Reusing Interactive Analysis Workflows

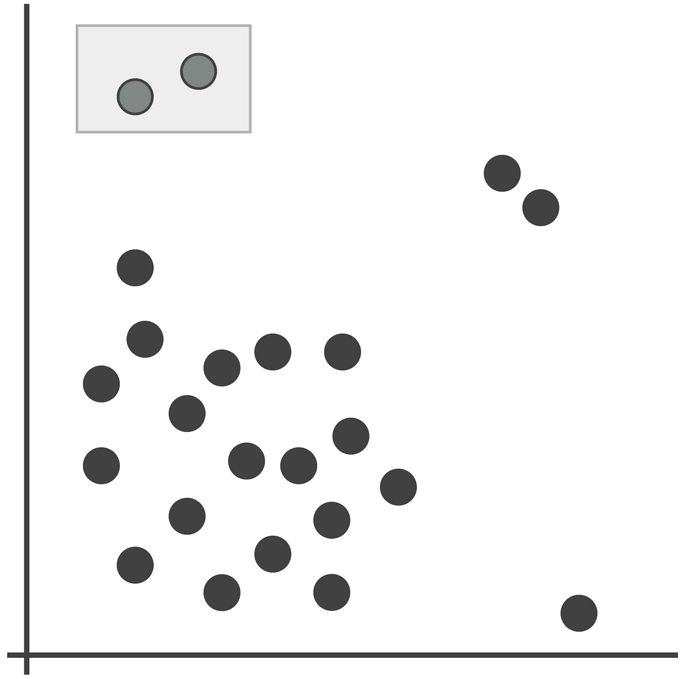
Kiran **Gadhave**, Zach **Cutler**, Alexander **Lex**
University of Utah

THE UNIVERSITY OF UTAH

visualization design lab

SCI
www.sci.utah.edu

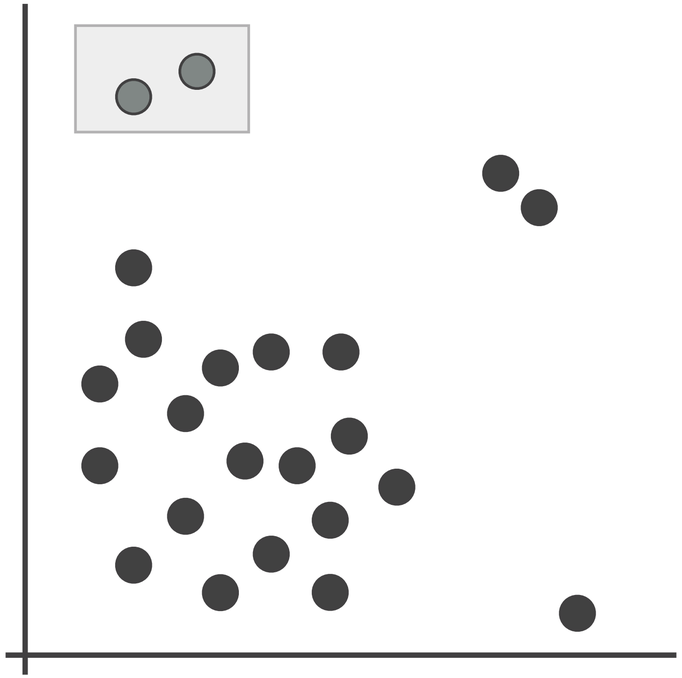
Interactive Visual Analysis



Computational Analysis

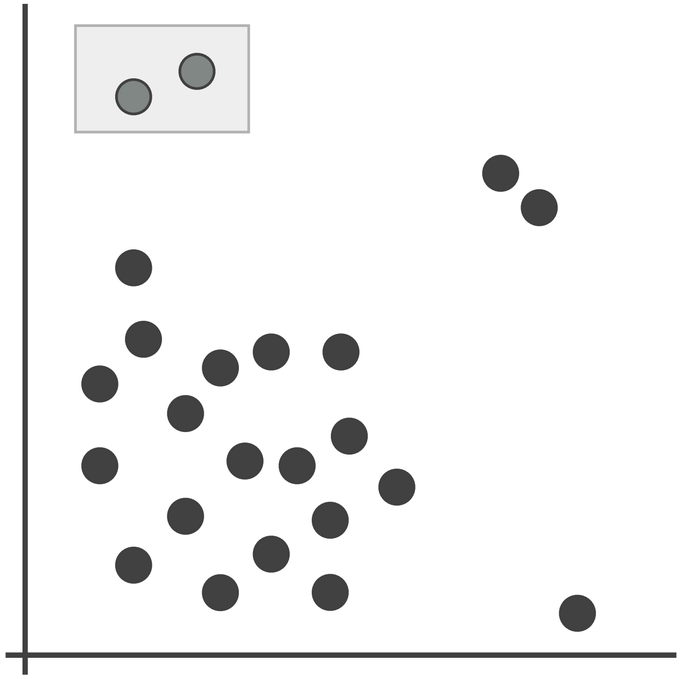
</>

Interactive Visual Analysis



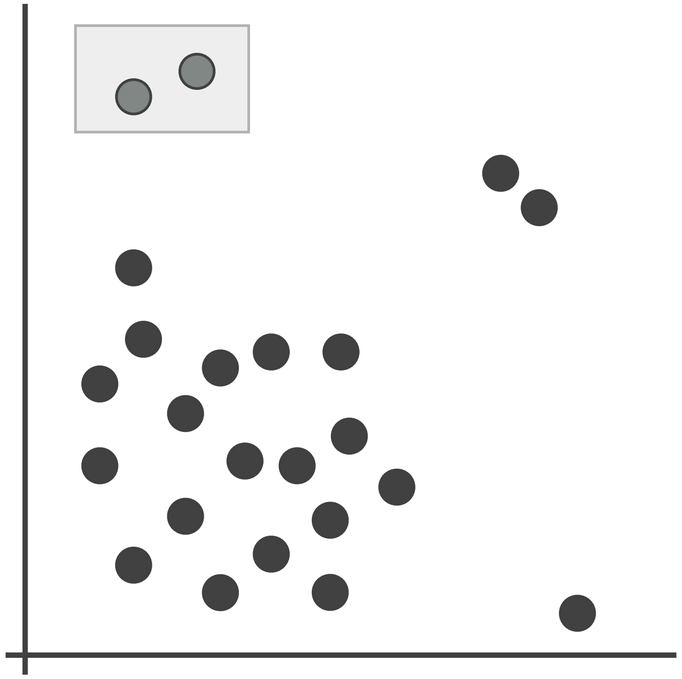
Interactive Visual Analysis

- Intuitive



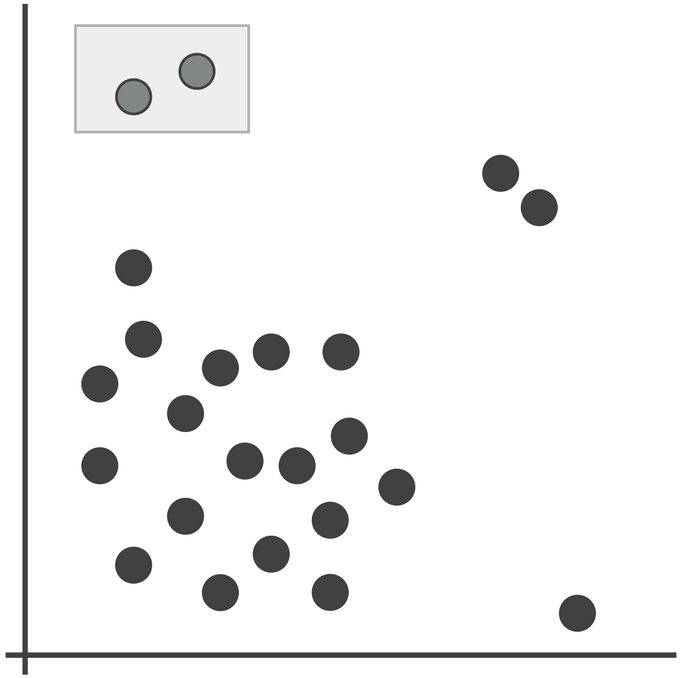
Interactive Visual Analysis

- Intuitive
- Uses human perceptual capabilities



Interactive Visual Analysis

- Intuitive
- Uses human perceptual capabilities



- Need to redo the analysis when the datasets update

Computational Analysis

Computational Analysis

- Flexible

Computational Analysis

- Flexible
- Reusable

Computational Analysis

- Flexible
- Reusable

- Have to know how to program

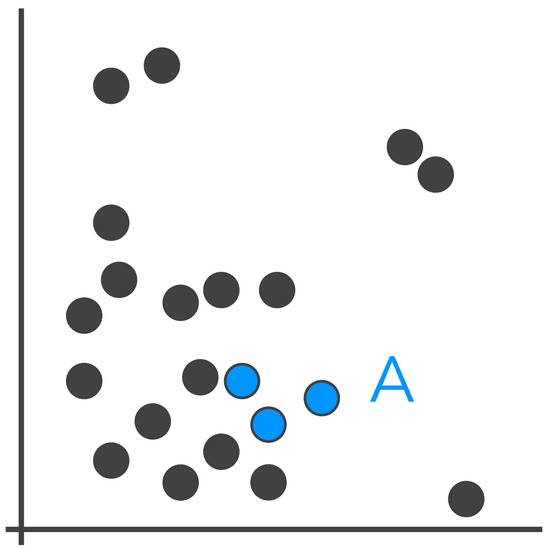
Computational Analysis

- Flexible
- Reusable

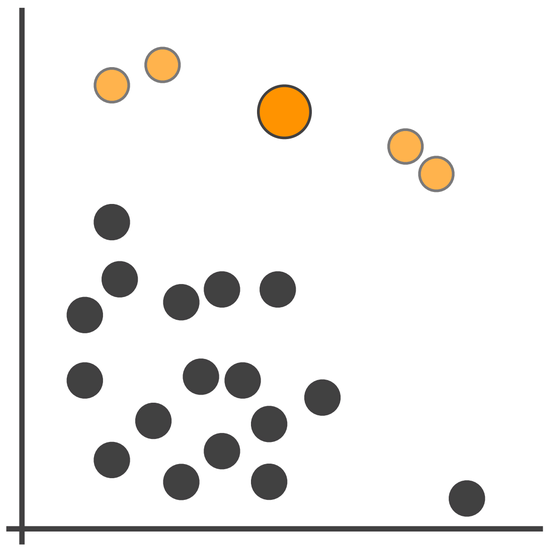
- Have to know how to program
- Time consuming

Certain tasks are easier with interactive visualizations

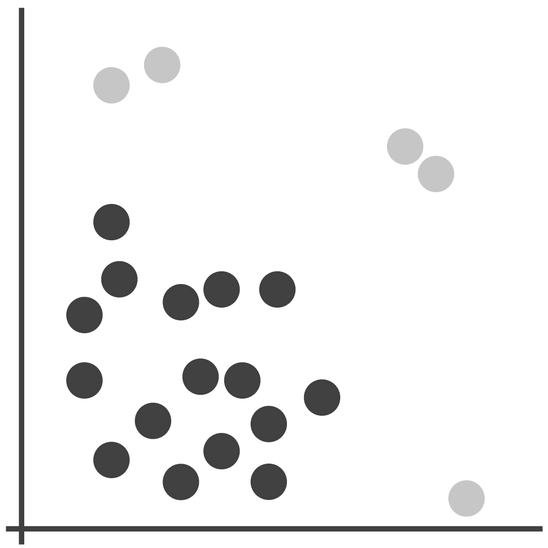
Certain tasks are easier with interactive visualizations



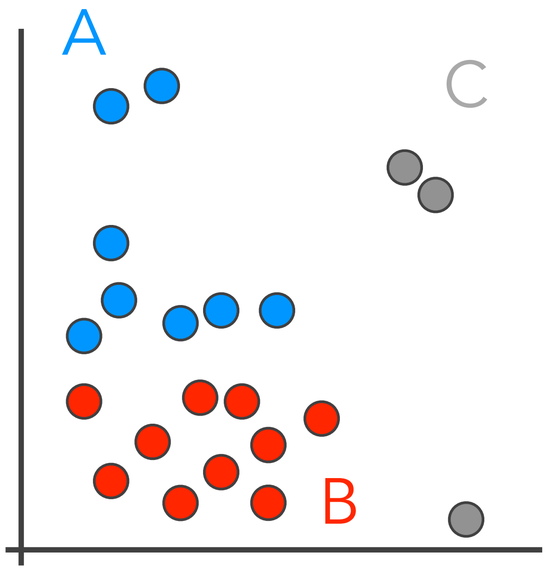
Labeling



Aggregation

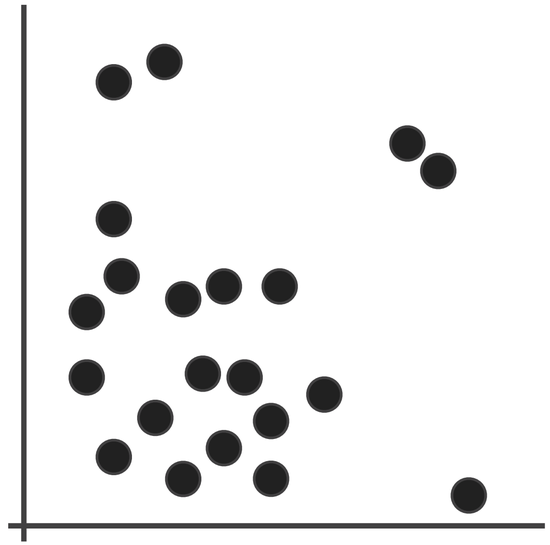


Filtering



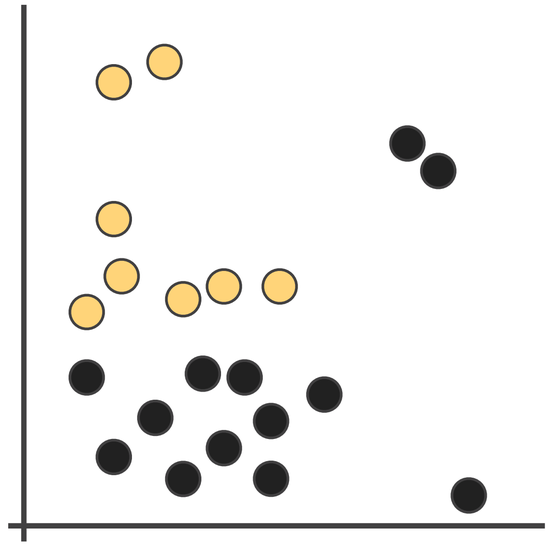
Categorization

In a computational notebook



In a computational notebook

```
sel = dataframe[:, …]
```

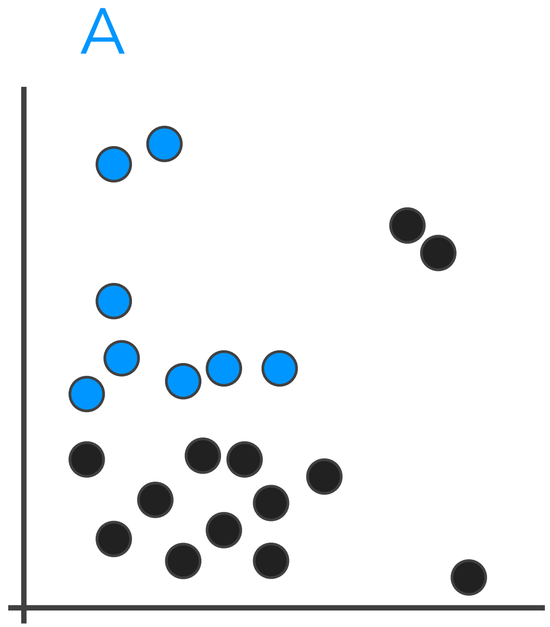


In a computational notebook

```
sel = dataframe[:, …]
```



```
sel.add_column(A)
```



In a computational notebook

```
sel = dataframe[:, …]
```

↓

```
sel.add_column(A)
```

```
sel = dataframe[:, …]
```

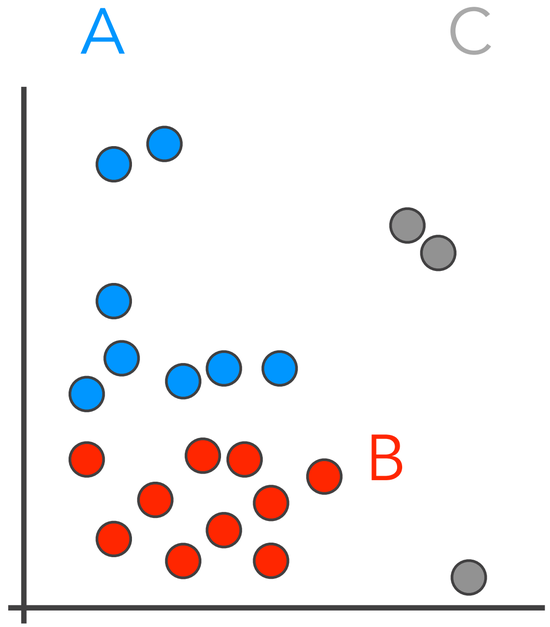
↓

```
sel.add_column(B)
```

```
sel = dataframe[:, …]
```

↓

```
sel.add_column(C)
```



In a computational notebook

```
sel = dataframe[:, …]
```

↓

```
sel.add_column(A)
```

```
sel = dataframe[:, …]
```

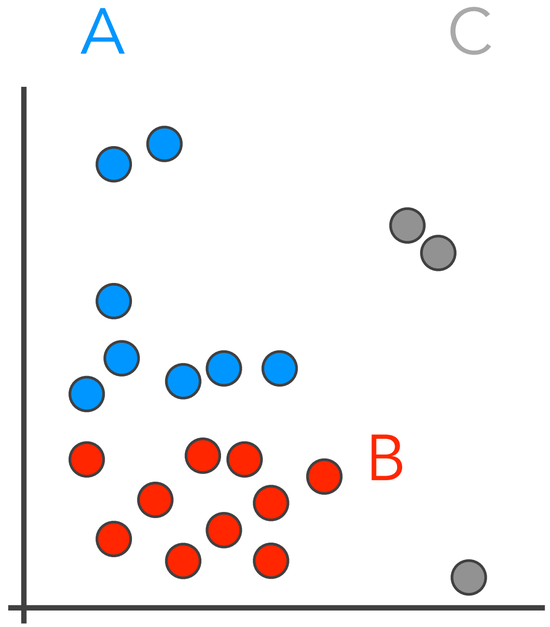
↓

```
sel.add_column(B)
```

```
sel = dataframe[:, …]
```

