e^i, x^j)$$

Because the random error vector e and the column entries x are uncorrelated, their covarianz is 0 and due to the noise being uncorrelated, $Cov(e^i, e^j)$ is also 0

$$= Cov(x^i, x^j) + 0 + 0 + 0 = Cov(x^i, x^j)$$

Statistical Properties of Uncorrelated Noise

- The variance $Var(y^i)$, which is a special case of the covariance $Cov(y^i, y^i)$ of a variable with itself, is not preserved but dependent on the proportionality constant α, with independence between x^i, e^i follows:

$$\rightarrow Var(y^i) = Var(x^i + e^i) = Var(x^i) + Var(e^i) = Var(x^i) + \alpha \cdot Var(x^i) = (1 + \alpha)Var(x^i)$$

- Uncorrelated noise addition doesn't preserve the correlations between different attributes, when the same proportionality constant is chosen for each transformed attribute, the dependence is as follows:

$$\rightarrow \rho_{y^i,y^j} = \frac{Cov(y^i,y^j)}{\sqrt{Var(y^i),Var(y^j)}} = \frac{Cov(x^i,x^j)}{\sqrt{(1+\alpha)^2 Var(x^i)Var(x^j)}} = \frac{1}{1+\alpha} \cdot \rho_{x^i,x^j}$$

Noise Addition Evaluation

Pros:

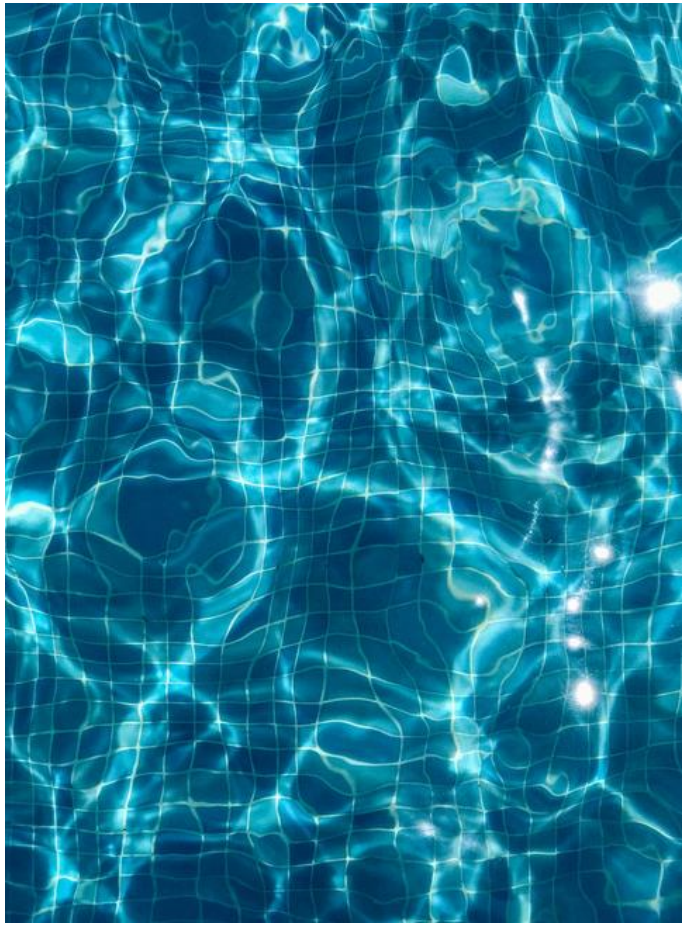
- ✓ Works well with continuous attributes
- ✓ Protects against exact matching with external data if specific attributes are known
- ✓ Preserves linear relationships while keeping disclosure risk low
- ✓ Performable without knowing the subject of the microdata

Cons:

- ○ Time-consuming and difficult to perform on categorical attributes
- ○ Data set needs to have minimum size and/or attribute value range for noise addition to be effective
- ○ Non-linear relationships between attributes may still be lost

Noise Addition Outlook

- When microdata is masked by noise addition, the effects will carry over to queries and sums derived from it, this is called „input perturbation“
- It is possible to reduce the noise needed to achieve the same privacy on processed data by adding the noise to the functions that compute the queries a.k.a. „output perturbation“
- Using other distributions to draw the noise from and switching the focus from preserving statistical properties to privacy protection leads to differential privacy models (more detailed later)



Data Swapping

- Proposed originally for databases consisting only of categorical attributes in 1970
- Aims at introducing doubt wether an attribute really corresponds to the respondent
- Data swapping tries to lower disclosure risk by exchanging values of sensitive attributes between individual records
- The basic procedure is implemented by creating pairs of records with similar attributes and then interchanging identifying or sensitive data values among the pairs
- Although a perturbative method, very little information (besides the initial swap) is lost, since statistics, low-frequency and marginal values are preserved across the data set
- Rarely used in its original form, because process is manual depending on the categorical attribute values
- Idea carries over to more sophisticated algorithms, some of which are also applicable to numerical and continuous attributes

ID	Profession	Age
234	Mechanic	26
235	Cook	31
236	Painter	39
237	Manager	79

A 79yo manager might be a QI*

Rank Swapping

- The principle of data swapping adapted for ordinal and numerical attributes
- Missing values or those representing the maximum or minimum value of the attributes range are excluded from the process
- Using the ordinal properties of the i-th column, the original table X is first ordered by its attribute X^i $\mathrm{in\ ascending\ order}$
- The input parameter p is used to randomly swap each attribute with another within a restricted range
- The larger p is, the larger information loss and lower the disclosure risk is
- The rank of a pair of values eligable for a swap cannot differ by more than p% of the total number of records
- This constrained swapping technique ensures that statistics computed from the data show only minor distortions dependent on the input parameters

Rank Swapping Example

- In our example, the third record is considered at risk of disclosure as the diagnosed illness is a sensitive attribute:

Age	Gender	Income/Day	Illness
15-24	m	100	n
15-24	m	10.000	n
>80	f	120	y

- First, the data set is sorted with respect to the chosen sorting variable „income“:

Age	Gender	Income/Day	Illness
15-24	m	100	n
>80	f	120	y
15-24	m	10.000	n

- The closest record within the swapping range (depending on the sorting variable) forms a swapping pair with the entry at risk and the sensitive attribute values are swapped:

Age	Gender	Income/Day	Illness
15-24	m	100	y
>80	f	120	n
15-24	m	10.000	n

(The set is then shuffled again to not disclose the sorting variable)

Effects of Rank Swapping

- Our example with a binary attribute is oversimplified, in practice categorical attribute values are preferably swapped with similar categories
- Rank swapping has been found to yield good results with respect to the trade-off between information loss and data protection, though the latter can be hard to measure
- The technique is generally less useful for variables with few different values (as in our example) or many missing values
- It is often used as a precision tool similar to our use in the last slide to protect QI-groups
- Due to rank swapping leaving the original values unperturbed, if an attacker knows to whom the highest or lowest value of an attribute belongs (e.g. income from the example), the level of this attribute is then disclosed

Data/Rank Swapping Evaluation

Pros:

- ✓ Preservation of univariate statistics and low-frequency values
- ✓ Adaptable method procedure
- ✓ Limited subject knowledge required
- ✓ Little interference for a perturbative method

Cons:

- ○ Original attribute values are left unperturbed
- ○ Limited attribute disclosure control
- ○ Relies in part on the doubt of truthfullness by the attacker

Related Techniques

PRAM

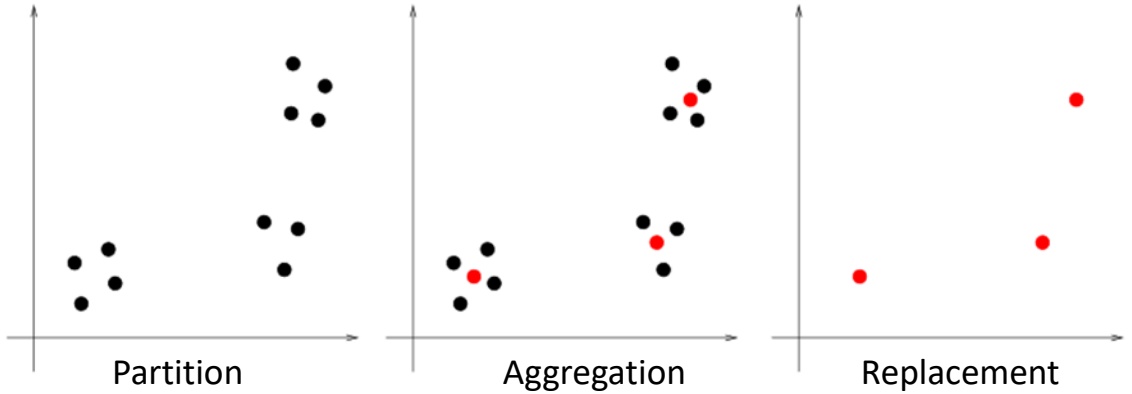
- Post-randomization is a perturbation method that can be applied to categorical variables
- Instead of swapping, the categories of an attribute are randomly transmuted into one another, providing an additional abstraction level with similar effects as data swapping
- Though highly flexible, a lot of subject knowledge is required as the transition matrix, which carries the probabilities for a transition of each category into every other, has to be prespecified along with additional constraints

Cross-set-swapping

- With the goal of preserving statistics while further abstracting from the original records, variations of swapping entries between separate data sets have been proposed
- The second data set has to have similar statistical properties to the original and swappable attribute values, which are difficult to find
- In practice often used in tandem with synthetic data generation as swap source

Microaggregation

- A perturbative masking method for continuous microdata
- Each row or record X_j of the microdata set is aggregated with at least k other rows to form groups of microaggregates
- The attribute values of each group are formed by taking the average value of their respective member values



