MetaInsight: Automatic Discovery of Structured Knowledge for Exploratory Data Analysis

Pingchuan Ma*
HKUST
Hong Kong, China
pmaab@ust.hk

Rui Ding†
Microsoft Research
Beijing, China
juding@microsoft.com

Shi Han
Microsoft Research
Beijing, China
shihan@microsoft.com

Dongmei Zhang
Microsoft Research
Beijing, China
dongmeiz@microsoft.com

ABSTRACT

Automatic Exploratory Data Analysis (EDA) focuses on automatically discovering pieces of knowledge in the form of interesting data patterns. However, the knowledge conveyed by these suggested data patterns is disjointed or lacks of organization. Therefore, it is difficult for users to gain structured knowledge. As the number of suggested patterns grows, these stand-alone patterns are less likely to motivate users to conduct follow-up analysis, which hinders the suggested patterns from being effectively utilized to facilitate EDA. In this paper, we propose *MetaInsight*, a structured representation of knowledge extracted from multi-dimensional data, which aims to facilitate EDA effectively. Specifically, we propose a novel formulation of basic data patterns to capture essential characteristics of the raw data distribution to achieve knowledge extraction. Then, based on the mined homogeneous data patterns (HDPs) and inter-pattern similarity, *MetaInsights* are identified by categorizing basic data patterns (within an HDP) into commonness(es) and exceptions, thus achieving structured knowledge representation. The commonness(es) and exceptions concretize knowledge obtained by the induction and validation processes, which are two typical analysis mechanisms conducted in EDA. We propose a novel scoring function to quantify the usefulness of *MetaInsights*, an effective and efficient mining procedure and a ranking algorithm to automatically discover high-quality *MetaInsights* from multi-dimensional data. We demonstrate the effectiveness and efficiency of *MetaInsight* (w.r.t. facilitating EDA) through experiments on real-world datasets and user studies on both expert and non-expert users.

ACM Reference Format:
Pingchuan Ma, Rui Ding, Shi Han, and Dongmei Zhang. 2021. MetaInsight: Automatic Discovery of Structured Knowledge for Exploratory Data Analysis. In *Proceedings of the 2021 International Conference on Management of Data (SIGMOD '21), June 20–25, 2021, Virtual Event, China.* ACM, New York, NY, USA, 18 pages. https://doi.org/10.1145/3448016.3457267

1 INTRODUCTION

Exploratory Data Analysis (EDA) emphasizes gaining knowledge of data and is a primary step in facilitating further in-depth analysis [9, 16, 20, 32]. Insights obtained via EDA have been increasingly important for decision making in various domains. As highlighted in Gartner Reviews in 2018 and 2019 [3, 5], automatic EDA [25] is an emerging topic which focuses on automatically discovering pieces of knowledge in the form of interesting data patterns.

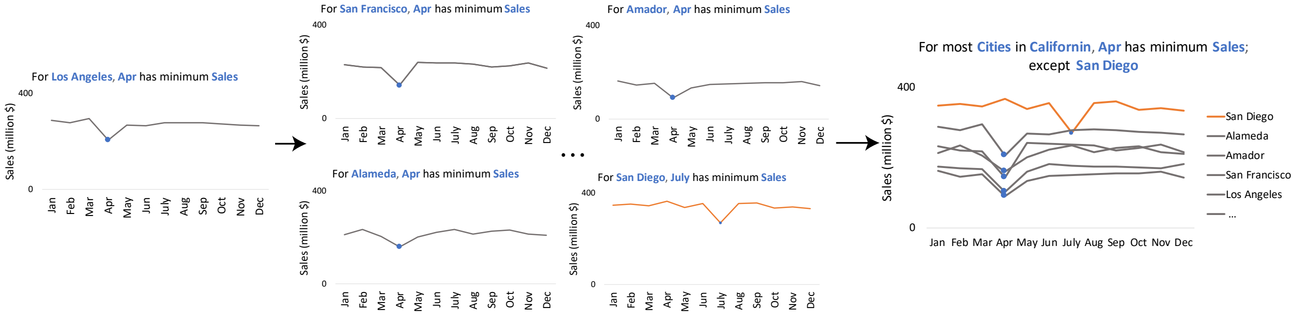
Typically, automatic pattern discovery boils down to the problem of the formulation of data patterns [11, 31]. For example, *QuickInsight* attempts to propose a unified formulation of data patterns as a 4-tuple called insights [11]. Commercial tools have been built on top of this, such as Microsoft Excel Ideas [1] or Microsoft Power BI Quick-Insights [2], which automatically suggests a list of insights to facilitate EDA. However, the knowledge conveyed by these suggested data patterns is disjointed or lacks of organization. Therefore, it is difficult for users to gain structured knowledge (i.e., knowledge of how facts or concepts are organized by certain relations [7, 8, 18]). As the number of suggested patterns grows, these stand-alone patterns are less likely to motivate users to conduct follow-up analysis [17, 22] (also evidenced by our user study). This in turn hinders the suggested patterns being effectively utilized to facilitate EDA.

Let us take a typical EDA iteration cycle as an example. Consider a multi-dimensional data about house sales in California in 2019, as shown in Figure 2(a). A real estate agent, Bob, is conducting EDA over the dataset. He first sees that compared with other months, Los Angeles has had minimal sales in April. Bob then raises an inductive hypothesis: "the other cities have similarly bad sales". With additional data exploration, he learns that this is a common case, because quite a few other cities also have bad sales in April (commonness). Based on the commonness, Bob now concentrates on exceptional cases, and he raises a validity inquiry: "did ALL cities have bad sales in April or are there any exceptions?" After a careful inspection of all cities in California, Bob identifies that San Diego had bad sales in July (exception). Bob has now completed an EDA iteration cycle, and he attains the following insights: he extracts a piece of knowledge (Los Angeles had bad sales in April) and further gains structured knowledge (most cities had bad sales in April except San Diego) from the data. He would thus like to regard the exception (San Diego) as a new entry point for further analysis. We depict the above analysis in Figure 1.

In this paper, we propose *MetaInsight*, a structured representation of knowledge extracted from multi-dimensional data, aiming to facilitate EDA automatically and effectively. Specifically, we propose a novel formulation of basic data patterns to capture essential characteristics of raw data distributions for knowledge extraction. A

[*] Pingchuan Ma's work was done during his internship at Microsoft, Beijing, China.
[†] Corresponding author.

SIGMOD '21, June 20–25, 2021, Virtual Event, China
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8343-1/21/06…$15.00
https://doi.org/10.1145/3448016.3457267



starting from a basic data pattern   induction and validation over homogenous data pattern   *MetaInsight*

Figure 1: An example scenario of leveraging *MetaInsight* to facilitate EDA

homogeneous data pattern (HDP) represents a set of basic data patterns that share certain relations. For example, the different cities in Figure1 form a sibling relation w.r.t. subspaces. Within an HDP, one commonness represents a set of similar basic data patterns. And the remaining data patterns, that do not belong to any commonness(es), are called exceptions. Based on the mined HDPs and inter-pattern similarity, *MetaInsights* are identified by categorizing basic data patterns (within an HDP) into commonness(es) and exceptions. Thus, *MetaInsight* achieves structured knowledge representation. The commonness(es) and exceptions concretize knowledge obtained by the induction and validation processes, which are typical analysis mechanisms conducted in EDA. Commonness(es) concisely expresses to what extent the inductive hypothesis is applicable (i.e., the generality of characteristics of a basic data pattern), and exceptions provide a concentration of unusual cases and how they differ from the general knowledge as a result of a validity inquiry. Note that induction and validation are typical bottom-up or data-driven sensemaking mechanisms in the cognitive process [40].

To automatically discover high-quality *MetaInsights* from multi-dimensional data, we propose a novel scoring function to quantify the usefulness of *MetaInsights*, an effective and efficient mining procedure and a ranking algorithm:

1) *Scoring*. To reflect its usefulness for facilitating EDA, the scoring of *MetaInsight* should consider three heterogeneous factors: importance, conciseness and actionability. A useful *MetaInsight* should convey information (e.g., the associated dimensions, measures or aggregates) that is important. A smaller number of commonness(es) convey more general knowledge and are easier to understand due to conciseness. Despite the purpose of data understanding, it is a further merit if a *MetaInsight* is actionable to motivate users for further analysis. These factors are particularly tailored to score *MetaInsight*, and also deemed as useful for EDA [12, 21].

2) *Mining*. The search space of mining *MetaInsights* grows exponentially w.r.t. the number of dimensions and dimension cardinalities. However, since EDA is typically conducted in an interactive manner, the *MetaInsight* mining procedure is expected to respond within a limited time budget. It is unrealistic to conduct exhaustive enumerations over all subspaces. Thus, designing a strategic and efficient mining procedure is crucial to make *MetaInsight* practical.

3) *Ranking*. In order to select the top-k useful *MetaInsights* among n candidates (which are obtained from the mining procedure), we should not only consider the score of each *MetaInsight* but also the inter-*MetaInsight* redundancy. Due to two *MetaInsights* possibly conveying knowledge from a similar perspective or containing redundant information, how a subset of *MetaInsights* with higher individual scores and lower inter-*MetaInsight* redundancy to be selected needs to be tackled.

Our main contributions are summarized as follows:

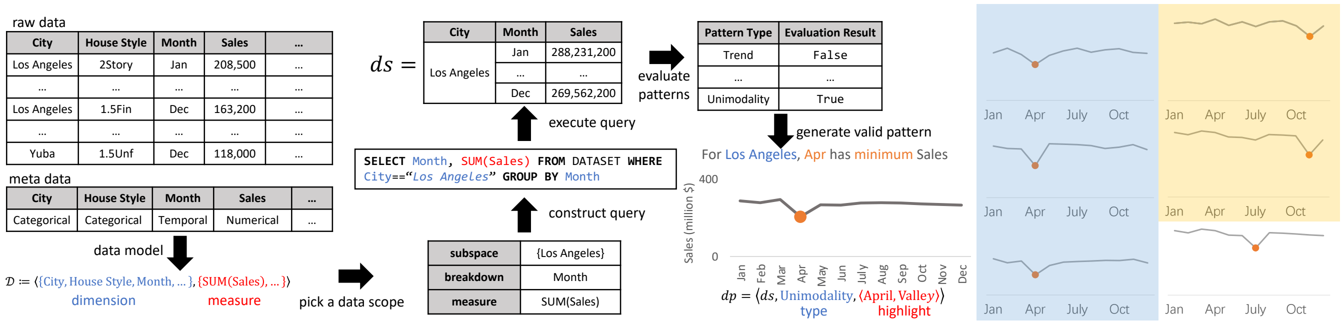
- We propose *MetaInsight*, a structured representation of knowledge from multi-dimensional data to facilitate EDA.
- We propose a novel scoring function to quantify *MetaInsight*'s usefulness w.r.t. facilitating EDA.
- We propose an effective and efficient mining procedure and ranking algorithm to automatically discover and suggest high-quality *MetaInsights* from multi-dimensional data.
- We comprehensively evaluate the effectiveness and efficiency of *MetaInsight* by experiments on real-world datasets and two user studies with expert and non-expert users.

2 PRELIMINARIES

2.1 Multi-Dimensional Data

Multi-dimensional data are conceptually organized in tabular format, where each row is a record and each column is an attribute. An illustrative example of multi-dimensional data is presented at the left-hand side of Figure 2(a). Generally, there are two types of columns in the table: dimensions and measures. Dimensions are used to group or filter records and the values of dimensions are either categorical (e.g., "City") or temporal (e.g., "Month"). Measures, meanwhile, are numerical columns (e.g., "Sales") on which certain aggregates (e.g., SUM) can be performed. Further, we use $\mathcal{D} := \langle Dim, M \rangle$ to represent a multi-dimensional data, where $Dim := \{dim_1, \ldots, dim_d\}$ is the set of d dimensions and M is the set of measures. In addition, we use $\mathbf{dom}(dim_i)$ to denote the domain of dim_i (e.g., $\mathbf{dom}(\text{Month}) = \{\text{Jan}, \cdots, \text{Dec}\}$) and $|dim_i| := |\mathbf{dom}(dim_i)|$ as the cardinality of dim_i (e.g., $|\text{Month}| := |\mathbf{dom}(\text{Month})| = 12$). In the following, we define some important notations in our work.

