DIVE: A Mixed-Initiative System Supporting Integrated Data Exploration Workflows

Kevin Hu
MIT Media Lab
Cambridge, Massachusetts
kzh@mit.edu

Diana Orghian
MIT Media Lab
Cambridge, Massachusetts
dorghian@mit.edu

César Hidalgo
MIT Media Lab
Cambridge, Massachusetts
hidalgo@mit.edu

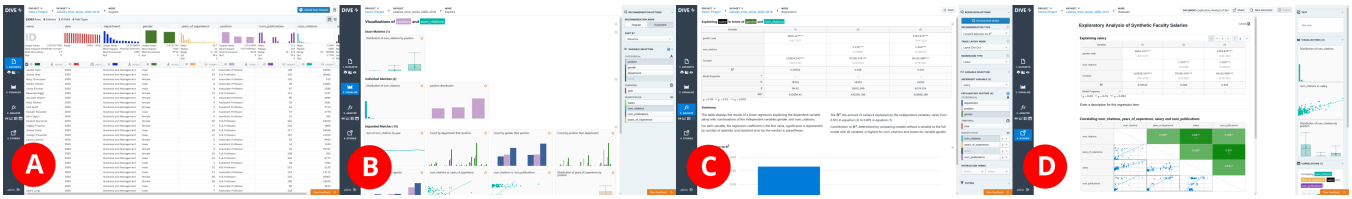

Figure 1: *Four stages of a data exploration workflow in DIVE: (A) Datasets, (B) Visualize, (C) Analysis, and (D) Stories.*

ABSTRACT

Generating knowledge from data is an increasingly important activity. This process of data exploration consists of multiple tasks: data ingestion, visualization, statistical analysis, and storytelling. Though these tasks are complementary, analysts often execute them in separate tools. Moreover, these tools have steep learning curves due to their reliance on manual query specification. Here, we describe the design and implementation of DIVE, a web-based system that integrates state-of-the-art data exploration features into a single tool. DIVE contributes a mixed-initiative interaction scheme that combines recommendation with point-and-click manual specification, and a consistent visual language that unifies different stages of the data exploration workflow. In a controlled user study with 67 professional data scientists, we find that DIVE users were significantly more successful and faster than Excel users at completing predefined data visualization and analysis tasks.

CCS CONCEPTS

• **Human-centered computing** → **Visualization systems and tools**; • **Statistical paradigms** → Exploratory data analysis;

KEYWORDS

Data exploration; data visualization; statistical analysis; visualization recommendation; mixed-initiative interfaces

ACM Reference Format:
Kevin Hu, Diana Orghian, and César Hidalgo. 2018. DIVE: A Mixed-Initiative System Supporting Integrated Data Exploration Workflows. In *HILDA'18: Workshop on Human-In-the-Loop Data Analytics, June 10, 2018, Houston, TX, USA.* ACM, New York, NY, USA, Article 4, 7 pages. https://doi.org/10.1145/3209900.3209910

HILDA'18, June 10, 2018, Houston, TX, USA

ACM ISBN 978-1-4503-5827-9/18/06.
https://doi.org/10.1145/3209900.3209910

1 INTRODUCTION

We live in a world of data. But existing tools for making sense of that data are labor-intensive due to their reliance on analysts manually specifying queries through code or clicks. To overcome these limitations, visualization recommender systems [17, 28, 36] are now being developed to lower the learning curve of working with data and facilitate broad exploration of the result space. Existing recommender systems, however, do not provide fine-grained control over results, which impedes an analyst's ability to create specific visualizations. This limitation gave rise to hybrid systems [37] with mixed-initiative interfaces [8] that support both broad and focused exploration by combining recommender systems with manual specification.

Yet, these mixed-initiative approaches only address isolated parts of an analyst's data exploration workflow. Ideal workflows involve the multiple stages of identifying aspects of a dataset that are relevant to questions of interest, bringing a diverse suite of analytical techniques to answer those questions, and communicating results to an audience [26]. In practice, data exploration is a non-linear and iterative process [7] often fragmented between multiple tools, even among advanced analysts [10]. Fragmented workflows incur tool and context switching costs, in addition to the learning costs of each individual tool.

Here, we introduce **DIVE**, a mixed-initiative system combining recommender systems with point-and-click manual specification to support state-of-the-art data model inference, visualization, statistical analysis, and storytelling capabilities. We contribute the design of a system that integrates the multiple stages of the data exploration pipeline, and the description of a system that extends the use of mixed-initiative approaches to data model inference and statistical analysis.

