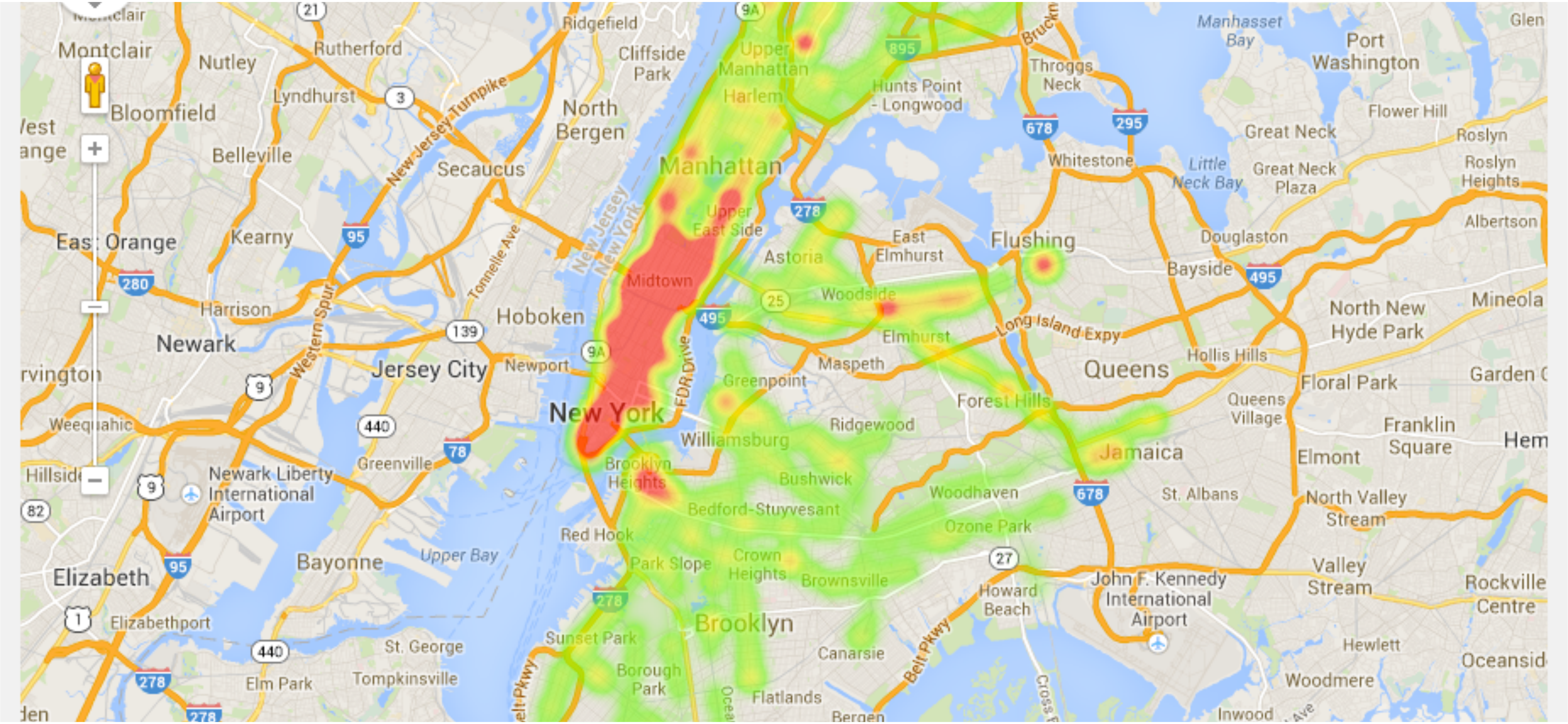
Programming Principles in Python (CSCI 503/490)

