Conditional Regression Rules

Rui Kang
BNRist, Tsinghua University
kr20@mails.tsinghua.edu.cn

Shaoxu Song
BNRist, Tsinghua University
sxsong@tsinghua.edu.cn

Chaokun Wang
BNRist, Tsinghua University
chaokun@tsinghua.edu.cn

***Abstract*—Mixed data distribution is widely observed, for example, the bird migration data consist of the observed locations of various birds in different years, varying in data distribution. Learning a single regression model over such a mixed data distribution is often ineffective, while manually segmenting the data, e.g., by bird, date or region, for learning individual models is truly labor-intensive. In this paper, we propose to automatically discover the regression models that apply conditionally to only a part of the data, namely *conditional regression rules* (CRRs), enlightened by the conditional functional dependencies (CFDs) that are FDs hold only in some data. Remarkably, a regression model may apply in different parts of data, e.g., the seasonal migration of birds is similar in different years. To capture the shared regression models, we investigate the inference of CRRs. An algorithm is devised to learn and discover CRRs from data, with the help of CRR inference. Extensive experiments on real-world datasets demonstrate that the discovered conditional regression rules are more effective than the regression models without conditions. In particular, with the inference of CRRs, the number of learned CRRs is significantly reduced without sacrificing rule semantics.**

***Index Terms*—Integrity Constraint, Regression Model**

I. Introduction

Mixed data distribution is widely observed. Making assumptions such as mixed Gaussian distribution [1] is not always valid in practice, e.g., in spatio-temporal data [2]. For instance, the bird migration dataset BirdMap [3] consists of the observed locations of various birds in different years, varying in data distribution, as illustrated in Table I and Figure 1(a) in Example 1 below. Our preliminary study [4] shows that learning individual regression models for each tuple is accurate and effective. Unfortunately, it is computational expensive and not always necessary. As shown in Figure 1(b), a regression model (lines with the same color) may apply to a set or multiple isolated sets of data, not necessary to learn for each individual tuple.

Example 1. *Table I shows a fraction of the GPS locations of different birds in several years from the BirdMap dataset [3]. Figure 1(a) visualizes part of the mixed data distribution, where each point corresponds to a GPS record of the bird* $2.Maria$*. We use different colors to denote the data in different seasons, showing the seasonal migration of a bird, which is expected to be automatically discovered below.*

This work is supported in part by the National Natural Science Foundation of China (62072265, 62021002), the National Key Research and Development Plan (2021YFB3300500, 2019YFB1705301, 2019YFB1707001), BNR2022RC01011, and the MIIT High Quality Development Program 2020. Shaoxu Song is the corresponding author.

TABLE I
Example of the BirdMap dataset

TID	Latitude	Longitude	BirdID	Date
t_1	56.20883	26.92067	2.Maria	2006-8-6
t_2	55.83867	26.2075	2.Maria	2006-8-7
t_3	21.988	22.56783	2.Maria	2007-8-28
t_4	53.04183	25.80183	3.Raivo	2007-3-30
t_5	47.39333	27.40033	1.Kalakotkas	2007-3-26
t_6	–	–	2.Maria	2007-3-20
t_7	38.5855	28.040333	4.Mart	2007-9-11
t_8	38.58567	28.03583	4.Mart	2007-9-1
t_9	9.0155	20.07167	2.Maria	2008-8-27
t_{10}	6.7465	19.073	2.Maria	2007-9-4
t_{11}	58.61833	28.66967	33.Erika	2007-8-13

Regression models are learned over the data for various applications, such as imputing the missing locations in t_6 *in Table I. To better illustrate the regression model, Figures 1(b) and 1(c) plot* Latitude *and* Longitude *over* Date*, respectively.*

The existing methods, such as [5], may learn the same model multiple times in different parts of data. For example, in Figure 1(b), it splits the domain of timestamp and learns regression models separately, e.g., the red line denotes the migration of the bird from north to south between 2006-8-11 and 2006-9-12. Unfortunately, the same regression model is redundantly learned again from 2008-8-18 to 2008-9-19, since the birds travel seasonally in the same way.

Obviously, manually splitting the data, e.g., by bird, date or region, for learning individual models is truly labor-intensive. Motivated by the extended conditional functional dependencies (eCFDs [6]) that introduce disjunctions to apply CFD to some data, in this paper, we propose to automatically discover the regression models that apply conditionally to some parts of the data, namely *conditional regression rules* (CRRs).

Informally, a CRR φ is in a form of triple (f, ρ, C), where $f : \mathbf{X} \rightarrow Y$ is a regression function from attributes $\mathbf{X}$ to Y, ρ is the maximum bias between the attribute value of Y and prediction $f(\mathbf{X})$, and C specifies the conditions. Motivated by denial constraints (DCs) [7], we use predicates to form conjunction and disjunction to specify the parts of data where a CRR conditionally applies.

Example 2 (Example 1 continued)**.** *For the light-blue points with Date from 2006-8-11 to 2006-9-12, a CRR* φ_1 *is observed*

2375-026X/22/$31.00 ©2022 IEEE
DOI 10.1109/ICDE53745.2022.00231

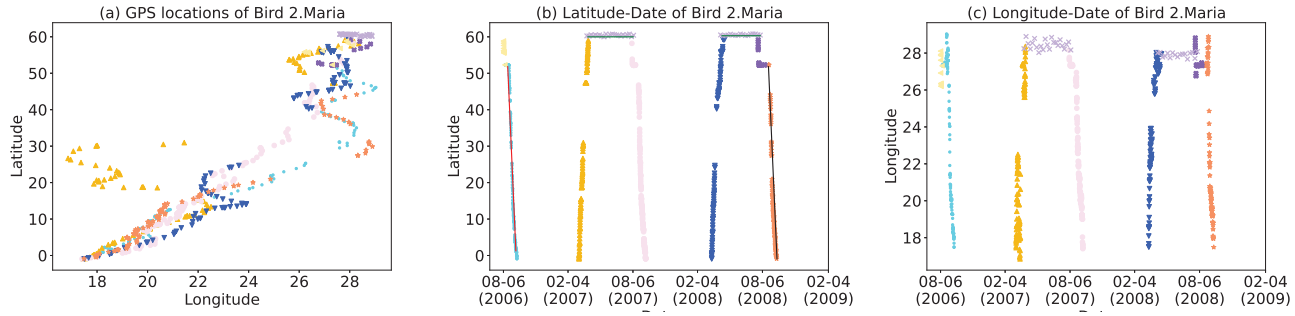

Fig. 1. Regression models conditionally apply to parts of the BirdMap data

in Figure 1(b).

$$\varphi_1 : f_1(\mathsf{Date}) = \mathsf{Latitude}, \rho_1 = 0.5,$$
$$C_1 = (\mathsf{Date} \geq \textit{2006-8-11} \wedge \mathsf{Date} < \textit{2006-9-12})$$

It denotes that the linear regression f_1 with the maximum bias $|\mathsf{Latitude} - f_1(\mathsf{Date})| \leq \rho_1$ applies to the tuples with $\mathsf{Date} \geq$ *2006-8-11* $\wedge\, \mathsf{Date} <$ *2006-9-12.*

The following CRR φ_2 applies to different parts of the data, from 2007-4-1 to 2007-8-8, as well as 2008-4-13 to 2008-7-31, denoted by disjunction in the condition $\mathbb{C}_2$.

$$\varphi_2 : f_2(\mathsf{Date}) = \mathsf{Latitude}, \rho_2 = 0.5, \mathbb{C}_2 = (\mathsf{Latitude} = 60.10 \wedge$$
$$((\mathsf{Date} \geq \textit{2007-4-1} \wedge \mathsf{Date} < \textit{2007-8-8}) \vee$$
$$(\mathsf{Date} \geq \textit{2008-4-13} \wedge \mathsf{Date} < \textit{2008-7-31})))$$

Analogous to CFDs specifying conditions on the dependent Y [8], the constant $\mathsf{Latitude} = 60.10$ *states that the prediction should always be* 60.10 *with the maximum bias $\rho_2 = 0.5$. It denotes that the bird stays in the location of the same* Latitude *from April to August each year.*

While CRRs such as φ_2 can be directly shared among different parts of the data, some others need translation. To share the aforesaid f_1 in another year, from 2008-8-18 to 2008-9-19, translation is needed, i.e., $f_1(\mathsf{Date} - x) = \mathsf{Latitude}$*, where $x = 744$ denotes the difference in date between two years.*

$$\varphi_3 : f_1(\mathsf{Date} - x) = \mathsf{Latitude}, \rho_3 = 0.5, \mathbb{C}_3 =$$
$$((\mathsf{Date} \geq \textit{2006-8-11} \wedge \mathsf{Date} < \textit{2006-9-12} \wedge x = 0) \vee$$
$$(\mathsf{Date} \geq \textit{2008-8-18} \wedge \mathsf{Date} < \textit{2008-9-19} \wedge x = 744))$$

It is worth noting that φ_3 subsumes the semantics of φ_1 given $x = 0$, i.e., using φ_3 is sufficient in practice and φ_1 is redundant.

The novelty of our CRR is thus to share the same regression model in different parts of data, specified by conditions. With model sharing, both the learning and evaluation of regression models could be improved. For example, in Figure 1(b), since the regression model f_1 in red line learned from *2006-8-11* to *2006-9-12* can also be applied in the time period from *2008-8-18* to *2008-9-19*, it does not need to be learned again, reducing the learning time cost. As the number of the rules is reduced, the evaluation time cost of the models is improved as well.

A. Challenges

Different from the existing studies for learning regression models [9]–[13], the main challenge of learning CRRs originates from the huge space of possible conditions, with disjunction and conjunction of predicates. In particular, regression models may be shared in different parts of data, specified by different conditions, such as φ_2 in Example 2. Moreover, some regression models might not be directly shared but with translation, e.g., translated from φ_1 to φ_3.

B. Contributions

In this paper, we will provide a deep insight into the problem of learning and inferring conditional regression rules. Our major contributions in this study are summarized as follows:

(1) We present a novel form of conditional regression rule (CRR) with disjunctions to capture the sharing of regression models among different parts of data, in Section III.

(2) We investigate the soundness of five inference rules for CRRs, in Section IV, which could be used to reduce the number of CRRs with shared models.

