Big Data and Advanced Analytics

Improving Data Quality For Big Data Using Advanced Analytics

Aytac Ozkan

Supervisor: Prof.Rachid Chelouah

Department of Engineering
Ecole internationale des sciences du traitement de l'information

This dissertation is submitted for the degree of
Master of Big Data

EISTI August 2019

Acknowledgements

I owe ***Mr. PICHOT Christian*** a debt of gratitude for his kind assistance and his endorsement through the preparation of this scientific paper. In the presence of yours, I would like to thank him. Also, I am appreciated working and collaborating with [36] PACA. Obviously without them it would hard to complete this study.

Abstract

Digital data play a crucial role in the information and communication technology (ICT) society: they are managed by business and governmental applications, by all kind of applications on the Web, and are fundamental in all relationships between governments, business, and citizens.

Data quality can be defined as "the measure of the agreement between the data views presented by an information system and that same data in the real world". Data quality is a multidimensional construct consisting of different data quality dimensions such as accuracy, completeness, consistency, and currency.

Furthermore, quality of data is also a significant usage for operational process of business and organizations. Some disasters are due to the presence of data quality problems, among them the use of inaccurate, incomplete, out-of-data.

As a consequence, the overall quality of the information that flows between information systems may rapidly degrade over time if both process and their inputs are not themselves subject to quality control.

On the other hand, the same networked information system offers new opportunities for data quality management, including possibility of selecting sources with better quality data, and of comparing sources for the purpose of error localization and correction, thus facilitating the control and improvement of data quality in the system.

Due to the described above motivations, researchers and organizations more and more need to understand and solve data quality problems, and thus answering the following questions: What is in essence, data quality? Which techniques, methodologies, and data quality issues are at a consolidated stage?

In this paper, we first review relevant works and discuss machine learning techniques, tools and statistical models. Second, we offer a creative data profiling framework based deep learning and statistical model algorithms for improving data quality.

Keywords: Deep Learning, Statistical Quality Control, Machine Learning, Data Cleaning

Table of contents

List of figures

List of tables

Listings

Chapter 1

Introduction to Data Quality

A Web search of terms "data quality" through the search engine Google, returns about three millions of pages and indicator that data quality issues are real and increasingly important (the term data quality will be shortened to the acronym DQ)

1.1 Data Quality: Wrangling and the Big Data Life Cycle

When talking about data quality in this document, it is done referring to the degree in which data fits to serve for its aimed purpose [73], for example, how well a medical record allows a nurse to identify the medicine that should be given to a patient, where “well” comprises, among general qualities, how accurate, complete, up to date, valid and consistent [3] is the information so that the task can be successfully achieved. There are several processes required to asses and improve data quality, which include data extraction, data profiling, data cleansing and data integration [38], as the major ones; altogether in “the process by which the data required by an application is identified, extracted, cleaned and integrated, to yield a data set that is suitable for exploration and analysis” [62] is known as Data Wrangling. Figure 1.1 shows how each process is placed within the big data life cycle [1] [38], where data profiling is the step in which this research contributes to.

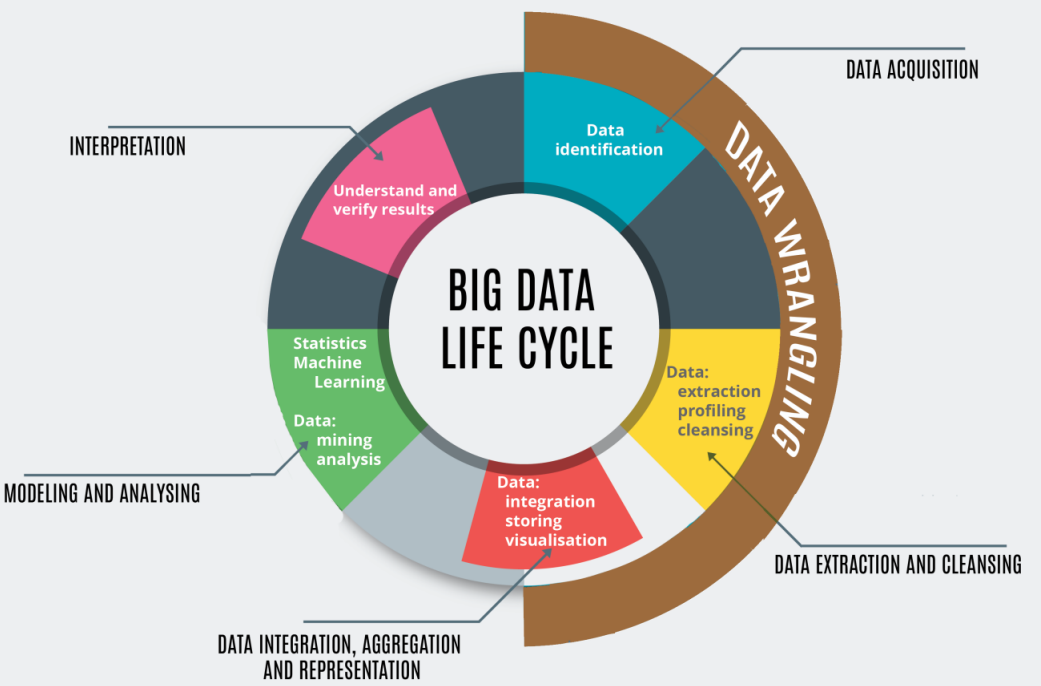

Fig. 1.1 The big data life cycle

Table 1.1 Big data life cycle processes definitions I.

Major Process	**Process included**	**Description**
DATA ACQUISITION	Data identification	Data has to be first generated from the real world, then converted to electrical signals so it can then be recorded in a machine. This process is called data acquisition [1]. Identifying data means to provide useful metadata about its provenance, intended use, recording place and motivation, etc. [17] [57].
DATA EXTRACTION AND CLEANSING	Data extraction	This is the process in which data from source systems is selected and transformed into a suitable type of data according to its purpose, e.g. coordinates from a set of stored satellite images [1].
	Data profiling	This refers to generating convenient metadata to support measurements against quality settings previously established, and to contribute towards "well known data", clearing up the structure, content and/or relationships among the data. E.g. data types,data domains, timeliness, completeness, statistics, etc. [76] [70].
	Data cleansing	Also known as cleaning or scrubbing. Requires solving errors found in invalid, inconsistent, incomplete or duplicated data so the quality of the data can be improved. To find the errors this process relies on profiling information [76] [69].
DATA INTEGRATION, AGGREGATION AND REPRESENTATION	Data integration	Integrating data involves combining data from multiple sources into one, meaningful and valuable set of data [47] [31] [76].

Table 1.2 Big data life cycle processes definitions II.

Major Process	Process included	Description
	Data storing	To preserve integrated data understandable, maintaining its quality, and adequacy to its purpose, it is also required to develop a suitable storage architecture design, taking into account the type of database suitableness (e.g. relational, non-relational), the capacities of the database management system, etc., among all the alternatives in which data could be stored. [1]
	Data visualization	Typically one step before analysis techniques. This process is about applying a graphical representation to the data, aimed at providing ease at future usage, transformation and understanding [23] [87] [9].
MODELING ANALYZING	Statistics & machine learning	This concerns about stating facts in this context, from a given dataset, by interpreting data and providing a numerical picture of it, as well as using computer systems that emulate the human learning process, saving new information and outcomes, closely related to artificial intelligence (AI). Parallel statistics algorithms have been proposed to approach big data [56] [10].
	Data mining	This process involves techniques to find latent valuable information, revealing patterns, cause-effect relations, implicit facts, etc., to hold up data analysis [55] [9] .

Table 1.3 Big data life cycle processes definitions III.

Major Process	**Process included**	**Description**
	Data analysis	A task done by the perceptual and cognitive system of an analyst, although nowadays machine learning and AI techniques can also be used. This is the ultimate phase in the process of obtaining the value from the data, by finding the significance on the insights provided by, for example, correlations or graphical representations. [87]
INTERPRETATION	Understand and verify results	This is the phase in which the information obtained is used and trans- formed into a decision that could lead to a tangible value (e.g. economic gaining, marketing advantages, scientific progress), differing each time according to the context on which it has been obtained and its purpose This phase also comprises retracing the results obtained, verifying them and testing them in several use cases [1].

In order to support the value characteristic of the data it is important to satisfy high quality conditions of the large data sets [7], where quality could be measured by its dimensions, having completeness, uniqueness, timeliness, validity, accuracy and consistency considered as the six core ones [3], among other identified dimensions, such as reputation, security, transactability, accesibility and interpretability [65] [53]. The processes, technologies and techniques aimed at obtaining the value from big data, are known as big data analytics [59], applied in several ambits, such as health care, social sciences, environmental and natural resources area, business and economic domains, and technology fields. Recently, big data quality has been approached from a big data analytics point of view [43] [4] [46]. Some studies might conclude that data quality is not a bigger challenge than the lack of knowledge from analysts to implement

the correct methods to manage big data value [46], however, data management is an inherent phase of the big data analytics, and it involves the capacity to gather, integrate and

analyze data as well, where data quality should not be considered as a separate phase. The Data Warehousing Institute estimated low quality data cost U.S. businesses more than 600 billion USD per annum [22]. The U.S. Postal Service estimated that wrong data cost 1.5 billion USD in 2013 from mailpieces that could not been delivered to the given addresses, facing data quality problems from around 158 billion mailpieces in that single year. Data quality management strategies were recommended to increase address information accuracy, timeliness and completeness [58]. In big data, "low" error rates translate into millions of faults annually, where the risk is to lose 10-25% of the total revenues from it [22]. Big data quality requires multidisciplinary participation to progress [4] and propel the development of simple and low-cost data quality techniques, reduce the cost of poor quality data, the data error rate, and the need of data cleansing processes which involve investing not only budget, but time and effort to manage. IS research is demanded to collaborate with data wrangling insights and advances, working together with statistical experts to leverage the techniques involved, where domain specific authorities are needed to set the data analytics management, which should support the right value retrieval out of relevant problems from each area [4].

1.2 The Concept of Data Quality

From a research perspective, data quality has been addressed in different areas, including statistics, management, and computer science. Statisticians were the first to investigate some of the problems related to data quality, by proposing a mathematical theory for considering duplicates in statistical data sets, in the late 1960's. They were followed by researchers in management, who at the beginning of the 1980's focused on how to control data manufacturing systems in order to detect and eliminate data quality problems. Only at the beginning of the 1990's computer scientists begin considering the problem of defining, measuring, and improving the quality of electronic data stored in databases, data warehouses, and legacy systems.

Dr. Genichi Taguchi [39], who was a world-renowned quality engineering expert from Japan, emphasized and established the relationship between poor quality and overall loss. Dr. Taguchi (1987) used a quality loss function (QLF) to measure the loss associated with quality characteristics or parameters. The QLF describes the losses that a system suffers from an adjustable characteristic. According to the QLF, the loss increases as the characteristic y (such as thickness or strength) gets further from the target value (m). In other words, there is a loss associated if the quality characteristic diverges from the target. Taguchi regards this loss as a loss to society, and somebody must pay for this loss. The results of such losses include system breakdowns, company failures, company bankruptcies, and so forth.

Figure 1.1 shows how the loss arising from varying (on either side) from the target by Δ_0 increases and is given by $L(y)$ when y is equal to m,

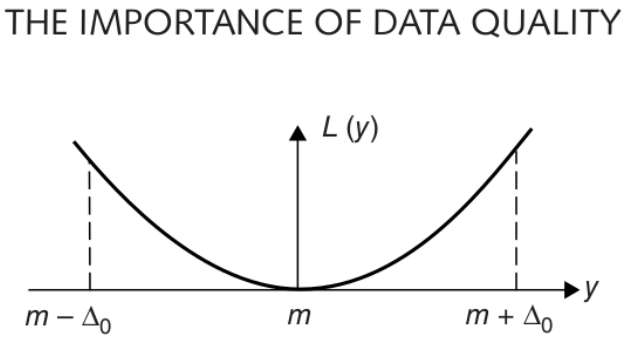

Fig. 1.2 Quality Loss Function (QLF)

the loss is zero, or at the minimum. The equation for the loss function can be expressed as follows:

$$L(y) = k(y-m)^2$$

where k is a factor that is expressed in dollars, based on direct costs, indirect costs, warranty costs, reputational costs, loss due to lost customers, and costs associated with rework and rejection. There are prescribed ways to determine the value of k. The loss function is usually not symmetrical-sometimes it is steep on one side or on both sides. Deming [18] says that the loss function need not be exact and that it is difficult to obtain the exact function. As most cost calculations are based on estimations or predictions, an approximate function is sufficient-that is, close approximation is good enough.

The concept of the loss function aptly applies in the DQ context, especially when we are measuring data quality associated with various data elements such as customer IDs, social security numbers, and account balances. Usually, the data elements are prioritized based on certain criteria, and the quality levels for data elements are measured in terms of percentages (of accuracy, completeness, etc.). The prioritized data elements are referred to as critical data elements (CDEs).

1.3 The Specificity of DQ in Big Data

The term big data itself comprises a deeper meaning, being not only a term to be defined but the name of a phenomenon and an emerging discipline [21]. The earliest mentions of big data were made, to the best of my knowledge, in 1979 by Lawrence Stone, when describing

the work of cliometricians dealing with "vast quantities of data using electronic computers to process it and applying mathematical procedures" [80], and by Charles Tilly, who in 1980 describing the work done by the former, used the term "big-data people" referring to the cliometricians Stone mentioned before [82]. Then, in 1997 Michael Cox and David Ellsworth, described the term as large data sets that surpasses the available size in main memory, local disk and even remote disk [13]. Following that year, Sholom M. Weiss and Nitin Indurkhya in their book "Predictive data mining: a practical guide", contextualize not only the term but the phenomenon, mentioning that "millions or even hundreds of millions of individual records can lead to much stronger conclusions, making use of powerful methods to examine data more comprehensively" and also acknowledges that analyzing big data, in practice, has many difficulties [88]. Two years later, Francis X. Diebold defined big data as "the explosion in the quantity of available and potentially relevant data, largely the result of recent and unprecedented advancements in data recording and storage technology". By this time, it was clear that big data was not only about size, but about the insights that a large set of data could eventually bring.

The Oxford English Dictionary, which added the term to its data base in 2013 [20], defines big data as "data of a very large size, typically to the extent that its manipulation and management present significant logistical challenges".

Nevertheless, those "logistical challenges" for one organization could be necessary to be done when facing a smaller size of data compared to another [49], in this sense, it seemed that relying only in the size depends on the available technology within each organization and its capability to handle a given amount of data, so, to scope the definition, the size of big data could be thought as the size in which using traditional techniques to process it, is not longer an option.

The above led to define big data not only regarding size, but taking into account another identified characteristics, known as the "V's of big data" [50] : Volume, Velocity, Variety, Veracity and Value; where Volume refers to the data size, Velocity evokes the high speed of change and fast generation of data, Variety relates to the different type of data (structured, unstructured and semistructured), Veracity is the degree of trustworthiness of the data (quality and accuracy) and Value indicates the worth or benefit of gathering, storing, processing and analyzing the data.

Because of the nature of big data, traditional approaches to managing data are not suitable, for example, since the main aim was to handle relational data, and as previously mentioned, big data is not always relational, so, traditional techniques are not expected to work correctly [25].

When processing data, one of the main challenges faced with big data is the large volume of it; this require scaling, and there have been two types of scaling for big data processing: scaling-up and scaling-out, where the former is about implementing powerful machines, with great memory and storage capacity, as well as quick but expensive processors, and the latter refers to the usage of several commodity machines connected as clusters, having a parallel environment, suitable to handle large volumes of data, and an advantageous price-performance relation, compared to the scaling-up approach [88] [25]. For big data frameworks, it is known that with the proper optimizations, scaling-up performs better than scaling-out [2], however, it is still unknown exactly when is better to opt for one approach or the other, this is, the results presented were not dependable on the framework solely, but on the processed data characteristics. Nevertheless, it might be strongly preferred to utilize several smaller machines than high performance computer systems (HPC), because of the higher cost scaling-up represents, and considering the support that new paradigms provide by avoiding the necessity to communicate data, but having the processing ap-publications running where the data is, which proposes a significant performance advantage [81], trading off performance in certain degree for a better cost-benefit ratio.

1.4 Why Data Quality is Relevant

The consequences of poor quality of data are often experienced in everyday life, but often, without making the necessary connections to their causes.

For example, the late or mistaken delivery of a letter is often blamed on a postal service, although a closer look often reveals data-related causes, typically an error in the address, originating in the address database.

Data quality has serious consequences of far-reaching significance, for the efficiency and effectiveness of organizations and business.

1.5 Why Data Quality is Matters

Poor data quality is enemy number one to the widespread, profitable use of machine learning. While the caustic observation, “garbage-in, garbage-out” has plagued analytics and decision-making for generations, it carries a special warning for machine learning. The quality demands of machine learning are steep, and bad data can rear its ugly head twice - first in the historical data used to train the predictive model and second in the new data used by that model to make future decisions. [71]

Data quality is no less troublesome in implementation. Consider an organization seeking productivity gains with its machine learning program. While the data science team that developed the predictive model may have done a solid job cleaning the training data, it can still be compromised by bad data going forward. Again, it takes people — lots of them — to find and correct the errors. This in turns subverts the hoped-for productivity gains. Further, as machine learning technologies penetrate organizations, the output of one predictive model will feed the next, and the next, and so on, even crossing company boundaries. The risk is that a minor error at one step will cascade, causing more errors and growing ever larger across an entire process.