Subspace. A subspace $s = \{s_1, \ldots, s_d\}$ is a size-d set of filters on each dimension, where $s_i \in \mathbf{dom}(\text{col}_i) \cup \{*\}$. Here, $s_i = \{*\}$ refers to "any" value in dim_i (i.e., empty filter). In the remainder of the paper, we omit empty filters for brevity; e.g., {City: Los Angeles, House Style: *, Month: *} = {City: Los Angeles}, and we further simplify it as {Los Angeles} for notational simplicity.


raw data
City | House Style | Month | Sales | ...
Los Angeles | 2Story | Jan | 208,500 | ...
Los Angeles | 1.5Fin | Dec | 163,200 | ...
Yuba | 1.5Unf | Dec | 118,000 | ...
meta data
City | House Style | Month | Sales | ...
Categorical | Categorical | Temporal | Numerical | ...
data model
$\mathcal{D} := \langle$ {City, House Style, Month, ... }, {SUM(Sales), ... } $\rangle$
dimension
measure
pick a data scope
subspace | {Los Angeles}
breakdown | Month
measure | SUM(Sales)
construct query
SELECT Month, SUM(Sales) FROM DATASET WHERE City=="Los Angeles" GROUP BY Month
execute query
ds =
City | Month | Sales
Los Angeles | Jan | 288,231,200
... | ...
Dec | 269,562,200
evaluate patterns
Pattern Type | Evaluation Result
Trend | False
... | ...
Unimodality | True
generate valid pattern
For Los Angeles, Apr has minimum Sales
400
0
Sales (million $)
Jan Feb Mar Apr May Jun July Aug Sep Oct Nov Dec
$dp = \langle ds$, Unimodality, (April, Valley)$\rangle$
type
highlight
Jan Apr July Oct

(a) Workflow of data pattern generation

(b) Partition of homogeneous data pattern

Figure 2: Illustrative examples

Sibling group. We define the sibling group of subspace s w.r.t. the i-th filter as $SG(s, i) := \{s' \mid \forall j \neq i, s'_j = s_j, s'_i \neq \{*\}\}$. Subspaces within the same sibling group only differ from each other in one non-empty filter, e.g., {City: Los Angeles, April} and {City: Yuba, April} are within the sibling group that is extended by "City".
Breakdown. A *breakdown* dimension is the dimension where group-by operator is performed. A sibling group is generated when a breakdown dimension is applied on a subspace; e.g., when we break down {Los Angeles} by dimension "Month", we obtain the corresponding sibling group SG ={{Los Angeles, Jan}, {Los Angeles, Feb}, $\cdots$}.

2.2 Data Scope

We define a data scope as a 3-tuple.

Definition 2.1. Data Scope

$$ds := \langle subspace, breakdown, measure \rangle \tag{1}$$

A data scope ds corresponds to a raw data distribution, which indicates a sibling group with corresponding aggregate values on the measure. Thus, a data scope specifies the content of interests for data analysis purpose. A data scope example and corresponding raw data distribution are illustrated in the middle of Figure 2(a).

We further define the *impact* of a data scope ds as:

$$\texttt{Impact}_{ds} = \frac{m_{\texttt{Impact}}(ds.subspace)}{m_{\texttt{Impact}}(\{*\})} \in [0, 1] \tag{2}$$

where $\texttt{Impact}_{ds}$ reflects the importance of the data scope against the entire dataset. It can be determined by promoting the data scope regarding any "meaningful measure". Here, we term such a "meaningful measure" as a impact-measure $m_{\texttt{Impact}}$, and denote the value of $m_{\texttt{Impact}}$ as $m_{\texttt{Impact}}(ds.subspace)$. For example, let SUM(Sales) be an impact measure for the multi-dimensional data in Figure 2(a). The impact of data scope ds is the proportion of total sales in Los Angeles against the total sales in California. A real estate agent naturally has more interest in cities with higher sales (i.e., cities with a higher impact). We note that impact is a commonly used metric to evaluate the importance of subspaces [11, 31].

3 FORMULATION

In this section, we give the formal definition of *MetaInsight*. We first define a *basic data pattern* to achieve knowledge extraction by symbolically representing the raw data distribution of a data scope. Grounded on basic data patterns, we design three feasible inter-pattern relations, which lead to the definition of *homogeneous data pattern*. Finally, we define a *MetaInsight* by categorizing basic data patterns within an HDP into commonness(es) and exceptions to achieve structured knowledge representation.

3.1 Basic Data Pattern

We define a *basic data pattern dp* as a 5-tuple:

Definition 3.1. Basic Data Pattern

$$dp := \langle ds, type, highlight \rangle \tag{3}$$

Here, ds is a data scope (3-tuple) which indicates raw data distributions in terms of aggregate values on measure over corresponding sibling groups. *type* indicates a certain perspective to interpret the data distribution. For example, when the breakdown of data scope is a temporal dimension, the distribution forms a time series which can be interpreted from the perspectives of *trend*, *outlier*, etc.

We use $Evaluate(ds, type) \in \{\texttt{true}, \texttt{false}\}$ to denote evaluation function conducted on raw data distribution of ds from a specific perspective, that is, *type*. Here, the function depicts the soundness of the given perspective via an evaluation criterion (e.g., detecting outliers by 3-sigma rule). Once the criterion is satisfied, $Evaluate(ds, type) = \texttt{true}$; otherwise, $Evaluate(ds, type) = \texttt{false}$. At the top-right of Figure 2(a), *Evaluate* is conducted on ds with different types. We also have $Evaluate(ds, \text{Trend}) = \texttt{false}$, but $Evaluate(ds, \text{Unimodality})$ is true, which indicates a valid pattern in ds w.r.t. the unimodality perspective. Note that Unimodality indicates a time series which forms a U-shaped valley or peak shape.

highlight is a type-dependent notion to encode essential characteristics as a result of $Evaluate(ds, type)$. For instance, a unimodality pattern's highlight is position of extreme points. Table 1 illustrates basic data patterns. In summary, our definition of basic data patterns provides a systematic way to extract essential characteristics of raw data distributions symbolically.
Type-induced basic data pattern. Denote $dp(ds, type)$ as a generative function that takes a data scope and a type as inputs and returns a basic data pattern. The detailed definition is as follows:

(1) $dp(ds, type) = \langle ds, type, highlight \rangle$, if $Evaluate(ds, type) = \texttt{true}$;

(2) $dp(ds, type) = \langle ds, \text{Other Pattern}\rangle$, if $Evaluate(ds, type) =$ `false` and $\exists t' \neq type$ s.t. $Evaluate(ds, t') =$ `true`;
(3) $dp(ds, type) = \langle ds, \text{No Pattern}\rangle$, otherwise.

In addition to pre-defined pattern types, we use "Other Pattern" and "No Pattern" as placeholders if the criterion of the specified type does not hold in the data scope. Take the right-hand side in Figure 2(a) as an example. If we evaluate the data scope by Unimodality, then $dp(ds, \text{Unimodality})$ is $\left\langle ds, \text{Unimodality}, \langle \text{Valley, April}\rangle\right\rangle$ since $Evaluate(ds, \text{Unimodality}) =$ `true`; in contrast, $dp(ds, \text{Trend})$ is $\langle ds, \text{Other Pattern}\rangle$ because $Evaluate(ds, \text{Trend}) =$ `false`, but the evaluation on Unimodality is `true`. If the evaluation function returns false for all types, then "No Pattern" is returned.

Table 1: Examples of basic data patterns

Type	Illustrations of Highlights
Outstand #1	subspace with the highest aggregate value
Trend	upward or downward trend
Outlier	position of outlier data points
Seasonality	length of seasonality period
Unimodality	position of extreme point of a U-shaped valley or peak shape.

We implement 11 types of basic data patterns corresponding to 11 perspectives which are commonly used in EDA. Table 1 lists some examples. Although both evaluation criteria and pattern types vary by domain, the design of *MetaInsight* is pattern-type agnostic and extendable. Users can add more domain-specific pattern types to facilitate EDA in their own scenarios. In subsequent discussions, we use "data pattern" as shorthand for "basic data pattern".

3.2 Homogeneous Data Patterns

Our definition of a data pattern makes it capable of extracting essential characteristics of the raw data distribution on a specific data scope in the form of type and highlight. To further provide structured knowledge representation grounded on data patterns, we first illustrate prospective relations among data scopes.

Recall that a data scope $ds = \langle s, b, m\rangle$ is a 3-tuple. Naturally, there are three ways to derive data scopes that are homogeneous w.r.t. ds (i.e., the data scopes that only differ from ds by one element under certain constraints). Considering this, we present the three amenable extending strategies, $\mathtt{Exd}_{s_i}$, $\mathtt{Exd}_m$, $\mathtt{Exd}_b$, on a data scope ds to constitute the homogeneous data scope. Below, we use $ds = \left\langle \text{Los Angeles, Month, } \mathtt{SUM(Sales)}\right\rangle$ as an example.

Subspace Extending. $\mathtt{Exd}_{s_i}$ extends the subspace s to its sibling group $SG(s, i)$ while keeping b and m unchanged:

$$\mathtt{Exd}_{s_i}(\langle s, b, m\rangle) = \left\{\left\langle s', b, m\right\rangle \mid s' \in SG(s, i)\right\} \tag{4}$$

where i specifies which filter of s is used to generate sibling groups. For example, applying $Exd_{\text{Los Angeles}_{\text{City}}}$ on ds, we obtain a set of data scopes that correspond to sales in different cities over months (middle part of Figure 2(a)).

Measure Extending. $\mathtt{Exd}_m$ extends the measure m to all measures while keeping s and b unchanged:

$$\mathtt{Exd}_m(\langle s, b, m\rangle) = \left\{\left\langle s, b, m'\right\rangle \mid m' \in M\right\} \tag{5}$$

Applying $\mathtt{Exd}_{\mathtt{SUM(Sales)}}$ on ds, we have a set of data scopes with different measures (e.g., SUM(Sales), SUM(Profits) and AVG(Profit Rate)) in Los Angeles over months.

Breakdown Extending. $\mathtt{Exd}_b$ extends the breakdown dimension b to all temporal dimensions while s and m remain unchanged. To ensure that the homogeneous data scope is semantically meaningful, we restrict all extended breakdown dimensions to being temporal:

$$\mathtt{Exd}_b(\langle s, b, m\rangle) = \left\{\left\langle s, b', m\right\rangle \mid b' \in Dim, b' \text{ is temporal}\right\} \tag{6}$$

Applying $\mathtt{Exd}_{\text{Month}}$ on ds, we obtain a set of time series of sales in Los Angeles with different time granularities, e.g., over "Day", "Week" and "Month".

With the above extending strategies, we derive a list of data scopes that form a homogeneous data scope:

Definition 3.2. (Homogeneous Data Scope) A homogeneous data scope (HDS) derived from a data scope ds is a set of data scopes that are extended by either one of the above extending strategies.

Definition 3.3. (Homogeneous Data Pattern) An HDP is a set of type-induced data patterns derived from an HDS.

$$\mathtt{HDP}_{type}(\mathtt{HDS}) := \{dp(type, ds) \mid ds \in \mathtt{HDS}\} \tag{7}$$

For example, as is shown in the middle part of Figure 1, the breakdown dimensions and measures of all the data scopes (indicated by charts) are identical (Month and SUM(Sales)) and these charts only differ in the subspace. Thus, they form a subspace extended HDS. These charts also convey HDP which is a set of unimodality-induced data patterns. According to Def. 3.3, data patterns within an HDP can be naturally and meaningfully compared. Thus, we define similarity of two data patterns as a Boolean function Sim:

$$\mathtt{Sim}(dp_i, dp_j) = \begin{cases} \mathtt{True} & dp_i\,.\,\mathtt{t} = dp_j\,.\,\mathtt{t}, dp_i\,.\,\mathtt{h} = dp_j\,.\,\mathtt{h} \\ \mathtt{False} & \text{Otherwise} \end{cases} \tag{8}$$

where $dp_i\,.\,\mathtt{t}$ and $dp_i\,.\,\mathtt{h}$ are short for the *type* and *highlight* of dp_i. In addition to Eqn. 8, $\mathtt{Sim}(dp_i, dp_j)$ always returns `False` when either type of dp_i or dp_j is "Other Pattern" or "No Pattern".

The definition of similarity follows intuition: when two data patterns belong to the same HDP (thus can be compared), and share the same essential characteristics (*highlight*) from the same perspective (*type*), then they are deemed to be similar. As is shown in Figure 2(b), patterns with a valley at April are similar (marked in blue), corresponding to the cities with bad sales in April.

Algebraically, Sim is an equivalence relation, and thus the corresponding equivalence classes form a partitioning of an HDP. We call an equivalence class a *commonness* if its ratio exceeds a threshold τ; the set of all *commonnesses* is called *commonness set.*

Definition 3.4. (Commonness Set) CommSet.

$$\mathtt{CommSet} := \{C \mid C \in \mathtt{HDP/Sim}, \text{ s.t. } \frac{|C|}{|\mathtt{HDP}|} > \tau\} \tag{9}$$

Here, '/' is a quotient operator and HDP/Sim is a quotient set w.r.t. the equivalence relation Sim, where HDP is partitioned into Sim-equivalence classes. Each element in CommSet is denoted as a commonness (a collection of data patterns that share the same type and highlight). Exceptions denote the remaining data patterns that do not belong to any commonness(es). τ is a pre-defined threshold indicating the acceptance rate of the generality of a specifically extracted piece of knowledge. That is, a specific data pattern is deemed as general in its HDP if the proportion of similar data patterns exceeds τ. For the example in Figure 2(b), there are two commonness(es) (one blue and one orange) with an exception, provided that $\tau = 0.3$. With definitions of commonness set and exceptions, we define *MetaInsight*.

