Keshif: Out-of-the-Box Visual and Interactive Data Exploration Environment

Mehmet Adil Yalçın[§], Niklas Elmqvist[°], and Benjamin B. Bederson[†]

University of Maryland, College Park

Abstract

Keshif is an open-source, web-based data exploration environment that enables data analytics novices to create effective visual and interactive dashboards and explore relations with minimal learning time, and data analytics experts to explore tabular data in multiple perspectives rapidly with minimal setup time. In this paper, we present a high-level overview of the exploratory features and design characteristics of Keshif, as well as its API and a selection of its implementation specifics. We conclude with a discussion of its use as an open-source project.

Keywords: Interactive data exploration, data visualization, data exploration, interaction, design, user-centered design.

Index Terms: H.5.2 [Information Interfaces and Presentation]: User Interfaces—Graphical User Interfaces (GUI).

1 Goals and Background

Creating effective data interfaces requires significant skills in visualization, interaction, and interface design, as well training and time to plan and execute actions that implement good strategies. Highly flexible and configurable tools, such as Tableau or Lyra [1] provide a vast design space that can be hard to learn, navigate, and use effectively. We built Keshif [‡] (www.keshif.me) as a data exploration space designed from ground-up to automatically convert data to an effective visual interface that can be interactively explored to reveal overviews, trends, and relations within a tabular dataset. Using Keshif, analytics and design novices benefit from the best practices with easy learning on its minimal and familiar basis. On the other hand, experts benefit from its rapid interaction, expressive alternative modes for visualizations and selections, and minimal setup time to bootstrap their data analysis, as well as its flexibility based on standards of the web (JavaScript, HTML, CSS) to transform data and customize styling.

Keshif is open-sourced at github.com/adilyalcin/keshif with BSD 3-clause license, first released on late 2013. By continuous improvements in its design and features, by its application to 160+ datasets across different domains, and through comparisons to existing practices and research, Keshif has grown to embody the best practices. It also has become a robust platform to explore new visualization techniques, such as AggreSet [3] which focuses on set-typed (multi-value categorical) data exploration.

2 Exploratory Features of Keshif

Keshif is designed to enable exploration of tabular datasets (records with a range of attributes). A Keshif data browser (Figure 1) converts selected attributes into visual summaries, where records are grouped into aggregates. Characteristics of these aggregates (such as count, total, average) are measured and visualized (Figure 3). Each data type has a specialized visualization design (Figure 4), extending upon the best practices to achieve perceptual effectiveness. Two visual scale modes support alternative analysis: measurements can be shown in either absolute or part-of mode (Figure 2). Each record can also be displayed individually, either as a sorted list, as a network (using an explicit link attribute), or as a map (using a unique geo-region attribute). As a result, the user is freed from data encoding decisions, and can focus on making queries, and observing trends and relations with minimal training.

The interactions in Keshif are designed to achieve seamless and automatic synchronization across summaries and the record display. To increase expressiveness in exploration, Keshif enables selection for filtering (by mouse-click), highlighting (by mouse-over), and comparison (by locking) of record groups, (Figure 1). Upon interaction, visualizations are refreshed with appropriate animations to guide through the flow of changes.

Keshif browser can be authored and configured using its graphical interface by drag-and-drop (Figure 5), or its API (Section 4). Authoring and exploration workflow is demonstrated by a 5-minute video. By defining a separate authoring mode (which allows adding/removing summaries, formulating custom attributes, and adjusting layout), Keshif achieves minimalism and efficiency for its primary use case: exploring the data features rapidly.

3 Design Approach of Keshif

The design of our tools is critical as it influences the paths taken in exploration, and effectively the outcomes. We designed Keshif to declutter the human-facing data search space by reducing configurations and improving out-of-the-box design for usability and learnability while maintaining high expressiveness. As John Maeda notes, "design is both the insanely radical, and the passionately incremental". Our radical approach is our strong emphasis in designing a minimalist data exploration space that avoids ineffective paths, enables explicit overview-to-detail flow, achieves high consistency in visual and interaction design, avoids multiple visual representations of a single data item, and prefers tight integration and efficiency over modularity. Our incremental approach is our extensions on the exploratory design space, such as compare-selection by locking, expressive visual scale modes, and synchronization from pointed records to aggregates as well as from aggregates to individual records and other summaries.

4 API for Keshif Browser Configuration

Keshif, including its API, is designed to let the user define *what* is being visualized and explored, not *how*. This is in contrast to visualization grammars such as Vega Lite [2] and ggplot, which have a compositional approach to create a range of chart designs. It also contrasts with chart templating approaches such Excel, Raw, Datamatic, and Quadrigram, since Keshif automates the visualizations and interaction, and the data dialogue is driven by the user based on key exploratory tasks rather than selecting charts and

[§]adil@keshif.me , [°]elm@umd.edu , [†]bederson@umd.edu

[‡] Keshif (keşif) means exploration and discovery in Turkish.