Bad data costs the U.S. $3 trillion per year, IBM's estimate of the yearly cost of poor quality data, in the US alone, in 2016. While most people who deal in data every day know that bad data is costly, this figure stuns. [66]

Importantly, the benefits of improving data quality go far beyond reduced costs. It is hard to imagine any sort of future in data when so much is so bad. Thus, improving data quality is a gift that keeps giving — it enables you to take out costs permanently and to more easily pursue other data strategies. For all but a few, there is no better opportunity in data.

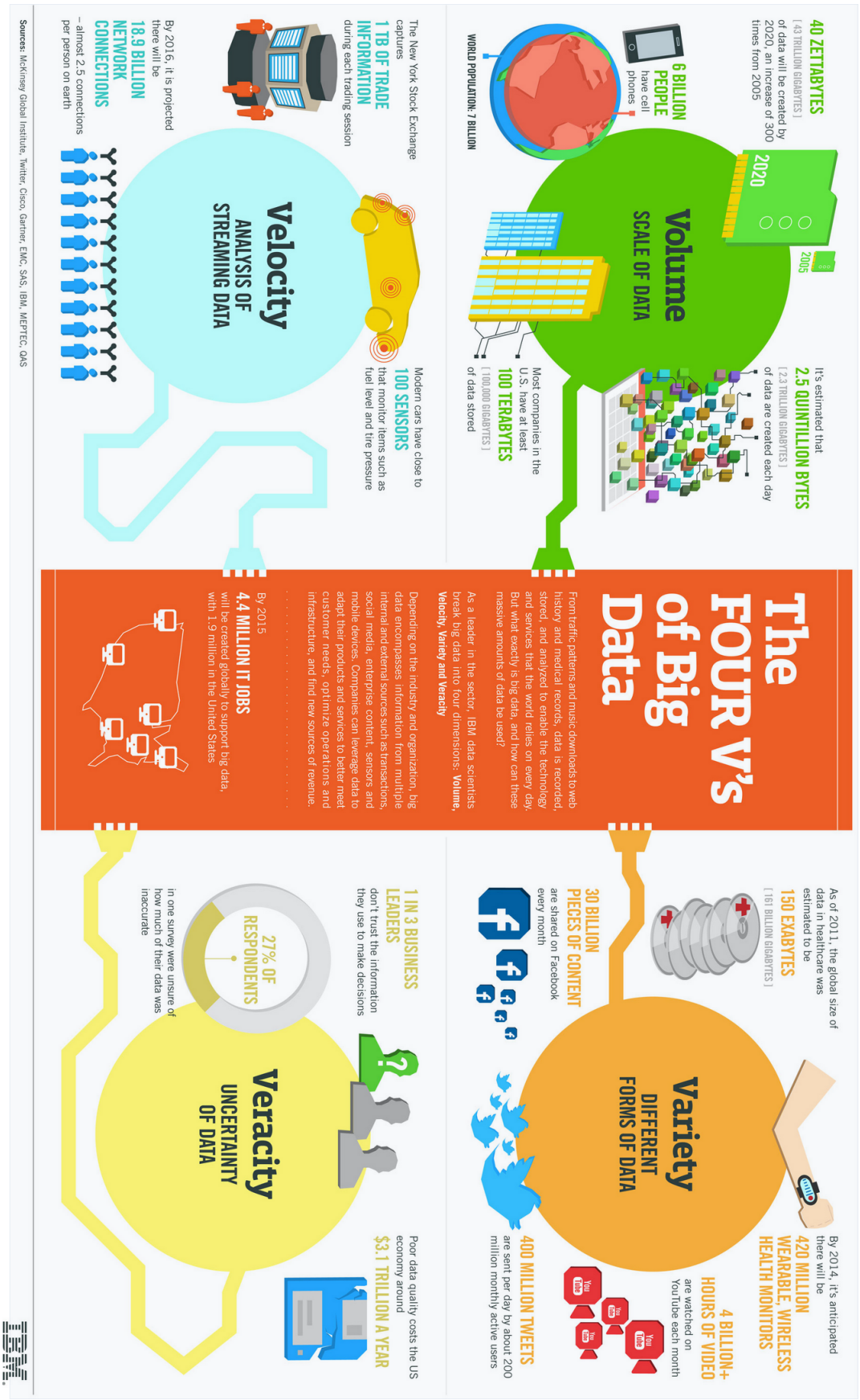

Fig. 1.3 IBM data scientists break big data into four dimensions: volume, variety, velocity and veracity. This infographic explains and gives examples of each. [66]

1.6 Data Quality and Types of Information Systems

Data are collected, stored, elaborated, retrieved, and exchanged in information systems used in organizations to provide services to business processes. Different criteria can be adopted for classifying the different types of information systems, and their corresponding architectures; they are usually related to the overall organizational model adopted by the organization or the set of the organizations that make use of the information system.

The three classifications are represented together in the classification space of Figure 1.2. Among all possible combinations, five main types of information systems are highlighted in the figure: Monolithic, Distributed, Data Warehouses,Cooperative, and Peer-to-Peer.

- In a *monolithic information system* presentation, application logic, and data management are merged into a single computational node. Many monolithic information systems are still in use. While being extremely rigid, they provide advantages to organizations, such as reduced costs due to homogenetiy of solutions and centralization of management. In monolithic systems data flows have a common format, and data quality control is facilitated by the homogenetiy and centralization of procedures and management rules.

- A *data warehouse* (DW) is a centralized set of data collected from different sources, designed to support management decision making. The most critical problem in DW design concerns the cleaning and integration of the different data sources that are loaded into the DW, in that much of the implementation budget is spent on data cleaning activities.

- A *distributed information system* relaxes the rigid centralization of monolithic systems, in that it allows the distribution of resources and applications across network of geographically distributed systems. The network can be organized in terms of several tiers, each made of one or more computational nodes. Presentation, application logic, and data management are distributed across tiers. Usually, the different tiers and nodes have a limited degree of autonomy, data design is usually performed centrally, but to a certain extent some degree of heterogenetiy can occur, due to the impossibility of establishing unified procedures. Problems of data management are more complex than in monolithic systems, due to the reduced level of centralization.

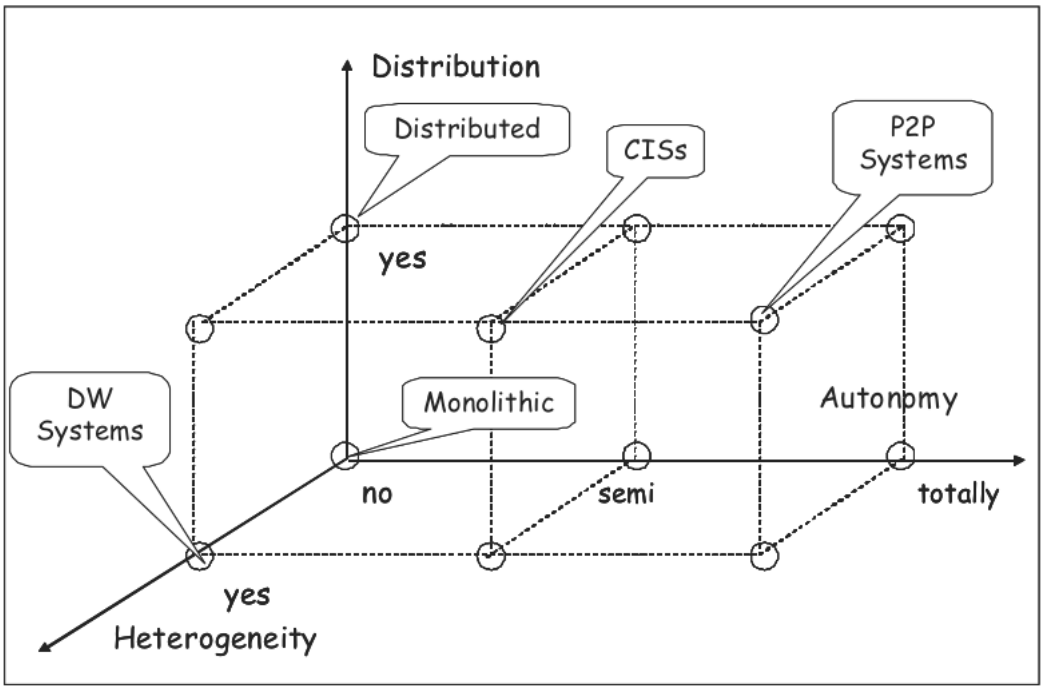

Fig. 1.4 Types of information systems

- A *coorparative information system* (CIS) can be defined as a large-scale information system that interconnects various systems of different and autonomous organizations, while sharing common objectives.
- In a *peer to peer information system* (usually abbreviated P2P), the traditional distinction between clients and servers typical of distributed systems is disappearing. A P2P system can be characterized by a number of properties: peers are highly autonomous and highly heterogeneous, they have no obligation for the quality of their services and data, no central coordination and no central database exist, no peer has a global view of the system, global behavior emerges from local intreactions.

1.7 Main Research Issues and Application Domains in Data Quality

Due to the relevance of data quality, its nature, and the variety of data types and information systems, achieving data quality is a complex, multidisciplinary area of investigation. I involves several research topics and real-life application areas

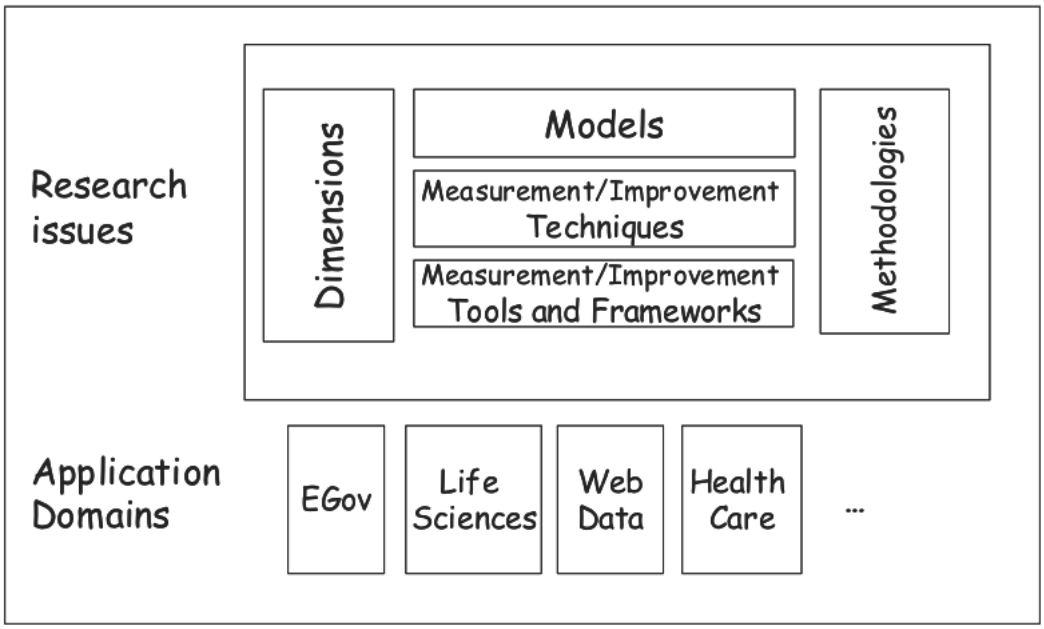

Fig. 1.5 Main issues in data quality

Chapter 2

Data Quality Dimensions

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. Both data dimensions are usually defined in qualitative way, referring to general properties of data and schemas, and related definitions do not provide any facility for assigning values to dimensions themselves.

Specifically, definitions do not provide quantitative measures, and one or more metrics are to be associated with dimensions as separate, distinct properties. For each metric, one or more measurement metadata are to be provided regarding (i) where the measurement is taken, (ii) what data are included, (iii) the measurement device, and (iv) the scale on which results are reported.

According to the literature, at times we will distinguish between dimensions and metrics, while other times we will directly metrics.

Table 2.1 Data Quality Dimensions I

Dimension Name	Description
Data Governance	Do organization-wide data standards exist and are they enforced? Do clearly defined roles and responsibilities exist for data quality related activities? Does data governance strive to acquire and maintain high-quality data through proactive management?
Data Specifications	Are data standards documented in the form of a data dictionary, data models, meta data, and integrity constraints?
Data Integrity	How is data integrity maintained? How are data integrity violations detected and resolved ?
Data Consistency	If data redundancy exists, how is data consistency achieved? What methods are used to bring consistency to data that has become inconsistent? If data is geographically replicated, how is the consistency and latency managed?
Data Currency	Is the data current? Do procedures exists to keep the data current and purge stale data?
Data Duplication	Are there effective procedures in place to detect and remove duplicate data?
Data Completeness	Is the data about entities complete? How is missing data managed?
Data Provenance	Is a historical record of data and its origination maintained? If the data is acquired through multiple sources and has undergone cleaning and transformations, does the organization maintain a history of all changes to the data?
Data Heterogeneity	If multi-modality data about an entity is available, is that data captured and used?
Streaming Data	How is streaming data sampled, filtered, stored, and managed for both real-time and batch processing?
Outliers	How are outliers detected and addressed? Are there versions of datasets that are outlier-free? Does each version correspond to a different method for outlier detection and treatment?
Dimensionality	Reduction Do the datasets feature dimensionality reduced versions? How many versions are available?
Feature Selection	Do datasets have versions that exclude features that are either redundant, highly correlated, or irrelevant? How many versions are available?

Table 2.2 Data Quality Dimensions II

Dimension Name	Description
Feature Extraction	Do the datasets provide a set of derived features that are informative and non- redundant, in addition to the original set of variables/features? How many such derived feature sets are available?
Business Rules	Does a process exist to identify, refine, consolidate, and maintain business rules that pertain to data quality? Do rules exist to govern data cleaning and transformations, and integrating related data of an entity from multiple sources? What business rules govern substitutions for missing data, deleting duplicate data, and archiving historical data? Are there rules for internal data audit and regulatory compliance?
Data Accuracy	Data can be syntactically accurate and yet semantically inaccurate. For example, a customer's mailing address may meet all the syntactic patterns specified by the postal service, yet it can be inaccurate. How does the organization establish the accuracy of data?
Gender Bias	Is the data free from factors that lead to gender bias in machine learning algorithms?
Confidentiality and Privacy	Are procedures and controls implemented for data encryption, data de- identification and re-identification, and differential privacy?
Availability and Access Controls	How is high data availability achieved? What security controls are implemented to protect data from unauthorized access? How are user entitlements to data access and modifications defined and implemented?

2.1 Accuracy

Accuracy is defined as the closeness between a value v and a value v' , considered as the correct representation of the real-life phenomenon that v aims to represent. As an example if the name of a person is Aytac, the value v' = Ayta? is correct, while the value v = Ayt is incorrect. Two kinds of accuracy can be identified, namely a syntactic accuracy and a semantic accuracy.

Let us consider a relation schema **R** consisting of **K** attributes and a relational table **r** consisting of N tuples.

Let $q_{ij}(i = 1..N, j = 1..K)$ be a boolean variable defined to correspond to the cell values y_{ij}, is syntactically accurate, while otherwise it is equal to 1.

In order to identify whether or not accuracy errors affect a matching of relational table **r** with a reference table **r'** containing correct values, we introduce a further boolean variable s_i equal to 0 if the tuple t_i matches a tuple in **r'**, and otherwise equal to 1. We can introduce three metrics to distinguish the relative importance of value accuracy in context of the tuple. The first two metrics have the purpose of giving a different importance to errors on attributes that have a higher identification power, in line with the above discussion.

The first metric called *weak accuracy error,* and is defined:

$$\sum_{i=1}^{N} \frac{\beta(q_i > 0) \wedge (s_i = 0)}{N}$$

where $\beta(.)$ is a boolean variable equal to 1 if the condition in parentheses is *true*, 0 otherwise, and $q_i = \sum_{j=1}^{K} q_{ij}$. Such metric considers the case in which for a tuple t_i accuracy errors occur $(q_i > 0)$ but do not affect identification $(s_i = 0)$.

The second metric is called *strong accuracy error*, and is defined assigning

$$\sum_{i=1}^{N} \frac{\beta(q_i > 0) \wedge (s_i = 1)}{N}$$

where $\beta(.)$ and q_i have the same meaning as above. Such a metric considers the case which accuracy errors occur $(q_i > 0)$ for a tuple t_i and actually do affect identification $(s_i = 1)$.

The third metric gives the percentage of accurate tuples matched with the reference table. It is expressed by the degree of syntactic accuracy of the relational instance **r**

$$\sum_{i=1}^{N} \frac{\beta(q_i = 0) \wedge (s_i = 0)}{N}$$

by actually considering the fraction of accurate $(q_i = 0)$ matched $(s_i = 0)$ tuples.

2.2 Completeness

Completeness can be generically defined as the extent to which data are of sufficient breadth, depth, and scope for the task at hand [86] three types of completeness are identified. Schema completeness is defined as the degree to which concepts and their properties are not missing from the schema. Column completeness is defined as a measure of the missing values for a specific property or column in a table. Population completeness evaluates missing values with respect to a reference population. If focusing on a specific data model, a more precise

characterization of completeness can be given. In the following we refer to the relational model.

2.2.1 Completeness of Web Data

Data that are published in Web information systems can be characterized by evolution in time. While in the traditional paper-based media, information is published once and for all, Web information systems are characterized by information that is continuously published.

We consider a function $C(t)$, defined as the value of completeness at the instant t, with $t \in [t\_pub, t\_max]$, where t pub is the initial instant of publication of data and t_max corresponds to the maximum time within which the series of the different scheduled updates will be completed. Starting from the function $C(t)$, we can define the completability of the published data as

$$\int_{t\_curr}^{t\_max} C(t),$$

where t_curr is the time at which completability is evaluated and $t\_curr < t\_max$.

