DashBot: An ML-Guided Dashboard Generation System

Sandrine Da Col
sandrine.da-col@univ-grenoble-alpes.fr
Univ. Grenoble Alpes

Radu Ciucanu
radu.ciucanu@insa-cvl.fr
INSA Centre Val de Loire, LIFO

Marta Soare
marta.soare@univ-orleans.fr
Univ. Orléans, LIFO

Nassim Bouarour
nassim.bouarour@univ-grenoble-alpes.fr
CNRS, Univ. Grenoble Alpes

Sihem Amer-Yahia
sihem.amer-yahia@univ-grenoble-alpes.fr
CNRS, Univ. Grenoble Alpes

ABSTRACT

Data summarization provides a bird's eye view of data and groupby queries have been the method of choice for data summarization. Such queries provide the ability to group by some attributes and aggregate by others, and their results can be coupled with a visualization to convey insights. The number of possible groupbys that can be computed over a dataset is quite large which naturally calls for developing approaches to aid users in choosing which groupbys best summarize data. We demonstrate DashBot, a system that leverages Machine Learning to guide users in generating data-driven and customized dashboards. A dashboard contains a set of panels, each of which is a groupby query. DashBot iteratively recommends the most relevant panel while ensuring coverage. Relevance is computed based on intrinsic measures of the dataset and coverage aims to provide comprehensive summaries. DashBot relies on a Multi-Armed Bandits (MABs) approach to balance exploitation of relevance and exploration of different regions of the data to achieve coverage. Users can provide feedback and explanations to customize recommended panels. We demonstrate the utility and features of DashBot on different datasets.

CCS CONCEPTS

• **Information systems** → **Data management systems**; *Database utilities and tools*; • **Mathematics of computing** → *Exploratory data analysis*; • **Theory of computation** → *Sequential decision making*; • **Computing methodologies** → *Reinforcement learning*.

KEYWORDS

Interactive data summarization and exploration; Grouping and aggregation queries; User feedback; Multi-armed bandits (MAB);

ACM Reference Format:
Sandrine Da Col, Radu Ciucanu, Marta Soare, Nassim Bouarour, and Sihem Amer-Yahia. 2021. DashBot: An ML-Guided Dashboard Generation System. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3459637.3481968

CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia

ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3481968

Name	Age	Gender	BMI	Age*	BMI*
Alice	20	F	25.4	[10, 25]	[18.3, 25.4]
Bob	25	M	31.8	[10, 25]	[31.8, 31.8]
Charlie	50	M	18.3	[50, 63]	[18.3, 25.4]
David	10	M	20	[10, 25]	[18.3, 25.4]
Eve	63	F	21	[50, 63]	[18.3, 25.4]

(a) Enriched relation R^*, obtained for input relation R containing the first 4 columns, and for parameter $k = 2$.

Gender	min(Age)	max(Age)	count(Age)	avg(Age)
F	20	63	2	41.5
M	10	50	3	28.33

BMI*	Gender	count(Name)	min(Age)	max(Age)
[18.3, 25.4]	F	2	20	63
[18.3, 25.4]	M	2	10	50
[31.8, 31.8]	M	1	25	25

(b) A dashboard consisting of two panels specified during interactions between the user and DashBot.

Figure 1: Example of data and panels.

1 INTRODUCTION

The need to summarize data arises with data availability. Groupby queries are a useful way to achieve such a goal. Notable examples are the dashboards generated in Behavior Analytics Systems such as Qualtrics.[1] Building such interfaces requires high data expertise, high user effort, and knowledge of the underlying data distributions, to best choose groupby and aggregation attributes. We demonstrate DashBot, a system that leverages Machine Learning to guide users in generating *data-driven* and *customized* dashboards.

A *dashboard* is a collection of panels, where each *panel* displays aggregated statistics of a selected data region in a user-friendly manner. That is, a panel is a groupby query of the form SELECT… FROM… GROUP BY….[2] The SELECT may have any of the five standard SQL aggregation functions (min, max, count, sum, avg). The number of possible groupbys that can be computed over a dataset is quite large necessitating high user effort to find relevant panels. DashBot offers guidance for building dashboards. To minimize effort, user interactions in DashBot are very simple: click on some predefined options and provide feedback for panel customization. Most of the work is done by DashBot while also letting the user maintain control over customization and writing other groupbys. This is an important departure from the related work on data summarization.

DashBot relies on *relevance*, *coverage*, and a *machine learning algorithm* to iteratively recommend panels. For each panel proposed

[1] https://www.qualtrics.com/support/vocalize/widgets/creating-cx-dashboard-pages/
[2] We assume a single relation, which may be an existing relation in the database or some denormalized data as result of some join query.

