A Framework for Measuring Differences in Data Characteristics

Venkatesh Ganti* Raghu Ramakrishnan†
Department of Computer Sciences, University of Wisconsin-Madison
1210 W. Dayton St, Madison, WI 53706
{vganti,raghu}@cs.wisc.edu

Johannes Gehrke
Department of Computer Sciences, Cornell University
johnannes@cs.cornell.edu

Wei-Yin Loh‡
Department of Statistics, University of Wisconsin-Madison
loh@stat.wisc.edu

Abstract

A data mining algorithm builds a model that captures interesting aspects of the underlying data. We develop a framework for quantifying the difference, called the *deviation*, between two datasets in terms of the models they induce. In addition to being a quantitative, intuitively interpretable measure of difference, the deviation between two datasets can also be computed very fast. Our framework covers a wide variety of models including frequent itemsets, decision tree classifiers, and clusters, and captures standard measures of deviation such as the *misclassification rate* and the *chi-squared* metric as special cases. We also show how statistical techniques can be applied to the deviation measure to assess whether the difference between two models

* Contact author. Supported by a Microsoft Graduate Fellowship.
† This research was supported by Grant 2053 from the IBM corporation.
‡ Supported by Army Research Office grant DAAG55-98-1-0333.

is significant (i.e., whether the underlying datasets have statistically significant differences in their characteristics), and discuss several practical applications.

1 Introduction

The goal of data mining is to discover (predictive) models based on the data maintained in the database [FPSSU96]. Several algorithms have been proposed for computing novel models [AGGR98, AIS93, AMS$^+$96, MAR96, NH94], for more efficient model construction [BMUT97, EKX95, GRG98, GKR98, GRS98, PCY95, RS98, SON95, SAM96, ZRL96], and to deal with new data types [GRG$^+$99, GKR98, GRS99, GGR99]. There is, however, no work addressing the important issue of how to measure the difference, or *deviation*, between two models.

As a motivating example, consider the following application. A sales analyst who is monitoring a dataset (e.g., weekly sales for Walmart) may want to analyze the data thoroughly only if the current snapshot differs significantly from previously analyzed snapshots. In general, since successive database snapshots overlap considerably, they are quite similar to each other [CHNW96, FAAM97, TBAR97]. Therefore, an algorithm that can quantify deviations can save the analyst considerable time and effort.

As a second example, a marketing analyst may want to analyze if and how data characteristics differ across several datasets of customer transactions collected from different stores. The analysis can then be used to decide whether different marketing strategies are needed for each store. Further, based on the deviation between pairs of datasets, a set of stores can be grouped together and earmarked for the same marketing strategy.

In this paper, we develop the `FOCUS` framework for computing an interpretable, qualifiable deviation measure between two datasets to quantify the differences between "interesting" characteristics in each dataset (as reflected in the model it induces when a data mining algorithm is applied on it [FPSSU96]). The central idea is that a broad class of models can be described in terms of a *structural component* and a *measure component*. The structural component identifies "interesting regions," and the measure component summarizes the subset of the data that is mapped to each region. The `FOCUS` framework has several desirable features:

- The deviation measure obtained from `FOCUS` is intuitively interpretable in terms of the work required to transform one model to the other (Section 3). It can be computed using a single scan of the underlying datasets; a good upper bound for frequent itemsets can be computed by simply examining the models (Section 4.1.1).
- The framework allows comparison of specific parts of two models. This makes it possible

to focus attention on interesting changes that might not significantly affect the model as a whole (Section 5).

- The framework covers the models obtained by several mining algorithms, including frequent itemsets, decision trees, and clusters (Sections 4.1 and 4.2). It also captures the *misclassification rate* (commonly used for evaluating decision trees) and *chi-squared statistic* as special cases of the deviation measure. (In Section 5.2.2, we also show how the chi-squared statistic can be applied to decision trees, using the bootstrapping technique to avoid some standard restrictions that would otherwise make it inapplicable.)

We illustrate the power of the framework through these additional contributions:

- We show how `FOCUS` can be used to interactively identify and explore subsets of data that lead to interesting changes in the model being studied (Section 5). We define a set of operators to discover regions where the differences between the datasets are interesting. We also instantiate `FOCUS` to derive a discovery-driven exploratory method for OLAP data proposed by Sarawagi et al. [SAM98] (Section 6).

- We apply our measure of deviation to study whether models based on a sample of the available data differ significantly from the model based on all the data. Interestingly, even for very large sample sizes, there is a statistically significant difference between the sample-based models and the model based on all data. However, the difference diminishes quickly with increasing sample size. In many situations, it may suffice to use a sample, and our measure of deviation can be used to determine the appropriate sample size (Section 7).

2 Examples Illustrating Deviation

In general, a data mining model constructed from a dataset is designed to capture the interesting characteristics in the data. Therefore, we use the difference between data mining models as the measure of deviation between the underlying datasets. In this paper, we consider several classes of data mining models widely studied in the data mining literature: lits-models (short form for frequent itemset models), dt-models (short form for decision tree models), and cluster-models. In this section, we illustrate the concepts and ideas behind the computation of deviation between two datasets first through the class of decision tree models and then through the class of frequent itemsets. In Section 3, we formalize these concepts.

2.1 Classes of Models

Before we discuss the computation of deviation between data mining models, we informally introduce the classes of models (along with applications) for which we instantiate the FOCUS framework. For a formal description, we refer the readers to [AMS$^+$96, BFOS84, DJ80].

dt-models: They arise in several application domains that predict, for a new tuple, the value taken by a specific attribute based on the values taken by a set of predictive attributes; the dependence model is learnt from a database of example tuples. For instance, consider a company that designs and mails catalogs for retail businesses. The dataset which the company accumulated may, besides other information, consist of a set of customer tuples that are classified into two classes: people who responded and people who did not respond to a mail order catalog. Now, consider a new dataset of tuples containing information about the customers of a retailer. To reduce the mailing expenses, the retailer wants the catalog to be mailed only to people who are likely to respond. Therefore, each customer in the new dataset needs to be classified into one of the two groups: responders and non-responders. A dt-model is a predictive model used to predict the group of a new customer based on the classification of the tuples in the dataset. The tuples have several attributes. One designated attribute is called the *dependent attribute*, the other attributes are called *predictor attributes*. The dependent attribute is a categorical attribute while the predictor attributes can either be categorical or numerical.[1] The goal is to build a dt-model that takes as input the values of the predictor attributes and predicts a value for the dependent attribute.

A dt-model is a graphical model in the form of a tree. Each edge originating from an internal node is labeled with a *splitting predicate*. The splitting predicates have the property that any tuple will take a unique path from the root to exactly one leaf node. An example is shown in Figure 1.

lits-models: They arise in the analysis of *market baskets*. A market basket is a collection of items purchased by a customer in an individual transaction, where a *transaction* is a well-defined business activity, for example a customer's visit to a grocery store or an online purchase from a virtual store such as www.amazon.com. Suppose the mail order catalog company (mentioned earlier) has access to very large collections of transactions from past business activity, which it wants to analyze to improve the layout of catalogs. For this purpose, the company may analyze a database of transactions to find sets of items (short, itemsets) that appear together in many transactions. Each pattern extracted through the analysis consists of an itemset and the number of transactions that contain the itemset. Knowledge of the patterns can then be used to improve the layout of mail order catalog pages.

[1] Attributes with totally ordered domains are called *numerical*, whereas attributes with unordered domains are called *categorical*.

Let $\mathcal{I} = \{i_1, \ldots, i_n\}$ be a set of literals called *items*. A *transaction* and an *itemset* are subsets of $\mathcal{I}$. A transaction T is said to *contain* an itemset X if $X \subseteq T$. Let D be a set of transactions. The *support* $sup_D(X)$ of an itemset X in D is the fraction of the total number of transactions in D that contain X. An itemset whose support is greater than a user-specified minimum support threshold is said to *frequent*.

cluster-models**:** A cluster-model is typically used for identifying hidden or unknown groups in the data. Suppose, in the mail order case study described earlier, the retailer does not have a set of example records classified a priori into responsive or non-responsive groups. In such cases, a cluster-model may be used to automatically classify customers into groups. In general, the goal of clustering is to partition the data into several groups, called *clusters*, such that “similar” objects are in the same cluster. Each cluster describes a region in the n-dimensional space where the set of objects in that region are similar to each other. Traditionally, a cluster-model is used to describe a set of n-dimensional points. More recently, several clustering algorithms have been proposed for clustering new data types [GKR98, GRS99, GGR99, GRG$^+$99, GS99]. In this paper, we only consider cluster-models for n-dimensional numerical data.

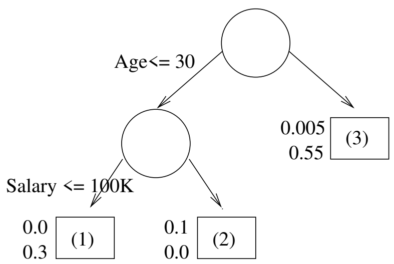


Figure 1: DT

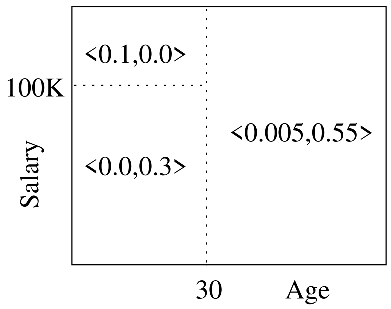
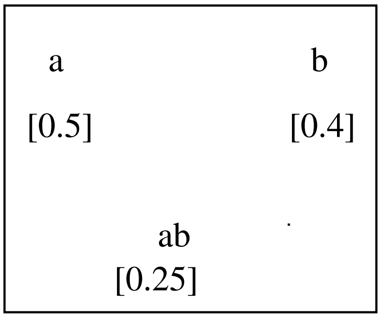


Figure 2: DT:two-components

2.2 dt-models

Let the decision tree constructed from a hypothetical dataset D with two classes—C_1 and C_2—be as shown in Figure 1. The decision tree consists of three leaf nodes. The class distribution at each leaf node is shown beside it (on the left side) with the top (bottom) number denoting the fraction of database tuples that belong to class C_1 (C_2, respectively). For instance, the fractions of database tuples that belong to the classes C_1 and C_2 in the leaf node (1) are 0.0 and 0.3, respectively. Each



L1: [D1]

Figure 3: Lits

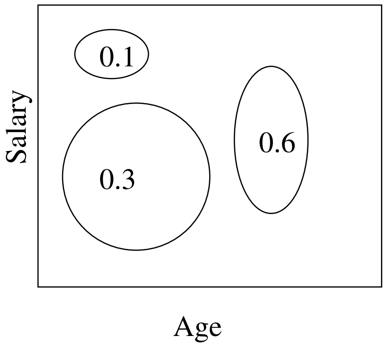


Figure 4: Clusters

leaf node in the decision tree corresponds to two regions (one region for class C_1 and one region for class C_2), and each region is associated with the fraction of tuples in the dataset that map into it; this fraction is called the *measure* of the region. Generalizing from this example, each leaf node of a decision tree for k classes is associated with k regions in the attribute space each of which is associated with its measure. These k regions differ only in the class label attribute. In fact, the set of regions associated with all the leaf nodes partition the attribute space.

We call the set of regions associated with all the leaf nodes in the dt-model the *structural component* of the model. We call the set of measures associated with each region in the structural component the *measure component* of the model. The property that a model consists of structural and measure components is called the *two-component* property. Figure 2 shows the set of regions in the structural component of the decision tree in Figure 1 where the two regions corresponding to a leaf node are collapsed together for clarity in presentation. The two measures of a leaf node are shown as an ordered pair, e.g., the ordered pair $\langle 0.0, 0.3\rangle$ consists of the measures for the two collapsed regions of the leaf node (1) in Figure 1.

We now illustrate the idea behind the computation of deviation between two datasets over a set of regions. Let D_1 and D_2 be two datasets. Given a region and the measures of that region from the two datasets, the *deviation* between D_1 and D_2 with respect to the region is a function (e.g., absolute difference) of the two measures; we call this function the *difference function*. A generalization to the deviation over a set of regions is a "combination" of all their deviations at each region; we represent this combination of deviations by a function called the *aggregate function*, e.g., sum.

If two datasets D_1 and D_2 induce decision tree models with identical structural components, we can combine the two ideas—the two-component property and the deviation with respect to a set of regions—to compute their deviation as follows: the deviation between D_1 and D_2 is the deviation between them with respect to the set of regions in their (identical) structural components.

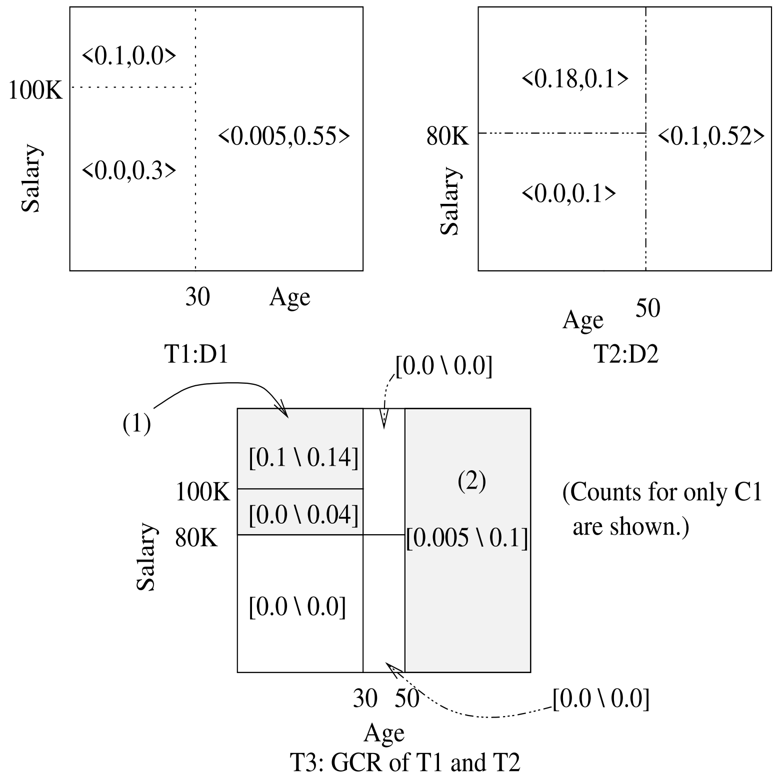


Figure 5: dt-model:$T_3 = \bigwedge(T_1, T_2)$

However, the decision tree models induced by two distinct datasets typically have different structural components, and hence the simple strategy described above for computing deviations may not apply. Therefore, we first make their structural components identical by "extending" them. The extension operation relies on the structural relationships between models, and involves refining the two structural components by splitting regions until the two sets become identical. Intuitively, the refined set of regions is the finer partition obtained by overlaying the two partitions of the attribute space induced by the structural components of both decision trees. We call the refined set of regions the *greatest common refinement* (GCR) of the two structural components. For instance, in Figure 5, T_3 is the GCR of the two trees T_1 induced by D_1 and T_2 induced by D_2. In each region of the GCR T_3, we show a hypothetical set of measures (only for class C_1) from the datasets D_1 and D_2. For instance, the measures for the region $\texttt{salary} \geq 100K$ and $\texttt{age} < 30$ for the class C_1 from D_1 and D_2 are 0.0 and 0.04, respectively. The property that the GCR of two models always exists, which we establish later for decision tree models, is called the *meet-semilattice* property of the class of models.

