Data Quality and Explainable AI

Leopoldo Bertossi[1] and Floris Geerts[2]

[1]Univ. Adolfo Ibáñez (Santiago, Chile) and RelationalAI Inc. (Toronto, Canada)
[2]University of Antwerp, Belgium

Abstract

In this work we provide some insights and develop some ideas, with few technical details, about the role of explanations in Data Quality in the context of data-based machine learning models (ML). In this direction there are, as expected, roles for causality and *explainable AI*. The latter area not only sheds light on the models, but also on the data that support model construction. There is also room for defining, identifying and explaining errors in data, in particular, in ML; and also for suggesting repair actions. More generally, explanations can be used as a basis for defining dirty data in the context of ML, and measuring or quantifying them. We think dirtiness as relative to the ML task at hand, e.g. classification.

1 Introduction

In this short paper we aim to bridge recent work on explanations in data quality with explainable AI. We start by describing a number of approaches for finding explanations of query results. For example, a query may extract potential inconsistencies or other forms of dirtiness in the data and one wishes to find explanations for the inconsistencies. The explanations correspond to typical questions like "*what data is dirty?*", "*why is it dirty?*", and "*how does a particular piece of data contribute to the overall dirtiness?*". As such, they explain query results at an increasing level of granularity.

In view of the rise of AI/machine learning methods to analyse data, the above explanation methods need to be revisited. Indeed, the query now becomes a more complex data analytical task. We remark that this paper is not a comprehensive survey. Instead our focus is to provide a bit of insight into this general problem and to identify a couple of promising research directions.

2 Inspecting data by queries

It is common that the cleanliness of a database is directly assessed on the basis of the database itself. One can directly inspect the data, at the record (tuple) or record value level. A more systematic way to do this inspection is by posing *queries* to the database; and even more sophisticated, one can define appropriate views that capture data with some "issues". Again, one can directly inspect the query results or the view contents. A good example of this is *consistency*, one particular dimension of data quality [2, 23].

Example 1. Consider the following database, D, below, and suppose that we expect D to be consistent in that it satisfies the *denial integrity constraint* ψ: $\neg\exists x\exists y(S(x) \wedge R(x,y) \wedge S(y))$, that prohibits a particular kind of joins.

R	A	B
	a	b
	c	d
	b	b

S	A
	a
	c
	b

To check whether this constraint holds, one can pose the query $\mathcal{Q}(x,y)$: $S(x) \wedge R(x,y) \wedge S(y)$, associated with ψ, to D. The query result

$\mathcal{Q}(D)$ will consist of all combinations of values ψ.
that do satisfy the join, i.e. inconsistencies for

In this case, the set of answers $\mathcal{Q}(D)$, which should be empty were the database consistent, is $\{\langle a, b\rangle, \langle b, b\rangle\}$. ■

This is an example of a particular and classical kind of dirtiness that is related to an also classical kind of integrity constraint. Other kinds of constraints have been proposed and investigated that capture other forms of dirtiness or can be used to address other dimensions of data quality [19].

As another example, but this time along the *redundancy dimension* of data quality, we can give an example related to duplicate detection, which is about identifying pairs (or sets) of records in a database that represent the same external entity.

Example 2. Consider a database which consists of records of the form $R(Name, Address, DOB, Occup)$. In the database there should not be two different records with same *Name* and *DOB* (date of birth), and similar *Address*. We can find such undesired pairs of records by means of the query:

$$\begin{aligned} \mathcal{Q}:\ & R(Name, Address_1, DOB, Occup_1) \wedge R(Name, Address_2, DOB, Occup_2) \\ & \wedge\ Address_1 \approx Address_2 \wedge\ (Address_1 \neq Address_2\ \vee\ Occup_1 \neq Occup_2)), \end{aligned} \tag{1}$$

that involves a built-in, application dependent similarity relation $\approx$, which is assumed to subsume equality.