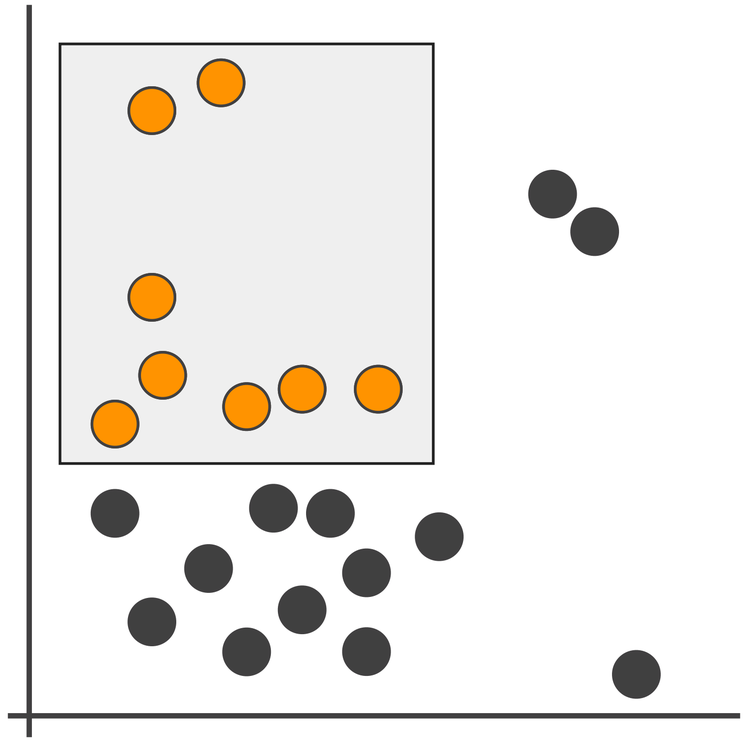
↓

```
sel.add_column(C)
```

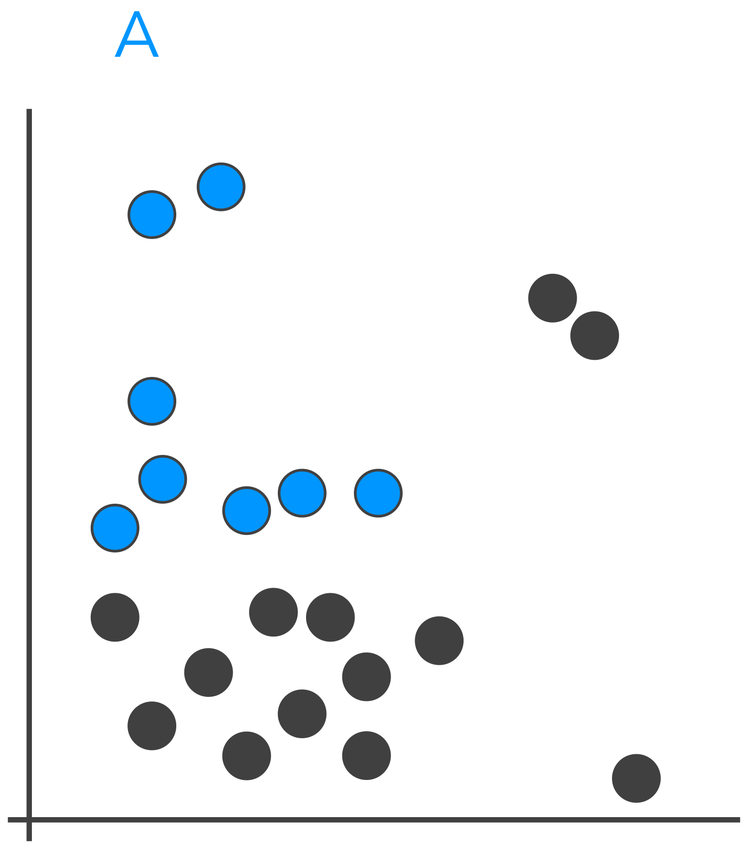


Depending on the query predicate, this can get real complex

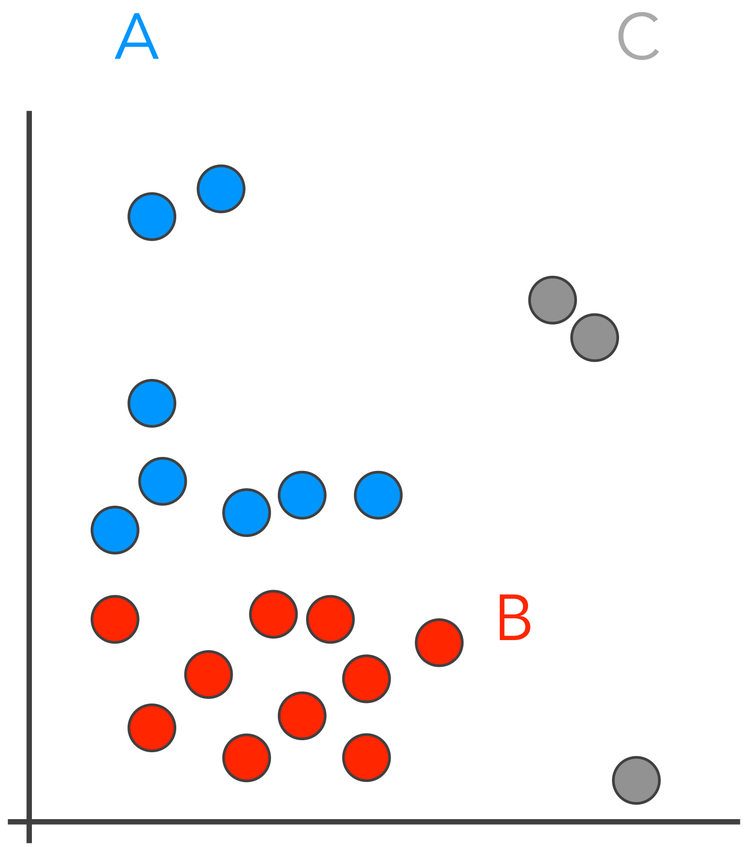
whereas with an interactive visualization



whereas with an interactive visualization



whereas with an interactive visualization



But what if we want to reuse our analysis?

But what if we want to reuse our analysis?

Functions can be parameterized and reused

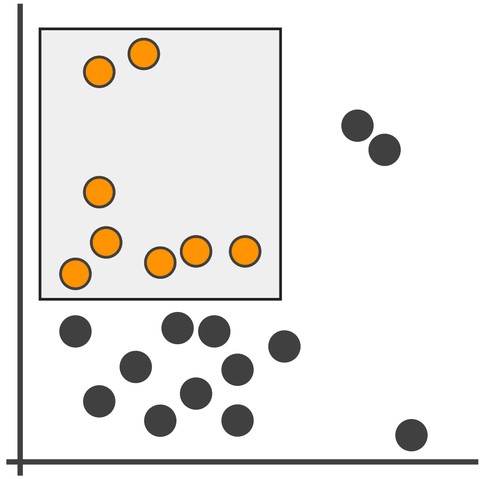
```
def add_category(dataframe, col):
    selection = dataframe[:, ...]
    selection.add_column(col)
```

But what if we want to reuse our analysis?

Functions can be parameterized and reused

```
def add_category(dataframe, col):
    selection = dataframe[:, ...]
    selection.add_column(col)
```

Interactions need to be repeated



Can we make visual analysis reusable?

Contribution

Capture analysis provenance

Contribution

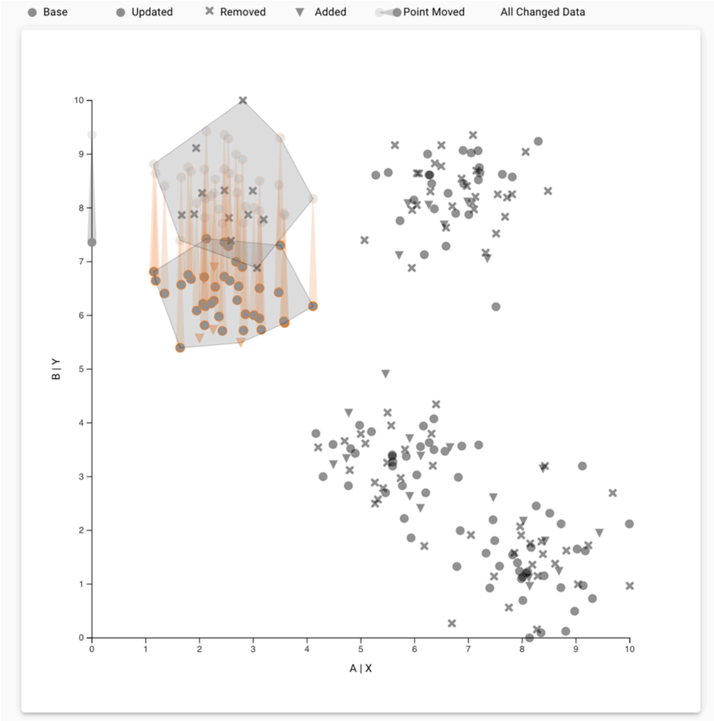
Capture analysis provenance

and curate reusable workflows

THE UNIVERSITY OF UTAH

Contribution

Reusable Workflows




```
# Get the desired workflow
wf = project.get_workflow("1638475878304")

# Description of the options in the workflow
wf.describe()
```

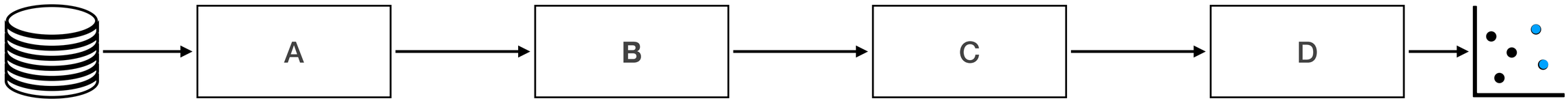
```
Categorize outliers
| Root
 +--| Adding scatterplot for new_cases_per_million-new_deaths_per_million
     +--| Apply Outlier selection
         +--| Filter In
             +--| Add Brush
                 +--| Categorize Selections
                     +--| Add Brush
                         +--| Update Brush
                             +--| Categorize Selections
                                 +--| Add Brush
                                     +--| Categorize Selections
```

Reapply the workflows on **updated datasets**

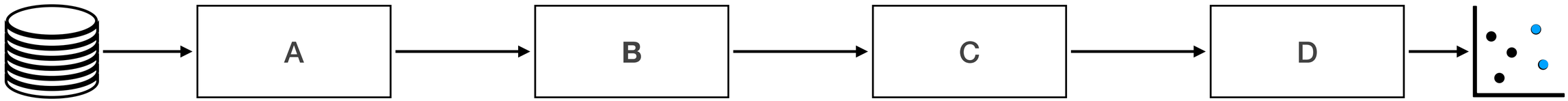
Apply the workflow in a **different environment**

Workflows

Workflows



Workflows



Sequence of tasks

Workflows

Computational Environment

```
gapminder %>%
    select(country, lifeExp, gdpPercap) %>%
    filter(country == 'India')
```

THE UNIVERSITY OF UTAH

Workflows

Interactive Visualizations

?

Workflow Creation

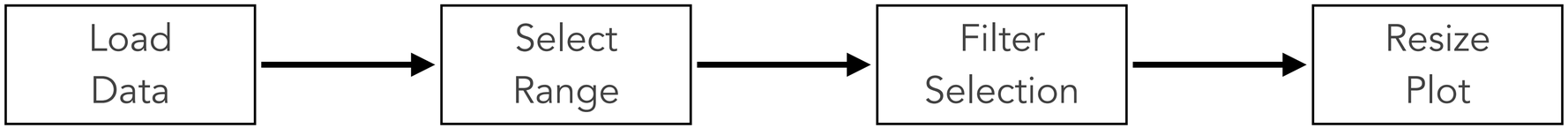
Workflow Creation

Explicit Modeling

Process-based

Explicit Modeling

Specify each task individually and specify their sequence



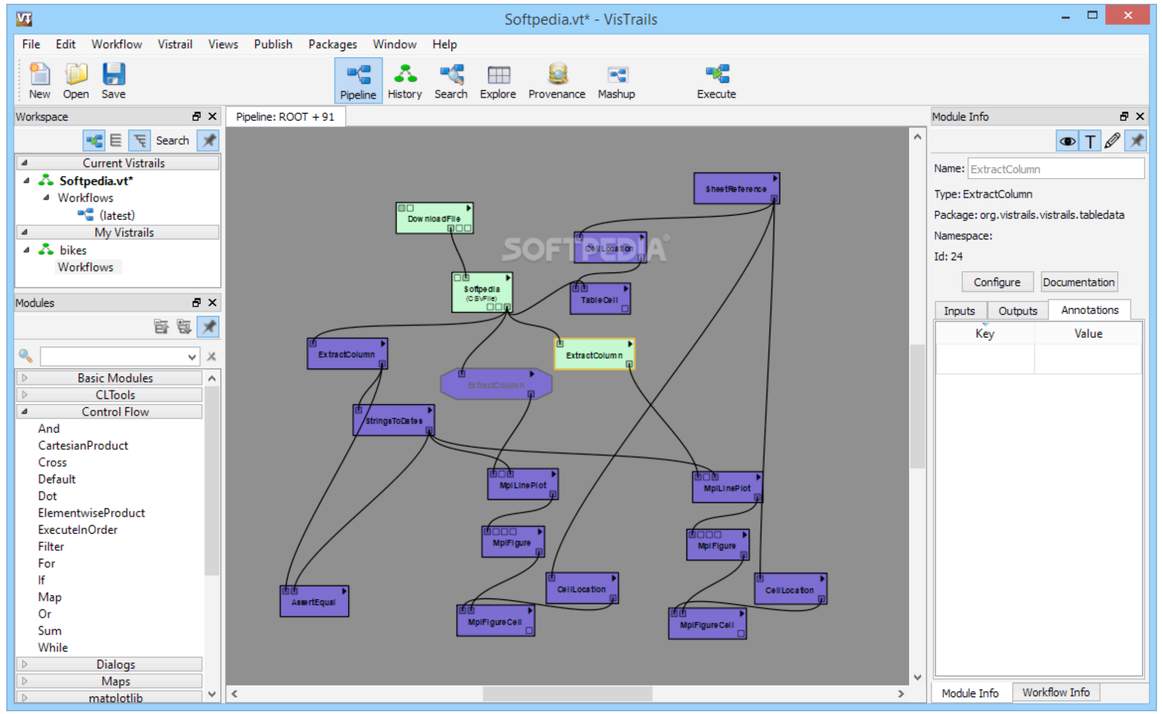
Similar to programming

Does not support rapid exploration

THE UNIVERSITY OF UTAH

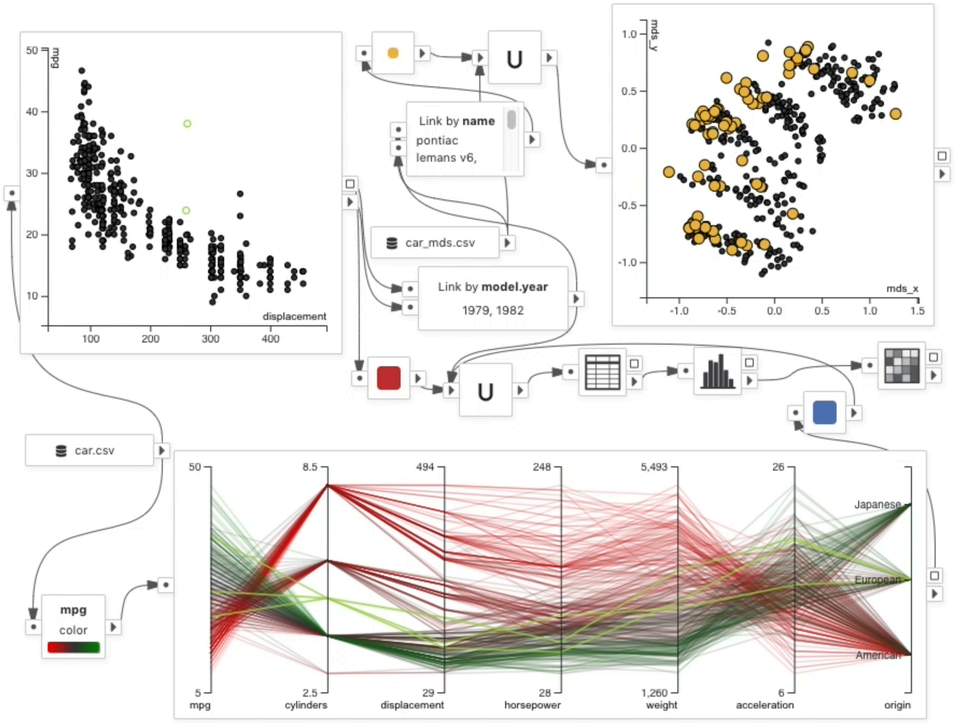
Explicit Modeling

VisTrails



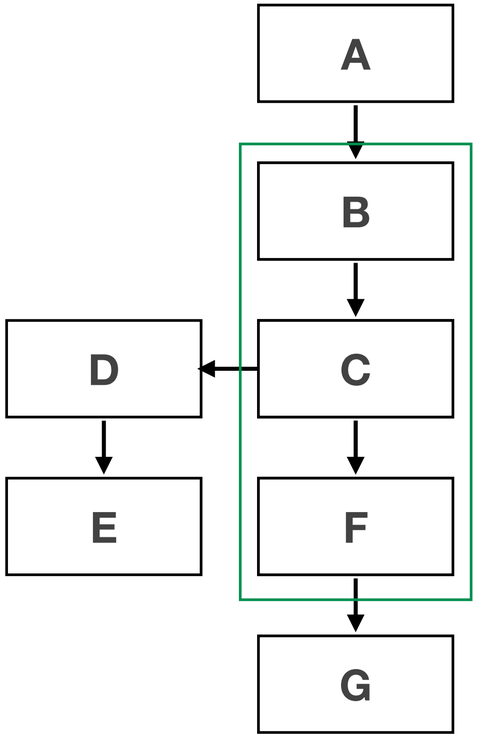
[Bavoil et. al., 2005]

VisFlow



[Yu and Silva, 2017]

Process-based



Captured analysis

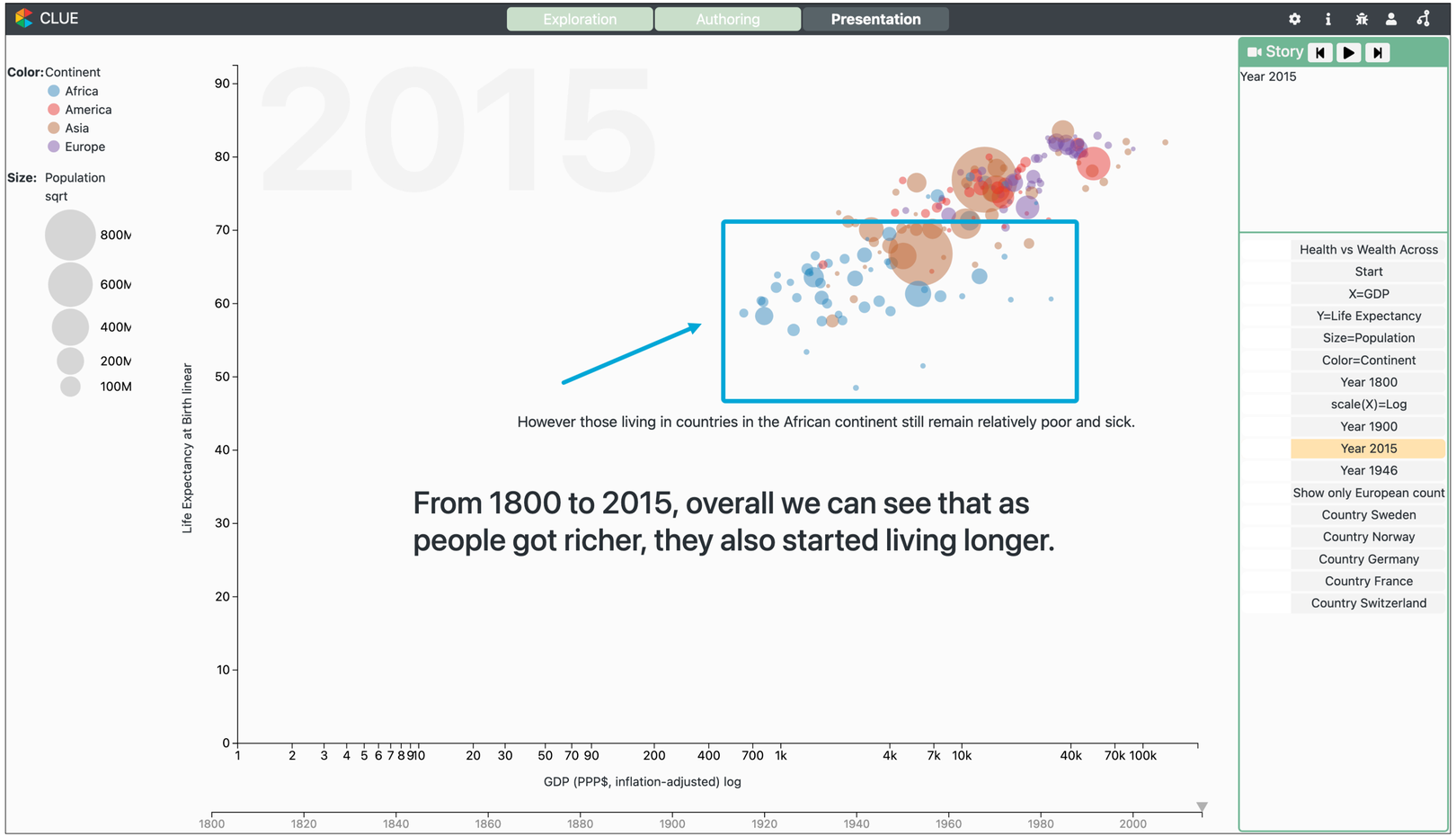
Curated workflow

Explore the data

After finding: Leverage analysis provenance to curate a workflow

Process-based

Vistories



[Gratzl et. al., 2016]

