Profiler: Integrated Statistical Analysis and Visualization for Data Quality Assessment

Sean Kandel*, Ravi Parikh*, Andreas Paepcke*, Joseph M. Hellerstein†, Jeffrey Heer*
*Stanford University †University of California, Berkeley

ACM AVI 2012

Presented by Yulun Du

CS 598 Human-in-the-loop Data Management

Fall 2015

Background

- Another work from Sean Kandel et al. one year after Wrangler.
- Data quality issues such as missing, erroneous, extreme and duplicate values undermine analysis and are time-consuming to find and fix.
- Automated methods can help identify anomalies, but determining what constitutes an error is context-dependent and so requires human judgment.
- While visualization tools can facilitate this process, analysts must often manually construct the necessary views to contextualize anomalies, requiring significant expertise.

Goals

- Using data mining methods to automatically identify data quality issues
- Suggesting coordinated summary visualizations for assessing the data in context
- Extensible system architecture: supports plug-in APIs
- Automatic view suggestion: view recommender
- Scalable summary visualizations: binning for brushing and linking

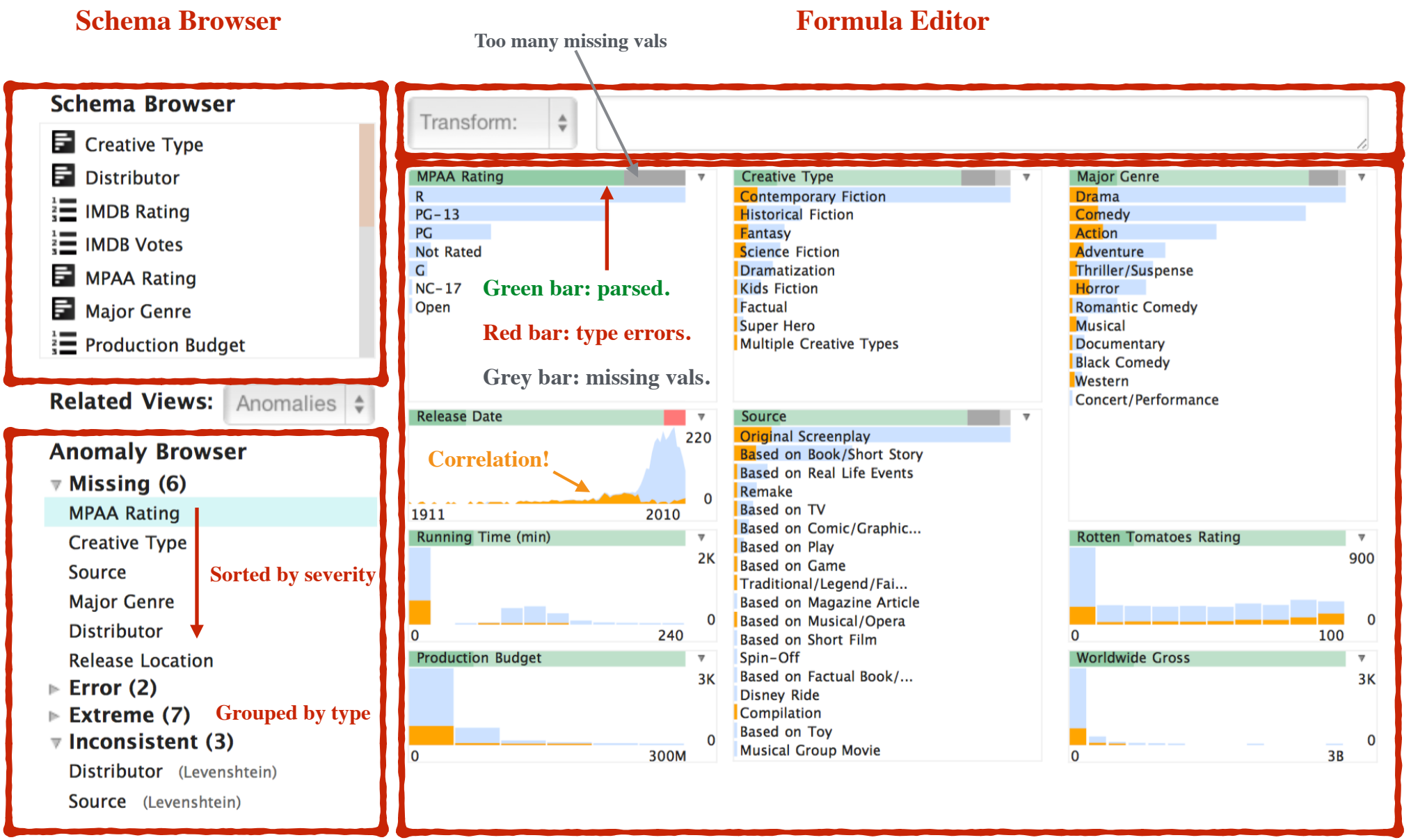
Related Work

- Taxonomies of anomalous data: Missing data, Erroneous data, Inconsistent data, Extreme values, Key violations.
- Existing data cleaning tools focus on: Data integration and entity resolution; Mass reformatting of raw input data; Specifications of data type definitions.
- Profiler focuses on data quality assessment.
- Unlike Potter's Wheel and Topes, Profiler generates visualizations.
- Unlike Google Refine, Profiler automatically suggests visualizations.
- Integrated with Wrangler's data transformation tool.

