# DimScanner: A Relation-based Visual Exploration Approach Towards Data Dimension Inspection

Jing Xia[*]
State Key Lab of CAD&CG, Zhejiang University

Wei Chen[†]
State Key Lab of CAD&CG, Zhejiang University

Yumeng Hou[‡]
State Key Lab of CAD&CG, Zhejiang University

Wanqi Hu[§]
State Key Lab of CAD&CG, Zhejiang University

Xinxin Huang[¶]
State Key Lab of CAD&CG, Zhejiang University

David S. Ebert[‖]
Purdue University

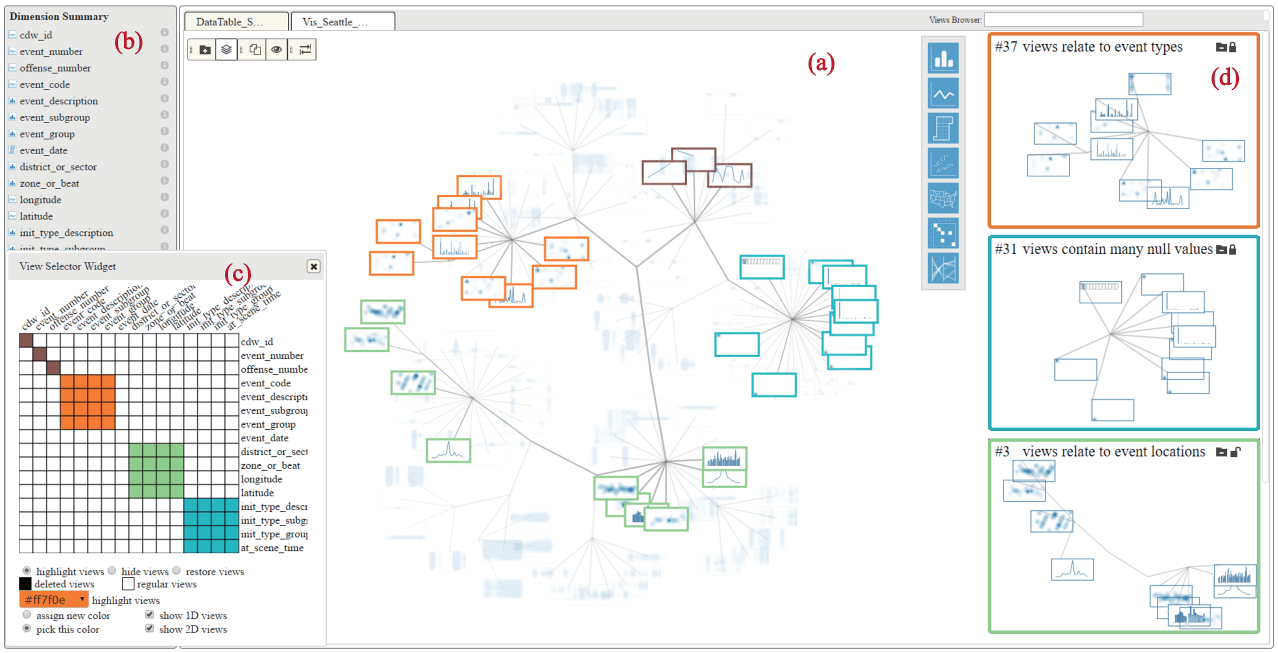


Figure 1: The DimScanner interface on the Seattle 911 calls and responses dataset whose time span ranges from October 28, 2013 to February 14, 2014. (a) The categorization tree is structured with 16 1D views and 120 2D views contributed by 16 dimensions of the data. The tree structure indicates the clustering hierarchy of pair-wise view relations such that views on the same branch are more likely to be related than those far apart. (b) The dimension summary column summarizes the data types (by icons) of the dimensions. It indicates that the dataset contains eight categorical dimensions, six numeric dimensions and two time-related dimensions. (c) The view selector widget displays a matrix of all views. It highlights three groups of views selected by analysts with colors. (d) The view summary snapshots keep track of analysts' observations (three highlighted groups are shown in this case).

## Abstract

Exploring multi-dimensional datasets can be cumbersome if data analysts have little knowledge about the data. Various dimension relation inspection tools and dimension exploration tools have been proposed for efficient data examining and understanding. However, the needed workload varies largely with respect to data complexity and user expertise, which can only be reduced with rich background knowledge over the data. In this paper we address the workload challenge with a data structuring and exploration scheme that affords dimension relation detection and that serves as the background knowledge for further investigation. We contribute a novel data structuring scheme that leverages an information-theoretic view structuring algorithm to uncover information-aware relations among different data views, and thereby discloses redundancy and other relation patterns among dimensions. The integrated system, DimScanner, empowers analysts with rich user controls and assistance widgets to interactively detect the relations of multi-dimensional data.

**Keywords:** High-dimensional data visualization, information-aware relation, data exploration.

**Index Terms:** Human-centered computing [Visualization]: Visualization application domains—Visual vnalytics;

[*]e-mail: jjane.summer@gmail.com
[†]e-mail: chenwei@cad.zju.edu.cn, corresponding author
[‡]e-mail: junesnow26@gmail.com
[§]e-mail: huwanqi1234@gmail.com
[¶]e-mail: huangxinxin07@gmail.com
[‖]e-mail: ebertd@ecn.purdue.edu

2016 IEEE Conference on Visual Analytics Science and Technology (VAST)
23-28 October, Baltimore, Maryland, USA
978-1-5090-5661-3/16/$31.00 ©2016 IEEE

## 1 Introduction

Although multi-dimensional data can have great value, it can be useless or even dangerous (e.g., considering misleading conclusions generated from data) without proper processing and interpretation. In particular, multi-dimensional data exploration is often used to find relations and frequent patterns among dimensions and requires need for semantic exploration for effectiveness. Since data analysts are exploring and often do not know the exact information they are looking for when they start the exploration, tools to help them understand the relationships among the multiple dimensions are necessary [4]. The following are typical challenges for multi-dimensional data exploration:

- Data analysts can easily become confused when navigating a large multi-dimensional data space. They explore the dataset by randomly navigating from one view to another, expecting to find useful patterns. They want to make the best use of visualization, but an overview of all the data views without proper organization is overwhelming.
- Redundant data can create unnecessary analysis workload. Even though the data are cleaned in terms of data format, redundant dimensions that encode useful but duplicate information may exist. If analysts can notice this similarity or the relationships across the dimensions, then cognitive complexity can be reduced.

Multiple approaches have been adopted for multi-dimensional data exploration. Script-based data processing tools such as Numpy[1] and database scripts are well adopted. However, efficiency is restricted by the users' programming skills and inexpressive data feedback. R[2] is a statistical analysis tool that supports not only data processing but also data distribution visualization. More complex visualization layouts such as parallel coordinates and scatter plot matrix can be employed to more effectively depict the data distributions. This multi-view approach can generate a set of data views for flexible investigation. However, even with these approaches, analysts still spend much time randomly exploring the many dimensions or data views in pursuit of valuable information.

These challenges can be addressed with an informative data context that depicts the basic distributions of data dimensions and the data dimension structure. The approach requires an efficient information-aware organization and exploration scheme whose primary task is to provide an information-rich understanding of the data and to provide clues of relation patterns. We achieve this by backing up the visualizations with not only data processing but also data organization. Data organization is especially effective in revealing basic structures between data dimensions and guiding the analysts' exploration process to a smaller subspace of the large multi-dimensional space.