Process-based

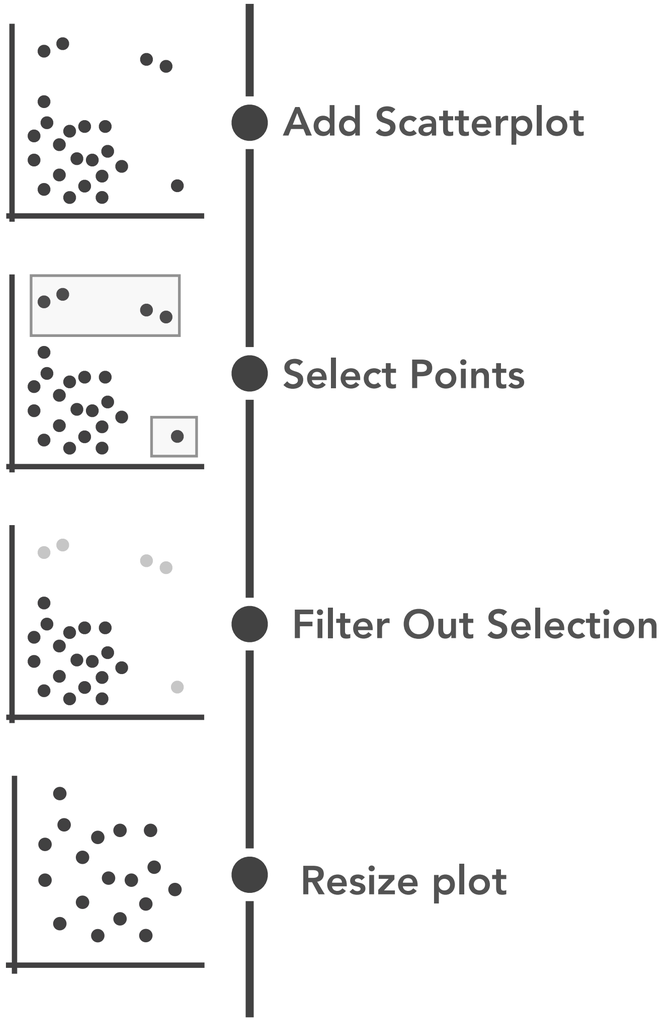
Easy & Natural

Freeform unencumbered exploration

Kiran Gadhave

THE UNIVERSITY OF UTAH

Capturing Workflows

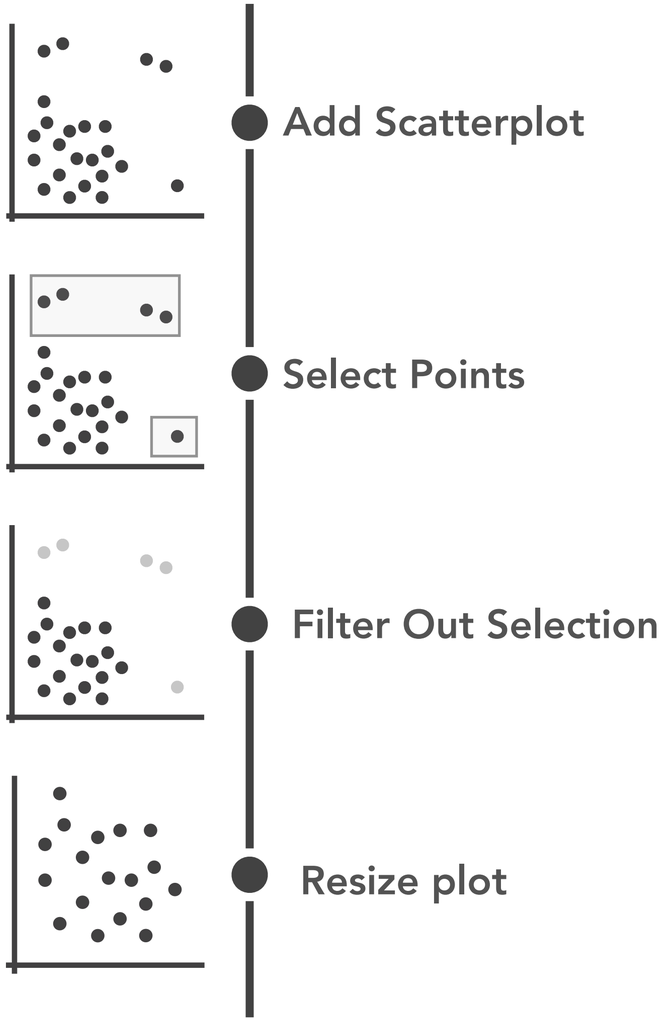


Captured Analysis

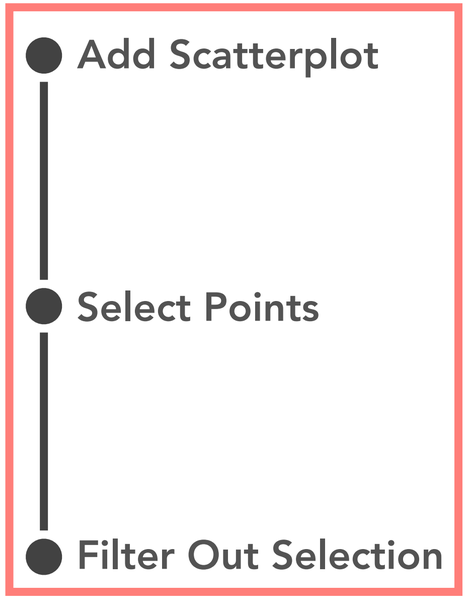
Kiran Gadhave

THE UNIVERSITY OF UTAH

Capturing Workflows



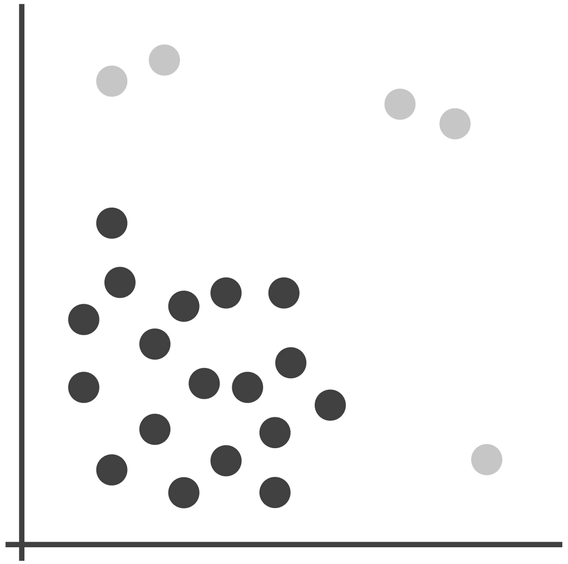
Captured Analysis



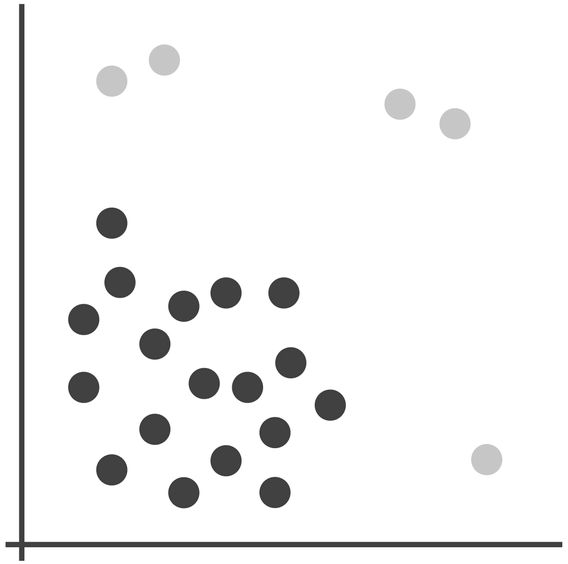
Filter Outliers Workflow

Such workflows enable reproducibility

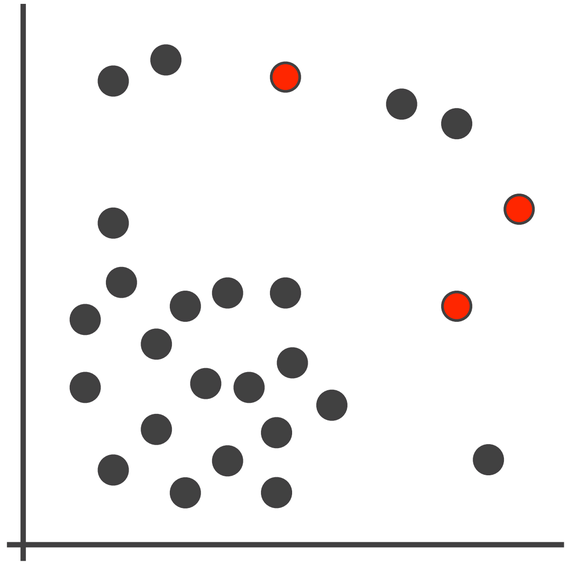
Such workflows enable reproducibility



Such workflows enable reproducibility



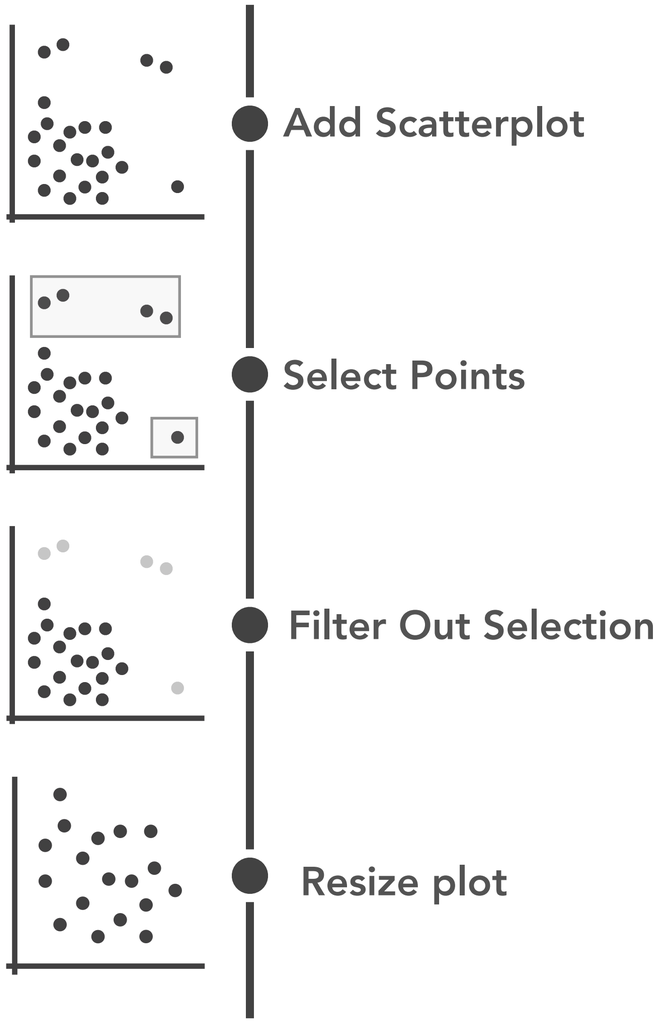
But what if the dataset changes?



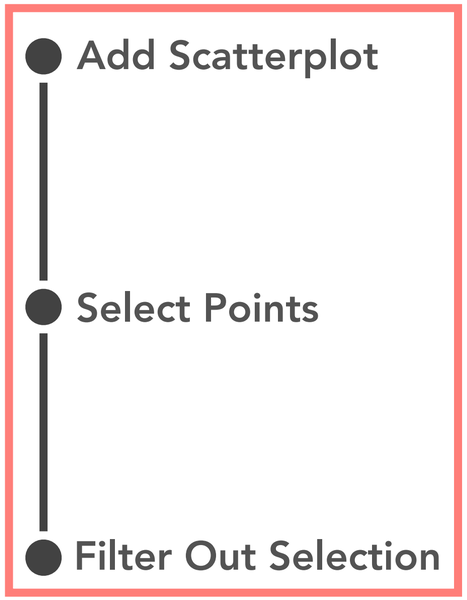
Kiran Gadhave

THE UNIVERSITY OF UTAH

Capturing Workflows



Captured Analysis



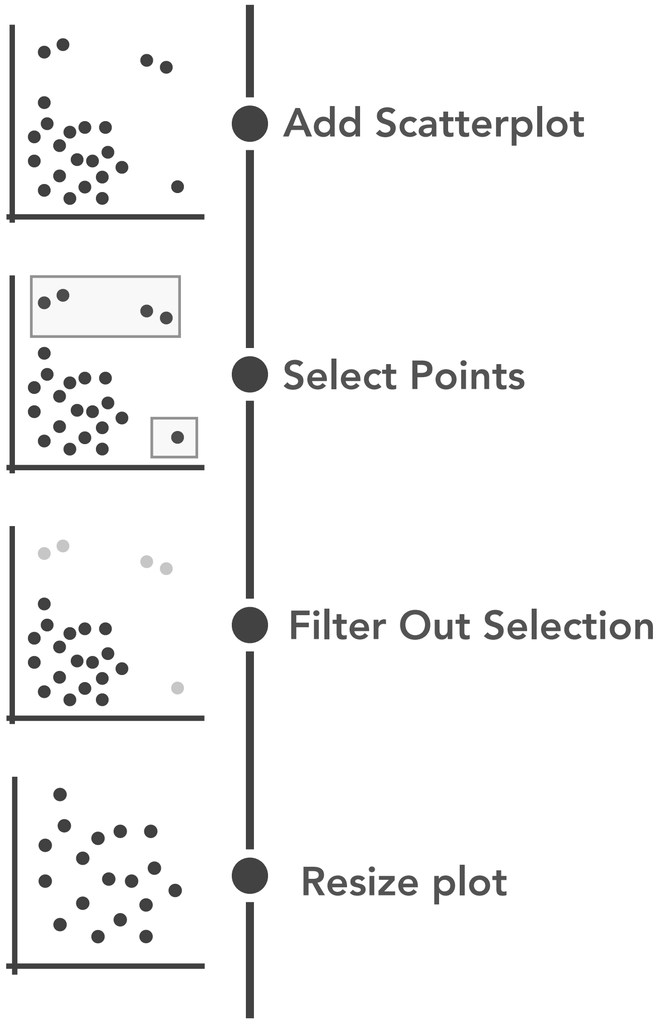
Filter Outliers Workflow

Apply workflow

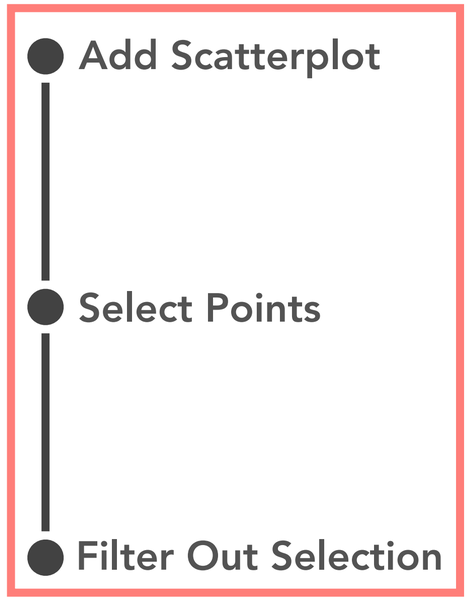
Kiran Gadhave

THE UNIVERSITY OF UTAH

Capturing Workflows



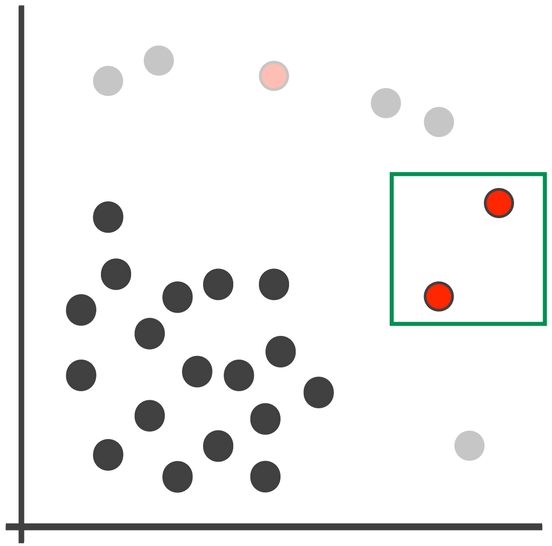
Captured Analysis



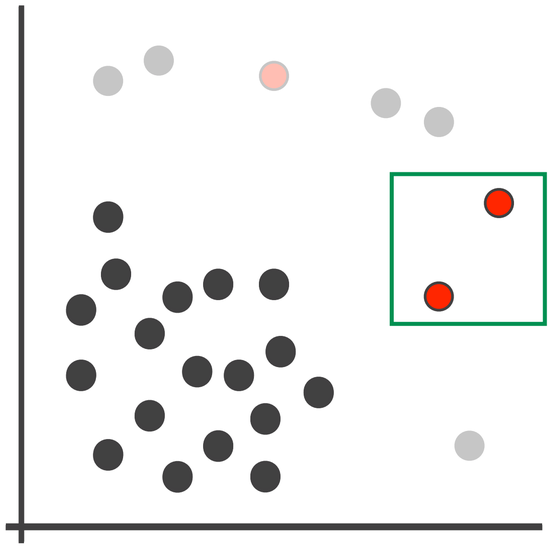
Filter Outliers Workflow

Apply workflow

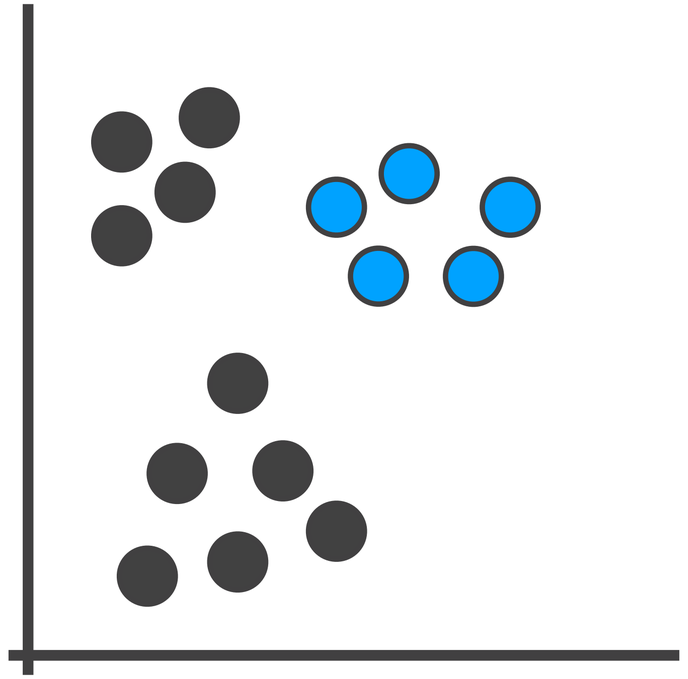
Why?



Why?

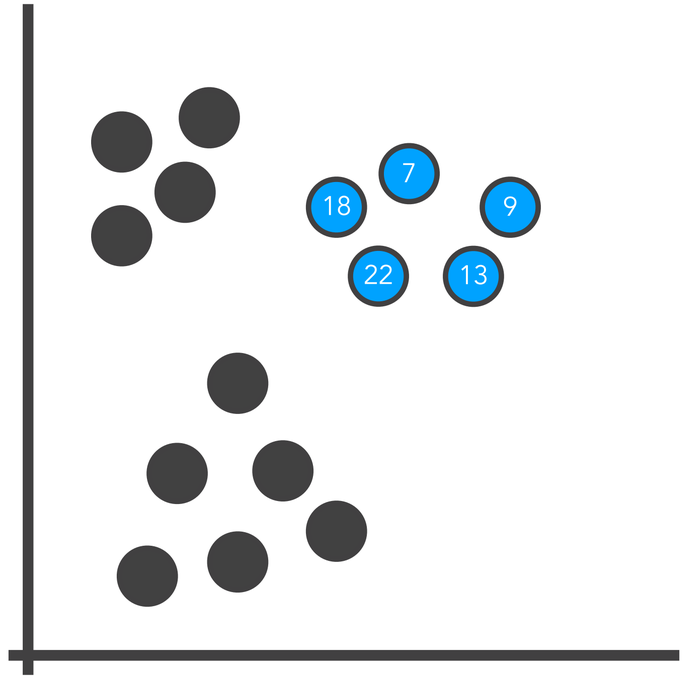


The way selections are captured in the provenance



ID Based Selection:

Selected Elements: 7, 9, 13, 18, 22

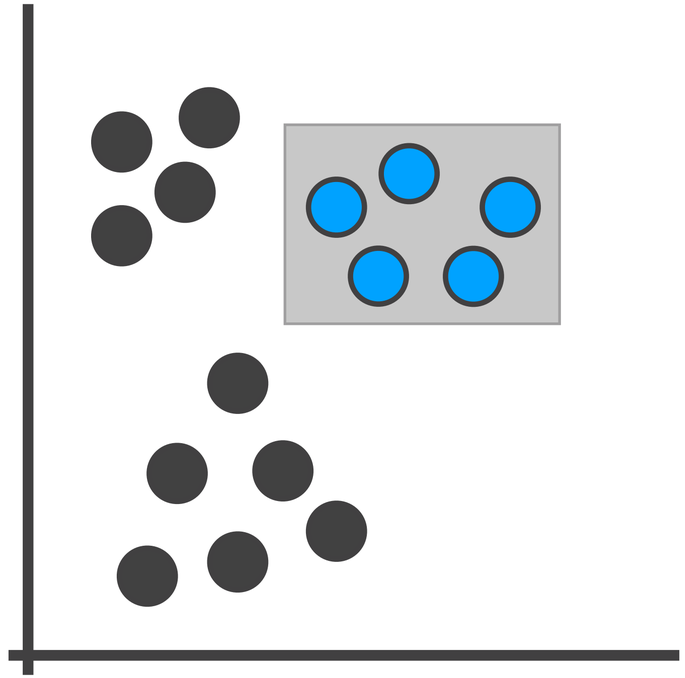


ID Based Selection:

Selected Elements: 7, 9, 13, 18, 22

Range Based Selection:

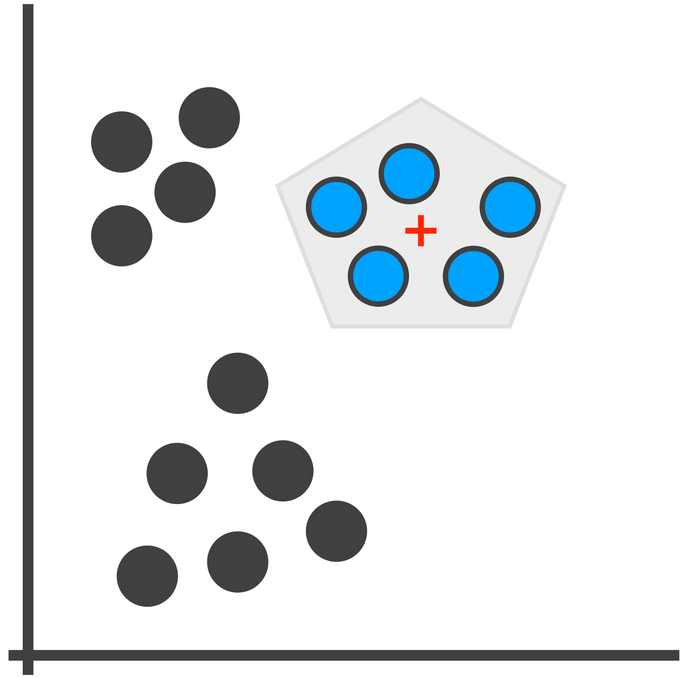
Rectangular area from (1,2) to (5,7)



ID Based Selection:
Selected Elements: 7, 9, 13, 18, 22

Range Based Selection:
Rectangular area from (1,2) to (5,7)

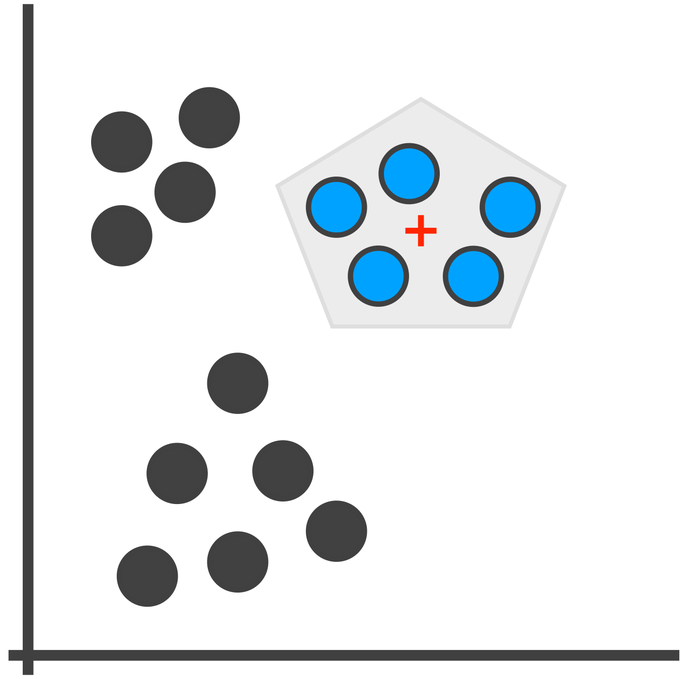
Semantic Selection:
Elements in K-Means cluster centered at [2, 3]



ID Based Selection:
Selected Elements: 7, 9, 13, 18, 22

Range Based Selection:
Rectangular area from (1,2) to (5,7)

Semantic Selection:
Elements in K-Means cluster centered at [2, 3]

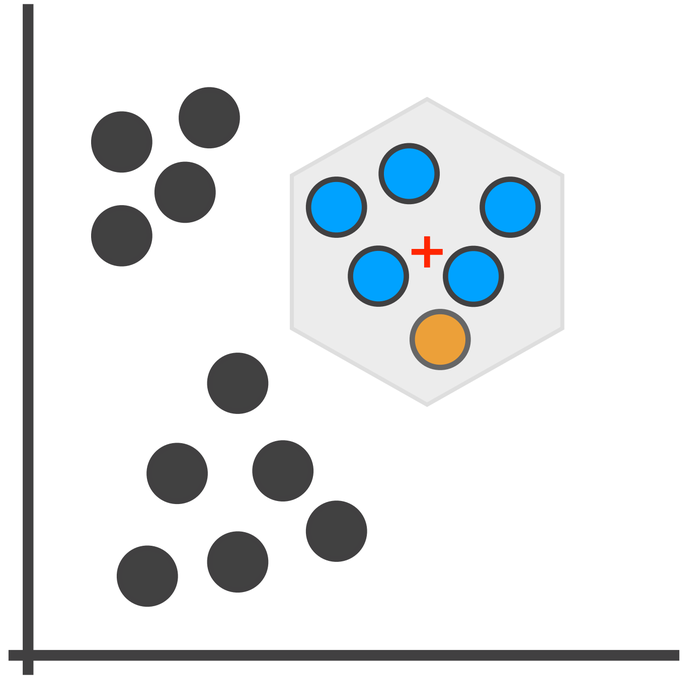


Meaningful, higher level concept:
improves reproducibility

ID Based Selection:
Selected Elements: 7, 9, 13, 18, 22

Range Based Selection:
Rectangular area from (1,2) to (5,7)

Semantic Selection:
Elements in K-Means cluster centered at [2, 3]



Meaningful, higher level concept:
improves reproducibility

Robust to changes and updates in dataset:
enables re-usability

Info Vis

Article

Information Visualization
2021, Vol. 20(4) 207–228

DOI: 10.1177/14738716211038604
journals.sagepub.com/home/ivi
SAGE

Predicting intent behind selections in scatterplot visualizations

Kiran Gadhave[1], Jochen Görtler[2], Zach Cutler[1], Carolina Nobre[3], Oliver Deussen[2], Miriah Meyer[1], Jeff M. Phillips[1] and Alexander Lex[1]

Abstract

Predicting and capturing an analyst's intent behind a selection in a data visualization is valuable in two scenarios: First, a successful prediction of a pattern an analyst intended to select can be used to auto-complete a partial selection which, in turn, can improve the correctness of the selection. Second, knowing the intent behind a selection can be used to improve recall and reproducibility. In this paper, we introduce methods to infer analyst's intents behind selections in data visualizations, such as scatterplots. We describe intents based on patterns in the data, and identify algorithms that can capture these patterns. Upon an interactive selection, we compare the selected items with the results of a large set of computed patterns, and use various ranking approaches to identify the best pattern for an analyst's selection. We store annotations and the metadata to reconstruct a selection, such as the type of algorithm and its parameterization, in a provenance graph. We present a prototype system that implements these methods for tabular data and scatterplots. Analysts can select a prediction to auto-complete partial selections and to seamlessly log their intents. We discuss implications of our approach for reproducibility and reuse of analysis workflows. We evaluate our approach in a crowd-sourced study, where we show that auto-completing selection improves accuracy, and that we can accurately capture pattern-based intent.

Keywords

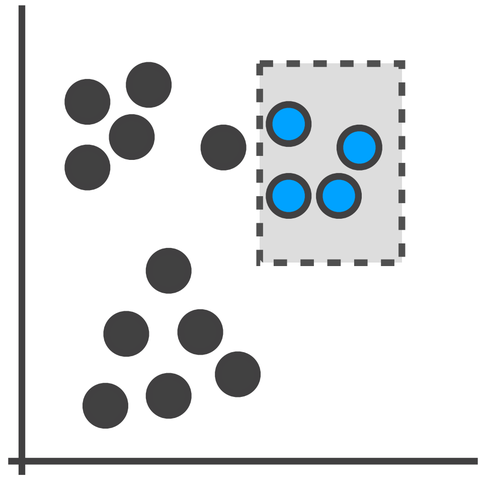
Provenance, reproducibility, intent, brushing, selections

Introduction

When experts interact with a visual analysis system, they are frequently guided by a domain-specific analysis question, such as identifying a gene that could be a drug target. To answer this question, they execute a series of intermediate tasks, such as selecting a set of correlated items for detailed analysis. In contrast to the high-level goal of answering a domain-specific question, these intermediate tasks are based on patterns in

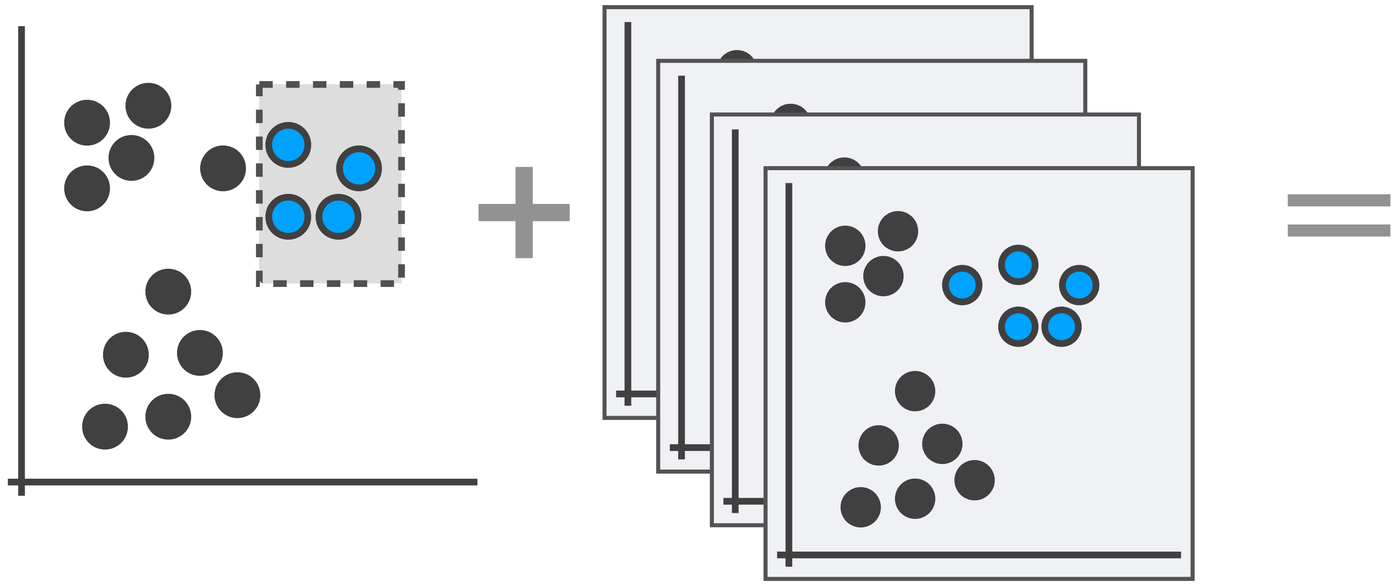
are distinct from higher level intents in that they are free of context and based solely on the data. They are also distinct from low-level intents, such as hovering over an item to read its label. In this paper, we introduce methods to infer these pattern-based intents for brushes in scatterplots. *We define pattern-based intents as the reasoning behind selections based on statistical patterns*

Capturing Intent



Selection

Capturing Intent



Selection

Predictions

K-Means
DBScan
Regression
Outlier Detection
Skyline
Decision Trees / Ranges
Categories

Capturing Intent

Selection

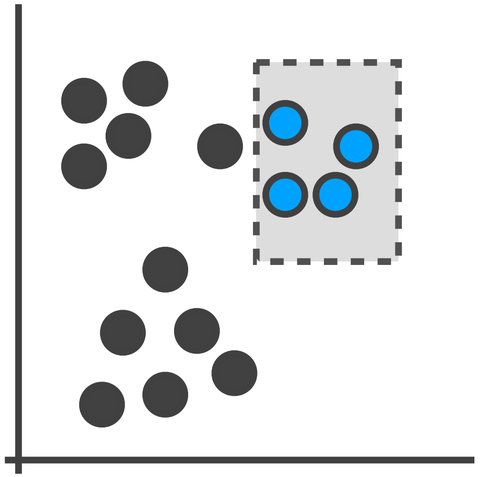
Predictions

K-Means
DBScan
Regression
Outlier Detection
Skyline
Decision Trees / Ranges
Categories

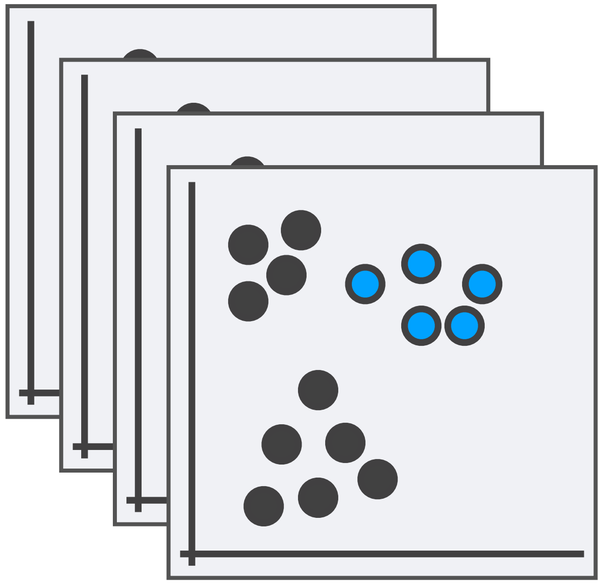
Ranking

Jaccard Distance
Naive Bayes
Classifier
Heuristic Measures

Capturing Intent



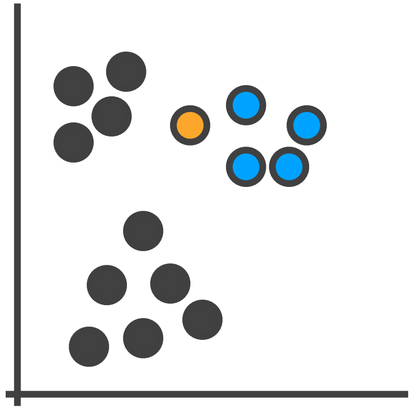
Selection



Predictions

K-Means
DBScan
Regression
Outlier Detection
Skyline
Decision Trees / Ranges
Categories

1. Range
2. Cluster
3. Outlier

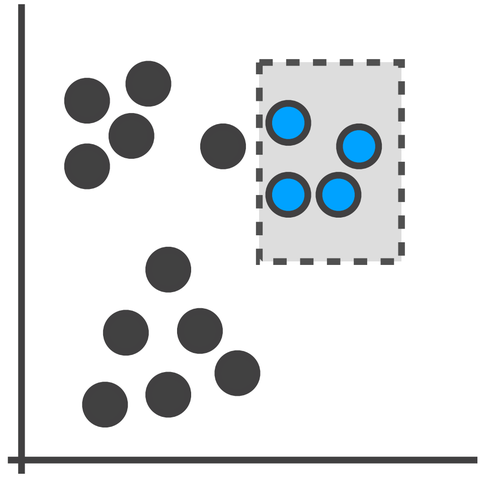


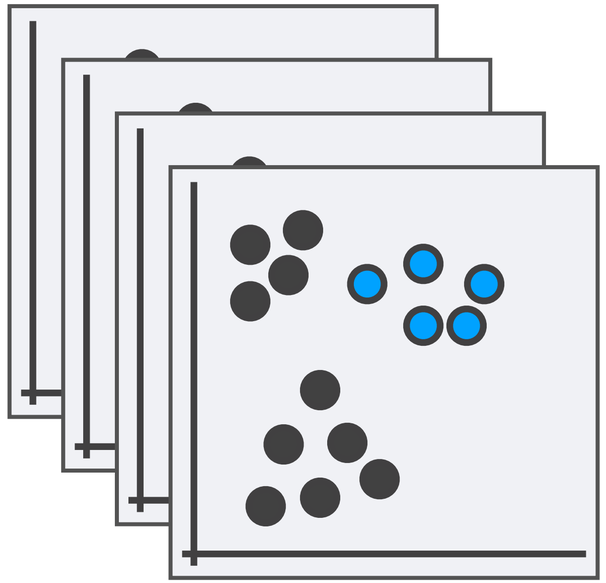
Ranking

Jaccard Distance
Naive Bayes
Classifier
Heuristic Measures

THE UNIVERSITY OF UTAH

Capturing Intent





1. Range
2. Cluster
3. Outlier

I think this cluster…

Selection

Predictions

K-Means
DBScan
Regression
Outlier Detection
Skyline
Decision Trees / Ranges
Categories