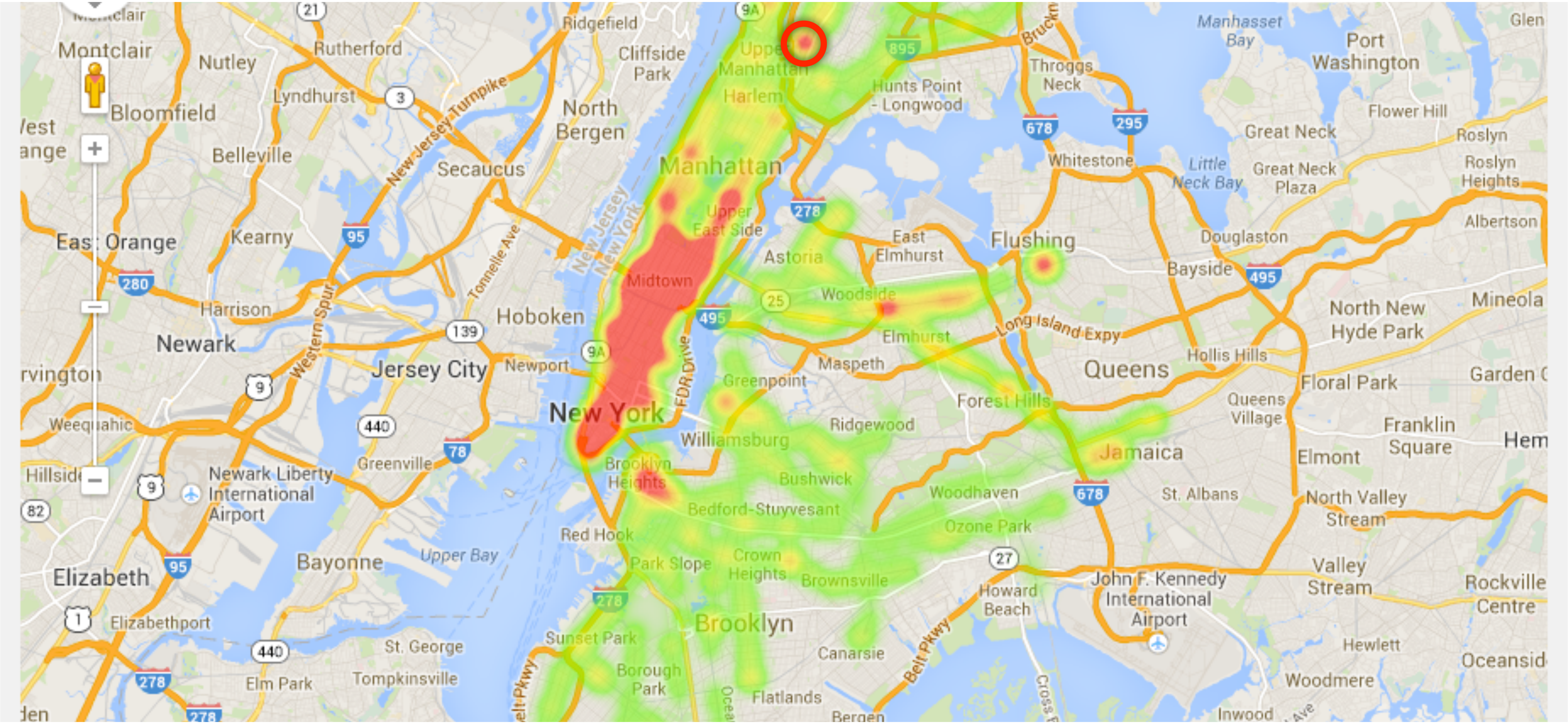
Data Visualization & Machine Learning

Dr. David Koop

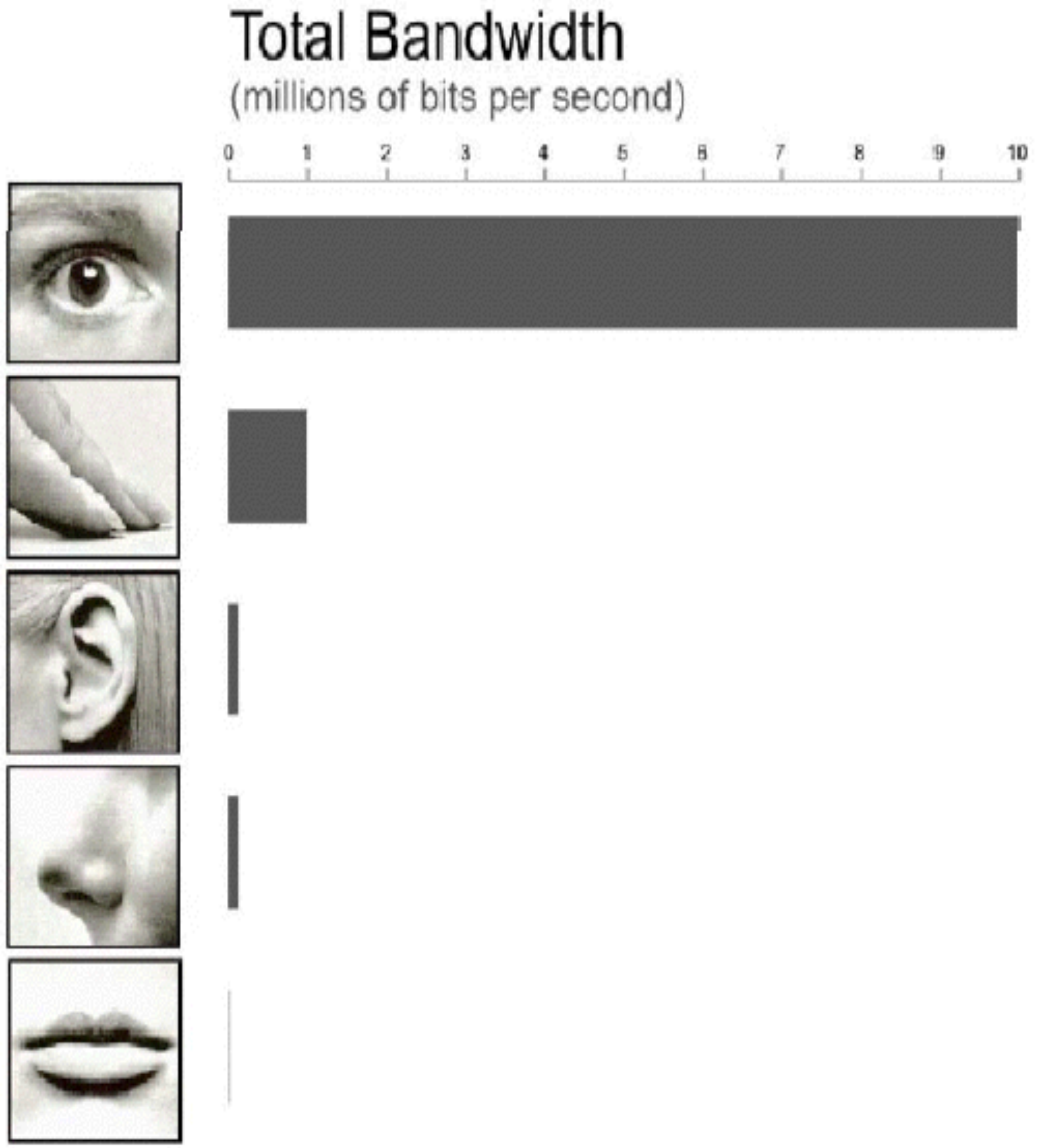
Exploring Data through Visualization



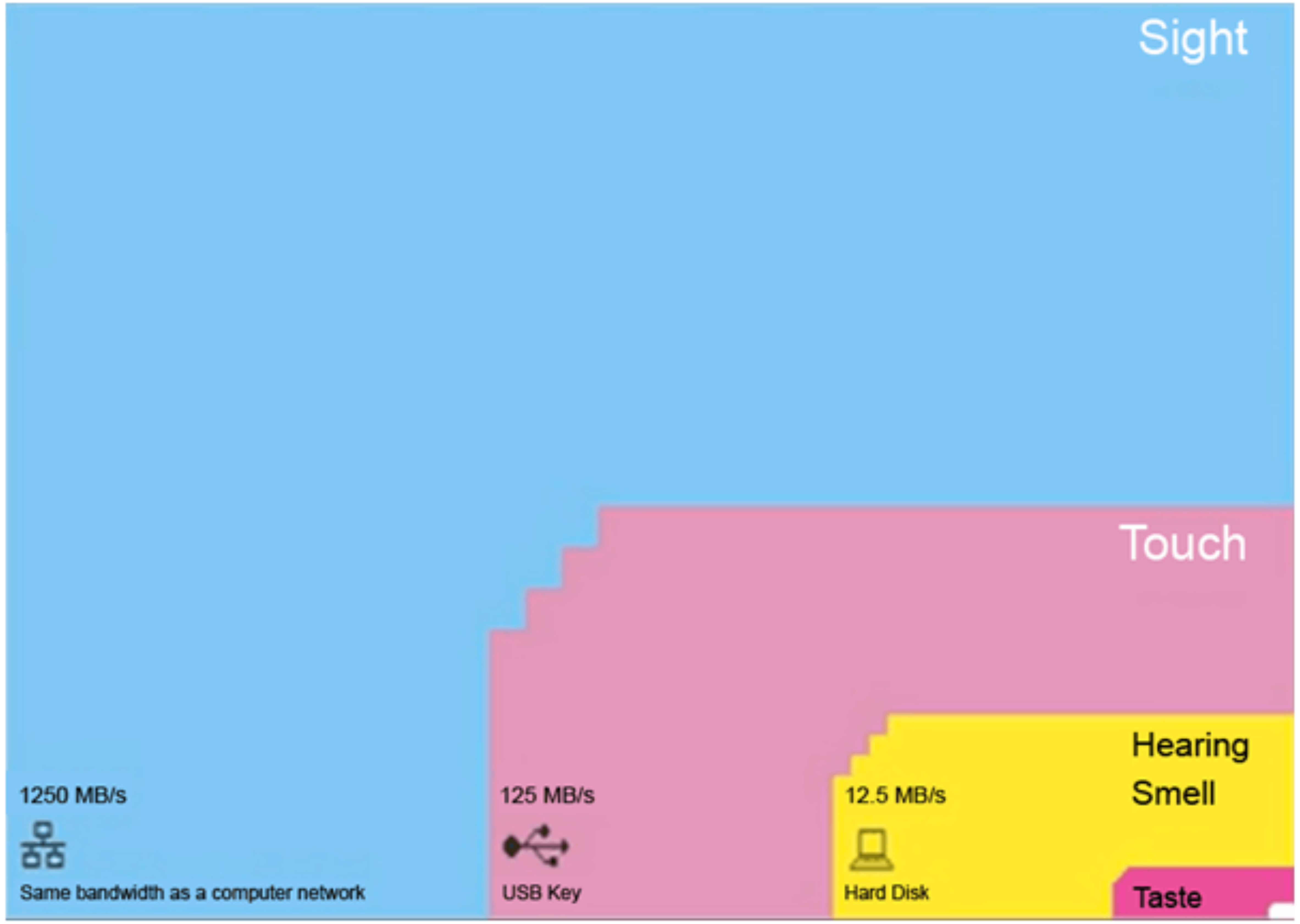
Exploring Data through Visualization



Why do we visualize data?

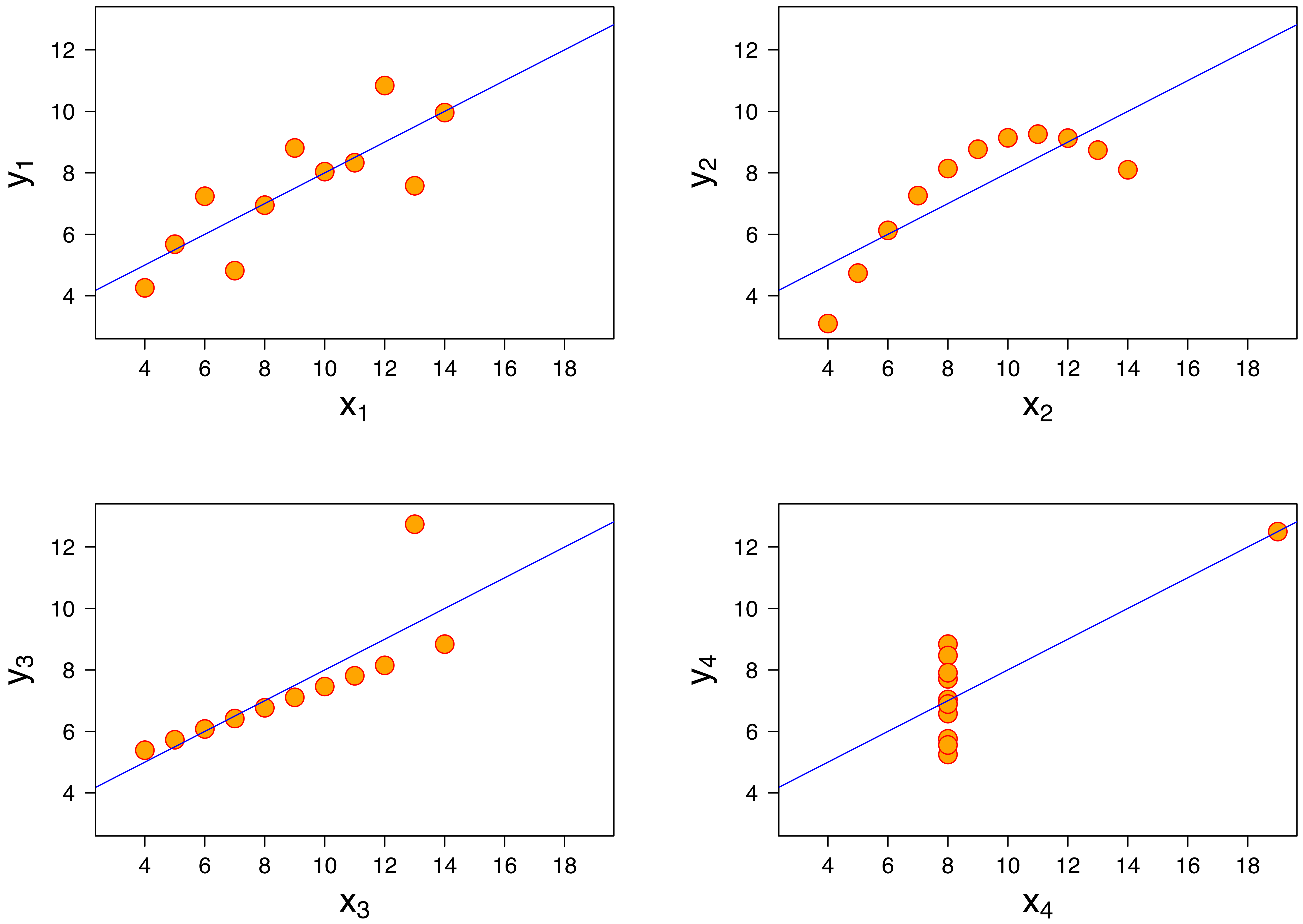


[via A. Lex]