To summarize, the deviation between two datasets D_1 and D_2 is computed as follows. The structural components of the two dt-models are extended to their GCR. Then, the deviation between D_1 and D_2 is the deviation between them over the set of all regions in the GCR. In Figure 5, if the

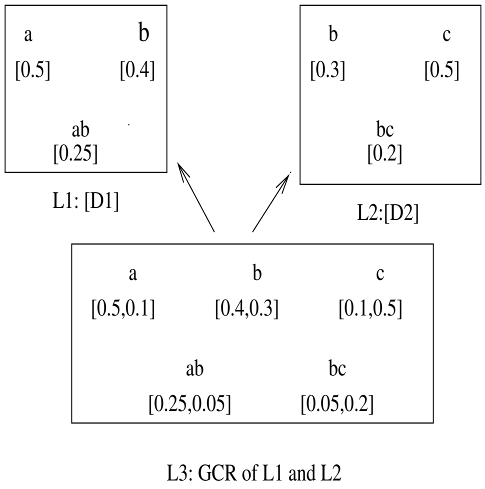


Figure 6: lits-model: $L_3 = \bigwedge(L_1, L_2)$

difference function is the absolute difference and the aggregate function is the sum then (part of) the deviation between D_1 and D_2 over the set of all C_1 regions is given by the sum of deviations at each region in T_3: $|0.0-0.0|+|0.0-0.04|+|0.1-0.14|+|0.0-0.0|+|0.0-0.0|+|0.005-0.1| = 0.175$.

2.3 lits-models

Paralleling the example computation using the class of decision tree models, we now illustrate the deviation computation through the class of frequent itemset models.

Figure 3 shows a simple itemset model where $\mathcal{I} = \{a, b\}$. It has three interesting regions identified by the frequent itemsets $\{a\}$, $\{b\}$, and $\{a, b\}$. Each itemset (equivalently, the corresponding region) is associated with its support: $\{a\}$ with 0.5, $\{b\}$ with 0.4, and $\{a, b\}$ with 0.25. The measure of a region identified by an itemset is the support of the itemset. Generalizing from this example, each frequent itemset X in a lits-model represents a region in the attribute space (where the support is higher than the threshold) whose measure is the support of X. The set of all frequent itemsets is the *structural component* and the set of their supports is the *measure component*.

As in the case of decision trees, if the structural components of two models are identical we compute the deviation between them to be the aggregate of the deviations between the measures at all regions in either structural component. However, if the structural components are different, we first make them identical by extending both models to their *greatest common refinement*. For the lits-models, the GCR is the union of the sets of frequent itemsets of both models. For example, Figure 6 shows the GCR of two lits-models L_1 induced by D_1 and L_2 induced by D_2. Then, the deviation between the datasets is the deviation between them over the set of all regions in the GCR. The measures (or supports) from D_1 and D_2 for each itemset in the GCR are shown below it. If the difference function is the absolute difference, and the aggregate function is the sum then the deviation between D_1 and D_2 is $|0.5-0.1|+|0.4-0.3|+|0.1-0.5|+|0.25-0.05|+|0.05-0.2| = 1.125$.

2.4 Focussed Deviations

In the above examples, we computed the deviation between two datasets over the entire attribute space. In cases where an analyst is interactively exploring two datasets to find regions where they differ considerably, it is necessary to "focus" the deviation computation with respect to a specific region R. The FOCUS framework covers such requirements. The computation is focussed with respect to region R by first intersecting each region in the GCR with R and then combining (using the aggregate function) the deviations over these intersected regions. The intersection with R ensures that the deviation is computed only over regions contained in R. In Figure 5, suppose the analyst is interested only in the difference between T_1 and T_2 over the region R: $\texttt{age} < 30$. The regions in the GCR T_3 intersected with R are the three leftmost regions that satisfy the condition $\texttt{age} < 30$. The deviation between T_1 and T_2 with respect to R is: $|0.0 - 0.0| + |0.0 - 0.04| + |0.1 - 0.14| = 0.08$.

A complementary approach is to declaratively specify a set of "interesting" regions in terms of the structural components of the two models and then rank the interesting regions in the order of their deviations. In Section 5, we introduce a set of structural operators and a ranking operator for declarative specification of interesting regions and region-ranking, respectively.

2.5 Additional Comments

A cluster-model induced by a dataset identifies a set of non-overlapping regions. As discussed above, a dt-model is also associated with a set of non-overlapping regions. The differences between a cluster-model and a dt-model are: (1) the set of regions associated with a dt-model is exhaustive, and (2) if decision trees that allow only univariate splits are considered then these regions are hyper-rectangles in the attribute space. But, since the instantiation of FOCUS for dt-models did not use these two properties unique to a dt-model, the instantiation of FOCUS for dt-models will extend directly to cluster-models. Hence, we do not discuss cluster-models in the rest of the paper.

Note that the derivation of the GCR of two models depends on the class of models being considered. We formalize this dependence in a later section. The computation of the deviation requires the measures from D_1 and D_2 over all the regions in the GCR to be computed; therefore, both the datasets need to be scanned once.

Suppose the deviation between D_1 and D_2 is 0.005, and that between D_1 and D_3 is 0.01. From just the deviation values, we are able to say the data characteristics of D_1 and D_2 are more similar than those of D_1 and D_3. But, we still do not know whether they have "different" data characteristics; a deviation of 0.01 may not be uncommon between two datasets generated by the same process. In other words, is the deviation value statistically "significant"? We answer these questions rigorously using statistical techniques in Section 3.4.

We instantiate the misclassification error metric (from Machine Learning and Statistics) and the chi-squared goodness of fit statistic (from Statistics) from the FOCUS framework. Both metrics only consider the class of dt-models; thus, our FOCUS framework which covers other classes of models as well is more general than the current approaches in Machine Learning and Statistics.

3 FOCUS

In this section, we formally describe the FOCUS framework for computing deviations between the "interesting characteristics" of two datasets. FOCUS can be applied to any class of data mining models that satisfy the two-component and meet-semilattice properties. (Both these concepts are defined below.) In Section 4, we will prove that these properties are satisfied by lits-models, dt-models, and cluster-models.

3.1 Preliminaries

We now introduce our notation, beginning with some standard terms. A *partially ordered* set $\langle P; \preceq\rangle$ consists of a non-empty set P and a reflexive, antisymmetric, transitive binary relation $\preceq$ on P. Let $\langle P; \preceq\rangle$ be a partially ordered set and let $H \subseteq P$. An element $a \in P$ is called a *lower bound* of H if $a \preceq h$ for all $h \in H$. A lower bound a of H is the *greatest lower bound* of H if, for any lower bound b of H, we have $b \preceq a$. We denote the greatest lower bound of H by $\bigwedge H$. A partially ordered set $\langle P; \preceq\rangle$ is a *meet-semilattice* if for all $a, b \in P$, $\bigwedge\{a, b\}$ exists. Let $\mathcal{I} = \{A_1, \ldots, A_n\}$ be a set of attributes. Let $\mathcal{D}_i$ be the *domain* of the attribute A_i, $i \in \{1, \ldots, n\}$.

Definition 3.1 The *attribute space* $\mathcal{A}(\mathcal{I})$ of $\mathcal{I}$ is the cross product of the domains of all attributes: $\mathcal{D}_1 \times \cdots \times \mathcal{D}_n$. A *region* γ is a subset of the attribute space $\mathcal{A}(\mathcal{I})$; $t = \langle t_1, \ldots, t_n\rangle$ is an n-tuple on $\mathcal{I}$ if $t \in \mathcal{A}(\mathcal{I})$. Each region γ has a corresponding predicate P_γ such that $\{P_\gamma(t)=\texttt{true}$ iff $t \in \gamma\}$. A *dataset* D is a finite set of n-tuples.

In contrast to a region, a dataset is an enumerated set of tuples in the attribute space. Let $\mathcal{I} = \{A_1, A_2\}$ with domains $[1, 10], [1, 10]$ respectively. $A_1 \leq 5$ and $D = \{\langle 1, 1\rangle, \langle 2, 1\rangle\}$ are examples of a region (defined by the predicate) and a dataset respectively.

Definition 3.2 The *selectivity* $\sigma(\gamma, D)$ of a region $\gamma \subseteq \mathcal{A}(\mathcal{I})$ with respect to a dataset D is the fraction of tuples in D that map into γ: $\sigma(\gamma, D) \stackrel{\text{def}}{=} \frac{|\{t : t \in D \, \wedge \, P_\gamma(t)\}|}{|D|}$.

3.2 Two-component, Meet-semilattice Models

The main idea behind FOCUS is that a model M has a structural component Γ_M that identifies interesting regions of the attribute space, and that each such region is summarized by a (or several) measure(s), e.g., a count. If the structural component satisfies some properties that allow us to "refine" two models naturally, we have the basis for an intuitive and quantitative deviation measure.

Consider the illustrative examples in Figures 2, 3, and 4 which show a decision tree, a set of frequent itemsets, and a set of clusters, respectively. As mentioned earlier, all three models have one common feature: certain regions in the attribute space are found to be interesting, and are associated with a measure, which is the fraction of tuples in the database that are mapped into the region. We use $\mathcal{M}$ generically to denote any one of the three classes.

Definition 3.3 A class of models $\mathcal{M}$ is said to have the *two component* property if any $M \in \mathcal{M}$ induced by a dataset D can be expressed as $\langle \Gamma_M, \Sigma(\Gamma_M, D) \rangle$ where $\Gamma_M = \{\gamma_M^i : 1 \leq i \leq |\Gamma_M|\}$ is a set of regions in $\mathcal{A}(\mathcal{I})$ and $\Sigma(\Gamma_M, D) = \{\sigma(\gamma_M^i, D) : \gamma_M^i \in \Gamma_M\}$. We use $\Gamma_{\mathcal{M}}$ to denote the set of structural components of all models in $\mathcal{M}$.

We now describe the meet-semilattice property, which captures the structural relationship between models in a class of models $\mathcal{M}$. Figure 5 illustrates the relationship between two decision trees T_1 and T_3. The structure of T_3 is "finer" than that of T_1 because we can deduce T_1's measure component with respect to any dataset D if the measure component of T_3 with respect to D is known. Intuitively, T_3 captures information at a finer level than T_1. Similarly, among the two sets of frequent itemsets L_1 and L_3 shown in Figure 6, L_3 is "finer" than L_1 because we can deduce the measure component of L_1 from that of L_3. We capture this relationship between the structural components of two models in $\mathcal{M}$ using a binary relation called the *refinement* relation.

For the classes of models we consider, given two models M_1 and M_2, the greatest lower bound of their structural components $\Gamma_{M_1}, \Gamma_{M_2}$ under the refinement relation always exists; we call this the *greatest common refinement (GCR)* of Γ_{M_1} and Γ_{M_2}, and denote it by $\Gamma_{\bigwedge(M_1, M_2)}$. The set of all structural components of models in $\mathcal{M}$ along with the refinement relation thus forms a *meet-semilattice*.

Definition 3.4 Let $\Gamma_{M_1}, \Gamma_{M_2} \in \Gamma_{\mathcal{M}}$. We say that a set of regions $\{\gamma_{j_1}, \ldots, \gamma_{j_k}\}$ *refines* a region γ_i if for any dataset D, $\sigma(\gamma_i, D) = \sum_{i=1}^{k} \sigma(\gamma_{j_i}, D)$.[2] We say that Γ_{M_1} *refines* Γ_{M_2} (denoted $\Gamma_{M_1} \preceq \Gamma_{M_2}$) if for every region $\gamma^j_{M_2} \in \Gamma_{M_2}$ there exists a set of regions $\{\gamma^{j_1}_{M_1}, \ldots, \gamma^{j_{k_j}}_{M_1}\} \subseteq \Gamma_{M_1}$ which refine $\gamma^j_{M_2}$. We call $\preceq$ a *refinement relation*.

Lemma 3.1 Let $\mathcal{M}$ be any one of the following three classes of models: lits-models, dt-models, cluster-models. Then $\mathcal{M}$ satisfies the two-component property and there exists a refinement relation $\preceq$ on $\Gamma_{\mathcal{M}}$ such that $\langle \Gamma_{\mathcal{M}}; \preceq \rangle$ is a meet-semilattice.

This observation summarizes results in Sections 4.1 and 4.2.

3.3 Measuring Deviations

We now develop our measure of deviation between two models M_1 and M_2, and thereby, between the underlying two datasets. Intuitively, the difference between the models is quantified as the amount of work required to transform one model into the other, which is small if the two models are "similar" to each other, and high if they are "different."

When the structural components are identical we can transform the measure component of one model to the other by making the measure at each region under the first model agree with that under the second model. Let $\Gamma_{M_1} = \Gamma_{M_2}$. Then, the amount of work for transforming $\Sigma(\Gamma_{M_1}, D_1)$ into $\Sigma(\Gamma_{M_2}, D_2)$ is the aggregate of the differences between $\sigma(\gamma^i_{M_1}, D_1)$ and $\sigma(\gamma^i_{M_2}, D_2)$, $i = 1, \ldots, |\Gamma_{M_1}|$. We assume that the difference, at a region, between the measures of the first and the second models is given by a *difference function* f (not necessarily the usual difference operator "-"), and that the aggregate of the differences is given by an *aggregate function* g. We discuss these functions, which enhance `FOCUS`'s ability to instantiate deviation functions for specialized applications, in Section 3.3.2. For now, it suffices to say that f and g are model-independent parameters of `FOCUS` with the signatures $f : \mathcal{Z}^4_+ \mapsto \mathcal{R}_+$, and $g : \mathcal{P}(\mathcal{R}_+) \mapsto \mathcal{R}_+$.[3]

We now formally define the deviation when the structural components of the two models are identical.

Definition 3.5 Let f be a difference function, g an aggregate function, and $M_1, M_2 \in \mathcal{M}$ be two models induced by the datasets D_1, D_2 respectively, such that $\Gamma_{M_1} = \Gamma_{M_2} = \{\gamma_1, \ldots, \gamma_l\}$. For

[2]The summation over $\sigma(\gamma_{j_i}, D)$ can be replaced by some other function, e.g., average, min, max, product. As we will show later, summation is summation is sufficient for all three common data mining models. Therefore, we stick with this definition for clarity in presentation. In Section 6.2.1, we revisit this note when we need to generalize summation to other functions.