Ranking

Jaccard Distance
Naive Bayes
Classifier
Heuristic Measures

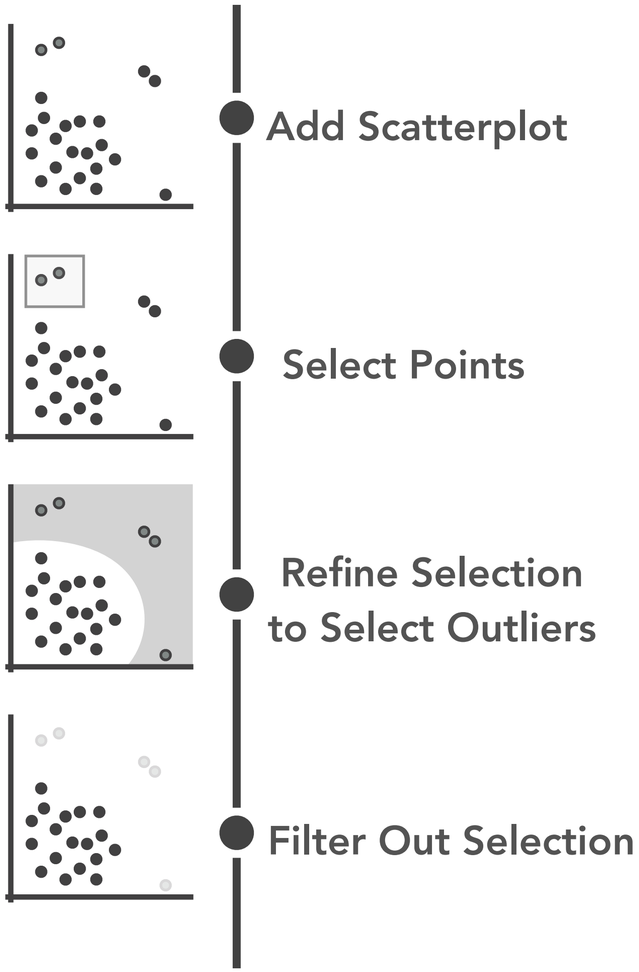
Confirming Intent & Annotation

Capturing Reusable Workflow

Kiran Gadhave

THE UNIVERSITY OF UTAH

Capturing Reusable Workflow

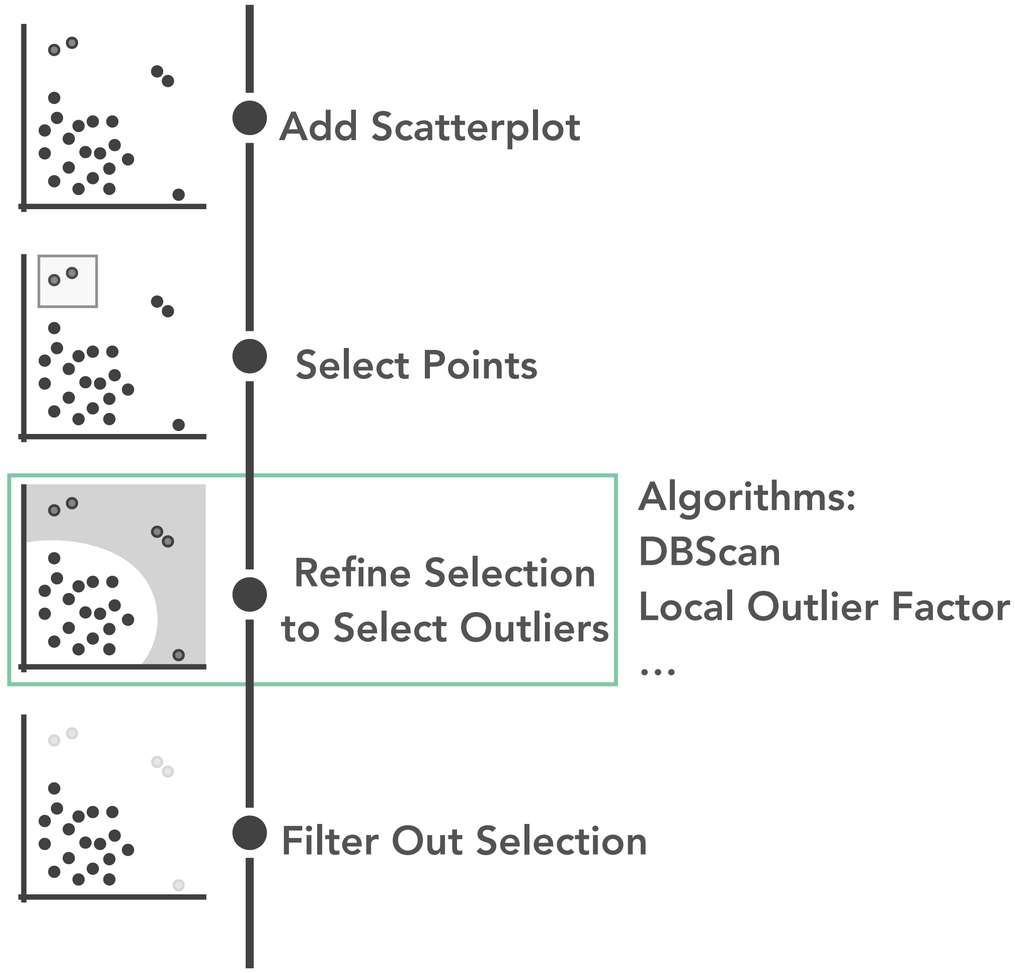


Captured Analysis

Kiran Gadhave

THE UNIVERSITY OF UTAH

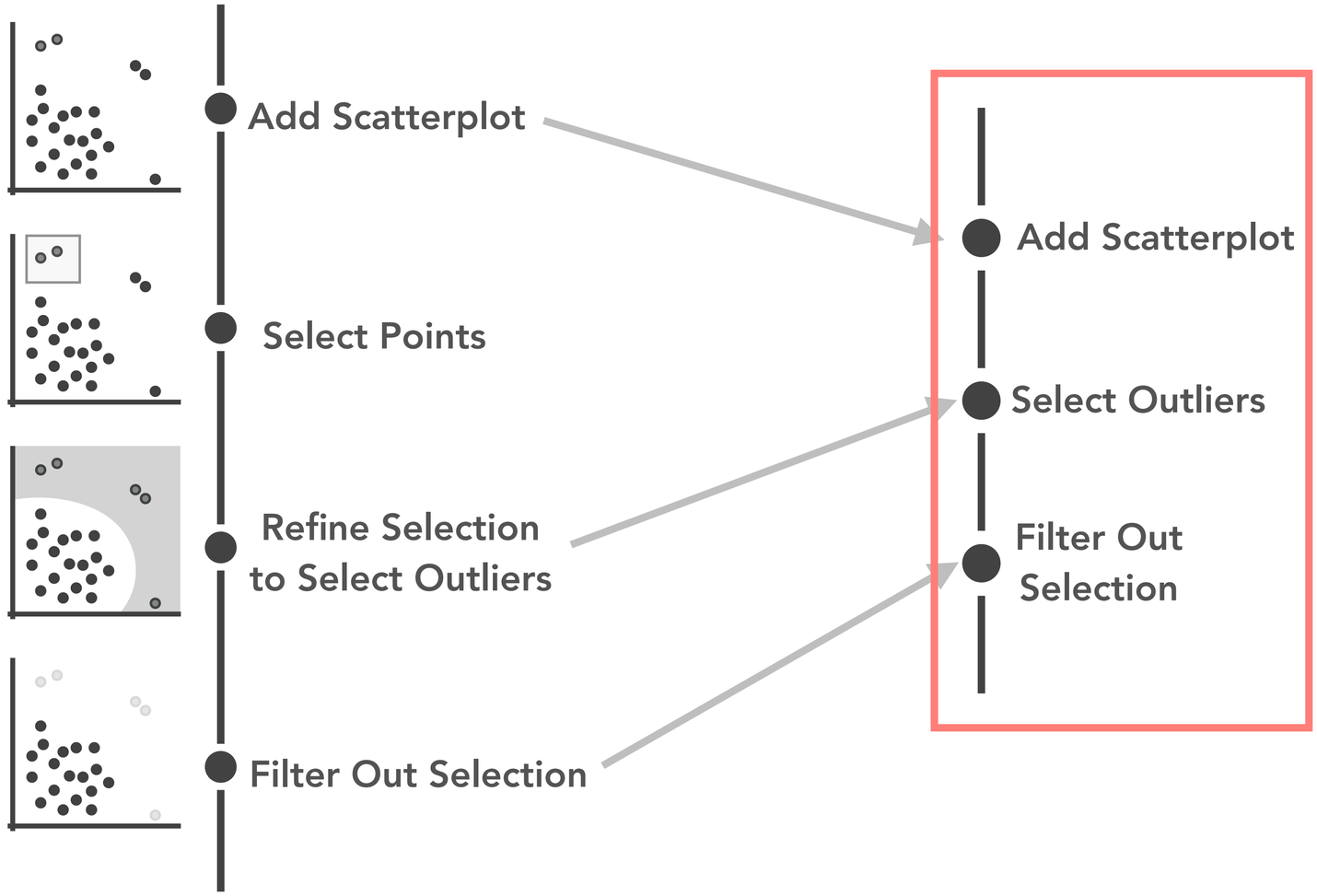
Capturing Reusable Workflow



Algorithms:
DBScan
Local Outlier Factor
...

