2023-2024

Data and Information Quality

Luca Gerin

Politecnico di Milano

2023-2024

Contents

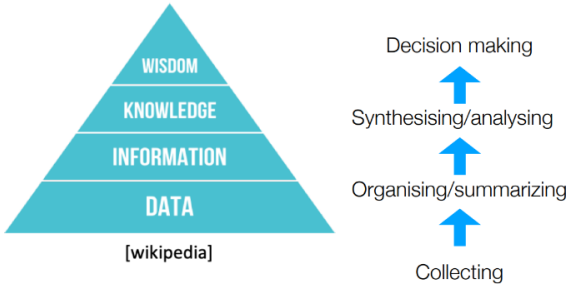
1. Introduction to Data Quality

The relevance of data quality

Companies operate through data, stored in datasets with a *primary purpose*, that is to support operations. But these data can have a *secondary purpose*, that is to be used to perform analytics. Companies aim to exploit the data gathered through the operational systems for the identification of information useful for business decisions. This is possible because there are a lot of data available and there is the technology to analyze big amounts of data. There is a shift of importance from the primary purpose to the secondary purpose of data.

Therefore, also data quality gains importance, because it's a deciding factor for a good or a bad analysis result, and thus a business decision.

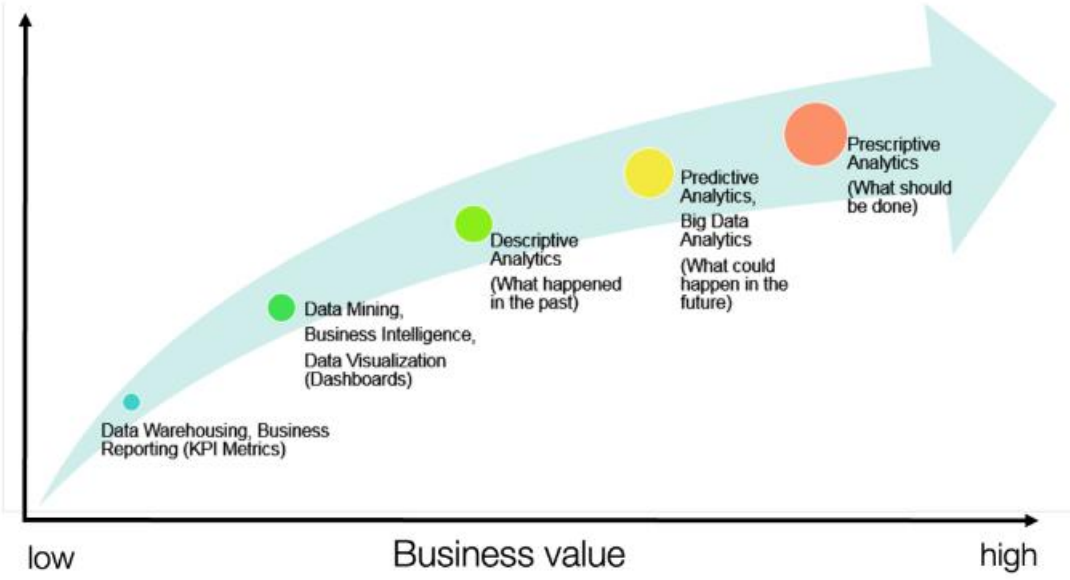
A **Decision Support System** (DSS) comprises the tools and methods to support the business in decision making activities, whit the aim to improve the effectiveness of the company. Thanks to these systems, data can be converted into useful information.



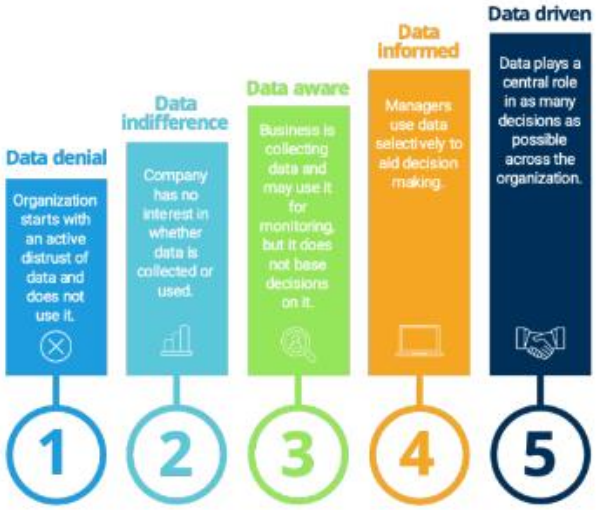
Data-driven Management is characterized by the practice of collecting data, analyzing it, and basing decision on insights derived from the gained information.

Data is collected, processed in various ways, and the results are visualized.

Technology is evolving towards solutions that aim at increasing the business value of data, through predictions based on data and Big Data analysis and prescriptive analytics.



Companies that evolve, eventually mature in the so-called *data-driven* stage, in which the data are used in different ways, from descriptive to diagnostic, predictive and prescriptive. Such companies devote a lot of attention to data collection, tools and technologies to interpret and use data (by making data widely accessible), new ideas and technologies, and ongoing improvement.



The success of data-driven decision-making heavily depends on two factors:

- The **quality of data** collected.
- The **methods used to analyze** data.

The problem is the so-called **Garbage In – Garbage Out** (GIGO) phenomenon: bad quality of the data used as input means bad quality of the output of the problem based on these data.

Thinking that the big quantity of data utilized can overcome the problem of quality is wrong. This is rarely the case: quantity cannot compensate bad quality. In fact, most of the times a small error in data can have high impact (snowball effect) on the output.

Data Quality definition

Traditionally, **data quality** is defined as the fitness for use of data, so the ability of a data collection to meet the user requirements.

In an Information System, data is considered of high quality when it correctly represents the real world, meaning that there are no contradictions between the real world and the derived user view stored in the Information System.

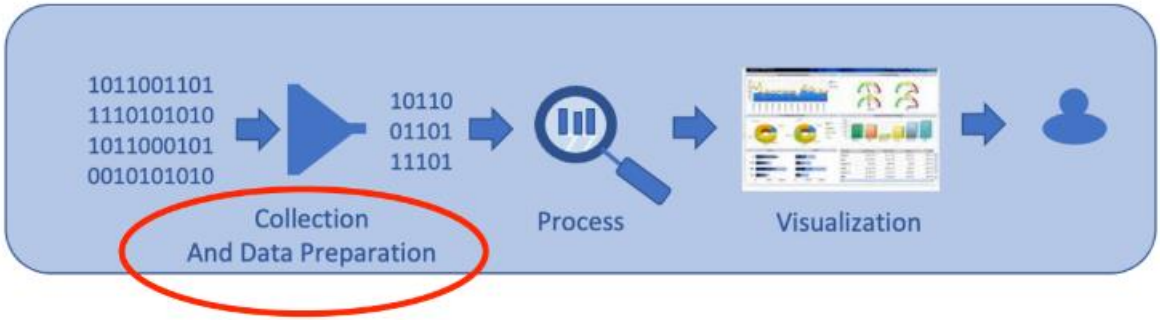
It's worth noticing though that, to assess data quality, it's not possible to directly compare all the data inside the information system with the real world, as this task is often infeasible.

The main causes for poor data quality are:

- Errors introduced by the manual inputting of data into the systems.
- Historical changes in the importance of pieces of data.
- Data usage, as data relevance depends on the process in which data are used.
- Corporate mergers in which difficulties arise during data integration.
- Data enrichment, as external sources might poison the internal data they are added to.

To avoid all these unwanted situations, and adequate architecture for analyzing data is needed.

Of particular importance is the **Collection and Data Preparation** pipeline, which is the set of methods to put in place to check and improve the data quality.



It happens very often, we might say almost always, that real world data is incomplete, inconsistent, and contains many errors. Unfortunately, data preparation, cleaning and transformation usually makes up the majority (90%) of the work in a data mining application.

It's important to note though that the relevance of each defect in a dataset depends on the particular application. One defect might be not significant for an analysis but of critical and deciding importance for another.

2. Data Governance

As already said, data-driven Management is characterized by the practice of collecting data, analyzing it, and basing decisions on insights derived from the information.

Nowadays, with the availability of both a lot of data and capabilities with them, a data-driven company has the following foundations:

- *Data at the center*: data is the core of the information system, and the strategy to provide business value is based on the collection of high volumes of data.
- *Data Culture*: the collective behaviors and beliefs of people who value, practice, and encourage the use of data to improve decision-making.
- *Data Governance*: the organization constantly defines the policies and data processing rules and checks the quality of data.
- *Dashboards and Analytics*

Data governance is a technical and organizational discipline. It is the practice of organizing and implementing policies, procedures and standards that maximize data access and interoperability for the business mission. Data governance defines roles, responsibilities, and processes for ensuring accountability for and ownership of data assets.

Getting data governance requires data management fundamentals aligned with organizational change. There is also need for rules and transparency about how data is used.

$$\text{Corporate Governance} \supseteq \text{IT Governance} \supseteq \text{Data Governance}$$

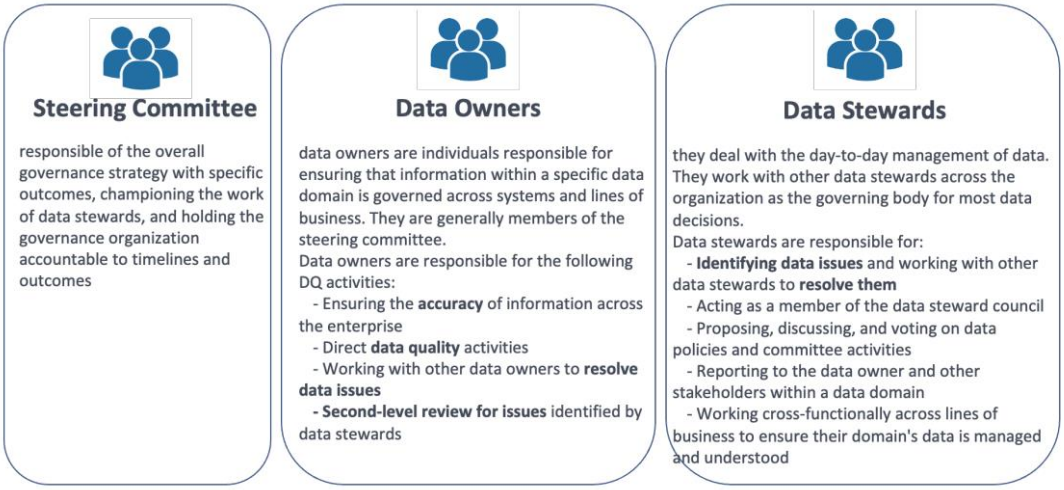
The main concepts behind data governance are:

- Data management
- Enterprise information management
- Data architecture

And the components of data governance are:

- Master data management – *master data* is the static data (for example about products or providers) that is often heavily linked to all transactions, and it needs to be correct and precise because an error would propagate in all the linked transactions.
- Data quality – the degree to which data are suitable for the processes in which they must be used.
- Security – authentication, confidentiality, integrity, data access, security management procedures, etc.
- *Metadata* – data about data, like data provenance.
- Integration – to reconcile different sources.

From a management perspective, it's possible to define three main roles associated to data quality:

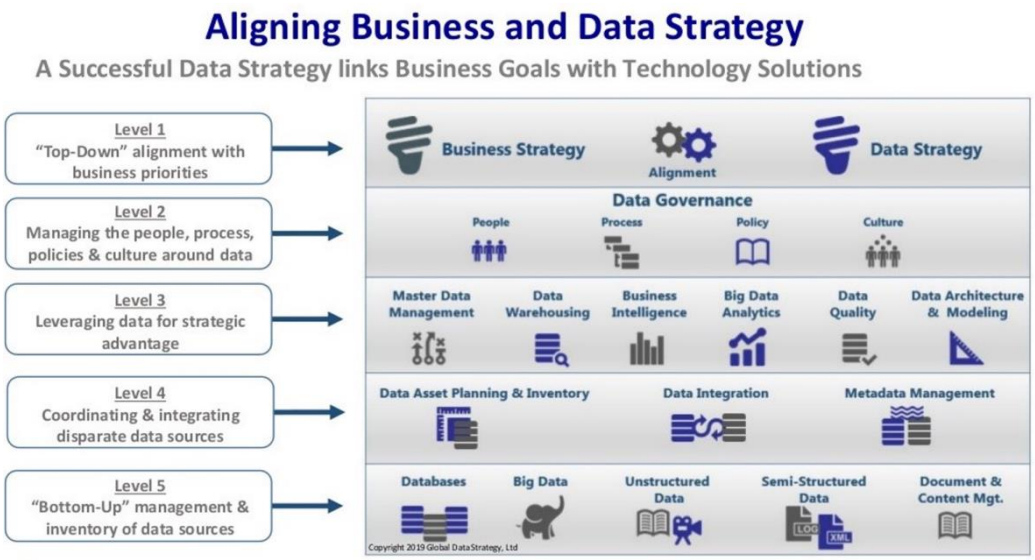


Note that the *data stewards*, who execute processes to guarantee data quality, are the ones who really know the dataset.

So, in summary, the goal of data governance is to establish the methods, set of responsibilities, and processes to standardize, integrate, protect, and store corporate data. More in details, the organizations goals should be:

- Minimize risks.
- Establish internal rules for data use.
- Implement compliance requirements and check periodically the compliance to them.
- Improve internal and external communication.
- Increase the value of data – and to do so data quality must be improved.
- Facilitate the administration of the above.
- Reduce costs.
- Help to ensure the continued existence of the company through risk management and optimization.

A **data strategy** assesses how well an existing data governance plan supports the business.



In practice, data governance builds the data structures that will give business the information it needs when it needs it. Thus, the data governance process acts as a central planning center to coordinate data design across organizations.

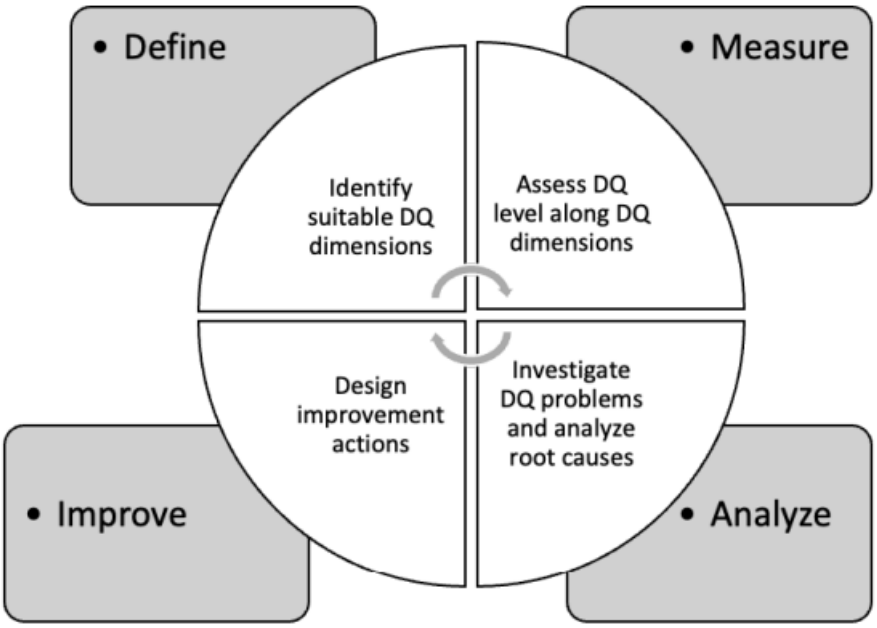
Data quality management

Data quality management provides a context-specific process for improving the fitness of data that is used for analysis and decision making. The goal is to create insights into the health of that data using various processes and technologies on increasingly bigger and more complex data sets.

Data quality management is performed in four phases:

1. Quality dimensions definition
2. Quality dimensions assessment
3. Quality issues analysis
4. Quality improvement

As data changes over time, data quality should be constantly and periodically checked, and this process repeated.



3. Data Quality Dimensions

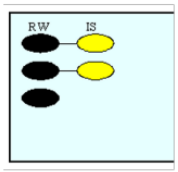
Data quality is measurable, at least partially, and the tools to do it are **data quality dimensions** and **metrics** associated to each dimension as distinct properties.

The selection of the dimensions or metrics to utilize depends on the particular dataset of interest. In addition, different potential problems in datasets are to be addressed with different dimensions.

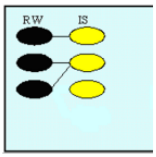
Remembering that a real world system is properly represented if there exists a correspondence between the real world and the information system, the main factors of poor data quality are:

- Incompleteness
- Ambiguity
- Insignificance

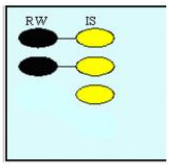
In general, problems may be in data formats, missing values, conflicts, duplicates, wrong values, wrong outliers, etc.



Incompleteness



Ambiguity



Insignificance

Recent classificiations distinguish between 179 different data quality dimensions.

Some of them are **objective** and some are **subjective**, because they regard some aspects that depend on use of the data. While objective measures can be assessed automatically, doing the same is harder for subjective dimensions.

The most used classification for data quality dimensions is the following, proposed by Wang and Strong in 1966:

Data Quality dimensions			
Intrinsic dimensions	**Contextual Dimensions**	**Representational Dimensions**	**Accessibility Dimensions**
Believability Accuracy Objectivity Reputation	Value-added Relevance Completeness Timeliness Appropriate amount of data	Interpretability Ease of understanding Representational Consistency Concise representation	Accessibility Access security

Each data quality dimension captures a specific aspect included under the general umbrella of data quality. More specifically, quality dimensions can refer either to the extension of data, i.e., to data values, or to their intension, i.e., to their schema.

Data and schema dimensions are important too, in fact data quality is given by both the quality of data and the quality of the schema of such data.

Objective dimensions

Accuracy

Accuracy is the extent to which data are correct, reliable and certified. Accuracy is defined as the closeness between a data value v and a data value v', considered as the correct representation of the real-life phenomenon that the data value v aims to represent.

The main difficulties in measuring accuracy are:

- the fact that sometimes the *ground truth* (the real value, to use for comparison to understand if a value is correct) is not available
- even if we have the ground truth, it's difficult to understand the best way to calculate the distance between values in the information systems and values in the real world

The first type of accuracy is the **Syntactic accuracy**: the closeness of a value v to the elements of the corresponding definition domain D. It is needed to check whether v is any one of the values in D, so if the value belongs to the domain.

While it is the easiest kind of accuracy to assess, it assumes that we know the specific domain of interest.

Syntactic accuracy can be measured in two ways:

- *exact matching approach*: check whether a value v is included (1) or not included (0) in D
- *similarity-based approach*: the accuracy value is a value included in the interval $[0,1]$ calculated by means of functions, called *comparison functions*, that evaluate the distance between v and the values in D. An example of similarity measure to use is the *edit distance*.

Id	Title	Director	Year	#Remakes	LastRemakeYear
1	Casablanca	Weir	1942	3	1940
2	Dead Poets Society	Curtiz	1989	0	NULL
3	Rman Holiday	Wylder	1953	0	NULL
4	Sabrina	NULL	1964	0	1985

Roman Holiday

Edit distance =1

The second kind of accuracy is the **Semantic accuracy**, defined as the closeness between a data value v and a data value v'. This kind of accuracy is better to be measured with an exact matching approach.

Id	Title	Director	Year	#Remakes	LastRemakeYear
1	Casablanca	Weir	1942	3	1940
2	Dead Poets Society	Curtiz	1989	0	NULL
3	Rman Holiday	Wylder	1953	0	NULL
4	Sabrina	NULL	1964	0	1985

Weir is a director

but not the right one

Additional knowledge is often needed to measure semantic accuracy. For example, most of the times semantic accuracy consists of looking for the same data in different data sources and finding the correct data by comparisons.

This latter approach also requires the solution of the object identification problem, which consists in understanding whether two tuples refer to the same real-world entity or not. The main issues to be addressed to solve this problem are the identification of tuples that match in some way and the decision strategy to decide whether two matching tuples correspond to the same entity or not.

Accuracy is also related to *duplication* that occurs when a real-world entity is stored twice or more in a data source.

Accuracy might also not be referred to single values of some relation attributes, but in practical cases coarser accuracy definitions and metrics may be utilized. We are talking about aggregating accuracy. As an example, it is possible to calculate the accuracy of an attribute, called *attribute* (or column) *accuracy*, of a relation (*relation accuracy*), or of a whole database (*database accuracy*).

We define the Relation accuracy as the ratio between accurate values and the total number of values.

Completeness

The **completeness** of a table characterizes the extent to which the table represents the corresponding real world. The question it answers is whether the data in the information system has a representation for everything in the real world.

Completeness in the relational model can be characterized with respect to:

- the presence/absence and meaning of null values
- the validity of one of the two assumptions of *open world assumption* (OWA) and *closed world assumption* (CWA).

To evaluate completeness, we need to understand what a NULL value means, thus why a given value is missing from the database. That value could be not existing, it could exist but be unknown, or we could not know whether it exists or not. It is important to understand why the value is missing.

According to the **closed world assumption** (CWA), if a tuple is not present in the dataset, then that entity doesn't exist in the real world. The absence of an entity or a value in the information system means that that entity doesn't exist, and in general what is not currently known to be true, is false. When evaluating the completeness with the closed world assumption then, we consider the NULL value whether as representing something not-existing or a mistake in the representation, and we can define completeness as:

$$C_{CWA}(r) = \frac{\#values}{\#expected\ values} = \left(\frac{\#nonNull}{\#Cells}\right)$$

However, in a model with null values, the presence of a null value has the general meaning of a missing value, i.e., a value that exists in the real world but for some reason is not available.

According to the **open world assumption** (OWA), we can state neither the truth nor the falsity of facts not represented in the tuples of a dataset.

Given the relation r, the reference relation of r, called $ref(r)$, is the relation containing all the tuples that satisfy the relational schema of r, i.e., that represent objects of the real world that constitute the present true extension of the schema.

$$C_{OWA}(r) = \frac{|r|}{|ref(r)|}$$

Consistency

The **consistency** dimension captures the violation of semantic rules defined over (a set of) data items, where items can be tuples of relational tables or records in a file.

Inconsistency is then given by the violation of rules in the dataset. To check consistency, we generally need to check row-by-row that the rules are respected.

Semantic rules can be integrity constraints, data edits, or business rules.

Integrity constraints are a set of rules used to maintain the quality of information. They ensure that the data insertion, updating, and other processes are performed in such a way that data integrity is not affected.

There are two main categories of integrity constraints:

- Intra-relation constraints: can regard single attributes (also called domain constraints) or multiple attributes of a relation. They have to do with accuracy.
- interrelation constraints: involve attributes of more than one relation.

Among integrity constraints, the following main types of dependencies can be considered:

- *Key dependency*: for a subset of attributes, a key dependency holds if no two rows have the same key values.
- *Inclusion dependency* (referential constraint): if some columns are contained in other columns or in the instances of another relational instance.
- *Functional dependency*: given a relational instance r, let X and Y be two nonempty sets of attributes in r. r satisfies the functional dependency $X \rightarrow Y$, if the following holds for every pair of tuples $t1$ and $t2$ in r:

$$If\ t1.X = t2.X;\ then\ t1.Y = t2.Y$$

AB—>C

A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_1	c_1	d_2
a_1	b_2	c_3	d_3

r_1

A	B	C	D
a_1	b_1	c_2	d_1
a_1	b_1	c_1	d_2
a_1	b_2	c_3	d_3

r_2

Where data are not relational, consistency rules can still be defined: **Data edits** are rules which denote error conditions. For example, in questionnaires data editing is defined as the task of detecting inconsistencies by formulating rules that must be respected by every correct set of answers.

Business rules are instead ad-hoc rules and constraints defined for the specific business case interested by the dataset.

Timeliness

Timeliness is the extent to which data are sufficiently up-to-date for a task, it's the validity in terms of time of the data.

Timeliness has two components:

- **Age** or **currency** is a measure of how old the information is, based on how long ago it was recorded.
- **Volatility** is a measure of information instability, the frequency of change of the value for an entity attribute.

Considering the volatility as the average time the data are valid, we can compute the timeliness as:

$$T = \max\left(0\,;\ 1 - \frac{Currency}{Volatility}\right)$$

Higher the timeliness T, better the validity of data. If T is close to 0, we are close to data "expiration", to data losing its validity.

Other dimensions

In order to access, for example, data on the web, a user needs to access a network, to understand the language to be used for navigating and querying the Web, and to perceive with his or her senses the information made available.

Accessibility measures the ability of the user to access the data from his or her own culture, physical status/functions, and technologies available.

Redundancy, or minimality, compactness, and conciseness refer to the capability of representing the aspects of the reality of interest with the minimal use of informative resources.

Usefulness, related to the advantage the user gains from the use of information. It's a subjective dimension and depends on the application.

Going back to the classification of data quality dimensions, it's now clear that:

- Intrinsic data quality regards capturing the quality that data has on its own
- Contextual data quality considers the context where data are used.
- Representational data quality captures aspects related to the quality of data representation.
- Accessibility data quality is related to the accessibility of data and to a further non-functional property of data access, namely, the level of security.

Data Quality dimensions			
Intrinsic dimensions	**Contextual Dimensions**	**Representational Dimensions**	**Accessibility Dimensions**
Believability Accuracy Objectivity Reputation	Value-added Relevance Completeness Timeliness Appropriate amount of data	Interpretability Ease of understanding Representational Consistency Concise representation	Accessibility Access security

Dimension Name	Type of dimension	Definition
Accuracy	data value	Distance between v and v', considered as correct
Completeness	data value	Degree to which values are present in a data collection
Currency	data value	Degree to which a datum is up-to-date
Consistency	data value	Coherence of the same datum, represented in multiple copies, or different data to respect integrity constraints and rules
Appropriateness	data format	One format is more appropriate than another if it is more suited to user needs
Interpretability	data format	Ability of the user to interpret correctly values from their format
Portability	data format	The format can be applied to as a wide set of situations as possible
Format precision	data format	Ability to distinguish between elements in the domain that must be distinguished by users
Format flexibility	data format	Changes in user needs and recording medium can be easily accommodated
Ability to represent null values	data format	Ability to distinguish neatly (without ambiguities) null and default values from applicable values of the domain
Efficient use of memory	data format	Efficiency in the physical representation. An icon is less efficient than a code
Representation consistency	data format	Coherence of physical instances of data with their formats

Schema quality dimensions

Data quality lies also in the quality of the schema utilized to store data.

Schema accuracy is the correctness with respect to the model, concerning the correct use of the constructs of the model in representing requirements, and the correctness with respect to requirements, concerning the correct representation of the requirements in terms of the model constructs.

Schema completeness measures the extent to which a conceptual schema includes all the conceptual elements necessary to meet some specified requirements.

Schema pertinence measures how many unnecessary conceptual elements are included in the conceptual schema.

Schema minimality considers a schema as minimal if every part of the requirements is represented only once in the schema. Avoidance of redundancies and normalization should be applied in order to guarantee minimality.

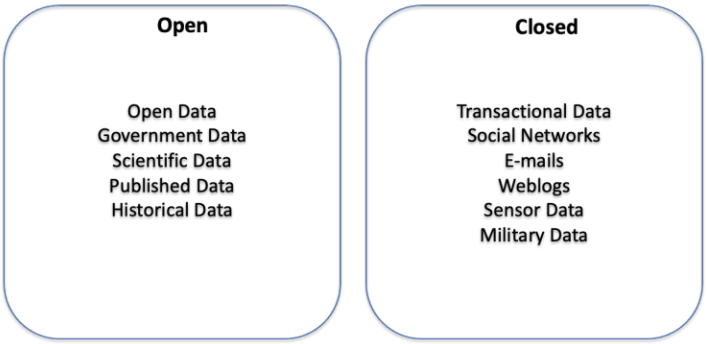
Schema readability measures how readable a schema is, so whenever it represents the meaning of the reality represented by the schema in a clear way for its intended use.

4. Data quality assessment

When a user evaluates the quality of data found on a web page, the first thing he thinks of are the sources of that data (e.g. the sources of what is written on Wikipedia), the timeliness and the completeness.

Each kind of information source has different quality metrics and quality models, as information is available in different formats and is represented according to different models.

We can distinguish between different types of sources:



The fact that there are different types of data sources implies the impossibility to define a unique Data Quality Model. Therefore, different set of dimensions are relevant in different contexts.

For example, for web data we value trustworthiness, credibility and coherence, while for scientific data we value accessibility, easy interchange, measurement properties, precision, etc.

In particular, data **trustworthiness** is based on data provenance. The trustworthiness of a data item is the probability that its value is correct, so probabilistic confidence models are used.

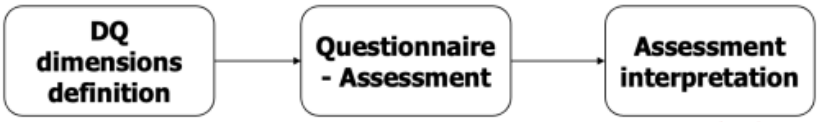
Sometimes the trustworthiness models do not consider the values but the reliability of the values is inferred by other characteristics of the sources.

In any case, whatever the data quality model, data quality dimensions are assessed by giving a data quality score to the data, that can be evaluated at different granularities (The entire database, the schema, some tables, some columns).

Typical assessment techniques are questionnaires and measuring/profiling.

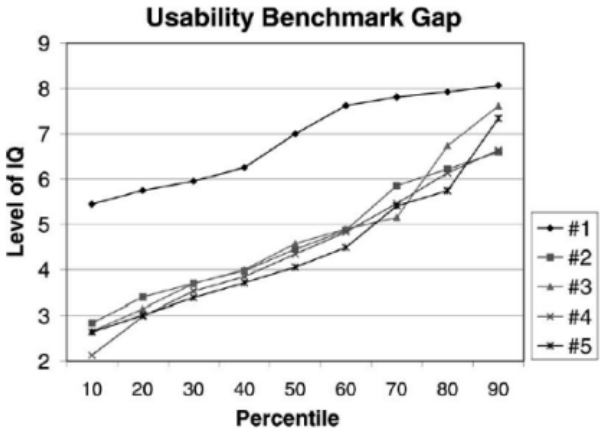
Subjective assessment

When **subjective assessment** is performed, the most used tools are **questionnaires**. So, the users are asked to evaluate the data quality and the collected answers are interpreted to understand the perceived data quality.



One of the problems with this kind of evaluation is that questionnaires are difficult to design properly.

After collecting the answers, some analysis can be performed. The first one is the Gap analysis, that aims to compare the perceived data quality inside a company with the benchmark one.



Another possible analysis is the analysis of the usability role gap, that compares the perceived quality of different kinds of users, consumers, and IT professionals in particular. The ideal scenario would be that users have a perceived data quality higher than the one perceived by IT experts, and not the contrary.

Objective assessment

On the other hand, when **Objective assessment** is performed, objective data quality dimensions are measured automatically.

It's possible to measure, for example, completeness, accuracy, consistency, and timeliness. Another interesting measure to perform is that of the **appropriate amount of data**, that tells if the available data are enough to perform a given analysis.

$$Accuracy = 1 - \frac{\text{Number of data units in error}}{\text{Total number of Not null data units}}$$

$$Completeness = \frac{\text{Number of not null values}}{\text{Total number of values}}$$

$$Consistency = 1 - \frac{\text{Number of violations}}{\text{Total number of consistency checks performed}}$$

$$Timeliness = max\left(0, 1 - \frac{currency}{volatility}\right)$$

$$Appropriate\ amount\ of\ data = min\left(\frac{\text{number of data units provided}}{\text{number of data needed}}, \frac{\text{number of data needed}}{\text{number of data units provided}}\right)$$

The problem with performing such measurements is that they produce several metadata associated with quality, that needs to be stored. Of course, it's inconvenient to store, for each cell of the database, an additional value for every measured dimension in addition to the value itself. The solution to this problem lies in the **aggregation** of the same quality dimension when possible.

We can evaluate completeness and accuracy with different granularities, from the single values to the entire source level; consistency can be evaluated at tuple level but also at source level, while timeliness can be evaluated at single value level, at tuple level or at source level. It's important to

notice, though, that the way the aggregation is done counts: we must be careful to use the average to aggregate on timeliness (averaging 0 with 1 gives an overall of 0,5 that would mean that the value is old but still usable, when in reality half the values are completely out of date and useless), and it would be better to see the minimum that, if equal to 0, would mean that there are out of date values.

There are different ways to aggregate data quality dimensions:

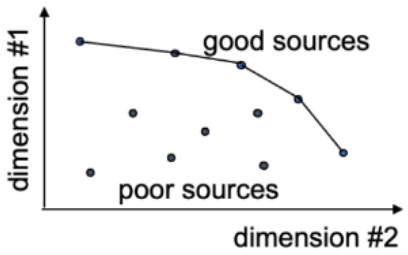
- Ratio (Completeness, Accuracy)
- Average (Completeness, Accuracy, Timeliness)
- Min/Max (Timeliness, Response time)
- Sum (Access costs)
- Product (Availability)

Aggregation can also be done between multiple different data quality dimensions, usually with the aim to compare different data sources regarding the same data to make a choice between them. Aggregation of multiple data quality dimensions gives a sort of score to each of the sources considered.

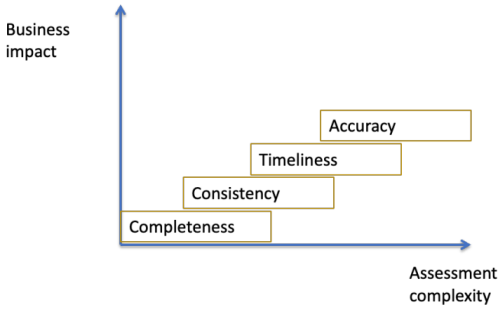
To do so, quality is seen as a vector of dimensions and these vectors, one for each source, are compared using simple additive weighting:

$$DQ(\text{Source } i) = \sum_j w_j v_{ij}$$

Weights are used because some dimensions may be more important than others in the specific context. However, assigning weights in a proper way is a difficult task to perform.



The assessment complexity of a data quality dimension is often higher for dimensions that have a bigger business impact.



Plotting data helps seeing data characteristics and checking value distribution, helping to identify strange behaviors.

Sampling

When a lot of data is available and needs to be worked with, a lot of time and resources are needed to examine everything. It is not always feasible to perform a census (that is examining all the records) of the database.

Under these circumstances, the analysis must use traditional **sampling** methods.

Note that by using sampling instead of analyzing the whole source, we are introducing *uncertainty* in the evaluation we are going to perform.

There are two types of sampling methods:

- **Probability sampling**: the most used approach, in which each unit is drawn from the population with known probability.
- **Nonprobability sampling**: it is not possible to know the probability with which each unit is drawn from the population. In this case there is no way to evaluate the reliability of the results.

The important steps of sampling are:

1. Set the objective of the sampling (e.g., percentage of errors, "I can tolerate at maximum this % of errors").
2. Define the elementary unit and the population (e.g., in a database the unit is the record or the tuple, while the population is the entire table or the entire dataset).
3. Define the degree of precision (amount of error) and the reliability level (that one can accept over repeated samples) required. This value affects the size of the sample. There is always a trade-off between precision, reliability, and sample size.

The sampling methods used for probability sampling are:

- *Simple Random sample*: definition of a random sample of the size required.
- *Systematic sample*: this is a variation of the simple random. The first row is randomly chosen: once the starting row is identified, every k-th row from that row on is included in the sample. (k is the ratio between table size and sample size)
- *Stratified Random sample*: if the quality distribution is not uniform, the parts with more issues should be represented in the sample. Subgroups should be created and each of them should have a uniform distribution of the quality. We take simple random samples from each stratum.
- *Cluster sample*: population is divided into clusters based on specific criteria and a subset of clusters is chosen at random, then all the elementary units can be analyzed, or a random sample form each cluster is inspected.

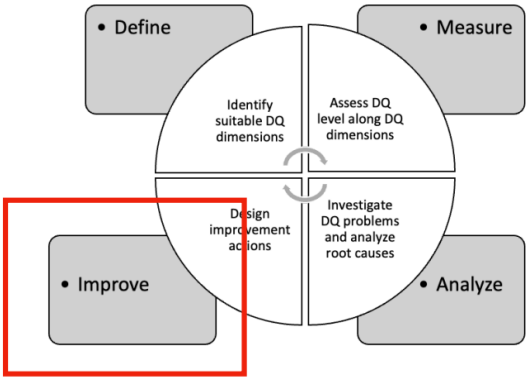
The sample size, and so the number of units to consider in a sample, depends on the error that we are willing to tolerate, so on the desired precision.

$$n = z^2 * p(1-p)/e^2$$

Where p is a preliminary estimate of the proportion, z is the two/tailed value of a standardized normal distribution and e is the desired precision.

5. Data Profiling

Once the data quality problems are understood, data needs to be modified in order to be improved.



There are two main types of improvement strategies:

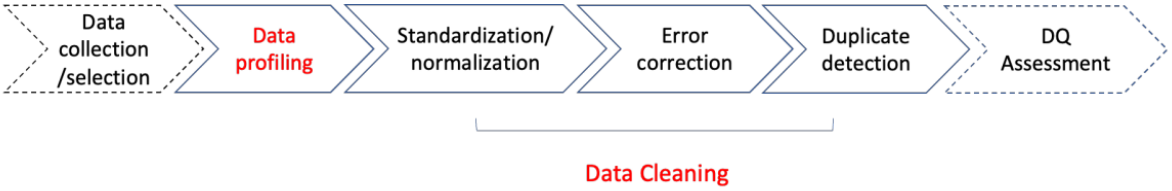
DATA-BASED APPROACHES
They focus on data values and aim to identify and correct errors without considering the process and context in which they will be used.
The dataset is provided as input to these techniques that produce as output the same datasets with a higher quality.
The aim is only to correct errors in data, without considering the context in which data are used, but only the dataset itself.
These approaches are good for the secondary purposes of data.

PROCESS-BASED APPROACHES
They are activated when an error occurs and aim to discover and eliminate the root cause of the error.
The objective is to find the root cause of errors and correct it to avoid such errors happen again and can propagate, by solving the problem originating the data quality issues.

Process-based approaches are the most effective but also the most expensive. A possible and often chosen strategy is to use data-based approaches for static data and process-based approaches for dynamic data.

Data cleaning is the process of identifying and eliminating inconsistencies, discrepancies and errors in data in order to improve quality. Data cleaning procedures are mostly carried out automatically by tools, but in some cases the user's interaction is requested when decisions are to be made on some data.

The steps of data cleaning process are the following:



Note that data cleaning implicates modifying the dataset.

Data quality assessment is performed also after data cleaning, to assess one more time the quality and see if the cleaning was effective.

Before data cleaning, data profiling is always performed, in order to analyze the dataset without making changes but getting information about the context and values and spot most evident errors.

Data profiling is the set of activities and processes designed to take as input the data source and determine and generate the metadata describing the dataset.

Data profiling also helps to understand and prepare data for subsequent cleansing, integration, and analysis, in which the generated metadata are utilized.

Systematic data profiling usually proceeds as follows:

1. The user specifies the data to be profiled and selects the types of metadata to be generated.
2. The tool computes the metadata in batch-mode, using SQL queries and/or specialized algorithms. Depending on the volume of the data and the selected profiling results, this step can last minutes to hours.
3. Results are usually displayed in a collection of tabs, tables, charts, and other visualizations to be explored by the user.
4. The discovered metadata (or the relevant parts of it) are applied to concrete use cases (e.g., data cleaning)

Data profiling is useful in several use cases:

- **Data exploration**: when datasets arrive at an organization and/or accumulate in so-called data lakes, experts need a basic understanding of their content. Manual data exploration can and should be supported with data profiling techniques.
- **Data integration**: often, datasets that need to be integrated are unfamiliar and the integration expert wants to explore the datasets first. Apart from exploring individual sources, data profiling can also reveal how and how well two datasets can be integrated.
- **Data quality/data cleansing**: a frequent and commercially relevant use case is profiling data to prepare a data cleansing process. Profiling results can be used to reveal data errors.
- **Big data analytics**: “Big data” is data that cannot be managed with traditional techniques, which underscores the importance of data profiling. Fetching, storing, querying, and integrating big data is expensive, despite many modern technologies. It might be worthwhile to know about properties of the data one is receiving.

Data Profiling tasks with a single source

Three are the main tasks of data profiling when a single source is involved:

- Single Column Analysis
- Dependence discovery
- Relaxed Dependencies

Single Column Analysis

Single Column Analysis consists in the analysis of the individual columns in a given table, so going column by column, feature by feature, attribute by attribute.

The kind of analysis are about cardinalities, value distribution, pattern and data types, domain classification, summary and sketches.

Cardinalities are numbers that summarize simple metadata. Some of them are:

- Number of rows,
- Number of attributes
- Number of null values
- Number of distinct values

While the number of null values can be used to measure completeness, the number of distinct values can be used to measure the uniqueness dimension (number of distinct values/number of rows) and the entropy of one column (how much the values are different from each other; in an ideal scenario they are neither too similar nor too different).

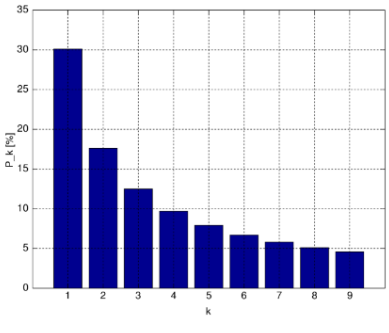
Field Name	NULL	Missing	Actual	Completeness	Cardinality	Uniqueness	Distinctness
Customer ID	0	0	3,338,190	100.00%	3,338,190	100.00%	100.00%
Account Number	0	0	3,338,190	100.00%	3,254,735	97.50%	97.50%
Customer Name 1	50,072	16,690	3,271,428	98.00%	2,997,864	89.81%	91.64%
Customer Name 2	2,450,670	53,077	834,443	25.00%	798,531	23.92%	95.70%
Tax ID	886,703	41,444	2,410,043	72.20%	2,120,837	63.53%	88.00%
Gender Code	1,204,060	50,264	2,083,866	62.43%	8	0.00%	0.00%
Birth Date	627,019	0	2,711,171	81.22%	25,275	0.76%	0.93%
Postal Address Line 1	196,536	5,193	3,136,461	93.96%	2,886,753	86.48%	92.04%
Postal Address Line 2	2,349,569	42,966	945,655	28.33%	875,578	26.23%	92.59%
City Name	171,517	15,171	3,151,502	94.41%	29,876	0.89%	0.95%
State Abbreviation	723,865	0	2,614,325	78.32%	72	0.00%	0.00%
Zip Code	925,591	0	2,412,599	72.27%	48,731	1.46%	2.02%
Country Code	0	0	3,338,190	100.00%	5	0.00%	0.00%
Telephone Number	515,781	0	2,822,409	84.55%	2,624,840	78.63%	93.00%
E-mail Address	1,204,608	0	2,133,582	63.91%	2,037,570	61.04%	95.50%

•**NULL –** count of the number of records with a NULL value
•**Missing –** count of the number of records with a missing value (i.e. non-NULL absence of data e.g. character spaces)
•**Actual –** count of the number of records with an actual value (i.e. non-NULL and non-missing)
•**Completeness –** percentage calculated as Actual divided by the total number of records
•**Cardinality –** count of the number of distinct actual values
•**Uniqueness –** percentage calculated as Cardinality divided by the total number of records
•**Distinctness –** percentage calculated as Cardinality divided by Actual

Value distributions summarize the distribution of values within a column.

A common representation for value distributions is a histogram, from which analysts can check if the values of some columns are (approximately) normally distributed, and the number of outliers. Strange patterns in the distribution can be spotted and can lead to discovering errors.

- The extremes of a numeric column include its maximum and minimum values. These values support the identification of outliers, i.e., datapoints that lie outside an expected range of values.
- The constancy of a column is defined as the frequency of the most frequent value divided by the total number of values. It might reveal the presence of standard values.
- The distribution over the first digits of a set of numeric values is another interesting value distribution. According to Benford's law (1938) the distribution of the first digit follows $P(d) = \log_{10}(d + 1/d)$. Benford's law has been used to detect accounting fraud and other suspicious features and anomalies in numeric datasets.



SQL queries can be used for basic profiling tasks. It's possible to query the data dictionary to gain information about the schema and data types, use MIN(), MAX(), COUNT(DISTINCT x) to inspect the domain of data, using queries like the following to find erroneous data and study default values:

```
SELECT City, COUNT(*) AS Cnt
FROM Customer
GROUP BY City ORDER BY City, Cnt
```

Data profiling tools or python can automatically detect data types, by labeling columns with their one. More advanced tools can identify frequently occurring patterns of values and tools that are trained on specific domains can identify the semantic domain of columns (e.g., identify a column as the column of phone numbers).