Proc. of IEEE VIS 2016 Workshop on Visualization in Practice: Open Source Visualization and Visual Analytics Software. Copyright retained by the authors.

mapping data to template parameters. Keshif customizations are commonly aimed to express metadata, such as *ordinal* categories and unit names of numeric attributes, as well as basic data transformations such as parsing time components from a text field, and splitting a text field into multiple categories by a delimiter. We created the API for Keshif browsers to concisely support the common needs we identified on 150+ public datasets.

The JavaScript API of Keshif (Code 1) enables concise, flexible, customizable, and persistent configuration of data browsers. Its single entry-point is instantiation of a `kshf.Browser` object with a browser configuration, which describes the data source, the list of summaries (position, name, function, and other configurations such as sorting of categorical data or unit name for integer values), and the record display (including sorting options, record view, etc). Multiple browsers can be added to a single web-page by instantiating multiple `kshf.Browsers` (see VisTools). Code 1 demonstrates `function`al customizations for key objectives including loading custom data (such as GeoJSON of a country, an XML file, or even BibTeX entries for literature surveys), describing a data feature to summarize (such as extracting months from a Date attribute), and describing HTML components of how a component should be rendered (such as merging multiple attributes, with custom styling). While the visual and interaction design is tightly controlled and not aimed to be end-user configurable, these callbacks provide key flexibility so that Keshif can fit many data sources, domains, and settings. In addition, Keshif browser configuration can be serialized to/from JSON objects. To handle custom callback functions in a configuration, we convert these functions to strings on export, and evaluate functions as string definitions using JavaScript `eval` on configuration load. The full end-user API documentation is at github.com/adilyalcin/Keshif/wiki.

To save, host, load, and edit browser configurations easily as JSON objects, we implemented a GitHub Gist-based storage and authentication. Gist configuration are stored and loaded using unique IDs (for example, keshif.me/gist/?82d0d3caed8e93ea5ff8 loads the configuration at gist.github.com/82d0d3caed8e93ea5ff8. This allows easy version-control and forking of browser configurations. Our Gist integration also can manage custom CSS style files along with browser configuration.

5 Implementation Notes

Keshif is implemeted as a cross-platform tool based on modern web standards of JavaScript, HTML and CSS. As a strictly client-side tool, Keshif is a lightweight system that does not require a server installation or maintenance. Datasets can be loaded from cloud services that host spreadsheets (such as Google Sheets) or documents (such as CSV or JSON files on Google Drive or Dropbox), in addition to files hosted at a local server, or uploaded from local computer (non-persistent). Essentially, a Keshif browser can be built on any data resource that a web browser can access, and Keshif does not control data authentication and security protocols of the data sources. Keshif's client-side basis puts a practical limit on the data volume that can be loaded into the browser's memory, while it may support up to 220k+ records (See NYC bike-trips).

Our implementation emphasizes a lean, minimalist approach as well. To keep our development stack minimal and have full control the interface design, we opted not to use frameworks such as React and Angular, or even jQuery. The *only* core dependency of the current Keshif implementation is D3, which is used to bind custom data structures to page components, create visualizations, and update these components interactively. We implemented custom internal aggregated and cached computations, since Keshif support query models not supported by off-the-shelf tools like Crossfilter. The JavaScript code is developed and maintained under a single file, keshif.js. Keshif also uses Leaflet to render interactive maps, and PapaParse to parse CSV files. Keshif browser styling is implemented using less, a CSS preprocessor, which simplifies hierarchical styling and cross-browser compatibility. Our current unminified JavaScript implementation is over 11kLOC (460KB), and stylesheet is over 4kLOC (138KB). Keshif also uses FontAwesome, which provides a clean, consistent, and familiar icon design for actions and objects in its interface. We implemented most animations using CSS3 transitions instead of using d3.transition(), making it more concise, simpler to develop and maintain. We used CSS flexbox display model to implement flexible and responsive layout components. Since rendering records individually can degrade performance given large datases, we implemented an infinite scrolling strategy, creating page DOM elements conservatively and dynamically on scroll and filter.