A publicly available version of DIVE is available at https://dive.media.mit.edu. A demo video is located at https://dive.media.mit.edu/video. To support further research, the DIVE codebase is available as open-source software under the MIT license at https://dive.media.mit.edu/frontend and https://dive.media.mit.edu/backend.

2 LITERATURE REVIEW

The *data model inference*, *visualization recommendation*, and *statistical analysis* components of DIVE are informed by prior work in these respective fields. The system design and implementation of DIVE build on lessons from previous *mixed-initiative systems* and *accessible data exploration systems*.

2.1 Data Model Detection

Commercial data tools frequently detect data models of user-uploaded datasets. **Alteryx** [3] includes an "Auto Field Tool" for field type detection. Microsoft **Power BI** [16] has an "autodetect" feature that supports both field type and inter-object relationship detection. Community Connectors in **Google Data Studio** [6] detect both field and semantic types. The **Trifacta Visual Profiler** [25] computes statistical properties and displays summary visualizations. In aggregate, these tools define the feature set of DIVE's data model detection component, which aims to detect semantic and scale types of fields, inter-object relationships, and compute statistical properties.

Data model detection is also provided by open-source software libraries. Examples include **Messytables** [12], a Python library for parsing tabular data; **DataTables** [22], a Javascript library for displaying HTML tables; **Datalib** [29], a JS data utility library; and **Profiler** [11], a JS library for assessing data quality. These libraries detect data types using heuristic approaches based on regular expressions, type casting, set membership, range constraints, or some combination of these techniques. DIVE employs a heuristic-based approach inspired by these systems.

2.2 Visualization Recommender Systems

The "generate and test" paradigm of visualization recommendation originated with **APT** [13]. APT is built on a composition algebra over a set of visual primitives to enumerate visualizations, and scored visualizations using design criteria. Similar encoding recommenders like **Tableau "Show Me" features** [14] and **Spotfire Recommendations** [9] also use heuristics to recommend mark encodings. This prior work informs the enumerate-score recommendation pipeline used by DIVE.

Several systems recommend galleries of charts in response to user uploaded data. **Autovis** [34] provides a set of predefined visualizations of varying abstraction based on the type of uploaded dataset. **Zenvisage** [21] also presents charts in response to visual queries. **Scagnostics**[33] visualizes pairs of fields as scatterplots in a matrix. While recommendations in DIVE are also organized as in a gallery, they include non-selected fields.

Another body of work facilitates visual search by characterizing visualizations by statistical properties. The **Rank-by-Feature Framework** [20] enumerates visualizations involving one or two fields, then scores the visualizations based on 1-D and 2-D statistical measures. **SeeDB** [28] uses a distance function between distributions to compute the *utility* of visualizations. The recommendation system described in this paper uses similar statistical measures to characterize visualizations, but supports more data types and recommends a wider variety of visualizations.

2.3 Mixed-Initiative Visualization Systems

Mixed-initiative visualization systems incorporate user interaction to inform recommender systems. **VizDeck** [17] presents users with a ranked list of 1-D and 2-D visualizations, which a user can vote up or down. VizDeck incorporates user votes to update visualization ranks. **Small Multiples, Large Singles** [27] presents a main visualization and a grid of small multiples that are variants of the main visualization. Users explore recommendations by specifying data queries or visual encodings. **Keshif** [38] enables users to create interactive web-based dashboards comprised of linked visualizations. Keshif automates visual encoding choices using heuristics informed by data type, but is still largely based on manual specification.

Voyager [36] and **Voyager 2** [37], and the associated Compass recommender engine [35], recommend visualizations involving user-selected fields and one non-selected field. **Explore in Google Sheets** [5, 31] provides similar recommendations that "look ahead" by one field. DIVE is heavily inspired by Voyager 2, and aims to extend its mixed-initiative visualization approach to other parts of the data exploration pipeline. That said, DIVE also extends this work by including more non-selected fields in recommendations, incorporating semantic types into recommendations, and introducing a distinction between exact, subset, and expanded recommendations.

2.4 Statistical Analysis Systems

Existing tools for statistical analysis are based on manual specification. Programming languages like **R** and **Julia** can be used in an interpreter, while other tool-specific languages are used in applications with visual interfaces like **SPSS**, **SAS**, and **Stata**. While powerful, these languages and tools have steep learning curves. Visual tools like **Wizard** and **Statwing** provide point-and-click interfaces to conduct simple statistical analyses. DIVE builds on these visual point-and-click statistical tools with recommender systems and default field selection.