A summary of the capabilities of single column analysis, so of what metadata can be provided with profiling when working with single column analysis, is the following:

Category	Task	Task Description
Cardinalities	num-rows	Number of rows
	null values	Number or percentage of null values
	distinct	Number of distinct values
	uniqueness	Number of distinct values divided by number of rows
Value Distributions	histogram	Frequency histograms (equi-width, equi-depth, etc.)
	extremes	Minimum and maximum values in a numeric column
	constancy	Frequency of most frequent value divided by number of rows
	quartiles	Three points that divide (numeric) values into four equal groups
	first digit	Distribution of first digit in numeric values; to check Benford's law
Data Types, Patterns, and Domains	basic type	Numeric, alphanumeric, date, time, etc.
	data type	DBMS-specific data type (varchar, timestamp, etc.)
	lengths	Minimum, maximum, median, and average lengths of values within a column
	size	Maximum number of digits in numeric values
	decimals	Maximum number of decimals in numeric values
	patterns	Histogram of value patterns (Aa9. . .)
	data class	Generic semantic data type, such as code, indicator, text, date/time, quantity, identier
	domain	Semantic domain, such as credit card, first name, city, phenotype

When high volumes of data are involved, tools can produce approximate statistics, by working with samples of the data instead of with the whole available data.

The simplest kind of summary of data is the uniform sample, when sampling is used to build approximate distributions and histograms, approximately count the number of distinct values, and detect data types and frequently occurring patterns. Unfortunately, some statistics such as minimum and maximum values cannot be reliably computed from a uniform sample.

A sketch is instead a small summary of a dataset or a column that can provide approximate statistics. Sketches typically use a combination of counters (uniform or non-uniform), samples, and transform data values to a smaller domain.

Dependency discovery

Dependency discovery is useful to analyze the consistency dimension. Dependencies can be leveraged to detect what are the correct values to expect and to improve the dataset.

Constraints, such as keys, foreign keys, and functional dependencies, are often constraints of a schema and are known at design time. However, many datasets do not come with key dependencies explicitly, which motivates dependency discovery algorithms. It may be the case that dependencies are not known even at design time and need to be discovered later, or that when getting hands on a dataset no specifications are available.

There are three kinds of multi-columns dependency:

- Unique column combination
- Functional dependencies
- Inclusion dependencies

A **unique column combination** (UCC) is a set of attributes that has no duplicates, and therefore can be selected to be a syntactically valid key. By finding unique column combinations, we find candidate columns to be the keys in the dataset.

A **functional dependency** (FD), written as $X \rightarrow A$, asserts that all pairs of records with same values in attribute combination X must also have same values in attribute A. X is a set and can be composed of just one attribute or more.

A functional dependency must hold in the entire dataset, without violations. If this is not the case, then the rule under consideration is not really a functional dependency.

An **inclusion dependency** (IND) $R_i[X] \subseteq R_j[Y]$ over the relational schemata R_i and R_j states that all values in X also occur in Y. This dependency has to do with foreign keys and consists in the case in which all values of some attribute must be included in the values of another attribute. For example, the *StudentIds* in some *Exam* table must also be included among all the students enrolled in the university in the *Students* table.

Relaxed dependencies

Errors in the database make it difficult to discover dependencies, and for this reason **relaxed dependencies** are used in the dependency discovery. New dependencies are defined by relaxing the requirements of existing dependencies.

The first way consists in relaxing the extent to which the dependency holds (it may hold in some subset of tuples but be violated by other tuples). This way, **partial dependencies**, that are rules that hold only for a subset of the data, can be found. These partial dependencies hold if the error is lower than a certain pre-defined threshold. E.g., we allow some percentage of the tuples in a dependency not to respect the dependency rule, and the ones respecting it are part of the partial dependency. A

popular error measure is the minimum number of records that must be removed to make the partial dependency exact.

Partial dependencies are useful if the data are expected to contain errors, that lead to functional dependencies not to be recognized. Or the other way around, partial dependencies can be used to find functional dependencies that contain errors, and measure how many errors are there.

Another way to relax the extent of a dependency is to consider **conditional dependencies** (CDF), that are partial dependencies that explicitly specify conditions to restrict their scope. This way, we look for functional dependencies that hold for only a subset of the data, but this subset also needs to satisfy some conditions. Conditions are typically sets of pattern tuples that summarize satisfying tuples. The set of pattern tuples is called a *pattern tableau*, or simply a *tableau*, and identifies syntactic conditions for satisfying tuples and therefore describe which parts of the data satisfy the underlying dependency.

To assess how often a CFD holds, confidence has been defined as the minimum number of tuples that must be removed to make the CFD hold.

Discovering conditional dependencies is more difficult than discovering exact, approximate, and partial dependencies. In fact, in addition to discovering the underlying dependency, we must also discover the pattern tableaux or pattern tuples.

Another way to relax the dependencies is to relax the corresponding attribute comparison (e.g., two values may satisfy a dependency if they are similar but not necessarily equal).

Metric dependencies relax the comparison method in a way that tolerates formatting differences. In particular:

- *Neighborhood dependencies*: a closeness function is defined for the two attributes in the rule. A rule is defined and values inside an acceptance interval are accepted.
  For example, the relaxed functional dependency age—> salary would hold for all persons in a database as long as the age and salary are similar within some thresholds.
- *Differential dependencies*: this relaxed dependency uses a differential function for each attribute in the form of a constraint.
  For example, in age—>salary the salary difference of two persons must be within some bounds according to their age difference, i.e., if the age difference is 2 years then the salary difference must not be higher than 200 USD.

Note that the bounds in differential dependencies can be variable, while the threshold in neighborhood dependencies is fixed.

Matching dependencies are in some way a generalization of metric dependencies, stating that if any two records $t1$ and $t2$ from instances of $R1$ and $R2$, respectively, are pairwise similar in all X attributes, they are also pairwise similar in all Y attributes. A threshold is always specified.

	Name	Tel	Street	City	Foot
t_1	Aina Bar	12-345	Oktober St.	Berlin	Singaporean
t_2	Jerry's Inn	00-000	Hill Ave.	New York	Mexican
t_3	Katsu Y.	42-911	Katsu Ln.	Tokyo	Sushi

	Label	Phone	Location	Town	Stars
t_1	Aina Bar	12345	Oktober St. 8	Berlin	1
t_2	Star Cafe	01357	Main St. 173	Paris	5
t_3	Katsu Yamo	42911	Katsu Ln. 1	Tokyo	4

1. $tel \approx_{Jaro,0.9} phone \wedge street \approx_{Levenshtein,0.6} location \rightarrow name \rightleftharpoons label.$
2. $name \approx_{Levenshtein,0.7} label \wedge city = town \rightarrow tel \rightleftharpoons phone.$

Order dependencies are dependencies between tuples existing when, if some kind of ordering exists between two attributes of different tuples, then this ordering also exists for other attributes of those tuples. E.g., Given two tuples X and Y, if $x1 < y1$ then also $x2 < y2$.

Order dependencies have been generalized to sequential dependencies, that state that when sorted on X, any two consecutive values of Y must be within a specific range.

Data Profiling tasks with multiple sources

When multiple sources are present, data profiling tools check for:

- Topical overlap: the overlap of topics, topic discovery and topical clustering
- Schematic overlap: the overlap of schemas, schema matching and cross-schema dependencies
- Data overlap: duplicate detection and record linkage

6. Functional and Inclusion dependencies

Dependency discover has some issues:

- The research time is high, in some cases is exponential. Ways to decrease it have to be defined, but there are approaches to minimize the number of comparisons and thus the research time.
- Dependencies should be valid, meaning that it is necessary to test the quality of the results, spending resources and more time.

Functional dependencies

A **functional dependency** (FD), written as $X \rightarrow A$, asserts that all pairs of records with same values in attribute combination X must also have same values in attribute A.

$X \rightarrow A$ is a statement about a relation R: when two tuples have same value in attribute set X, they must have same values in attribute A.

Given a rule $X \rightarrow Y$, there are different types of functional dependencies:

- Trivial: Attributes on Y are subset of attributes on X
  Street, City → City
- Non-trivial: At least one attribute on Y does not appear on X
  Street, City → Zip, City
- Completely non-trivial: Attributes on Y and X are disjoint.
  Street, City → Zip

A functional dependency $X \rightarrow Y$ is minimal if Y does not depend on any subset of X. There is no overlapping between the two columns that have different values and are independent.

Usually, the typical goal is to look for all *minimal completely non-trivial* functional dependencies.

The properties of functional dependencies are the followings:

R1	Reflexivity -> Trivial FDs	$X \supseteq Y \Longrightarrow X \rightarrow Y$ (also $X \rightarrow X$)
R2	Accumulation (Augmentation)	$\{X \rightarrow Y\} \Longrightarrow XZ \rightarrow YZ$
R3	Transitivity	$\{X \rightarrow Y, Y \rightarrow Z\} \Longrightarrow X \rightarrow Z$

These 3 properties (Reflexivity, Accumulation and Transitivity) are also known as the Armstrong-Axioms. If these axioms are satisfied, then the requirements are *sound* and *complete*.

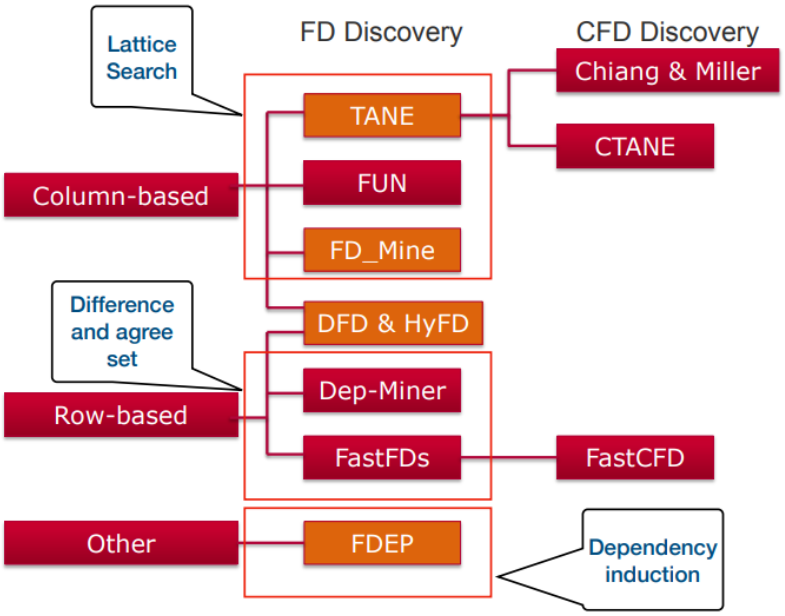
R4	Decomposition	$\{X \rightarrow YZ\} \Longrightarrow X \rightarrow Y$
R5	Union	$\{X \rightarrow Y, X \rightarrow Z\} \Longrightarrow X \rightarrow YZ$
R6	Pseudo-transitivity	$\{X \rightarrow Y, WY \rightarrow Z\} \Longrightarrow WX \rightarrow Z$

To find all minimal, non-trivial FDs $X \rightarrow A$, the algorithm is:

For each column combination X:
  For each $k \epsilon K$
  For each pair of tuples $(t1, t2)$
  If $t1[X \backslash k] = t2[X \backslash k]$ *and* $t1[A] <> t2[A]$*: Break*

We are scanning the column combinations and if we find that the same value on the left side of a functional dependency corresponds to 2 different values on the right side then we stop and go to the next, because in that case no functional dependency holds.

There are different tools for functional dependencies discovery:



One in particular, **Tane**, is based on two key ideas. It reduces tuple sets through partitioning and reduces column combinations through pruning.

Partitioning consists in creating groups of tuples with the same attribute.

Tuple ID	*A*	*B*	*C*	*D*
1	1	a	$	Flower
2	1	A		Tulip
3	2	A	$	Daffodil
4	2	A	$	Flower
5	2	b		Lily
6	3	b	$	Orchid
7	3	C		Flower
8	3	C	#	Rose

Partitions of attributes:

$\pi_{\{A\}} = \{\{1, 2\}, \{3, 4, 5\}, \{6, 7, 8\}\}$
$\pi_{\{B\}} = \{\{1\}, \{2, 3, 4\}, \{5, 6\}, \{7, 8\}\}$
$\pi_{\{C\}} = \{\{1, 3, 4, 6\}, \{2, 5, 7\}, \{8\}\}$
$\pi_{\{D\}} = \{\{1, 4, 7\}, \{2\}, \{3\}, \{5\}, \{6\}, \{8\}\}$

Definition: Partition π *refines* partition π' if every equivalence class in π is a subset of some equivalence class in π'.

Theorem: A functional dependency $X \rightarrow A$ holds if and only if π_X refines π_A.

Thanks to this theory, it's possible to understand if there is a functional dependency by just looking at partitions.

To make an example:

	A	B
1	1	A
2	1	A
3	2	A
4	2	A
5	2	A
6	3	B
7	3	B
8	3	B

$\pi_A = \{12, 345, 678\}$

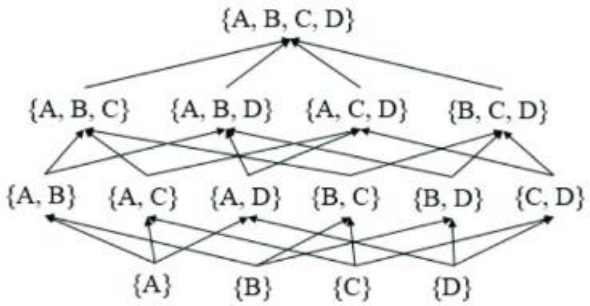
$\pi_B = \{12345, 678\}$

$\pi_{AB} = \{12, 345, 678\}$

The partition on A is a refinement of the partition on the B column. Therefore $A \rightarrow B$.

In general: $X \rightarrow A \Leftrightarrow \pi_X$ refines $\pi_A \Leftrightarrow |\pi_X| = |\pi_{X \cup A}|$

In Tane, there is a bottom-up search in the lattice. At first level, $X \rightarrow Y$ is tested, and then $XX \rightarrow Y$ at the second level and so on. We start from the bottom to consider dependencies that start with an atomic value and then go on with subsequent stages to consider dependencies with more than one value.



When a functional dependency in which the left side is a key or a super-key is found, **pruning** is performed. So, pruning is performed every time we find a key and every time a super-key is detected.

Execution of the algorithm on bridges.csv

```
List of all FDs:  [['A', 'M'], ['A', 'J'], ['A', 'I'], ['A', 'F'], ['A', 'C'], ['A', 'B'], ['A', 'H'], ['A', 'L']
['A', 'K'], ['A', 'E'], ['A', 'G'], ['A', 'D'], ['C', 'B'], ['CD', 'M'], ['CD', 'J'], ['CD', 'I'], ['CD', 'F'], [
D', 'H'], ['CD', 'L'], ['CD', 'A'], ['CD', 'K'], ['CD', 'E'], ['CD', 'G'], ['DF', 'B'], ['DF', 'G'], ['DF', 'H'],
'DF', 'I'], ['DF', 'J'], ['DF', 'K'], ['DEF', 'M'], ['DEF', 'C'], ['DEF', 'L'], ['DEF', 'A'], ['DFL', 'M'], ['DFL
 'C'], ['DFL', 'A'], ['DFL', 'E'], ['DFM', 'C'], ['DFM', 'L'], ['DFM', 'A'], ['DFM', 'E'], ['BDE', 'J'], ['BDL',
'], ['CGJ', 'E'], ['CGL', 'E'], ['CFJ', 'H'], ['CFL', 'H'], ['CFM', 'H'], ['CFJ', 'L'], ['CGJ', 'L'], ['CJM', 'H'
 ['DEK', 'J'], ['DKM', 'I'], ['DKL', 'J'], ['FJK', 'H'], ['BDIL', 'E'], ['BDJM', 'E'], ['BDLM', 'E'], ['BEFJ', 'H
, ['BFJL', 'H'], ['BFKM', 'L'], ['CFGI', 'E'], ['CFGK', 'E'], ['CEFM', 'G'], ['CEFH', 'J'], ['CEFH', 'L'], ['CEFL
 'J'], ['CEFM', 'J'], ['CEFM', 'L'], ['CGHI', 'E'], ['CGHK', 'E'], ['CGIM', 'E'], ['CGKM', 'E'], ['CFGI', 'H'], [
FGH', 'M'], ['CFGI', 'J'], ['CFGI', 'L'], ['CFGI', 'M'], ['CFGL', 'J'], ['CFGJ', 'M'], ['CFGL', 'M'], ['CFHI', 'L
, ['CFHK', 'J'], ['CFHK', 'L'], ['CFIM', 'J'], ['CFIJ', 'M'], ['CFIM', 'L'], ['CFKL', 'J'], ['CFKM', 'J'], ['CFLM
 'J'], ['CFKM', 'L'], ['CGLM', 'H'], ['CGLM', 'J'], ['CHLM', 'J'], ['DGHL', 'E'], ['DGIL', 'E'], ['DGJM', 'E'], [
GLM', 'E'], ['EFGJ', 'H'], ['FGJL', 'H'], ['BDEHK', 'G'], ['BDEIK', 'M'], ['BDHIK', 'G'], ['BDHKM', 'G'], ['BDGIM
 'K'], ['BDIKL', 'M'], ['BEFGJ', 'M'], ['BEFIL', 'H'], ['BEFHK', 'L'], ['BEFIJ', 'M'], ['BEFJK', 'L'], ['BFGIL',
'], ['BFIKM', 'H'], ['BFILM', 'H'], ['CEFHI', 'G'], ['CEFIJ', 'G'], ['CEFIK', 'G'], ['CEFIL', 'G'], ['CEFIK', 'H'
 ['CEFHI', 'M'], ['CEFIK', 'J'], ['CEFIK', 'L'], ['CEFIK', 'M'], ['CEFIL', 'M'], ['CFHIK', 'M'], ['CFIKL', 'M'],
FGILM', 'H'], ['FIKLM', 'H'], ['BEFGHI', 'L'], ['BEFGHK', 'M'], ['BEFGIJ', 'L'], ['BEFGKL', 'M'], ['BEFHIK', 'M']
['BEFIKL', 'M'], ['BFGHIM', 'L'], ['BFGIJL', 'M'], ['EFGHIM', 'L'], ['EFGHKL', 'M'], ['EFGIJM', 'L'], ['EFGJKL',
'], ['EFHIKM', 'L'], ['EFIJKM', 'L']]
```

From the example, it can be seen how the search happens from dependencies with less atomic values and goes bottom up.

It's of course possible to consider relaxed dependencies, using **approximate Tane**. Since the dependency rule we want to consider does not hold for all the tuples, we need to specify the error that defines the acceptance area for dependencies. The definition is based on the minimum number of tuples to be removed from R for $X \rightarrow A$ to hold in R. The error answers the question of how many tuples should be ignored in order for the considered dependency to always hold and thus to effectively be a dependency.

When looking for refinements, we do not search for exact refinements, but for refinements that hold for a certain percentage of tuples.

- π_A = {12, 345, 678}
- π_B = {1, 234, 56, 78}
- π_{AB} = {1, 2, 34, 5, 6, 78}
- $|\pi_B| \neq |\pi_{BA}|$
- e(B→A) = 8/8 − (1+2+1+2)/8 = 2/8

	A	B
1	1	a
2	1	A
3	2	A
4	2	A
5	2	b
6	3	b
7	3	C
8	3	C

A more advanced tool is **FD_Mine**, that is an evolution of Tane. FD_Mine considers and exploits other properties of functional dependencies, like for example symmetry, to eliminate redundancy in the search process and to further prune the lattice.

Example
A→D and D→A ⇒ A↔D

Examination: AB → C and BC → A

Inferred:

– BD → C (property 1)

– BC → D (property 2)
D can be removed from table

	A	B	C	D	E
1	0	0	0	1	0
2	0	1	0	1	0
3	0	2	0	1	2
4	0	3	1	1	0
5	4	1	1	2	4

Different algorithms for dependencies discovery exist and have different performances depending on the properties of the dataset in use. FD_Mine is often quicker than Tane, but not in all cases.

dataSet	Columns	Rows	FDs	Tane	FUN	FD_Mine	Dep-Miner	FastFDs	FDep	DFD
iris	5	150	4	0.6s	**0.1s**	**0.1s**	**0.1s**	**0.1s**	**0.1s**	**0.1s**
balance-scale	5	625	1	0.9s	0.4s	0.3s	**0.2s**	0.5s	0.3s	**0.2s**
chess	7	28,056	1	2.0s	1.0s	3.0s	200.8s	200.1s	202.5s	**0.9s**
abalone	9	4,177	137	1.0s	**0.3s**	1.0s	2.9s	3.0s	4.1s	0.9s
nursery	9	12,960	1	3.1s	1.5s	6.0s	132.0s	131.9s	56.6s	**1.1s**
breast-cancer	11	699	46	1.4s	**0.4s**	1.5s	0.9s	1.0s	**0.4s**	0.9s
bridges	13	108	142	1.3s	0.5s	2.9s	**0.2s**	**0.2s**	**0.2s**	0.9s
echocardiogram	13	132	538	0.8s	**0.1s**	69.9s	**0.1s**	**0.1s**	**0.1s**	1.6s
adult	14	48,842	78	81.2s	150.2s	485.3s	5982s	5946s	760.7s	**6.8s**
letter	17	20,000	61	326s	553.9s	ML	865.4s	853.9s	292.3s	**9.1s**
hepatitis	20	155	8,250	10.9s	321.6s	TL	5363.1s	9.3s	**0.5s**	317.8s
horse	27	368	128,726	5451.s	TL	TL	TL	386.8s	**15.7s**	TL
fd-reduced-30	30	250,000	89,571	**41.1s**	78.4s	TL	391.9s	391.3s	TL	TL
flight	109	1,000	982,631	ML	TL	ML	TL	TL	**213.5s**	TL
plista	125	1,000	178,152	ML	TL	TL	TL	TL	**26.4s**	TL

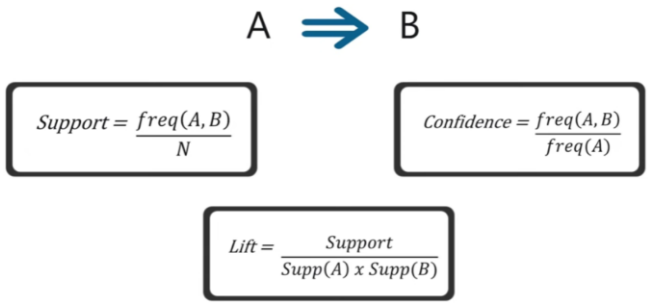
Another way to find dependencies is to exploit **association rules**, that are a data mining model and method that assumes that all data are categorical. Algorithms based on association rules are however generally not good for numeric data.

Given a set of items and a set of transactions, an association rule is an implication in the form:

$X \dashrightarrow Y$, where X, Y are items, and $X \cap Y = \emptyset$

The goal is to find the values that occur most often inside the transactions together, to derive implied association rules. So, an association rule is a pattern that states when X occurs, Y occurs with certain probability.

In mathematical terms, it's possible to define: the **support** as the percentage of transactions containing both the items involved, and the **confidence** as the percentage of transactions containing the first item that also contain the second item.



Or we can write them as:

$$sup = Pr(X \cup Y)$$

$$conf = Pr(Y \mid X)$$

The goal becomes to find all rules that satisfy the user-specified minimum support (*minsup*) and minimum confidence (*minconf*).

```
Best rules found:

 1. outlook=overcast 4 ==> play=yes 4    <conf:(1)> lift:(1.56) lev:(0.1) [1] conv:(1.43)
 2. temperature=cool 4 ==> humidity=normal 4    <conf:(1)> lift:(2) lev:(0.14) [2] conv:(2)
 3. humidity=normal windy=FALSE 4 ==> play=yes 4    <conf:(1)> lift:(1.56) lev:(0.1) [1] conv:(1.43)
 4. outlook=sunny play=no 3 ==> humidity=high 3    <conf:(1)> lift:(2) lev:(0.11) [1] conv:(1.5)
 5. outlook=sunny humidity=high 3 ==> play=no 3    <conf:(1)> lift:(2.8) lev:(0.14) [1] conv:(1.93)
 6. outlook=rainy play=yes 3 ==> windy=FALSE 3    <conf:(1)> lift:(1.75) lev:(0.09) [1] conv:(1.29)
 7. outlook=rainy windy=FALSE 3 ==> play=yes 3    <conf:(1)> lift:(1.56) lev:(0.08) [1] conv:(1.07)
 8. temperature=cool play=yes 3 ==> humidity=normal 3    <conf:(1)> lift:(2) lev:(0.11) [1] conv:(1.5)
 9. outlook=sunny temperature=hot 2 ==> humidity=high 2    <conf:(1)> lift:(2) lev:(0.07) [1] conv:(1)
10. temperature=hot play=no 2 ==> outlook=sunny 2    <conf:(1)> lift:(2.8) lev:(0.09) [1] conv:(1.29)
```

Inclusion dependencies

Inclusion dependencies typically involve more than one relation, and one column of one relation is linked through an inclusion dependency to another column of another relation if all the values of the first column are contained inside the values of the other column.

More formally:

Let D be a relational schema and let I be an instance of D.

R[A1, ..., An] denotes projection of I on attributes A1, ... An of relation R

IND R[A1, ..., An] ⊆S[B1, ..., Bn], where R, S are (possibly identical) relations of D. Projection on R and S must have same number of attributes.

Values of R: "dependent values"
Values of S: "referenced values"

An example of inclusion dependency can be seen considering the "Title" columns in the following tables.

Movies

Title	*Director*	*Actor*
The Birds	Hitchcock	Hedren
The Birds	Hitchcock	Taylor
Bladerunner	Scott	Hannah
Apocalypse Now	Coppola	Brando

Showings

Theater	*Screen*	*Title*	*Snack*
Rex	1	The Birds	coffee
Rex	1	The Birds	popcorn
Rex	2	Bladerunner	coffee
Rex	2	Bladerunner	popcorn
Le Champo	1	The Birds	tea
Le Champo	1	The Birds	popcorn
Cinoche	1	The Birds	Coke
Cinoche	1	The Birds	wine
Cinoche	2	Bladerunner	Coke
Cinoche	2	Bladerunner	wine
Action Christine	1	The Birds	tea
Action Christine	1	The Birds	popcorn

The usual task is to find all maximal and non-trivial inclusion dependencies.

Inclusion dependencies have the following properties:

- Reflexivity $R[X] \subseteq R[X]$
- Transitivity $R[X] \subseteq S[Y]\ and\ S[Y] \subseteq T[Z] \Rightarrow R[X] \subseteq T[Z]$

Inclusion dependencies can be of different types:

- Unary INDs: INDs on single attributes: $R[A] \subseteq S[B]$
- n-ary INDs: INDs on multiple attributes: $R[X] \subseteq S[Y], |X| = |Y|$
- Partial INDs: IND $R[A] \subseteq S[B]$ is satisfied for $x\%$ of all tuples in R or is satisfied for all but x tuples in R
- **Approximate INDs**: IND $R[A] \subseteq S[B]$ is satisfied with probability p
- Prefix/Suffix INDs: IND $R[A] \subseteq S[B]$ is satisfied after removing a fixed (or variable) prefix/suffix from each value of A. So, values are included in another column but just partially (if we do not consider some prefix/suffix)

R	A	B	C
	1	x	1
	2	x	1
	3	y	2
	5	z	4

S	F	G	H
	1	x	1
	2	y	3
	3	z	4
	4	z	4

Unary
$R[C] \subseteq S[F]$

N-ary: $R[B,C] \subseteq S[G,F]$

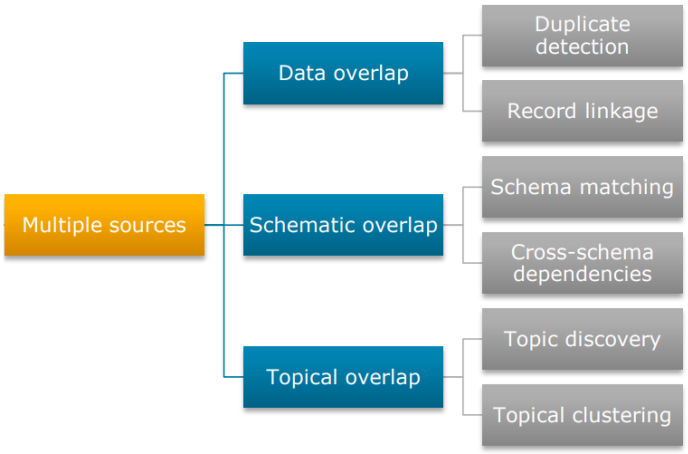
Partial: $R[A] \subseteq_{75\%} S[F]$

Approximate: $R[BA] \subseteq S[GH]$

A	B
bbc	b
	bb

$A \subseteq_s B$
(suffix with variable length)

Profiling for multiple sources



Schematic overlap is detected most of the times manually, by looking at attributes that correspond.

Schema matching is performed to semi-automatically determine cross-schema value correspondences between attributes. Data profiling can extract features from schema and data, compare these features and determine the closeness of two schemata.

Cross schema dependencies include inclusion dependencies across schemata (Join paths between data sources) and conditional inclusion dependencies (typical pattern among cross-referencing sources).

Data overlap is present when two relations contain values related to the same real-world object. When multiple (different) representations of the same real-world entity are detected, we are in front of duplicates. Duplicate detection (with total or only partial overlap) and subsequent record linkage are the performed phases.

Topic overlap is found when two data sources represent the same thing from different domains, using different attributes and data.

The challenges of profiling regard: efficient profiling, scalable profiling, online profiling to interact quickly and with real time results with the tools, profiling query results, profiling new types of data (we have good tools for structured and relational databases, but the same cannot be said for different kinds of databases dealing with semi-structured data), data generation and testing, data profiling benchmark.

7. Data Cleaning

Data cleaning is composed of three steps that must be performed in the order specified in a cyclic way.

Standardization/normalization step can be repeated different times before proceeding to the next step in order to better the results.

Normalization/standardization

- Datatype conversion
- Discretization
- Domain specific

Error correction

Localization and correction of inconsistencies

Missing values

- Detection
- Imputing

Outlier detection

- Model
- Distance

Duplicate Detection

Data transformation and normalization

During this step, different operations are performed.

During **data type conversion**, data are converted to homogeneous types if needed. Afterwards, data is mapped into a common format through **normalization**.

date: 03/01/15 → 01-MAR-2015
currency: $ → €
tokenizing: „Smith, Paul“→ „Smith“, „Paul“

When normalization happens, it’s important not to lose information in the process.

Tokenizing is generally a good practice but must be done carefully. If, for example, some tuples containing the full name of people start with the first name and others start with the last name, tokenizing them could produce mixed results in which first and last names are confused and we would end up with columns with different meaning.

Then, **discretization of numerical values** and **domain-specific transformations** happen, both aimed at eliminating differences in representation for the same values. For example, abbreviations could be eliminated, or everything could be made abbreviated (New York – NY).

Codd, Edgar Frank → Edgar Frank Codd
St. → Street
Address transformation using address databases
Domain-specific product names/codes (e.g. in pharmacy)

Error localization and correction

This second activity can be seen as composed by three phases:

1. Localization and correction of inconsistencies.
2. Localize and correction of incomplete data.
3. Localization of outliers.

Localization and correction of inconsistencies

Data must verify a set of properties, like edit rules, or a check plan, or a compatibility plan.

When such rules are collected, it is crucial that they be proven to be consistent, i.e., without contradictions; otherwise, every conceivable procedure to use edits in order to localize errors will fail. Furthermore, they should be non-redundant. Once we have a valid, so at least consistent, set of edits, we can use them to perform the activity of error localization.

The activity of using edits to correct erroneous fields by restoring correct values is called **error correction** or **imputation**. The main goals of this edit-imputation problem are:

- Minimum change principle: the data in each record should satisfy all edits by changing the fewest fields possible.
- When imputation is necessary, it is desirable to maintain the marginal and joint frequency distribution of values in the different fields.

Localization and correction of incomplete data.

Identifying missing data is generally easy because it only requires to detect null values. The problem is in giving a meaning to the found null values.

However, information may be missing at different levels: at instance level (values, tuples, relation fragments, …) and at schema level (attributes, …).

Talking about instance level, the issues that might arise regard:

- Treating null values: are they missing values or default values?
- Data truncation: in a database, data truncation occurs when data or a data stream is stored in a location too short to hold its entire length, or data are collected leaving intentionally out some categories (e.g., collect data only of people with age>x)
- Data obfuscation: when all elements are available but for some of them a standard value is present instead of the real one
- Biased data (e.g., caused by null values)

Procedures to detect missing values include analysis of the number of null values, comparisons with expected values involving the need of knowledge of the domain, and the analysis of the order of the tuples, followed by the detection that usually happens by analyzing data distribution, but often domain knowledge is required.

Missing values are of course a drawback for the quality of data, and when the analyst or tool decides to solve this issue in a different way than dropping the rows containing missing values, a choice about what to put instead of them must be made.

Imputation is the process of replacing missing data with substituted values. Imputation for some applications is necessary since missing values can introduce bias, make the analysis difficult and decrease efficiency.

To perform imputing of missing values, we can:

- Use Unbiased estimators: to estimate missing values without changing the characteristics of existing dataset (mean, variance, ...)
- Exploit functional dependencies to infer the correct value
- Employ techniques from statistics, like linear regression or techniques for non-linear dependencies based on neural networks and machine learning.

As mentioned, an alternative method to imputation is the deletion of all the tuples that contain a missing value. It easy to implement but the decreases the effectiveness of the analysis.

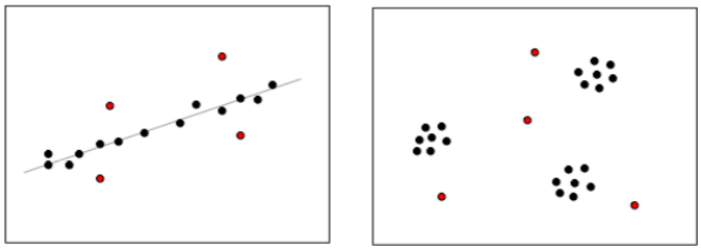
The main imputation methods are:

- Deterministic imputation: It imputes a missing value by using logical relations between variables and derive a value for the missing item.
- Mean/median imputation: the missing values are replaced by the mean/median of the observed values. Also, the Minimum or maximum values can be used, along with other statistical descriptors.
- Regression imputation: this method replaces the missing values by predicted values from a regression of the missing item on items observed for the unit.
- Hot deck imputation (K-NN Imputation): impute for each missing item the response of a randomly selected similar record.
- ML- Based imputation: based on ML methods (e.g., random forest)

Outlier detection

An **outlier** is a "suspicious" observation that deviates too much from other observations. An outlier is then a value that is unusually larger or smaller in relation to other values in a set of data. Sometimes outliers are also values that are close to normal values.

Outliers must be detected, by studying distribution, geometry and time series, and they must be interpreted, to understand if facing a data or observation error or instead a real event. In fact, an outlier is not always an error, but it could be a real value but different from others for some valid motivation.



When finding outliers, if they are not the focus of the analysis, then they can be removed. Otherwise, if they are needed by the analysis or are the focus of it, like for example in heating management or fraud detection problems, they need to be analyzed more closely.

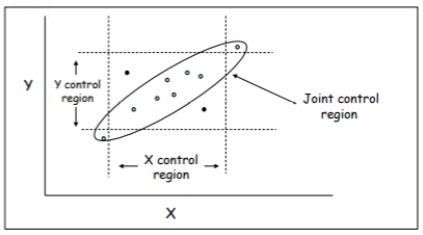
An outlier can be generated because:

- ❖ It is incorrectly observed, recorded, or entered in a dataset.
- ❖ It comes from a different population, in relation to other values.
- ❖ It is correct, but represents a rare event.

False values regard only the first and second situation, and are called data glitches.

Methods for managing outliers are characterized by two phases: first, discovering outliers, and then deciding between rare data and data glitches. Some of these methods are:

- *Control charts*: developed by the manufacturing industry to measure the quality of products, consist of data samples and statistics that are computed and analyzed



- *Distributional outliers*: outliers are seen as points which are in a region of low density; the intuition is that outliers are likely to be at a large distance from the other data points.
- *Time series outliers*: time series are vectors of measurements performed over time. A trajectory can be glitched, or it can make radical but valid changes, and models based on entities past behavior and all entity behavior are employed to identify these deviations.

Once the outliers are identified, we have to decide whether they represent an abnormal but legitimate behavior or a data glitch.

In the time series methods, two different measures of deviation are considered for the decision: the *relative deviation* represents the movement of a data point relative to other data points over time, while the *within deviation* measures the dynamics of a data point in relation to its own expected behavior.

The relative deviation is more robust, since state changes require significant changes in attributes. Instead, the within deviation is sensitive to minor changes and is better for analyzing long-term changes; thus, it is more suitable for discriminating between rare data and glitches. In fact, genuine changes are usually persistent over time, whereas glitches appear and disappear unpredictably.

Patterns in glitches reveal systematic causes, such as data in particular missing intervals.

More methods are available for outlier detection. Among them, we mention: attribute interrelationships, distribution and statistics, geometry methods, distance methods, time series.

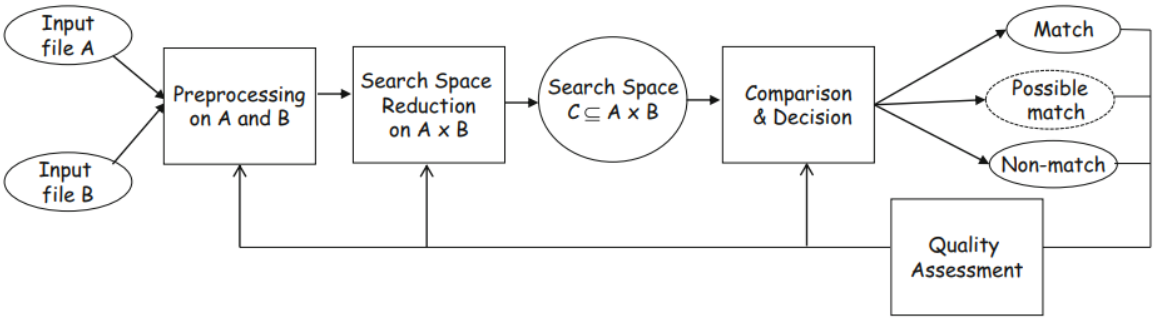
Duplicate detection

Duplicate detection is the discovery of multiple representations of the same real-world object.

Duplicate detection techniques can be also called record linkage, object identification or object consolidation, entity resolution or clustering, merge, purge techniques.

There are some issues to deal with when doing duplicate detection:

- Representations of the same entities are not identical, hence the use of similarity measures
- Data sets are large, and so algorithms to decrease the complexity of the problem are needed



At a high level, the process of duplicate detection starts with the preprocessing of the input data sources, consisting in the other steps of data cleaning, i.e., normalizing and error detection and correction. Afterwards, the complexity of the problem must be reduced by reducing the number of comparisons to perform: the search space $A \times B$ is reduced. The result is a new search space $C \subseteq A \times B$. Comparisons and decisions are then carried out using thresholds for the similarity. The outputs can be “match”, “non-match”, or even “possible match” that means a grey area in the middle of the two in which the tool cannot decide if there is a match or not and the user is asked to make the decision.

At the end, when multiple representations are found, data fusion is performed to have just one tuple representing one real world entity. As last step, data quality is assessed again to decide if to start again the whole data cleaning process from standardization and normalization.

mailBatch for USA : Consumer-Comparison : Examples

Clustering: Individual person level / Household level

Duplicates: Only certain / All

Sandra Powels	3349 North Ridge Avenue	33706	St. Pete Beach		
Powells Sandra	3349 North Ridge Avenue	33706	Pete Beach		
Poweles Donna S.	3347 North Ridge	33706	Saint Pete Beach		
Lowe Ruth-Hanna	25 Peachtree Lane	02114	Boston	10.10.50	(0617)-8845342
Ruthanna Lowe	1201 Oak Street	02132	Boston		0617-8845342
Lowe Ruth Anna		02110	Boston		
Ruth Lowe	1 Becton Drive	21030	Cockeysville	10.10.50	
Johnstone, Jeffrey	3300 Sylvester Rd	92020	El Cajon		
Jeffrey Johnstane	3300 Sylversterroad	92020	El Cajon		
J.R. Johnstone	3302 Sylvester	92020	Cajon		
Jeff Johnston	3300 S. Road	92020	El Cajon		
Gray-David Richard Crewson	Mail Stop, 300 Constitution Drive	33186	Miami		
Richard Crawson	300 constitution drive	33186	Miami		
Crewson, Gray Dave	Mail Stop, Constitution Dr. 301	33186	Miami		
Graham Crewsons	30 Constitution Drive	33186	Miami		
Michael & Nicole Goodman	Regionel Campuses, 711 Lincoln Bldg	10022	New York		
Ph. D. M. Goodnam	711 Lincoln Bldg	10022	New York		
Nicole Goodman	Regional Campuses, 711 Lincoln Bldg	10020	New York		
Michael Goodman	Regional Campuses, 711 Lincoln Bldg	10010	New York		
Mike Goodnan	711 Bldg	10020	New York		
Haddou, Judith Ben	137 Victoriacourt	22153	Springfield		
Benhaddou, Judith	137 S. Viktoria Court	22153	Springfield		
Haddou, Ben	137 Vicctoria Court	22153	Springfield		

Getting more into details, during preprocessing, **standardization** consists of reorganization of composed fields, data type checks, and replacement of alternative spellings with a single one. In the context of object identification, this type of reorganization has the purpose of making comparisons easier.

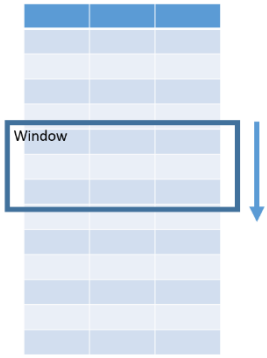
addresses→ StreetName, CivicNumber, City, and State.
dates must be expressed in the same format: 1 Jan 2001, 01-1-2001, and 1st January 2001 should be homogenized to a single format.

Replacement of alternative spellings includes abbreviations that can be replaced by the corresponding complete word, e.g., rd. by road.

Conversion of upper/lower cases, in which data to be compared corresponding to alphabetic strings are transformed to be homogeneous in terms of upper and lower cases.

Schema reconciliation addresses conflicts in schema definition.

Search space reduction can happen through:



- *Blocking*: the file is partitioned in exclusive blocks and comparisons are limited to records within the same block.
- *Sorted neighborhood*: consists of sorting a file and then moving a window of a fixed size on the file, comparing only records within the window.
- *Pruning* (or filtering) has the objective of first removing from the search space all records that cannot match each other, without actually comparing them.

Comparisons functions are usually distance-based:

- String-Based Distance Functions (e.g., edit distance, sounded code, jaro algorithm)
- Item-Based Distance Functions (e.g., Jaccard distance, TF-IDF)

More distance functions are often used together to better understand the similarity, as using only one of them could be more effective in some cases and less effective in others, so using more gives a better overview on the distance between values.

Techniques for the comparison and decision include:

- Probabilistic techniques, based on the extremely relevant set of methods developed in statistics and probability theory, ranging from classical statistical inference to Bayesian networks to data mining approaches.
- Empirical techniques that make use in the different phases of the process of algorithmic techniques such as sorting, tree traversal, neighbor comparison, and pruning.
- Knowledge-based techniques, in which domain knowledge is extracted from the files involved and reasoning strategies are applied to make the process more effective.

8. Data transformation

Data cleaning operations such as outlier and duplicate detection expect the input data to be in the right format and with the right structure. **Data transformation** refers to the data preparation activity of running user-defined programs, rules, or scripts to convert data from one structure into another format or structure.

Sometimes data transformation is also used at the end of the cleaning to better prepare data to be consumed by analytics tools.