- Which records get aggregated into a group is decided by following the criterion of maximal similarity, avoiding individual record values to exert a dominating effect on the average
- The value of k is a lower bound, but beyond that the group size can either be fixed for all groups or determined for each group individually depending on the data values, finding the optimal size is the crux of any algorithm trying to implement microaggregation
- Microaggregation algorithms can further be divided into the uni- or multivariant categories, depending on wether they simultaneously deal with one or several attributes respectively

Measurements for Microaggregation

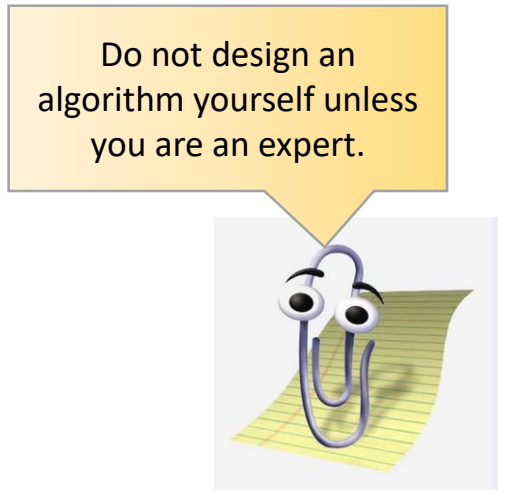
- Let the example set X consisting of n records X_j (also referred to as data vectors) be divided into g groups
- The j-th data vector within the i-th group will be denoted as $X_{ij},\ j \in [1, \dots, k, \dots, n_i]$ with each group containing a minimum of k vectors so that $\sum_{i=1}^{g} n_i = n$
- The computed average group vector for group i will be referred to by $\overline{X_i}$, while the average data vector for the whole microdata set is $\overline{X} = \sum_{i=1}^{g} \frac{\overline{X_i}}{g}$
- Taking the sum over the squares of deviations of each rows entries from its group average leads to:
  $$SSE = \sum_{i=1}^{g} \sum_{j=1}^{n_i} (X_{ij} - \overline{X_i})'(X_{ij} - \overline{X_i})$$
  with squares represented by transposed vector multiplication for our multidimensional case
- Summing over the squares of deviations of each group from the global average likewise leads to:
  $$SSA = \sum_{i=1}^{g} \sum_{j=1}^{n_i} n_i (X_i - \overline{X})'(X_i - \overline{X})$$
- Adding these two results in the summed squares of deviations of each vector from the set average
  $$SSE + SSA = SST = \sum_{i=1}^{g} \sum_{j=1}^{n_i} (X_{ij} - \overline{X})'(X_{ij} - \overline{X})$$

Designing a microaggregating Algorithm

- Finding a partition of X into g groups at first glance shows parallels to the NP-complete problem of number partition discussed in theoretical informatics:

  > "Given a list of positive integers, find a subset such that the difference between the sum of numbers part of it and the sum of those who are not is minimized"

- Following the principle of maximum similarity required for the groups, our problem is yet more related to the clustering problem, whose goal it is to split a population into a fixed number of disjoint, similar groups
- But the k-partition issue we are dealing with in microaggregation, while also being an optimization problem, has additional constraints:
  - The group size, disregarded in the classical clustering problem, has a lower bound
  - The number of groups doesn't need to be fixed
  - Dealing with vectors instead of data points adds an additional layer of complexity



The *k*-Partition Problem

- The desired property of a good partition to have similar group members has 2 beneficial effects:
  - Avoiding the linking of records with extreme values to a group, which would be possible for randomly assigned groups and an attacker with knowledge of the attributes distribution
  - Minimizing information loss when publishing the group average, since the within-group deviations are low
- To solidify this: The optimal *k*-partition is the one that minimizes the SSE-Messurement (which is equavalent to maximizing SSA)
- A standardized way to quantify the information loss is $L = \frac{SSE}{SST}$, weighting the sums of squares of the within-group deviations against that of all vectors from the global average to produce values between 0 and 1
- Deterministic ways to find the optimal *k*-partition proves difficult in most cases, heuristic methods are the only practical ones

Univariate, Fixed-size Solutions

- A simple approach to the *k*-partition problem is sorting the vectors in relation to some criterion and combining successive *k* vectors into a group, replacing the attribute inside each group by ist average
- This leads to *g-1* groups of size k and one bigger group that holds the surplus, hence the name „fixed-size“ for this family of algorithms
- The description „univariate“ refers to the straightforward, one-dimensional sorting methods used, yet those can still be subdivided into sorting by:
  - Single-axis: If all attributes are highly correlated, the easiest way to sort them is by choosing one that somehow reflects the „size“ of the vector
  - Principal component: Finding a function of the sets attributes, that highly correlates them to measure their contribution to the subject the vectors represent
  - *Z*-score: A subject-independent case of principle component sorting that standardizes the value-ranges of each attribute and them up for each vector
- The last univariate approach sorts each attribute seperately and forms groups, ripping the original vector composition appart, besides the conceptual problems this carries on release, an attacker can infere bounds for variables by comparing the sorted averages, so this method is unsuitable if those aren‘t far between
- This approach is called „individual sorting“, it preserves the most information compared to the others, but suffers from the same disclosure problems as data swapping

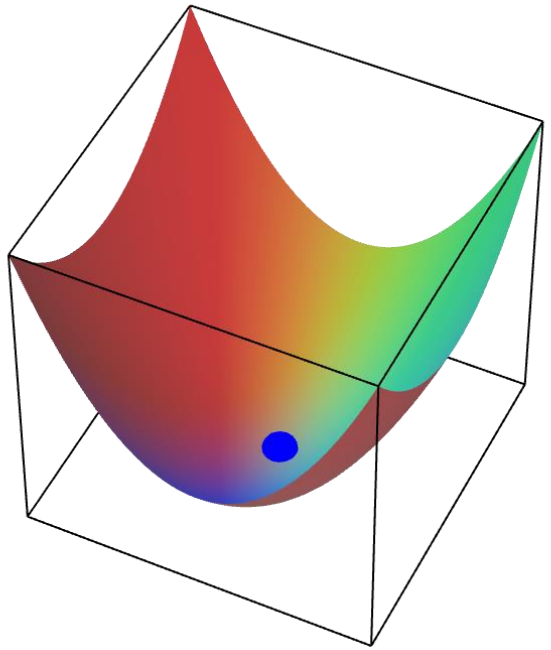
Multivariate Solutions

- The microaggregation SDC-method suffers from 2 fundamental problems:
  - A data vector consisting of multible (continuous) attributes, which have varying correlations and hold different real world information, must be assigned a distance analogy in order to compare it to other data vectors
  - An optimal *k*-partition must be found, minimizing information loss and disclosure risk simultaneously
- The first problem can be approached in 2 ways: Either the previously discussed projections of multiple attributes onto a one-dimensional measurement is used or distance is computed by using a multidimensional distance matrix (except individual sorting which bypasses the problem but comes with its own severe problems)
- The second problem splits into the data-oriented and fixed-size approaches, each of which can use one of the two measurement variants for distances between vectors, leading to univariate or multivariate versions
- While projecting real world data sets onto a single measurement distorts the anonymisation effect subsequent *k*-partitioning algorithms have across different attributes, optimal multivariate microaggregation has been shown to be a NP-hard problem, hence heuristics are used that are resource-intensive and/or step-wise optimal at best

A good overview and more on the multivariate case in „A comparative study of microaggregation methods“ J.M. Mateo Sanchez, J Domingo-Ferrer

Data-oriented Solutions and Optimization

- The optimization (or mathematical programming) problem:
  - Max- or Minimization of a real function by choosing input values from an allowed set and computing the value
  - The subfield of constraint satisfaction tries to find a range of values with respect to prespecified conditions
- Not every distribution of values and combination of attributes is optimally divided by setting a fixed group size in advance, so to minimize information loss, more complex algorithms that form variable-size groups are constructed
- The 2 examples presented are heuristic algorithms that need to be combined with individual, single-axis or principal component sorting to deal with multible attribute sets (which makes them univariate), multivariate versions exist and build on the same principles



Microaggregation with Genetic Algorithms

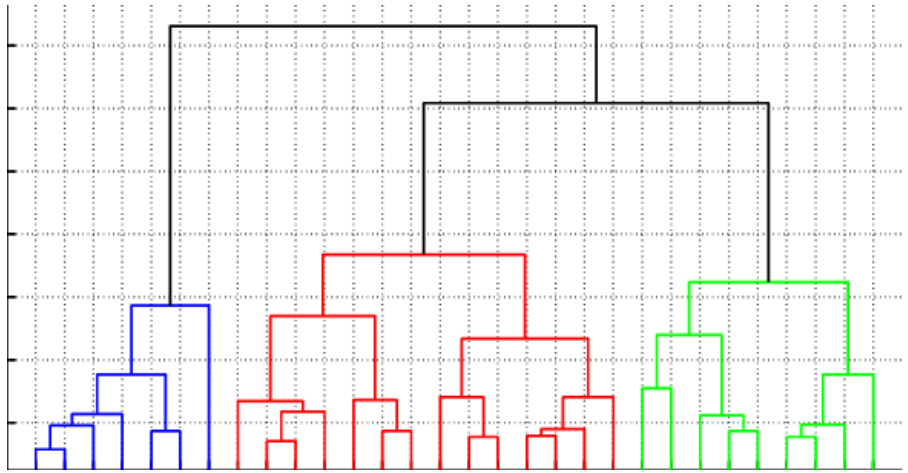
- Being a subclass of evolutionary algorithms, a genetic design tries to find solutions to the optimization (or search) problem by iterative, heuristic use of mutation and selection inspired by biology
- Solutions (*k*-partitions in our case) are encoded in binary strings, with each candidate having a set of mutatable properties (e.g. the group vectors or their variables)
- The algorithm is initialized with a starting population of solution candidates, this can be randomly generated or computed by another approach to the problems solutions (some implementations feed the genetic algorithm with a fixed-size solution)
- With respect to the constraints set by maximum similarity, the mutatable properties of the solution candidates are varied (or combined between solutions) step-by-step, with evolutionary selection of promising candidates
- This process should lead a well designed genetic algorithm towards the global optimum value of all possible populations, solving the optimization problem, but since designed as a heuristic the near optimum is sufficient
- Runtime and memory usage are usually the problem with this approach when applied to large data sets, therefor it is often consulted as a last step to optimize a partial solution