2.5 Accessible Data Exploration Systems

There is a rich history of desktop and web applications aiming to make data exploration more accessible. **Tableau**, formerly Polaris [24], allows users to drag-and-drop data fields into "shelves" in order to create visualizations. The business intelligence tools **Spotfire** [1] and **Qlik Sense** [19], among others, provide similar drag-and-drop interfaces to construct dashboards consisting of multiple visualization types. While these tools are widely adopted, most still rely heavily on manual specification.

The web applications **IBM ManyEyes** [30], **Raw Graphs** [15], and **Plot.ly Chart Studio** [18] are web-based tools that allow users to create pre-defined types of visualizations from uploaded data. Unlike these applications, DIVE supplements visualization with multivariate statistical analysis, and incorporates a mixed-initiative approach to both.

3 USAGE SCENARIO

We start by describing the example use case of an analyst using DIVE to investigate the factors influencing faculty salary in a hypothetical university. The analyst uses an eight column dataset containing demographic information (`name`, `gender`, `department`, `position`, `years of experience`), measures of performance (`number`

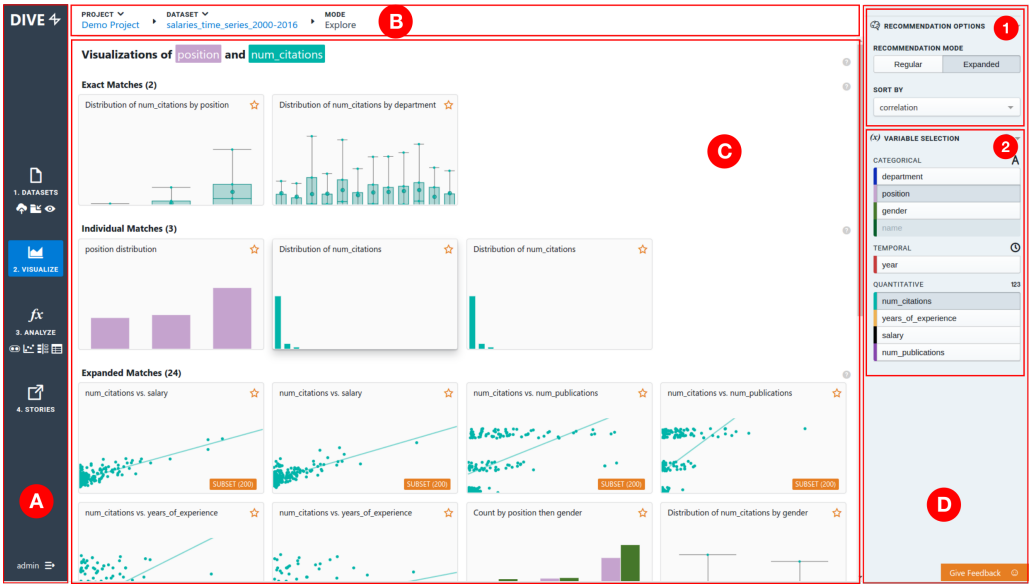

Figure 2: *The DIVE user interface. The left navigation bar labeled (A) is used to navigate between modes and modify project properties. The top navigation bar labeled (B) marks the current user state (project, dataset, and mode) and lets users switch projects or datasets. The main area labeled (C) displays the main results of the mode. The right selection menu (D) lets users change mode-specific parameters (D1) or selecting fields (D2). All other modes in DIVE follow this hierarchical four-section layout, though some modes do not include a selection menu (D).*

of publications, number of citations), and salary of 1000 faculty members.

She begins by creating a project, after which she is navigated to the **Upload** task associated with her project. Upon successful upload, the **Inspect** task presents her with the inferred data model describing her dataset. For example, name is correctly detected as a string acting as a unique identifier. Here, she can also manipulate the data model by either changing the types of fields, or marking fields as IDs.

Upon clicking the **Visualize** button on the left navigation bar shown in Figure 2-A, she is presented with a set of univariate summary visualizations. Because she is interested in salaries, she selects the salary field on the right hand menu, as seen in Figure 2-D. DIVE presents her with visualizations associated with the salary field, such as a scatter plot showing a positive relationship between years of experience and salary.

The analyst can go beyond visualizations by using the **Analysis** functionality, which supports four types of statistical analysis. First, she enters the **Correlation** task, which automatically generates a correlation matrix indicating a positive relationship between salary and number of citations and number of publications. She proceeds to the **Regression** task, selects salary as her dependent variable, and clicks "Recommend Model." DIVE recommends a set of models indicating not only the contribution of measures of accomplishment to salary, but also the contribution of demographic factors like gender.

