

Kiran Gadhave, Zach Cutler, **Alexander Lex**

<http://vdl.sci.utah.edu>



Persist: Persistent and Reusable Interactions in Computational Notebooks



visualization
design lab



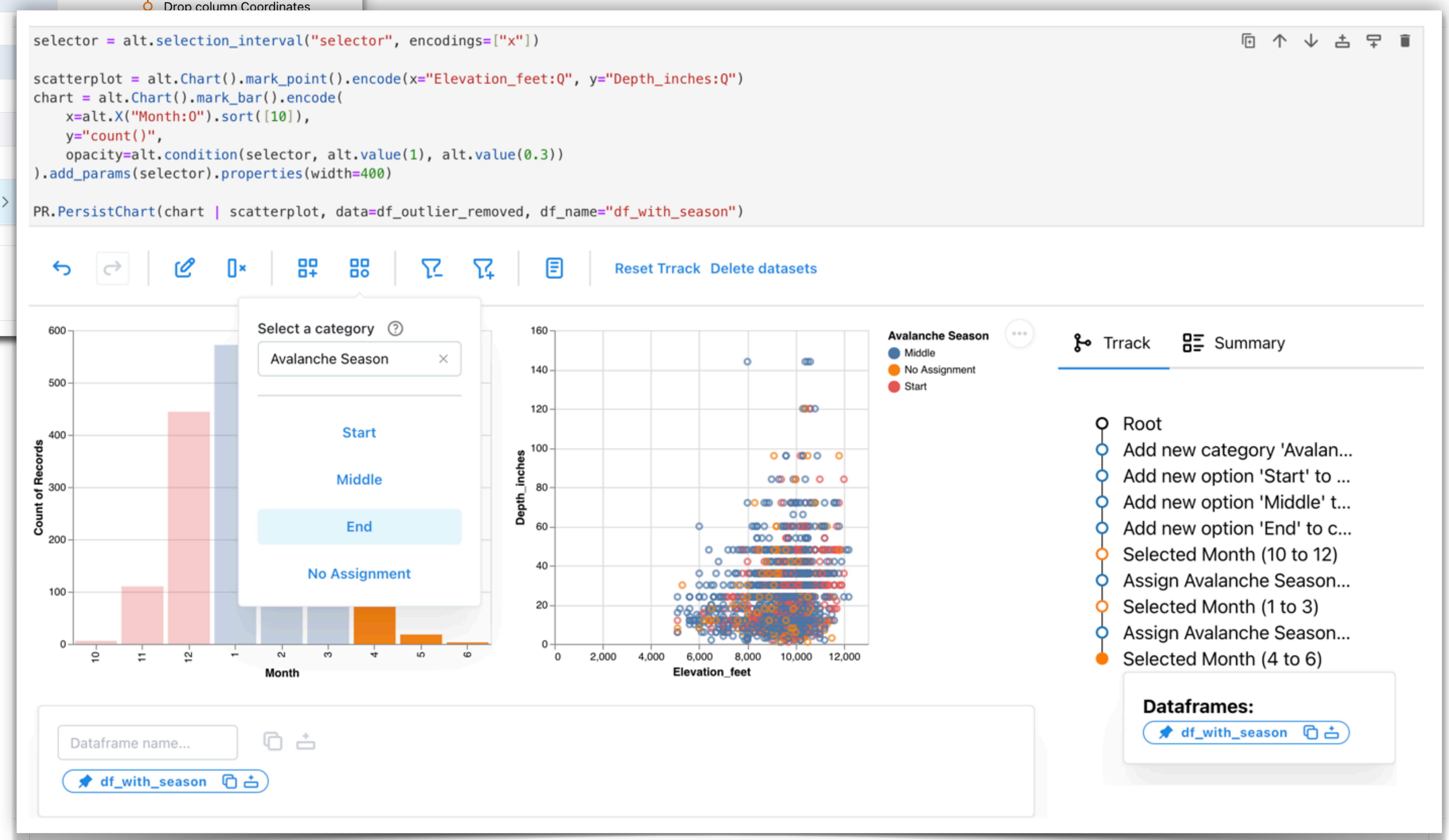
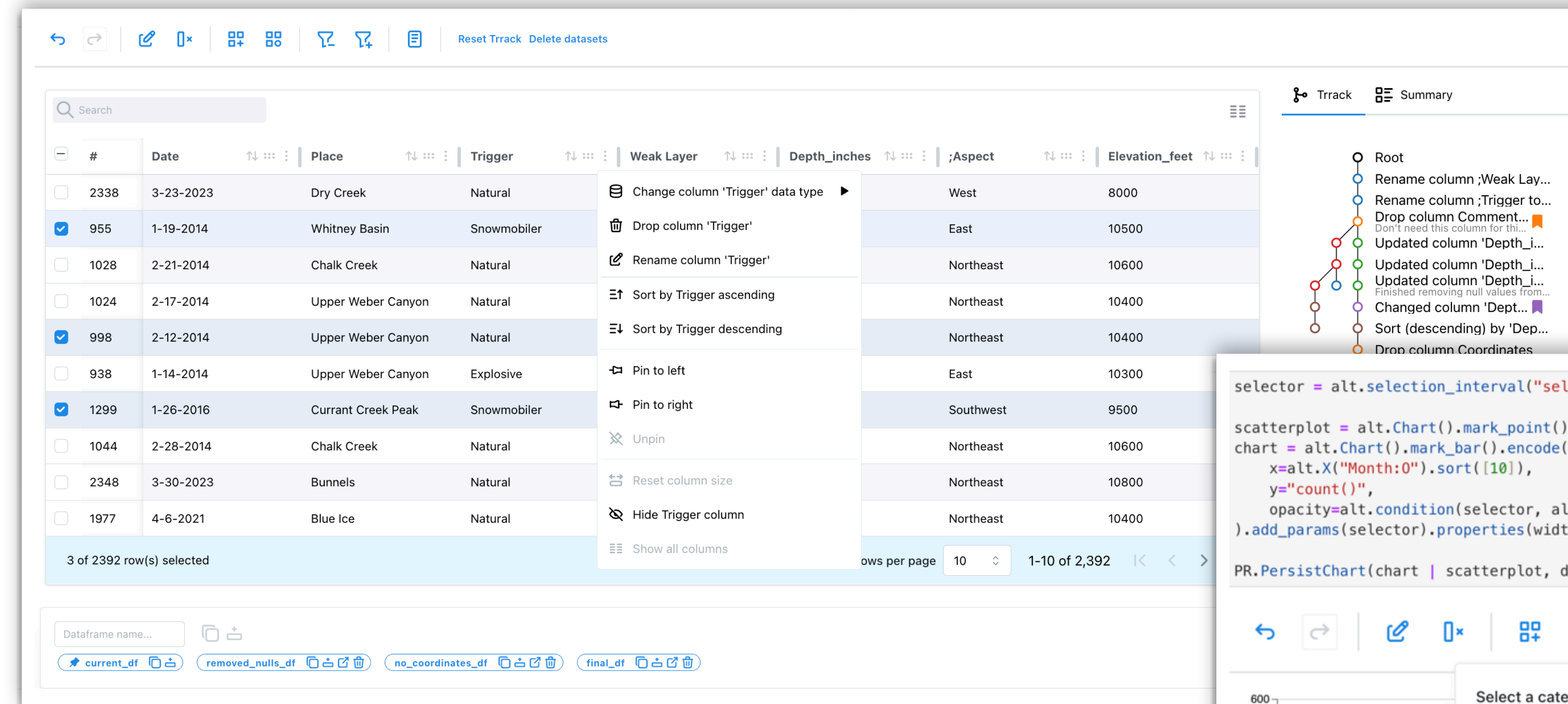
THIS SHOULD BE
PRESENTED BY
KIRAN GADHAVE

www.kirangadhave.me

On the Job Market!



PERSISTENT AND REUSABLE INTERACTIONS IN COMPUTATIONAL NOTEBOOKS



A Jupyter Plugin

<https://vdl.sci.utah.edu/persist/>

`pip install persist_ext`

Demo Notebook: <https://tinyurl.com/5db7nynn>

DATASET EXAMPLE: HISTORICAL AVALANCHES IN UTAH

Avalanches are a major hazard

in Utah

Utah Avalanche Center collects data about avalanches, including **where it occurred (location, elevation), how it occurred, how big it was,** etc.



WHAT IS THIS TALK ABOUT?

Supposed you're doing data analysis in Python

What's the pandas code...

- ...to **change the order of columns?**
- ...to **drop a column?**
- ...to **change the label of a column?**

Nothing here is hard, but it's annoying.

PERSIST MAKES THIS EASY

[4]: PR.PersistTable(avalanches, df_name="avalanches")

Reset Ttrack

Delete datasets

Search

<input type="checkbox"/>	#	;Region	Month	Day	Year	;Trigger	;Weak Layer
<input type="checkbox"/>	1	Salt Lake	11	9	2012	Snowboarder	New Snow/Old Snow
<input type="checkbox"/>	2	Salt Lake	11	11	2012	Skier	New Snow/Old Snow
<input type="checkbox"/>	3	Salt Lake	11	11	2012	Skier	Facets
<input type="checkbox"/>	4	Salt Lake	11	11	2012	Skier	New Snow
<input type="checkbox"/>	5	Salt Lake	11	11	2012	Skier	Facets
<input type="checkbox"/>	6	Salt Lake	11	10	2012	Skier	New Snow/Old Snow
<input type="checkbox"/>	7	Salt Lake	11	12	2012	Skier	Facets
<input type="checkbox"/>	8	Salt Lake	12	8	2012	Skier	Facets
<input type="checkbox"/>	9	Salt Lake	12	9	2012	Skier	Facets
<input type="checkbox"/>	10	Salt Lake	12	10	2012	Skier	Facets

Rows per page

10

1-10 of 2,392

Ttrack

Summary

● Root

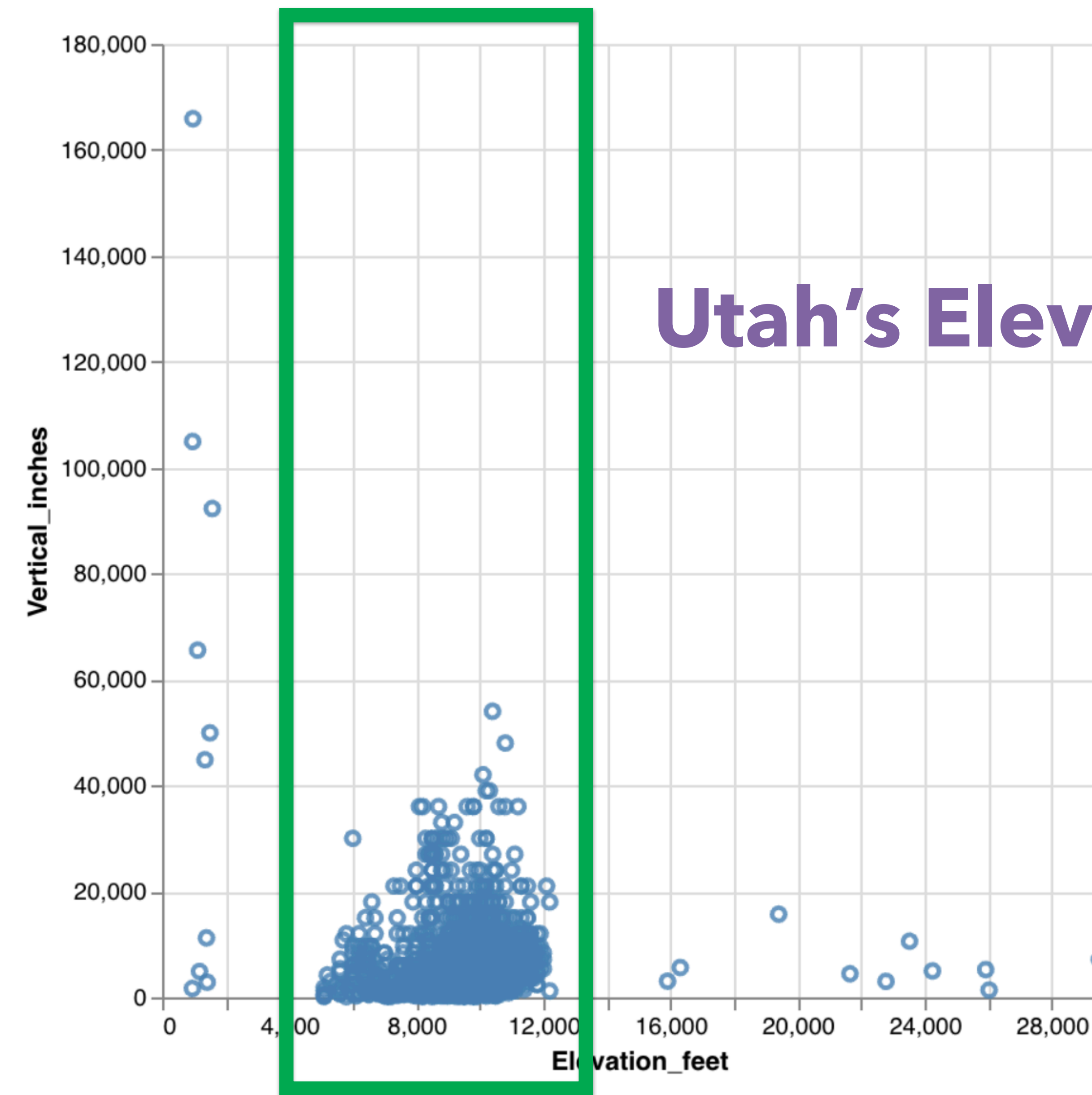
Dataframe name...

avalanches

WHAT IS THIS TALK ABOUT?

Have you ever plotted something
and wished you could just “fix”
things as you spot them?

How deep (big) the avalanche was



Utah's Elevation Range

Elevation where the avalanche occurred

PERSIST MAKES THIS EASY

```
[6]: PR.plot.scatterplot(avalanches, "Elevation_feet:Q", "Vertical_inches:Q", df_name="avalanches")
```

↶

↷

✎

×

⊞

⊞

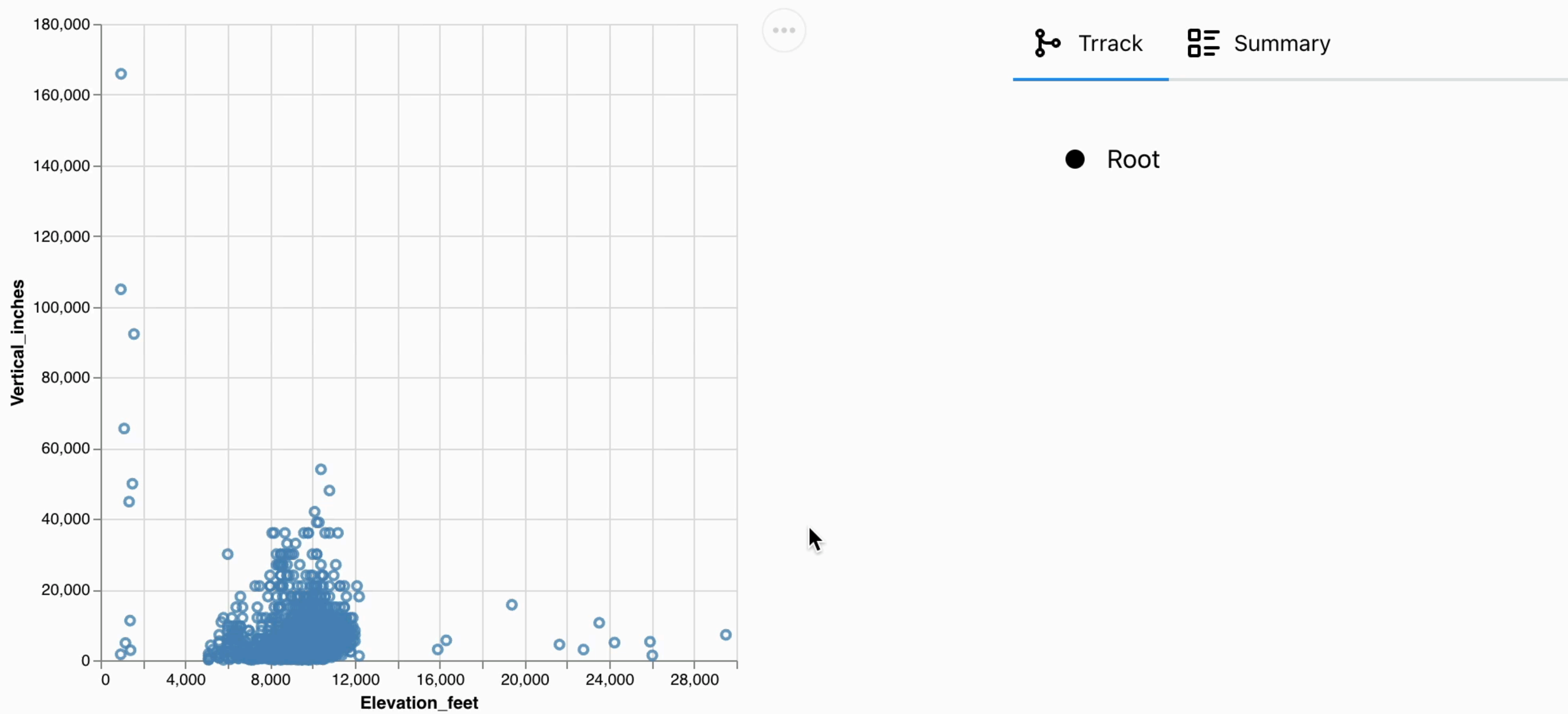
⌵

⌵

☰

Reset Ttrack

Delete datasets



SO WHAT'S SPECIAL HERE?

