



Lux: Towards Intelligent Visual Data Discovery

<https://github.com/lux-org/lux>

Doris Jung-Lin Lee, Thyne Boonmark, Aditya Parameswaran (Berkeley)
Vidya Setlur, Melanie Tory (Tableau Research)
Jaewoo Kim, Karrie Karahalios (UIUC)

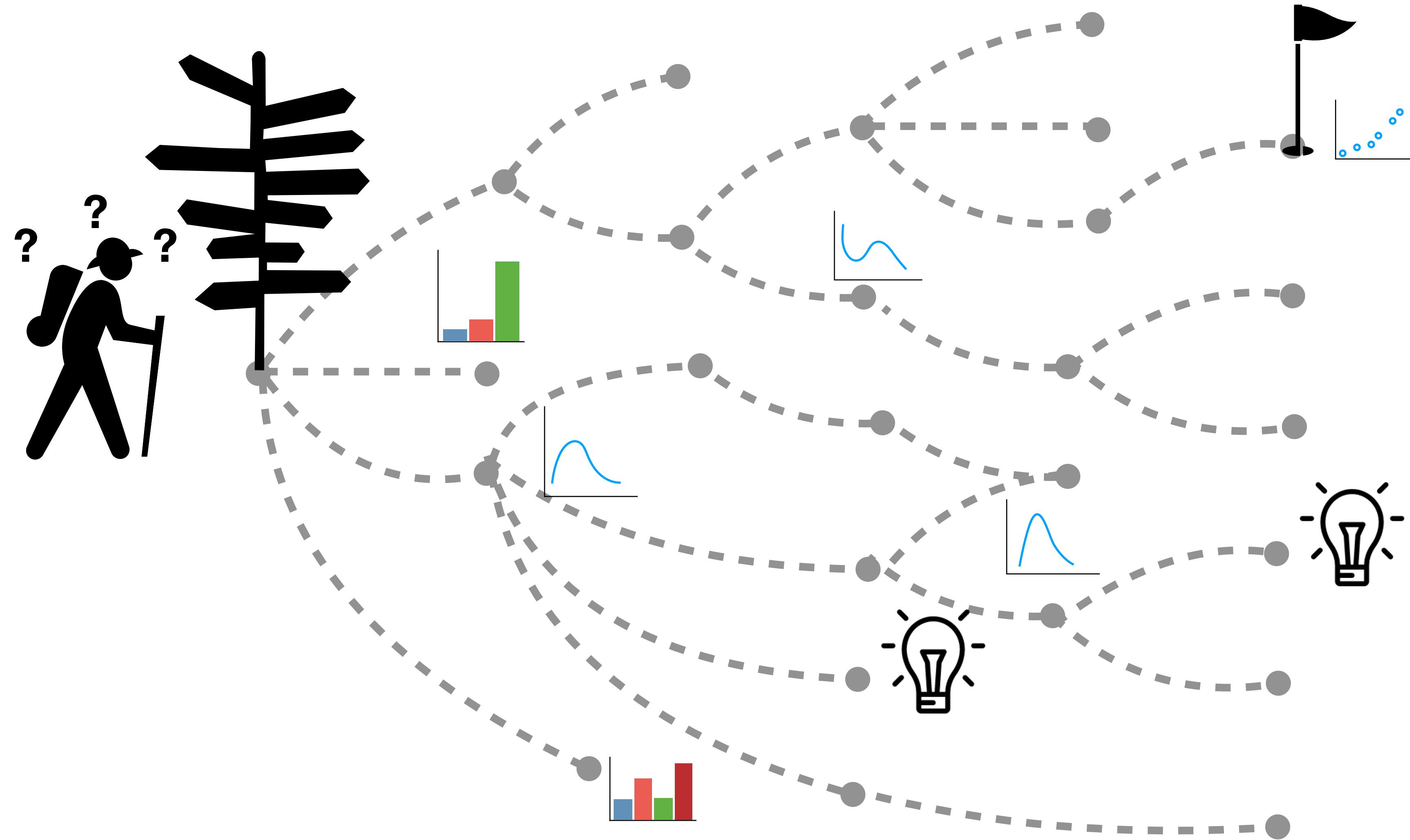


“The purpose of computing is insight, not numbers”

–Richard Hamming



Data exploration is tedious, iterative, and overwhelming

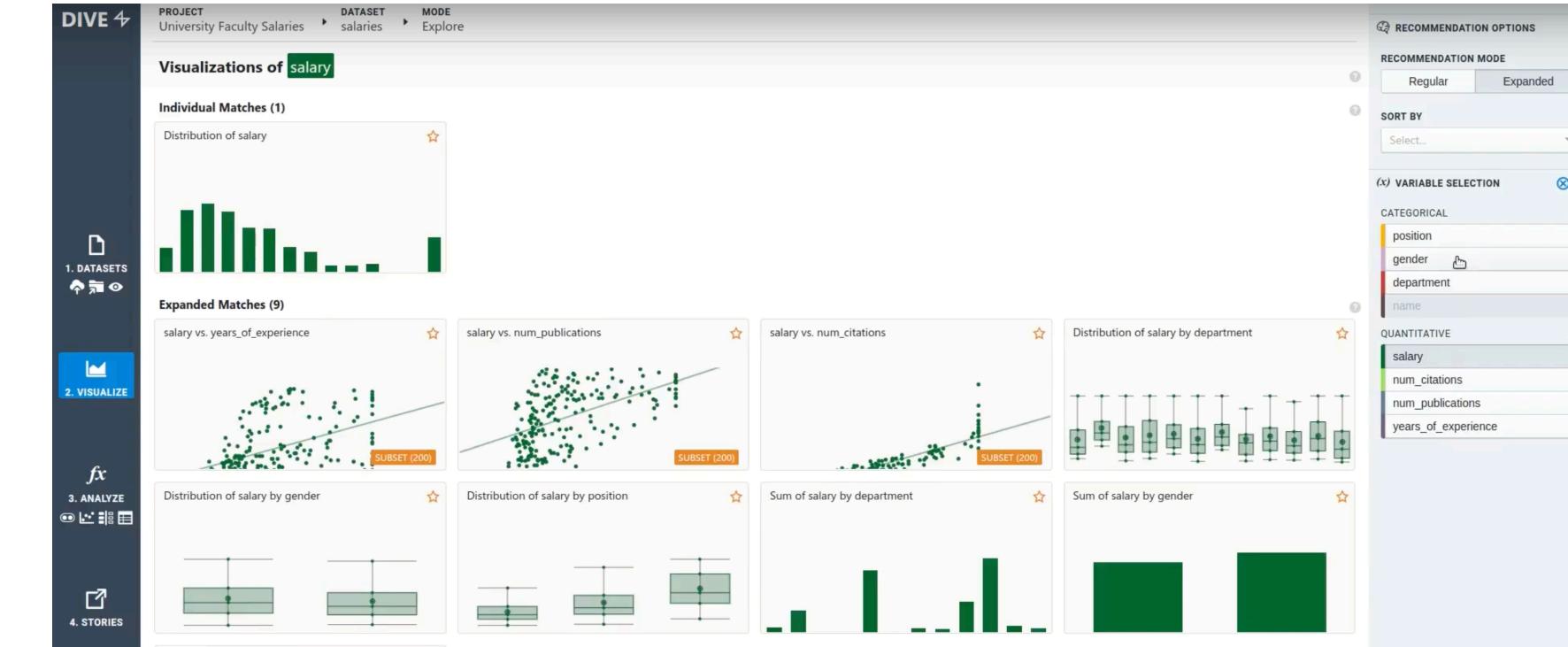


Visualization Recommendation Systems

Suggest interesting patterns and trends in data



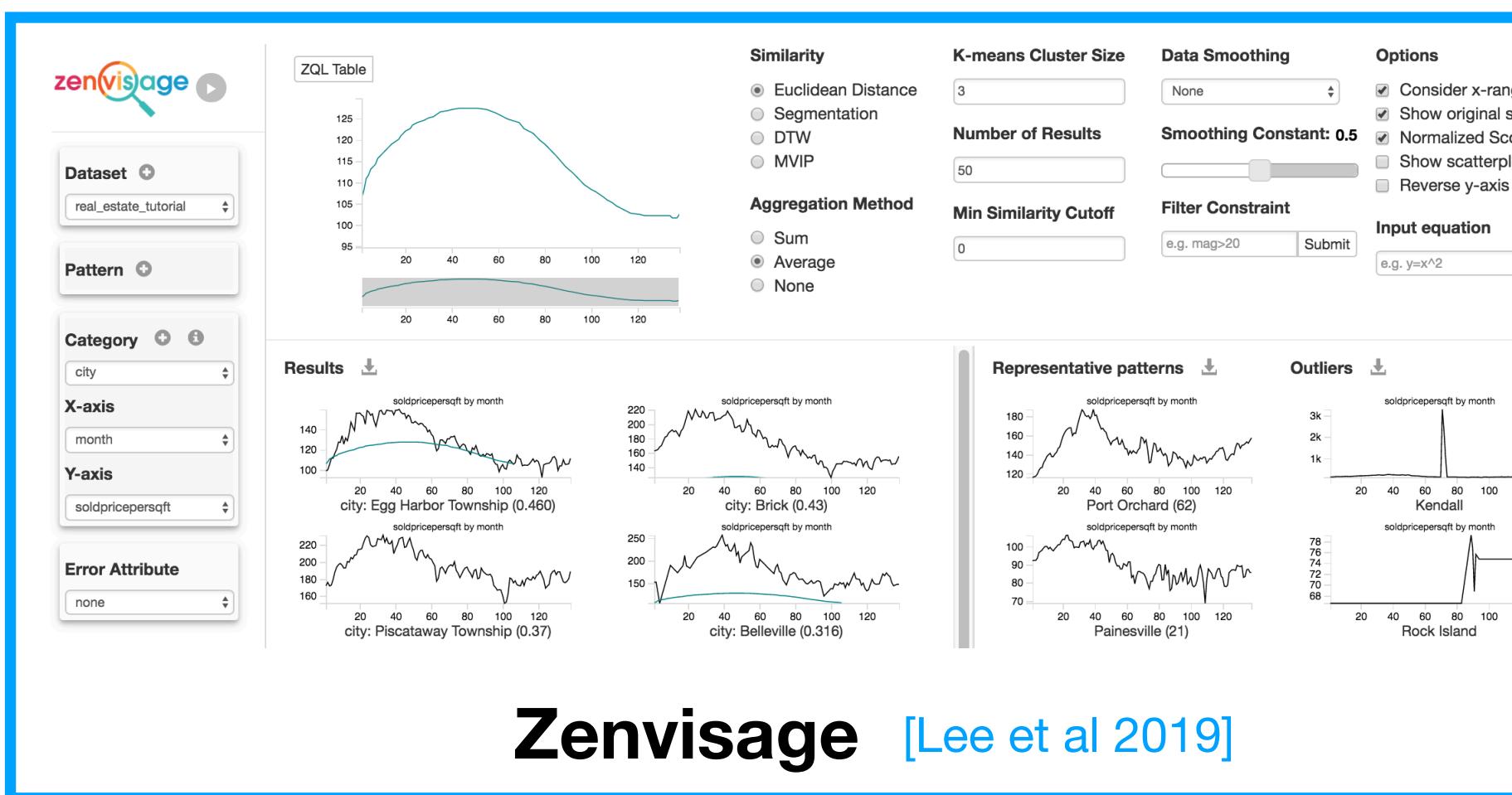
Voyager [Wongsuphasawat et al 2017]



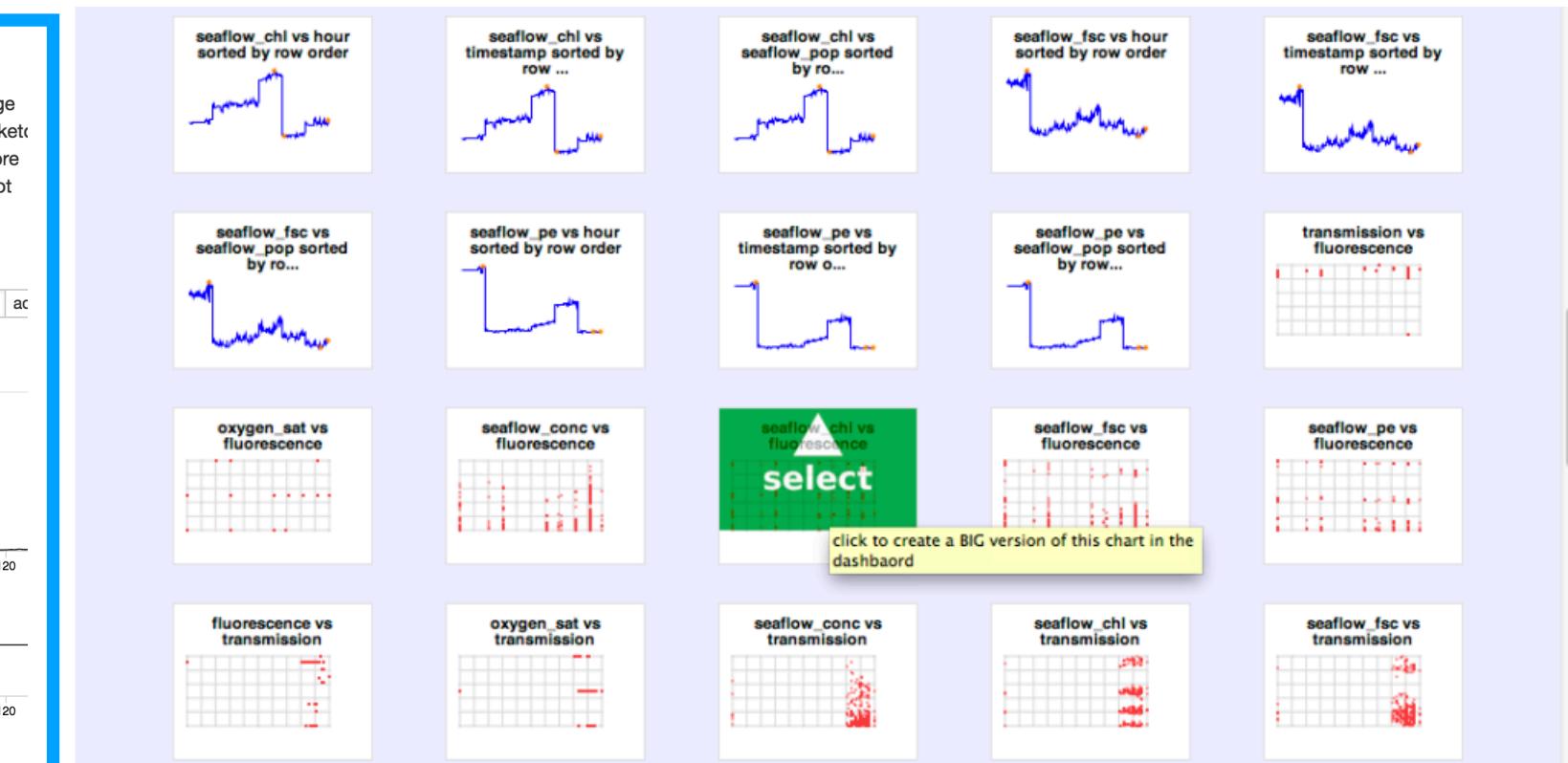
DIVE [Hu et al 2018]