Data transformation tasks can be classified in two types:

- **Syntactic data transformations**: aim to transform a table from one syntactic format to another, changing the structure of tables and of data, often without requiring external knowledge or reference data but relying only on the present structure.
  E.g., transforming phone numbers to a standard format, concatenating or splitting columns

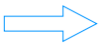
Name	
John	Tel: 7188751243
	Fax: 7188751200
Mike	Tel: 7186359763
	Fax: 7186359700
Frank	Tel: 5198780763
	Fax: 5198780700
Julie	Tel: 5176543809
	Fax: 5176543800

Name	Tel	Fax
John	718-875-1243	718-875-1200
Mike	718-635-9763	718-635-9700
Frank	519-878-0763	519-878-0700
Julie	517-654-3809	517-654-3800

Syntactic transformation

- **Semantic data transformations**: involve understanding the meaning/semantics or the typical use of the data. Therefore, they usually require additional external knowledge, usually referencing external data sources.
  E.g., transforming acronyms into full names requires an external table that contains the full names

Country code
US
CA
CN
DE



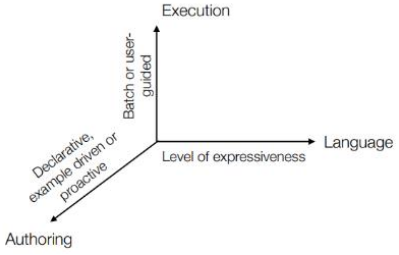
Country
United States
Canada
China
Germany

Semantic transformation

Syntactic data transformations

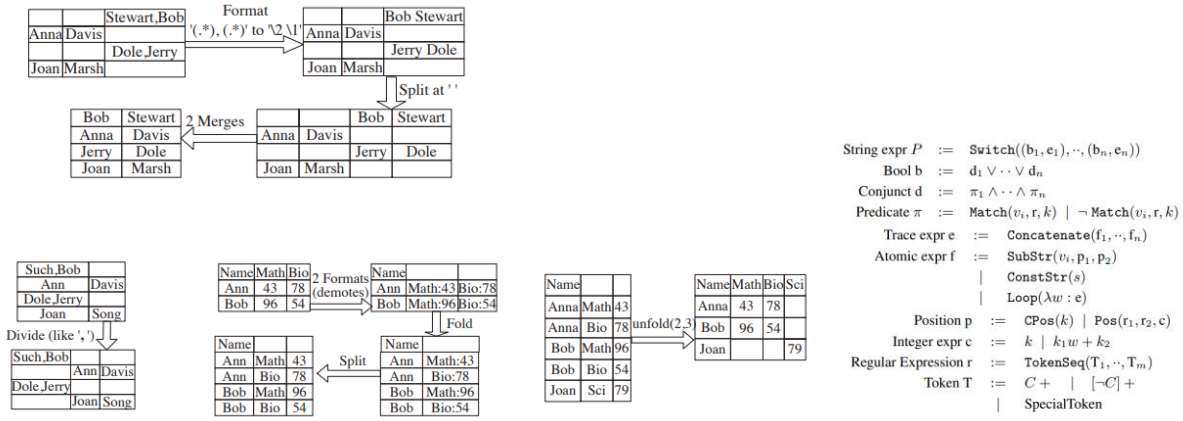
When coming to **syntactic data transformations**, there are three major components/dimensions of a syntactic data transformation system:

- Language (how the transformation is defined, with a certain expressiveness)
- Authoring
- Execution



The **transformation language** limits the space of possible transformations, as it has its own power, or level of expressiveness. In fact, it's the language to define the set of operations allowed on a tablet, like, for example, splitting and merging columns, extracting and manipulating substrings. The language defines expressions that identify the transformations.

A language needs to be expressive enough so that it captures many real-word transformation tasks and at the same time restricted enough to allow for effective and easy authoring of the transformations.



Syntactic transformation tools must allow an easy and effective **authoring** of transformation programs.

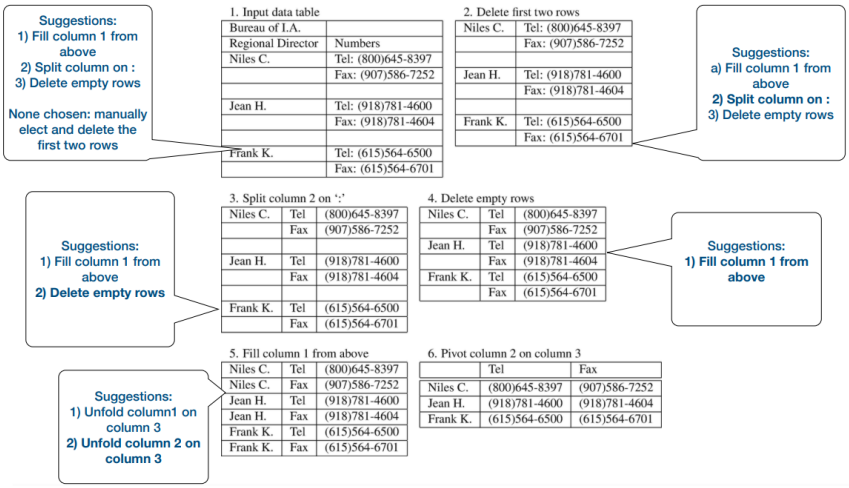
The possible interaction models are:

- *Declarative transformation*: the user specifies transformations directly.
- *Transformation by example*: users give a few input-output examples based on which the tools infer plausible transformations. This modality could however limit the set of allowed transformations.
- *Proactive transformation*: automatically suggests potential transformations without requiring or with only minimal user input.

Some tools, like *Data Wrangler*, that provide proactive transformation mechanisms, suggest the appropriate transformation by considering a suitability score. These tools expect the data to be in a relational format where every column has atomic/simple data type, and each row describes a single entity.

The suitability score can consider as important: column type homogeneity, fewer empty values, or smaller number of delimiters.

Examples of potential proactive suggestions can be: delete all empty columns/rows, delete all mostly empty rows, fill all empty cells with standard values, split columns, et cetera.

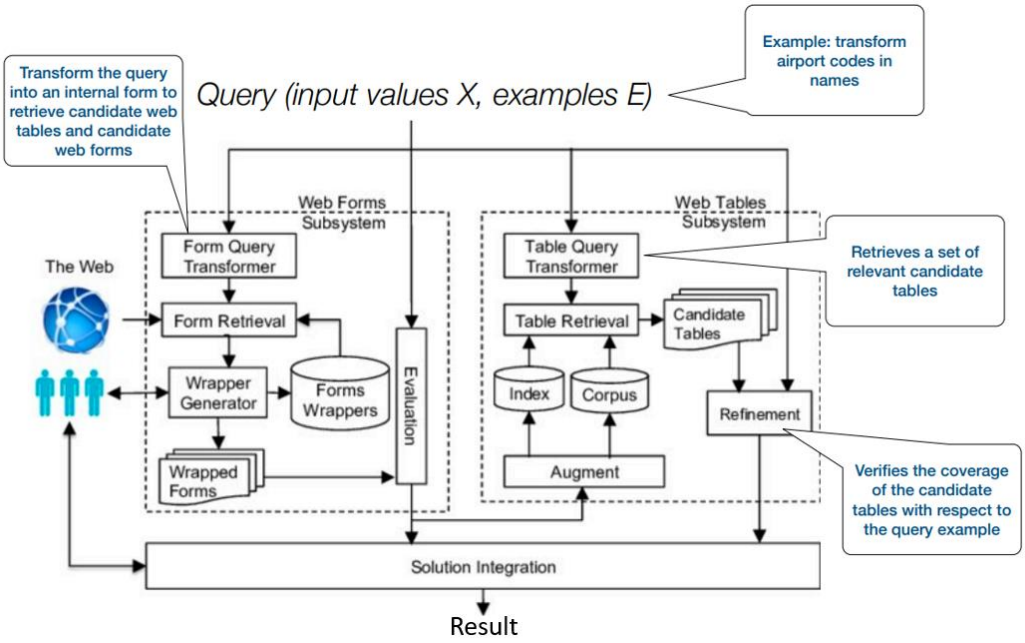


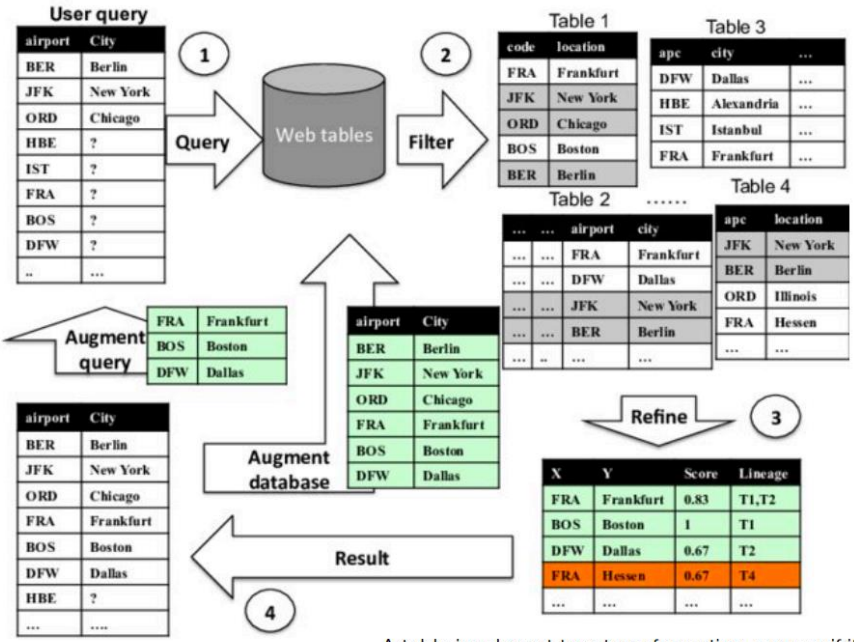
For what concerns **transformation execution**, transformation tools usually offer assistance to the users in selecting the correct transformation. In order to do so, they display the effect of specified transformation on sample data, or they provide natural language descriptions of expected outcomes of the specified transformations to guide the execution of these transformations.

Semantic data transformations

Semantic transformations involve understanding the meanings/semantics or the typical use of data, instead of simply considering it as a set of characters. This means that external data is needed.

An example of a tool that does semantic data transformations is *DataXFormer*, that works by consulting additional external data sources from the web.



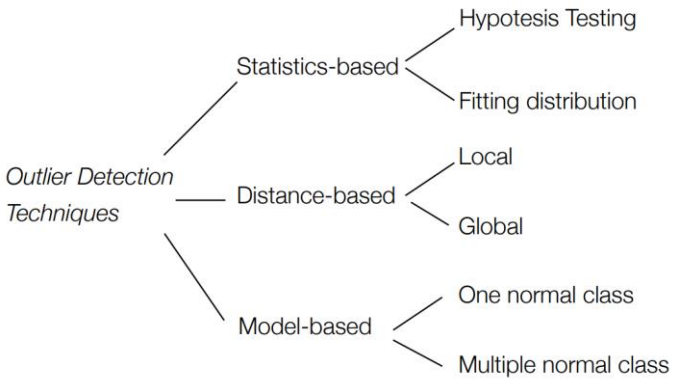


A table is relevant to a transformation n query if it contains at least a certain number of example contained in the query

9. Outlier detection

Outlier detection methods

There are many methods available in literature for outlier detection, divided in three families:



Each family starts from a different assumption and aims at overcoming the limits of the preceding one.

Statistics based outlier detection methods start from the assumption that normal data points would appear in high probability regions of a stochastic model, while outliers would occur in the low probability regions of a stochastic model.

They are divided into two classes:

- Hypothesis testing methods: they calculate a test statistic, based on the observed data points, which is used to determine whether the null hypothesis (there is no outlier in the dataset) should be rejected.
- Fitting distribution methods: they aim to fit a distribution or infer a probability density function based on the observed data (data in low probability areas are then outliers). They can be:
    - Parametric: the assumption is that data follows a normal distribution and aim to find the parameters of the distribution from the observed data (mean and standard deviation).
    - Non-parametric: no assumptions about the distribution, they infer the distribution.

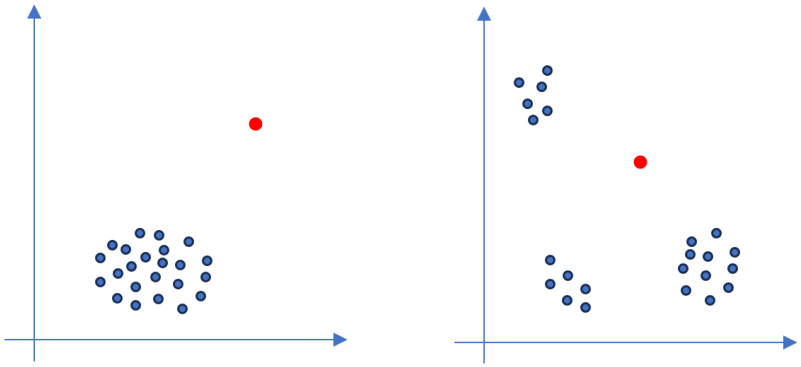
Statistics-based outlier detection methods	
Advantages: ➢ If data follows a specific distribution, statistical outlier detection techniques can provide a statistical interpretation for discovered outliers. ➢ Statistical techniques provide a score or a confidence interval for every data point, rather than making a binary decision. This can be an additional and useful information for making a decision. ➢ Statistical techniques do not need labeled training data.	Disadvantages: ➢ The assumption of an underlying distribution does not hold for high dimensional real datasets. ➢ Even when it holds, the selection of the best statistic is not a straightforward task.

Distance-based outlier detection methods start from the assumption that a normal data point should be close to many other data points. So, these methods define a distance between data points that is used for defining a normal behavior, that outliers do not follow.

On the basis of the reference population we can distinguish:

- Global distance-based outlier detection: they consider the distance between that data point and all the other data points.
- Local distance-based outlier detection: they consider the distance between a point and its neighborhood points.

The choice between Global distance and Local distance depends on the distribution of the data. In some cases one can be more effective than in others.



Distance-based outlier detection methods	
Advantages: ➢ Data driven nature: they are unsupervised and do not make assumption regarding the distribution of data. ➢ It is possible to adapt such techniques to different data types.	Disadvantages: ➢ The techniques fail if outliers have enough close data points. ➢ The computational complexity is also a challenge since we have to compute the distance of every pair of data points. ➢ The performances of the techniques depend on the distance measure. Such measure is complex to find if data are complex (e.g., graphs).

Model-based outlier detection methods start from the assumption that a classifier can be trained to distinguish between the normal data points and the anomalous data points. These Machine Learning based methods learn a classifier model from a set of labeled data points and then apply the trained classifier to a test data point to determine whether it is an outlier or not.

Based on the labels available to train the classifier, model-based approach can be divided into two subcategories:

- Multi-class: labels belong to multiple normal classes.
- One-class: the label refers to one class.

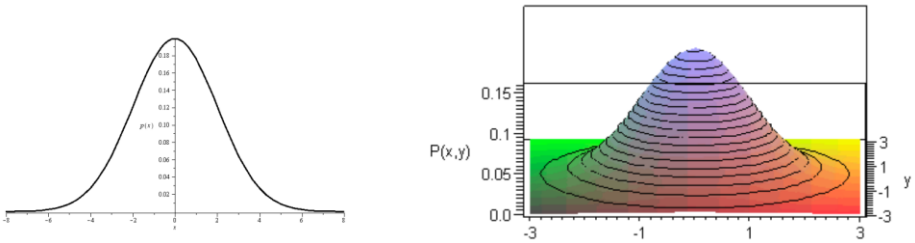
Model-based outlier detection methods	
Advantages: ➢ These techniques can make use of powerful algorithms that can distinguish between instances belonging to different classes. ➢ The testing phase is fast.	Disadvantages: ➢ They must rely on the availability of accurate labels, which is often not possible.

Statistics-based outlier detection

When making use of statistic-based outlier detection mechanisms, we start from the concept of data distribution.

Data practitioners are usually focused on descriptive statistics about data. To find those, statisticians often treat data as a sample of some data-generating process. They aim to find a model or a distribution that better describes the available sample data.

The most used distribution is the Gaussian or normal distribution, that is characterized by a mean μ and a standard variance σ. So, for a normal distribution, we expect most of the values to be in the high probability region and the outliers to be in the tails of the Gaussian curve.



Hypothesis testing refers to the formal procedures for accepting or rejecting hypotheses. This is usually done after some tests are carried out to verify the hypothesis.

1. formulate the null and alternative hypothesis
2. decide the appropriate test and state the relevant test statistic T
3. choose a significance level α, a threshold below which the null hypothesis will be rejected
4. define the critical region (values) for which the null hypothesis is rejected
5. compute from the current data the observed value of the statistic
6. reject the null hypothesis if the observed value falls under the critical region

A frequently utilized test id the **Grubbs test**, devoted to detecting a single outlier. When the outlier is found it is discarded, and the test is repeated until no outliers are detected.

$$G = \frac{\max_{i=1,\ldots,N} |Y_i - \bar{Y}|}{s}$$

Value, Mean, Variance

The Grubbs test statistic is the largest absolute deviation from the sample mean in units of the sample standard deviation.

Note that multiple iterations change the probabilities of detection, and the test is not to be used for a small sample size. In fact, every time an outlier is discarded, the distribution changes and it becomes more difficult to detect more.

The null hypothesis (That there are no outliers) is rejected at a certain significant level α if:

$$G > \frac{N-1}{\sqrt{N}} \sqrt{\frac{t^2_{\alpha/(2N),N-2}}{N-2+t^2_{\alpha/(2N),N-2}}}$$

If this inequality is true, then there are outliers.

As already said, among the *fitting distribution methods*, that instead of testing an hypothesis aim to fit a distribution or infer a probability density function based on the observed data, it's possible to distinguish parametric and non-parametric approaches.

Parametric approaches

The assumption of parametric approaches is that data follows a normal distribution.

Let us consider **univariate** outlier detection, a simple way to identify outliers is to compute a z-score for every x_i, which is defined as the number of standard deviations away x_i is from the mean:

$$z - score(x_i) = \frac{|x_i - \mu|}{\sigma}$$

It's the distance of every value from the mean in terms of variance.

Data values that have a z-score > than a certain threshold are declared to be outlier.

However, the mean is not a robust value, as it is affected by outliers. So, it's possible to use instead **robust statistics**, for example the median and the median absolute deviation (MAD) to replace respectively the mean and the standard deviation.

Masking effect: There is a single data point that has shifted the mean and the standard deviation so much to mask other outliers. The mean is not a robust estimator, so the breakdown point is low.

The Median is the value separating the higher half from the lower half of a data sample, and has a breakdown point of 50%: as long as no more than half the data are outliers, the median will not give bad results. It's not as sensitive as the average of values.

The Median absolute deviation (MAD) is the median of the absolute deviations from the data's median.

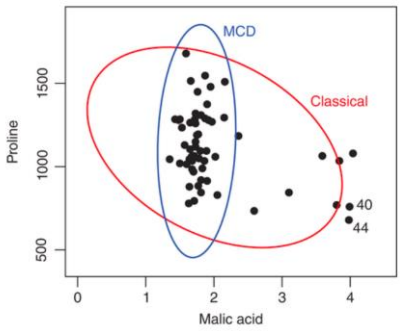
$$MAD = median_i\big(|x_i - median_j(x_j)|\big)$$

Let's consider now those outliers that can be revealed only considering multiple dimensions; they are called **multivariate** outliers. An example can be found in income and tax rate, where a couple of high income with low tax rate is an outlier, but the two values alone are not enough to determine it.

In these cases, it is necessary to measure the distance between a data point to the mean using the covariance matrix and define how much distance should quantify an outlier. One possible distance is the *Mahalanobis distance*:

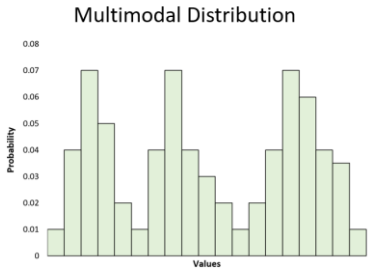
$$\sqrt{(x-\mu)^T\Sigma^{-1}(x-\mu)}$$

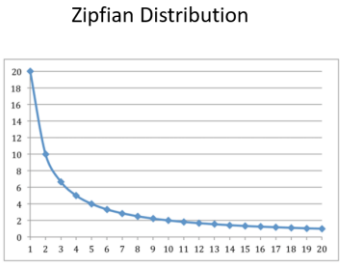
One of the famous methods for robustification of multivariate mean and covariance matrix is called the *Minimum Covariance Determinant*. The goal is to find a set of h points that minimize the determinant of the covariance matrix. These points will be used to compute the Mahalanobis distance.



The problem with parametric approaches is that not all the datasets are normally distributed.

Two other distributions are often observed: multimodal and Zipfian distributions.





Non-parametric approaches

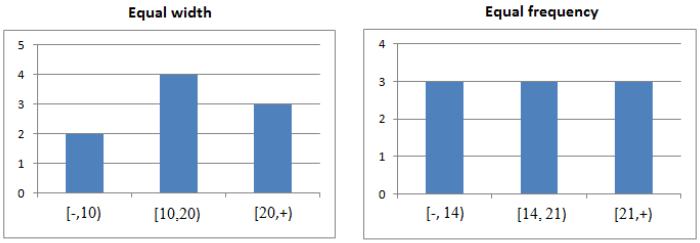
Non-parametric approaches make no assumptions about the distribution that generates the data. They infer the distribution from the data.

There are mainly two types: Histogram-based and Kernel density-based.

A histogram is a graphical representation of the distribution of numerical data values and is used to estimate the probability distribution of a continuous random variable. To create a histogram, the range of values is discretized or bin, and for each interval the number of values falling under it are counted.

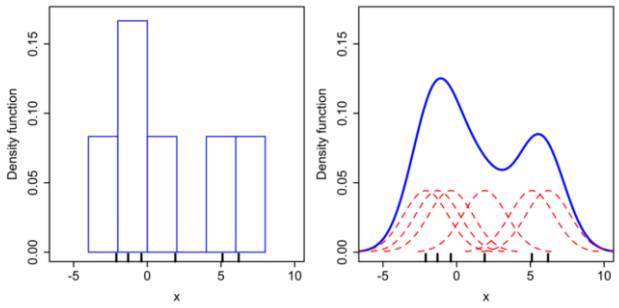
Considering the way in which bins are created, it's possible to have:

- *equi-width histograms*: if the bins have the same width. Then outliers are in bins with less values. The problem shifts to how large to take the bins.
- *equi-depth histograms*: if the bin has the same frequency.



In **histogram-based** non-parametric outlier detection approaches, equi-width histograms can be used to detect outliers: data points that belong to bins that have a very low frequency are reported as outliers. The problem is to identify the right width of the interval.

Instead, **Kernel-density estimation** consists in estimating the probability density function f(x) of a random variable starting from a data sample on the basis of properties such as smoothness or continuity. Outliers are then detected by observing properties such as the smoothness and identifying strange behaviors.



Distance-based outlier detection

Distance-based outlier detection mechanisms do not assume an underlying model and work by defining a distance between data points, which is used to define a normal behavior.

On the basis of the reference population, it's possible to distinguish between: Global distance-based outlier detection and Local distance-based outlier detection.

Global distance-based outlier detection

These methods simply compute the distance between each point with all the other points and use this information to determine if that point is an outlier. In particular, a point is an outlier if at least a fraction p of the other objects lies at a distance greater than distance D from that point.

The problem lies in defining what is the correct threshold (for the distance) to distinguish outliers from non-outliers.

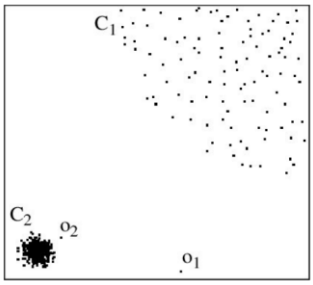
The definition of outlier used does not provide a ranking or a score of outliers and needs specifying a distance parameter D that is not always easy to determine and may involve a trial-and-error process.

The methods used include:

- nested loops to find the distance between all the points
- use of spatial index structures (e.g., trees)
- blocking to minimize the number of comparisons

Local distance-based outlier detection

Outliers in the same region could be recognized as normal values by global distance-based algorithms but be in fact outliers. In these and other similar cases, it might be difficult to understand which ones effectively are the outliers. In particular, clusters of points create problems to global distance-based outlier detection techniques.



To overcome this issue, it's possible to define a *local outlier factor* (LOF) that scores data points based on the density of their neighboring data points.

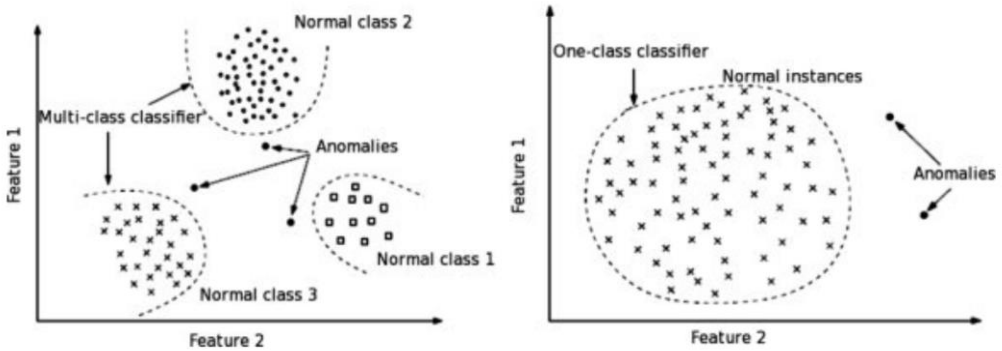
Given a positive integer k, the k-distance between an object p and another object o (distance (p, o)), it is possible to define:

- A reachability distance that is the distance between p and o if they are far, and on the other hand, if they are close, it is the average distance among close points.
- The local reachability density is the inverse of the reachability distance.
- The local outlier factor is the average of the ratio of the local reachability density of p and the points in the k-distance neighborhood of p.

Model-based outlier detection

These methods learn a classifier model from a set of labeled data points and then apply the trained classifier to a test data point to determine whether it is an outlier or not.

Validation and application of these models are generally fast but come with the disadvantage that we need a set of labeled data to start with, in order to train the classifier. Moreover, to properly train a classifier, the amount of needed (labeled) data is huge and it's not easy to find.



Imputing missing values

Under the closed world assumption, a **missing value** can be defined as the data value that is not captured nor stored for a variable in the observation of interest. There are three types of missing values:

- **Missing Completely at Random** (MCAR): MCAR occurs when the missing of the variable is completely unsystematic. When our dataset is missing values completely at random, the probability of missing data is unrelated to any other variable and unrelated to the variable with missing values itself.
- **Missing at Random** (MAR): MAR occurs when the probability of the missing data on a variable is related to some other measured variable but unrelated to the variable with missing values itself.
- **Missing Not at Random** (MNAR): MNAR occurs when the missing values on a variable are related to the variable with the missing values itself.

Missing data can limit our ability to perform important data science tasks such as converting data types or visualizing data, it can reduce the statistical power of the developed models, it can reduce the representativeness of the samples in the dataset, and it can distort the validity of the scientific trials and can lead to invalid conclusions.

To make up for these unwanted effects, it's possible to perform imputation. **Imputation** is the process of replacing missing data with substituted values.

Imputation for some applications is necessary since missing values can introduce bias, make the analysis difficult and decrease efficiency.

An alternative method to imputation is the deletion of all the tuples that contain a missing value. It easy to implement but the decreases the effectiveness of the analysis.

Among imputation methods we can list:

- *Deterministic imputation*: It imputes a missing value by using logical relations between variables and derive a value for the missing item.
- *Mean/median imputation*: the missing values are replaced by the mean/median of the observed values.
- *Regression imputation*: this method replaces the missing values by predicted values from a regression of the missing item on items observed for the unit.
- *Hot deck imputation* (K-NN Imputation): is a simple way is to impute for each missing item the response of a randomly selected similar record.
- *ML- Based imputation*: based on ML methods (e.g., random forest)

10. Data deduplication

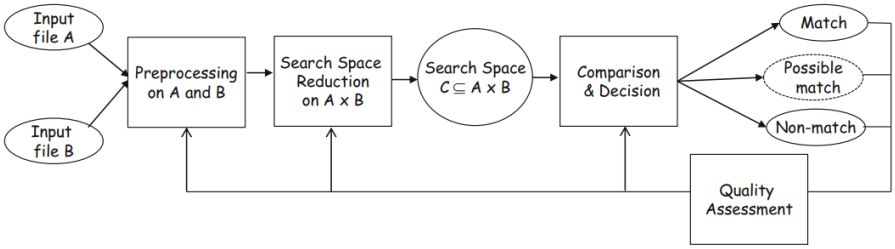
Duplicate detection is the discovery of multiple representations of the same real-world object. To perform it and to remove redundancies is the last step of data cleaning.

Duplicate detection techniques can be also called *record linkage*, *object identification* or object consolidation, *entity resolution* or entity clustering, merge, or purge.

Duplicates can be found inside the same table (having thus the same schema) or may be tuples in different tables representing the same entity (in this case the schema is different but there is an evident overlapping of some if not all attributes).

Issues in duplicate detection arise when representations are not identical and so similarity measures, with their accuracy problems, are needed to find duplicates, or when data sets are large and so efficient algorithms are needed to reduce complexity and costs.

The high-level process of duplicate detection can be summarized with the following schema:



Distance-Based Comparison Functions can be divided into two families:

- String-Based Distance Functions (e.g., edit distance, sounded code, Jaro algorithm), suitable to compare two values (two strings)
- Item-Based Distance Functions (e.g., Jaccard distance, TF-IDF), good to compare sentences.

It's important to keep in mind that, more than often, more different distance functions are applied together to get better and more accurate and comprehensive results at the end.

String-based distance functions

The **Edit-distance** / **Levenshtein-distance** counts the minimum number of edits from one word to the other to determine the distance from the two. Edit operations to count are substitutions, deletion, insertion.

E.g., Between "intention" and "Execution" there is an edit distance of 5.

So, the similarity score is computed as:

$$1 - \frac{edit\ distance(m,n)}{max(|m|,|n|)}$$

In **Longest common subsequence** (LCS), distance substitution costs 2 since this variation allows only deletion and insertion.

Another alternative is **Hamming distance**, which allows only substitutions (the two words must be of the same length).

The **Jaro-Winkler** string comparator counts the number c of common characters between two strings and the number of transpositions that are the number of pairs of common characters that are out of order. So, given the number of common characters c, the length m of the first word, the length n of the second word, and the number t of transpositions to perform in order to pass from one word to the other, the similarity score can be computed as:

$$S = \frac{1}{3}\left(\frac{c}{m} + \frac{c}{n} + \frac{c-t}{c}\right)$$

E.g., comparing the two surnames “Barnes” and “Anderson”, we find that common characters are “arnes” while one transposition “rne”→“ner” is needed. So we have a score: S = 1/3*(5/6+5/8+(5-1)/5) = 0.75

These distances can be ineffective in some cases, for example when dealing with people’s names, that are translated from sounds using phonetical rules ending with representations with the same sound but different written words. **Soundex** was defined for problems with names, that can be spelt in different ways. It consists of phonetically oriented algorithms that can find similar sounding terms and names. So, similarity between words is based on their sound.

Soundex algorithm is the following:

1. The first character of the word is retained as the first character of the Soundex code.
1. The following letters are discarded: a, e, i, o, u, h, w, and y.
2. Remaining consonants are given a code number.
3. If consonants having the same code number appear consecutively, the number will only be coded once. (e.g. "B233" becomes "B23").
4. The resulting code is modified so that it becomes exactly four characters long: If it is less than 4 characters, zeroes are added to the end (e.g. "B2" becomes "B200"), if instead it is more than 4 characters, the code is truncated (e.g. "B2435" becomes "B243").

The English Soundex table for substitutions is provided below, with some examples of Soundex at work:

b, p, f, and v	1
c, s, k, g, j, q, x, z	2
d, t	3
l	4
m,n	5
r	6

AC/DC	Ay See Dee Ci
A232	A232

Our	Hour
0600	H600

Robert	Rupert
R161	R161

Bejing	Pecking
B252	B252

Philadelphia	Filadelfia
P434	F434

Using different metrics leads to different distances and different results with relation to what other words are more similar to the one selected as starting point. The following example shows different metrics when evaluating the similarity scores of surnames similar to “Mire”:

SURNAME	EDIT_DIST	JARO_WINK	SOUNDEX	SOUNDEX(SURNAME)
Mayer	40	67	similar to Mire	M600
Meier	40	80	similar to Mire	M600
Meiyar	34	67	similar to Mire	M600
Meiyer	50	67	similar to Mire	M600
Miere	80	87	similar to Mire	M600
Miere	80	87	similar to Mire	M600
Miero	60	87	similar to Mire	M600
Miire	80	94	similar to Mire	M600
Mire	100	100	similar to Mire	M600
Mirea	80	96	similar to Mire	M600
Mireh	80	96	similar to Mire	M600
Mirew	80	96	similar to Mire	M600
Mirhe	80	95	similar to Mire	M600
Mirre	80	95	similar to Mire	M600
Mmire	80	94	similar to Mire	M600
More	75	85	similar to Mire	M600
Myre	75	85	similar to Mire	M600
Admiral	29	59	not similar to Mire	A356
Admire	50	75	not similar to Mire	A356
Amire	60	78	not similar to Mire	A560
Mayes	40	67	not similar to Mire	M200
Mayo	25	55	not similar to Mire	M000
Meiers	50	82	not similar to Mire	M620
Mierins	43	81	not similar to Mire	M652
Miers	60	87	not similar to Mire	M620
Mimre	80	94	not similar to Mire	M560
Miranda	43	80	not similar to Mire	M653
Mirfe	80	95	not similar to Mire	M610
Mirissimo	34	78	not similar to Mire	M625
Smire	60	78	not similar to Mire	S560
cat	0	0	not similar to Mire	C300

Item-Based Distance Functions

With string-based distance functions, the distance between two values is measured on a syntactical level. Therefore, strings that are syntactically similar but have a completely different meaning, would be considered very similar, or strings that are very different syntactically but have the same meaning would be considered different. Using for example the edit distance, we would have:

$$editDistance(\text{AT\&T};\ \text{AT\&T Corporation}) = 12 \implies Similarity = 1 - 12/16 = 0{,}25$$

$$editDistance(\text{IBM Corporation};\ \text{AT\&T Corporation}) = 5 \implies \text{Similarity} = 1 - 5/16 = 0{,}6875$$

This case demonstrates how string-based distance functions may lead to erroneous conclusions. Therefore, a way to measure the similarity also on a semantical level is needed.

Item-based distance functions take into account the semantical meaning of strings. They start from the assumption that two strings can also be considered as bags (or multisets) of tokens (or words).

The **Jaccard similarity index** is defined as the cardinality of the intersection of the two sets of words divided by the cardinality of the union of them. So, considering two sentences $S1$ and $S2$ as sets of words, let A and B the set of words in $S1$ and $S2$, then:

$$Jaccard(S1, S2) = \frac{|A \cap B|}{|A \cup B|}$$

For example, applying the Jaccard distance to the initial example, there's an improvement:

$$JaccardIndex(\text{AT\&T};\ \text{AT\&T Corporation}) = {}^{1}/_{2} = 0{,}5$$

$$JaccardIndex(\text{IBM Corporation};\ \text{AT\&T Corporation}) = {}^{1}/_{3}$$

The problem with Jaccard distance is that it considers all words equivalently, without considering some of them more important than others, while in reality they should have different weights when deciding whether two sentences are similar or not. Jaccard distance gives the same importance to common words as to others, and so texts with different meanings could result similar if they just have a lot of common words.

To solve this problem and give lower weights to words that are common to more documents compared, it's possible to make use of the **TF-IDF- term frequency inverse document frequency** (or cosine similarity), often employed when comparing sets of documents or long and complex texts.

The idea is to assign higher weights to words appearing frequently in a document (TF weight) and to assign lower weight to words that appear frequently in the whole set of documents (IDF weight).

$TF_{w,s}$ is the number of times word w appears divided by the total number of words in the considered document.

IDF_w measures the importance of a word. $IDF(t) = \ln\left(\frac{\text{Total number of documents}}{\text{Number of documents with term } t \text{ in it}}\right)$

$V'_{w,s}$: $TF_{w,s} \times IDF_w$

Each weight should be scaled: $V_{w,s} = \frac{V'_{w,s}}{\sqrt{\sum_w V'_{w,s}}}$

We obtain vectors of weights (e.g., U and V)

We then calculate the similarity using cosine similarity with these vectors.

Domain-dependent similarity metrics

In some cases, the context has relevance and it's possible to employ **domain-dependent similarity metrics**.

There are domain-dependent similarity measures specialized for data types, for example special similarity metrics for dates, to compare them specifically, or special similarity metrics for numbers.

Others are instead built for a specified domain and are specific for it, and are based on rules that can be specified by the user and that can make use of other distances. For example, a rule could establish that two tuples are equivalent if the edit distance for some attributes is lower than a certain threshold and some attributes are the same. In order to define these rules, domain knowledge is needed along with the ability to define the right thresholds.

Search space reduction

Two are the main issues when coming to duplicate detection: finding the right distance metrics to use and reducing the size of the search space to lower the complexity of the problem.

The three classical methods to perform search space reduction are:

- **Blocking**: the file is partitioned in exclusive blocks and comparisons are limited to records within the same block. Blocking can be implemented by choosing a blocking key and grouping into a block all records that have the same values on the blocking key.
  The smaller the blocks, the less complexity when doing comparisons and the more efficient duplicate detection is, but accuracy decreases with the decrease of the size of blocks and their number raising. There is a tradeoff between the size of blocks that needs to be smaller to achieve efficiency and the overall accuracy that needs bigger blocks to be high.
- **Sorted neighborhood** consists of sorting a file and then moving a window of a fixed size on the file, comparing only records within the window.
- **Pruning** (or filtering) has the objective of first removing from the search space all records that cannot match each other, without actually comparing them.

Nowadays several other methods are available, but not necessarily they overtake these classical ones. In any case, all the approaches available in literature are divided into three sets:

- **Empirical techniques** that make use in the different phases of the process of algorithmic techniques such as sorting, tree traversal, neighbor comparison, and pruning.
- **Probabilistic techniques**, based on the extremely relevant set of methods developed in the last two centuries in statistics and probability theory.
- **Knowledge-based techniques**, in which domain knowledge is extracted from the files involved and reasoning strategies are applied.

All these methods are part of fuzzy matching, used when univocal identifiers (e.g., the person code) are not available to directly understand if there are representations of the same entity, or this identifier is present but cannot be trusted.

Name	Technical Area	Type of data
Fellegi and Sunter and extensions	probabilistic	Two files
Cost-based	probabilistic	Two files
Sorted Neighborhood and variants	empirical	Two files
Delphi	empirical	Two relational hierarchies
DogmatiX	empirical	Two XML documents
Intelliclean	knowledge-based	Two files
Atlas	knowledge-based	Two files

Empirical techniques

Among the empirical techniques, the **Sorted Neighborhood** is the most famous and is usually utilized as anchor for comparison with other techniques.

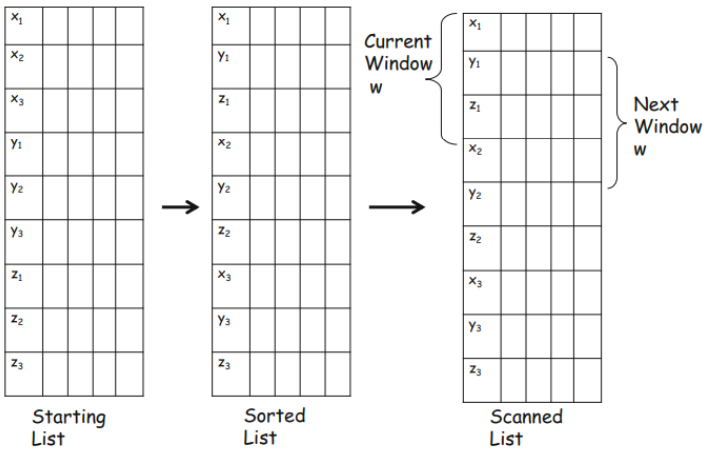
It is based on three phases:

1. Create a Key for each record: Compute a key for each record by extracting relevant fields or portions of fields. rationale is that similar data will have closely matching keys.

First	Last	Address	ID	Key
Sal	Stolfo	123 First Street	45678987	STLSAL123FRST456

2. Sort records on this key: Sort the records in the data list using the key in step 1.
3. Merge/Purge records: Move a fixed size window through the sequential list of records. This limits the comparisons to the records in the window. Data are compared within a rule and a similarity function.

The use of a sliding window decreases the search space and, as the dataset was sorted by the key and so the similar tuples will be near, most likely inside the window, the probability of having duplicates inside the window is at a good level.



There are three issues with this method:

- Defining the optimal size for the window is not trivial, as there is a tradeoff between maximizing accuracy and minimizing the computational cost. The usual method is to go by trial and error and iteratively adjust the size of the window until the results are acceptable.
- Selecting and deciding how to generate the key. The key has to provide sufficient discriminating power and must be defined using relevant attributes for this task with the right importance.
- Even after sorting by the key, there is still possibility that similar records remain too distant and are not compared because never inside the window at the same time, and they might have same meaning but different key.

A partial solution is the **multi-pass approach** of the Sorted Neighborhood method. After the first execution, the key generation mechanism is changed, and other passes of the window are performed. This way, the accuracy should be higher.

The multi-pass SNM is based on the consideration that running the SNM on a single sorting key does not produce the most suitable results.

Therefore, the idea is to have several runs of the method, each with a different key and very small windows. Different keys reasonably ensure that if there are errors on some of them, the subsequent runs will compensate such errors. Also, running SNM with small windows corresponds to run less expensive steps instead of a single expensive one, keeping the complexity at an acceptable level.

Each run of the multi-pass approach produces a set of pairs of records that can be merged. A transitive closure step is then applied to such pairs of records, and the result is the union of all pairs found in the independent runs, with the addition of pairs that can be inferred by transitive closure.

The experimental evidence is that the multi-pass approach drastically improves the accuracy of the basic SNM with a single run on large varying windows.

Probabilistic techniques

The **Fellegi and Sunter** theory consists in a supervised technique that needs a training dataset for labelling records pairs as matching or non-matching. Then, the similarity scores serve as features for training a classifier to be applied to the rest of data.

Given couples of records taken from different sets, we define two disjoint sets: M is the set of couples of records that are equal, U is the set of couples of records that are not equal, where with "equal" we mean that two records represent the same real-world entity. M is named the *matched* set and U is named the *unmatched* set. A third set P can also be introduced as the *possible matches*.

A function γ returns, for each couple of records, a certain level of agreement between the two. So, for each couple of records, we have a probability that they represent the same entity, and so the probability that they belong to M or U.

The way this γ function works is by giving a score based on some rules (e.g., if the attribute X is the same then +4, if the attribute Y is different than -10, etc.), and then to transform this score into a probability.

Afterwards, we are interested in:

- The probability $m(\gamma)$ that a couple of records that has an agreement level higher than a certain threshold is actually composed by duplicates, so the function was right.
- The probability $u(\gamma)$ that a couple of records that has an agreement level higher than a certain threshold is not composed by duplicates, so the function was wrong.

These last two probabilities are the base for future calculations that are performed.

They can also be used to assess the data quality of some dataset. In fact, having high probability of matching when distances are high means that there are a lot of errors in the database.

In the case in which we are able to estimate such probabilities, they become crucial in a possible assignment decision procedure.

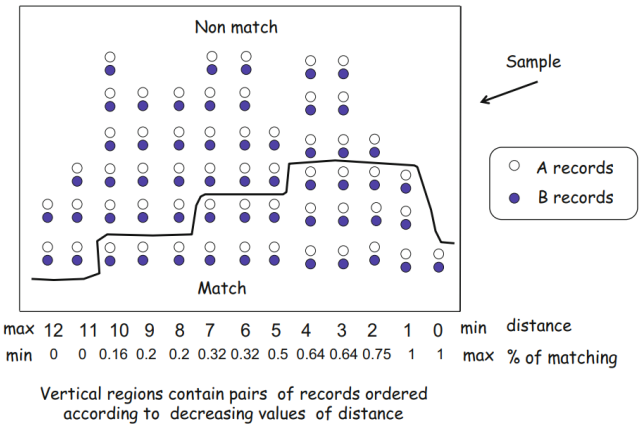
Given the ratio: $R = {m(\gamma)}/{u(\gamma)}$

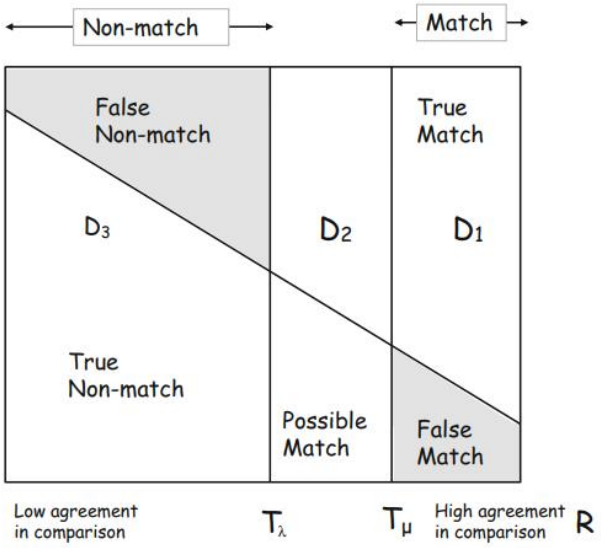
Fellegi and Sunter defined the following decision rule, where T_μ and T_λ are two thresholds:

If $R > T_\mu$ then designate the pair as a match.