6 Use of Keshif

We created Keshif data browsers for 160+ public datasets across many domains, including, but not limited to, journalism, open government, surveys, health, and arts. www.keshif.me and its various demos have been visited more than 100k times in the last year. keshif.me/demo/VisTools stands as the most visited and shared among our demos, giving a multi-faceted overview of 400+ visualization tools and 80 visualization books. At the time of this submission, Keshif GitHub repository has been ★starred 400+ times, and forked 100+ times. While the project is open source and we are open to code contributions, we have not yet received contributions from third parties, although we have received questions and comments on our issues page, our mailing list, or through personal communications. This may be due to our integrated design and implementation that does not prioritize modularity, customization, or extensive documentation of internal implementation. Project updates can be followed on social media at twitter.com/keshifme and facebook.com/keshifme.

7 Conclusion

We presented Keshif, a new out-of-the-box data exploration environment that automates the visual and interactive design to enable rapid and flexible tabular data exploration. Our main motivation is to improve the exploration space for data-driven insights by pruning complexities and ineffective choices. We briefly introduced the API for Keshif browser configurations, as well as some of our implementation strategies and components. We are looking forward to extending supported data types, graphical authoring capabilities, and making the design of Keshif easier to learn, while extending the application of Keshif to new domains and datasets.

References

[1] A. Satyanarayan and J. Heer, "Lyra: An Interactive Visualization Design Environment," *Computer Graphics Forum*, vol. 33, no. 3, pp. 351–360, Jun. 2014.

[2] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer, "Vega-Lite: A Grammar of Interactive Graphics," *IEEE Transactions on Visualization and Computer Graphics*, vol. PP, no. 99, pp. 1–1, 2016.

[3] M. A. Yalçın, N. Elmqvist, and B. B. Bederson, "AggreSet: Rich and Scalable Set Exploration using Visualizations of Element Aggregations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 688–697, Jan. 2016.

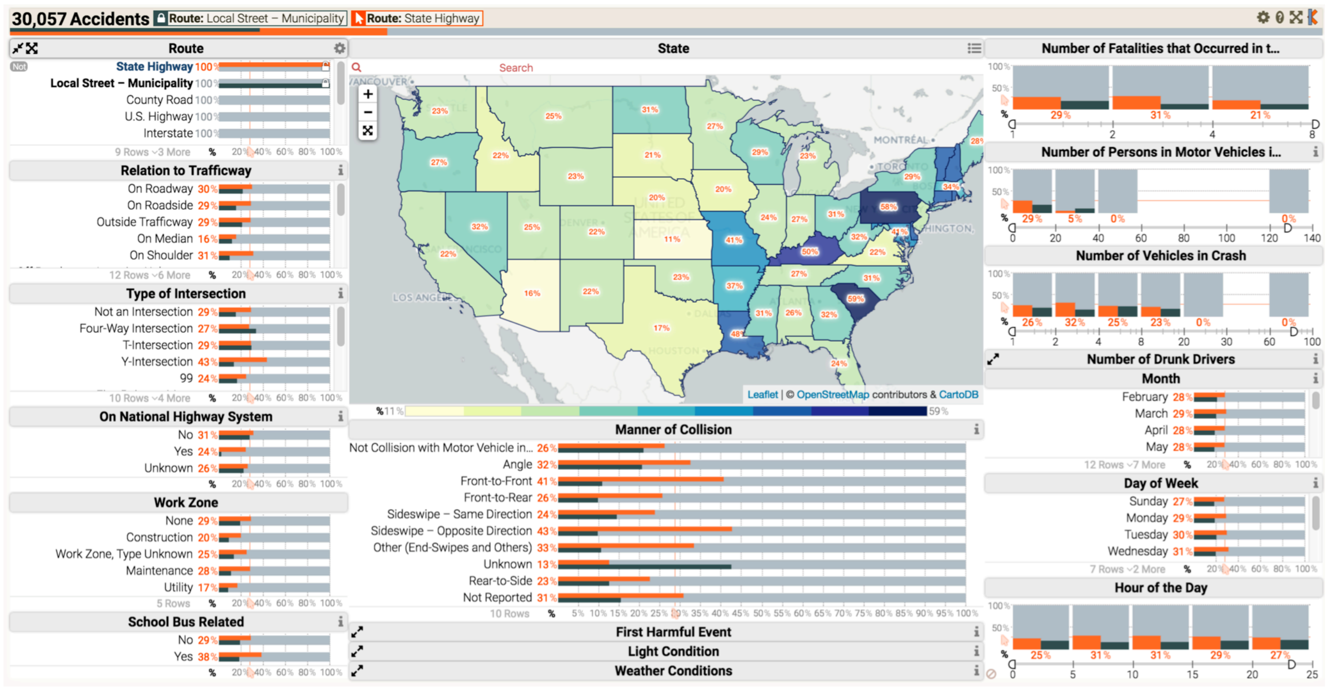
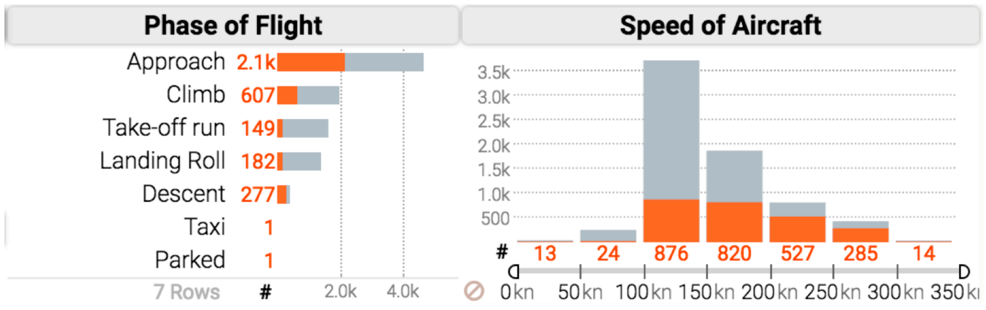
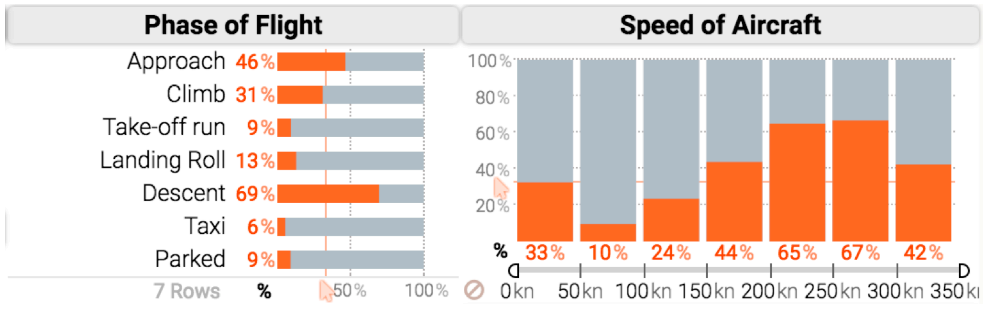