In this paper we contribute the design and implementation of such **a data structuring and exploration scheme** that improves the conventional multi-dimensional data exploration process with a novel information-theoretic view matching and structuring technique. We first extract data views and characterize the view-wise relations with mutual information. We then organize extracted views in a configurable "categorization tree" [27] structure by means of a quartet analysis technique, to present the information-based relations among data views to provide context and structure information. In addition, the constructed categorization tree results in a hierarchical relation structure of data views, where views with close relations are near each other and views with remote relations are far apart.

**DimScanner** is an implementation of this multi-dimensional data exploration process that integrates the relation matching technique and the relation structuring technique. It accommodates a suite of interactions and assistance widgets to empower analysts with comprehensive preview and rich controls over data views. In summary, the contributions of this paper are as follows:

- A view matching and structuring scheme that constructs view-wise relation structures of multi-dimensional data;
- An exploration scheme that supports interactive investigation and view-wise relation validation of multi-dimensional data.

[1]http://www.numpy.org/

[2]https://www.r-project.org/

## 2 Related Work

Our discussion of relevant previous work contains two aspects: inspecting dimensional relations and exploring multi-dimensional data.

### 2.1 Dimension Relationship Inspection

Bertini et al. described the goal of multi-dimensional data analysis as "the extraction of relevant and meaningful information" [4]. To achieve this goal, a variety of techniques (quality metrics) have been proposed to measure the relations among multiple dimensions. Many solutions [8, 31, 32] have been proposed to measure the quality in the image space, i.e., the projection of the underlying data onto 2D planes. For instance, graph-theoretic scagnostics [32] constructs convex hull, alpha hull, and minimal spanning tree with the projected plots and defines the quality of data with shape-based metrics, such as outlying, skewed, and clumpy. Alternatively, Pixnostics [25] measures quality not only in the image space with a jigsaw map and pixel bar chart but also in the data space with correlation metrics and classification metrics. Once the quality metrics are obtained, clustering can be applied to reduce visual clutter [3], select features [26], and find relevant subspaces [11].

In addition to various quality metrics, a large body of visualization techniques have been applied for interactive exploration of dimensional relations. Parallel coordinates is a typical visualization for multi-dimensional data and has been adopted in many quality metrics studies [8, 17, 29]. Another scheme is the use of multiple views for faceted inspection of dimensional relations, which can be either automatically generated [7, 15] or dynamically created [16, 35]. A commonly adopted form is the matrix view, in which each element encodes a visualization of a facet of the underlying data. For instance, GPLOM [15] visualizes dimension-wise relations with a generalized plot matrix. Recently developed dimension projection matrix/tree [35] represents the exploration flow of subspace projection with a tree structure, effectively favoring dimension inspection of multi-level subspaces. TimeSeer [7] implements the scagnostics measurements and characterizes relations between time and other dimensions. Explates [16] is designed to dynamically create customized visualization of user-selected dimensions. Linkages between data dimensions and visualization methods are explicitly displayed as chains to indicate view-wise relations.

In this paper, we construct a categorization tree structure of the data views based on quartet analysis and employ an information-aware relation for characterizing correlations. The information-aware relation differs from the above quality metrics by integrating non-linear correlation. The categorization tree generated through quartet analysis is superior to dimension projection methods which suffer heavily from dimension evaluation and unreliable clustering problems.

### 2.2 Multi-dimensional Data Exploration

Exploring multi-dimensional data relies heavily on user interactions, such as semantic zooming, sorting, and highlighting. Specialized techniques and toolkits have been well-studied for multivariate tabular data. For instance, TableLens [24] can effectively support the exploration of large multivariate tables with a semantic zooming scheme. SimulSort [14] and ParallelTable [13] employ sorting and highlighting techniques, respectively, to enhance the exploration efficiency. Our scheme differs from the previous methods in that ours presents the data with a tree-based structure whose nodes encode 1D or 2D views of the input data.

Commercialized data exploration tools are available, such as the business intelligence software Tableau[3] and the visual statistical software JMP[4]. The research community has also paid considerable attention to the visualization of multi-dimensional datasets. Data views can be configured with different layouts [23, 30, 34]. For instance, Improvise [30] is a coordinated-view environment that supports visualization of a multivariate dataset with orthogonal view layout. MosaicJS [23] creates visual representations from multivariate dataset with a mosaic view layout model. Voyager [34] supports faceted browsing of multivariate data.

## 3 Approach Overview

The main goal of DimScanner is to provide an informative relational structure of the data views, and thereby study the relations and frequent patterns in the data space. Data analysts typically explore the data space by identifying various relational patterns between views [1]. One important relation pattern is the one-to-one relationship that creates redundancy between two views: if each value in one view corresponds to exactly one value in the other, and vice versa (hence a one-to-one correspondence), then one view can be considered redundant. For instance, the view "city" is redundant with the presence of the view "city (abbreviation)". Other representative important relations in the data space are one-to-many correspondences and frequent value pairs. The former indicates a one-to-many correspondence between two views, e.g., the view "category" and the view "subcategory". The latter indicates a strong co-occurrence between two values in two views despite the co-occurrence of other values. An example is the strong correlation between "18-years old" (in the "age" view) and "student" (in the "occupation" view).

DimScanner is designed to achieve its goal by leveraging two components: a structuring component that generates the view relations and an exploration component that supports view navigation. The structuring process starts by enumerating all 1D and 2D views of the input dataset. We employ both 1D and 2D views because, in some situations, 2D views can provide combinational perspectives to the data space. For example, a view that combines "latitude" and "longitude" is often more reasonable than two 1D views. The pair-wise relations [see the view matching step in Figure 2 (a)] among views are computed by means of an information-theoretic metric. A collection of quartets for all views are then generated based on the relations of view pairs [see the quartets construction step in Figure 2 (b)]. Subsequently, a view categorization tree is constructed [see the structuring step in Figure 2 (c)] such that views on the same branch are more likely to be related.

The visualization and interaction design [see the exploration component in Figure 3] favors fast validation of relations among views, and progressive visual exploration. DimScanner presents the categorization with two types of tree-based layouts: a dendrogram or a force-directed layout. Analysts can select a single view to observe distribution or two views to analyze their relation (via relation constructing). Distinctive views and associated interactions are designed for depicting time-oriented, geographical, numerical, and categorical dimensions. For instance, analysts can aggregate the data by a customized time level such as hour, month, or year (via time aggregation), within a time-related view. In addition, analysts can convert a geo-related 2D view to a map with a drag-and-drop interaction. Appropriate visualizations for numerical and categorical dimensions are also supported (e.g., histograms and line charts). The widely applied interaction brushing and linking is supported in each view as well. To escape from the overwhelming number of views, analysts can select, group, or filter views with the view selector. Analysts can also summarize groups of views of interest with the help of a summary snapshot list.

[3]http://www.tableau.com/

[4]http://www.jmp.com

## 4 Structuring Component

We now describe two key techniques of the structuring component: view matching and view structuring. The view matching process seeks to extract highly related view pairs, while the view structuring process generates a relation-based organization within the view pairs.

### 4.1 Information-aware view matching

Our system design needs a measure of the relationship between pairs of views. In accordance with the relation patterns described in Section 3, we need to define the relation appropriately so that two views are highly related if data clusters in one view are also clustered in the other. Formally, this relation measures the discrepancy among the distributions of individual views and the joint distributions of view pairs. Mutual information (MI) [22] meets this requirement by measuring the mutual dependence among variables. It can be used to characterize the relations among views. We favor MI over the common Pearson's correlation coefficient because MI evaluates the statistical correlation that may present a non-linear correlation.