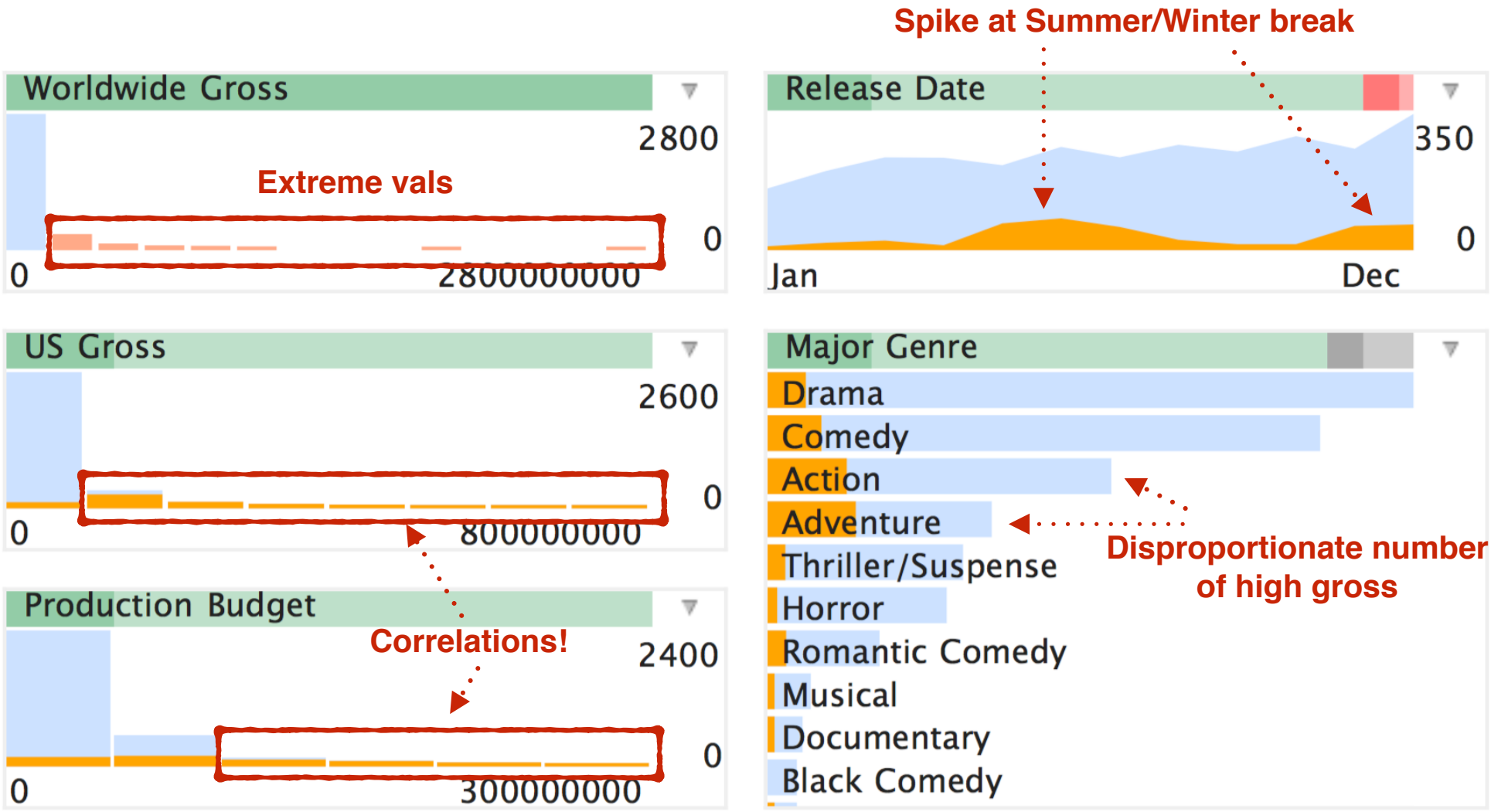
Related Work

- Unlike existing visualization tools:
- Coordinated multiple views enable assessment of relationships between data dimensions. Profiler extends this with a set of type-specific aggregate visualizations.
- Profiler automatically suggests combinations of data subsets for multi-dimensional views.
- Automates the choice of data columns, aggregation functions, and visual encodings.

Usage Scenario


Schema Browser
Formula Editor
Too many missing vals
Schema Browser
Creative Type
Distributor
IMDB Rating
IMDB Votes
MPAA Rating
Major Genre
Production Budget
Related Views:
Anomalies
Transform:
MPAA Rating
R
PG-13
PG
Not Rated
G
NC-17
Open
Green bar: parsed.
Red bar: type errors.
Grey bar: missing vals.
Creative Type
Contemporary Fiction
Historical Fiction
Fantasy
Science Fiction
Dramatization
Kids Fiction
Factual
Super Hero
Multiple Creative Types
Major Genre
Drama
Comedy
Action
Adventure
Thriller/Suspense
Horror
Romantic Comedy
Musical
Documentary
Black Comedy
Western
Concert/Performance
Release Date
Correlation!
220
0
1911
2010
Source
Original Screenplay
Based on Book/Short Story
Based on Real Life Events
Remake
Based on TV
Based on Comic/Graphic...
Based on Play
Based on Game
Traditional/Legend/Fai...
Based on Magazine Article
Based on Musical/Opera
Based on Short Film
Spin-Off
Based on Factual Book/...
Disney Ride
Compilation
Based on Toy
Musical Group Movie
Running Time (min)
2K
0
0
240
Rotten Tomatoes Rating
900
0
0
100
Production Budget
3K
0
0
300M
Worldwide Gross
3K
0
0
3B
Anomaly Browser
Missing (6)
MPAA Rating
Creative Type
Source
Major Genre
Distributor
Release Location
Sorted by severity
Error (2)
Extreme (7)
Grouped by type
Inconsistent (3)
Distributor (Levenshtein)
Source (Levenshtein)
Anomaly Browser
Canvas of linked summary visualizations