[3]$\mathcal{Z}_+$ and $\mathcal{R}_+$ denote the sets of non-negative integers and non-negative real numbers respectively.

$j \in \{1, 2\}$, let $\kappa^i_{D_j} = \sigma(\gamma_i, D_j) \cdot |D_j|$ denote the absolute number of tuples in D_j that are mapped into $\gamma^i_{M_j} \in \Gamma_{M_j}$. The deviation $\delta^1_{(f,g)}(M_1, M_2)$ between M_1 and M_2 is defined as follows.

$$\delta^1_{(f,g)}(M_1, M_2) \stackrel{\text{def}}{=} g(\{f(\kappa^1_{D_1}, \kappa^1_{D_2}, |D_1|, |D_2|), \ldots, f(\kappa^l_{D_1}, \kappa^l_{D_2}, |D_1|, |D_2|)\})$$

In general, two models induced from different datasets have significantly different structural components. Therefore we first have to reconcile the differences in the structural components of two models to make them comparable. To do this, we rely on the meet-semilattice property exhibited by many classes of data mining models (see Observation 3.1). The idea is to "extend" both models to the GCR of their structural components, and then compare the extensions. Intuitively, to extend a model M to $\Gamma_{M'}$ ($\preceq \Gamma_M$) we find the measure component $\Sigma(\Gamma_{M'}, D)$ for $\Gamma_{M'}$ using the dataset D, i.e., we find the selectivity of each region in $\Gamma_{M'}$ with respect to D.

Definition 3.6 Let $M_1, M_2 \in \mathcal{M}$ be two models induced by D_1, D_2 respectively. We define the deviation $\delta_{(f,g)}(M_1, M_2)$ between M_1 and M_2 as follows.

$$\delta_{(f,g)}(M_1, M_2) \stackrel{\text{def}}{=} \delta^1_{(f,g)}(\langle \Gamma_{\bigwedge(M_1,M_2)}, \Sigma(\Gamma_{\bigwedge(M_1,M_2)}, D_1)\rangle, \langle \Gamma_{\bigwedge(M_1,M_2)}, \Sigma(\Gamma_{\bigwedge(M_1,M_2)}, D_2)\rangle)$$

Usually, we drop f and g because they are clear from the context. The deviation between two models M_1 and M_2 computed using the GCR has some attractive properties. For certain choices of f and g (identified in Sections 4.1 and 4.2), using the GCR gives the least value for δ over all common refinements. This property of the least deviation then corresponds to the least-work transformation between the two models.
Summarizing, the instantiation of `FOCUS` requires:

1. A refinement relation $\preceq$.

2. A difference function f and an aggregate function g.

3.3.1 Computational Requirements for δ

The computation of $\delta(M_1, M_2)$ requires the selectivities of all regions in $\Gamma_{\bigwedge(M_1,M_2)}$ to be computed with respect to both the datasets D_1 and D_2. For the three classes of data mining models we consider, this requires D_1 and D_2 to be scanned once.

3.3.2 Difference and Aggregate Functions

In this section, we motivate the use of parameters f and g in the FOCUS framework. We then present two example instantiations each for f and g.

We first consider f. Let L_1 and L_2 be two lits-models induced by D_1 and D_2. Without loss of generality, let us assume that L_1 and L_2 have identical structural components Γ. (Otherwise, we can extend them to their GCR.) Consider two itemsets X_1 and X_2 in Γ. Suppose $\sigma(\gamma_{L_1}^{X_1}, D_1) = 0.5, \sigma(\gamma_{L_2}^{X_1}, D_2) = 0.55$, and $\sigma(\gamma_{L_1}^{X_2}, D_1) = 0.0, \sigma(\gamma_{L_2}^{X_2}, D_2) = 0.05$. So, X_1 varies between a "significant" 50% and a "more significant" 55% whereas X_2 varies between a "non-existent" 0% and a "noticeable" 5%. For some applications, the variation in X_2 is more significant than the variation in X_1 because noticing an itemset for the first time is more important than a slight increase in an already significant itemset. For some other applications which just concentrate on the absolute changes in support, the variations in X_1 and X_2 are equally important. To allow both cases, our first instantiation f_a finds the absolute difference between the supports, while the second instantiation f_s "scales." We now define the two instantiations.[4]

Definition 3.7 Let $\kappa_1, \kappa_2, N_1, N_2 \in \mathcal{I}_+$ such that $\kappa_1 < N_1$ and $\kappa_2 < N_2$. The *absolute difference function* f_a and the *scaled difference function* f_s are defined as follows:

$$f_a(\kappa_1, \kappa_2, N_1, N_2) \stackrel{\text{def}}{=} |\frac{\kappa_1}{N_1} - \frac{\kappa_2}{N_2}|$$

$$f_s(\kappa_1, \kappa_2, N_1, N_2) \stackrel{\text{def}}{=} \begin{cases} \frac{|\frac{\kappa_1}{N_1} - \frac{\kappa_2}{N_2}|}{(\frac{\kappa_1}{N_1} + \frac{\kappa_2}{N_2})/2}, & \text{if } (\kappa_1 + \kappa_2) > 0 \\ 0, & \text{otherwise} \end{cases}$$

The aggregate function g takes as input a set of values. The two most commonly used aggregate functions are *sum* and *max*. Since the instantiations of f and g are independent of each other, these example instantiations generate four different instantiations of δ.

3.4 The Qualification Procedure

Is the deviation sufficiently large that it is unlikely that the two datasets are generated by the same underlying generating process? The availability of a quantitative deviation measure makes it possible to answer such questions rigorously. If we assume that the distribution $\mathcal{F}$ of deviation values under the hypothesis that the two datasets are generated by the same process is known, we

[4]The signature $f : \mathcal{R}_+ \times \mathcal{R}_+ \mapsto \mathcal{R}_+$ for f where the two arguments correspond to the selectivities of a region with respect to both datasets suffices for most purposes. However, some functions require absolute measures. We give one such example in Section 5.2.2. Therefore, we use absolute measures.

Procedure 3.1 Qualify($D_1, M_1, D_2, M_2, \delta, m, n$)
/* M_1, M_2 are models induced by D_1, D_2. */
/* Output: $\hat{p}_d = P(F > \delta(M_1, M_2))$ */
begin
Draw random samples $S_{11}, S_{12}, \ldots, S_{n1}, S_{n2}$ of size $m\%$ each from D_1.
Compute $D = \{d_1, \ldots, d_n\}$ where $d_i = \delta(M_{i1}, M_{i2})$ and M_{i1}, M_{i2} are induced by S_{i1}, S_{i2}.
Let $D_{(<)} = \{d_{(0)}, d_{(1)}, \ldots, d_{(n+1)}\}$ be the list of values in $D \cup \{0, \infty\}$ sorted in increasing order.
Find i such that $d_{(i)} \leq \delta(M_1, M_2) < d_{(i+1)}$.
$\hat{p}_d = \frac{i}{n+1}$.
end

Figure 7: Bootstrapping procedure for estimating $\mathcal{F}$

can use standard statistical tests to *compute the significance $sig(d)$ of the deviation d* between two datasets. We use *bootstrapping* techniques from Statistics [ET93] to compute $\mathcal{F}$.

Let M_1 and M_2 be two models induced by datasets D_1 and D_2, respectively. Let $\mathcal{G}$ be the hypothetical process that generated D_1. Let us assume that we know the distribution $\mathcal{F}$ of deviations between models induced by a randomly selected pair of datasets generated by $\mathcal{G}$. (We will describe a procedure to compute $\mathcal{F}$.) If D_1 and D_2 have the same data characteristics (equivalently, if D_2 was also generated by $\mathcal{G}$), then $d = \delta(M_1, M_2)$ would be a value drawn from $\mathcal{F}$. Since we know $\mathcal{F}$, we compute the probability $\hat{p}_d$ of two models induced by datasets generated by $\mathcal{G}$ having a deviation greater than or equal to d. ($\hat{p}_d$ equals $P(F > d)$ where F is a random variable with distribution $\mathcal{F}$.) If $\hat{p}_d$ is "significantly" low, then the probability that D_2 was also generated by $\mathcal{G}$ is very low, and we can conclude that D_1 and D_2 have different data characteristics with confidence $100(1-\hat{p}_d)\%$. Typically, most statistical methods consider a value less than 0.05 to be significantly low. $100(1 - \hat{p}_d)\%$ is the *percentage significance* of the deviation between D_1 and D_2. We now present the details of the bootstrapping procedure to generate $\mathcal{F}$ and compute $\hat{p}_d$.

We now discuss the computational requirements of the boostrapped estimation of $\mathcal{F}$. The bootstrapping procedure repeatedly computes (n times, to be precise) the deviation between pairs of samples. We note the following three points regarding the cost-benefit tradeoff. First, it is much less expensive to construct a model on the sample than on the entire dataset. Second, the automatic procedure of computing the significance of deviation between models is many orders of magnitude faster than a domain expert manually inspecting both datasets and the models they induce. Third, the bootstrapped estimate of $\mathcal{F}$ for a dataset can be reused, if necessary.

We note the following comments on the values taken by the parameters n and m in the bootstrapping procedure. A value between 50 and 100 for the number n of bootstrap iterations works well in practice [ET93]. For now, let us assume that there is an oracle which gives us the right value for m. In Section 7, we show that a value between 20% and 30% for m is a good choice.

4 Instantiations

In this section, we instantiate the FOCUS framework for lits-models, dt-models, and cluster-models. Wherever possible, we analyze the properties of the instantiated deviation functions.

4.1 lits-models

We first show that the class of lits-models exhibits the meet-semilattice property. Next, we analyze the deviation functions and discuss interesting characteristics that arise due to the use of the GCR. We then derive an upper bound for the deviation functions $\delta_{(f_a,g)}$ where $g \in \{g_{sum}, g_{max}\}$.

The refinement relation between the structural components of two sets of frequent itemsets is defined by the *superset* relation. Let $\Gamma_{M_1} = L_{D_1}^{m_s}$ and $\Gamma_{M_2} = L_{D_2}^{m_s}$ be two sets of frequent itemsets[5]. Formally, $\Gamma_{M_1} \preceq_L \Gamma_{M_2}$ if $L_{D_1}^{m_s} \supseteq L_{D_2}^{m_s}$. The powerset of a set (here, $\mathcal{I}$) of objects along with the superset relation forms a meet-semilattice [Gra70]. (In fact, it forms a lattice.)

Proposition 4.1 The class of lits-models $\mathcal{M}$ on the set of items $\mathcal{I}$ exhibits the two-component property and $\langle \Gamma_{\mathcal{M}}; \preceq_L \rangle$ is a meet-semilattice.

Once again, consider the example in Figure 6. L_3 is the GCR of L_1 and L_2. The supports from D_1 and D_2 for each itemset in the GCR are shown below it. $\delta_{(f_a,g_{sum})}(L_1, L_2) = 0.4 + 0.1 + 0.4 + 0.2 + 0.15 = 1.125$, and $\delta_{(f_a,g_{max})}(L_1, L_2) = 0.4$.

We now show that using the GCR of two models rather than any common refinement gives the least deviation, which reinforces the interpretation of the notion of work required to transform one model to the other because, as always, we prefer a transformation of lower cost to a higher cost transformation.

Theorem 4.1 Let $f \in \{f_a, f_s\}$ and $g \in \{g_{sum}, g_{max}\}$. Let Γ_M be a common refinement of Γ_{M_1} and Γ_{M_2}. Then,

$$\delta(M_1, M_2) \leq \delta^1_{(f,g)}(\langle \Gamma_M, \Sigma(\Gamma_M, D_1) \rangle, \langle \Gamma_M, \Sigma(\Gamma_M, D_2) \rangle)$$

[5] $L_{D_1}^{m_s}$ is the set of itemsets in D_1 with support greater than m_s.

4.1.1 Upper Bound δ^* for δ

In an exploratory, interactive environment where δ is repeatedly computed, we can typically work with estimates of the actual answers, but require fast responses. For the case where the difference function is f_a, we now derive an upper bound δ^* of δ that can be computed fast using just the two models (which will probably fit in main memory, unlike the datasets). Thus, the entire computation uses in-memory data structures and does not scan either dataset. Using the upper bound δ^* instead of δ is safe; we will not ignore significant deviations. δ^* also satisfies the *triangle inequality*, and can therefore be used to embed a collection of datasets in a k-dimensional space for visually comparing their relative differences.

Definition 4.1 Let $\mathcal{M}$ be the class of lits-models and $M_1, M_2 \in \mathcal{M}$ be two models at minimum support level m_s induced by D_1 and D_2. Let $\kappa_1, \kappa_2 \in \mathcal{I}_+$. Let

$$f^*(\kappa_1, \kappa_2, |D_1|, |D_2|) \stackrel{\text{def}}{=} \begin{cases} f_a(\kappa_1, \kappa_2, |D_1|, |D_2|), \text{ if } \frac{\kappa_1}{|D_1|}, \frac{\kappa_2}{|D_2|} > m_s \\ f_a(\kappa_1, 0, |D_1|, |D_2|), \text{ if } \frac{\kappa_1}{|D_1|} > m_s \text{ and } \frac{\kappa_2}{|D_2|} < m_s \\ f_a(0, \kappa_2, |D_1|, |D_2|), \text{ if } \frac{\kappa_1}{|D_1|} < m_s \text{ and } \frac{\kappa_2}{|D_2|} > m_s \end{cases}$$

We define $\delta^*_{(g)}(M_1, M_2) \stackrel{\text{def}}{=} \delta_{(f^*,g)}(M_1, M_2)$.

Theorem 4.2 Let $M_1, M_2 \in \mathcal{M}$ be two models induced by D_1, D_2 and let $g \in \{g_{sum}, g_{max}\}$. Then the following properties hold:

(1) $\delta^*_{(g)}(M_1, M_2) \geq \delta_{(f_a,g)}(M_1, M_2)$
(2) $\delta^*_{(g)}$ satisfies the triangle inequality.
(3) $\delta^*_{(g)}$ can be computed without scanning D_1 or D_2.

4.2 dt-models

We now show that the class of dt-models exhibits the meet-semilattice property. Next, we discuss certain attractive properties that arise due to the use of the GCR of the two models.

Intuitively, a region γ in one model is refined by a set of regions in another model if they partition γ. The idea is extended to the entire structural component below. We use the predicate representation for regions while formally defining the refinement relation. For the rest of the section, let $M_1, M_2 \in \mathcal{M}$ be two dt-models induced by D_1, D_2 respectively, and let P_γ denote the predicate identifying a region γ.