Figure 1. A screenshot from a Keshif browser that visualizes all 30,057 fatal traffic accidents in the United States that happened in 2013. The visualization scale mode is set to show part-of-whole relations. Black colors in the visualization show the locked (compared) selection 🔒 of accidents that happened on Local Street routes, while orange colors show the highlighted selections of State Highway accidents. This status is displayed as a bread-crumb pattern on top of the browser. All summaries use the same visual mode and language. This example is taken from www.keshif.me/demo/FatalTrafficAccidents2013.

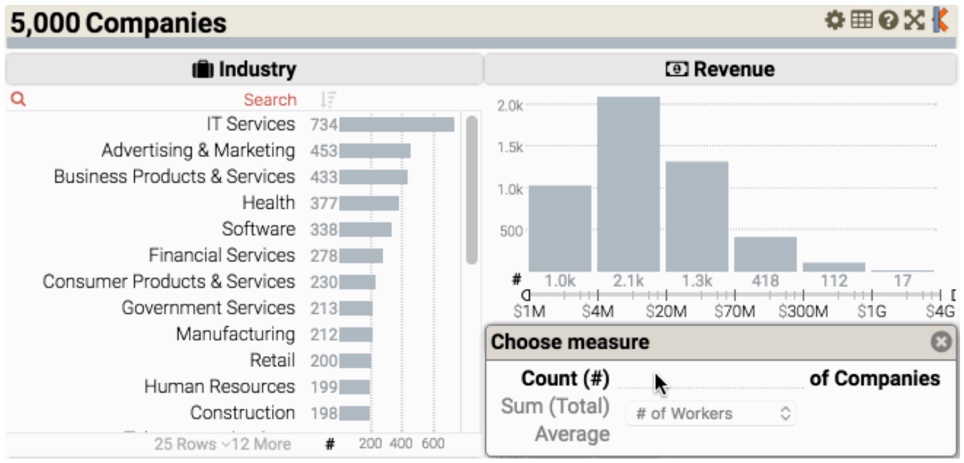


a) Absolute mode: Measurements are shown in absolute values.