Orange is the Worldwide Gross



Conclusion: High Worldwide Gross outliers are exceptional values, not errors

Worldwide Gross vs U.S. Gross

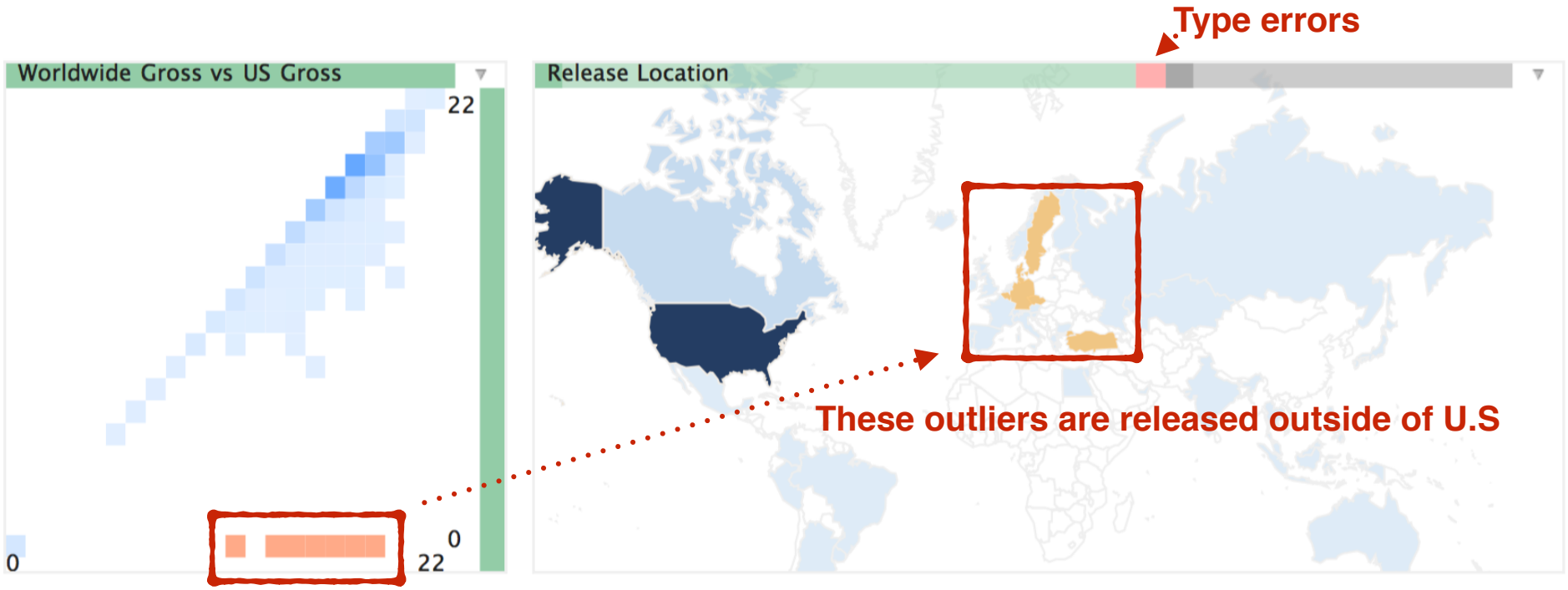


Figure 3: Map assessing 2D outliers in a binned scatter plot. Selected in the scatter plot are movies with high Worldwide Gross but low US Gross (in orange). Linked highlights on the map confirm that the movies were released outside of the US.

Duplicate detection by textual similarity

Figure 4: Conditioned duplicate detection. Left: Movie titles clustered by Levenshtein distance reveal over 200 potential duplicates. Right: Conditioning the clustering routine on 'Release Year' reduces the number of potential duplicates to 10.

System Architecture

High-level view

- Extensible system
- Statistical algorithms
- Coordinated visualizations
- Run inside browsers; implemented with JS
- Five major components

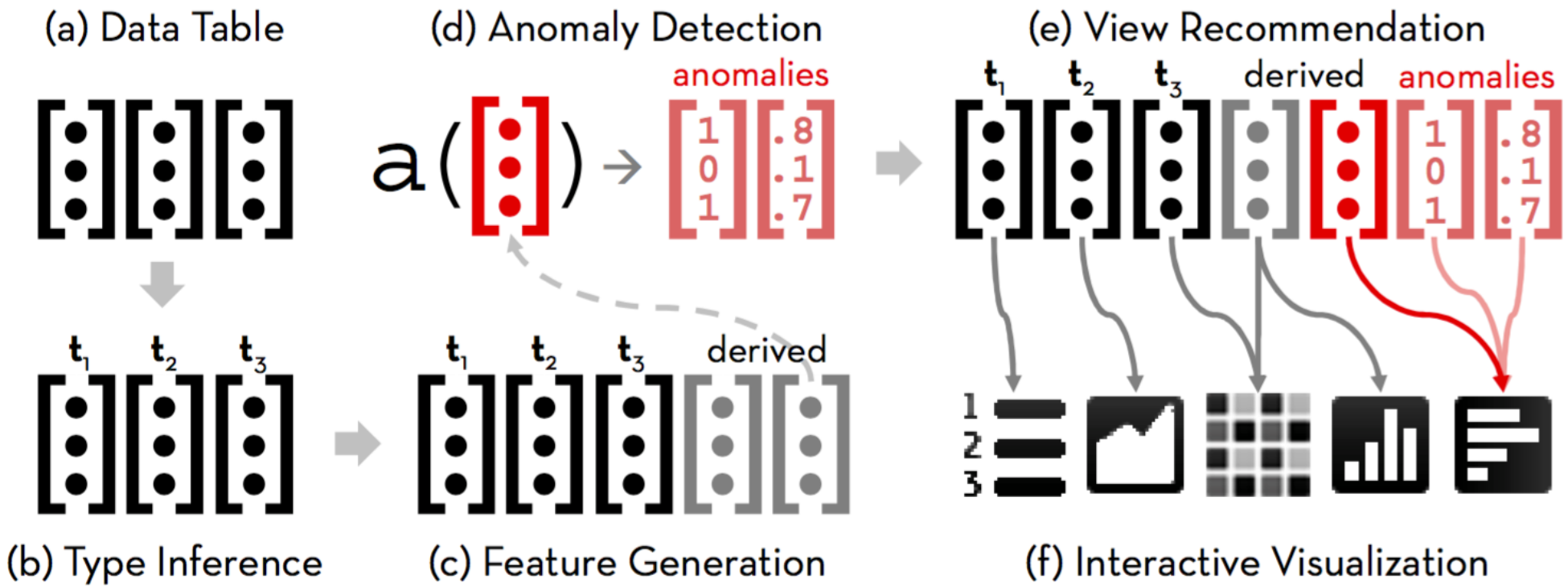


Figure 5: The Profiler Architecture. An (a) input table is analyzed to (b) infer types for each column. Type information is used to (c) generate features prior to running (d) anomaly detection routines. The results of anomaly detection and mutual information analysis are used to perform (e) view recommendation and populate a set of (f) interactive visualizations.