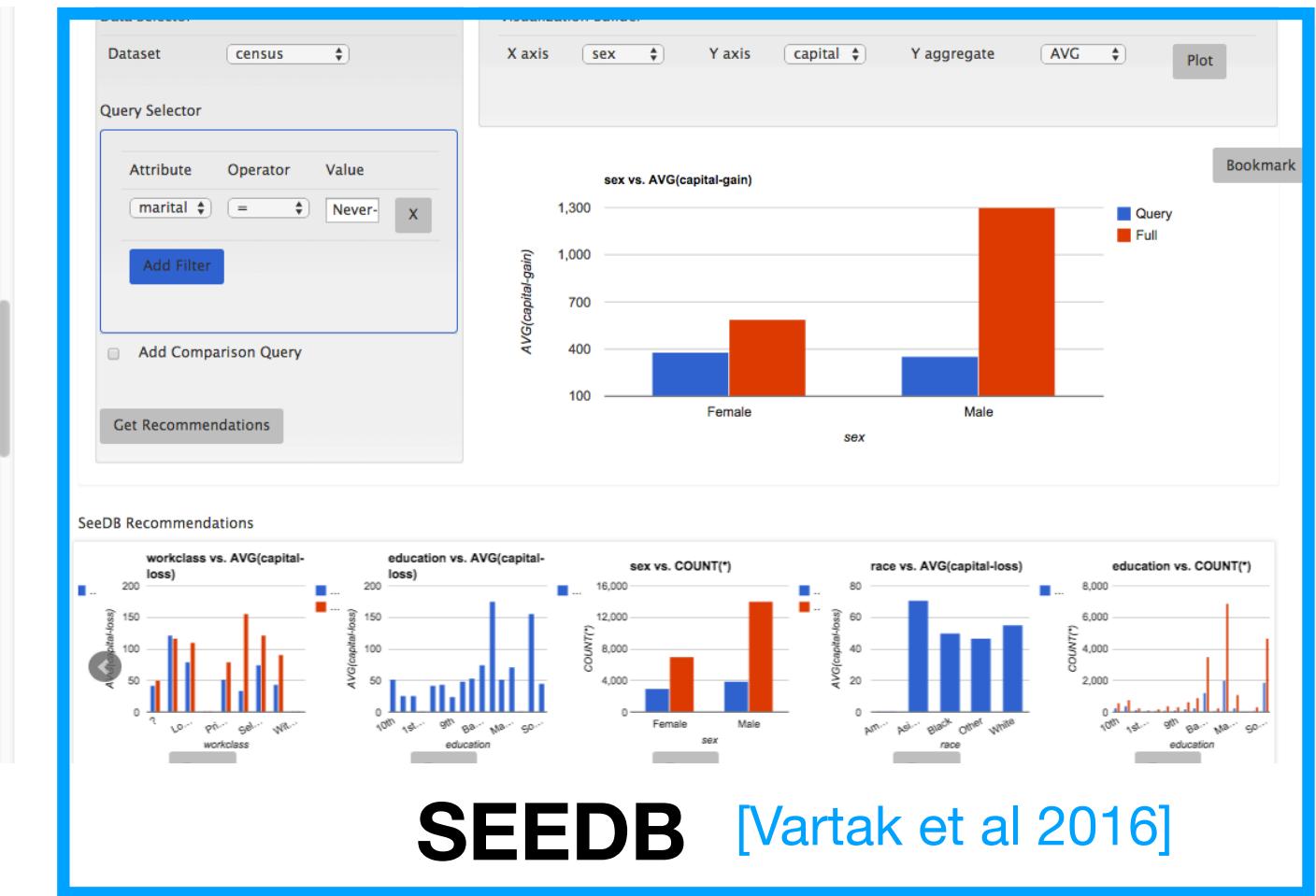
VisPilot [Lee et al 2019]



Zenvisage [Lee et al 2019]



VizDeck [Key et al 2012]



SEEDB [Vartak et al 2016]

Challenge #1: Limited Task Support

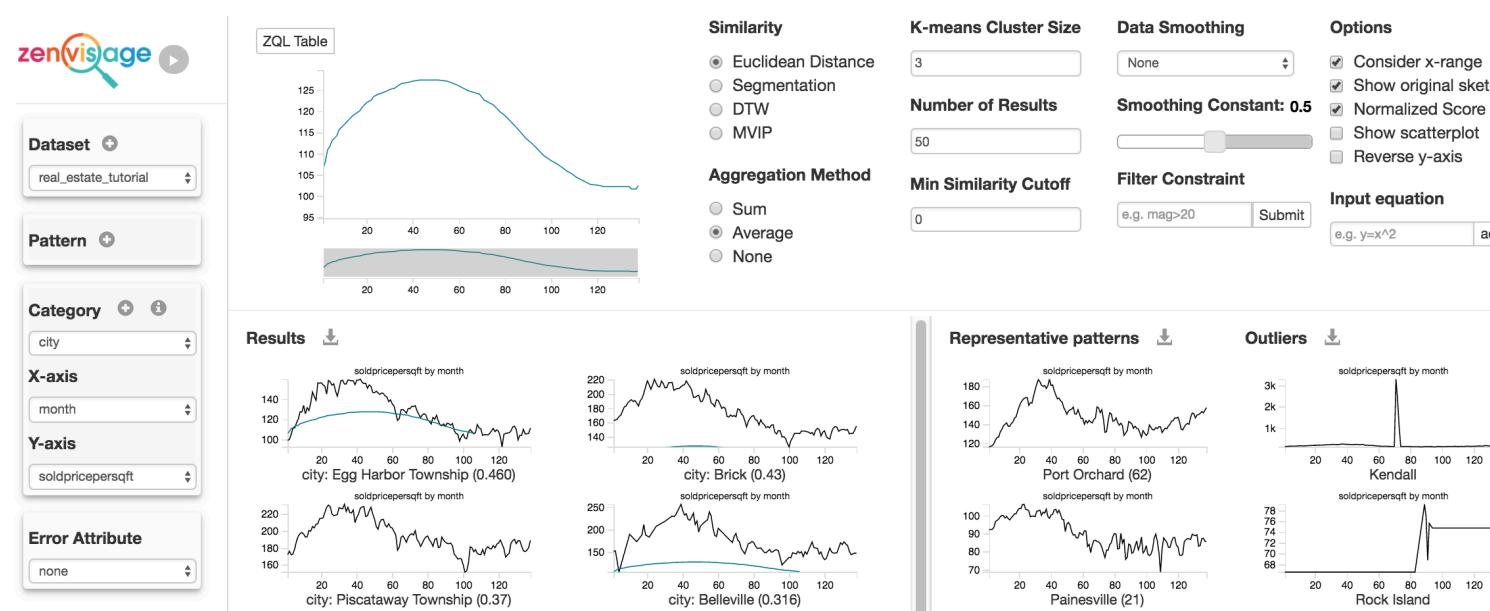
Voyager

[Wongsuphasawat et al 2017]



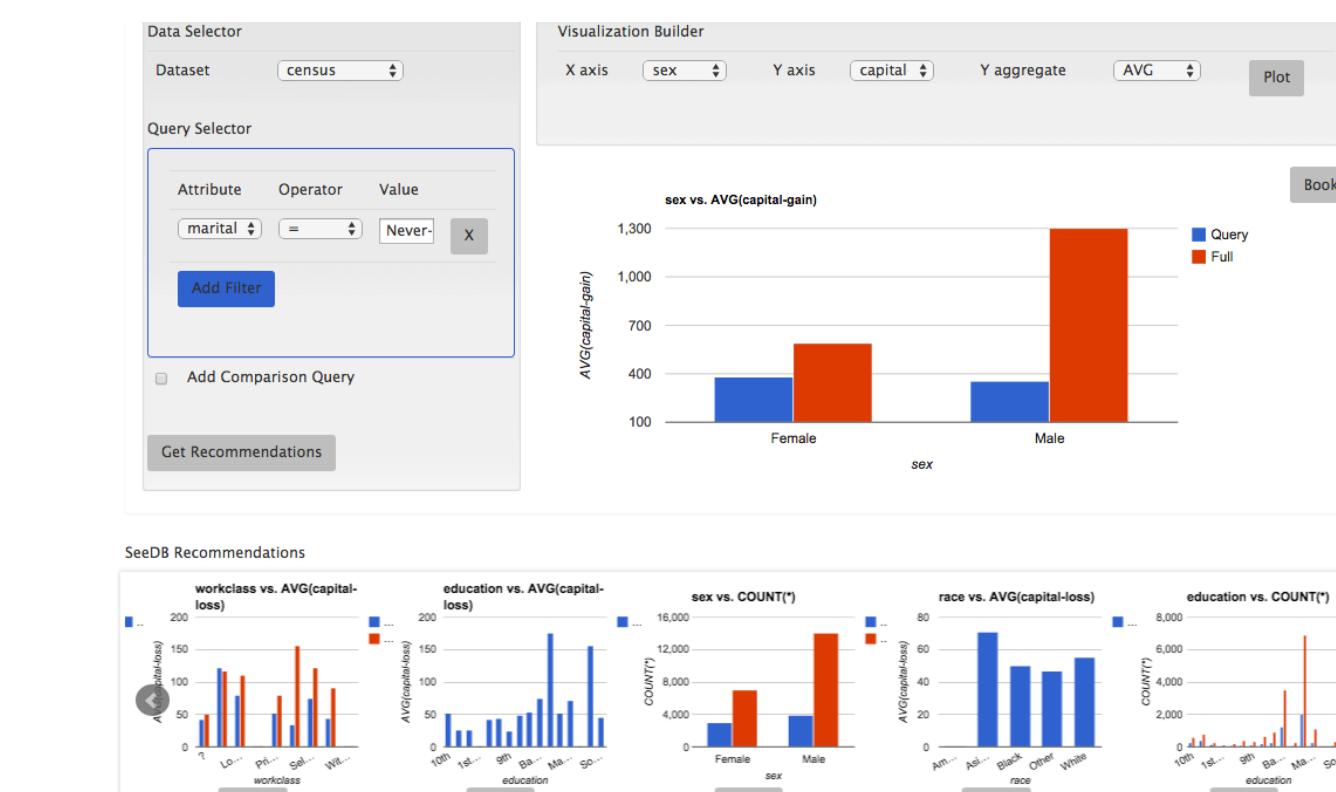
Zenvisage

[Lee et al 2019]

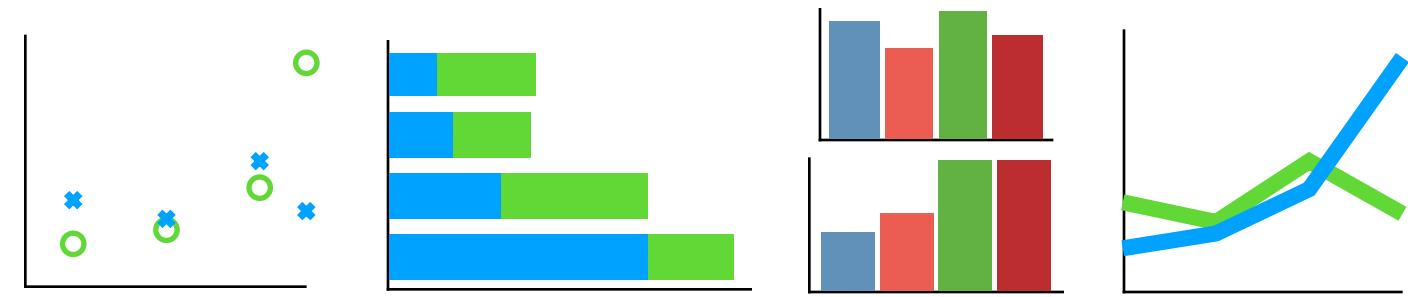


SEEDB

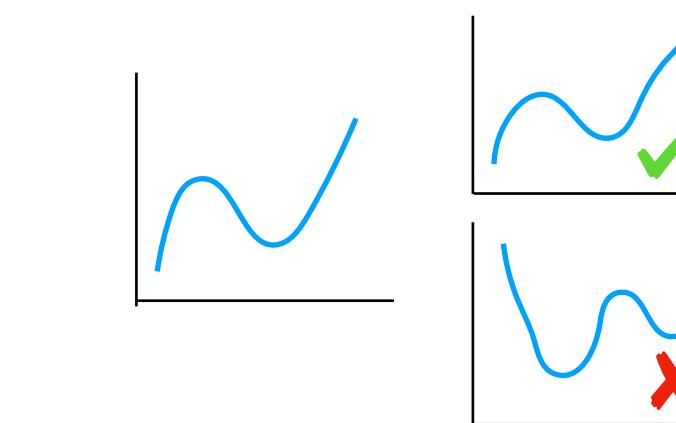
[Vartak et al 2016]