Lots of **vis tools** support these operations

Most **data wrangling happens in code**: it's just more powerful

Opportunity: bring **interactive operations to code**!

Persist works **INSIDE** your Jupyter Notebook

BRIDGING BETWEEN DATA ANALYSIS MODALITIES

**BRIDGING BETWEEN
DATA ANALYSIS
MODALITIES**

What are Modalities?

1. Interactive Vis

2. Code

INTERACTIVE VISUALIZATION: BENEFITS

Intuitive

Easy to use

Uses human perceptual capabilities



INTERACTIVE VISUALIZATION: DOWNSIDES

Limited Expressivity

Some operations are difficult

e.g., conditional queries..

Not reusable

need to redo analysis when data changes

Not reproducible



CODE: BENEFITS

Flexible and powerful

you basically can do anything

Reusable

if your data changes, re-run

Reproducible

everything is documented

```
1 # Keep this cell
2 avy_df = pd.read_csv('./avalanches.csv')
3
4 # Remove NaN coordinates
5 avy_df = avy_df[avy_df['Coordinates']==avy_df['Coordinates']]
6
7 # Split into latitude & longitude
8 avy_df[['lat', 'lon']] = avy_df['Coordinates'].str.split(',', expand=True)
9
10 # Remove values outside of Utah bounds
11 avy_df = avy_df[(36 < avy_df['lat'].astype('float')) & (avy_df['lat'].astype('float') < 42)]
12 avy_df = avy_df[(-114 < avy_df['lon'].astype('float')) & (avy_df['lon'].astype('float') < -108)]
13
14 # Keep columns we need
15 avy_df = avy_df[['Date', 'Region', 'Trigger', 'lat', 'lon']]
16 avy_df.head()
```


CODE / SCRIPTING: DOWNSIDES

It's hard

requires extensive training

reading documentation

not discoverable

It's time consuming

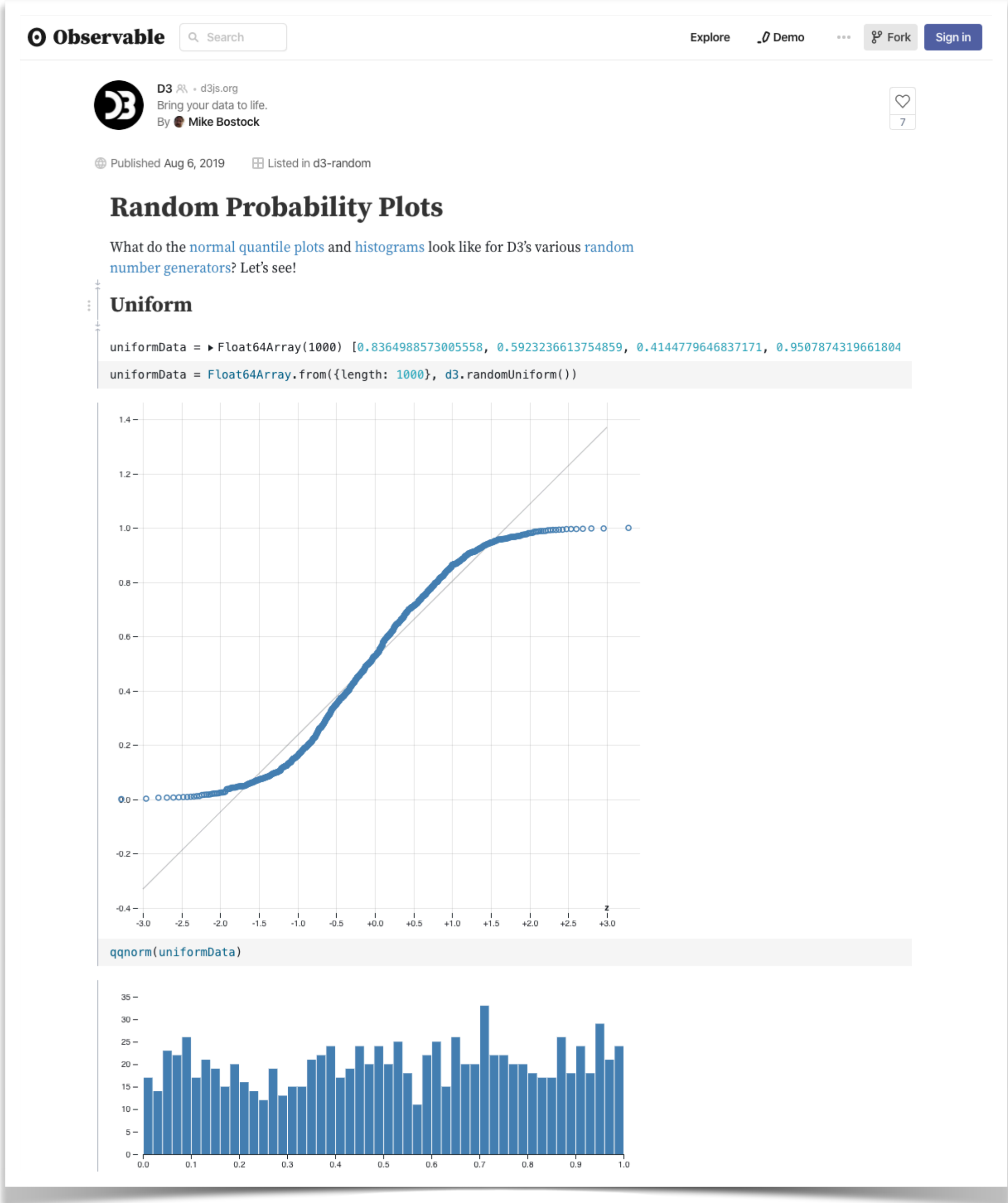
even simple things require effort

Some operations are difficult

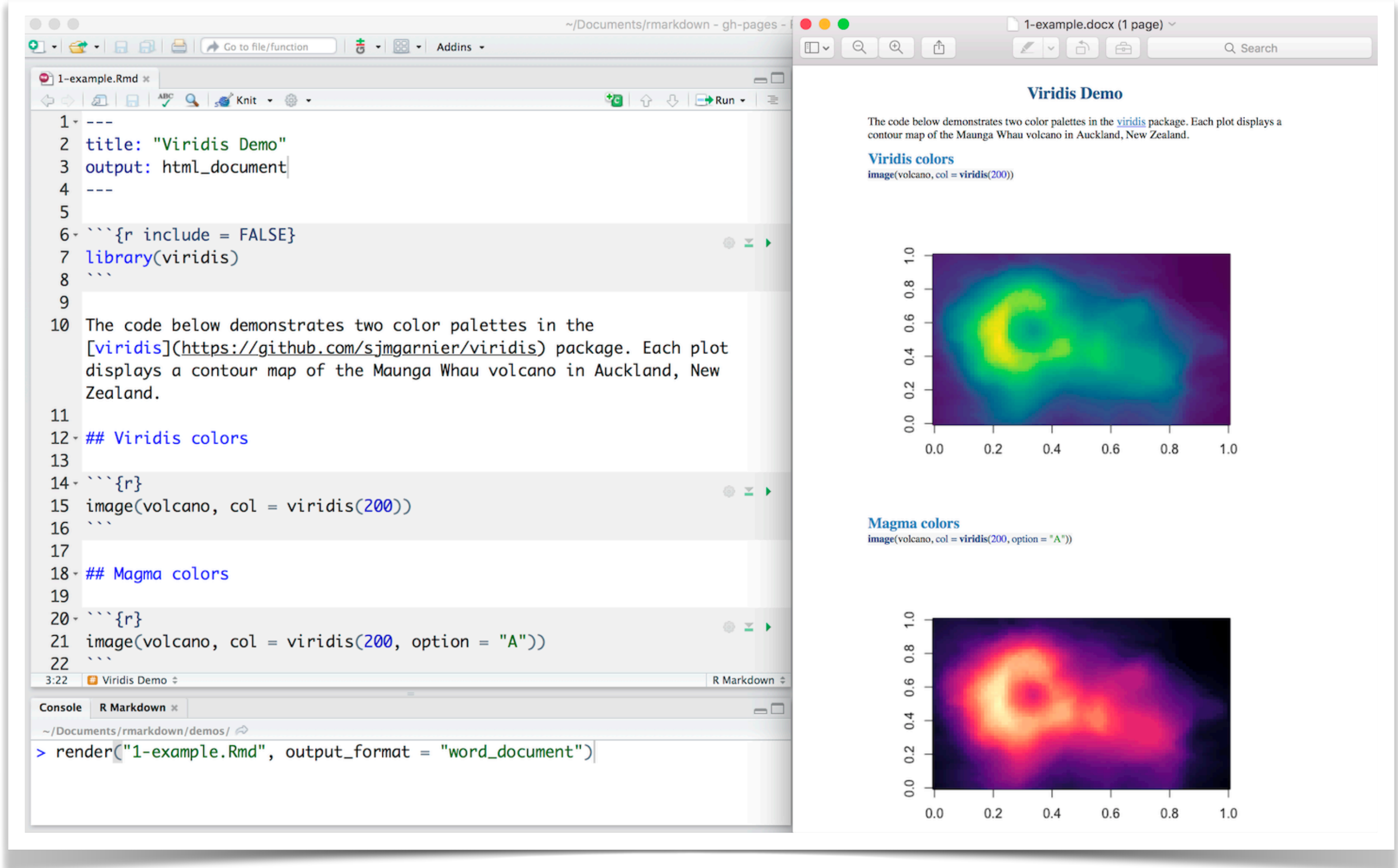
e.g., labeling data points

```
1 # Keep this cell
2 avy_df = pd.read_csv('./avalanches.csv')
3
4 # Remove NaN coordinates
5 avy_df = avy_df[avy_df['Coordinates']!=avy_df['Coordinates']]
6
7 # Split into latitude & longitude
8 avy_df[['lat', 'lon']] = avy_df['Coordinates'].str.split(',', expand=True)
9
10 # Remove values outside of Utah bounds
11 avy_df = avy_df[(36 < avy_df['lat'].astype('float')) & (avy_df['lat'].astype('float') < 42)]
12 avy_df = avy_df[(-114 < avy_df['lon'].astype('float')) & (avy_df['lon'].astype('float') < -108)]
13
14 # Keep columns we need
15 avy_df = avy_df[['Date', 'Region', 'Trigger', 'lat', 'lon']]
16 avy_df.head()
```


COMPUTATIONAL NOTEBOOKS: A MIDDLE GROUND?



Observable



COMPUTATIONAL NOTEBOOKS: A MIDDLE GROUND?

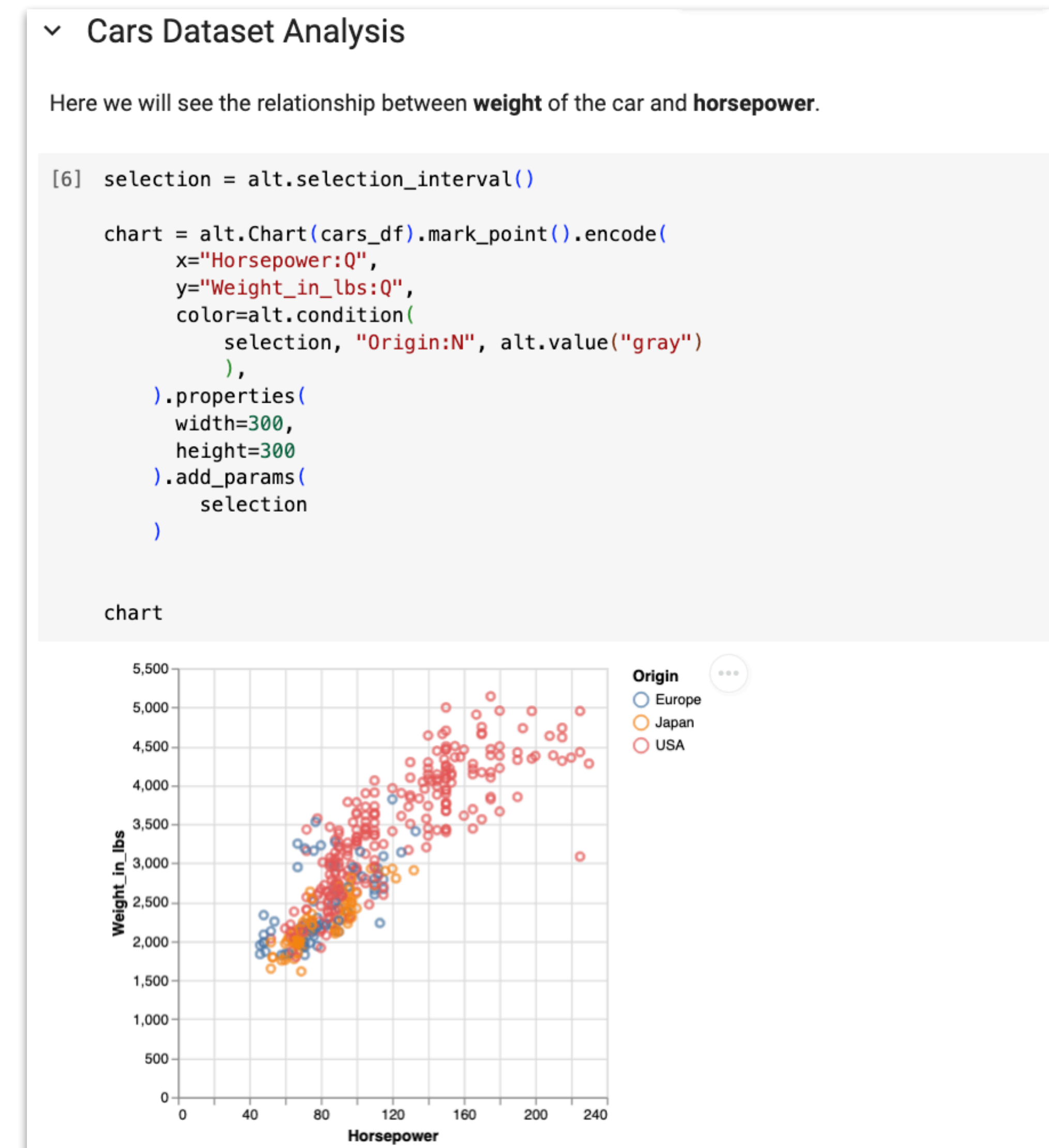
Yes!