Figure 2: Interface of DashBot.

by DashBot, the user can provide Boolean feedback: if she likes the candidate panel (Yes feedback), it is added to the dashboard; otherwise, she may customize it by providing an explanation, or by relying on a machine learning algorithm when no explanation is given. To achieve that, DashBot combines three appealing features:

(i) Panel Generation. The input data is a relation R (see example in Fig. 1(a)). Fig. 1(b) shows two panels, i.e., groupby queries, generated from R. To generate a panel, DashBot uses a notion of *relevance* that is *data-driven* and combines statistical measures on groupby and aggregation attributes. DashBot also ensures *coverage* of attributes, by continuously updating relevance according to panels added to the dashboard.

(ii) Panel Customization. The number of groupby queries that one can specify over a relation grows exponentially with its number of attributes.[3] Additionally, some groupbys may be of interest to users more than others. Therefore, we propose to solicit user's feedback on a recommended panel. If the user says Yes, the panel is added to the current dashboard. If the user says No and provides an explanation, it is used to customize panel relevance and generate a new panel. If the user says No and does not provide an explanation, we model panel customization with *Multi-Armed Bandits (MABs)*, a popular *reinforcement learning* model [9, Ch. 2]. The goal is to identify the next candidate panel by seeking a trade-off between *exploiting* panels close to what was shown to the user based on relevance vs *exploring* completely different panels.

(iii) Panel Visualization. Given a groupby query, the user has the option of mapping its results to different visual elements (e.g., pie chart or histogram). The user has also the option to personalize panel visualization by providing attribute values to bias the mapping of panels into visual elements. Those values could be the user's own information which leads to *personalizing* panel visualization.

In Section 2, we present an overview of DashBot. Section 3 describes our demonstration scenarios.

Related work. The closest work to ours is interactive summarization and exploration of top-k aggregate query answers [11]. While this work assumes that data is sorted according to a score attribute that is known in advance and relies on drill-down wildcards [6, 7], DashBot does not make assumptions on the existence of a score attribute, and computes panel relevance based on data properties.

Similarly to SeeDB [10], DashBot recommends interesting visualizations using grouping and aggregation. Multiple other works rely on utility functions to recommend visualizations [1, 4, 12, 13]. In particular, SeeDB relies on a utility function for assessing the interestingness of a visualization: the larger its deviation from some reference, the more likely it will be chosen. DashBot distinguishes itself by providing to users the freedom to guide panel generation, via Yes/No feedback and explanations, instead of assuming a reference. Additionally, panel relevance relies on data properties, which is different from the aforementioned research where data-driven means some utility function computed based on data.

Similarly to AIDE [3], DashBot incorporates user feedback in interactive data exploration. The focus of DashBot is on groupby queries while AIDE builds a query based on generating a decision tree to classify tuples into relevant/irrelevant. Additionally, DashBot relies on MABs to select a panel under uncertainty, i.e., when user feedback is limited.

Two other related systems are Vizdom [2] and QUDE [5]. Vizdom visualizes results of ML algorithms, in a scenario where tasks and subsets of training data are interactively chosen by the user. QUDE focuses on detecting statistical pitfalls (e.g., Simpson's paradox) during data exploration. These systems assume that the user has prior knowledge on interesting attributes and data analysis tasks. The goals and hypotheses of DashBot are different: it rather aims at summarization based on data-driven measures and subsequent user interactions. No assumption is made on user knowledge.

2 SYSTEM OVERVIEW

DashBot provides a user-friendly interface depicted in Fig. 2. The user first chooses a dataset and starts the dashboard generation. Zone A is where the recommended panel is displayed. DashBot visualizations rely on the VegaLite grammar[4] which provides a flexible and powerful mapping of groupby results to visual elements. Zone B contains the dashboard under construction with the chosen panels so far. Zone C is the interactivity zone where users can provide feedback on the current panel, add data to personalized visualization, and specify groupby queries from scratch. Zone D displays information on attributes: coverage, relevance and value distribution. The workflow of DashBot is shown in Fig. 3.

[3] Let x be the number of attributes of a relation R. The total number of distinct SQL groupbys that can be specified over R is: $\sum_{y=1}^{x-1}\left(\binom{x}{y} \times \left(\sum_{z=1}^{y}\binom{y}{z}\right) \times (2^{5\times(x-y)}-1)\right)$.