Similarly as in Example 1, the query $\mathcal{Q}$ corresponds to some kind of data quality constraint. Indeed, it is associated to a *matching dependency* (MD), a relatively new kind of constraint proposed for duplicate detection and reconciliation, the latter being the process of fusing duplicates into single records [20, 7]. In this case, the MD corresponding to $\mathcal{Q}$ would be the (implicitly universally quantified) formula:

$$\begin{aligned} & R(Name, Address_1, DOB, Occup_1) \wedge R(Name, Address_2, DOB, Occup_2) \\ & \wedge\ Address_1 \approx Address_2 \rightarrow\ Address_1 \doteq_1 Address_2 \wedge Occup_1 \doteq_2 Occup_2, \end{aligned} \tag{2}$$

which specifies that for any two records that share the same name and date of birth and have similar addresses, the addresses and the occupations have to be made identical. This requirement is application dependent. In this case, an application where the attributes $Name, DOB, Address$ are expected to act as a key, but with only similarity required for *Address* instead of the usual equality used in key constraints. Here, the way to make values identical is built into the operators $\doteq_1$ and $\doteq_2$, and is application dependent.

We can see that the first three atoms of query $\mathcal{Q}$ in (1) detect records that satisfy the left-hand side of the constraint in (2). The disjunction in (1) captures the records that do not satisfy one (or both) of the equalities required by the right-hand side of the MD in (2). ■

What we just described is a simply way to detect inconsistencies in the data and to answer "*what is dirty?*". One often wants to go beyond simple queries to detect inconsistencies (or other forms of dirtiness), and go deeper into the *causes*. That is, one wants the identify the root causes of the reported inconsistencies, e.g. at the tuple level. This is the role of causality in databases.

3 Causality in Databases

Causality in databases was first introduced in [31], inspired by [21] (c.f. also [32]). We use an example to give the intuition.

Example 3. (ex. 1 cont.) With the same database D, consider the existential closure of the original query, i.e. the Boolean conjunctive query: $\mathcal{Q}$: $\exists x \exists y (S(x) \wedge R(x,y) \wedge S(y))$, which happens to be true in D, denoted $D \models \mathcal{Q}$. We ask "*why?*", and we want the causes, as explanations for $\mathcal{Q}$ being true in D.

A tuple $t \in D$ (that is a *fact* or, equivalently, a row in a relation in D) is a *counterfactual cause* for $\mathcal{Q}$ in D if $D \models \mathcal{Q}$ and $D \smallsetminus \{t\} \not\models \mathcal{Q}$. Here, $S(b)$ is counterfactual cause for $\mathcal{Q}$ in D: if $S(b)$ is removed from D, then $\mathcal{Q}$ is no longer true (and then, no inconsistencies remain in D).

A tuple $t \in D$ is an *actual cause* for $\mathcal{Q}$ if there is a *contingency set* $\Gamma \subseteq D$, such that t is a counterfactual cause for $\mathcal{Q}$ in $D \smallsetminus \Gamma$. Here, $R(a,b)$ is an actual cause for $\mathcal{Q}$ with contingency set $\{R(b,b)\}$: if $R(a,b)$ is removed from D, then $\mathcal{Q}$ is still true, but further removing $R(b,b)$ makes $\mathcal{Q}$ false. It indicates that $R(a,b)$ is less of a good cause than $S(b)$ for the inconsistencies in D. Every counterfactual cause is also an actual cause, with empty contingent set, but actual, but non-counterfactual, causes need company to invalidate a query result.

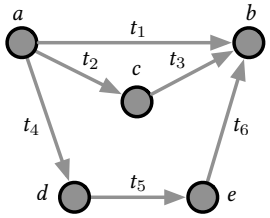
A related question is "*Can we quantify how good causes are?*". An answer comes from the notion of responsibility [13]. The *responsibility* of an actual cause t for $\mathcal{Q}$ is $\rho_D(t) := \frac{1}{|\Gamma| + 1}$, with $|\Gamma|$ the size of smallest contingency set for t. In this example, the responsibility of $R(a,b)$ is $\frac{1}{2} = \frac{1}{1+1}$ (its several smallest contingency sets have all size 1). Hence, $R(b,b)$ and $S(a)$ are also actual causes with responsibility $\frac{1}{2}$. We note that $S(b)$ is an actual (counterfactual) cause with responsibility $1 = \frac{1}{1+0}$. This quantitatively validates our earlier observation that $S(b)$ is more causal than $R(a,b)$ for the inconsistencies in D. ■

We can see that causality gives us an additional knowledge on the reasons why a query is true, or similarly, an integrity constraint is violated. In the end, we gain insight into the underlying data, and their quality.

In particular, high responsibility tuples provide more interesting explanations. Causes in this case are tuples that come with their responsibilities as "scores". Actually, for a particular data phenomenon, e.g. a query result, all tuples can be seen as actual causes and only the scores matter. This idea of assigning scores to tuples (or more fine-grained, to tuple values [5]), can be taken much further. Other score functions could be applied in situations where others do not provide intuitive results, as the following example shows.