We apply the definition of MI [22] to two views ($A$ and $B$) as $I(A;B)$:

$$I(A;B) = \sum_{a \in D(A)} \sum_{b \in D(B)} p(a,b) log(\frac{p(a,b)}{p(a)p(b)}). \qquad (1)$$

$D(A)$ and $D(B)$ denote the sets of values in view A and view B, respectively. $p(a,b)$ is the joint probability distribution of values in view $A$ and view $B$. $p(a)$ and $p(b)$ are the marginal probability distributions of view $A$ and view $B$, respectively. In practice, we generate the probability from the input data instead of a normal distribution. If the values of a dimension are non-categorical, then this dimension is appropriately binned. For a 2D view $C$, $D(C)$ denotes the set of values from both dimensions.

MI can be expressed as $I(A;B) = H(A) - H(A|B)$, where $H(A)$ is the marginal entropy and $H(A|B)$ is the conditional entropy. When values in views $A$ and $B$ are distributed independently, $H(A|B)$ is identical to $H(A)$, and therefore MI $I(A;B)$ is zero. Otherwise, $H(A)$ is greater than the conditional entropy $H(A|B)$. Typically, a large MI indicates a large co-occurrence of clusters in two views.

### 4.2 Data view structuring

For view structuring, we formulate the collection of relations of pairs of data views as a relation matrix. This relation matrix contains the pairwise relations subject to the chosen information-aware relation measures described above, and can be regarded as an abstraction of the original data. Consequently, structuring the relations among views needs to reduce the complex information to achieve a cognitively acceptable illustration (A similar situation arises in hierarchical clustering [9], which extracts a hierarchy of clusters from a distance matrix). The most natural and efficient way to achieve this is to represent the relations in the form of a dendrogram, which is a directed binary tree or undirected ternary tree [6]. Inspired by prior work on phylogenetics [28] that achieves hierarchical clustering by topologically arranging four-item tuples, we choose to construct a categorization tree structure with view quartets.

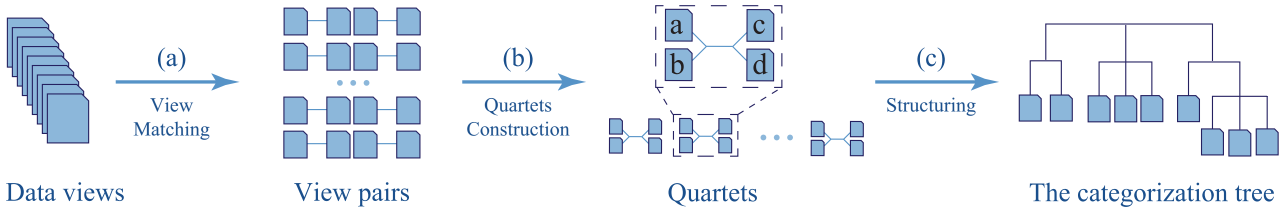


Figure 2: The structuring component takes three steps: (a) the views are paired using an information measure; (b) a collection of quartets are generated for all views based on this information-based relations of view pairs; (c) finally, a view categorization tree is constructed.

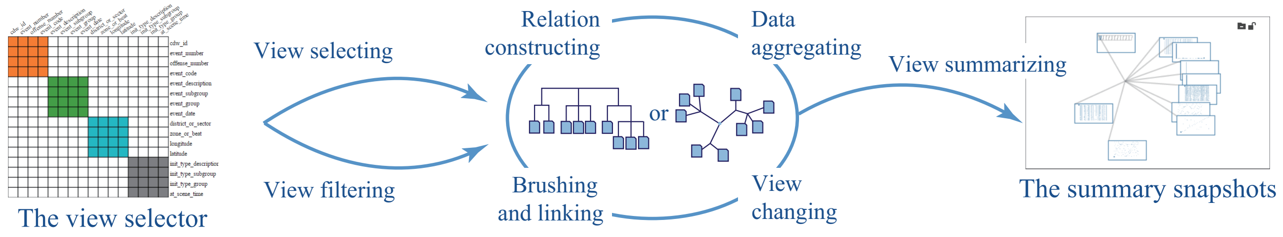


Figure 3: The exploration component depicts the user controls that support interactive data investigation. Direct view controls include relation constructing, brushing and linking, data aggregating and view changing. The view selector helps select and filter the views while the summary snapshots helps summarize the groups.

Quartet analysis is a topology-based phylogenetic reconstruction method in biology [33]. A quartet in this scope is a four-species tuple in two sets where the species in the same sets are genetically related and the species in different sets are not, denoted as $AB|CD$. A quartet puzzling procedure [28] is conducted to generate a categorization tree based on all the quartets.

With a data view regarded as a high-dimensional point, embedding all data views in a 2D space is feasible using various dimension projection approaches, including MDS [5], PCA [18], and self-organizing maps [19]. However, our approach does not employ dimension projection methods for two reasons. First, the dimension of the embedding is a key parameter for manifold projection methods [21]. Without a proper computation of the intrinsic dimension of the underlying dataset, 2D projection approaches may easily lead to mixed features in the 2D plane. The subsequent clustering of the projected 2D points is not very reliable. Second, a dimension projection process requires an evaluation of the dimensional dissimilarity subject to specific distance functions [10], which typically match some conditions such as triangle inequality [20]. Instead, MI measures the discrepancy of the data distributions in the two views (dimensions) and is thus independent to other views (dimensions) by its definition.

However, the quartet-based analysis is a qualitative method. Quartets can be generated with heterogeneous dissimilarities or even without quantitative measures as long as the four-tuple structure is qualified [12]. In addition, the quartet-based analysis is essentially a maximum-likelihood method that aims to construct a tree structure that preserves as many qualified quartets as possible [28], thereby naturally producing a clustering representation of the input data points. In this sense, the quartet-based analysis is superior to dimension projection methods that suffer heavily from dimension evaluation and unreliable clustering problems.

**Construction of the views:** The data view structuring process seeks to construct a categorization tree structure with quartets. On the basis of quartet analysis, a view quartet is a four-view tuple $(ab|cd)$ (see a quartet example in Figure 2) in which the views to the same side ($AB$ and $CD$) are related, whereas the views to the different side ($AC$, $AD$, $BC$, and $BD$) are unrelated. We generate the quartets based on the information-aware relations described in Section 4.1 for all view pairs. According to the definition, a quartet is qualified when the two most related view pairs are disjoint and their relations are much more significant than the other four relations.

Algorithm 1 explains the qualification process of a quartet with four views $A$, $B$, $C$, and $D$. We first obtain the first and second maximum relation values from the relation list $(I(A;B)$, $I(A;C)$, $I(A;D)$, $I(B;C)$, $I(B;D)$, $I(C;D))$. Assuming that the quartet is qualified, the sum of the indexes of the two values should be 5 in order to generate two disjoint view pairs. The qualification does not hold if a more significant relation exists in the other four relations. Alternatively, we can also sort six relations and check whether the first two view pairs are disjoint. Our algorithm is slightly faster than a conventional sorting scheme (of average time complexity $6log6$) because it traverses the relation list at most twice (lines 1 and 5). For $N$ views, $N*(N-1)*(N-2)*(N-3)/24$ quartets exist to determine their qualification. Therefore, this difference is much more significant for the $O(N^4)$ quartets.