[4] https://vega.github.io/vega-lite/

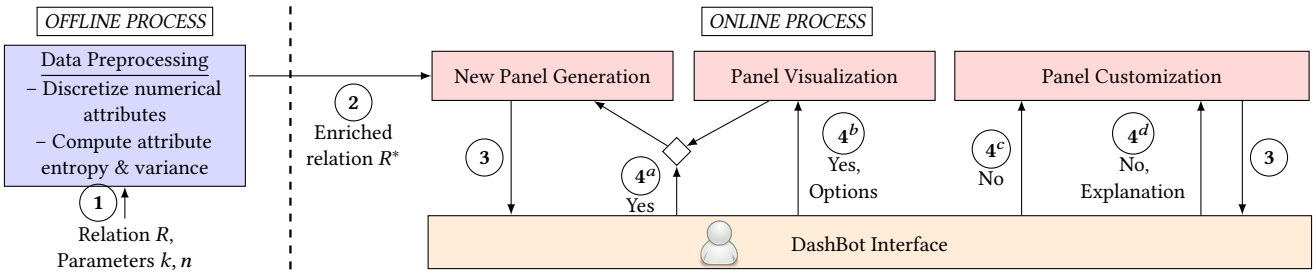

Figure 3: Architecture of DashBot. Each edge labeled 3 coming into the DashBot interface should be read: "*Ask feedback*". The diamond should be read: "*Add panel on dashboard. If the halt condition is satisfied, then output all Yes-labeled panels; otherwise, go to New Panel Generation*".

2.1 Data Preprocessor

Preprocessing (cf. Fig. 3) is done offline to prepare attributes. Categorical attributes can be used for groupby or aggregated with count. Numerical attributes can be used as such for all aggregation functions. As for groupby, numerical attributes with more than n distinct values (threshold parameter) are discretized using k-means clustering, with $k \leq n$.[5] The idea is to avoid generating as many groups as roughly the number of tuples in the dataset. The second part of preprocessing focuses on precomputing *entropy* and *variance* of attributes, which are then used to compute panel relevance.

We define *axioms* to prioritize attributes to be chosen as groupby or aggregation dimensions:

(1) *Attributes kept in their initial form*, since discretization yields information loss. Hence, the priorities for groupby are: (a) attributes for which the number of distinct values is $\leq n$ without need for discretization; (b) numerical attributes that are discretized into k clusters; (c) non-numerical attributes for which the number of distinct values is $> n$.

(2) *Attributes that yield balanced groups.* Functions min/max/avg have more sense if we apply them on groups of similar sizes. Hence, we favor to group by attributes with high entropy.

(3) *Attributes with many applicable aggregation functions.* Hence, the priorities for aggregation are: (a) numerical attributes with high variance, for which all aggregation functions might be interesting; (b) numerical attributes with low variance, for which min/max/avg might be redundant; (c) non-numerical attributes for which count is the only possible function.

For example, $n = k = 2$ leads to adding two discretized attributes to R to obtain the enriched relation R^* shown in Fig. 1(a). Using *axioms* 1 and 2, the proposed ranking of groupby dimensions is: `Gender` > `Age*`> `BMI*`> `Name`. Indeed, Gender and Age* have the same entropy, but Gender is not discretized. BMI* has the smallest entropy. Name is a string attribute that cannot be discretized, hence it has the lowest rank. For aggregation dimensions, since var(`Age`) > var(`BMI`), *axiom* 3 gives: `Age` > `BMI` > `Gender` = `Name`.

2.2 New Panel Generator

As shown in Fig. 3, the new panel generation is called once at the beginning of the online process, and subsequently as many times as needed after a Yes feedback. DashBot displays the most relevant panel according to entropy, variance, and coverage, while making sure to not show panels that have been seen. To illustrate how DashBot relies on *entropy* and *variance* to propose a panel, we recall the rankings computed at the end of Section 2.1, based on which DashBot first proposes a panel that shows the result of a groupby on Gender, and aggregation on Age:

Gender	min(Age)	max(Age)	count(Age)	sum(Age)	avg(Age)
F	20	63	2	83	41.5
M	10	50	3	85	28.33

Assuming a Yes feedback (Steps 4^a and 4^b in Fig. 3), the panel is added to the dashboard. At each iteration, DashBot aims at *covering* remaining attributes, according to groupby and aggregation rankings. Hence, proposing a new panel takes into account the panels already present on the dashboard. On our example, the next generated panel should group by BMI* and aggregate on Name.

We discuss the case where a user gives a No feedback with explanation in Section 2.3 and the case of No feedback without explanation in Section 2.4.