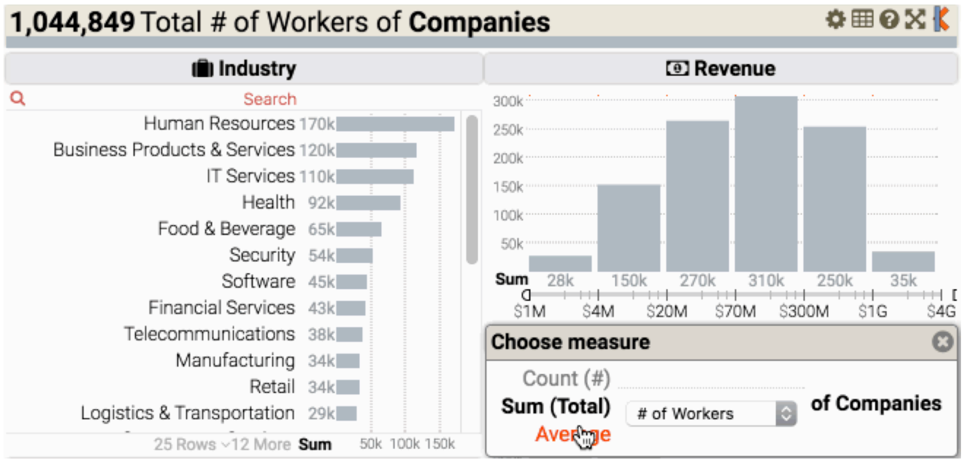


b) Part-of mode: Measurements are shown in percentages of the total (or filtered) values. The part-of mode is only valid (and applicable) for count or sum measure functions.

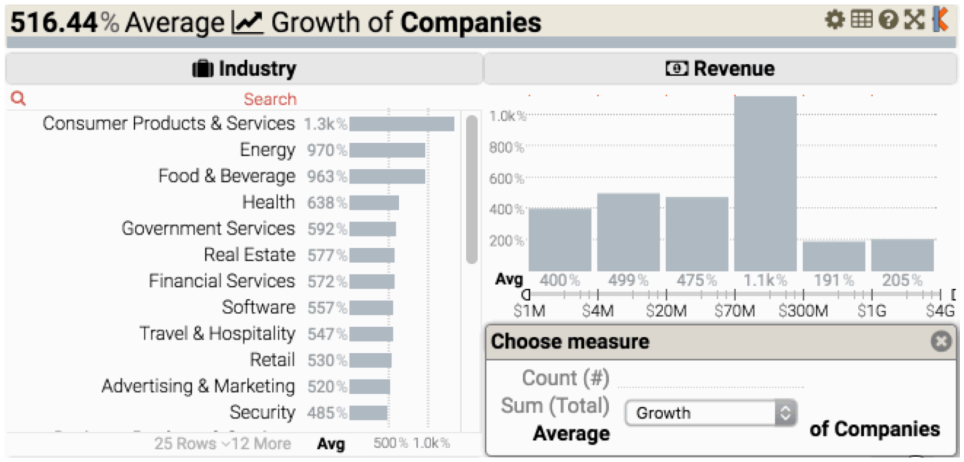
Figure 2. Alternative visual scale modes for aggregations, using an example from BirdFlights dataset. This example is taken from www.keshif.me/demo/BirdStrikes.



a) Each aggregate shows the number of companies.
This is the default mode as it provides a familiar and basic overview of the data distribution.

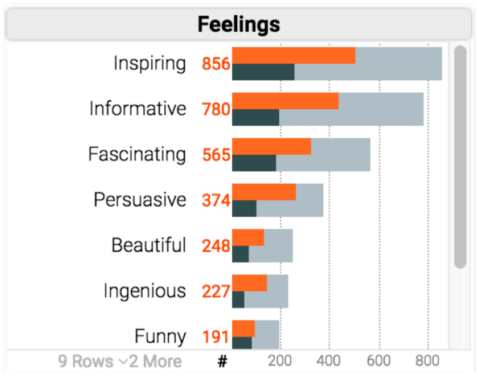


b) Each aggregate shows the total number of workers.
Example: There are 92k workers in companies in health industry.

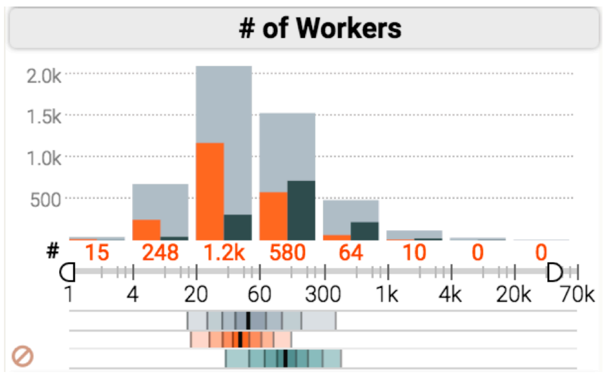


c) Each aggregate shows the average growth per each aggregation.
Example: Energy sector is the second highest-average-growth industry, with 970% growth in average across all energy industry companies.