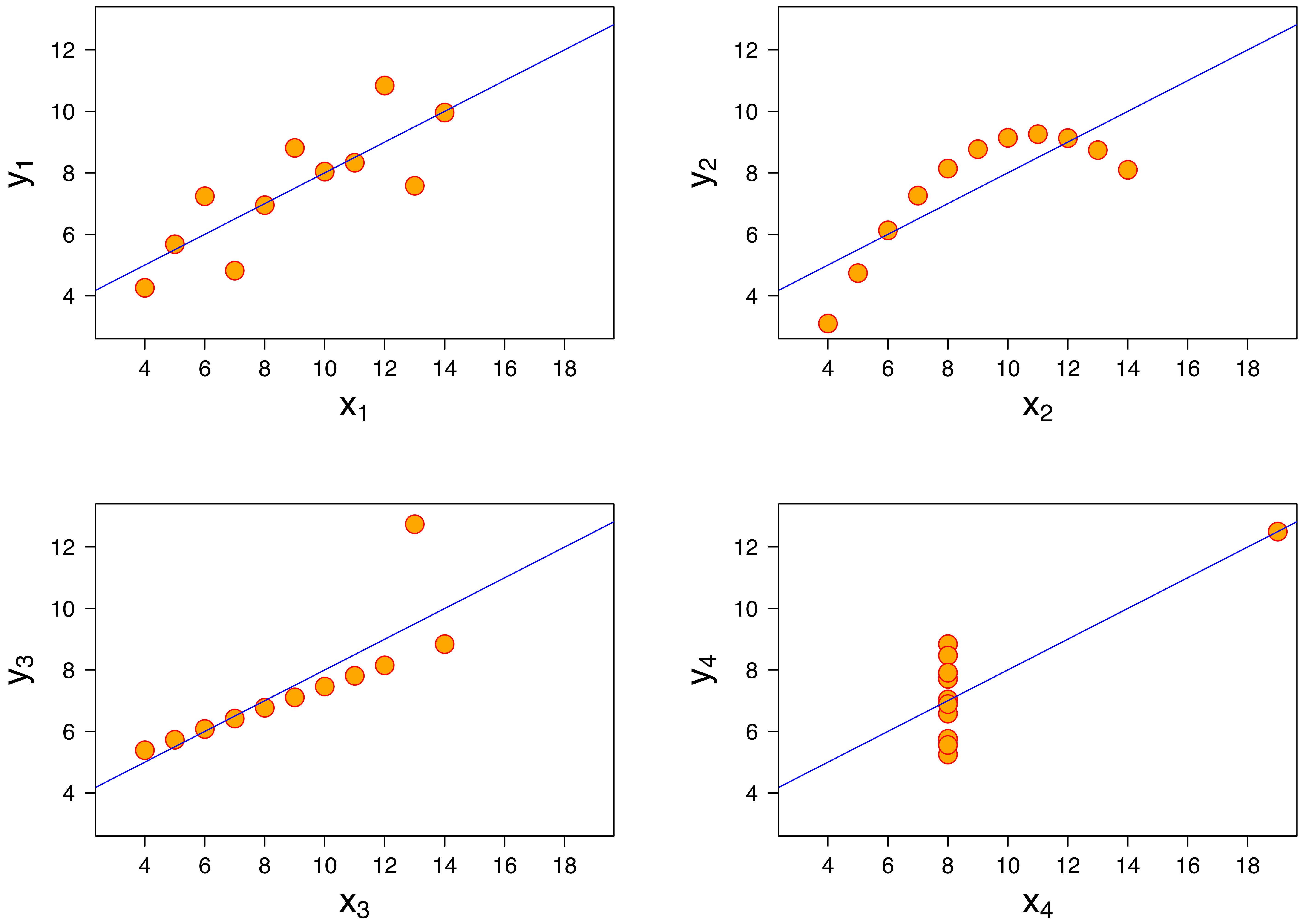
[T. Nørretranders]

Why Visual?



[F. J. Anscombe]

Why Visual?



Mean of x	9
Variance of x	11
Mean of y	7.50
Variance of y	4.122
Correlation	0.816

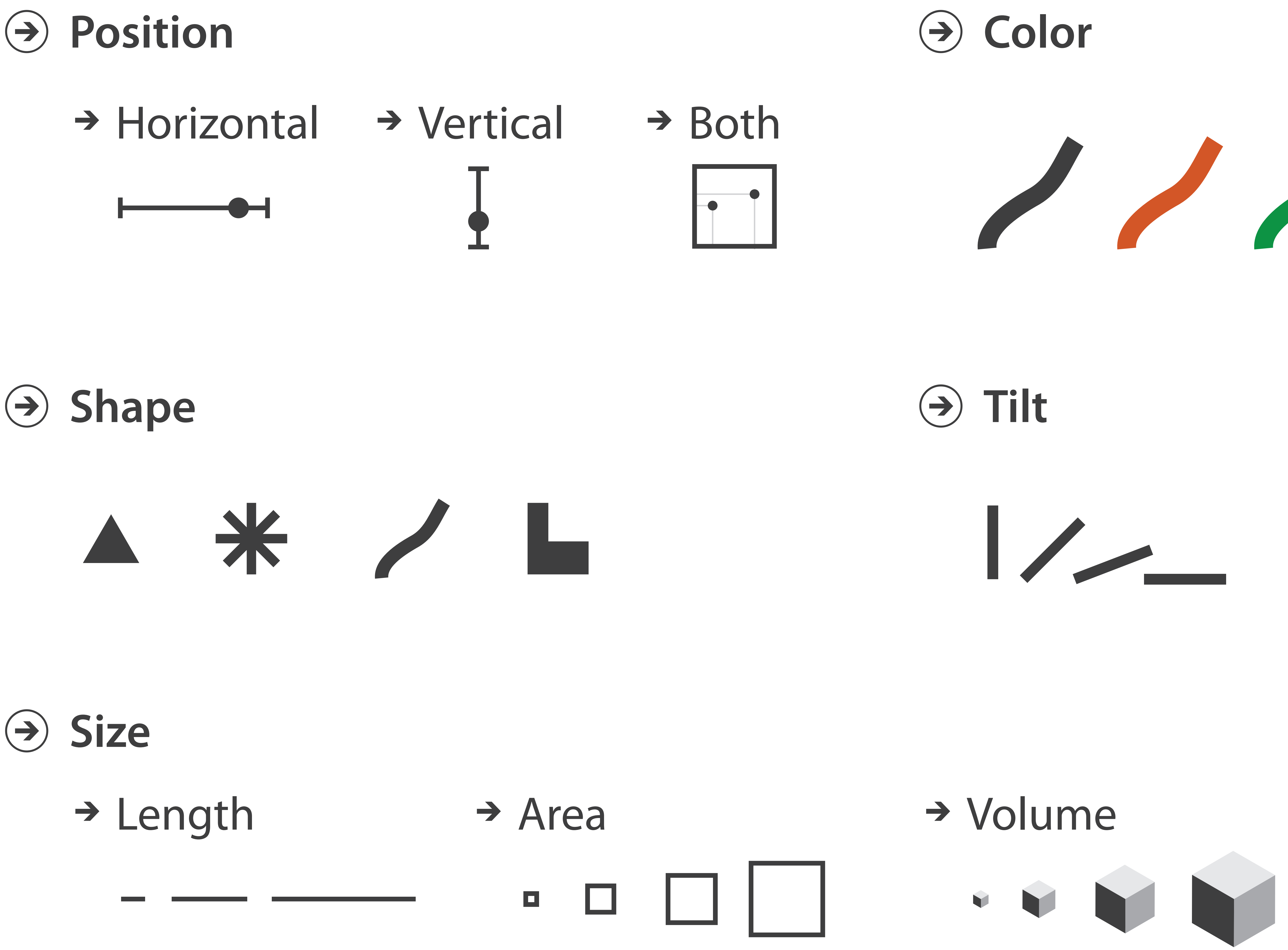
[F. J. Anscombe]