Ward‘s Algorithm

- Ward‘s algorithm is an agglomerative hierachical clustering procedure:
  1. Each record in the dataset forms a group
  2. Find the nearest pair of distinct groups and merge them into one
  3. Repeat step 2 until all records are merged into a single group
- Between-group distance measurement used is based on the euklidean measurements discussed earlier and minimizes *SSE*-increase on merge

$$d(G_i, G_j) = \frac{n_i n_j}{n_i + n_j} (\overline{X_i} - \overline{X_j})^2$$

- Ward‘s algorithm in its original form is used to compute a group hierarchy, with the results displayed in a dendogram
- It can be adapted to recursively compute a *k*-partition, resulting in the *k*-Ward algorithm



Ward‘s hierachical clustering algorithm in a dendogram

Ward‘s Method

- Agglomerative (in contrast to divisive) approach to hierarchical clustering
- Employs a bottom-up strategy, starting with n groups for each of the n elements in the set
- The distance measurement between groups $d(G_i, G_j)$ is called “merging cost” and its minimization is the criterion for the next merging step, making the algorithm both greedy and constraint by the previous choices in the procedure

$$d(G_i, G_j) = \frac{n_i n_j}{n_i + n_j} (\overline{X_i} - \overline{X_j})^2 = \frac{n_i n_j}{n_i + n_j} (\overline{X_i} - \overline{X_j})' (\overline{X_i} - \overline{X_j})$$

- Since each group average is a vector in datasets containing more than one attribute, the square is transformed into transposed vector multiplication to condense the distance down to a single number
- For simplicity, we will perform the algorithm on a one-dimensional sample set of numbers

Hierarchical Clustering Example

- Our data set consists of 6 records with one numerical attribute: $\{2, 12, 16, 25, 29, 45\}$
- In step 1, every record forms its own group and the distances between them are computed:

$$d(\{2\},\{12\}) = \frac{1}{1+1}\left(\frac{2}{1} - \frac{12}{1}\right)^2 = 50 \text{ and with the same formular}$$

$$d(\{12\},\{16\}) = 8, \qquad d(\{16\},\{25\}) = 40{,}5, \qquad d(\{25\},\{29\}) = 8, \qquad d(\{29\},\{45\}) = 128$$

- Since the formular (group sizes) stays the same, it is sufficient to only compute the distance between neighbors of the ordered numbers, because it is reasonable to assume not-neighboring pairs will result in an even greater square, when the group sizes vary or more than 1 attribute is present, this relation is hard to see and all between-group-distances need to be computed
- We now look for the lowest value of d to determine which group(s) to merge in this step, these are the groups {12} and {16} as well as {25} and {29} both with a distance of 8

Example (cont.)

- After the merge, we are left with 4 groups {2}, {12,16}, {25,29} and {45}
- We once again need to compute the distance between each of them e.g.:

$$d(\{2\},\{12,16\}) = \frac{2}{1+2}\left(\frac{2}{1} - \frac{12+16}{2}\right)^2 = 96$$

- The results of the distance measurements between groups are usually displayed in a "(dis)similarity matrix" which is symmetric and has 0s as diagonal values in Ward's method:

	{2}	{12,16}	{25,29}	{45}
{2}	0	96	416,33	924,5
{12,16}	96	0	169	640,33
{25,29}	416,33	169	0	216
{45}	924,5	640,33	216	0

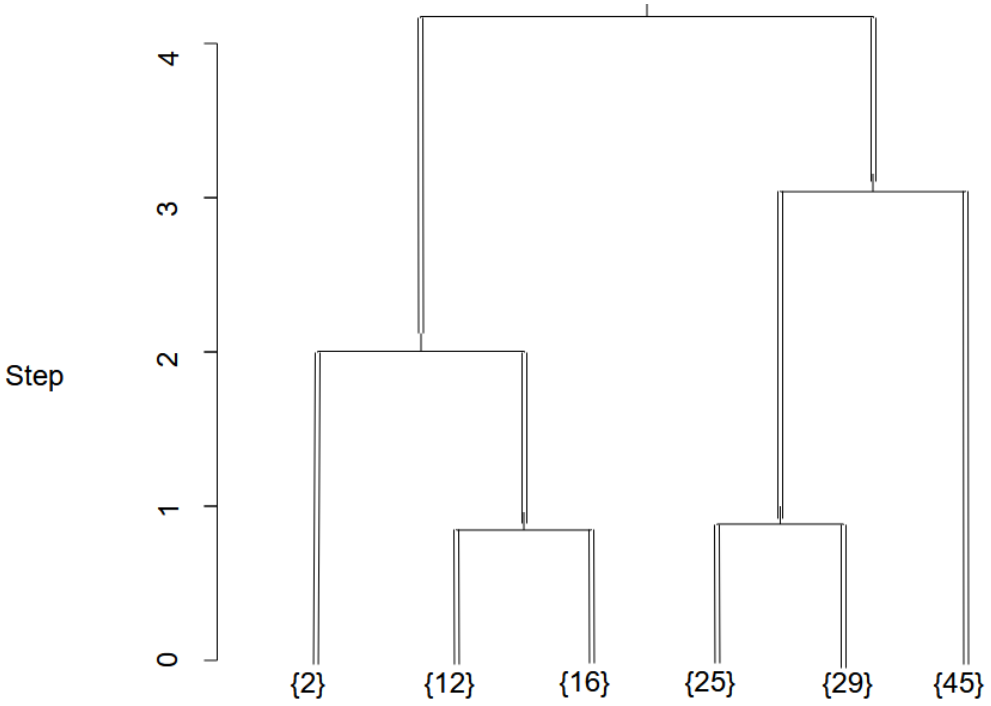
- This qualifies the groups {2} and {12,16} for a merger in step 2

Example (cont.)

- The remaining step is analogous, with the dissimilarity matrix given below:

	{2,12,16}	{25,29}	{45}
{2,12,16}	0	346,8	918,75
{25,29}	346,8	0	216
{45}	918,75	216	0

- After merging groups {25,29} and {45} we are only left with 2 groups and merging those subsequently terminates the algorithm, the process (and the resulting hierarchy) are displayed in a dendogram like this:



The *k*-Ward Algorithm

- The first adaption that needs to be made is to add a constraint to Ward‘s procedure by forbiding the merging of groups that both contain a record number greater than, or equal to *k*
- Though this imposes a stop condition before all groups are merged into one, the adapted version still tends to form large groups, merging as long as only one of the groups has fewer than *k* records
- With this adaptation in mind, the complete *k*-Ward procedure, which also relies on the distance $d(G_i, G_j)$, follows the steps below:
  1. Locate the 2 records with the greatest distance from each other in the dataset
  2. Form 2 groups with the nearest k-1 records from those 2 records respectively, all the left over records form their own group
  3. Use Ward’s adapted procedure until all records in the dataset belong to a group containing *k* or more elements
  4. In the final partition, use any group that contains 2*k* or more elements as input to apply this algorithm recursively
- The recursive nature follows the build-all-groups-simultaneously style and outperformes the genetic and many other approaches to the *k*-partition problem when it comes to ressource usage
- Its tendency to build large groups (which contributes to information loss) can be combated by introducing an additional constraint *t* (“mixability”) to the adapted Ward’s algorithm, leading to the generalized version of our *t*=1 case

For more information on the generalized version see: „Generalized k-Ward Microaggregation”, Juni-Lin Lin, Laksamee Khomnotai, Cia-Chun Hung

Microaggregation Overview

- Microaggregation has one of the best information-loss/disclosure-risk ratios if the dataset is suitable and the k-partition problem was solved optimally
- A suitable data set needs to have a sufficient attribute value range in order for the distance measurements and subsequent aggregations to be subtle enough, it is therefor best applicable to continuous attributes
- Since most datasets contain a mix of attribute data types, an optimal solution to the k-partition problem with a multivariate approach, that takes all attributes into consideration simultaneously, is computationally intensive and difficult, yet the basic univariate approaches are easily feasable without subject knowledge
- Some models that try to define privacy in data structures (e.g. k-anonymity) can be satisfied very effectively by using microaggregation (more on this in the next chapter)

Microaggregation Evaluation

Pros:

- ✓ Effective disclosure control when dataset is suitable (except individual sorting)
- ✓ Minimal information loss
- ✓ Strongly coupled with privacy model *k*-anonymity
- ✓ Subject knowledge is otional

Cons:

- ○ Only effective for continuous data or a large enough (numerical) attribute range
- ○ Univariate projections often to coarse for real data sets
- ○ Multivariate case is NP-Hard problem

Differential Privacy

- Another promising approach, that doesn‘t rely on attribute classification measures is differential privacy
- Instead of the static data, the model focuses primarily on operations on the data set, e.g. entries being added or deleted
- A central concept of the model states, that an attacker shouldn‘t be able to infer any clues, if a specific entry was added or removed and what its contents were
- Mathematically, the probabilities (Prob) of getting a result-set within a domain (S) from a query (K) are computed between the original table (D_2) and the modified one (D_1)
- The model limits the difference between the two through a factor ϵ

$$\text{Prob}[K(D_1) \in S] \leq e^{\epsilon} \times \text{Prob}[K(D_2) \in S]$$

(ϵ, δ)-Differential Privacy

- Because some extreme attribute values can disproportionally blow up the computation, an extension to the model was added to make it more practical
- To achieve this, another parameter δ was introduced
- It acts as an additive weight to the upper limit and rectifies its strictness

$$\text{Prob}[K(D_1) \in S] \leq e^{\epsilon} \times \text{Prob}[K(D_2) \in S] + \delta$$