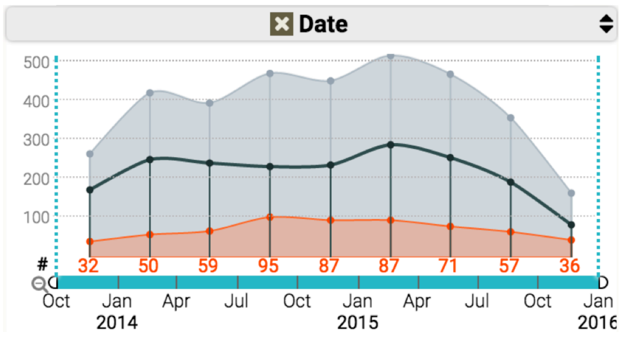
Figure 3. Alternative measurement functions for aggregations, using an example from Companies dataset (from www.inc.com)
This example is taken from www.keshif.me/demo/inc5000.



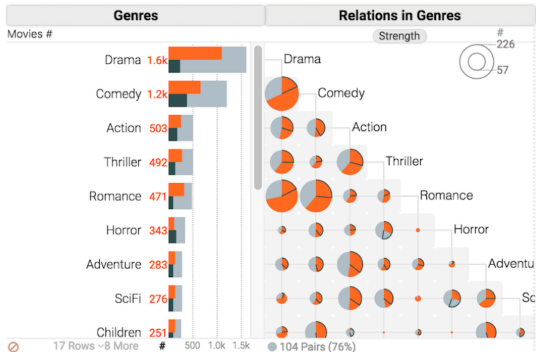
Categorical data → Vertical histogram



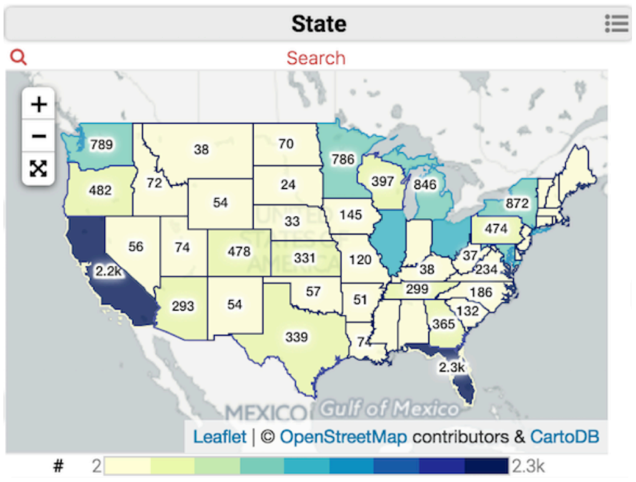
Numeric Data → Horizontal histogram and percentile chart



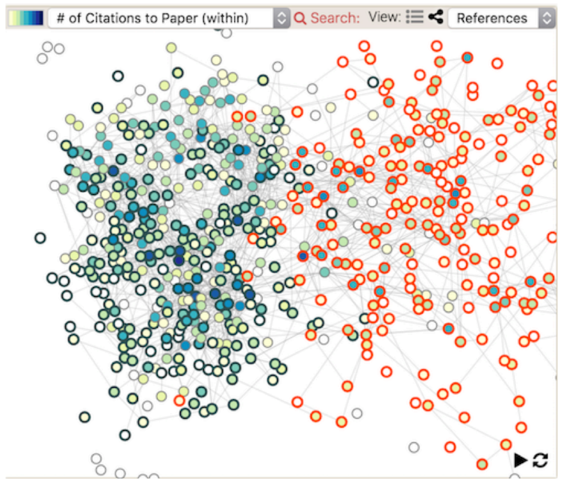
Time data (timestamp) → Line chart



Set-typed data (set relations) → Set matrix chart [3]