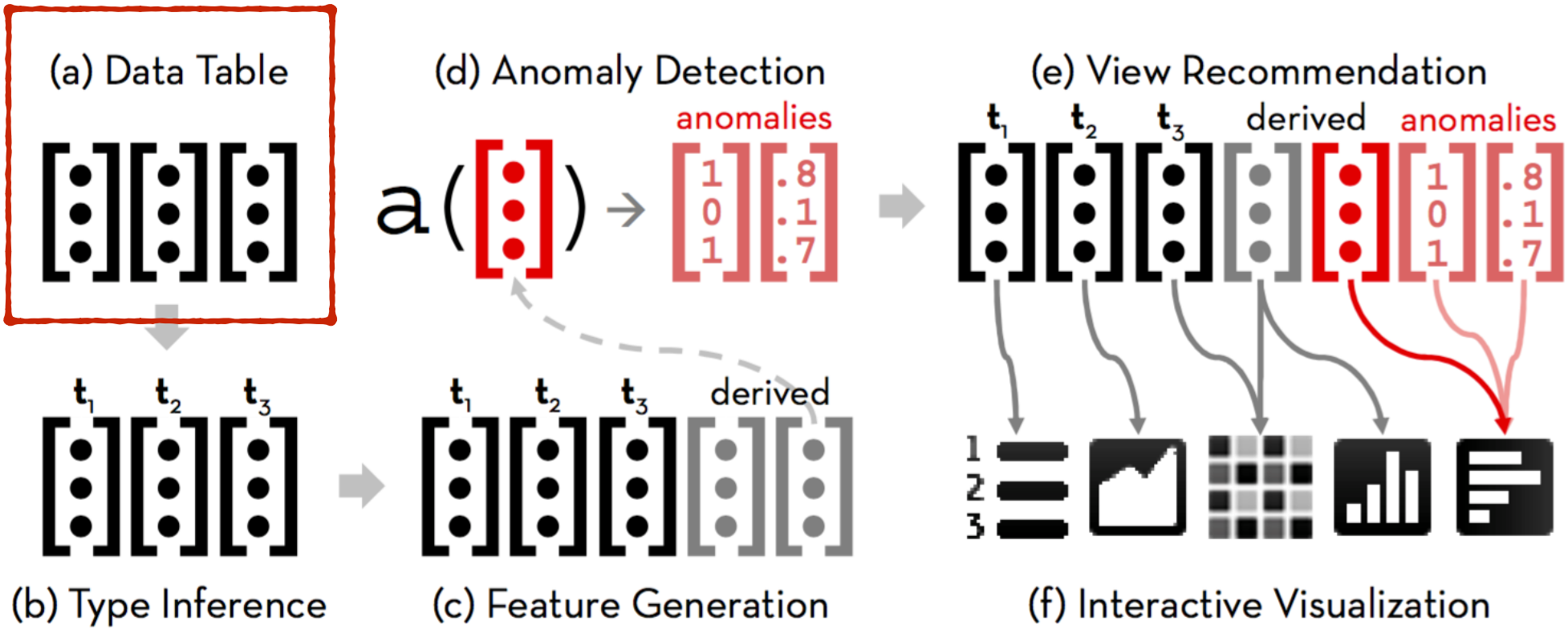


Figure 5: The Profiler Architecture. An (a) input table is analyzed to (b) infer types for each column. Type information is used to (c) generate features prior to running (d) anomaly detection routines. The results of anomaly detection and mutual information analysis are used to perform (e) view recommendation and populate a set of (f) interactive visualizations.

1. Data Tables

- A memory-resilient column-oriented RDBMS
- Standard SQL-style queries: filtering, aggregation, and generating derived columns
- Unlike standard SQL DB: Relaxed type system allows type deviation of values and only flags inconsistent values
- Wrangler's data transformation language: extends with additional transforms. eg. more advanced binning aggregation.

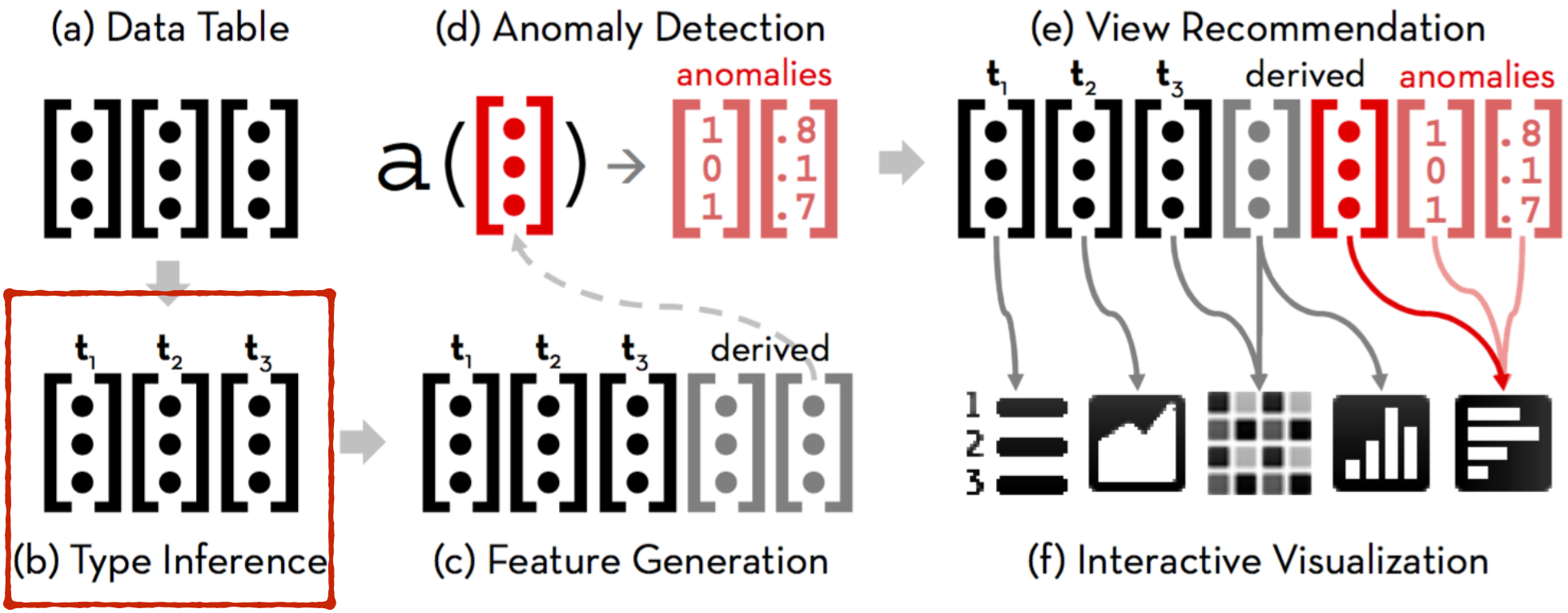


Figure 5: The Profiler Architecture. An (a) input table is analyzed to (b) infer types for each column. Type information is used to (c) generate features prior to running (d) anomaly detection routines. The results of anomaly detection and mutual information analysis are used to perform (e) view recommendation and populate a set of (f) interactive visualizations.

2. Type Registry

- Each column must have a type: inferred or by user
- Type defined by a binary verification function: true or false by regex match, set membership, range constraints
- Primitive types: boolean, string, numeric
- Higher order types: country name, zip code, etc.
- Extensibility: define new types, new specifications…

2. Type Registry (cont.)

- Type definitions may also include a set of type transforms and group-by functions
  - Type transforms: do mappings between types. eg. mapping zip codes to lat-lon coordinates.
  - Group-by functions: grouping values to drive scalable visualizations. eg. binned numerics
- Type inference: Minimum Description Length (MDL); same principle used in Potter's Wheel.
- MDL: selects type that minimizes the number of bits needed to encode the values in a column.