Once all quartets are obtained, we construct a tree structure that contains as many quartets as possible. We adopt the Quartet MaxCut (QMC) algorithm [27], which recursively takes two steps to generate the tree. QMC first parses the quartets and generates a graph, whose nodes represent the views and whose edges encode the relations between view pairs. An edge is regarded as good if the associated views are related, and vice versa. The weight of each edge is also derived from quartets, as $NUM(good\ edges) - NUM(bad\ edges)$. Good edges indicate two related views and vice versa. The algorithm then partitions the graph into two subgraphs with the standard graph-cut algorithm to maintain as many good edges as possible. Once the subgraphs are obtained, the weights of edges are updated based on the remaining quartets, and the partitioning step is recalled. The recursion stops when the graph is fully subdivided or when no more bad edges exist.

As discussed in [12], a reliable categorization tree can be constructed by $O(m^2)$ quartets ($m$ being the number of views). Algorithm 1 generates qualified quartets for tree construction by adjusting the weight $k$ in line 6. A small $k$ results in fewer qualified quartets.

**Algorithm 1** Determine if there exits a qualified quartet with four views.

```
Input: Four views A, B, C, D and their correspondence relations
    $relations = [I(A;B), I(A;C), I(A;D), I(B;C), I(B;D), I(C;D)]$,
    a threshold weight $k$.
Output: A qualified quartet if one exists.
 1: Get the max value of $relations$ and its index respectively, as
    $MaxRelation$ and $MaxIndex$.
 2: $SecondMaxIndex = 5 - MaxIndex$
 3: $SecondMaxRelation = relations[SecondMaxIndex]$
 4: $qualified = TRUE$
 5: for Each relation other than the largest and the second largest do
 6:     if There exists a relation larger than $k * SecondMaxRelation$ then
 7:         $qualified = FALSE$
 8:         Break
 9:     end if
10: end for
11: return The corresponding quartet if $qualified = TRUE$ still holds
```

## 5 Exploration Component

The exploration component visualizes layouts of the structured categorization tree, and enables data analysts to interactively compare and explore data views and their relations. It also affords rich assistance widgets for flexible investigation and summarization.

### 5.1 Data view layout

Theoretically, the categorization tree can be represented as a dendrogram (Figure 10), which presents a clear tree structure with a limited number of data views, or a flexible force-directed layout [Figure 1 (a)] exists, which can compactly encode a tree with many nodes. Analysts can switch to an appropriate layout based on the given data configuration. In either case, the leaf nodes encode all data views. The thickness of the branches is defined by its depth such that the thickness increases with greater proximity to the root. To avoid an overwhelming information display, we use a view thumbnail in place of a leaf node, whose shape indicates the overall distribution of the view.

The view thumbnails can be triggered to complete charts to show details, such as the data values and the value counts. We apply different visualizations to data views based on dimension data types. Examples include bar charts for categorical views, line charts for numerical views, calendars for time-related views, and scatter plots for 2D views. We change the thumbnail of a line chart to a stacked graph [see view (a) and its thumbnail in Figure 10] because lines are difficult to recognize in a small display area. When we hover over a view thumbnail, it is enlarged, and a path from the node to the root is highlighted in green. When we hover over a non-leaf node, all its leaf thumbnails and the paths are highlighted.

### 5.2 Direct view controls

Rich controls over the views are supported.

**Relation construction** - Although the layout is based on information-aware relations among views, data analysts are not convinced unless they explicitly see how two views on the same branch are related. When analysts select two views, a parallel coordinate view is shown for dimensional inspection and comparison [Figure 4 (c)]. The data count is encoded with opacity, thereby easing the inspection of data clusters and relation patterns.

Figure 4 shows the comparison between view "district_or_sector" [Figure 4 (a)] and view "zone_or_beat" [Figure 4 (b)]. One value in "district_or_sector" corresponds to three values in "zone_or_beat" (highlighted in red). To be specific, the correspondence is a one-to-three pattern as seen in the parallel coordinate plot [Figure 4 (c)]. For example, district N covers beat N1, N2 and N3. It indicates that "zone_or_beat" is a subdivision of "district_or_sector". Noticing the similarity, data analysts can hide either view to reduce the workload.

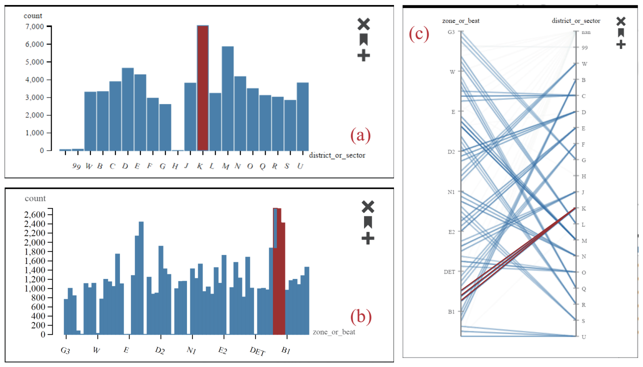


Figure 4: A comparison between view "district_or_sector" and view "zone_or_beat" (some labels of "zone_or_beat" are not shown), which indicates that "zone_or_beat" is a subdivision of "district_or_sector".

**Data aggregation** - In a time-related view, analysts can interactively aggregate records by various time levels such as year, month, day, hour, and minute. The aggregation results in a simplified view that may reveal a periodical pattern, which is useful for hourly/daily/weekly/monthly pattern identification.

**Brushing and linking** - Brushing-and-linking is implemented to highlight values in other views. If analysts highlight a subset of data in one view, then corresponding data in other views are highlighted. Analysts can compare highlighted parts with non-highlighted parts for combinational distribution exploration.

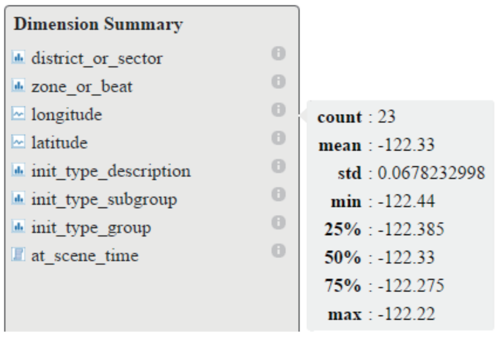
**View changing** - A histogram, a line chart or a calendar is applied for a categorical dimension, a numeric dimension and a time dimension respectively. A scatter plot is adopted for 2D data views. Specifying a different visual form is also allowed by dragging the thumbnail that represents a visual form onto the underlying view. Available alternatives include a map for geospatial views, a parallel coordinate view and a matrix view.

With the structured layouts and user interactions, data analysts are able to discover important data views and highly related views. For time-related views, analysts can explore the patterns with different aggregation levels. They can also select a reasonable visualization for a specific view.

### 5.3 Assistance widgets

Aside from the direct controls, several assistance widgets are provided for flexible data exploration.

**Dimension summary widget -** The dimension summary widget (Figure 5) lists all dimensions of the dataset, their data types, and basic statistical values. The data types are indicated by small thumbnails. When we hover over a dimension, the statistical information of the dimension is automatically shown, i.e., quartile values for numerical dimensions, time duration for time-related dimensions, and counts of identical values for categorical dimensions.