Afford both scripting and interactive visualization

But visualizations are a dead end

can't "use" interaction in code

e.g., changing a label, or filtering a value



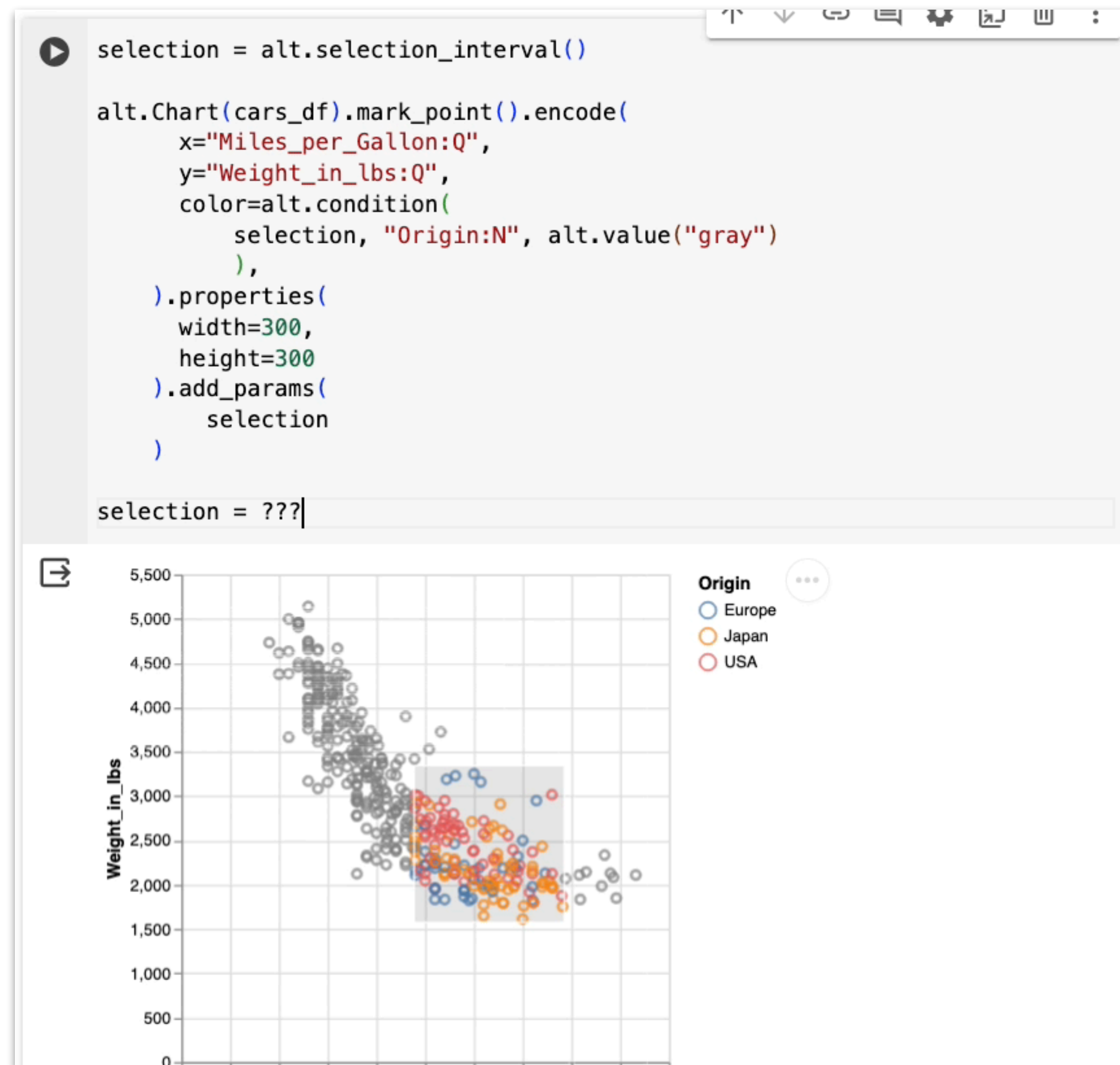
Documentation

Code

Visualizations

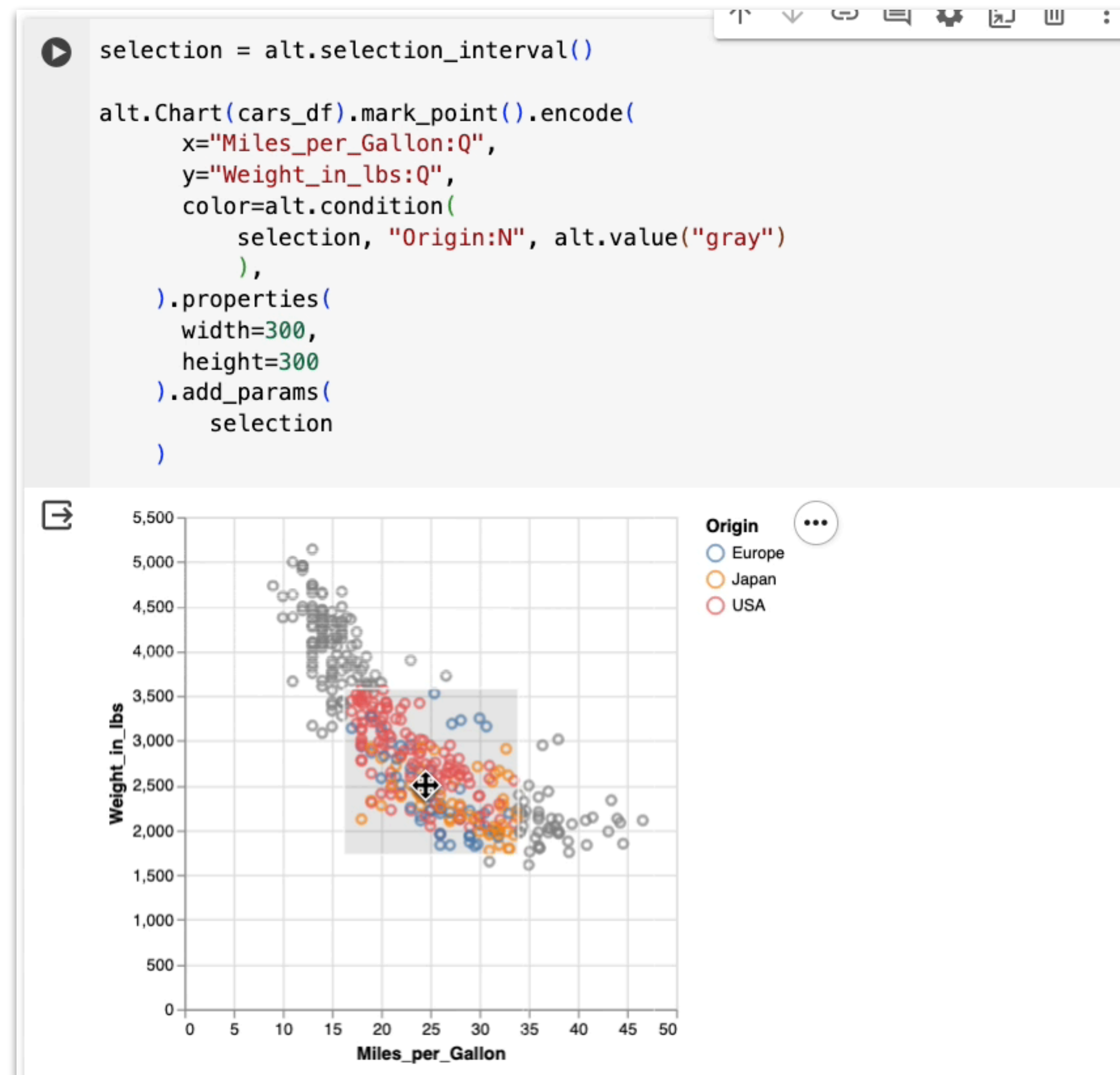
GAPS BETWEEN CODE AND INTERACTIVE OUTPUTS*

Semantic Gap



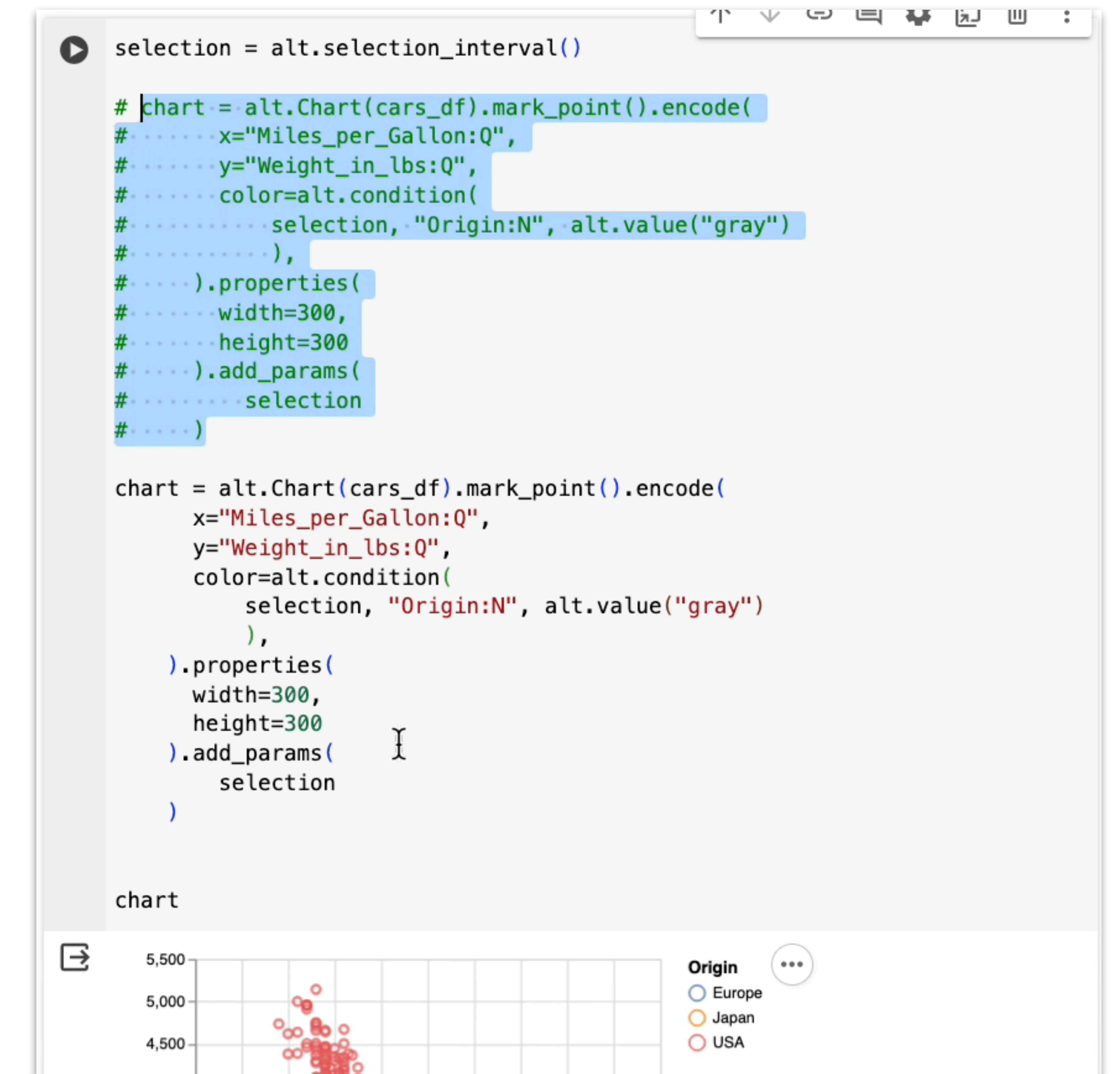
Information only flows from
code to visualization

Temporal Gap



Changes made to code are preserved
Changes made to vis are lost

Layout Gap



Changes in code are messy

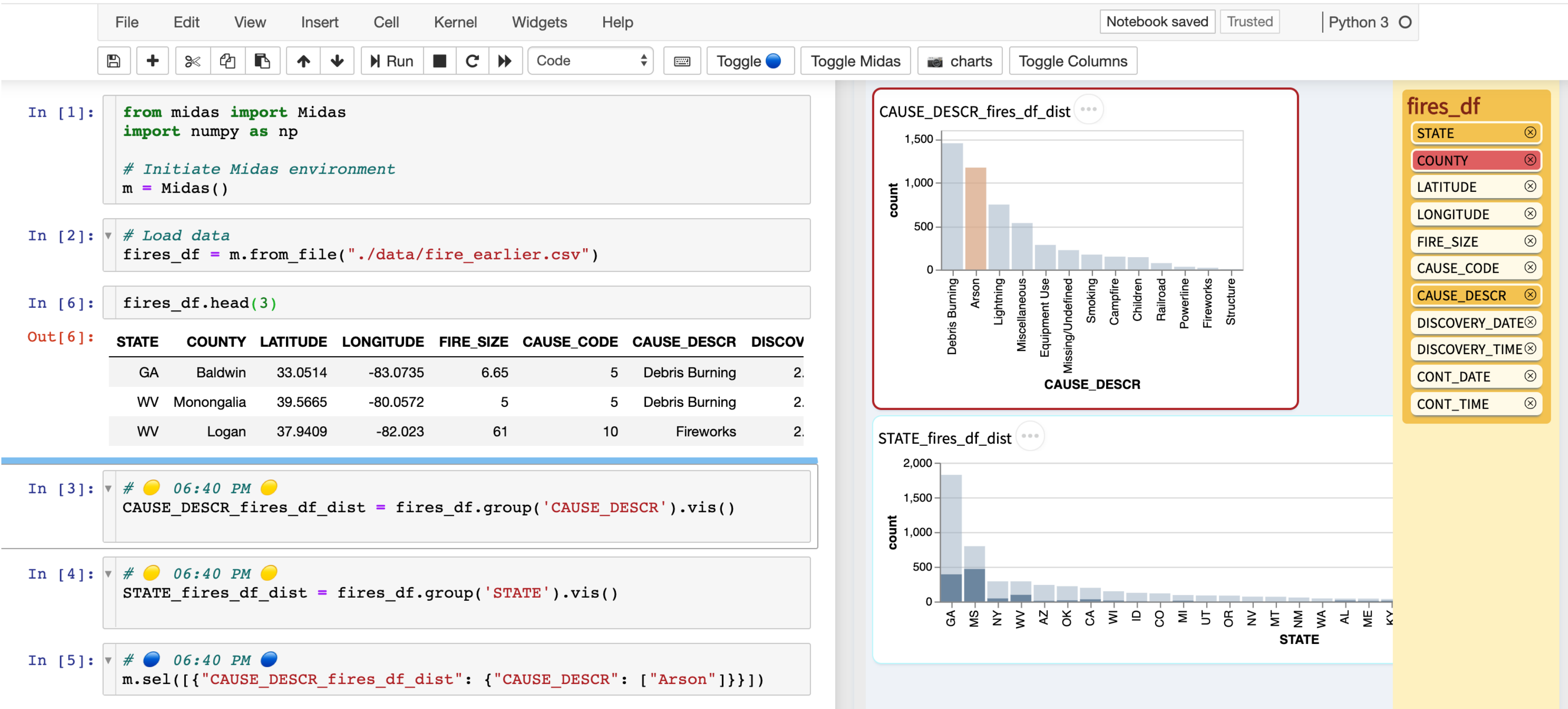
*[Wu, Hellerstein, Satyanarayan, UIST 2020]

THESIS: **BRIDGING** BETWEEN CODE
AND INTERACTIVE VIS IS **USEFUL**

Easy handoffs are important!