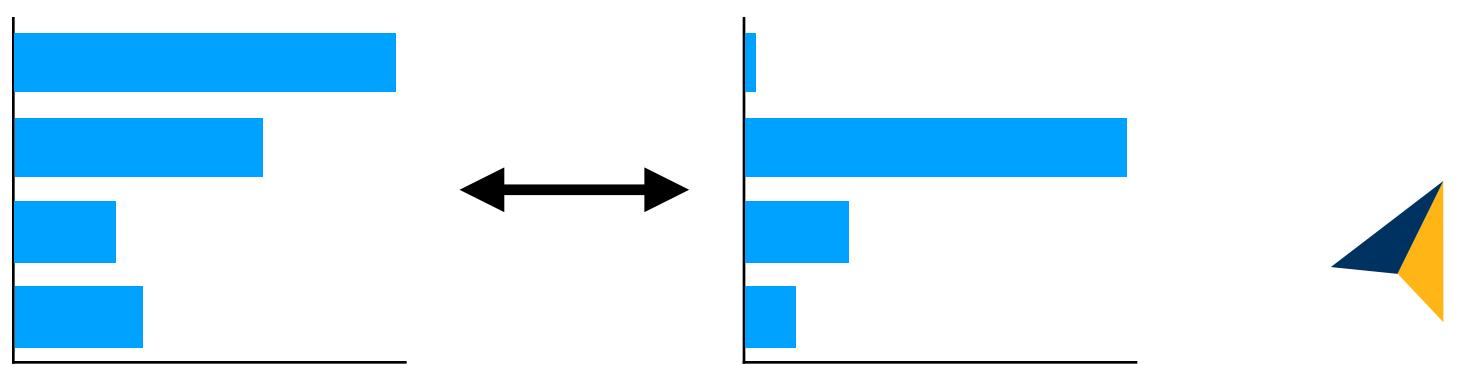
Perceptual effectiveness



Similarity



Deviation



Challenge #2: Disconnected Workflow

jupyter ChurnAnalysis Last Checkpoint: a minute ago (autosaved) 

File Edit View Insert Cell Kernel Widgets Help Not Trusted

In [1]: `from datetime import datetime, timedelta, date
import pandas as pd
%matplotlib inline
from sklearn.metrics import classification_report,confusion_matrix
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from __future__ import division
from sklearn.cluster import KMeans`

In [2]: `import chart_studio.plotly as py # DORIS: change to make this import work
import plotly.plotly as py
import plotly.offline as pyoff
import plotly.graph_objs as go`

In [3]: `import xgboost as xgb
from sklearn.model_selection import KFold, cross_val_score, train_test_split`

In [4]: `pyoff.init_notebook_mode()`

In []: `df_data = pd.read_csv('churn_data.csv')`

In []: `df_data.head(10)`

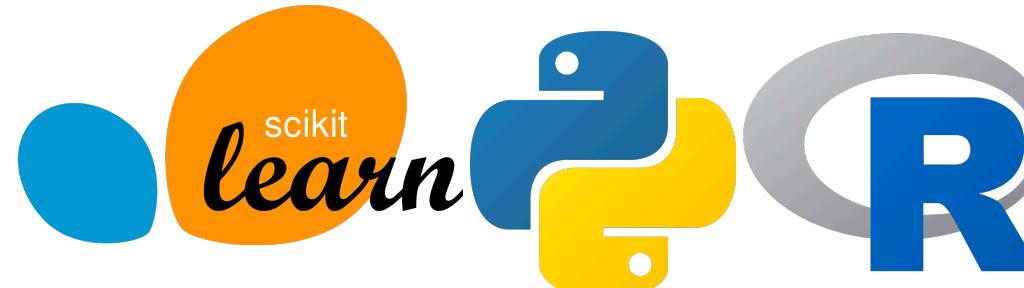
In []: `df_data.info()`

In []: `df_data.loc[df_data.Churn=='No','Churn'] = 0
df_data.loc[df_data.Churn=='Yes','Churn'] = 1`

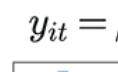
In []: `df_data.groupby('gender').Churn.mean()`

In []: `df_plot = df_data.groupby('gender').Churn.mean().reset_index()
plot_data = [
 go.Bar(
 x=df_plot['gender'],`



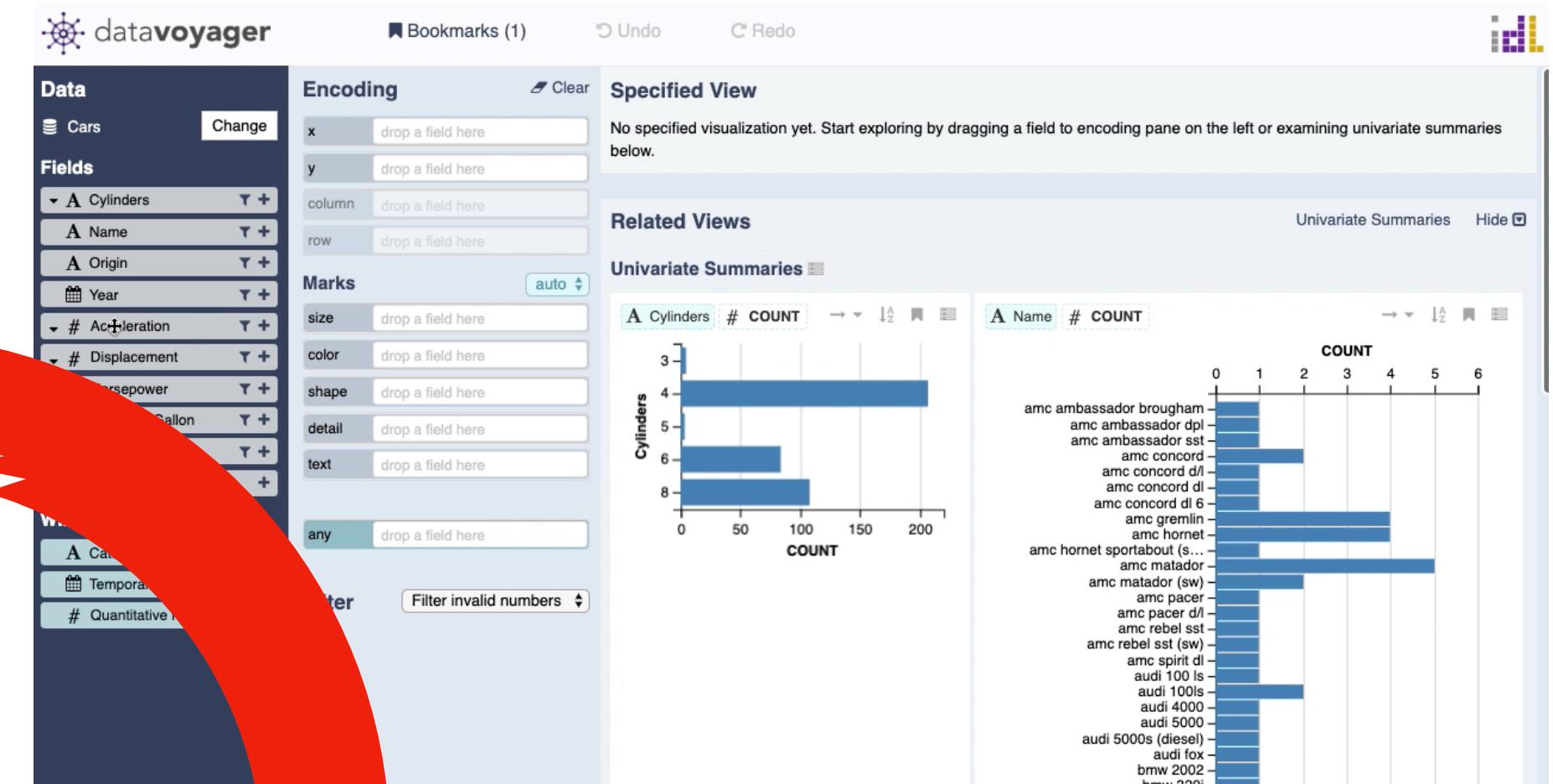
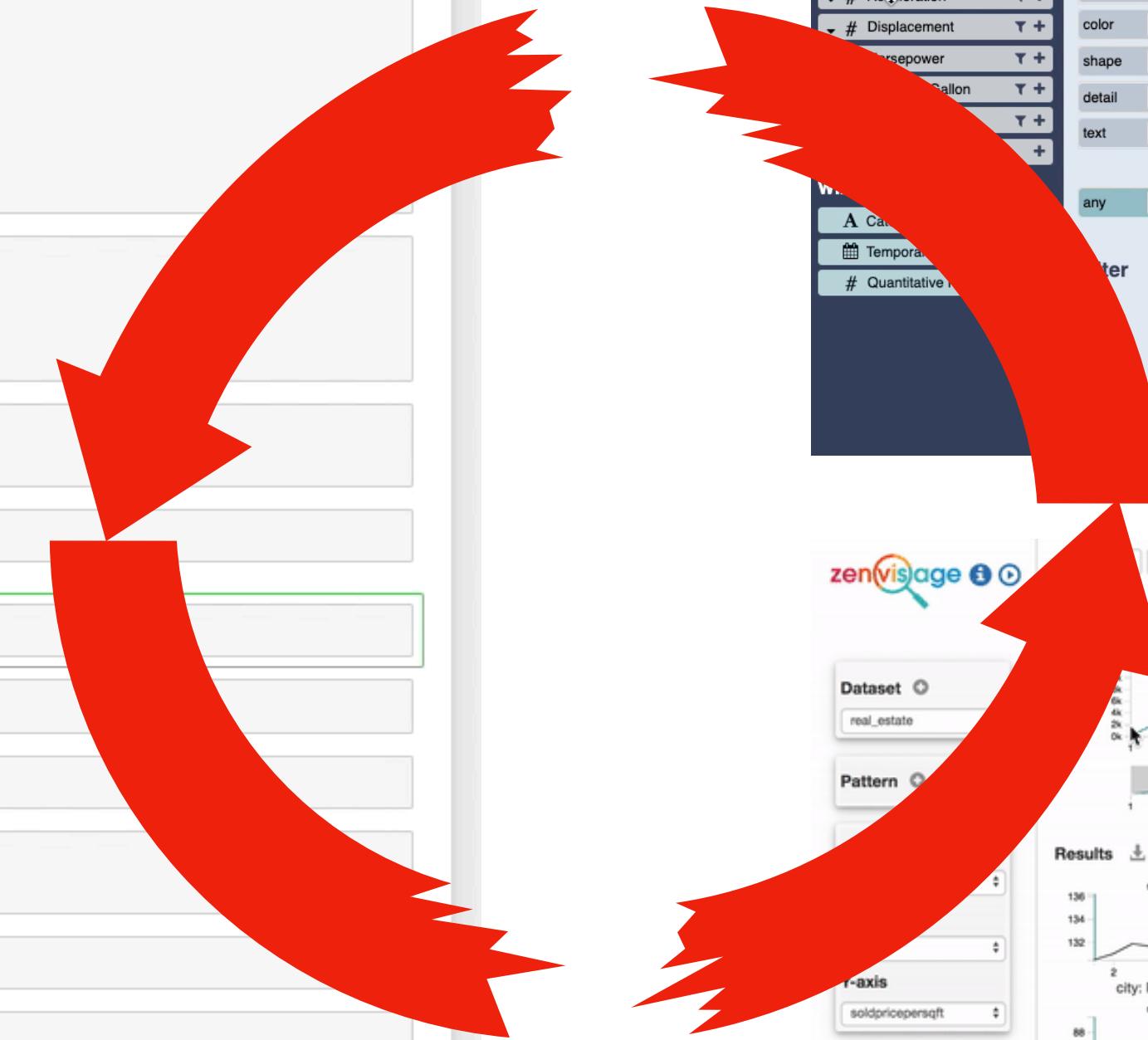


pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$


The logo for plotly consists of a stylized lowercase 'v' composed of three colored triangles (dark grey, purple, yellow) followed by the word 'plotly' in a lowercase sans-serif font.

The ggplot2 logo is a hexagonal icon containing a blue line plot with three points and the text "ggplot2". The website address "www.rstudio.com" is visible at the bottom right of the hexagon.



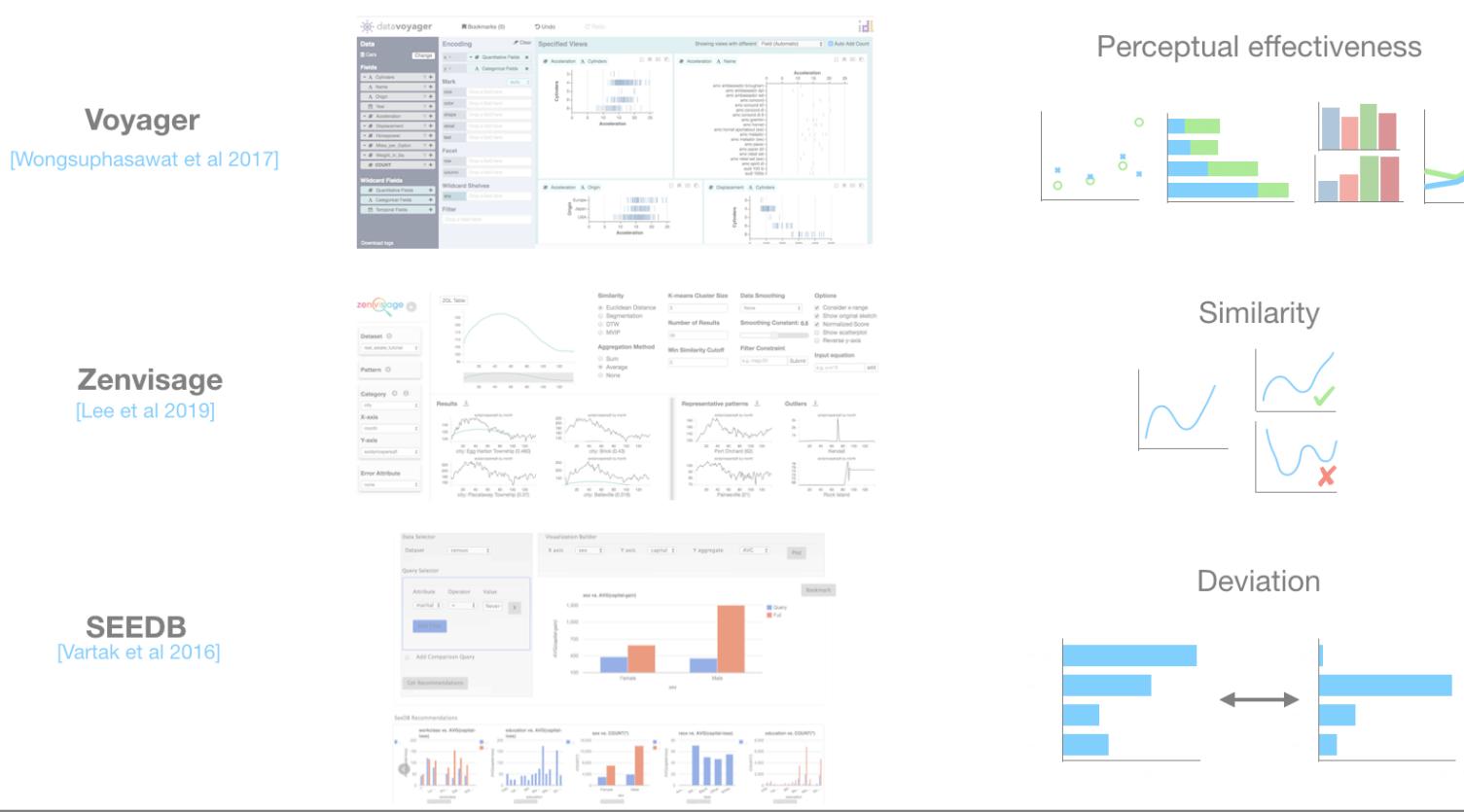
The image displays four distinct logos arranged horizontally. On the left is the logo for 'datavoyager', featuring a purple stylized gear or network icon followed by the brand name in a bold, lowercase sans-serif font. Next is the '4DIVE' logo, which consists of a black square containing a white stylized '4' icon and the word 'DIVE' in large, white, uppercase letters. In the center is the 'tableau' logo, where each letter of the word 'tableau' is rendered in a different shade of blue. To the right is the 'zenvisage' logo, which features the word 'zenvisage' in a stylized font where 'zen' is red, 'visage' is green, and the 'e' is partially obscured by a magnifying glass icon.