Dimension Summary

- district_or_sector
- zone_or_beat
- longitude
- latitude
- init_type_description
- init_type_subgroup
- init_type_group
- at_scene_time

| | |
|---|---|
| **count** | 23 |
| **mean** | -122.33 |
| **std** | 0.0678232998 |
| **min** | -122.44 |
| **25%** | -122.385 |
| **50%** | -122.33 |
| **75%** | -122.275 |
| **max** | -122.22 |

Figure 5: The dimension summary column summarizes the data types (by icons) and the basic statistical attributes (the tooltip) of dimension "longitude".

**View selector widget -** The view selector widget allows analysts to select or deselect one or multiple views with pre-defined colors (Figure 6) by clicking the entries of the matrix. All entries have three configurations indicated by different colors: white for regular views, black for hidden views, and other colors for highlighted views.

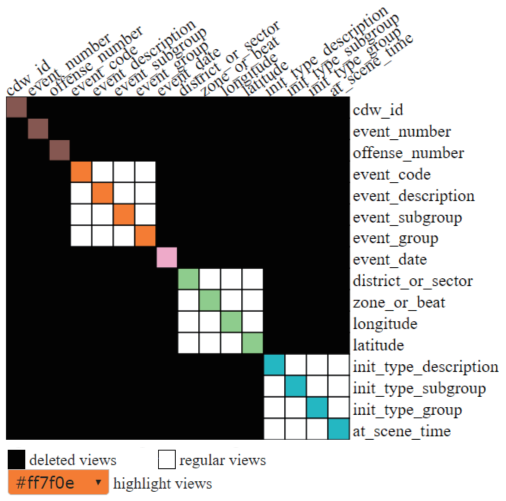


Figure 6: The view selector widget displays a matrix of all views. All entries have three configurations indicated by different colors: white for regular views, black for hidden views and other colors for highlighted views.

Data analysts may select a 1D view by clicking the diagonal entries or select a 2D view by clicking non-diagonal entries. All views related to one dimension can be selected by clicking the rectangle entries in the right of the matrix. Analysts can also choose to highlight views with a specific color. Analysts may choose to assign a new color automatically whenever they click an entry or to pick a color manually from the color scheme list. Manual color selection is particularly useful when analysts want to assign a single color to multiple entries. When analysts specify the colors to be black, views in the categorization tree will be filtered. The view selector also supports immediate filtering of 1D and 2D views with each individual check box.

**Summary snapshots -** The summary snapshots (Figure 7) capture local tree structures of the categorization tree. The organization of a summarized subset is copied from the global tree, thereby making it easy to recognize. When multiple snapshots are created, only one snapshot, either the latest created one or the one unlocked by analysts, is activated.

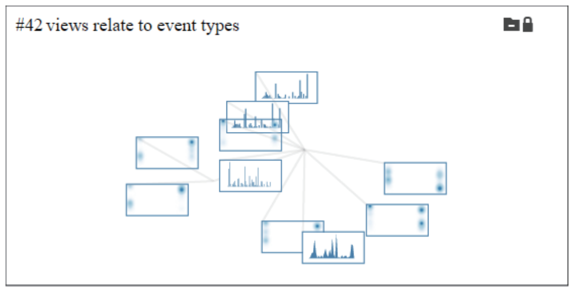


Figure 7: The three view summary snapshots keep track of analysts' observation and extract two of the highlighted groups individually.

### 5.4 Implementation

The DimScanner system is based on a browser/server architecture, where the server is responsible for the structuring component and the browser is responsible for the navigating component. We utilized Django[5], a lightweight Python web framework, as the server, and D3.js and jQuery UI for front-end implementation. The python data analysis tool Pandas[6] is adopted for back-end data computation. The architecture enables a feasible system on all platforms (Windows, Mac OS, and Linux). To provide smooth interactions, a set of preprocesses are employed, including mutual information computations, quartet generation, and categorization tree construction.

## 6 Case Study

We evaluate the usability of DimScanner with two cases.

### 6.1 Seattle 911 calls and responses dataset

We gathered more than 60K Seattle 911 calls and responses dataset with time ranging from October 28, 2013 to February 14, 2014. The categorization tree (Figure 1) is constructed with 136 1D and 2D views generated by 16 dimensions of the data.

Without prior knowledge of the dataset, we start the exploration by filtering all 1D views. Partitioned by the categorization tree, four groups of views become clear, indicating that the dimensions may be divided into four groups in the data space. To verify our hypothesis, we highlight four groups in different colors (Figure 8) and check them individually.

Starting with the group in green [Figure 8 (a)], by highlighting the 2D views of those 1D variables, we find that all 2D views belong to the same partition of the tree, which confirms that all four dimensions are closely related. We further investigate the relation view of the dimensions because the 2D views do not reveal the relation pattern clearly. By sliding across the "district_or_sector" dimension, we discover the mapping pattern such that one district corresponds to three or four beats that are closely located on map (the view "latitude$\times$longitude"). In terms of the group in blue [Figure 8 (b)], we apply a similar exploration process and realize that these views contain many null values. And the dark line in the relation view indicates that null values have strong co-occurrences among dimensions (as shown in the relation view in the blue box), namely data with null values in one dimension usually contain null values in others. The group in orange [Figure 8 (c)] shares a similar relation pattern with the group in green: one value in one dimension corresponds to several values in the other, which can

[5]https://www.djangoproject.com/
[6]http://pandas.pydata.org

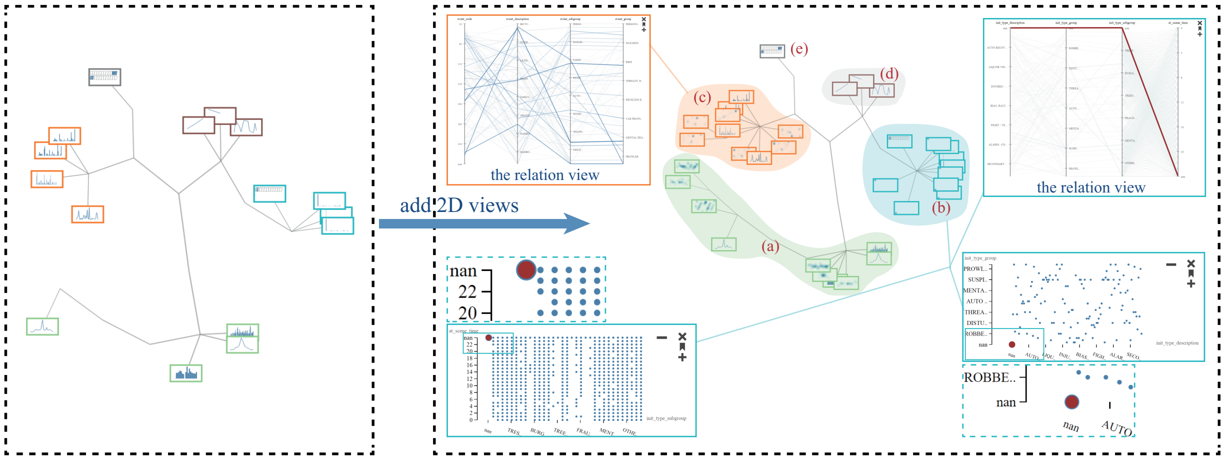


Figure 8: Four groups of data views are extracted from the categorization tree of the Seattle 911 calls and response dataset. Left: The categorization tree of 1D views reveals four groups. Right: 2D views that correspond to the dimensions of the 1D views are added. The pattern of views in green is described in Figure 4. No explicit relation patterns can be uncovered from views in gray. The colored highlights are generated manually just in the paper, not in the system.