Example 4. The following Boolean Datalog query Π (right) becomes true on database E (left) if there is a path from a to b. We can alternatively interpret this by saying that E is consistent if no path between a and b exists.

E	X	Y
t_1	a	b
t_2	a	c
t_3	c	b
t_4	a	d
t_5	d	e
t_6	e	b



$$\begin{aligned} yes &\leftarrow P(a,b) \\ P(x,y) &\leftarrow E(x,y) \\ P(x,y) &\leftarrow P(x,z), E(z,y) \end{aligned}$$

One can verify that all tuples are actual causes for the answer "*yes*" since every tuple appears in a path from a to b. However, all the tuples have the same causal responsibility of $\frac{1}{3}$, which may be counter-intuitive. Indeed, t_1 provides a direct path between a and b, and one would expect t_1 to be more responsible than the other tuples that are only involved in indirect paths between a and b. ■

In [37], an alternative, quantitative notion of *causal effect* was introduced. This is done by transforming the database into a probabilistic one [40], where tuples have independent probabilities of $\frac{1}{2}$ of being true; and performing counterfactual interventions on it. The causal effect is then the difference of the expected values of the query being true and false. The causal effect turned out to give much more intuitive results. In the previous example, the causal effect for tuple t_1 is 0.65625, for tuples t_2, t_3 it is 0.21875, and for tuples t_4, t_5, t_6 it is 0.09375. Interestingly, it was shown later [29] that the causal effect coincides with the *Banzhaf power index* [18], which is defined in a similar way as the *Shapley value*, a well known measure used in coalition game theory.

4 Coalition Games and Shapley

The use of the Shapley value to measure how much a database tuple contributes to a query answer from a database has been recently investigated in [29]. In this situation, several tuples together are necessary to produce a query result. Like *players in a coalition game*, some may contribute more than others. The query (Boolean or aggregate numerical) becomes the *wealth-distribution or game function*, $\mathcal{G}$, that sends subsets of players to values $v \in \mathbb{R}$. Since the players are the database tuples, the set of players is the database D. We can apply standard measures of players' contributions that are used in game theory, economics, etc. One established measure is the Shapley value of a player. It is based on counterfactual interventions, of the kind *"What would happen if we change ...?"*.

More precisely, the Shapley value of player p (i.e. a tuple) among a set of players D (i.e. the database) is given by:

$$\mathit{Shapley}(D, \mathcal{G}, p) := \sum_{S \subseteq D \setminus \{p\}} \frac{|S|!(|D| - |S| - 1)!}{|D|!} (\mathcal{G}(S \cup \{p\}) - \mathcal{G}(S)),$$

where $|S|!(|D| - |S| - 1)!$ is the number of permutations of D with all players in S coming first, then p, and then all the others. This is an average of the game-function differences between a set having player p and not having it, over all sets of players. Again, a form of local counterfactual intervention.

Example 5. (ex. 4 cont.) It can be verified (see [29] for details) that $\mathit{Shapley}(E, \mathcal{G}, t_1) = \frac{35}{60}$, $\mathit{Shapley}(E, \mathcal{G}, t_2) = \mathit{Shapley}(E, \mathcal{G}, t_3) = \frac{8}{60}$, and $\mathit{Shapley}(E, \mathcal{G}, t_4) = \mathit{Shapley}(E, \mathcal{G}, t_5) = \mathit{Shapley}(E, \mathcal{G}, t_6) = \frac{3}{60}$. Hence, although these values differ from what was obtained for causal effect, the ordering on the tuples induced by the Shapley value and causal effect coincide. They thus rank the tuples in the same way in accordance to their relevance for the Boolean query Π.■

The Shapley value provides a more sophisticated measure of the contributions of database tuples to a result from the database. With it and other scores, we can shed some light on the underlying data; and from the combination of results and scores, we can assess if unexpected results are being obtained from the data, and then, their quality. This idea becomes even more critical if the result is something much more complex than a query result, e.g. the outcome from a machine learning model that has been built using the data, and is being applied to those data.

5 Explainable AI/ML

We next highlight a couple of interesting research directions, all related to finding explanations when queries are replaced with complex data analytical tasks. We here illustrate the problems in the context of machine learning and data quality.