Completability, as shown in Figure 2.1, can be graphically depicted as an area Cb of a function that represents how completeness evolves between an instant t_curr of observation and t max. Observe that the value corresponding to t_curr is indicated as c_curr; c_max is the value for completeness estimated for t max. The value c_max is a real reachable limit that can be specified for the completeness of the series of elements; if this real limit does not exist, c_max is equal to 1. In Figure 2.5, a reference area A is also shown, defined as

$$(t\_max - t\_curr) * \frac{c\_max - c\_pub}{2},$$

that, by comparison with Cb, allows us to define ranges [High, Medium, Low] for completability.

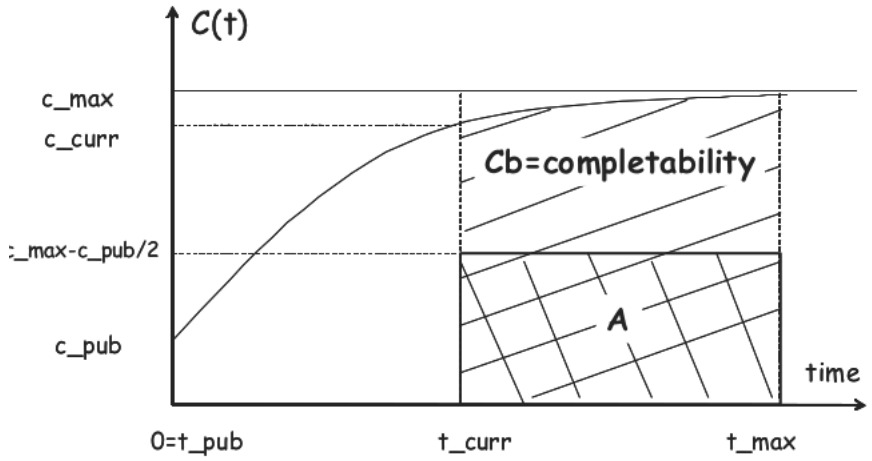

Fig. 2.1 A graphical representation of completability

With respect to the example above, considering the list of courses published on a university Web site, the completeness dimension gives information about the current degree of completeness; the completability information gives the information about how fast this degree will grow in time, i.e., how fast the list of courses will be completed. The interested reader can find further details in [64].

2.3 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. With reference to relational theory, integrity constraints are an instantiation of such semantic rules. In statistics, data edits are another example of semantic rules that allow for the checking of consistency.

2.3.1 Integrity Constraints

Integrity constraints are properties that must be satisfied by all instances of a database schema. Although integrity constraints are typically defined on schemas, they can at the same time be checked on a specific instance of the schema that presently represents the extension of the database. Therefore, we may define integrity constraints for schemas, describing a schema quality dimension, and for instances, representing a data dimension.

Most of the considered integrity constraints are dependencies. The following main types of dependencies can be considered:

- *Key Dependency.* This is the simplest type of dependency. Given a relation instance r, defined over a set of attributes, we say that for a subset K of the attributes a key dependency holds in r, if no two rows of r have the same K-values. For instance, an attribute like SocialSecurityNumber can serve as a key in any relation instance of a relation schema Person.

- *Inclusion Dependency.* Inclusion dependency is a very common type of constraint, and is also known as referential constraint. An inclusion de- pendency over a relational instance r states that some columns of r are contained in other columns of r or in the instances of another relational in- stance s. A foreign key constraint is an example of inclusion dependency, stating that the referring columns in one relation must be contained in the primary key columns of the referenced relation.

- *Functional Dependency.* Given a relational instance **r**, let **X** and **Y** be two nonempty sets of attributes in **r**. **r** satisfies the functional dependency $\mathbf{X} \rightarrow \mathbf{Y}$ if the following holds for every pair of tuples t_1 and t_2 in **r**:

$$\boxed{If, t_1.X = t_2.X, then, t_1.Y = t_2.Y,}$$

where the notation $t_1.X$ means the projection of the tuple t_1 onto the attributes in X.

2.4 Other Data Quality Dimensions

There are general proposals for sets of dimensions that aim to fully specify the data quality concept in a general setting Some other proposals are related to specific domains that need ad hoc dimensions in order to capture the peculiarities of the domain. For instance, specific data quality dimensions are proposed in the following domains:

1. The archival domain (see [89] and [45]) which makes use of dimensions such as condition (of a document) that refers to the physical suitability of the document for scanning.

2. The statistical domain; every National bureau of census and international organizations such as the European Union or the International Monetary Fund define several dimensions for statistical and scientific data, such as integrity, on the notion that statistical systems should be based on adherence to the principle of objectivity in the collection, compilation, and dissemination of statistics

3. The geographical and geospatial domain (see [61] [30]), where the following dimensions are proposed: (i) positional accuracy, defined as a quality parameter indicating the accuracy of geographical positions, and (ii) attribute/thematic accuracy, defined as the positional and/or value accuracy of properties such as sociodemographic attributes in thematic maps.

Chapter 3

Data Quality (DQ) Evaluation

3.1 Metrics and Measurement

Any data can have its quality measured. Using a data driven strategy, the measurements acts on the data itself to quantify the DQD (Data Quality Dimension). As mentioned before, our work is based on structured data represented in [75] a set of attributes, columns, and rows with their values. Any data quality metric should specify whether the values of data respect or not the quality attributes. The data quality measurement metrics tend to evaluate a binary results correct or incorrect or a value between 0 and 100, and use universal formulas to compute these attributes. This will apply to many quality dimensions such as accuracy, completeness, and consistency.

		Data Quality Dimensions Related		
	Data Quality Issues	**Accuracy**	**Completeness**	**Consistency**
Instance Level	*Missing Data*	X	X	
	Incorrect data, Data entry errors,	X		
	Irrelevant data			X
	Outdated data	X		
	Misfiled and Contradictory values	X	X	X
Schema Level	*Uniqueness constrains, Functional dependency violation*	X		
	Wrong data type, poor schema design			X
	Lack of integrity constraints	X	X	X

Table 3.1 Data Quality Issues vs DQD

The DQDs (Data Quality Dimensions) must be relevant to the DQ problems as identified In Table 3.1 Therefore DQ Metrics are designed for each DQD to measure if the attributes respect the previously defined DQD. These measures are done for each attribute given it type, data ranges values, and if it is collected from data profiling.

For example a metric that calculates the accuracy of a data attribute is defined as follows:

- The data type of an attribute and its values.
- For numerical attributes a range or sets of acceptable values (Textual also) are defined. Any other values are incorrect.
- The accuracy of an attribute is calculated based on the number of correct values divided by number of observations or rows.
- For another data types/formats like images, videos, audio files, another type of metrics must be defined to evaluate accuracy or any other quality dimensions. The authors of [24] describe usefulness as an aspect of data quality for images. For this kind of data, feature extraction functions are defined on the data and extracted for each data item. These features have constraints that characterize the goodness or badness of data values. Some of quality metrics functions are designed based on the extracted features such as, usefulness, accuracy , completeness and any other data quality dimensions judged by domain experts to be candidate for such data type.

3.2 DQ Issues and Big Data Characteristics

Data characteristics commonly named V's are initially, Volume, Velocity, Variety, and Veracity. Since the Big Data inception; we reached now 7 V's and probably we will keep going [83]. The veracity tends more to express and describe trust and certainty of data that can be expressed mostly as quality of the data. The DQD accuracy is often related to precision, reliability and veracity [84].

A mapping tentative between these characteristics, data and data quality is complied in [35] [24] [8]. The authors attempted to link the V's to the quality dimensions.

3.3 Big Data Quality Evaluation Scheme

The purpose of Big Data Quality Evaluation (BDQ) Scheme is to address the data quality before starting data analytics. This is done by estimating the quality of data attributes or

DQ Dimensions	Metric functions
Accuracy	Acc = (Ncv / N)
Completeness	Comp = (Nmv / N)
Consistency	Cons = (Nvrc / N)
Ncv	Number of correct values
Nmv	Number of missing values
Nvrc	Number of values that respects the constraints
N	Total number of values (rows) of the sample Dataset

Table 3.2 DQD metric functions

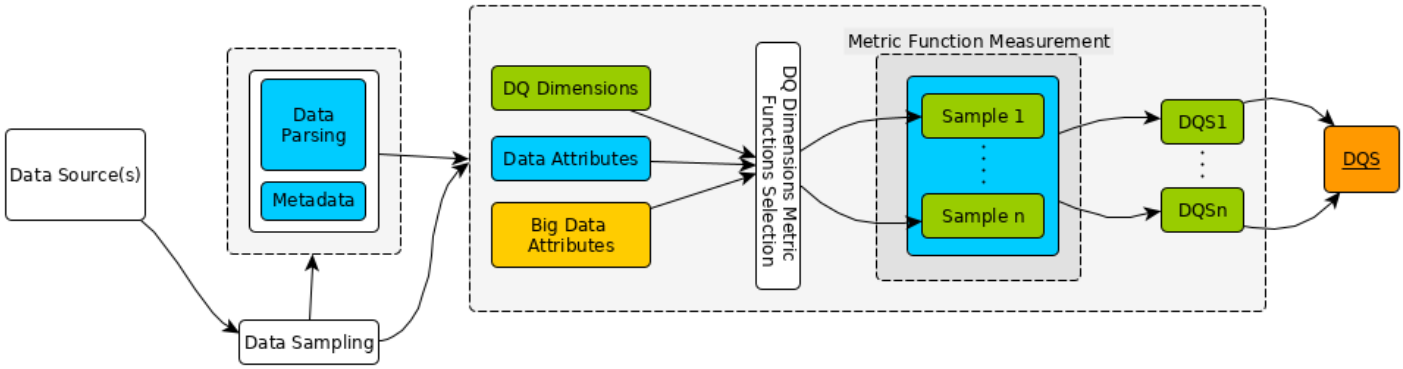

Fig. 3.1 Big Data Quality Evaluation Scheme

features by applying a DQD metric to measure the quality characterized by its accuracy, completeness or/and consistency. The expected result is data quality assessment suggestions indicating the quality constraints that will increase or decrease the data quality.

The BDQ Evaluation scheme is illustrated in Figure 3.1 where the data goes through many module to estimate its quality. The key modules of our scheme consist of: (a) data sampling; and data profiling, (b) DQD vs attributes selection, (c) data quality Metric selection, (d) samples data quality evaluation. In the following sections, we describe each module, its input(s), output(s), and the main functions.

3.3.1 Big Data Sampling

A sample is representative of a whole population. Based on a sample, we make several decisions about a population. A sample is also called a subgroup. The number of observations or units in a sample is called sample size.

The number of times a sample is collected is usually referred to as the sampling frequency. In designing a control chart, we must specify both of these parameters. [39]

There are several sampling strategies that can be applied on Big Data as expressed in [26] [12]. They evaluated the effect of sampling methods on Big Data and believed that sampling large datasets reduces run time and computational footprint of link prediction algorithms though maintaining sufficient prediction performance. In statistic, Bootstrap sampling technique evaluates the sampling distribution of an estimator by sampling with replacement from the original sample. In the context of Big Data, Bootstrap sampling has been addressed in many works [48] [77].

In our data evaluation scheme will used the Bag of Little Bootstrap (BLB) [44], which combines the results of bootstrapping multiple small subsets of a Big data dataset. THe BLB algorithm use an original Big dataset used generate small samples without replacements. For each generated sample another set of samples are created by resampling with replacement.

3.3.2 Data Profiling

Data profiling is an exploratory approach to data quality analysis. Statistical approaches are used to reveal data usage patterns as well as patterns in the data [60] [51]. Several tools exist for data quality assessment using data profiling and exploratory data analysis. Such tools include Tableau and Talend Open Studio.

Data profiling module performs screening of data quality based on statistics and information summary. Since profiling is meant to discover data characteristics from data sources. It is considered as data assessment process that provides a first summary of the data quality. Such information include: data format description, different attributes, their types and values. data constraints (if any), data range, max and min. More precisely information about the data are presented in two types; technical and functional. This information can be extracted from the data itself without any additional representation using it metadata or any descriptive header file, or by parsing the data using any analysis tools. This task may become very costly in Big Data. To avoid costs generated due the data size we will use the same sampling process BLB to reduce the data into a representative population sample, in addition to the combination of profiling results.

3.3.3 Data Quality Evaluation

The data profiling provides information about dataset,

- Data attributes (eg. type format)
- Data summary (eg. max, min)

- Big data attributes; size number of sources speed of data generation (eg. data streams)
- What DQD evaluate.

The previous information's are used to select the appropriate quality metrics functions $\boldsymbol{F}$ to evaluate a data quality dimensions d_k for an attribute a_i with a weight w_j

In the Fig 3.2 we describe how data quality evaluated using bootstrap sampling for Big data. The process follows these steps:

1. Sampling from the data set **S n** bootstrap sample size of **ss** size without replacement DS_j.
2. Each sample generated from step 1 is sampled into n' samples of size SS with replacements DS_{ij}
3. For the Each sample DS_{ij} generated in step 2, evaluate the data quality score Q_{ij}

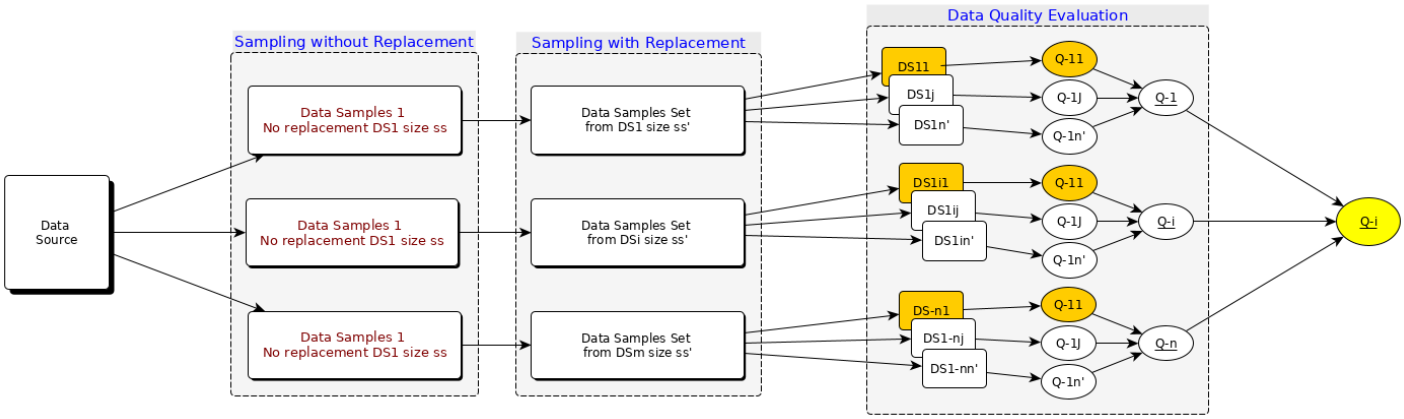

Fig. 3.2 Big Data Quality Sampling Evaluation

Table 3.3 Big Data Quality Evaluation Algorithm

Algorithm: Big Data Quality Evaluation

let ds a Original Data Set with size **SS and Observation (N-SS)**;
let ss (**b(SS)**) the samples with ss < SS ;
let n samples s_i of size **ss** and ***M*** Observation **(M-ss)**;
let D a set of DQD $D = \{d_0, ...d_k, ...d_q\}$;
let F a metric function **F (completeness, accuracy,...)** ;
let cc $\leftarrow 0$ counter of correct valid attribute value(when **F** *is true* $cc = cc + 1$);
let S ={ DS_0**,...**DS_i**,...**DS_n **}** without replacement ;
for $i \leftarrow 0$ **to** n **do**
    Generate sample s_i *of size SS from ds*;
    for $j \leftarrow 0$ **to** n' **do**
        Generate a sample $_{ij}$ *of size SS from sample si*;
        for $k \leftarrow 0$ **to** j **do**
            `MetricFunctionTuple(`d_k, F`)`
            for $a \leftarrow 0$ **to** j **do**
                for $a_{ij}(x)$ ***ss*** *values* **do**
                    if ***F(***$a_{ij}(x)$***, value) == 1*** **then**
                        measure metric ;
                        $c \leftarrow$ cc + 1 ;
                        Calculate the scores vector DQD(F, d_k***,***a_{ij}***,*** DS_i***)=*** $\frac{cc}{N}$ *cc* $\leftarrow$0 *counter of correct valid attribute value* ***(***d_k***,F)***
                    end
                end
                DQD d_k *computed for all attributes for a sample* ds_{ij}
            end
            DQS_{ijk} *is the* D_K *scores for an attribute* a_ij *for sample* DS_{ij}
            Q_{ijk} *sum of all* d_k *scores for attributes* a_{ij} *for* DS_{ij}
        end
        $Q_{ik} += 1/n'(Q_{ijk})$
    end
end
Q_k *is the mean of all* Q_{ik} *for a specif* d_k $Q_k += 1/n(Q_{ik})$

3.3.4 BDQ Evaluation Algorithm

Let ***F*** represents a set of data quality metrics, ***F*****=**$\{f_0 \ldots f_1, \ldots f_m\}$ where f_1 a quality metric function that will measure and evaluate a DQD d_k for each value of an attribute a_i in the sample s_i and returns 1 if correct, 0 if not. Each f_1 function will compute if the value of the attribute reflects the d_k constraints. For example, the metric accuracy of an attribute is defined as a range of values between 0 and 100, otherwise it is correct. Similarly, it can be defined to satisfy a certain number of constraints related to the type of data such as a zip code, email, social security number, or an address. If we are evaluating the same DQD d_k for a set of attributes, if the weights are all equal, a simple mean is computed. The metric f_i will be evaluated to measure if all the attributes individually have their f_i correct. This is done for each instance (cell or row) of the sample s_i.