Definition 3.5. (*MetaInsight*)

$$\mathit{MetaInsight}_{\text{HDP}} := \langle \texttt{CommSet}, \texttt{Exc} \rangle \text{ s.t. } \texttt{CommSet} \neq \emptyset \tag{10}$$

MetaInsight categorizes an HDP into a `CommSet` and exceptions `Exc` to provide a structured (by homogeneous relations) knowledge representation. *MetaInsight* concretizes knowledge obtained by the induction and validation processes which are typically conducted in EDA: each commonness indicates a piece of general knowledge, and exceptions are used to focus the users on unusual cases and how they differ from general knowledge.

4 APPROACH

In this section, we elaborate on the design of the *MetaInsight* scoring function, mining procedure and ranking algorithm.

4.1 Scoring

In facilitating EDA, there are three essential factors to assess the usefulness of *MetaInsight*: conciseness, actionability and impact. Among these three factors, conciseness plays a central role and is defined on the commonness set and categorization of exceptions. We start by elaborating on how exceptions are categorized.

Categorization of exceptions. As previous mentioned, in the context of EDA, exceptions are typically used to focus the users on unusual cases and how they differ from general knowledge. Therefore, users tend to focus on identifying contrasts against commonness(es) before further inspecting the details of a specific exception pattern. For example, in Figure 1, after obtaining general knowledge ("most cities had bad sales in April"), users can easily understand that San Diego has behaved differently because its highlight (the month of bad sales) is on July rather than April. We categorize such an exception as a "highlight change". Based on the definition of data patterns, we propose an exception categorization that divides exceptions into three categories:

1) `Highlight-Change` denotes an exceptional pattern whose highlight is different from all patterns in the commonness set.

2) `Type-Change` denotes an exceptional pattern whose pattern type is "Other Pattern". For example, "For most Cities in California, Month:Apr has the lowest Sales, except for San Jose, where Sales are distributed evenly."

3) `No-Pattern` denotes an exceptional pattern whose pattern type is "No Pattern". For example, "For most Cities in California, Month:Apr has the lowest Sales, except for Riverside, where Sales do not exhibit any particular patterns."

These three categories are typical ways to categorize exceptions, though, in general, there may be more categories of exceptions by considering the finer granularity of relations of highlights (e.g., partial/complete highlight change). In the following discussion, we denote k as the number of categories of exceptions. Here, we propose a fine-grained *MetaInsight* representation as follows:

Definition 4.1. (*MetaInsight* representation) A *MetaInsight* from an n-length HDP is represented by the $u + v$ categories:

$$\{c_1, \cdots, c_u, e_1, \cdots, e_v\} \tag{11}$$

where $u \geq 1$, $0 \leq v \leq k$, c_i denotes the i-th commonness, and e_j denotes the j-th exception category. The proportion of each category in the *MetaInsight* is denoted as

$$\begin{gathered}\{\alpha_1, \cdots, \alpha_u, \beta_1, \cdots, \beta_v\} \\ \text{s.t.} \sum_{i=1}^{u} \alpha_i + \sum_{j=1}^{v} \beta_j = 1, \alpha_i, \beta_j \in (0, 1], \forall i, \alpha_i > \tau\end{gathered} \tag{12}$$

Conciseness as a core element. The implication of conciseness is two-fold: First, conciseness measures the generality of knowledge from *MetaInsight*; second, conciseness can quantify the human effort to understand *MetaInsight*. As we have categorized all data patterns in *MetaInsight* in Def. 4.1, we first propose an entropy-like formula to quantify the complexity of a *MetaInsight* representation:

$$S = -\left(\sum_{i=1}^{u} \alpha_i \log \alpha_i + r \sum_{j=1}^{v} \beta_j \log \beta_j\right) \tag{13}$$

Intuitively, smaller entropy S indicates better conciseness, and thus it requires less effort to understand. E.g., suppose a *MetaInsight* with $\alpha_1 = 1$; then $S = 0$, which indicates perfect conciseness because this *MetaInsight* is the easiest to understand. Here, r serves as a balancing parameter that distinguishes complexity brought by the commonness(es) and exception categories.

Based on such a consideration, we define conciseness as follows:

$$\texttt{Conciseness}_{\text{raw}} = 1 - \frac{S}{S^*} \tag{14}$$

where S^* is the upper bound of S, which serves as a normalization factor that ensures $\texttt{Conciseness}_{\text{raw}}$ is bounded in $[0, 1]$.

Lemma 4.1. The maximal S of a *MetaInsight* is:

$$S^*(\tau) = \begin{cases} -\log(\tau) + \frac{rk\tau^{1/r}}{e} \log e & k < \frac{(1-\tau)e}{\tau^{1/r}} \\ -\tau \log \tau - r(1-\tau) \log \frac{1-\tau}{k} & k \geq \frac{(1-\tau)e}{\tau^{1/r}} \end{cases} \tag{15}$$

where e is the Natural Base, k is the number of exception categories, r is the same balancing parameter as in Eqn. 13, τ is the threshold of commonness(es) and $S^*(\tau)$ is continuous w.r.t. τ.

Corollary 4.1.1. (Continuity & Monotonicity) $S^*(\tau)$ is continuous and monotonically decreasing w.r.t. τ.

According to the corollary, higher τ brings lower upper bound of entropy S, which is compliant with intuition since higher τ means higher generality of commonness, which offers less complexity.

Regularization for actionability. Despite commonness(es) providing general knowledge concisely, exceptions are more actionable for follow-up analysis. To encourage the presence of exceptions, we design a regularization term to penalize cases without exceptions, namely an indicator function $I_{\sum_i^n \alpha=1}$:

$$\mathtt{Conciseness}_{\text{reg}} = 1 - \frac{S + \gamma I_{\sum_i^n \alpha=1}}{S^*} \tag{16}$$

where γ is a hyperparameter which should be chosen such that $S + \gamma I \leq S^*$ for any possible *MetaInsights*.

Impact as importance measure. We adapt Eqn. 2 for homogeneous data scopes as follows:

$$\mathtt{Impact}_{\mathtt{HDS}} = \sum_{ds \in \mathtt{HDS}} \mathtt{Impact}_{ds} = \frac{\sum m_{\mathtt{Impact}}(ds.subspace)}{m_{\mathtt{Impact}}(\{*\})} \tag{17}$$

We quantify the usefulness of a *MetaInsight* as follows.

$$\mathtt{Score}(I) = f(\mathtt{Conciseness}_{\text{reg}}) \times g(\mathtt{Impact}_{\mathtt{HDS}}) \tag{18}$$

where $f(x)$ and $g(x)$ are monotonically increasing functions $\in [0, 1]$.

Parameters in our implementation. We set hyper-parameter $\tau = 0.5$, which is a simple and natural threshold indicates majority (a well-known ratio to reflect generality). Note that τ could vary in different EDA scenarios. According to our empirical study, the identified *MetaInsight* are not sensitive to the changes of τ in real-world datasets. We further set $k = 3$ and $r = 1$. To ensure that $\mathtt{Conciseness}_{\text{reg}}$ is within $[0, 1]$, we restrict $0 < \gamma < 1 + 0.5 \log k$ and let $\gamma = 0.1$. For Eqn. 18, we let $f(x) = g(x) = x$.

4.2 Mining

Design. To achieve efficient and interactive EDA, *MetaInsight* mining procedure is first designed to be progressive, returning best-so-far results within a pre-specified time budget. We decompose the mining procedure into search, query and evaluation functionalities, and optimize each functionality to improve mining efficiency. During mining process, all qualified *MetaInsight* candidates are stored, which will be used for downstream tasks, such as pattern-indexing, ranking or diversification. Therefore, we intend to make our mining procedure generic for the needs of various downstream tasks.

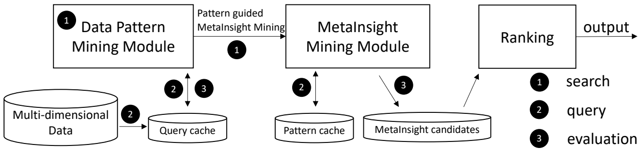


Figure 3: *MetaInsight* Mining Procedure

Workflow. As shown in Figure 3, data pattern mining module first takes multi-dimensional data as input and discovers data patterns in the dataset. Once a data pattern is found, its data scope is extended into several HDSs and corresponding *MetaInsight* compute units are emitted to examine the existence of *MetaInsight* in the *MetaInsight* mining module. Discovered *MetaInsights* are stored and sent to the ranking module when the time budget has been used up.

Conceptually, the mining procedure consists of three main functionalities: search, query and evaluation (shown in Figure 3). The search functionality explores possible (homogeneous) data scopes, emits corresponding compute units and schedules the priority of compute units; the query functionality is executed within each compute unit, and is responsible for querying and aggregating specified (homogeneous) data scopes from raw data; and the evaluation functionality conducts data pattern evaluation or *MetaInsight* evaluation to determine the existence of data patterns or *MetaInsights*. Now, we elaborate on how we optimize these functionalities to achieve efficient *MetaInsight* mining.

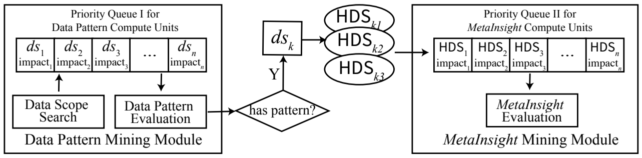


Figure 4: Pattern guided *MetaInsight* mining

4.2.1 *Search.*

To deal with the typical scale of real-world data, it is impossible to enumerate all possible HDP(s) within a realistic time budget. Thus, we are motivated to optimize the search strategy.

Search space analysis. Let dataset $\mathcal{D} = \langle Dim, M \rangle$ where $Dim = \{dim_1, \cdots, dim_d\}$. Recall that the subspace consists of d filters (including empty and non-empty filters). Hence, the size of the subspace $|S|$ is exponential w.r.t. dimensionality and cardinalities, i.e., $\prod \left(|dim_i| + 1\right)$. By further considering the combination with measures, the number of distinct data scopes is greater than or equal to$|M| \times \prod \left(|dim_i| + 1\right)$. Therefore, the number of distinct data scopes is exponential to the number of dimensions and cardinalities.

Pattern-guided mining. To search for *MetaInsight*, a straightforward approach is directly enumerating all HDSs and evaluate *MetaInsight* accordingly. For example, a subspace-extended HDS can be viewed as a data scope augmented with an additional group-by from the subspace-extending dimension. Generally, the number of subspace extended HDSs is about $\times d$ more than all possible data scope size. Thus, the cost of directly enumerating HDSs is prohibitively large. On the other hand, considering data patterns rarely appear in data and a valid *MetaInsight* contains at least $n\tau$ data patterns, most HDSs triggered by straightforward approach contains no *MetaInsight*, in which computations are drastically wasted. Instead, we propose pattern-guided *MetaInsight* mining. Here, we use data pattern mining module to trigger compute units for *MetaInsight* evaluation. The data pattern mining module is responsible for enumerating data scopes and evaluating corresponding data patterns. *MetaInsight* compute units are emitted from three extending strategies in terms of dp, only when dp is identified by data pattern mining module. In this way, the number of emitted *MetaInsight* compute units (and the workload of *MetaInsight* mining module) is drastically reduced. Considering the evaluation of data patterns is relatively lightweight (cf. *MetaInsight* evaluation), pattern-guided mining strategy saves significant computational cost.

Priority queue. As shown at the right-hand side of Figure 4, emitted *MetaInsight* compute units are stored in a queue. Ideally, we would evaluate the compute units with higher scores earlier. At first glance, considering each compute unit only has information of the HDS and its corresponding impact, without knowing its conciseness, we can see that, without conducting an evaluation, the high-scoring *MetaInsights* are randomly hidden in compute units. However, considering that the definition of the *MetaInsights* score (see Eqn. 18), score is monotonic w.r.t. impact. We can sort these compute units by impact (in descending order) and serve the compute units with higher impact earlier. This can be viewed as a best-effort search if we treat the conciseness as a random variable which is independent of impacts. As a result, we propose a priority queue to store *MetaInsight* compute units by setting impact as a priority. Compared with a FIFO (first-in, first-out) queue, our strategy achieves more efficient utilization of the time budget. Likewise, we employ another priority queue for data pattern evaluation.

Table 2: Query APIs

Query API	SQL Query
BasicQuery(ds)	SELECT $ds.measure$, $ds.breakdown$ FROM D WHERE filter = $ds.subspace$ GROUP BY $ds.breakdown$
AugmentedQuery(ds, d)	SELECT $m_1, \cdots, m_{\vert M \vert}$, $ds.breakdown$, d FROM D WHERE filter = $ds.subspace \setminus d$ GROUP BY $ds.breakdown$, d

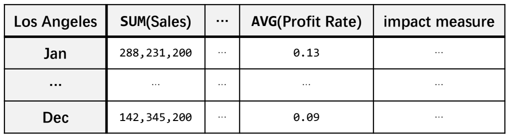
4.2.2 Query. *MetaInsight* mainly conducts queries to obtain data (raw data distributions) of data scopes and HDSs. We elaborate the design of query and cache for efficient *MetaInsight* mining.

Basic query. Denote the query for obtaining data of a data scope as a basic query, which is shown in the 1st row of Table 2. An example of basic query and query result is shown in middle of Figure 2(a).