Visualization Goals

- "The purpose of visualization is **insight**, not pictures" – B. Schneiderman

- Identify patterns, trends
- Spot outliers
- Find similarities, correlation

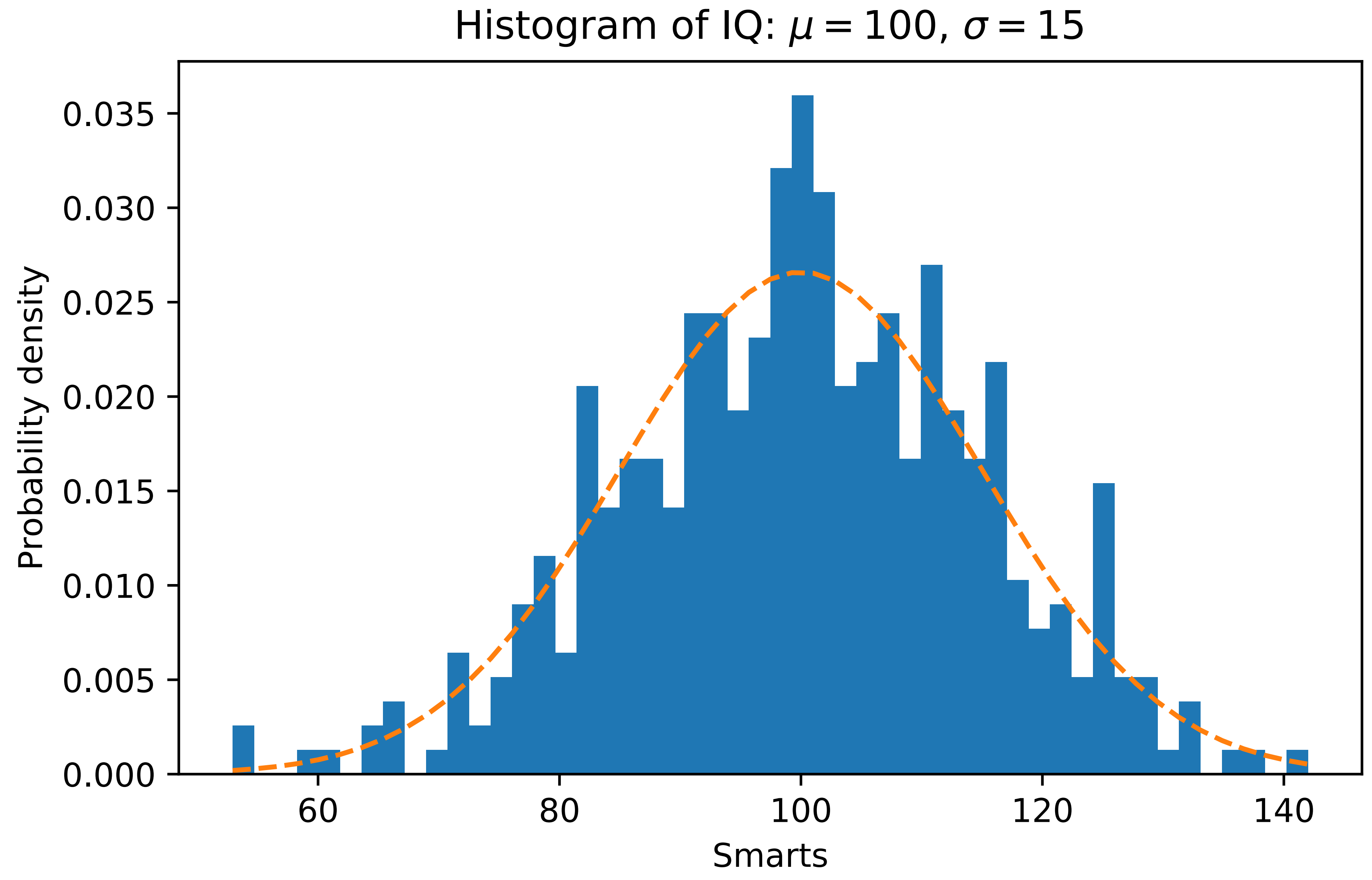
Data is Encoded via Visual Channels



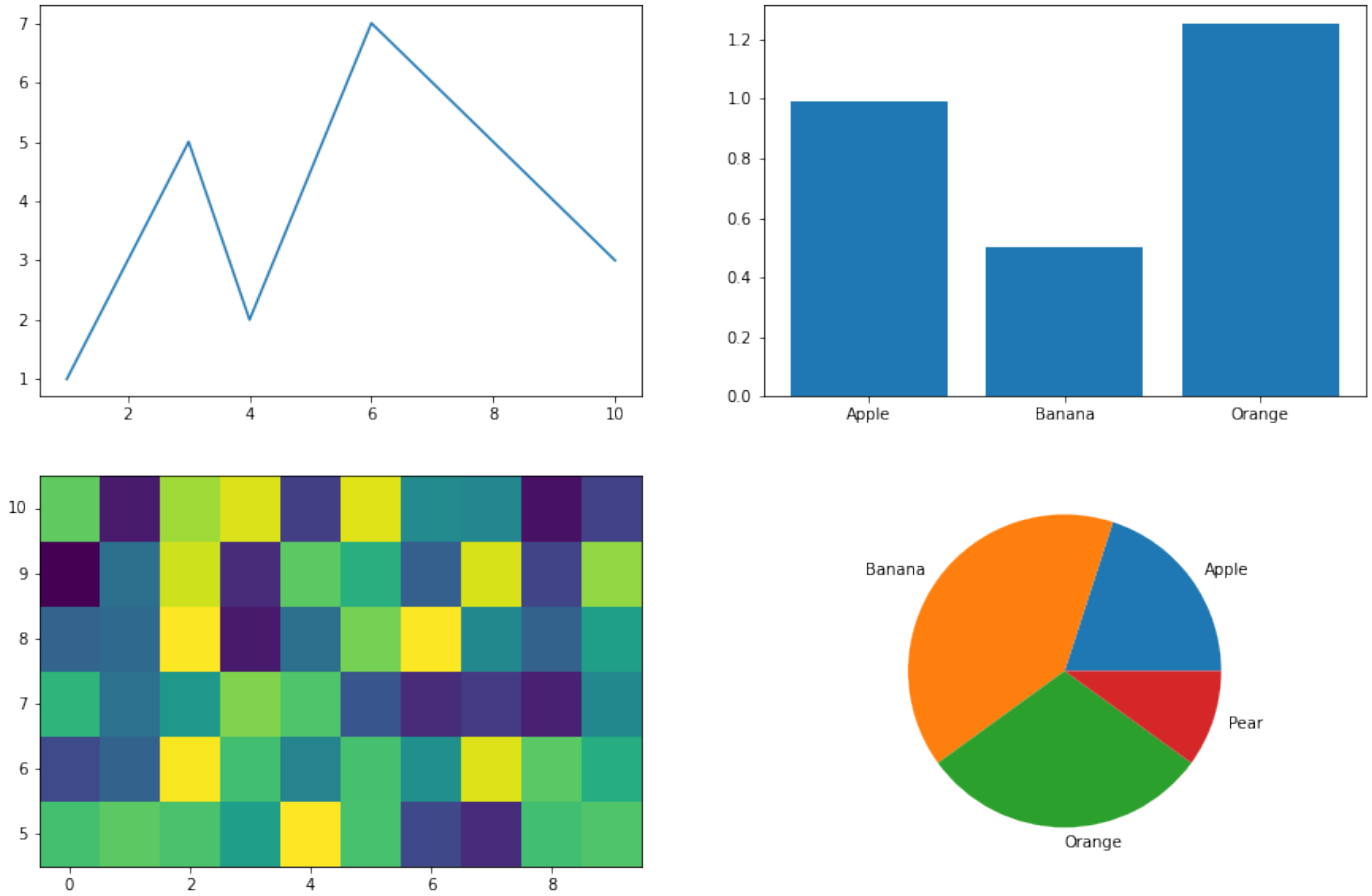
[Munzner (ill. Maguire), 2014]

matplotlib

- Strengths:
  - Designed like Matlab
  - Many rendering backends
  - Can reproduce almost any plot
  - Proven, well-tested
- Weaknesses:
  - API is imperative
  - Not originally designed for the web
  - Dated styles