(3) We devise an algorithm to learn and discover CRRs from data, in Section V, where model sharing further improves the discovery time performance by avoiding redundant learning.

(4) We compare with strong baselines, in Section VI, to demonstrate the reduced number of regression rules and time performance without sacrifice of RMSE by model sharing.

II. Related Work

Our proposed CRR is motivated by both the integrity constraints and regression models that apply to part of the data, discussed in Sections II-A and II-B, respectively.

A. Conditions for Integrity Constraints

Integrity constraints apply conditions to build data models such as CFD [6], [8], [14], [15] and CDD [16], [17]. Different from other existing conditional integrity rules, CRR only considers single tuple relationship among attributes, which does not rely on the existence of other tuples. Moreover, CRR applies disjunctions based on predicates to form the condition, integrating multiple parts of data. Based on predicates, denial constraints (DCs) [7] define the false-negative part of data.

Different from denial constraints, CRRs generate regression models and use predicates for true-positive data partitions. In this way, CRRs provide precise models in integrity constraints for mixed data distribution.

B. Conditions for Regression Models

In general, our idea of model sharing is complementary to the existing models, such as logistic regression [18], by introducing the conditions for shared models in different parts of data. Conditional logistic regression [19], [20] trains logistic regression models over the part of data according to conjunction. However, model sharing is not studied, which is enabled by DNF conditions in our CRR. Conditional regression forest [21] builds an additive model to average the predictions from multiple regression trees, each of which is trained through data sampling from the same training set, leading to redundant regression models. Again, the model sharing is not studied.

C. Conditions for Time Series Analysis

For time-series analysis/forecasting, regression models are also considered, such as harmonic regression [22]. The harmonic regression [22] fits data by cosine functions and use different Fourier frequencies to capture short- and long-term periodic features of the whole dataset. In contrast, we use conditions to specify the periods, and thus enable model sharing. Recurrence time modeling [23] uses random variable T_j to represent the period and learn regression models over each period. Again, model sharing is not supported, the main motivation and novelty of our study.

III. Definition

In this section, we introduce the definitions of each component of CRRs formally in Section III-A. Then, we have the semantics and formal definition of CRRs in Section III-B.

A. Preliminaries

We start with the formal definition of predicate p, condition $\mathbb{C}$, regression function f, regression bias ρ in Section III-A1, Section III-A2, Section III-A3 and Section III-A4, respectively.

The CRRs are considered over a relational database D. The database D is a collection of tuples of schema $R(A_1, ...A_n), attr(R) = \{A_1, ...A_n\}$ as data samples from mixed data distribution. Given tuple $t \in D$, we denote $t.A_i = t[A_i]$ as the value of attribute A_i in tuple t.

1) Predicates p: As shown in Figure 1(b) and Example 2, a conjunction of predicates could describe a simple set of data for conditional regression model. Consider a set of operators $\Phi = \{=, >, \geq, <, \leq\}$, the predicate space $\mathbb{P}$ contains a set of single tuple predicates p with $p := A\phi c$, where $\phi \in \Phi$ and $A \in attr(R)$, c as a constant from the domain of A. As the usability of function translation illustrated by φ_3, we also have built-in predicates $\mathbf{x} = \Delta, y = \delta$ telling the parameter of function translation in $\mathbb{P}$ defined in Section III-A3. Given tuple t, assume that t is satisfied by any built-in predicates, we have $t \vDash p$ iff $t.A\phi c$.

2) Conditions $\mathbb{C}$: For describing simple part of data, e.g., light-blue part of data in Figure 1, a conjunction C_i connects a set of predicates $pred(C_i) \subseteq \mathbb{P}$ with the conjunction i.e.,

$$C_i : \bigwedge_{p \in pred(C_i)} p.$$

For each tuple t, we have $t \vDash C_i$ iff $\forall p \in pred(C_i), t \vDash p$.

Mixed data distributions may vary in the shape of data, and the same distribution could be shared over more complex data sets, e.g., the horizontal green lines in Figure 1 (b). Thereby, we use a disjunction normal form (DNF) composed by countable conjunctions $C_1 ... C_n$, i.e.,

$$\mathbb{C} = (\bigwedge_{p \in pred(C_1)} p) \vee ... \vee (\bigwedge_{p \in pred(C_n)} p).$$

For each tuple t and the DNF condition $\mathbb{C}$, we have $t \vDash \mathbb{C}$ iff $\exists C_i, t \vDash C_i$. Given the condition $\mathbb{C}$ and instance $I \subseteq D$, we denote $I_{\mathbb{C}}$ as the subset of tuples in instance I with each tuple satisfying condition $\mathbb{C}$.

3) Regression f: Observing in Example 1 and Figure 1(b), each curve corresponds to a regression function f that predicts the Latitude of a bird in a time slice. We have the definition of regression function as follows. Given attributes $\mathbf{X} \subset attr(R)$, built-in variables $\mathbf{B} = \{\mathbf{x}, y\}$ and regression target $Y \in attr(R)$, a regression function $f : \mathbf{X}, \mathbf{B} \rightarrow Y$ generates predictions with

$$Y = f(\mathbf{X} + \mathbf{x}) + y = f(\mathbf{X}_1 + \mathbf{x}_1, ...) + y.$$

For built-in variables in f, the value of $\mathbf{x}, y$ is defined by built-in predicates $\mathbf{x} = \Delta, y = \delta$ associated with the conjunction of each part of data, where the constant Δ_i is one of the differences between two values in the domain of $X_i \in \mathbf{X}$, and so as the constant δ of target Y. Thus, the constants Δ and δ denote the difference of translating one tuple to another. In Example 2, the data in $\mathsf{Date} \geq 725 \wedge \mathsf{Date} < 769$ is associated with built-in predicates $d = 725$ in condition, representing the difference between two data parts on applying model f_1.

4) Bias ρ: In Figure 1(b), slight differences are observed between $f(\mathsf{Date})$ and $t.\mathsf{Latitude}$ even if the model f is fine-tuned. In Table I, the precision of the GPS value of each tuple is limited, e.g., for predicting Latitude, the precision 10^{-3} should be a lower bound of bias ρ. Thus, to decide the bias ρ for regression function f and target value Y, we consider both the precision of attribute Y and model bias $\max_t |t.Y - f(t.\mathbf{X})|$. The regression model of data is composed of regression function f and bias ρ, denoted by (f, ρ).

B. Semantics

The regression model (f, ρ) with $f : \mathbf{X}, \mathbf{B} \rightarrow Y$ estimates the target value Y based on attributes $\mathbf{X} \subset attr(R)$ under some data by the regression function f and bias ρ. Thus, to capture the semantic relationship between the conditions to some data and the regression models, we introduce the definition of CRR.

Definition 1. *Given condition $\mathbb{C}$ and database D of schema R, conditional regression rule (CRR) φ is in the form of*

$$\varphi : (f, \rho, \mathbb{C}),$$

where $f : \mathbf{X}, \mathbf{B} \to Y$ is a regression function mapping from the value of attributes $\mathbf{X} \subseteq R$ to the prediction of target $Y \in R$ with $\rho \geq 0$ as the maximum error between prediction and the value of Y, and $\mathbb{C}$ is a condition in DNF format with no predicates on attribute Y.

Based on CRR definition, we could see that conditional regression model refines the regression model $(f : \mathbf{X}, \mathbf{B} \to Y, \rho)$ by applying the data semantically satisfied by condition C. Based on the condition semantics defined in Section III-A, we interpret the semantics of CRRs as below. A tuple t of schema R satisfies CRR $\varphi : (f, \rho, \mathbb{C})$, denoted as $t \vDash \varphi$, if $t \vDash \mathbb{C}$, then

$$|t.Y - (f(t.\mathbf{X} + \mathbf{x}) + y)| \leq \rho.$$

The value of $\mathbf{x}$ and y is determined by built-in predicates $\mathbf{x} = \Delta, y = \delta$ in each conjunction with the default values of $\mathbf{x} = \mathbf{0}, y = 0$.

IV. Inference

As illustrated in Examples 1 and 2, some CRRs can be implied from others. The implication is essential to identify the shared regression models among different parts of data, and discover a concise set of CRRs in Section V below. To study the extension and reasoning of CRRs, we analyze the inference of CRRs in Section IV-B, based on the inference of conditions in Section IV-A.

A. Inference on Conditions

We briefly introduce the inferences on conditions based on DNF and conjunction, studied as predicate calculus in [7], [24]. Conjunction inference $C_1 \vdash C_2$ denotes the refinement of conditions, as defined in [7]. It means that for any tuple t of R, $t \vDash C_1$ implies $t \vDash C_2$. Based on conjunction inferences, we restate the inference of DNFs for illustrating the CRR inference.

Definition 2. *For two DNFs $\mathbb{C}_1$ and $\mathbb{C}_2$, we say $\mathbb{C}_1 \vdash \mathbb{C}_2$ if and only if, for all conjunction $C_1 \in \mathbb{C}_1$, there exists a conjunction $C_2 \in \mathbb{C}_2$ s.t. $C_1 \vdash C_2$.*

B. Inference of CRRs

We define, state and prove the soundness of the inference rules for CRR, as below.

Proposition 1 (Reflexivity)**.** *If $Y \in \mathbf{X}$, then we have $\varphi : (f, \rho, \mathbb{C})$, where $f(\mathbf{X}) = Y, \rho = 0, \mathbb{C} = \emptyset$.*

Proof. For any tuple t of schema R, since $Y \in \mathbf{X}$, we have $t.Y = f(t.\mathbf{X})$, i.e., $|t.Y - f(t.\mathbf{X})| = 0 \leq \rho$. That is, $t \vDash \varphi$ for any tuple t. □

Reflexivity extends the reflexivity rule in Armstrong's Axiom [25] on FDs to regression models. It means that any tuple t of schema R always satisfies a CRR $\varphi : (f, \rho, \mathbb{C})$ with $f(\mathbf{X}) = Y, Y \in \mathbf{X}$. For instance, any tuple t satisfies $\varphi : (f, \rho, \mathbb{C})$ with $f(\mathsf{Salary}, \mathsf{Tax}) = \mathsf{Tax}, \rho = 0, \mathbb{C} = \emptyset$. For CRR discovery, Reflexivity could avoid generating trivial rules by ignoring $Y \in \mathbf{X}$ as regression target.