Augmented query. According to Def. 3.2, an HDS is a collection of data scopes. Thus, a natural way to query for HDS is to use basic queries as building blocks. However, such an approach results in massive calls to an underlying database which further leads to significant performance degradation. For instance, we need $|M|$ basic queries for a measure-extending HDS. Instead, we propose augmented query (2nd row in Table 2) to reduce the number of queries. Compared with basic query, each augmented query requests aggregate values over all measures (highlighted by blue in Table 2), and conducts an additional group-by from dimension d (highlighted by red in Table 2). The result can be represented by multiple 2-dimensional grids, where each grid corresponds to aggregation over a specific measure and grouped by $ds.breakdown$ and d. It is easy to verify that, query result contains not only raw data of data scope *ds* (used in the data pattern mining module, shown at left-most of Figure 3), but also raw data for measure-extending and subspace-extending HDSs generated from *ds* (used in *MetaInsight* mining module, shown at right of Figure 3). Moreover, augmented query pre-fetches measure-extending HDSs for sibling group of *ds.subspace* and subspace-extending HDSs for all the other measures. These pre-fetched results are stored in query cache, which is useful to serve upcoming compute units. Given above considerations, we apply augmented query thorough the mining procedure.

Query cache. We choose the cache unit to be in one-to-one correspondence to data of basic query, which efficiently serves queries for data scopes, HDSs and impact calculation (i.e., basic query from the impact measure). Each cache unit is a 2-dimensional aggregation result grouped-by breakdown, and across all measures with a specific filter. Data generated by augmented queries are stored in such multiple units, where the filter of each unit forms a sibling group *SG*(*ds.subspace*, *d*). An example is shown in Figure 5.

4.2.3 Evaluation. After query results are obtained in a specific compute unit, data pattern mining module and *MetaInsight* mining module start to evaluate existence of (homogeneous) data patterns.



Los Angeles	SUM(Sales)	···	AVG(Profit Rate)	impact measure
Jan	288,231,200	···	0.13	···
···	···	···	···	···
Dec	142,345,200	···	0.09	···

Figure 5: An Example of Query Cache Units

Pattern cache. Data pattern evaluations could be duplicated between the data pattern mining module and the *MetaInsight* mining module, since evaluation of an HDP is almost equal to (without considering pruning) evaluating n data patterns (corresponding to the n data scopes of HDS). We propose a pattern cache to store the data patterns which have been evaluated. According to Def. 3.1, we use the combination of data scope and type as a key to query the evaluation result from the pattern cache.

Pruning. The goal of pruning follows two principles: 1) HDP evaluation should be terminated based on *MetaInsights*' criteria; 2) trivial *MetaInsights* with extremely low scores should be discarded. Based on the principles, we propose two pruning strategies.

Pruning 1. According to Def. 3.5, a valid *MetaInsight* requires at least one commonness with $\frac{|C|}{|\text{HDP}|}$ greater than τ. Therefore, during the HDP evaluation process, which sequentially evaluates data patterns over n data scopes, after evaluating the j-th data scope, we record the ratio of the largest commonness as τ'_j. If $\tau'_j + (n-j)/n < \tau$, then terminate the HDP evaluation process.

Pruning 2. Users often have no interest in a *MetaInsight* with an extremely small score. According to Eqn. 18, we have $\texttt{score} = g(\texttt{Impact}_{\text{HDS}}) \times f(\texttt{Conciseness}) \leq g(\texttt{Impact}_{\text{HDS}})$. Thus, if $g(\texttt{Impact}_{\text{HDS}})$ of a compute unit is already smaller than a threshold (e.g., 0.01), we discard this compute unit from the priority queue.

4.3 Ranking

When suggesting a set of *MetaInsights* to users, two or more *MetaInsights* may convey knowledge with certain overlaps, which is called inter-*MetaInsight* redundancy. Given N *MetaInsights* discovered by the mining procedure (Sec. 4.2), our ranking problem aims to pick $k \ll N$ *MetaInsights* with high individual scores and low inter-*MetaInsight* redundancy. In other words, maximize the "total usefulness" (w.r.t. facilitating EDA) of the k *MetaInsights*.

Here we use $|I|$ to denote the usefulness (w.r.t. facilitating EDA) of an individual *MetaInsight* I.

Definition 4.2. $|I| := \texttt{Score}(I)$

We further define the total usefulness of p *MetaInsights* $I_1, \cdots, I_p$ by adopting inclusion-exclusion principle:

Definition 4.3. $\texttt{TotalUse}(I_1, \cdots, I_p)$

$$\left|I_1 \cup \cdots \cup I_p\right| = \sum_i |I_i| - \sum_{1 \leq i < j \leq p} \left|I_i \cap I_j\right| + \cdots + (-1)^{p-1} \left|I_1 \cap \cdots \cap I_p\right| \tag{19}$$

To fulfill the above definition, we define the overlap of knowledge among p *MetaInsights*:

Definition 4.4. $\texttt{Overlap}(I_1, \cdots, I_p)$

$$\left|I_1 \cap \cdots \cap I_p\right| := \min(|I_1|, \cdots, |I_p|) r(I_1, \cdots, I_p) \tag{20}$$

$r(I_1, \cdots, I_p)$ denotes overlap ratio, which reflects inter-*MetaInsight* redundancy among $I_1, \cdots, I_p$. If $r = 1$, $I_1, \cdots, I_p$ are fully identical, whereas $r = 0$ means there is no overlap of knowledge among $I_1, \cdots, I_p$. Hence, higher overlap ratio indicates more redundancy among these p *MetaInsights*. In our implementation, we propose a heuristic-based method to quantify the overlap ratio of a collection of *MetaInsights*, which can be viewed as a generalization of commonly used overlap coefficient [34].

We further formulate our ranking problem as follows:

$$\mathcal{M} = \underset{\{j_1, \cdots, j_k\} \subseteq \{1, \cdots, N\}}{\operatorname{argmax}} \quad \texttt{TotalUse}(I_{j_1}, \cdots, I_{j_k}) \tag{21}$$

where $\mathcal{M}$ is the suggested *MetaInsights*. Our problem is similar to the result diversification problem discussed in [13]. In result diversification problem, 8 axioms are presented, and it is easy to verify that our objective function Eqn. 21 satisfies all the axioms except stability (note that in the language of [13], our problem has a generalized relevance function but without a distance function). According to [13], maximizing the objective is NP-hard. Therefore, approximations are necessary for efficiency.

Second-order approximation. We make a second-order approximation of Eqn. 21 by discarding higher-order terms (i.e., only preserve first- and second-order terms):

$$\texttt{TotalUse}_{approx}(I_1, \cdots, I_p) \coloneqq \sum_i |I_i| - \sum_{1 \le i < j \le p} \left| I_i \cap I_j \right| \tag{22}$$

Greedy algorithm. To further improve efficiency, we adopt greedy algorithm to solve Eqn. 22. We maintain a selected *MetaInsight* set S (which initially contains the *MetaInsight* with highest score). In each iteration, we pick a new *MetaInsight* which increases $\texttt{TotalUse}_{approx}$ the most and insert it into S. We terminate the iteration when size of S reaches k, and then return S.

Our ranking algorithm (second-order approximation with greedy algorithm) makes reasonable trade-off between accuracy and efficiency. Based on our experiment (Sec. 5.1.3), it is rare to have multiple *MetaInsights* sharing non-trivial overlap and our greedy algorithm often achieves near-optimum.

5 EVALUATION

We evaluate the effectiveness and efficiency of *MetaInsight* by quantitative experiments on real-world datasets (Sec. 5.1), and its usefulness in assisting EDA by two user studies with 18 non-expert users and three expert users (Sec. 5.2).

5.1 Experiments on Real-world Datasets

5.1.1 Setup. **Dataset.** We select 35 datasets from more than 10 domains (e.g., Sales, Environment and Healthcare) to evaluate the effectiveness and efficiency of *MetaInsight*. These datasets are of different scales, ranging from one thousand cells to over one million cells. In particular, we pick four large datasets to evaluate *MetaInsight*'s efficiency at the extreme.

Environment. We conduct all experiments on Windows 10 with an Intel Core i7-8700 and 16 GB RAM. The *MetaInsight* mining procedure is implemented by C#, and we implement a plugin on Microsoft Excel to work with its query interface. We set the configuration as follows: #worker threads = 8 and the maximal filter size in a subspace = 3. We set `COUNT(*)` as impact-measure for all datasets, for simplicity, and we note that the choice of impact-measure only has a negligible effect on evaluating efficiency.

5.1.2 Experiment Design. We evaluate *MetaInsight* from two aspects, namely, the efficiency of the *MetaInsight* mining procedure, and the optimality of our ranking algorithm.

In terms of mining efficiency, we employ several optimizations to boost the search, query and evaluation functionalities. We conduct an ablation study by disabling one of the optimizations or replacing it with a baseline. Specifically, we compare the efficiency by using priority queues vs. using FIFO queues, by enabling query cache vs. disabling query cache, and enabling pattern cache vs. disabling pattern cache. We define *MetaInsight* precision to measure the effectiveness of each optimization. In addition, we also conduct end-to-end comparisons between *MetaInsight* and *QuickInsight* to evaluate *MetaInsight* extra cost, since *MetaInsight* conducts not only mining of basic data patterns (which *QuickInsight* does), but also mining of HDPs. We use total number of emitted queries as metric which reflects major performance cost.

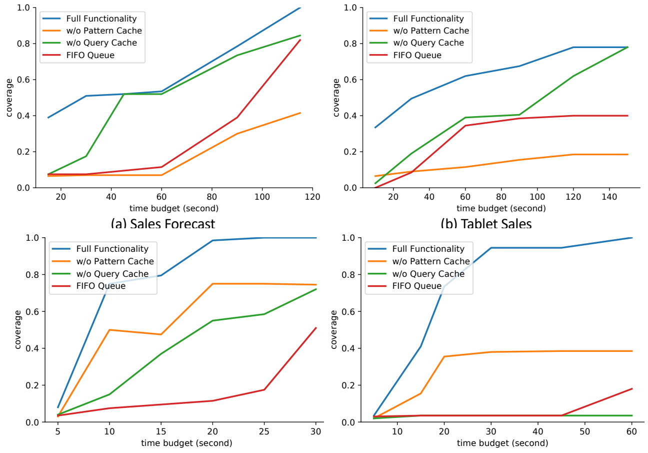
Definition 5.1. (*MetaInsight* Precision) Let $\mathcal{M}$ and $\mathcal{M}_0$ denote two sets of *MetaInsights*, where $\mathcal{M}_0$ is the golden set and $\mathcal{M}$ is the comparative set. The *MetaInsight* precision β is

$$\beta = \frac{|\mathcal{M}_0 \cap \mathcal{M}|}{|\mathcal{M}_0|} \tag{23}$$

In our golden setting, all optimizations are enabled and the time budget is 600 seconds. Then, we compare the golden set $\mathcal{M}_0$ with other results in different methods and compute *MetaInsight* precisions. Each comparative method is repeated under different time budgets on four datasets. Higher precision denotes better performance. $\beta = 1$ means that the mining procedure can achieve equivalent performance under this setting. As a part of our optimizations, in addition to *MetaInsight* precision, we also report the statistics of the query cache and the pattern cache in the 35 datasets to show how much communication and computational cost is saved.

To understand the optimality of our ranking algorithm, we first run a standalone baseline algorithm (neither second-order approximation or greedy algorithm is enabled) to find the globally optimal top-k set of Eqn. 19. Then we use this set as the golden set, and evaluate the optimality of our ranking algorithm. We also compare our algorithm with a rank-by-score algorithm (sorting all the *MetaInsights* by scores in descending order and picking the top-k ones).

5.1.3 Results. Figure 6 reports the precision over different time budgets for the four different settings. The full functionality mining procedure achieves the best performance on all datasets among all time budgets. As we use the results of 600 seconds to be the golden set, we would like to note that the satisfactory results are generated in a much shorter time (e.g., the full functionality mining procedure achieves equivalent performance within 120s in the "Sales Forecast" dataset). However, when either of these optimizations is disabled, there is notable performance degradation in most time budgets. We also find that, with the scale of the dataset growing, the implications of optimizations become increasingly important to the overall performance. In the "Hotel Booking" dataset (the largest dataset among the four, w/ 1M+ cells), the effectiveness of our optimizations is obvious. For a medium-sized dataset (w/



(a) Sales Forecast

(b) Tablet Sales

(c) Credit Card

(d) Hotel Booking

Figure 6: Efficiency of *MetaInsight* mining procedure (in terms of precision) under different datasets and settings

100k-1M cells), e.g., the "Tablet Sales" dataset, when either of the caches is disabled, the mining procedure takes 12x longer (5s vs. 60s) to achieve the same performance, let alone the mining procedure without a priority queue. Also, the importance of the query cache is relatively smaller than the others for the medium-sized datasets; given enough time, the mining procedure can discover good *MetaInsights* as well. However, as there are many duplicated data pattern evaluations, the absence of pattern cache may greatly slow down the performance.