Definition 4.2 We say that $\Gamma_{M_1} \preceq_T \Gamma_{M_2}$ if, $\forall \gamma^i_{M_2} \in \Gamma_{M_2}$, $\exists \{\gamma^{i_1}_{M_1}, \ldots, \gamma^{i_{j_i}}_{M_1}\} \subseteq \Gamma_{M_1}$: $\{(P_{\gamma^{i_1}_{M_1}} \vee \cdots \vee P_{\gamma^{i_{j_i}}_{M}})$ iff $P_{\gamma^i_{M_2}}\}$.

Intuitively, the GCR of the structural components of two dt-models is the finer partition of $\mathcal{A}(\mathcal{I})$ obtained by overlaying the two structural components Γ_{M_1} and Γ_{M_2}. The corresponding set of predicates is obtained by "anding" all possible pairs of predicates from both the structural components. For example, Figure 5 shows the finer partition formed by overlaying the partitions of the models T_1 and T_2. For the sake of clarity, we show the measures only for regions of class label C_1 in the GCR. (An identical structure exists for the second class label.) Formally, the GCR $\Gamma_{\bigwedge(M_1,M_2)}$ of Γ_{M_1} and Γ_{M_2} is:

$$\{\gamma : \gamma \text{ is identified by } P_{\gamma_1} \wedge P_{\gamma_2} \ \ni \ \gamma_1 \in \Gamma_{M_1} \ \wedge \ \gamma_2 \in \Gamma_{M_2}\}$$

Proposition 4.2 Let $\mathcal{M}$ be the class of dt-models with refinement relation $\preceq_T$. Then $\mathcal{M}$ exhibits the two-component property and $\langle \Gamma_{\mathcal{M}}; \preceq_T \rangle$ is a meet-semilattice.

Once again, we consider the example in Figure 5. T_3's structural component is the GCR of the structural components of T_1 and T_2. For the sake of clarity, only the measures of class C_1 from both D_1 and D_2 are shown in T_3. $\delta_{(f_a,g_{sum})}(T_1, T_2)$ over regions corresponding to class C_1 is: $|0.1 - 0.14| + |0.0 - 0.04| + |0 - 0| + |0 - 0| + |0 - 0| + |0.005 - 0.1| = 0.175$.

The following theorem shows that using the greatest common refinement, rather than any common refinement, gives the least deviation value for the case $g = g_{sum}$.

Theorem 4.3 Let Γ_M be a common refinement of Γ_{M_1} and Γ_{M_2}. Let $g = g_{sum}$, and $f \in \{f_a, f_s\}$. Then, $\delta_{(f,g)}(M_1, M_2) \leq \delta^1_{(f,g)}(\langle \Gamma_M, \Sigma(\Gamma_M, D_1)\rangle, \langle \Gamma_M, \Sigma(\Gamma_M, D_2)\rangle)$

Observe that this theorem is less general than Theorem 4.1 for lits-models. It is not difficult to generate a counter example for $g = g_{max}$ that violates the above theorem. (We discuss one counter example in the appendix.)

5 Focussed Deviations

In this section, we illustrate the power of the FOCUS framework by applying it to two different scenarios: *exploratory analysis* and *change monitoring*. The objective in the first setting is to interactively explore and understand the differences between two datasets, similar to the drill-down and roll-up strategies in OLAP databases [Cod93] and the *ad hoc mining* approach emphasized in [IM96, NLHP98]. The objective in the second setting is to check how well a model built from an old dataset fits a new dataset.

For both application scenarios, a very useful property of FOCUS is that we can compute deviations with respect to a specific region $\gamma \subseteq \mathcal{A}(\mathcal{I})$. Each region in the structural component $\Gamma_M = \{\gamma_M^i, i = 1, \ldots, |\Gamma_M|\}$ of the model M can be independently focussed with respect to γ by taking its intersection with γ. The measure with respect to a dataset D for each region γ_M^i focussed with respect to γ is $\sigma(\gamma \cap \gamma_M^i, D)$.

Definition 5.1 Let $M \in \mathcal{M}$ be a model induced by the dataset D and $\gamma \subset \mathcal{A}(\mathcal{I})$ be a region, called the *focussing region*. Then the *focus M^γ of M* with respect to γ is defined as:

$$M^\gamma \stackrel{\text{def}}{=} \langle \Gamma_M^\gamma, \Sigma(\Gamma_M^\gamma, D) \rangle$$

where $\Gamma_M^\gamma = \{\gamma \cap \gamma_M^i : \gamma_M^i \in \Gamma_M\}$. We use $\mathcal{M}^\gamma$ and $\Gamma_{\mathcal{M}}^\gamma$ to denote the sets of all models in $\mathcal{M}$ and structural components in $\Gamma_{\mathcal{M}}$ focussed with respect to γ.

The following theorem shows that the theory we developed for the class of models $\mathcal{M}$ can be applied to $\mathcal{M}^\gamma$ as well.

Theorem 5.1 Let $\mathcal{M}$ be one of the following three classes of models: lits-models, dt-models, and cluster-models. Let $\preceq$ be a refinement relation such that $\langle \Gamma_{\mathcal{M}}; \preceq \rangle$ forms a meet-semilattice. Let $\gamma \subseteq \mathcal{A}(\mathcal{I})$ be a focussing region. Then $\langle \Gamma_{\mathcal{M}}^\gamma; \preceq \rangle$ is a meet-semilattice.

Definition 5.2 Let f be a difference function, g an aggregate function, and M_1, M_2 be two models induced by D_1, D_2, respectively. The deviation $\delta_{(f,g)}^\gamma(M_1, M_2)$ between M_1 and M_2 focussed with respect to a region $\gamma \subseteq \mathcal{A}(\mathcal{I})$ is defined as:

$$\delta_{(f,g)}^\gamma(M_1, M_2) \stackrel{\text{def}}{=} \delta_{(f,g)}({M_1}^\gamma, {M_2}^\gamma)$$

We emphasize that the deviation function may not be monotonic, i.e., if $\gamma \subset \gamma'$ then the deviation over γ may not be less than that over γ'. For example, if M_1 and M_2 are two models constructed from D_1 and D_2 respectively and $g \in \{g_{sum}, g_{max}\}$ then

$$\gamma \subseteq \gamma' \Rightarrow \delta_{(f_a,g)}^\gamma(M_1, M_2) \leq \delta_{(f_a,g)}^{\gamma'}(M_1, M_2)$$

However, the same is not true for $\delta_{(f_s,g)}(M_1, M_2)$.

The ability to compute region-specific deviations is enhanced by adding operators to manipulate sets of regions. We now introduce a small collection of such operators, divided into two groups: *structural* and *rank* operators. The basic intuition behind the structural operators is that the structural components of models can be viewed as sets of regions. However, the usual set operations like union, intersection, and difference operations do not completely capture the structural relationships between the models. We define the structural operators to obviate this shortcoming. The structural operators take as input two sets of regions Γ_1 and Γ_2 and return as output a set of regions Γ such that the result satisfies the model-specific constraints. Here, we rely on the interpretation of the structural component as a set of regions. For example, if $\mathcal{M}$ is the class of dt-models then each of the sets of regions $\Gamma_1, \Gamma_2, \Gamma$ consists of non-overlapping regions. They may not be exhaustive because we are not dealing with a complete `dt-model` but only one part of such a model.

1. `Structural Union` ($\sqcup$)**:** The structural union $\bigwedge(\Gamma_1, \Gamma_2)$ of two sets of regions Γ_1 and Γ_2 is given by their GCR.

2. `Structural Intersection` ($\sqcap$)**:** The structural intersection $\Gamma_1 \sqcap \Gamma_2$ of Γ_1 and Γ_2 is the set of regions Γ such that each region in Γ is a member of both Γ_1 and Γ_2. This is identical to the standard intersection operation on sets.

3. `Structural Difference` $\ominus$**:** Informally, the structural difference $\Gamma_1 \ominus \Gamma_2$ of Γ_1 and Γ_2 consists of those regions where Γ_1 and Γ_2 differ structurally. Formally,

$$\Gamma_1 \ominus \Gamma_2 \stackrel{\text{def}}{=} (\Gamma_1 \sqcup \Gamma_2) - (\Gamma_1 \sqcap \Gamma_2)$$

4. `Predicate` p**:** The predicate region is a subset of the attribute space identified by p.

Given a set of regions, the rank operator orders them by the "interestingness" of change between the two datasets. The interestingness of a region is captured by a deviation function. Later, we will give a few example instantiations of f and g, which capture a variety of interestingness notions.)

- `Rank`**:** Given a set of regions Γ, two datasets D_1, D_2, and a deviation function $\delta_{(f,g)}$, the `rank` operator $\rho(\Gamma, \delta_{(f,g)}, D_1, D_2)$[6] returns as output a list $\vec{\Gamma}$ of regions in the decreasing order of interestingness.

[6]Since D_1 and D_2 are usually clear from the context, we omit them from the notation.

- `Select:` Given a set of regions ordered according to some criterion, the selection operator selects a subset of the output. For example, `top-region`, `top-n regions`, `bottom-region`, and `bottom-n regions` are common selections; we denote these selections by θ^{top}, θ^{n}, θ^{bot}, and θ^{-n} respectively. We expect the select operator to be typically employed on the output of the rank operator, which orders a set of regions according to an interestingness criterion.

5.1 Exploratory Analysis

The objective in exploratory analysis is to find a set of interesting regions in terms of the differences between the two datasets. Consider the decision trees T_1 and T_2 constructed from D_1 and D_2 shown in Figure 5. Suppose that deviations above 0.05 are considered significant. D_1 and D_2 differ considerably in the shaded regions `(1)` and `(2)`. If $f = f_a$ then these regions have a deviation (with respect to class C_1) of 0.08 and 0.095 respectively. Note that region `(1)` is a leaf node of T_1 but region `(2)` is a sub-region of a leaf node in T_2. Moreover, the sub-regions of `(1)` in T_3 do not cause significant differences between D_1 and D_2. Therefore, we have to find regions that are significantly different at all levels of the tree in addition to the regions of T_3. The following expressions find the regions `(1)` and `(2)` respectively:

$$\theta^{top}(\rho(\Gamma_{T_1} \cup \Gamma_{T_2}, \delta_{(f_a, g_{sum})})), \quad \theta^{top}(\rho(\Gamma_{T_1} \sqcup \Gamma_{T_2}, \delta_{(f_a, g_{sum})}))$$

Next, consider an example in the frequent itemset domain. The shoes and clothes departments in the Walmart super market sell sets of items $\mathcal{I}_1$ and $\mathcal{I}_2$, respectively. Suppose D_1 and D_2 are datasets collected at two different outlets. An analyst compares the top-10 itemsets in each department to see if the popular itemsets are similar across the two departments. Let L_1 and L_2 be the sets of frequent itemsets computed from D_1 and D_2 respectively. Let f and g be chosen appropriately. The following expressions return the top-10 lists from each department, and the combined top-20:

$$\rho(\theta^{10}(\rho(\mathcal{P}(\mathcal{I}_1) \cap (\Gamma_{L_1} \sqcup \Gamma_{L_2})), \delta) \cup \theta^{10}(\rho(\mathcal{P}(\mathcal{I}_2) \cap (\Gamma_{L_1} \sqcup \Gamma_{L_2}))), \delta)$$

$$\theta^{20}(\rho(\mathcal{P}(\mathcal{I}_1) \cup \mathcal{P}(\mathcal{I}_2)) \cap (\Gamma_{L_1} \sqcup \Gamma_{L_2}), \delta)$$

5.2 Monitoring Change

The objective in this setting is to know how well the model constructed from the old dataset fits a new dataset. Therefore, the structural component for the model on the new dataset is expected to be that of the old dataset, and the question can be cast as "By how much does the old model misrepresent the new data?" For decision trees, the misclassification error is widely used for this purpose (e.g.,[BFOS84, LV88, LS97]); as we show, the chi-squared metric can also be adapted (using bootstrapping) to address this question. We show that these two traditional measures can be captured as special cases of the FOCUS framework by appropriate choices of f and g. Thus, FOCUS generalizes change monitoring in two ways: (1) to models other than decision trees, and (2) to change monitoring over specific regions.

5.2.1 Misclassification Error

Let $T = \langle \Gamma_T, \Sigma(\Gamma_T, D_1)\rangle$ be a dt-model constructed on a dataset D_1, and let D_2 be an independent dataset. For each tuple $t \in D_2$, let $C' = T(t)$ be the class label predicted by T for t. If the true class C of t is different from C' then t is said to be *misclassified* by T. The misclassification error $ME^T(D_2)$ of T with respect to D_2 is the fraction of the number of tuples in D_2 misclassified by T.

$$ME^T(D_2) \stackrel{\text{def}}{=} \frac{|\{t \in D_2 \text{ and } T \text{ misclassifies } t\}|}{|D_2|}$$

We define the *predicted dataset* D_2^T of D_2 with respect to T to be the set of tuples formed by replacing the class label of each tuple $t \in D_2$ with T's prediction for t. Denoting the replacement of the class label of a tuple t with c by $t|c$,

$$D_2^T \stackrel{\text{def}}{=} \{t' : t' = t|T(t), t \in D_2\}$$

The following theorem shows that $ME^T(D_2)$ is the deviation between D_2 and D_2^T at Γ_T.

Theorem 5.2 Let T be a dt-model induced by D_1. Let D_2 be another dataset. Then

$$ME^T(D_2) = \frac{1}{2}\delta_{(f_a, g_{sum})}(\langle \Gamma_T, \Sigma(\Gamma_T, D_2)\rangle, \langle \Gamma_T, \Sigma(\Gamma_T, D_2^T)\rangle)$$

5.2.2 Chi-squared Goodness of Fit Statistic

The computation of the chi-squared statistic X^2 assumes that the entire space is partitioned into cells each of which is associated with "expected" and "observed" measures. (See [DS86] for details.) To apply the chi-squared test to dt-models, we use the regions associated with a decision tree T as the cells since these regions partition the entire attribute space. The expected and observed measures are: $E(\gamma_i, D_2) = \sigma(\gamma_i, D_1) \cdot |D_2|, O(\gamma_i, D_2) = \sigma(\gamma_i, D_2) \cdot |D_2|$. The statistic X^2 can now be computed in a straightforward way except for two problems:

(1) For the chi-squared statistic to be well-defined, $E(\gamma_i, D_2)$ should not be zero. We follow the standard practice in Statistics and add a small constant $c > 0$ (0.5 is a common choice) to ensure this [DS86].

(2) At least 80% of the expected counts must be greater than 5 in order to use the standard X^2 tables. In a decision tree, this condition is often violated. For example, if all tuples in node n are of class i, the expected measures for regions $\gamma_j^n, j \neq i$ will be zero. The solution to this problem is to use an exact calculation for the probability distribution of the X^2 statistic under the null hypothesis, i.e., distribution of X^2 values when the new dataset fits the old model [DS86]. The procedure (see Section 3.4) to estimate the exact distribution using the bootstrapping technique can be used to perform the test.

It is easy to show that chi-squared statistic, adapted as described above, can be instantiated from FOCUS.