In Table 3.3, we describe the detail of BDQ Evaluation Algorithm. The Q_k represents the mean quality score for a DQD d_k for measurable attributes. For data set let note. A as a set of attributes or features. The Q_k values respectively for each attribute are represented by a set of quality scores:

$$V = \{Q_{ka_1}, \ldots Q_{ka_m}\}$$

where **A** is set of m attributes. With this evaluation, we have more insights, statistics and benefits about the Big data quality to ensure a well-refined analytics that targets the best precision.

3.3.5 After Evaluation Analysis

The data evaluation process done on Big data set provides data quality information and scores of quality dimensions of each attributes or features. These scores are used to identify the data that must be targeted and omitted. A set of proposals actions is generated based on many parameters, like DQD, or data quality issue. If a data attribute got a lower score than the required level (%) of accuracy or completeness the following actions are proposed:

- Discard it from the dataset.
- Tune, reformat, and normalize its values.
- Replace values, as in missing data.

Whatever the Quality evaluation results, it always contains actions to be taken on the dataset to remove any irregularities using techniques like cleaning, filtering and pre-processing based on the quality assessment.

Chapter 4

Experimentations , Results and Analysis

In this chapter, we will explain how data quality dimensions are implemented with Machine Learning (ML) techniques towards dirty datasets.

4.1 Experimental Methodology

The impact of the dirt data and data cleaning on ML in a dataset depends on a number of factors – some factors depend on the data cleaning process,

i.e., the error types to be cleaned and the cleaning methods; some factors depend on the ML process, i.e., the model types used; and some factors depend on where the cleaning is performed during the ML process. Hence, in order to comprehensively investigate the impacts, we need to consider data cleaning an ML jointly in our experiments.

4.2 The NettoyageML Schema

The NettoyageML relational schema consists of three relations as shown in Table 4.1 . Firstly will introduced the attributes of NettoyageML relational models, and then will explained the differences between these three relations.

- **Attributes for Dataset.** The first attribute is dataset, which is the input to the data cleaning and ML pipeline. Each dataset can have multiply types of errors and has an associated ML task.

R1 Vanilla

Dataset	Error Type	Detection	Repair	ML Model	Scenario	Flag

R2 (With Model Section)

Dataset	Error Type	Detection	Repair	Scenario	Flag

R3 (With Model Selection and Cleaning Method Selection)

Dataset	Error Type	Scenario	Flag

Table 4.1 NettoyageML Relational Schema

Even for one error type, they might appear in a dataset in various distributions and hence affect ML models in complicated ways. To study the impact of the real-world error types and distribution on ML models, we mostly use real-world-dataset with real errors, and we apply various cleaning methods to detect and repair the errors in them.

- **Attributes for Data Cleaning.** The error type attribute describes which type of dirtiness we are testing. We consider five most common types of dirtiness that are considered in the ML and DB communities: missing values, outliers, duplicates, inconsistencies, and mislabels. For each error type, there exist multiple cleaning methods, and each cleaning method includes an error detection component and an error repair component.
- **Attributes for ML.** The ML model describes the algorithm of training and prediction. Different ML models may have different robustness or sensitivity to dirty data.
- **Flag Attribute.** *The flag attribute summarizes* the impact of data cleaning on ML of an experiment into three categories "**P (positive)**", "**N (negative)**" or "**S (insignificant)**", indicating whether the cleaning has a positive, negative, or insignificant impact on the ML performance respectively.

Given these attributes, we designed three relations as shown Table 4.1 *R1* shows the vanilla of the a NettoyageML relation schema with the key *{ dataset, error type, detect, repair, ML model, scenario, flag }* Every tuple of *R1* represents a hypothesis or an experiment: how

does cleaning some type of error using a detection method and a repair method affect a ML model for a given dataset? We also consider other two versions of relations in NettoyageML.

Compared with *R1*, *R2* eliminates the ML model attribute. In this case, we try different models during training and select the model that has the best validation accuracy (or F1 score) as the model to be considered in an experiment in *R2*.

Every tuple of R2 represents a hypothesis or an experiment: how does cleaning some type of error using a detection method and a repair method affect the best ML model for a given dataset?

R3 further eliminates the cleaning method (detection and repair) attributes. In this case, in addition to model selection, we also try different cleaning methods and select the cleaning method that results in the best validation accuracy as the cleaning method to be considered in an experiment in *R3*. Every tuple of *R3* represents a hypothesis or an experiment: how does the best cleaning method affect the predictive performance of the best model for a given dataset?

All three relations can also be extended with other attributes that are associated with an experiment, which may help obtain insights and interpret the results, such as the accuracy scores before and after cleaning and p-values of hypothesis testing associated with each experiment.

4.3 Analysis Methodology

We first fix the error type and group by flags. The percentage of each type of flag indicates the general impact of cleaning this type of error on ML. For example, if flag “P” dominates in the error type outliers, it indicates cleaning outliers generally improves the performance of ML. Then we group by the other attributes (e.g., ML models, datasets, etc.) in addition to flags to see how each attribute affects the impact.

For example, if flag “S” dominates under the ML algorithm decision tree, it indicates decision tree is insensitive to this error. We also investigate the changes of percentage in each type of flag when moving from *R1*, to *R2* and *R3*.

This indicates how model selection and cleaning algorithm selection affect the impacts.

For example, if the percentage of flag “N” significantly decreases from *R1* to *R2*, it indicates the model selection reduces the negative impact of data cleaning on ML.

We investigate the impact of dirty data on ML models by simply running SQL queries on three relations *R1*, *R2* and *R3*.

We formally present the SQL query templates as follows, where E $\in$ inconsistencies, duplicates, mislabels, outliers, missing values.

Q1: Flag

```
SELECT flag, COUNT(*) FROM R
WHERE error_type = E
GROUP BY flag
```

Listing 4.1 Q1: Flag

Q2: Scenario

```
SELECT scenario, flag, COUNT(*)
FROM R
WHERE error_type = E
GROUP BY scenario, flag
```

Listing 4.2 Q2: Scenario

Q3: ML Model. (Not applicable to R2, R3)

```
SELECT model, flag, COUNT(*)
FROM R
WHERE error_type = E
GROUP BY model, flag
```

Listing 4.3 Q3: ML Model

Q4: Clean Method (Not applicable to R3 or E $\in$ inconsistencies, duplicates, mislabels, where only one cleaning method is applied

```
SELECT detect_method, flag, COUNT(*)
FROM R
WHERE error_type = E
GROUP BY detect_method, flag
```

Listing 4.4 Q4.1:Detect Method

```
SELECT repair_method, flag, COUNT(*)
FROM R
WHERE error_type = E
GROUP BY repair_method, flag
```

Listing 4.5 Q4.2:Repair Method

-

```
SELECT dataset, flag, COUNT(*)
FROM R
WHERE error_type = E
GROUP BY dataset, flag
```

Listing 4.6 Q5: Dataset

4.4 Design of Benchmark

Based on NettoyageML Relational Schema, we design NettoyageML Benchmark by specifying the domain of each key attribute. In the following sections we present all datasets, error types, cleaning methods, ML models, and cleaning scenarios we consider in the benchmark.

4.5 Error Types and Cleaning Methods

We consider five error types that are prevalent in the real-world datasets, including missing values, outliers, duplicates, inconsistencies and mislabels.

Table 4.2 Cleaning Methods

Error Type	**Detect Method**	**Repair Method**
Missing Values	Empty Entry	Deletion
		Imputation: Mean Mode Median Mode Mode Mode Mean Dummy Mode Dummy Median Dummy
Outliers	SD	Deletion
	IQR	Imputation: Mean Median Mode
	IF	
Duplicates	Key Collision	Deletion
Inconsistencies	OpenRefine	Merge
Mislabels	Ground Truth	Flip Labels

While there are many cleaning methods in the literature we consider the most straightforward one in this study. The definition and the cleaning methods of each error type are described below and summarized in Table 4.2

4.5.1 Missing Values

Missing values occur when no value is stored for some attribute in an observation. We detect missing values by finding empty or *NaN* entries in datasets. We use two methods to repair missing values:

- **Deletion:** Delete records with missing values.
- **Imputation:** For numerical missing values, we consider three types of imputation methods: mean, median and mode. For categorical missing values, we use two types of imputation methods: the mode (most frequent class) or a dummy variable named "missing". Therefore, we have six imputation methods. In Table 4.2 we denote each imputation method by the numerical imputation and categorical imputation (e.g. "Mean Dummy" represents imputing numerical missing values by mean and imputing categorical missing values by dummy variables).

4.5.2 Outliers

An outlier is an observation that is distant from other observations. We only consider numerical outliers in our experiments. We use three methods to detect numerical outliers.

- **Standard Deviation Method (SD:)** A value is considered to be an outlier if it is n numbers of standard deviations away from the mean of the attribute it belongs to. We use $n = 3$.
- **Interquartile Range Method (IQR):** Let Q_1 and Q_3 be the $25th$ and the $75th$ percentiles of an attribute. Then, interquartile range $IQR = Q_3 - Q_1$. A value is considered to be an outlier of it is outside range of $[Q_1 - k * IQR, Q_3 + k * IQR]$ We use $k = 1.5$.
- **Isolation Forest Method (IF):** The isolation forest isolates observations by randomly selecting a feature and randomly selecting a split value of the selected feature. This partition can be represented by a tree structure and outliers will have noticeably shorter paths in the random trees. We use **the scikit-learn IsolationForest** and set the contamination parameter to be $\mathbf{0.01}$

We use two methods to repair numerical outliers in the datasets.

- **Deletion:** Records with outliers entries are removed from the datasets.
- **Imputation:** Outliers entries are imputed. We consider three types of imputation: **mean**, **median** and **mode**.

4.5.3 Duplicates

Duplicates refer to the records that correspond to the identical real-world entity. We detect duplicates by defining the key attribute that is unique for each entity in the dataset. If two records have an identical value on the key attribute, they will be considered as duplicates. We repair the duplicates by keeping the first and deleting all the others.

4.5.4 Inconsistencies

Inconsistencies occur when two values correspond to the identical real-world entity but have different representations. We detect inconsistencies in datasets using OpenRefine [85] and repair them by merging different representations into one in OpenRefine.

4.5.5 Mislabels

Mislabels occur when an observation is incorrectly labeled. Since mislabels in our datasets come from error injection, we already know which label is incorrect. We repair mislabels by flipping the label. Our protocol in injecting class noise follows the recommendation in [27]

4.6 Datasets

Collected 13 real-world datasets from different sources to conduct our experiments. Each dataset contains one or more types of error summarized in Table 4.3. For mislabels, we cannot find existing real-world datasets with both mislabels and ground truth. Since it is difficult to clean mislabels without ground truth (we do not know whether a label is mislabeled unless we have ground truth or domain knowledge), we injected mislabels in three real-world datasets.

-

Table 4.3 Dataset and Error Types

	Error Types				
Datasets	**Inconsistencies**	**Duplicates**	**Missing Values**	**Outliers**	***Mislabels Data**
Airbnb		X	X	X	
Citation		X			
Company	X				

Credit			X	X	
EEG				X	X
KDD			X	X	X
Marketing			X		
Movie	X	X			
Restaurant	X	X			
Sensor				X	
Titanic			X		
University	X				
USCensus			X		X

- **Airbnb :** This dataset contains 42,492 records on hotels in the top 10 tourist destinations and major US metropolitan areas scraped from Airbnb.com. Each record has 40 attributes including the number of bedrooms, price, location, etc. Demographic and economic attributes were scraped from city-data.com. The classification task is to determine whether the rating of each hotel is 5 or not. This dataset contains missing values, numerical outliers and duplicates.

- **Citation: [15]** This dataset consists of titles of 5,005 publications from Google Scholar and DBLP. Given a publication title, the classification task is to determine whether the paper is related to Computer Science or not. This dataset contains duplicates.

- **Company: [34]** The original dataset contains over 2.5 million records about companies including company names and locations. Because of its large size, we randomly sampled 5% records (128,889 records) from the original dataset. Each record has seven attributes including company name, country, city, etc. The classification task is to predict whether the company sentiment is negative or not. This dataset contains inconsistent company names.

- **Credit: [40]** This dataset consists of 150,000 credit data with 10 attributes including monthly income, age, then number of dependents, etc. The classification task for this dataset is to predict whether a client will experience financial distress in the next two years. This dataset has a class imbalance problem. There are only 6.7in minority class. This dataset primarily contains missing values and numerical outliers.

- **EEG: [19]** This is a dataset of 14,980 EEG recordings. Each record has 14 EEG attributes. The classification task is to predict whether the eye-state is closed or open.

This dataset contains numerical outliers. We inject mislabels into this dataset by randomly flip labels.

- **KDD: [41]** his dataset contains 131,329 records about projects and donations from DonorsChoose.org. Each record has 100 attributes. The classification task is to predict whether a project is "exciting". This dataset has a class imbalance problem. There are 11% records in the minority class. This dataset contains missing values and numerical outliers. We inject mislabels into this dataset by randomly flip labels.

- **Marketing:** This dataset consists of 8,993 data about household in- come from a survey. Each record has 14 demographic attributes including sex, age, education, etc. The classification task is to predict if the annual household income is less than $25000. This dataset contains missing values.

- **Movie: [29] [90]** This dataset consists of 9,329 movie reviews, which we obtained by merging data from IMDB and TMDB datasets. Each record has seven attributes including title, language, score, etc. The classification task is to predict the genre of the movie (romance or comedy). It contains duplicates and inconsistent representations of languages.

- **Restaurant: [28]** This dataset contains 12,007 records about restaurants, which we obtained by merging data from the Yelp and Yellowpages datasets. Each record has 10 attributes including city, category, rating, etc. The classification task is to predict whether the categorical price range of a restaurant is "$" or not. This dataset consists of duplicates and inconsistent restaurant names and categories.

- **Titanic: [42]** This dataset contains 891 records and 11 attributes from the Titanic including name, sex, age, etc. The classification task is to determine whether the passenger survived or not. This dataset has a significant number of missing values.

- **Sensor: [16]** The original sensor dataset contains 928,991 sensor recordings with eight attributes including temperature, humidity, light, etc. Because of the large size, we only used recordings from sensor 1 and sensor 2 and sampled the dataset to include 1 observation per hour, and day for each sensor. The sampled dataset contains 62,076 records. The classification task is to predict whether the readings came from a particular sensor (sensor 1 or sensor 2). This dataset contains outliers.

- **University: [79]** This dataset contains 286 records about universities. Each record has 17 attributes including state, university name, SAT scores, etc. The classification

task is to predict whether the expenses are greater than 7, 000 for each university. This dataset contains inconsistent representations for states and locations.

- **USCensus: [54]** This dataset contains 32,561 US Census records for adults. Each record has 14 attributes including age, education, sex, etc. The classification goal is to predict whether the adult earns more than $50,000. This dataset contains missing values.

4.7 ML Models

We select seven classical and competitive classification algorithms in our experiments, including **Logistic Regression, KNN, Decision Tree, Random Forest, Adaboost, XGBoost and Naive Bayes**. We used **scikit-learn** [63] to train models. Each model is described below.

- **Logistic Regression:** Logistic regression is a binary classifier that uses a Sigmoid function to create a linear classification boundary. Logistic regression uses optimization methods to determine the best regression coefficient of the function based on the training data. [52]

- **KNN Classifier:** *KNN classifier* uses a distance metric (*Euclidean distance* in our experiments), and the number of nearest neighbors (k) to calculate the distance between records in the training set. After calculating the distance it then retrieves the k nearest neighbors. Once the algorithm has found those neighbors, it can classify a record in the test set by computing its distance to other training records and using the class labels of the nearest neighbors to deter- mine the label class of the unknown record. [33]

- **Decision Tree:** CART (Classification& Regression Trees) decision trees were used in this analysis. During training, the decision tree splits the data based on homogeneity. Gini index is used to measure node impurity and the attribute with minimum Gini index is used as the split node. This algorithm recursively partitions data until the splitting is completed. [67]

- **Random Forest:** Random forests are an ensemble learning method for classification. During training, the random forests algorithm constructs several decision trees and outputs an aggregated prediction (often the mode of the classes). Predictions in the test set are then made using this output [14].

- **Adaboost**: Adaboost, also known as "Adaptive boost", is a meta- learning algorithm with high theoretical and empirical performance. It uses weak learners and transforms them into high performance learners by repeatedly emphasizing mispredicted instances. In ex- periments, the decision tree is our base learner. [32]

- **XGBoost**: XGBoost is short for "Extreme Gradient Boosting". It is an implementation of gradient tree boosting system designed to be highly efficient and scalable. It is one of the most popular packages used by competitors to win ML challenges [11]. We use gradient boosted tree as our base learner in the experiments.

- **Naive Bayes:** Naive Bayes predicts a class given a set of features using Bayes Theorem. This algorithm assumes independence among all attributes when the class is given [72].

- We preprocess features before training ML models following these common practice: **(1)** Categorical variables were encoded using one hot encoding. **(2)** Text embeddings were used for non- categorical text attributes. We computed their tf-idf matrix using **TfidfVectorizer** from **scikit-learn** [63]. **(3)** Data were standardized to a mean of 0 and variance of 1. **(4)** Class-imbalanced datasets were downsampled, i.e., for every observation of the minor class, we randomly sample from the major class without replacement, to make the number of the instances in the major class equal to that in the minor class during the training.