Finally, on the right hand panel of the **Compose** task, she is able to view thumbnails of the visualizations and analyses she previously saved. By clicking on the thumbnails, she is able to add results to her linear narrative. She can publicly distribute this narrative by sharing an interactive web page tied to the current state of her dataset.

4 DESIGN CONSIDERATIONS

In this section, we propose five design considerations to guide the design of mixed-initiative systems and multi-stage data exploration workflows.

4.1 Discretize Workflow into Tasks Grouped by Ordered Modes

Idealized data exploration workflows consist of a sequential progression of stages flowing from data upload to presentation of results. In practice, analyst workflows are non-linear and iterative [10]. Furthermore, each stage may consist of multiple discrete but related tasks. *By grouping tasks into ordered modes, we can support non-linear workflows while encouraging a natural progression from data to presentation.*

DIVE is organized into four *modes*, which can be thought of as self-contained, but linked, applications. For example, the **Datasets** mode contains all functionality for uploading, inspecting, and transforming datasets. Each mode contains multiple *tasks*, such as building aggregation tables. The current scheme of modes and tasks is shown in Figure 3. Some modes are linked, like the transition from **Upload** to **Inspect** after successful ingestion of an uploaded dataset.

4.2 Hierarchically Distinguish Between Navigation, Configuration, and Results

User interface elements either enable users to navigate between tasks and modes, configure inputs to tasks, or view results of a task. Note that these three groupings of elements are dependent, such that tasks determine valid inputs, which then determine results. *Hierarchical visual layouts minimize visual overload while using a uniform visual language across tasks .*

The DIVE interface is organized into four sections, as shown in Figure 2. The left navigation bar, shown in every project, (Figure 2-A) provides controls for users to navigate between tasks. The top

Figure 3: *Map of the four user modes and associated tasks in DIVE. Tasks are connected by an arrow if they are connected by an action. A complete, linear use case of DIVE would begin on the top-left [1. Datasets > Upload] task and progress downwards through modes, towards the bottom-center [4. Stories > Story] task.*

state bar (Figure 2-B) shows the user's current project, dataset, and task, and lets users switch between projects and datasets. The task-specific selection menu (Figure 2-D) aggregates selectors used to specify or modify inputs to that mode. Results are shown in the main center area (Figure 2-C). The results shown in the center are are uniquely specified by the state of the selection menu.

4.3 Organize Input Components into Tightly-Coupled Hierarchies

Within a task, users work with their data by specifying task configuration as input. *Because task configuration types are inter-dependent, associated interface elements should be hierarchically organized and tightly-coupled [2].* In DIVE, task input is specified by five types of configuration. Users can specify which model to use, the parameters of a given model, the data used by the model, how to display results, and filter down results based on conditional selectors.

4.4 Combine Populated Defaults with Incremental Selection

When a user navigates to a new task without pre-specified configuration, they should be presented results by default, which can then be incrementally modified. Populated defaults encourage users to engage with their datasets by bypassing the cold-start problem. For example, on the **Explore** page, users are shown descriptive visualizations for each field in their selected dataset. On the **Regression** page, users are shown a set of simple linear regressions involving a random dependent variable and independent variables selected through the LASSO method.

This default result set is modified using the selection menu (Figure 2-D), which lets users change the parameters relevant to that specific task (Figure 2-D1) or the field selection (Figure 2-D2). As users incrementally select fields, the main view updates to reflect the current state, as shown in Figure 4.

4.5 Distinguish Recommendation Types

As shown in Figure 4-B, there is a visual separation between different kinds of results: exact, subset, individual, and expanded. *This allows users to both understand how a recommendation is created and groups results to avoid overwhelming users with recommendations.* Additionally, this separation lets users adjust the number of returned results by toggling certain types of recommendations.

5 SYSTEM DESCRIPTION

5.1 Datasets: Upload and Inspect

Meaningful data exploration requires an accurate and relevant data model. For each dataset, DIVE assigned a data model that is comprised of three components: *field properties* such as ID, contiguity, name, semantic type, scale type, statistical properties, and unique values; *inter-field relationships*, including hierarchical relationships between fields; and *inter-object relationships*, like the existence and cardinality of one-to-one, one-to-many, and many-to-many relations. We assume that uploaded data is tabular and *tidy* [32], such that each dataset represents an object, with columns representing attributes of that object and rows representing instances of that object.