If $T_\lambda < R < T_\mu$ then designate the pair as a possible match.

If $R < T_\lambda$ then designate the pair as a non-match.





In any case, the Fellegi and Sunter theory is based on the knowledge of the u and m-probabilities. Several methods have been proposed to compute or estimate such probabilities.

Knowledge-based techniques

Knowledge-based techniques are techniques that make use of knowledge about the scenario to update the decision rules.

Choice maker is based on rules, called clues, that are domain-independent or domain-dependent relevant properties of data. Clues are used in two phases: first offline when a training set is employed to determine the importance of each rule, and then at run-time when the trained model is applied to the clues to compute a match probability that is compared with a given threshold.

Several types of clues can be defined in Choice Maker, such as:

- Swaps of groups of fields, e.g., swaps of first and last names.
- Multi-clues, i.e., groups of clues that differ only by a parameter.
- Stacked data describe data that store multiple values for certain fields. For example, current and old addresses may be stored in a relation to retrieve the old if needed.
- Complex clues that capture a wider set of properties of the application domain.7

	Record #	First Name	Last Name	State	Area	Age	Salary
→	1	Ann	Albright	Arizona	SW	65	70.000
	2	Ann	Allbrit	Florida	SE	25	15.000
	3	Ann	Alson	Louisiana	SE	72	70.000
→	4	Annie	Olbrght	Washington	NW	65	70.000
→	5	Georg	Allison	Vermont	NE	71	66.000
	6	Annie	Albight	Vermont	NE	25	15.000
	7	Annie	Allson	Florida	SE	72	70.000
→	8	George	Alson	Florida	SE	71	66.000

Another technique is that of **Intelliclean**, whose main idea is to exploit rules as an evolution of previously proposed distance functions.

Rules are extracted from domain knowledge, and they are of two types, with different goals:

- Duplicate identification rules, specifying conditions according to which two tuples can be classified as duplicates. They allow more complex logic expressions for determining tuple equivalence and they work by abstraction of the similarity between two tuples.

```
Define rule Restaurant_Rule
Input tuples: R1, R2
IF  (R1.telephone = R2.telephone)
AND  (ANY_SUBSTRING (R1.ID, R2.ID) = TRUE)
AND (FIELDSIMILARITY (R1.address =  R2.address) > 0.8)
THEN
DUPLICATES (R1,R2) CERTAINTY = 0.8
```

- Merge-purge rules, specifying how duplicate records are to be handled.

In Intelliclean a *certainty factor* (CF) is applied to each duplicate identification rule. During the computation of the transitive closure, we compare the resulting certainty factor of the merged group to a user-defined threshold, which represents how tight or confident we want the merges to be. Any merges that result in a certainty factor less than the threshold will not be executed.

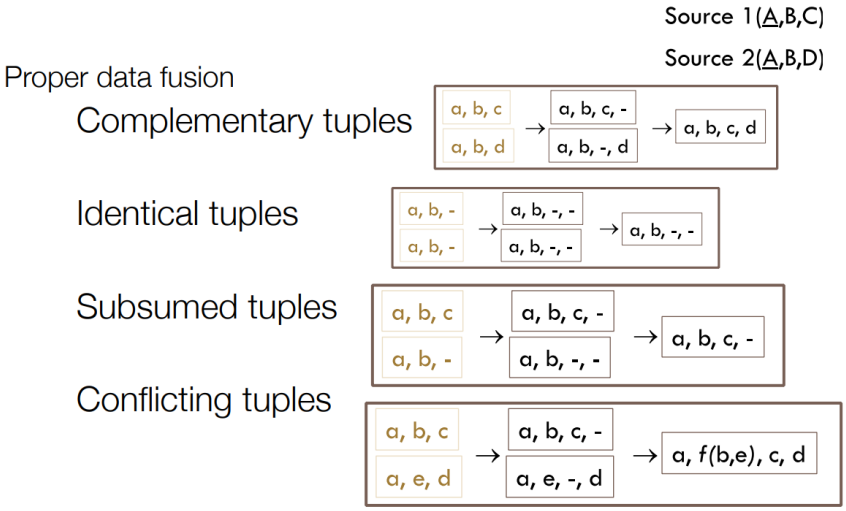
Data fusion

Once it's understood that two tuples are duplicates, only one record representing that real-world entity must be kept. **Data fusion** has the aim to resolve uncertainties and contradictions, and works by, given a duplicate, creating a single object representation while resolving conflicting data values.

This process presents many difficulties:

- Null values: Subsumption and complementation.
- Contradictions in data values.
- Uncertainty & truth: Discover the true value and model uncertainty in this process.
- Metadata: Preferences, recency, correctness.
- Lineage: Keep original values and their origin.

Data fusion goals can be summarized in the following schema:



There are different strategies to handle conflicts:

- **Conflict ignorance**: it's possible to decide to deal only with certain tuples and some of the conflicts between multiple records, passing them to the users or to the application, while the majority of the conflicts are intentionally ignored, assuming that they are negligible or too expensive to address.
- **Conflict avoidance**: involves taking measures to prevent conflicts from occurring during the data fusion process, thanks to careful planning and data preprocessing to minimize the chances of conflicts. It may involve standardizing data formats, cleaning and validating data before integration. It can be:
    - Instance based,
    - Metadata based.
- **Conflict resolution**: actively addressing and resolving conflicts that arise during the integration of data from multiple sources. It can be:
    - Instance based - deciding, mediating,
    - Metadata based - deciding, mediating.

When **conflict ignorance** is the chosen strategy, we ignore the conflicts between multiple records that refer to the same entity and pass the conflicts to the users or to applications. The problem is not solved but we are asking someone else to decide whether to solve it and how to do it.

There are two possible options for this strategy: escalate or to consider all possible resolution strategies.

Another strategy may be the "possible worlds (or models)", consisting in building all possible solutions and annotate them with a likelihood. This can be done in probabilistic databases, in which for each tuple a probability for its correctness is given. The issues in this approach are in extending the algebra to produce probabilities and in extending the query language to query and export them. Moreover, by keeping all the possible situations (when integrating more sources, more combinations involving the same tuple appear with different probability) the databases acquire higher complexity and size.

If the volume of data is huge, then deleting conflicting tuples without consequences may be possible, and a viable option. Of course, attention must be paid to what is effectively deleted, because it may constitute important information.

Conflict avoidance applies a simple rule to take a unique decision based on either the data instance (e.g. prefer not null values) or the metadata (e.g. prefer values from tables with higher reputation).

Conflict resolution consists in deciding the true value between the available ones by considering the ones that I have. These strategies solve the conflicts by picking the value from already present values (deciding), for example by taking the most frequent value, or by choosing a value that does not necessarily exist among present values (mediating), for example doing the average of all values. Metadata can of course be used to make a decision.

Strategy	Classification	Description
Pass it	Conflict ignorance	Escalates conflicts to users or applications
Consider all the possibilities	Conflict ignorance	Creates all the possible value combinations
Take the information	Conflict avoidance – instance based	Prefers values over null values
No Gossiping	Conflict avoidance – instance based	Returns only consistent tuples
Trust your friends	Conflict avoidance – metadata based	Takes the value of a preferred source
Cry with the wolves	Conflict resolution – instance based - deciding	Takes the most often occuring value (Vote)
Roll the dice	Conflict resolution – instance based - deciding	Takes a random value
Meet in the middle	Conflict resolution – instance based - mediating	Takes an average value
Keep up to date	Conflict resolution – instance based - mediating	Takes the most recent value

Function	Description	Example
Min, max, sum, count, avg…	Standard aggregation	Salary, height
Random	Random choice	Shoe size
First, last, longest, shortest	Adopt a strategy	First name
Choose source	Give a higher reputation to a source	
ChooseDepending (val, col)	Value depends on value chosen in other column	City&zip, e-mail & employer
Vote	Majority decision	rating
Coalesce	Choose first non-null value	First name
Group, concat	Group or concatenate all values	Reviews
Most recent	Most recent value	address
Most abstract or specific term	Use a taxonomy	Location
….		

The process of solving conflicts may introduce accuracy problems, decreasing the overall accuracy.

It may be the case that a lot of the data has high uncertainty, and no ground truth is available. In this case, advanced techniques to detect uncertainty and estimate if a given fact in the data is true or not are needed. The sources reputation gains this way importance, and data provenance problems arise.

Advanced techniques for conflict resolution consider and analyze the quality of the sources in terms of:

- Source Accuracy: probability that a value provided by the source is a true value.
- Source Dependency: considers the fact that sources are often not independent - a source may be a copy of another source or may contain values taken from other sources. Then, sources that do not take data from elsewhere have higher reputation.
- Source freshness: how quick a value change is captured by a source. The idea is that more updated sources are the one more taken care of, so the level of update of the tables gains importance.

Sometimes it is necessary to involve humans in the fusion phase to improve the accuracy. In this case we are talking about *Human-involved Data Deduplication*. Some systems may also rely on a crowd, usually paid, to complete some tasks, performing the process called **crowdsourcing**.

"Answer" is the result of a query to an integrated information system, that is:

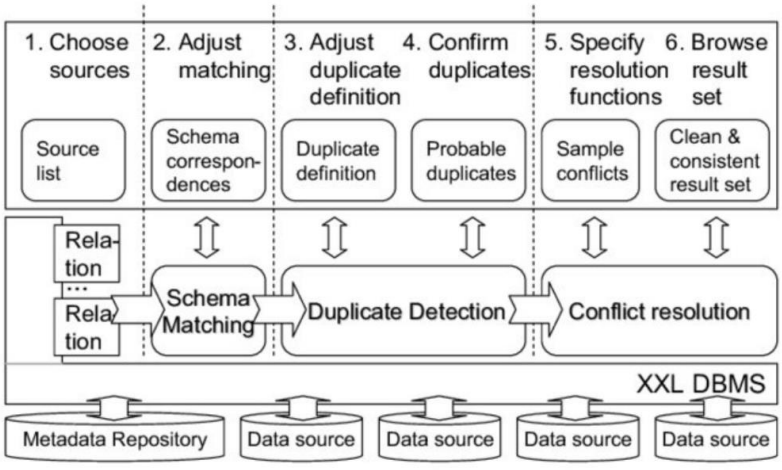
- *Complete*: the answer should contain all the objects and attributes that have been present in the sources
- *Concise*: all the objects and attributes are described only once
- *Consistent*: all the tuples that are consistent with respect to a specified set of integrity constraints are present
- *Complete and consistent*: it additionally fulfills a key constraint on some real world ID (contains all attributes from the sources and combines semantically equivalent ones into only one attribute).

When one single tuple is produced as the result of the resolution of a conflict, we have a so-called **data fusion answer**.

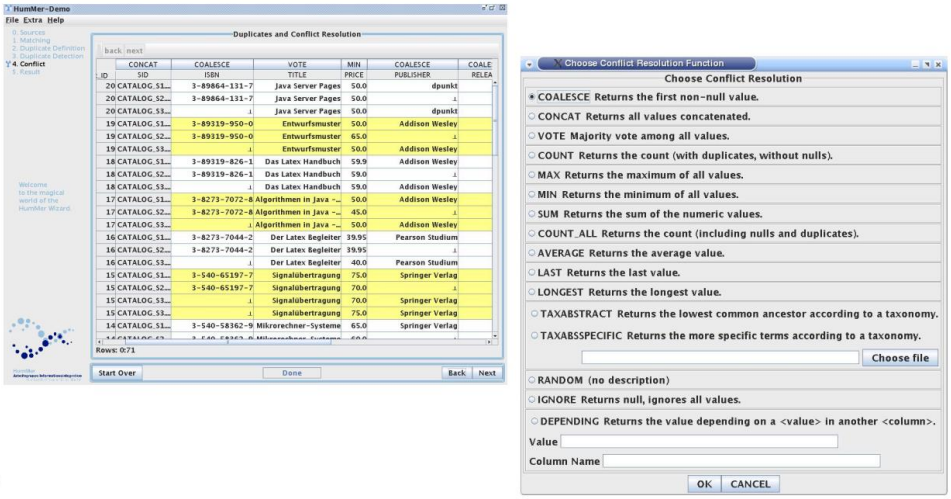
HumMer tool

HumMer is a tool devoted to data fusion.

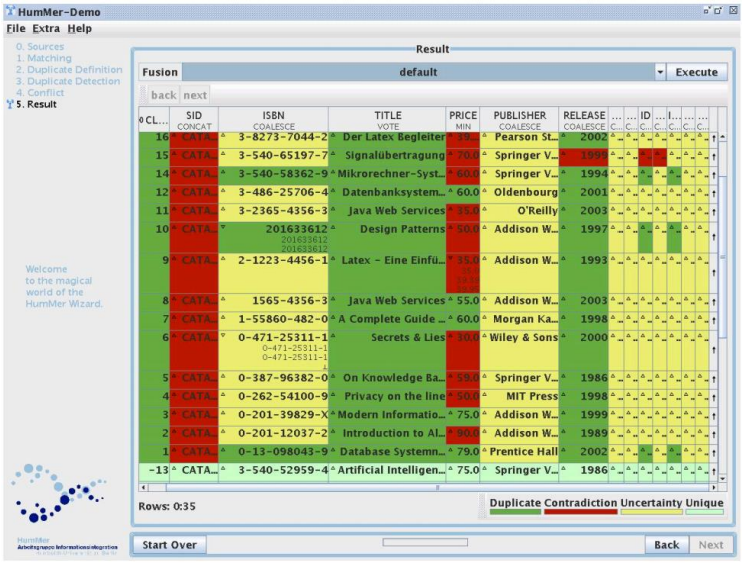
It supports the user in the definition of an integrated table, when doing duplicate detection between different databases.



Similarities between attributes are discovered and a list of potential and sure duplicates is produced. Then, the conflict resolution functions that are available and suggested are shown along with hints on how to deal with conflicts based on the kind of attribute.



At the end, a final integrated schema is produced, containing only one instance of the found duplicated values.



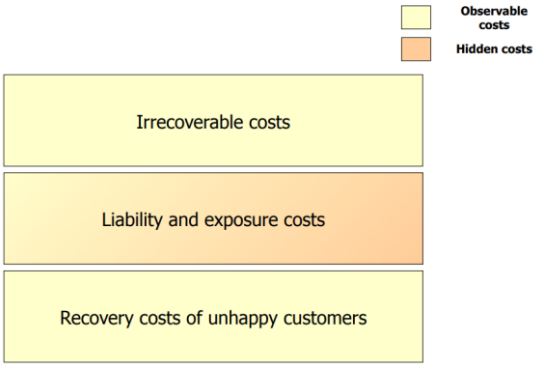
11. Data streams

Data Quality costs

Poor quality of data can have a high cost. The costs associated with poor data quality include:

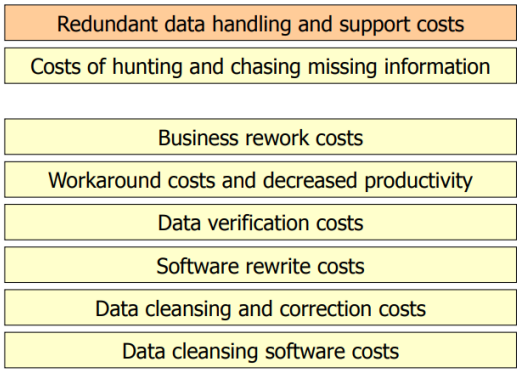
- Process failure costs,
- Information scrap and rework costs,
- Lost and missed opportunity costs.

Some costs are direct, and we can often translate them directly in money terms, while others are not and have a latent effect.



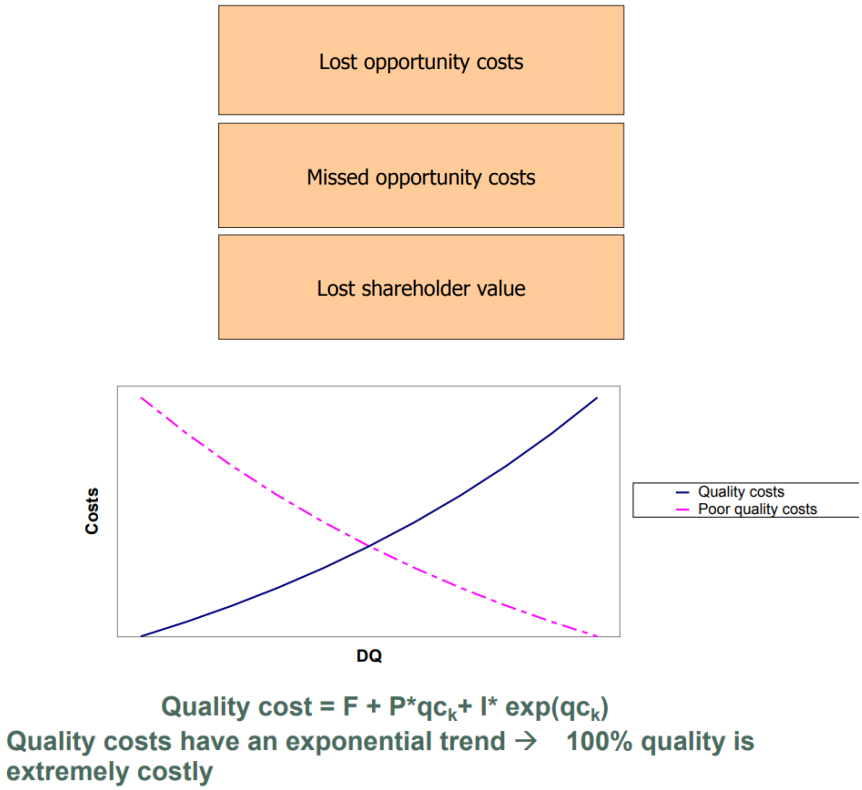
Process failure costs are expenses incurred as a result of defects, errors, or failures in a process or product. Ensuring high-quality data is crucial for organizations to avoid the negative impacts of data-related failures.

A failure in a process leads to the need of re-doing something and repeating the process while repairing the error. Of course, this has a cost, that is the **Information scrap and rework costs**. *Information scrap* refers to the discarding or rejection of data or information that is deemed unusable, inaccurate, or irrelevant. If a dataset contains a significant number of inaccurate records, organizations may need to discard or ignore that portion of the data, essentially treating it as "information scrap." *Rework costs* in the context of data management refer to the additional efforts and resources needed to correct, modify, or improve data that was initially processed or managed incorrectly. If errors are identified in a dataset after it has been processed and used for analysis, rework costs may include the time and resources required to go back, correct the errors, and ensure the data is accurate.



Indirect costs are constituted mainly by **Lost and missed opportunity costs**, that describe the potential value that is foregone when a particular choice or decision is made. Poor data quality can

have a negative impact on informed decision making, strategic planning, and the potential to capitalize on opportunities. By improving data quality, organizations can enhance their ability to identify and seize opportunities, ultimately reducing the risk of lost or missed opportunity costs.



Quality cost = F + P*qc_k+ I* exp(qc_k)
Quality costs have an exponential trend → 100% quality is extremely costly

Data Stream

A **data stream** is an infinite sequence of elements:

$$\{(x_1, T_1), (x_2, T_2), \dots, (x_m, T_m), \dots.\}$$

Each element is composed by a value and a timestamp.

There is no static table to analyze, and it is not possible to reason only in terms of input to pass through a static sequence of phases to produce an output. There is a flow of data, and other things such as the *velocity* need to be considered.

The general goal is to compute a function of a stream, like for example an average, usually applying a lambda architecture.

The real-world situation usually involves monitoring data from one or more sensors.

Challenges are in accessing data sequentially and in processing each element quickly.

The behavior of a data stream can be continuous or periodical, meaning that data arrives continuously, or the sensors collect the data for a while and then send it all together. In any case, data analysis must keep up with a data rate to prevent from the loss of important information.

Moreover, streaming data evolves over time, and consequently the value of data decreases over time because the recent streaming data is more valuable than the old one in most applications. For some applications, the recent data alone is even sufficient.

However, streaming data are more than often noisy and corrupted. There is an issue of data quality.

In fact, the quality of sensor data is restricted due to limited sensor precision and sensor failures, and, additionally, data stream processing introduces additional noise and decreases data quality in order to meet resource constraints in streaming environments.

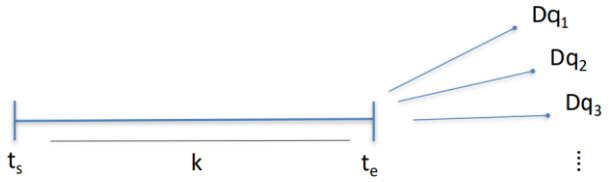
Quality characteristics need to be captured, processed, and provided to the respective business tasks.

Since data streams are noisy, a way to monitor the quality of the incoming data is needed. To handle data deficiencies, there are two ways:

- **Optimistic approach**: we rely on sensors with high precision and assume that the arising errors are so small to be negligible in the considered context. In few words, we are trusting the sensors.
- **Data quality-aware strategy**: data quality information must be recorded at the sensor nodes, propagated through the data processing, and finally presented to the user, after the received values are analyzed. Of course, this process introduces an overhead for data transfer and management.

A data quality-aware strategy needs new dimensions and rules for aggregating values.

The model defined for data quality management relies on the concept of **data quality windows**. The employed strategy consists in waiting for a window of values to arrive, collecting data for a specific timeframe, and then performing data quality evaluation on the values inside this window.



This way, the data stream is transformed in a bunch of tables of intervals of incoming values, and analysis is performed on these tables.

It's important to update the previous results from past windows with the findings from the new posterior tables. This way, tables are not analyzed independently.

The model can be used for a variable number of *data quality dimension*s that are adaptable to various user requirements. Five of them are:

- Accuracy: maximal systematic numeric error of a sensor measurement
- Confidence: maximal statistical error
- Completeness: missing values in a dataset
- Data volume: amount of underlying raw data
- Timeliness: temporal context of a data stream

A dimension like Consistency is not considered, because it's rare to have dependencies between the values in a stream from sensors. However, if there are multiple sensors in the same environment, there might be some correlation between the captured values.

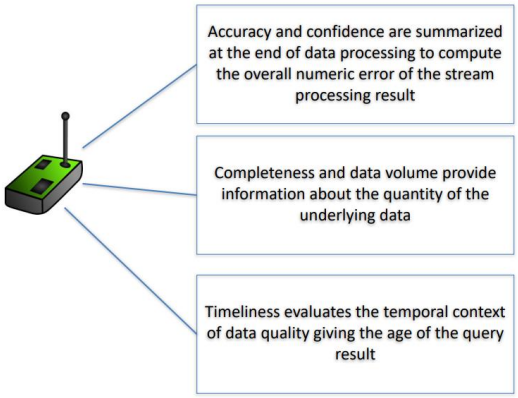
Accuracy is related to the precision of the sensors and to measurement errors. It defines the maximal absolute systematic error a such that the real value v' lies in the interval $[v - a;\ v + a]$ around the measured value v. In summary, we expect an error from a sensor that has a certain accuracy, and we admit a small, controlled tolerance interval.

Confidence is related to uncertainty that can be present in the environment. It defines the statistical measurement error due to random environmental interferences. It is the statistic error ε defining the interval $[v - \varepsilon;\ v + \varepsilon]$ around the measurement value v containing the true value v' with the confidence probability p. Confidence plays a fundamental role during data stream processing where selection or sampling aim to reduce the data volume.

Completeness is defined as the ratio of measured values compared with the size of the analyzed stream partition. If we know the frequency of a sensor, we can compare the number of received values with the number of expected one, to check how many of them we have effectively received. If we cannot predict the correct number of values we expect to arrive from a sensor, we can study the time series to find a pattern of arriving values and see what to expect next.

Data volume is the amount of raw data used to compute the results of a data stream subquery and thus to derive a data item.

Thanks to the timestamp in each arriving value, we can define the **timeliness** as the age of a data item as the difference between the current system and the timestamp of data recording. Timeliness in data streams often defines the relevance of the values.



Different **operators** can be applied to modify a data stream, among the we have:

Operator Origin	Operator Type	Example	Data Manipulation				
			Modifying	Generating	Reducing		Merging
					Attribute	Item	
CQL	Projection				×		
	Selection					×	
	Join	Equi-Join	×				
	Aggregation	Slope Calculation					×
Signal Analysis	Sampling	Simple Random Sampling				×	
	Interpolation	Linear Interpolation		×			
	Spectral Analysis	Fourier Transformation					×

- **Data Generation operators**: new data items are inserted into the data stream based on existing sensor data (e.g., linear interpolation). It increases the completeness dimension and also influences the accuracy, confidence and data volume dimensions.
- **Data Reducing operators**: they decrease the volume of the data stream to meet resource constraints such as limited communication capability, restricted memory capacity and processing power. They can be divided in:
    - Selection: During selection tuples are extracted on the basis of the selection criteria, or also on the basis of quality thresholds. Selection only impacts the confidence of data.
    - Sampling: is a general technique for dealing with high amounts of data by selecting a subset of them in a proper way for the use-case.
- **Data merging and aggregation**: the aggregation is commonly executed in combination with data grouping, by compressing the incoming data to one output value. In this case, data quality dimensions are calculated as the average of the dimensions of the incoming tuples.
- **Data Modifying operators**: such operators have no effect on the data volume. Completeness does not change since data are neither deleted nor generated.

An important issue is the definition of the window size to use.

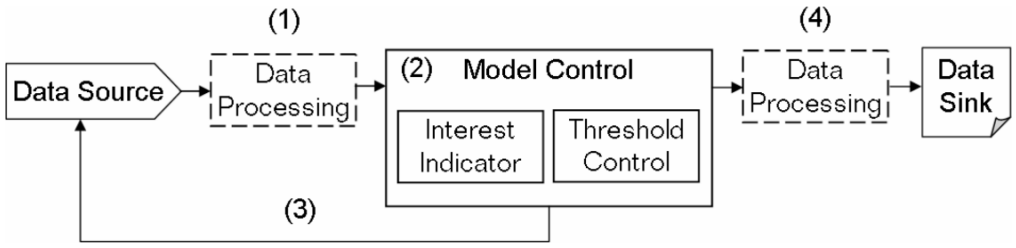
In fact, small windows result in high granular data quality information, leading to having a higher data overhead.

On the other hand, a wider window guarantees resource savings that are essential for data stream environment, but data quality information is available only at the end of the application.

A solution to this problem is an approach based on the interestingness of data stream characteristics. **Interestingness** is not a data quality dimension but characterizes the data stream in the context of a specific application scenario. It depends on the characteristics of data, not on their quality.

It's especially useful when data streams are used to monitor events or phenomena, for example if we are looking for changes in the data stream, not all values are interesting but only the ones that go outside the normal trend.

Going back to its use for window-size adaptation, the data quality model control is based on the idea that more regular is the data, wider the window can be. If there are changes in values, so data with high interestingness arrives, the window needs to be smaller in order to have checkpoints and be able to process anomalies in a fast way.



- (1) The procedure starts with a fixed window size defined by the model user.
- (2) the control operator indicates the interestingness of the current data stream and through the threshold control evaluates the current interest indicator. If the threshold is exceeded, the window size must be decreased, and data source should be informed (3).
- (4) The DQ window size is adapted, so that the data stream provides data quality information with finer granularity.
- In order to support different levels of interestingness more than one threshold can be applied, resulting in a set of interest classes.

To measure interestingness, interest indicators and operators are used:

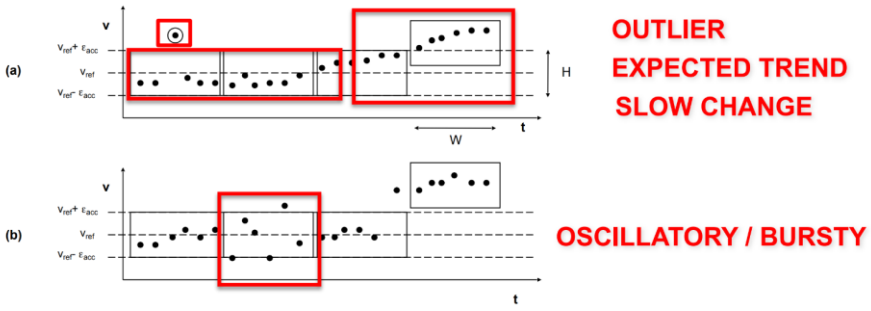
	Interest Indicator	DQ Control Operator Pattern
currentValue()	Extraordinary value ranges	→ Threshold Control →
slidingSlope()	Extraordinary value alterations	→ Slope Calculation → Threshold Control →
fft()	Unsteadiness	→ FFT → Threshold Control →
fftSlope()	Changing periodicity	→ FFT → Slope Calculation → Threshold Control →

- *currentValue()*: use a threshold operator, which in this case also takes over the role of the interest indicator, to detect extraordinary measurement values that are over a defined threshold.
- *slidingSlope()*: extraordinary value alterations like the fast rising of a measurement value can be perceived with the help of a sliding slope aggregation over the respective data quality.
- *fft()*: use the unsteadiness of measurement values, measured with the help of the Fast Fourier Transformation (FFT), is used as indicator for important stream partitions.
- *fftSlope()*: control evaluates the alteration of the data stream's frequency spectrum. After the FFT has transformed the signal from time to frequency domain, the slope calculation together with the threshold control recovers significant shifts.

Any data stream can be described as the combination of three cases:

- Expected trend: the trend is regular, and values are precise and accurate. There is no high variance, and the trend is normal.
- Slow change: the trend shows an unpredicted but lasting variation. In this case, values are still precise, but not accurate. When a potential outlier comes, we need to wait for the following values to say if effectively it is an outlier, because if the next values are precise and so are near the outlier, then we may have a slow change or a problem with the sensors.
- Oscillatory / bursty / sudden change: the trend shows discontinuous variations and values are not precise, but they can be both accurate and not.

The recorded trend is also useful to make choices regarding, for example, whether to aggregate or not the received data.



DQ improvement tools based on statistical techniques

In static table, a method for imputing missing data is that of forwarding values, meaning that a missing value is substituted by the last value observed for that missing attribute. Forwarding values is not always the best method when working on static tables but can work very well when applied to data streams.

The simple imputation method is based considering every attribute individually or using interpolation.

We can use mean or median to replace missing values or we can simulate the distribution of incoming values.

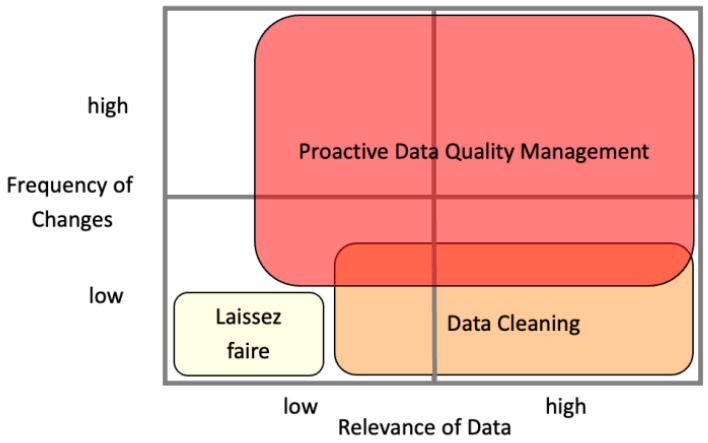
When dealing with outliers instead, many approaches are possible:

- Error bounds, tolerance limits – control charts,
- Model based – regression depth, analysis of residuals,
- Geometric,
- Distributional,
- Time Series outliers.

12. Process-based data cleaning

If not following data-based data improvement, a viable and complementary way is process-based data improvement.

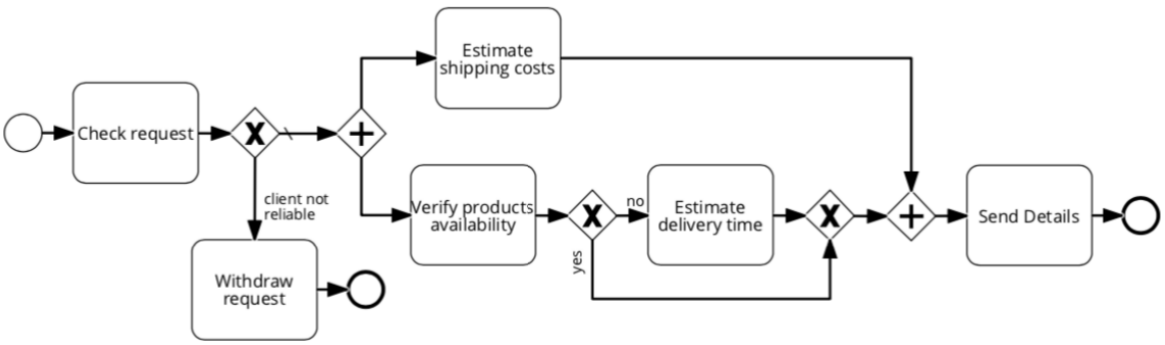
In a quite static database, data quality issues are solved by working directly on data with data-based data quality improvement approaches. When instead the frequency of change in a database is high, a proactive technique may be preferrable. **Process-based data cleaning** is proactive and consists in finding the causes for bad data quality and solve them, so the next generated data will not be affected by the found issues.



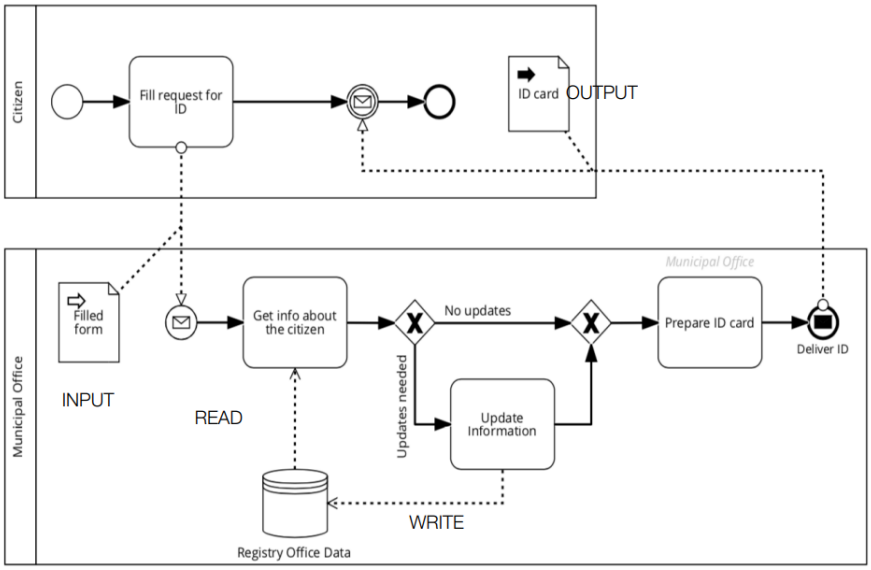
It's worth noting that a third possible viable solution is to just ignore the data quality issues: not all databases are relevant and deserve to be spent money on for data cleaning purposes.

A **Business Process** is a sequence of activities carried out starting from an input received from a supplier, aimed at the realization of an output to which the customer assigns a value. The focus of a business process is on the activities and on the exchange of information they generate (information flows).

Business processes can be modeled using **BPMN**, or Business Process Model and Notation, that is a standardized graphical notation used for representing and designing business processes.



Business processes allows to represent the data used during their execution. A Data Object represents the information used during the process execution. Examples of data objects are documents, mails, etc. From business processes representation through BPMN, it's possible to see how the data are flowing and the dependencies between different data objects and activities.



Poor quality (inaccurate, incomplete or out-of-date data) negatively affects the efficiency and effectiveness of business processes, leading to process delays, money losses and stakeholders dissatisfaction.

The implications of poor data quality in processes generally are:

- Wrong outputs
- Different courses of action
- Wrong analyses
- Failures
- Delays and timeouts

But poor data quality can also have no effect: some errors do not affect the execution of the process, and this usually means that the data are not relevant for that particular process.

In a business process, each activity depends not only on the data sources but also on the outcome of previous activities. This means that errors propagate from one activity to one other, through activities themselves or through data objects that are used along the way by more activities. As a consequence, data quality cannot be evaluated independently for each activity in the process, because every activity can introduce an error. To solve problems, it is needed to study interdependencies and the information flow inside the process.

Typical causes of errors include: input data that can be wrong or missing, external sources access and message exchange, work-arounds to the process, temporal aspects involving untimely information and delayed recording of information.

The objective of process-based analysis is to understand which activity generates a problem. To do so, checkpoints that perform quality checks are introduced so that data quality can be evaluated in each point of the process before and after failures.

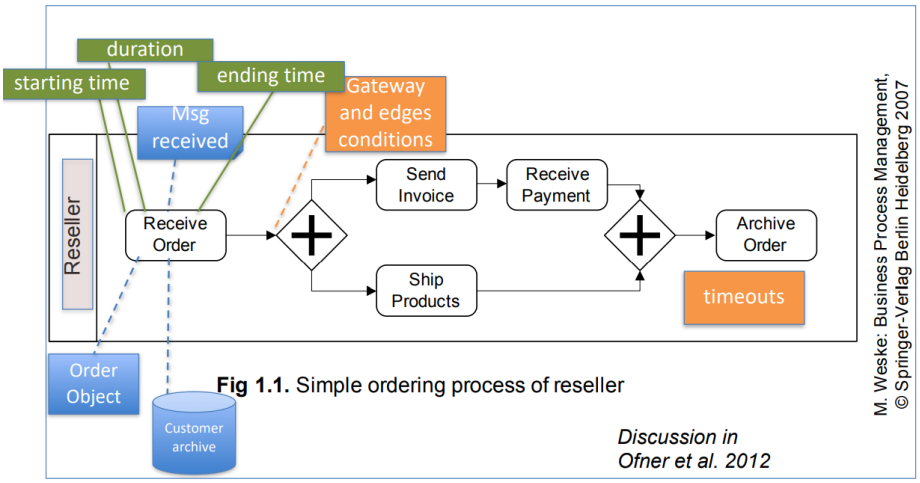


Fig 1.1. Simple ordering process of reseller

Discussion in Ofner et al. 2012

At the end, the activity that generated the error needs to be corrected, and the process needs to be modified in order for it to be able to detect future errors and avoid process failures.

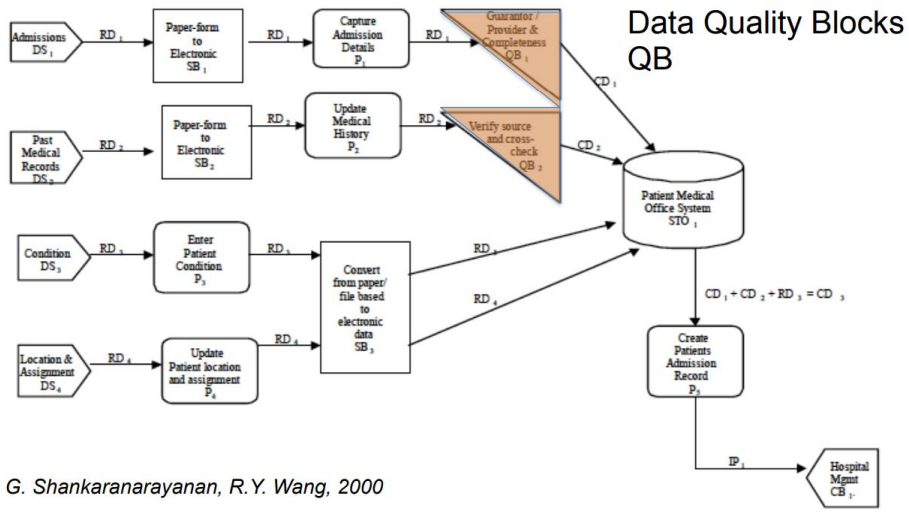
Notice that it is not necessary to catch all the errors, but just the ones that have a significant enough implication for the processes and are costly. To decide if it is necessary to catch an error, poor quality issues are evaluated, considering economic aspects and the entity of the impact of the error.

Another way, instead of BPMN, to model processes is **IP-MAP**, that stands for Information Product MAP. In it, information may be treated as a Product and the steps involved in creating it as a set of manufacturing processes. The data is followed inside the process and treated as the product of that process.

The Information Product (IP) manager visualizes the most important phases in the manufacture of an IP and identifies the critical phases that affect its quality. To do so, Data quality Blocks are introduced into the process to check the quality of data. Data Quality Blocks are used to represent the checks for DQ on those data items that are essential in producing a “defect-free” IP. So, whenever there is an exchange of information between two activities, a data quality block is placed.

It's important to take into account that data quality checks take time and inserting them does slow down the process.

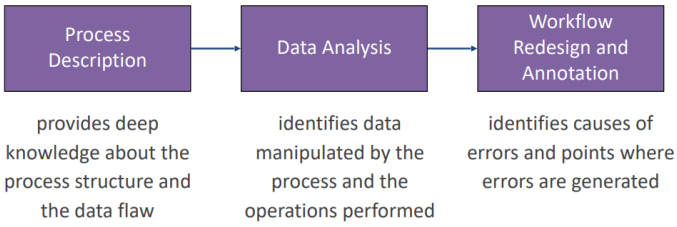
Figure 1: IP-MAP for Patients Admissions Record



G. Shankaranarayanan, R.Y. Wang, 2000

The issues to answer to are: how to design the data quality blocks? Where to position them? How many of them to position inside the process under analysis?

Methodologies do exist to suggest the process designer where to insert data quality blocks.

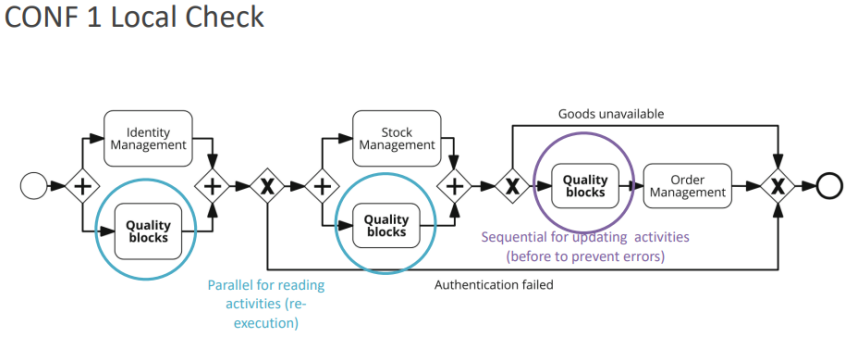


A *data quality control matrix* can be useful to link data problems to the quality controls that should detect them and identify the sources of data quality problems, the quality checks and corrective actions, the cost of data quality issues and the rating of the effectiveness of the IP-MAP constructs at reducing the level of data errors and irregularities.

	Information Product					
	Source of Data Errors or Irregularities That Occurred					
	Duplicate data in Component Data produced during Process 1	Data became obsolete during Storage	Typos from Data Source 1	Missing data from Data Source 1	Bad data from Data Source 1	Wrong format used in Process 2 to produce Component Data
Estimated Frequency of Error	2% of transactions	3% of transactions per month	5% of transactions	10% of transactions	6% of transactions	4% of transactions
Estimated Cost of Data Error per Information Product	$1	$1	$2	$5	$3	$4
Reliability Ratings of IP-Map constructs						
Quality Check 1	98% of transactions					
Corrective Process 1		90% of transactions per month				
Quality Check 3			85% of transactions	95% of transactions	88% of transactions	
Quality Check 4						97% of transactions
Overall Quality = Error Rate x (1 - Reliability Rate of IP-Map Construct)	.04% of transactions lead to IP's that are duplicates	.3% of data in storage lead to IP's that contain obsolete data	,75% of transactions lead to IP's that have typos	.5% of transactions lead to IP's that have missing data	.72% of transactions lead to IP's that have wrong data.	.12% of transactions lead to IP's that have the wrong format

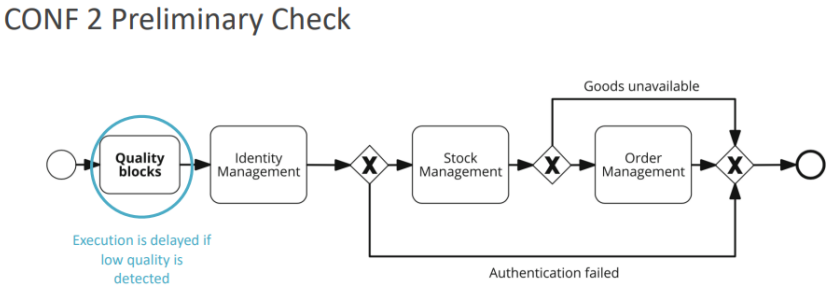
Several configurations of the Business Process are possible with a trade-off between execution time and DQ level.