When it comes to machine learning and data quality, one can either use ML to assess the quality of data, or assess the quality of data for the ML task at hand.

Indeed, integrity constraints are often used in combination with more complex methods to assess the quality of data, including machine learning tasks [22]. A typical example is when each tuple t in a database D is represented by features $F(t)$ with $F \in \mathcal{F}$, some set of features. This can be followed by a classification task with the aim to label, with 0 (dirty) or 1 (clean), a tuple t based on all its features $\mathcal{F}(t)$. As shown in Figure 1, the classification model can be a black-box model (left in the figure), or an explicit, open-box model, e.g. a classification tree (right), that returns label $L(t) = 0$, for "dirty tuple".

In a similar vein, ML methods can be used to predict the best repair actions, e.g. to make the database consistent [28]. In other words, ML is used to assess the quality of data, and as explanation, we may ask about the feature of t that contributes the most to the outcome $L(t)$ of t being dirty or clean.

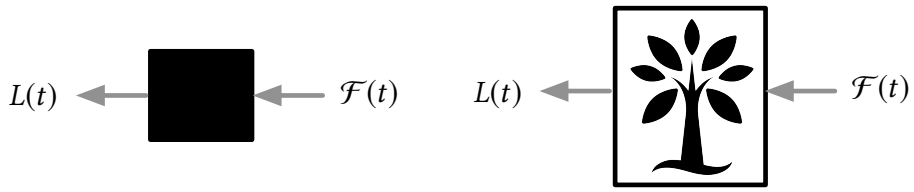


Figure 1: Classification models

In a different, but not unrelated direction, we can ask for explanations of outcomes of general ML tasks. For example, in a typical application, $\mathcal{F}(t)$ in Figure 1 is a record of feature values, e.g. for *income*, *age*, *address*, *job*, etc., that represent an entity that is applying for a loan at a financial institution, and the classifier returns 1 if the applicant is risky, and 0 if not (and the loan should be granted). We thus want to assess the quality of data for the ML task at hand by finding explanations for unexpected outcomes.

We would like to find explanations in these settings, as captured by *numerical scores* assigned to them. The question here is what kind of scoring functions can be used.

As it turns out, the Shapley value has already been used with black-box models [30]. Different scores can be applied to investigate the relevance of the feature values for this decision, in particular, the *Shapley-score* [8]. If we obtain "strange" scores, or, indirectly, rankings of feature values, we may want to inspect more deeply the data, the model, or both.

For the Shapley-score, the players are the features $F \in \mathcal{F}$, and the game function that depends on t (and then, on its feature values) becomes, for $S \subseteq \mathcal{F}$: $\mathcal{G}_t(S) := \mathbb{E}(L(t') \mid \mathcal{F}_S(t') = \mathcal{F}_S(t))$, where $\mathcal{F}_S$ are features in S. This reflects the fact that feature values outside S are counterfactually intervened in all possible ways. Thus, $\mathcal{G}_t(S)$ is the expected value of the label over tuples that coincide with t on features in S, and can be estimated via the empirical distribution of the available data. So, for a feature $F \in \mathcal{F}$, one computes: $\mathit{Shapley}(\mathcal{F}(t), \mathcal{G}_t, F(t))$. This is just one example, but more research is required to properly understand which score-functions are good in this context.

We also point out that score-based explanations for outcomes from black-box ML models can also be applied with open-box models, using only the input/output relation. In some applications, it could be more appealing and more informative to use score-based, and ranking-based explanation methodology that does use the internal components of the model. An example of such a methodology is given in, e.g. [12]. C.f. [35] for thoughts on explanations and black box models.

Explanations come in different forms, and are at the center of one of the most effervescent areas of research in AI and data science.

6 Ontological Contexts and Counterfactuals

There is the general agreement that data quality is context dependent [4]. On this basis, and to assess quality of data, and also extract quality data from a possible dirty source, contexts formalized as ontologies have been proposed [3].

In more general terms, ontologies that describe the domain the data are about can be useful when finding explanations for results obtained from the data source. This is intuitively clear, because we have more elements, actually declarative ones, that can be used when looking for explanations [14, 17, 9, 10].

Furthermore, ontologies could be handy when performing counterfactual interventions to obtain causes or compute score-based explanations. As a particular case of the former, causes for query answers in the presence of integrity constraints were investigated in [6]. In relation to counterfactual updates under ontologies, we give a simple example, modified from [15].