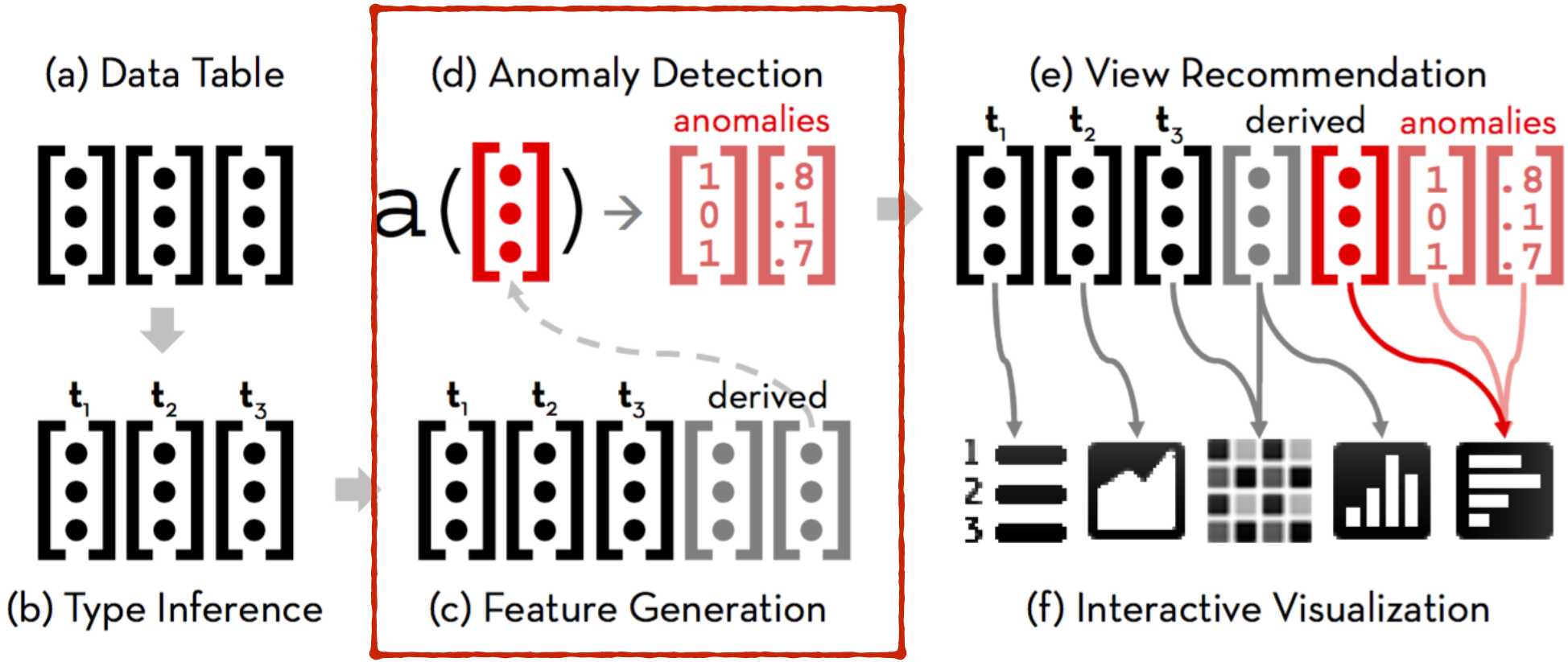


Figure 5: The Profiler Architecture. An (a) input table is analyzed to (b) infer types for each column. Type information is used to (c) generate features prior to running (d) anomaly detection routines. The results of anomaly detection and mutual information analysis are used to perform (e) view recommendation and populate a set of (f) interactive visualizations.

3.1 Detector - Pipeline

- 2 phases: feature generation & anomaly detection
- Features extracted by generators. eg. len(str)
- Features are fed to anomaly detection routines. eg. len(str) —A.D.routines—>z-score. Too long?
- Detector maintains a list of appropriate generators for each type.

3.1 Detector - Pipeline

- (cont.)
- Anomaly Detection routines accepts feature columns as input, then outputs two columns: a class column and a certainty column.
- Class column: integers
- For each row, 0—>no anomaly; non-zero—>other classes
- Certainty column: strength of prediction. eg. z-score as distance from mean.

3.1 Detector - Pipeline

- (cont.)
- Detection routines run on all of the columns, including generated feature columns, with compatible type. eg. z-score on all numerics
- Anomaly browser: list all the anomaly results detected by routines in decreasing anomaly count.
- Output class and certainty are handled by View Recommender (more on this later…)

3.2 Detector - Routines

- Five basic routines:
  - Missing value detection. i.e. empty cells
  - Type verification. i.e. type errors or restraint violations
  - Clustering. Nearest Neighbor clustering with chosen distance metric
  - Univariate outlier detection. i.e. extreme values
  - Frequency outlier detection. eg. Unique value ratio

Type	Issue	Detection Method(s)	Visualization
Missing	Missing record	Outlier Detection | Residuals then Moving Average w/ Hampel X84	Histogram, Area Chart
		Frequency Outlier Detection | Hampel X84	Histogram, Area Chart
	Missing value	Find NULL/empty values	Quality Bar
Inconsistent	Measurement units	Clustering | Euclidean Distance	Histogram, Scatter Plot
		Outlier Detection | z-score, Hampel X84	Histogram, Scatter Plot
	Misspelling	Clustering | Levenshtein Distance	Grouped Bar Chart
	Ordering	Clustering | Atomic Strings	Grouped Bar Chart
	Representation	Clustering | Structure Extraction	Grouped Bar Chart
	Special characters	Clustering | Structure Extraction	Grouped Bar Chart
Incorrect	Erroneous entry	Outlier Detection | z-score, Hampel X84	Histogram
	Extraneous data	Type Verification Function	Quality Bar
	Misfielded	Type Verification Function	Quality Bar
	Wrong physical data type	Type Verification Function	Quality Bar
Extreme	Numeric outliers	Outlier Detection | z-score, Hampel X84, Mahalanobis distance	Histogram, Scatter Plot
	Time-series outliers	Outlier Detection | Residuals vs. Moving Average then Hampel X84	Area Chart
Schema	Primary key violation	Frequency Outlier Detection | Unique Value Ratio	Bar Chart

Figure 6: Taxonomy of Data Quality Issues. We list classes of methods for detecting each issue, example routines used in Profiler, and visualizations for assessing their output.