Proposition 2 (Induction)**.** *If $\mathbb{C}_2 \vdash \mathbb{C}_1$, then $\varphi_1 : (f, \rho, \mathbb{C}_1)$ implies $\varphi_2 : (f, \rho, \mathbb{C}_2)$.*

Proof. Consider any tuple t with $t \vDash \varphi_1$. If $t \vDash \mathbb{C}_2$, there exist some $C_2 \in \mathbb{C}_2$ having $t \vDash C_2$. Referring to Definition 2, given $\mathbb{C}_2 \vdash \mathbb{C}_1$, for any conjunction $C_2 \in \mathbb{C}_2$, there exists $C_1 \in \mathbb{C}_1$ s.t. $C_2 \vdash C_1$. Thus, for any conjunction $C_2 \in \mathbb{C}_2$, if $t \vDash C_2$, then $\exists C_1 \in \mathbb{C}_1, t \vDash C_1$. It follows $t \vDash \mathbb{C}_1$. According to $t \vDash \varphi_1$, we have $|t.Y - (f(t.\mathbf{X} + \mathbf{x}) + y)| \leq \rho$, i.e., $t \vDash \varphi_2$ as well. Otherwise, for $t \nvDash \mathbb{C}_2$, it naturally has $t \vDash \varphi_2$. To sum up, φ_1 implies φ_2. □

Induction means that for any tuple t, if t satisfies φ_1, then t also satisfies the CRR $\varphi_2 : (f, \rho, \mathbb{C}_2)$ under refined condition $\mathbb{C}_2$ with $\mathbb{C}_2 \vdash \mathbb{C}_1$. For instance, CRR $\varphi_5 : (f_5, \rho_5, C_5)$ with $f_5(\mathsf{Salary}) = 0.04 * \mathsf{Salary} - 230, \rho_5 = 1, C_5 = (\mathsf{S} = IA)$ implies $\varphi_5' : (f_5, \rho_5, C_5' = (\mathsf{S} = IA \wedge \mathsf{MS} = S))$ by applying Induction. When discovering CRRs, Induction lays the foundation for the addition of the default built-in predicates $\mathbf{x} = \mathbf{0} \wedge y = 0$, and also helps the comparison among counterpart conditions, e.g., conditions on the same set of attributes.

Proposition 3 (Fusion)**.** *For $\varphi_1 : (f, \rho, \mathbb{C}_1)$, $\varphi_2 : (f, \rho, \mathbb{C}_2)$, φ_1 and φ_2 imply $\varphi_3 : (f, \rho, \mathbb{C}_3)$, where $\mathbb{C}_3 = \mathbb{C}_1 \vee \mathbb{C}_2$.*

Proof. Consider any tuple t with $t \vDash \varphi_1$ and $t \vDash \varphi_2$. If $t \vDash \mathbb{C}_3 = \mathbb{C}_1 \vee \mathbb{C}_2$, there exists some conjunction $C \in \mathbb{C}_3$ having $t \vDash C$. If $C \in \mathbb{C}_1$, then $t \vDash \mathbb{C}_1$. It follows $|t.Y - (f(t.\mathbf{X} + \mathbf{x}) + y)| \leq \rho$ since $t \vDash \varphi_1$. Thus, we have $t \vDash \varphi_3$. Otherwise, when $C \in \mathbb{C}_2$, following similar proof, we also have $t \vDash \varphi_3$. To conclude, φ_1, φ_2 imply φ_3. □

Fusion means that for any tuple t satisfies φ_1 and φ_2, if the condition $\mathbb{C}_3 = \mathbb{C}_1 \vee \mathbb{C}_2$, then t satisfies $\varphi_3 : (f, \rho, \mathbb{C}_3)$. That is, Fusion is binary for combining the conditions of two CRRs sharing the same regression model. In this way, Fusion helps the reduction of total number of CRRs for modeling data within database D. Recall φ_2 in Example 2, which is composed by the same regression model shared by two parts of conjunction $\mathbb{C}_2 = \mathsf{Latitude} = 60.10 \wedge (C_{2,1} \vee C_{2,2})$. Given $(f_2, \rho_2, C_{2,1})$ and $(f_2, \rho_2, C_{2,2})$, Fusion helps in obtaining φ_2.

Proposition 4 (Generalization)**.** *If $\rho_2 \geq \rho_1$, then $\varphi_1 : (f, \rho_1, \mathbb{C})$ implies $\varphi_2 : (f, \rho_2, \mathbb{C})$.*

Proof. Consider any tuple t with $t \vDash \varphi_1$. If $t \vDash \mathbb{C}$, then $|t.Y - (f(t.\mathbf{X} + \mathbf{x}) + y)| \leq \rho_1 \leq \rho_2$. It follows $t \vDash \varphi_2 : (f, \rho_2, \mathbb{C})$. Otherwise, it is natural to reach $t \vDash \varphi_2$ when $t \nvDash \mathbb{C}$. To conclude, φ_1 implies φ_2. □

Generalization means that for any tuple t satisfies $\varphi : (f, \rho_1, \mathbb{C})$, if $\rho_2 \geq \rho_1$, then the tuple t satisfies φ_2. Generalization considers a tolerance on prediction bias, especially when applying Fusion over two CRRs with different bias ρ. For example, given $\varphi_2 : (f_2, \rho_2 = 0.5, \mathbb{C}_2)$, we have φ_2 implies $(f_2, \rho_3 = 1.0, \mathbb{C}_2)$, as $\rho_3 \geq \rho_2$.

Proposition 5 (Translation). *For* $\varphi_1 : (f_1, \rho, \mathbb{C}_1)$, $\varphi_2 : (f_2, \rho, \mathbb{C}_2)$, *if* $f_2(\mathbf{X}) = f_1(\mathbf{X} + \Delta) + \delta$, *then* φ_1 *and* φ_2 *imply* $\varphi_3 : (f_3, \rho, \mathbb{C}_3)$, *where* $f_3(\mathbf{X}) = f_1(\mathbf{X} + \mathbf{x}) + y$, *and* $\mathbb{C}_3 = (\mathbb{C}_1 \wedge \mathbf{x} = \mathbf{0} \wedge y = 0) \vee (\mathbb{C}_2 \wedge \mathbf{x} = \Delta \wedge y = \delta)$.

Proof. Consider any tuple t with $t \vDash \varphi_1$ and $t \vDash \varphi_2$. If $t \vDash \mathbb{C}_3$, there exists some $C \in \mathbb{C}_3$ having $t \vDash C$. When $C \in \mathbb{C}_1$ with built-in predicate $\mathbf{x} = \mathbf{0} \wedge y = 0$, we have $t \vDash \mathbb{C}_1$. According to $t \vDash \varphi_1$, it has $|t.Y - (f_1(t.\mathbf{X} + \mathbf{x}) + y)| \leq \rho$, i.e., $t \vDash \varphi_3$. When $C \in \mathbb{C}_2$ with built-in predicate $\mathbf{x} = \Delta \wedge y = \delta$, it follows $t \vDash \mathbb{C}_2$. Given $t \vDash \varphi_2$, we have $|t.Y - (f_2(t.\mathbf{X} + \mathbf{x}) + y)| \leq \rho$. According to $f_2(\mathbf{X}) = f_1(\mathbf{X} + \Delta) + \delta$, it has $|t.Y - (f_1(t.\mathbf{X} + \mathbf{x} + \Delta) + y + \delta)| \leq \rho$, i.e., $t \vDash \varphi_3$. Otherwise, for $t \nvDash \mathbb{C}_3$, it naturally has $t \vDash \varphi_3$. To sum up, φ_1, φ_2 imply φ_3. □

Translation means that for any tuple t satisfies φ_1 and φ_2, if models f_1 and f_2 have $f_2(\mathbf{X}) = f_1(\mathbf{X} + \Delta) + \delta$, then tuple t satisfies φ_3, composed by the model from f_1. Translation replaces the regression model in a CRR by the model in another CRR to share regression models. For instance, CRRs φ_4 and φ_5 adopt different models with the relationship $f_5(\mathsf{Salary}) = 0.04 * \mathsf{Salary} - 230 = f_4(\mathsf{Salary}) - 230$. Thus, φ_4, φ_5 imply $(f_4, \rho_5, (C_5 \wedge y = -230))$ after applying Translation. It supports CRR discovery by comparing and integrating regression models, which forms the equivalence relationship among the shared regression models, discussed in Section V-B1. Proposition 9 further uses f_1 as the regression model, and decides the built-in predicates inside each conjunction after applying Translation.

In Problem 1 below, we propose to find a set Σ of CRRs that cannot be further reduced using the five inference rules. The algorithms in Section V avoid enumerating all the CRRs, some of which can be implied by others using the five inference rules. It thus reduces the number of discovered CRRs and improves the discovery efficiency.

Unfortunately, it is still open whether the five inference rules for CRRs form a complete set. As the first study of model sharing using conditions, we propose to remove redundant semantics with our best effort, referring to the five inference rules we have thus far.

V. Discovery

With the help of the CRR inference rules in Section IV, we could devise discovery algorithms for learning non-trivial CRRs on the database D, where each CRR holds a regression function $f : \mathbf{X}, \mathbf{B} \to Y, Y \notin \mathbf{X}$.

With Translation, Fusion inferences and DNF conditions, we could merge multiple CRRs with shared models for more general CRRs. Applications such as data imputations could take advantage of the merged CRRs by saving time on rule locating and predicting. For example, in Example 2 and Figure 1, rather than finding CRRs for each part of data, we use CRR φ_3, which is more efficient for the imputation task e.g., 4 parts could be seen from *2008-8-18* to *2008-9-19*.

In this sense, we aim to find a concise set of CRRs from data. To begin with, we first formalize this discovery problem of CRRs in Problem 1.

Problem 1. *Given a database* D *of schema* R, *attributes* $\mathbf{X}$, *target* Y, *maximum bias* ρ_M *and predicates* $\mathbb{P}$ *for conditions, the CRR discovery problem aims to generate CRRs* Σ *such that* Σ *covers the database* D, *i.e.,* $\forall t \in D$, $\exists \varphi : (f, \rho, \mathbb{C}) \in \Sigma, t \vDash \mathbb{C}$, *and is not reducible by inference rules, i.e., (1)* $\nexists \varphi \in \Sigma$ *where* Reflexivity *applies, (2)* $\nexists \varphi, \varphi' \in \Sigma$ *s.t.* φ *implies* φ' *by applying* Induction *or* Generalization, *(3)* $\nexists \varphi_1, \varphi_2, \varphi_3 \in \Sigma$ *s.t.* φ_1, φ_2 *imply* φ_3 *by* Fusion *or* Translation.