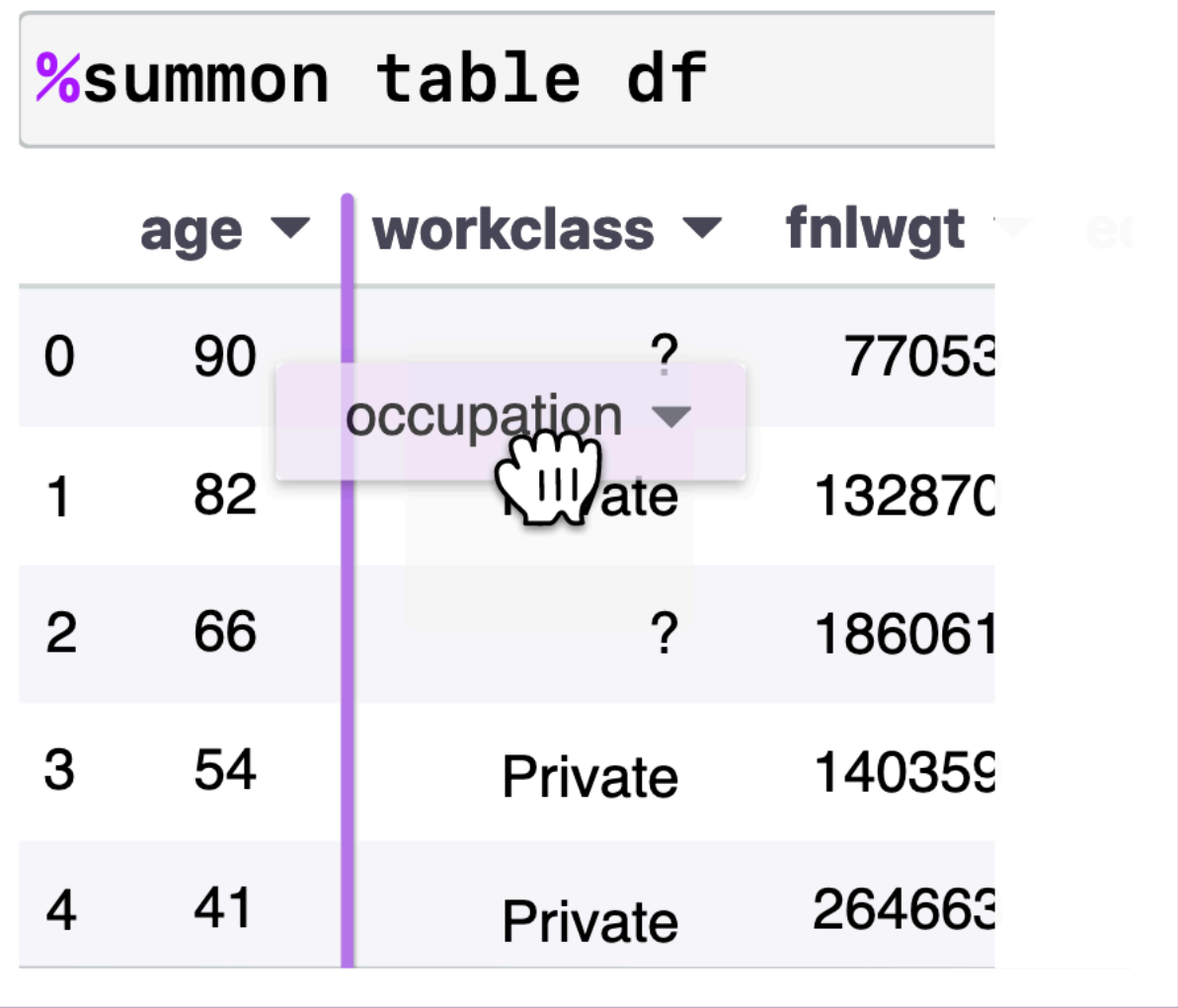
RELATED WORK

B2 – Wu, Hellerstein, Satyanarayan, UIST 2020

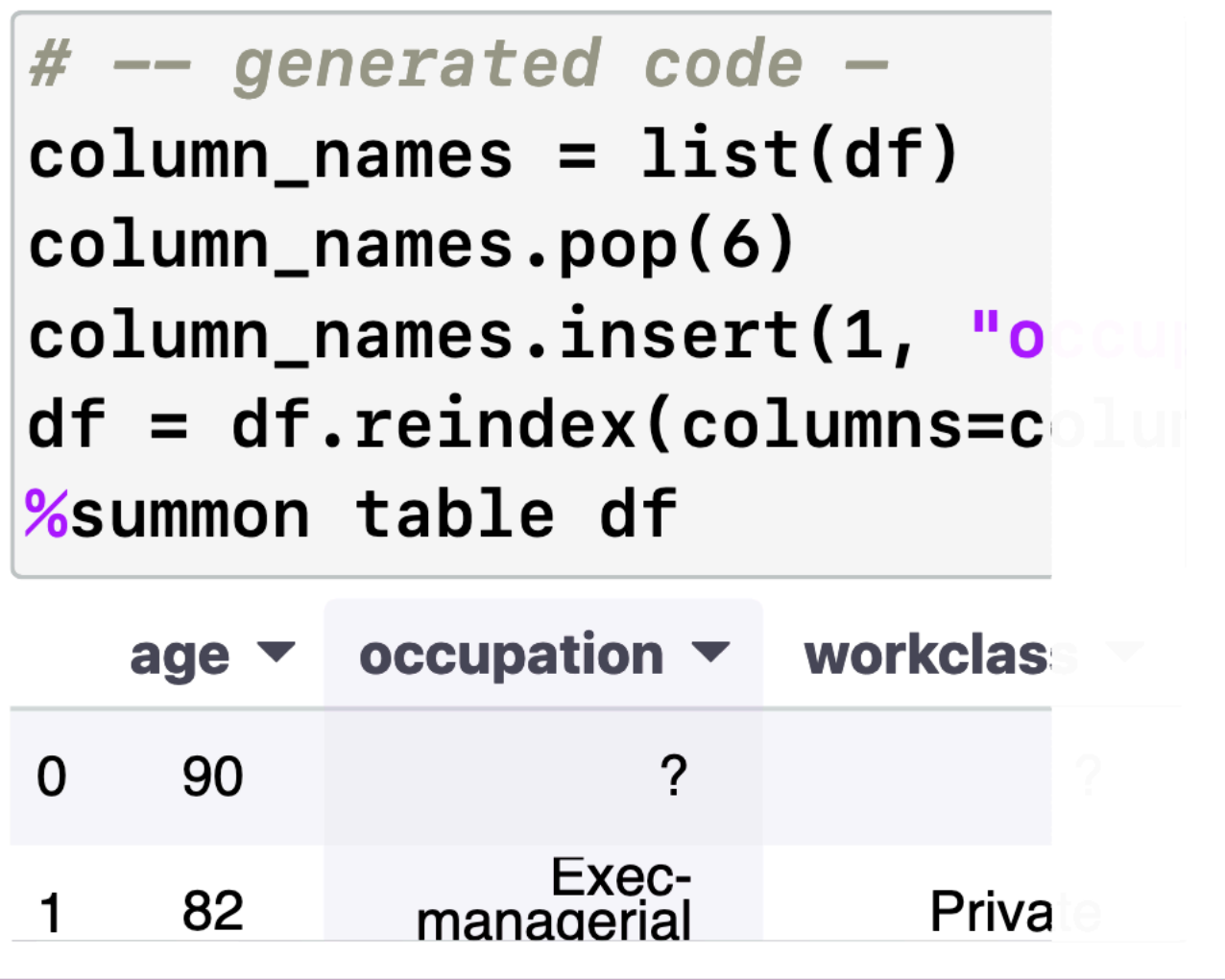


Mage – Kery et al., UIST 2020

1 mage : user edits table



2 mage : edits reflect in code



THE **PERSIT** APPROACH

PRINCIPLE

Track events in interactive visualizations

Map them to **data frame operations**

Operations then **applied to data frame**






HOW IT WORKS

Code to create chart

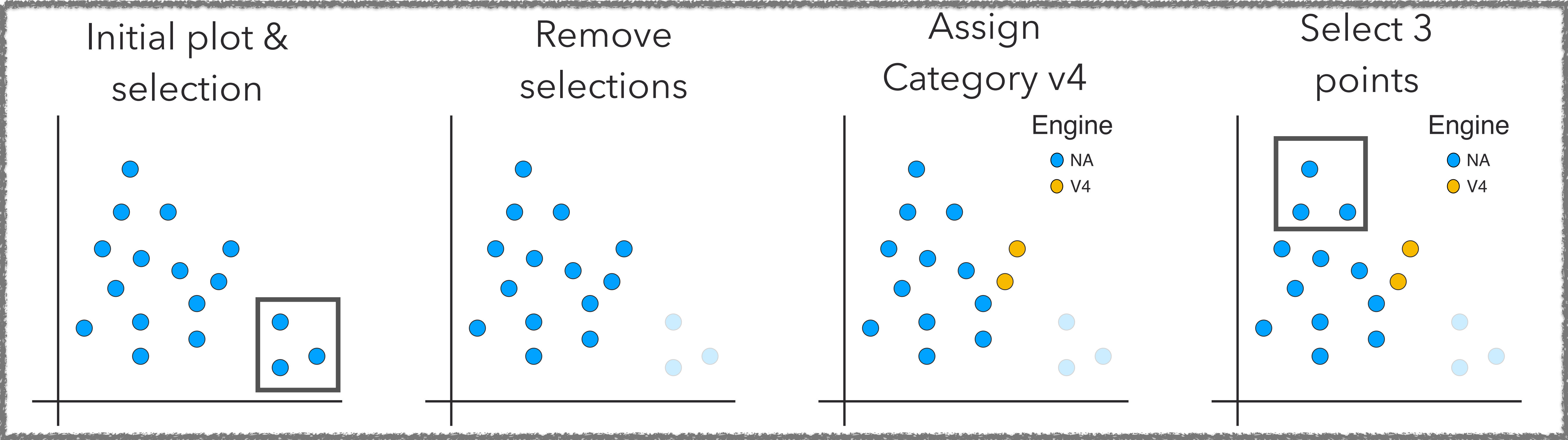


```
df = pd.read_csv('cars.csv')  
PersistChart(scatterplot(df))
```

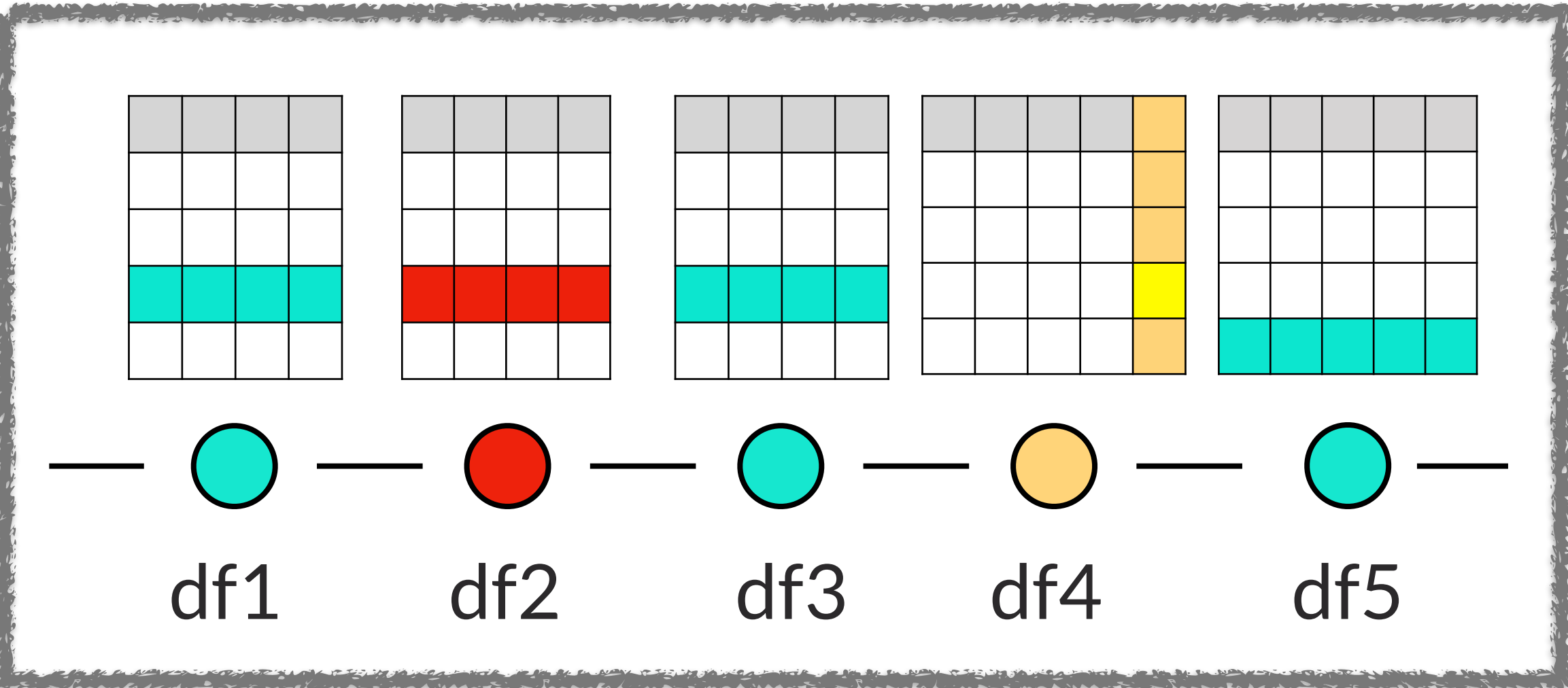
Provenance

- df1  Select 3 points
- df2  Remove selections
- df3  Select 2 points
- df4  Assign category v4 to selection
- df5  Select 3 points

Interactive Visualization



Dataframe updates



In code











```
df5.head()
```

Name	MPG	Cylinders	origin	Engine	is_selected
ford	18	8	USA	NA	FALSE
dodge	15	8	USA	NA	FALSE
volkswagen	22	4	Europe	V4	FALSE
amc	16	8	USA	NA	TRUE

27 rows x 8 columns

OPERATIONS

-  **Selection**
-  **Edit column names, edit cells**
-  **Sort rows/columns**
-  **Drop columns**
-  **Filter (in/out) items**
-  **Label items**
-  **Categorize items**
-  **Change data types**

PERSIST WORKFLOW

**Standard
Vega-Altair Chart**

```
import altair as alt
from vega_datasets import data
import persist_ext as PR

cars_df = data.cars() # Load cars dataset

brush = alt.selection_interval() # Create a 2d brush

# create scatterplot and link to brush
scatterplot = alt.Chart(cars_df).mark_point().encode(
    x="Miles_per_Gallon:Q",
    y="Weight_in_lbs:Q",
    color=alt.condition(brush, alt.value("steelblue"), alt.value("gray"))
).add_params(
    brush
)

PR.PersistChart(scatterplot)
```

Call to Persist with Chart as Parameter


```
import altair as alt
from vega_datasets import data
import persist_ext as PR

cars_df = data.cars() # Load cars dataset

brush = alt.selection_interval() # Create a 2d brush

# create scatterplot and link to brush
scatterplot = alt.Chart(cars_df).mark_point().encode(
    x="Miles_per_Gallon:Q",
    y="Weight_in_lbs:Q",
    color=alt.condition(brush, alt.value("steelblue"), alt.value("gray"))
).add_params(
    brush
)

PR.PersistChart(scatterplot)
```