Proposition 5.1 Let T be the decision tree induced by D_1, and let D_2 be another dataset. Let c be a (small) constant. Then the chi-squared statistic X^2 is given by:

$$X^2 = \delta_{(f,g_{sum})}(\langle T, \Sigma(T, D_1)\rangle, \langle T, \Sigma(T, D_2)\rangle) \text{ where}$$

$$f(v_1, v_2, |D_1|, |D_2|) = \begin{cases} \frac{|D_2|(\frac{v_1}{|D_1|} - \frac{v_2}{|D_2|})^2}{\frac{v_1}{|D_1|}}, \text{ if } v_1 > 0 \\ c, \text{ otherwise} \end{cases}$$

6 Exploratory Analysis of OLAP Data

On-line analytic processing (OLAP) is the interactive exploratory analysis of relational data, and is commonly used in marketing research and analysis. As a typical example, consider an analyst who wants to understand the impact of current marketing strategies and devise new ones. Since the analysis is interactive, decision support systems that support on-line analytic processing require fast processing of complex aggregate queries on large databases. Several methods to expedite the processing of aggregate queries in the OLAP context have been proposed (e.g., [AAD+96b, AAD+96a, ZDN97]). However, Sarawagi et al. [SAM98] argue that, in addition to enhancing the speed of aggregate query processing, it is necessary to automatically aid the analyst in discovering regions which "deviate significantly" from the anticipated behavior. For instance, if the total profit in the city of Madison is higher than anticipated, then the analyst needs to be informed. Sarawagi et al. then propose a method that emulates an analyst's discovery-driven exploration to identify regions with anomalous behavior. They capture the notion of anomalous behavior by constructing a series of predictive statistical models on the data and comparing the actual behavior with the anticipated behavior. In this section, we show that the comparison between the actual and the anticipated behavior can be instantiated from the FOCUS framework. In Section 6.1, we briefly describe the method of discovery-driven data exploration. In Section 6.2, we illustrate its instantiation from the FOCUS framework.

6.1 Discovery-driven Exploration of OLAP Data

OLAP applications analyze relational data whose schema consists of a set of *dimensional* attributes and a set of *dependent* attributes. Some examples for dimensional attributes are product name, store location, and examples for dependent attributes are sales, profit. The analyst is interested in understanding the influence of dimensional attributes on dependent attributes through a series of aggregate queries on subsets of dimensional attributes. Each aggregate query is associated with a set S of *grouping attributes* over which it groups by, and a *measure function* to compute the value for each group. The functions *sum, average, min*, and *max* are some examples of measure functions typically used in the analysis.[7]

Consider a relation R, in the database of a retail grocery chain, with dimensional attributes `city` and `product type`, and a dependent attribute `profit`. Let the *sum* function be the measure function of interest. There are four possible sets of grouping attributes each corresponding to a subset of the set $\{$`city, profit`$\}$. Some example aggregate queries on R are shown below.

Q1: *select* sum(profit)	Q2:*select* city, sum(profit)	Q3:*select* city, product type, sum(profit)
from R;	*from* R	*from* R
	group by city;	*group by* city, product type;

Assuming that a technique to identify tuples with unusual measures in the answer to an aggregate query exists, Sarawagi et al. argue that a typical session of an analyst discovering exceptions proceeds as follows. The analyst starts by looking at the profit aggregated over all of R, then the profits aggregated over each city to isolate a city that behaves unusually. Then, the analyst looks at profits aggregated over each product type for this city. Suppose the profit for the city of Madison aggregated over all product types is exceptional (compared with the overall profit), then the aggregate profits for all product types in Madison will be analyzed to understand the reason for the exception. Since the number of dimensional attributes that in R is two, the *downward* path from the most summarized to the most detailed information stops here. The analyst may also start by isolating a product type (instead of a city) that behaves unusually.

Sarawagi et al. annotate the downward path—most summarized to the most detailed—with "indicators" of exceptions to aid the analyst's search for regions of unusual behavior. Each tuple

[7]The measure function is sometimes referred to as the *aggregate function*. To avoid the confusion between the aggregate function used in deviation computation, we use *measure function*.

in an answer to an aggregate query is associated with an exception value, which is computed as the deviation of the actual measure value from a measure value predicted using a statistical model. For the purpose of instantiating the discovery-driven analysis method from the FOCUS framework, it is sufficient to say that the exception value is a function of the actual measure value y and the predicted value $\hat{y}$. A statistical model predicts $\hat{y}$. The prediction uses all measures on the downward path from the most summarized until y (not including). For instance, the predicted measure for the pair {Madison, coffee} (where coffee is a product type) in the illustrative session discussed above uses the overall profit for R and the total profit from Madison. We skip the details of the statistical model employed. (Interested readers can refer to [SAM98].) Given y and $\hat{y}$, the actual and predicted measure values, of a tuple t the function f for computing the exception value of t is $\frac{(y-\hat{y})^2}{\hat{y}^{\rho}}$ where ρ is a constant (derived from a likelihood model of the data).

6.2 Instantiation of the Discovery-driven Exploration

For the above method of discovering exceptions to be instantiated from the FOCUS framework, we need to show the following two properties. First, the answers to aggregate queries are two-component models and the set of answers to all possible aggregate queries forms a meet-semilattice. Second, the most exceptional tuple in the answer to an aggregate query can be identified. The following two sections discuss each step in detail.

6.2.1 Meet-semilattice induced by Aggregate Queries

In this section, we show that the answer to an aggregate query is a two-component model and the set of answers to all possible aggregate queries on R forms a meet-semilattice. We first illustrate the observation through an example before formalizing it. Consider the relation R mentioned in the previous section. Let the sum function be the measure function. The answer to the aggregate query Q2 consists of a set of tuples—one for each city in R—each of which is associated with a measure, the total profit, for that city. In this example, the set of cities is the structural component and the corresponding set of per-city profit sums is the measure component. Generalizing, the set of tuples in the answer to an aggregate query constitutes the *structural component* and the corresponding set of per-tuple measures constitutes the *measure component*.

Consider the queries Q2 and Q3. It can easily be seen that the answer to Q2 can be computed from the answer to Q3. That is, the profit of every tuple in the answer to Q2 is the sum of the measures (here, profit sums) of a set of tuples in the answer to Q3.[8] Thus, the answer to Q3 is a

[8]Note that if the measure function is different from sum, then the definition of refinement needs to be changed so that the measure of a tuple in the answer to Q2 can be computed from a set of tuples in the answer to Q3. (Recall the

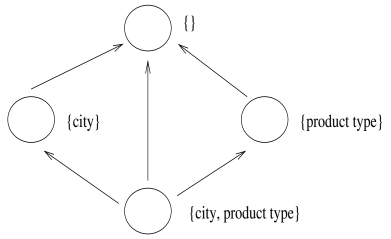


Figure 8: An Example Lattice of Answers to Aggregate Queries

refinement of the answer to `Q2`.

In general, the refinement relation is defined by the *superset* relation on the sets of attributes involved in the group by operator. The answer to a query Q_1 grouping on the set S_1 of attributes is a refinement of an answer to a query Q_2 grouping on the set S_2 of attributes if $S_1 \supseteq S_2$. For instance, the grouping set of attributes $\{\text{city}\}$ of `Q2` is a superset of the corresponding set ϕ for `Q3`. It was already shown in Section 4.1 that the superset relation induces a meet-semilattice. The following theorem formalizes the above discussion.

Theorem 6.1 Let $S = \{A_1, \ldots, A_n\}$ be the set of dimensional attributes, and Y be a dependent attribute of a relation R.[9] Let the measure function be one of *sum, average, min, max*, and $\mathcal{Q}$ be the set of aggregate queries on S. The answer $A(Q)$ to any query $Q \in \mathcal{Q}$ is a two-component model. The set of answers $\mathcal{A}(\mathcal{Q})$ to all queries in $\mathcal{Q}$ forms a meet-semilattice.

As an illustration, the lattice formed by the answers to aggregate queries is shown in Figure 8. An answer to a query is represented by a vertex labelled by the grouping attributes. For instance, the answer to a query grouping on the attribute `city` is represented by a vertex labelled $\{$`city`$\}$. If the answer $A(Q_1)$ to a query Q_1 refines that of another query Q_2 then there is a directed edge from the vertex representing $A(Q_1)$ to the vertex representing $A(Q_2)$.

Finally, note that queries with non-trivial where clauses correspond to models focussed with respect to the where clause predicate. From Section 5, we know that models restricted using predicates are still two-component models, and that they form a meet-semilattice. Therefore, the above instantiation extends in a straight-forward manner to a set of aggregate queries, each of which consists of a specific where-clause predicate.

footnote in Definition 3.4.) However, deriving such functions for the common measure functions (average, min, max) is straight-forward. Therefore, we assume that the measure function is sum.

[9]We only consider one dependent attribute. If R has more than one dependent attribute, we analyze them one at a time.

6.2.2 Identifying Exceptional Regions

In this section, we show that an expression formed from the operators discussed in Section 5 identifies the most exceptional tuple in the answer to an aggregate query. Informally, the expression merely picks the tuple associated with the highest exception value.

Let Q be an aggregate query on a relation R. Let the set of tuples (also, structural component) in the answer to Q be Γ_Q. Let $\Sigma(\Gamma_Q, R)$ and $\Sigma(\Gamma_Q, R)'$ be the measure component and the (statistically) predicted measure component of Q, respectively. Let ρ be a constant. Let f be the difference function for computing the exception value of a tuple; the input to f consists of the actual measure value and the predicted measure value of a tuple. Formally,

$$f(y, \hat{y}) = \frac{(y - \hat{y})^2}{\hat{y}^\rho}$$

Let g be any aggregate function (e.g., sum, max). The functions f and g instantiate a deviation function $\delta_{(f,g)}$. The following expression identifies the most exceptional tuple in Γ_Q whose actual and predicted measure components are $\Sigma(\Gamma_Q, R)$ and $\Sigma(\Gamma_Q, R)'$, respectively.

$$\theta^{top}(\rho(\Gamma_Q, \delta_{(f,g)}))$$

Replacing the θ^{top} operator with the θ^n operator yields the n most exceptional tuples. The analyst can choose one of these n tuples for further exploration.

7 Effect of Sample Size

A popular solution to improve the speed and scalability of data mining algorithms is to induce models from a random sample instead of the entire dataset. The argument here is that a random sample captures most of the characteristics of the underlying dataset. However, this argument is debatable, and there has not been any work on resolving this issue. In this section, we address this issue and quantitatively answer the following question. *While constructing a model using a random sample drawn from the dataset, do bigger sample sizes necessarily yield better models?* We apply `FOCUS` to quantify the notion of "representativeness" of a random sample in inducing the "true" model, that is, the model induced by the entire dataset.

The intuition behind our approach is as follows. The deviation obtained from an instantiation of `FOCUS` quantifies the difference between the models induced by two datasets. If one of the datasets is a sample randomly drawn from the other, the deviation between the models they induce is then a measure of the *representativeness* of the sample in inducing the true model.

Let M be the model induced by D, and M_S the model induced by a random sample S drawn from D. We define the *sample deviation (SD)* of S to be $\delta(M, M_S)$. The smaller the SD of S, the more representative S is of D. This definition gives us a handle to study the influence of the size of the sample on its representativeness.

Using the SD, we now address two questions. Does increasing the size of the sample decrease its SD? If so, is the decrease significant or is it merely an artifact of random fluctuation? If the answer to the first question is affirmative, then the SDs of two sample sizes can be compared to answer the second question; in Sections 7.1.1 and 7.1.2, we carry out this comparison for a wide variety of datasets and models. If the answer to the first question is negative, then the second question is irrelevant. We now describe a procedure that returns the statistical *significance* of the decrease in SD due to an increase in the sample size. The significance is the percentage confidence $100(1-\alpha)\%$ with which the null hypothesis that the two sample sizes are equally representative is rejected.

The basic intuition behind the procedure is as follows. Consider two sets of random samples where the first set S_1 contains samples of size s_{i+1}, and the second set S_2 contains samples of size s_i $(< s_{i+1})$. If the SD measures for size s_{i+1} is smaller than that of s_i $(< s_{i+1})$ then we expect a large number of SD values for S_1 to be smaller than those for S_2. We use the Wilcoxon two-sample test to check the significance of this hypothesis [BD76].

7.1 Empirical Study

In this section, we present an empirical study of the representativeness of a sample versus its size for lits-models and dt-models.

Sample Fraction	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
Significance	99.99	99.99	99.99	99.99	99.99	99.99	99.99	99.99	99.99	-

Table 1: lits-models:% significance of decrease in SD with SF from s_i to s_{i+1}

Sample Fraction	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
Significance	99.99	99.99	99.99	99.97	99.69	79	99.22	99.93	95.25	-

Table 2: dt-models:% significance of decrease in SD with SF from s_i to s_{i+1}

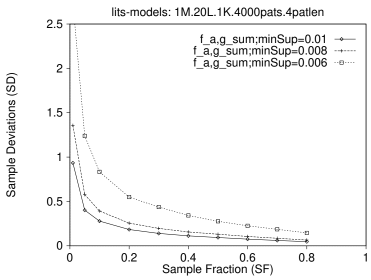


Figure 9: SD vs SF

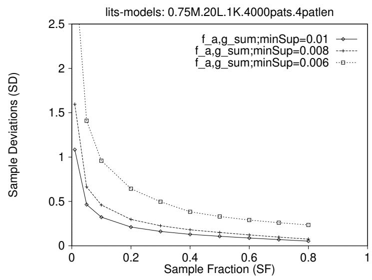


Figure 10: SD vs SF

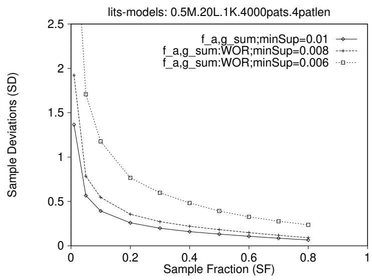


Figure 11: SD vs SF

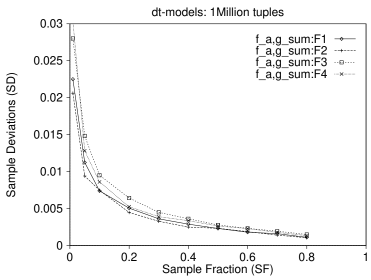


Figure 12: SD vs SF

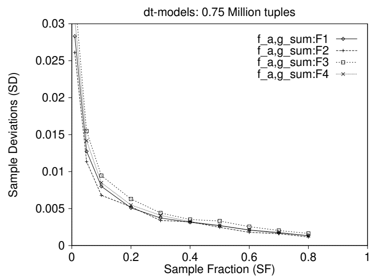


Figure 13: SD vs SF

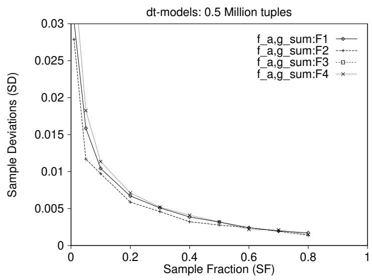


Figure 14: SD vs SF

7.1.1 lits-models

We used the synthetic data generator from the IBM Quest Data Mining group. (The data generator can be downloaded from http://www.almaden.ibm.com/cs/quest/syndata.html.) We use NM.t_lL.$|\mathcal{I}|$I.$N_p$pats.$p$plen to refer to a dataset with N million transactions, average transaction length t_l, $|\mathcal{I}|$ thousand items, N_p thousand patterns, and average pattern length p. We used the Apriori algorithm [AS94] to compute the set of frequent itemsets from a dataset.