2.3 Panel Customization with Explanation

The user may give a No feedback with explanation (Step 4^d in Fig. 3): "Bad groupby attribute", "Bad aggregation attribute", "Change aggregation functions". DashBot then modifies the last proposed panel accordingly. For example, assume that the user visualizes the panel from Section 2.2 and gives No feedback with explanation "Change aggregation functions" on "sum", DashBot drops the sum which results in suggesting the first panel from Fig. 1(b).

During panel customization, DashBot also allows the user to choose other aggregate or groupby attributes, which may yield a panel that is completely different from the proposed panel. DashBot ensures that a panel that already received user feedback (either Yes or No) is not shown anymore.

2.4 Panel Customization without Explanation

After each No feedback without explanation (Step 4^c in Fig. 3), DashBot models panel customization as a MAB problem.

DashBot currently supports two algorithms (ε-greedy and Softmax), each providing a different exploration-exploitation trade-off.

ε-greedy (with $\varepsilon = 0.1$)

- *Exploit* with probability $1-\varepsilon$: most of the time, shows a panel that only slightly differs from the rejected one.

[5] DashBot proposes default values for these parameters, and also allows the user to tune them. Other discretization methods than k-means (e.g., Tableau's auto-binning) can be easily added in DashBot.

- *Explore* with probability ε: once in a while, shows a panel that is very different from the one showed at the previous iteration.

Choosing with high probability to show a panel that is close to the last one helps preserve the user's stream-of-consciousness [8]. Furthermore, seeing several similar panels helps the user better understand what she does not want and provide an explanation. Proposing with low probability a customized panel that is very different from previous ones avoids local optima and reveals currently unexplored data regions. When exploiting (resp., exploring), the newly recommended panel is randomly chosen in the close (resp., distant) space of possible panels, provided that it has not already been suggested or discarded by a previous explanation "Bad groupby attributes".

Softmax. A more sophisticated way of customizing a panel is based on *scores* associated to the possible reasons for a No feedback. Even if the user does not provide an explanation, we can compute scores for the *likelihood of an explanation.* For each possible explanation of a No feedback, we store (i) the number of times the explanation has been used before in panel customization, and (ii) the number of times the explanation has been used before in panel customization that led to a panel for which the user provided a Yes feedback. To initialize Softmax, DashBot tries each explanation once and initializes the aforementioned variables. During the exploration-exploitation phase, DashBot computes a score (i.e., a probability matching) for each explanation according to a Boltzmann distribution [9, Ch. 2], chooses an explanation based on the probability matching, proposes a new panel accordingly, observes user feedback, and updates variables for the chosen explanation.

We stress that the semantics of the exploration-exploitation dilemma is different from the ε-greedy strategy introduced earlier: for ε-greedy, changes are randomly-chosen over a small (resp. large) number of features in the panel to be customized, whereas for Softmax, we target the most (resp. least) likely explanation.

For both algorithms, the goal is to minimize the number of panels that the user labels No before converging to a dashboard that satisfies her. We model each panel with a binary vector having 6 bits for each attribute A, encoding the possible roles of A in a query (part of the groupby or of some of the 5 aggregate functions). We measure the distance between panels as the distance between vectors. Although the new panel generation always proposes a panel that groups by a single attribute, panel customization can propose a panel that groups by multiple attributes. In practice, ε-greedy is more efficient for dashboards consisting of few panels, while Softmax is better for dashboards consisting of many panels, where same explanations apply to customize panels at different iterations.

2.5 Panel Visualization

DashBot displays panels either as tables e.g., Fig. 1(b), or using visual elements e.g., Fig 4. DashBot proposes visualization by default depending on the type of attributes. For instance, in Fig 4, each bar represents the average value of the aggregation attribute for a group. The min and max values are the extremes of the arrow and the counts are displayed with a color gradient. The user may request to switch to different visual elements (see examples in Fig. 2) or provide data against which the visualization is personalized. For example, Fig. 4 shows a personalized panel where the user entered age and BMI information that is shown in the visualization.

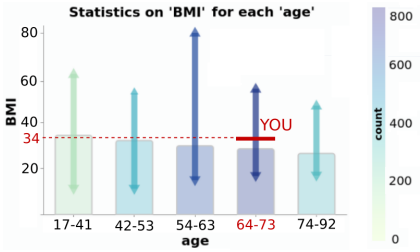

Figure 4: A personalized panel visualization.