The image contains three logos stacked vertically. The top logo is for SAS, featuring a blue stylized 'S' followed by the word 'sas' in black. To its right is a black icon of a mobile phone with signal bars. The middle logo is for Spotfire, showing a red target symbol followed by the word 'Spotfire' in blue with a registered trademark symbol. The bottom logo is for TIBCO Software, with the word 'TIBCO' in a smaller font above 'Software'.

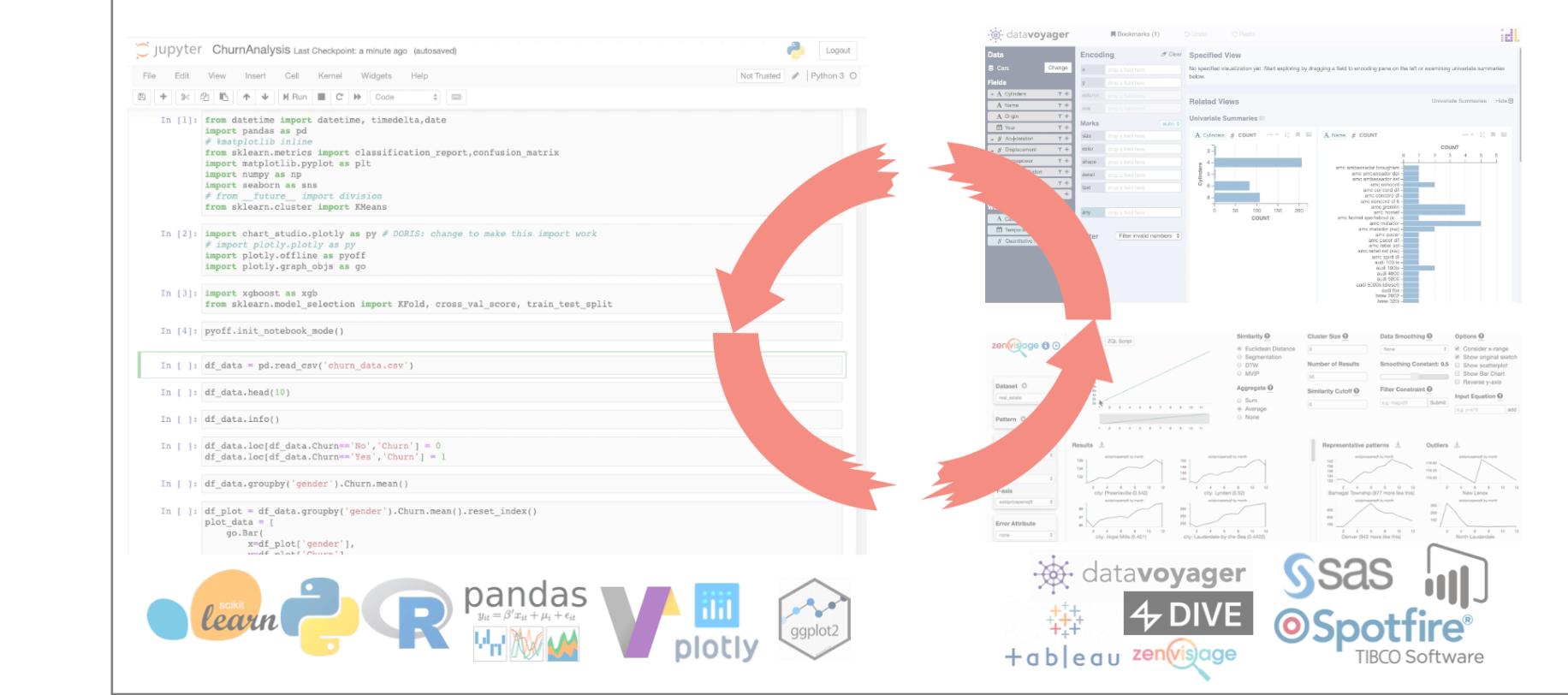


LUX

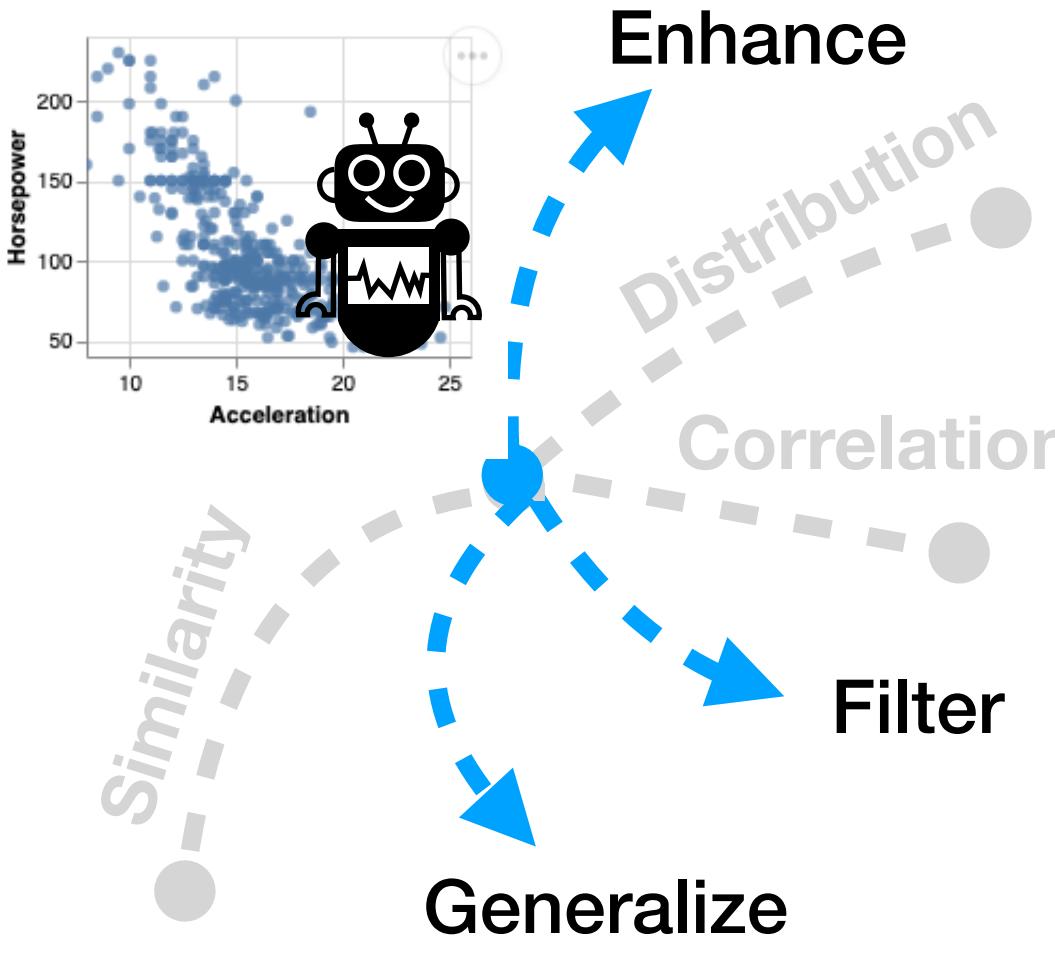
Challenge #1: Limited Task Support



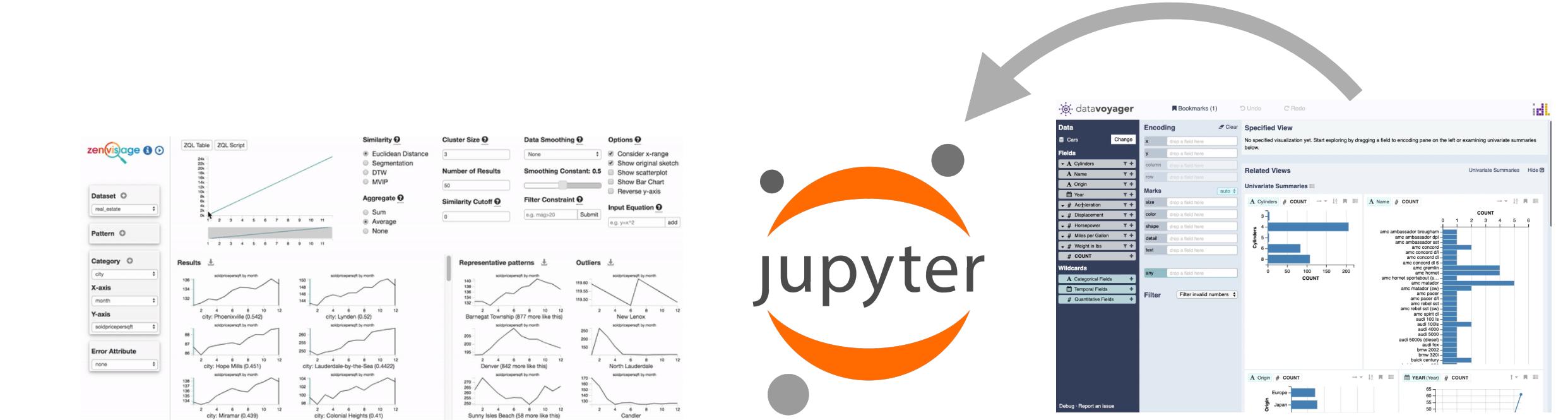
Challenge #2: Disconnected Workflow



General, Extensible Analytics Framework



In-situ Jupyter Widget



File Edit View Insert Cell Kernel Widgets Help

Not Trusted

Python 3 C



In [1]: `import lux`

Load in a dataset of 392 different cars from 1970-1982:

```
In [ ]: dataset = lux.Dataset("lux/data/car.csv", schema=[{"Year": {"dataType": "date"} }])
```

```
In [ ]: dataset.df.head()
```

```
In [ ]: dobj = lux.DataObj(dataset)

result = dobj.showMore()

result.display()
```

Intuitively, we expect cars with more horsepower means higher acceleration, but we are actually seeing the opposite of that trend.

Let's learn more about whether there are additional factors that is affecting this relationship.