A first approach with relation to where to put quality checks is the configuration in which checks are local, in parallel with activities.



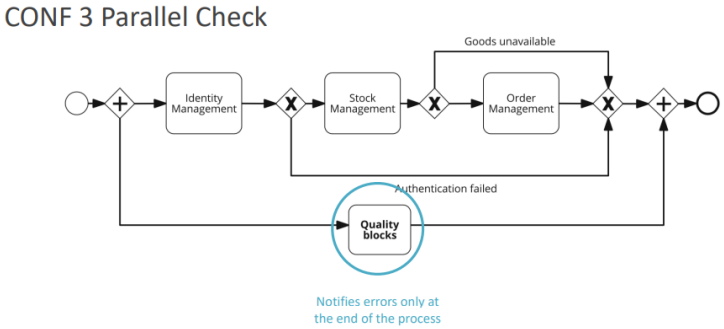
This approach is good for studying the single activities. If the activities write on a dataset, it's in however better to check the quality before executing the process.

A second configuration is the one in which all data is checked before starting the process, doing a preliminary check.



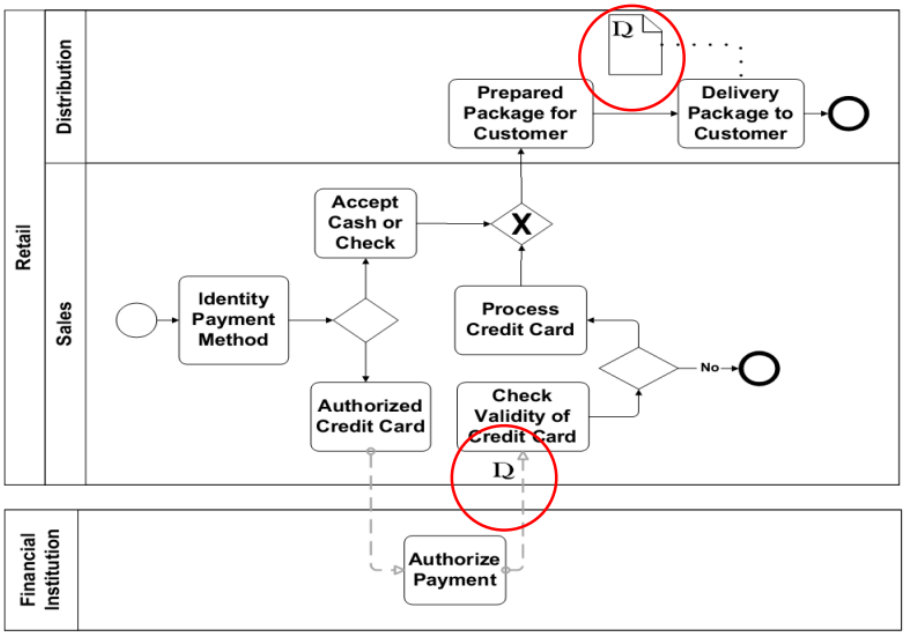
As a downside, if the data quality is low, the process is delayed a lot, and additionally errors inside the process are not detected precisely.

Always considering the whole process as a single unit, it's possible in a third configuration to perform a check parallel to the entire process.



This way, the notification of an error can happen only at the end of the process.

Flags can be inserted into the process to create a BPMN model with DQ flags and perform a Flag Analysis, aimed at finding the root of cause and relevance, considering data dependencies along with probabilities of occurrence of errors and computing the cost of poor data quality.



The analysis and selection of improvement activities must consider:

- The effect of the adoption of the data quality activity on the data quality dimension.
- The impact of the improvement action of the business process.
- The cost of implementing the improvement actions compared with the cost of allowing poor quality.

It's important to keep in mind that process-based data quality improvement is more expensive and complicated than traditional data-based data quality improvement.

Data streams are important because most of big data sources are data streams.

13. Big Data

Big Data is a collection of very huge data sets with a great diversity. Big Data are high-volume, high-velocity, and/or high-variety information assets that require new forms of processing to enable enhanced decision making, insight discovery and process optimization.

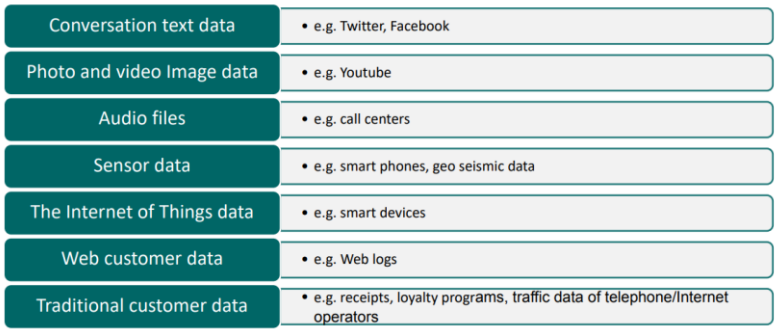
These characteristics are obstacles for classical databases, that employ solutions such as sampling that must be carefully utilized. In particular, the speed of data and its amount raise performance and scalability problems.

Big data should be thought of as a process, mostly aimed at getting to new insights, understanding how to turn them into action, resulting in business value.

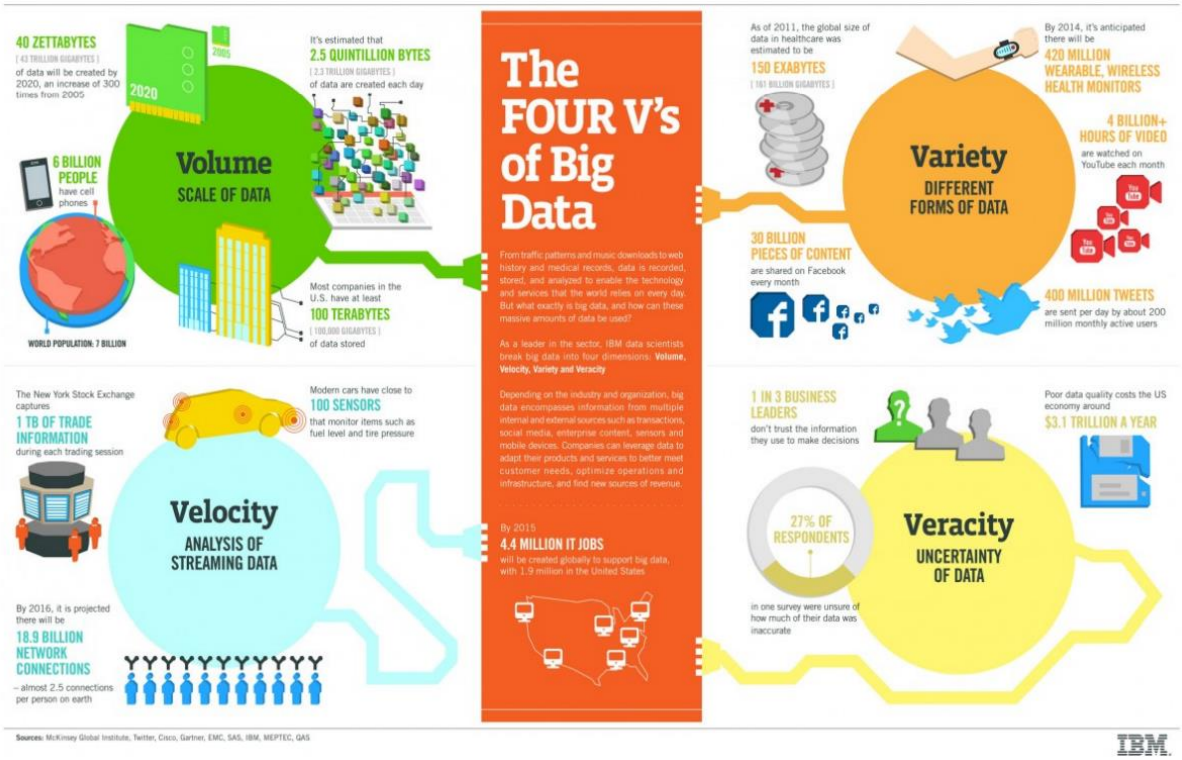
The Big data era is the consequence of **datafication**: our ability to transform each event and every interaction into digital data and our concomitant desire to analyze and extract value from this data.

There are three main kinds of data sources:

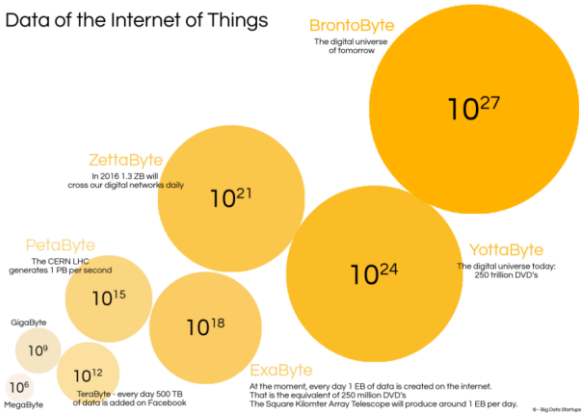
- Human-sourced information sources: data directly produced by humans (e.g., social networks, blogs, internet searches, picture archives)
- Process-mediated sources: data produced by companies and organizations (e.g., data produced by public bodies and institutions and data produced by the private sector)
- Machine-generated sources: data produced by sensors and machines (e.g., data from fixed sensors, data from mobile sensors, data from computer systems)



The four V of Big Data



Volume refers to the sheer size of the data generated or collected. With the increasing use of digital technologies, data is being generated at an unprecedented scale. Managing and processing large volumes of data is a key challenge in big data analytics.



To solve the issue, NoSQL (Not Only SQL) databases are employed with their capabilities in terms of scalability and flexibility. Also, paradigms like MapReduce are useful to process large amounts of data together.

Variety refers to the heterogeneity of data and encompasses the different types of data, both structured and unstructured, that are encountered in big data. Traditional databases mainly deal with structured data, but big data includes a diverse range of data types, such as text, images, videos, and more. Managing and analyzing this diverse set of data is a challenge in big data analytics.

Data are not generated by a single source but come from different sources. Managing heterogeneous information requires integration. Data Warehouse and relational databases are able

to manage this feature but require knowing in advance the data structure. In Big Data, instead, the data to be stored does not have a predefined structure.

Velocity represents the speed at which data is generated, processed, and analyzed. In the era of real-time or near-real-time data processing, the velocity of data is crucial. Examples include social media posts, sensor data, and financial transactions that occur rapidly and require quick processing. So, there are mostly not data to query on demand but dynamic data to analyze at real time (Data streaming).

Veracity refers to the quality and reliability of the data. With the vast amount of data generated, ensuring the accuracy and trustworthiness of the data becomes crucial. Data may come from various sources, and its quality can vary. Veracity emphasizes the need to ensure data quality for effective decision-making.

In fact, data are useful only if they can be trusted, and if data are not reliable then results obtained from their analysis are not reliable either.

Data veracity is about how accurate or truthful a data set may be and how trustworthy the data source, type, and processing of it is. Uncertainty with big data is increasing and there is no ground truth but a lot of conflicting sources.

Veracity is the V interesting for data quality.

Data Quality and big data

In the big data era, many sources are available online, but there is no guarantee that they are accurate, up to date and trustworthy.

When dealing with datasets that should be accurate, low accuracy is often found. In addition, sources quality changes over time, so a source with high quality may become bad quality or vice-versa.

The available data is not always up to date, but most of the sources contain data that represents the past.

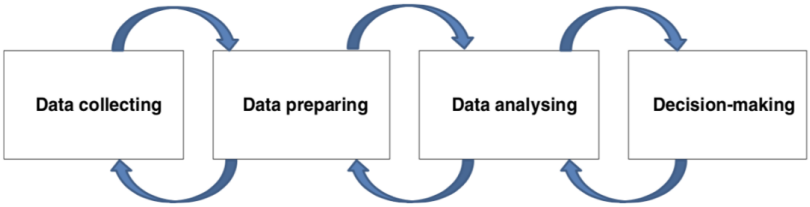
Trustworthiness is almost never granted, and usually depends on the chain of derivation of the data and on the reputation of the source. Fake news is everywhere.

However, high quality data are the precondition for guaranteeing the quality of the results of Big Data analysis. Big Data tried to overcome Data Quality issues with Data quantity, but it isn't enough and data quality is still an issue.

The challenges of data quality for big data are correlated with the 4 Vs:

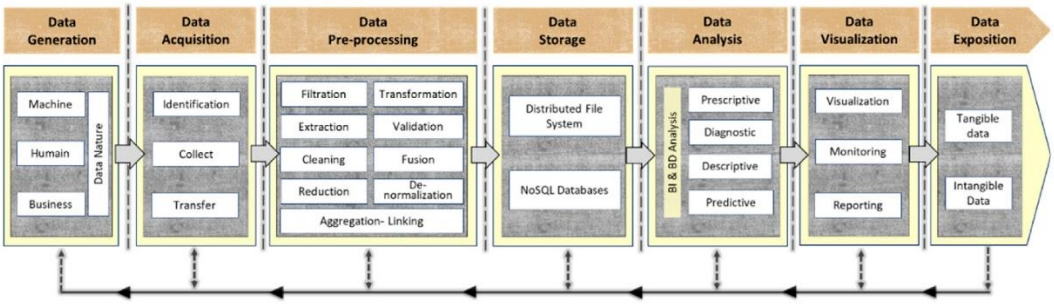
- *Diversity of data sources* (Variety): we need to adapt the algorithms for data quality and quality models to different situations.
- *Tremendous data volume* (Volume): the cost of cleaning data in terms of both time and money is high, even when parallelization is exploited.
- *Timeliness of data is very short* (Velocity): meaning that data becomes outdated and not valid quickly.
- *Missing standard for Data Quality* (Veracity): Standards have been proposed for DQ of traditional data sources but not for big data, we need to understand what should be expected.

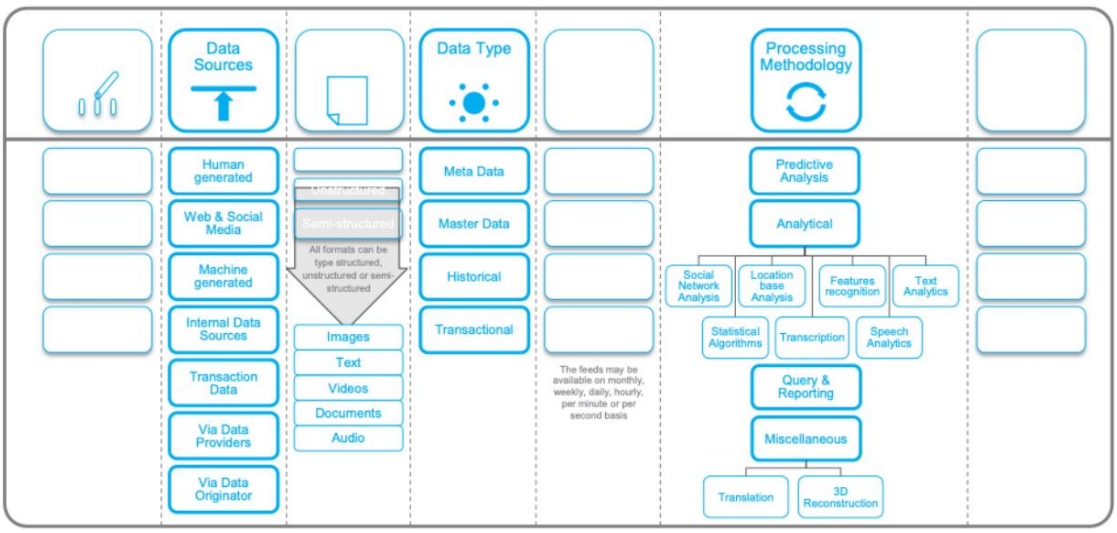
Big data can be seen as passing through a **big data chain**, that is a pipeline composed by a flow of activities, from data collection to decision making, for the big data process.



The main issues are related to velocity and veracity:

- Processing and manipulation: The velocity of the data can mean that only one part of the data is provided, which might give a different picture than when the whole dataset can be viewed.
- Noise: Data are incorrectly connected. Wrong values are entered. Some data from different periods are linked. This is not a problem for data analytics revealing patterns but for inferring about individual cases.
- Error: The context in which the data is collected is often unknown, and only the source has this information. If there are any changes in the way data is collected, which is not communicated, this results in erroneous results. More understanding of the context and of the changes in the context is needed.





Data quality assessment

To assess data quality for Big Data, other important dimensions need to complement the traditional ones.

Dimensions	Elements	Indicators
1) Availability	1) Accessibility	▪ Whether a data access interface is provided ▪ Data can be easily made public or easy to purchase
	2) Timeliness	▪ Within a given time, whether the data arrive on time ▪ Whether data are regularly updated ▪ Whether the time interval from data collection and processing to release meets requirements
2) Usability	1) Credibility	▪ Data come from specialized organizations of a country, field, or industry ▪ Experts or specialists regularly audit and check the correctness of the data content ▪ Data exist in the range of known or acceptable values
3) Reliability	1) Accuracy	▪ Data provided are accurate ▪ Data representation (or value) well reflects the true state of the source information ▪ Information (data) representation will not cause ambiguity
	2) Consistency	▪ After data have been processed, their concepts, value domains, and formats still match as before processing ▪ During a certain time, data remain consistent and verifiable ▪ Data and the data from other data sources are consistent or verifiable
	3) Integrity	▪ Data format is clear and meets the criteria ▪ Data are consistent with structural integrity ▪ Data are consistent with content integrity
	4) Completeness	▪ Whether the deficiency of a component will impact use of the data for data with multi-components ▪ Whether the deficiency of a component will impact data accuracy and integrity
4) Relevance	1) Fitness	▪ The data collected do not completely match the theme, but they expound one aspect ▪ Most datasets retrieved are within the retrieval theme users need ▪ Information theme provides matches with users' retrieval theme
5) Presentation Quality	1) Readability	▪ Data (content, format, etc.) are clear and understandable ▪ It is easy to judge that the data provided meet needs ▪ Data description, classification, and coding content satisfy specification and are easy to understand

This way, other aspects of data that are proper of Big Data are considered, like for example the source trustworthiness and reliability.

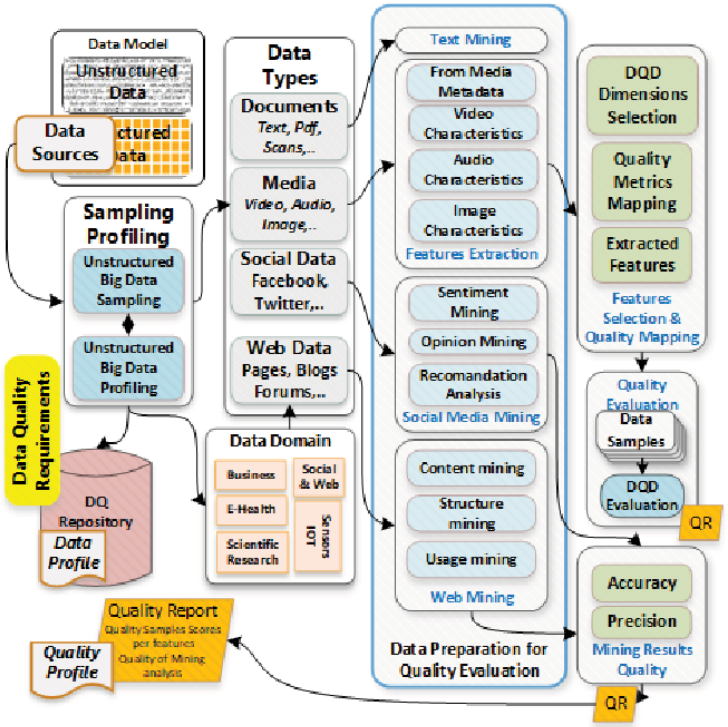
Data quality in use refers to identifying critical DQ dimensions that are important for the data processed by the Big Data project. Data quality is intended as the perceived quality of different attributes evaluated for the specific task and intended purpose, in the moment the data is meant to be used.

The **adequacy** is defined as the state or ability of data of being good enough or satisfactory for some need, purpose, or requirement. It's possible to make a distinction between:

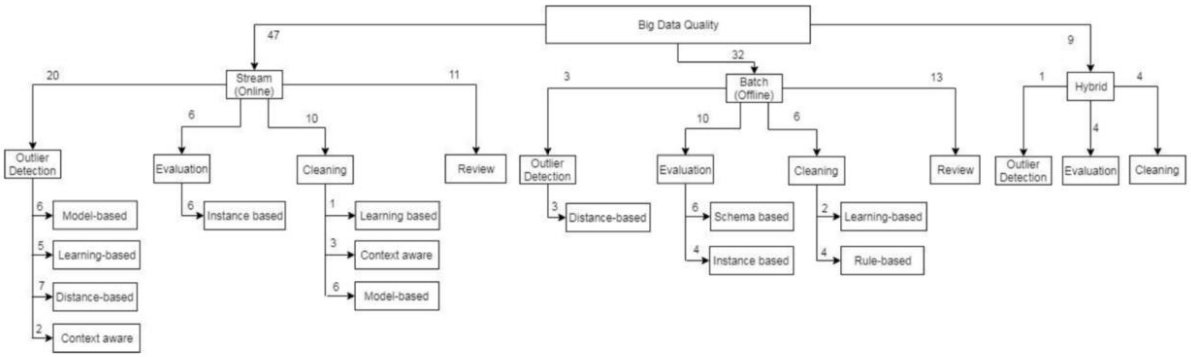
- Contextual Adequacy - capability of datasets to be used in a specific domain (relevant, accurate, credible, confidential, compliant).
- Temporal Adequacy - the data is within an adequate time slot (time-concurrent, current, timely, frequent).
- Operational Adequacy - capability to process the data by an adequate set of technologies.

	Velocity	Volume	Variety
Contextual Adequacy	Completeness	Completeness Consistency Confidenciality	Accuracy Consistency Credibility Compliance Confidenciality Understandability
Temporal Adequacy	Accuracy Currentness	Currentness	Consistency Currentness
Operational Adequacy	Confidenciality Efficiency	Efficiency	Accessibility Confidenciality Efficiency

Big data of new, evolving and heterogenous types and structures call for new data quality models and techniques. We need an increasing number of models, methods and algorithms, especially for unstructured data (around 80% of the data of companies is unstructured).



Data Quality improvement



Data quality improvement needs to face different limitations, starting from source dependent limitations:

- Resource constraints: requires computational capacity and time for assessment and cleaning execution. Computational, communication and memory limitations limit data quality.
- Source heterogeneity: differences in structure lead to unusual behavior. Detecting and cleaning data is challenging.
- Scalability: data and algorithms have to be distributed. If it is not possible, scalability cannot be reached.

And inherent limitations:

- Variety of arrival rate: processing existing data before the arrival of new data.
- Infinite data: with streams, data arrives continuously. Evaluation must be done online without interruptions.
- Transient Data: data expires and lose credibility. Processing must anticipate expiration.
- Distributed data points: integrate and extract correlations of data collected by different sources to obtain context information.

Big Data integration

The era of Big Data provides a high availability of sources, but these sources are often conflicting and there is no ground truth available.

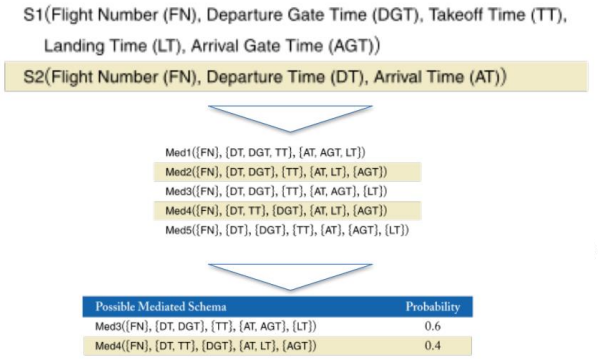
A possible solution may be to use probabilistic approaches to build and keep all of the available information with a relative probability that it is correct.

In any case, data integration is based on three major steps: schema alignment, record linkage and data fusion.

There can be thousands to millions of data sources in the same domain, but they often describe the domain using different schemata. **Schema alignment** aims at obtaining a single schema to conform all the data to use.

Probabilistic schema alignment embeds the uncertainty on how to model the domain and on the creation of the mediated schema, offering two solutions:

- Probabilistic mediated schema: it consists of a set of mediated schemas, each with a probability indicating the likelihood that the schema correctly describes the domain of the sources.



- Probabilistic schema mapping: schema mappings describe the relationship between the contents of the sources and that of the mediated data. In many applications it is impossible to provide all schema mappings upfront and probabilistic schema mappings can capture this uncertainty on mappings between schemas.

	Possible Mapping Between S1 and Med3	Probability		Possible Mapping Between S1 and Med4	Probability
M_1	{(FN, FN), (DGT, DDGT), (TT, TT), (AGT, AAGT), (LT, LT)}	0.64	M_5	{(FN, FN), (DGT, DGT), (TT, DTT), (AGT, AGT), (LT, ALT)}	0.64
M_2	{(FN, FN), (DGT, DDGT), (TT, TT), (AGT, LT), (LT, AAGT)}	0.16	M_6	{(FN, FN), (DGT, DGT), (TT, DTT), (AGT, ALT), (LT, AGT)}	0.16
M_3	{(FN, FN), (DGT, TT), (TT, DDGT), (AGT, AAGT), (LT, LT)}	0.16	M_7	{(FN, FN), (DGT, DTT), (TT, DGT), (AGT, AGT), (LT, ALT)}	0.16
M_4	{(FN, FN), (DGT, TT), (TT, DDGT), (AGT, LT), (LT, AAGT)}	0.04	M_8	{(FN, FN), (DGT, DTT), (TT, DGT), (AGT, ALT), (LT, AGT)}	0.04

Record linkage

Record linkage is the process of identifying and linking similar or related records from different data sources representing the same world object, enabling the consolidation of information for more comprehensive analysis.

Record linkage provide satisfying answers when data are traditional records that is well structured information with clearly identified metadata describing values, but with Big Data we do not usually have structured tables to integrate.

Object matching, or object linkage, refers to the process of connecting or associating related objects in a system or dataset, also of different types. It is very huge, goes from images to completely unstructured data like documents.

A possible solution to link objects of different data types (e.g., images, posts, etc.) is to use **ontologies** to explore the content of resources and compare them. An ontology is a formal representation of knowledge that defines concepts, relationships, and properties within a domain. In object linkage, ontologies provide a standardized framework for defining and describing entities, enabling more accurate and meaningful connections between objects based on shared ontological characteristics.

Challenges that object linkage needs to face include:

- Size of data: high volume of data requires a way higher number of comparisons.

- Time variance of data: different sources refer to unknown times that may be defined but unknown or even not defined at all.
- Schema information is poor: there might not be sufficient information or metadata to support object linkage.

Entity resolution is a data intensive, and performance critical tasks and cloud computing might help the execution of such procedure. In any case, blocking techniques become necessary in order to reduce the number of entity comparisons.

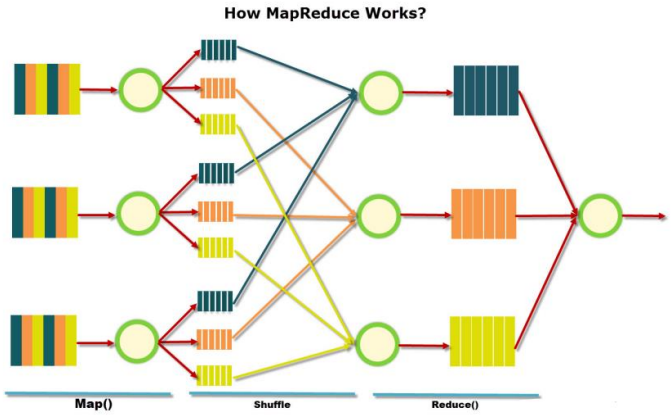
A way to reduce complexity of data deduplication operations is to exploit **MapReduce**, with a combination of blocking and parallel processing.

So, algorithms for data deduplication and reconciliation are developed with the MapReduce framework to parallelize operations and exploit data partitioning and redistribution. Map and Reduce functions are executed in parallel across many nodes utilizing data parallelism.

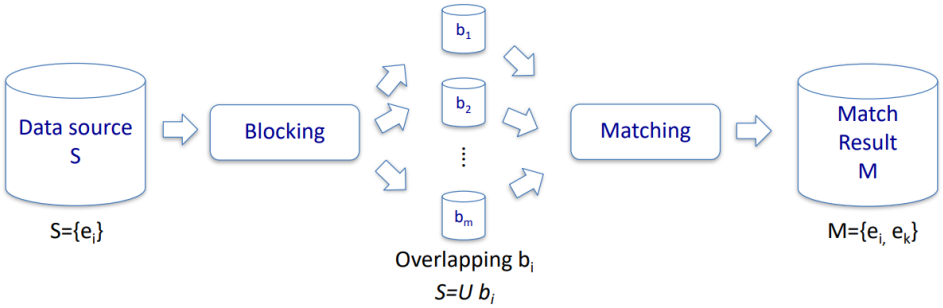
MapReduce is a framework for algorithms for processing data stored in a distributed file system in parallel using distributed computation nodes, based on two procedures:

- Map: performs filtering and sorting on each computational node on a single file block, and mappers analyse data resulting in couples <key, value>.
- Reduce: subsequently performs a summary operation.

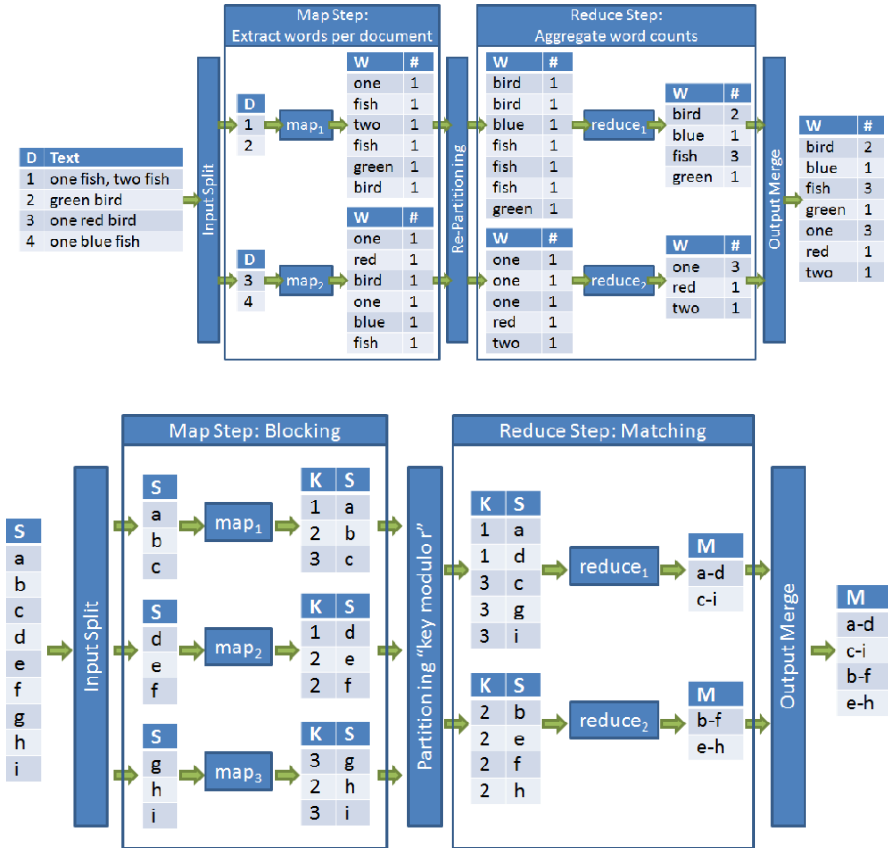
Data is distributed to the mappers that perform their tasks creating <key, value> pairs as results. These pairs are redistributed based on the key by the shuffler to the reducers, that combine them to produce the final output that is extracted with a committer.



Considering the problem of entity resolution, the mappers can perform the blocking of the data and the reducers can perform the matching.



Sorted neighborhood algorithms can be applied using the MapReduce framework. The simulation of the sliding of the window is assigned to the reduce workers. This way, advantages in terms of complexity and robustness are gained and sliding and multi-pass strategy windows can be applied.



The disadvantages of using MapReduce for record linkage lie in:

- Disjoint data partitioning: the map output is partitioned on the basis of its key value, which is not the best for sliding window approaches.
- Load balancing: Partitions can be characterized by a different size that depends on the key values. Execution time may be dominated by few reduce tasks.
- Memory bottlenecks: All the entities in the same block are passed to a single reduce call. Reduce task implies that data are processed row-by-row and all the entities within the same reduce block are compared with each other. All the entities are stored in the main memory. This is computationally intensive.

Data provenance

Data lineage is information and data about who created the source and why.

Data provenance is the collection of all the historical life cycle of the source and of data. Provenance is a record of metadata that describes entities and activities involved in producing and delivering or otherwise influencing a given object.

The main usages of provenance are related to:

- understanding where data come from.
- ownership and rights over a resource.

- making judgments about a resource to determine whether to trust it.
- verifying that the process used to obtain a result complies with given requirements and reproducing it.

Data provenance is needed for data integration and reuse because data comes from various and diverse data sources, has a varying quality and different scope, and is stored based on different assumptions.

Provenance can be analyzed and recorded from different perspectives:

- Agent-centered provenance, that is, what people or organizations were involved in generating or manipulating a resource (focus on the people manipulating the data).
- Object-centered provenance, by tracing the origins of portions of an entity, i.e., an object or a resource, to other entities (focus on the changes in the data object).
- Process-centered provenance, capturing the activities and steps taken to generate a resource (focus on describing accurately the activities performed on the data).

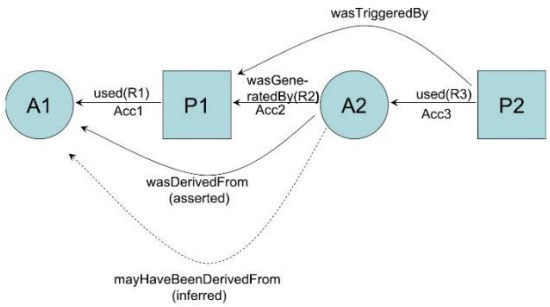
These three perspectives are usually used all together to have a good understanding of data provenance and produce useful metadata.

The important data quality dimensions for provenance are:

- **Data Trustworthiness**
  - Data Authenticity
  - Data Reliability
- **Dimensions of Believability**
  - Trustworthiness of source
    - Data Lineage – The origin of data
    - Related Artifacts and actors
  - Reasonableness of data
    - Possibility – The extent to which data value is possible.
    - Consistency – The extent to which a data value is consistent with other values of same data.

In summary, in order to evaluate the level of trust in data, it must be considered who created the content, how it was manipulated and by what process or source, and from where the content was taken.

To model provenance, the **Open Provenance model** (OPM) can be utilized. It is a conceptual model and set of specifications designed to represent and exchange information about the provenance of data, processes, and resources in a distributed and heterogeneous environment. It provides a standard way to capture and represent the lineage or history of how data was created, processed, and transformed, allowing for greater transparency and reproducibility in computational workflows.



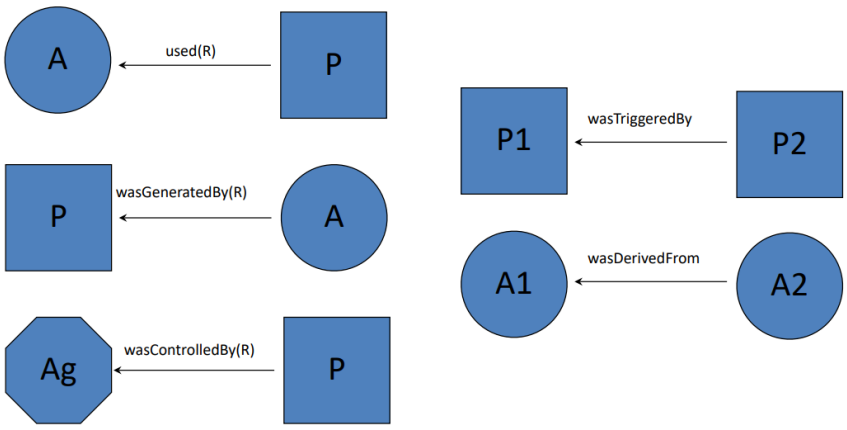
Relationships between agents, processes and objects are modeled, allowing to express al the causes of an item, allowing for process-oriented and dataflow-oriented views.

Nodes are constituted by artifacts, processes and agents, and different edges between them are possible, so to describe the history and all the activities performed on an object and by whom.

Artifact: Immutable piece of state, which may have a physical embodiment in a physical object, or a digital representation in a computer system.

Process: Action or series of actions performed on or caused by artifacts, and resulting in new artifacts.

Agent: Contextual entity acting as a catalyst of a process, enabling, facilitating, controlling, affecting its execution.



14. Data quality for Machine Learning

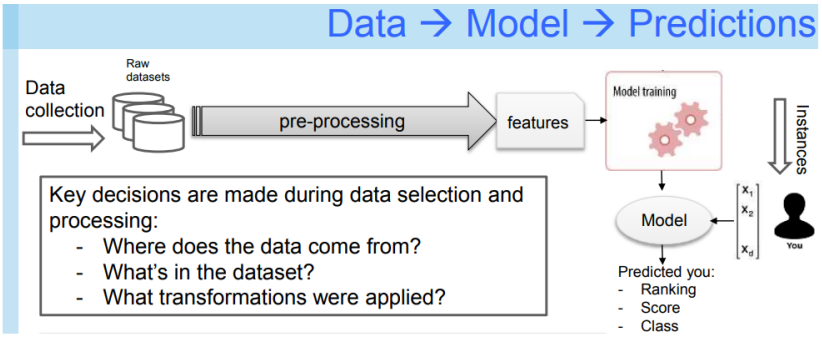
Machine learning is a branch of artificial intelligence that empowers computers to learn patterns and make predictions from data, without explicit programming, facilitating autonomous decision-making and problem-solving.

We can make the distinction between:

- **Unsupervised ML**: extract information from a distribution that does not require human labor to annotate examples (e.g., clustering)
- **Supervised ML**: learn to associate some input (x) to some output (y), performing a task consisting in assigning a label or target, learning from examples (training set). In this case, information about what we want to obtain is already present in the dataset and is utilized to train the model starting from already labeled examples.

In summary, supervised machine learning involves training a model on labeled data with known outcomes, enabling it to make predictions on new data. Unsupervised learning deals with unlabeled data, where the algorithm identifies patterns and relationships without predefined outcomes, often used for clustering and dimensionality reduction.

A normal pipeline for supervised learning involves data collection, splitting into training and testing sets, feature engineering, model training, evaluation on the test set, and, finally, using the trained model to make predictions on new, unseen data for real-world applications.



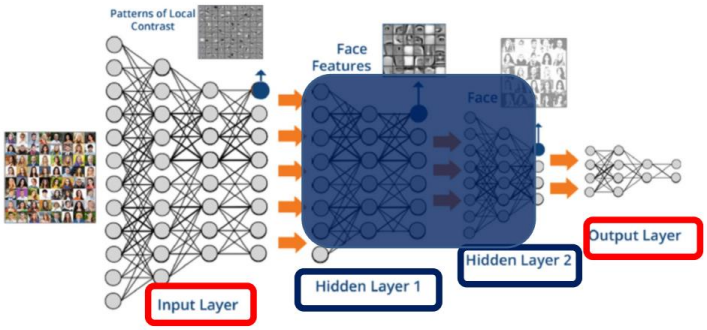
The model must be trained and then used for tasks such as prediction or classification, and the quality of the prediction of the model depends on how good it is trained. Bad input for the training means bad output afterwards, and data quality can heavily influence the quality of training. We are in front of the usual GIGO – Garbage In Garbage Out situation.

For Machine Learning, data quality is intended in terms of fitness for use, as a measurement of how the data fits the purposes of building a machine learning system.

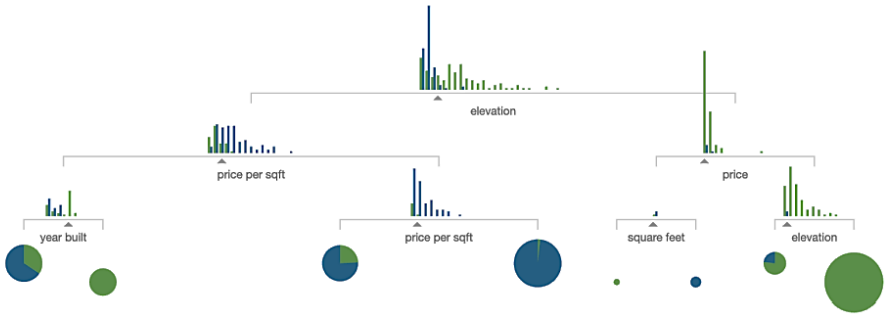
However, data preparation requires a lot of resources that may be needed somewhere else along the pipeline to train a Machine Learning model. Therefore, there is a tradeoff between properly cleaning the data and investing resources somewhere else in the training.

Machine learning uses include object recognition, image and speech, social media post analysis, also with location extraction, natural language processing, recommendation systems, healthcare diagnosis, fraud detection, predictive analytics, autonomous vehicles, predictive maintenance, financial modeling, customer churn prediction, and so on.

A **neural network** is a computational model inspired by the human brain's structure, consisting of interconnected nodes organized in layers. It's used in machine learning for pattern recognition and decision-making. Why a result is obtained with respect to another is not known, as these software are not transparent on how decisions are taken (research is ongoing on this topic).



A **decision tree** is a tree-like model in machine learning that makes decisions based on a series of rules and conditions. It's used for classification and regression tasks.



For decision trees, the way in which a result is obtained is known.

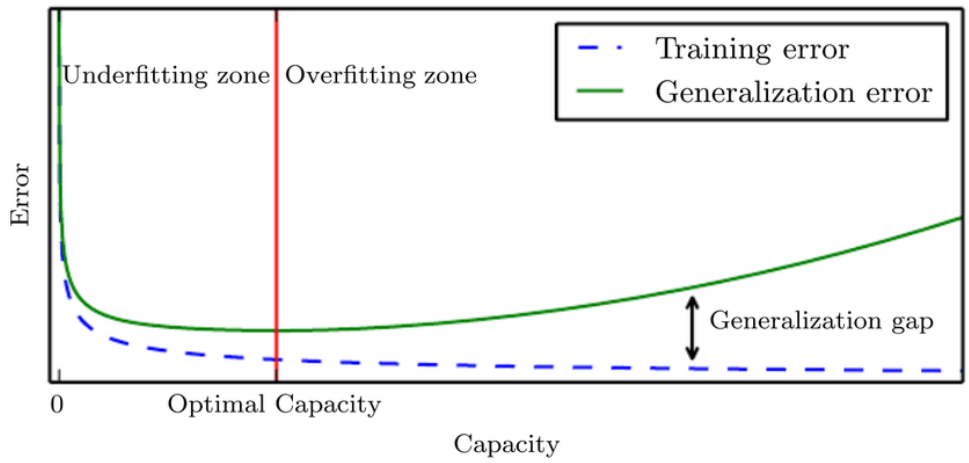
Back to Machine Learning models, the two essential phases in data science pipelines are:

- The training of the dataset, to prepare the model
- The testing of the dataset, to evaluate correctness of the model

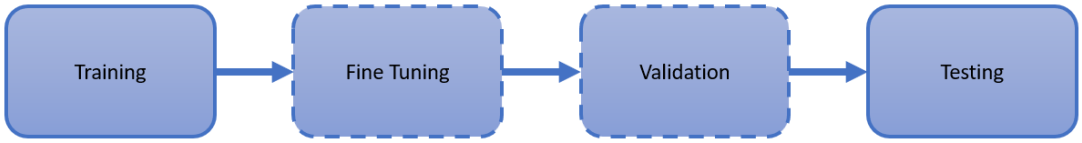
The **training error** refers to the error or the difference between the predicted outcome and the actual outcome of the data that was used to train the model. Essentially, it measures how well the model fits the training data. The goal of training is often to minimize this error, but a model that fits the training data too closely might not generalize well to new, unseen data and produce bad results for it.