Captured Analysis

Capturing Reusable Workflow



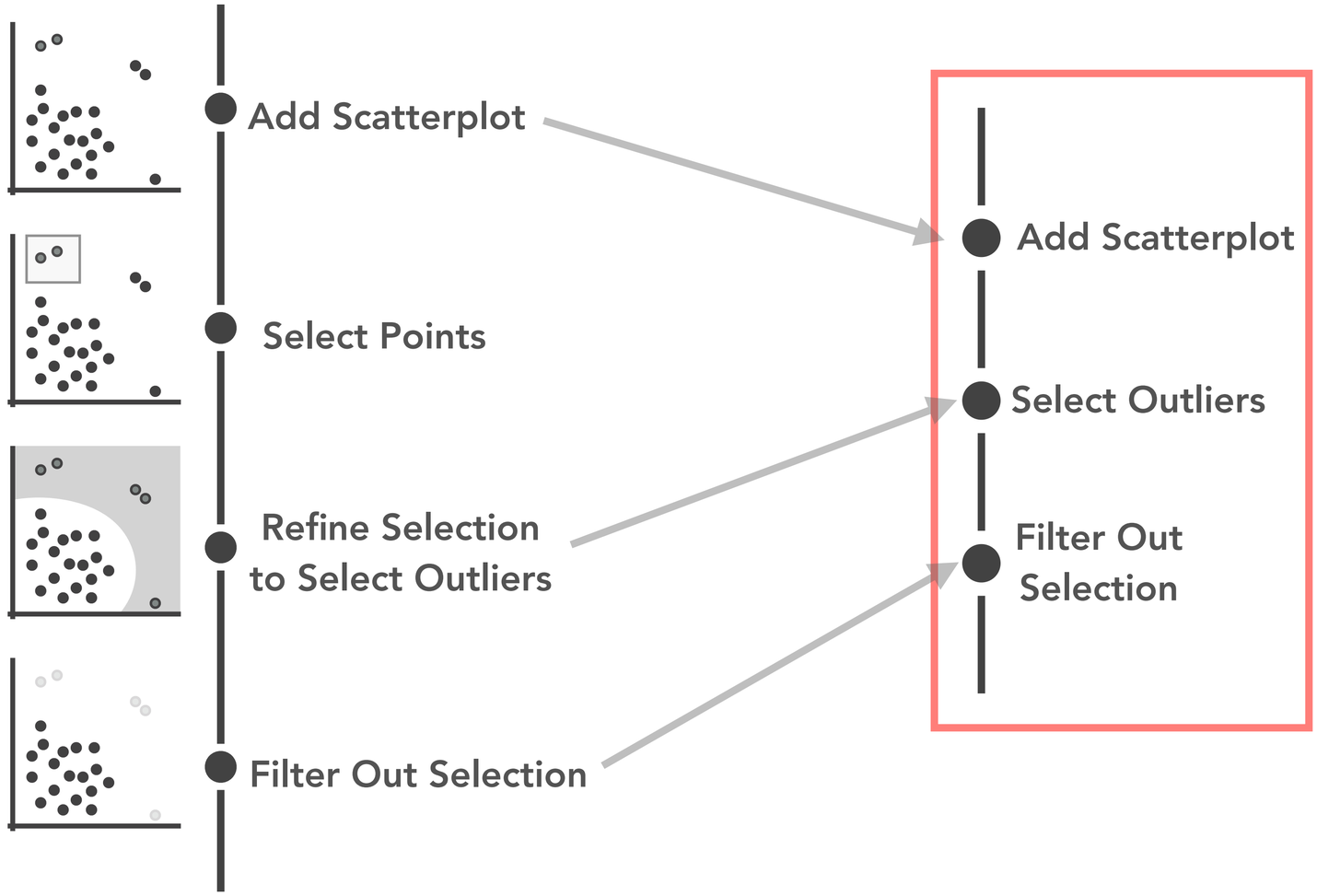
Captured Analysis

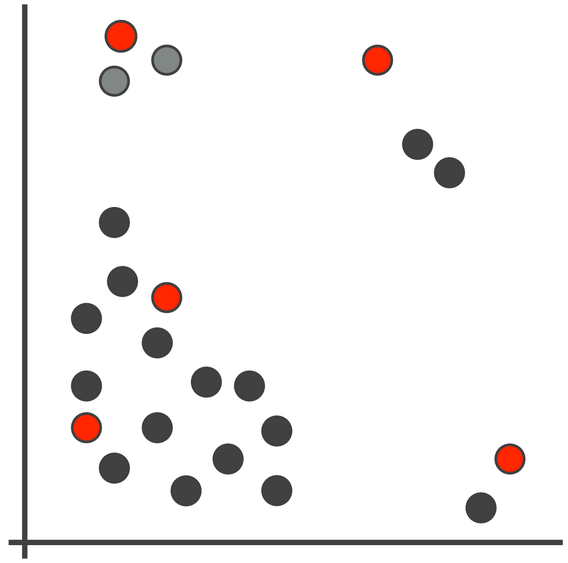
Semantic Filter Outlier Workflow

Kiran Gadhave

THE UNIVERSITY OF UTAH

Capturing Reusable Workflow





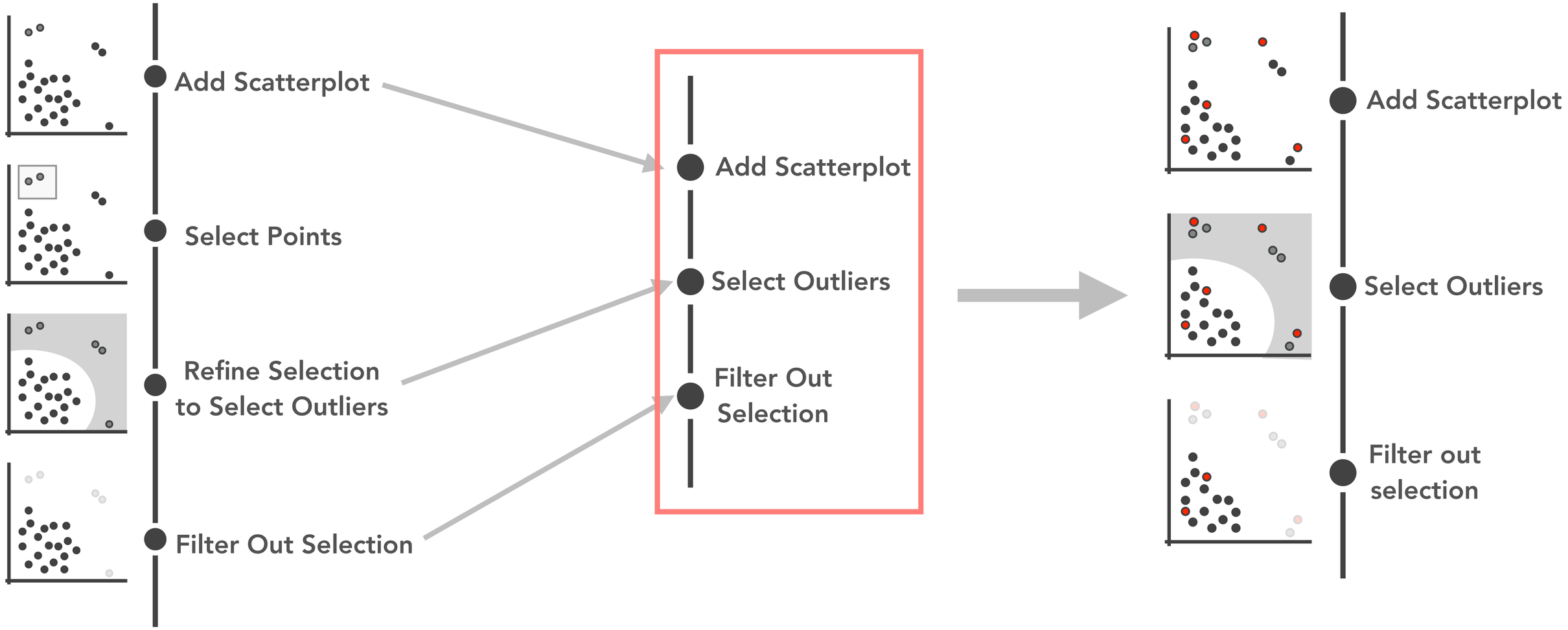
Updated Dataset

Captured Analysis

Semantic Filter Outlier Workflow

THE UNIVERSITY OF UTAH

Capturing Reusable Workflow

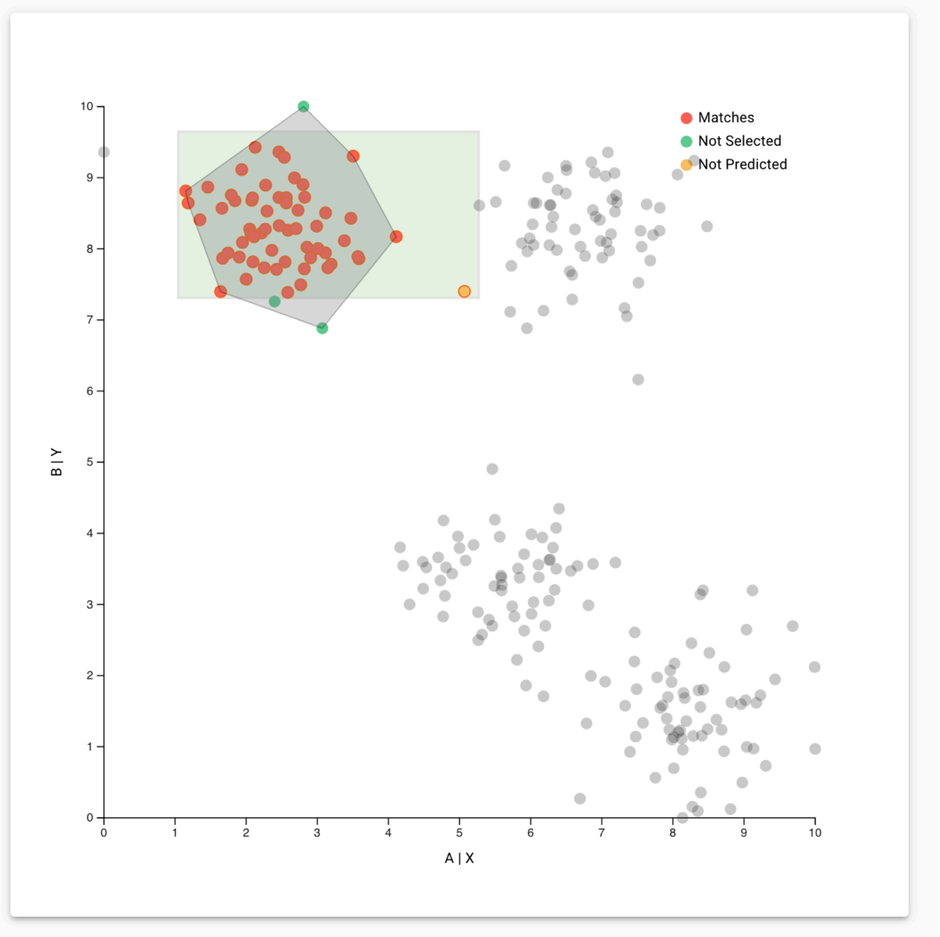


Captured Analysis

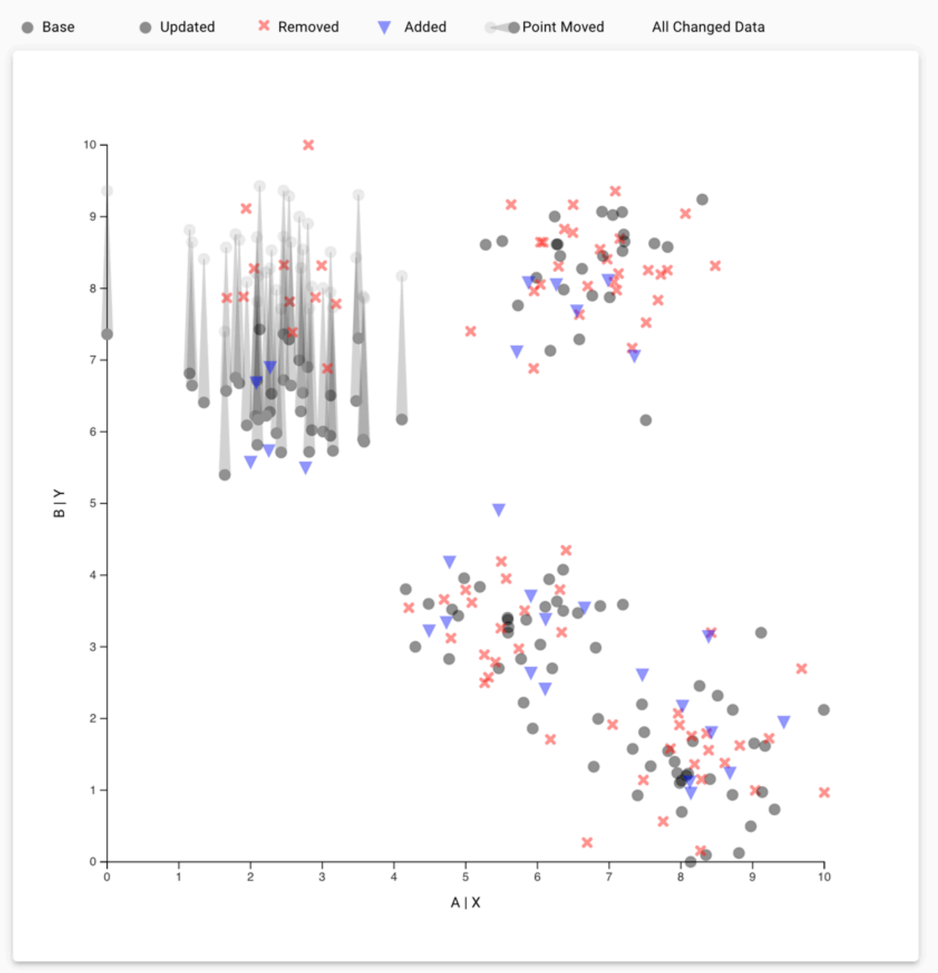
Semantic Filter Outlier Workflow

Apply Semantic Workflow

Example: Selecting a cluster

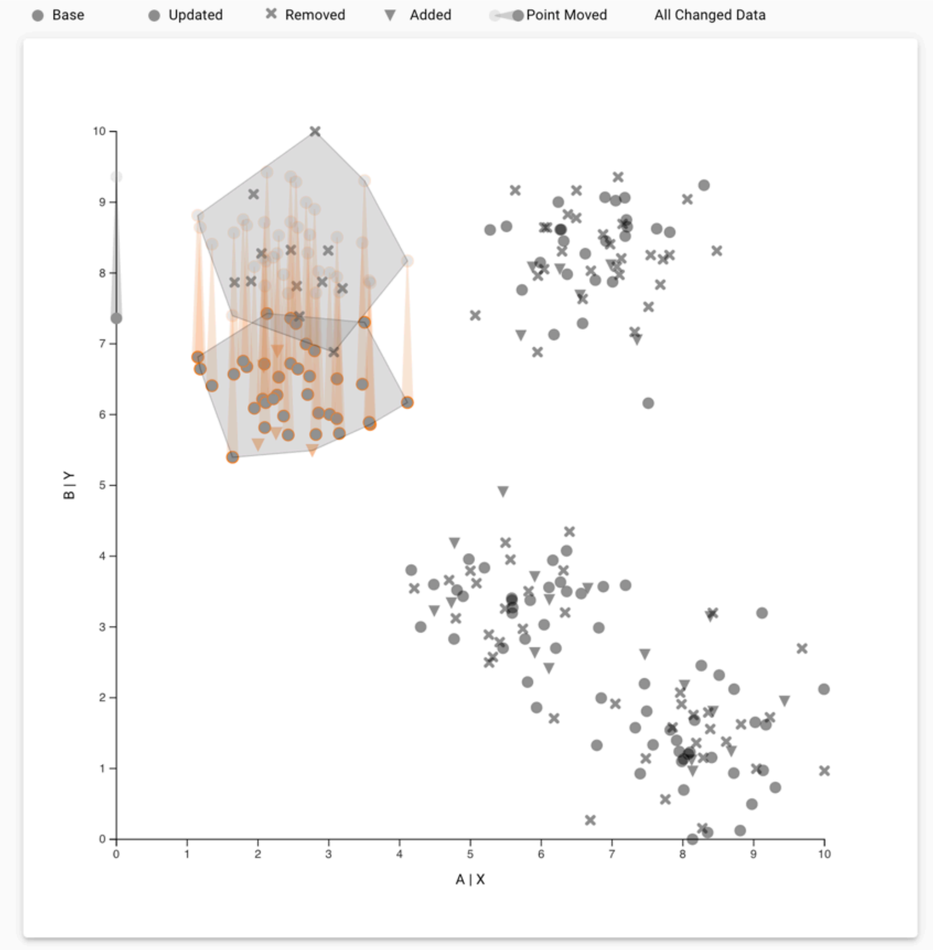


Example: Selecting a cluster

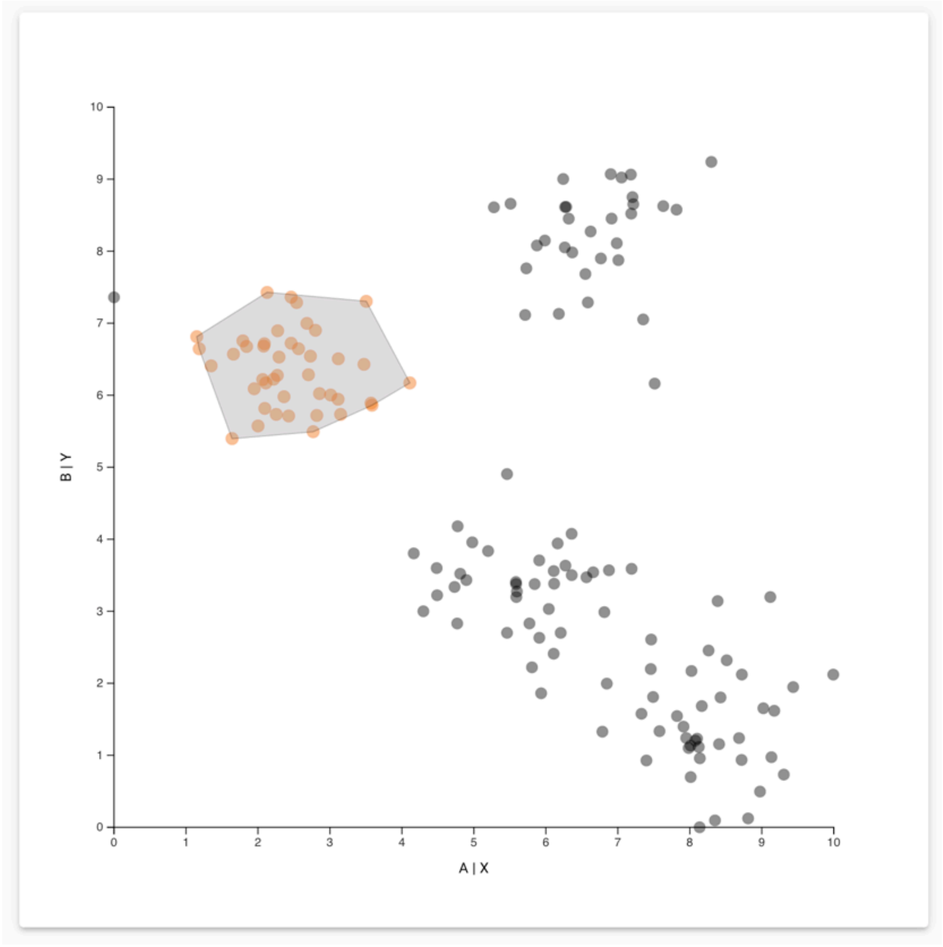


Dataset updates

Comparing the selections

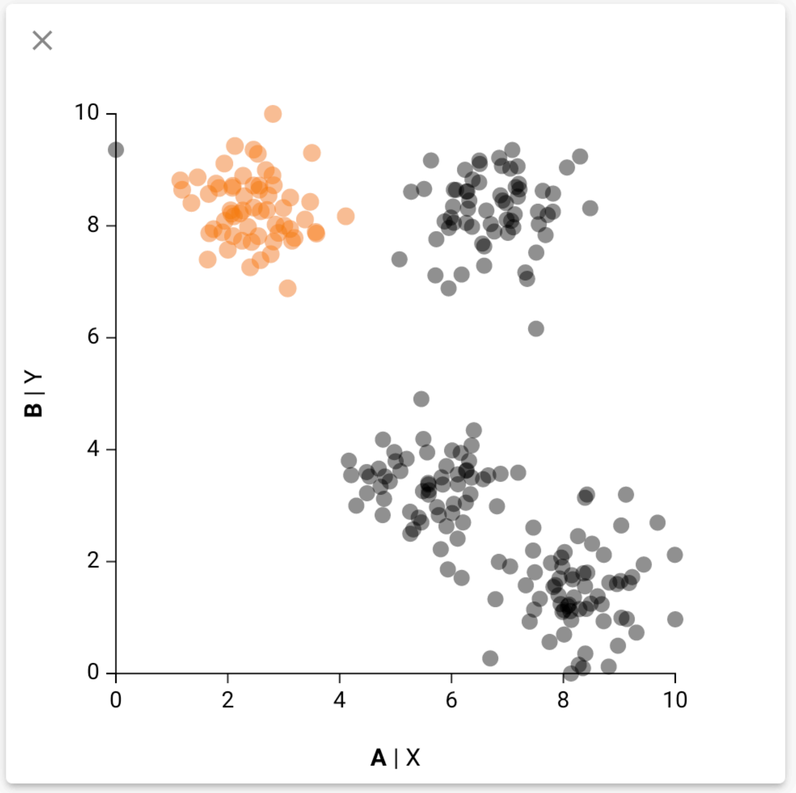


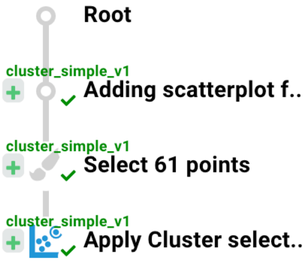
Final selection after automatic reapplication



Reviewing applied workflows

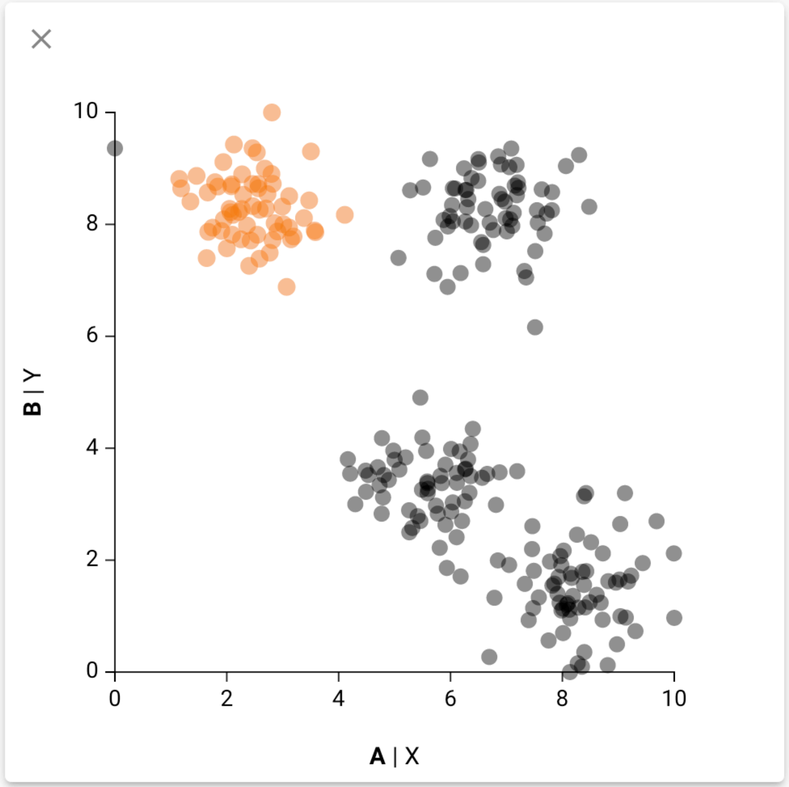
Original cluster selection

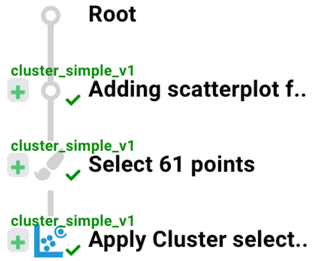




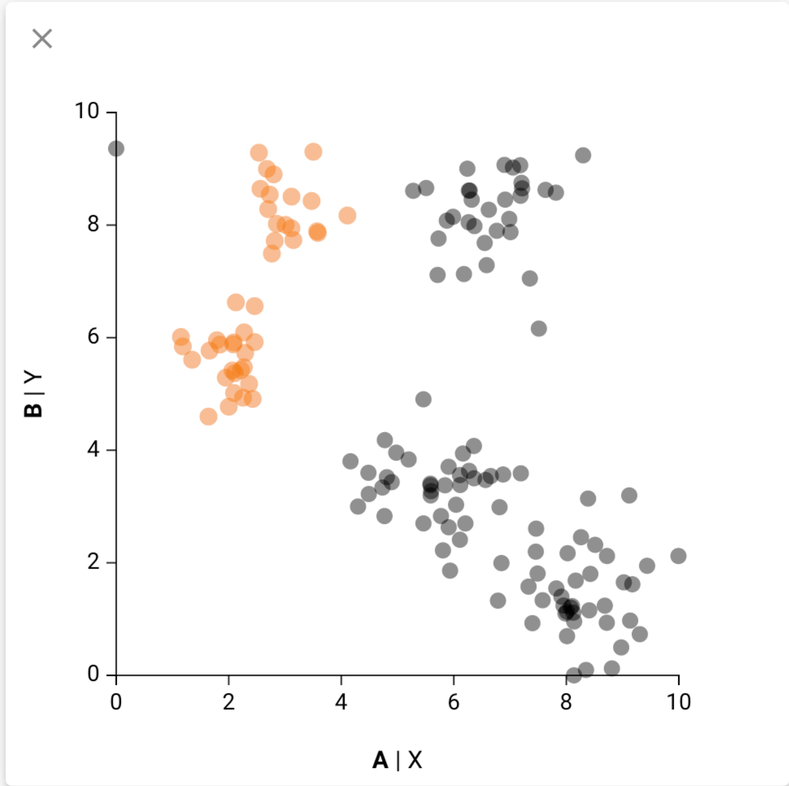
Reviewing applied workflows

Original cluster selection





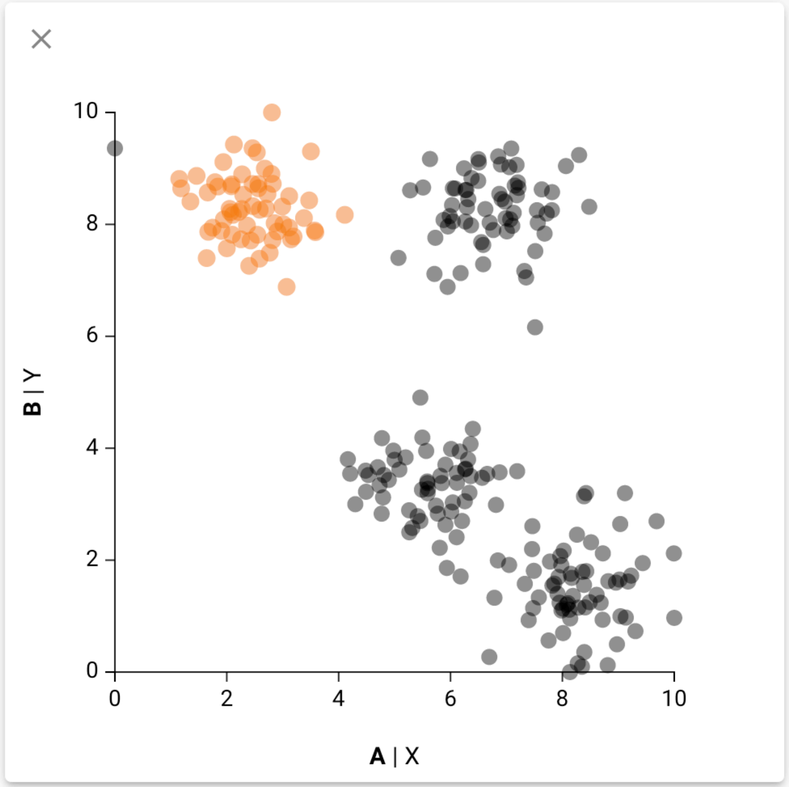
Broken cluster

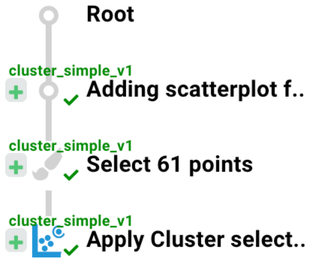


THE UNIVERSITY OF UTAH

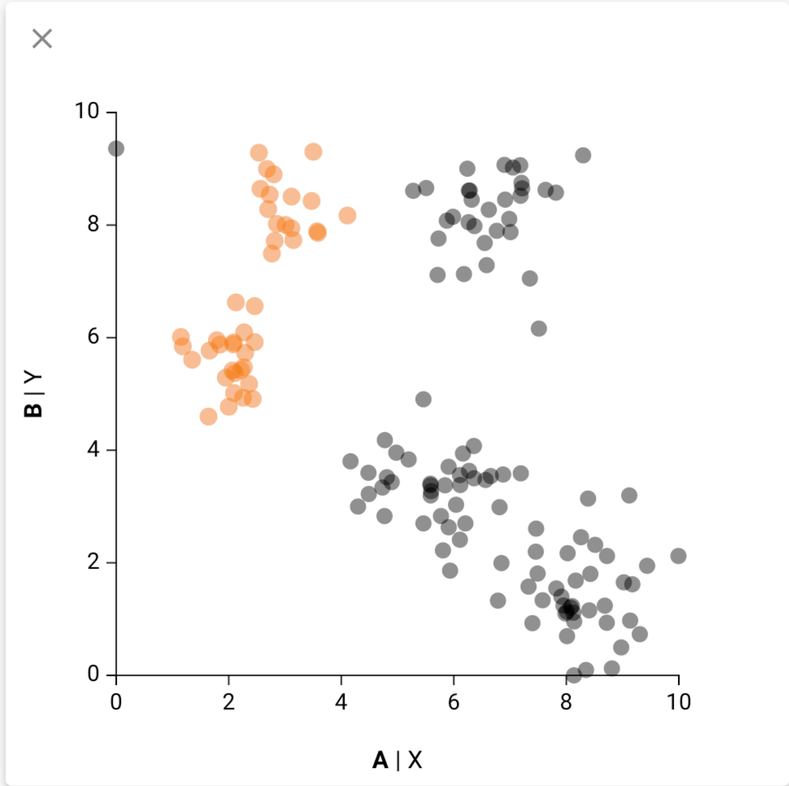
Reviewing applied workflows

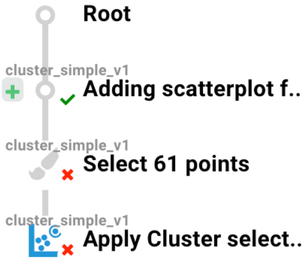
Original cluster selection





Broken cluster





Kiran Gadhave

THE UNIVERSITY OF UTAH

Demo

Brush Type

Transforms

FILTER

LABEL

AGGREGATE

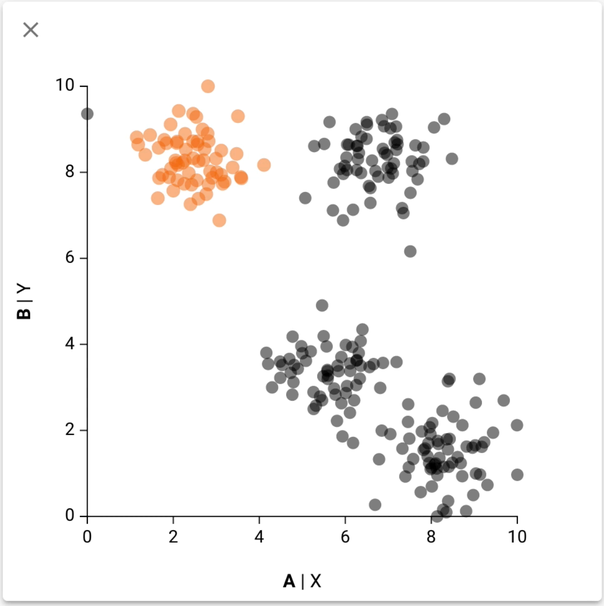
Categories

Enable category encoding

No category column in the dataset!

cluster_simple_v6 | cluster_simple_v5 | cluster_simple_v4 | cluster_simple_v1 | cluster_simple_v2 | cluster_simple_v3

Compare



CLUSTER (1.000)

CLUSTER (0.984)

CLUSTER (0.984)

CLUSTER (0.968)

CLUSTER (0.968)

CLUSTER (0.934)

OUTLIER (0.923)

CLUSTER (0.918)

CLUSTER (0.754)

CLUSTER (0.672)

CLUSTER (0.574)

CLUSTER (0.492)

CLUSTER (0.419)

POLYNOMIAL REGRES

CLUSTER (0.295)

Graph | Bookmarks/Annotations

Undo | Redo

Root

cluster_simple_v1 Adding scatterplot f..

cluster_simple_v1 Add Brush

cluster_simple_v1 Update Brush

cluster_simple_v1 Apply Cluster select..

Kiran Gadhave

THE UNIVERSITY OF UTAH

gapminder_2010 | gapminder_2000 | gapminder_1990 | gapminder_1980 | Compare

Brush Type

Transforms

FILTER

LABEL

AGGREGATE

Categories

Enable category encoding

Showing categories

continent

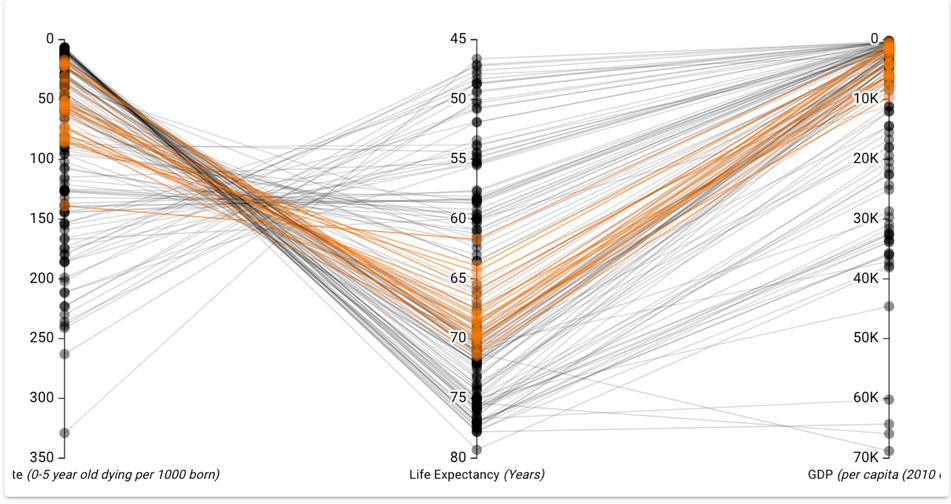
UNASSIGNED

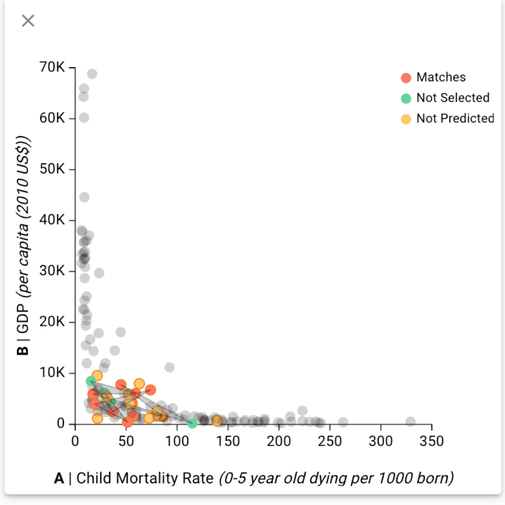
ASIA

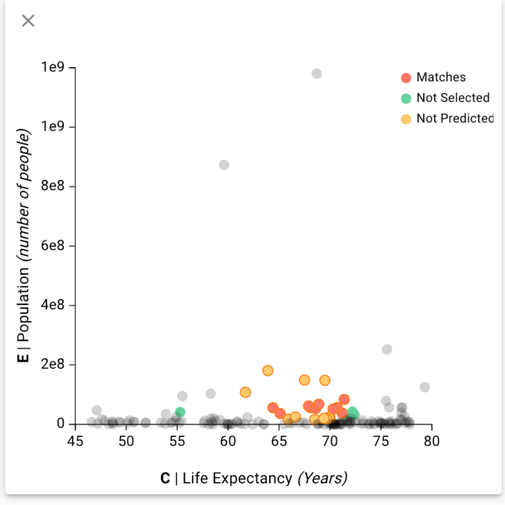
EUROPE

AMERICAS

AFRICA







CLUSTER (0.423)

CLUSTER (0.400)

POLYNOMIAL REGRESSION (0.391)

POLYNOMIAL REGRESSION (0.381)

CLUSTER (0.333)

POLYNOMIAL REGRESSION (0.323)

OUTLIER (0.308)

OUTLIER (0.270)

OUTLIER (0.250)

CLUSTER (0.238)

OUTLIER (0.236)

OUTLIER (0.236)

OUTLIER (0.226)

MULTIVARIATE OPTIMIZATION (0.221)

MULTIVARIATE OPTIMIZATION (0.215)

OUTLIER (0.212)

LINEAR REGRESSION (0.211)

OUTLIER (0.209)

OUTLIER (0.207)

Graph | Bookmarks/Annotatio

Undo | Redo

Root

gapminder_1990 Adding scatterplot f..

gapminder_1990 Adding PCP

gapminder_1990 Adding scatterplot f..