- When the parameter δ is equal to 0, the original formula reappears, the extension is called (ϵ, δ)-differential privacy

Differential Privacy and k-anonymity

- As we‘ve already seen, the k-anonymity model makes use of hierarchical generalization
  - 94032 -> 9403* -> 940** -> …
  - 192.168.1.127 -> 192.168.1.* -> …
- Thereby making sure that a sufficient number of equivalence classes are able to be formed
- The hierarchy formation process discussed so far was static and attributes are assigned to their equivalence classes from the start, which makes the process revertible
- If random hierarchy formation is used, the traceability disappears and the criteria of differential privacy can be met (with fitting parameter values δ and ε)

Practical Differential Privacy

For Details on Differential Privacy visit the Lecture: Privacy Enhancing Techniques, 5881V

- The industry has high expectations for the model
- With the reality of volatile, heterogeneous storage systems holding personal data, the transaction focus of differential privacy is more suitable than the static-state focus of classical models
- The information loss on published data could be controlled through indistinguishability of operations, even though new entries are being added and removed at a rapid rate when:
  - New user data is added
  - The user revokes his data processing agreement
  - Maximum allowed storage time is exceeded
- The model is incorporated into query parsers to automatically ensure, that attackers cannot infer any information by probing the data base through queries
- A typical pitfall: If the query parser itself processes different queries at different rates, the time needed can also allow inferences

4.4
Measuring Utility

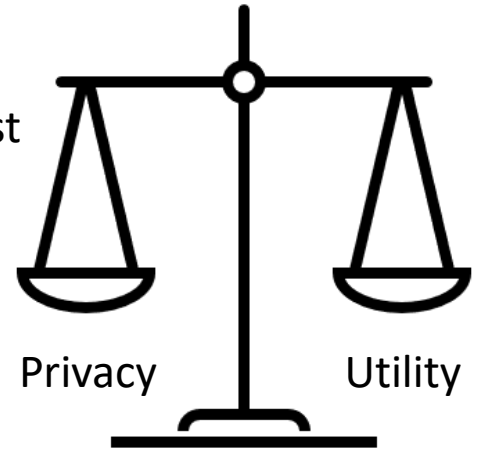
Privacy-Preservation Technologies in Information Systems

Dr. Armin Gerl

WS 2021/2022

Requirements

- There is an inherent trade-off between anonymity and utility of a data set
- A hypothetical data set with the highest achievable anonymity would then contain almost no information
- Everyone has the right to privacy, but:
- Since there are no reasons to store a data set without information, maximum anonymisation isn't a good thing either
- For data collectors, a balance between costs and risks is the name of the game
- Practical implementations of the privacy models we've seen so far need to:
  - Produce a data set that satisfies the model
  - Limit the usefulness of the data as little as possible at the same time
- Measurements for data privacy are as important as measurements for its utility

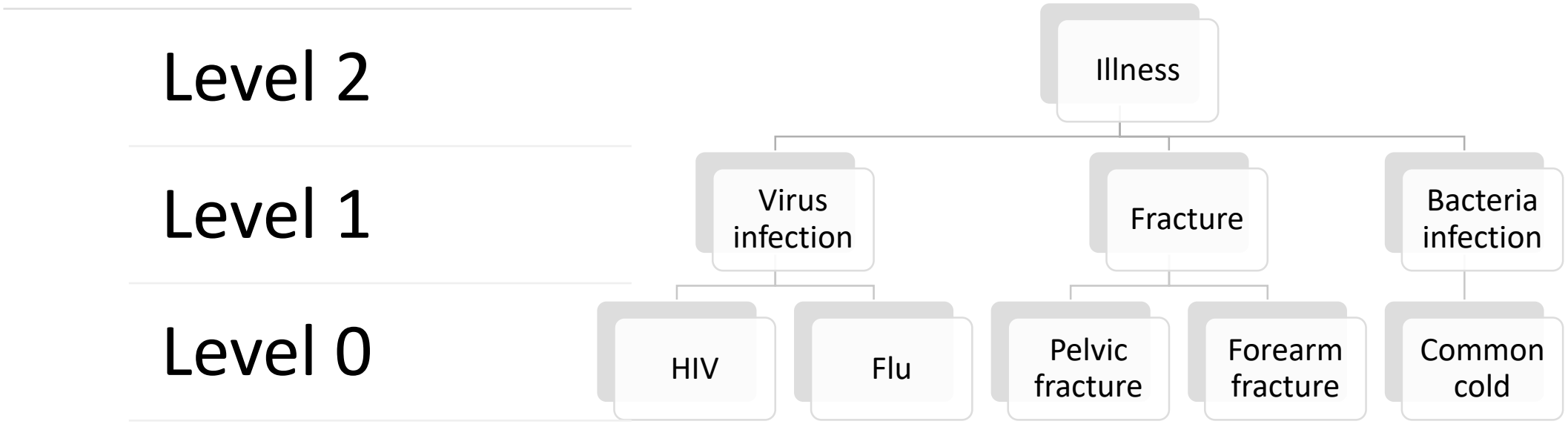


Measuring Utility

- Data utility is strongly connected to the subject of the data set
- Our approach to measuring it will be a mathematical and objective one
- The following illustrations show principles, that are applicable to singular attributes (e.g. postal code, illness or sex)
- It is important to differentiate between categorical and numerical attributes
- When measuring the whole data set, numerical attributes need to be aggregated and sometimes attached with weights
- At first, we will take a look at generally applicable methods, as many measurements are tailored to specific use-cases

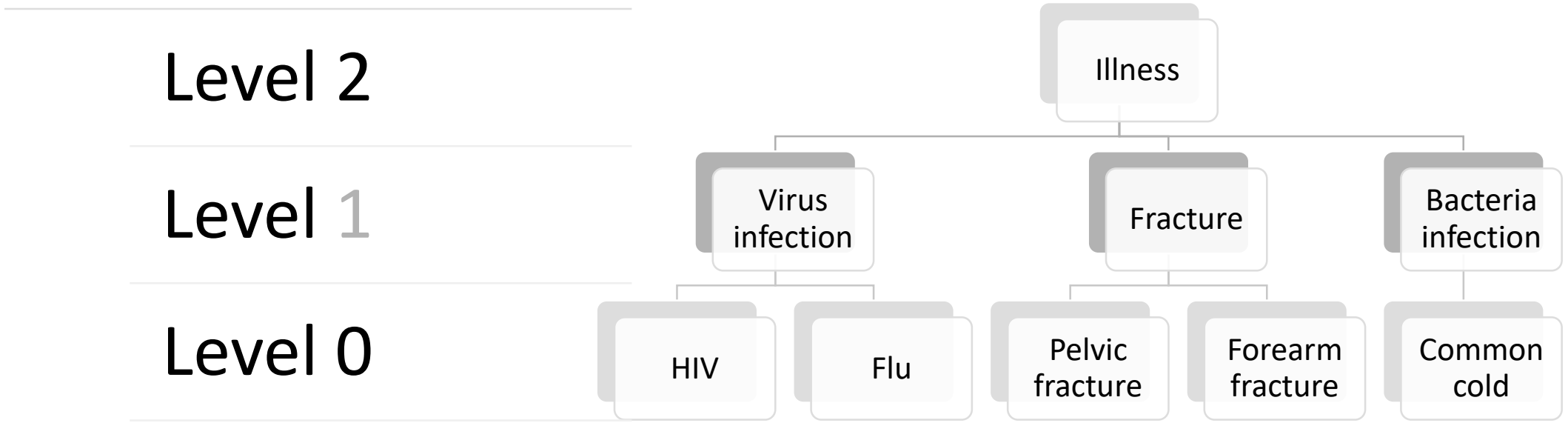
The Height Metric (or tree measure)

- The measurement focusses on rating the usefulness of hierarchical generalization
- It assesses the selected group through its height in the hierarchy tree and normalizes it with respect to the maximum height of the tree
- It is computed as follows: The height or level of the generalization is divided by the total height H minus 1



Tree Metric Example

- For example: If the target data set is generalized to fractures, virus and bacterial infections, we measure the following:
- H = 3 the maximum height of our hierarchy tree
- h/(H-1) for level 1 would produce 1/(3-1) which is 50%



Limitations of the Tree Metric

- There tree metric measurements gets inaccurate in certain circumstances:
  - When maximum heights H differ a lot between different attributes
  - When the information content between different attributes varies, that is when missing information in some attributes impacts the data sets utility disproportionally more than in others
- In our running example set, missing information about a persons set would be much less impactful as a missing illness entry

Sex	Illness
female	HIV
male	HIV
male	Common cold
male	Palvic fracture

Sex	Illness
female	*
male	*
male	*
male	*

Sex	Illness
*	HIV
*	HIV
*	Common cold
*	Palvic fracture

- Using the tree metric, both anonymisation options would produce the same value and are equivalent in the eyes of the measurement

Granularity/Loss

- A different approach to data utility evaluation is the loss metric *LM*
- In contrast to the other 2 we looked at, *LM* doesn‘t have it‘s scope around the entire attribute but looks at every value of a column separately
- From the separate results, a mean value is then computed

$$LM = \frac{(M_P - 1)}{(M - 1)} = \frac{(U_i - L_i)}{(U - L)}$$

- Within the hierarchy tree *T,* there are a maximum of *M* leafs
- The middle Node *P* for a given leaf has paths to at max M_P other leafs
- For numerical attributes, U_i and L_i represent the upper and lower limit of the value range, while *U* and *L* stand for those limits within a given generalization group

Granularity/Loss (cont.)

