`VizInNuce`: Efficient Data-Driven Visual Summarization to Support Visual Analytics

Himel Dev, Doris Lee
hdev3,jlee782@illinois.edu

June 27, 2017

Abstract

We present `VizInNuce`, an interactive visualization recommendation (`VizRec`) system to summarize patterns or trends: given a subset of data and attributes to be studied, `VizInNuce` intelligently explores the lattice of visualizations, evaluates the trends of visualizations with respect to their predecessors in the lattice, and recommends those it deems most "interesting" and "informative". Motivated by collaborations with domain scientists who search for patterns or trends in hundreds of subsets of data, `VizInNuce` automatically recommends appropriate visualizations based on "interestingness" and "informativeness", and adopts a directed acyclic graph (DAG) structure to organize the recommended visualizations into an interactive visual dashboard.

1 Problem Formulation

Given a subset of data and attributes to be studied, the goal for `VizInNuce` is to identify the most interesting and informative visualizations summarizing patterns or trends. There are two major challenges in achieving this goal; (i) scale: evaluating a large number of candidate visualizations in lattice while responding within interactive time scales, and (ii) utility: identifying appropriate metrics for assessing interestingness and informativeness of visualizations. For now, we focus on the latter. We first describe two conceptual visualization lattices, and then describe the .

1.1 Preliminaries

A dataset D consists of a set of dimension attributes A, and a set of measure attributes M. We define a metric family z as the result of a SQL query using a group of dimension attributes $A_z \subseteq A$, a single measure attribute $m_z \in M$, and an aggregate function f_z as follows: `SELECT` A_z `AS` $metric\_id, f(m_z)$ `AS` $metric\_value$ `FROM` D `GROUP BY` A_z. The result of this query can be presented using a table with two columns, where the first column accommodates the metrics within the metric family, and the second column accommodates the values corresponding to these metrics. This two column table can be visualized with a bar chart, where each metric is visualized using a separate bar.

Notice that the aforementioned query does not have a `WHERE` clause. If we include a `WHERE` clause in this query incorporating constraints on the remaining dimension attributes $A'_z = A - A_z$, we shall get different instances of z corresponding to different data subsets defined by the constraints. These different instances of z can be visualized with separate bar charts that present the same metric family for different data subsets.

1.2 Data Subset Lattice

A data subset can be contained within another data subset. Specifically, given a data subset defined by a set of constraints on dimension attributes, adding one or more new constraints in the set will generate a new data subset that is contained within the former subset. We specify a data subset as the *parent* of another data subset if the latter is defined by a set of constraints that includes one additional constraint along with the ones that define the former subset. Notice that, as per our definition, a data subset can have multiple parent subsets.

We extend the idea of parent data subset to introduce the concept of parent visualization. Let $V_s(z)$ represent the set of visualizations that visualize a metric family z for different data subsets. Given a visualization $v_i \in V_s(z)$ (corresponding to a particular data subset), parent of v_i, $v_i^j \in V_s(z)$ is a visualization that corresponds to a parent data subset of the former. Analogous to a data subset, a visualization $v_i \in V_s(z)$ can have multiple parents.