4.8 Scenarios

Given a dataset with a train/test split, and a cleaning method, we can have different model performance depending on where the cleaning is performed. Table 4.4 shows the four cases: **Case A** represents a model built using the original dirty training set and tested on the original dirty test set; **Case B** represents a model built using the original dirty training set and tested on the cleaned test set; **Case C** represents a model built using the cleaned training set and tested on the original dirty test set; and **Case D** represents a model built using the cleaned training set and tested on the cleaned test set. **Our goal is compare two of them to evaluate how cleaning affects model performance**, and a chosen comparison between two cases is what we call a scenario in Table 4.1

Table 4.4 Where Cleaning is Performed

	Dirty Test Set	**Cleaned Test Set**
Dirty Training Set	A	B
Cleaned Training Set	C	D

Table 4.5 Where Cleaning is Performed (Missing Values)

	Deletion Test set	**Imputation Test set**
Deletion Training set	A	B
Imputation Training set	C	D

Do we then have a total of $C_4^2 = 6$ scenarios ? The answer is no, and in fact, only two of them ("BD" and "CD") make sense as explained as follows:

- **Scenario "BD"**. This shows the effects of cleaning dirty data in the training set when models are evaluated on the clean test set. This is reflective of the real-world model building phase, where we are given a dirty training set and we would like to know whether we need to clean the training set. Of course, in order to test whether cleaning training set helps, the two models need to be evaluated on the same cleaned test set.

- **Scenario "CD"**. This shows the effects of cleaning dirty data in the test set when models are trained on the clean training set. This is reflective of the real-world model deployment phase, where the model is deployed and is being used for new test data, and we would like to know whether cleaning incoming dirty test data helps with the predictive performance.

- **Scenario "AB"**. We do not compare the entries A and B because in real-world data cleaning, we do not consider evaluating a test set on a model trained with dirtiness, especially we are not interested in the performance improvement/degradation when swap- ping dirty with clean test sets.

- **Scenario "AC"**. The comparison between A and C is also not reported because in production we are not interested in the performance of models evaluated on a dirty test set. It is common practice to ensure the test set is clean for evaluating the model performance.

- **Scenario "AD" and "BC"**. The two scenarios are based on two different settings, which are not directly comparable.

Special Handling for Missing Values. Missing values need special attention; they occur when no value is stored for a variable in an observation. We cannot train models when some data is missing. Thus, Cases A and B in Table 4.4 are not available for missing values. Instead of comparing dirty and cleaned datasets, we compare the difference between a deletion dataset (a dataset with missing values deleted) and an imputation dataset (a dataset with missing values imputed).

The four cases for missing values are shown in Table 4.5. We only consider the "BD" scenario for missing values, which captures the difference of imputing and deleting training samples while testing the model performance in the imputed test set. This scenario is the authentic scenario we encounter in production, where it is not allowed to delete instances from the test set, so missing values in the test set have to be repaired using imputation methods.

4.9 Running The Benchmark

We have defined the domain of each key attribute. We call each valid assignment of key attributes an experiment specification. By definition, the experiment specification for a tuple t in relation is $t[R-Flag]$, where $R \in \{R1, R2, R3\}$. For example in Table 4.6 s_1, s_2 and s_3 are there experiment specifications in $R1$, $R2$ and $R3$, respectively.

s_1

Dataset	Error Type	Detection	Repair	ML Model	Scenario
EEG	Outliers	IQR	Mean Imputation	Logistic Regression	BD

s_2

Dataset	Error Type	Detection	Repair	Scenario
EEG	Outliers	IQR	Mean Imputation	BD

s_3

Dataset	Error Type	Scenario
EEG	Outliers	BD

Table 4.6 Example of Experiment specifications

In this section, we present how to run the benchmark to generate flags for each experiment specification in $R1$, $R2$ and $R3$.

4.9.1 Generating One Metric Pair

Given an experiment specification in R1, we can generate a pair of metrics through following steps:

1. **Split dataset.** We first split dataset randomly into a training and a test set with $70/30$ ratio.

2. **Clean dataset.** We clean the error in the training set and test set with the specific cleaning methods. To avoid the data leakage problem, all statistics needed for data cleaning, such as mean and standard deviation, are computed only on the training set and used to clean both training and test set.

3. **Training ML models.** If the scenario is BD, we train two ML models, one on dirty training set and one on clean training set. If the scenario is **CD**, we only train one ML model on the clean training set. We tune hyper-parameters of ML models using random search and 5-fold cross validation.

4. **Evaluating ML models.** If the scenario is **BD**, we evaluate two ML models (one trained on a dirty training set, anther trained on a clean training set) on the clean test set to get a pair of metrics. If the scenario is **CD**, the model trained on clean training set will be evaluated on dirty test set and clean test set respectively to get a pair of metrics. The evaluation metric is selected based on class imbalance. For class- imbalanced datasets *(i.e., KDD and Credit)*, we use F1 score to evaluate the performance of models but for all the other datasets, we use accuracy as the evaluation metric.

For experiment specifications in $R2$, the difference is that we train various ML models at step **(3)** and select the model with the best validation accuracy from cross validation as the model evaluated in step **(4)**. For R3, in addition to model selection, we use various cleaning methods to clean dataset at step **(2)** and select the cleaning method resulting in the best validation accuracy. At step **(4),** the test set cleaned by the best cleaning method is used to evaluate the best model.

Example 1. *We take the specifications in Table 4.6 as an example to show how to generate one metric pair for each specification.*

To generate metric pair for s_1 , we first split EEG into training and test datasets. We detect outliers in the training set and test set using **IQR** detection and repair them with mean imputation. The quantiles used in detection and mean used in repair are computed on the training set. Then, since the scenario here is **BD**, we train two logistic regression models on

a dirty training set and a cleaned training set respectively. Finally, we evaluate two models on the cleaned test set and get two test accuracy scores to form a metric pair as shown in Table 4.7

	Train on Dirty Training Set		**Train on Clean Training Set**	
Model	**Validation Accuracy**	**Clean Test Accuracy**	**Validation Accuracy**	**Clean Test Accuracy**
Logistic Regression	0.638849	0.634179	0.673467	0.668892
Metric Pair: (0.634179, 0.668892)				

Table 4.7 s_1 Metric Pairs

To generate the metric pair for s_2 , we train various ML models. Table 4.8 shows that based on the validation accuracy, **XGBoost** is the best model trained on the dirty training set and **KNN** is the best model trained on the clean training set. We then evaluate two best models on the cleaned test set to get two test accuracy scores to form a metric pair.

	Train on Dirty Training Set		**Train on Clean Training Set**	
Model	**Validation Accuracy**	**Clean Test Accuracy**	**Validation Accuracy**	**Clean Test Accuracy**
AdaBoost	0.763205	0.711393	0.718193	0.715176
Decision Tree	0.822621	0.754784	0.796487	0.810414
KNN	0.895481	0.821095	0.948312	0.956386
Logistic Regression	0.638849	0.634179	0.673467	0.668892
Naive Bayes	0.453365	0.457276	0.634745	0.638407
Random Forest	0.918556	0.854695	0.903680	0.903680
XGBoost	0.932098	0.862706	0.920369	0.922786

Table 4.8 s_2 Metric Pairs

To generate the metric pair for s_3 , in addition to model selection, we use various cleaning methods. Table 4.9 shows the clean test accuracy of best models under each cleaning methods.Based on the validation accuracy, detecting outliers by **SD** and repairing by deletion is the best cleaning method. Hence, we use its metric pair as the metric pair for s_3.

Detect Method	Repair Method	Validation Accuracy of Best Model on Clean Training set	Clean Test Accuracy of Best Model on Dirty Training set	Clean Test Accuracy of Best Model on Clean Training set
SD	Delete	0.959370	0.937612	0.969928
SD	Mean Imputation	0.955179	0.938140	0.964174
SD	Median Imputation	0.955179	0.938140	0.964174
SD	Mode Imputation	0.955179	0.937917	0.964174
IQR	Delete	0.958072	0.929052	0.967190
IQR	Mean Imputation	0.948312	0.862706	0.956386
IQR	Median Imputation	0.944115	0.868046	0.951268
IQR	Mode Imputation	0.946308	0.870049	0.957499
IF	Delete	0.959250	0.935250	0.969846
IF	Mean Imputation	0.957466	0.925456	0.966845
IF	Median Imputation	0.957371	0.924789	0.966177
IF	Mode Imputation	0.956990	0.927236	0.966400
Metric Pair: (0.937612, 0.969928)				

Table 4.9 s_3 Metric Pairs

4.10 Handling Randomness

The above procedure has randomness, which mainly comes two sources: (1) Search Randomness. This is introduced by random search in tuning hyper-parameters. Different search spaces will result in different performances, which may affect the evaluation of ML models. (2) Split Randomness. This is introduced by random train/test split. Different train/test splits may result in different error distribution, which may affect the quality of data cleaning.

4.10.1 Handling Search Randomness

To handle the randomness from random search, we repeat step (3) and step (4) 5 times with different seeds for random search. Each random search will generate a pair of metrics. To aggregate 5 pairs, for specifications in $R1$, since we care more about the performance of a model on average, we averages each metric in the pair over 5 random searches. For specifications in $R2$ and $R3$, similarly as we select the best model based on the validation accuracy, we select the one with the best validation accuracy from 5 random searches for each metric. After repeating experiments with 5 times random search, we still have one metric pair for each specification, but it provides a better evaluation of the model performance and reduces the effect of randomness caused by random search.

Example 1. Table 4.10 shows the five metric pairs we get from repeating random search with 5 different seeds. For s_1 , we average over 5 random search for each metric. For s_2 ,

as shown in Table 4.11, we select the one with the best validation accuracy from 5 random search for each metric. s_3 can be generated in a similar way.

Train on Dirty Training Set			Train on Clean Training Set	
Random Search Seed	**Validation Accuracy**	**Clean Test Accuracy**	**Validation Accuracy**	**Clean Test Accuracy**
8006	0.638849	0.634179	0.673467	0.668892
6130	0.638849	0.635292	0.673562	0.667557
5824	0.638849	0.634846	0.673372	0.668669
3659	0.638754	0.635291	0.672323	0.668892
3239	0.639040	0.634179	0.672323	0.669114
Average		0.634767		0.668625
Aggregated Metric Pair: (0.634767, 0.668625)				

Table 4.10 Aggregate Five Random Search For s_1

Train on Dirty Training Set			Train on Clean Training Set	
Random Search Seed	**Validation Accuracy**	**Clean Test Accuracy**	**Validation Accuracy**	**Clean Test Accuracy**
8006	0.932098	0.862706	0.948312	0.956386
6130	0.930381	0.868046	0.948312	0.956386
5824	0.932098	0.862706	0.920369	0.922786
3659	0.930381	0.868046	0.948312	0.956386
3239	0.932098	0.862706	0.948312	0.956386
Metric Pair: (0.862706, 0.956386)				

Table 4.11 Aggregate Five Random Search For s_2

4.10.2 Handling Split Randomness

To avoid the occasionality caused by a train/test split, we randomly split each dataset 20 times with different seeds and repeat the experiments under the same protocol on each train/test split. Each split will generate one pair of metrics. Hence, we end up with 20 metric pairs for each specification.

Example 2. Table 4.12 shows 20 pairs of metrics from 20 different train/test splits for s_1.

Given 20 pairs of metrics for each specification $s = t[R-Flag]$, we generate the flag $t[Flag]$ using paired sample $t-test$. We consider 20 metric pairs as two sets of 20 observations from the same dataset before and after data cleaning. Then paired sample $t-testcan$ determine whether the mean difference between two sets of observations is zero, positive or negative. The paired sample $t-test$ is formally defined below.

For each specification $t[R-Flag]$, let μ^t be the mean difference of the metrics of the ML model before and after we clean the error in the dataset with the detection and repair method. We define null and alternative hypotheses for three types of paired sample $t-test$ as:

Split Seed	B	D	Split Seed	B	D
v144	0.632488	0.657321	v5192	0.631954	0.67401
v235	0.634757	0.668625	v5374	0.638362	0.676992
v2516	0.625812	0.666266	v5396	0.641032	0.672452
v2895	0.636404	0.662394	v6542	0.63992	0.670049
v2962	0.637161	0.674633	v7751	0.640098	0.669871
v3462	0.644726	0.673654	v7813	0.634535	0.676591
v3562	0.635514	0.67401	v8093	0.636271	0.666489
v4225	0.641478	0.674989	v8444	0.632443	0.673431
v4764	0.649177	0.680196	v905	0.636671	0.673565
v5056	0.629773	0.669381	v9394	0.632176	0.668803

Table 4.12 Accuracy Evaluated on the Clean Test Set

Hypothesis	**Two-tailed t-test**	**Upper-tailed t-test**	**Lower-tailed t-test**
Null	$H_0^t : \mu^t = 0$	$H_1^t : \mu^t \leq 0$	$H_2^t : \mu^t \geq 0$
Alternative	$H_a^t : \mu^t \neq 0$	$H_b^t : \mu^t > 0$	$H_c^t : \mu^t < 0$

Table 4.13 Hypotheses

We run three types of paired sample $t-test$ on 20 metric pairs. Let p_0, p_1, p_2 be the $p-values$ of two tailed $t-test$, upper-tailed $t-test$ and lower-tailed t-test, respectively.

Let α be the significant level.The procedure for determining flags using paired sample t-test is described below:

$$
\begin{aligned}
&(1) \quad if \quad p_0 \geq \alpha, \quad t[Flag] = \text{"}S\text{"} \\
&(2) \quad if \quad p_0 < \alpha \quad \text{and} \quad p_1 < \alpha, \quad t[Flag] = \text{"}P\text{"} \\
&(3) \quad if \quad p_0 < \alpha \quad \text{and} \quad p_2 < \alpha, \quad t[Flag] = \text{"}N\text{"}
\end{aligned}
$$

The intricacy of conducting two-tailed test and one-tailed test together lies in the fact that if the test statistics distribution is symmetric (e.g., Gaussian), the p-value in one of one-tailed tests is half of the p-value in a two-tailed test. Hence, a two-tailed test with significance does imply that the one-tailed test under the same distribution is also significant; yet if the one-tailed test is significant, the two-tailed one is not necessarily significant. What is criticized often is that people only report a one-tailed test p-value because the two-tailed test is insignificant. However, in our case, we do not face this claim because we conduct

three tests and only report the one- tailed test results if its corresponding two-tailed test is significant.

Example 3. For s_1 , we run two-tailed, upper-tailed and lower-tailed sample t-test on 20 metric (4.12) pairs $Assume \alpha = 0.05$. Table 4.14 *shows* $p_0 < \alpha$ *and* $p_1 < \alpha$. Thus, the flag of s_1 is determined to be "P"

Type	p-value
Two-Tailed (p_0)	3.82E-17
Upper-tailed (p_1)	1.91E-17
Lower-tailed (p_2)	1

Table 4.14 p-values in t-test and Hypothesis Testing

4.11 Controlling False Discoveries

Since we aim at studying the significant impacts of data cleaning techniques on ML performances in spite of the search and split randomness, we face the challenges that not all statically significant results in individual hypothesis tests are true discoveries. Some results are significant simply due to the large number of tests examined.

This commonly known as the multiple hypothesis testing or the multiple comparisons problem in the statistics literature [74].

To see the effect of multiple testing, consider a case where there are 20 hypotheses to test and we set a significance level of $\alpha = 0.05$. The probability of observing at least one significant result just due to chance is $1 - (1 - \alpha)^{20} \approx 0.64$

With just 20 test considered, we have a 64% chance of observing at least one significant result, even if all of the tests are actually not significant.

With 3, 990, 570 and 150 hypotheses in our relations $R1$, $R2$ and $R3$, respectively, it is highly likely that our results contain many false discoveries by chance. Strategies to control the false discoveries caused by the multiple hypothesis testing problem usually adjust the significance level α in some way [74] [5].

For example a simple way to adjust α is called the *Bonferroni correction* [6] [1] which test each hypothesis at the significance level of $\frac{\alpha}{m}$ instead of α, where m is the number of false negatives because it can miss a lot of true significant tests.

[1] Bonferroni correction is one of many familywise error rate (FWER) methods, for an extensive survey, c.f. [91]

Instead of adjusting the significance level α for every test, another strategy is to rank the tests by their p-values, which indicate the statistical significance levels of tests [68].

This is called the **FDR** approach [37] where we ensure that in expectation, $\frac{V}{R} = FDR$, where R is the total number of rejections, and V the number of false rejections. Common FDR approaches include Benjamini-Hochberg (BH) and Benjamini-Yekutieli (BY) procedures, where we try to control the FDR that is "(upper) bounded by a user-defined level α" [37].We employed the BY procedure since it controls the FDR under arbitrary dependence assumptions[1] For each relation, we conduct a separate multiple hypothesis testing. We assign α to be 0.05 in our experiments.

Example 1. P-values for s_1 Table 4.14 are collected in a multiple testing setting where we run BY-Procedure on p-values of all hypotheses in $R1$. Table 4.15 shows the collected the corrected p-values for s_1. Since $p_0 < \alpha \quad and \quad p_1 < \alpha$. Thus, the flag of s_1 is finally determined to be "P".

Test-Type	Corrected p-value
Two-Tailed (p_0)	6.28E-17
Upper-tailed (p_1)	3.25E-17
Lower-tailed (p_2)	1

Table 4.15 Corrected p-values of the Example Analysis (Outlier)

4.12 Analyzing Benchmark Results

We inspect the correlation between the test accuracy scores in the scenarios BD and CD. In Figure 4.1 The scatter plots are generated based R3. We plot the test accuracy scores of 20 splits in each dataset, given the best model and the best data cleaning method. Different colors correspond to various datasets. The visualization show that:

1. Cleaning does not improve ML performance.
2. Cleaning can help improve accuracy scores up to 10% (e.g., cleaning outliers in the scenario CD).
3. The improvements vary largely from one error type to another.
4. We need a systematic and principled approach to analyze the results.