Many different types of charts



Many different types of charts

- Bar chart
  - `plt.bar(['Apple','Banana','Orange'],[0.99,0.50,1.25])`
- Grid Heatmap
  - `plt.pcolormesh(x, y, Z)`
- Pie chart:
  - ```
    plt.pie([20,40,30,10],
            labels=['Apple','Banana','Orange','Pear'])
    ```
    ```

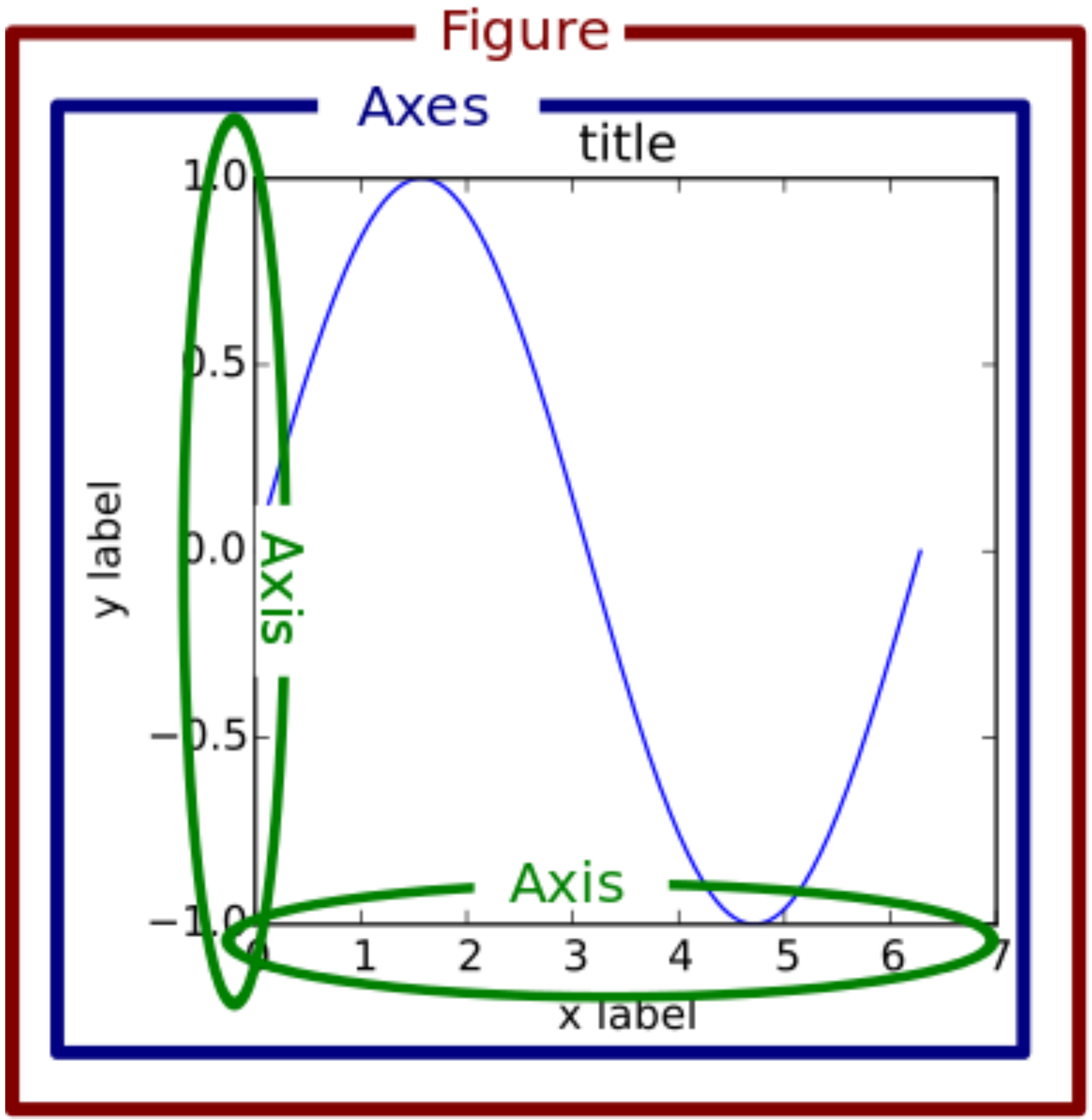
matplotlib tutorials

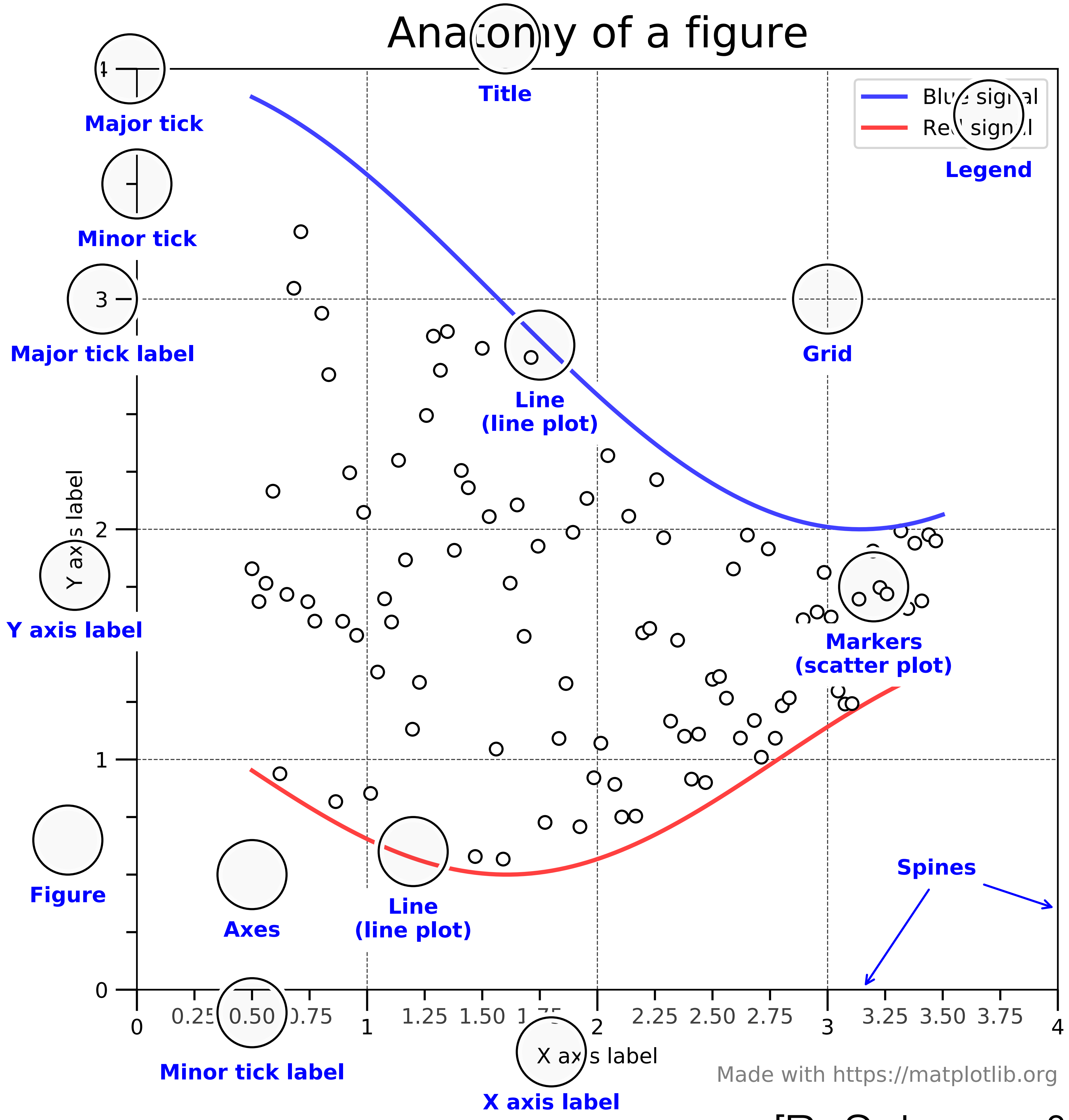
- https://matplotlib.org/stable/tutorials/index.html
- https://github.com/rougier/matplotlib-tutorial

Adding Labels

- `plt.xlabel`: set x label
- `plt.ylabel`: set y label
- `plt.title`: set title
- ```
  plt.plot([1,3,4,6,10],[1,5,2,7,3])
  plt.xlabel('Age')
  plt.ylabel('Number of Jumps')
  plt.title('Kangaroo Jumps Today')
  ```
  ```

Anatomy of a Figure





[B. Solomon & matplotlib]