Thus, based on Problem 1, to find a concise set of CRRs, we need to share the models and merge CRRs through inferences. As noticed in Example 2 model sharing avoids training new regression models too frequently such as tree-based models [9], [10]. However, as mentioned in challenges I-A, models generate predictions over parts of data specified by different conditions, and sometimes, model sharing asks for model translation with built-in predicates decision.

In this section, to generate CRRs as described in Problem 1, we perform CRR discovery in two steps, i.e., generating CRRs from data with model sharing in Section V-A, and compacting the generated CRRs for a concise set of CRRs in Section V-B.

A. CRR Searching with Model Sharing

For finding CRRs under data satisfaction, in this section, we start with the most general condition and refine the condition by adding predicates. Similar to tree-based regression learning [9], [10], the top-down strategy may find a regression model on each part of data specified by conjunction. However, in this way, the model sharing over data partitions would not be processed until all CRRs are sufficiently discovered.

Moreover, noisy data leads to the hardness of generating precise regression models. For example, for $\mathsf{Latitude} - \mathsf{Date}$ data shown in Figure 1, slight differences in data lead to the failure of sharing the models on light-blue and pink points. Based on the Translation inference, two CRRs with linear models are able to translate into a new CRR when they share the same slope, which could be sensitive to data noise.

Motivated by the above discussions, we propose an incremental CRR discovery algorithm that trains new models to fit the data only when no existing models could be shared, as shown in Algorithm 1.

In Algorithm 1, the incremental discovery maintains the set of shared models $\mathcal{F}$ as shown in Line 2, and finds CRRs starting from the most general condition $C = \emptyset$ as shown in Line 3. We introduce the incremental discovery of CRRs by model sharing in Section V-A1 for Line 7-10 and top-down searching in Section V-A2 for Line 4-5 and 12-22.

1) Model Sharing: In this section, we focus on a simple way of model sharing on the part of data under condition C.

Instead of sharing the model based on the Translation inference, the model sharing based on data could generalize to noisy database. We also prove the correctness of sharing model in this section, by Proposition 7.

For a conjunction C, Line 7-10 focus on the model sharing on all tuples from D_C, where D_C is the collection of tuples satisfying the condition C. We first consider two cases, $\delta = 0$ and $\delta \neq 0$, to discuss model sharing based on data.

Algorithm 1 CRR Searching with Model Sharing

Input: Database D of schema R, attributes $\mathbf{X}$, target Y with $Y \notin \mathbf{X} \subseteq attr(R)$, maximum bias ρ_M and predicate space $\mathbb{P}$ with no predicates on Y

Output: Σ CRRs with conjunction conditions

1: CRR set $\Sigma = \emptyset$
2: Model set $\mathcal{F} = \emptyset$
3: Priority Queue $Q = \{(C = \emptyset, i(C) = 0)\}$
4: **while** queue Q is not empty **do**
5: Conjunction $C = Top(Q)$
6: // Find possible model sharing
7: **if** $\exists f \in \mathcal{F}$ s.t. $\exists \delta, \forall t \in D_C, |t.Y - (f(t.\mathbf{X}) + \delta)| \leq \rho_M$ **then**
8: Decide $\rho = \max_{t \in D_C} |t.Y - (f(t.\mathbf{X}) + \delta)|$
9: $C = (C \wedge (y = \delta))$
10: $\Sigma = \Sigma \cup \{(f, \rho, C)\}$
11: **else**
12: $ind(C) = \max_{f \in \mathcal{F}} \frac{|\{t | t \in D_C \wedge |t.Y - (f(t.\mathbf{X}) + \delta_0)| \leq \rho_M\}|}{|D_C|}$
13: Train f under data D_C
14: **if** error $\max_{t \in D_C} |t.Y - f(t.\mathbf{X})| \leq \rho_M$ **then**
15: $\rho = \max_{t \in D_C} |t.Y - f(t.\mathbf{X})|$
16: $\Sigma = \Sigma \cup \{(f, \rho, C)\}$
17: $\mathcal{F} = \mathcal{F} \cup \{f\}$
18: **else**
19: Decide split predicate $p_1, ...p_n \in \mathbb{P}' \subset \mathbb{P}$
20: **for** $p_i \in \{p_1, ...p_n\}$ **do**
21: Build $C_i = (C \wedge p_i)$
22: Add to queue $Q = Q \cup \{(C_i, ind(C))\}$
23: **return** Σ.

Firstly, when the condition in Line 7 is satisfied by $\delta = 0$, we have $\forall t \in D_C, |t.Y - f(t.\mathbf{X})| \leq \rho_M$, indicating that the function f fits for the part of data D_C perfectly by allowing maximum bias ρ_M. In this way, f is considered to be the regression function of tuples from D_C, and we add a CRR $(f, \rho, C \wedge (y = 0))$ to the result Σ (Line 10).

Then, when $\delta \neq 0$, the condition $\forall t \in D_C, |t.Y - (f(t.\mathbf{X}) + \delta)| \leq \rho_M$ indicates that by adding the prediction $f(t.\mathbf{X})$ to a constant δ, the model $f(\mathbf{X}) + \delta$ fits all data from D_C within bias ρ_M. Thus, the function f could be shared by data D_C with a built-in predicate $y = \delta$. With an update on condition $C = (C \wedge (y = \delta))$, we add the CRR (f, ρ, C) to Σ (Line 10).

The value of δ could be determined by the average $\delta_0 = (\mathsf{max}\{t.Y - f(t.\mathbf{X}) | t \in D_C\} + \mathsf{min}\{t.Y - f(t.\mathbf{X}) | t \in D_C\})/2$, indicating the value δ_0 in $f(\mathbf{X}) + \delta_0$ for the minimum absolute error $|Y - (f(\mathbf{X}) + \delta_0)|$. The following conclusion considers the determination of δ.

Proposition 6. *Only if* $\forall t \in D_C, |t.Y - (f(t.\mathbf{X}) + \delta_0)| \leq \rho_M$ *is satisfied, the model* f *could be shared by tuples under condition* C *by a built-in predicate* $y = \delta$*, where* $\delta_0 = (\mathsf{max}\{t.Y - f(t.\mathbf{X}) | t \in D_C\} + \mathsf{min}\{t.Y - f(t.\mathbf{X}) | t \in D_C\})/2$.

Thus, δ_0 provides the necessary condition of whether we could share model f over all tuples in D_C. Based on the above conclusion, we could examine whether data D_C can translate to model f under the condition in Line 7 by assigning $\delta = \delta_0$. Further, we have the following conclusion for the correctness of the data-based model-sharing.

Proposition 7. *If the model* f_1 *in CRR* $\varphi_1 : (f_1, \rho_1, C_1)$ *could be shared by instance* D_C *by applying a built-in predicate* $y = \delta$*, then there exists a CRR* $\varphi_2 : (f_2, \rho_2, C_2 = C)$ *s.t.* $\varphi_3 : (f_3, \rho_3, C_3)$ *with* $f_3 = f_1, C_3 = (C \wedge (y = \delta))$ *could be reached by applying* Translation *inference on* φ_1, φ_2.

Proposition 7 actually points out the existence of a model for a data partition when the data-based model sharing conditions are satisfied. Without fine-tuning over noisy data, Proposition 7 could avoid the influence of noisy data to model sharing through Translation inference. Thus, Proposition 7 provides a practical solution to model sharing when CRRs need to learn over noisy data.

2) Top-down Searching: In this section, we consider the top-down searching for CRRs among Line 12-22 in Algorithm 1. Similar to the top-down approaches [9], [10] in tree-based structures, we briefly introduce Line 13-22 here for completeness and its adjustments to CRR discovery.

Among Line 13-17, we start with the condition C to train new model by Line 13, as we cannot observe any model sharing for data in D_C. To find whether the model could generalize to all data in D_C, we test the new model f with all data in D_C and compare the difference between the predictions of the model f and the observed values of the attribute values on Y at Line 14, i.e., $\max_{t \in D_C} |t.Y - f(t.\mathbf{X})| \leq \rho_M$. When the new model f is satisfied by the above condition, we calculate the bias ρ for f under data D_C and add the new CRR into Σ (Line 15-16). To potentially utilize f as the existing model later, we add the trained model f in the model set $\mathcal{F}$ (Line 13).

When no model could be shared in $\mathcal{F}$ among Line 7-10, we search for more models and more data partitions. Line 19-22 finds split predicates p for data partition from $\mathbb{P}' \subset \mathbb{P}$ decided by applicable strategies from tree-based approaches [9]. We then add all new formed data partitions with conditions $C \wedge p$ to the queue Q.

We use VC dimension [26] to determine the number of tuples for learning a regression model. If it cannot find a suitable regression model for the smallest data part so that the error is less than ρ_M, we have to return this model anyway to ensure the coverage. The edge case is indeed a regression model per tuple, i.e., our preliminary study [4]. For example, for linear regression, any tuple (the smallest data part) could learn a regression model.

3) Conjunction Ordering: In the above two parts, the Algorithm 1 mainly focuses on model reuse among Line 5-10 that tries to fit the current part of data with existing models w.r.t. the error ρ_M. It speeds up the search by avoiding learning the same model multiple times over different parts of the data. Based on Propositions 6 and 7, the model reuse enables model sharing in Algorithm 1 that combines the conditions of the same regression model. Existing top-down approaches like regression tree [9]–[13] cannot capture the model sharing

among different parts of data, and thus may learn the same regression model multiple times in different data parts.

In addition to model sharing, we develop another new innovation of the method, by discovering those CRRs first which could be more likely in model sharing. Let the index $ind(C)$ denote the probability of conjunction C being shared. We estimate the probability by Line 12 of Algorithm 1. Consequently, Line 5 considers C in the decreasing order of $ind(C)$, using a priority queue Q. In addition, the split decision could be adjusted in Line 19 based on $ind(C)$ for the guarantee on model sharing.