Field Types Definitions. Following the example of Stevens [23], we distinguish between three general scale types, *nominal*, *ordinal*, and *continuous*, each of which permit specific mathematical transformations and operations. However, similar to Google Data Studio [6], we also distinguish between three general semantic types, *categorical*, *temporal*, and *quantitative*, which inform the families of valid analyses. Each general data type is divided into more specific types, forming a taxonomy of scale and data types.

Figure 4: *Recommended visualizations in the Explore mode. (A) shows the default view when a user first navigates to Explore, while (B) shows recommendations when the fields* `position`, `year`, `salary` *are selected. The sections marked (B1), (B2), (B3), and (B4) contain exact, subset, individual, and expanded matches, respectively.*

Field Type Detection. DIVE employs a heuristic-based approach for detecting semantic types, considering both the name of the field and its values. Regular expression matches of names involve comparing the field name against a list of matches and their associated scores. Some semantic types, like `datetime`, involve matching against a set of regular expressions. Others, like `country`, are tested by comparing field values against a list of fixed set of instances. This rank-ordering of types is used to assign confidence scores and suggest field type updates.

Field Property Detection. With this detected field type, we can determine the statistical properties of the field. For categorical fields, we determine the number of unique values, and the frequency of each value. For ordinal and continuous fields, we calculate summary statistics. For all data types, we calculate the number of null values, uniqueness, and assign a likelihood as to whether a field is an identifier.

5.2 Visualize: Explore and Drill-down

Visualization recommendation in DIVE is a two-stage process starting with *enumeration* of visualization specifications, then *scoring* of visualizations. Given a user data model $\boldsymbol{D}$, user selection $\boldsymbol{S} = \{s_i\}$, and un-selected fields $\boldsymbol{U} = \{u_j\} = \boldsymbol{D} \setminus \boldsymbol{S}$, the DIVE recommendation system iteratively constructs different sets of *considered fields*, denoted as $\mathbf{C}$. By default, if the user does not select any fields, DIVE returns univariate descriptive visualizations of each field.

Enumeration of Visualization Specifications. We define a *visualization specification* as a statement defining an exact and unique mapping from data to a visualization. Our enumeration approach begins with functions mapping from input fields to specifications, following an approach similar to that of Bertin [4].

Exact matches ($\boldsymbol{S} = \boldsymbol{C}$) involve all user selected fields. *Subset matches* ($\boldsymbol{S} \supset \boldsymbol{C}$) consider a subset of user selected fields, as shown in Figure 4-B2. A special case of *subset matches* is *individual matches*, which consider only single user selected fields in $\boldsymbol{S}$. Figure 4-B3 shows univariate descriptive visualizations of the three selected fields. *Expanded matches* ($\boldsymbol{S} \subset \boldsymbol{C}$) involve at least one user selected field s_i and at least one un-selected field u_j, following the approach of Voyager [36] and Google Sheets Explore [5, 31].

Visualization Scoring. For each visualization, we first compute the *relevance* score $R = |\boldsymbol{S} \cap \boldsymbol{C}|/|\boldsymbol{C}|$, marking the number of user-selected fields included in the visualizations. We also compute statistical properties of the visualizations [20], such as entropy, normality, in addition to standard descriptive statistics (min, max, mode, average) for 1-D visualizations. For 2-D visualizations, we compute the correlation and the coefficients of a linear fit. For all, we attach the number of visual elements and null values.

5.3 Analysis: Aggregation, Correlation, Comparison, and Regression

DIVE supports four common statistical analysis tasks. The **Aggregation** task lets users create 1-D and 2-D aggregation tables displaying the count, mean, or sum of elements in a group. **Correlation** lets users create correlation matrices between pairs of quantitative fields. **Comparison** lets users compare means of groups using one-way or two-way ANOVA. By default, on these three pages, analyses are conducted on randomly selected valid fields or previously selected user fields. **Regression** supports simple linear or logistic regressions. Users can also introduce interaction terms or transform independent variables by taking the log or square. By default, independent variables are chosen by a forward selection algorithm. On all pages, results can be saved or shared.

5.4 Stories: Compose and Share

On the **Compose** page, users can assemble blocks containing saved visualizations, statistical analysis results, and text entries into a linear story. Each block can also be annotated with a title or description. These interactive stories can be shared with a public URL.

6 ARCHITECTURE AND IMPLEMENTATION

DIVE is implemented as a web application with a thin client front end connected to API endpoints exposed by a worker-server back end. The front end uses the React web framework with Redux to manage application state, Google Charts as a visualization library, and Palantir Blueprint as a user interface framework. The back end consists of a RESTful API using Flask as a web server, a PostgreSQL database for persistence, and Celery on RabbitMQ as an asynchronous task queue. Data manipulation and statistical analysis are performed using the standard Python scientific computing libraries: pandas, NumPy, SciPy, and StatsModels libraries.