gapminder_1990 Select 21 points

THE UNIVERSITY OF UTAH

Bridging between environments

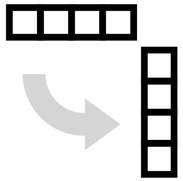
Data analysis rarely takes place in a single tool

THE UNIVERSITY OF UTAH

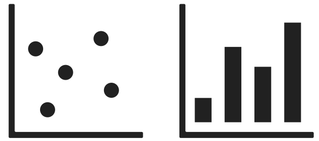
Bridging between environments

Data analysis rarely takes place in a single tool

Raw data exploration



Wrangling



Exploratory analysis

Reporting

Tableau/PowerBI

Jupyter/R notebooks

SQL

Spreadsheets

Bridging between environments

Exporting the transformed data is difficult

THE UNIVERSITY OF UTAH

Bridging between environments

Exporting the transformed data is difficult

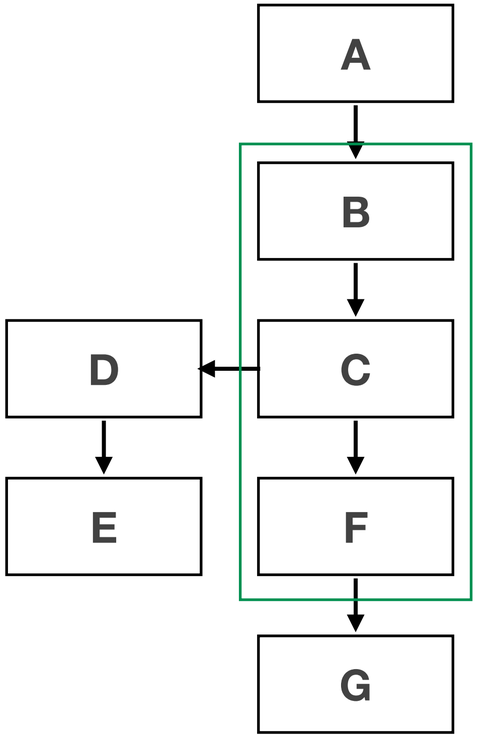
Need to **redo parts of analysis** when switching tools

Contribution

Reuse workflows in a different environment

THE UNIVERSITY OF UTAH

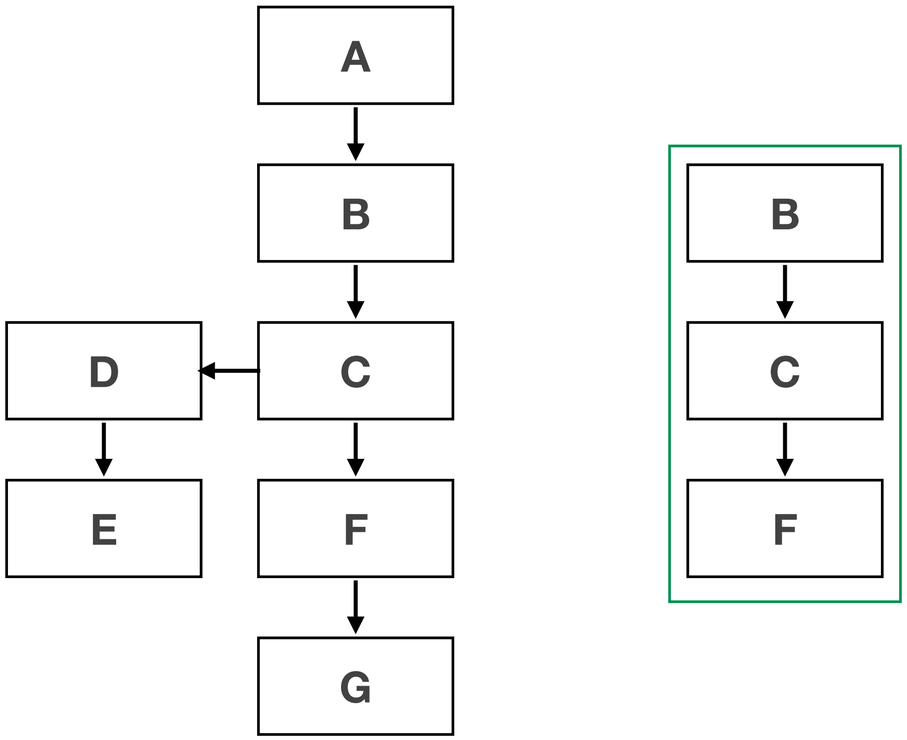
Reusing workflows in Jupyter



Captured analysis

THE UNIVERSITY OF UTAH

Reusing workflows in Jupyter

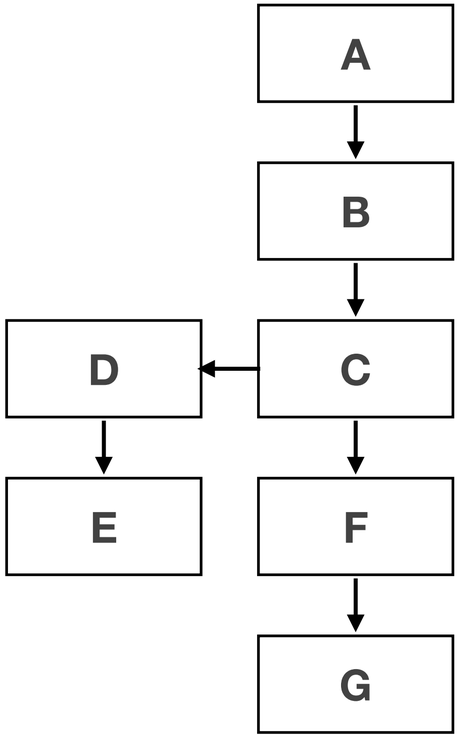


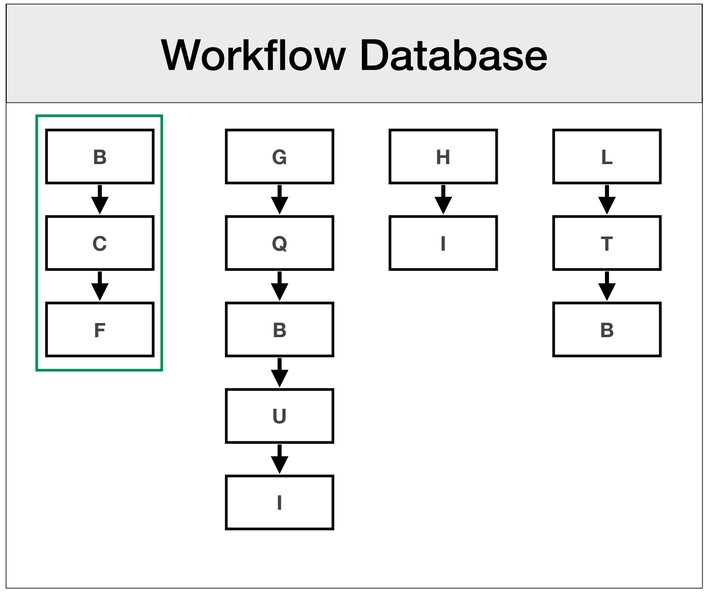
Captured analysis

Curated workflow

THE UNIVERSITY OF UTAH

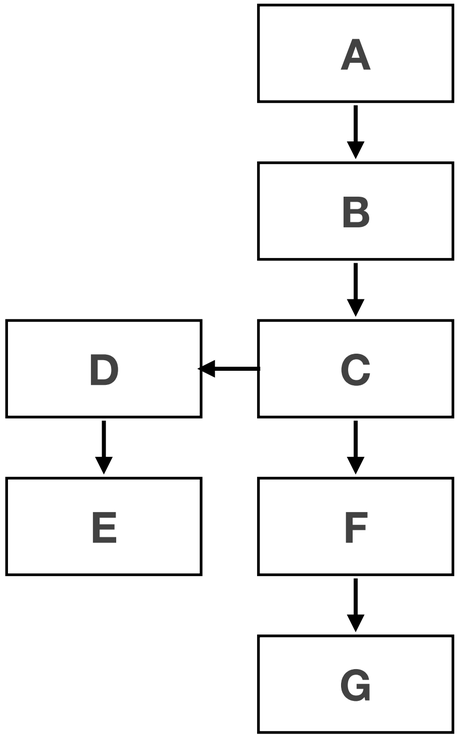
Reusing workflows in Jupyter

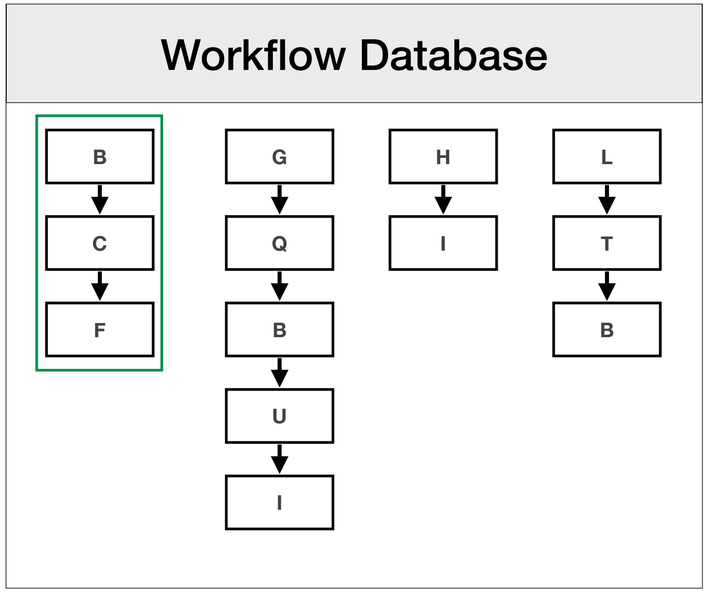




THE UNIVERSITY OF UTAH

Reusing workflows in Jupyter





Jupyter Notebook

```
import workflow

apply workflow

further analysis
```

Reapply Library

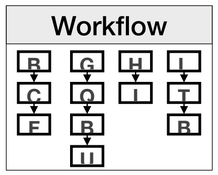
Reapply Library

Python Library

THE UNIVERSITY OF UTAH

Reapply Library

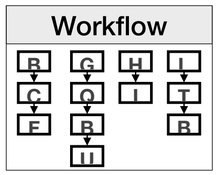
Python Library



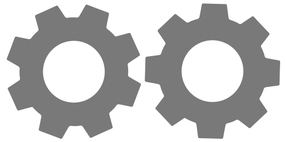
Load workflows

Reapply Library

Python Library



Load workflows

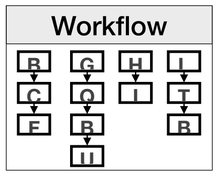


Core logic for capturing and applying workflows

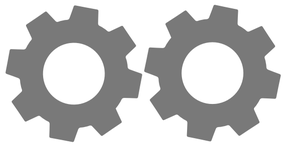
THE UNIVERSITY OF UTAH

Reapply Library

Python Library



Load workflows



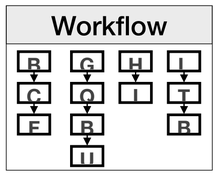
Core logic for capturing and applying workflows

Apply to pandas dataframe

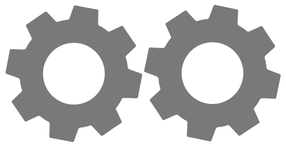
THE UNIVERSITY OF UTAH

Reapply Library

Python Library



Load workflows



Core logic for capturing and applying workflows

Apply to pandas dataframe

Proof-of-concept

Demo

```
# This module exposes the Reapply class
from reapply_workflows import Reapply
```

Demo

```
# This module exposes the Reapply class
from reapply_workflows import Reapply
```

Demo

```
# Initialize the Reapply library
reapply = Reapply()

# Load the Covid Dataset project
project = reapply.load("Covid OWID")
project.list_workflows()
```

```
Categorize outliers - 1638475878304
```

THE UNIVERSITY OF UTAH

Demo

```
# This module exposes the Reapply class
from reapply_workflows import Reapply
```

```
# Initialize the Reapply library
reapply = Reapply()

# Load the Covid Dataset project
project = reapply.load("Covid OWID")
project.list_workflows()
```

```
Categorize outliers - 1638475878304
```

Demo

```
# Get the desired workflow
wf = project.get_workflow("16384758783O4")

# Description of the options in the workflow
wf.describe()
```

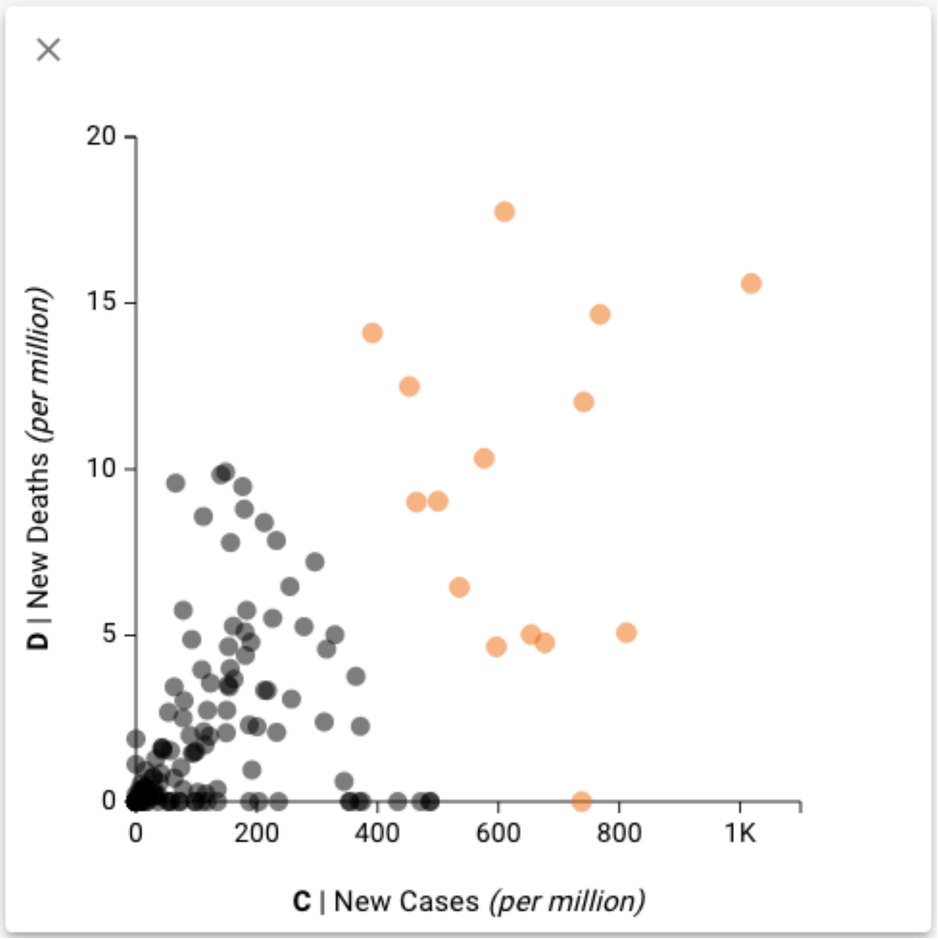
```
Categorize outliers
| Root
 +--| Adding scatterplot for new_cases_per_million-new_deaths_per_million
     +--| Apply Outlier selection
         +--| Filter In
             +--| Add Brush
                 +--| Categorize Selections
                     +--| Add Brush
                         +--| Update Brush
                             +--| Categorize Selections
                                 +--| Add Brush
                                     +--| Categorize Selections
```

THE UNIVERSITY OF UTAH

Demo

```
# Get the desired workflow
wf = project.get_workflow("1638475878304")

# Description of the options in the workflow
wf.describe()
```

```
Categorize outliers
| Root
 +--| Adding scatterplot for new_cases_per_million-new_deaths_per_million
     +--| Apply Outlier selection
         +--| Filter In
             +--| Add Brush
                 +--| Categorize Selections
                     +--| Add Brush
                         +--| Update Brush
                             +--| Categorize Selections
                                 +--| Add Brush
                                     +--| Categorize Selections
```

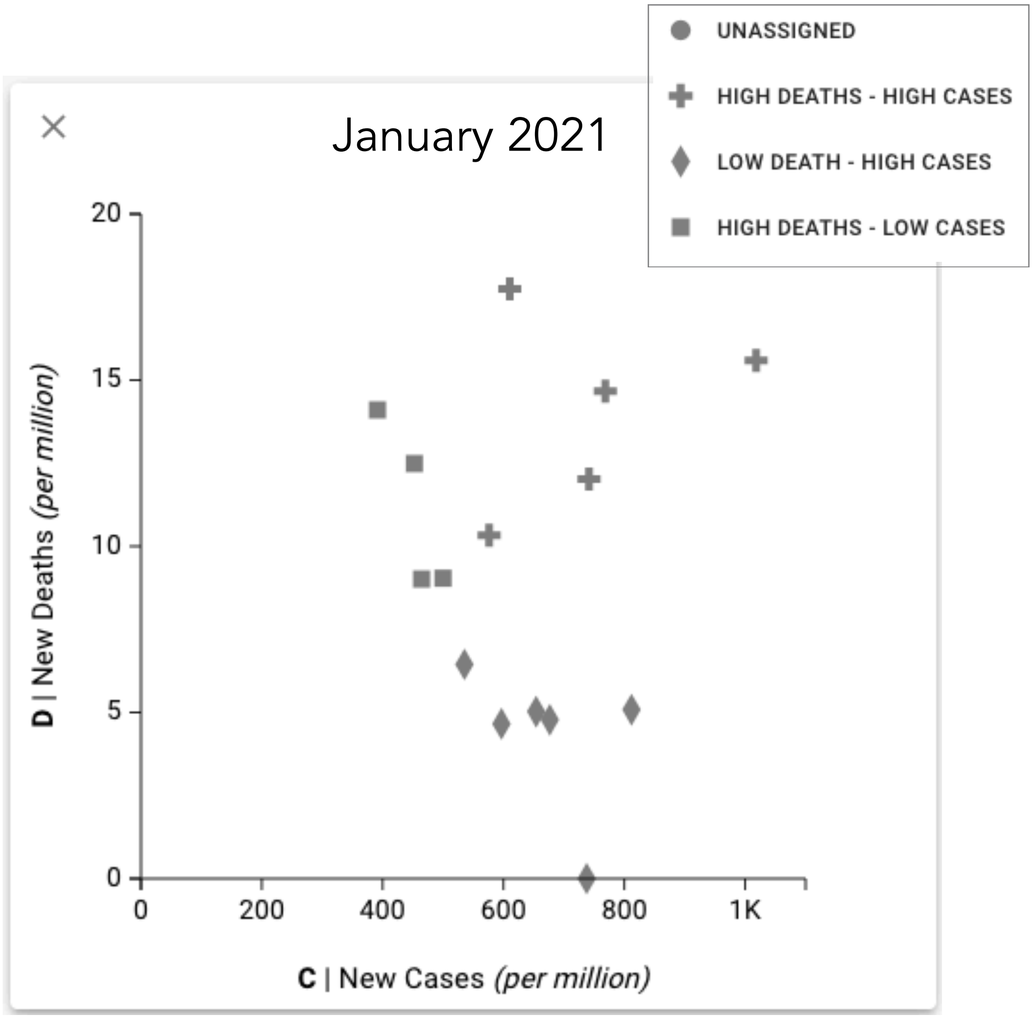


THE UNIVERSITY OF UTAH

Demo

```
# Get the desired workflow
wf = project.get_workflow("1638475878304")

# Description of the options in the workflow
wf.describe()
```

```
Categorize outliers
| Root
 +--| Adding scatterplot for new_cases_per_million-new_deaths_per_million
     +--| Apply Outlier selection
         +--| Filter In
             +--| Add Brush
                 +--| Categorize Selections
                     +--| Add Brush
                         +--| Update Brush
                             +--| Categorize Selections
                                 +--| Add Brush
                                     +--| Categorize Selections
```



THE UNIVERSITY OF UTAH

Demo

```
# This module exposes the Reapply class
from reapply_workflows import Reapply
```

```
# Initialize the Reapply library
reapply = Reapply()

# Load the Covid Dataset project
project = reapply.load("Covid OWID")
project.list_workflows()
```

```
Categorize outliers - 1638475878304
```