Proposition 8. *Given the conjunction C, the index $ind(C)$ and predicates $\mathbb{P}$, when we take $\lceil(1 - ind(C))|D_C|\rceil \leq |\mathbb{P}|$ as the number of predicates for splitting C, at least one of the resulting new conjunctions could be shared by an existing regression model.*

That is, when we choose $\lceil(1-ind(C))|D_C|\rceil$ as the number of predicates for splitting conjunction C, it guarantees that at least one of the resulting new conjunctions could be shared by an existing CRR. Thus, considering C with index $ind(C)$ in a decrease order could speed up the model sharing and discovery of CRRs by fewer conjunction splits.

4) Complexity Analysis: Algorithm 1 has time complexity $O(|D|^2 log|D|)$. Similar to the condition discovery in regression trees, the predicates could make binary searching for further data partitions in Line 19. In Line 17 of Algorithm 1, we add a regression model when a new CRR is formed. Thus, the total number of tested models in Line 7 is bounded by $|\Sigma|$. As the CRR discovery terminates when applying each tuple to a regression model, we have $|\Sigma| \leq |D|$. When we make binary searching for further data partitions in Line 19, the searching terminates within $O(\log|D|)$ partitions. Thus, the worst case is to test $|D|$ models in Line 7 by all tuples $O(\log|D|)$ times, i.e., the time complexity is $O(|D|^2 \log|D|)$.

B. CRR Compaction with Inference

A more concise set of CRRs without sacrifice of rule semantics could obviously improve the efficiency of downstream applications such as data imputation. In this section, we aim to find a concise set of CRRs Σ^* from the aforesaid Σ discovered from data. To reduce the size of CRR set, we use inferences in Proposition 5 to translate and merge CRRs.

In Propositions 1 and 2, Reflexivity inference aim to identify trivial CRR patterns, and Induction refines the condition of CRRs. As discussed in Section IV-B, Generalization considers a tolerance on ρ that could be the first step of Fusion. The other two rules, Translation and Fusion, share the regression models and merge CRRs of the same regression models.

In this section, we apply the inferences Translation, Generalization and Fusion in rule compaction. Based on the above three inferences of CRRs, we devise Algorithm 2 for compacting CRRs. Algorithm 2 processes mainly in two steps, i.e., rule translation (Line 3-11) in Section V-B1 and rule fusion (Line 12-15) in Section V-B2.

Algorithm 2 CRR Compaction with Inference

Input: Database D of schema R, maximum bias ρ_M and CRRs Σ
Output: a concise set of CRRs Σ^*

```
1: Initial CRR set Σ* = Σ
2: Queue Q = Σ
3: while queue Q is not empty do
4:   CRR φ : (f, ρ, ℂ) = Top(Q)
5:   for CRR φ' : (f', ρ', ℂ') ∈ Σ* and f' ≠ f do
6:     if ∃Δ, δ s.t. f'(X) = f(X + Δ) + δ then
7:       // Apply Translation by solving Line 8-10
8:       for conjunction C' ∈ ℂ' with built-in predicates
         x = Δ', y = δ' do
9:         Replace built-in predicates in C' by x = Δ' +
           Δ, y = δ' + δ
10:      Update Σ* = Σ* ∪ {(f, ρ', ℂ')}
11:      Remove φ' from Σ* and Q
12: while ∃φ : (f, ρ, ℂ), φ' : (f', ρ', ℂ') ∈ Σ* having f = f'
    do
13:   Apply Generalization: ρ'' = max(ρ, ρ')
14:   Apply Fusion: ℂ'' = (ℂ ∨ ℂ')
15:   Remove φ, φ' from Σ*
16:   Update Σ* = Σ* ∪ {φ'' : (f, ρ'', ℂ'')}
17: return Σ*
```

1) Rule Translation: In this section, we first introduce the basic property of the Translation inference. Then, we introduce the operations among Line 3-11 in Algorithm 2.

As the input CRRs Σ for Algorithm 2 may include built-in predicates for model sharing, e.g., data-based model sharing among Line 8-10 in Algorithm 1, the model sharing based on the Translation inference also introduces built-in predicates for each conjunction. After applying multiple Translation inference or model sharing, each conjunction may need to combine multiple built-in predicates, and keep one built-in predicate $\mathbf{x}, y$ in each conjunction. Thus, it is important to decide the built-in predicates under this case.

As shown in Proposition 5, Translation inference consumes two CRRs φ_1, φ_2 and generates φ_3 with built-in predicates. We have Proposition 9 for built-in predicate decision when φ_3 is again translated to get φ_4.

Proposition 9. *Given CRR $\varphi_1 : (f_1, \rho_1, C_1)$, $\varphi_2 : (f_2, \rho_2, C_2)$ and $\varphi_3 : (f_3, \rho_3, C_3)$, if $f_1(\mathbf{X}) = f_2(\mathbf{X}+\Delta)+\delta$ and $f_2(\mathbf{X}) = f_3(\mathbf{X} + \Delta') + \delta'$, then φ_2, φ_1 imply $\varphi_1' : (f_1', \rho_1, C_1 \wedge (\mathbf{x} = \Delta \wedge y = \delta))$ with $f_1' = f_2$, and φ_3, φ_1' imply $\varphi_1'' : (f_1'', \rho_1'', C_1'')$, where $f_1'' = f_3$, $\rho_1'' = \rho_1$ and $C_1'' = (C_1' \wedge (\mathbf{x} = \Delta' \wedge y = \delta'))$. We have the decision of built-in predicates of C_1'' as $C_1'' = (C \wedge (\mathbf{x} = \Delta' + \Delta \wedge y = \delta' + \delta))$.*

Proposition 9 decides the built-in predicates when the Translation inference is applied multiple times to the same CRR and its translations. Based on Proposition 9, we have the correctness of Line 8-10.

Thus, when Translation inference applied φ_1, φ_2 for φ_3, we could also reach φ_2 after applying the Translation inference on

φ_1, φ_3. Based on the Translation inference and Proposition 9, we could introduce the equivalence relationship among CRRs, i.e., when φ_3 is obtained by Translation on φ_1, φ_2, it indicates that φ_2 is equivalent to φ_3. Thus, for each step of applying the Translation inference, we would obtain the equivalent set of CRRs as the input. Also, the results of Line 3-11 will not be influenced by the order of applying CRR Translation.

Among Line 3-11, applying Translation inference aims to unify the regression models in Σ^* for condition fusion. In Line 4, we take a CRR $\varphi : (f, \rho, C)$ from the queue Q each time. Among Line 6-11, we try the Translation inference for all rules φ' with $f'(\mathbf{X}) = f(\mathbf{X}+\Delta)+\delta$. Besides the equivalence relationship among CRRs with multiple Translations, considering the equivalence as the existence of built-in predicates $\mathbf{x} = \Delta, y = \delta$ between two functions, we can construct the equivalence relationship among functions.

Thereby, when φ_1 is used at Line 4 for chances of Translation, φ_1 will translate all CRRs with the same equivalent class with function f_1. When φ_2 is translated into φ_2' by φ_1, there is no need for using φ_2 again, as all CRRs with their functions equivalent to f_1 and f_2 are already translated by φ_1 with function f_1. Considering this, at Line 10, we add a new CRR $(f, \rho', \mathbb{C}')$ to Σ^*, and remove its equivalent CRR from queue Q (Line 11).

2) Rule Fusion: Rule fusion among Line 12-16 utilizes two inferences Generalization and Fusion. As discussed above, the functions of the same equivalent class could use the built-in predicates translating each other. Thus, after applying Translation inference, all functions from the same equivalence class will obtain a unique form for ease of fusion.

The inference Fusion requires two CRRs with the same regression function f and bias ρ, and combines their conditions with a DNF. For any pair of CRRs share the same regression function, we could find the same regression model after applying the Generalization inference. Thus, at Line 12, we enumerate all pairs of CRRs with the same regression functions, and apply Generalization (Line 13) and Fusion to combine two rules.

In this way, we could finally merge all CRRs with the functions in the same equivalence class. Based on the above discussions, we have a concise set of CRRs Σ^* from the input CRRs Σ. When CRRs Σ are discovered by Algorithm 1 with a top-down approach to cover all tuples, we have Σ^* as the equivalent set of CRRs with data satisfaction.

3) Complexity Analysis: Algorithm 2 has time complexity $O(|\Sigma|^2)$, since Translation tests every two rules. It merges the CRRs sharing the same regression model. As Line 5 in Algorithm 2 suggests, for each time, we test a pair of CRRs for chances of Translation. Since the queue Q contains at most $|\Sigma|$ CRRs, the time complexity between Line 3 and Line 11 is $O(|\Sigma|^2)$. The rest part of Algorithm 2 finds CRRs with shared regression functions, which is terminated in $O(|\Sigma|)$. Thus, the time complexity of Algorithm 2 is $O(|\Sigma|^2)$.

TABLE II
DATASET STATISTICS

Dataset	#Row	#Column	Category
Air Quality [28]	9.4k	18	Time series
Electricity [29]	2075k	12	Time series
BirdMap [3]	407k	4	Time series
Tax	100k	17	Relational
Abalone	4.2k	9	Relational

VI. EXPERIMENTS

We evaluate our algorithms on CRRs discovery by experiments in this section. The evaluations show the advance of sharing models and learning regression under disjunctions (Section VI-B). Through the scalability study of conditional regression discovery in Section VI-C, the performance of our algorithm is stable under the increasing size of instances, and the reports on scalability verify our analysis in previous sections. Under the parameter study in Section VI-D, we could adjust regression bias ρ, varying the efficiency on CRR discovery. For a special case of CRRs, we evaluate the performance of rule compaction on regression trees, and the downstream application, data imputation, in Section VI-E.

A. Experimental Settings

All the experiments are performed on a machine with an i7-10700 CPU and 32 GB RAM. The code and data are available online [27].

1) Datasets: We consider several real datasets to test the effectiveness of CRRs, which fall into two different categories, relational and time-series. The BirdMap dataset [3] traces the GPS position of multiple birds in Africa and Europe from the year 2006 to 2020. The Abalone dataset [30] collects 4.2k tuples about the information of one abalone with its age, sizes such as height, diameter and weights such as its shell, viscera. AirQuality collects time series data with air quality measurements, used in time series statistical analysis like [31]. Electricity is an IoT dataset that collects the data on household energy consumption every minute, included in the benchmark study [32]. Tax is a relational dataset for tax payments, widely used for constraint discovery such as [33]. The dataset statistics are shown in Table II.