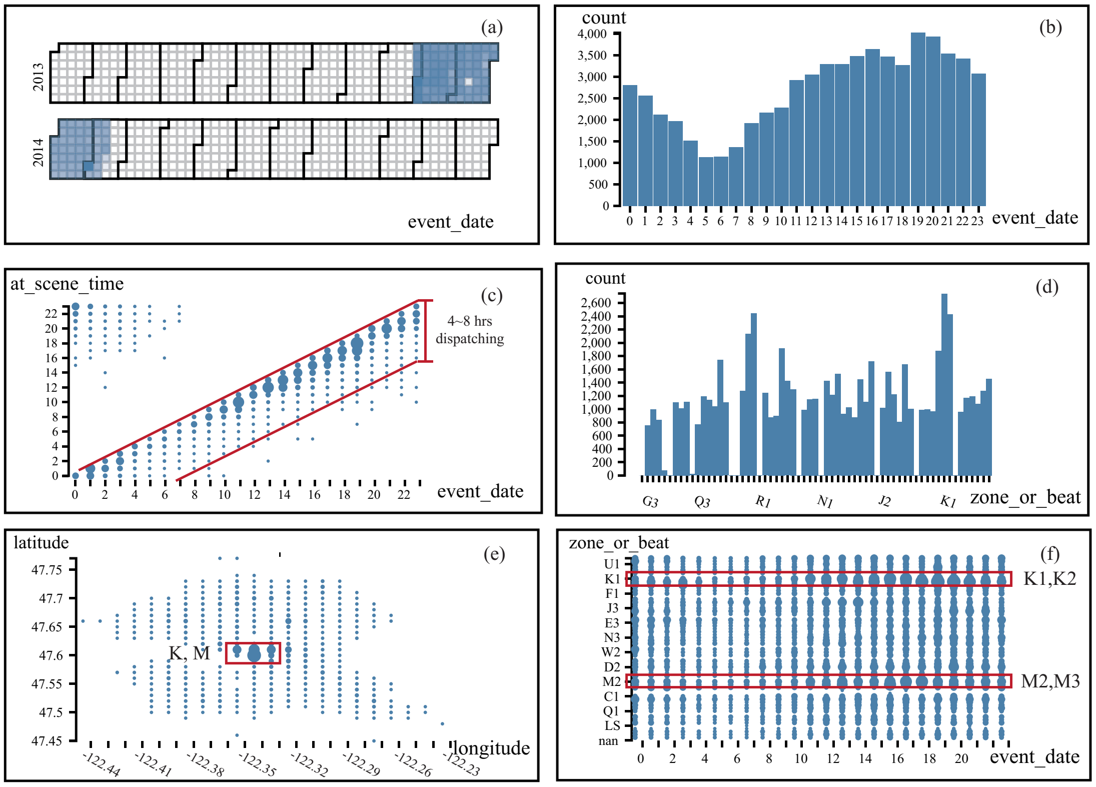


Figure 9: The views of dimension “at_scene_time”, “event_date” and “zone_or_beat”. (a, b) Aggregating view “event_date” (a) by hour (b), we can infer that the least amount of 911 calls were made in the early morning. (c) The police response time, namely the time difference, ranges from less than one hour to 8 hours. (d, f) More police should staff to district K1, K2 and M2, M3 while less police are needed in district E; more patrols are also needed in the evening and less police from 4 am. to 10 am.. (e) The 2D “latitude$\times$longitude” view indicates the most “dangerous” district in Seattle as downtown Seattle.

also be revealed from the name of the dimensions (see the relation view in the orange box). We state that although some patterns are common sense for domain experts given the dimension names, our method can help analysts discover the patterns even without the names. It can also help show groups of errors/empty records in the data as seen in this example. The group in light gray [Figure 8 (d)] also contains three 1D views, in which no explicit relation pattern is uncovered.

Based on the above observations, we keep the two time-related dimensions, the geo-related dimensions and one dimension in each group, and filter out other dimensions as well as the dimensions related to IDs. We utilize our system to suggest a police dispatching policy in terms of time and beat. The time-related view "event_date" concerns the time of the reports being responded. We aggregate the view by hour and realize that the least amount of 911 calls were made in the early morning [Figure 9 (b)] when citizens are asleep. The view "at_scene_time" concerns the time that the calls were made. By observing the corresponding 2D view of the two time-related views [Figure 9 (c)], we can infer that the police response time, namely the time difference, ranges from less than one hour to 8 hours. Most of the events can be cleared in less than 3 hours. The 2D view "zone_or_beat×event_date" (aggregated by hour) can be an indicator for police dispatching. After exploring this view [Figure 9 (f)], we recommend more police should staff to district K1, K2 and M2, M3 while less police are needed in district E; more patrols are also needed in the evening and less police from 4 am. to 10 am. By studying the 2D "latitude×longitude" view [Figure 9 (e)], we locate the most crime ridden district in Seattle as downtown Seattle. By highlighting the district K, we discover which type of events were mostly reported in view "event_group": "liquor_violations".

### 6.2 The mobile user profile dataset

Our second case study uses mobile user profiles from a data modeling challenge[7]. The dataset contains 380,000 user profiles (15D). The categorization tree is structured with 120 1D and 2D views generated by the 15 dimensions of the data.

To our surprise, many 1D views in this dataset are placed on one branch (Figure 10). After inspecting each view, we draw the conclusion that the values of some dimensions are distributed non-uniformly (as peaks) and values at the peaks co-occurred across dimensions. First, some views contain unreasonable values, e.g., some values in the "age" dimension are greater than 1000 or below 0 [Figure 10 (a)]. Second, we notice that the value distributions of numeric dimensions are often heavily influenced by some outliers, e.g., the "terminal_price" dimension [Figure 10 (b)] (the price of the mobile phone) are distributed non-uniformly because the price of some mobile phones can be up to 8000.

Therefore, we perform data cleaning operations before the structuring component. First, we filter missing values and unreasonable values. Second, for a dimension with outliers, we transform the dimension with a non-linear scaling to make sure that outliers gather together in one or two ranges. The new categorization tree (Figure 11) reveals some clustered structures. The structure (a) in orange shows a local branch of four views ("prob_level", "age", "cust_level", and "is_VIP") as well as their 2D views. We can infer that "cust_level" is more or less equivalent to "is_VIP" and positively related to "prob_level" (the level of service plan). The structure (b) in green depicts the relation between "value_added_fee", "web_fee" and "terminal_price". We can infer that "total_consumption" is independent of "web_fee" or "value_added_fee". In addition, according to the distribution of "web_fee", very few users will spend much money on "web_fee". It covers a discount package of web service and better web user experience to encourage more web usage. The structure (c) in blue describes the major contents of the total consumption, which includes call fees, roam call fees, long distance call fees, and SMS fees. It indicates that the total consumption mostly relates to call fee and SMS fee. Additionally, "long_call_fee" is most closely related to "roam_call_fee", both regarding calls between two cities.

## 7 Discussion

Our discussion contains four aspects: the choice of data views, comparisons with other relation measures and layouts, system scalability and limitations.

[7]http://www.mcm.edu.cn/html_cn/node/a4e8393c0d57cf879520183520b-34a9f.html

### 7.1 Data Views

Our approach employs 1D and 2D views for two reasons. First, 1D views alone are insufficient for a detailed exploration of multivariate datasets. In many situations, a 2D view indicates a specific context. For instance, the view "latitude × longitude" is much more relevant to "district_or_sector" and "zone_or_beat" than the view "latitude" or the view "longitude". Second, presenting views of arbitrary dimensions is time-consuming and unnecessary.