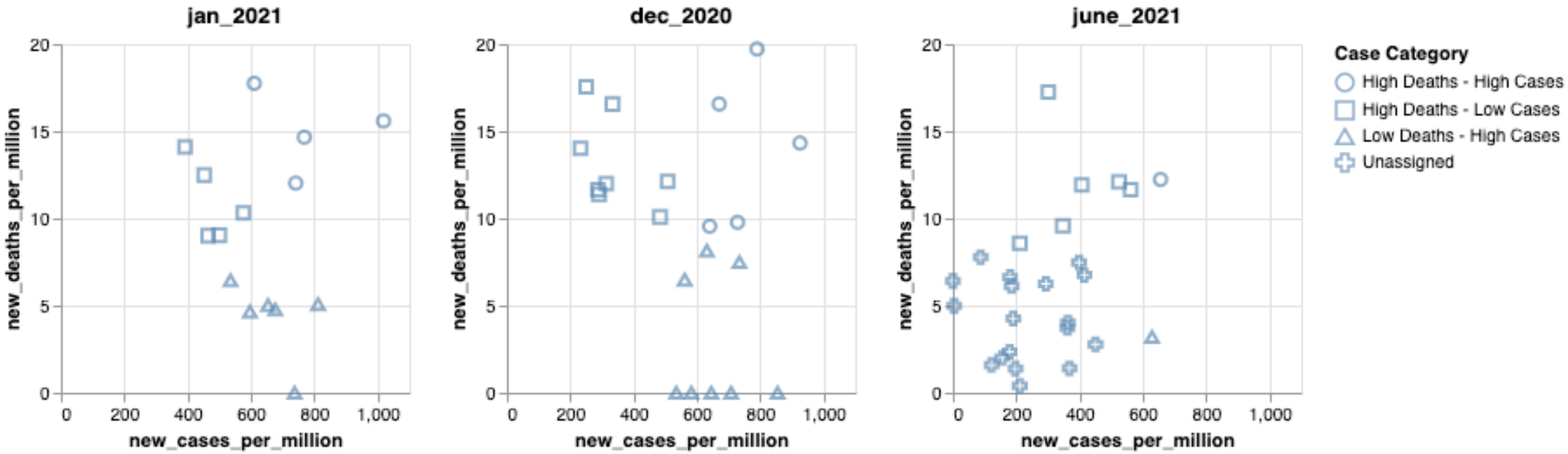
```
# Get the desired workflow
wf = project.get_workflow("1638475878304")

# Description of the options in the workflow
wf.describe()
```

```
Categorize outliers
| Root
 +--| Adding scatterplot for new_cases_per_million-new_deaths_per_million
     +--| Apply Outlier selection
         +--| Filter In
             +--| Add Brush
                 +--| Categorize Selections
                     +--| Add Brush
                         +--| Update Brush
                             +--| Categorize Selections
                                 +--| Add Brush
                                     +--| Categorize Selections
```

THE UNIVERSITY OF UTAH

Demo

```
# Apply the workflow to three versions of the dataset
results_jan_2021 = wf.apply(jan_2021, "location")
results_dec_2020 = wf.apply(dec_2020, "location")
results_june_2021 = wf.apply(june_2021, "location")
```



THE UNIVERSITY OF UTAH

Demo

```
# This module exposes the Reapply class
from reapply_workflows import Reapply
```

```
# Initialize the Reapply library
reapply = Reapply()

# Load the Covid Dataset project
project = reapply.load("Covid OWID")
project.list_workflows()
```

```
Categorize outliers - 1638475878304
```

```
# Get the desired workflow
wf = project.get_workflow("1638475878304")

# Description of the options in the workflow
wf.describe()
```

```
Categorize outliers
| Root
 +--| Adding scatterplot for new_cases_per_million-new_deaths_per_million
     +--| Apply Outlier selection
         +--| Filter In
             +--| Add Brush
                 +--| Categorize Selections
                     +--| Add Brush
                         +--| Update Brush
                             +--| Categorize Selections
                                 +--| Add Brush
                                     +--| Categorize Selections
```

```
# Apply the workflow to three versions of the dataset
results_jan_2021 = wf.apply(jan_2021, "location")
results_dec_2020 = wf.apply(dec_2020, "location")
results_june_2021 = wf.apply(june_2021, "location")
```

THE UNIVERSITY OF UTAH

Demo

```
# This module exposes the Reapply class
from reapply_workflows import Reapply
```

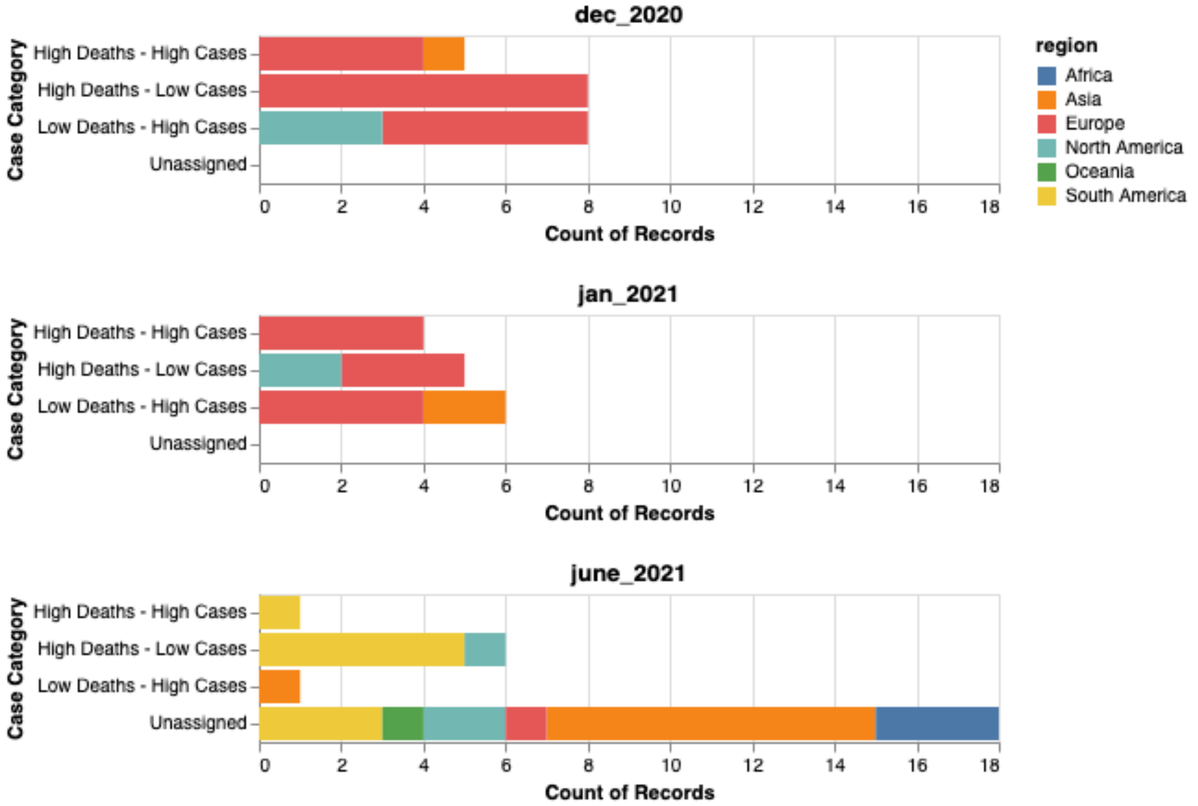
```
# Initialize the Reapply library
reapply = Reapply()

# Load the Covid Dataset project
project = reapply.load("Covid OWID")
project.list_workflows()
```

```
Categorize outliers - 1638475878304
```

```
# Get the desired workflow
wf = project.get_workflow("1638475878304")

# Description of the options in the workflow
wf.describe()
```

```
Categorize outliers
| Root
 +--| Adding scatterplot for new_cases_per_million-new_deaths_per_million
     +--| Apply Outlier selection
         +--| Filter In
             +--| Add Brush
                 +--| Categorize Selections
                     +--| Add Brush
                         +--| Update Brush
                             +--| Categorize Selections
                                 +--| Add Brush
                                     +--| Categorize Selections
```

```
# Apply the workflow to three versions of the dataset
results_jan_2021 = wf.apply(jan_2021, "location")
results_dec_2020 = wf.apply(dec_2020, "location")
results_june_2021 = wf.apply(june_2021, "location")
```



Evaluation

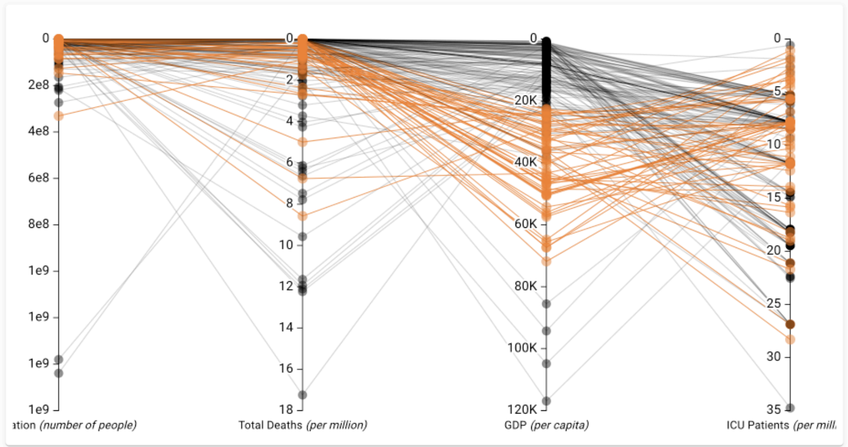
Evaluation

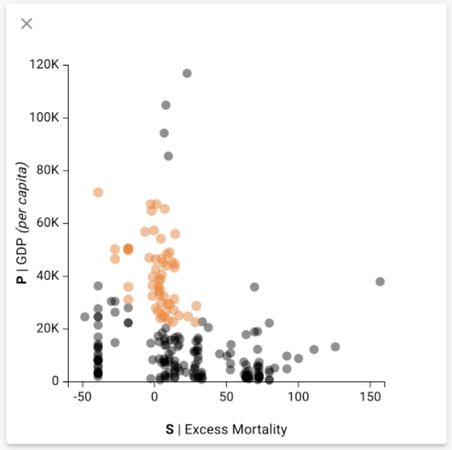
Usage Scenarios

THE UNIVERSITY OF UTAH

Evaluation

Usage Scenarios



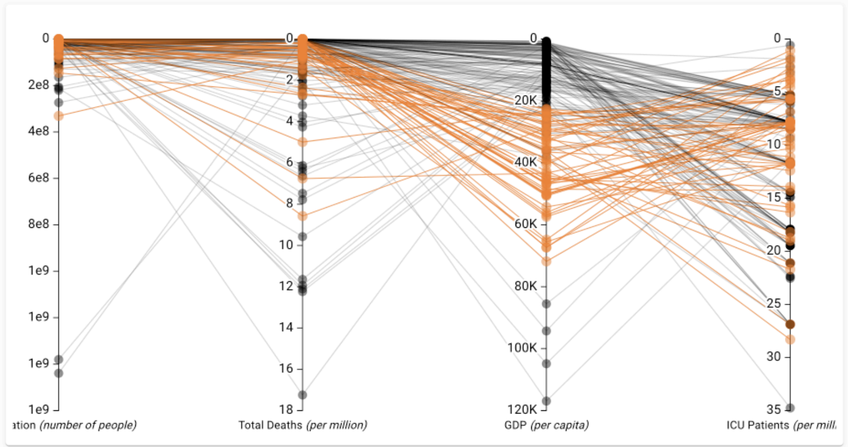


Covid - Our World in Data

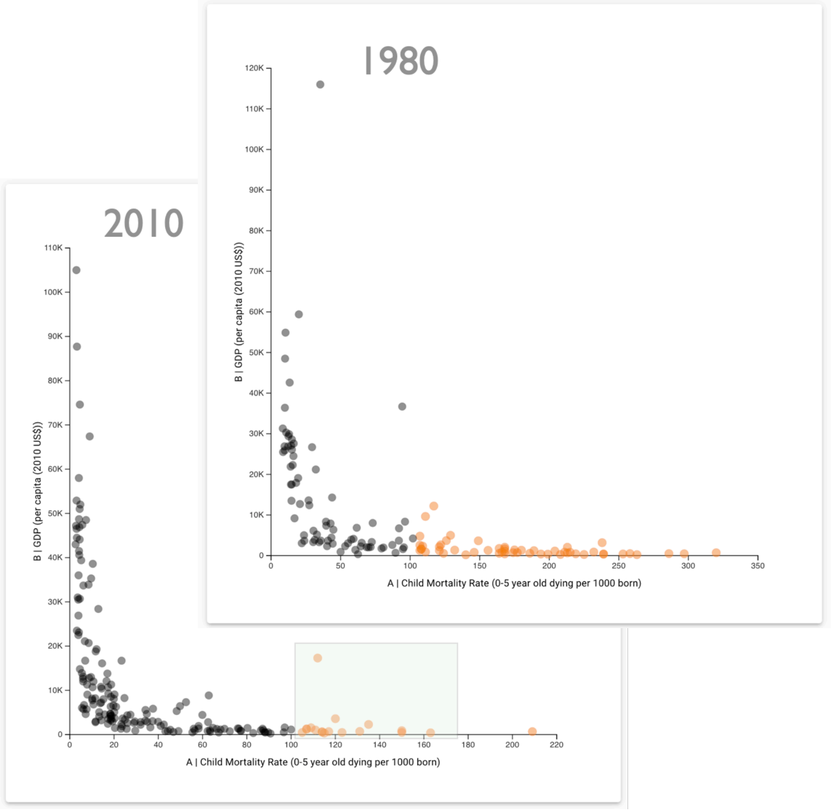
THE UNIVERSITY OF UTAH

Evaluation

Usage Scenarios



Covid - Our World in Data



Gapminder Public Health

Evaluation

Expert Feedback

4

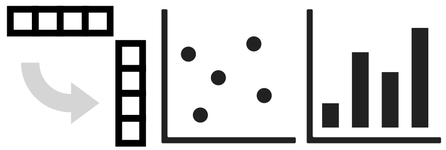
data practitioners

THE UNIVERSITY OF UTAH

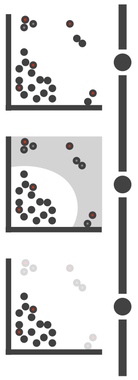
Evaluation

Expert Feedback

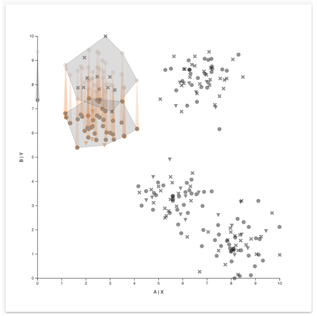
Interview



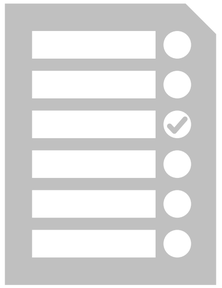
Introduction to the techniques



Demo



Feedback



Evaluation

Expert Feedback

About curating workflows from provenance

*I like this, because it's much **more natural**.*

About reusing the workflows on updated datasets

*I definitely think it will be applicable because most of the time, we **actually don't inherently change the method itself** … so I definitely can see this to be helpful*

About reusing the workflows in a computational environment

*Great to be able to **click on the Select 53 points**, and then **see the all code**, you know, to that would select the 53 points.*

Future Work

Future Work

Workflows as templates

Future Work

Workflows as templates

Reapplying captured workflows on unrelated datasets

Future Work

Workflows as templates

Reapplying captured workflows on unrelated datasets

Automate repetitive data preprocessing steps

Future Work

Workflows as templates

Reapplying captured workflows on unrelated datasets

Automate repetitive data preprocessing steps

Training using workflows curated by experts

Future Work

Integration with interactive visualizations in the notebook environment

THE UNIVERSITY OF UTAH

Future Work

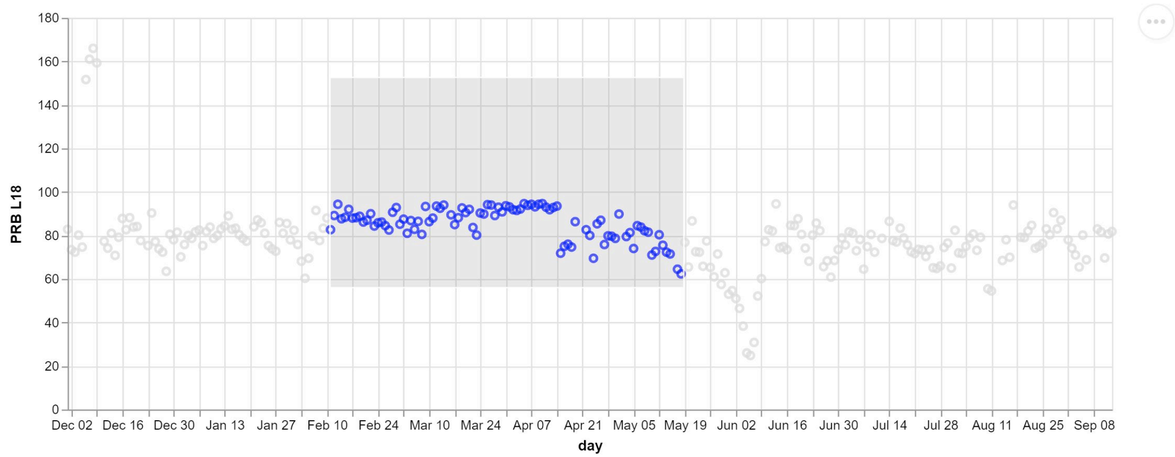
Integration with interactive visualizations in the notebook environment

Libraries like Altair support interactive visualizations in notebook environment.

Typically the interactions cannot manipulate the data

[83]:
```
brush = alt.selection_interval()
chart = alt.Chart(df).encode(
        x="day:T",
        ).properties(
            width=800
        ).add_selection(
            brush
        )
chart.mark_point().encode(y="PRB L18:Q", color=alt.condition(brush,
                                                            alt.value('blue'),
                                                            alt.value('lightgray')))
```

[83]:



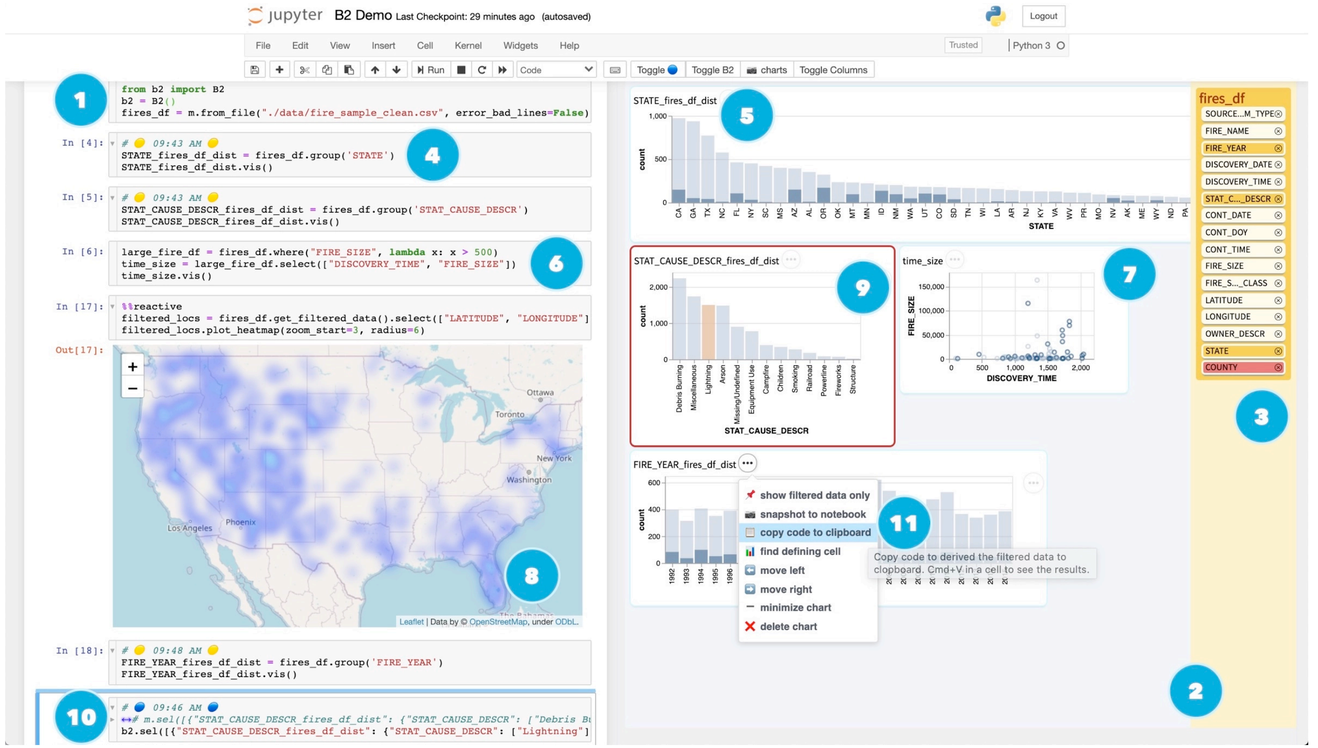
THE UNIVERSITY OF UTAH

Future Work

Integration with interactive visualizations in the notebook environment

B2 — introduces techniques to coordinate interactive visualizations and code cells.

Data queries (or selections) act as the bridge between interactive visualizations and the code.



[Wu et. al., 2020]

Future Work

Integration with interactive visualizations in the notebook environment

Integrate interactive visualizations with the notebook

Semantic analysis provenance as a shared abstraction

Kiran Gadhave

Twitter: @kbgadhave

Email: kirangadhave2@gmail.com

Paper Website: https://tinyurl.com/yvw3xmw4

Thank You

Questions

We thank Derya Akbaba and Jack Wilburn from VDL for help with the expert interviews

Supported by National Science Foundation (IIS 1751238)

THE UNIVERSITY OF UTAH

visualization design lab

SCI www.sci.utah.edu