2) Default Parameters: The parameters of CRR discovery are composed by the maximum bias ρ_M, and the predicate set $\mathbb{P}$. We set the parameter ρ_M according to the data distribution of each dataset, with 1.0 as default. In regard to the predicates in $\mathbb{P}$, for the domain of each attribute A, we generate predicates $A\phi c$ on each value c with $\phi \in \{>, \leq\}$, in binary by default.

3) Basic Regression Models $\mathcal{F}$: For the regression learning of each part of data, we select simple models as some regression trees [5], [12], i.e., F1 (Linear) [34], F2 (Ridge Regression) [35], and F3 (Multi-layer Perceptron) [36]. Among the selected regression models, F1 and F2 are linear models

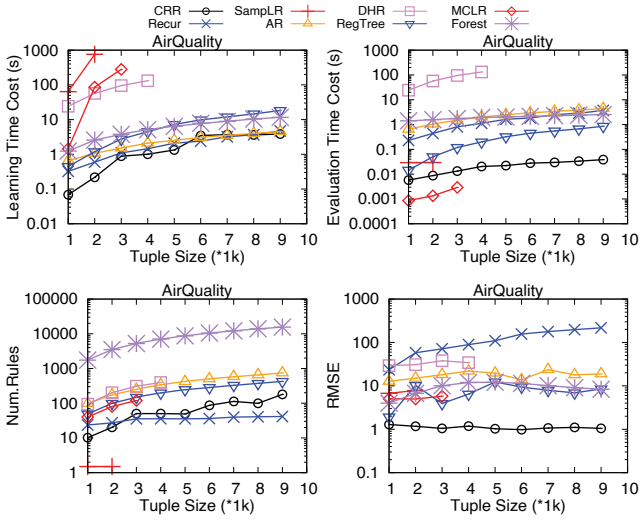

Fig. 2. Training and evaluation instance scalability compared to conditional regression approaches over AirQuality dataset [28]

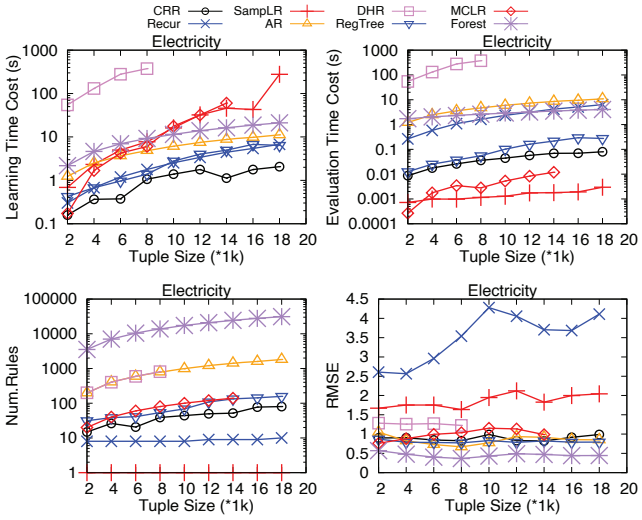

Fig. 3. Training and evaluation instance scalability compared to conditional regression approaches over Electricity dataset [29]

supporting the CRR inference Translation by solving parameters Δ, δ in built-in predicates $\mathbf{x} = \Delta, y = \delta$, while F3 only supports the variance of δ, i.e., when choosing F3 as regression models, model sharing only considers built-in predicate $y = \delta$.

4) Baselines: We consider regression tree, RegTree [5], for regression models, and auto-regression, AR [37], for time series, as baselines. Moreover, the conditional logistic regression methods, SampLR [19] and MCLR [20], generate logistic regression models over large datasets by sampling. The regression forest, Forest [21], builds an additive model for predictions. The harmonic regression, DHR [22], fits the cosine models on time series data by specifying seasonal parameters. Recurrence regression, Recur [23], models the recurrence time of each value in time series.

B. Effectiveness of Conditional Regression Rules

1) Effectiveness of Model Sharing: Figures 2, 3 and 4 demonstrates that model sharing could avoid redundant models and reduce the number of models to train and evaluate. Owing to the extremely high time costs of SampLR, MCLR and DHR in training, some of their results are omitted in larger data sizes. In Figures 2 and 3, we compare all the regression models/time series analysis approaches, since they are all able to handle time series data, with timestamps as conditions. Figure 4 only includes SampLR, MCLR and regression tree, as others are not designed for relational data. Similar results are also observed in the other datasets BirdMap and Abalone, and reported in the full version technique report [38], owing to limited space.

To show that the regression model sharing is meaningful, we demonstrate that both the learning and evaluation time could be reduced by model sharing. Again, the results are shown in Figures 2(a)(b), 3(a)(b) and 4(a)(b). The reduction of learning time is achieved as follows. Rather than learning models over

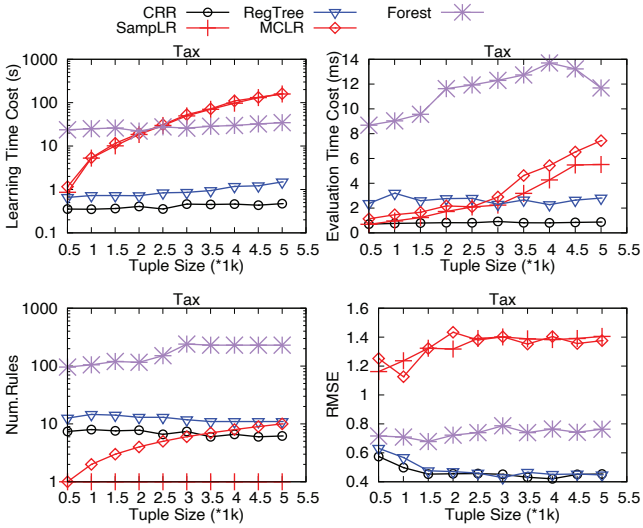

Fig. 4. Training and evaluation instance scalability compared to conditional regression approaches over Tax dataset

each part of data and then detecting the same learned models, our proposal first tries to apply the existing models to a part of data before learning a new model over them. If they can be applied, there is no need to conduct the model learning. Note that learning is more costly than evaluating a model. With possible model sharing, the overall learning time is reduced.

To illustrate that such shared models cannot be obtained by state-of-the-art, we compare the number of regression rules as in Fig. 9. Compared to existing studies, our proposal uses fewer rules to achieve comparable RMSE, in Figures 2(c)(d), 3(c)(d) and 4(c)(d), owing to the shared models. The RMSE of Forest in Figure 3(d) is a bit lower, but needs 100 times more

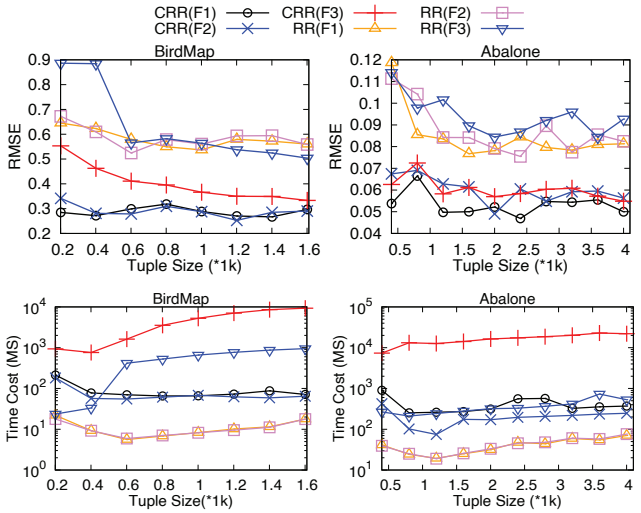

Fig. 5. Instance scalability performance on RMSE and time consumption

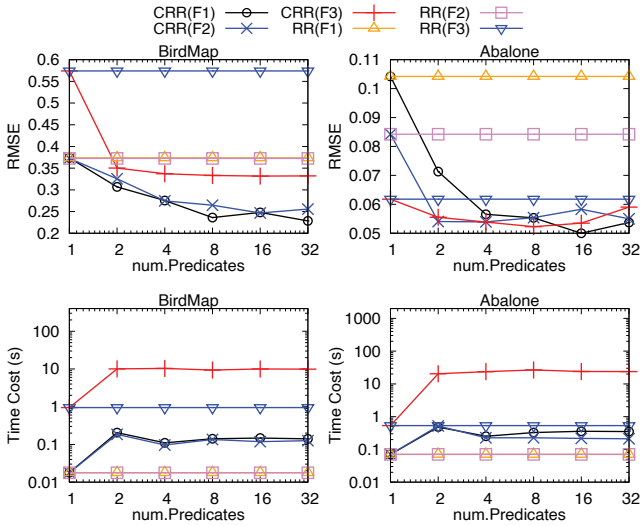

Fig. 6. Predicate scalability performance on RMSE and time consumption

rules and thus significantly higher learning and evaluation time costs than CRR.

2) Effectiveness of Conditions: We here study the performance of using conditions, as one major component of CRR. Figure 5 shows that CRR discovery provides more precise predictions than RRs and is stable as the instance size grows. In Figure 6, with more predicates provided, CRRs show more clear improvement compared to RR. Such an improvement is observed on all the columns in Figure 7. Finally, by ignoring those inaccurate rules with the maximum tolerated bias ρ_M in Figure 8, CRRs show again better RMSE than RR. In short, compared with RRs, CRRs lead to lower regression error but higher time cost, a worthwhile trade-off.

C. Discovery of Conditional Regression Rules

1) Instance Scalability: To find whether the proposed approach can be extended to larger databases with varied regression functions, we introduce the instance scalability as shown in Figure 5, where we find CRRs for all attributes and vary the size of input instance. As the subsets of BirdMap on different birds belong to several isolated data partitions, we include the predicates on each bird, which leads to a natural segregation on the whole dataset. In this way, we study the instance scalability onto one of the years among 2006 to 2020. The first two figures consider the RMSE evaluation over different tuples sizes, while the last two ones study the time consumption on different instance sizes $|I|, I \subseteq D$.