Table 3: Cache Statistics

#Cells	$\#C_q$	r_q	$\#C_p$	r_p
0-1k	0.5	80.4%	179	26.0%
1k-10k	35.9	75.6%	7175	34.8%
10k-100k	38.5	65.8%	1497	37.9%
100k-1M	99.4	74.9%	5594	42.2%
1M+	294	70.6%	3584	50.1%

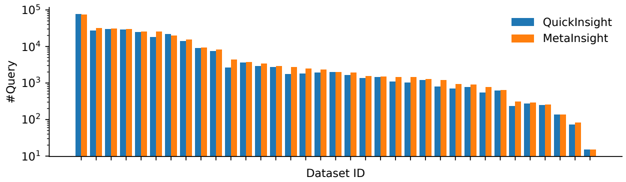


Figure 7: Query Count

We use a total of 35 representative datasets to show the effectiveness of the query cache and pattern cache. We report the cache statistics in Table. 3, where $\#C_q$ and $\#C_p$ denote the size of the query cache and pattern cache, respectively, and r_q and r_p denote the cache hit rate of the query cache and pattern cache. Since the query cache employs a compound data structure (see Figure 5), $\#C_q$ is the memory size in MB (megabytes). As given in the table, a substantial number of data queries and pattern evaluations are eliminated by prefetching in the query cache and pattern cache. On the "Hotel Booking" dataset, we find that more than 80% of queries are prefetched in the query cache, which explains the large performance degradation in Figure 6(d). The results of the pattern cache hit rate also validate our statement in Sec. 4.2, where data pattern evaluations are often duplicated (i.e., a basic data pattern may serve for different HDPs).. For datasets with more than one million cells, more than half the data pattern evaluations are saved.

We further report the total emitted number of queries by *QuickInsight* and *MetaInsight* in Figure 7. *MetaInsight* conducts 17.1% extra queries on average compared with *QuickInsight*. On large datasets (*QuickInsight* emits more than 10, 000 queries), the extra cost of *MetaInsight* is only 7.9% due to the better utilization of query and pattern cache. Overall, *MetaInsight* has modest extra cost and is feasible to be integrated into production.

Table 4: Optimality of *MetaInsight*'s Ranking

Dataset	Algorithm	Time	TotalUse	Precision
Sales Forecast	Baseline	>1min	3.31	N/A
	Our	0.93 s	3.27	0.90
	Rank-by-Score	0.52 s	2.48	0.40
Tablet Sales	Baseline	>1min	6.59	N/A
	Our	1.04 s	6.59	1.0
	Rank-by-Score	0.23 s	4.23	0.30
Credit Card	Baseline	>1min	8.52	N/A
	Our	1.90 s	8.52	1.0
	Rank-by-Score	0.36 s	4.82	0.30
Hotel Booking	Baseline	>1min	8.43	N/A
	Our	2.37 s	8.43	1.0
	Rank-by-Score	0.39 s	3.78	0.30

Table 4 reports the results of *MetaInsight* ranking algorithm optimality (second-order approximation) compared with baseline (accurate algorithm) and rank-by-score algorithm (first-order approximation). The results of our ranking algorithm are in line with baseline algorithm in above datasets. Owing to high-order redundancy being small to have much of an effect on the total usefulness, our second-order approximation captures major redundancy and with much lower time complexity (compared with baseline). It is worth noting that the time cost of baseline is more than one minute (even more than one hour in some datasets) which impedes its use in interactive EDA. However, if we do not take redundancy into account in the ranking algorithm, i.e., the rank-by-score algorithm, the precision is dropped significantly. In summary, our algorithm takes acceptable time and gives a plausible ranking result.

5.2 User Study

5.2.1 Methodology. *MetaInsight* is designed to serve both expert and non-expert users, whose analytical needs are often different. Non-expert users would like to gain a better understanding of data, whereas expert users are more familiar with the data to be analyzed, so they would like *MetaInsight* to aid their in-depth data analysis and decision making. Thus, we conduct two user studies on expert and non-expert users, respectively.

Overall design. For expert users, we compare *MetaInsight* examples (generated by *MetaInsight* mining framework) with *QuickInsight* examples (generated by Microsoft Excel Ideas [1], a commercial tool for facilitating EDA built on top of *QuickInsight*) on the same dataset. For non-expert users, we design an alternative representation of *MetaInsight* called Flat-List Representation (FLR). An FLR is obtained by unfolding all the data patterns within an HDP (of a *MetaInsight*), which presents each data pattern separately. We use FLR as a reference to evaluate *MetaInsight* conciseness and potential information loss, since FLR conveys complete information of HDP (i.e., no conciseness and no information loss).

For both studies, participants need to answer two rating questions to assess each *MetaInsight* example:

Q1: How helpful is this fact for you to understand the data characteristics?

Q2: To what extent do you feel interested to take follow-up analysis?

Here, the answers to both questions range from 1 to 5. A higher score indicates this fact is more helpful (Q1) or the willingness to do some follow-up analysis is higher (Q2). Q1 and Q2 are designed according to the two typical usage scenarios of EDA (data understanding and facilitating further analysis). In addition, for expert users, we also ask the same questions for each *QuickInsight* example to make direct comparison. For non-expert users, we instead ask two more single-choice questions to each *MetaInsight* example to assess the conciseness and potential information loss by using corresponding FLR as reference:

Q3: Compared with FLR, how much easier is it to gain knowledge by *MetaInsight*?

Q4: Compared with FLR, how much useful information is lost (for your purpose of data analysis) by *MetaInsight*?

We assign five choices for Q3: "much easier", "easier", "neutral", "harder", and "much harder". We assign three choices for Q4: "none", "a few (with some information loss but does not affect data analysis)" and "a lot (with noticeable information loss which affects data analysis)". The setup for each user study is elaborated below.

Table 5: Dataset Description

Dataset	User Group	#Rows	#Cols
Survey on Remote Working	Expert	474	24
Car Sales	Non-expert	275	5
Air Pollution Emissions	Non-expert	4862	8
Hiking Trail	Non-expert	141	7

User study for expert users. We invite three data analysts to participate in our expert user study. They are responsible for analyzing a survey about employees' experience of remote work (i.e., working from home) due to COVID-19.

The dataset is the survey feedback collected from 473 respondents from a large organization in the form of an Excel spreadsheet. As shown in the first row of Table 5, the dataset has 24 dimensions corresponding to the 24 single-choice questions in the survey. A total of 473 records are collected, where each record is the response from a specific respondent, with specific answers on the 24 questions. Most questions, such as "I have flexible work hours", are designed with 5 answer choices:: {strongly disagree, disagree, neutral, agree, strongly agree}. The dataset contains only one measure, `COUNT(*)`, and we also use it as an impact measure.

Our study consists of three stages. In the first stage, we compare the top-10 *MetaInsight* examples generated by our implementation and top-10 *QuickInsight* examples generated by Microsoft Excel Ideas [1]. Thus, the comparison is in an end-to-end manner. We follow the convention of *QuickInsight* to create text descriptions and visual representations for *MetaInsight* examples (similar to the right-hand-side part of Figure 1), to minimize potential bias or unfairness during presentation. We educate the participants how to interpret a *QuickInsight* or a *MetaInsight* from text description and visual representation. In the second stage, participants assign scores to the questions of each example. They are encouraged to provide additional comments as well. In the last stage, we ask participants for their overall feedback, and whether or how *MetaInsight* helps to facilitate their analytical tasks.

Since there is no temporal dimension, and with only one measure, all *MetaInsight* examples are extracted from the dataset with subspace-extended HDS. This can be interpreted as the cross-analysis of two questions: one question (primary question) is used to form a sibling group, and another (secondary question) is used as a breakdown. For instance, a *MetaInsight* example is extracted from "I have insufficient workspace setup" (primary question) and "How has your productivity changed vs. working in office" (secondary question). Here, all participants provide positive feedback on the secondary question (represented by an outstanding top-two data pattern, with "about the same" and "more productive" as the two top answers). However, those, who answer "strongly agree" to the primary question, are more negative to the secondary question ("less productive" becomes a highlight).

User study for non-expert users. We invite 18 participants for this user study. They have certain data analysis needs but none of them are professional data analysts. To minimize potential bias, we select participants with diverse roles and experience. Most are experienced in data analysis: more than half of the participants (10/18) conduct data analysis daily or weekly, and about a third of participants (5/18) conduct data analysis monthly. Since non-expert users normally do not have dedicated analytical tasks, we select three public datasets which are common and easy to understand. Table 5 (row 2 to 4) lists the information of these datasets. We present the top three *MetaInsight* examples of each dataset (nine examples in total). For each *MetaInsight* example, we also present it in the FLR, where each data pattern is presented using the same style of *QuickInsight* (similar to the middle part of Figure 1). For non-expert study, we only assign scores to *MetaInsight* examples, where no direct comparison between *MetaInsight* and FLR is conducted.

5.2.2 Key Findings. We report the feedback statistics in Figure 8. The first row shows the feedback histograms from expert users on Q1 and Q2. For each question, the distribution of the ratings of *MetaInsight* examples is compared with the *QuickInsight* examples. The other four charts show results for non-expert users on Q1–Q4 respectively. We identify five findings from the two user studies.

1. *MetaInsight* is helpful for gaining knowledge of data. Both expert and non-expert users provide positive feedback for *MetaInsight* w.r.t. data understanding. For expert users, the average rating of *MetaInsight* examples on Q1 is about 4.0 (3.90 ± 0.74), which is a positive indicator. Moreover, the rating of *MetaInsight* examples is significantly higher than the rating of *QuickInsight* examples in Q1 (3.90 ± 0.74 vs. 2.46 ± 0.99, as is shown at the top-left of Figure 8). As reported by two of expert users, the results of *QuickInsight* are often consistent with their prior knowledge

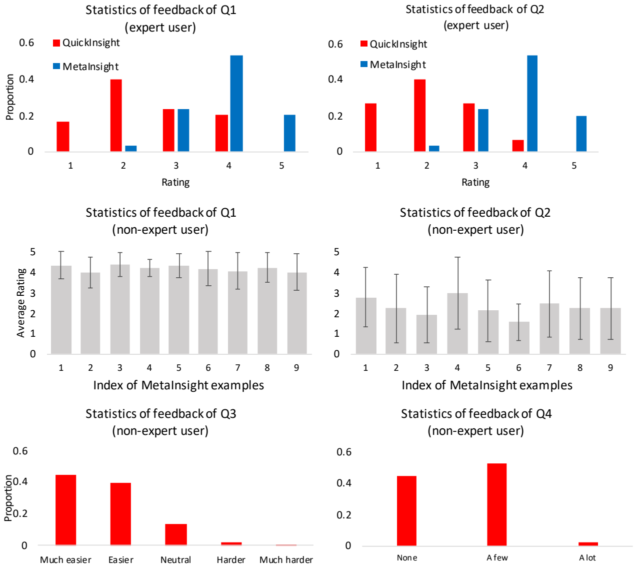


Figure 8: Statistics of feedback.

(note that they are quite familiar with the dataset). For example, the first *QuickInsight* example shows that a significant number of respondents strongly agree with "I feel good spending less time on commute". Such expected knowledge leads to low rating scores. In contrast, the knowledge conveyed by *MetaInsight* examples is more general and robust, and the exceptions of *MetaInsights* often convey "surprising" information contrary to their prior knowledge, which provides valuable information to gain a deeper understanding of data. For non-expert users, the average rating on Q1 is 4.19 ± 0.14 (middle-left in Figure 8) which is higher and has a much smaller variance (compared with 3.90 ± 0.74), indicating that *MetaInsight* provides more value on data understanding for non-expert users.

2. *MetaInsight* motivates follow-up analysis. Both expert and non-expert users provide positive feedback for *MetaInsight* w.r.t. follow-up analysis. For expert users, the average rating of *MetaInsight* examples on Q2 is 3.07 ± 1.06 (top-right of Figure 8), which is significantly higher than the rating of the *QuickInsight* examples (2.13 ± 0.88). For non-expert users, a non-negligible portion of feedback on Q2 (30/162) shows very strong willingness (i.e., rating=5) to perform follow-up analysis after inspecting a *MetaInsight* example, which indicates the effectiveness of *MetaInsight* to motivate follow-up analysis since non-expert users did not have any analytical need for these common datasets beforehand.

We further notice that the presence of exceptions of *MetaInsights* has a strong correlation with the rating on Q2 (willingness to conduct follow-up analysis). For non-expert users, there are three *MetaInsight* examples (#3, #6 and #8) with no exceptions. As shown at the mid-right-side of Figure 8, these three examples have a significantly lower rating than the rest (we apply a t-test on this case and confirm it with a p-value $p = 0.018 \leq 0.05$). The same observation also appears for expert users (2.00 ± 0.81 vs. 3.07 ± 1.06 on Q2), which strongly suggests that exceptions of *MetaInsights* provide good entry points for follow-up analysis.

Below, we present two findings based on detailed feedback from expert users under the context of their analytical tasks.

3. *MetaInsight* helps verify a hypothesis. The question "How has your productivity changed vs. working in office?" (productivity, for short) is important in the survey. An analytical task is to answer the causes of productivity change. The analysts hypothesize that the sufficiency of workspace may be a driving factor for productivity. During the user study, one *MetaInsight* example directly verifies this hypothesis. It uses insufficient workspace as the primary question (short for "I have insufficient workspace setup") and productivity as the secondary question. This *MetaInsight* conveys that all participants are positive for productivity, except those who answer "strongly agree" to insufficient workspace and whose answers to the secondary question are more negative ("less productive" becomes a highlight). Therefore, this *MetaInsight* provides strong evidence that workspace sufficiency is an important factor in productivity.