Example 6. A moving company makes hiring decisions based on feature values that become entries in records representing applicants, say $R = \langle \mathit{appCode}, \mathit{ability\ to\ lift}, \mathit{gender}, \mathit{weight}, \mathit{height}, \mathit{age} \rangle$.

Mary, represented by the record $R^\star = \langle 101, 1, F, \textit{160 pounds}, \textit{6 feet}, 28\rangle$, applies and is denied the job. That is, the classification returns: $L(R^\star) = 1$. To explain the decision, we can hypothetically change Mary's gender, from F into M, obtaining record $R^{\star\prime}$, for which we now observe $L(R^{\star\prime}) = 0$. Thus, *gender*, or better its value F, can be seen as a counterfactual explanation for the initial decision.

There are other alternatives we might consider, e.g. keeping the value of *gender*, counterfactually change the other feature values, to see if the original decision stays the same. If this is the case, there would be evidence that the value F for *gender* is relevant for the decision made.

However, when considering counterfactual interventions we might be constrained or guided by an ontology containing, e.g. a denial semantic constraint, such as $\neg(R[2] = 1 \wedge R[6] > 80)$, with 2 and 6 indicating position values in the record, that prohibits someone over 80 to be qualified as fit to lift. We could also have a rule, such as $(R[3] = M \wedge R[4] > 100 \wedge R[6] < 70) \rightarrow R[3] = 1$, specifying that men who weight over 100 pounds and are younger than 70 are automatically qualified to lift weight. ■

This example shows that counterfactual interventions can be affected by ontological knowledge that has to be brought into the game. Not every counterfactual intervention (or combination of them) may be admissible, and some may automatically generate additional interventions. In other words, counterfactual interventions and associated scores have to be *semantically correct*, in the sense that they comply with the imposed integrity constraints. Furthermore, the ontological knowledge has to be consistent with the data at hand. It may not make much sense imposing conditions on the explanations if they are not satisfied by the underlying data and the model trained with them. This is an area that deserves much more investigation.

More speculatively, there are situations where high score explanations may not be very useful. For example, a loan applicant who has been rejected might want to know *what to do* to revert the decision. If *age* has the highest score, there is not much to do, but maybe a *a combination* of other features could be more useful, e.g. changing *address* and *job*. This would benefit from a declarative and usable specification of preferences about what could be useful counterfactuals to consider, and extending scores from single features to admissible combinations thereof [8].

7 Final Remarks

We have argued that explanation methodologies need further investigation, especially when they are to explain outcomes of complex data analytical tasks. When decisions or predictions are made based on features, as is common practice in ML, we described explanation methodologies based on these features. One can go, however, one step further. More precisely, it is often the case that the features originate from queries posed to an underlying database, as is illustrated in the following example.

Example 7. Consider the following two queries written as rules (taken from [26]):

$$\pi_1(t) \leftarrow \textit{TxnInfo}(t, n, c, s), \textit{Card}(n, c, s) \quad \pi_2(t) \leftarrow \textit{TxnInfo}(t, n, c, s), \textit{Card}(n, c, s')$$

over relations $\textit{TxnInfo}(\textit{txn}, \textit{card}, \textit{country}, \textit{state})$ and $\textit{Card}(\textit{card}, \textit{country}, \textit{state})$ representing transaction and credit card information. When posed over a database D, these queries correspond to two Boolean features of transactions t: $F_i(t) = 1$ if $t \in \pi_i(D)$ and $F_i(t) = 0$ if $t \notin \pi_i(D)$, for $i = 1, 2$. These correspond to whether or not a transaction t happened in the same state or not. These features can, e.g. be used to predict credit-card fraud. ■

Explanations should then be sought in the underlying data, i.e. tuples in the database, from which the features are derived, rather than on the feature level. This leads to an interesting question of how to combine the explanation methodologies for queries, which we discussed earlier, with explanation methodologies for ML. It is likely that a Shapley-value approach can be applied here as well, but this needs further investigation.

Moreover, the design of explanation methods for query-derived features should probably go hand-in-hand with *in-database learning* methods. In such methods, ML tasks are performed *without* an explicit materialisation of the features [24]. This may bring additional benefits in terms of computational aspects of explanation methods.

We also like to point out that it is often desirable to provide declarative descriptions of explanations, rather than explanations based on data alone. In the context of data quality, one can consider learning succinct declarative explanations of collections of inconsistencies [11] or of user-made repairs of the data [33]. An interesting line of research is to develop similar techniques, alongside scoring-based techniques, in the ML context described earlier.