Figure and Axes Objects

- pyplot is stateful, functions affect the "current" figure and axes
  - `plt.gcf()`: gets current figure
  - `plt.gca()`: gets current axes
  - Creates one if it doesn't exist!
- This is not aligned with object-based programming ideas
- Most methods in pyplot are translated to methods on the current axes (gca)
- We can instead call these directly, but first need to create them:
  - ```
    fig, ax = plt.subplots() # "constructor-like" method
    ax.scatter([1,3,4,6,10],[1,5,2,7,3])
    ```
    ```

Object-Based Plotting

- `fig, ax = plt.subplots() # "constructor-like" method`
  `ax.scatter([1,3,4,6,10],[1,5,2,7,3])`
- Use getters/setters for labels and title
  - `ax.set_xlabel('Age')`
  - `ax.set_ylabel('Number of Jumps')`
  - `ax.set_title('Kangaroo Jumps Today')`
- We can also call methods on the figure:
  - `fig.tight_layout() # reduce margins`

Multiple Figures

- subplots allows multiple axes in the same figure:
  - `fig, ax = plt.subplots(2, 2, figsize=(10, 10)) # rows, then columns`
- `ax` is now a 2x2 numpy array
- Can put any type of visualization on each pair of axes
- ```
  ax[0,0].plot([1,3,4,6,10],[1,5,2,7,3])
  ax[0,1].bar(['Apple','Banana','Orange'],[0.99,0.50,1.25])
  ax[1,0].pcolormesh(x, y, Z)
  ax[1,1].pie([20,40,30,10],
              labels=['Apple','Banana','Orange','Pear'])
  ```
  ```

pandas Integration

- Can call many of these methods directly from pandas
- Handled through `kind` kwarg or `.plot` accessor
- It will try to guess a reasonable visualization, but may fail:
  - `fruit.plot()`
- Instead, specify x and y and other parameters:
  - `fruit.plot(kind='bar',x='name',y='price')`
  - `plt.bar(x='name',height='price',data=fruit) # SIMILAR`
  - `fruit.plot.scatter(x='price',y='count',c='name') # ERROR`
  - ```
    colors = {'Apple': 'red','Orange': 'orange',
              'Banana': 'yellow','Pear': 'green'}
    fruit.plot.scatter(x='price',y='count',
                       c=fruit['name'].map(colors))
    ```
    ```

Assignment 8

- Due Friday, Dec. 7
- Last Assignment
- Data and Visualization
- Same Illinois Unemployment Data
- Use polars or pandas
- Must use matplotlib or altair where directed

Final Exam

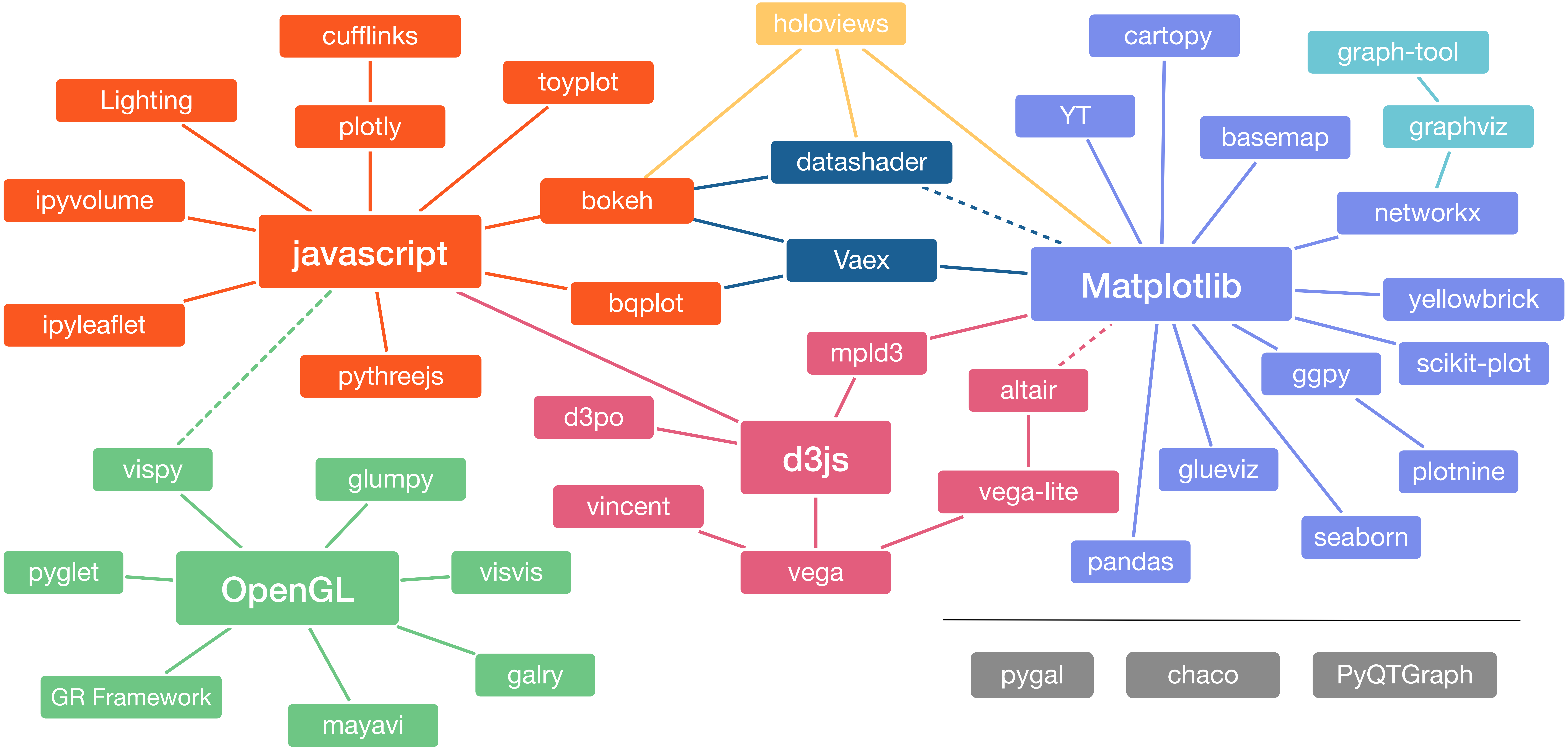
- Wednesday, Dec. 11, **8:00**-9:50am in PM 103
- **More comprehensive** than Test 2
- Expect questions from topics covered on Test 1 and 2
- Expect questions from the last few weeks of class (data, visualization, machine learning)
- Similar format

-

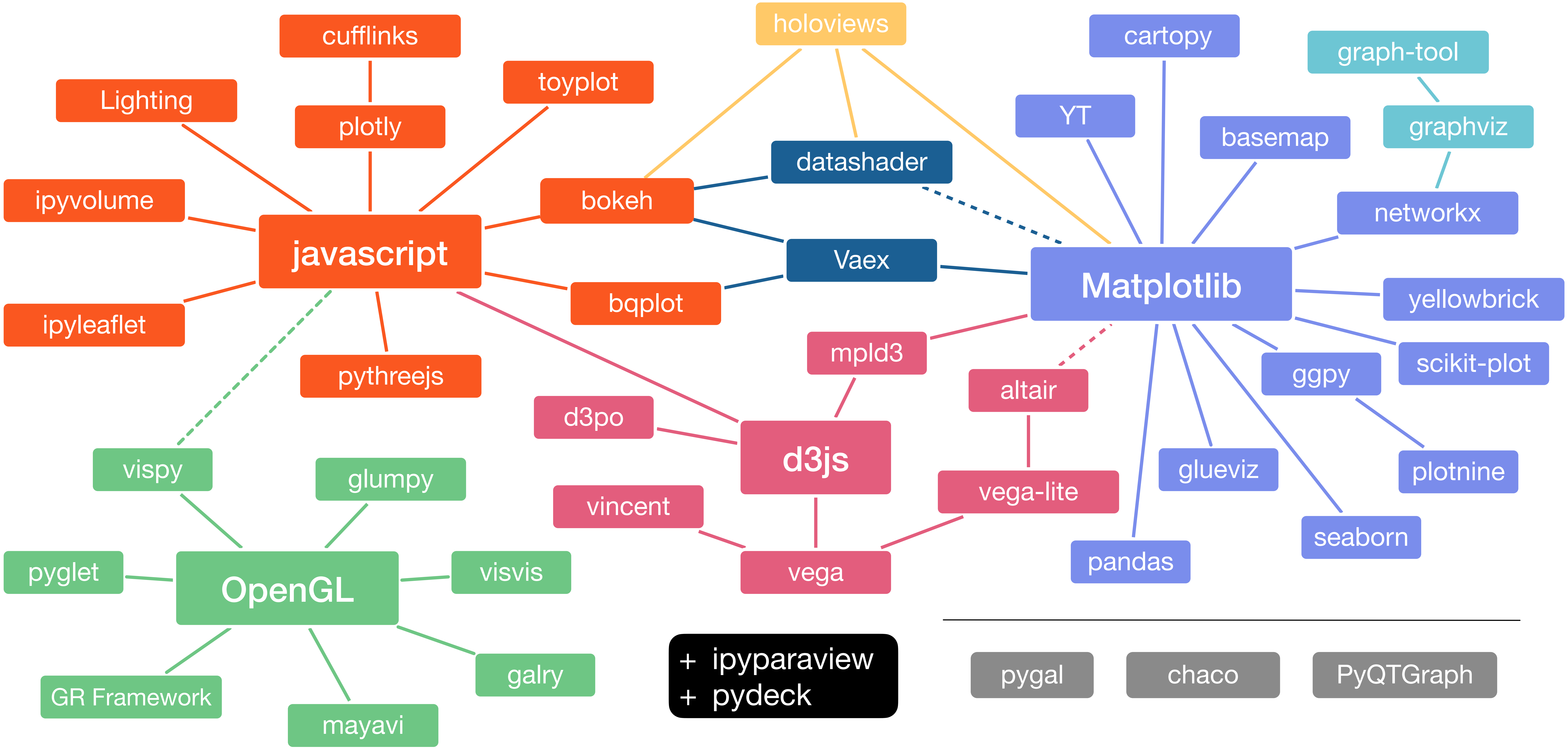
Wednesday

- Wrap up Machine Learning
- Review
  - Bring your questions!
  - Your last chance to ask questions before the final
- Also, teaching evaluations

The Python Visualization Landscape



The Python Visualization Landscape



+ ipyparaview
+ pydeck

History of Vega-Lite & Altair

- "Grammar of Graphics", L. Wilkinson
- "A Layered Grammar of Graphics", H. Wickham
- ggplot: plotting library for R
- Vega: similar idea for Javascript/JSON (U. Washington, A. Satyanarayan)
  - "Declarative language for creating, saving, and sharing interactive visualization designs"
  - More focus on interaction and reactive signals
  - Separation between specification and runtime
- Vega-Lite: higher-level language than Vega (U. Washington, D. Moritz)
  - uses carefully designed rules to default settings

History of Vega-Lite & Altair

- Altair: Python interface to Vega-Lite (J. VanderPlas)
  - "spend more time understanding your data and its meaning"
  - Specify the what, minimize the amount of code directing the how
  - Python can write JSON specification just as well as any other language
  - Bindings make it more Python-friendly, integrate with pandas, add support for Jupyter, etc.
- Vega Fusion (J. Mease)
  - Scaling to larger datasets
  - Serverside scaling

Basic Example

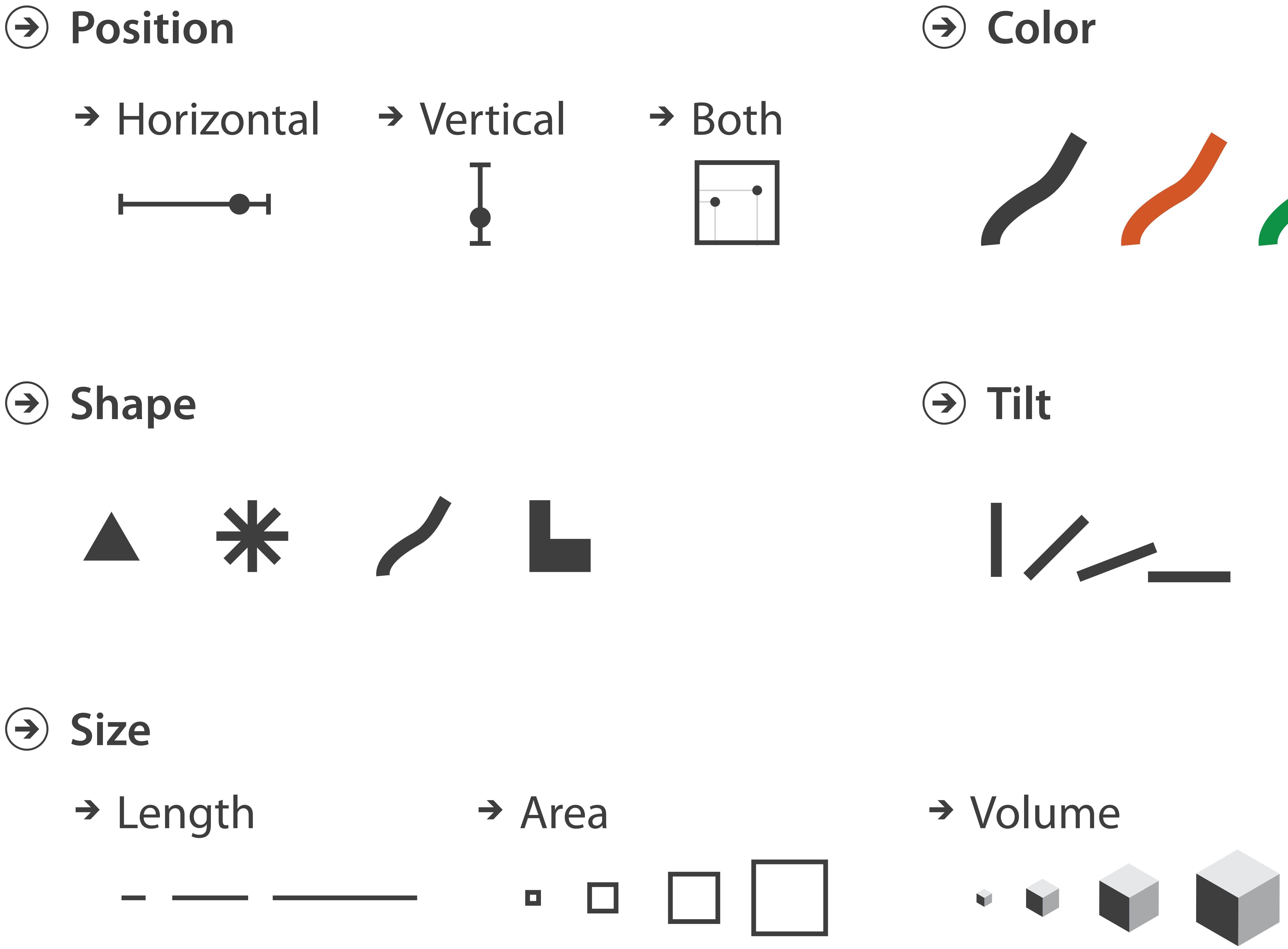
- ```
  import altair as alt
  import pandas as pd
  data = pd.DataFrame({'x': [1,3,4,6,10],'y': [1,5,2,7,3]})
  alt.Chart(data).mark_line().encode(x='x', y='y')
  ```
- Easiest to use data from a pandas data frame
  - Another option is a csv or json file
  - Can support geo_interface, too
- `Chart` is the basic unit
- Mark: `.mark_*()` indicates the geometry created for each data item
- Encode: `.encode()` allows visual properties to be set to data attributes
  ```