We studied all four combinations of f and g. In this section, we only present the results of $\delta_{(f_a, g_{sum})}$. The trends for the remaining experiments are similar. (They are discussed in Section 8.3 in the context of sensitivity of the deviation functions.) We varied two parameters: the size of the dataset and the minimum support level. The datasets used for this study have three different sizes: 1 million, 0.75 million, and 0.5 million transactions. All other parameters to the data generator are set as follows: $|\mathcal{I}| = 1000, t_l = 20, N_p = 4000, p = 4$. Figures 9, 10, and 11 show the sample deviation (SD) versus the sample fraction (SF) values. We draw the following conclusions. (1) As the minimum support level decreases, the size of the sample required to achieve a certain level of representativeness increases. This is to be expected because the lower the minimum support level the more difficult it is to estimate the model. (2) For a given SF value, the representativeness of samples of a fixed size increases with the dataset size. This observation confirms the common perception in Statistics that the absolute size of the sample is more important than its relative size (with respect to the dataset).

Table 1 shows the significance of the decrease in SD as we increase the size of the sample drawn from the dataset `1M.20L.1I.4pats.4plen`. We measured the significance using the Wilcoxon test on sets of 50 sample deviation values for each size. We conclude that the representativeness of samples increases with the size of the sample. However, from Figures 9, 10, and 11 we see that the decrease in the SD values is not large when sample fraction values are larger than 30%.

7.1.2 dt-models

We use the synthetic generator introduced in [AIS93]. It has several classification functions to generate datasets with different characteristics. We selected four functions (Functions F1, F2, F3, and F4) for our performance study. We use $N\texttt{M.F}num$ to denote a dataset with N million tuples generated using classification function num. We used a scalable version of the widely studied CART [BFOS84] algorithm implemented in the RainForest framework [GRG98] to construct decision tree models. We used $\delta_{(f_a, g_{sum})}$ to compute the deviation between two models.

Table 2 shows the significance of the decrease in sample deviations for the dataset `1M.F1` as the sample size is increased. The significance is measured using the Wilcoxon test on sets of 50 sample deviation values for each sample size. The decrease in sample deviation values is quite significant even at SF=70%.

Figures 12, 13, and 14 show the plots for different classification functions (F1, F2, F3, and F4) in the IBM data generator and for varying dataset sizes.

7.1.3 Conclusions from this study

For both classes of models, based on the significance values from the Wilcoxon tests, we conclude that it is better to use larger samples because the decrease in sample deviations is statistically significant even for sample sizes as large as 70-80%. On the other hand, the SD versus SF plots suggest that the rate of additional information obtained decreases with increasing sample size, and for many applications, it may be sufficient to take a sample of size 20-30% of the original dataset.

Dataset	δ	$\% \, sig(\delta)$	δ^*	Time for δ	Time for δ^*
D_1	0.0913	1	0.0913	0	0.01
D_2	3.2198	99	3.6893	46.27	0.01
D_3	6.0957	99	6.60874	46.16	0.01
D_4	6.0096	99	6.4435	44.19	0.01
$D + \delta_5$	0.1511	2	0.1610	17.37	0.0
$D + \delta_6$	0.2760	99	0.3645	19.53	0.01
$D + \delta_7$	0.2784	99	0.3668	18.86	0.0

Figure 15: Deviation with D: 1M.20L.1I.4pats.4plen

8 Experimental Evaluation

In this section, we evaluate the deviation computation in terms of its speed and ability to detect significant deviations. We first consider lits-models and then dt-models. We also study the sensitivity of some deviation functions, and the consequent applications. The datasets we used for this study are also generated from the IBM data generators described in Section 7.1, and the naming conventions are the same as in Section 7.1.

8.1 Set of Frequent Itemsets

In this section, through controlled experiments on synthetic datasets, we first evaluate the procedure for detecting significant deviations. We then evaluate the quality and speed of the upper bound of the deviation function δ^*.

Let D=`1M.20L.1I.4pats.4plen`. We compute deviations between D and a variety of datasets. All datasets $D_1 - D_7$ are generated with an average transaction length 20, and 1000 items; D_1 consists of 500K transactions, $D_2 - D_4$ consist of a million transactions each, and $\delta_5 - \delta_7$ consist of 50K transactions each. The number of patterns and the average pattern length for each dataset is as follows. D_1: (4K,4); D_2, δ_5: (6K,4); D_3, δ_6: (4K,5); D_4, δ_7: (5K,5). In each case, we set the minimum support level to 1% to compute the set of frequent itemsets from both datasets. Figure 15 shows the deviation values and their significance. The deviation value $\delta_{(f_a, g_{sum})}$ and its significance in row `(1)` reflect the fact that D_1 has the same distribution as that of D. As expected, D_2, D_3, D_4 differ significantly from D. Moreover, the deviation values suggest that the parameter `patlen` has a large influence on data characteristics. The addition of δ_5 and δ_6 to D (rows `(6)`, `(7)`) cause significant deviations because they differ in the `patlen` parameter whereas the addition of δ_7 which differs only in the parameter `pats` does not cause a significant

ID	δ	$\%\ sig(\delta)$
$D_{(1)}$	0.0022	10
$D_{(2)}$	1.2068	99
$D_{(3)}$	0.8146	99
$D_{(4)}$	1.4819	99
$D+\delta_{(5)}$	0.0569	99
$D+\delta_{(6)}$	0.03722	99
$D+\delta_{(7)}$	0.0689	99

Figure 16: Deviation with D: 1M.F1

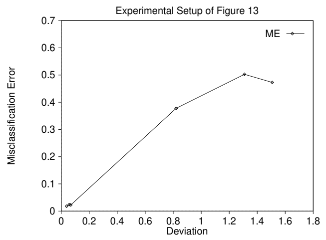


Figure 17: $Deviation$ vs. ME

deviation (row `(5)`).

The last three columns in Figure 15 show that δ^* delivers a good estimate instantaneously. The equality of the times in the row `(1)` is due to the fact that D and D_1 have identical distributions. Therefore, the sets of frequent itemsets were identical; so all the measures necessary to compute the deviation are obtained directly from the models.

8.2 Decision Tree Classifiers

We evaluate the significance detection procedure (see Section 3.4) for dt-models using the same experimental framework as in Section 7.1.2. In this experiment, we compute the deviations using $\delta_{(f_a,g_{sum})}$ and their significance values between D=`1M.F1` and a variety of datasets. The datasets for the first four rows are generated using the functions F1, F2, F3, and F4 respectively. The datasets used for the last three rows are obtained by extending D with a new block of 50000 tuples generated using F_2, F_3, and F_4. D_1=0.5M.F1, D_2=1M.F2, D_3=1M.F3, D_4=1M.F4, D_5=D+δ_5=D + 0.05M.F2, D_6=D+δ_6=D + 0.05M.F3, and D_7=D+δ_7=D + 0.05M.F4.

The significance of the deviation for D_1 in row `(1)` is low because it has the same distribution as that of D. The significance of deviations in rows `(2)`, `(3)`, `(4)` are high, as expected. From rows `(5)`, `(6)`, `(7)`, we see that even the addition of new blocks of size 50K to D causes significant deviations.

In Figure 17, we plot the misclassification error (ME) for the tree constructed from D with respect to a second dataset (chosen from δ_5-δ_7 and $D_2 - D_4$) against the deviation between the two datasets. We see that they exhibit a strong positive correlation.

8.3 Sensitivity of Deviation Functions

In this section, we analyze the deviation functions instantiated by some more combinations of the difference function f and the aggregate function g. We study the sample deviation versus sample fraction plots for each deviation function. The intuition is that the behavior of these plots—smooth or choppy—indicates the sensitivity of the deviation function to the variability in the datasets.

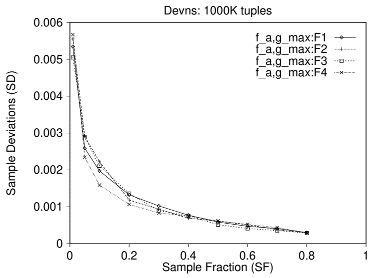


Figure 18: SD vs SF

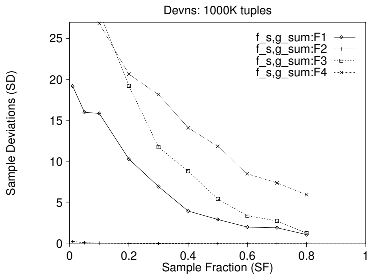


Figure 19: SD vs SF

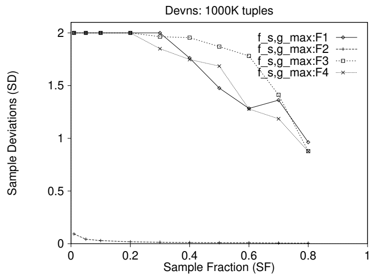


Figure 20: SD vs SF

First, we study the deviation functions for dt-models. Figures 18, 19, and 20 show the plots of sample deviation versus sample fraction on datasets generated using classification functions F1, F2, F3, and F4 for the following deviation functions: $\delta_{(f_a, g_{max})}$, $\delta_{(f_s, g_{sum})}$, and $\delta_{(f_s, g_{max})}$. All plots exhibit a general downward trend as the sample fraction is increased. However, the plots involving the difference function f_s are not as smooth as those for f_a.

Why is the sensitivity of the deviation functions derived from f_s for decision tree models so high? Consider two decision tree models M_1 and M_2 induced by D_1 and D_2 respectively. Let γ be a region with measures v_1 in D_1 and v_2 in D_2. If the structural components Γ_{M_1} and Γ_{M_2} are different, it is possible that $v_1 > 0$ and $v_2 = 0$ in which case $f_s(v_1, v_2, |D_1|, |D_2|) = 2$. (Recall that the motivation behind f_s was to recognize such differences.) This case occurs often, as illustrated by Figures 19 and 20, because the decision tree model induced from the sample almost always has a slightly different structural component. When g_{max} is used, the sensitivity is extremely pronounced. Thus f_s is not a good function to be used when quantifying the representativeness of a sample. In contrast, if the application wants to detect "exceptions" while monitoring changes in a dataset that evolves with time, then f_s is a good function to use.

In Figures 19 and 20, note that the sample deviation values for the classification function F2 are much smaller and smoother than the sample deviation values of the other classification functions. (The plot almost coincides with the x-axis.) The reason for this difference is as follows. The structural component of the classification tree models constructed using datasets from F2 involve only categorical attributes. Due to the small domains of the categorical attributes, the exact structural component of the classification tree model for F2 is recognized even with a very small sample. Datasets from other classification functions induce classification tree models with numerical attributes in their structural component. Due to random fluctuations, the structural components of models induced by a sample differ slightly from the structural component of the model induced by the entire dataset. When $f = f_a$, this difference is very small (see Figure 18) whereas it is pronounced when f_s is used (see Figures 19 and 20). The plots involving f_s illustrate its sensitivity to even minor differences in the structural component between two decision tree models.

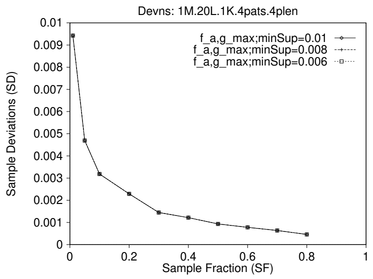


Figure 21: SD vs SF

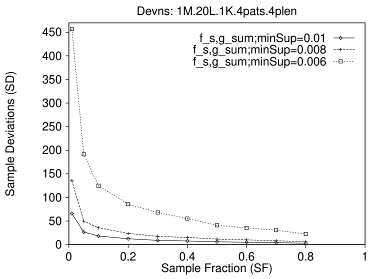


Figure 22: SD vs SF

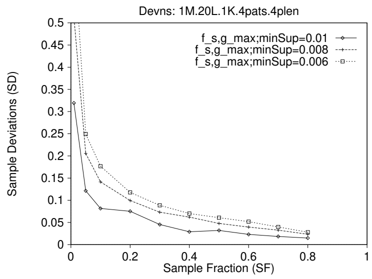


Figure 23: SD vs SF

We now look at the deviation functions for lits-models. Figures 21, 22, and 23 show the plots of sample deviation versus sample fraction on the dataset `1M.20L.1I.4pats.4plen` for the following deviation functions: $\delta_{(f_a, g_{max})}$, $\delta_{(f_s, g_{sum})}$, and $\delta_{(f_s, g_{max})}$. We see that the conclusions drawn from the plot for $f = f_a, g = g_{sum}$ hold even for these deviation functions. Also, the sensitivity is not as pronounced for lits-models because an itemset X appears in the GCR of two models only if X has the required minimum support in either the sample or the entire dataset. And, the measures v_1 and v_2 for such itemsets are fairly close. Thus the case described in the previous paragraph ($v_1 > 0$ and $v_2 = 0$ or cases where v_1 and v_2 differ a lot) is extremely rare. Therefore, the deviation functions are smoother for lits-models.

9 Related and Future Work

A lot of research on clustering concentrated on detecting "outliers" within the dataset as noise and devised special strategies to handle them [EKX95, GRS98, NH94, SD90, ZRL96]. In contrast to the work on clustering, [AAR96, GMV96, KN98] concentrated primarily on discovering outliers in a dataset. They characterized outliers in the data and proposed algorithms for discovering them. None of this work, however, addressed the quantification of differences between datasets.

Interestingness measures to monitor variation in a single pattern were proposed in [ST96].

A similar problem of monitoring the support of an individual itemset was addressed in [AP95, CSD98]. Given a pattern (or itemset) their algorithms propose to track its variation over a temporally ordered set of transactions. However, they do not detect variations at levels higher than that of a single pattern.

In future work, we intend to apply our framework to approximate query answering to evaluate the quality of approximate query answers. The intuition is that each tuple in the result of an aggregate query identifies a region in the attribute space (based on the grouping attribute values), and the aggregate value is the measure of the region.

10 Conclusions

In this paper, we proposed the `FOCUS` framework for quantifying and qualifying changes between datasets. We instantiated the framework for lits-models, dt-models, and cluster-models. We also applied `FOCUS` to two very interesting applications: (1) interactive, exploratory paradigm for finding differences between two datasets, and (2) studying representativeness of a sample.