4. *MetaInsight* helps improve questionnaire design. Two *MetaInsight* examples use "I feel good wearing more comfortable clothing" (comfortable for short) as the secondary question, with the primary questions "I have clear work-life boundary" (work-life balance for short) and "It is difficult to find dining options" (unpleasant dinner for short), respectively. Both *MetaInsight* examples have a commonness and one exception. Each commonness conveys the same knowledge: people wear more comfortable clothes at home (the answers "agree" and "strongly agree" are about half-and-half). However, for respondents who answer "strongly agree" on work-life balance or "strongly disagree" on unpleasant dinner, the answers to the secondary question are almost all "strongly agree". These respondents (corresponding to exceptions) deliver strong emotions which favor working from home over working in the office. As a result, the analysts would want to add more targeted questions to these "optimistic" people.

Below, we present one finding from the non-expert users.

5. *MetaInsight* is a concise and compact representation of knowledge. As shown at the bottom-left, 84% feedback is in agreement that the knowledge conveyed by *MetaInsight* is easier to understand (compared with FLR). More specifically, only 2% of feedback reports the inconvenience of the *MetaInsight* representation. Therefore, *MetaInsight* can be viewed as a concise representation of knowledge. As shown at the bottom-right, regarding information loss in comparison to FLR (Q4), almost all of the feedback (97%) reports that the information loss of *MetaInsights* has no effect on data analysis. The feedback on Q3 and Q4 together indicates that *MetaInsight* is a concise and compact representation of knowledge.

6 DISCUSSION

Alternative structured representation. We use data patterns to encode essential characteristics of raw data distribution to achieve knowledge extraction. We further define similarity over data patterns to identify commonness(es) and exceptions of HDP. An alternative representation can be achieved by directly applying similarity measure over raw data distributions (e.g., KL distance), bypassing pattern extraction stage. Clustering analysis can be conducted subsequently to obtain clusters as commonness(es) and outliers as exceptions. However, compared with raw distributions, extracted patterns effectively encode analysis semantics for EDA. Hence, our similarity measure, based on extracted patterns, is more robust to categorize commonness(es) and exceptions w.r.t. facilitating EDA.

Extensibility of *MetaInsight* formulation. Our definitions of data scope (Def. 2.1) and HDS (Def. 3.2) support a wide range of data analysis needs in practice. According to a recent study which collects over 121, 000 pivot tables (each pivot table can be viewed as a specific analysis result) from 74, 000 real-world Excel spreadsheets [42], about 3/4 of pivot tables can be either represented by data scope (≈ 38%) or by HDS (≈ 36%). We can easily enrich our supported types of data patterns to cover the content of these pivot tables. For the remaining quarter of pivot tables, most of them have multiple breakdown dimensions (e.g., hierarchical pivot table) or multiple measures (e.g., scatter plot), which cannot be directly represented by data scope. We leave extending definition of data scope for multiple breakdown dimensions or measures for future work.

7 RELATED WORK

Pattern mining in multi-dimensional data. Research on mining various types of data patterns from multi-dimensional data has been ongoing for decades. Sarawagi et al. [28] aim to find regions of anomalies in OLAP data cubes. Chen et al. [6] investigate approaches to multi-dimensional regression analysis of time series data for trend and outlier detection. Wu et al. [38] propose promotion analysis to discover top ranked subspaces w.r.t. a given promotion object. Vartak et al. [33] focus on recommending high-deviation patterns by visual interaction. Tang et al. [31] propose complex measure aggregations (composite extractors) to discover latent yet interesting patterns. [11] was the first to propose a unified formulation of various types of data patterns, called *QuickInsight*, where each pattern is represented as a 4-tuple (subspace, breakdown, measure, type). *MetaInsight* extends this formulation by including *highlights* to define a 5-tuple basic data pattern (Def. 3.1), which effectively captures essential characteristics of raw data distributions. By using data patterns as building blocks, *MetaInsight* further constructs HDP and categorizes these patterns into commonness(es) and exceptions based on inter-pattern similarity, achieving a structured knowledge representation. *MetaInsight* overcomes the limitations of *QuickInsight*, such that the suggested commonness(es) and exceptions are tailored to facilitate the induction and validation analysis mechanisms for EDA.

Analysis mechanisms for EDA. It is evident that induction and validation are typical analysis mechanisms for EDA, with support from domains of data analysis, sensemaking and cognitive science. Sarawagi et al. [27, 29] proposes two operators, RELAX and SURPRISE, to facilitate EDA. RELAX evaluates to what extent a specific data pattern is generally applicable by rolling up the subspace (of the given data pattern) to a higher level and searches on sibling subspaces. It follows the induction process on gaining general knowledge of data, which is a typical sub-task in EDA. SURPRISE quantifies how much a fact is contrary to the user's prior knowledge. In EDA iteration cycles, users' prior knowledge is typically gained or refined by the induction process. Therefore, SURPRISE facilitates the validation process by suggesting exceptions w.r.t. general knowledge. In comparison, *MetaInsight* is an automatic and systematic approach to concretize the processes of induction and validation. Ground on a unified formulation of data patterns, we propose three extending strategies to obtain HDP, where RELAX can be viewed as the subspace extending strategy, and we consider heterogeneous factors (importance, consciousness and actionability) to systematically quantify *MetaInsight* usefulness w.r.t. facilitating EDA. In the domain of sensemaking, Zhang et al. [40] state that bottom-up sensemaking activities (e.g., EDA) involve recognizing patterns from data and building on the patterns of similarity and differences to generalize to the structured representation of knowledge. Mai [24] proposes two typical information-seeking processes which are highly relevant to EDA: (1) induction and deduction to examine particular instances and reason towards generalization; (2) validity and reliability to study how compelling the knowledge will be. In summary, our formulation of *MetaInsight* is consistent with human sensemaking mechanisms w.r.t. facilitating EDA.

OLAP and cubing. OLAP cubing techniques have been developed for decades [14]. These techniques provide different strategies to pre-compute cubes to facilitate EDA, including bottom-up [41], top-down [4] or hybrid [39]. Instead of pre-constructing data cubes, *MetaInsight* mining procedure adopts on-demand querying and caching to avoid generating unnecessary cubes. Moreover, *MetaInsight* mining procedure conducts augmented query to efficiently obtain HDS. The design of either caching or pruning is tailored for *MetaInsight*, which is not addressed by general cubing techniques.

Visualization recommendation. To facilitate interactive EDA, a complementary approach is visualization recommendation-aided pattern discovery. Recent works [10, 15, 23, 30, 33, 35, 37] suggest data patterns (in the form of visualization candidates) based on various perspectives [26, 36]. For example, Profiler [19] finds anomalies; SeeDB [33] identifies charts that are largely deviated from a given reference; Zenvisage [30] targets charts that are similar to a given input. We believe the concepts introduced by *MetaInsight*, such as *highlight* of a data pattern, commonness(es) and exceptions to concretize knowledge obtained by the induction and validation processes, can help visualization recommendation systems produce more useful and actionable visualizations.

8 CONCLUSION

We propose *MetaInsight* as a structured representation of knowledge from multi-dimensional data to facilitate EDA. We design a novel scoring function to quantify the usefulness of *MetaInsights*, an effective and efficient mining procedure and a ranking algorithm to automatically discover high-quality *MetaInsights*. We conduct thorough evaluation on both real-world datasets and user studies, demonstrating the effectiveness and efficiency of *MetaInsight* in facilitating EDA.

ACKNOWLEDGE

The authors would like to thank Dr. Shuai Wang for proofreading our manuscript and helpful discussions.

REFERENCES

[1] [n.d.]. https://aka.ms/Excel-Ideas.
[2] [n.d.]. https://aka.ms/QuickInsights.
[3] J Belissent, E Cullen, G Leganza, and J Lee. 2019. *Gartner identifies top 10 data and analytics technology trends for 2019.* Technical Report. Technical report, Gartner.

[4] Kevin Beyer and Raghu Ramakrishnan. 1999. Bottom-up computation of sparse and iceberg cube. In *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*. 359–370.

[5] David CeArley, Brian Burke, Samantha Searle, and Mike J Walker. 2017. *Gartner Top 10 strategic technology trends for 2018*. Technical Report. Technical report, Gartner.

[6] Yixin Chen, Guozhu Dong, Jiawei Han, Benjamin W Wah, and Jianyoung Wang. 2002. Multi-dimensional regression analysis of time-series data streams. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*. Elsevier, 323–334.

[7] Roy B Clariana. 2010. Multi-decision approaches for eliciting knowledge structure. In *Computer-based diagnostics and systematic analysis of knowledge*. Springer, 41–59.

[8] James L Crowley. 2012. Intelligent Systems: Reasoning and Recognition. *training* 1000 (2012), 1.

[9] Jay Devore. 2007. Making sense of data: A practical guide to exploratory data analysis and data mining.

[10] Victor Dibia and Çağatay Demiralp. 2019. Data2vis: Automatic generation of data visualizations using sequence-to-sequence recurrent neural networks. *IEEE computer graphics and applications* 39, 5 (2019), 33–46.

[11] Rui Ding, Shi Han, Yong Xu, Haidong Zhang, and Dongmei Zhang. 2019. Quickinsights: Quick and automatic discovery of insights from multi-dimensional data. In *Proceedings of the 2019 International Conference on Management of Data*. 317–332.

[12] Liqiang Geng and Howard J Hamilton. 2006. Interestingness measures for data mining: A survey. *Comput. Surveys* 38, 3 (2006), 9–es.

[13] Sreenivas Gollapudi and Aneesh Sharma. 2009. An axiomatic approach for result diversification. In *Proceedings of the 18th international conference on World wide web*. 381–390.

[14] Jim Gray, Surajit Chaudhuri, Adam Bosworth, Andrew Layman, Don Reichart, Murali Venkatrao, Frank Pellow, and Hamid Pirahesh. 1997. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. *Data mining and knowledge discovery* 1, 1 (1997), 29–53.

[15] Kevin Hu, Michiel A Bakker, Stephen Li, Tim Kraska, and César Hidalgo. 2019. Vizml: A machine learning approach to visualization recommendation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12.

[16] Andrew T Jebb, Scott Parrigon, and Sang Eun Woo. 2017. Exploratory data analysis as a foundation of inductive research. *Human Resource Management Review* 27, 2 (2017), 265–276.

[17] Manas Joglekar, Hector Garcia-Molina, and Aditya Parameswaran. 2015. Smart drill-down: A new data exploration operator. In *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, Vol. 8. NIH Public Access, 1928.

[18] David H Jonassen. 2000. Toward a design theory of problem solving. *Educational technology research and development* 48, 4 (2000), 63–85.

[19] Sean Kandel, Ravi Parikh, Andreas Paepcke, Joseph M Hellerstein, and Jeffrey Heer. 2012. Profiler: Integrated statistical analysis and visualization for data quality assessment. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*. 547–554.

[20] Matthieu Komorowski, Dominic C. Marshall, Justin D. Salciccioli, and Yves Crutain. 2016. *Exploratory Data Analysis*. Springer International Publishing, Cham, 185–203. https://doi.org/10.1007/978-3-319-43742-2_15

[21] Po-Ming Law, Alex Endert, and John Stasko. 2020. What are Data Insights to Professional Visualization Users? *arXiv preprint arXiv:2008.13057* (2020).

[22] Doris Jung-Lin Lee, Himel Dev, Huizi Hu, Hazem Elmeleegy, and Aditya Parameswaran. 2019. Avoiding drill-down fallacies with VisPilot: assisted exploration of data subsets. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*. 186–196.

[23] Yuyu Luo, Xuedi Qin, Nan Tang, and Guoliang Li. 2018. DeepEye: Towards Automatic Data Visualization. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 101–112.

[24] Jens-Erik Mai. 2016. *Looking for information: A survey of research on information seeking, needs, and behavior*. Emerald Group Publishing.

[25] Tova Milo and Amit Somech. 2020. Automating exploratory data analysis via machine learning: An overview. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 2617–2622.

[26] Xuedi Qin, Yuyu Luo, Nan Tang, and Guoliang Li. 2019. Making data visualization more efficient and effective: a survey. *The VLDB Journal* (2019), 1–25.

[27] Sunita Sarawagi. 2000. User-adaptive exploration of multidimensional data. In *Proceedings of the VLDB Endowment*, Vol. 2000. 307–316.

[28] Sunita Sarawagi, Rakesh Agrawal, and Nimrod Megiddo. 1998. Discovery-driven exploration of OLAP data cubes. In *International Conference on Extending Database Technology*. Springer, 168–182.

[29] Sunita Sarawagi and Gayatri Sathe. 2000. i3: intelligent, interactive investigation of OLAP data cubes. *ACM SIGMOD Record* 29, 2 (2000), 589.