The **generalization error** is the error that occurs when the model is tested on new, unseen data, i.e. data that was not used during the training phase. It measures how well the model can make predictions on data it has never seen before. A good model should not only perform well on the training data, but also generalize well to new data, keeping this error low.

The goal in machine learning is to find a balance between minimizing the training error (ensuring the model fits the training data well enough) and minimizing the generalization error (ensuring the model performs well on new, unseen data). This is often referred to as achieving good model performance without **overfitting**, that means fitting too closely to the training data, or **underfitting**, that means not to capture the underlaying patterns in the data because the model is not trained well enough.



Basic configurations for machine learning include training and testing phases, but fine tuning and validation phases can be singularly or both added between the two.



A model with hyperparameters is a machine learning model whose architecture or settings must be specified before training to optimize performance and control the algorithm behavior. **Validation** is the phase in which these parameters are set.

Transfer learning is a machine learning technique where a pre-trained model is adapted for a new, but related, task. So, a general model is trained first with very large datasets, and then in the **fine tuning** phase domain-specific training data are added to make it more specific.

Data quality issues can regard the training data and test data, which can both contain problems of different types, the majority of which are in training. The model quality depends on both the training and test data. The quality of results that are analyzed in turn depends on the quality of the input data.

Data quality in building a model

When building a model, usually the initial dataset is not created ad hoc, but an already existing one is taken. We need to assess its fitness for use, to see if it is appropriate, and this depends on the quality of data. We need to keep in mind that we are repurposing the data, because datasets are usually not built for machine learning purposes.

Formalized and fine-grained annotation of input data is still considered costly to produce. Consequently, a significant amount of workflow processing still deals with metadata wrangling, format transformations and identifier mapping. When putting together data and looking for matches, metadata is heavily utilized.

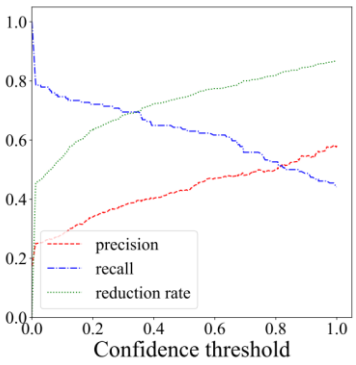
Data cleaning is largely a manual process and large quantities of already high-quality data are difficult to obtain. Then, usually data imputation needs to be performed because completeness is rarely present.

Data augmentation is a technique in machine learning that involves artificially increasing the size of a dataset by creating data or by applying various transformations to the existing data, such as rotations or flips. It helps improve model generalization by exposing it to a wider range of variations in the training data, enhancing performance.

Evaluating ML models

Machine learning models are evaluated using various metrics depending on the type of task (classification, regression, etc.). Common evaluation metrics include:

- Accuracy: Proportion of correctly classified instances.
- Precision: Accuracy of positive predictions.
- Recall (Sensitivity): Proportion of true positives correctly predicted out of all the instances that are actually positive (true positives and false negatives).
- F1 Score: Harmonic mean of precision and recall.
- Mean Squared Error (MSE): Measures regression model performance.



With the precision going up and becoming higher, the recall gets lower.

Model selection depends on the specific goals and characteristics of the problem at hand.

High variance in models can be good in some cases, because it is then possible to see both the relevant and not relevant information.

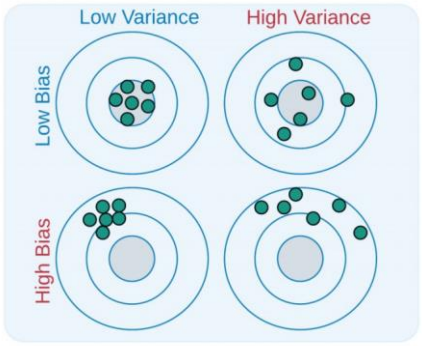
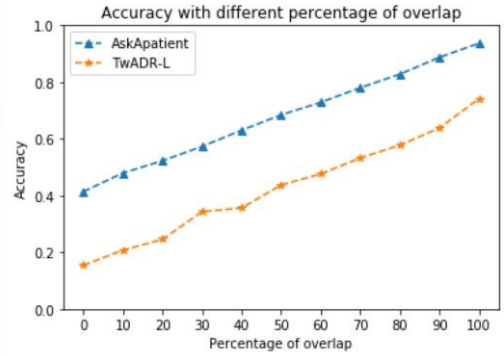


Figure 1: Visualizing bias and variance tradeoff using a bulls-eye diagram

Data quality in model training

Usually, the initial dataset composed by a collection of examples, that are a collection of features, is divided in around 80% of training data and 20% of test data.

There might be overlapping records between training set and test dataset, and this can impact the accuracy score.



Overlapping could be due to duplicate records, and, in this case, deduplication would reduce the overlapping but change the representativeness of the dataset.

It's difficult to estimate if a certain quantity of data is enough to properly train the model, also because this data can be of different data quality and so more or less fit for the intended use of training.

Data quality dimensions according to Chen are:

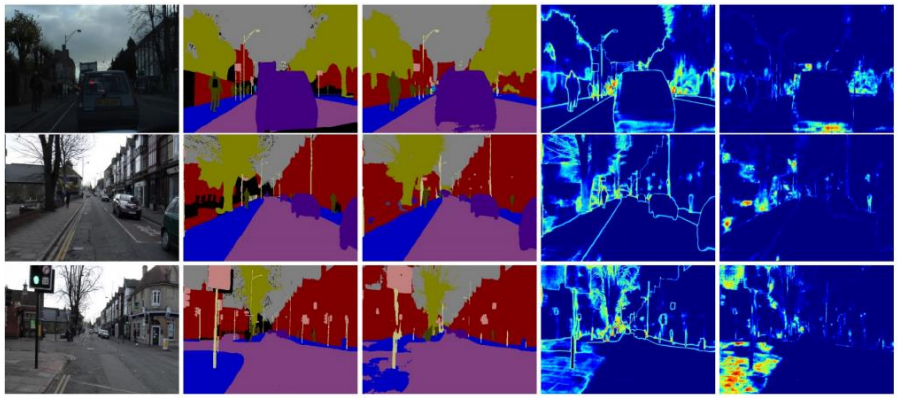
- Comprehensiveness: if an issue, the model could suffer the generalization issue
- Correctness: avoid label noise in training data
- Variety: to ensure the confidence of the evaluation results, cover all different cases, while increasing the quality of the sampling. Variety should be similar to the distribution of the population for each considered feature.

Possible improvements for these quality dimensions include:

- Comprehensiveness can be improved with:
  - Very large volumes of training data
  - Start from pretrained models, then fine-tuning
  - Check all concepts are covered and retrieve incrementally new data until very few will be found in a new iteration
  - Use domain knowledge
- Correctness can be improved with:
  - Data cleaning: outliers detection, imputation of missing labels, manual curation, evaluation with relation to other sources, etc.
- Variety can be improved with:
  - Data augmentation
  - Wang and Strong categories: intrinsic data quality, context DQ, representational DQ, accessibility DQ
  - Data with noise can be effective for training

Data quality in the results

There are two types of uncertainty in the context of machine learning and probabilistic modeling. **Aleatoric uncertainty**, or statistical uncertainty, arises from inherent variability in the data itself and is associated with randomness or variability that is irreducible, even with perfect information. It captures the noise inherent in the observations that cannot be eliminated. **Epistemic Uncertainty**, or model uncertainty, arises from a lack of knowledge or information about the true underlying model that generated the collected data. It is associated with uncertainty that could be reduced with more data or a better model. These uncertainties can be modeled.



(a) Input Image (b) Ground Truth (c) Semantic Segmentation (d) Aleatoric Uncertainty (e) Epistemic Uncertainty

The presence of these uncertainties means, for example, that even with a well-trained model but with bad input it's possible to get bad results. How much of the results is due to the model and how much is due to the input data quality is a question that can be addressed with probabilistic methods.

Uncertainty evaluation frameworks are based on multiple evaluation criteria, including confidence curves, error calibrators, confidence calibration, dispersion, and out-o-domain analysis.

In 2021, the EU Commission proposed new rules and actions for excellence and trust in Artificial Intelligence, and the obligations include the request for high quality of the datasets feeding the systems, for example to minimize the risks and discriminatory outcomes. Also, along with robustness and security, a high level of accuracy is required, and it depends on data quality.

Rule-based data cleaning

Rule-based data cleaning for machine learning involves defining and applying predetermined rules to identify and rectify errors or inconsistencies in a dataset. It uses dependencies, business rules and filters to define rules to ensure data quality, enhancing the performance and reliability of machine learning models.

A set of data quality rules Σ is specified for a schema by an expert or with automatic discovery of rules.

Rule-based techniques clean a dataset in two main steps: violation detection and error repair.

Violation detection is the phase in which the violations of some rules, if present, are detected.

There are multiple challenges associated with violation detection. A violation points out that a set of values are wrong but some values in the set are correct and other ones are wrong, so it is difficult to

understand exactly which values are erroneous. Violations can also be detected at a later stage with respect to when the error is introduced, so violations need to be traced back to identify errors and their cause with error propagation techniques. Moreover, the cost of violation detection can become very high and scalable violation detection is needed.

A possible answer to these difficulties is holistic error detection, which has the aim to pinpoint which values are more likely to be wrong by compiling all the violations. It considers all the DQ rules and define in one homogeneous representation (conflict hypergraph) all the values affected by the violations. The idea is that a value involved in multiple violations is more likely to be wrong with respect another value involved in fewer violations. It is a probabilistic method.

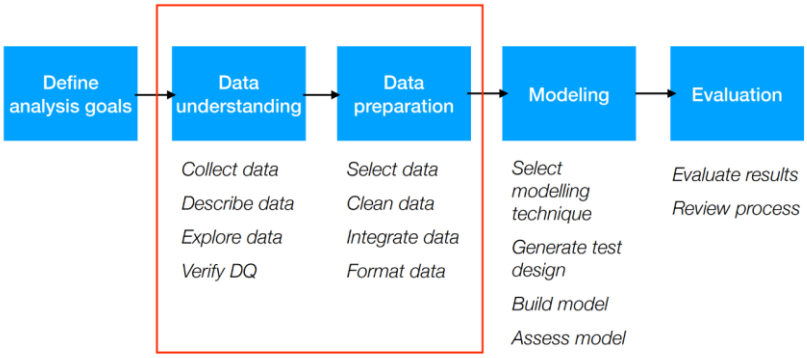
The second phase is that of **error repair**. The error repair is the process of finding another database instance that conforms to the DQ rules.

The main questions to answer in the repairing process are:

- Which is the repair target (what to repair?): If data is repaired then we trust the rules and data must be changed to remove errors. If the rules need to be repaired, then we are trusting the data and need a relaxation of the rules. So, in this second case, we assume that the data is correct, and the rule needs to be changed. Both the data and the rules can however need repair.
- Automation (how to repair?): errors can be repaired by using automatic tools or semi-automatic human guided tools.
- Repair model (where to repair?): it is possible to change the database or to build a model to describe the repair.

Data quality for Machine Learning

Machine learning phases and tasks are the followings:

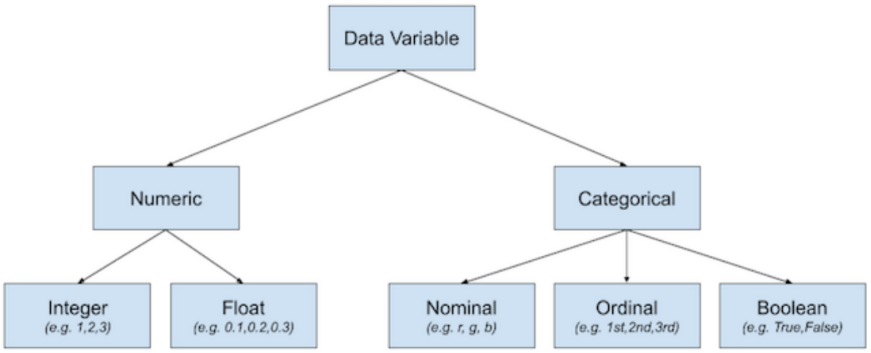


Raw data cannot be used directly for machine learning, and the reasons are that machine learning algorithms require data to be numbers and sometimes impose requirements on data, statistical noise and errors may need to be corrected, and linear or nonlinear relationships can be extracted by the data. A phase of **data preparation** is thus needed, comprising the tasks of:

- Data transforms
- Data cleaning
- Feature selection
- Feature engineering
- Dimensionality reduction

Data can be of different types, in particular it can be divided into numeric and categorical.

- **Numeric data** consists of numerical values and is continuous and is used for quantitative measurements.
- **Categorical data** represents categories and is discrete. It includes labels or groups and is often used for qualitative distinctions.



Some tools can deal with nominal values but other need numeric values: hence the need for data conversion, to convert ordinal fields to numeric ones in order to be able to use comparison operators. This is called *Ordinal Transform*.

Possible solutions for this conversion include:

- each value you have to create a binary flag which is 1 if the variable is associated with that value 0 otherwise.
- In other cases, groups are defined naturally - es. Attribute Profession: select most frequent ones, group the rest.

Discretization converts a numeric variable to an ordinal variable and is very useful for generating a summary of data. To perform it, the range of a continuous attribute is divided into intervals.

For distance-based methods it is important to perform normalization of values, since it prevents that the attributes with large ranges out-weight attributes with small ranges.

Missing values may have an impact on modelling: some tools ignore them while other tools use metric to fill in replacements. The modeler should avoid default automated replacement techniques: it is difficult to know limitations, problems and introduced biases.

To handle missing data in machine learning, possible solutions are:

- Ignore records (use only records with all the values): not viable when the percentage of missing values is high or varies considerably as it can lead to insufficient and/or biased sample sizes
- Ignore attributes with missing values (use only features -attributes- with all the values): the risk is to leave out important features
- Fill in the missing values manually: it must be feasible and face volume problems
- Use a global constant to fill in the missing value, for example a string “missing”, but the risk is that it can create a new class
- Use the attribute mean (or other properties) to fill in the missing value, but the chosen property must be unbiased
- Use the most probable value to fill in the missing value

All the techniques come to a price. In any case, with imputation more uncertainty is introduced, therefore sometimes the better solution is just to drop records with missing values. When you add a wrong or estimated but not precise value, bias is added, and such bias can affect the accuracy and validation of the mining results.

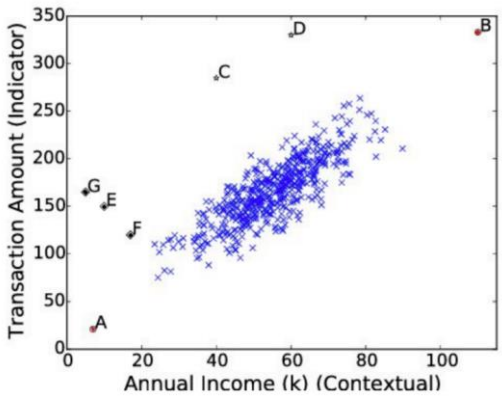
Basic cleaning operations include:

- Identify and remove column variables that only have a single value
- Identify and consider column variables with very few unique values
- Identify and remove rows that contain duplicate observations

Outlier Detection in high dimensional data, in real datasets that contain a lot of attributes and dimensions, can be difficult with traditional techniques. An option to have more effective approaches and to deal with high-dimensional data is the **dimensionality reduction**. These techniques use all the available dimensions of the original dataset and aim to find new datasets that preserve certain properties. Methods include the use of random projections or the generation of independent features from correlated attributes.

There are techniques that discover outliers using a subset of dimensions from the original dataset, for example by working separately on each subset of attributes or using functions of them.

Pair of attributes can be compared to see the points that are not in the usual behavior and find potential outliers.



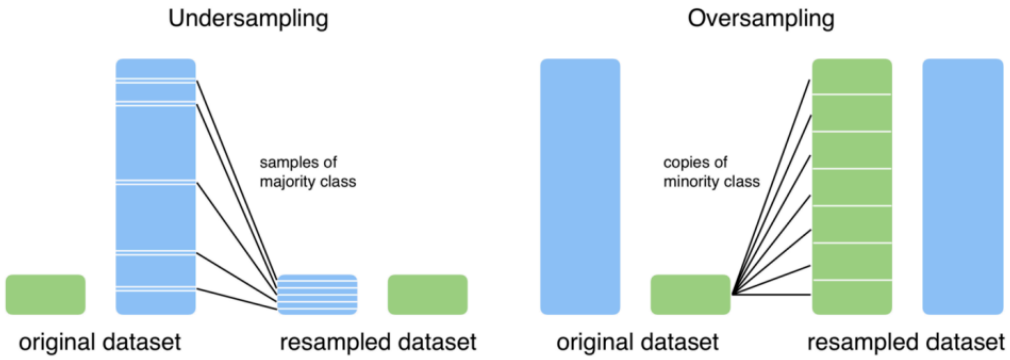
Redundant data occur when integrating databases and this redundancy needs to be detected and handling.

Another complexity lies in the fact that datasets can be **unbalanced datasets**, meaning that their classes have unequal frequency. The machine learning algorithm receives significantly more examples from one class, prompting it to be biased towards that particular class. It does not learn what makes the other class “different” and fails to understand the underlying patterns that allow us to distinguish classes. The majority class classifier can be characterized by a high accuracy, but it become ultimately useless.

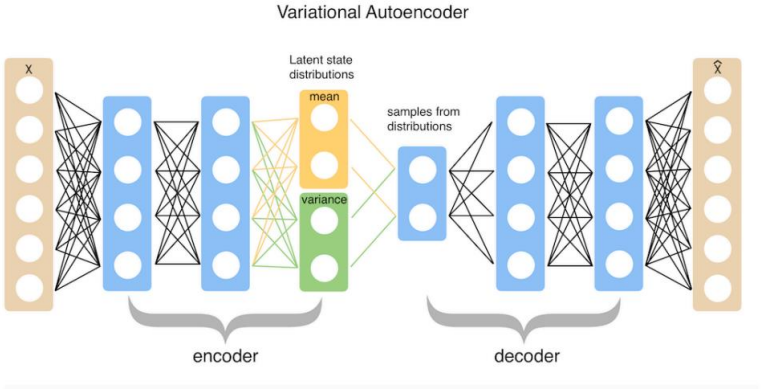
There are methods to adjust the dataset in order to produce an un-biased dataset:

- Resample the dataset: *undersampling* (the majority class is transformed into a smaller one) vs *oversampling* (the smaller class is enriched to reach the dimension of the majority class)
- Collect more data from the minority class
- Change algorithm

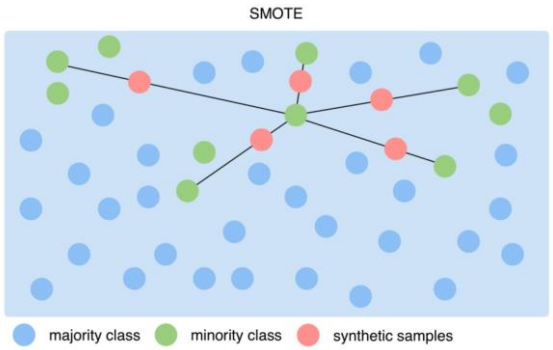
The first two methods aim to eliminate the bias with data enrichment.



Variational autoencoders: The encoder registers input data and turns them into a smaller, dense representation. The decoder network uses this representation and converts it back to the original input.



SMOTE: use a distance measure to synthetically generate a new instance with the same "properties" for the available features. It creates new, synthetic data points.



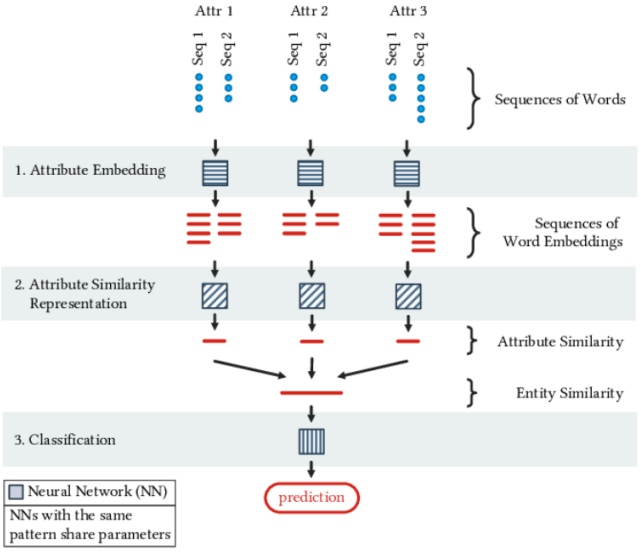
Machine Learning for Data Quality

An effective approach to data imputing is to use a machine learning model to predict the missing values. A model is created for each feature that has missing values, taking as input values all other input features.

One popular technique for imputation is a K-nearest neighbor model. A new sample is imputed by finding the samples in the training set "closest" to it (its k-nearest neighbors) and averaging these nearby points to fill in the value. It calculates the distance between instances, assigns weights, and imputes missing values based on the average or weighted average of neighboring values. The use of this model to predict or fill missing data is referred to as **Nearest Neighbor Imputation** or KNN imputation.

Configuration of KNN imputation often involves selecting the distance measure and the number of contributing neighbors for each prediction.

A binary classifier can also be trained to predict whether a pair of records are duplicates or not. This training needs a large training set and thus human labelling that can be expensive.

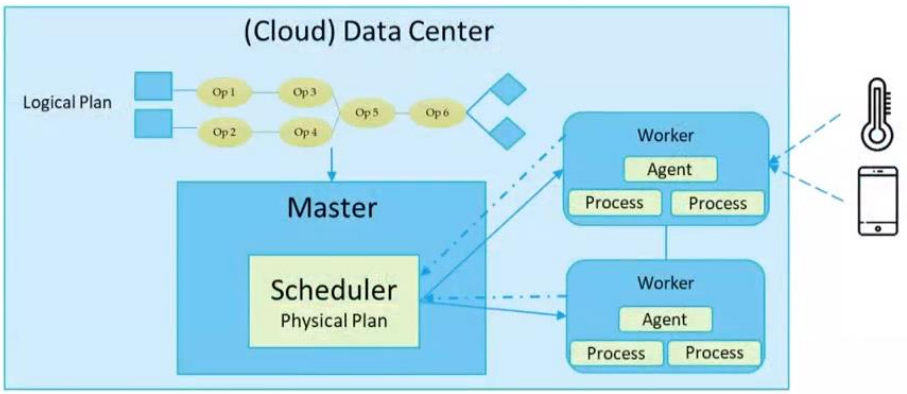


15. Data quality management for data streams

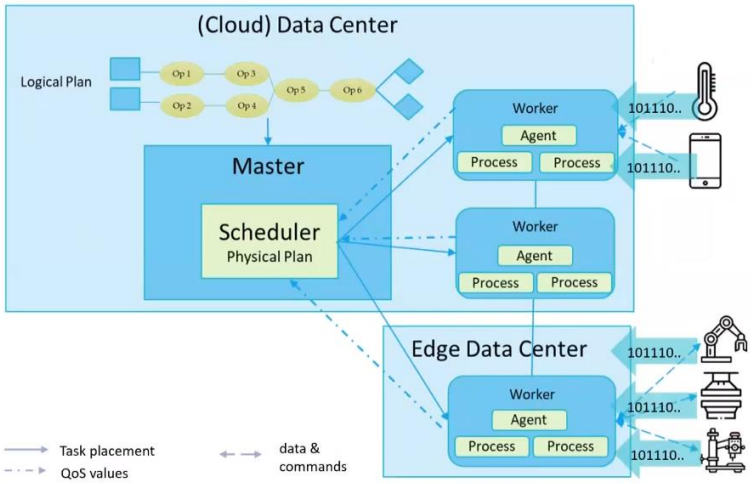
Sensors produce data continuously and at a high frequency, and this data stream can be useful in many ways, from smart manufacturing (smart equipment maintenance, product quality control, process monitoring, smart planning) to artificial intelligence applications.

An example of stream can be of 20 gigabytes of data per second.

The classical setup for data stream processes includes a data center with several servers with a master node responsible for executing the data stream processing by creating a physical plan to distribute the work to different workers receiving data from outside. Data comes in and is processed in a logical plan composed by a sequence of operations applied to data.



It's possible and often convenient to process part of the data nearer the resources producing it, thanks to **edge computing**. This way we limit the data size coming into the central data center and can be more efficient for what concerns the transportation of data.



In any case, a lot of challenges arise:

- Heterogeneity of data: it is an issue especially when deploying edge computing.
- Dynamics: we need to distribute our processing plan.
- Security: protect business secrets and production continuity
- Scalability: to handle burst of data and different loads

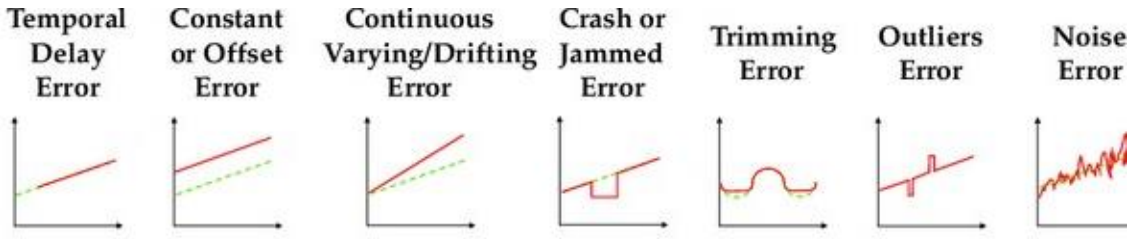
- Semantics: data is not valuable if you do not understand it; semantics also helps in understanding the quality level of data.
- Online analysis: what algorithm to use for analyzing data streams?
- Data quality: difficult to represent, measure and manage in data streams.

Internet of Things characteristics that effect the data quality of data streams are:

- Deployment scale: heterogeneous data sources, many devices
- Resource constraints: complexity of operations must be restricted
- Network: connection losses, package losses, delays
- Sensors: are often cheap hardware with issues regarding lack of precision, calibration problems, low accuracy, inconsistencies
- Environment & vandalism: climate or what happens influences hardware and may lead to failure or inconsistencies
- Fail-dirty: sensor fails, but sends erroneous data (e.g., creating outliers)
- Privacy: intentional reduction of Data Quality due to changed information to preserve privacy
- Security vulnerabilities: malicious attacks tampering with the data
- Data stream processing: makes use of approximation, load shedding when data is too much to process

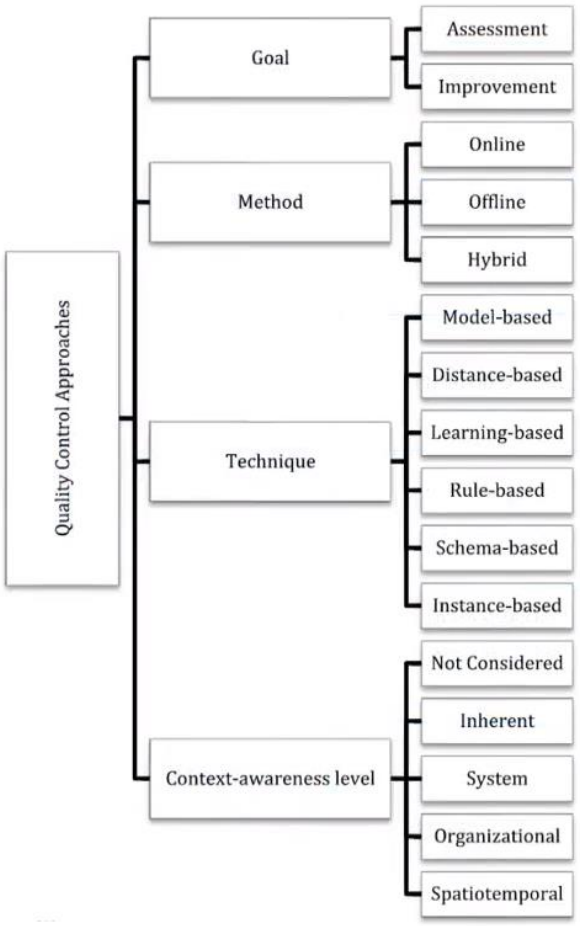
There are different kinds of **sensor errors** that can cause erroneous data to be produced and be present in the stream:

- Constant or offset error: observations deviate from expected value by a constant offset
- Continuous varying or drifting error: the deviation is continuously changing according to some continuous time-dependent function
- Crash or jammed error: the sensor stops providing values or gets jammed/stuck in some incorrect value
- Trimming error: data is correct for values in some interval, but incorrect outside
- Outliers error: observations occasionally deviate from expected value, randomly
- Noise error: observations deviate from expected value stochastically in the value domain and permanently in the time domain

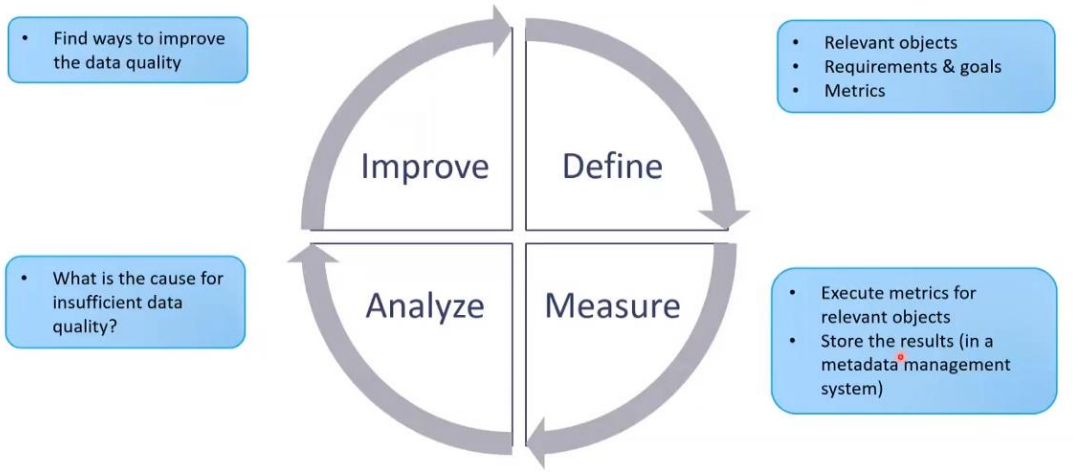


To deal with these errors, there are different viable quality management approaches. The approach to manage data quality is based on assessing data quality in the streams and optionally improving it.

This can be done online, so looking at the data in real-time while it is passing by, offline, so looking at already stored data, or in a hybrid way, using different kinds of techniques.



The **total data quality management** (TDQM) **cycle** comprises four phases.

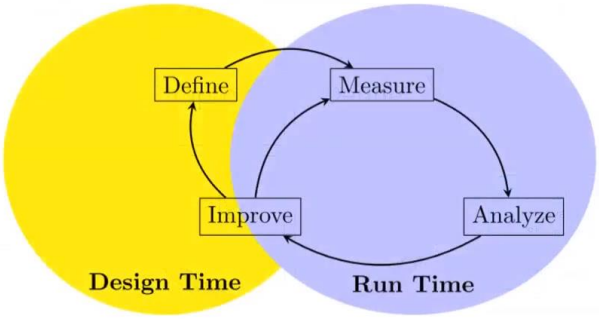


The first phase is the define phase in which the objects of interest are identified, and the requirements and goals for them along with metric to measure their accomplishment are picked.

Secondly, during the measure phase, metrics for the objects are computed and the results are stored in metadata management systems or monitored in real time to see how data quality is evolving.

Then, the analyze phase includes understanding the causes for insufficient data quality in order to improve it in the following improve phase, aimed at finding ways to improve the data quality by modifying both the data itself and the processes creating this data.

For data streams, the data quality dimensions and metrics are defined at design time, before executing the data stream processing pipeline. When the system works, the data quality is measured, and data is analyzed to see deviations or behaviors and understand the causes of the found issues. Improvement can happen both in real time (so, run time) and at design time by re-configuring machines and processes.



Define phase

Common data quality dimensions for streams are:

DQ Dimension	Description	Example Metric
Completeness	Fraction of missing values or tuples of received values/tuples	The number of non-null values divided by all values including null values in a window
Data Volume	Number of tuples a result is based on	Quantity of tuples in a window
Timeliness	The age of a tuple	Difference between creation time and current system time
Accuracy	Indicates the accuracy of the data, e.g., a constant measurement error or an estimation of result quality	An externally calculated or set value, e.g., the result confidence of a data mining algorithm
Consistency	Indicates the degree to which a value of an attribute adheres to defined constraints	Rule evaluation, check of constraints
Drop Rate	Indicator for the system performance	The number of tuples dropped during stream processing due to latencies.

Usually, these dimensions are looked at and measured for a certain defined period of time, as it is not possible for a stream to give a general measurement for these dimensions but only one relative to a time lapse.

Further dimensions that are relevant for internet of things are:

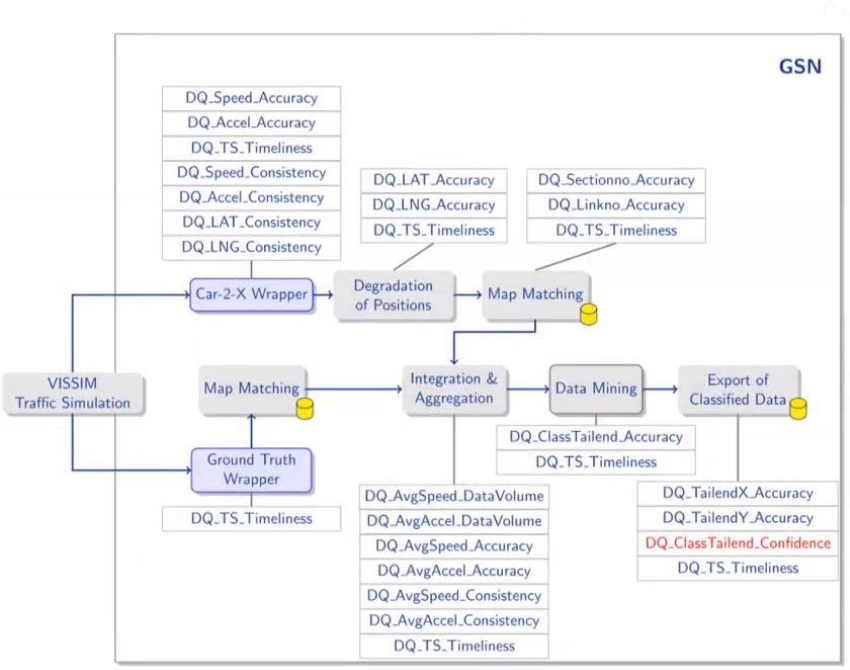
- *Trustworthiness*: how much we can rely on the data for some specific requirements
- *Security*
- *Privacy*: if the data contains sensitive information
- *Ease of access*
- *Utility*

They are generally less important than others, depending on the use case.

Measure phase

There are different ways to measure the data quality dimensions in data that is rapidly passing by in data streams.

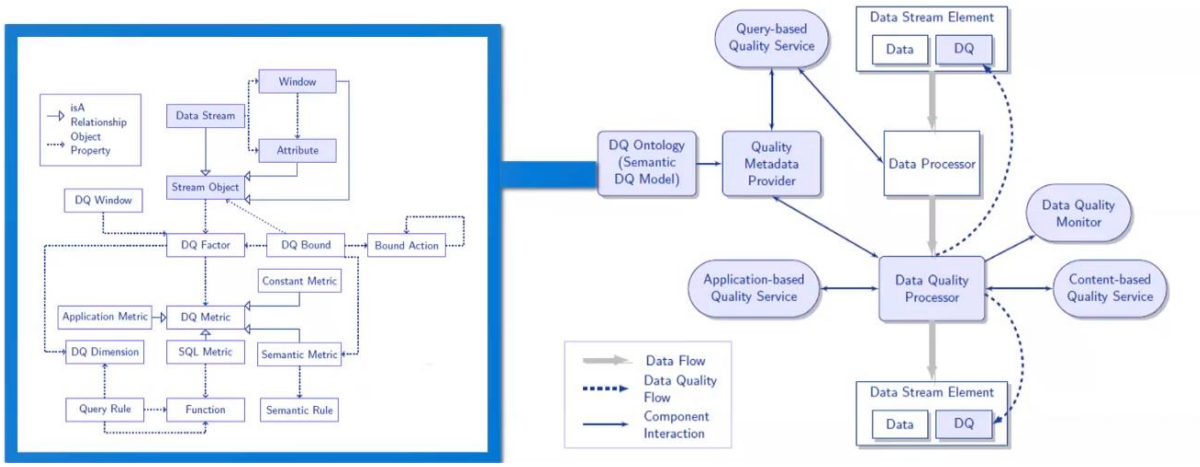
For example, in a project in which data streams are utilized to communicate events between different cars on the roads, a queue with different data quality components is implemented as follows:



The dimensions measured depend on what the data will be used for and what steps it will undergo.

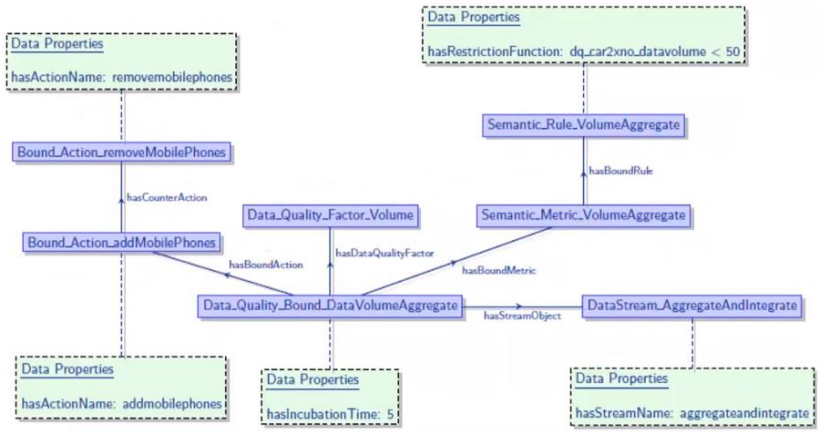
In **ontology-based data quality management**, data quality is considered a component of the data and is processed like the rest of the data that composes the data stream. Different services help to measure the data quality in real time.

Dimensions to be measured are defined in terms of ontologies, meaning that some metadata data quality model exactly defines which attributes should be measured in different ways. Ontologies help in defining the semantics of data, specifying data quality metrics, and capturing the relationships between different data elements.



An ontology that represents the data quality dimensions, metrics, and rules relevant to the organization's data serves as a framework for understanding and improving data quality.

Ontologies facilitate semantic integration of data by providing a common understanding of concepts and relationships. This helps in resolving semantic conflicts and ensuring consistency across diverse data sources.



Other approaches to measure data quality consist in the use of **quality windows**. The data stream is divided into non-overlapping, jumping data quality windows for each attribute, and the dimensions are evaluated for each window. In fact, windows contain the values for the attribute, a timestamp, and other attributes containing the data quality dimensions for that window. The quality values are calculated with adapted operators' functions.

Timestamp	...	210	220	230	240	250	260	270	280	290	300	310	320	330	340	350	360	370	380	390	400	...
Lifetime	...	300	298	295	292	292	292	292	283	274	265	255	252	250	242	233	206	195	190	187	184	...
Accuracy	...	3.0					3.3					2.78					2.86					...
Completeness	...	0.9					0.8					0.9					1					...

Adaptive window size algorithms are based on **interestingness**, in order to have finer granularity of windows when facing high peaks, threshold excess and fluctuations in the stream.

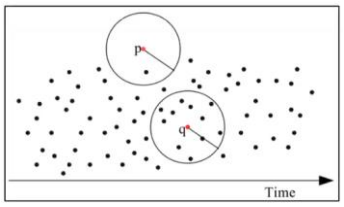
Anomaly detection is another important aspect of measuring data quality.

It's possible to distinguish between:

- Single point anomalies: one data point deviates from the normal trend of the data stream
- Subsequence anomalies: more than one subsequent point deviates from the trend of data

One approach to detect **single point anomalies** is to use a predictive model, in which predicted and observed value are compared for each data point and that data point is considered an anomaly if the difference between observed and predicted value exceeds a certain threshold.

Another kind of approaches, always for single point anomalies detection, are simple distance-based approaches that mark an object as outlier if there are less than a number k of objects within a given distance.

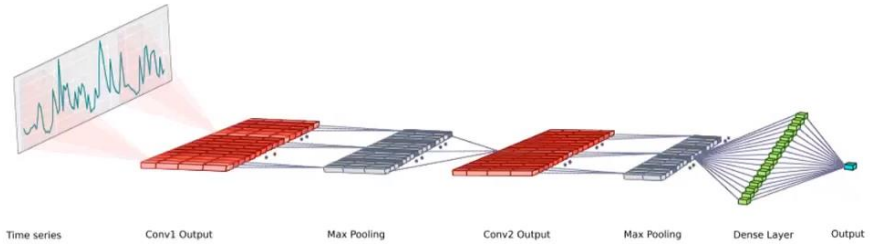


Other techniques are the ones based on the K-Nearest neighbor (KNN), for example distance-based techniques, that calculate a weight for each point by considering the sum of the distances of its k nearest neighbors, and then rank the points according to these computed weights, in order to classify as outliers those with largest weight.

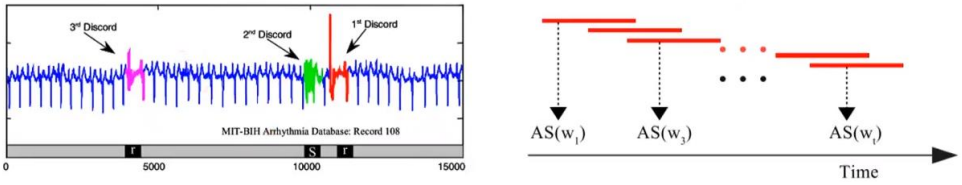
Some approaches make use of autoencoder (AE) that learns approximation to the identity function, with an encoder compressing the data and a decoder reconstructing it and using the error between the original and reconstructed data to understand if a point is an anomaly.

Further techniques exist, based on support vector machines, principal component analysis (PCA), histograms, clustering, and deep learning.

An example of an unsupervised technique based on deep learning is *DeepAnT*, that detects point anomalies, contextual anomalies and subsequences. It exploits a time series predictor that makes use of a neural network to predict the next value on a defined horizon (prediction window) based on history windows (context). Then, an anomaly detector determines anomalies by calculating error between actual and predicted value using Euclidean distance as anomaly score and using a threshold.



To detect **subsequence anomalies**, we identify a discord as a subsequence that has the largest distance from other subsequences in tis vicinity and is a deviation.



The distance between each subsequence with a certain length and other subsequences is calculated, using sliding windows to create the subsequences, and calculating an abnormal score (AS) of windows first and then of the whole test sequence.

Analyze phase

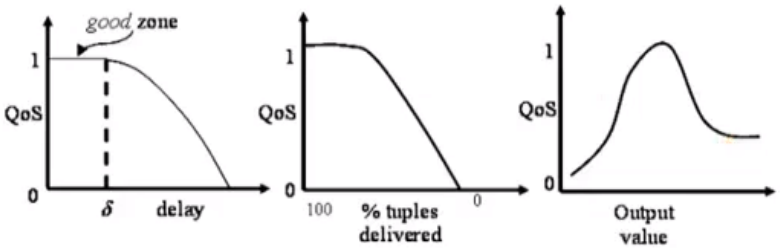
One way to analyze the dataset and data quality is to look at the **quality of service**. We look at what is coming out of the data processing and verify if the results satisfy some quality-of-service requirements, such as, for example, latency, throughput, etc.

We need to understand what resources are needed to compute a set of queries with quality-of-service specifications and satisfy the requirements. If, on the opposite, we have resources available, we need to understand if they are enough and well distributed to satisfy the requirements.

Quality of service monitoring is implemented in many data stream processing processes. One tool doing this is *Aurora*, that aims at maximizing the quality of service for some produced outputs.

Quality of service is considered a multi-dimensional function over several attributes and dimensions: response time, tuple drops, utility of output values.