Persist Toolbar

↶

↷

✎

✖

⊞

⊞

🔍

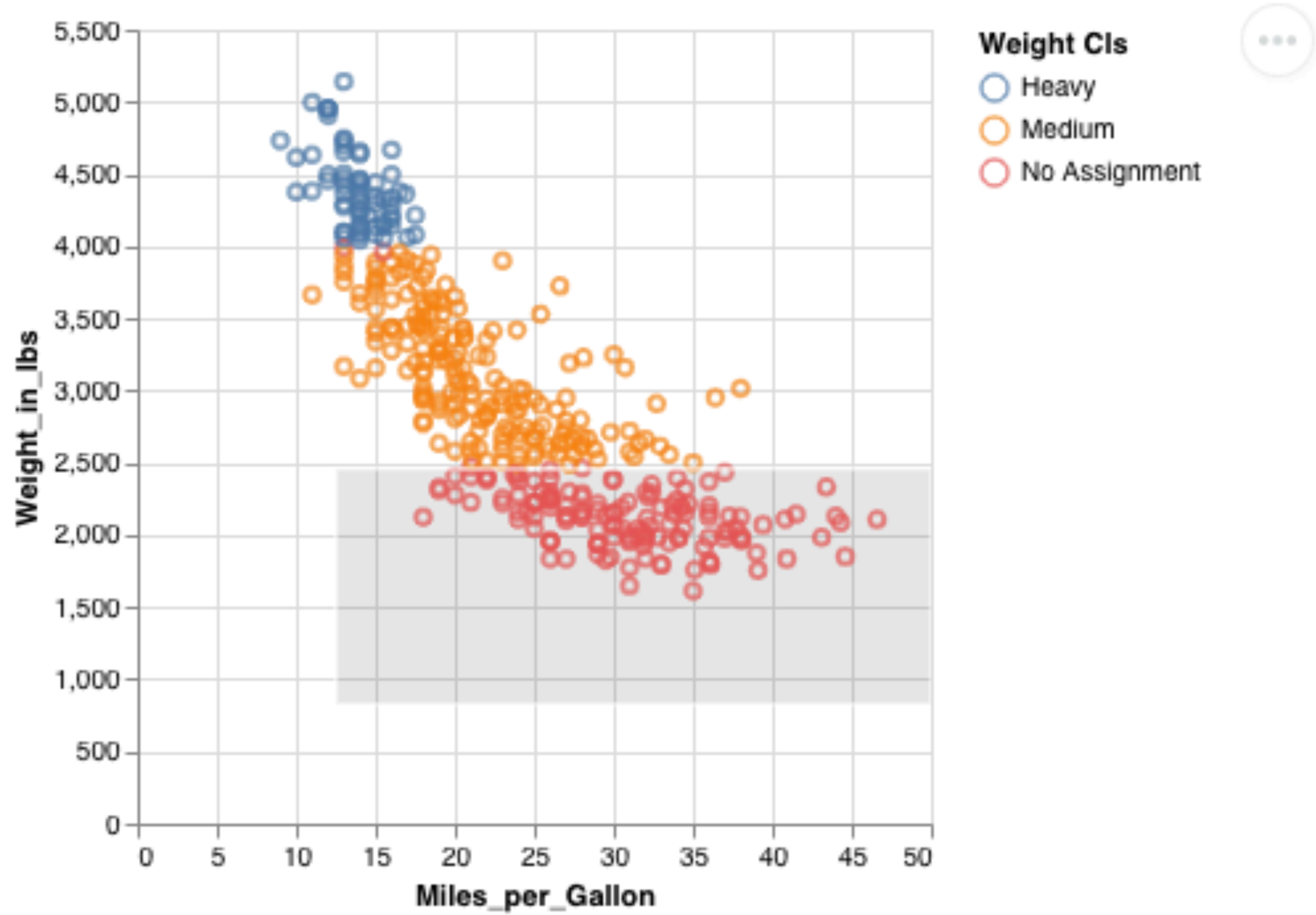
🔍

📄

Reset Ttrack

Delete datasets

Vega-Altair Chart



Dataframe Manager

Dataframe name...

📄

+

🚀 persist_df_2

📄

+

Provenance History

Ttrack

Summary

○ Root

○ Add new category 'We...

○ Add new option 'Heav...

○ Add new option 'Medi...

○ Add new option 'Light'...

○ Selected Miles_per_G...

○ Assign Weight Cls (He...

○ Selected Miles_per_G...

○ Assign Weight Cls (Me...

● Selected Miles_per_G...

○ Assign Weight Cls (Lig...

Dataframes:

🚀 persist_df_2

📄

+


```

import altair as alt
from vega_datasets import data
import persist_ext as PR

cars_df = data.cars() # Load cars dataset

brush = alt.selection_interval() # Create a 2d brush

# create scatterplot and link to brush
scatterplot = alt.Chart(cars_df).mark_point().encode(
    x="Miles_per_Gallon:Q",
    y="Weight_in_lbs:Q",
    color=alt.condition(brush, alt.value("steelblue"), alt.value("gray"))
).add_params(
    brush
)

PR.PersistChart(scatterplot)

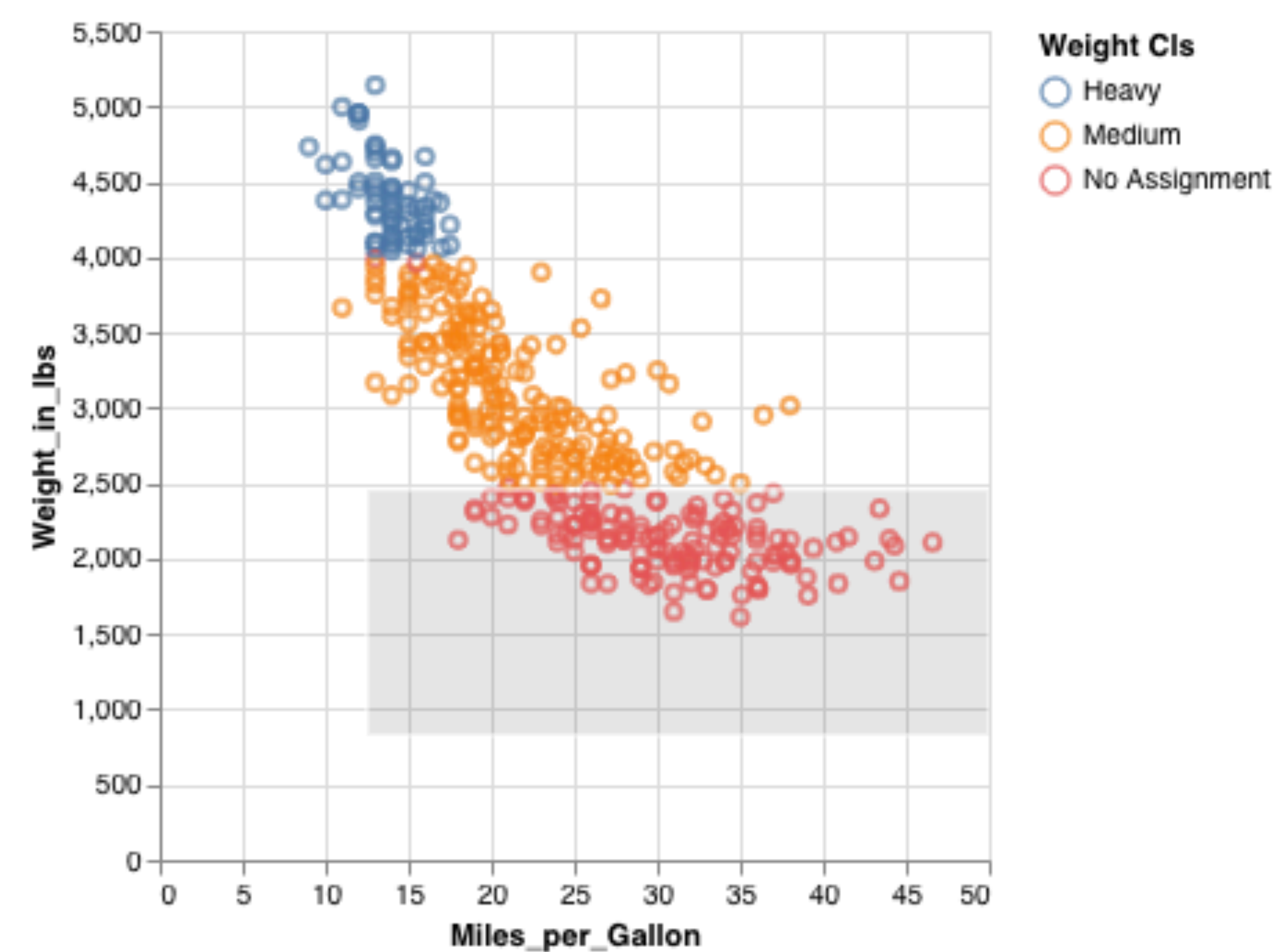
```

Categorization

Filtering

Labelling

[↩](#)
[→](#)
[✎](#)
[✕](#)
[⌵](#)
[⌶](#)
[🔍](#)
[🔍](#)
[📄](#)
[Reset Trrack](#)
[Delete datasets](#)



Dataframe name...

[✎](#)
[persist_df_2](#)
[📄](#)
[+](#)

[🔗 Trrack](#)
[☰ Summary](#)

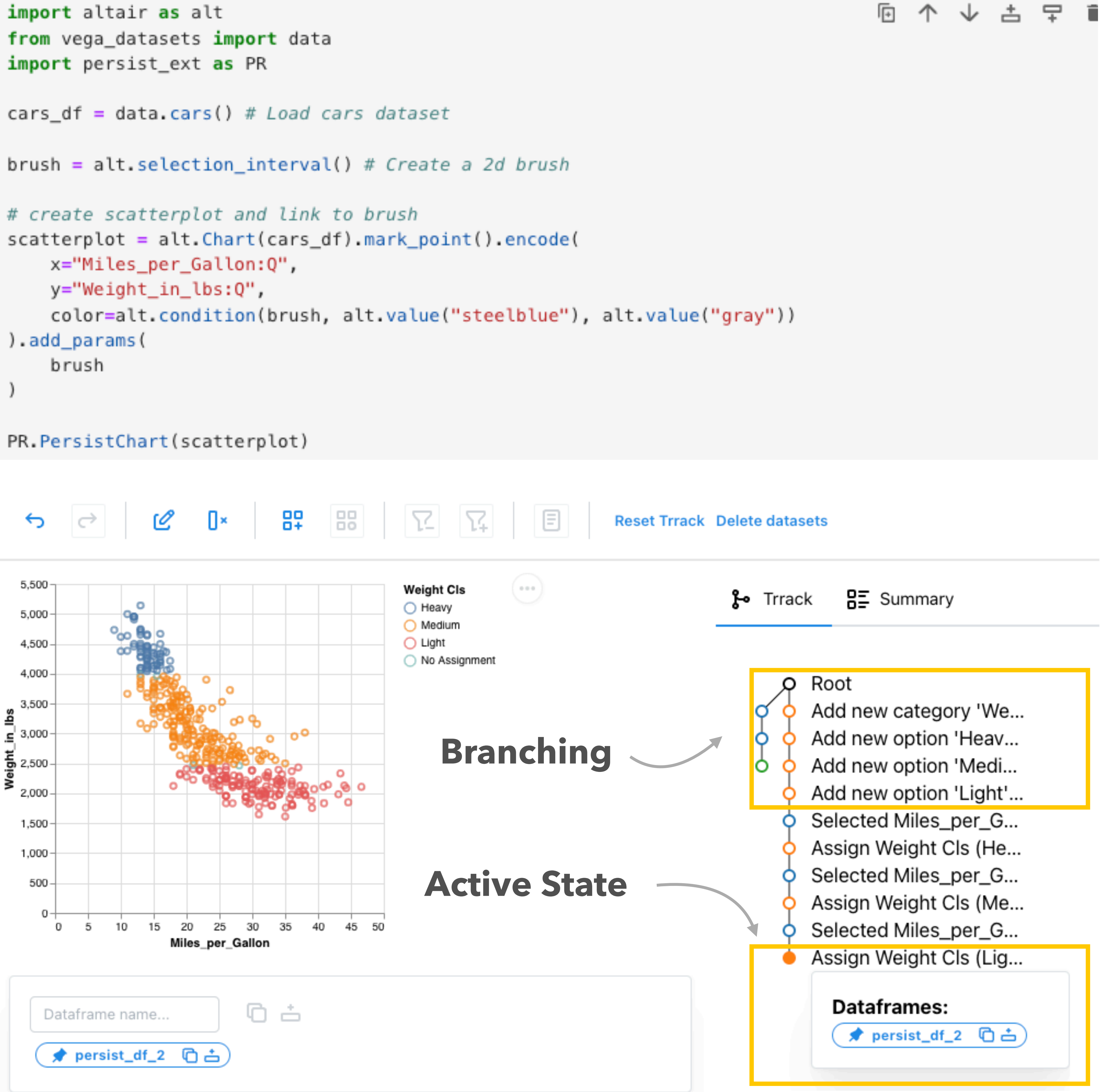
- Root
- Add new category 'We...
 - Add new option 'Heav...
 - Add new option 'Medi...
 - Add new option 'Light'...
 - Selected Miles_per_G...
 - Assign Weight Cls (He...
 - Selected Miles_per_G...
 - Assign Weight Cls (Me...
 - Selected Miles_per_G...

Dataframes:

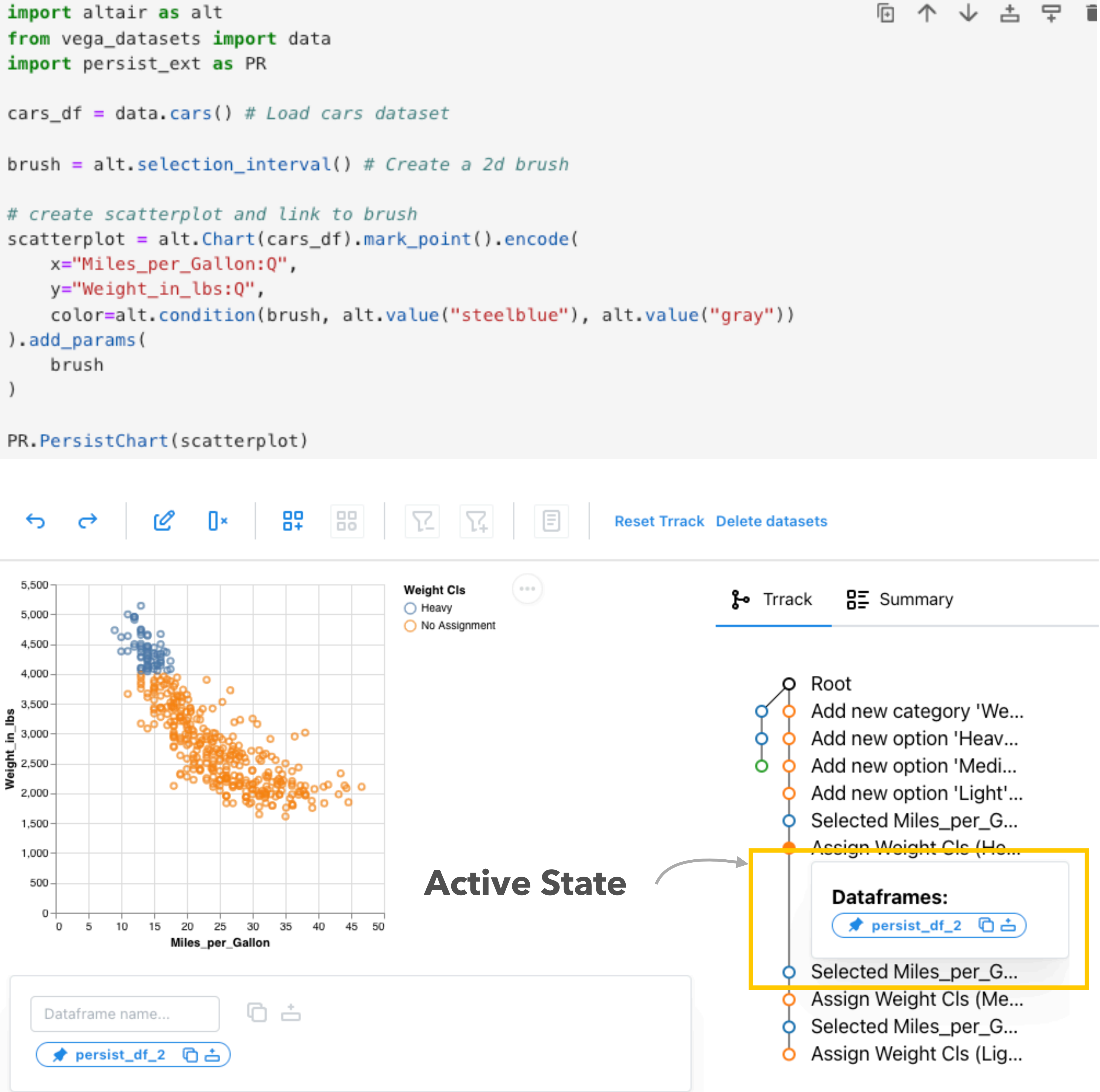
[✎ persist_df_2](#)
[📄](#)
[+](#)

Assign Weight Cls (Lig...