- To illustrate the concept, let‘s take a look the example used for the height metric
- All the different illnesses were generalized into 3 groups: Fractures, virus and bacterial infections
- For an entry with the original value ‚HIV‘, the generalization process would group it into ‚virus infection‘
- The loss metric for this:

$$LM_{HIV} = \frac{(2-1)}{(5-1)} = \frac{1}{4} = 0{,}25 = 25\%$$

- On the other hand, an entry with ‚common cold‘, that was generalized to ‚bacterial infection‘, wouldn‘t show any loss as $LM_{CC} = 0$

Precision

- Another model that decides singular values (or cells Z) instead of entire columns
- The number of rows is the basis for a mean value of the cell results
- At the beginning, the maximum possible informational value of the data is 100%
- After generalization was applied, the metric computes the $Distortion(Z)$ that resulted from the transformation and subtracts it from 100%
- Just like for the tree metric, the hierarchy tree is vital here

$$Distortion(Z) = \frac{Level\ of\ the\ tree\ for\ selected\ cell}{Total\ tree\ height}$$

$$Precision(Z) = 1 - Distortion(Z)$$

Precision (cont.)

- Summarizing this metric into a formal definition:
  - Let *PT* be a table with attributes $A_1, \dots, A_n$ and *RT* be a anonymized version of said table with the same attributes $A_1, \dots, A_n$ and let $t_{P_j} \in PT$ as well as $t_{R_j} \in RT$ be instances of those tables
  - For a given attribute, DGH_A represents the hierarchy tree for the generalization functions $g_1\left(t_{P_j}[A_1]\right), \dots, g_n\left(t_{P_j}[A_n]\right)$, that form $t_{R_j}[A_i]$
  - With this, we conclude for *RT:*

$$Prec(RT) = 1 - \frac{\sum_{i=1}^{n}\sum_{j=1}^{N}\frac{h}{|DGH_{A_i}|}}{|PT| \times |n|}$$

- A good anonymisation within this metric is gravely influenced by the maximum height of the hierarchy tree

(Sum of) Squared Error

- We‘ve looked at singular cell metrics and column metrics, the metric at hand works on rows instead
- The data set X consists of rows x_j
  - a_j^i is denoted as the original, and $\left(a_j^i\right)'$ as the generalized *i*-th attribute value of the *j*-th row
- From these two attribute values, the normalised Euclidean distance NED is then calculated, squared and summed up for the selected row and then aligned with the set of all attributes m
- As we’ve seen bevor, this result is then subsequently normalized with respect to the total number of rows *n,* resulting in:

$$SSE = \frac{1}{n} \times \sum_{x_j \in X} \frac{1}{m} \times \sum_{a_j^i \in x_j} \left(NED\left(a_j^i, \left(a_j^i\right)'\right)\right)^2$$

Record-oriented metrics

- We will now go over the remaining metrics in quick succession
- Some gloss over attribute values entirely and rely solely on the size of equivalence classes
- The following points illustrate some approaches:

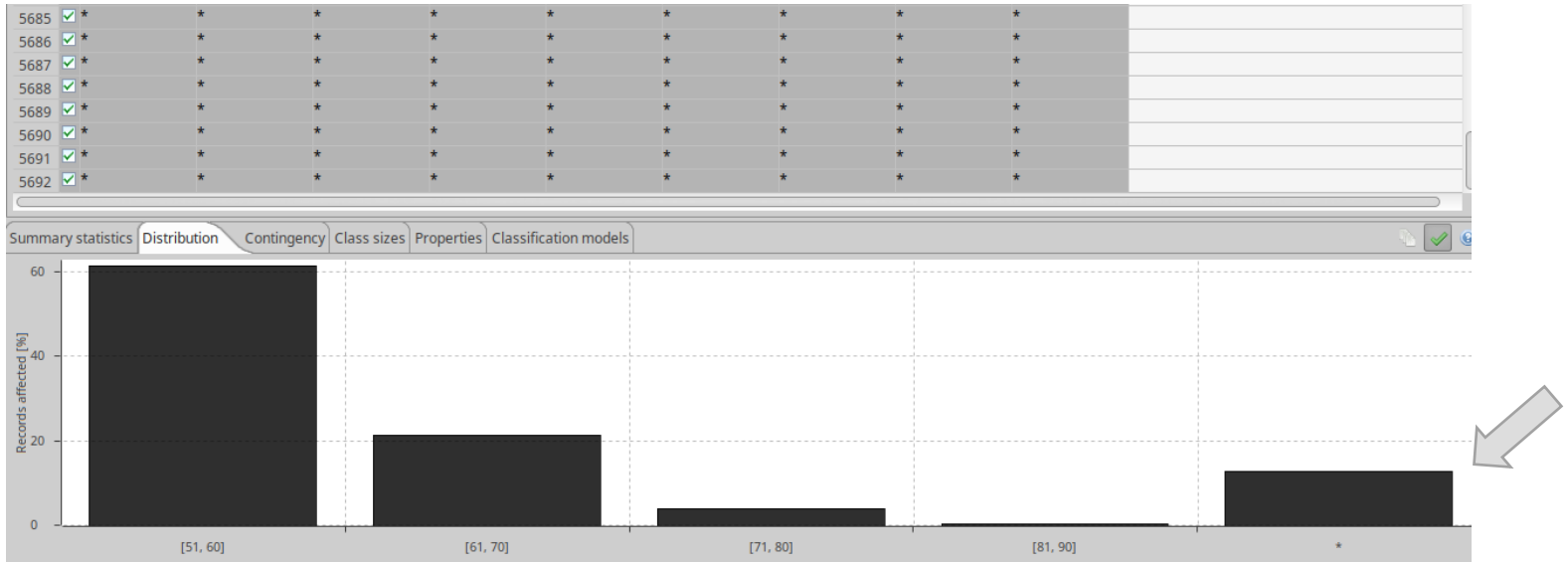
- The metric ‚average equivalence class size‘ only looks at the amount of in differentiable equivalence classes (eqs) that were formed:

$$C_{AVG} = \frac{\frac{Total\ number\ of\ entries\ n}{Total\ number\ of\ formed\ eqs}}{minimum\ number\ of\ indifferentiable\ eqs\ k}$$

Record-oriented metrics (cont.)

- The ‘discernability‘ metric counts entries that lead to the formation of equivalence class *E* and then uses this count to discern the complete suppression of entries by counting them all into a single equivalence class

$$C_{DM} = \sum_E |E|^2 \geq k \times n \geq 1$$



Record-oriented metrics (cont.)

- The next metric is called ‘Ambiguity‘, when an entry is masked, it measures how many possible values were possible in its place. We‘re using the notation from Entropy:

$$\Pi_{AM}(D, g(D)) = \frac{1}{n} \times \sum_{i=1}^{n} \prod_{j=1}^{r} |\bar{R}_i(j)|$$

- Another candidate is ‘Record Level squared Error’, which follows the same principles as ‘Squared Error’ but instead of focussing on the distance between singular attributes, the tables rows are taken as vectors and a distance is computed

Outlook Data Quality

- Additional Measures can be considered to calculate “Utility”
- 1. Completeness: The ratio of stored data against the potential of `100% complete' data
- 2. Uniqueness: Nothing will be recorded more than once based on how that thing is identified"
- 3. Timeliness: The degree to which data represents reality from the required point in time
- 4. Validity: Data is valid if it conforms to the syntax ,i.e. format, type and range of its definition
- 5. Accuracy: The degree to which data correctly describes the `real world' object or event being described"
- 6. Consistency: The absence of difference, when comparing two or more representations of a thing against a definition

https://smartbridge.com/data-done-right-6-dimensions-of-data-quality/

4.5
Anonymization and Privacy Languages

Privacy-Preservation Technologies in Information Systems

Dr. Armin Gerl

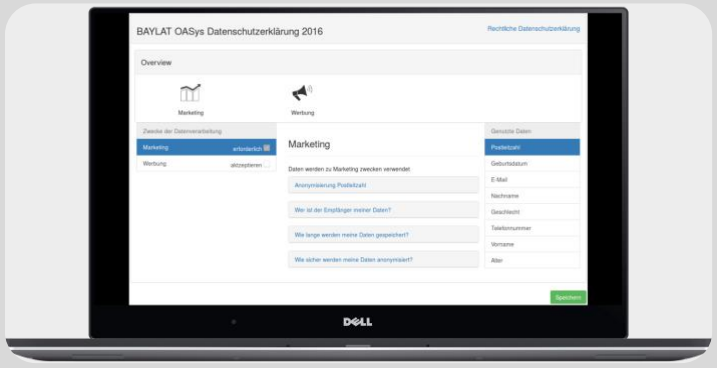
WS 2021/2022

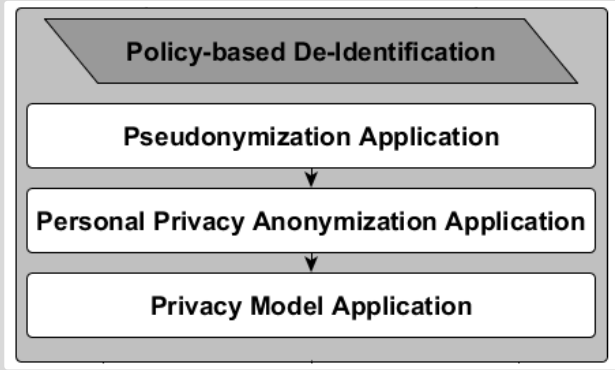
From Privacy Policy to De-Identification

Personalization of Privacy Policy

Layered Privacy Language (LPL)

Policy-based De-Identification





Policy-based De-identification

Order is essential to preserve utility and privacy!