Normalized graphs for each output with relative thresholds are produced on these quality of service attributes or on other ones or on combinations of them.

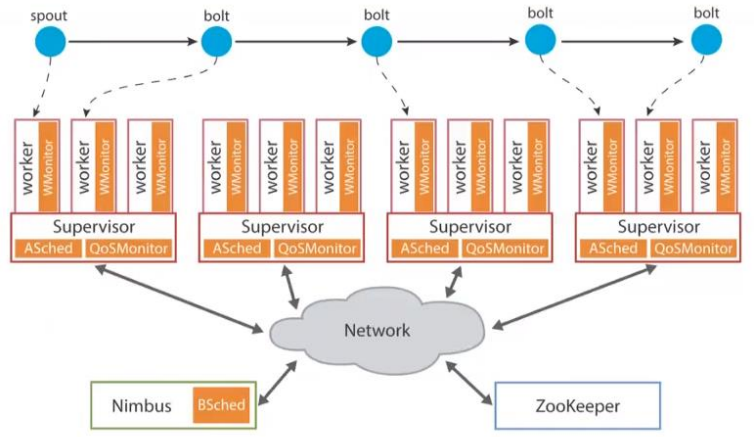


If some requirement is not fulfilled, actions such as activating a load shedder on bad performance or overload can be taken.

Apache Storm is a more modern system for quality-of-service monitoring that analyzes data streams and puts in place countermeasures when needed.

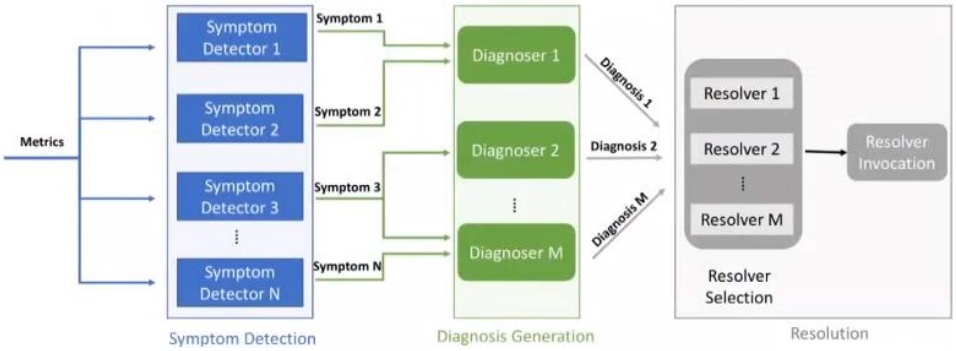
Metrics bolts (operators that process the data) are present in each topology of data, and different kinds of metrics, so system metrics and topology metrics, are collected to be analyzed.

As a consequence of the process, adaptive scheduling is possible to take countermeasures to modify the original query plan and, for example, redistribute tasks to workers.



Quality of service monitors keep track of network latency across nodes and monitor availability and resource utilization, while worker monitors compute exchanged data rate.

Dhalion is a self-regulatory system based on quality of service implemented on top of twitter Heron, that is self-tuning, self-stabilizing and self-healing. A health manager process periodically invokes policies (invasive and non-invasive) evaluating the state of the topologies and detecting symptoms and reasons for them in order to solve the issues.



Improve phase

The usual requirements for cleaning time series data include:

- High throughput
- Monitor the data in real-time and push immediately an alert to perform a cleaning step if needed
- Apply as less modifications as possible to not change the data too much and leave it authentic, and for performance reasons

The possible approaches include **smoothing-based** approaches, using moving averages, autoregressive, Kalman filter models, etc., or **constraint-based** approaches using dependencies and constraints in data for cleaning, or **statistical-based** approaches, exploiting concepts like the maximum likelihood, Markov models, binomial sampling, spatial-temporal probabilistic models, etc., or **anomaly detection**-based approaches using clustering and distance measures.

Smoothing-based cleaning has the goal to eliminate noise in data to obtain a better representation of it. This approach is not used a lot in time series as data is modified largely, but mostly exploited when there is no possibility to solve the issues differently.

Moving averages (MA) can be used to move over the data and calculate the average of the last n values to predict the value at current time t. *Autoregressive* (AR) models for random processes use regression to describe previous variables. Combinations of the two are of course possible, like ARMA or ARIMA models. *Kelman filter* models describe the evolution of states over time based on previously observed measurements and on the description of the assumed linear dynamical system to recursively estimate, predict and update.

Constraint-based cleaning techniques could use *order dependencies* (OD) that define an order of values and compare the records to see if these dependencies are respected, or *sequential dependencies* (SD) that focus on differences in values between consecutive data points in time series.

Speed constraints describe how fast values change in a certain window, and detect unexpected behaviors to correct them.

16. Truth discovery

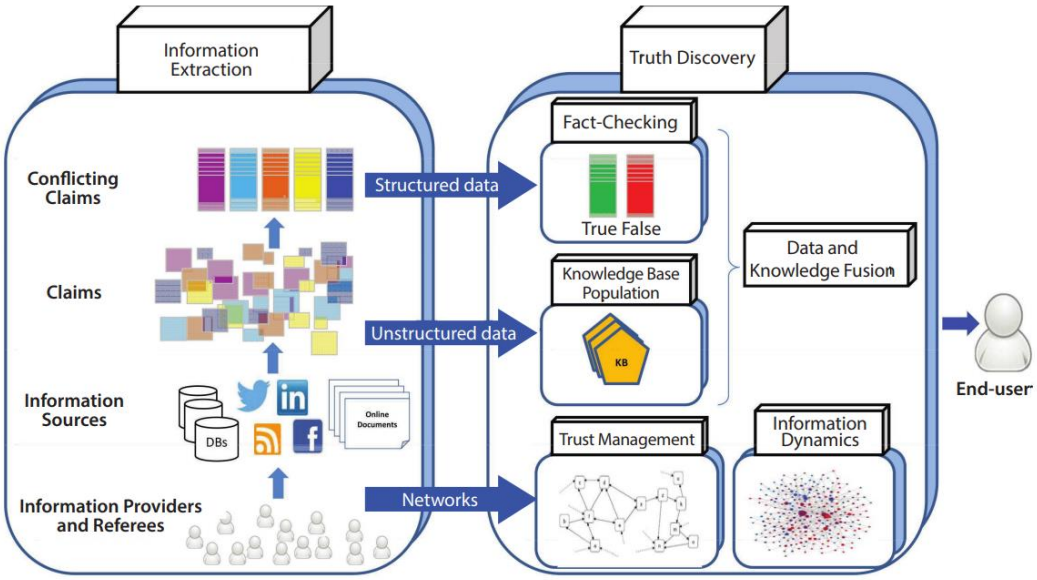
Especially in the era of big data, the ground truth may not be available to verify the correctness of some information, with regard to how truthfully it describes the real word. Different research directions focus on how to compare data that is available and make decisions about which is the right one.

Truth discovery is the process of determining the true or correct values of information from a set of conflicting and potentially unreliable sources. In many real-world scenarios, data may be obtained from various sources, each with its own biases, errors, or inconsistencies. Truth discovery aims to identify the most accurate and reliable information by reconciling conflicting data from different sources.

The truth discovery pipeline starts from extracting the information from the sources, then processing it, and ultimately performing the truth discovery phase.

Different kind of sources may be involved, comprising both structured, semi-structured and unstructured data, or combinations of the three. These sources are analyzed because, with no ground truth available to compare their data values with, their source reputation gains increasing importance. In fact, assessing the reputation of each data source is a critical step in determining how much weight or trust should be assigned to the information it contributes.

The **reputation** of a data source refers to the perceived reliability, credibility, and trustworthiness of the information provided by that source. Key factors contributing to the reputation of a data source include: accuracy, consistency, reliability, transparency, historical performance, independence and expertise.

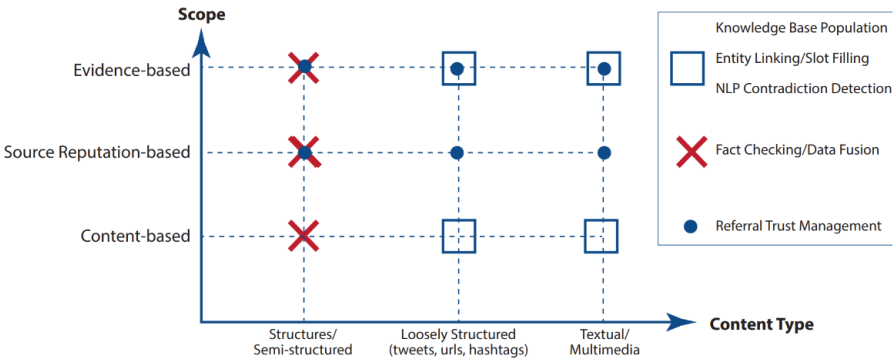


After the claims are extracted from the sources and conflicts arise between them, fact checking happens. Claims are also saved in a *knowledge base* and used afterwards when new claims come in, to have more information to use to do fact checking.

There exist three main kinds of truth discovery approaches:

- Content-based: primarily relies on analyzing the information provided by different sources and compares the data itself and the values associated with the same entities or events across multiple sources to identify inconsistencies or conflicts.
- Source reputation-based: emphasizes the reliability and trustworthiness of the sources themselves, by considering the historical performance and credibility of each source to determine the likelihood of accurate information.
- Evidence-based: considers the supporting evidence for the same piece of information, by analyzing the consistency and agreement of evidence across sources, also making use of probabilistic machine learning models to integrate evidence and estimate the likelihood of true values.

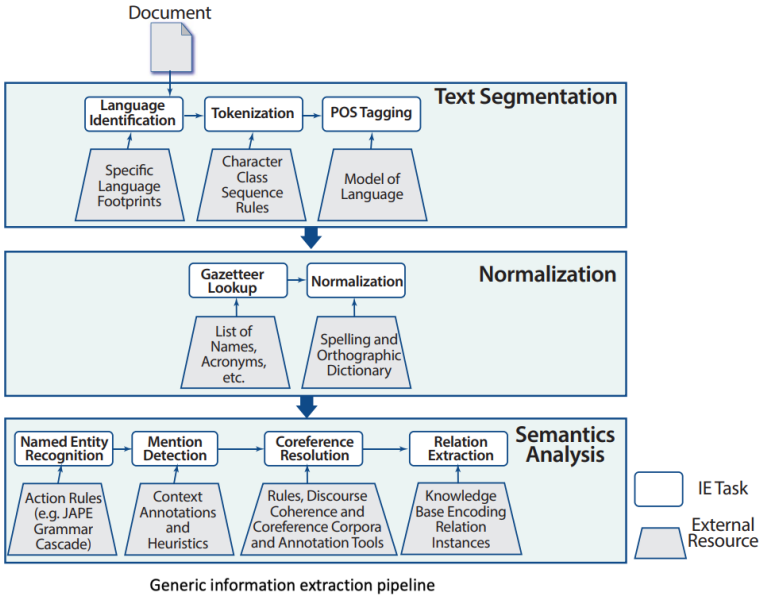
The choice of approach depends on the nature of the data and the characteristics of the sources involved in a particular scenario.



Information extraction

In any case, the first step of the pipeline is information extraction.

Starting from a document containing the desired information, the language is identified and divided into tokens, that are all processed to be transformed in a homogeneous format for comparisons, and last phases take care of understanding the content of the text under analysis.



Generic information extraction pipeline

Text segmentation is the first of the three phases of information extraction. It aims to divide a text into linguistically meaningful units. In particular, tokenization consists in identifying tokens so that matches between them occur despite differences in the character sequences of the word they contain. Techniques like stemming (reducing words to their root form.) and lemmatization (transforming words to their base or dictionary form) can be employed.

"President Obama on Sunday pressed the nation of his father's birth to root out corruption, treat women and minorities as equal citizens, and take responsibility for its own future. (July 26) AP."

Algorithm	Text
Porter [Porter, 1997]	Presid Obama on Sundai press the nation of hi father s birth to root out corrupt treat women and minor as equal citizen and take respons for it own futur Juli 26 AP
Lancaster [Paice, 1990]	presid obam on sunday press the nat of his fath ' s bir to root out corrupt , tre wom and min as eq cit , and tak respons for it own fut . (july 26) ap
Snowball [Agichtein and Gravano, 2000]	presid obama on sunday press the nation of his father ' s birth to root out corrupt , treat women and minor as equal citizen , and take respons for it own futur . (juli 26) ap

The last part of text segmentation is POS tagging, consisting in marking each word in a text with labels corresponding to the part-of-speech of the word in its grammatical context. Grammatical analysis of the text is thus performed.

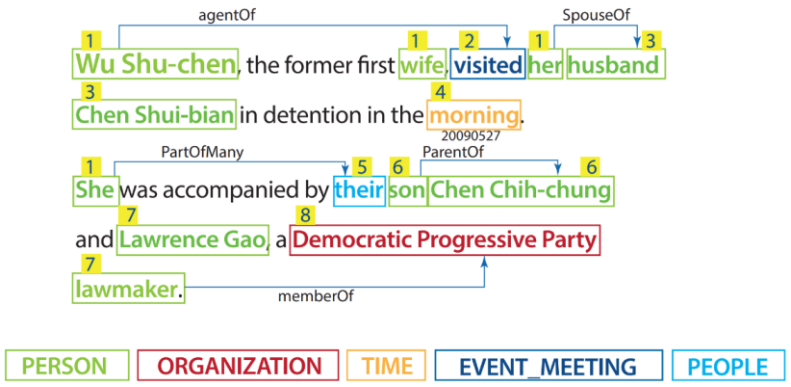
Tagged text	President/NNP Obama/NNP on/IN Sunday/NNP pressed/VBN the/DT nation/NN of/IN his/PRP$ father/NN's/-NONE- birth/JJ to/TO root/NN out/IN corruption/NN ,/, treat/VB women/NNS and/CC minorities/NNS as/RB equal/JJ citizens/NNS ,/, and/CC take/VB responsibility/NN for/IN its/PRP$ own/JJ future/JJ ./.. (/-NONE- July/NNP 26/CD)/-NONE- AP/-NONE-
with tag list:	NN (Noun, singular), NNS (Noun, plural), NNP (Proper noun, singular), NNPS (Proper noun, plural), DT (Determiner), VB (Verb, base form), VBD (Verb, past tense), VBG (Verb, gerund or present participle, IN (preposition or subordinating conjunction), JJ (adjective), CC (conjunction, e.g., "and," "or"), PRP (Personal pronoun), MD (modal auxiliary, e.g., "can," "will"), etc.

The second phase out of the three phases of information extraction is **Normalization**, that aims to reduce linguistic noise and name variance, to eliminate multiple representations of the same entity. It is comprised of two steps: identification of orthographic errors and corrections of errors and transformation of abbreviations.

Normalization creates a homogeneous situation in which the way things are written is not an obstacle for analysis, but the same syntax is present everywhere and, as a consequence, different parts can be easily compared.

The last phase of **semantic analysis** is composed by 4 phases, the two most important of which are:

- Named Entity Recognition identifies and classifies some types of information elements. Given concrete types of semantics (e.g., person, organization, localization…), the goal is to locate the elements in the text that fit the semantics. For example, in the sentence "Mark Zuckerberg is one of the founders of Facebook, a company from the United States" we can identify three types of entities: "Person": Mark Zuckerberg. "Company": Facebook. "Location": United States.
- Mention detection is defined as a language-dependent step of marking potential coreferences in a text. It is followed by coreference resolution, which links detected mentions in groups referring to the same entity.



Truth discovery

Truth discovery was born after sentiment analysis, and tackles the complementary part to opinions, that are facts.

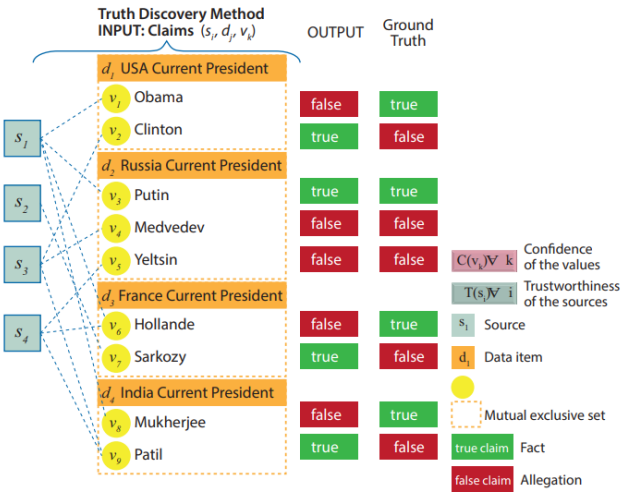
Term	Definition
Fact	*A fact is a true claim.*
Allegation	*An allegation is a false claim.*
Claim	*A claim is a value provided by an identified source.*
Value	*A value refers to the value of a property of a real-world entity.*
Object	*An object refers to a real-world entity. An object can be described by multiple properties, attributes, or features.*
Data item	*A data item is a valued property for a given object instance.*
Source	*A source is a provider of data items.*
Mutual exclusive set	*A mutually exclusive set is a set of values claimed for a given object property by multiple sources.*
Source trustworthiness	*The trustworthiness (or accuracy or reliability) of a source is a score that quantifies how reliable the source is, given the confidence of its claims.*
Value confidence	*The confidence of a value is a score that quantifies the veracity of the value, given the trustworthiness of the sources claiming it.*
Truth label	*A truth label is a Boolean value determining whether the value of a claim for a given object property is true or false.*
Ground truth	*Ground truth (sometimes called golden standard) is the set of facts (i.e., values of object properties known to be true in the real world). This set is usually manually verified/labeled and used for quality performance evaluation of the truth discovery methods.*

Some important definitions for truth discovery are those of trustworthiness and confidence.

The **trustworthiness** of a source is a measure of how reliable and credible that source is, given the confidence of its claims.

The **value confidence**, or confidence of a value, is a measure of the veracity of that value. It represents the degree of certainty or belief in the accuracy and reliability of that particular data point or measurement and indicates the level of trust one can place in the reported value.

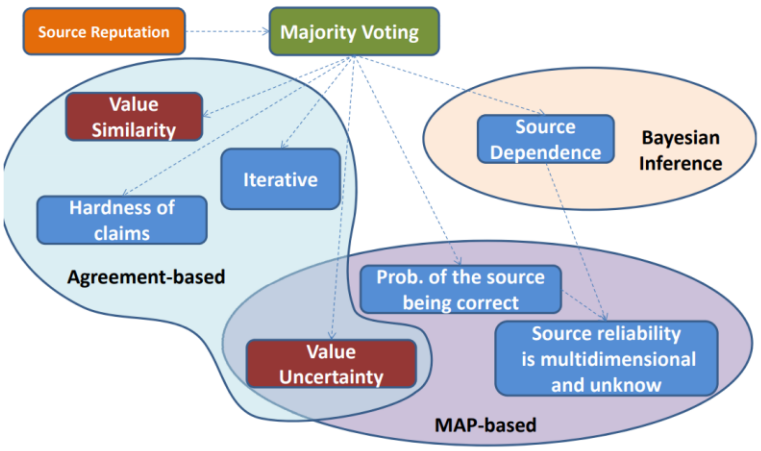
In some cases, the ground truth is easy to find, and so assigning a level of trustworthiness to a source depending on how many correct values it contains is easy, but in most cases it is not available.



		S_1	S_2	S_3	S_4	Ground Truth	Conflicts
d_1	*USA*	Obama	-	Clinton	-	Obama	2
d_2	*Russia*	Putin	-	Medvedev	Yeltsin	Putin	3
d_3	*France*	Hollande	Sarkozy	-	Hollande	Hollande	2
d_4	*India*	Mukherjee	-	Patil	Patil	Mukherjee	2
Source Coverage		1	.25	.75	.75		

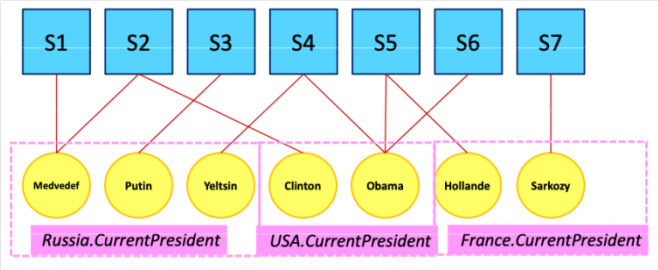
The **coverage** of a source is the extent to which a particular set of data sources comprehensively represents or covers the relevant information within a given domain or dataset. It assesses the proportion or completeness of the available data from all potential sources.

Different methods have been proposed to “discover the truth” and tell which claims from the sources are true and which are instead false.



The two main factors utilized for truth discovery computation are the source reputation and majority voting.

When **simple majority voting** it the chosen method, the true value is simply considered to be the one provided by the majority of the sources.



Of course, if the sources are not reliable, there is no guarantee that the result is correct.

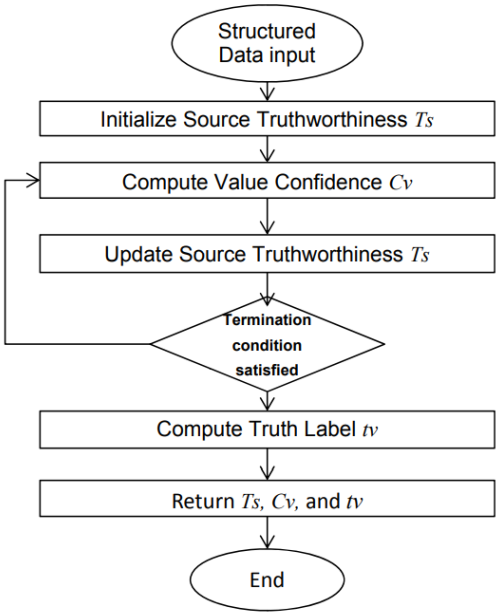
More criteria are needed to enrich the results of majority voting and have more robust results. Reputation of each source, for example, can be one of them.

Agreement based methods mainly rely on the counting of the number of agreeing/disagreeing sources for each data item.

The first group of methods refers to Web link analysis and trust metrics approaches. They generally consist in computing the relative importance of a source in the Web graph based on the probability of landing on the source node by a random surfer.

The second group of methods relies on the iterative voting algorithm which iteratively computes the source trustworthiness as a function of the confidence scores of the values it claims, and the confidence score of a value is a function of its source trustworthiness. The relationship between the trustworthiness and confidence score of a value is then considered to get to the results.

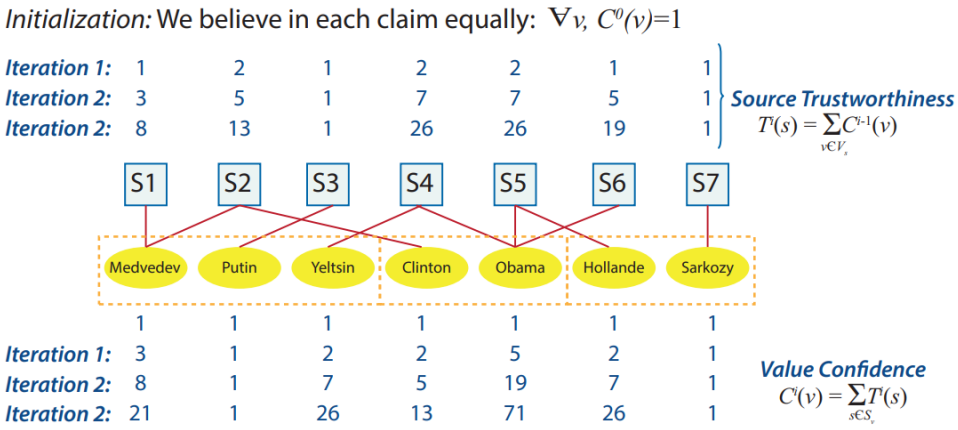
Iterative and transitive voting algorithm works in the following way:



At each iteration, value confidence for some values of a piece of information of interest is computed, then the source trustworthiness is calculated based on confidence of values. If a termination

condition is satisfied, for example if a value has a confidence way higher than other values, then a result is reached, and the iterative process stops.

In the following example, trustworthiness for each source is equally initialized, and so the confidence for each value answering to a question. At each iteration, first the trustworthiness of each source is computed simply as the sum of the confidence of all the values provided by that source. Afterwards, the confidence for each value is computed as the sum of the trustworthiness of the sources providing that value.

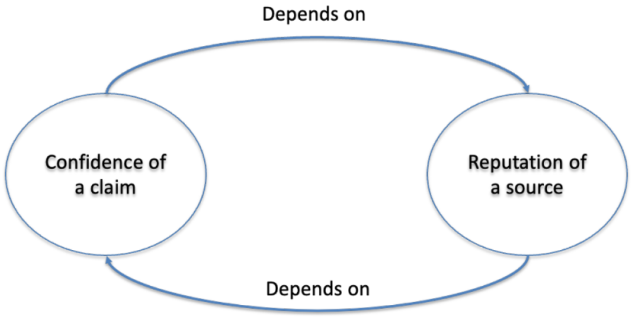


In the example then, taking source number 2, we can see it initially provides two values with confidence initialized at 1, and so at the first iteration gains a trustworthiness of $1 + 1 = 2$. The value “Medvedev” is provided by source 1 with trustworthiness 1 and source 2 with trustworthiness 2 and gets a confidence of $1 + 2 = 3$.

In the second iteration, the source 2 provides two values that now have confidence 3 and 2, and so gains a trustworthiness of $3 + 2 = 5$. The value “Medvedev” is now backed by source 1 that has now trustworthiness 3 and source 2 with trustworthiness 5 and gets a confidence of $3 + 5 = 8$.

At a certain point, a termination situation will be reached, and the iterative process will stop, decreeing the value with higher confidence as the true value, and having a produced a better understanding of the sources’ trustworthiness.

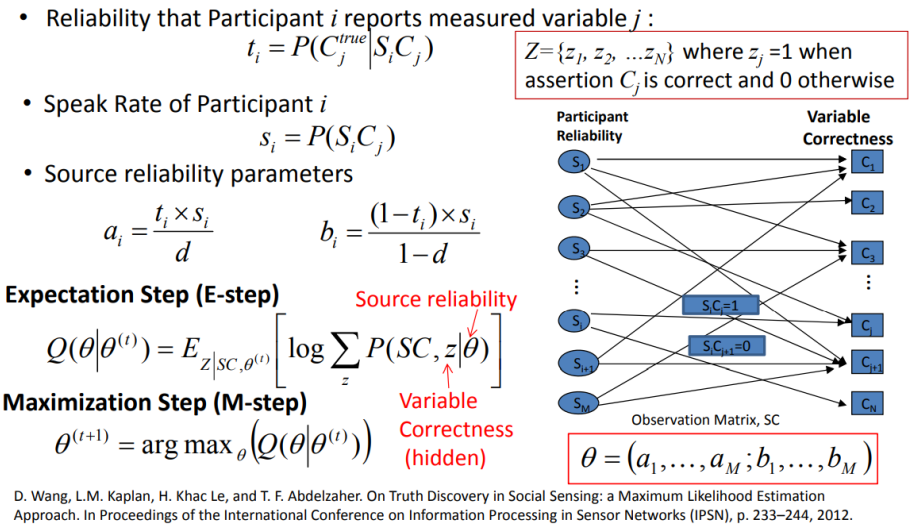
This way, the sources with higher coverage have better chances to be preferred and considered more reliable.



MAP (Maximum a posterior) estimation-based methods are different approaches that differ from the agreement-based methods mainly in the modeling of the source trustworthiness. In the

agreement-based methods, source trustworthiness is explicitly defined by a measure while MAP estimation-base capture it as a latent variable to estimate.

So, a more complex model for the trustworthiness of a source is employed. An example is the following:



D. Wang, L.M. Kaplan, H. Khac Le, and T. F. Abdelzaher. On Truth Discovery in Social Sensing: a Maximum Likelihood Estimation Approach. In Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN), p. 233–244, 2012.

Another way is to use **Bayesian inference-based methods**. One of them is **DEPEN**, that is the first Bayesian truth detection model that takes into consideration the copying relationships between sources.

Some concepts are copied from one source to another, and the assumption is that, if there are the same errors in different sources, then probably they are not independent. The underlying intuition is that sharing the same errors is unlikely if sources are independent.

DEPEN penalizes the vote count of a source if the source is detected to be a copy of another source. Sources don't have the same probability of providing a true value. So, lower reputation is assigned to sources that are detected as even partial copies of other sources, giving higher reputation to the master source.

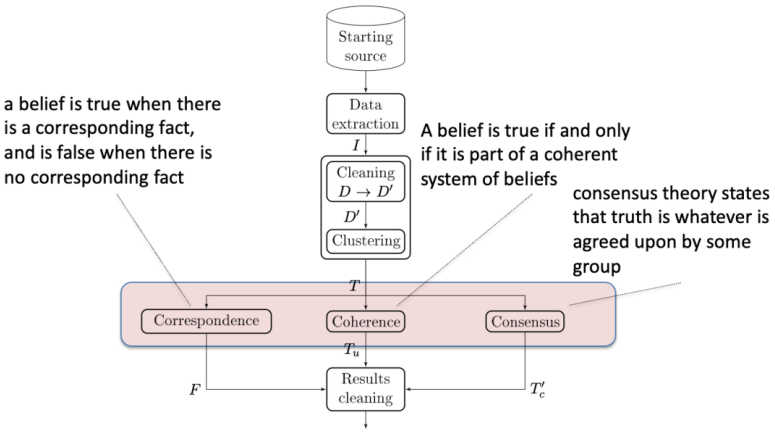
New methods are being developed because of the new challenges arising.

True values can evolve over time, what is true today may be false tomorrow and this fact needs to be taken into account by truth discovery approaches. Also, the quality of sources can change over time, especially when data streams are involved. Moreover, more sources or new claims are continuously added, and this new information must be used to modify the sources' reputation and confidence in claims with the passing of time.

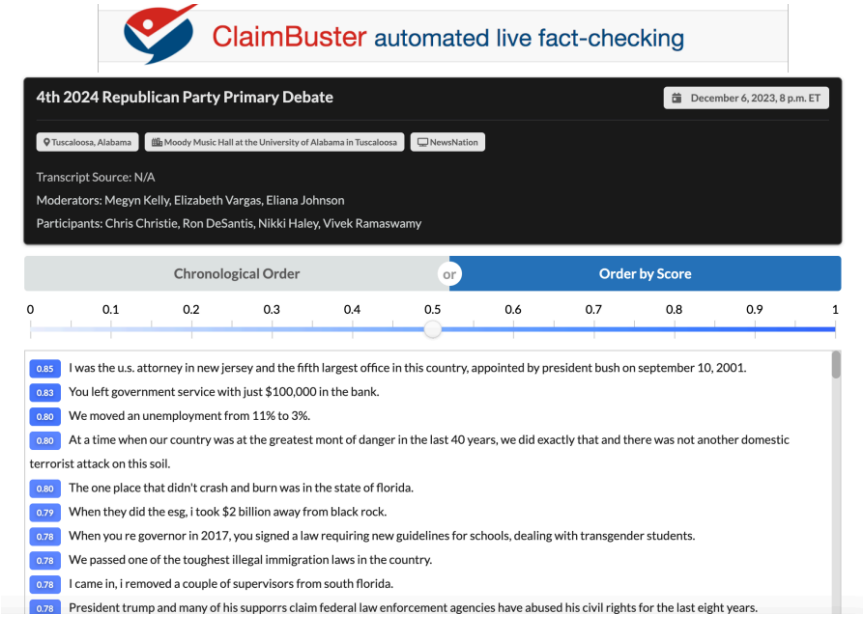
Multiple true values are sometimes possible for a question, this problem need new algorithms to be addressed.

Crowdsourcing is a resource more and more often utilized for truth discovery, but poses new risks, as quality of the output of this process is difficult to understand and depends on the accuracy of different workers.

Approaches are being developed on new criteria, for example using philosophical concept and perspectives on truth.



Fact checking to claim if something said is true or not exists in the USA from years, ad was one after political debates, without necessarily the aid of information technology. It is now performed automatically.



17. New challenges in data quality

New challenges

There are new challenges related to data quality.

Multi-modal error detection and repair

Most data cleaning solutions focus on categorical and real-valued data. We have robust methods to clean tabular sources, but what about non tabular or unstructured data? Data used in modern analytical pipelines are multi-modal and can combine structured data (e.g., categorical, and real-valued values) with unstructured data (e.g., text and images).

We cannot apply the same techniques that we use for tabular data to data that is saved in a combination of other forms. It is important that next generation data cleaning solutions are multi-modal and can operate over text, categorical, real-valued, and potentially image data. We need methods that are general enough and can adapt to the way in which data is stored.

Cleaning over Sparse Data models Like Knowledge Graphs

Databases such as graph databases are catching on and have a schema-less approach to saving data. Graphs have no regular structure, and this is an obstacle for traditional techniques based on the presence of some schema.

The increasing availability of graph-structured data sources, such as knowledge graph, opens new research directions for data cleaning. Specifically, graph-structured data captures complex and highly heterogeneous relationships between entities, making it hard to identify strong structural priors that allow for regular compute patterns and easy generalization of ML-models.

This irregularity of structure poses a significant challenge to existing ML-based data cleaning solutions since most of them are designed for data with a strict, regular schema.

One direction to address this challenge is to explore ML-based data cleaning solutions over graph-learning models such as Graph Neural Networks and Graph Embedding.

Consuming probabilistic cleaning

The data cleaning methods typically generate probabilistic predictions (i.e., predictions that are accompanied with a confidence score). For instance, they can generate a confidence score for the prediction associated with each cell in the entries of an input dataset.

To consume such probabilistic data, we need to develop new human-in-the-loop solutions that will allow users to explore the effects of these confidence scores to downstream applications in an interactive and iterative manner. Keeping humans in the loop allows to explore the confidence value associated with predictions and better understand it and the application work.

This problem is especially significant for the many ML methods that are now utilized for data cleaning.

Privacy constraints

Dealing with private and sensitive data adds another complication to data cleaning: in many cases, the underlying data contains sensitive and confidential information, and hence conducting data cleaning and machine learning on those data faces privacy constraints.

Privacy can often be an obstacle for data cleaning, because some values that may be needed won't be available. Privacy constraints can prevent from properly analyzing the data and performing subsequent operations.

To address the privacy concerns, the security and privacy community has developed various techniques that usually come at the cost of efficiency (e.g., longer running time), utility of data cleaning and machine learning tasks (e.g., lower F1-score of entity matching), or both.

One direction is to privatize the tasks, while ensuring utility and efficiency. This means that the task or computation itself is outsourced or delegated to an external entity (Cleaning as a service), that is the owner of the data itself, often without disclosing the sensitive details of the data to the organization providing the tools.

Another direction is to privatize the data, while ensuring the usefulness of the released synthetic data for downstream data cleaning and machine learning tasks. In this approach, the raw data is shared, but it is anonymized, masked, or transformed in a way that preserves privacy while still allowing certain analyses or computations.

Fair and Explainable Data Cleaning

The emerging fields of Fairness, as a new data quality dimension, and Explainable AI seek to ensure that model decisions are not directly correlated (in direct or indirect ways) to 'sensitive' features in the input data.

Datasets without biases give better results and are thus of better quality (GIGO).

These concerns can be especially relevant for data cleaning problems where data entries can be removed from an input data set due to being identified as erroneous due to their rarity or attributes can be repaired always to popular values present in the data.

We foresee engineering and mathematical challenges in learning data cleaning methods with strict fairness guarantees and developing explainable data cleaning models.

Ethical Dimensions for Data Quality

Ethical dimensions are crucial for data quality because they ensure that data is collected, processed, and used in a manner that aligns with the following principles:

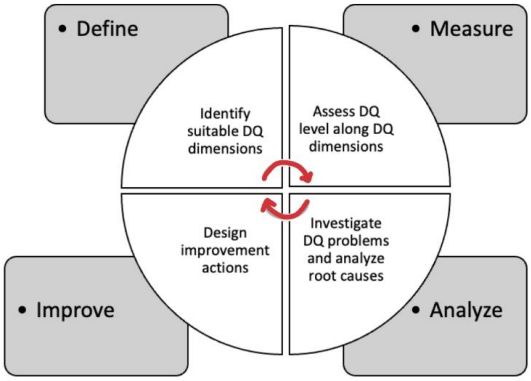
- **Fairness** of data is defined as the lack of bias.
- **Transparency** is the ability to interpret the information extraction process in order to verify which aspects of the data determine its results. In this context, transparency metrics can use the notions of data provenance by measuring the amount of meta-data describing where the original data come from and explanation by describing how a result has been obtained.
- **Diversity** is the degree to which different kinds of objects are represented in a dataset. Ensuring diversity at the beginning of the information extraction process may be useful for enforcing fairness at the end. However, using synthetic data to transform a population into a not biased one in terms of diversity decreases the overall quality of the data, generating a situation in which there is a trade-off between ensuring diversity and more accurate but also more biased results.
- **Data protection** concerns the ways to secure data, algorithms, and models against unauthorized access.

Bias is “a concentration on, or interest in one particular area, or subject”. Bias in data refers to systematic errors or prejudices that skew results, often due to a lack of representativeness. It can lead to unfair and inaccurate outcomes in analyses, perpetuating disparities and inequalities.

Metrics to understand if a dataset is biased do exist, and some of them are:

- Coverage: is the degree to which the dataset is representative of the real-world. It can be measured as the number of real-world entities represented by data compared to the total universe or population of interest. The number of real-world entities can be gathered from external sources or estimated (e.g., Elbow method).
- Density: is the degree to which different entities occur into the dataset. Density refers only to the single attribute. For each distinct value of an attribute, it is the percentage occurrence of the same value in one column. 100% density means that each distinct value appears the same number of times.
- Diversity: is the degree to which different kinds of objects are represented in a dataset. Basically, diversity is associated with the concept of entropy. The entropy of a variable, in information theory, is the average level of information contained in it.

A1. Summary schemas



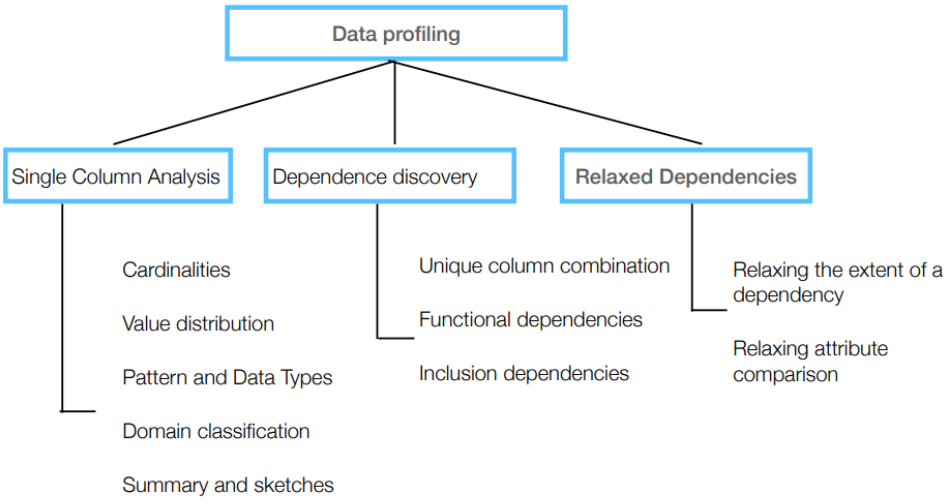
Data Quality dimensions			
Intrinsic dimensions	**Contextual Dimensions**	**Representational Dimensions**	**Accessibility Dimensions**
Believability Accuracy Objectivity Reputation	Value-added Relevance Completeness Timeliness Appropriate amount of data	Interpretability Ease of understanding Representational Consistency Concise representation	Accessibility Access security

The main data quality dimensions include:

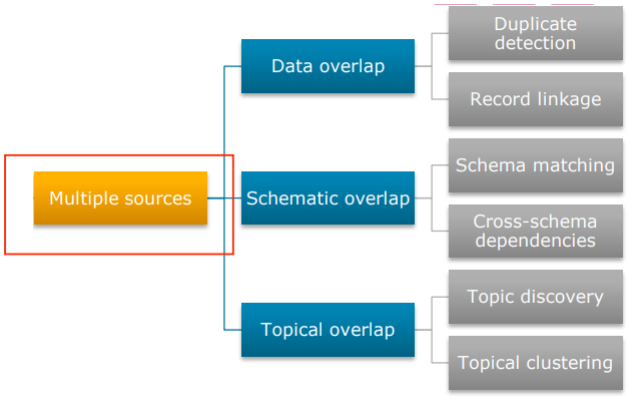
- Accuracy: the extent to which data are correct, reliable, and certified
- Completeness: the degree to which a given data collection includes the data describing the corresponding set of real-world objects
- Consistency: the satisfaction of semantic rules defined over a set of data items
- Timeliness: the extent to which data are sufficiently up to date for a task

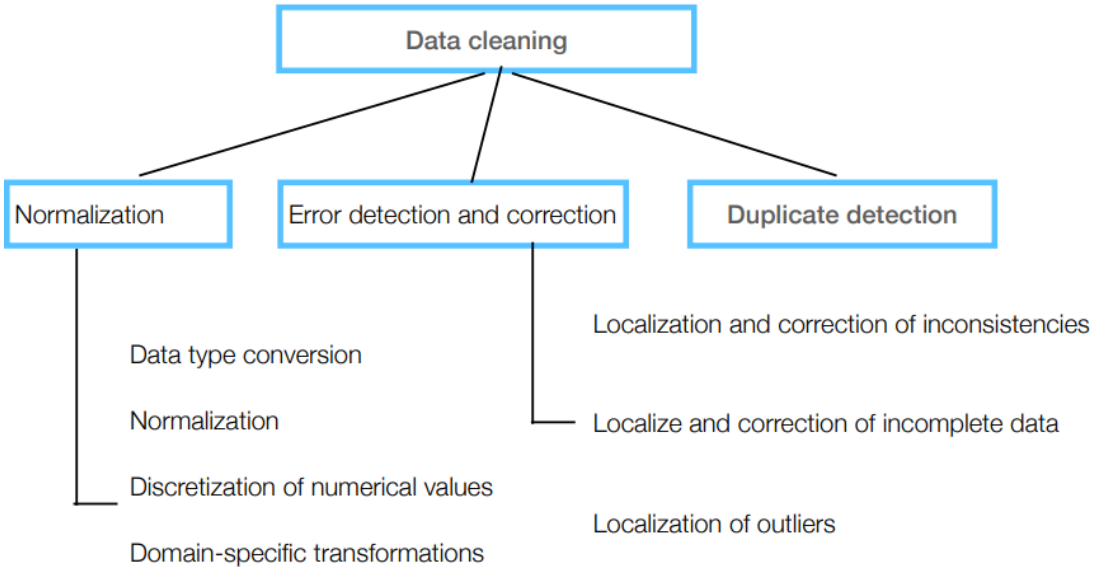
Data improvement methods are divided into:

- Data-based improvement methods
- Process-based improvement methods



Category	Task	Task Description
Cardinalities	num-rows	Number of rows
	null values	Number or percentage of null values
	distinct	Number of distinct values
	uniqueness	Number of distinct values divided by number of rows
Value Distributions	histogram	Frequency histograms (equi-width, equi-depth, etc.)
	extremes	Minimum and maximum values in a numeric column
	constancy	Frequency of most frequent value divided by number of rows
	quartiles	Three points that divide (numeric) values into four equal groups
	first digit	Distribution of first digit in numeric values; to check Benford's law
Data Types, Patterns, and Domains	basic type	Numeric, alphanumeric, date, time, etc.
	data type	DBMS-specific data type (varchar, timestamp, etc.)
	lengths	Minimum, maximum, median, and average lengths of values within a column
	size	Maximum number of digits in numeric values
	decimals	Maximum number of decimals in numeric values
	patterns	Histogram of value patterns (Aa9. . .)
	data class	Generic semantic data type, such as code, indicator, text, date/time, quantity, identier
	domain	Semantic domain, such as credit card, first name, city, phenotype

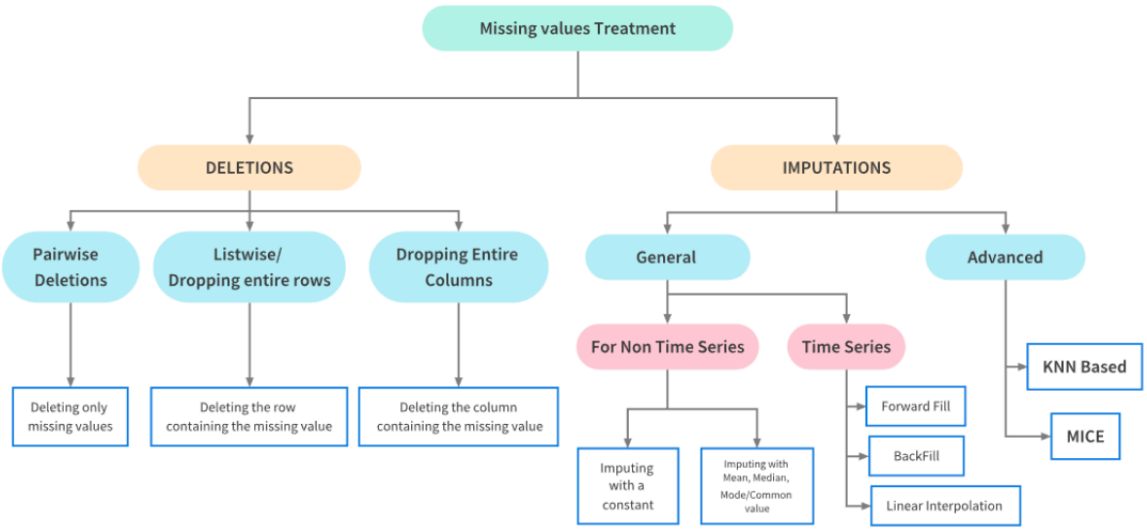




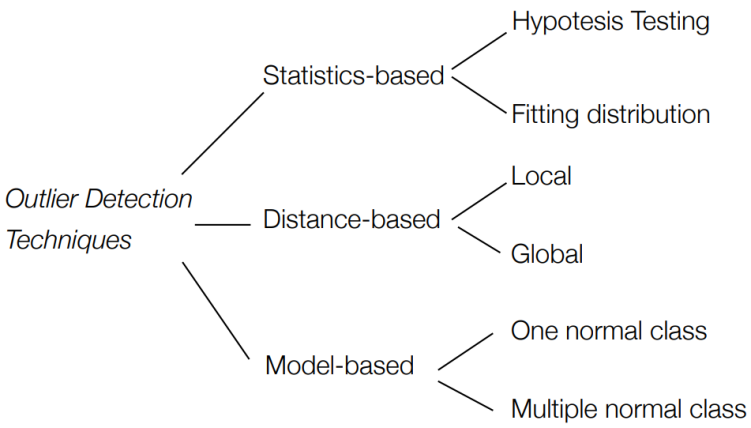
Data transformations include:

- Syntactic data transformation: E.g., transforming phone numbers to a standard format, concatenating or splitting to columns, altering the layout of a table
- Semantic data transformation: E.g., transforming acronyms into full names requires an external table that contains the full names

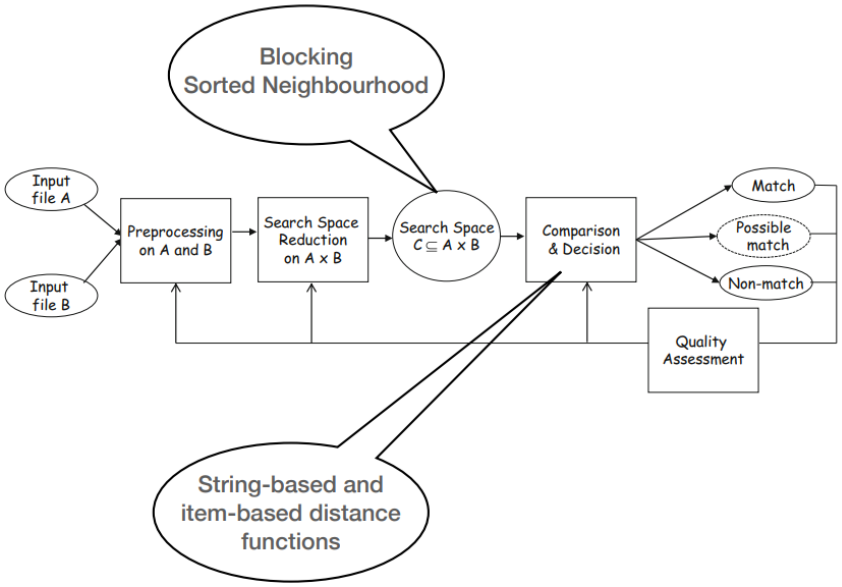
Ways to handle missing values:



Ways to handle outliers:



Duplicate detection:



Conflict handling strategies for data fusion include:

1. conflict ignorance
2. conflict avoidance
    - Instance based
    - Metadata based
3. conflict resolution
    - Instance based - deciding, mediating
    - Metadata based - deciding, mediating

Conflict resolution functions:

Function	Description	Example
Min, max, sum, count, avg…	Standard aggregation	Salary, height
Random	Random choice	Shoe size
First, last, longest, shortest	Adopt a strategy	First name
Choose source	Give a higher reputation to a source	
ChooseDepending (val, col)	Value depends on value chosen in other column	City&zip, e-mail & employer
Vote	Majority decision	rating
Coalesce	Choose first non-null value	First name
Group, concat	Group or concatenate all values	Reviews
Most recent	Most recent value	address
Most abstract or specific term	Use a taxonomy	Location
….		