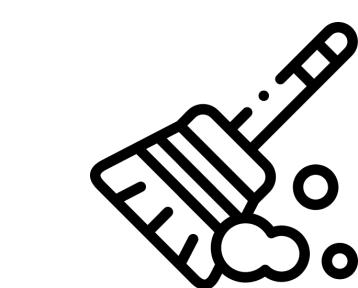
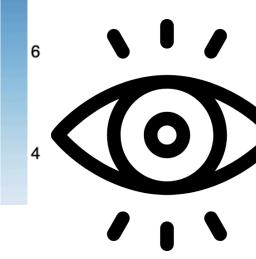
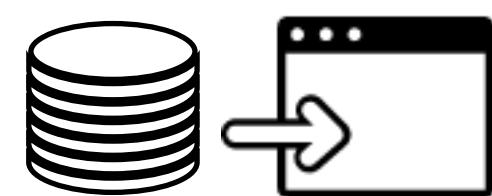
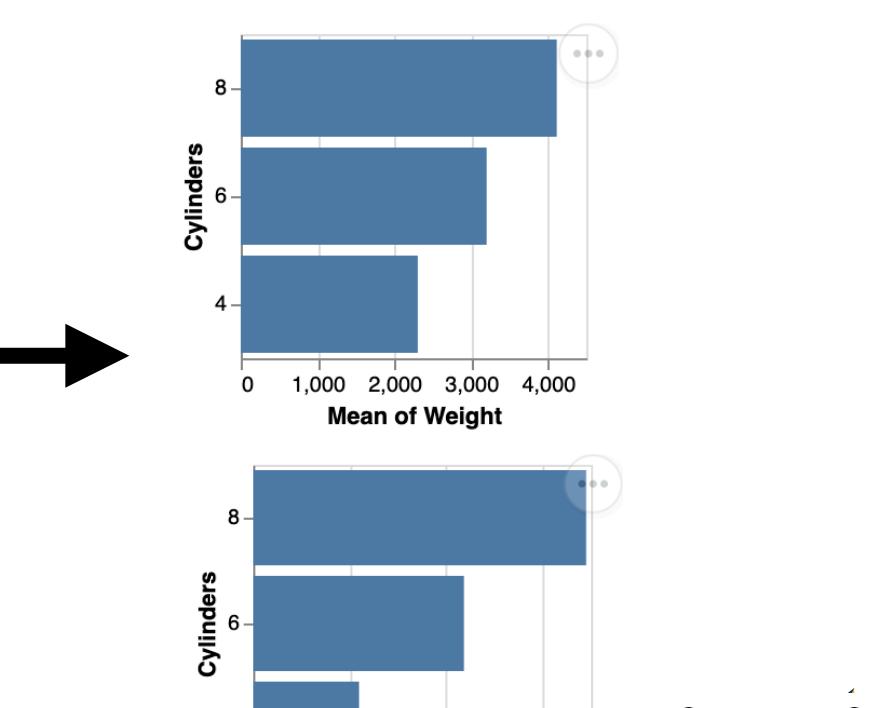
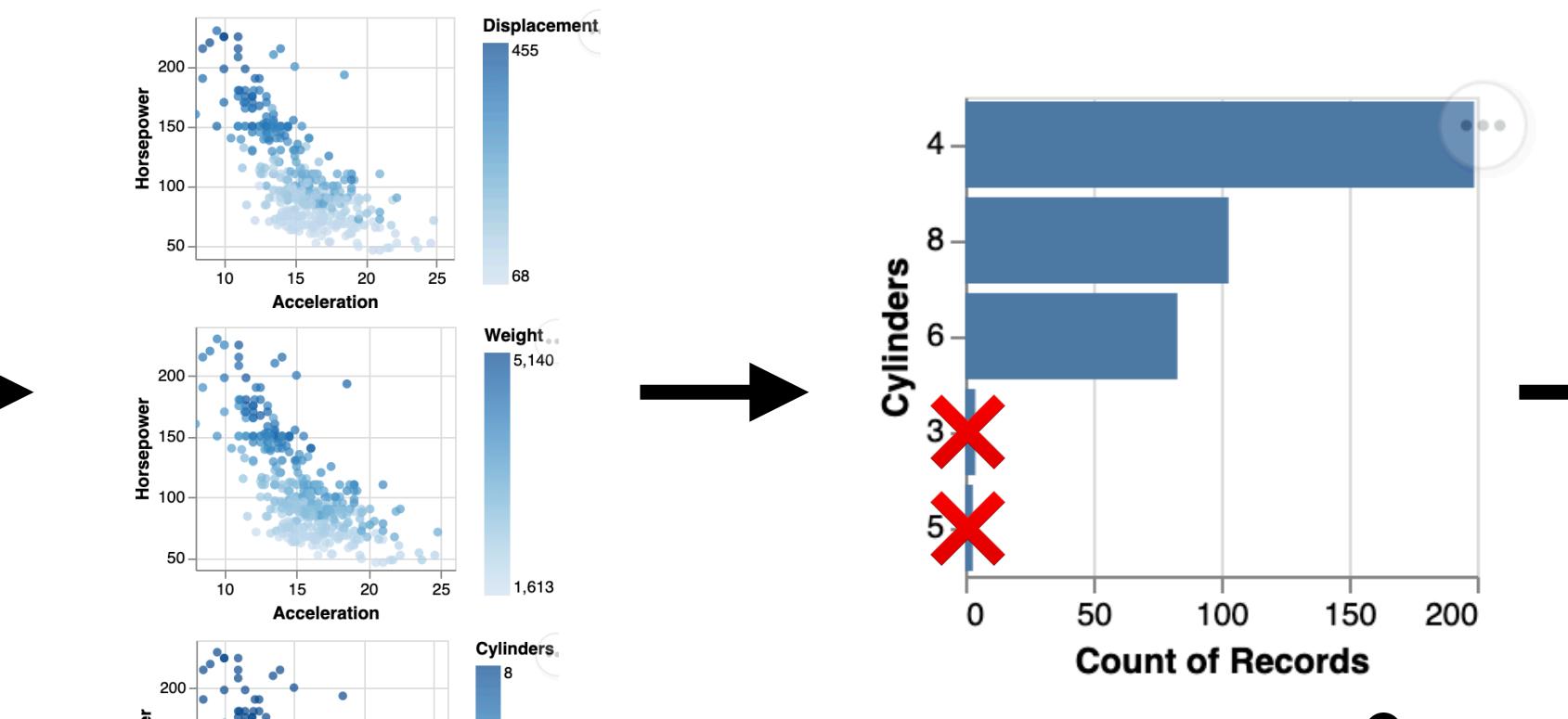
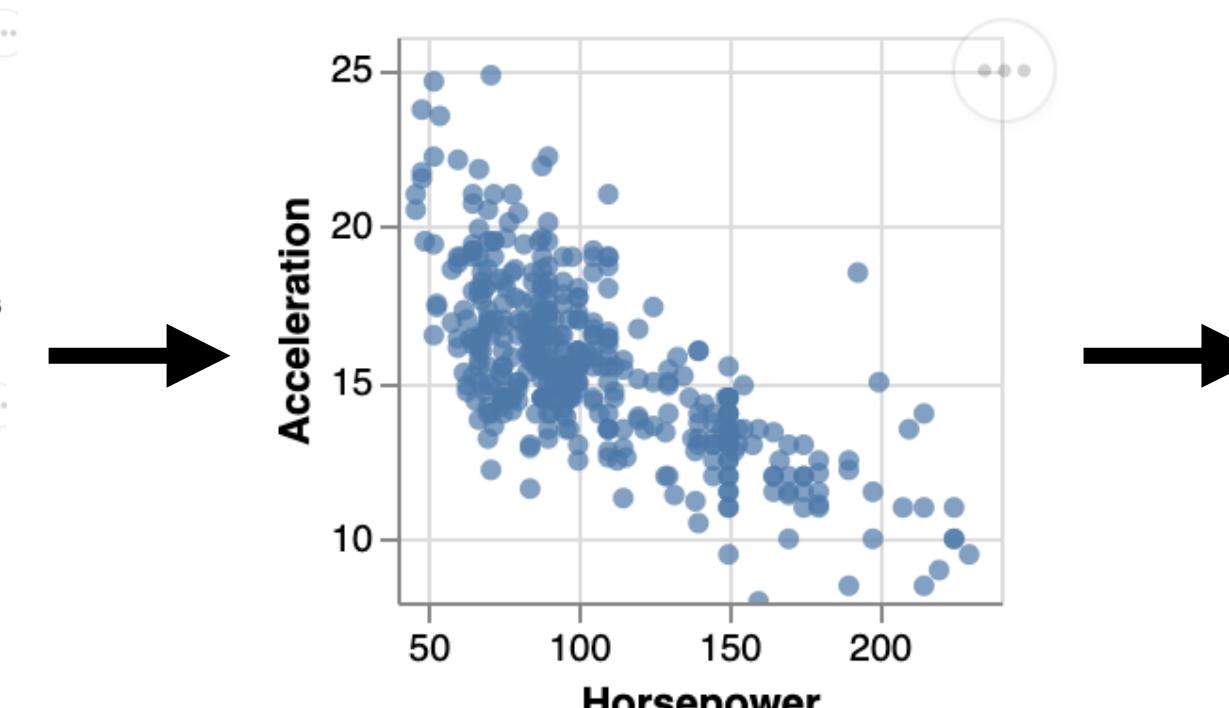
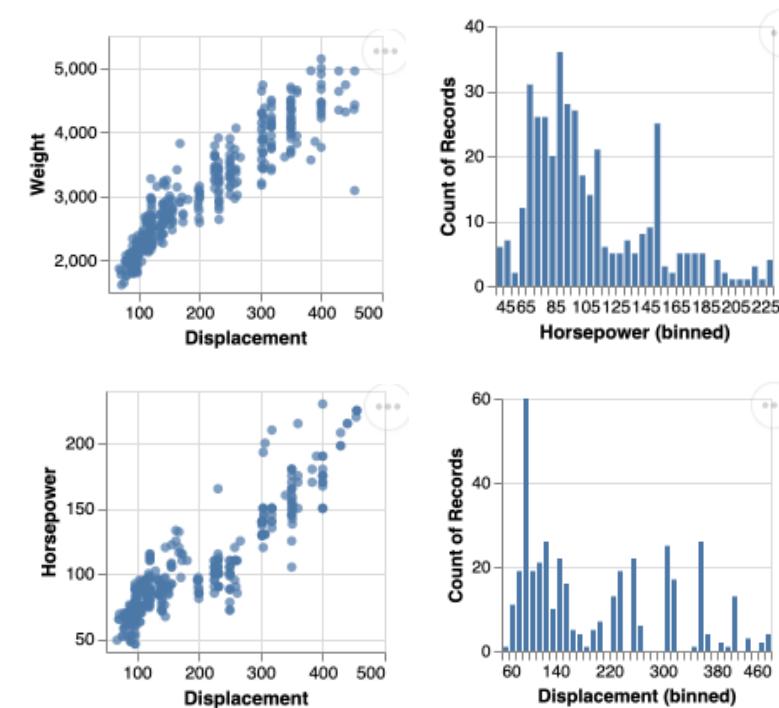
```
In [ ]: dobj = lux.DataObj(dataset,[lux.Column("Acceleration", dataModel="measure"),
                                   lux.Column("Horsepower", dataModel="measure")])
result = dobj.showMore()
result.display()
```

In Enhance, all the added variable (color), except MilesPerGal, shows a trend for the value being higher on the upper-left end, and value decreases towards the bottom-right.

Now given these three other variables, let's look at what the Displacement and Weight is like for different Cylinder cars.

Demo Summary

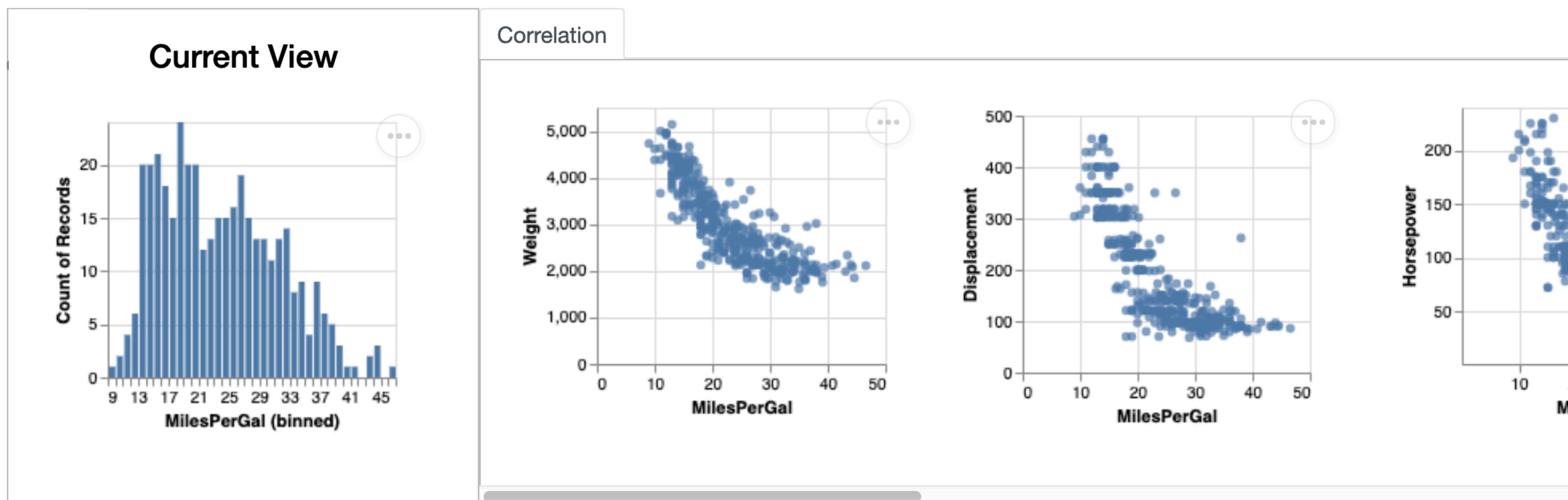
- Seamless integration: Load, clean data with pandas
- Overview: Correlation, Distribution
- ShowMore: Enhance, Filter, Generalize
- Generate vis collection on-the-fly and for rapid browsing

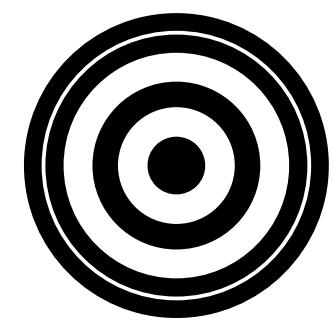


Lux language design



```
dobj = DataObj(dataset,[Column("MilesPerGal"),Column("?",dataModel="measure")])  
dobj.correlation()
```