FIRST Pseudonymization

- Create Pseudonyms
  - More on Pseudonymization in Chapter 5
- "Preservation" of information

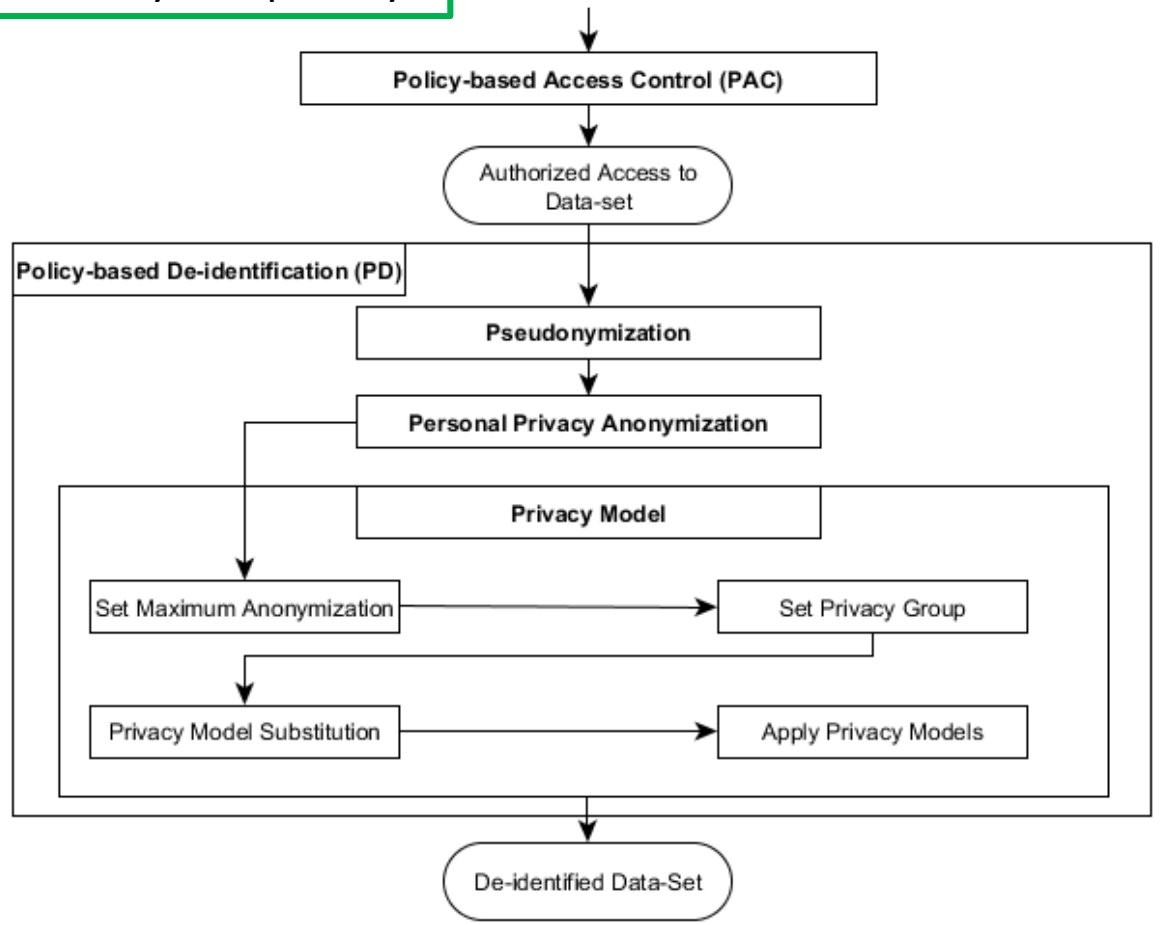
Name	Age	Zipcode	Disease
Alice	25	53711	Flu
Bob	25	53712	Hepatitis
Axel	26	53711	Bronchitis

Pseudonymization

Name	Age	Zipcode	Disease
445A	25	53711	Flu
646B	25	53712	Hepatitis
C33C	26	53711	Bronchitis

Optional: Bijective Mapping

Name	Pseudonym
Alice	445A
Bob	646B
Axel	C33C



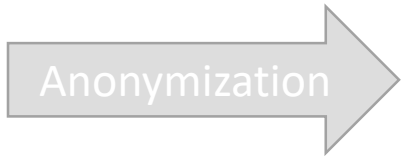
Policy-based De-identification

Order is essential to preserve utility and privacy!

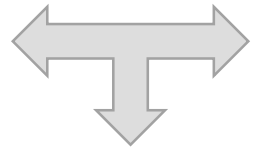
Bad Alternative

- FIRST Anonymization
- SECOND Pseudonymization

Name	Age	Zipcode	Disease
Alice	25	53711	Flu
Bob	25	53712	Hepatitis
Axel	26	53711	Bronchitis



Name	Age	Zipcode	Disease
Ali**	<30	537**	Flu
B**	<30	537**	Hepatitis
Ax**	<30	537**	Bronchitis



Name	Age	Zipcode	Disease
445A	<30	537**	Flu
646B	<30	537**	Hepatitis
C33C	<30	537**	Bronchitis

Name	Pseudonym
Ali**	445A
B**	646B
Ax**	C33C

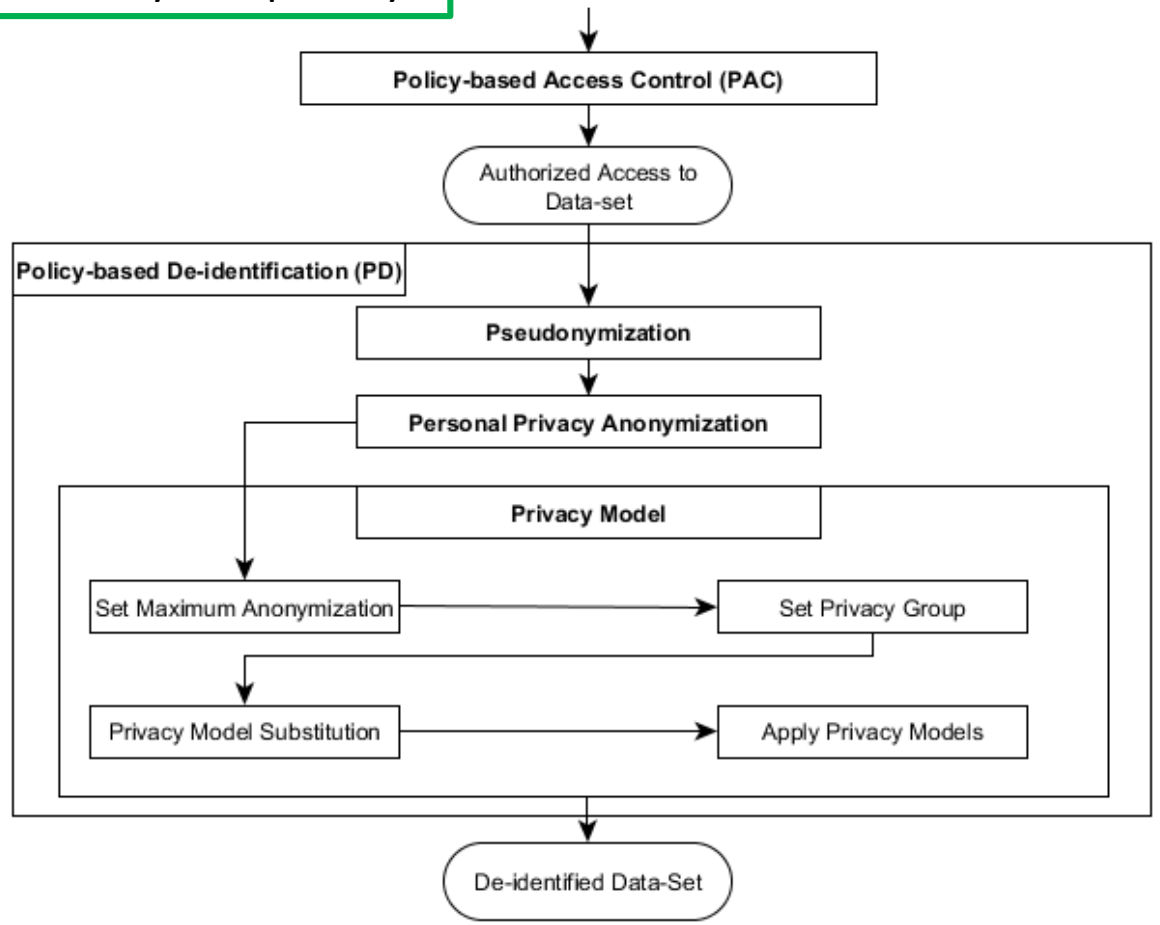
Policy-based De-identification

Order is essential to preserve utility and privacy!

SECOND Personal Privacy Anonymization

- Apply **Personal Privacy** requirements of each user/policy to the Data-set
- Localized Anonymization

- No Data-Set Properties considered!
- No Privacy Guarantees given without PM Step!