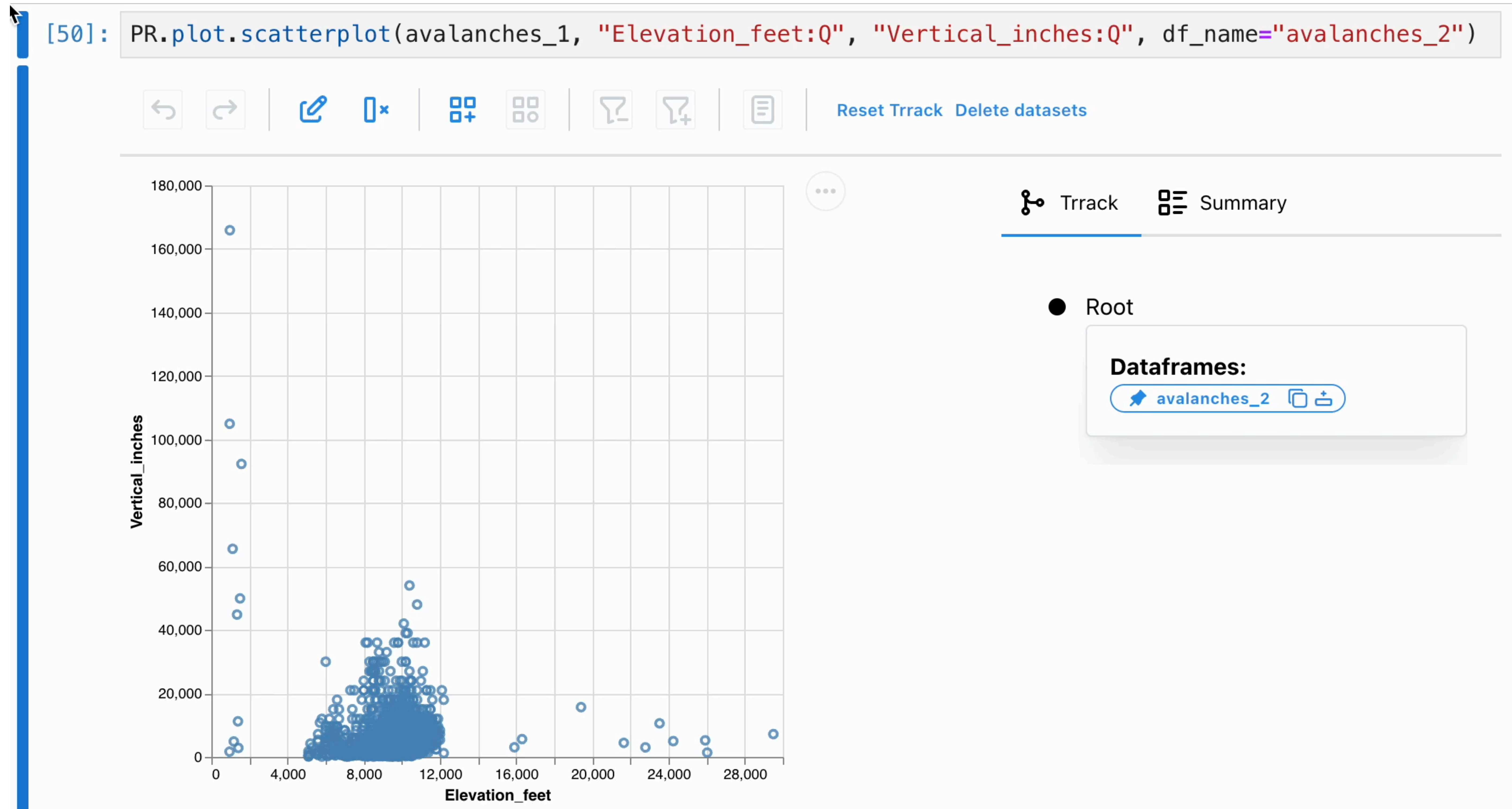
Branches and **Choosing a State** in
provenance support *non-linear*
analysis, addressing **the layout gap**



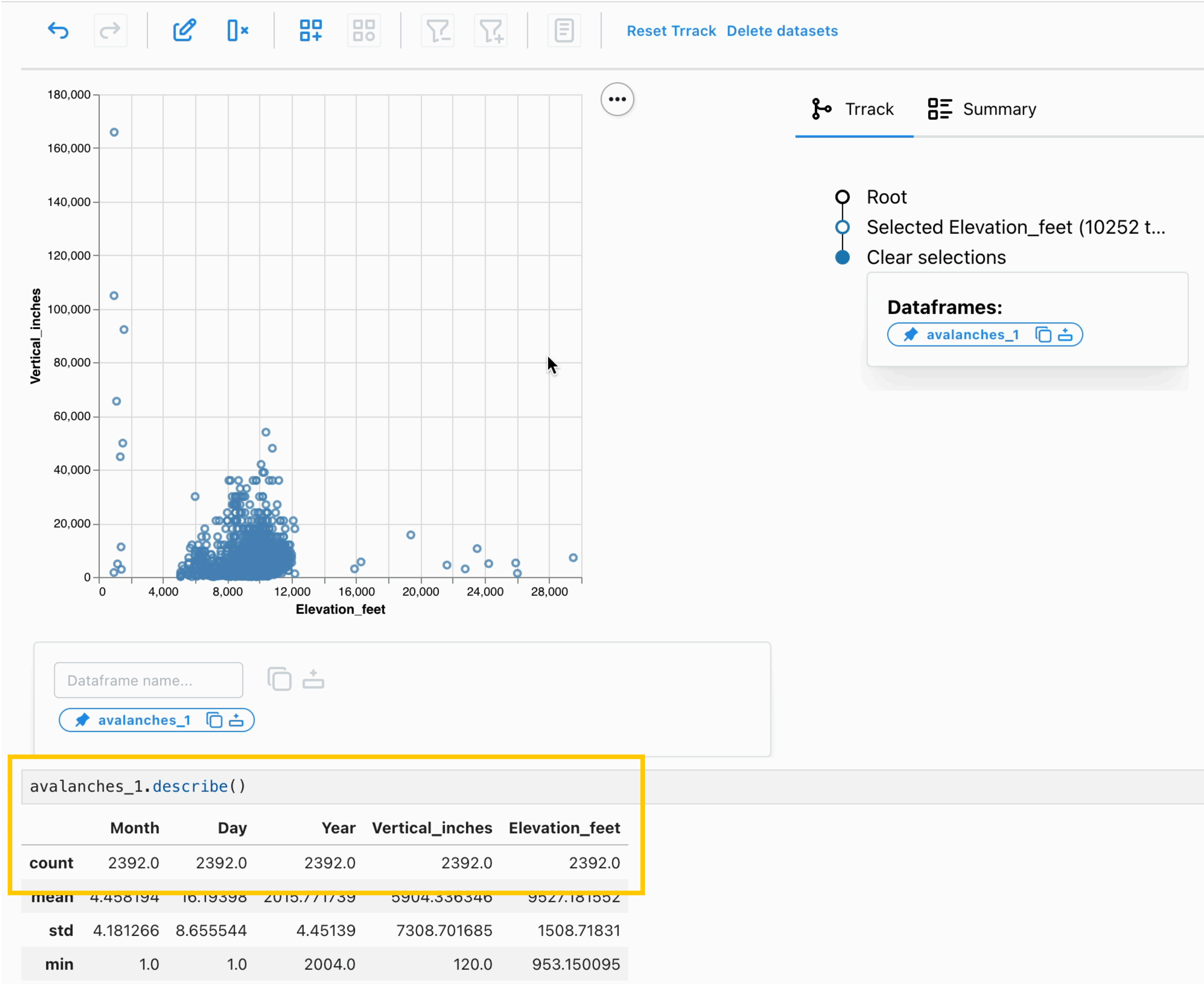
Branches and **Choosing a State** in
provenance support *non-linear*
analysis, addressing **the layout gap**



Persist **re-runs the interactions** in the output,
addressing **the temporal gap**



Persist *applies interactions to data frames* that can be accessed in code, addressing the **semantic gap**



Manipulated
data frame accessed in
code

count: 2392→79

VISUALIZATION OPTIONS



Arbitrary Vega-Altair Charts

The screenshot displays an interactive data table interface. The table has columns: #, Date, Place, Trigger, Weak Layer, Depth_inches, Aspect, and Elevation_feet. The first three rows are visible: 2338 (3-23-2023, Dry Creek, Natural, West, 8000), 955 (1-19-2014, Whitney Basin, Snowmobiler, East, 10500), and 1028 (2-21-2014, Chalk Creek, Natural, Northeast, 10600). A 'Track' panel on the right shows a sequence of operations: Root, Rename column ;Trigger to..., Drop column Comment..., Updated column 'Depth_i...', Updated column 'Depth_i...', Updated column 'Depth_i...', Changed column 'Dept...', Sort (descending) by 'Dep...', Drop column Coordinates, Selected 1 point, Selected 2 points, Selected 3 points, Drop column ;Region, Drop column Width_inches, and Drop column Vertical_inch... A 'Dataframes' panel at the bottom shows 'current_df' as the current dataframe.

#	Date	Place	Trigger	Weak Layer	Depth_inches	Aspect	Elevation_feet
2338	3-23-2023	Dry Creek	Natural			West	8000
955	1-19-2014	Whitney Basin	Snowmobiler			East	10500
1028	2-21-2014	Chalk Creek	Natural			Northeast	10600
1024	2-17-2014	Upper Weber Canyon	Natural			Northeast	10400
998	2-12-2014	Upper Weber Canyon	Natural			Northeast	10400
938	1-14-2014	Upper Weber Canyon	Explosive			East	10300
1299	1-26-2016	Currant Creek Peak	Snowmobiler			Southwest	9500
1044	2-28-2014	Chalk Creek	Natural			Northeast	10600
2348	3-30-2023	Bunnels	Natural			Northeast	10800
1977	4-6-2021	Blue Ice	Natural			Northeast	10400