1. Our framework can instantiate intuitively interpretable deviation measures for several classes of data mining models. The deviation measures can be computed using a single scan of the underlying datasets. The framework allows deviation computation to be focussed to specific parts of the model.

2. We described a procedure to qualify the statistical significance of the deviation measure.

3. We instantiated the framework for the commonly studied classes of data mining models in the database literature: frequent itemsets, decision tree classifiers, and clusters. We also instantiated the misclassification error and the chi-squared goodness of fit statistics.

4. We showed how our framework can support a set of operators to interactively explore the structural components of two models to understand the differences between the datasets inducing the models. We also illustrate the instantiation of the discovery-driven exploration of OLAP data proposed by Sarawagi et al. [SAM98].

5. We applied our framework to study the impact of sample size on its representativeness for both frequent itemsets and decision tree classifiers. Our conclusion is that it is better to use all the data to extract all the information from the data. However, in many cases models constructed from a sample of size between 20-30% of the dataset size are quite close to that constructed from the entire dataset.

References

[AAD+96a] S. Agarwal, R. Agrawal, P.M. Deshpande, A. Gupta, J.F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. In *Proc. of the 22nd Int'l Conference on Very Large Databases*, pages 506–521, Mumbai (Bombay), India, September 1996.

[AAD+96b] Sameet Agarwal, Rakesh Agarwal, Prasad M. Deshpande, Ashish Gupta, Jeffrey F. Naughton, and Raghu Ramakrishnan. On the computation of multidimensional aggregates. *vldb96*, pages 506–522, 1996.

[AAR96] Andreas Arning, Rakesh Agrawal, and Prabhakar Raghavan. A linear method for deviation detection in large databases. In *Proc. of the 2nd Int'l Conference on Knowledge Discovery in Databases and Data Mining*, Portland, Oregon, August 1996.

[AGGR98] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, 1998.

[AIS93] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Database mining: A performance perspective. *IEEE Transactions on Knowledge and Data Engineering*, 5(6):914–925, December 1993.

[AMS+96] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A. Inkeri Verkamo. Fast Discovery of Association Rules. In Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, chapter 12, pages 307–328. AAAI/MIT Press, 1996.

[AP95] Rakesh Agrawal and Giuseppe Psaila. Acive data mining. *Proceedings of the first international conference on knowledge discovery and data mining*, 1995.

[AS94] Rakesh Agrawal and Ramakrishnan Srikant. Fast Algorithms for Mining Association Rules. In *Proc. of the 20th Int'l Conference on Very Large Databases*, Santiago, Chile, September 1994.

[BD76] Peter J. Bickel and Kjell A. Doksum. *Mathematical Statistics: Basic Ideas and Selected Topics*. Prentice Hall, 1976.

[BFOS84] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, 1984.

[BMUT97] Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, and Shalom Tsur. Dynamic itemset counting and implication rules for market basket data. In *Proc. of the ACM SIGMOD Conference on Management of Data*, May 1997.

[CHNW96] D. Cheung, J. Han, V. Ng, and C.Y. Wong. Maintenance of discovered association rules in large databases: An incremental updating techniques. In *Proc. of 1996 Int'l Conference on Data Engineering*, New Orleans, USA, February 1996.

[Cod93] E. F. Codd. Providing OLAP (on-line analytical processing) to user-analysts: An IT mandate. Technical report, E. F. Codd and Associates, 1993.

[CSD98] Soumen Chakrabarti, Sunita Sarawagi, and Byron Dom. Mining surprising patterns using temporal description length. In *Proceedings of the 24th International Conference on Very Large Databases*, pages 606–617, August 1998.

[DJ80] R. Dubes and A.K. Jain. *Clustering methodologies in exploratory data analysis, Advances in Computers*. Academic Press, New York, 1980.

[DS86] Ralph B. D'Agostino and Michael A. Stephens. *Goodness-of-fit techniques*. New York: M.Dekker, 1986.

[EKX95] Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. A database interface for clustering in large spatial databases. In *Proc. of the 1st Int'l Conference on Knowledge Discovery in Databases and Data Mining*, Montreal, Canada, August 1995.

[ET93] Bradley Efron and Robert J. Tibshirani. *An introduction to the bootstrap*. Chapman & Hall, 1993.

[FAAM97] Ronen Feldman, Yonatan Aumann, Amihood Amir, and Heikki Mannila. Efficient algorithms for discovering frequent sets in incremental databases. *Workshop on Research issues on Data Mining and Knowledge Discovery*, 1997.

[FPSSU96] Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1996.

[GGR99] Venkatesh Ganti, Johannes Gehrke, and Raghu Ramakrishnan. Cactus–clustering categorical data using summaries. In *Proceedings of the ACM SIGKDD fifth international conference on knowledge discovery in databases*, August 15-18 1999.

[GKR98] David Gibson, Jon Kleinberg, and Prabhakar Raghavan. Clustering categorical data: An approach based on dynamical systems. In *Proceedings of the 24th International Conference on Very Large Databases*, pages 311–323, New York City, New York, August 24-27 1998.

[GMV96] Isabelle Guyon, Nada Matic, and Vladmir Vapnik. Discovering informative patterns and data cleaning. In Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 181–204. AAAI Press, 1996.

[Gra70] George Gratzer. *Lattice Theory: First Concepts and Distributive Lattices*. W. H. Freemand and Company, 1970.

[GRG98] Johannes Gehrke, Raghu Ramakrishnan, and Venkatesh Ganti. Rainforest – a framework for fast decision tree construction of large datasets. In *Proceedings of the 24th International Conference on Very Large Databases*, pages 416–427. Morgan Kaufmann, August 1998.

[GRG+99] Venkatesh Ganti, Raghu Ramakrishnan, Johannes Gehrke, Allison Powell, and James French. Clustering large datasets in arbitrary metric spaces. In *Proceedings of the 15th IEEE International Conference on Data Engineering*, pages 502–511, Sydney, March 1999.

[GRS98] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Cure: An efficient clustering algorithm for large databases. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, June 1998.

[GRS99] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Rock: A robust clustering algorithm for categorical attributes. In *Proceedings of the IEEE International Conference on Data Engineering*, Sydney, March 1999.

[GS99] Scott Gaffney and Padhraic Smyth. Trajectory clustering. In *Proceedings of the ACM SIGKDD fifth international conference on knowledge discovery in databases*, August 15-19 1999.

[IM96] T. Imielinski and Heikki Mannila. A database perspective on knowledge discovery. *Communication of the ACM*, 39(11):58–64, Nov 1996.

[KN98] Edwin M. Knorr and Raymond T. Ng. Algorithms for distance-based outliers in large databases. In *Proceedings of the 24th International Conference on Very Large Databases*, pages 392–403, August 1998.

[LS97] Wei-Yin Loh and Yu-Shan Shih. Split selection methods for classification trees. *Statistica Sinica*, 7(4), October 1997.

[LV88] W.-Y. Loh and N. Vanichsetakul. Tree-structured classification via generalized disriminant analysis (with discussion). *Journal of the American Statistical Association*, 83:715–728, 1988.

[MAR96] Manish Mehta, Rakesh Agrawal, and Jorma Rissanen. SLIQ: A fast scalable classifier for data mining. In *Proc. of the Fifth Int'l Conference on Extending Database Technology (EDBT)*, Avignon, France, March 1996.

[NH94] Raymond T. Ng and Jiawei Han. Efficient and effective clustering methods for spatial data mining. In *Proc. of the VLDB Conference*, Santiago, Chile, September 1994.

[NLHP98] Raymond T. Ng, Laks V.S. Lakshmanan, Jaiwei Han, and Alex Pang. Exploratory mining and pruning optimizations of constrained association rules. In Laura Hass and Ashutosh Tiwary, editors, *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 13–24, June 1998.

[PCY95] Jong Soo Park, Ming-Syan Chen, and Philip S. Yu. An effective hash based algorithm for mining association rules. In *Proc. of the ACM-SIGMOD Conference on Management of Data*, San Jose, California, May 1995.

[RS98] Rajeev Rastogi and Kyuseok Shim. PUBLIC: A decision tree classifier that integrates building and pruning. In *Proceedings of the 24th International Conference on Very Large Databases*, pages 404–415, New York City, New York, August 24-27 1998.

[SAM96] John Shafer, Rakesh Agrawal, and Manish Mehta. SPRINT: A scalable parallel classifier for data mining. In *Proc. of the 22nd Int'l Conference on Very Large Databases*, Bombay, India, September 1996.

[SAM98] Sunita Sarawagi, Rakesh Agrawal, and Nimrod Megiddo. Discovery-driven exploration of olap data cubes. In *Proceedings of the 6th International Conference on Extending Database Technology*, Valencia, Spain, 1998.

[SD90] J. W. Shavlik and T.G. Dietterich. *Readings in Machine Learning*. Morgan Kaufmann, 1990.

[SON95] A. Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases. In *Proc. of the VLDB Conference*, Zurich, Switzerland, September 1995.

[ST96] Avi Silbershatz and Alexander Tuzhilin. What makes patterns interesting in knowledge discovery systems. *IEEE Transactions on Knowledge and Data Engineering*, 8(6), 1996.

[TBAR97] Shiby Thomas, Sreenath Bodagala, Khaled Alsabti, and Sanjay Ranka. An efficient algorithm for the incremental updation of association rules in large databases. In *Proceedings of 3rd International Conference on Knowledge Discovery in Databases*, 1997.

[ZDN97] Yihong Zhao, Prasad M. Deshpande, and Jeffrey F. Naughton. An array-based algorithm for simultaneous multidimensional aggregates. *sigmod*, 1997.

[ZRL96] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: An efficient data clustering method for very large databases. In *Proc. of the ACM SIGMOD Conference on Management of Data*, Montreal, Canada, June 1996.

A Proofs of Results

In this section, we give proofs of all claims made in the paper. The proofs are ordered by the class of models and thus do not appear in the same order as in the main part of the paper.

A.1 Proofs for lits-models

Proposition 4.1: The class of lits-models $\mathcal{M}$ on the set of items $\mathcal{I}$ exhibits the 2-component property and $\langle \Gamma_{\mathcal{M}}; \preceq_L \rangle$ is a meet-semilattice.
Proof: That the class of lits-models exhibits the 2-component property was illustrated in Section 4.1. The meet-semilattice property follows from the existence of the unique greatest common refinement under the refinement relation $\preceq_L$: $\Gamma_{\bigwedge(M_1,M_2)} = L_{D_1}^{m_s} \cup L_{D_2}^{m_s}$ is unique. □

Lemma 4.1: Let $f \in \{f_a, f_s\}$ and $g \in \{g_{sum}, g_{max}\}$. Let Γ_M be a common refinement of Γ_{M_1} and Γ_{M_2}. Then,

$$\delta(M_1, M_2) \leq \delta^1_{(f,g)}(\langle \Gamma_M, \Sigma(\Gamma_M, D_1)\rangle, \langle \Gamma_M, \Sigma(\Gamma_M, D_2)\rangle)$$

Sketch of proof: Let $\delta'(M_1, M_2) = \delta^1_{(f,g)}(\langle \Gamma_M, \Sigma(\Gamma_M, D_1)\rangle, \langle \Gamma_M, \Sigma(\Gamma_M, D_2)\rangle)$. The result follows from these observations:
(1) $\Gamma_M \supseteq \Gamma_{\bigwedge(M_1,M_2)}$
(2) f is a non-negative function
(3) Let S_1 and S_2 be sets of positive numbers. Since $g \in \{g_{sum}, g_{max}\}$ it follows that $S_1 \subseteq S_2 \Rightarrow g(S_1) \leq g(S_2)$. Let X be an itemset and let $\kappa_1^X = sup_{D_1}(X) \cdot |D_1|$ and $\kappa_2^X = sup_{D_2}(X) \cdot |D_2|$. Then the following equation shows that Lemma 4.1 holds:

$$\begin{aligned}
\delta'_{(f,g)}(M_1, M_2) &= g(\{f(\kappa_1^X, \kappa_2^X, |D_1|, |D_2|) : X \in \Gamma_M\}) \\
&= g(\{f(\kappa_1^X, \kappa_2^X, |D_1|, |D_2|) : X \in \Gamma_{\bigwedge(M_1,M_2)}\})\} \cup \\
&\quad \{f(\kappa_1^X, \kappa_2^X, |D_1|, |D_2|) : X \in \Gamma_M - \Gamma_{\bigwedge(M_1,M_2)}\})\}) \\
&\geq g(\{f(\kappa_1^X, \kappa_2^X, |D_1|, |D_2|) : X \in \Gamma_{\bigwedge(M_1,M_2)}\}) \\
&= \delta_{(f,g)}(M_1, M_2)
\end{aligned}$$

□

Theorem 4.2: Let $M_1, M_2 \in \mathcal{M}$ be two models induced by D_1, D_2 and let $g \in \{g_{sum}, g_{max}\}$. Then the following properties hold:

(1) $\delta^*_{(g)}(M_1, M_2) \geq \delta_{(f_a,g)}(M_1, M_2)$
(2) $\delta^*_{(g)}$ satisfies the triangular inequality.
(3) $\delta^*_{(g)}$ can be computed without scanning D_1 or D_2.

Sketch of proof: The GCR of two sets of frequent itemsets $L_{D_1}^{m_s}$ and $L_{D_2}^{m_s}$ is given by their union. Let $X \in L_{D_1}^{m_s} \cup L_{D_2}^{m_s}$. Let $\kappa_1^X = sup_{D_1}(X) \cdot |D_1|$ and $\kappa_2^X = sup_{D_2}(X) \cdot |D_2|$. Then exactly one of the following three cases holds.

(C1) $X \in L_{D_1}^{m_s} \cap L_{D_2}^{m_s}$: $sup(X)$ is known for D_1 and D_2.
$f^*(\kappa_1^X, \kappa_2^X, |D_1|, |D_2|) = f_a(\kappa_1^X, \kappa_2^X, |D_1|, |D_2|)$

(C2) $X \in L_{D_1}^{m_s} - L_{D_2}^{m_s}$: $sup(X)$ is known for D_1 only.

$f^*(\kappa_1^X, \kappa_2^X, |D_1|, |D_2|) \geq f_a(\kappa_1^X, 0, |D_1|, |D_2|)$

(C3) $X \in L_{D_2}^{m_s} - L_{D_1}^{m_s}$: $sup(X)$ is known for D_2 only.

$f^*(\kappa_1^X, \kappa_2^X, |D_1|, |D_2|) \geq f_a(0, \kappa_2^X, |D_1|, |D_2|)$

From the above three observations and since $g \in \{g_{sum}, g_{max}\}$, (1) holds. Since the computation of δ^* requires only information from the measure components of the two models M_1 and M_2, (3) is true.