[1] https://en.wikipedia.org/wiki/False_discovery_rate (last accessed:July 31, 2019

Table 4.16 present the results according to the query templates we define in Section 4.3.

In section 4.13 to 4.17 we present the impacts of each type of error on ML by analyzing the query results. In Section 4.18 we summarize the key insights.

4.13 Inconsistencies

1. Cleaning inconsistencies is more likely to have insignificant impact and unlikely to have negative impact on ML.
2. Model selection increases the probability of having positive impacts on ML.

- **Q1:** We first group by flags on the tuples in R1, R2 and R3. Table 4.16 :Q1(E=Inconsistencies) shows no negative flags ("N") in the impact directionality. For every relation, the insignificant changes ("S") have the largest likelihood. This implies that cleaning inconsistency in both training and test sets is unlikely to produce negative impacts on the ML model performance. Furthermore, comparing the percentages of "P" among all the relations, selecting the best ML model and the best data cleaning strategy helps to gradually introduce positive changes to ML performances after data cleaning.
- **Q2:** Table 4.16: Q2(E=Inconsistencies) shows the query results of grouping by flags and scenarios, which follows the tendency we observe in Q1, i.e., no negative impacts on ML performance after cleaning inconsistency in both scenarios. Adding an auto ML model/cleaning tuner increases the changes of positive impacts.

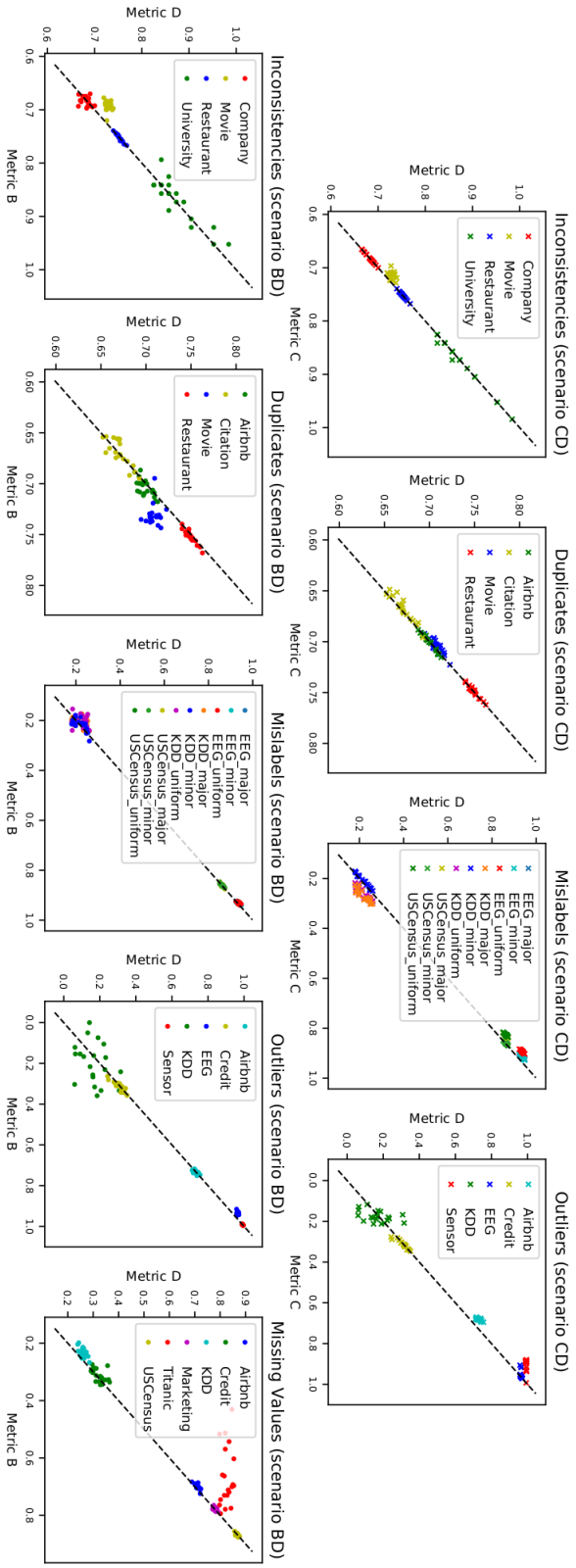

Fig. 4.1 Test Accuracy Scores in Scenarios BD and CD of 20 Splits for Five Error Types

Table 4.16 Benchmark Results(Organized by Query)

Q1

Q1(E=Inconsistencies)

R	P	S	N
R1	14.29% (8)	85.71% (48)	0%(0)
R2	5.0% (2)	75.0% (6)	0%(0)
R3	37.5% (3)	62.5% (5)	0%(0)

Q1(E=Duplicates)

R	P	S	N
R1	17.86% (10)	71.43% (40)	10.71% (6)
R2	12.5% (1)	62.5% (5)	25.0% (2)
R3	25.0% (2)	50.0% (4)	25.0% (2)

Q1(E=Mislabels)

R	P	S	N
R1	59.52% (75)	26.19% (33)	14.29% (18)
R2	61.11% (11)	27.78% (5)	11.11% (2)
R3	61.11% (11)	27.78% (5)	11.11% (2)

Q1(E=Outliers)

R	P	S	N
R1	31.55% (265)	57.02% (479)	11.43% (96)
R2	33.33% (40)	60% (72)	6.67% (8)
R3	30% (3)	70% (7)	0% (0)

Q1(E=Missing Values)

R	P	S	N
R1	61.51% (155)	34.92% (88)	3.57% (9)
R2	50.00% (18)	50.00% (18)	0.00% (0)
R3	50.00% (3)	50.00% (3)	0.00% (0)

Q2

Q2(E=Inconsistencies)

R	Scenario	P	S
R1	BD	7.14% (2)	92.86% (26)
	CD	21.43% (6)	78.57% (22)
R2	BD	25.0% (1)	75.0% (3)
	CD	25.0% (1)	75.0% (3)
R3	BD	25.0% (1)	75.0% (3)
	CD	50.0% (2)	50.0% (2)

Q2(E=Duplicates)

R	Scenario	P	S	N
R1	BD	10.71% (3)	75.0% (21)	14.29% (4)
	CD	25.0% (7)	67.86% (19)	7.14% (2)
R2	BD	0.0% (0)	50.0% (2)	50.0% (2)
	CD	25.0% (1)	75.0% (3)	0.0% (0)
R3	BD	0.0% (0)	50.0% (2)	50.0% (2)
	CD	50.0% (2)	50.0% (2)	0.0% (0)

Q2(E=Mislabels)

R	Scenario	P	S	N
R1	BD	50.79% (32)	49.21% (31)	0.0% (0)
	CD	68.25% (43)	3.17% (2)	28.57% (18)
R2	BD	44.44% (4)	55.56% (5)	0.0% (0)
	CD	77.78% (7)	0.0% (0)	22.22% (2)
R3	BD	44.44% (4)	55.56% (5)	0.0% (0)
	CD	77.78% (7)	0.0% (0)	22.22% (2)

Q2(E=Outliers)

R	Scenario	P	S	N
R1	BD	50.79% (32)	49.2% (31)	0.0% (0)
	CD	68.25% (43)	3.17% (2)	28.57% (18)
R2	BD	44.44% (4)	55.56% (5)	0.0% (0)
	CD	77.78% (7)	0.0% (0)	22.22% (2)
R3	BD	44.44% (4)	55.56% (5)	0.0% (0)
	CD	77.78% (7)	0.0% (0)	22.22% (2)

Q3

Q3(E=Inconsistencies)

R	Model	P	S
R1	AdaBoost	12.5% (1)	87.5% (7)
	Decision Tree	0.0% (0)	100.0% (8)
	KNN	25.0% (2)	75.0% (6)
	Logistic Regression	12.5% (1)	87.5% (7)
	Naive Bayes	12.5% (1)	87.5% (7)
	Random Forest	25.0% (2)	75.0% (6)
	XGBoost	12.5% (1)	87.5% (7)

Q3(E=Duplicates)

R	Model	P	S	N
R1	AdaBoost	25.0% (2)	75.0% (6)	0.0% (0)
	Decision Tree	0.0% (0)	100.0% (8)	0.0% (0)
	KNN	12.5% (1)	87.5% (7)	0.0% (0)
	Logistic Regression	25.0% (2)	62.5% (5)	12.5% (1)
	Naive Bayes	37.5% (3)	50.0% (4)	12.5% (1)
	Random Forest	12.5% (1)	62.5% (5)	25.0% (2)
	XGBoost	12.5% (1)	62.5% (5)	25.0% (2)

Q3(E=Mislabels)

R	Model	P	S	N
R1	AdaBoost	77.78% (14)	11.11% (2)	11.11% (2)
	Decision Tree	66.67% (12)	22.22% (4)	11.11% (2)
	KNN	50.0% (9)	38.89% (7)	11.11% (2)
	Logistic Regression	55.56% (10)	33.33% (6)	11.11% (2)
	Naive Bayes	27.78% (5)	38.89% (7)	33.33% (6)
	Random Forest	61.11% (11)	27.78% (5)	11.11% (2)
	XGBoost	77.78% (14)	11.11% (2)	11.11% (2)

Q3(E=Outliers)

R	Model	P	S	N
R1	AdaBoost	20% (24)	62.50% (75)	17.50% (21)
	Decision Tree	27.50% (33)	65.83% (79)	6.67% (8)
	KNN	50% (60)	44.17% (53)	5.83% (7)
	Logistic Regression	28.33% (34)	56.67% (68)	15% (18)
	Naive Bayes	34.17% (41)	58.33% (70)	7.5% (9)
	RandomForest	29.17% (35)	56.67% (68)	14.17% (17)
	XGBoost	31.67% (38)	55% (66)	13.33% (16)

Q3(E=Missing Values)

R	Model	P	S	N
R1	AdaBoost	20% (24)	62.50% (75)	17.50% (21)
	Decision Tree	27.50% (33)	65.83% (79)	6.67% (8)
	KNN	50% (60)	44.17% (53)	5.83% (7)
	Logistic Regression	28.33% (34)	56.67% (68)	15% (18)
	Naive Bayes	34.17% (41)	58.33% (70)	7.5% (9)
	Random Forest	29.17% (35)	56.67% (68)	14.17% (17)
	XGBoost	31.67% (38)	55% (66)	13.33% (16)

Q4

Q4.1(E=Outliers)

R	Detect Method	P	S	N
R1	IF	32.14% (90)	47.14% (132)	20.71% (58)
	IQR	50.71% (142)	38.21% (107)	11.07% (31)
	SD	11.79% (33)	85.71% (240)	2.50% (7)
R2	IF	32.50% (13)	60% (24)	7.50% (3)
	IQR	52.50% (21)	35% (14)	12.50% (5)
	SD	15% (6)	85% (34)	0% (0)

Q4.2(E=Outliers)

R	Repair Method	P	S	N
R1	Delete	32.86% (69)	50.48% (106)	16.67% (35)
	Mean Imputation	31.90% (67)	60.95% (128)	7.14% (15)
	Mode Imputation	30.95% (65)	57.14% (120)	11.90% (25)
	Median Imputation	30.48% (64)	59.52% (125)	10.00% (21)
R2	Delete	33.33% (10)	53.33% (16)	13.33% (4)
	Mean Imputation	33.33% (10)	63.33% (19)	3.33% (1)
	Mode Imputation	30.00% (9)	63.33% (19)	6.67% (2)
	Median Imputation	36.67% (11)	60.00% (18)	3.33% (1)

Q4.2(E=Missing Values)

R	Imputation Method	P	S	N
R1	Mean Dummy	54.76% (23)	40.48% (17)	4.76% (2)
	Median Dummy	59.52% (25)	35.71% (15)	4.76% (2)
	Mode Dummy	64.29% (27)	30.95% (13)	4.76% (2)
	Mean Mode	64.29% (27)	33.33% (14)	2.38% (1)
	Median Mode	61.90% (26)	35.71% (15)	2.38% (1)
	Mode Mode	64.29% (27)	33.33% (14)	2.38% (1)
R2	Mean Dummy	50.00% (3)	50.00% (3)	0.00% (0)
	Median Dummy	50.00% (3)	50.00% (3)	0.00% (0)
	Mode Dummy	50.00% (3)	50.00% (3)	0.00% (0)
	Mean Mode	50.00% (3)	50.00% (3)	0.00% (0)
	Median Mode	50.00% (3)	50.00% (3)	0.00% (0)
	Mode Mode	50.00% (3)	50.00% (3)	0.00% (0)

Q5

Q5(E=Inconsistencies)

R	Dataset	P	S
R1	Company	28.57% (4)	71.43% (10)
	Movie	21.43% (3)	78.57% (11)
	Restaurant	0.0% (0)	100.0% (14)
	University	7.14% (1)	92.86% (13)
R2	Company	0.0% (0)	100.0% (2)
	Movie	100.0% (2)	0.0% (0)
	Restaurant	0.0% (0)	100.0% (2)
	University	0.0% (0)	100.0% (2)
R3	Company	0.0% (0)	100.0% (2)
	Movie	100.0% (2)	0.0% (0)
	Restaurant	50.0% (1)	50.0% (1)
	University	0.0% (0)	100.0% (2)

Q5(E=Duplicates)

R	Dataset	P	S	N
R1	Airbnb	0.0% (0)	85.71% (12)	14.29% (2)
	Citation	7.14% (1)	85.71% (12)	7.14% (1)
	Movie	64.29% (9)	28.57% (4)	7.14% (1)
	Restaurant	0.0% (0)	85.71% (12)	14.29% (2)
R2	Airbnb	0.0% (0)	100.0% (2)	0.0% (0)
	Citation	0.0% (0)	100.0% (2)	0.0% (0)
	Movie	50.0% (1)	0.0% (0)	50.0% (1)
	Restaurant	0.0% (0)	50.0% (1)	50.0% (1)
R3	Airbnb	0.0% (0)	100.0% (2)	0.0% (0)
	Citation	50.0% (1)	50.0% (1)	0.0% (0)
	Movie	50.0% (1)	0.0% (0)	50.0% (1)
	Restaurant	0.0% (0)	50.0% (1)	50.0% (1)

Q5(E=Outliers)

R	Dataset	P	S	N
R1	Airbnb	0.0% (0)	85.71% (12)	14.29% (2)
	Citation	7.14% (1)	85.71% (12)	7.14% (1)
	Movie	64.29% (9)	28.57% (4)	7.14% (1)
	Restaurant	0.0% (0)	85.71% (12)	14.29% (2)
R2	Airbnb	0.0% (0)	100.0% (2)	0.0% (0)
	Citation	0.0% (0)	100.0% (2)	0.0% (0)
	Movie	50.0% (1)	0.0% (0)	50.0% (1)
	Restaurant	0.0% (0)	50.0% (1)	50.0% (1)
R3	Airbnb	0.0% (0)	100.0% (2)	0.0% (0)
	Citation	50.0% (1)	50.0% (1)	0.0% (0)
	Movie	50.0% (1)	0.0% (0)	50.0% (1)
	Restaurant	0.0% (0)	50.0% (1)	50.0% (1)

Q5(E=Missing Values)

R	Dataset	P	S	N
R1	Airbnb	23.21% (39)	70.83% (119)	5.95% (10)
	Credit	22.62% (38)	43.33% (73)	33.93% (57)
	EEG	60.12% (101)	33.93% (57)	5.95% (10)
	KDD	3.57% (6)	93.45% (157)	2.98% (5)
	Sensor	48.21% (81)	43.45% (73)	8.33% (14)
R2	Airbnb	37.50% (9)	62.50%(15)	0.0% (0)
	Credit	0.0% (0)	66.67% (16)	33.33% (8)
	EEG	79.17% (19)	20.83% (5)	0% (0)
	KDD	0.0% (0)	100% (0)	0% (0)
	Sensor	50% (12)	50% (12)	0% (0)
R3	Airbnb	50% (1)	50% (1)	0.0% (0)
	Credit	0% (0)	100% (2)	0.0% (0)
	EEG	50.0% (1)	50% (1)	0% (0)
	KDD	0.0% (0)	100% (2)	0% (0)
	Sensor	50% (1)	50% (1)	0% (0)

Q5(E=Mislabels)

R	Dataset	P	S	N
R1	EEG major	78.57% (11)	14.29% (2)	7.14% (1)
	EEG minor	78.57% (11)	21.43% (3)	0.0% (0)
	EEG uniform	78.57% (11)	14.29% (2)	7.14% (1)
	KDD major	28.57% (4)	21.43% (3)	50.0% (7)
	KDD minor	50.0% (7)	50.0% (7)	0.0% (0)
	KDD uniform	35.71% (5)	14.29% (2)	50.0% (7)
	USCensus major	50.0% (7)	42.86% (6)	7.14% (1)
	USCensus minor	64.29% (9)	35.71% (5)	0.0% (0)
	USCensus uniform	71.43% (10)	21.43% (3)	7.14% (1)
	EEG major	100.0% (2)	0.0% (0)	0.0% (0)
	EEG minor	100.0% (2)	0.0% (0)	0.0% (0)
	EEG uniform	100.0% (2)	0.0% (0)	0.0% (0)
	KDD major	0.0% (0)	50.0% (1)	50.0% (1)
	KDD minor	50.0% (1)	50.0% (1)	0.0% (0)
	KDD uniform	0.0% (0)	50.0% (1)	50.0% (1)
	USCensus major	50.0% (1)	50.0% (1)	0.0% (0)
	USCensus minor	50.0% (1)	50.0% (1)	0.0% (0)
	USCensus uniform	100.0% (2)	0.0% (0)	0.0% (0)
R2	EEG major	100.0% (2)	0.0% (0)	0.0% (0)
	EEG minor	100.0% (2)	0.0% (0)	0.0% (0)
	EEG uniform	100.0% (2)	0.0% (0)	0.0% (0)
	KDD major	0.0% (0)	50.0% (1)	50.0% (1)
	KDD minor	50.0% (1)	50.0% (1)	0.0% (0)
	KDD uniform	0.0% (0)	50.0% (1)	50.0% (1)
	USCensus major	50.0% (1)	50.0% (1)	0.0% (0)
	USCensus minor	50.0% (1)	50.0% (1)	0.0% (0)
	USCensus uniform	100.0% (2)	0.0% (0)	0.0% (0)

- **Q3:** Grouped by ML models, it is noticeable that all the ML models have demonstrated the same tendency in the impacts of data cleaning, as shown in Table 4.16: Q3(E=Inconsistencies). Again, there is no negative impact on ML performances that cleaning inconsistency can induce.