3 DEMONSTRATION SCENARIOS

Our first scenario motivates the need for iterative data summarization. Our second scenario showcases the features of DashBot: computing panel relevance, customization by applying MABs, customization with user feedback, and visualization. In both scenarios, attendees will get to choose among several datasets including medical data, customer data in retail, or a synthetic dataset consisting of ~10 attributes and few hundreds tuples. They can also choose among proposed questions that constitute the purpose of their summarization. For instance, on medical data they will be asked questions such as "Do middle-aged patients have higher BMI than others?" and "Do they use ventilation very often?"

Motivating DashBot. The goal of this first scenario is to illustrate the need of interactive systems that allow efficient and user-friendly dashboard generation. Attendees will first be invited to explore data without DashBot. Attendees will use DashBot to seek answers to the proposed questions and will see that they can be obtained in a few exploration steps only. To further appreciate that, they will be shown a quantification of the number of possible groupbys that one may specify (e.g., 10^{15} possible queries for 10 attributes). At any point, attendees may choose to express their own groupby query via the interface. This scenario will show that unless one perfectly masters the schema of a dataset, it is not trivial to quickly get insightful summaries and find answers to the proposed questions. That remains true even for relatively small data sizes.

Showcasing DashBot. Attendees will effectively use DashBot and run an entire workflow to generate a dashboard. They will first see that relevance computation will allow them to generate panels *without prior knowledge on attributes.* Various settings of parameters n and k will be tried to point out their impact on DashBot's rankings of attributes for groupby and aggregation. Attendees will then get *to choose between the two implementations of MABs for panel customization.* They will discover the ability to provide an explanation to discard attributes ("Bad groupby attribute", "Bad aggregation attribute") or choose the most relevant aggregation functions for a given aggregation attribute ("Change aggregation functions"). They will see how that is used to *customize the current panel,* as well as how it is taken into account by the system in subsequent iterations. At any time, attendees will be able to switch to a different mapping to visual elements as well as provide data against which *panel visualization will be personalized.*

REFERENCES

[1] O. Bar El, T. Milo, and A. Somech. 2020. Automatically Generating Data Exploration Sessions Using Deep Reinforcement Learning. In *SIGMOD*. 1527–1537.

[2] A. Crotty, A. Galakatos, E. Zgraggen, C. Binnig, and T. Kraska. 2015. Vizdom: Interactive Analytics through Pen and Touch. *PVLDB* 8, 12 (2015), 2024–2027.

[3] K. Dimitriadou, O. Papaemmanouil, and Y. Diao. 2016. AIDE: An Active Learning-Based Approach for Interactive Data Exploration. *TKDE* 28, 11 (2016), 2842–2856.

[4] R. Ding, S. Han, Y. Xu, H. Zhang, and D. Zhang. 2019. QuickInsights: Quick and Automatic Discovery of Insights from Multi-Dimensional Data. In *SIGMOD*. 317–332.

[5] Y. Guo, C. Binnig, and T. Kraska. 2017. What you see is not what you get!: Detecting Simpson's Paradoxes during Data Exploration. In *HILDA@SIGMOD*. 2:1–2:5.

[6] M. Joglekar, H. Garcia-Molina, and A. G. Parameswaran. 2019. Interactive Data Exploration with Smart Drill-Down. *TKDE* 31, 1 (2019), 46–60.

[7] S. Sarawagi. 2001. User-Cognizant Multidimensional Analysis. *VLDB J.* 10, 2-3 (2001), 224–239.

[8] D. Shahaf and C. Guestrin. 2010. Connecting the Dots Between News Articles. In *KDD*. 623–632.

[9] R. S. Sutton and A. G. Barto. 2018. *Reinforcement Learning: An Introduction* (second ed.). The MIT Press. http://incompleteideas.net/book/the-book-2nd.html

[10] M. Vartak, S. Rahman, S. Madden, A. G. Parameswaran, and N. Polyzotis. 2015. SEEDB: Efficient Data-Driven Visualization Recommendations to Support Visual Analytics. *PVLDB* 8, 13 (2015), 2182–2193.

[11] Y. Wen, X. Zhu, S. Roy, and J. Yang. 2018. Interactive Summarization and Exploration of Top Aggregate Query Answers. *PVLDB* 11, 13 (2018), 2196–2208.

[12] K. Wongsuphasawat, Z. Qu, D. Moritz, R. Chang, F. Ouk, A. Anand, J. D. Mackinlay, B. Howe, and J. Heer. 2017. Voyager 2: Augmenting Visual Analysis with Partial View Specifications. In *CHI*. 2648–2659.

[13] X. Zhang, X. Ge, and P. Chrysanthis. 2020. Interactive View Recommendation with a Utility Function of a General Form. In *HILDA@SIGMOD*.