[30] Tarique Siddiqui, Albert Kim, John Lee, Karrie Karahalios, and Aditya Parameswaran. 2016. Effortless Data Exploration with zenvisage: An Expressive and Interactive Visual Analytics System. *Proceedings of the VLDB Endowment* 10, 4 (2016).

[31] Bo Tang, Shi Han, Man Lung Yiu, Rui Ding, and Dongmei Zhang. 2017. Extracting Top-K Insights from Multi-Dimensional Data. In *Proceedings of the 2017 ACM International Conference on Management of Data* (Chicago, Illinois, USA). 1509–1524.

[32] John W Tukey. 1977. *Exploratory data analysis*. Vol. 2. Reading, Mass.

[33] Manasi Vartak, Samuel Madden, Aditya Parameswaran, and Neoklis Polyzotis. 2014. SeeDB: automatically generating query visualizations. *Proceedings of the VLDB Endowment* 7, 13, 1581–1584.

[34] MK Vijaymeena and K Kavitha. 2016. A survey on similarity measures in text mining. *Machine Learning and Applications: An International Journal* 3, 2 (2016), 19–28.

[35] Kanit Wongsuphasawat, Dominik Moritz, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. 2015. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE transactions on visualization and computer graphics* 22, 1 (2015), 649–658.

[36] Kanit Wongsuphasawat, Dominik Moritz, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. 2016. Towards a general-purpose query language for visualization recommendation. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*. 1–6.

[37] Kanit Wongsuphasawat, Zening Qu, Dominik Moritz, Riley Chang, Felix Ouk, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. 2017. Voyager 2: Augmenting visual analysis with partial view specifications. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 2648–2659.

[38] Tianyi Wu, Dong Xin, Qiaozhu Mei, and Jiawei Han. 2009. Promotion analysis in multi-dimensional space. *Proceedings of the VLDB Endowment* 2, 1, 109–120.

[39] Dong Xin, Jiawei Han, Xiaolei Li, and Benjamin W Wah. 2003. Star-cubing: Computing iceberg cubes by top-down and bottom-up integration. In *Proceedings 2003 VLDB Conference*. Elsevier, 476–487.

[40] Pengyi Zhang and Dagobert Soergel. 2014. Towards a comprehensive model of the cognitive process and mechanisms of individual sensemaking. *Journal of the Association for Information Science and Technology* 65, 9 (2014), 1733–1756.

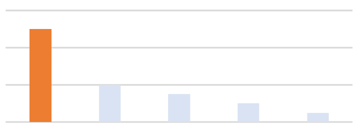
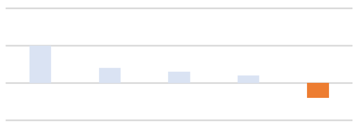
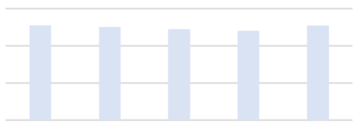
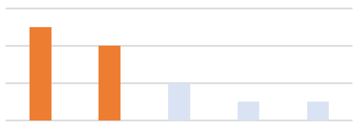
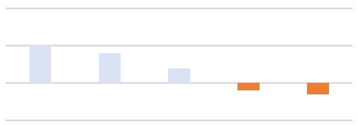
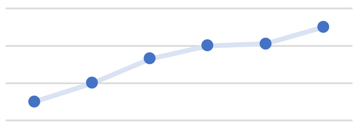
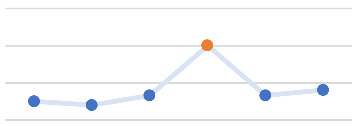
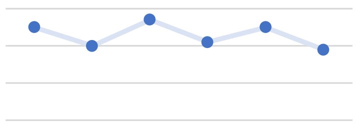
[41] Yihong Zhao, Prasad M Deshpande, and Jeffrey F Naughton. 1997. An array-based algorithm for simultaneous multidimensional aggregates. In *Proceedings of the 1997 ACM SIGMOD international conference on Management of data*. 159–170.

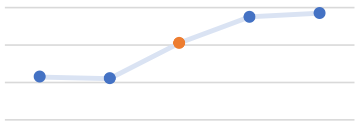
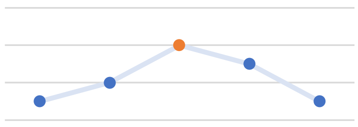
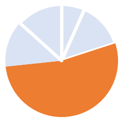
[42] Mengyu Zhou, Wang Tao, Ji Pengxin, Han Shi, and Zhang Dongmei. 2020. Table2Analysis: Modeling and Recommendation of Common Analysis Patterns for Multi-Dimensional Data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 320–328.

9 APPENDIX

9.1 Supported basic data pattern

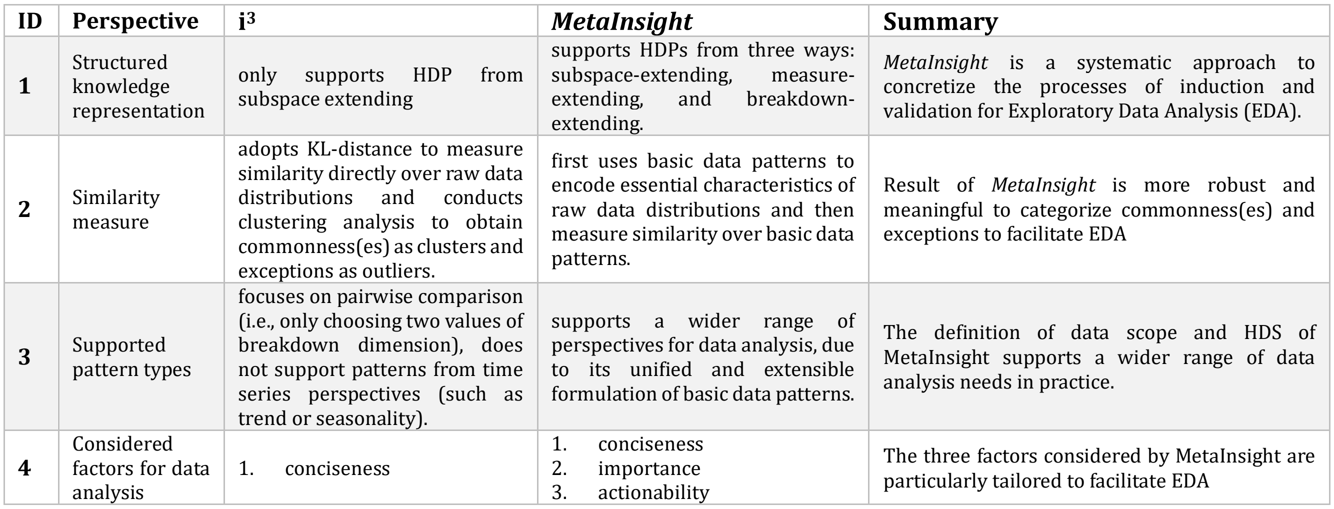
We elaborate on the specification (with textual and visual examples) of supported types of basic data patterns of *MetaInsight* as follows.

Outstanding #1 pattern presents the subspace that is outstanding first in the data scope. The highlight in the pattern is the outstanding #1 subspace. **Example**: San Diego has noticeably higher Sales across all Cities **Highlight**: City: San Diego	
Outstanding #Last pattern presents the subspace that is outstanding last in the data scope. The highlight in the pattern is the outstanding #Last subspace. **Example**: Los Angeles has noticeably lower Profit across all Cities **Highlight**: City: Los Angeles	
Evenness pattern presents that all subspaces are distributed evenly (in probabilistic manner). **Example**: For Los Angeles, The Sales of all HouseStyle are relatively even.	
Outstanding Top-two pattern presents the two subspaces that are outstanding top two in the data scope. The highlight in the pattern is the outstanding top-two subspaces. **Example**: For Amador, 2Story and 1.5Fin have noticeably higher Sales across all HouseStyle. **Highlight**: HouseStyle: 2Story, HouseStyle: 1.5Fin	
Outstanding Last-two pattern presents the two subspaces that are outstanding last two in the data scope. The highlight in the pattern is the outstanding last-two subspaces. **Example**: For San Diego, 2Story and 1.5Fin have noticeably lower Profit across all HouseStyle. **Highlight**: HouseStyle: 2Story, HouseStyle: 1.5Fin	
Trend pattern presents the trend of the measure in the temporal data scope. The highlight in the pattern is the Boolean on the direction of the trend (increasing or decreasing). **Example**: Sales of 2Story is trending upwards over Months. **Highlight**: increasing	
Outlier pattern presents the outliers in a time series w.r.t. 3-sigma rule based on certain non-parametric regression algorithm. The highlight in the pattern is a point list that includes a boolean on the types of outliers (above or below baseline) and their subspaces. **Example**: For San Francisco, Sales has an outlier at April. **Highlight**: above, {Month: April}	
Seasonality pattern presents the repeated pattern in a time series. The highlight in the pattern is the period. **Example**: For San Francisco, Sales shows a repeating pattern over months. **Highlight**: every two months	

Change Point pattern presents that the subspace is a change point in a time series. Specifically, the mean value of the measure before and after the subspace changes significantly. The highlight in the pattern is the position of the change point. **Example**: For Amador, Sales changed significantly from at Month: April. **Highlight**: Month: April	
Unimodality pattern presents that whether a time series forms a U-shaped valley or peak shape. The highlight in the pattern is the position of the extremum and an indication of whether it is a peak or a valley **Example**: For San Diego, April has the maximum Sales. **Highlight**: peak, Month:April	
Attribution pattern presents that the aggregation value of a subspace dominates. The highlight is this subspace. **Example**: For Amador, 2Story accounts for the majority of Sales. **Highlight**: HouseStyle: 2Story	

9.2 Comparison and analysis on i^3

i^3 (eye-cube) [29] is an OLAP system, which integrates pioneering work [27, 28] to facilitate automatic data analysis. The code of i^3 was open sourced in 2001, but it is currently out-of-date due to lack of maintenance (last maintained date was in 2002). To make fair and comprehensive comparison, we refine some of its functionalities and improve query efficiency, as will be elaborated shortly. We commit to release our refined version of i^3 for reproducibility.



ID	Perspective	i³	*MetaInsight*	Summary
1	Structured knowledge representation	only supports HDP from subspace extending	supports HDPs from three ways: subspace-extending, measure-extending, and breakdown-extending.	*MetaInsight* is a systematic approach to concretize the processes of induction and validation for Exploratory Data Analysis (EDA).
2	Similarity measure	adopts KL-distance to measure similarity directly over raw data distributions and conducts clustering analysis to obtain commonness(es) as clusters and exceptions as outliers.	first uses basic data patterns to encode essential characteristics of raw data distributions and then measure similarity over basic data patterns.	Result of *MetaInsight* is more robust and meaningful to categorize commonness(es) and exceptions to facilitate EDA
3	Supported pattern types	focuses on pairwise comparison (i.e., only choosing two values of breakdown dimension), does not support patterns from time series perspectives (such as trend or seasonality).	supports a wider range of perspectives for data analysis, due to its unified and extensible formulation of basic data patterns.	The definition of data scope and HDS of MetaInsight supports a wider range of data analysis needs in practice.
4	Considered factors for data analysis	1. conciseness	1. conciseness 2. importance 3. actionability	The three factors considered by MetaInsight are particularly tailored to facilitate EDA

Figure 9: Summary of comparison between *MetaInsight* and i^3

9.2.1 Conclusion. Figure 9 summarizes the key results. #1, #3 and #4 indicates that *MetaInsight* is a systematic approach to facilitate EDA. Regarding #2, i^3 adopts a similarity measure (KL-distance) over raw data distributions whereas *MetaInsight* adopts a similarity measure based on pattern extraction. Our experimental results show that, in terms of categorization of commonness(es) and exceptions, *MetaInsight* is more robust and meaningful. #2 provides additional support to our discussion on “alternative structured representation” (see Sec. 6). Details are elaborated in the following.

9.2.2 Experiment. **Formulation.** Figure 10 shows an illustrative example of visualizing the result of RELAX operator. This example further indicates the way to formulate i^3 result. To be precise, it can be viewed as a subspace-extending HDS (extended by country), where the breakdown contains only two values (breakdown is “year”, with values are 1993 and 1994).

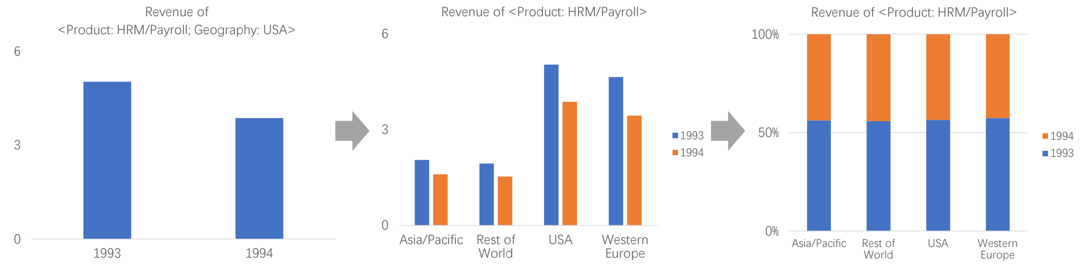


Figure 10: Visualization of i^3 results. Left Chart: an initial pattern presented to analyst – a drop of revenue from 1993 to 1994 at USA; Mid Chart: by applying RELAX on Geography – all regions have the similar revenue drop from 1993 to 1994; Right Chart: a refined visualization – using stacked column chart to better convey similarity of distributions.