We now prove property (2). Let D_3 be another dataset which induces the model M_3. We partition $L_{D_1}^{m_s} \cup L_{D_2}^{m_s} \cup L_{D_3}^{m_s}$ into seven disjoint regions (A) − (G) as shown in Figure 24. We now show that

$$\delta^*_{(g)}(M_1, M_2) + \delta^*_{(g)}(M_2, M_3) \geq \delta^*_{(g)}(M_1, M_3).$$

$$\begin{aligned} \delta^*_{(g)}(M_1, M_3) &= \sum_{X \in A \cup B} sup_{D_1}(X) + \sum_{X \in D} |sup_{D_1}(X) - sup_{D_3}(X)| \\ &\quad + \sum_{X \in E} |sup_{D_1}(X) - sup_{D_3}(X)| + \sum_{X \in F \cup G} sup_{D_3}(X) \end{aligned}$$

$$\begin{aligned} \delta^*_{(g)}(M_1, M_2) + \delta^*(g)(M_2, M_3) = & \textstyle\sum_{X \in A} sup_{D_1}(X) + \sum_{X \in C} sup_{D_2}(X) + \sum_{X \in G} sup_{D_3}(X) + \\ & \textstyle\sum_{X \in B}(|sup_{D_1}(X) - sup_{D_2}(X)| + sup_{D_2}(X)) + \\ & \textstyle\sum_{X \in D}(|sup_{D_1}(X) - sup_{D_3}(X)| + sup_{D_3}(X)) + \\ & \textstyle\sum_{X \in E}(|sup_{D_1}(X) - sup_{D_2}(X)| + |sup_{D_2}(X) - sup_{D_3}(X)|) + \\ & \textstyle\sum_{X \in F}(|sup_{D_2}(X) - sup_{D_3}(X)| + sup_{D_3}(X)) \end{aligned}$$

The result follows from the following inequalities. Let $X \in L_{D_1}^{m_s} \cup L_{D_2}^{m_s} \cup L_{D_3}^{m_s}$ be an itemset. Then,

$$\begin{aligned} X \in B \Rightarrow |sup_{D_1}(X) - sup_{D_2}(X)| + sup_{D_2}(X) &\geq sup_{D_1}(X) \\ X \in D \Rightarrow |sup_{D_1}(X) - sup_{D_3}(X)| + sup_{D_3}(X) &\geq |sup_{D_1}(X) - sup_{D_3}(X)| \\ X \in E \Rightarrow |sup_{D_1}(X) - sup_{D_2}(X)| + |sup_{D_2}(X) - sup_{D_3}(X)| &\geq |sup_{D_1}(X) - sup_{D_3}(X)| \\ X \in F \Rightarrow |sup_{D_2}(X) - sup_{D_3}(X)| + sup_{D_3}(X) &\geq sup_{D_2}(X) \end{aligned}$$

□

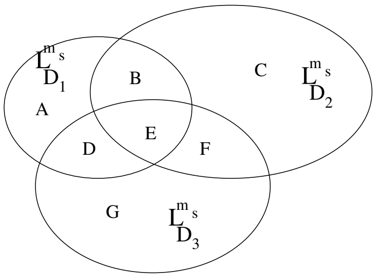


Figure 24: Three sets of frequent itemsets: $L^{m_s}_{D_1}, L^{m_s}_{D_2}, L^{m_s}_{D_3}$

A.2 Proofs for dt-models

Proposition 4.2: Let $\mathcal{M}$ be the class of dt-models with refinement relation $\preceq_T$. Then $\mathcal{M}$ exhibits the 2-component property and $\langle \Gamma_{\mathcal{M}}; \preceq_T \rangle$ is a meet-semilattice.

Proof: The 2-component property has already been discussed in Section 4.2. The meet-semilattice property follows from the uniqueness of the GCR, which we prove by showing that if

$$(\Gamma_{M'} \preceq_T \Gamma_{M_1}) \wedge (\Gamma_{M'} \preceq_T \Gamma_{M_2}) \Rightarrow (strucM' \preceq_T \Gamma_{\bigwedge(M_1,M_2)}$$

Let $M_1, M_2 \in \mathcal{M}$ be induced by two datasets D_1, D_2. Let $\Gamma_M = \Gamma_{\bigwedge(M_1,M_2)}$ be defined as follows:

$$\Gamma_{\bigwedge(M_1,M_2)} \stackrel{\text{def}}{=} \{\gamma : \gamma \text{ is identified by } P_{\gamma_1} \wedge P_{\gamma_2} \ \ni \ \gamma_1 \in \Gamma_{M_1} \ \wedge \ \gamma_2 \in \Gamma_{M_2}\}$$

Let $\Gamma_{M'}$ also refine Γ_{M_1} and Γ_{M_2}. Let $\gamma \in \Gamma_{M'}$. Then, there exist $\gamma_1 \in \Gamma_{M_1}$ and $\gamma_2 \in \Gamma_{M_2}$ such that γ is a sub-region of both γ_1 and γ_2. Therefore, γ is a sub-region of $\gamma_{12} \in \Gamma_M$ identified by the predicate $P_{\gamma_1} \wedge P_{\gamma_2}$. Similarly, every region in $\Gamma_{M'}$ is a sub-region of some region in Γ_M. Since $\Gamma_{M'}$ is a partition of the attribute space every region in Γ_M is refined by a set of regions in $\Gamma_{M'}$. Therefore, $\Gamma_{M'} \preceq_T \Gamma_M$. $\square$

Theorem 4.3: Let Γ_M be a common refinement of Γ_{M_1} and Γ_{M_2}. Let $g = g_{sum}$, and $f \in \{f_a, f_s\}$. Then,

$$\delta_{(f,g_{sum})}(M_1, M_2) \leq \delta^1_{(f,g)}(\langle \Gamma_M, \Sigma(\Gamma_M, D_1)\rangle, \langle \Gamma_M, \Sigma(\Gamma_M, D_2)\rangle)$$

Proof: Let $\gamma \in \Gamma_{\bigwedge(M_1,M_2)}$ be a region. Let x and y be the number of tuples in D_1 and D_2 respectively that are mapped into γ. Since Γ_M refines $\Gamma_{\bigwedge(M_1,M_2)}$, there exists a set of regions $\gamma_1, \ldots, \gamma_k \in \Gamma_M$ such that they partition γ. Then, x is split into $x_1, \ldots, x_k$ at $\gamma_1, \ldots, \gamma_k$ respectively such that $(x_1 + \ldots + x_k) = x$. Similarly, y is split into $y_1, \ldots, y_k$ such that $y = y_1 + \ldots + y_k$. Therefore, the computation of δ' further splits the term $f(x, y, |D_1|, |D_2|)$ into $\sum_{i=1}^{k} f(x_i, y_i, |D_1|, |D_2|)$. The computation of δ sums $f(x, y, |D_1|, |D_2|)$ at each region γ in $\Gamma_{\bigwedge(M_1,M_2)}$. We consider the relationship between δ and δ' at an arbitrary region $\gamma \in \Gamma_{\bigwedge(M_1,M_2)}$, which extends to all of $\Gamma_{\bigwedge(M_1,M_2)}$.

`Case 1:` $f = f_a$.

$$\begin{aligned} f_a(x, y, |D_1|, |D_2|) &= |\frac{x}{|D_1|} - \frac{y}{|D_2|}| \\ &= \left|\sum_{i=1}^{k}\left(\frac{x_i}{|D_1|} - \frac{y_i}{|D_2|}\right)\right| \\ &\leq \sum_{i=1}^{k}\left|\frac{x_i}{|D_1|} - \frac{y_i}{|D_2|}\right| \\ &= \sum_{i=1}^{k} f_a(x_i, y_i, |D_1|, |D_2|) \end{aligned}$$

`Case 2:` $f = f_s$.

$$\begin{aligned} f_s(x, y, |D_1|, |D_2|) &= \frac{|\frac{x}{|D_1|} - \frac{y}{|D_2|}|}{\frac{1}{2}(\frac{x}{|D_1|} + \frac{y}{|D_2|})} \\ &\leq \frac{\sum_{i=1}^{k}|\frac{x_i}{|D_1|} - \frac{y_i}{|D_2|}|}{\frac{1}{2}(\frac{x}{|D_1|} + \frac{y}{|D_2|})} \\ &\leq \sum_{i=1}^{k}\frac{|\frac{x_i}{|D_1|} - \frac{y_i}{|D_2|}|}{\frac{1}{2}(\frac{x_i}{|D_1|} + \frac{y_i}{|D_2|})} \\ &= \sum_{i=1}^{k} f_s(x_i, y_i, |D_1|, |D_2|) \end{aligned}$$

Since γ is arbitrarily chosen, the relationship extends to the entire structural component of $\Gamma_{\bigwedge(M_1, M_2)}$ in a straight-forward way. □

Counter Example for $g = g_{max}$:
Using the above notation, let $x = 0.5$, $y = 0.0$ be the measures from D_1 and D_2 respectively at a region γ. Let $x_1 = x_2 = 0.25$ and $y_1 = y_2 = 0.0$ be the measures of the partition of γ into $\{\gamma_1, \gamma_2\}$. If we use g_{max}, clearly the partition may lead to a lower value for δ. Therefore, the above lemma is not valid for g_{max}.

Theorem 5.2: Let T be a dt-model induced by D_1. Let D_2 be another dataset. Then $ME^T(D_2)$=

$$\frac{1}{2}\delta_{(f_a, g_{sum})}(\langle\Gamma_T, \Sigma(\Gamma_T, D_2)\rangle, \langle\Gamma_T, \Sigma(\Gamma_T, D_2^T)\rangle)$$

Proof: Let n be a leaf node in T associated with class label n_C, predicate P_n, and regions $\gamma_i^n = P_n \wedge (C = i)$, $1 \leq i \leq k$. Let $x_i, 1 \leq i \leq k$ be the number of tuples in D_2 that satisfy the predicate P_n and belong to the class `i`. Let $x = x_1 + \cdots x_k$. Without loss of generality assume that $n_C = 1$. The misclassification error of T at n is $\frac{x - x_1}{|D_2|}$. Since T predicts that all tuples in D_2 satisfying P_n are in class `1`, the number of tuples in D_2^T at regions $\gamma_1^n, \ldots, \gamma_k^n$ are $(x_1 + \ldots + x_k), 0, \ldots, 0$ respectively.

Aggregating the differences between the measures in D_2 and in D_2^T over regions $\gamma_1^n, \ldots, \gamma_k^n$ and noting that $|D_2| = |D_2^T|$, we get: $f_a(x_1, x_1 + \ldots + x_k, |D_2|, |D_2|) + \sum_{i=2}^{k} f_a(x_i, 0, |D_2|, |D_2|)$
$= \frac{x - x_1}{|D_2|} + \sum_{i=2}^{k} \frac{x_i}{|D_2|} = \frac{2(x-x_1)}{|D_2|}$
Summation over all leaf nodes results in $2ME^T(D_2)$. □

Proposition 5.1:
Let T be the decision tree induced by D_1, and let D_2 be another dataset. Then the chi-squared statistic X^2 is given by:

$$X^2 = \delta_{(f, g_{sum})}(\langle T, \Sigma(T, D_1)\rangle, \langle T, \Sigma(T, D_2)\rangle) \text{ where}$$
$$f(v_1, v_2, |D_1|, |D_2|) = \begin{cases} \frac{|D_2|(\frac{v_1}{|D_1|} - \frac{v_2}{|D_2|})^2}{\frac{v_1}{|D_1|}}, \text{ if } v_1 > 0 \\ c, \text{ otherwise} \end{cases}$$

Sketch of proof: The result follows from the definitions of f and the chi-squared statistic. □

A.3 Focussed Deviations

Theorem 5.1: Let $\mathcal{M}$ be one of the following three classes of models: lits-models, dt-models, cluster-models. Let $\preceq$ be a refinement relation such that $\langle \Gamma_{\mathcal{M}}; \preceq \rangle$ forms a meet-semilattice. Let $\gamma \subseteq \mathcal{A}(\mathcal{I})$ be the focussing region. Then $\langle \Gamma_{\mathcal{M}}^{\gamma}; \preceq \rangle$ is a meet-semilattice.
Sketch of proof: Let $M_1, M_2 \in \mathcal{M}$. The result follows from the following two observations.

First, if $\Gamma_{M_1} \preceq \Gamma_{M_2}$ then $\Gamma_{M_1}^{\gamma} \preceq \Gamma_{M_2}^{\gamma}$. Because, $\{\gamma_{M_1}^{j_1}, \ldots, \gamma_{M_1}^{j_k}\}$ refines $\gamma_{M_2}^{i}$ then $\{\gamma \cap \gamma_{M_1}^{j_1}, \ldots, \gamma \cap \gamma_{M_1}^{j_k}\}$ refines $\gamma \cap \gamma_{M_2}^{i}$.

Second, focussing (with respect to γ) the GCR of two structural components is identical to the GCR of the focussed structural components, i.e., $\Gamma^{\gamma}_{\bigwedge(M_1, M_2)} = \bigwedge(\Gamma_{M_1}^{\gamma}, \Gamma_{M_2}^{\gamma})$. We prove this observation for lits-models and dt-models separately.

lits-models:

$$\begin{aligned} \Gamma^{\gamma}_{\bigwedge(M_1, M_2)} &= \gamma \cap \Big\{\{\gamma_{11}, \ldots, \gamma_{1|\Gamma_{M_1}}\} \cup \{\gamma_{21}, \ldots, \gamma_{2|\Gamma_{M_2}}\}\Big\} \\ &= \Big\{\{\gamma \cap \gamma_{11}, \ldots, \gamma \cap \gamma_{1|\Gamma_{M_1}}\} \cup \\ &\qquad \{\gamma \cap \gamma_{21}, \ldots, \gamma \cap \gamma_{2|\Gamma_{M_2}}\}\Big\} \\ &= \bigwedge(\Gamma_{M_1}^{\gamma}, \Gamma_{M_2}^{\gamma}) \end{aligned}$$

dt-models: We use the predicate notation for regions in a dt-model.

$$\begin{aligned} \Gamma^{\gamma}_{\bigwedge(M_1, M_2)} &= P_{\gamma} \wedge \{P_{\gamma_1} \wedge P_{\gamma_2} \ni \gamma_1 \in \Gamma_{M_1} \wedge \gamma_2 \in \Gamma_{M_2}\} \\ &= \{(P_{\gamma} \wedge P_{\gamma_1}) \wedge (P_{\gamma} \wedge P_{\gamma_2}) \ni \gamma_1 \in \Gamma_{M_1} \wedge \gamma_2 \in \Gamma_{M_2}\} \\ &= \bigwedge(\Gamma_{M_1}^{\gamma}, \Gamma_{M_2}^{\gamma}) \end{aligned}$$

Therefore, $\langle \Gamma_{\mathcal{M}}^{\gamma}; \preceq \rangle$ forms a meet-semilattice. □