If the explanation methodology is well-established and properly conceived, then the provided explanations say something about the quality of the data itself, and the quality of data we used to train the model, specially if we receive unexpected explanations. However, in these cases, the quality dimension of data seems to be most naturally related to statistical concerns, such as *bias* [38, 36], or to the (also data fed) algorithm at hand, whose outcomes can also be assessed from the point of view of *fairness* [27, 39]. More research is needed here to identify good score-based explanation methods for such statistical aspects.

It is well-known that classic aspects of data quality are critical for machine learning and data-based AI: good quality data are necessary to learn and create the right models. It is also clear that ML has an important role to play in data quality. There are many ML-based solutions to data quality problems, e.g. entity resolution [25, 16, 1]. In this article we have shown how ML and AI bring to the table new dimensions of data quality, and how techniques from explainable AI can help in this direction.

References

[1] Bahmani, Z., Bertossi, L. and Nikolaos Vasiloglou, N. ERBlox: Combining Matching Dependencies with Machine Learning for Entity Resolution. *Int. J. Approx. Reason.*, 2017, 83:118-141.

[2] Batini, C. and Scannapieco, M. *Data Quality: Concepts, Methodologies and Techniques*. Second edition, Springer, 2016.

[3] Bertossi, L. and Milani, M. Ontological Multidimensional Data Models and Contextual Data Quality. *Journal of Data and Information Quality*. 2018, 9(3):14.1-14.36.

[4] Bertossi, L., Rizzolo, F. and Lei, J. Data Quality is Context Dependent. In *Proc. of the Workshop on Enabling Real-Time Business Intelligence (BIRTE) Collocated with the International Conference on Very Large Data Bases (VLDB)*, Springer LNBIP 84, 2011, pp. 52-67.

[5] Bertossi, L. and Salimi, B. From Causes for Database Queries to Repairs and Model-Based Diagnosis and Back. *Theory of Computing Systems*, 2017, 61(1):191-232.

[6] Bertossi, L. and Salimi, B. Causes for Query Answers from Databases: Datalog Abduction, View-Updates, and Integrity Constraints. *International Journal of Approximate Reasoning*, 2017, 90:226-252.

[7] Bertossi, L., Kolahi, S. and Lakshmanan, L. Data Cleaning and Query Answering with Matching Dependencies and Matching Functions. *Theory of Computing Systems*, 2013, 52(3):441-482.

[8] Bertossi, L., Li, J., Schleich, M, Suciu, D. and Vagena, Z. Experimenting with Score-Based Explanations for Classification Outcomes. Forthcoming.

[9] Calvanese, D., Ortiz, M., Simkus, M. and Stefanoni, G. Reasoning about Explanations for Negative Query Answers in DL-Lite. *Journal of Artificial Intelligence Research*, 2013, 48:635-669.

[10] Calvanese, D., Lanti, D., Ozaki, A., Peñaloza, R. and Xiao, G. Enriching Ontology-based Data Access with Provenance. Proc. IJCAI, 2019.

[11] Chalamalla, A., Ilyas, I.F.,Ouzzani, M. and Papotti, P. Descriptive and Prescriptive Data Cleaning. Proc. SIGMOD, 2017.

[12] Chen, C., Lin, K., Rudin, C., Shaposhnik, Y. and Wang, S. and Wang, T. An Interpretable Model with Globally Consistent Explanations for Credit Risk. Proc. NIPS 2018 Workshop on Challenges and Opportunities for AI in Financial Services: the Impact of Fairness, Explainability, Accuracy, and Privacy.

[13] Chockler, H. and Halpern, J. Y. Responsibility and Blame: A Structural-Model Approach. *J. Artif. Intell. Res.*, 2004, 22:93-115.

[14] Croce, F. and Lenzerini, M. A Framework for Explaining Query Answers in DL-Lite. Proc. EKAW, 2018.

[15] Datta, A., Sen, S. and Zick, Y. Algorithmic Transparency via Quantitative Input Influence: Theory and Experiments with Learning Systems. IEEE Symposium on Security and Privacy, 2016.

[16] Draisbach, U., Christen, P. and Naumann, F. Transforming Pairwise Duplicates to Entity Clusters for High-quality Duplicate Detection. *J. Data and Information Quality*, 2019, 12(1): 3:1-3:30.