3.2 Detector - Routines

- (cont.)
- Two multivariate outlier detection routines:
    - 1. Accepting multiple columns as input. eg. Mahalanobis distance.
    - 2. Conditioning on grouped data. eg. categorical data, binned numerics.
- Not applied by default due to high complexity; User can initiate by adding conditioning columns.

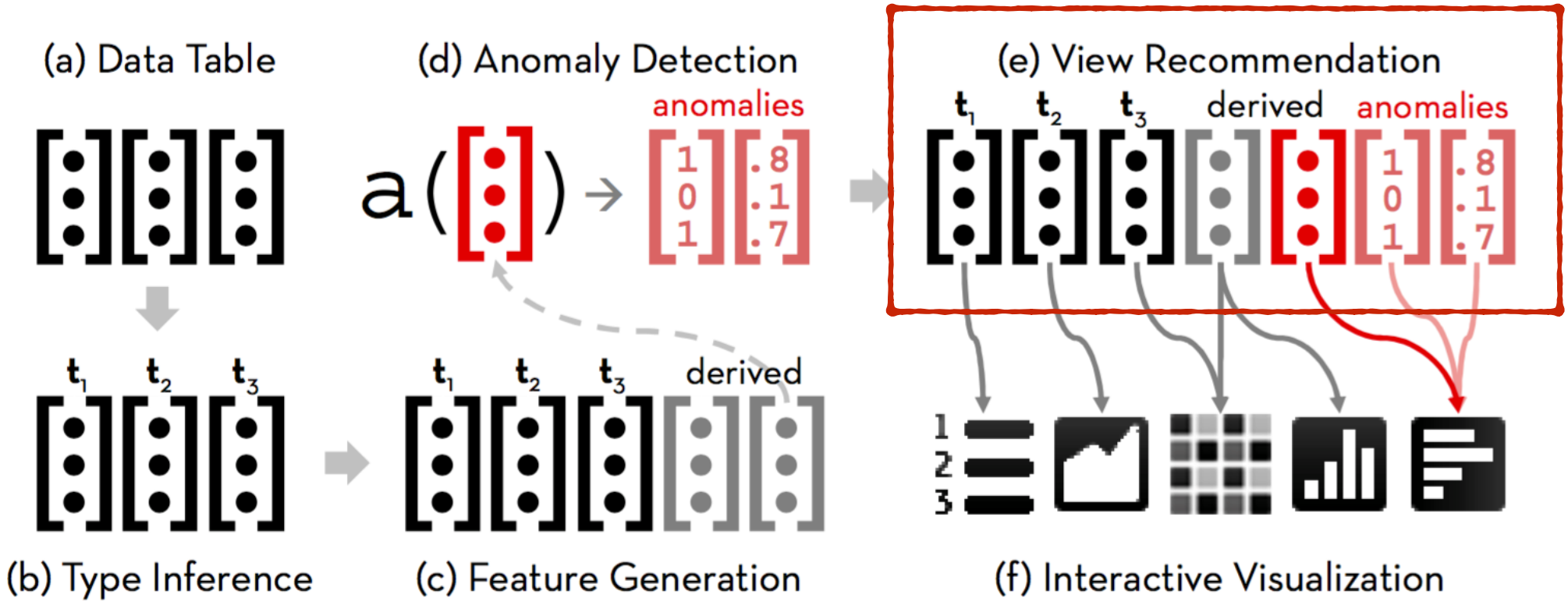


Figure 5: The Profiler Architecture. An (a) input table is analyzed to (b) infer types for each column. Type information is used to (c) generate features prior to running (d) anomaly detection routines. The results of anomaly detection and mutual information analysis are used to perform (e) view recommendation and populate a set of (f) interactive visualizations.

4. View Recommendation

- Recommends a view specification for the View Manager (more on this later…)
- A view specification: a set of columns to visualize and type-appropriate group-by functions for aggregation. May also include class and certainty columns to parameterize a view.
- Primary View: visualize the column that contains the anomaly.
- Related Views: a set of related views by mutual information.
    - Two types: Anomaly-oriented and Value-oriented

4.1 Mutual Information

- Formal definition: Reduction in entropy attained by knowing a second variable.
- My interpretation: The dependence between two variables.
- Should be non-negative values.
- Minimum value = 0 —> independent
- Distance metric D: $$D(X,Y) = 1 - \left(\frac{I(X,Y)}{max(H(X), H(Y))} \right)$$

4.2 Recommendation

- Some definitions:
  - ViewToColumn: view specification—>column of group ids
  - VS_c: a set of all possible view specifications containing one column from a set of columns C and a type-appropriate group-by function.

4.2 Recommendation

- To suggest the primary view: produce a summary view with bins that minimize the overlap of anomalies and non-anomalies so that analysts can better discriminate them.

- More formally, if A is the set of columns containing the anomaly, we recommend the view specification vs in set VS_A that minimizes the quantity D(ViewToColumn(vs), class). This primary view specification (denoted pvs) is assigned the class and certainty columns as parameters.

4.2 Recommendation

- To suggest anomaly-oriented views: find other columns that best predict the class column.

- We consider the set of all columns R that exclude the columns in C. We then choose view specifications from VS_R that predict the class column. We sort specifications vs in set VS_R by increasing values of D(ViewToColumn(vs), class). The Recommender populates the View Manager with the corresponding visual summaries in sort order until the canvas is full, discarding summaries that contain columns already visualized.

4.2 Recommendation

- To recommend value-oriented views: Value-oriented views show visualizations related to the entire distribution of values in the primary view, not just anomalies. Instead of predicting the class column, we predict the group ids generated by the primary view specification.(psv)

- We sort view specifications vs in set VS_R by D(ViewToColumn(vs), ViewToColumn(pvs)). Because VS_R only contains view specifications with one column, only univariate summaries are suggested. Our approach extends to multiple columns if we augment R to include larger subsets of columns.