For either 1D or 2D views, both the structuring component and the exploration component can be adapted to nonorthogonal projected facets or other customized facets. The adaption does not affect the information-aware relation computation nor to view structuring and visualization. With nonorthogonal projections and user-defined facets, data analysts can randomly explore and dissect the data in combination with the semantic properties. Taking the police dispatching issue in 911 calls dataset as an example, a customized facet view of time difference can easily reveal the high variation in the service time for calls.

### 7.2 Comparisons

The DimScanner approach is a hierarchical clustering approach of data views based on dimensional relations. As a qualitative approach, the quartet-based analysis constructs the categorization (clustering) tree by maintaining as many close MI relations as possible.

Compared with techniques based on quality metrics [4], DimScanner can better reveal dimension-wise relationships in the data space. Scagnostics [32] is an image space technique that focuses on the "shape" of the plotted relations between dimensions. However, plots of similar shape do not reveal the relations between the underlying dimensions unless the plots all share a common dimension, as demonstrated in the case of TimeSeer [7]. Pixnostics [25] measures dimension-wise relation in the data space with Pearson's correlation or the normalized Euclidean distance alternatively. It also employs the standard K-means method as the clustering approach. Unlike Pixnostics and other data space approaches, the relation metric MI can effectively characterize both linear and non-linear relations, which can be further presented in a 2D plane with the proximity-preserving quartet analysis approach.

Coordinated views and parallel coordinate plots (PCPs) are two common techniques for multi-dimensional data exploration. GPLOM [15] places 2D data views in a lower-triangle matrix and integrates exploratory interactions, such as linking (highlighting) and filtering. By contrast, DimScanner reveals the clustering pattern of the dimensions with a categorization tree structure instead of a view matrix. PCPs depict distributions of all individual dimensions and selective selected dimension pairs. Meanwhile, with the increasing number of dimensions, the axis order in PCPs is crucial to discover dimension-wise correlations or clustering patterns. DimScanner avoids this problem by restricting the dimensions only to those that are highly related.

### 7.3 System scalability

The system scalability is mainly related to the number of dimensions, which determines not only the number of data views, but also the number of MI relations and the quartets. The number of quartets determines the computation complexity of the QMC algorithm. For an $N$-dimensional data, $O(N^2)$ data views are generated, thereby requiring $O(N^4)$ computations of MI relations for both 1D and 2D data views. To obtain a reliable categorization tree, $O(N^4)$ quartets are required for the QMC algorithm. With much larger dimension sizes, speed up techniques may be required to compute quartets. The number of data records has little influence on the computation complexity because a binning process is applied to data points as preprocessing. The tree layouts also restrict the number of data

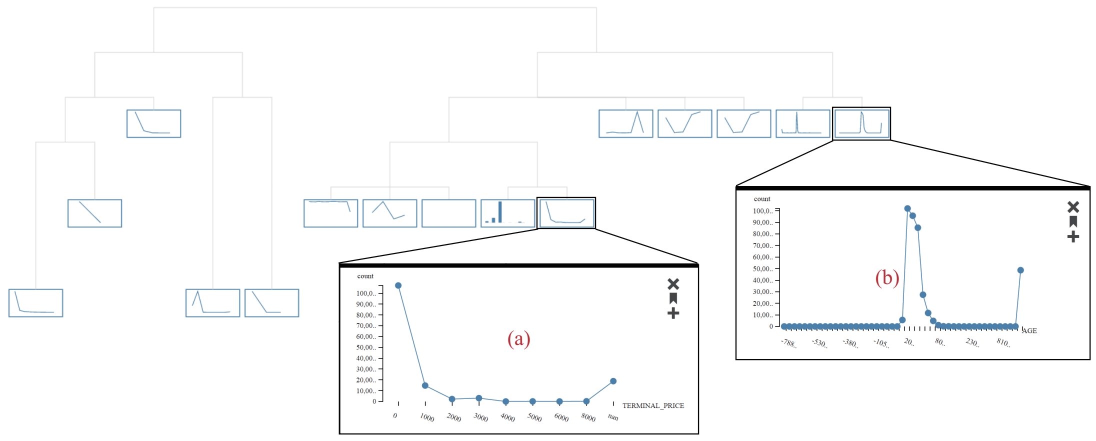


Figure 10: The categorization tree of the mobile user dataset before data cleaning. Some 1D views are grouped on one branch because they commonly contain one or two peaks (valleys) and the peak values co-occurred across dimensions. This implies that the values of some dimensions are distributed non-uniformly. (a) The "terminal price" view indicates a non-uniform distribution. (b) The "age" view exhibits a sharp and narrow peak. The colored highlights are generated manually just in the paper, not in the system.

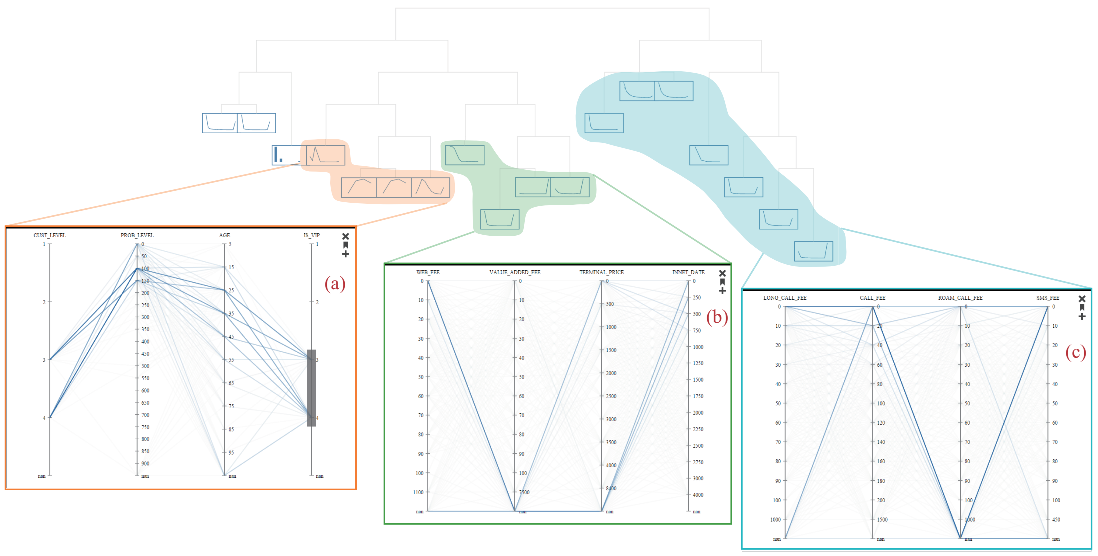


Figure 11: The categorization tree of the mobile user dataset after data cleaning. (c) The group in blue indicates the major components of the consumptions, which contains "long_call_fee", "call_fee", "roam_call_fee", and "SMS_fee".

views to a few hundred. In the two cases presented in this paper, the number of dimensions is limited to less than 20.

### 7.4 Limitations

Our system has some limitations.

**Data preprocessing** From the experiences of examining the mobile service dataset, we learn that the understanding of the categorization tree varies greatly with data quality. Nevertheless, the performance of data preprocessing (e.g., data cleaning) for improving data quality is proportional to the data size. Sampling might be a solution to address this problem, that is, a sample of raw data can be cleaned and categorized for an initial inspection. One of our future plans is to design more efficient solutions that allow for dynamic addition of data views. In this way, the categorization tree can be iteratively updated with the addition of new views.