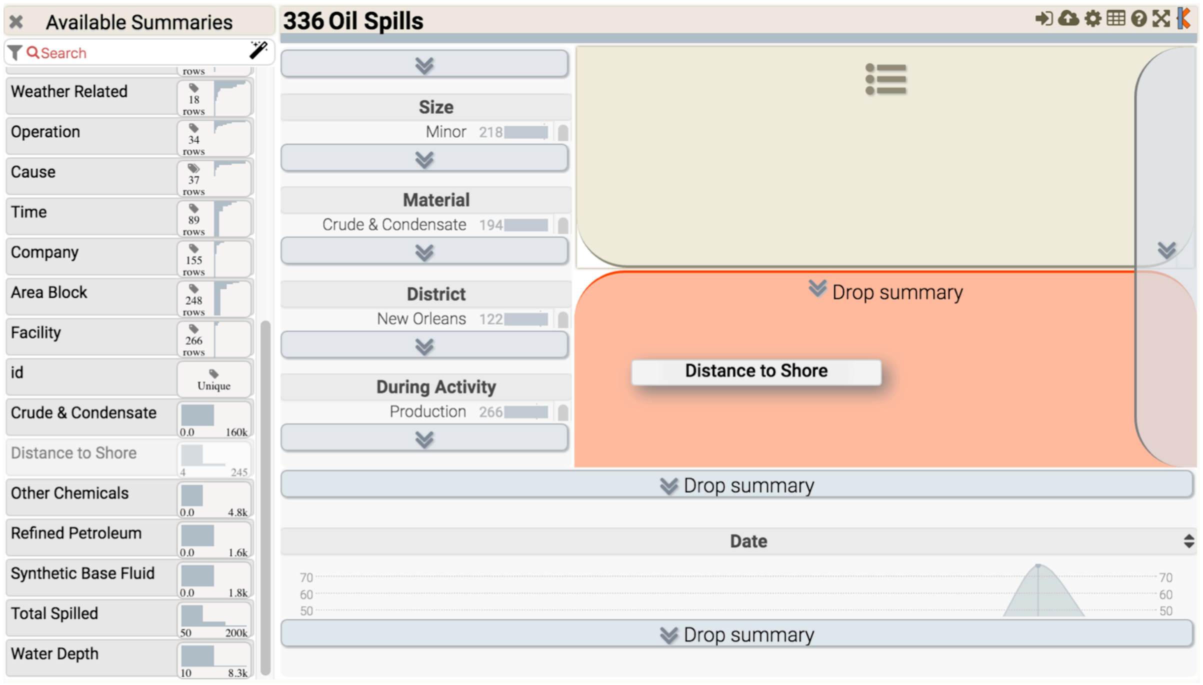


Spatial Region data → Maps (aggregated or per record)

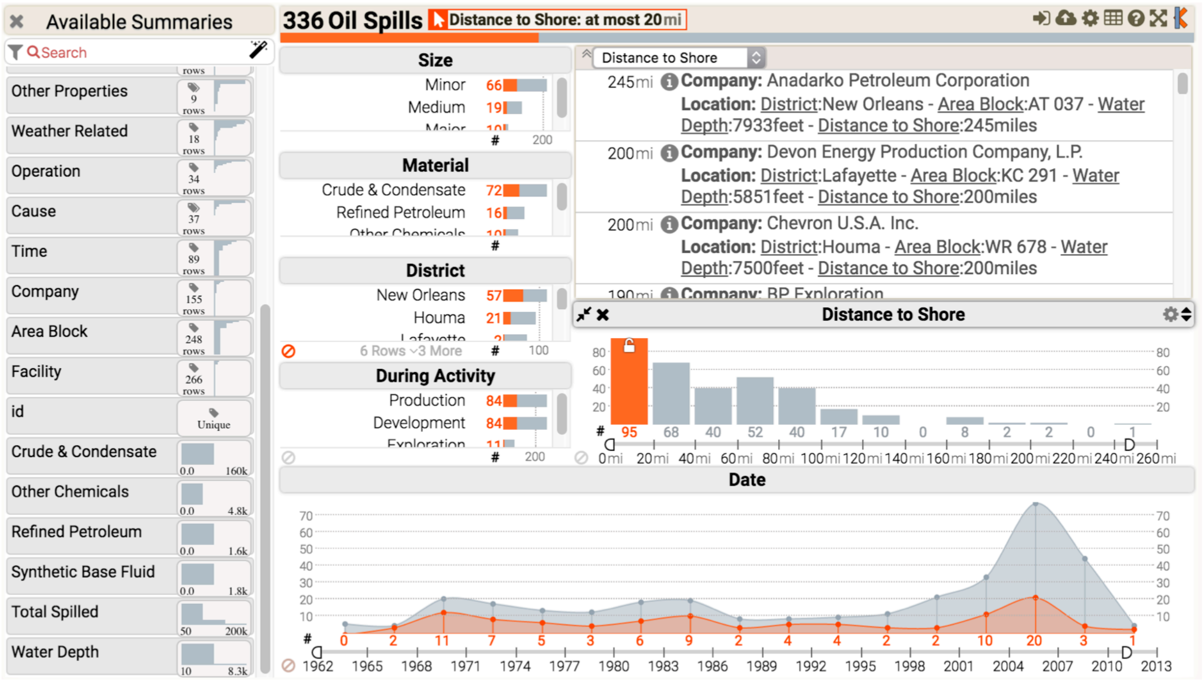


Network data (explicit links between records) → Node-link diagram

Figure 4. A selection of the visualizations integrated into Keshif from various datasets.



a) The attribute is dragged from Available Summaries panel to under the record display ☰.