Target: Representing Data Aspects of Interest

Data Analytics

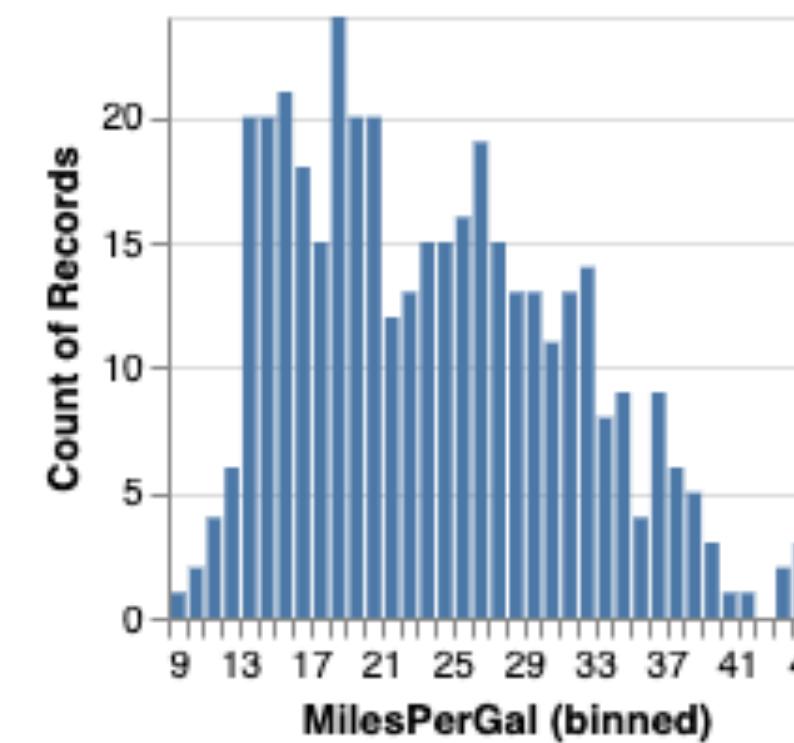
Sample - Superstore

Dimensions

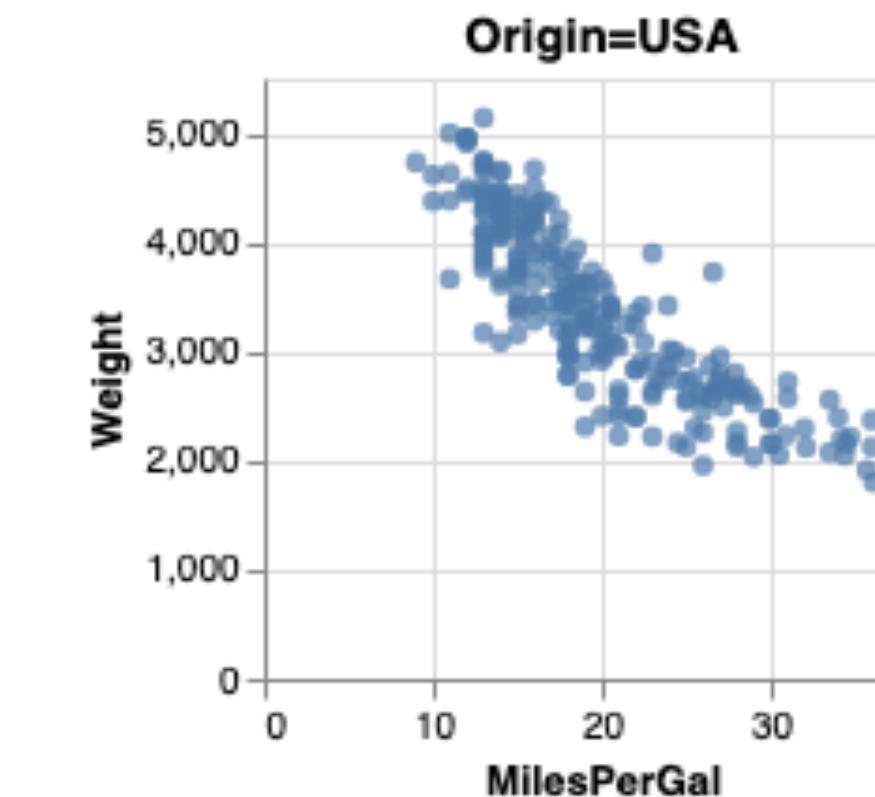
- Customer Name
- Segment
- Order
 - Order Date
 - Order ID
 - Ship Date
 - Ship Mode
- Location
 - Country
 - State
 - City
 - Postal Code
- Product
 - Category
 - Sub-Category
 - Manufacturer
 - Product Name
 - Profit (bin)
 - Region
 - Measure Names



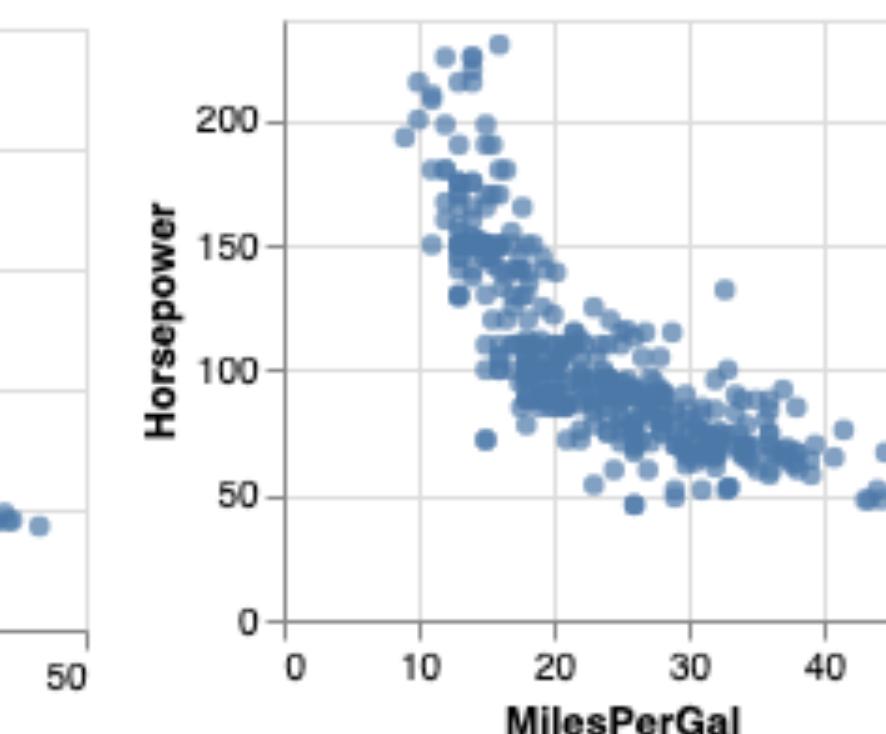
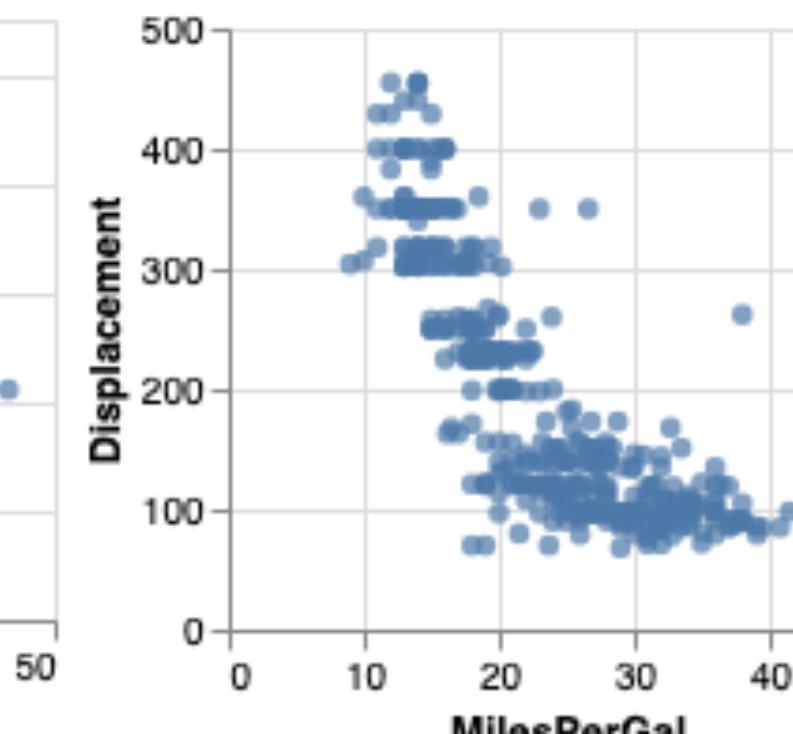
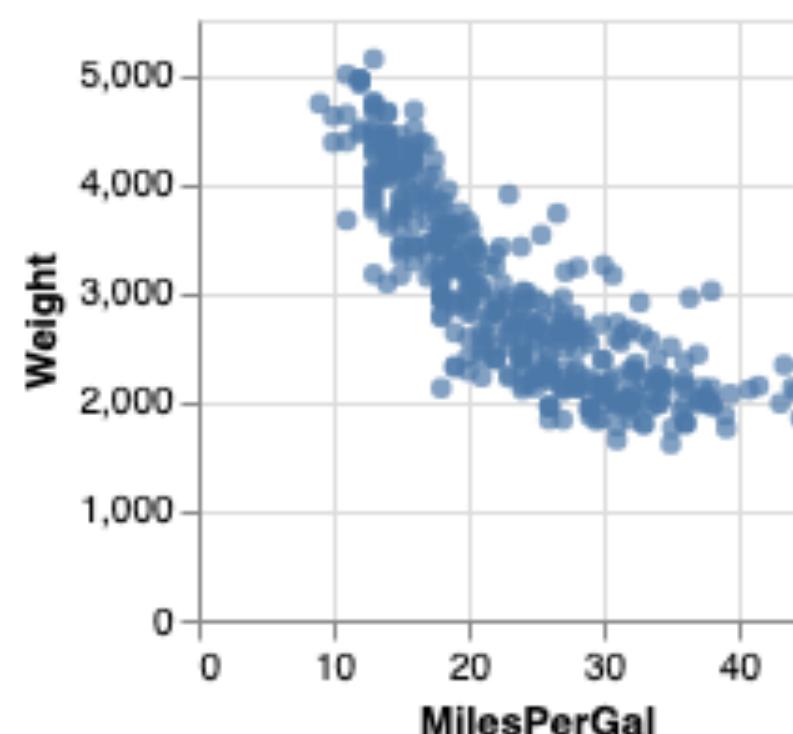
[Column("MilesPerGal")]



[Column("MilesPerGal"), Column("Weight"),
Row("Origin", "USA")]

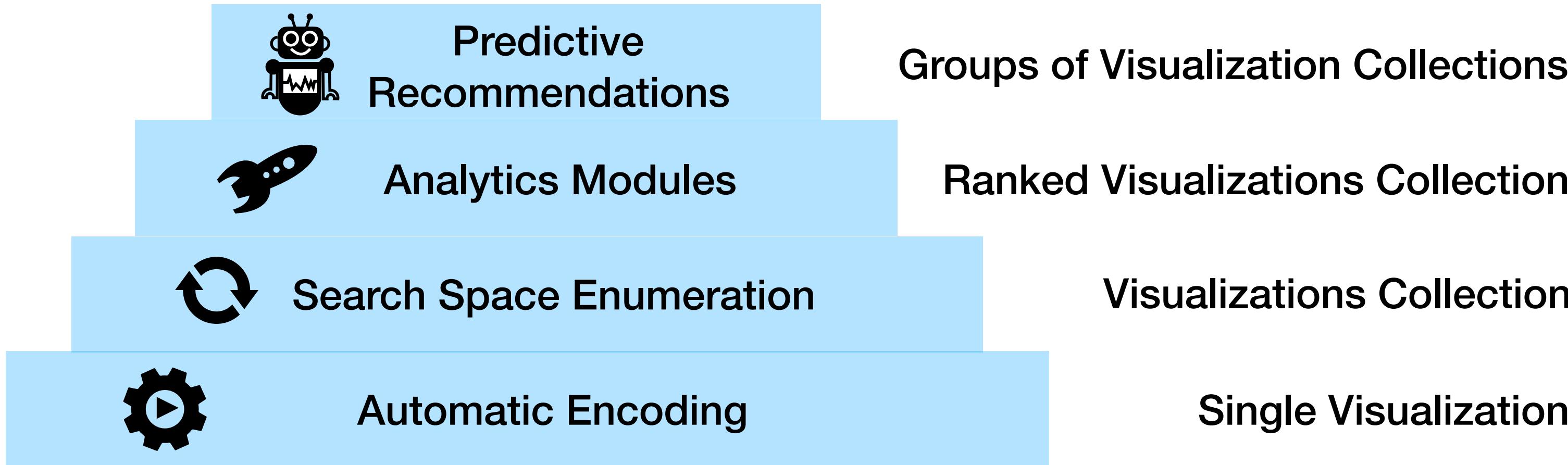


[Column("MilesPerGal"), Column("?", dataModel="measure")]



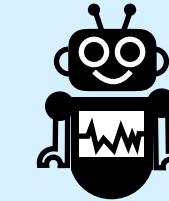


Actions: Hierarchy of Lux capabilities





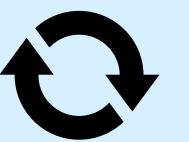
Actions: Hierarchy of Lux capabilities



Predictive
Recommendations



Analytics Modules



Search Space Enumeration



Automatic Encoding

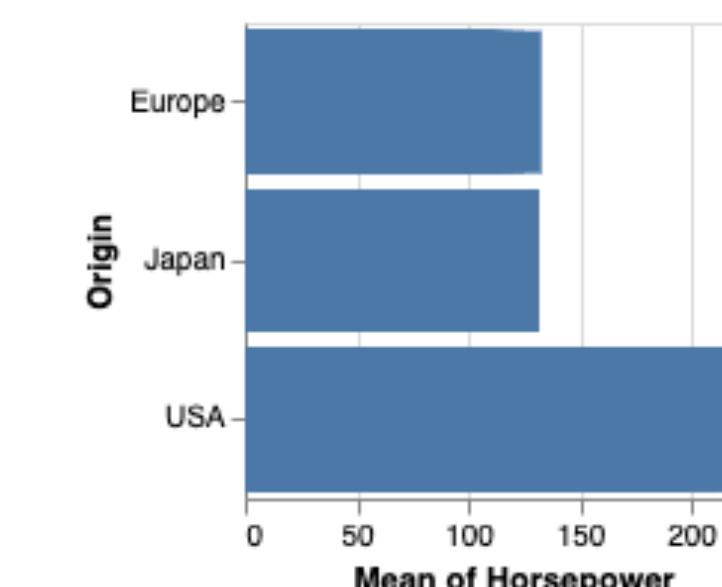
Groups of Visualization Collections

Ranked Visualizations Collection

Visualizations Collection

Single Visualization

```
import lux
dataset = lux.Dataset("data/cars.csv")
dobj = lux.DataObj(dataset,[lux.Column("Horsepower"),lux.Column("Origin")])
dobj.display()
```



```
import plotly.graph_objects as go
import pandas as pd
df = pd.read_csv("data/cars.csv")
barVal = df.groupby("Origin").mean()["Horsepower"]
fig = go.Figure(go.Bar(
    x= barVal,
    y= barVal.index,
    orientation='h'))
fig.update_layout(
    xaxis_title="Mean of Horsepower",
    yaxis_title="Origin",
)
fig.show()
```

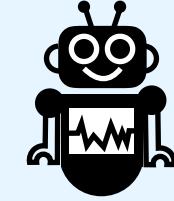
plotly

```
import matplotlib.pyplot as plt
import pandas as pd
df = pd.read_csv("data/cars.csv")
barVal = df.groupby("Origin").mean()["Horsepower"]
y_pos = range(len(barVal))
plt.barh(y_pos,barVal, align='center', alpha=0.5)
plt.yticks(y_pos, list(barVal.index))
plt.xlabel('Mean of Horsepower')
plt.ylabel('Origin')
plt.show()
```