Visual Marks

- **Marks** are the basic graphical elements in a visualization
- Marks classified by dimensionality:

- Also can have surfaces, volumes
- Think of marks as a mathematical definition, or if familiar with tools like Adobe Illustrator or Inkscape, the path & point definitions
- Altair: area, bar, circle, geoshape, image, line, point, rect, rule, square, text, tick
  - Also compound marks: boxplot, errorband, errorbar

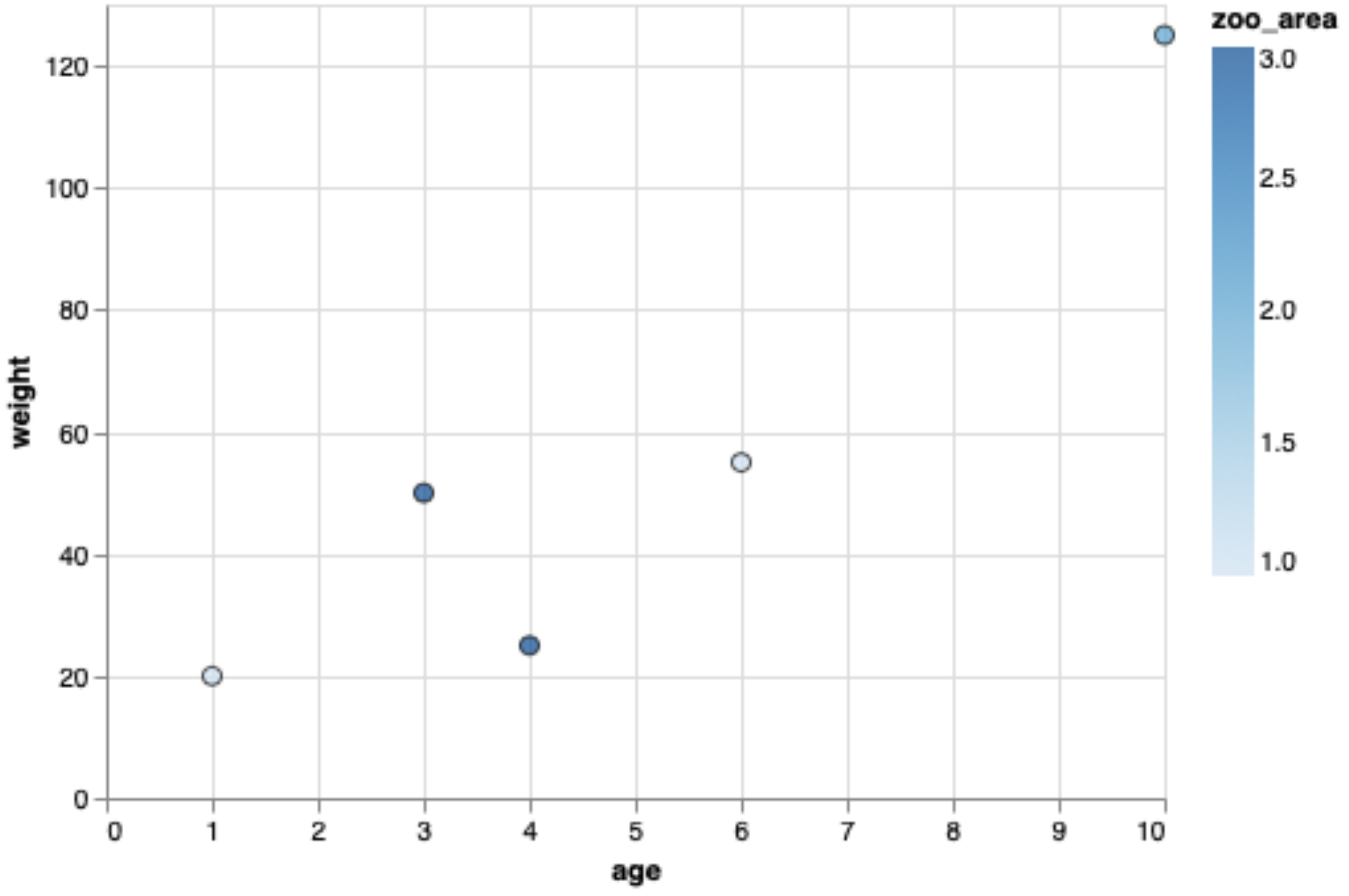
Encode via Visual Channels



[Munzner (ill. Maguire), 2014]