245mi **Company:** Anadarko Petroleum Corporation
Location: District:New Orleans - Area Block:AT 037 - Water Depth:7933feet - Distance to Shore:245miles

200mi **Company:** Devon Energy Production Company, L.P.
Location: District:Lafayette - Area Block:KC 291 - Water Depth:5851feet - Distance to Shore:200miles

200mi **Company:** Chevron U.S.A. Inc.
Location: District:Houma - Area Block:WR 678 - Water Depth:7500feet - Distance to Shore:200miles

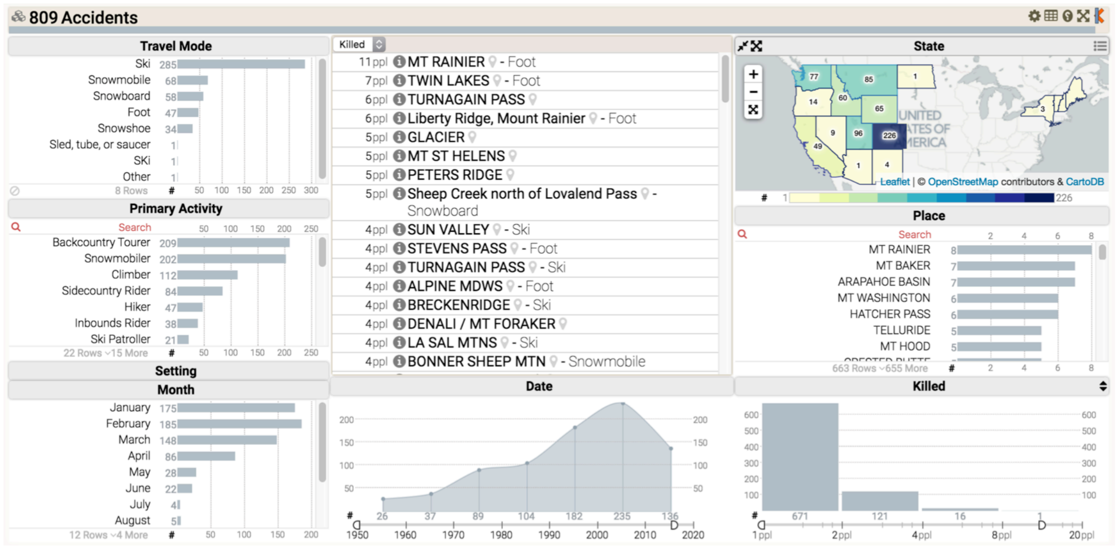
190mi **Company:** BP Exploration

b) The 0-20 mile range is higlighted in the summary, with distributions reflecting on all summaries and the record display. Records are also sorted by distance to shore, and the unit (mile) is automatically applied to all representations of this attribute.

Figure 5. A new summary (Distance to Shore) is added to the data browser in the authoring mode. This example is taken from www.keshif.me/demo/oilspills.

```
google.setOnLoadCallback(function(){
  browser = new kshf.Browser({
    domID: "#demo_Browser",
    recordName: "Accidents",
    categoryTextWidth: 160,
    rightPanelLabelWidth: 220,
    barChartWidth: 150,
    source: {
      gdocId: "1G2umrkAYK5nI-mElgQNYy2HditxQAR4K9wu7ia1V_TY",
      tables: "Accidents",
      callback: function(){ US_States.loadGeo(this); }
    },
    summaries: [
      { name: "Travel Mode", panel: "left" },
      { name: "Primary Activity", panel: "left" },
      { name: "Setting", panel: "left", collapsed: true },
      { name: "State", panel: "right", viewAs: "map",
        catLabel: function(){
          var v = US_States.index_code[this.id];
          return v ? v.name : this.id;
        },
        catMap: function(){ return US_States.index_code[this.id].geo; } },
      { name: "Place", value: "PLACE", panel: "right" },
      { name: "Date", panel: "middle" },
      { name: "Month", value: function(){ return this.Date.getUTCMonth(); }, catSortBy: "id", catLabel: _demo.Month },
      { name: "Killed", panel: "right", unitName: 'ppl' },
    ],
    recordDisplay: {
      sortBy: [ 'Killed', 'Date' ],
      sortColWidth: 65,
      recordView: function(){
        return this.PLACE + " <i class='fa fa-map-marker' style='color: lightgray;'></i>"+
          (this["Travel Mode"]?
            (" - <span style='font-weight: 100;'>"+this["Travel Mode"]+"</span>"): "");
      }
    }
  });
});
```

Code 1. Keshif configuration for an avalanche accidents dataset.



A screenshot of the corresponding Keshif browser defined by the Keshif configuration above.

This browser can be accessed at keshif.me/demo/AvalancheAccidents.
The full source of the web-page is available at github.com/adilyalcin/Keshif/blob/master/demo/AvalancheAccidents.html