**Enhanced view representations** DimScanner can be utilized to display random projection views. To calculate the relations between randomly projected views, one solution is to compute the relation in the image space, and integrate the distributions of data in the image space into computing MI. For categorical dimensions, we maintain their original orders in the dataset. Bertin stated the importance of ordering [2] and suggested that a better order may reveal cluster patterns of the dataset immediately. We aim to implement a reordering algorithm with multi-dimensional view/relation representation. Other reordering controls, such as ordering by the number of values, can also be implemented.

## 8 Conclusion

We have presented a novel data dissection scheme that constructs multiple data views and structures an information-aware data categorization tree. We demonstrate that a fair incorporation of quartet analysis effectively reduces cognitive load by focusing attention on the highly related dimensions.Our system offers rich user controls, and enables data analysts to get the contexts of an unfamiliar dataset.

In the future we plan to enable user-defined data views and nonorthogonal projection data views. We also expect to improve the computation efficiency by using in-memory databases and parallelizing the scanning process.

## Acknowledgements

This work is supported by National 973 Program of China (2015CB352503), Major Program of National Natural Science Foundation of China (61232012), National Natural Science Foundation of China (61422211), National Natural Science Foundation of China (61303134), Zhejiang Provincial Natural Science Foundation of China (LR13F020001), the Fundamental Research Funds for the Central Universities.

## References

[1] M. Ankerst, S. Berchtold, and D. A. Keim. Similarity clustering of dimensions for an enhanced visualization of multidimensional data. In *IEEE Symposium on Information Visualization*, pages 52–60. IEEE, 1998.

[2] J. Bertin. Semiology of graphics: diagrams, networks, maps. 1983.

[3] E. Bertini and G. Santucci. Quality metrics for 2d scatterplot graphics: automatically reducing visual clutter. In *Smart Graphics*, pages 77–89. Springer, 2004.

[4] E. Bertini, A. Tatu, and D. Keim. Quality metrics in high-dimensional data visualization: An overview and systematization. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2203–2212, 2011.

[5] I. Borg and P. J. Groenen. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005.

[6] R. Cilibrasi and P. Vitanyi. A new quartet tree heuristic for hierarchical clustering. *Theory of Evolutionary Algorithms*, 2006.

[7] T. N. Dang, A. Anand, and L. Wilkinson. Timeseer: Scagnostics for high-dimensional time series. *IEEE Transactions on Visualization and Computer Graphics*, 19(3):470–483, 2013.

[8] A. Dasgupta and R. Kosara. Pargnostics: Screen-space metrics for parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1017–1026, 2010.

[9] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. John Wiley & Sons, 2012.

[10] D. Engel, L. Hüttenberger, and B. Hamann. A survey of dimension reduction methods for high-dimensional data analysis and visualization. In *OASIcs-OpenAccess Series in Informatics*, volume 27. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2012.

[11] B. J. Ferdosi, H. Buddelmeijer, S. Trager, M. H. Wilkinson, and J. B. Roerdink. Finding and visualizing relevant subspaces for clustering high-dimensional astronomical data using connected morphological operators. In *IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pages 35–42. IEEE, 2010.

[12] S.-S. Huang, A. Shamir, C.-H. Shen, H. Zhang, A. Sheffer, S.-M. Hu, and D. Cohen-Or. Qualitative organization of collections of shapes via quartet analysis. *ACM Transactions on Graphics*, 32(4):71:1–71:10, 2013.

[13] I. Hur, S.-H. Kim, A. Samak, and J. S. Yi. A comparative study of three sorting techniques in performing cognitive tasks on a tabular representation. *International Journal of Human-Computer Interaction*, 29(6):379–390, 2013.

[14] I. Hur and J. S. Yi. Simulsort: Multivariate data exploration through an enhanced sorting technique. In *Proceedings of the 13th International Conference on Human-Computer Interaction. Part II: Novel Interaction Methods and Techniques*, pages 684–693. Springer, 2009.

[15] J.-F. Im, M. J. McGuffin, and R. Leung. GPLOM: the generalized plot matrix for visualizing multidimensional multivariate data. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2606–2614, 2013.

[16] W. Javed and N. Elmqvist. Explates: spatializing interactive analysis to scaffold visual exploration. *Computer Graphics Forum*, 32(3):441–450, 2013.

[17] S. Johansson and J. Johansson. Interactive dimensionality reduction through user-defined combinations of quality metrics. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):993–1000, 2009.

[18] I. Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[19] T. Kohonen. *Self-organizing maps*, volume 30. Springer Science & Business Media, 2001.

[20] I. Kramosil and J. Michálek. Fuzzy metrics and statistical metric spaces. *Kybernetika*, 11(5):336–344, 1975.

[21] E. Levina and P. J. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in neural information processing systems*, pages 777–784, 2004.

[22] D. J. C. Mackay. Information theory, inference and learning algorithms. *Information Theory*, page 640, 2003.

[23] S. MacNeil and N. Elmqvist. Visualization mosaics for multivariate visual exploration. *Computer Graphics Forum*, 32(6):38–50, 2013.

[24] R. Rao and S. K. Card. The table lens: Merging graphical and symbolic representations in an interactive focus + context visualization for tabular information. In *Proceedings of Conference Companion on Human Factors in Computing Systems*, pages 318–322. ACM, 1994.

[25] J. Schneidewind, M. Sips, D. Keim, et al. Pixnostics: Towards measuring the value of visualization. In *Proceedings of IEEE Symposium On Visual Analytics Science And Technology*, pages 199–206. IEEE, 2006.

[26] J. Seo and B. Shneiderman. A rank-by-feature framework for unsupervised multidimensional data exploration using low dimensional projections. In *IEEE Symposium on Information Visualization*, pages 65–72. IEEE, 2004.

[27] S. Snir and S. Rao. Quartets maxcut: A divide and conquer quartets algorithm. *IEEE/ACM Transaction on Computational Biology and Bioinformatics*, 7(4):704 – 718, 2010.

[28] K. Strimmer and A. Von Haeseler. Quartet puzzling: a quartet maximum-likelihood method for reconstructing tree topologies. *Molecular Biology and Evolution*, 13(7):964 – 969, 1996.

[29] A. Tatu, G. Albuquerque, M. Eisemann, J. Schneidewind, H. Theisel, M. Magnor, and D. Keim. Combining automated analysis and visualization techniques for effective exploration of high-dimensional data. In *Proceedings of IEEE Symposium on Visual Analytics Science and Technology, 2009*, pages 59–66. IEEE, 2009.

[30] C. E. Weaver. *Improvise: A user interface for interactive construction of highly-coordinated visualizations*. PhD thesis, University of Wisconsin at Madison, 2006.

[31] L. Wilkinson, A. Anand, and R. Grossman. High-dimensional visual analytics: Interactive exploration guided by pairwise views of point distributions. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1363–1372, 2006.

[32] L. Wilkinson, A. Anand, and R. L. Grossman. Graph-theoretic scagnostics. In *Proceedings of IEEE Symposium on Information Visualization*, volume 5, page 21, 2005.

[33] S. J. Wilson. Building phylogenetic trees from quartets by using local inconsistency measures. *Molecular Biology and Evolution*, 16(5):685 – 693, 1998.

[34] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):649–658, 2016.

[35] X. Yuan, D. Ren, Z. Wang, and C. Guo. Dimension projection matrix/tree: Interactive subspace visual exploration and analysis of high dimensional data. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2625–2633, 2013.