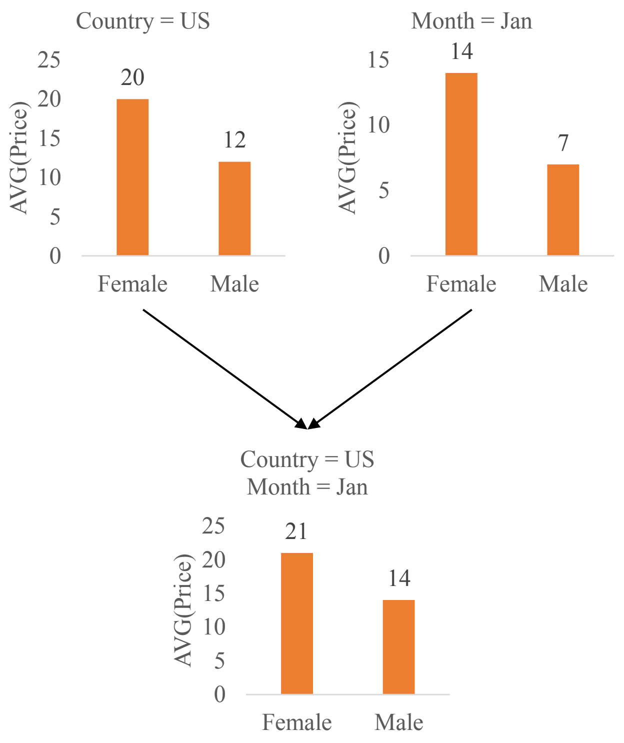

Figure 1: Parent Child Relationship in Data Subset Lattice

1.3 Dimension Combination Lattice

Given a combination of dimension attributes, inducing one or more new dimensions in the combination will generate a new combination. We specify a dimension combination as the *parent* of another dimension combination if the latter is defined by a set of dimensions that includes one additional dimension along with the ones that define the former combination. Notice that, as per our definition, a dimension combination can have multiple parent combinations.

Let $V_c(Z)$ represent the set of visualizations that visualize the different metric families with same measure of interest and same data subset. We extend the idea of parent dimension combination to introduce the concept of parent visualization. Specifically, given a visualization $v_i \in V_c(Z)$ (corresponding to a particular dimension combination), parent of v_i, $v_i^j \in V_c(Z)$ is a visualization that corresponds to a parent dimension combination of the former. Analogous to a dimension combination, a visualization $v_i \in V_c(Z)$ can have multiple parents.

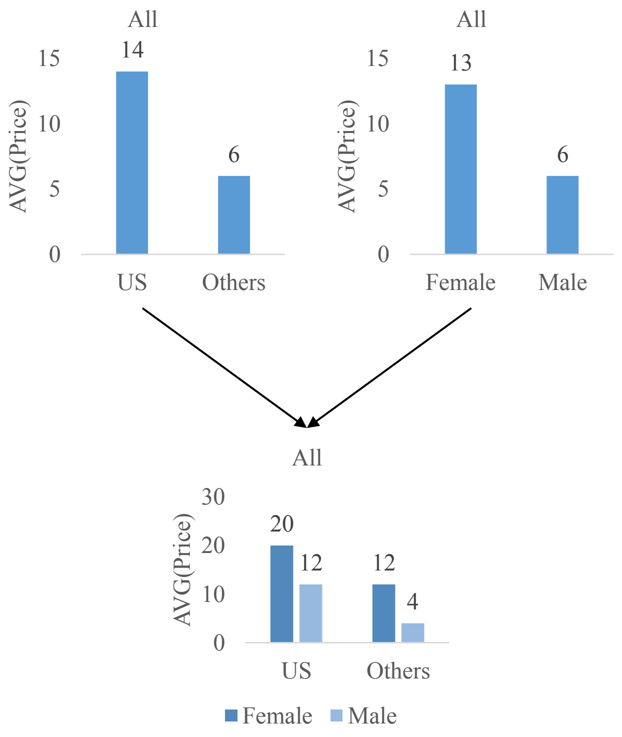

Figure 2: Parent Child Relationship in Dimension Combination Lattice

2 Parent Child Relationship Model

We observe that, while exploring a dataset, users first look at the top level visualizations before looking at the next level. Further, the knowledge learnt from the top level visualizations set up user expectation for the next level. Based on these two observations, we quantify the parent-child relationship mentioned above.

2.1 User Expectation Model

We formulate user expectation ($\hat{v}_i$) corresponding to a visualization v_i, based on its parents, $v_i^1, \ldots, v_i^\lambda$.

$$\hat{v}_i = g(v_i^1, \ldots, v_i^\lambda)$$

Here, the function $g()$ takes the parents of v_i as argument and returns user expectation $\hat{v}_i$.

We assert that the parents of v_i do not contribute equally in setting up user expectation $\hat{v}_i$. We care about the p parents that contribute most in setting up $\hat{v}_i$. We call these p parents the *true parents* of v_i and represent them as $[v_i^*, p]$.

$$[v_i^*, p] = \text{arg } \underset{v_i^j}{\text{max\_p}}\, d(v_i, v_i^j)$$

Here, the function $d()$ takes v_i and its parent v_i^j as argument and returns the contribution of v_i^j in setting up $\hat{v}_i$.