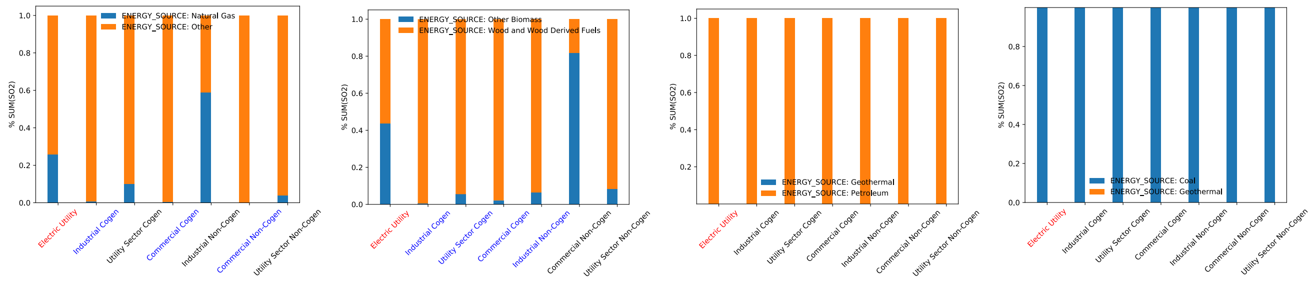
Code and Additional Implementation. The original codebase of i^3 is available at https://www.cse.iitb.ac.in/~sunita/icube. We commit to release our refined version for reproducibility. In particular, we made following refinements to enable fair and comprehensive comparison:

(1) We make full automation on cell selection and allow i^3 progressively explore all specified ranges of cells.
(2) We design a query cache module to improve efficiency.
(3) We provide natural language descriptions and charting (which is used for *MetaInsight*) to present its result.
(4) We design a reasonable ranking algorithm to prioritize the presentation of i^3 patterns (i^3 does not have ranking module, whose original result is overwhelming).
(5) We design a handy interface to make hyper-parameters configurable.

Setup. We conducted experiments on “Air Pollution Emissions”, which is also used in user study. We use specific cases to present our findings. Based on our observation (by inspecting results generated by i^3), our findings are generally applicable to arbitrary datasets.

9.2.3 *Key Result.*

We present primary two findings based on comparison between i^3 and *MetaInsight*. We omit several other findings which are less relevant (mainly due to other limitations of i^3, such as high redundancy, failure of applying KL-distance to negative values, etc.)



(a) Result #1

(b) Result #2

(c) Result #3

(d) Result #4

Figure 11: Illustrative results of i^3

1. results of i^3 is less robust due to measuring similarity over raw data distributions. Figure 11(a) shows an example of result obtained by i^3 RELAX operator, where x-axis denotes the extended subspaces, orange and blue bars indicate the two values (“Other” and “Natural Gas”) from breakdown dimension “ENERGY_SOURCE”, y-axis denotes the proportion of measure (`SUM(SO2)`, SO2 pollution)). In the stacked column chart, exceptions are marked in blue and the remaining bars form a commonness set.

From data analysis perspective, users may feel “Other” dominates the SO2 pollution, except “Industrial Non-Cogen”. However, i^3 reports that the distribution of “Industrial Non-Cogen” is not an exception, but the 2nd, 4th and 6th bars are exceptional. The unwanted result is due to the limitations of KL-distance, which does not consider analysis semantics, but directly calculates the similarity over raw data distributions. Figure 11(b) shows another similar example, where “Commercial Non-Cogen” is categorized as a commonness, while it actually appears different.

2. results of i^3 may be trivial due to the lack of impact and actionability considerations. As shown in Figure 11(c) and Figure 11(d), the results with higher scores from i^3 tend to be with single commonness set but without exceptions. Because “Geothermal” has no emission on SO2. Hence, for any pair of values from “ENERGY_SOURCE”, if contains “Geothermal”, the distribution is identical across all extended subspaces. As a result, the result is trivial and highly redundant for data analysis.

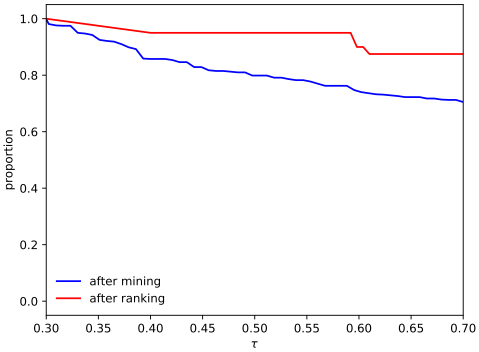


Figure 12: Proportion of identified *MetaInsight* when τ increases

Empirical Analysis. We note that above low-quality cases take a considerable proportion of the results produced by i^3. We manually inspect 100 outputs of i^3 for empirical analysis. First, we find that exceptions are incorrectly categorized in 12 out of top-100 outputs, which are similar to Figure 11(a) and Figure 11(b). Second, in 25 out of these outputs, the aggregation value of measure takes a positive value on one breakdown value but zero on the other breakdown value, across all extended subspaces. This is similar to to Figure 11(c) and Figure 11(d). In summary, over 1/3 results of i^3 are less useful for EDA corresponding to the above findings.

9.3 Choose appropriate threshold τ

Setup and Design. We conducted an experiment to see how the identified *MetaInsight* changed by varying τ in real-world datasets. We choose the same four large datasets which are used for evaluation in the paper. Considering when τ increases, the output result is a strict subset of the result with a smaller τ (according to the definition of *MetaInsight*, Def. 3.5). We use the result under $\tau = 0.3$ are a reference (i.e., the collection of *MetaInsights* identified by using $\tau = 0.3$) and calculate the proportion of *MetaInsight* remained for different $\tau \in [0.3, 0.7]$.

We consider the identified *MetaInsights* in two stages: a full set of *MetaInsight* identified at mining stage ("after mining" in Figure 12), and the top-k *MetaInsight* suggested after ranking ("after ranking" in Figure 12). In the experiment, we set $k = 10$.

Result. As shown in Figure 12, our experiment shows stable behavior in both two curves of "after mining" and "after ranking". Both of the two curves are decreasing slowly. For instance, when τ increases from 0.3 to 0.5, only $\approx$ 20% of the identified *MetaInsights* are discarded on "after mining", and only $\approx$ 5% of the suggested top-10 *MetaInsights* are changed on "after ranking". This is not surprising, because *MetaInsights* with larger size of commonness(es) tend to have higher score, which are more likely to be retained in the final suggested top-10 list. In summary, our experiment shows that the identified *MetaInsights* are not sensitive to the changes of τ.

9.4 Inter-*MetaInsight* redundancy

Because *MetaInsight* can be presented in the form of HDPs, we consider inter-*MetaInsight* redundancy as inter-HDP redundancy. Considering Def. 3.3, an HDP is derived from an HDS with a pattern type. We only consider redundancy if two HDPs have same HDS extending strategy and pattern type.

Definition 9.1. Subspace overlap ratio

$$r_{\mathsf{sub}}\left(s_1, \ldots, s_p\right) = \frac{s_1 \cap \cdots \cap s_p}{\min\left(s_1, \cdots, s_p\right)} \tag{24}$$

Where s_i denotes the set of non-empty filters of the subspace w.r.t. I_i. Note that Def. 9.1 is a generalization of overlap coefficient [34]. For p subspace-extending HDPs $h_1 \cdots h_p$ (with same pattern type), we propose the following overlap ratio:

$$\begin{aligned} r_{\mathsf{s}}(h_1, \cdots, h_p) :=& w_{11} r_{\mathsf{sub}}(h_1.s^*, \cdots, h_p.s^*) + w_{12}\mathbb{1}_i \\ &+ w_{13}\mathbb{1}_{\mathsf{m}} + w_{14}\mathbb{1}_{\mathsf{b}} \end{aligned} \tag{25}$$

where $\mathbb{1}_i$ is the indicator function that returns 1 if the extended filters s_i are identical in all HDPs, $\mathbb{1}_{\mathsf{m}}$ and $\mathbb{1}_{\mathsf{b}}$ return 1 if the measure and breakdown dimension, respectively, are the same in all HDPs, $w_{11}, \cdots, w_{14}$ are weights of different factors and $w_{11} + \cdots + w_{14} = 1$. Likewise, we have the definition of overlap ratio among measure-extended and breakdown-extended HDPs:

$$r_{\mathsf{m}}(h_1, \cdots, h_p) := w_{21} r_{\mathsf{sub}}(h_1.s^*, \cdots, h_p.s^*) + w_{22}\mathbb{1}_b \tag{26}$$

$$r_{\mathsf{b}}(h_1, \cdots, h_p) := w_{31} r_{\mathsf{sub}}(h_1.s^*, \cdots, h_p.s^*) + w_{32}\mathbb{1}_m \tag{27}$$

Now we are ready to define overlap ratio r in the general form. Given p *MetaInsights* $I_1, \cdots, I_p$, the overlap ratio among these *MetaInsights* is

$$r\left(I_1, \cdots, I_p\right) = \begin{cases} 0 & \text{Cond}\left(I_1, \cdots, I_p\right) \\ r\left(I_1.\text{hdp}, \cdots, I_p.\text{hdp}\right) & \text{otherwise} \end{cases} \tag{28}$$

where Cond $\left(I_1, \cdots, I_p\right)$ holds if $I_1, \cdots, I_p$ have different extending strategies or different pattern types.

9.5 Proof of Lemma. 4.1

To prove the Lemma. 4.1, we first present and prove the following lemma.

Lemma 9.1.

$$x \log x + y \log y < (x+y)\log(x+y) \tag{29}$$

where $x > 0, y > 0$.

Proof.

$$\begin{aligned} & x \log x + y \log y < (x+y)\log(x+y) \\ \Leftrightarrow & x(\log x - \log(x+y)) < y(\log(x+y) - \log y) \end{aligned} \tag{30}$$

Since $\log x$ is a monotonically increasing function, $\log(x+y) - \log x > 0$ when $y > 0$. We have $x(\log x - \log(x+y)) < 0$ and $y(\log(x+y) - \log y) > 0$. The inequality holds.

Then, we analyze the upper bound of S when fixing λ, where $\lambda = \sum_i^n \alpha_i$.

Lemma 9.2. (Upper bound of S when fixing λ)

$$S(\lambda) \leq -\left(\lambda \log \tau + r(1-\lambda)\log\frac{1-\lambda}{k}\right) \tag{31}$$

Proof. Recall that $S = -\left(\sum_{i=1}^{u} \alpha_i \log \alpha_i + r \sum_{j=1}^{v} \beta_j \log \beta_j\right)$. Applying Lemma. 9.1, we have that $-\sum_{j=1}^{v} \beta_j \log \beta_j$ reaches the upper bound when $v = k$. And

$$-r\sum_{j=1}^{k} \beta_j \log \beta_j \leq -r \sum_{j=1}^{k} \frac{1-\lambda}{k} \log \frac{1-\lambda}{k} = -r(1-\lambda)\log\frac{1-\lambda}{k} \tag{32}$$

since the information entropy is maximized when the random variable follows uniform distribution. Now we apply the above strategy to the left term in Eqn. 13. Since there are at most $\left\lfloor \frac{\lambda}{\tau} \right\rfloor$ commonness(es), we have

$$-\sum_{i=1}^{u} \alpha_i \log \alpha_i \leq -\sum_{i=1}^{\lambda/\tau} \tau \log \tau = -\lambda \log \tau \tag{33}$$

The upper bound is reached when $\left\lfloor \frac{\lambda}{\tau} \right\rfloor = \frac{\lambda}{\tau}$ and α_i are equal. Hence, we apply Eqn. 32 and Eqn. 33 and prove the lemma.

Now, we prove Lemma. 4.1.

Proof. Let $S^*(\lambda)$ denote the upper bound of $S(\lambda)$, according to Lemma. 9.2. We have

$$\frac{dS^*(\lambda)}{d\lambda} = \frac{1}{\ln 2}[-\ln \tau - r \ln k + r \ln(1-\lambda) + r] \tag{34}$$

Let $\frac{dS^*(\lambda)}{d\lambda} = 0$, we have $\lambda^* = 1 - \frac{k\tau^{1/r}}{e}$. However, $\lambda^* = 1 - \frac{k\tau^{1/r}}{e}$ is achievable only when $k < \frac{(1-\tau)e}{\tau^{1/r}}$ since $\lambda = \sum \alpha_i \geq \alpha_i > \tau$. Considering that $\frac{d^2S^*(\lambda)}{d\lambda^2} = -\frac{r}{(1-\lambda)\ln 2} < 0$, λ^* is the only maximum. So, when $k \geq \frac{(1-\tau)e}{\tau^{1/r}}$, the maximum of $S^*(\tau)$ is achieved when $\lambda = \tau + \epsilon$, where ϵ is an infinitesimal positive value. Put the two condition together, we have the lemma. Also, the continuity can be simply obtained by checking the value of the boundary.