In the first two figures, we notice that our algorithm is stable under different instance scales. Based on proposed Algorithm 1, we generate CRRs through model sharing and avoid discovering new models, which is observed as a major time-consuming workload. Thus, from the last two figures on time consumption over various instance sizes, we notice that our algorithms are scalable to larger data sets with stable performance on prediction errors, e.g., the simple model F1.

2) Predicate Scalability: To find out how our algorithm performances over more precise conditions, we conduct this experiment to study the predicate scalability of our approach. We use Figure 6 to report the behavior of our algorithm over different conditions. As discussed in Section V-A2, the predicates size $|\mathbb{P}|$ decides refinement of conjunctions. When no suitable models found, the results may contain CRRs with higher RMSE.

The first two figures of Figure 6 show the RMSE of the CRRs and three basic regression models, with conditions under different refinements. The last two figures in Figure 6 show the time consumption on varied predicates. As the predicate space $\mathbb{P}$ increases, we would have more conditions in queue for regression learning. From the last two figures, the time cost of Linear method (F1) is almost the same after adding sufficient predicates in $\mathbb{P}$ because a small size of $\mathbb{P}$ is enough to generate reliable CRRs.

3) Column Scalability: Column scalability aims to illustrate whether our approach has performance tilt over the real-world datasets. The experimental results are shown in Figure 7.

Based on the first two figures, we could notice that the RMSE over multiple variables is stable. According to the last two figures, we could notice that the time consumption increases almost linearly with the number of target columns. Thus, based on statistics in Figure 7, our approach provides stable performance on each target variables.

D. Parameter Sensitivity

1) Regression Bias: We evaluate the performances with various regression bias ρ_M over two datasets. The bias ρ_M studied here is applied to all continuous attributes. Among the four figures in Figure 8, we vary the parameter ρ_M, and compare the results of RMSE and learning time consumption. Based on Section III-B and Section V, the parameter ρ_M

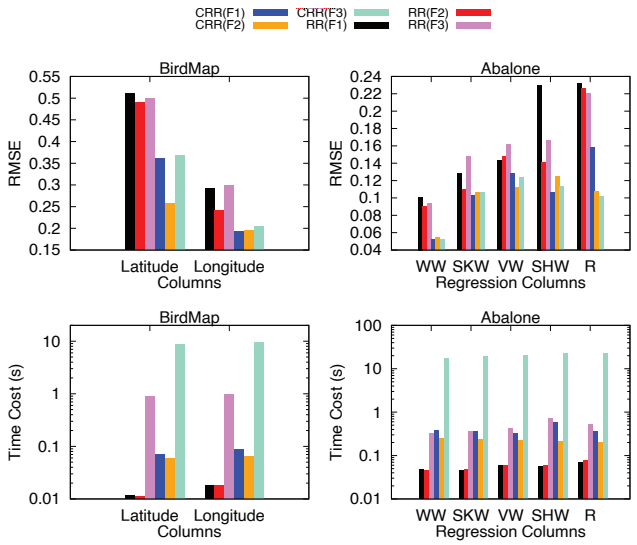

Fig. 7. Column scalability performance on RMSE and time consumption

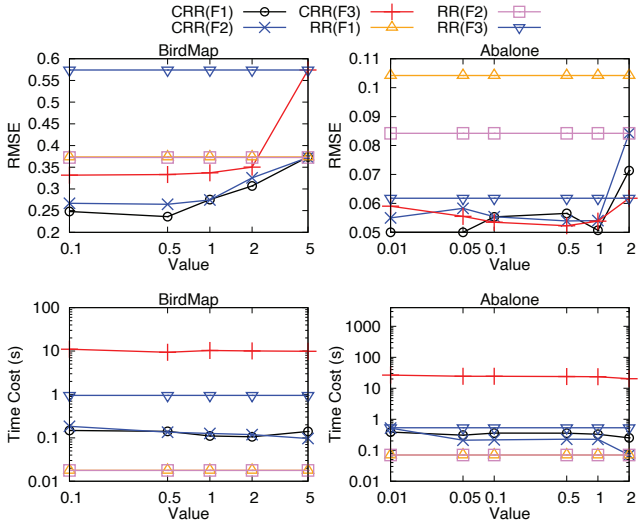

Fig. 8. Parameter study on different regression bias ρ

relates to the validation of regression models. During the CRR discovery, a failure on validating the regression model on condition C will lead to the refinement of C, which could be either precise or overrefined. Thus, a better choice of ρ_M results in lower RMSE of CRR discovery.

In Figure 8, similar to refined conditions caused by large predicate set $\mathbb{P}$, the RMSE results indicate that large regression bias could lead to worse performances, such as $\rho_M = 5$ for attribute Latitude and Longitude in BirdMap dataset. For smaller ρ_M, Algorithm 1 would find conditions with more refinements. Noticed in [4], learning individual models could generate high performance, which is also proved by our data. We could notice that each dataset achieves lowest RMSE at

TABLE III
PERFORMANCE OVER VARIED PREDICATE GENERATORS

Data	Method	Learning (S)	Evaluation (MS)	RMSE	# Rules
BirdMap	Expert	0.496	0.79	**0.254**	6.7
	Binary	0.348	0.76	0.275	5.0
	Random	0.372	0.78	0.283	5.9
Abalone	Expert	0.684	0.91	**0.056**	7.6
	Binary	0.641	0.93	0.057	7.4
	Random	0.644	0.93	0.059	7.6

TABLE IV
PERFORMANCE OF MODEL SHARING PRIORITY

Data	Order	Learning (S)	Evaluation (MS)	RMSE	# Rules
BirdMap	Decrease	**0.331**	0.77	0.269	6.0
	Increase	0.577	0.75	0.275	6.5
	Random	0.496	0.78	0.264	6.5
Abalone	Decrease	**0.681**	0.91	0.047	7.2
	Increase	0.970	0.87	0.046	8.0
	Random	0.730	0.93	0.051	7.0

different ρ_M, indicating that the decision of ρ_M should be varied under different data distributions.

2) Predicate Generation: The predicates $\mathbb{P}$ could be generated both automatically [9]–[13] and manually by expert knowledge [39], and different predicates $\mathbb{P}$ could influence the experimental performance. In addition to Figure 6 on different number of predicates, we add experiments to compare different ways of generating predicates, in Table III.

In short, we include three ways of generating predicates, expert knowledge, binary separation and random separation, and keep the size $|\mathbb{P}|$ of generated predicates the same for fair comparison. Given the size of predicates $\mathbb{P}$, for expert knowledge, we specify several useful separations for subset of $\mathbb{P}$, such as Date $\geq$ 2006-8-11 in Example 2. Then, we use the predicates $A\phi c$ with c decided by random values from the domain, e.g., 2006-9-12 in Date. Inspired by a regression tree with depth limited by n [5], when the specified size is 2^n, the predicates generated by binary separation would segment the domain into 2^{n-1} sections. Random generation finds $\frac{|\mathbb{P}|}{2}$ constants from the domain to construct the predicates $A\phi c$ with operator $\phi \in \{>, \leq\}$.

We use the same strategy [5] on deciding splits in Line 19 of Algorithm 1 with the same predicate size $|\mathbb{P}|$ for fair comparison, and report the performance on different predicate generations in Table III.

In short, Table III shows that different ways of generating predicates $\mathbb{P}$ could influence the experimental effects, even if they share the same size. Among different predicate generations, expert knowledge helps the Algorithm 1 to make better splits in Line 19, and thus leads to lower RMSE. In addition, different predicate generations also lead to differences on time efficiency of Algorithm 1. Since expert knowledge may lead to skewness on using predicates to separate the domains,

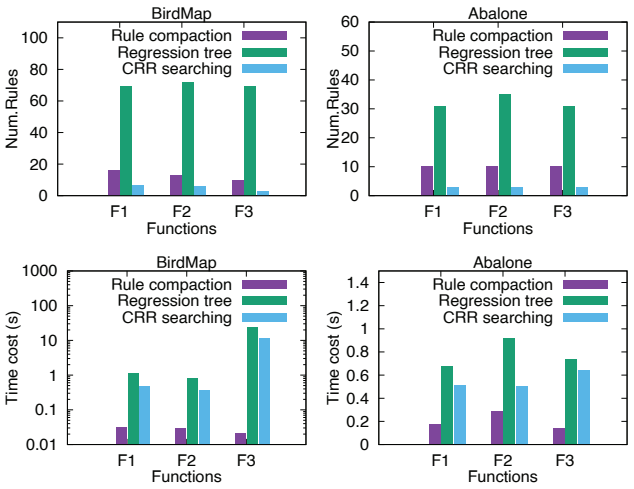

Fig. 9. Performance of rule compaction via translation and fusion on regression tree over different basic regression models

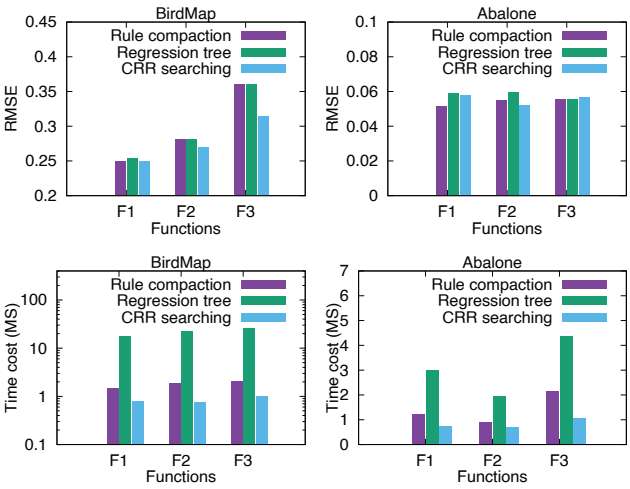

Fig. 10. Performance of missing data imputation using rules with/without compaction via translation and fusion

the corresponding learning time is a bit higher than others, i.e., binary and random generations are observed more time efficient on learning CRRs.

3) Conjunction Ordering: To illustrate the effectiveness of ordering conjunctions, an experiment is conducted in Table IV, by comparing the decreasing, increasing and random index order of conjunctions in Q (Line 3 of Algorithm 1). As Proposition 8 points out, the decreasing order of enumerating conjunctions could share the existing models through fewer predicate splits. As shown in Table IV, considering first the conjunctions with higher probability of being shared (i.e., decreasing order) could enable model sharing cases handled ahead and thus speed up the discovery.