An Interactive Data Table

VEGA-ALTair

Persist works with **most Vega-Altair** charts

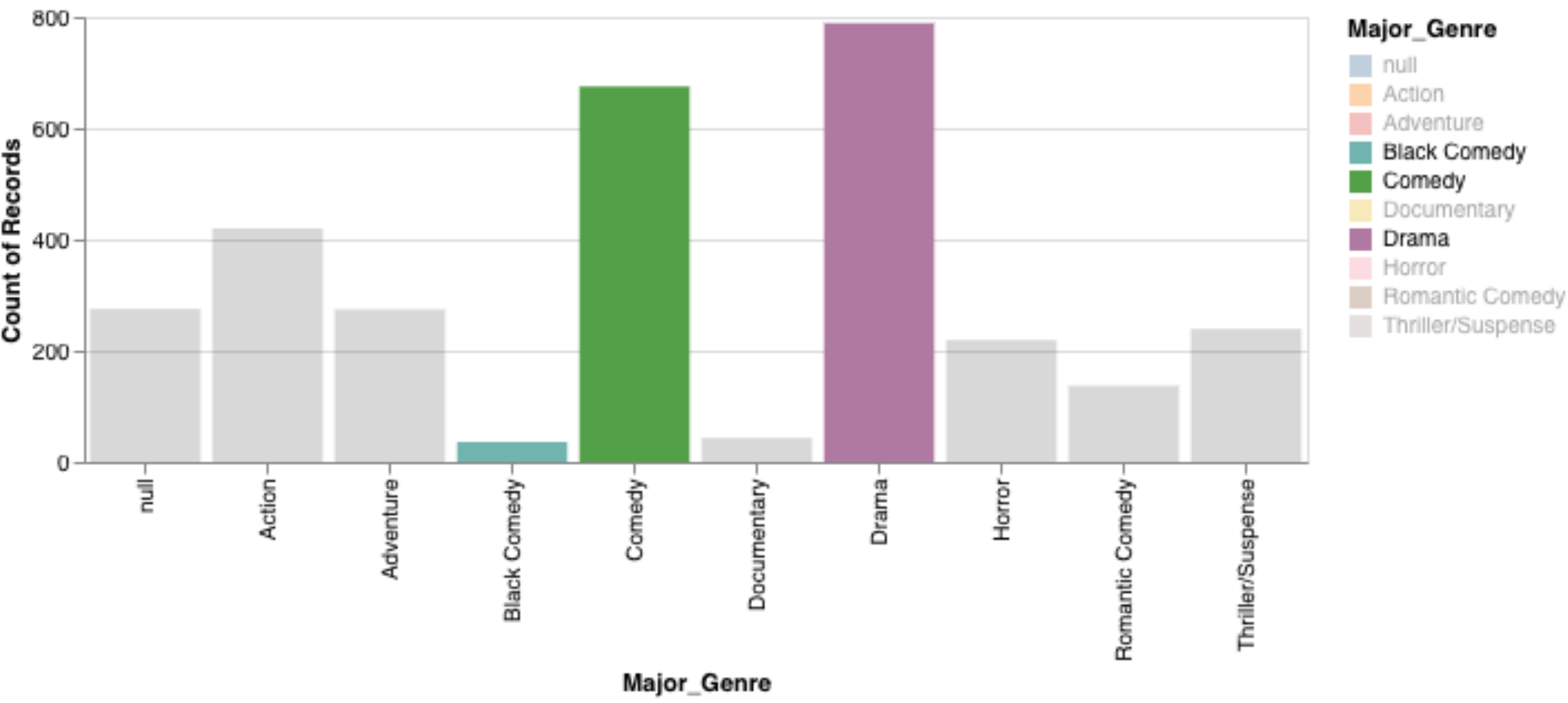
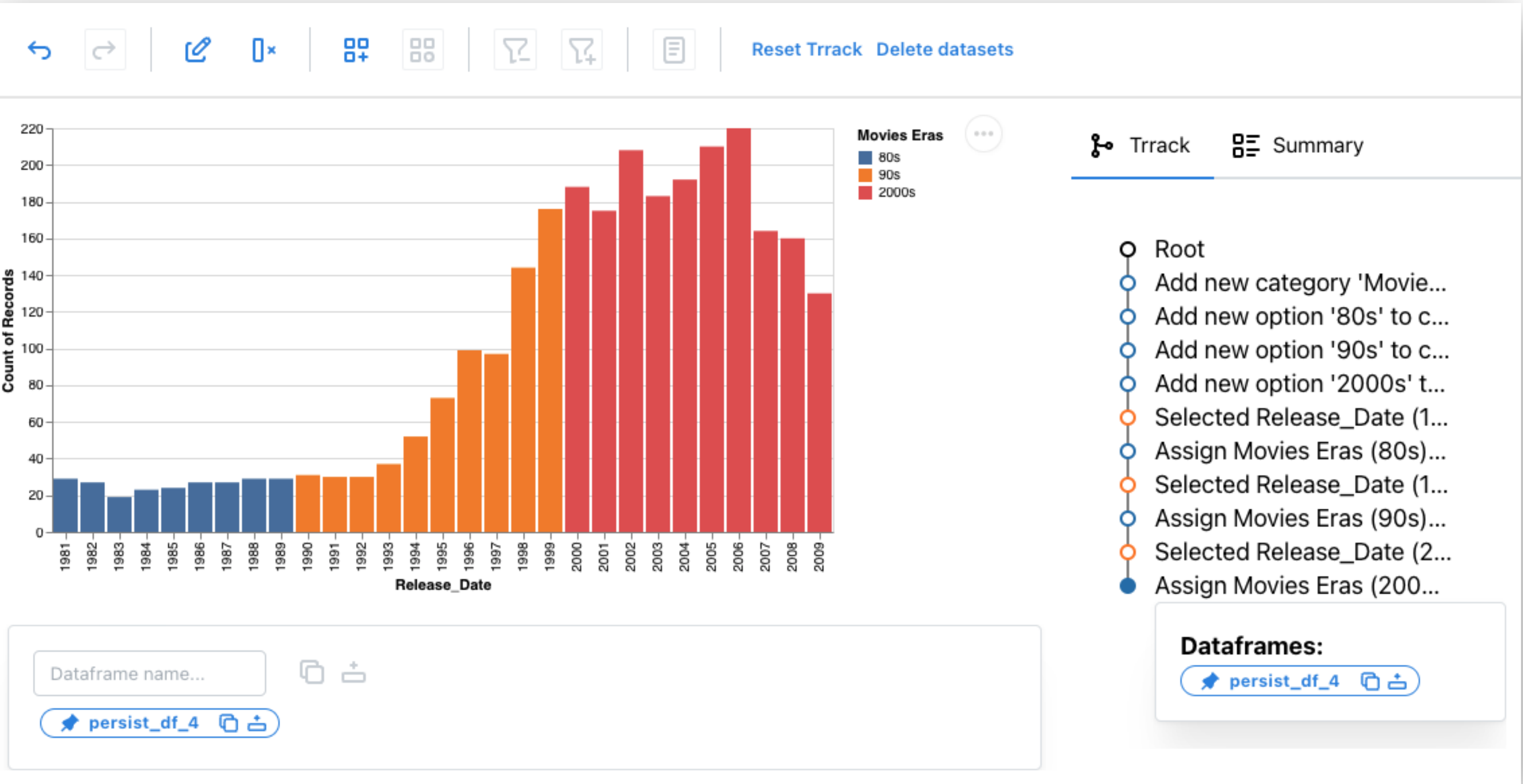
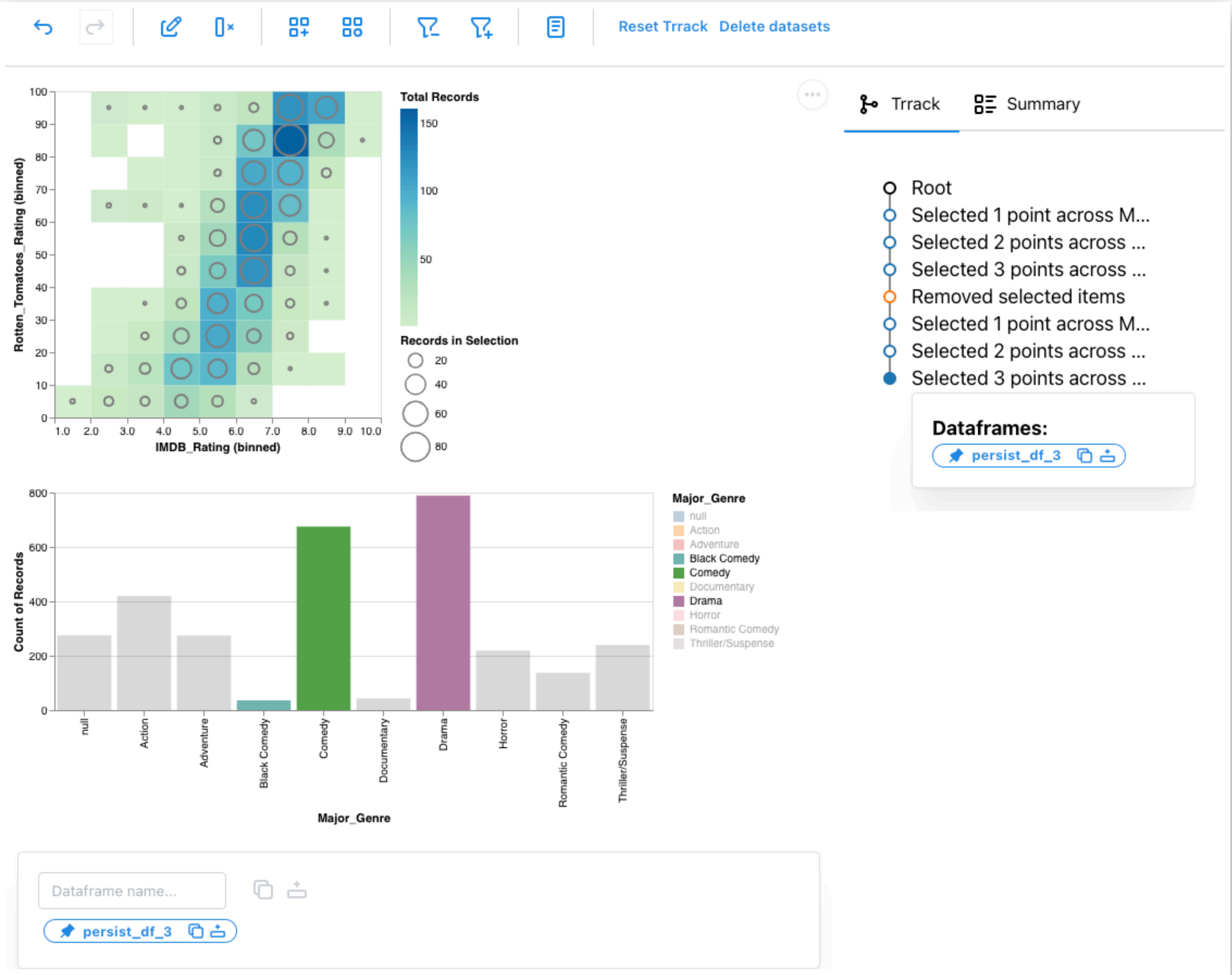
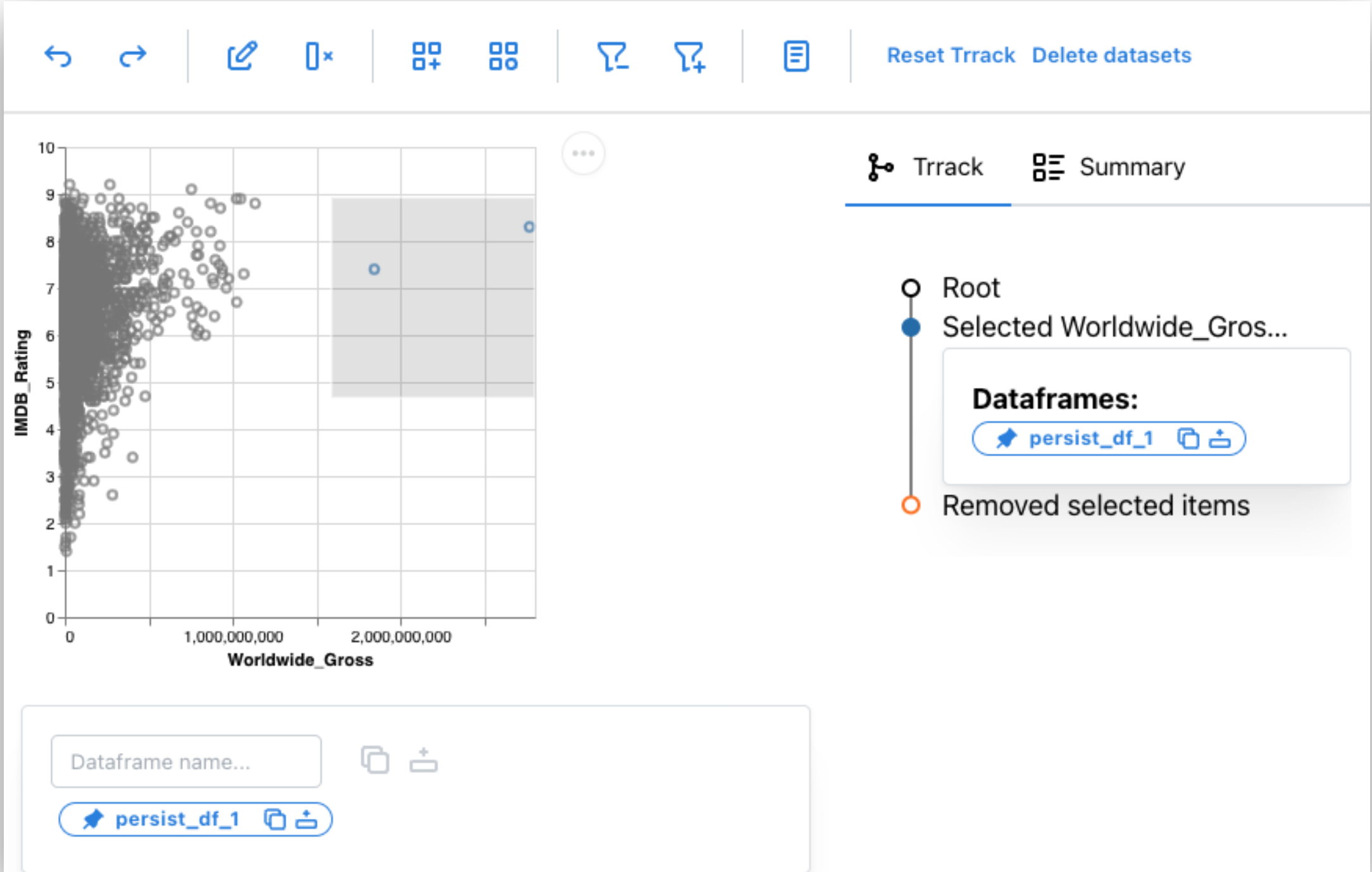
"Listens" to native operations (selections)

Updates Vega charts:

- Use original chart spec when possible (e.g., filters)

- Update spec when necessary (categories, labels)

EXAMPLE CHARTS



DEMO

EVALUATION

IN-LAB STUDY

III STUDY DESIGN

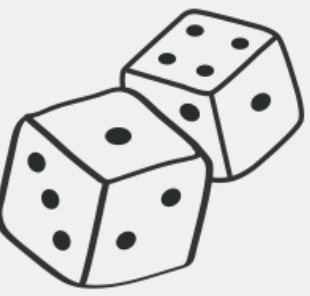
WE RECRUITED ELEVEN PARTICIPANTS FOR THE STUDY. PARTICIPANTS ALL HAD PRIOR EXPERIENCE WITH PYTHON AND PANDAS.

FULL FACTORIAL DESIGN

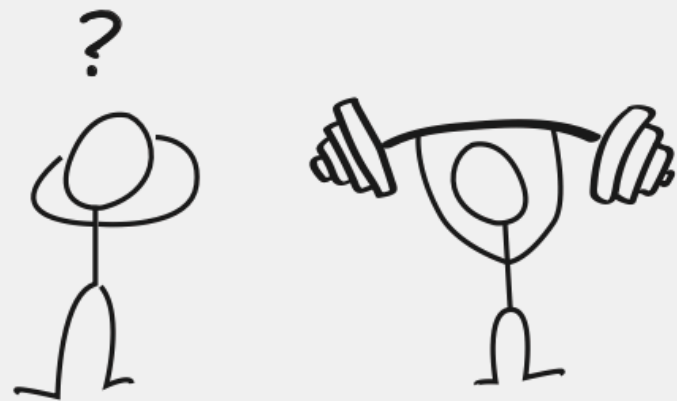
2 DATASETS **X** 2 CONDITIONS **X**

THE ORDER OF CONDITIONS WAS RANDOMLY ASSIGNED.

FOR EACH CONDITION, DATASETS WERE RANDOMLY ASSIGNED. PARTICIPANTS NEVER SAW THE SAME DATASET TWICE



— STUDY METRICS —



SUBJECTIVE PERFORMANCE



TIME



ERROR



REPRODUCIBILITY

— CONDITIONS —

PANDAS CODING

```
[ 1 ]  IMPORT PANDAS AS PD
        DF = PD.READ_CSV(...)
```

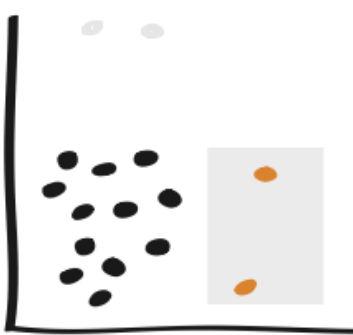
```
[ 2 ]  DF = DF.DROP("AGE")

        DF = DF.RENAME(
            COLUMNS={'JOB': 'PROFESSION'}
        )

        DF.LOC[1, 'JOB'] = 'PRINCIPAL'
```

PERSIST EXTENSION

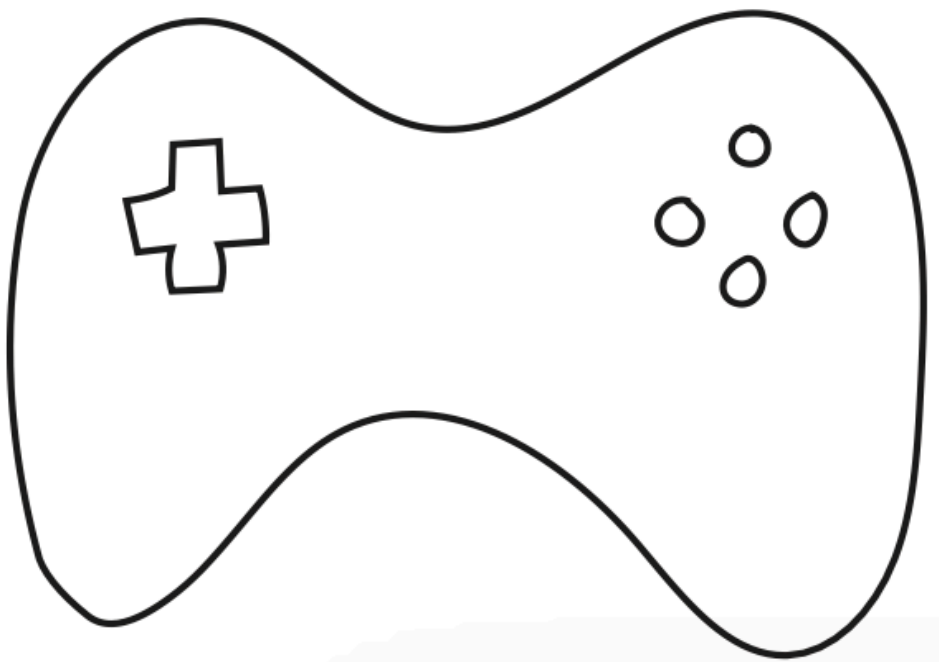
```
[ 1 ]  IMPORT PERSIST_EXT AS PR
        IMPORT PANDAS AS PD
        DF = PD.READ_CSV(...)
        PR.PLOT.SCATTERPLOT(...)
```