[17] Du, J., Wang, K. and Shen, Y. A Tractable Approach to ABox Abduction over Description Logic Ontologies. Proc. AAAI, 2014.

[18] Dubey, P. and Shapley, L. S. Mathematical Properties of the Banzhaf Power Index. *Mathematics of Operations Research*, 1979, 4(2):99–131.

[19] Fan, W. and Geerts, F. *Foundations of Data Quality Management.* Morgan & Claypool, 2012.

[20] Fan, W., Gao, H., Ji, X., Li, J. and Ma, S. Dynamic Constraints for Record Matching. *The International Journal on Very Large Data Bases (VLDBJ)*, 2009, 20(4): 495-520.

[21] Halpern, J. and Pearl, J. Causes and Explanations: A Structural-Model Approach: Part 1. *British J. Philosophy of Science*, 2005, 56:843-887.

[22] Heidari, A., McGrath, J., Ilyas, I.F., and Rekatsinas, Th. HoloDetect: Few-Shot Learning for Error Detection. Proc. Sigmod, 2019.

[23] Jiang, L., Borgida, A. and Mylopoulos, J. Towards a Compositional Semantic Account of Data Quality Atrributes. Proc. International Conference on Conceptual Modeling (ER), 2008, pp. 55-68.

[24] Khamis, M.A., Ngo, H.Q., Nguyen, X., Olteanu, D. and Schleich, M. AC/DC: In-Database Learning Thunderstruck. Proc. DEEM, 2018.

[25] Kouki, P., Pujara, J., Marcum, C., Koehly, L. and Getoor, L. Collective Entity Resolution in Multi-Relational Familial Networks. *Knowl. Inf. Syst.*, 2019, 61(3):1547-1581.

[26] Kimelfeld, B. and Ré, C. A Relational Framework for Classifier Engineering. Proc. PODS, 2017.

[27] Kleinberg, J., Ludwig, J., Mullainathan, S. and Rambachan, A. Algorithmic Fairness. AEA Papers and Proceedings 2018, 108: 22–27.

[28] Krishnan, J., Franklin, M.J., Goldberg, K., Wang, J. and Wu, E.. BoostClean: Automated Error Detection and Repair for Machine Learning. arXiv:1711.01299, 2017.

[29] Livshits, E., Bertossi, L., Kimelfeld, B. and Sebag, M. The Shapley Value of Tuples in Query Answering. Proc. ICDT, 2020. arXiv:1904.08679.

[30] Lundberg, S. and Lee, S-I. A Unified Approach to Interpreting Model Predictions. Proc. NIPS, 2017.

[31] Meliou, A., Gatterbauer, W., Moore, K. F. and Suciu, D. The Complexity of Causality and Responsibility for Query Answers and Non-Answers. Proc. VLDB 2010.

[32] Pearl, J. *Causality: Models, Reasoning and Inference.* Cambridge Univ. Press, 2nd ed., 2009.

[33] Rammelaere, J. and Geerts, F. Explaining Repaired Data with CFDs. Proc. VLDB, 2018.

[34] Roth, A. (ed.) *The Shapley Value: Essays in Honor of Lloyd S. Shapley.* Cambridge University Press, 1988.

[35] Rudin, C. Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and use Interpretable Models Instead. *Nature Machine Intelligence*, 2019, 1(5): 206-215. arXiv:1811.10154

[36] Saleiro, P., Kuester, B., Stevens, A., Anisfeld, A., Hinkson, L. London, J. and Ghani, R. Aequitas: A Bias and Fairness Audit Toolkit. CoRR abs/1811.05577, 2018.

[37] Salimi, B., Bertossi, L., Suciu, D. and Van den Broeck, G. Quantifying Causal Effects on Query Answering in Databases. Proc. TaPP, 2016.

[38] Salimi, B., Gehrke, J. and Dan Suciu, D. Bias in OLAP Queries: Detection, Explanation, and Removal. Proc. SIGMOD 2018, pp. 1021-1035.

[39] Salimi, B., Howe, B. and Suciu, D. Data Management for Causal Algorithmic Fairness. *IEEE Data Eng. Bull.*, 2019, 42(3):24-35.

[40] Suciu, D., Olteanu, D., Re, C. and Koch, C. *Probabilistic Databases.* Synthesis Lectures on Data Management, Morgan & Claypool Publishers, 2011.