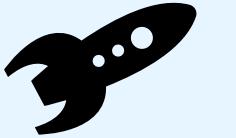
matplotlib



Actions: Hierarchy of Lux capabilities



Predictive
Recommendations



Analytics Modules



Search Space Enumeration



Automatic Encoding

Groups of Visualization Collections

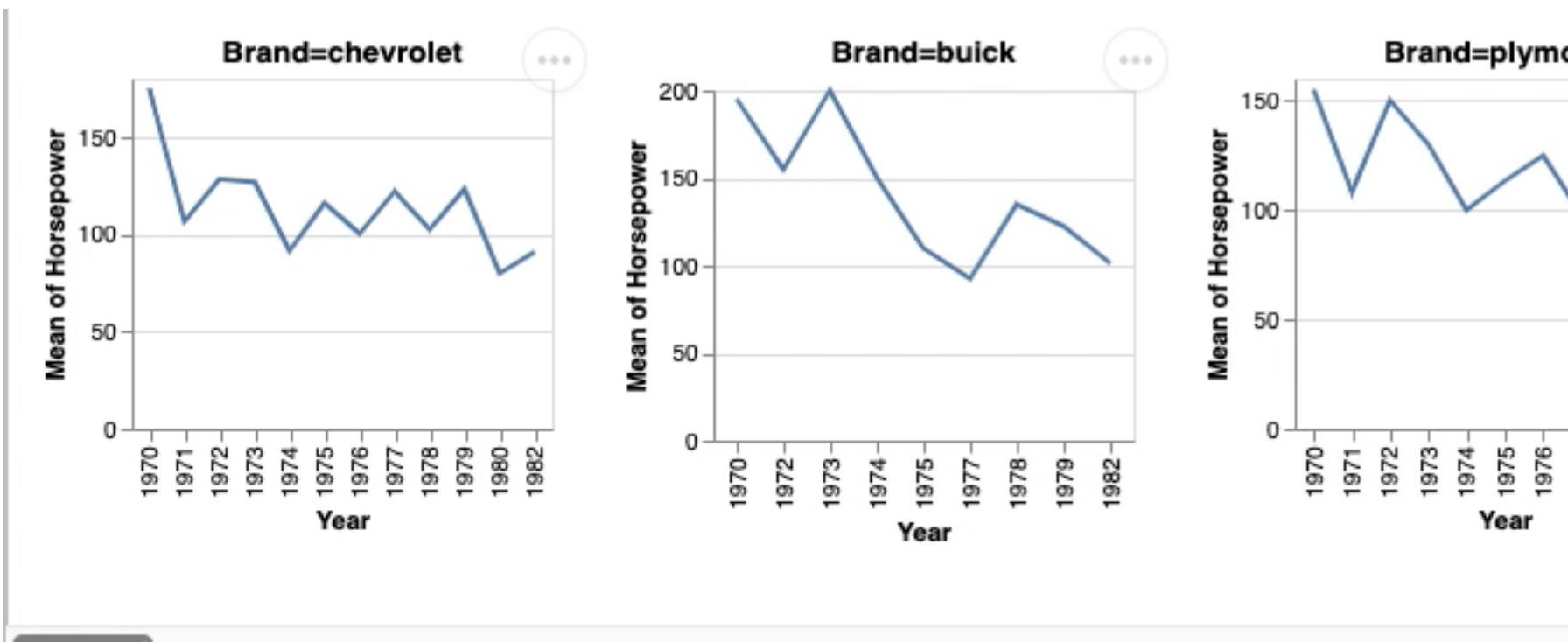
Ranked Visualizations Collection

Visualizations Collection

Single Visualization

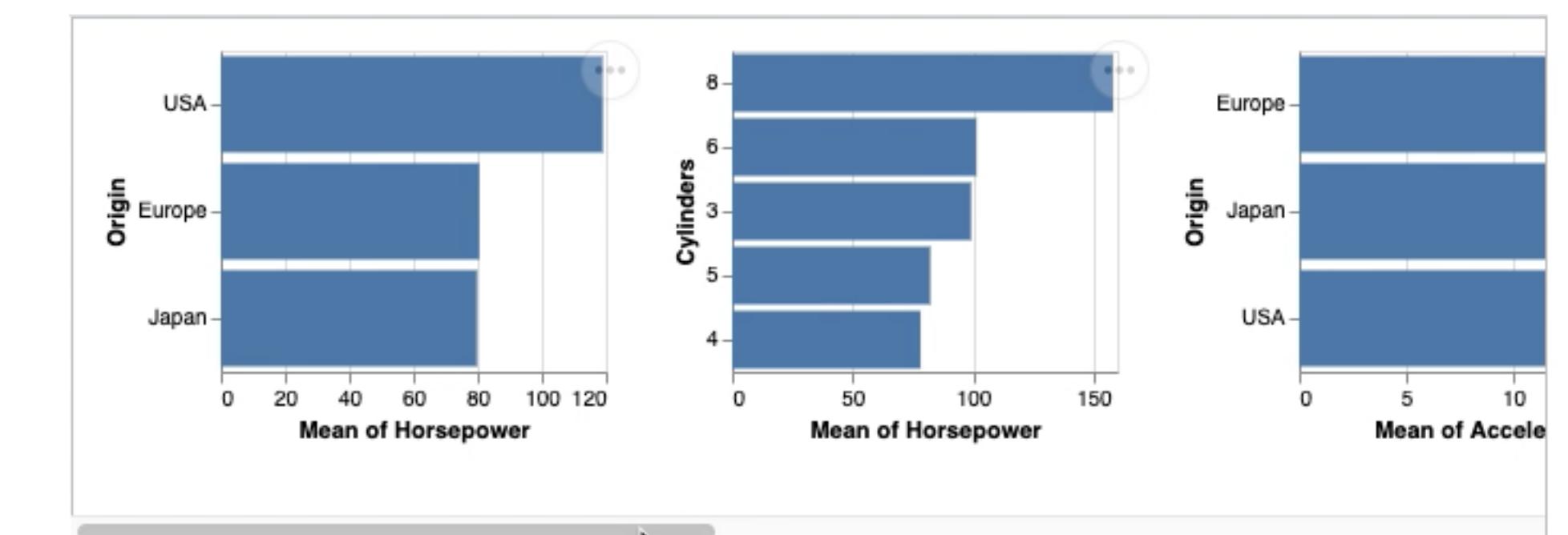
Iterate over filter values

```
dobj = DataObj(dataset,[Column("Horsepower",channel="y"),  
Column("Year",channel="x"),  
Row("Brand","?")])
```



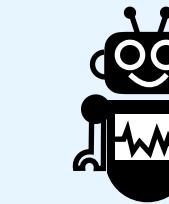
Iterate over attributes

```
dobj = DataObj(dataset,[  
Column(["Horsepower","Acceleration","Weight"]),  
Column(["Origin","Cylinders"],channel="x")])
```