- **Q5:** At last, we group by datasets and provide a view on dataset choice and its influence on ML performance with data cleaning strategy incorporated. As shown in Table 4.16:(E=Inconsistencies), in general, the pattern holds that insignificant impact of cleaning inconsistency prevails; no negative impacts of cleaning inconsistency was found. It is probably due to the idiosyncrasy in the Movie dataset, which has 48% of inconsistencies [1] the improvement an auto ML/cleaning strategy tuner brings Hatton2019is 78.57% in the direction of positive changes.

4.14 Duplicates

1. Cleaning duplicates is more likely to have insignificant impacts on ML and it is possible to produce negative impacts.

2. With model selection, negative impacts caused by cleaning test set can be eliminated.

- **Q1:** From the query on flags shown in Table 4.16:Q1(E=Duplicates), it is unclear that if cleaning duplicates could bring either positive or negative impacts. In all the relations, the number of insignificant flags has the largest percentage compared with the positive and negative flags.

- **Q2:** Looking scenarios grouped by flags in Table 4.16: Q2 (E=Duplicates), using a clean test set with a clean training model (CD) is highly unlikely to result in negative changes of ML performance. Especially when we utilize an auto ML/data cleaning tuner, the negative impacts disappear. In the scenario BD, utilizing an ML/cleaning tuner does not decrease the chance of yielding negative impacts when we clean the duplicates. This observation is also made from the scatter plot on BD in Figure 4.1, where there are more points with the "N" flag than the other two flags. The reason is due to the various duplication rates of features in the datasets, as we see when analyzing the results of Q5 below.

[1] We calculate the inconsistency of the datasets using the percentage of minority class. For instance, in the dataset Movie, we correct the values English and en under the variable "Language". The value English takes 52% in all the attribute values; the value en 48%. We therefore replace en with English. This gives us an inconsistency rate of 48%.

- **Q3**: We observe from Table 4.16:Q3(E=Duplicates) that for all the ML models, AdaBoost, KNN and Decision Tree tend to have no negative impact.

- **Q5**: The discrepancies of datasets affect the ML performances largely as we observe from Table 4.16:Q5(E=Duplicates). For the datasets "Airbnb" and "Citation", the negative impacts could be dampened to zero if we add an auto ML/cleaning tuner. Yet for the datasets "Movie" and "Restaurant", the negative impacts even increase after we use an auto ML/cleaning tuner. This might be due to the protocol of generating the duplicates, where we have combined the two sources of datasets and do not correlate the duplication rate with the class distribution. Finally, "Movie" has a duplication rate of 40%, "Citation" 10%, "Airbnb" 10%, and "Restaurant" 10%.

4.15 Mislabels

1. Cleaning mislabels are highly likely to have positive impacts on ML.
2. Cleaning mislabels for models that have bad performances may produce negative impacts.

- **Q1:** Table 4.16:Q1(E=Mislabels) is generated by grouping flags, which demonstrates strong positive impacts of cleaning mislabels in all relations. The likelihood of improving ML model performances after cleaning mislabels (the "P" flag) is higher than that of insignificant changes (the "S" flag) , which is more likely than reducing the ML model performances (the "N" flag).

- **Q2**: In the scenario CD shown in Table 4.16:Q2(E=Mislabels), cleaning mislabels has always a higher likelihood of rendering a positive impact on the ML performances, which corresponds to the observation from Figure 4.1. This could be understood by the fact that using a clean training set and a clean test set, i.e., sets without label flipping, ML performances are generally better than using a clean training set and a corrupted test set. There are no negative flags in the scenario BD under the error type "Mislabel". We find out that if we clean the dirtiness in the training set, it is highly unlikely that the cleaning method has a negative impact on the ML model performance.

- **Q3**: Apart from Naive Bayes (Table 4.16:Q2(E=Mislabels)), all the other ML models have demonstrated a stronger tendency in producing more accurate predictions after cleaning the mislabels.

- **Q5:** We observe that the negative impact only occurs when the model has a bad performance (accuracy < 50%). This is because when we inject mislabels by flipping labels, we are more likely to flip a label that bad models predict incorrectly. Then accuracy of bad models is likely to be improved after injecting mislabels. Hence, cleaning mislabels may reduce the performance of bad models. With model selection, the negative impact is reduced, since we are less likely to have a bad model.

4.16 Outliers

1. Cleaning outliers is more likely to have insignificant and positive impacts, but it may produce negative impacts on ML.
2. With model selection, the probability of having negative impacts can be reduced. With cleaning method and model selection, it is unlikely to have negative impacts.
3. The probability of having positive and negative impacts is highly related to datasets and detection methods.

- **Q1:** Table 4.16:Q1(E=Outliers) shows the results of Q1 for outliers. The results of R1 indicate that cleaning outliers mostly have no impact or positive impact on ML, but sometimes it may negatively affect ML. This is because outlier detection and repair are not completely accurate. The detection methods are usually based on the assumptions of the error distribution, which may not be the underlying authentic distribution. Some outliers may be true data instead of errors although they are distant from other instances. Cleaning such data will distort the true distribution of the dataset, which results in negative impact on ML. The results of R2 and R3 show that with the model selection and with the cleaning method selection the percentage of flag "N" decreases to 0 and the percentage of "S" increases, while the flag "P" remains at the similar percentage. This indicates that using model selection and cleaning method selection can eliminate the negative impact of cleaning outliers and improve the robustness without losing too much benefit.

- **Q2:** Table 4.16:Q1(E=Outliers) shows the results of Q2 for outliers. In R1, R2 and R3, BD and CD have similar percentage scores of "P", "S" and "N". This indicates that cleaning outliers in the training and test sets have similar impacts on ML models.

- **Q3:** Table 4.16:Q3(E=Outliers) shows the results of Q3 for outliers. KNN has more "P" flags, less "N" and "S" flags than other models. Therefore, KNN is the most

sensitive model to outliers and gains most benefit from cleaning outliers. Other models are affected similarly.

- **Q4.1:** Table 4.16:Q4.1(E=Outliers) shows the results of Q4.1. In R1, IQR and IF have more "P" flags and "N" flags than the SD method. This indicates that IQR and IF are more aggressive than SD and SD is more conservative. In R2, the negative impact of IF and IQR is largely eliminated by model selection, but the positive impact remains.
- **Q4.2:** Table 4.16:Q4.2(E=Outliers) shows that there is no significant difference between repair methods in both R1 and R2.
- **Q5:** Table 4.16:Q5(E=Outliers) shows the result of Q5. In R1, most negative flags are from "Credit" and "KDD" datasets. In R2, all of negative flags are from "Credit". EEG and Sensor have more "P" flags than other datasets. This echoes our interpretation in Q1 that the impact of outliers on ML models largely depends on the error distribution in the dataset.

4.17 Missing Values

1. Cleaning missing values by imputation are more likely to improves the performance or achieves similar performance as deleting missing values.
2. With model selection and imputation method selection, ML models tend to be more robust to missing values.

As mentioned in section 4.8, we only consider one scenario (BD) for missing values. Therefore, we do not run Q2 for missing values.

- **Q1:** Table 4.16:Q1(E=Missing Values) shows the results of Q1 for missing values. The result of R1 exhibits that cleaning missing values by imputation mostly improves the performance or achieves similar performance as deleting missing values. But there are still few "N" flags, which indicates that imputation can sometimes hurt the performance. The reason is that imputation is simply an approximation of ground truth. If the imputation is distant from ground truth, it may introduce bias to data and reduce the performance of ML.

  The results of R2 and R3 show that, with model selection, "N" flags are eliminated and the percentage of "S" flags increases. There- fore, ML becomes more robust to missing values.

- **Q3:** Table 4.16:Q3(E=Missing Values) presents the results of Q3. Only Naive Bayes has "N" flags. Therefore, Naive Bayes is the most vulnerable model to missing value imputation. Other models have similar results.
- **Q4.2:** Table 4.16:Q4.2(E=Missing Values) shows the results for Q4.2. In both R1 and R2, different imputation methods have similar results. Therefore, different imputation methods have similar impacts.
- **Q5:** Table 4.16:Q5(E=Missing Values) shows the results of Q5. In R1, "USCensus" has much less "P" flags and more "N" flags than other datasets. The reason may be that the imputation in this dataset is distant from ground truth. All of flags in "KDD" are "P", which indicates that imputation in this dataset may be close to ground truth. R2 and R3 show that with the model selection the negative impact in "UScensus" caused by missing value imputation is eliminated.

4.18 Summary of Key Insights

Data cleaning does not necessarily improve the quality of downstream ML models. We see from the result analyses that applying cleaning methods blindly could negatively impact model performances. Cleaning methods could introduce biases and sometimes this bias is larger than the original bias:

1. It is unclear that if cleaning duplicates could bring either positive or negative impacts, defining a better duplicate injection protocol is key.
2. Cleaning outliers mostly have no impact or positive impact on ML, but some- times it may negatively affect ML. The effects are highly dependent on detection and repair techniques.
3. If the imputation of missing values is distant from ground truth, it may introduce bias to data and reduce the performance of ML.

Interpretation of the experimental results should take into the following factors: the errors and their distributions on the datasets (which are unknown), the correctness of the cleaning algorithms (which are also unknown without ground truth), and the internal structures of the ML models (which can be hard to interpret for some models). Since these factors are jointly at work, it could be hard to interpret the results:

1. Cleaning *inconsistency* in both training and test sets is unlikely to produce negative impacts on the ML model performance. Dataset choice and dirtiness in key features matter.

2. Dataset choices and how to inject duplicates when combining the datasets are crucial in the experiment setups of duplicates.

3. Class distribution has a huge impact on how mislabels should be cleaned.

Performing model selection can significantly increase the robustness of impacts of cleaning on ML. This effect has been identified in cleaning all error types. In particular, the negative effect of data cleaning can be eliminated by selecting the best ML model.

Performing cleaning algorithm selection further increases the robustness of impacts of cleaning on ML. This effect has been identified in cleaning all the error types. Since the data cleaning techniques are dependent of error distributions in datasets, no single cleaning algorithm is the best, and any future joint cleaning and ML work must devise "adaptive" cleaning solutions.

Before the conclusion, as we understand that only implementation of machine learning algorithms to datasets do not enough to increase data quality by itself. So that on next section, we are going to focused on the data quality verification methods.

4.19 Unit Tests' for Data

We generally write unit tests for our code, but do we also test your data? Incorrect or malformed data can have a large impact on production systems. Examples of data quality issues are:

- Missing values can lead to failures in production system that require non-null values (NullPointerException).

- Changes in the distribution of data can lead to unexpected outputs of machine learning models.

- Aggregations of incorrect data can lead to wrong business decisions.

In this section, we introduce Deequ [78], an open source tool developed and used at Amazon. Deequ [78] allows you to calculate data quality metrics on your dataset,define and verify data quality constraints, and be informed about changes in the data distribution. Instead of implementing checks and verification algorithms on your own, you can focus

on describing how your data should look. Deequ [78] supports you by suggesting checks for you. Deequ [78] is implemented on top of Apache Spark and is designed to scale with large datasets (think billions of rows) that typically live in a distributed filesystem or a data warehouse.

4.19.1 Deequ at Amazon

Deequ [78] is being used internally at Amazon for verifying the quality of many large production datasets. Dataset producers can add and edit data quality constraints. The system computes data quality metrics on a regular basis (with every new version of a dataset), verifies constraints defined by dataset producers, and publishes datasets to consumers in case of success. In error cases, dataset publication can be stopped, and producers are notified to take action. Data quality issues do not propagate to consumer data pipelines, reducing their blast radius.

4.19.2 Overview of Deequ

To use Deequ, let's look at its main components (Figure 4.2)

- **Metrics Computation** — Deequ computes data quality metrics, that is, statistics such as completeness, maximum, or correlation. Deequ uses Spark to read from sources such as Amazon S3, and to compute metrics through an optimized set of aggregation queries. You have direct access to the raw metrics computed on the data.
- **Constraint Verification** — As a user, you focus on defining a set of data quality constraints to be verified. Deequ [78] takes care of deriving the required set of metrics to be computed on the data. Deequ generates a data quality report, which contains the result of the constraint verification.
- **Constraint Suggestion** — You can choose to define your own custom data quality constraints, or use the automated constraint suggestion methods that profile the data to infer useful constraints.

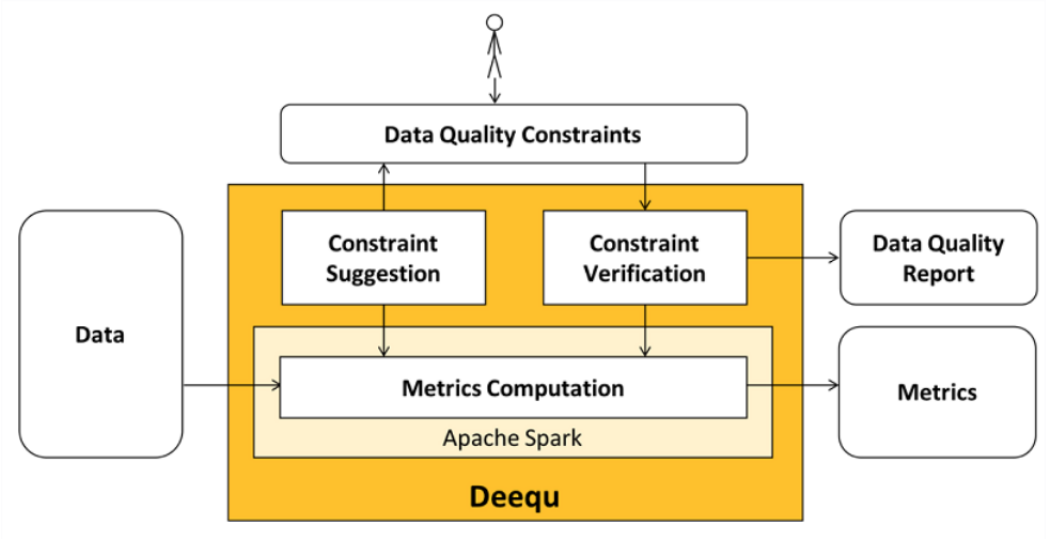

Fig. 4.2 Overview of Deequ components.

References

[1] Agrawal, D., Bernstein, P., Bertino, E., Davidson, S., Dayal, U., Franklin, M., Gehrke, J., Hass, L., and Han, J. (2015). Challenges and opportunities with big data. In *Challenges and Opportunities with Big Data*. A white paper prepared for the Computing Community Consortium committee of the Computing Research Association.

[2] Appuswamy, R., Gkantsidis, C., Narayanan, D., Hodson, O., and Rowstron, A. (2013). Scale-up vs scale-out for hadoop: Time to rethink? In editor, editor, *Proceedings of the 4th Annual Symposium on Cloud Computing*, page 20:1–20:13. ACM.

[3] Askham, N., Cook, D., Doyle, M., Fereday, H., Gibson, M., and Landbeck, U. (2013). *The six primary dimensions for data*. Data Management Association (DAMA), United Kingdom.

[4] Benjamin T. Hazen, Christopher A. Boone, J. D. E. and Jones-Farmer, L. A. (2014). Data quality for data science, predictive analytics, and big data in supply chain management: An introduction to the problem and suggestions for research and applications. *International Journal of Production Economics*, 154:72–80.

[5] Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society*, 57(1):289–300.

[6] Bonferroni, C. E. (1936). Teoria statistica delle classi e calcolo delle probabilità. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commerciali di Firenze*, 8:3–62.

[7] Cai, L. and Zhu, Y. (2015a). The challenges of data quality and data quality assessment in the big data era. *The Challenges of Data Quality and Data Quality Assessment in the Big Data Era*, page 14:2.

[8] Cai, L. and Zhu, Y. (May 2015b). The challenges of data quality and data quality assessment in the big data era. *Data Sci. J.*, 14:2.

[9] Chen, C. P. and Zhang, C.-Y. (2014a). Data-intensive applications, challenges, techniques and technologies: A survey on big data. *A survey on big data*, page 314–347.

[10] Chen, C. P. and Zhang, C.-Y. (2014b). Data-intensive applications, challenges, techniques and technologies: A survey on big data. *Information Sciences*, 275:314–347.

[11] Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Xgboost: A scalable tree boosting system*, page 785–794. The 22nd acm sigkdd international conference on knowledge discovery and data mining, ACM.

[12] Cormode, G. and Duffield, N. (2014). Sampling for big data: A tutorial. In *Sampling for Big Data: A Tutorial*, page 1975–1975, New York, NY, USA. 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

[13] Cox, M. and Ellsworth, D. (1997). Application-controlled demand paging for out-of-core visualization. *journal*.

[14] Cui, Z., Chen, W., He, Y., and Chen, Y. (2015). Optimal action extraction for random forests and boosted trees. In *Optimal action extraction for random forests and boosted trees*, page 179–188. 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM.