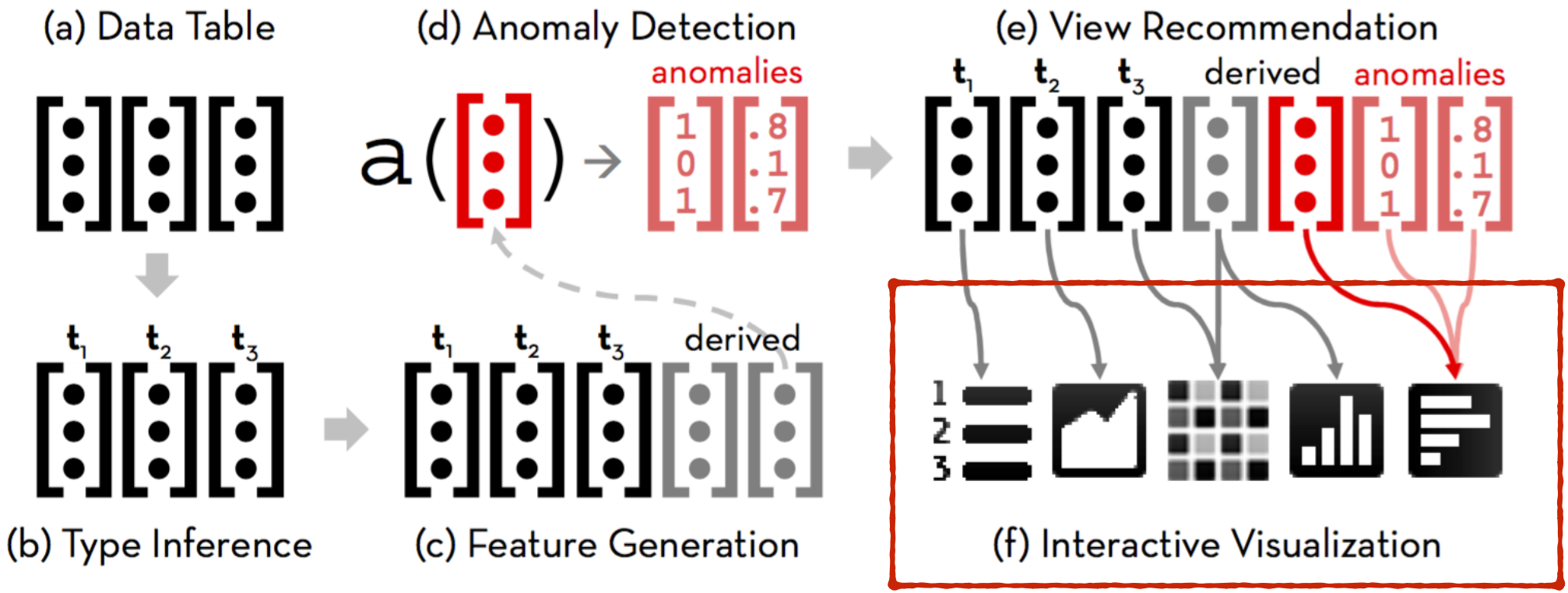


Figure 5: The Profiler Architecture. An (a) input table is analyzed to (b) infer types for each column. Type information is used to (c) generate features prior to running (d) anomaly detection routines. The results of anomaly detection and mutual information analysis are used to perform (e) view recommendation and populate a set of (f) interactive visualizations.

5. View Manager

- View manager: View specifications—VM—>a set of linked visual summaries.
- Type-specific views to reveal patterns. eg. gaps, clusters, and outliers.
- Query Engine for filtering and aggregating: To support Brushing and Linking
- Manual construction of views by user interactions.

5.1 Summary Visualizations

- Scalability: The number of marks depends on the number of bins, not on the number of records.
- Requires a group-by function with a binning strategy
  - Binning for automatically generated view: determined by Recommender.
  - Binning for user selected view: determined by Profiler based on the range of data value.
- User preference of GBF and type transform: by user

5.1 Summary Visualizations

- (cont.)
- Histograms: Numeric data
- Area charts: Temporal data
- Choropleth maps: Geographic data
- Binned scatter plots: 2D Numeric or Temporal
- Rectangular binning for better query and rendering performance.

5.1 Summary Visualizations

- (cont.)
- Bar charts: Frequencies of distinct nominal values.
- Grouped bar charts: Frequencies of clustered values. eg. possible duplicates
- Columns with high cardinality? Scroll
- Continuous bars: Windowed aggregation over continuous bars to form summery counts.

5.1 Summary Visualizations

- (cont.)
- Data quality bars: valid, type errors, missing
- ~~Naive scaling of bar heights and color ramps~~: Results in invisible low-frequency bins, where outliers reside.
- Solution: perception discontinuity. Introduce minimum heights, make every color distinguishable. (next slide)
- Other: TS binning by time spans; Map panning and zooming
- Each view can be parameterized by class and certainty. eg. sorting on certainty