7 EVALUATION: DIVE VERSUS EXCEL

We conducted a user study in which we asked users to create visualizations and conduct analyses with either DIVE or Excel, and then compared task performance between these two groups. While Excel is not state-of-the-art in terms of its functionality, it is still one of the most commonly used general tools. Additionally, other tools would most likely require previous training. In our study, we used a cold-start setting, meaning that participants were not trained in DIVE prior to the evaluation.

We recruited 67 data analysts from two large consulting firms in the US to participate in the evaluation. All participants had previous exposure to Excel, with an average of 11 years of experience. The group had a strong technical background, having had taken, on average, 3.94 courses in statistics and 3.49 courses in computer science. The average age of our sample was 33.57 years old (σ^2 = 7.30 years). There were 12 women in the sample.

Participants were randomly assigned to the DIVE (33 participants) or Excel condition (34 participants). Subsequently, participants were given a synthetic dataset of faculty salaries from a hypothetical university, which was structurally identical to the dataset described in the **Usage Scenario** section. Each of the 1000 rows corresponds to one faculty member, and the 8 columns correspond to attributes of the faculty members.

Then, participants were asked to complete a set of tasks. In the **Visualization Section**, they were asked to create a scatter plot of salaries versus number of publications, a bar chart of the number of people by department, and a bar chart of the average wage by gender. In the **Analysis Section**, participants were asked to answer questions concerning inferential statistics and to paste the screenshots of the evidence that led to their conclusions. The questions were: "Is the difference between the average wage of males and females statistically significant?"; "Is the difference between the average wage of males and females statistically significant after controlling for number of citations, publications, department, and years of experience?"; and "Did the difference between the average wage of males and females increase or decrease after controlling for number of citations, publications, department, and years of experience?"

7.1 Quantitative Evaluation Results

Participants using DIVE were significantly more successful in creating visualizations ($\mu = 0.89$, $\sigma^2 = 0.18$) than the Excel users ($\mu = 0.77$, $\sigma^2 = 0.29$), with $t(65) = 2.10, p = 0.04$. DIVE users were also much faster ($\mu = 123.50s$, $\sigma^2 = 35.53s$) than those using Excel ($\mu = 168.74s$, $\sigma^2 = 63.59s$) at completing the same tasks, with $t(64) = 3.60, p = 0.001$.

Participants using DIVE were significantly more successful in conducting the specified statistical analysis tasks. We looked at the number of people that did not reach an answer to the questions (did not answer or selected the `I cannot answer this question` option). 73.53% of DIVE users were able to reach an answer, while only 52.52% in the group using Excel were able to answer the questions.

In sum, the evaluation results indicate that participants using DIVE were more successful and faster than those using Excel in completing the same data exploration tasks. Because the two groups were balanced in terms of experience and proficiency with data exploration tasks, these results suggest that that DIVE supports the completion of simple, but commonly used, visualization and analysis tasks.

7.2 Qualitative Participant Feedback

At the end of the session, participants in the DIVE group were asked to give feedback about the tool. The first three questions, and associated responses that were reported on a 5-point Likert scale, were: **"Was DIVE easy to learn?"** ($\mu = 3.77$, $\sigma^2 = 0.31$); **"Would you use DIVE again?"** ($\mu = 3.27$, $\sigma^2 = 0.36$); **"If DIVE was available online, would you recommend it to others?"** ($\mu = 3.53$, $\sigma^2 = 0.32$).

Free-form text responses to the remaining questions reinforce the quantitative results. In response to **"What features of DIVE did you like most?"** many participants noted the ease of use: *"How easy it was to visualize information across different charts"* and *"Intuitive to learn with some stats and data analysis background"*. Some commented specifically on the visualization recommendation: *"Proactive graph proposal"* and *"Automatic analysis when choosing variables"*. Others appreciated the integrated workflow: *"integrated place for many tasks"* and *"The ubiquity of visualization as part of the analysis."*

In response to the question **"What should we improve about DIVE?"** participants had a diverse set of responses, centered around clarity: *"The flow: where to begin, what is used for what."*; *"Felt that the explanation boxes for logistic, linear, anova were somewhat confusin to interpret"*; and *"Why to we need to have another data vizualization tool ?"* Some of these concerns would have been addressed with an introduction to the tool. Answers to the final question **"What additional features would you like to see in DIVE?"** were also diverse: *"integrated decision-support/help"*; *"customize graphs to show different things"*; and *"More machine learning algorithms."*

8 CONCLUSIONS AND FUTURE WORK

We introduce DIVE, a mixed-initiative data exploration tool developed to lower the learning curve to working with data. We described the heuristic-based approaches to data model inference and visualization recommendation system central to the construction of DIVE. We also described considerations taken into account in the interface, interaction, and system design of workflows that integrate the multiple stages of data exploration.