Name	Age	Zipcode	Disease
Alice	25	53711	Flu
Bob	25	53712	Hepatitis
Axel	26	53711	Bronchitis

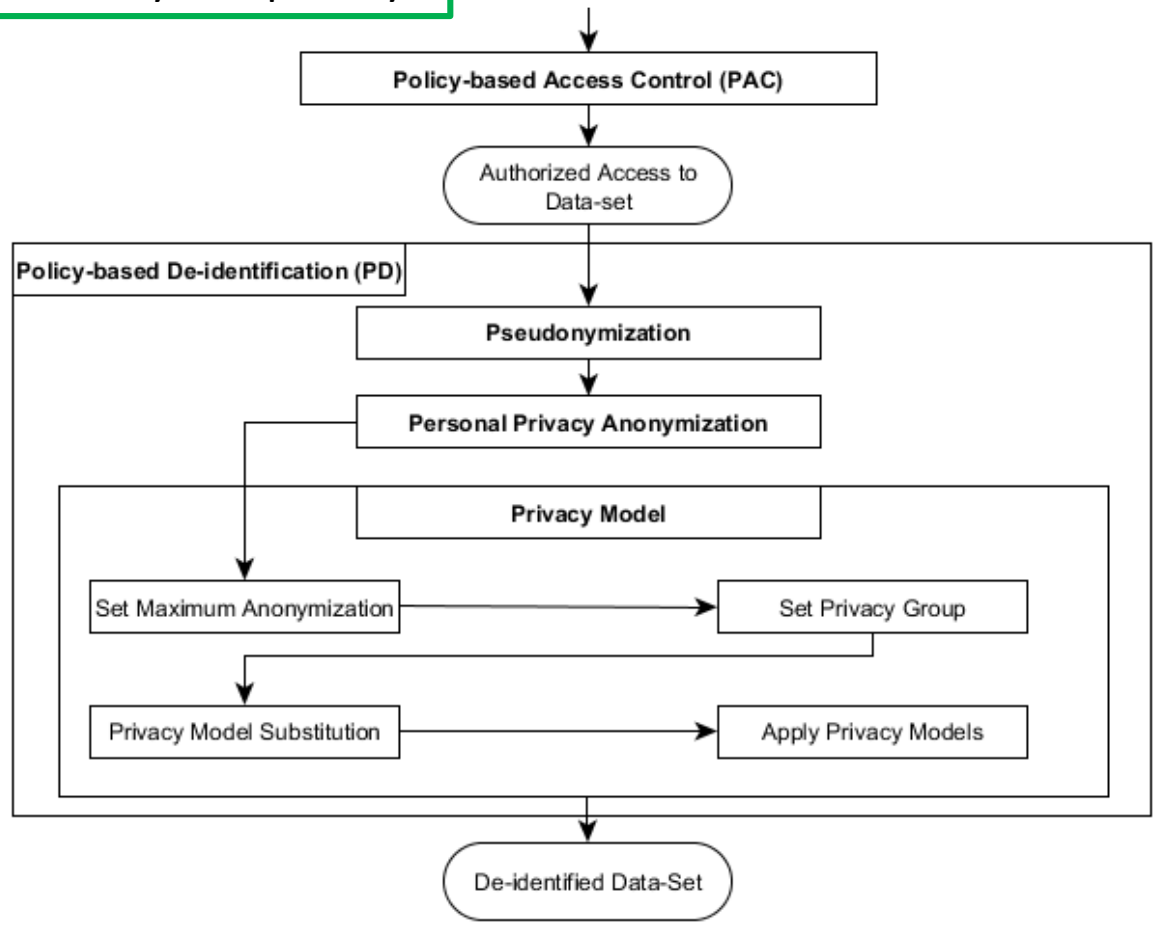
Name	Age	Zipcode	Disease
Alice	20-30	53711	Flu
Bob	25	537**	Hepatitis
Axel	26	53711	Bronchitis

Policy-based De-identification

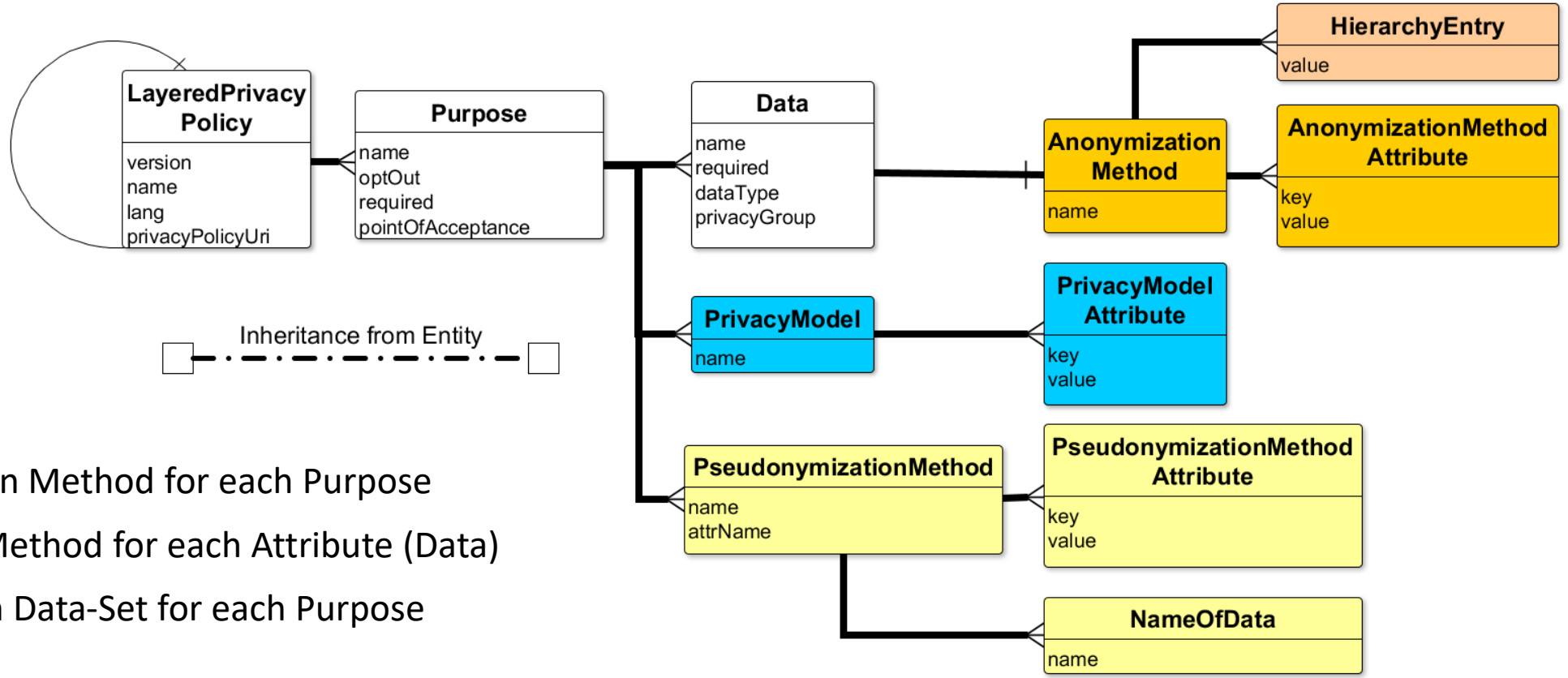
Order is essential to preserve utility and privacy!

THIRD Privacy Model(s) on Data-Set

- Based on all LPL Policies of the Data-Set a “common minimum privacy guarantee” is defined
- Set Maximum Anonymization
  - Limits Anonymization Level for each Attribute
- Set Privacy Group
  - Defines if Attribute is EI, QI, SA or NSA
- Privacy Model Substitution
  - Defines (Set of) Privacy Model to be guaranteed
- Apply Privacy Models
  - Executes Anonymization to achieve Privacy Model(s)



Defining De-identification Enforcement in LPL



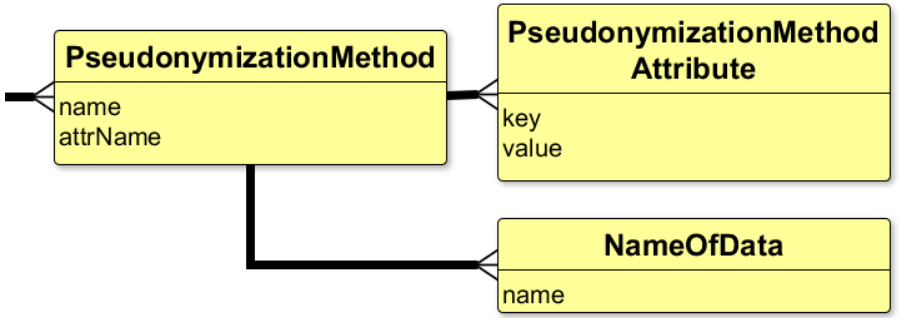
Definition of

- Pseudonymization Method for each Purpose
- Anonymization Method for each Attribute (Data)
- Privacy Model on Data-Set for each Purpose

PseudonymizationMethod

Pseudonymization Method for each Purpose

- PseudonymizationMethod:
  - PID, SHA-1, etc. (see Chapter 5)
- PseudonymizationMethodAttribute:
  - Pseudonymization Method Specific Parameters
  - E.g. a secret key, seed, or other parameters
- List of NameOfData
  - Defines attributes that the method is applied to



Example for "age" of Alice:
PseudonymizationMethod: (name="PID",
PseudonymizationMethodAttribute: {(key="seed", value="11657464985987654 5289")},
NameOfData: {(name="Name)})

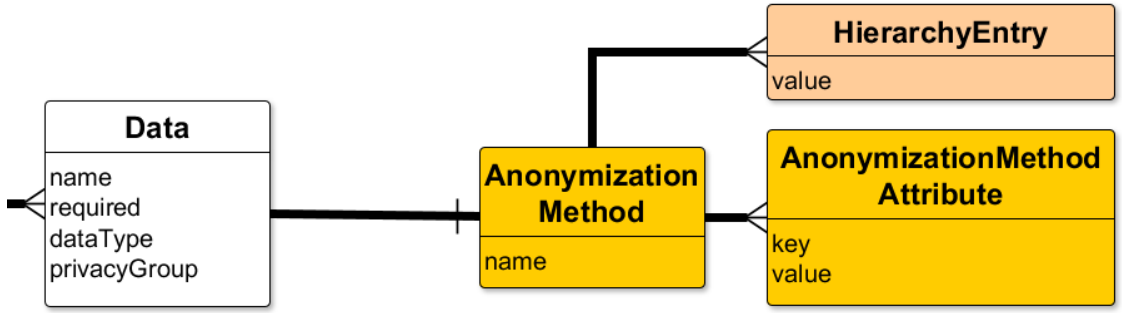
Name	Age	Zipcode	Disease
Alice	25	53711	Flu
Bob	25	53712	Hepatitis
Axel	26	53711	Bronchitis