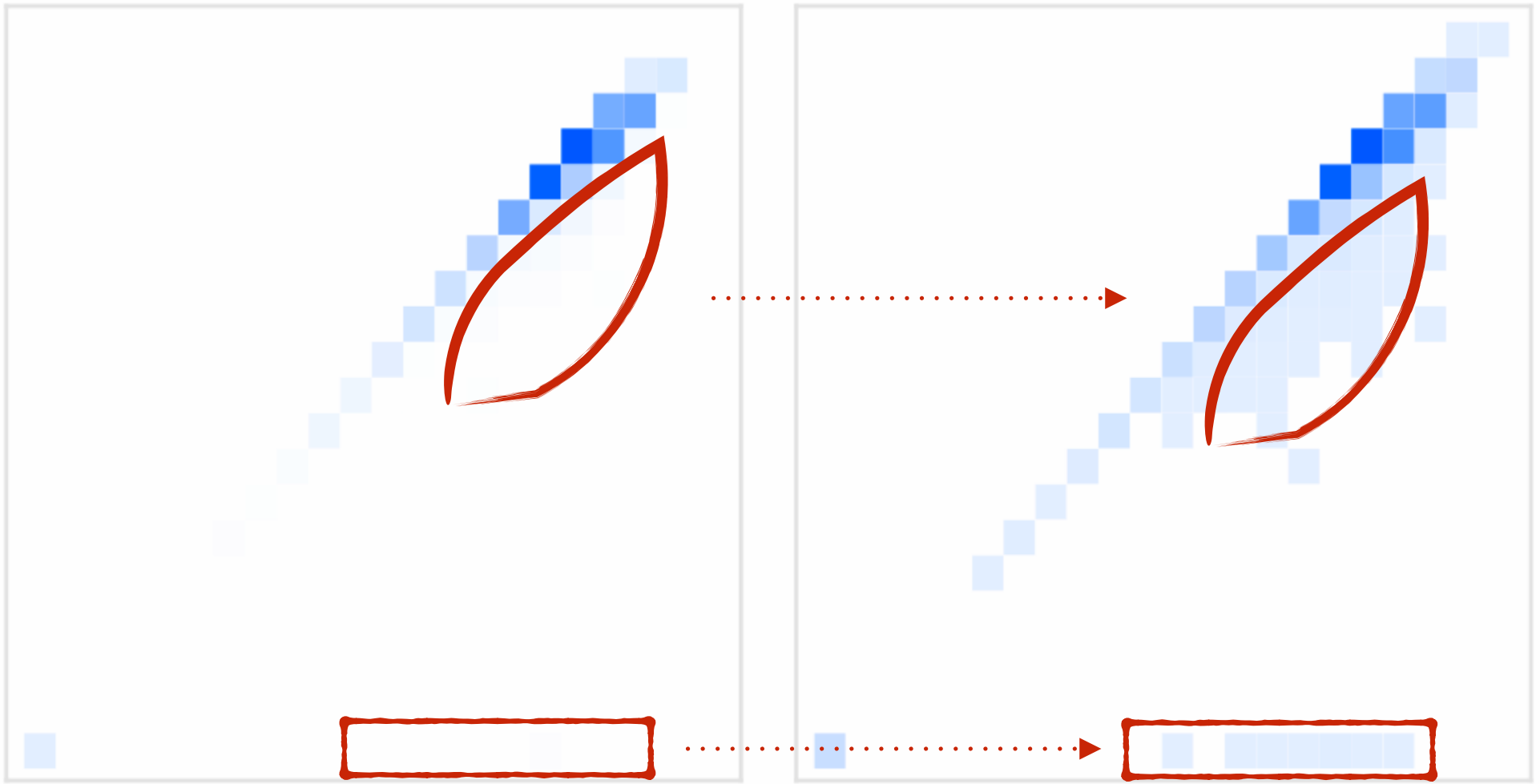


Figure 7: Adding perceptual discontinuity to summary views. Left: A binned scatter plot using a naive opacity ramp from 0-1. Right: An opacity scale with a minimum non-zero opacity ensures perception of bins containing relatively few data points.

5.2 Scalable Linked Highlight

- Profiler highlights the projection of a range of values across all views.
- For scalable, real-time interaction: Should optimize query execution and rendering

5.2 Scalable Linked Highlight

- (cont.)
- Query Execution Optimization:
  - Reduce query load: Pre-aggregate data into a suitable number of bins—>reduce #records by one or two orders of magnitude.
  - Reduce query time: Encode non-numeric types as zero-based integers. Store original values in sorting order in a lookup table.
  - Inner loop of the query executor avoids function calls.
  - Cache query results for possible reuse of data.

5.2 Scalable Linked Highlight

- (cont.)
- Rendering Optimization:
  - Minimize modifications to the DOM in each interactive update.
  - To avoid churn, introduce all SVG DOM elements upon initialization.

5.3 Performance Benchmark

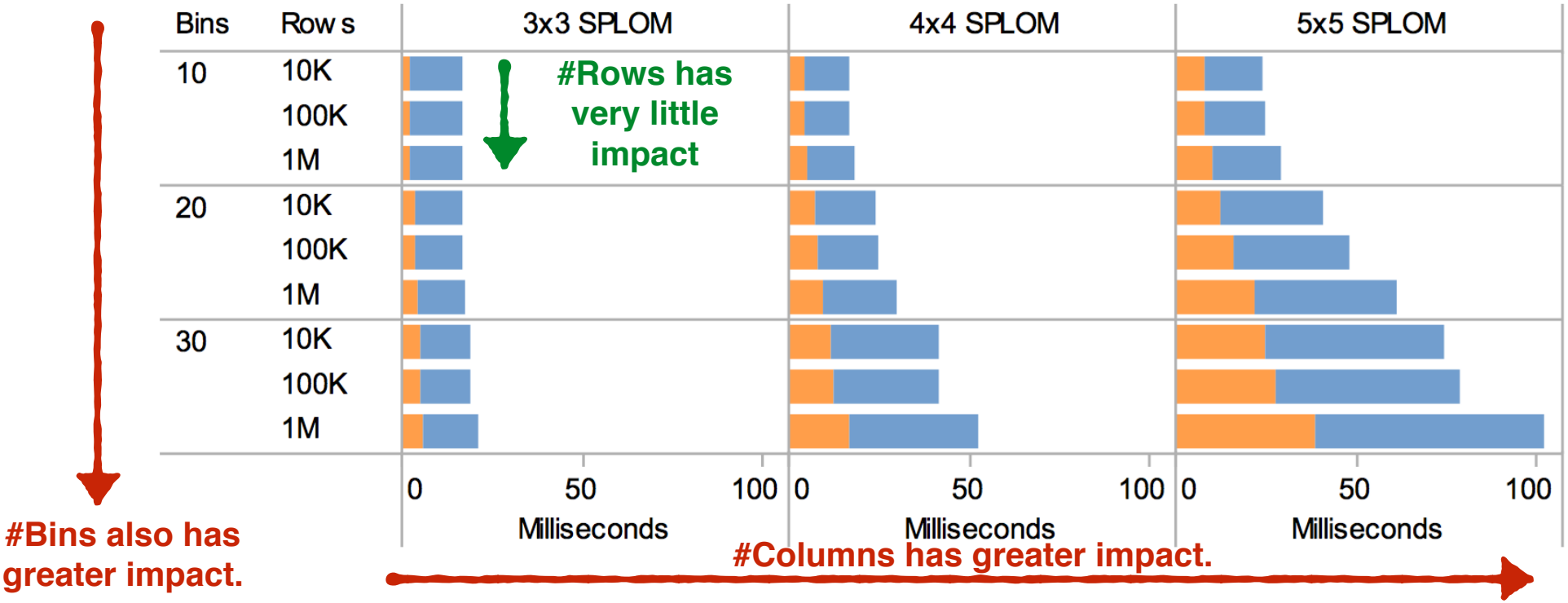


Figure 8: Performance (in ms) of linked highlighting in a scatter plot matrix (SPLOM). Orange bars represent query processing time, blue bars represent rendering time. We varied the number of dimensions, bins per dimension and data set size. In most cases we achieve interactive (sub-100ms) response rates.

Initial Usage

- A disasters database
- World Water Monitoring Day data
- All suggest rapid assessment of data with the aid of Profiler.

Conclusion

- Profiler can reduce the time spent diagnosing data quality issues, allowing domain experts to discover issues and spend more time performing meaningful analysis.

Future Work

- Further evaluations: controlled studies and public deployments on the web
- Define custom types for additional data types. eg. free-form text
- Hybrid query engine: combine server and client to solve the limited memory problem.
- Consider Bayesian network for conditional dependancies to improve ranking in Recommender.

Thank you!