Actions: Hierarchy of Lux capabilities



Predictive
Recommendations



Analytics Modules



Search Space Enumeration



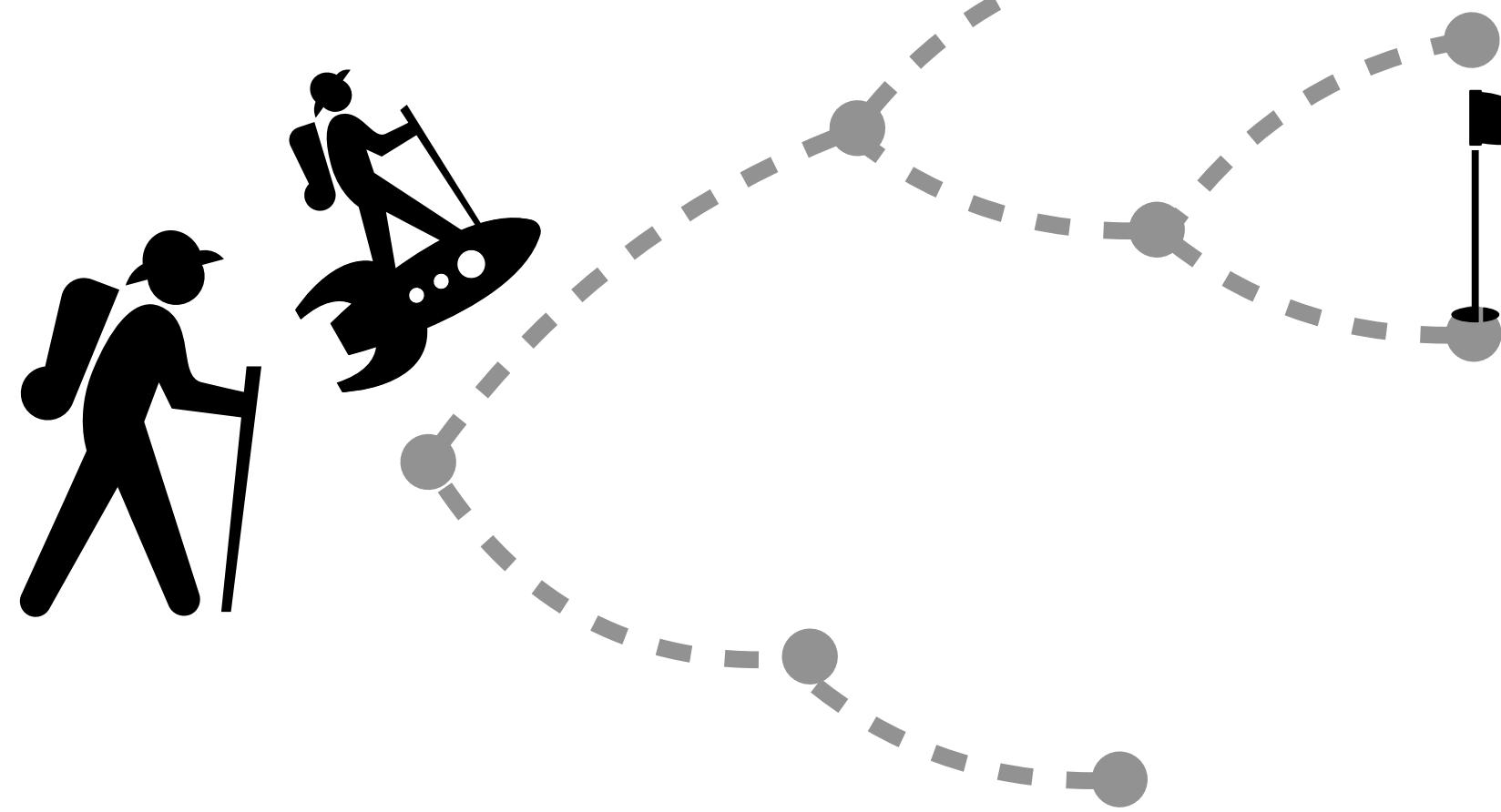
Automatic Encoding

Groups of Visualization Collections

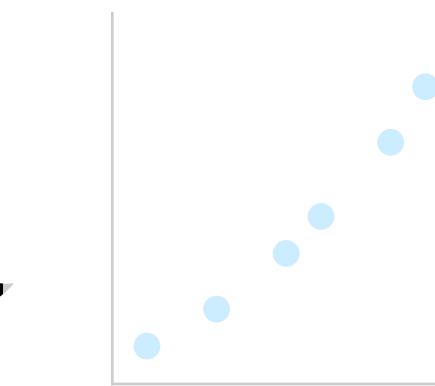
Ranked Visualizations Collection

Visualizations Collection

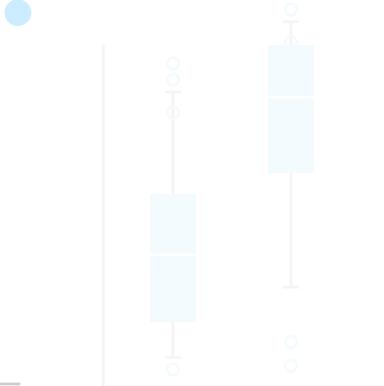
Single Visualization



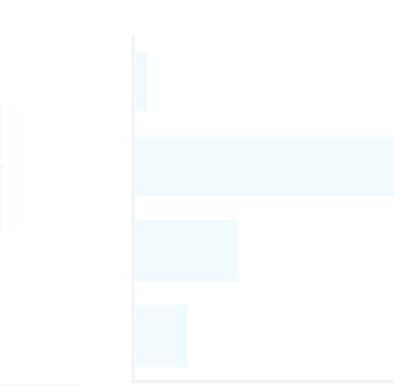
Correlation



Outlying



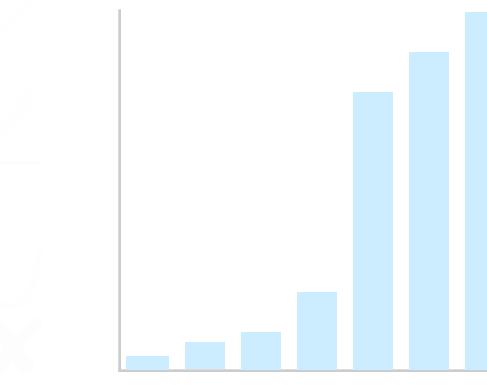
Deviation



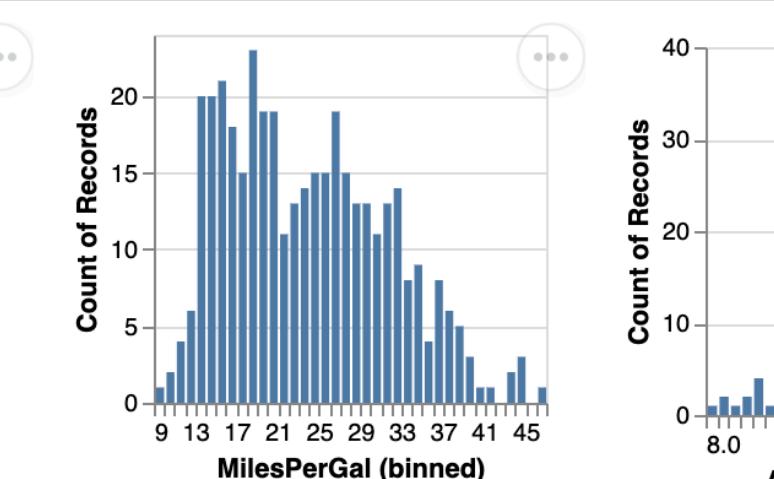
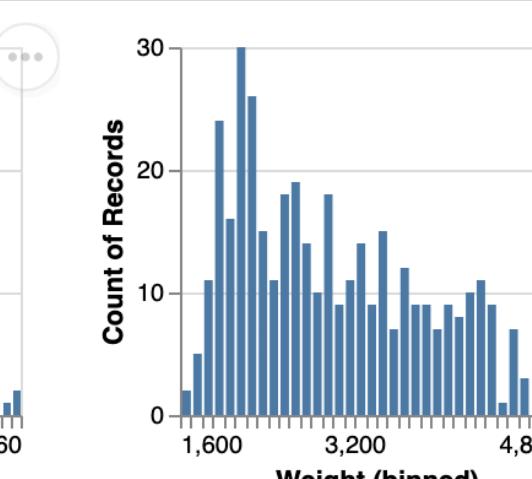
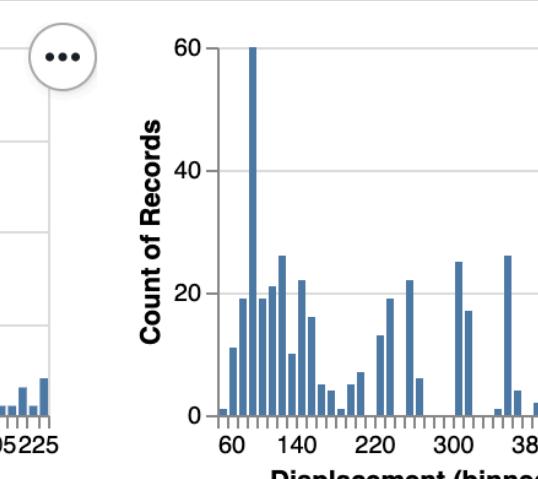
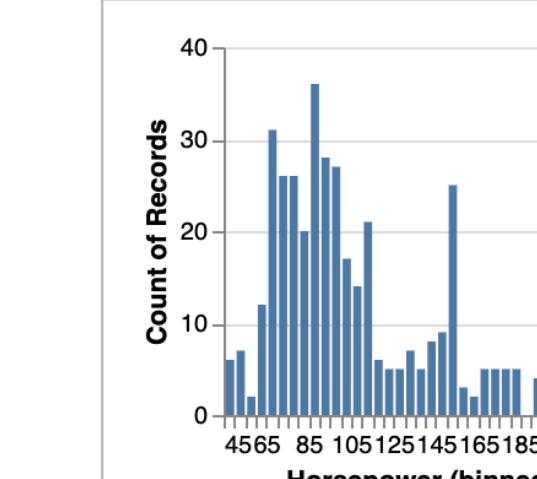
Similarity

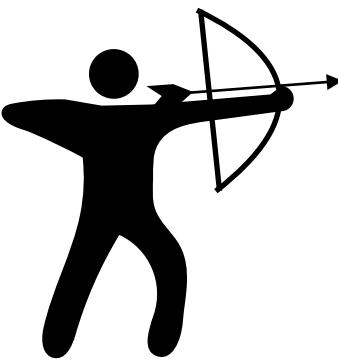


Skewness

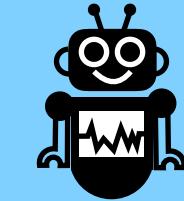


Distribution

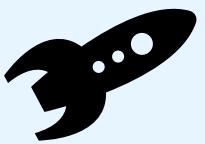




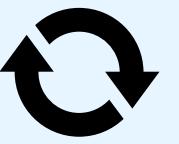
Actions: Hierarchy of Lux capabilities



Predictive
Recommendations



Analytics Modules



Search Space Enumeration



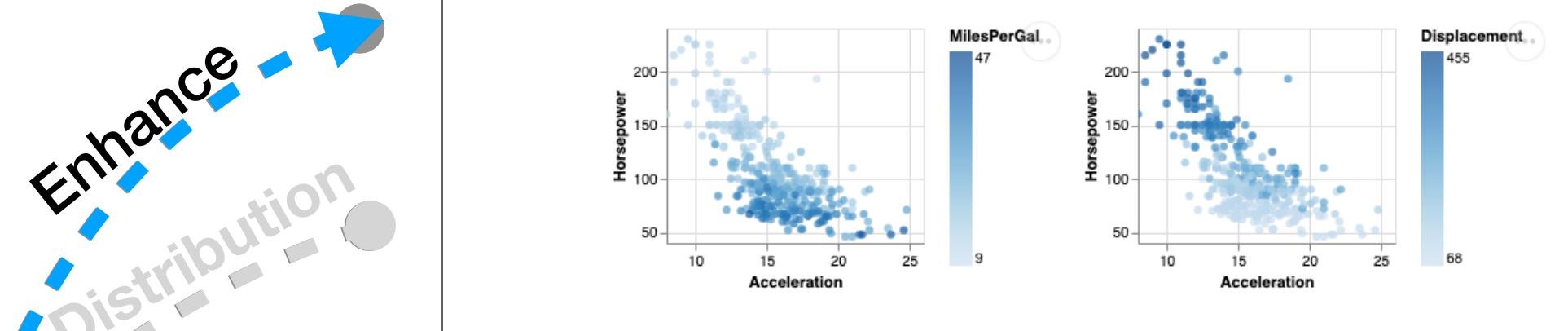
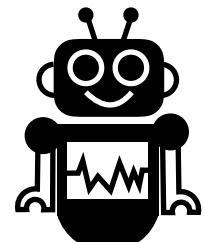
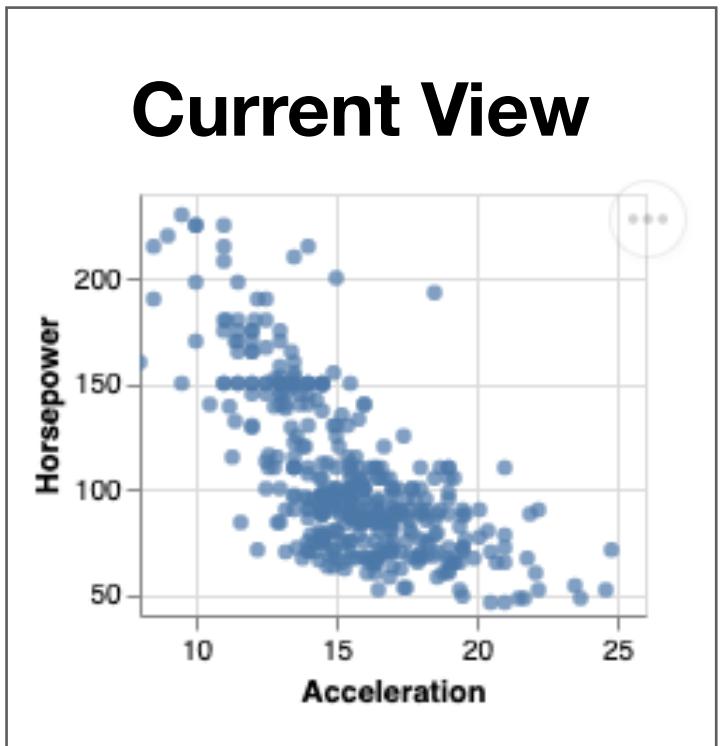
Automatic Encoding

Groups of Visualization Collections

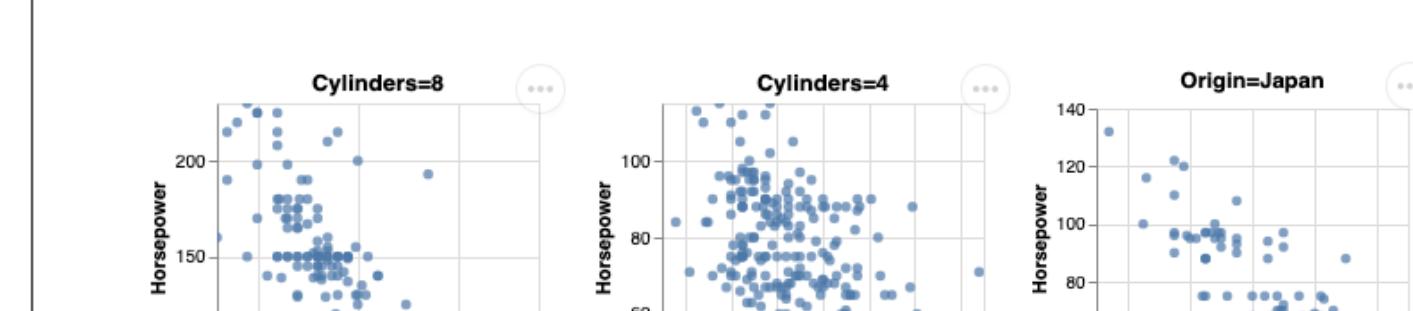
Ranked Visualizations Collection

Visualizations Collection

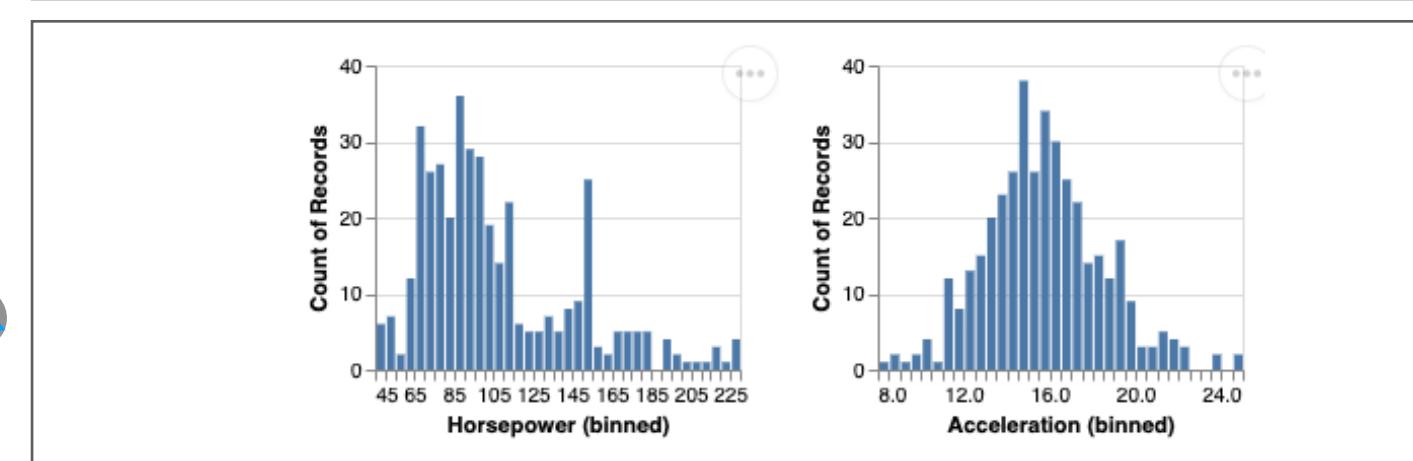
Single Visualization



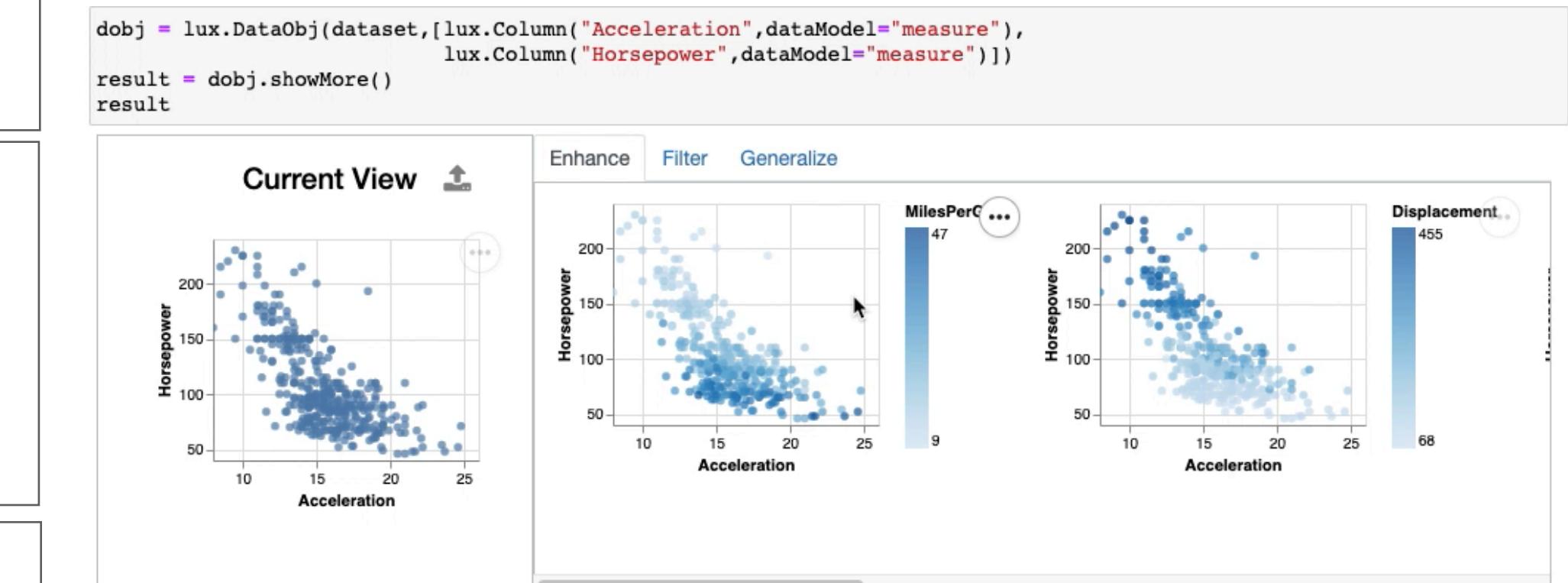
Distribution



Filter



Generalize



Lux: Towards Intelligent Visual Data Discovery



github.com/lux-org/lux

- ▲ Expressive partial specification of data aspects
- ▲ In-situ browsing of visualization collection
- ▲ Analytics modules accelerating discovery
- ▲ Predictive “next-steps” in user’s exploration



Contact:



dorislee@berkeley.edu

UC Berkeley



Tableau Research



University of Illinois