Name	Age	Zipcode	Disease
445A	20-30	53711	Flu
646B	25	53712	Hepatitis
C33C	26	53711	Bronchitis

AnonymizationMethod

Anonymization Method for each Attribute (Data)

- AnonymizationMethod:
  - Generalization, Suppression, Deletion, etc.
- AnonymizationMethodAttribute:
  - Defines “min” and “max” Anonymization Level
- List of HierarchyEntry
  - Defines Hierarchy for specific attribute value
- Remember: Each LPL Policy for 1 User



Example for “age” of Alice:
AnonymizationMethod: (name=“Generalization”,
AnonymizationMethodAttribute: {(key=“min”, value=“1”)},
HierarchyEntry: {value=“25”, value=“20-30”, value=“<50”})

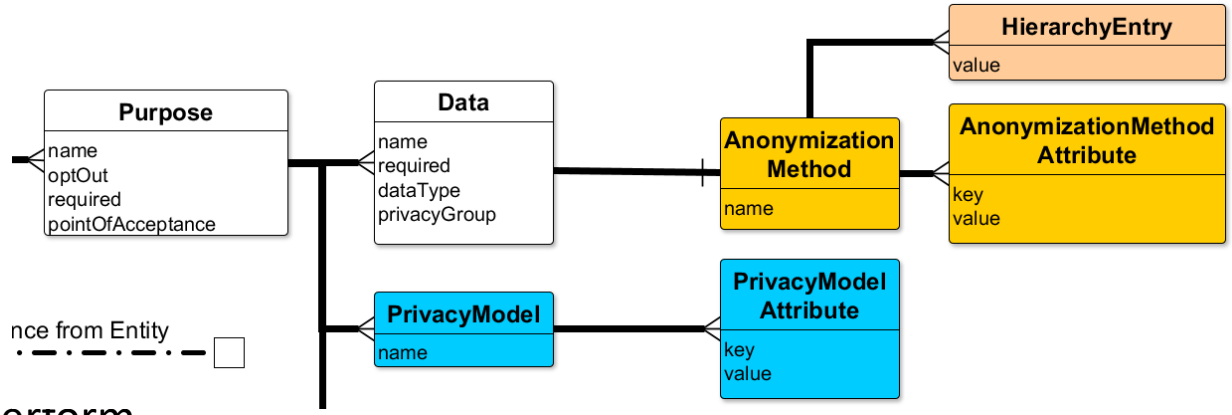
Name	Age	Zipcode	Disease
Alice	25	53711	Flu
Bob	25	53712	Hepatitis
Axel	26	53711	Bronchitis

Name	Age	Zipcode	Disease
Alice	20-30	53711	Flu
Bob	25	53712	Hepatitis
Axel	26	53711	Bronchitis

PrivacyModel

Privacy Model on Data-Set for each Purpose

- PrivacyModel
  - k-Anonymity, l-Diverstiy, etc.
- PrivacyModelAttribute:
  - Defines Privacy Model Attributes/Konfiguration
- Anonymization Method and Hierarchy are used to perform Anonymization
- Multiple Privacy Model guarantees can be defined!



Example:
PrivacyModel: (name="k-Anonymity",
PrivacyModelAttribute: {(key="k", value="3")},

Name	Age	Zipcode	Disease
Alice	20-30	53711	Flu
Bob	25	53712	Hepatitis
Axel	26	53711	Bronchitis

Name	Age	Zipcode	Disease
	20-30	537**	Flu
	20-30	537**	Hepatitis
	20-30	537**	Bronchitis

SPECIAL Approach

- SPECIAL defines “Usage Policy”: specification of a set of authorized operations
- In the minimum core model (MCM) specifies authorized operations:
  - the data processed by the operation
  - the purpose of the operation
  - a description of the operation itself (e.g. “query”, “classification”, “disclosure”, etc.)
  - a description of where the result is stored and for how long
  - the entities that can access the result of the operation (recipients)
- That is, in abstract, mathematical terms a usage policy is just a set of tuples like

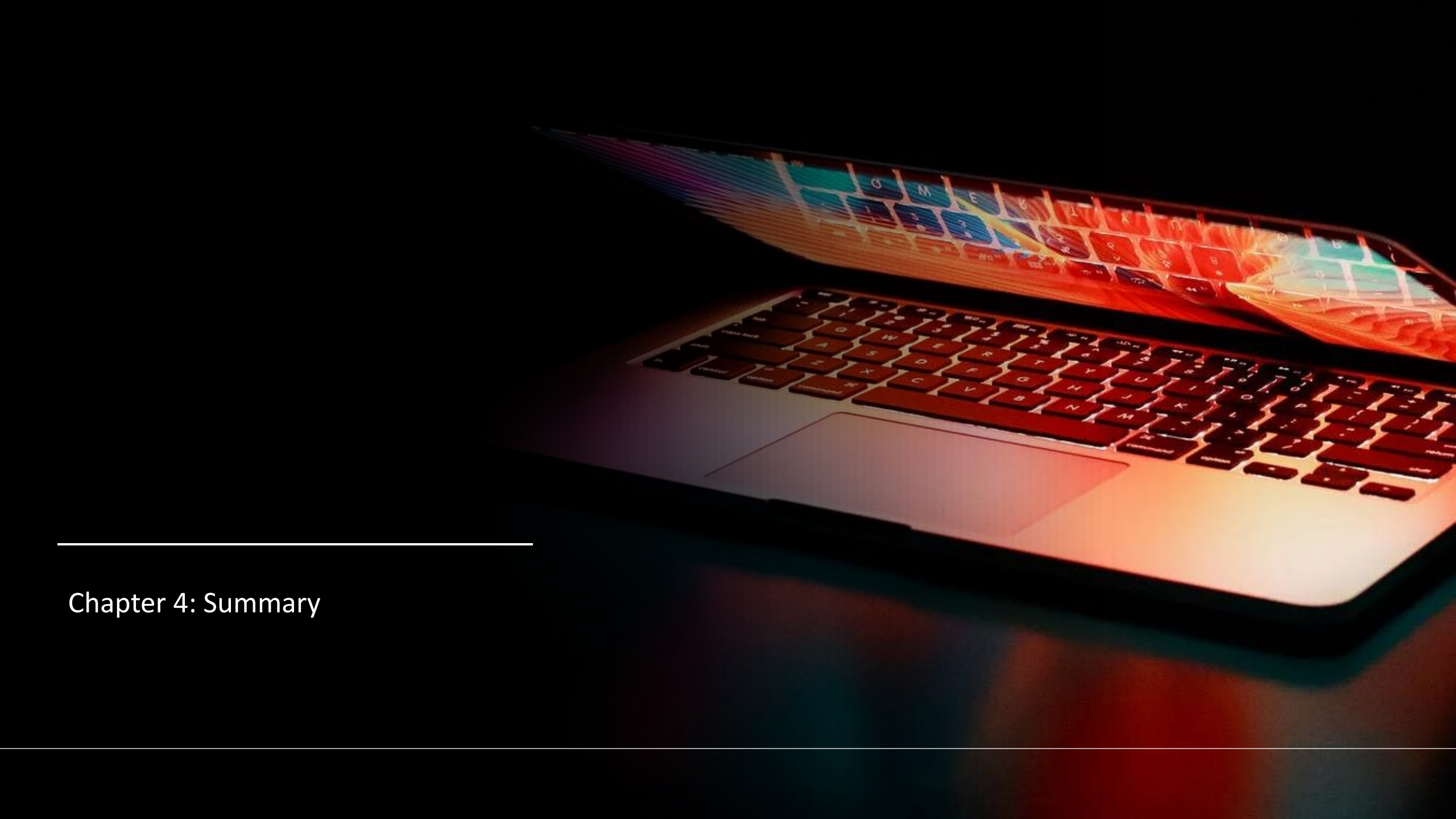
  <data item,purpose,operation,storage,recipiention>
- that will be called authorizations, each of which specifies a permitted operation.

- **Definition of Anonymization** are supposed to be intersected with a Data Category that specify data contents, as in

  ObjectIntersectionOf(Demographic kAnonymous DataSomeValueFrom(has_k "10"|xsd:integer))
- that denotes 10-anonymous demographic data

What is the difference in the approach to LPL?

https://specialprivacy.ercim.eu/images/documents/SPECIAL_D25_M21_V10.pdf



Chapter 4: Summary

Summary

- Introduction of Privacy Models
- Perturbative and Non-Perturbative Approaches
  - Several Methods (Data Swapping, Noise Addition, etc.)
  - Advantages and Disadvantages of the Approaches and Methods
- Privacy Models
  - K-Anonymity
  - L-Diversity
  - T-Closeness
  - Differential Privacy
- Utility Measures
- Definition of De-identification Methods in Privacy Languages
  - LPL
  - SPECIAL

Overview of Lecture Topics

We are here

Chapter	Est. Extent
Chapter 1: Introduction	~1 Lecture
Chapter 2: From GDPR to Privacy Languages	~3 Lecture
Chapter 3: Basics on Data Anonymization in IS	~1 Lecture
Chapter 4: Privacy Risks and Anonymization Techniques	~4 Lectures
Chapter 5: Privacy in Health-Care	~2 Lectures
Chapter 6: Privacy in Data Warehouses	~2 Lectures
Chapter 7: Privacy in Social Networks	~2 Lectures
Chapter 8: Current Research and Outlook	~2 Lectures
Exam Preparation Lecture	1 Lecture