ML: How to handle missing data in ML?

- Ignore records (use only records with all the values): not viable when the percentage of missing values is high or varies considerably as it can lead to insufficient and/or biased sample sizes
- Ignore attributes with missing values (use only features -attributes- with all the values): the risk is to leave out important features
- Fill in the missing values manually: feasible?
- Use a global constant to fill in the missing value: e.g. “missing” (the risk is that it can create a new class)
- Use the attribute mean (or other properties) to fill in the missing value: if the mean is unbiased
- Use the most probable value to fill in the missing value
- All the techniques come to a price. When you add a wrong value bias is added and such bias can affect the accuracy and validation of the mining results.

Basic cleaning operations include:

- Identify and remove column variables that only have a single value: Columns that have a single observation or value are probably useless for modeling. These columns or predictors are referred to zero-variance predictors as if we measured the variance (average value from the mean), it would be zero. Usually, there columns are deleted
- Identify and consider column variables with very few unique values: columns or predictors as near-zero variance predictors, as their variance is not zero, but a very small number close to zero. These columns may or may not contribute to the skill of a model. We can’t assume that they are useless to modeling.
- Identify and remove rows that contain duplicate observations. Rows that have identical data are could be useless to the modeling process, if not dangerously misleading during model evaluation. Here, a duplicate row is a row where each value in each column for that row appears in identically the same order (same column values) in another row.

Different and particular scenarios to carefully consider for data quality are:

- Data streams
- Big Data

A2. Open questions

DISCLAIMER: The answers to these example open questions are written by the author of this text based on his notes and may contain errors and/or be incomplete. No responsibility is accounted for the use that will be done of them.

From a DATA GOVERNANCE perspective define the goal of data quality and describe the DATA QUALITY ROLES that should be defined in an organization.

Data governance has the aim to build the data and organizational structures and policies to ensure the data-driven business receives the information it needs when it needs, and this information is of high quality. With high quality data, from a data governance perspective, we then mean data that is suitable for the processes in which it must be used and is accessible and interoperable because it respects a set of defined rules.

Three are the main roles that can be defined from a management perspective and have to do with data quality:

- The steering committee: composed by high-level executives and stakeholders, it defines the governance strategy and oversees the work of others to ensure the high-level processes are correctly carried out and produce the desired outputs.
- Data owners: generally members of the steering committee, they have overall accountability for the quality and accuracy of specific data domains and make decisions regarding data activities, usage, access, and policies.
- Data stewards: the people dealing everyday with data, responsible of identifying the data quality issues and resolving them (profiling, cleaning, metadata managements) by working with other data stewards and of reporting to the data owners.

Provide a definition of DATA QUALITY and of ACCURACY discussing the way in which it can be measured.

Data quality is the fitness for use of data, meaning it is the ability of data to meet user requirements and to produce a good output with respect to the user needs.

In an information system, data is considered of high quality when it coherently and correctly represents the real world, and the degree to which data accurately represents the real-world entities or events it is meant to describe is measured by an objective dimension of data quality called accuracy. Accuracy is the extent to which data are correct, reliable and certified and is defined as the closeness between a stored data value and the correct representation of the real-world phenomenon that it aims to represent.

Accuracy can refer to a single tuple, an attribute or column, a relation or a whole dataset.

To measure accuracy, the values must be compared to the actual real-world values, so either the ground truth or another set of data we consider trustworthy is needed.

Syntactic accuracy measures the closeness of a value to the elements of its domain, while semantic accuracy measures the closeness of a value to another value considered correct.

List a provide a brief definition of the most relevant DATA QUALITY DIMENSIONS.

A data quality dimension is a specific aspect or characteristic used to evaluate and measure the quality of data. These dimensions provide a framework for understanding and assessing the properties and quality of data. Dimensions can be objective and subjective, because they regard aspects that depend on the intended use of data. A dimension can thus be more or less important, depending on the way the data will be used.

Most relevant data qualities dimension include:

- Accuracy: the extent to which data is correct, reliable and certified, in terms of the closeness between the stored values and the real-word objects they represent.
- Completeness: the extent to which the table represents the corresponding real world, in terms of how many objects are represented with respect to the number of objects to represent.
- Consistency: measures how good the data is in respecting the rules, integrity constraints, data edits and business rules that are imposed on it.
- Timeliness: the extent to which data are sufficiently up-to-date for their intended use, so how valid they are in terms of time (age and volatility) to perform a task.
- Accessibility
- Redundancy
- Usefulness
- Relevance

Moreover, data quality dimensions that refer to the quality of the schema utilized to store data exist, and some of them are schema completeness, schema pertinence, schema minimality, schema readability.

SAMPLING is an important step in the data quality assessment phase. Discuss when it is required and the different sampling methods it is possible to adopt.

When a lot of data is available and needs to undergo some process, examining all of it may require too many resources and time. The solution is to use sampling methods to analyze a subset of the data and generalize the results, saving resources at the cost of introducing some uncertainty in the evaluation.

The two kinds of sampling methods are:

- Probability sampling: each unit is drawn from the population with known probability.
- Non-probability sampling: it is not possible to know the probability with which each unit is drawn from the population and so there is no way to evaluate the reliability of the results.

And the possible methods are:

- Simple random sample: random sample of a required size.
- Systematic sample: chose randomly a fist row and the others are picked at a given distance from it.
- Stratified random sample: create subgroups based on the parts of the data with different issues and take a random sample from each group.
- Cluster sample: create clusters and take a subset of them to pick random samples from

List and briefly describe the DATA CLEANING STEPS.

Data cleaning is the process, happening after data profiling, of identifying and eliminating inconsistencies, discrepancies, and errors in data in order to improve quality.

It is composed of three steps, that must be performed in the right order.

The first step is data normalization/standardization. Data is converted to homogeneous types and common formats, paying attention not to lose information in the process. Discretization of numerical values and domain-specific transformations are also included in this phase.

The second step is error localization and correction, in turn divided into three main tasks:

1. *Localization and correction of inconsistencies*: check that the data respects a set of rules and correct the values not following them.
2. *Localization and correction of incomplete data*: identify missing data and, once understood its meaning, perform imputation to replace missing data with appropriate substituted values.
3. *Localization of outliers*: study distribution, geometry, and time series of the data to identify outliers and understand their meaning to decide if analyze them more deeply or discard them.

The last phase is duplicate detection, consisting in the localization of multiple representations in the data of the same real-world object and in the consequent actions to reach a unique representation for it. Problems of search space reduction must be faced in this phase with different strategies, then the correct measures to compare values must be employed and at the end a choice about what to do with duplicated values must be made.

Explain the main analyses usually conducted in DATA PROFILING.

Data profiling is the set of activities and processes designed to take as input the data source and determine and generate the metadata describing the dataset, preparing the data for the subsequent data cleaning activities.

Data profiling can be useful for data exploration purposes, data integration, the following data cleaning, and big data analytics.

When data profiling is conducted on a *single source*, there are three main steps:

1. Single Column Analysis: analysis of the properties of single columns and attributes, studying cardinalities, data types, patterns and value distribution, domain classification and summaries of data properties.
2. Dependence discovery: analyze the consistency by searching for unique column combinations, functional dependencies, and inclusion dependencies.
3. Relaxed Dependencies discovery: as errors in data can prevent from finding dependencies, they are relaxed, allowing to find them but considering some data that do not respect them as erroneous values or considering some conditions that define when they are respected.

When *multiple sources* are involved, data profiling tools also check for topical overlap, semantic overlap, and data overlap.

List and describe the main types of DATA DEPENDENCIES.

There are different types of multi-columns dependencies.

A unique column combination (UCC) is a set of attributes within a database table where the combination of values across those columns must be unique for each record or row. Columns that are part of a UCC are candidates to compose the keys of the dataset.

A functional dependency (FD) written as $X \rightarrow A$, asserts that all pairs of records with same values in attribute combination X must also have same values in attribute A. Different tools devoted to finding them exist and are based on their properties.

Relaxed functional dependencies are functional dependencies that hold for some extent but not completely. They can be *partial dependencies*, that are FD that hold for a subset of the data and allow a certain error, *conditional dependencies*, that are FD that explicitly define conditions for them to hold, or *metric dependencies* and *matching dependencies*, that relax the comparison method to tolerate some differences, or *order dependencies*, that are dependencies existing when some ordering of values must be respected.

Inclusion dependencies are dependencies between two columns, usually from different relations, that state that all the values of a column must be contained in the set of values of the other column.

Dependencies can be found also between the schemata of different sources.

From a data cleaning perspective, discuss the goals and techniques related to the DATA TRANSFORMATION step.

Data transformation is the set of data preparation activities consisting in running user-defined programs and rules to convert data from one structure into another that is suitable for the subsequent phases of error localization and correction and duplicates detection.

Data transformation tasks can be of two types:

- Syntactic data transformation: to transform a table from one format to another, by changing the structure of the table itself and of data, without requiring external and additional domain information. Examples of operations are splitting and merging columns or rows, or deleting empty columns or mostly empty rows.
- Semantic data transformation: to transform data itself into a homogeneous and more correct format, by using domain knowledge, often from external sources. An example of operation is transforming abbreviations into full names or converting numeric values to the same unit of measurement.

From a data cleaning perspective, present the STATISTIC-BASED OUTLIER DETECTION techniques.

Statistics-based outlier detection methods are based on the assumption that normal data points would appear in high probability regions of a stochastic model, while outliers would occur in the low probability regions of a stochastic model.

A first family of methods is that of Hypothesis testing methods, that work by calculating a test statistic (e.g., the Grubbs test) for each point to determine if that point effectively is an outlier or not, and stop when the hypothesis that there are no outliers in the data is confirmed.

A second family is that of fitting distribution methods, that aim at fitting a distribution or at inferring a probability density function on the data. They in turn divide in:

- Parametric approaches: assume that data follows a normal distribution and compute a score for each data point, using robust statistics, to determine their nature.
- Non-parametric approaches: they make no assumption on the distribution of the data and try to infer the distribution with the aid of histograms or other tools, to detect outliers in the areas with less observations.

These statistic-based techniques can provide a statistical interpretation of outliers and with each detected one a score or confidence interval is computed. However, the assumptions made may not be right in all cases and even when they hold, the selection of the best statistics to employ is a difficult task.

Describe the process for detecting DUPLICATES and present two methods that can be used to evaluate the similarity.

Duplicate detection is the discovery of multiple representations of the same real-world object and is the last step of data cleaning.

The process consists of first pre-processing the input and to perform search space reduction, as it is usually too expensive to compare all records with each other's, so techniques like blocking or sorted neighborhood are employed in combination with pruning. Then, comparisons between values inside the reduced search space are carried out and decisions on whether two or more tuples match or do not match are taken. At the end, the quality is assessed again to decide if the process needs to be repeated.

To make comparisons and evaluate similarity between two objects, distance functions are utilized together with thresholds and more complex rules for discriminating matches and non-matches. Distance functions can be divided into two families:

- String-based, like the edit distance, humming distance, Jaro Winkler distance or Soundex
- Item-based, like the Jaccard distance or TF-IDF

The *edit distance* counts the minimum number of edits (substitutions, deletions, insertions) needed to transform one string into another. Similar strings are then the ones with a higher similarity score computed in function of this distance.

Soundex is instead based on the pronunciation of strings. It is an algorithm to transform strings into other 4-character long strings to be compared via substitutions.

TF-IDF measures the importance of a word in a text (under the BOW assumption) by considering its frequency of appearance and relative importance.

Let us assume that data must be analyzed with a ML application. Which additional data quality checks should be performed?

When data is used for ML tasks, the GIGO paradigm still holds. Data quality is intended in terms of fitness for use, as a measurement of how the data fits the purposes of building a machine learning system. The model quality depends on both the training and test data.

Data quality dimensions that are important to consider are:

- Comprehensiveness: how well the model could suffer the generalization issue
- Correctness: related to the label noise in training data
- Variety: how well the data covers all different cases

Rule based data cleaning, composed of error detection and correction, is often applied to improve the performance of ML models.

As ML algorithms have some requirements regarding data, some operations must be performed on it before feeding it to the model.

The first one is data conversion, in order to obtain only numerical values (through encoding of categorical one) and a homogeneous format.

Some methods require normalization of values, so to avoid having attributes with large ranges that may confuse the model.

Data should not have missing values, so imputing methods or other strategies are used to create a dataset with full completeness.

Outlier detection is another aspect, performed using dimensionality reduction techniques to face the high dimensionality of data.

Redundancy needs to be detected and eliminated.

Moreover, unbalanced datasets need to be transformed into balanced ones by collecting more data to fix them or by using undersampling or oversampling.

Which are main differences related to data quality management with respect to structured data if we are dealing with DATA STREAMS?

A data stream is a sequence of elements, each one composed by a value and a timestamp, that is produced and comes in with high velocity. When processing this data, we cannot treat it as we do for static data, with a static pipeline, but we need a process taking into account the continuous changes in the data stream.

Streaming data is often noisy and corrupted, mainly due to the limited sensors precision, and has a value that decreases with the passing of time. The main data quality dimensions for streaming data are accuracy, confidence, completeness, data volume and timeliness, while others like consistency are not considered.

Another useful dimension is *interestingness*, that measures how relevant a data point is in a stream with relation to the values near it and is used to adapt the size of the *window* utilized to measure quality of data.

Data quality improvement techniques need to be adapted for data streams, and some of them, like imputation of missing values with the last observed, have a different level of effectiveness.

Present and discuss all the types of OUTLIER DETECTION METHODS.

Outlier detection methods can be divided into three categories:

- Statistics-based
- Distance-based
- Model-based

Statistics based methods detect outliers by observing the distribution of data and detecting strange points with relation to it, by testing hypothesis by calculating statistics (like the Grubbs test) to classify the points as outliers and by fitting distributions to data with parametric and non-parametric approaches to account for non-normally distributed data.

Distance based methods calculate the distance of each point from other points and consider outliers those values that are in some way isolated. The distance can be calculated globally, between all points, or be a local distance relative to the points that are in the same neighborhood (that are near a certain point tested).

Model based methods make use of trained classifiers to detect outliers, but need a proper set of labeled data points to be trained, and this training sets are not easy to get or generate.

Describe COMPLETENESS dimension and discuss the way in which it could be assessed and improved.

Completeness describes the extent to which a table represents the real world, answering the question of whether the real world is completely represented, or something is missing in the data.

To evaluate completeness, it's important to consider what assumption between the open world and closed world assumption we are working under.

When evaluating completeness under the closed world assumption, a NULL value can be interpreted either as something that does not exist in the real world, or as a mistake or something not represented. Completeness is then the ration between the number of present (not null) values and the total number of expected values.

With the open world assumption, a NULL value simply means that something is not represented, but something that exists may also have no corresponding row in the dataset. So, completeness is evaluated as the ratio between the number of present tuples and expected ones.

To improve completeness, the easiest way is to drop the rows or columns that are incomplete. This however causes a loss of information.

Imputation is instead the process of replacing missing data with substitute data appositely generated or picked. Many imputation methods exist, from propagating values, to using statistical properties or estimators, regression, or ML techniques.

Describe CONSISTENCY dimension and discuss the way in which it could be assessed and improved.

Consistency measures how well the data satisfies a set of rules in terms of integrity constraints, data edits and business rules.

Integrity constraints can regard single attributes or involve attributes of different tables and relations together, and are divided in key dependencies, functional dependencies, and inclusion dependencies.

Data edits are rules which denote error conditions, while business rules are constraints defined by the specific business case the data is used in.

Rules can be automatically checked by tools, to estimate the consistency of data. Dependencies can be automatically discovered, and relaxed dependencies are a good instrument to find them and measure how well they are respected.

Describe TIMELINESS dimension and discuss the way in which it could be assessed and when it is important.

Timeliness is the data quality dimension related to how up-to-date the data is for a certain task. Certain data may lose value getting old and become less and less useful, especially when talking about data coming from sensors in data streams.

Timeliness has two components:

- Age or currency: measures how old the information is
- Volatility: the frequency of change of the value

Considering volatility as the average time the data is valid, timeliness is:

$$T = \max\left(0\ ;\ 1 - \frac{Currency}{Volatility}\right)$$

Lower the timeliness value, the nearer the data is to its expiration.

A3. Exercise guide

Titanic dataset

PassengerId	Survived	Class	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owen Harris	male	22	1		A/5. 21171	7.25	E — 46	S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	feale	38	1		PC. 17599	71.2833	C 85	C
3	1	3	Miss. Laina Heikkinen	female	26	0		STON/O2. 3101282	7.925	G — 56	Southampton
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1		113803	53.1	C 123	S
5	0	3	Allen, Mr. William Henry	male	444	0		373450		E 4	S
6	0	3	Moran, Mr. James	male		11		330877	8.4583	C — 67	Q
7	0	1	Mr. Timothy J McCarthy	male	54	0		17463	51.8625	E 46	S
8	0	3	Palsson, Master. Gosta Leonard	male	2	3	1	PC.349909	21.075	E — 98	S
9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27	0	2	347742		G — 09	S
10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	F	14	1		A/5. 237736		G — 07	C
11	1	3	Sandstrom, Miss. Marguerite Rut	F	4	1	1	PP 9549	16.7	G 6	Southampton
12	1	1	Miss. Elizabeth Bonnell	female		0		STON/O2. 113783	26.55	C 103	S
13	0	3	Saundercock, Mr. William Henry	mle	20	0		A/5. 2151		E — 89	S
14	0	3	Mr. Anders Johan Andersson	mae	39	1	5	347082	31.275	E 6	S
15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female		0		350406	7.8542	C 15	Southampton
16	0	3	Vestrom, Miss. Hulda Amanda Adolfina	F		0		A/5. 350406	7.8542	C —15	Southampton
17	0	3	Rice, Master. Eugene	M	2	4	1	382652		C 12	Queenstown
18	1	2	Williams, Mr. Charles Eugene	M		0		STON/O2. 244373		E 44	S
19	0	3	Vander Planke, Mrs. Julius (Emelia Maria Vandemoortele)	female	999	12		345763	18	C — 2	S
20	1	3	Mrs. Fatima Masselmani	femle	0	0		2649	7.225	C 7	Cherbourg

1) Detect and list all the error types contained in the dataset.

The possible errors that can be found are:

- **Typos**: errors in the spelling of words
- **Null values**
- **Data formats**: the same attribute is represented in different ways (e.g. the order of name and surname, the format of the date, prefixes, abbreviations, etc.)
- **Outliers**
- **Duplicates**
- **Inconsistencies** with relation to integrity constraints

2) List the data preparation activities that should be performed to solve those errors, and specify:

a. the order in which the data preparation activities should be executed
b. describe the method that you would apply for each data preparation activity
c. which libraries or functions should be used in python to perform each data preparation activity.

Being specific about what to do with the provided DataFrame and listing the attributes with their issues and what to do with them specifically, the structure should be the following:

1) Transformation/standardization
    a. Typos correction
        i. dataframe.loc(dataframe[attribute]==condition)=new_value
    b. Abbreviations correction
        i. dataframe.loc(dataframe[attribute]==condition)=new_value
    c. Change data types
        i. dataframe["col_name"] = dataframe["col_name "].astype("new_type")

   d. Tokenization and splitting/merge
      i. Dataframe[[new_1, new_2]]=Dataframe.column_name.str.split("separator", n=num_of_splits, expand=true)
2) Error detection and correction
   a. Integrity constraints and dependencies
      i. dataframe.loc() for corrections
      ii. Rules() to detect dependencies
   b. Missing values
      i. Dataframe.fillna(method='ffill') to forward the values
      ii. Dataframe.fillna(method='bfill') to forward back the values
      iii. Dataframe.fillna("value")
      iv. Dataframe.fillna(dataframe.col_name.mean())
   c. Outliers
      i. Using ground rules with dataframe.loc()
      ii. Using grubbs.test(dataframe, alpha=threshold)
      iii. Using z-score with outliers-utils library
      iv. Plotting histograms
3) Duplicate detection and data fusion
   a. Detection
      i. Dataframe.drop_duplicates() for exact duplicates
      ii. Recordlinkage library for non-exact duplicates
         i. indexer = recordlinkage.index.SortedNeighbourhood(on='col_name, window=9)
         ii. compare = recordlinkage.Compare()
         iii. Define rules with compare.exact() or compare.string()
         iv. features = compare.compute()
         v. Then filter the features with a desired score to have the list of duplicated values
   b. Dropping or fusion
      i. User defined functions: drop, or take one value, or take the mean, …

Other useful functions:

- Dataframe.drop()
- Soundex().distance()
- TfidfVectorizer()

3) List and briefly describe at least 2 methods you would perform to discover if this dataset contains non-exact duplicated rows, and list the libraries and functions you would use in python.

A first possible method is to use blocking to reduce the search space. The data is divided into blocks and the comparisons between values, based on different possible distance measures (edit distance, hummin, soundex, jaro-Wiinker, TF-IDF, Jaccard), are executed block by block. There is a trade-off between the sizes of blocks and the accuracy of the algorithm.

Another possible method is to use sorted neighborhood, consisting of first ordering the dataset according to some attribute or created key, and then sliding a window on it, performing comparisons inside this window. Also with sorted neighborhood, there is a trade-off between the size of the

window and the accuracy, but the overall performances should be better than in the case of blocking, because sorted data should have similar values near and inside the same window.

Multi-pass sorted neighborhood algorithms perform the sorted neighborhood method multiple times with smaller windows but changing every time the way the data is ordered, generally obtaining better results than the standard method.

To perform these operations, the *recordlinkage* library can be utilized.

To create an indexer:
For blocking: indexer = recordlinkage.index.block()
For sorted neighborhood: indexer = recordlinkage.index.SortedNeighbourhood

To find candidate links:
candidate_links = indexer.index()

To define ways to compare values:
compare = recordlinkage.Compare()
compare.exact() or compare.string() or compare.numeric()

To compute features of each candidate couple:
features = compare.compute()

Then features are filtered according to user defined critereia to decide which one are matches

3) List the data profiling tasks and the libraries or functions you would use in python to execute them

Data profiling tasks include:

- Single column analysis
  - Cardinalities:
    - DATA.columns
    - DATA.describe()
    - DATA.dtypes
    - DATA.duplicated()
    - DATA.corr()
  - Value distributions:
    - DATA.hist()
  - Panda.ProfileReport()
- Dependencies discovery
  - Association rules
    - Apriori()
    - Association_rules()
    - ECLAT()
  - Functional dependencies
    - Tane.compute()
    - Fdtool.main()
- Relaxed dependencies
  - Relaxed functional dependencies
    - Ctane.compute()

A4. Multiple choices

DISCLAIMER: The answers to these example multiple choices questions are written by the author of this text based on his notes and may contain errors and/or be incomplete. No responsibility is accounted for the use that will be done of them.

Which of the following sentences about DQ dimensions are true?

- *DQ dimensions can refer either to the data values or to the data schema* – True, DQ dimensions for the single values include accuracy, consistency, completeness, timeliness, while schema dimensions include schema accuracy, schema completeness, schema pertinence, and others.
- *Each DQ dimension is always associated with only one specific metric* – False, different metrics can be used, some are more suitable for the specific use case and others are not.
- *Some DQ dimensions cannot be measured in an objective way* – True, some DQ dimensions are subjective, and their measurement depends on user defined thresholds and on the intended use of data.
- *Each DQ dimension is defined to assess data suitability from a different perspective* – True, that is the reason there exist a lot of DQ dimensions and news are defined.

Sampling in data quality assessment

- *It is always needed* – False, when data are not too much, it's better to avoid doing sampling and working directly on the entirety of data.
- *It is needed when it is not possible to analyze all the records or the entire dataset* – True, it is not always possible or convenient to work on all the data.
- *It Is usually performed by using nonprobability sampling methods* – False, the probability method is the most used and allows to know the probability with which each value is extracted, allowing better control and precision over the sampling.
- *It can be performed adopting a systematic sample approach* – True, systematic sample is a variation of the simple random sample, in which one starting value is initially extracted randomly and others are extracted from positions depending on the position of this first value.

Data profiling

- *Gathers characteristics of the data source* – True, the data source is analyzed along with the data values.
- *Generates metadata describing a dataset* – True, data characteristics are saved in form of metadata, to be used in subsequent phases.
- *Detects and corrects errors* – False, errors are corrected in later phases, data profiling is only about generating metadata describing the dataset.
- *Provides input data for data cleaning* – True, data profiling prepares the data for subsequent cleaning, integration and analytics.

Relaxed data dependencies

- *May hold on some subset of tuples* – True, partial dependencies are relaxed data dependencies that hold only for a subset of data.
- *Are useful if data are expected to contain errors* – True, data containing errors may prevent from finding functional dependencies, while relaxed dependencies may indicate that there is a functional dependency not detected because of some errors.
- *Are, in general, not considered in data profiling* – False, dependencies discovery is a phase of data profiling.
- *Might require considering the similarity between attributes* – True, one way to relax the dependencies is to relax the corresponding attribute comparison.

In a data preparation pipeline, data transformation

- *Is part of data profiling* – False, data transformation is part of normalization/standardization phase of data cleaning, while data profiling happens before the data cleaning starts.
- *Is part of the data cleaning* – True, is in fact part of its first phase of normalization/standardization.
- *Is mainly focused on convert data in the right format* – True, but not limited to it: syntactic data transformations change the structure of data and tables, while semantic data transformations transform data by using external knowledge.
- *Should be performed between error detection and duplicate detection* – False, it must be performed as the first phase of data cleaning, so before both error detection and duplicate detection.

Which of the following sentences about outlier detection are true?

- *Outlier detection is a data profiling task* – False, outlier detection is part of the data cleaning process, in particular part of the error correction phase. It's true that outliers can be spotted from histograms produced during data profiling, but the ad-hoc task of outlier detection is carried out later during data cleaning, complementing histograms with other techniques.
- *Statistics-based outlier detection methods often assume that data are normally distributed* – True, but it is not always the case. Statistics based parametric approaches assume that data follows a normal distribution, while non-parametric approaches infer the distribution from the data.
- *Model-based detection methods define a normal behavior on the basis of the distance between data points* – False, that is what is done by distance-based approaches, that work by defining a distance between data points, which is used to define a normal behavior.
- *Histograms can be an effective tool for detecting outliers* – True, they are especially useful in statistics-based non-parametric approaches, in which equi-width histograms show potential outliers in bins with low frequency.

Record linkage and duplicate detection are mostly based on the same techniques – True, both can use the same similarities and approaches.

Which of the following are true in the context of data fusion?

- *Data fusion is a synonym for data integration* – False, we can say that data fusion is a step in the data integration process, but only a part of it.
- *Data fusion aims to solve conflicts and contradictions* – True, Data fusion has the aim to resolve uncertainties and contradictions, and works by, given a duplicate, creating a single object representation while resolving conflicting data values.
- *Selecting a random value between two conflicting values is a data fusion resolution function* – True, it is one of the possible conflict resolution strategies, along with taking the average of the two values, taking the most updated one, taking the one occurring more often, and others.
- *Data fusion is performed before duplicate detection* – False, Data fusion requires that duplicate detection is performed before, to know the matching tuples to merge.

Process-based data improvement techniques

- *In the long term, they are more effective than data-based improvement techniques* – True, because they aim to solve the problem generating the data quality issues under analysis, so to avoid that in the future they are generated again.
- *In the long term, they are less effective than data-based improvement techniques* – False, data-based improvement techniques only correct the present data, leaving the possibility that the data generated in the future will present the same issue, while process-based improvement techniques try to avoid this scenario.
- *Once detected an error, they aim to correct it* – True, if we are referring to an error inside the process generating the data, and we mean correcting the process or activity generating the data error.
- *Require modelling the processes in which data are used* – True, this is done with tools like BPMN or IPMAP to examine the flow of data inside the processes and understand what needs correcting.

Interestingness in data streams increases for ranges in which there are low fluctuations – False, it's the contrary: interestingness increases when there are behaviors in the data that go outside the normal trend. This fact is utilized to resize the window in data streams, making it smaller when data are "interesting" and bigger when nothing "of interest" seems to happen.

Which of the following are causes for poor data quality?

- *Historical data* – False, the fact itself that the data is historical is not a cause for poor data quality, but we must nevertheless consider that the importance of data may change over time.
- *Manual data entry* – True, manual inputting of data may be done erroneously.
- *Data integration* – True, it's easy that data integration is not completely correctly performed.
- *Data provenance* – False, data provenance is a record of the history of the life cycle of data.

The main causes for poor data quality include: errors introduced by the manual inputting of data into the systems, historical changes in the importance of pieces of data, data usage (as data relevance depends on the process in which data are used), corporate mergers in which difficulties arise during data integration, data enrichment (as external sources might poison the internal data to which they are added).

Which of the following sentences about Data Governance are true?

- *Data Governance is important for a data-driven company* – True, it's one of the foundations of a data driven company, along with "data at the center", data culture, and analytics.
- *Data Governance is just an organizational discipline* – False, it is also a technical discipline. It is the practice of organizing and implementing policies, procedures and standards that maximize data access and interoperability for the business mission, and works defining roles, responsibilities, and processes for ensuring accountability for and ownership of data assets.
- *Data Governance includes security* – True, the components of data governance are: master data management, data quality, security, metadata management, integration.
- *Minimize risks is one of the goal of data governance* – True, the objective of data governance is to establish the methods, set of responsibilities, and processes to standardize, integrate, protect, and store corporate data, and to do so one of the goals is minimizing risks.

Which of the following sentences about accuracy are true?

- *Accuracy assessment is related to the satisfaction of business rules* – False, that is a matter relevant for the consistency dimension.
- *Accuracy is a synonym for correctness* – True, Accuracy is the extent to which data are correct, reliable and certified.
- *A value of an attribute is syntactically accurate if it is included in the domain of the attribute* – True, syntactic accuracy is the closeness of a value to the elements of the corresponding definition domain.
- *Semantic accuracy is easier to assess than syntactic accuracy* – False, additional knowledge is often needed to measure semantic accuracy, which is the closeness between a data value and the value it should represent, and needs the solution of the object identification problem in order to be evaluated.

Which of the following sentences about Sampling are true?

- *Sampling should be used when analyzing all the records is not feasible* – True, it is not always feasible to perform a census of the database, and sampling is a technique that allows to run some analyses on the data at the cost of introducing some uncertainty. When analyzing all the records is possible in terms of resources, it should be preferred to sampling.
- *Cluster sampling is a non-probability sampling method* – False, cluster sampling is a probability sampling method. In cluster sampling, the population is divided into clusters, and then a random sample of clusters is selected.
- *The size of the sample does not affect the desired precision* – False, the size of the sample depends on the precision and reliability level required by the analysis, and vice versa.
- *Systematic sampling requires to divide the population in groups* – False, systematic sampling consists in selecting randomly a first row, and then pick the others based on a function of the position of this first row.

Which of the following sentences about Data Profiling are true?

- *It is part of the data cleaning process* – False, it is performed before data cleaning.
- *Analyzing value distributions is part of the single column analysis* – True, single column analysis includes the analysis of cardinalities, value distribution, pattern and data types, domain classification, the generation of summaries and sketches.
- *It evaluates completeness* – True, completeness is one of the aspects that can be evaluated during data profiling.
- *It does not include dependency discovery* – False, dependency discovery is part of data profiling, and mainly involves finding unique column combinations, functional dependencies and inclusion dependencies.

Which of the following sentences about Dependency discovery are true?

- *Unique column combination is a set of attributes of a relation X whose values are contained also in a relation Y* – False, a unique column combination (UCC) is a set of attributes that has no duplicates in the other rows, and thus can be used a s a key in the database. Instead, a set of attributes of a relation X, whose values are contained also in a relation Y, is an inclusion dependency.
- *Functional dependencies are important for evaluating consistency* – True, functional dependencies are a type of the integrity constrains (along with key dependencies and inclusion dependencies) that are evaluated to check for the consistency of data.
- *Partial dependencies are associated with conditions that restrict their scope* – False, partial dependencies are rules that hold only for a subset of the data, but not necessarily there is condition restricting their scope. Partial dependencies that explicitly specify conditions to restrict their scope are conditional dependencies and are only a subset of partial dependencies.
- *Metric dependencies are relaxed dependencies* – True, they are relaxed dependencies that relax the comparison method in a way that tolerates formatting differences.

Statistics-based outlier detection methods need labeled training data – False, Statistical techniques do not need labeled training data, but infer if a value is an outlier from the analysis of probability regions of stochastic models.

Which of the following sentences in the context of duplicate detection are true?

- *It is possible to use blocking techniques for the search space reduction phase* – True, when blocking techniques are employed, the file is partitioned in exclusive blocks according to a defined key and comparisons are limited to records within the same block.
- *Jaccard distance can be used to evaluate similarities between sentences* – True, Jaccard distance is based on Jaccard similarity, which compares two sentences (bag of words) by considering the number of words that appear in both of them.
- *At the end of the process, it classifies the tuples in two categories: match and non-match* – False, a third category is possible, that is the “possible match” and usually requires additional work or the intervention of a human to resolve the conflict.
- *The sorted neighborhood approach uses sampling techniques for the search space reduction phase* – False, the sorted neighborhood approach consists of sorting a file according to a key and then moving a window of a fixed size on the file, comparing only records within the window.

Which of the following sentences in the context of big data are true?

- *Variety is managed by using the MapReduce approach* – False, MapReduce algorithms are useful to deal with the Volume of Big Data.
- *In big data timeliness is usually very short* – True, this is because of the Velocity of Big Data, meaning also that data becomes outdated and invalid quickly.
- *Sampling is often used for data profiling and assessment* – True, but sampling must be utilized carefully.
- *Most of the data sources are data streams* – True, most Big Data sources are data streams, in which the Velocity of generation and required processing of data is high.

Data provenance is a value that indicates the author of a data source – False, Data lineage is the information about who created a source and why, but data provenance includes a lot more aspects, representing the historical life cycle of a source or of some data, including who modified it, how it was modified, where it comes from, and other stuff.

Which of the following sentences about syntactic accuracy are true?

- *It is the closeness of a value v to the elements of the corresponding domain D* – True, it is the definition of syntactic accuracy, that requires checking whether v is any one of the values in D, so if the value belongs to the domain.
- *It can be assessed only by using questionnaires* – False, exact matching and similarity-based approaches can be used to check the distance between a value and the values of a domain.
- *It can be assessed by using similarity functions* – True, similarity-based approaches make use of similarity functions to give a syntactic accuracy score between 0 and 1 to a value according to its distance to the values in the domain of interest.
- *Its assessment always requires external sources* – False, knowledge of the domain is needed, but not necessarily external sources need to be involved, like it happens in most cases when evaluating semantic accuracy.

Which of the following sentences about accessibility are true?

- *Accessibility is referred to the presence of large amounts of data in the web* – False, data can be present and accessible also elsewhere.
- *Data are accessible if the user is able to access them and use them* – True, accessibility is the ability of the user to access the data.
- *It is not a proper data quality dimension* – False, it is a data quality dimension, also present in the famous classification for DQ dimensions presented by Wang and Strong in 1966.
- *It measures the ability of the user to access the data on the basis of his/her context* – True, it takes into account the ability of the user to access the data from his or her own culture, physical status/functions, and technologies available.

Which of the following sentences about sampling are true?

- *Nonprobability methods are the ones used in most of the studies* – False, the most used approach is probability sampling, in which each unit is drawn from the population with known probability.
- *Simple random is a probability method* – True, it consists in the definition of a random sample of the size required.
- *Degree of precision is the amount of error* – True, it is the amount of tolerated error and affects the size of the sample, together with the reliability level.
- *In Cluster sample values in cluster should be characterized by a uniform distribution* – False, the values within each selected cluster may exhibit various distributions.

Which of the following sentences about Data profiling are true?

- *Data profiling creates metadata* – True, data profiling is the set of activities and processes designed to take as input the data source and determine and generate the metadata describing the dataset.
- *Data profiling can support data exploration* – True, especially in the cases when datasets arrive at an organization and/or accumulate in data lakes, and experts need a basic understanding of their content, data profiling techniques support data exploration. Other use cases in which data profiling is useful include data integration, data cleaning and big data analytics.
- *Data profiling detects truncation* – True, by analyzing the distribution of data lengths, patterns, and potential anomalies, data profiling tools and techniques can help identify truncation issues and contribute to improving data quality. (data truncation occurs when data or a data stream is stored in a location too short to hold its entire length, or data are collected leaving intentionally out some categories)
- *Data profiling is able to identify the semantic domain of a column* – True, data profiling can identify the domain (e.g., credit card, first name, city, etc.)

Which of the following sentences about Outlier detection are true?

- *Statistics-based outlier detection methods are not suitable for high dimensional datasets* – True, the assumption of an underlying distribution does not hold for high dimensional real datasets.
- *Distance-based methods can be used only with numerical data* – False, different distance measures can be utilized to compare values and compute distances.
- *Model based techniques have a fast testing phase* – True, testing phase, validation and application of these models are generally fast.
- *Parametric approaches assume the distribution that generates data* – True, the assumption of parametric approaches is that data follows a normal distribution.

Majority voting in data fusion is a rule-based technique – True, Voting is a conflict resolution technique. (Majority voting is also a method utilized by truth discovery)

Which of the following sentences about Functional dependencies (X->Y) are true?

- *In non-trivial FD attributes on Y and X are disjoint* – False, non-trivial FDs are FDs X->Y in which at least one attribute on Y does not appear on X (e.g., Street, City → Zip, City). Instead, when attributes on Y and X are disjoint, we are talking about completely non-trivial FDs (e.g., Street, City → Zip).
- *A FD asserts that all pairs of records with same values in attribute combination X must also have same values in attribute Y* – True, this is the definition of functional dependency.
- *The goal of dependency discovery is to find all minimal and completely non-trivial FDs* – True, this is the typical goal of dependency discovery, but other kinds of FDs can be searched for.
- *Association rules are a data mining tool to find FDs* – True, association rules can be exploited to find the values that occur most often inside the transactions together, to ultimately find functional dependencies.

Fellegi and Sunter theory need labeled data – True, the Fellegi and Sunter theory consists in a supervised technique that needs a training dataset for labelling records pairs as matching or non-matching.

Which of the following sentences in the context of similarity measures are true?

- *Jaro-Winkler method considers on the number of common characters* – True, the Jaro-Winkler string comparator counts the number *c* of common characters between two strings and the number of transpositions that are the number of pairs of common characters that are out of order.
- *Jaccard distance can be used to evaluate similarities between texts* – True, Jaccard distance measures the distance between two sets of words, so also texts.
- *To compare two tuples, it is necessary to use the same similarity method for all the attributes* – False, more different distance functions should be applied together to different attributes according to their format, and also more similarity methods can be applied on the same attribute to have a better overall score.
- *The sorted neighborhood approach suggests using the Levenshtein distance* – False, there is not a preferred distance to use, but the combination of distances to use should be chosen according to the use case.

Which of the following sentences in the context of data streams are true?

- *Accuracy, consistency and completeness are still relevant for data streams* – False, while accuracy and completeness are still relevant, consistency is generally not considered, because it's rare to have dependencies between the values in a stream from sensors. However, it's still possible to measure consistency for data streams.
- *Data quality evolves over time* – True, data can be more or less interesting, and be more or less subject to environmental factors, and these characteristics evolve with time. Moreover, the value of data gets lower as data becomes older.
- *Optimistic approach relies of the precision of the sensors* – True, it relies on sensors with high precision and assumes that the arising errors are so small to be negligible in the considered context.
- *Social networks generate data streams* – True, they generate data with a high velocity and that gets quicky outdated.