We plan to extend DIVE by implementing tasks for simple machine learning, clustering, and time series analysis. Several beta users also requested more flexibility in data input, such as integration with web and database APIs. Other future work includes developing precautions to ensure valid statistical analysis, recommending field or view transformations, and more tightly integrating visualizations and analyses.

Other systems-level questions remain. While DIVE is intentionally a domain-agnostic system, domain-specific approaches could lead to more useful data exploration tools. Many current systems also lack any modelling of user preferences of behavior, both of which can inform recommendation. Lastly, few recent systems have addressed the social dimension of visualization, which is increasingly relevant for collaborative systems.

REFERENCES

[1] Christopher Ahlberg. 1996. Spotfire: An Information Exploration Environment. *SIGMOD Rec.* 25, 4 (Dec. 1996), 25–29. https://doi.org/10.1145/245882.245893

[2] Christopher Ahlberg and Ben Shneiderman. 1994. Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '94)*. ACM, New York, NY, USA, 313–317. https://doi.org/10.1145/191666.191775

[3] Alteryx. 2010. Alteryx. (2010). https://alteryx.com

[4] Jacques Bertin. 1983. *Semiology of Graphics*. University of Wisconsin Press.

[5] Google. 2015. Explore in Google Sheets. (2015). https://www.youtube.com/watch?v=9TiXR5wwqPs

[6] Google. 2016. Google Data Studio. (2016). https://datastudio.google.com

[7] Jeffrey Heer and Ben Shneiderman. 2012. Interactive Dynamics for Visual Analysis. *Commun. ACM* 55, 4 (April 2012), 45–54. https://doi.org/10.1145/2133806.2133821

[8] Eric Horvitz. 1999. Principles of Mixed-initiative User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. ACM, New York, NY, USA, 159–166. https://doi.org/10.1145/302979.303030

[9] Tibco Software Inc. 2015. Spotfire Recommendations. (2015). https://spotfire.tibco.com/products/data-discovery-and-visualization

[10] Sean Kandel, Andreas Paepcke, Joseph M. Hellerstein, and Jeffrey Heer. 2012. Enterprise Data Analysis and Visualization: An Interview Study. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (Dec. 2012), 2917–2926. https://doi.org/10.1109/TVCG.2012.219

[11] Sean Kandel, Ravi Parikh, Andreas Paepcke, Joseph M. Hellerstein, and Jeffrey Heer. 2012. Profiler: Integrated Statistical Analysis and Visualization for Data Quality Assessment. In *Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI '12)*. ACM, New York, NY, USA, 547–554. https://doi.org/10.1145/2254556.2254659

[12] Open Knowledge Labs. 2013. messytables. (2013). https://github.com/okfn/messytables

[13] Jock Mackinlay. 1986. Automating the Design of Graphical Presentations of Relational Information. *ACM Trans. Graphics* 5, 2 (1986), 110–141.

[14] Jock Mackinlay, Pat Hanrahan, and Chris Stolte. 2007. Show Me: Automatic Presentation for Visual Analysis. *IEEE Transactions on Visualization and Computer Graphics* 13, 6 (Nov. 2007), 1137–1144. https://doi.org/10.1109/TVCG.2007.70594

[15] Michele Mauri, Tommaso Elli, Giorgio Caviglia, Giorgio Uboldi, and Matteo Azzi. 2017. RAWGraphs: A Visualisation Platform to Create Open Outputs. In *Proceedings of the 12th Biannual Conference on Italian SIGCHI Chapter (CHItaly '17)*. ACM, New York, NY, USA, Article 28, 5 pages. https://doi.org/10.1145/3125571.3125585

[16] Microsoft. 2015. Power BI. (2015). https://powerbi.microsoft.com

[17] Daniel Perry, Bill Howe, Alicia M.F. Key, and Cecilia Aragon. 2013. VizDeck: Streamlining exploratory visual analytics of scientific data. In *iConference*.