[15] Das, S., Doan, A., G. C., P. S., Gokhale, C., Konda, P., Govind, Y., and Paulsen, D. (year). The magellan data repository. https://sites.google.com/site/anhaidgroup/projects/data.

[16] Data, I. L. (2019). Sensor dataset. http://db.csail.mit.edu/labdata/labdata.html. Accessed: 2019-07-30.

[17] dataone.org (2016). Data observation network for earth. dentify and use relevant metadata standards. https://www.dataone.org/best-practices/identify-and-use-relevant-metadata-standards. Accessed: 2019-07-19.

[18] Deming, W. E. (1960). *Sample Design in Business Research*. ohn Wiley and Sons, Inc., New York.

[19] Dheeru, D. and Casey, G. (2017). UCI machine learning repository.

[20] Dictionaries, O. (2013). An oxford english dictionary update — oxfordwords blog. http://blog.oxforddictionaries.com/2013/06/oed-june-2013-update/.

[21] Diebold, F. X. (2012). A personal perspective on the origin(s) and development of big data. *IER Working Paper No. 13-003*.

[22] Eckerson, W. W. (2002). Data quality and the bottom line. achieving business success through a commitment to high quality data.

[23] Fayyad, U. M., Wierse, A., and Grinstein, G. G. (2002). *Information visualization in data mining and knowledge discovery*. Morgan Kaufmann.

[24] Firmani, D., Mecella, M., Scannapieco, M., and Batini, C. (2015). *On the Meaningfulness of Big Data Quality*. Springer Berlin Heidelberg.

[25] FUJITSU (2015). *White paper Solution Approaches for Big Data*. FUJITSU Technology Solutions GmbH.

[26] Gadepally, V., Herr, T., Johnson, L., Milechin, L., Milosavljevic, M., and Miller, B. A. (2015). Sampling operations on big data. In editor, editor, *Sampling operations on big data*, page 1515–1519. 49th Asilomar Conference on Signals, Systems and Computers.

[27] Garcia, S., Luengo, J., and Herrera, F. (2015). *Data preprocessing in data mining*. Springer.

[28] Group, A. (2019a). Restaurant dataset. https://sites.google.com/site/anhaidgroup/useful-stuff/data. Accessed: 2019-07-30.

[29] Group, A. (2019b). The tmdb 5000 movie dataset. https://www.kaggle.com/tmdb/tmdb-movie-metadata. Accessed: 2019-07-30.

[30] Guptil, C. and Morrison, J. (1995). *Elements of Spatial Data Quality.* Elsevier Science Ltd, Oxford, UK.

[31] Halevy, A., Rajaraman, A., and Ordille, J. (2006). Data integration: the teenage years. In *Data integration: the teenage years*, page 9–16, Endowment. In Proceedings of the 32nd international conference on Very large data bases.

[32] Hastie, T., Rosset, S., Zhu, J., and Zou, H. (2009). *Multi-class adaboost. Statistics and its Interface*, volume 2. International Press.

[33] Hastie, T. and Tibshirani, R. (1996). In advances in neural. *Discriminant adaptive nearest neighbor classification and regression*.

[34] Hatton, J. A. (2019). Company dataset. https://www.kaggle.com/jacksapper/company-sentiment-by-location. Accessed: 2019-07-30.

[35] I. Caballero, M. S. and Piattini, M. (2014). *A Data Quality in Use Model for Big Data*. Springer International Publishing.

[36] INRA (2019). Centre de recherche inra paca, provence-alpes-côte d'azur. http://institut.inra.fr/Organisation/Annuaire-des-sites/Agriculture/Vegetal/Sante-des-plantes/Centre-de-recherche-Inra-PACA-Provence-Alpes-Cote-d-Azur. Accessed: 2019-07-30.

[37] J. Friedman, T. H. and Tibshirani, R. (2001). *The elements of statistical learning*. Springer series in statistics.

[38] Jagadish, H., Gehrke, J., Labrindis, A., Papakonstantino, Y., Patel, J. M., Ramakrishnan, R., and Shahabi, C. (2014). Big data and its technical challenges.

[39] Jugulum, R. (2014). *Computing with High Quality Data*, page 1. John Wiley and Sons, Inc., New Jersey.

[40] Kaggle (2019a). The credit dataset. https://www.kaggle.com/c/GiveMeSomeCredit/data. Accessed: 2019-07-30.

[41] Kaggle (2019b). The kdd dataset. https://www.kaggle.com/c/kdd-cup-2014-predicting-excitement-at-donors-choose/data. Accessed: 2019-07-30.

[42] Kaggle (2019c). The titanic dataset. https://www.kaggle.com/upendr/titanic-machine-learning-from-disaster/data. Accessed: 2019-07-30.

[43] Karthik Kambatla, Giorgos Kollias, V. K. and Grama, A. (2014). Journal of parallel and distributed computing. *Trends in big data analytics*, 74:2561–2573.

[44] Kleiner, A., Talwalkar, A., Sarkar, P., and Jordan, M. (2012). The big data bootstrap.

[45] Krawczyk, H. and Wiszniewski, B. (2003). Visual gqm approach to quality-driven development of electronic documents. In *Visual GQM Approach to Quality-driven Development of Electronic Documents*, Edinburgh UK. 2nd International Workshop on Web Document Analysis.

[46] Lavalle, S., Lesser, E., Shockley, R., Hopkins, M. S., and Kruschwitz, N. (2011). Big data , analytics and the path from insights to value big data. *Analytics and the Path From Insights to Value*, 52205.

[47] Lenzerini, M. (2002). Data integration: A theoretical perspective. *Data integration: A theoretical perspective. In Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 23(4):233–246.

[48] Liang, F., Kim, J., and Song, Q. (2016). *A Bootstrap Metropolis-Hastings Algorithm for Bayesian Analysis of Big Data*. Technometrics.

[49] Magoulas, R. and Lorica, B. (2009). Big data: Technologies and techniques for large-scale data. *Big Data: Technologies and Techniques for Large-Scale Data*, 2.

[50] Marr, B. (2015). Why only one of the 5 vs of big data really matters — ibm big data and analytics hub. http://www.ibmbigdatahub.com/blog/why-only-one-5-vs-big-data-really-matters.

[51] Maydanchik, A. (2007). *Data quality assessment*. Bradley Beach, Technics Publications, New Jersey, USA.

[52] McFadden, D. (1973). *Conditional logit analysis of qualitative choice behavior*. University of California Berkeley.

[53] McGilvary, D. (2008). *Executing data quality projects: ten steps to quality data and trusted information*. Elsevier Inc.

[54] Meek, C., Thiesson, B., and Heckerman, D. (2019). UCI machine learning repository us census data (1990) data set.

[55] Michael J. Shaw, Chandrasekar Subramaniam, G. W. T. and Welge, M. E. (2001). Decision support systems. *Knowledge management and data mining for marketing*, 31:127–137.

[56] Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (2013). *Machine learning: An artificial intelligence approach*. Springer Science and Business Media.

[57] Microsoft (2016). Data identification. https://msdn.microsoft.com/en-us/library/aa291809(v=vs.71).aspx. Accessed: 2019-07-19.

[58] of Inspector General, G. O. (2014). United states postal service. undeliverable as addressed mail audit report. (MS-AR-14-006).

[59] Ohbyung Kwon, N. L. and Shin, B. (2014). Data quality management, data usage experience and acquisition intention of big data analytics. *International Journal of Information Management*, 34(3):387–394.

[60] Osborne, J. W. (2013). *Best practices in data cleaning: a complete guide to everything you need to do before and after collecting your data*. Thousand Oaks.

[61] Ostman, A. (1997). The specifications and evaluation of spatial data quality. In *18th ICA/ACI International Conference*, Stockholm, Sweden.

[62] Paton, N. (2016). Data wrangling. http://www.cs.manchester.ac.uk/study/postgraduate-research/programmes/phd/programme-details/projects/description/?projectid=6981. Accessed: 2019-07-19.

[63] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

[64] Pernici and Scannapieco, M. (2003). Data quality in web information systems. *Journal of Data Semantics*.

[65] Pipino, L., Lee, Y. W., Wang, R. Y., Lowell, W., Lee, Y., and Yang, R. Y. (2002). Data quality assessment. *Data Quality Assessment.*, 45(4):211–218.

[66] Platform, I. A. T. (2018). Extracting business value from the 4 v's of big data. https://www.ibmbigdatahub.com/infographic/four-vs-big-data. Accessed: 2019-07-19.

[67] Quinlan, J. R. (1986). Induction of decision trees. *Induction of decision trees*, (1):81–106.

[68] R. L. Wasserstein, N. A. L. (2016). The asas statement on p-values: context, process, and purpose. *The asas statement on p-values: context, process, and purpose*, 70(2):129–133.

[69] Rahm, E. and Do, H. H. (2000). Data cleaning: Problems and current approaches. *Data cleaning: Problems and current approaches*, 23(4):3–13.

[70] Ralph Kimball, Laura Reeves, M. R. and Thornthwaite, W. (2008). *The Data Warehouse Lifecycle Toolkit*. Wiley.

[71] Redman, T. C. (2018). If your data is bad, your machine learning tools are useless. https://hbr.org/2018/04/if-your-data-is-bad-your-machine-learning-tools-are-useless. Accessed: 2019-07-19.

[72] Rish, I. (2001). An empirical study of the naive bayes classifier. In *An empirical study of the naive bayes classifier*, volume 3, page 41–46. IJCAI 2001 workshop on empirical methods in artificial intelligence, IBM.

[73] Rouse, M. (2005). What is data quality? http://searchdatamanagement.techtarget.com/definition/data-quality. Accessed: 2019-07-19.

[74] Rupert, G. (2012). *Simultaneous statistical inference*. Springer Science and Business Media.

[75] S, J. (2015). Overview of data quality challenges in the context of big data. In *Overview of data quality challenges in the context of Big Data*, pages 1–9. International Conference on Computing Communication and Security.

[76] Sandra de F. Mendes Sampaio, C. D. and Sampaio, P. R. F. (2015). *DQ2S - A framework for data quality-aware information management*. Elsevier.

[77] Satyanarayana, A. (2014). Intelligent sampling for big data using bootstrap sampling and chebyshev inequality. In *Intelligent sampling for big data using bootstrap sampling and chebyshev inequality*, pages 1–6. 27th Canadian Conference on Electrical and Computer Engineering (CCECE).

[78] Schelter, S., Lange, D., Schmidt, P., Celikel, M., Biessmann, F., and Grafberger, A. (2018). Automating large-scale data quality verification. *Proc. VLDB Endow.*, 11(12):1781–1794.

[79] Souders, S. (2019). UCI machine learning repository.

[80] Stone, L. (1979). The revival of narrative: Reflections on a new old history.past and present. *The Revival of Narrative*, 85(85):3–24.

[81] Tekiner, F. and Keane, J. A. (2013). Big data framework. In *Big data framework*, page 1494–1499. EEE International Conference on Systems, Man, and Cybernetics, IEEE.

[82] Tilly, C. (1980). The old new social history and the new old social history. *CRSO Working Paper No. 218*.

[83] ud-din Khan, M. A., Uddin, M. F., and Gupta, N. (2014). Seven v's of big data understanding big data to extract value. In *Seven V's of Big Data understanding Big Data to extract value*, page 1–5. American Society for Engineering Education (ASEE Zone 1), 2014 Zone 1 Conference of the, 2014.

[84] V. Goasdoué, S. Nugier, D. D. and Laboisse, B. (2007). An evaluation framework for data quality tools. *An Evaluation Framework For Data Quality Tools*, page 280–294.

[85] Verborgh, R. and Wilde, M. D. (2013). *Using OpenRefine*. Packt Publishing Ltd.

[86] Wang and Strong, D. M. (1996). Beyond accuracy: What data quality means to data consumers. *Journal of Management Information Systems*, page 12.

[87] Ware, C. (2012). *Information visualization: perception for design*. Elsevier.

[88] Weiss, S. M. and Indurkhya, N. (1998). *Predictive Data Mining: A Practical Guide*. Morgan Kaufmann Publishers Inc.

[89] Wiszniewski, B. and Krawczyk, H. (2003). Digital document life cycle development. In *1st International Symposium on Information and Communica- tion Technologies*, Dublin Ireland.

[90] World, D. (2019). The imdb 5000 movie dataset. https://data.world/popculture/imdb-5000-movie-dataset. Accessed: 2019-07-30.

[91] Zhao, Z., Stefani, L. D., Zgraggen, E., Binnig, C., Upfal, E., and Kraska, T. (2017). Controlling false discoveries during interactive data exploration. In *Controlling false discoveries during interactive data exploration*, page 527–540. In Proceedings of the 2017 ACM International Conference on Management of Data, ACM.

Appendix A

How to use NettoyageML

This is the EISTI.NettoyageML Benchmark for Joint Data Cleaning and Machine Learning. The codebase is located in: https://github.com/aytacozkan/EISTI.NettoyageML

A.1 Basic Usage

Run Experiments

To run experiments, download and unzip the datasets. Place it under the project home directory and execute the following command from the project home directory:

```
1    python3 main.py --run_experiments [--dataset <name>] \
2    \[--cpu <num_cpu>][--log]
```

Options

- **–dataset:** the experiment dataset. If not specified, the program will run experiments on all datasets.
- **–cpu:** the number of cpu used for experiment. Default is 1.
- **–log:** whether to log experiment process.

Output

The experimental results for each dataset will be saved in ‘/result‘ directory as a json file named as <dataset name>_result.json. Each result is a key-value pair. The key is a string in format "<dataset><split seed><error type><clean method><ML model><random search

seed>". The value is a set of key-value pairs for each evaluation metric and result. Our experimental results are provided in ‘result.zip‘.

Run Analysis

To run analysis for populating relations described in the paper, unzip **‘result.zip‘** and execute the following command from the project home directory:

```
python3 main.py --run_analysis [--alpha <value>]
```

Options

- **–alpha:** the significance level for multiple hypothesis test. Default is 0.05.

Output

The relations R1, R2 and R3 will be saved in **‘/analysis‘** directory. Our analysis results are provided in ‘analysis.zip‘.

A.2 Extend Domain of Attributes

Add new datasets

To add a new dataset, first, create a new folder with dataset name under **‘/data‘** and create a ‘raw‘ folder under the new folder. The ‘raw‘ folder must contain raw data named **‘raw.csv‘**. For dataset with inconsistencies, it must also contain the inconsistency-cleaned version data named **‘inconsistency_clean_raw.csv‘**. For dataset with mislabels, it must also contain the mislabel-cleaned version data named **‘mislabel_clean_raw.csv‘**. The structure of the directory looks like:

```
/
└── data
    └── new_dataset
        └── raw
            ├── raw.csv
            ├── inconsistency_clean_raw.csv (for dataset with inconsistencies)
            └── mislabel_clean_raw.csv (for dataset with mislabels)
```

Then add a dictionary to **‘/schema/dataset.py‘** and append it to **‘datasets‘** array at the end of the file.

The new dictionary must contain the following keys:

```
data_dir: the name of the dataset.
error_types: a list of error types that the dataset contains.
label: the label of ML task.
```

The following keys are optional:

```
class_imbalance: whether the dataset is class imbalanced.
categorical_variables: a list of categorical attributes.
text_variables: a list of text attributes.
key_columns: a list of key columns used for deduplication.
drop_variables: a list of irrelevant attributes.
```

Add new error types

To add a new error type, add a dictionary to '/schema/error_type.py' and append it to 'error_types' array at the end of the file.

The new dictionary must contain the following keys:

```
name: the name of the error type.
cleaning_methods: a dictionary, {cleaning method name: cleaning
    methods object}.
```

Add new models

To add a new ML model, add a dictionary to '/schema/model.py' and append it to 'models' array at the end of the file.

The new dictionary must contain the following keys:

```
name: the name of the model.
fn: the function of the model.
fixed_params: parameters not to be tuned.
hyperparams: the hyperparameter to be tuned.
hyperparams_type: the type of hyperparameter "real" or "int".
hyperparams_range: range of search. Use log base for real type
    hyperparameters.
```

Add new cleaning methods

To add a new cleaning methods, add a class to '/schema/cleaning_method.py'.
The class must contain two methods:

‘fit(dataset, dirty_train)‘: take in the dataset dictionary and dirty training set. Compute statistics or train models on training set for data cleaning.

‘clean(dirty_train, dirty_test)‘: take in the dirty training set and dirty test set. Clean the error in the training set and test set. Return **‘(clean_train, indicator_train, clean_test, indicator_test)‘**, which are the clean version datasets and indicators that indicate the location of error.

Add new scenarios

We consider "BD" and "CD" scenarios in our paper. To investigate other scenarios, add scenarios to ‘/schema/scenario.py‘.

Appendix B

Installing the CUED class file

LaTeX.cls files can be accessed system-wide when they are placed in the <texmf>/tex/latex directory, where <texmf> is the root directory of the user's TeXinstallation. On systems that have a local texmf tree (<texmflocal>), which may be named "texmf-local" or "localtexmf", it may be advisable to install packages in <texmflocal>, rather than <texmf> as the contents of the former, unlike that of the latter, are preserved after the LaTeXsystem is reinstalled and/or upgraded.

It is recommended that the user create a subdirectory <texmf>/tex/latex/CUED for all CUED related LaTeXclass and package files. On some LaTeXsystems, the directory look-up tables will need to be refreshed after making additions or deletions to the system files. For TeXLive systems this is accomplished via executing "texhash" as root. MIKTeXusers can run "initexmf -u" to accomplish the same thing.

Users not willing or able to install the files system-wide can install them in their personal directories, but will then have to provide the path (full or relative) in addition to the filename when referring to them in LaTeX.