E. Rule Compaction

In this experiment, we evaluate the performance of rule compaction by inference with translation and fusion over the BirdMap and Abalone datasets. We compare the regression tree [5], [12] with regression models F_1, F_2 and F_3 mentioned above as the RR model. As each node in a regression tree represents a CRR with the condition on conjunction, we use the inferences translation and fusion (implemented in Algorithm 2) to merge the nodes and produce a concise set of CRRs as shown in Figure 9.

We use the green bars to represent the number of CRRs a regression tree generated, and use the purple bars for the number of CRRs in the concise set returned by our algorithm. The results show that through rule compaction with fusion and translation inference, our algorithm could cut down the number of CRRs efficiently.

While the discovered concise set of CRRs can be utilized in various downstream application, we provide one of them, missing data imputation, as case study. Again, the compaction of CRRs reduces the size of rules for more efficient imputation, whereas the semantics of the CRR set remains the same after compaction. Therefore, as illustrated in Figure 10, while the imputation RMSE is somewhat comparable, the time cost of the imputation task, using our proposed rule compaction via CRR inference, is significantly reduced compared to the regression tree without rule compaction.

We also compare the results of Algorithm 1, denoted as CRR searching. Figure 9 shows that the rule compaction is efficient on reducing the amount of regression rules learned by regression tree. Even without rule compaction, CRR searching returns less rules than regression tree + rule compaction. The corresponding RMSE however is comparable in Figure 10. It indicates that model sharing is also useful in avoiding redundant model generation.

VII. Conclusion

In this paper, in order to capture the regressions over mixed data distribution, we study conditional regression rules (CRRs) that apply conditionally to only a part of the data. Interestingly, the same regression model may also be shared among different parts of data, such as the seasonal migration of birds similar in different years. To capture such conditionally shared regression models, i.e., redundant semantics, we study the inference of CRRs. With the help of CRR inference, we devise an algorithm for learning and discovering a concise set of CRRs. The extensive experiments on real-world datasets demonstrate that the discovered CRRs are more effective than the corresponding regression models without conditions. In particular, with CRR inference, the number of learned CRRs is significantly reduced without sacrifice of rule semantics.

Note that pruning through chi-squared independence test has been studied as a post-pruning method for decision trees [40]. It is interesting to apply similar strategies to the CRRs discovered by Algorithm 1 to avoid overfitting of conditions. We leave this pruning opportunity on the independence of the variables in the future study.

REFERENCES

[1] L. Xu and M. I. Jordan, "On convergence properties of the EM algorithm for gaussian mixtures," *Neural Comput.*, vol. 8, no. 1, pp. 129–151, 1996. [Online]. Available: https://doi.org/10.1162/neco.1996.8.1.129

[2] H. Yuan and G. Li, "A survey of traffic prediction: from spatio-temporal data to intelligent transportation," *Data Sci. Eng.*, vol. 6, no. 1, pp. 63–85, 2021. [Online]. Available: https://doi.org/10.1007/s41019-020-00151-z

[3] "Birdmap data - gps tracking of storks, cranes and birds of prey, breeding in northern and eastern europe," https://doi.org/10.15468/vnwmrx, 2022.

[4] A. Zhang, S. Song, Y. Sun, and J. Wang, "Learning individual models for imputation," in *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019*, 2019, pp. 160–171. [Online]. Available: https://doi.org/10.1109/ICDE.2019.00023

[5] I. Ilic, B. Görgülü, M. Cevik, and M. G. Baydogan, "Explainable boosted linear regression for time series forecasting," *Pattern Recognit.*, vol. 120, p. 108144, 2021. [Online]. Available: https://doi.org/10.1016/j.patcog.2021.108144

[6] L. Bravo, W. Fan, F. Geerts, and S. Ma, "Increasing the expressivity of conditional functional dependencies without extra complexity," in *ICDE*, 2008, pp. 516–525.

[7] X. Chu, I. F. Ilyas, and P. Papotti, "Discovering denial constraints," *PVLDB*, vol. 6, no. 13, pp. 1498–1509, 2013.

[8] P. Bohannon, W. Fan, F. Geerts, X. Jia, and A. Kementsietsidis, "Conditional functional dependencies for data cleaning," in *ICDE*, 2007, pp. 746–755.

[9] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Wadsworth, 1984.

[10] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986. [Online]. Available: https://doi.org/10.1023/A:1022643204877

[11] J. Ross Quinlan, "Improved use of continuous attributes in C4.5," *J. Artif. Intell. Res.*, vol. 4, pp. 77–90, 1996. [Online]. Available: https://doi.org/10.1613/jair.279

[12] W.-Y. Loh, "Regression tress with unbiased variable selection and interaction detection," *Statistica sinica*, pp. 361–386, 2002.

[13] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Machine Learning, Proceedings of the Thirteenth International Conference (ICML '96), Bari, Italy, July 3-6, 1996*, L. Saitta, Ed. Morgan Kaufmann, 1996, pp. 148–156.

[14] W. Fan, F. Geerts, J. Li, and M. Xiong, "Discovering conditional functional dependencies," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 5, pp. 683–698, 2011. [Online]. Available: https://doi.org/10.1109/TKDE.2010.154

[15] W. Fan, F. Geerts, L. V. S. Lakshmanan, and M. Xiong, "Discovering conditional functional dependencies," in *ICDE*, 2009, pp. 1231–1234.

[16] S. Song and L. Chen, "Differential dependencies: Reasoning and discovery," *ACM Trans. Database Syst.*, vol. 36, no. 3, p. 16, 2011.

[17] S. Kwashie, J. Liu, J. Li, and F. Ye, "Conditional differential dependencies (cdds)," in *Advances in Databases and Information Systems - 19th East European Conference, ADBIS 2015, Poitiers, France, September 8-11, 2015, Proceedings*, ser. Lecture Notes in Computer Science, T. Morzy, P. Valduriez, and L. Bellatreche, Eds., vol. 9282. Springer, 2015, pp. 3–17. [Online]. Available: https://doi.org/10.1007/978-3-319-23135-8_1

[18] B. K. Ray, "Regression models for time series analysis," *Technometrics*, vol. 45, no. 4, p. 364, 2003. [Online]. Available: https://doi.org/10.1198/tech.2003.s166

[19] S. Greenland, J. A. Schwartzbaum, and W. D. Finkle, "Problems due to small samples and sparse data in conditional logistic regression analysis," *American journal of epidemiology*, vol. 151, no. 5, pp. 531–539, 2000.

[20] C. R. Mehta, N. R. Patel, and P. Senchaudhuri, "Efficient monte carlo methods for conditional logistic regression," *Journal of the American Statistical Association*, vol. 95, no. 449, pp. 99–108, 2000.

[21] M. Dantone, J. Gall, G. Fanelli, and L. V. Gool, "Real-time facial feature detection using conditional regression forests," in *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*. IEEE Computer Society, 2012, pp. 2578–2585. [Online]. Available: https://doi.org/10.1109/CVPR.2012.6247976

[22] P. C. Young, D. J. Pedregal, and W. Tych, "Dynamic harmonic regression," *Journal of forecasting*, vol. 18, no. 6, pp. 369–394, 1999.

[23] S.-H. Chang and M.-C. Wang, "Conditional regression analysis for recurrence time data," *Journal of the American Statistical Association*, vol. 94, no. 448, pp. 1221–1230, 1999.

[24] H. Hermes, *Introduction to mathematical logic*, ser. Universitext. Springer, 1973.

[25] W. W. Armstrong, "Dependency structures of data base relationships," in *IFIP Congress*, 1974, pp. 580–583.

[26] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition*, ser. Springer Series in Statistics. Springer, 2009. [Online]. Available: https://doi.org/10.1007/978-0-387-84858-7

[27] R. Kang, "Learn CRRs," https://github.com/marisuki/LearnCRR, 2022.

[28] "Air quality dataset," https://archive.ics.uci.edu/ml/datasets/Air+Quality.

[29] "Individual household electric power consumption dataset," https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption.

[30] "Abalone dataset," https://archive.ics.uci.edu/ml/datasets/abalone.

[31] X. Yi, Y. Zheng, J. Zhang, and T. Li, "ST-MVL: filling missing values in geo-sensory time series data," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, S. Kambhampati, Ed. IJCAI/AAAI Press, 2016, pp. 2704–2710. [Online]. Available: http://www.ijcai.org/Abstract/16/384

[32] M. Khayati, A. Lerner, Z. Tymchenko, and P. Cudré-Mauroux, "Mind the gap: An experimental evaluation of imputation of missing values techniques in time series," *Proc. VLDB Endow.*, vol. 13, no. 5, pp. 768–782, 2020. [Online]. Available: http://www.vldb.org/pvldb/vol13/p768-khayati.pdf

[33] W. Fan, F. Geerts, X. Jia, and A. Kementsietsidis, "Conditional functional dependencies for capturing data inconsistencies," *ACM Trans. Database Syst.*, vol. 33, no. 2, 2008.

[34] E. Hazan and T. Koren, "Linear regression with limited observation," in *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress, 2012. [Online]. Available: http://icml.cc/2012/papers/433.pdf

[35] C. Bishop and M. Tipping, *Bayesian Regression and Classification*, ser. NATO Science Series, III: Computer and Systems Sciences. Netherlands: IOS Press, 2003, pp. 267–285.

[36] "Multi-layer perceptron regressor, scikit-learn," https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html.

[37] K. S. Tuncel and M. G. Baydogan, "Autoregressive forests for multivariate time series modeling," *Pattern Recognit.*, vol. 73, pp. 202–215, 2018. [Online]. Available: https://doi.org/10.1016/j.patcog.2017.08.016

[38] Full Version, "https://sxsong.github.io/doc/crr.pdf."

[39] J. Cai and J. Durkin, "Use of expert knowledge for decision tree pruning," in *Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA*, M. M. Veloso and S. Kambhampati, Eds. AAAI Press / The MIT Press, 2005, pp. 1600–1601. [Online]. Available: http://www.aaai.org/Library/AAAI/2005/sa05-009.php

[40] A. P. White and W. Z. Liu, "Bias in information-based measures in decision tree induction," *Mach. Learn.*, vol. 15, no. 3, pp. 321–329, 1994. [Online]. Available: https://doi.org/10.1023/A:1022694010754