[18] Plotly. 2017. Plot.ly Chart Studio. (2017). https://plot.ly/online-chart-maker/

[19] Qlik. 2017. Qlik Sense. (2017). http://www.qlik.com/us/products/qlik-sense

[20] Jinwook Seo and Ben Shneiderman. 2005. A Rank-by-Feature Framework for Interactive Exploration of Multidimensional Data. 4 (2005), 96–113. Issue 2.

[21] Tarique Siddiqui, Albert Kim, John Lee, Karrie Karahalios, and Aditya G. Parameswaran. 2016. zenvisage: Effortless Visual Data Exploration. *CoRR* abs/1604.03583 (2016). http://arxiv.org/abs/1604.03583

[22] SpryMedia. 2007. DataTables. (2007). https://datatables.net

[23] Stanley Smith Stevens. 1946. On the Theory of Scales of Measurement. *Science, New Series* 103, 2684 (1946), 677–680.

[24] Chris Stolte, Diane Tang, and Pat Hanrahan. 2008. Polaris: a system for query, analysis, and visualization of multidimensional databases. *Commun. ACM* 51, 11 (2008), 75–84. https://doi.org/10.1145/1400214.1400234

[25] Trifacta. 2014. Visual Profiler. (2014). https://docs.trifacta.com/display/PE/Profiling+Basics

[26] John W. Tukey. 1977. *Exploratory Data Analysis*. Addison-Wesley Publishing Company. https://books.google.com/books?id=UT9dAAAAIAAJ

[27] Stef van den Elzen and Jarke J. van Wijk. 2013. Small Multiples, Large Singles: A New Approach for Visual Data Exploration. In *Proceedings of the 15th Eurographics Conference on Visualization (EuroVis '13)*. The Eurographs Association John Wiley Sons, Ltd., Chichester, UK, 191–200. https://doi.org/10.1111/cgf.12106

[28] Manasi Vartak, Samuel Madden, Aditya Parameswaran, and Neoklis Polyzotis. 2014. SeeDB: Automatically Generating Query Visualizations. *Proceedings of the VLDB Endowment* 7, 13 (2014), 1581–1584.

[29] Vega. 2017. datalib. (2017). http://vega.github.io/datalib/

[30] Fernanda B. Viegas, Martin Wattenberg, Frank van Ham, Jesse Kriss, and Matt McKeon. 2007. ManyEyes: A Site for Visualization at Internet Scale. *IEEE Transactions on Visualization and Computer Graphics* 13, 6 (Nov. 2007), 1121–1128. https://doi.org/10.1109/TVCG.2007.70577

[31] Fernanda Viégas, Martin Wattenberg, Daniel Smilkov, James Wexler, and Daniel Gundrum. 2018. Generating charts from data in a data table. US 20180088753 A1. (2018).

[32] Hadley Wickham. 2014. Tidy data. *The Journal of Statistical Software* 59 (2014). Issue 10. http://www.jstatsoft.org/v59/i10/

[33] Leland Wilkinson, Anushka Anand, and Robert Grossman. 2005. Graph-Theoretic Scagnostics. In *Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization (INFOVIS '05)*. IEEE Computer Society, Washington, DC, USA, 21–. https://doi.org/10.1109/INFOVIS.2005.14

[34] Graham Wills and Leland Wilkinson. 2010. AutoVis: Automatic Visualization. *Information Visualization* 9 (2010), 47–6927. https://doi.org/10.1057/ivs.2008.27

[35] Kanit Wongsuphasawat, Dominik Moritz, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. 2016. Towards A General-Purpose Query Language for Visualization Recommendation. In *ACM SIGMOD Human-in-the-Loop Data Analysis (HILDA)*. http://idl.cs.washington.edu/papers/compassql

[36] Kanit Wongsuphasawat, Dominik Moritz, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. 2016. Voyager: Exploratory Analysis via Faceted Browsing of Visualization Recommendations. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)* (2016). http://idl.cs.washington.edu/papers/voyager

[37] Kanit Wongsuphasawat, Zening Qu, Dominik Moritz, Riley Chang, Felix Ouk, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. 2017. Voyager 2: Augmenting Visual Analysis with Partial View Specifications. In *ACM Human Factors in Computing Systems (CHI)*. http://idl.cs.washington.edu/papers/voyager2

[38] Mehmet Adil Yalcin, Niklas Elmqvist, and Benjamin B. Bederson. 2017. Keshif: Rapid and Expressive Tabular Data Exploration for Novices. *IEEE Transactions on Visualization and Computer Graphics* PP, 99 (2017), 1–14. https://doi.org/10.1109/TVCG.2017.2723393