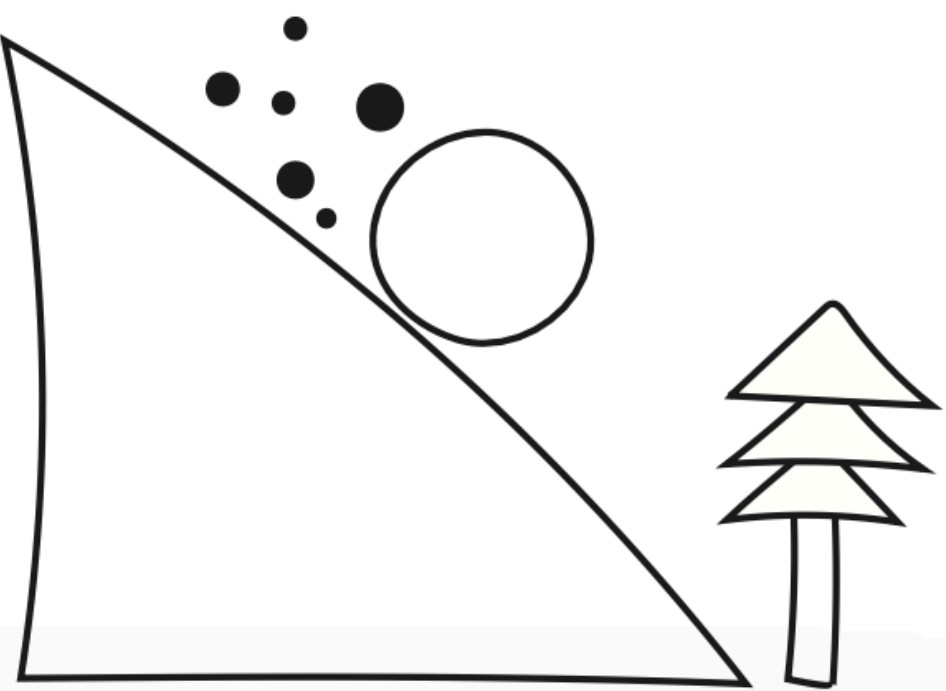
SELECT 2 VALUES
REMOVE 2 VALUES
SELECT 2 VALUES

— DATASETS —

VIDEO GAMES



AVALANCHES



IN-LAB STUDY

TASKS

PARTICIPANTS MADE THE FOLLOWING CHANGES TO A DATASET

NAME	AGE	JOB
STEVE	32	PLUMBER
JILL	24	TEACHER
ANN	42	ENGINEER

REMOVE COLUMNS

NAME	AGE	PROFESSION
STEVE	32	PLUMBER
JILL	24	TEACHER
ANN	42	ENGINEER

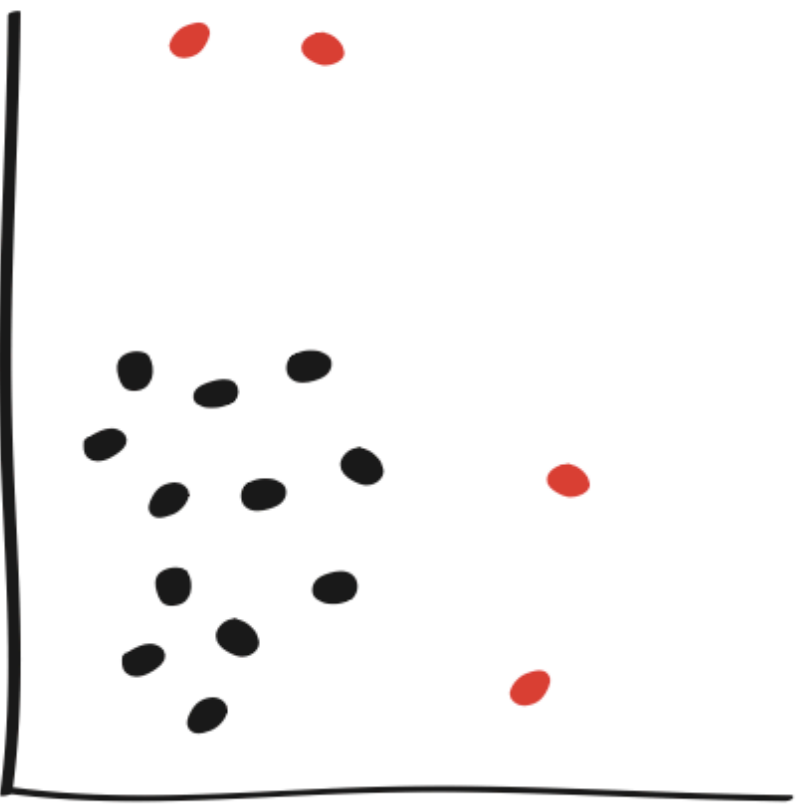
RENAME COLUMNS

AGE	AGE
"32"	32
"24"	24
"42"	42

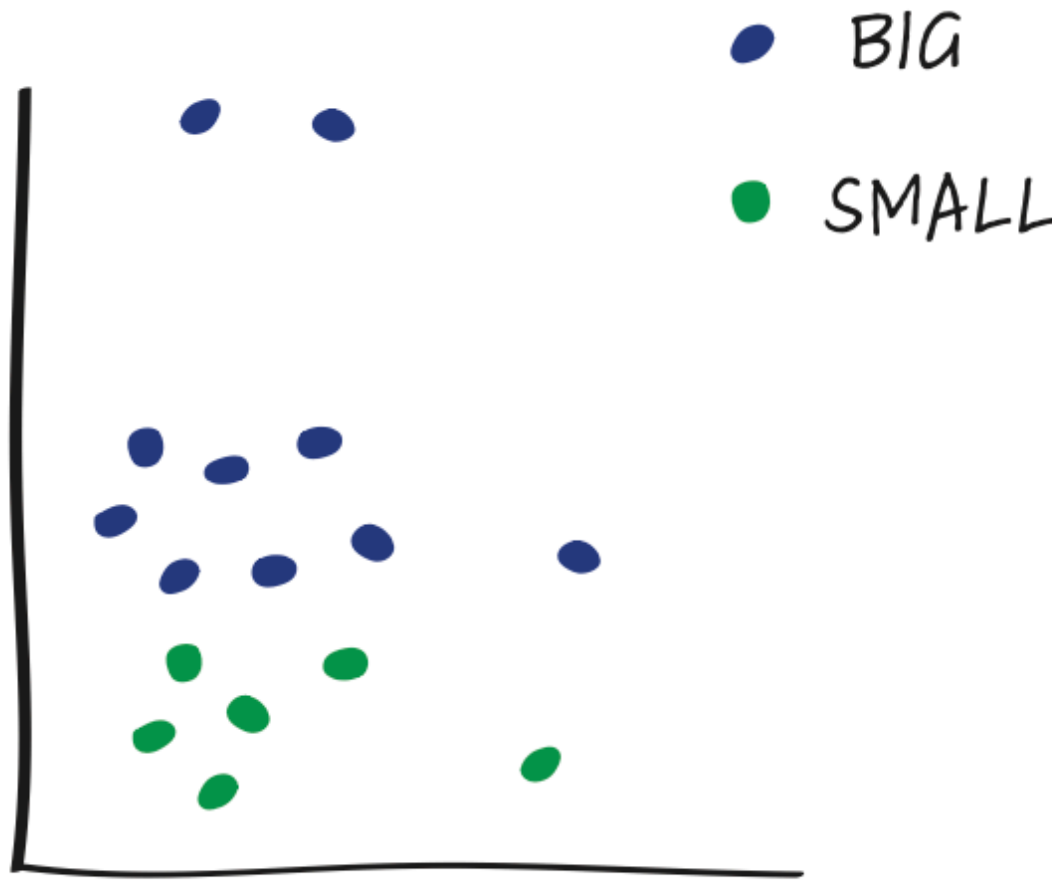
CHANGE DATA TYPE

NAME	AGE	JOB
STEVE	32	PLUMBER
JILL	24	PRINCIPAL
ANN	42	ENGINEER

EDIT VALUES



FILTER DATA



ADD CATEGORICAL
COLUMN

RESULTS

3x

times faster
with Persist

97%

tasks **correctly** using Persist,
compared to 85% for Pandas

11/11

notebooks using Persist
were **reproducible**

only 7/11 using pandas
were

RESULTS

QUOTES

“so much easier than manually coding.”
- M4

“easier as compared to the code and everything was visible [...] and it didn’t take much time.”
- M2

“Changing the category type, or adding new categories or removing anomalies from data, they were very much easier in [Persist] than coding.”
- M7

“The thing I really liked about is version control, which shows the history of all operations [...] and also saves the changes [...] into a data frame.”
- M14

DISCUSSION

GENERATING CODE VS PROVENANCE TRACKING

Provenance **better for most cases**

No code clutter

Undo/redo

Consistent semantics

But code generation might be

more robust

works outside of notebooks

works w/o the library

GENERATE CODE ON DEMAND!

[Beta]

Reset Ttrack Delete datasets

Search

#	Region	Year	;Trigger	;Weak Layer	Dep
3	Salt Lake	2012	Skier		Facets
4	Salt Lake	2012	Skier		New Snow
5	Salt Lake	2012	Skier		Facets
6	Salt Lake	2012	Skier		New Snow/Old Snow Interface
7	Salt Lake	2012	Skier		Facets
8	Salt Lake	2012	Skier		Facets
9	Salt Lake	2012	Skier		Facets
10	Salt Lake	2012	Skier		Facets
11	Salt Lake	2012	Skier		Facets
12	Salt Lake	2012	Unknown		Ground Interface

Rows per page

10

1-10 of 2,390

Ttrack Summary

Dataframes:
[persist_df_1](#)

Dataframe name...

[persist_df_1](#)

```
def create_persist_df_1(df):
    df = df.copy(deep=True)

    # Add "__id_column" as the ID column
    df.insert(0, "__id_column", df.index + 1)
    df["__id_column"] = df["__id_column"].astype(str)

    # Add selection column
    df["__selected"] = False

    # Rename column
    df = df.rename(columns={';Region': 'Region'})
    df.loc[df["__id_column"].isin(['2']), "__selected"] = True
    df.loc[df["__id_column"].isin(['1', '2']), "__selected"] = True
```

Copy

Insert

BEYOND JUPYTER



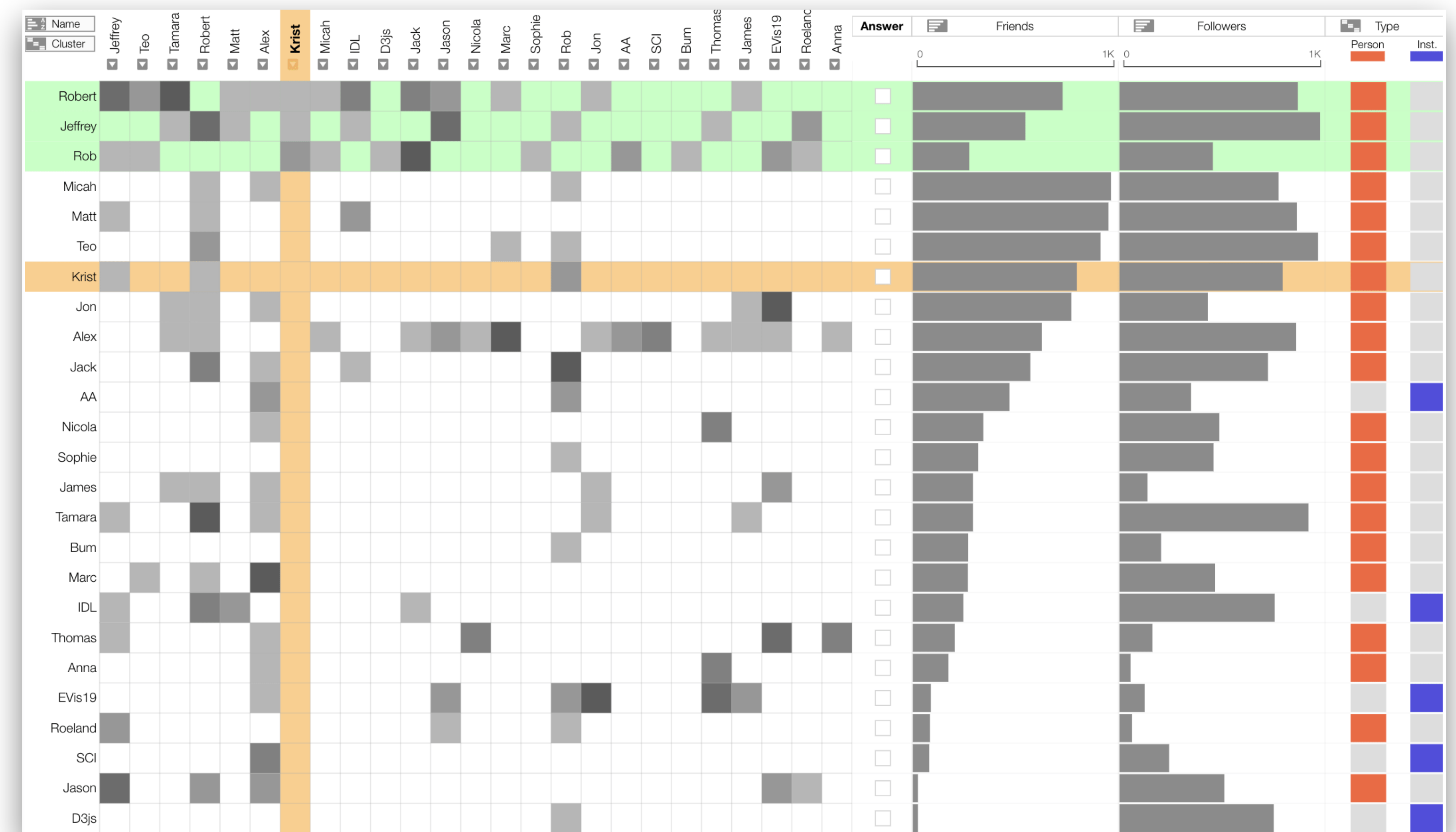
DIFFERENT DATA TYPES
DIFFERENT CHARTING LIBRARIES

Other Interactive **Plotting Libraries**

Plot.ly, Bokeh

Other **data types**

Maps, Networks, ...



TRY OUT PERSIST!

Persist is **available now!**

<https://vdl.sci.utah.edu/persist/>

Documentation & examples

Feedback / bug reports
appreciated!

README BSD-3-Clause license

Getting Started

Requirements

- JupyterLab >= 4.0.0 or Jupyter Notebook >= 7.0.0
- pandas >= 0.25
- altair >= 5
- ipywidgets
- anywidget

Install

To install the extension, execute:

```
pip install persist_ext
```

If the Jupyter server was already running, you might have to reload the browser page and restart the kernel.

Uninstall

To remove the extension, execute:

```
pip uninstall persist_ext
```

Usage

Persist supports two types of interactive outputs — a custom data table and [Vega-Altair](#) (>=5.0.0, see [requirements](#) and [caveats](#)) charts. The following examples will walk you through creating each one. The examples are also available as notebooks in the `examples` folder of the repository. Each section will link to the corresponding notebook as well as a binder link for the notebook.

Persist currently works with pandas dataframes, so load/convert the data to pandas dataframe before using.

Examples

Kiran Gadhave, Zach Cutler, **Alexander Lex**

<http://vdl.sci.utah.edu>



Persist. Persistent and Reusable Interactions in Computational Notebooks



visualization
design lab



www.sci.utah.edu

Thanks to Jake Wagoner
Thanks to the NSF for Funding
(IIS 1751238, CNS 213756,
and CNS-2313998)