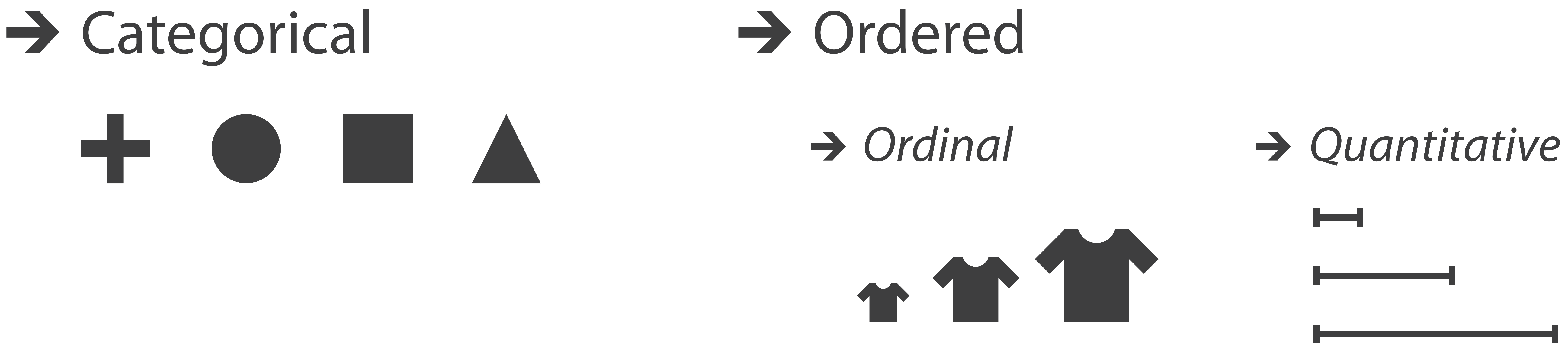
Easily Explore Different Encodings

- ```
  data = pd.DataFrame({
        'age': [1,3,4,6,10],
        'weight': [20,50,25,55,125],
        'zoo_area': [1,3,3,1,2],
        'num_scoops': [3,2,4,2,3]
  })
  alt.Chart(data).mark_point(
        filled=True, size=50,
        stroke='black',strokeWidth=1
  ).encode(
        x='age',
        y='weight',
        color='zoo_area'
  )
  ```


  ```

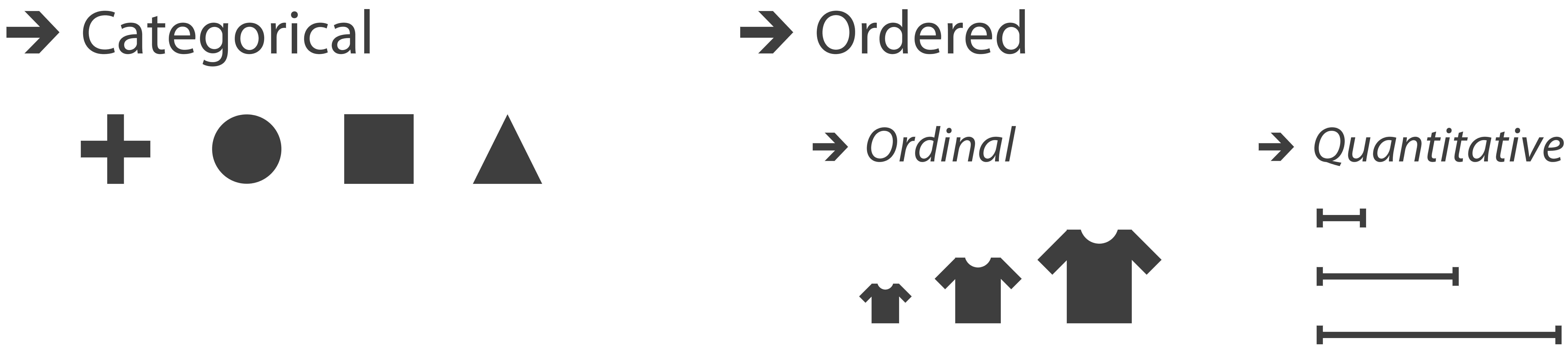
Problem: zoo_area is not a continuous value, nor is it ordered in any way!

Data Attributes and Altair Types



[Munzner (ill. Maguire), 2014]

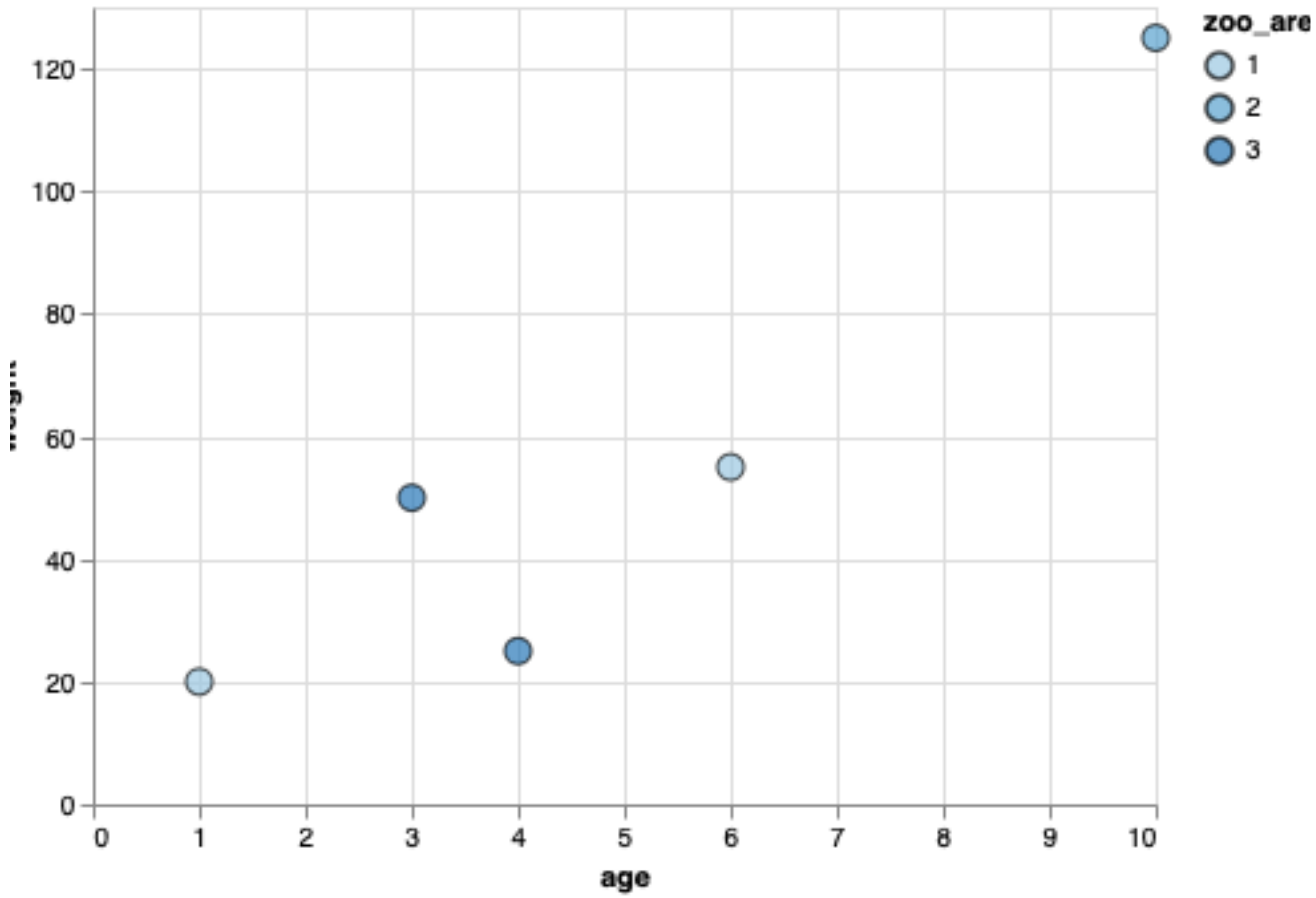
Data Attributes and Altair Types



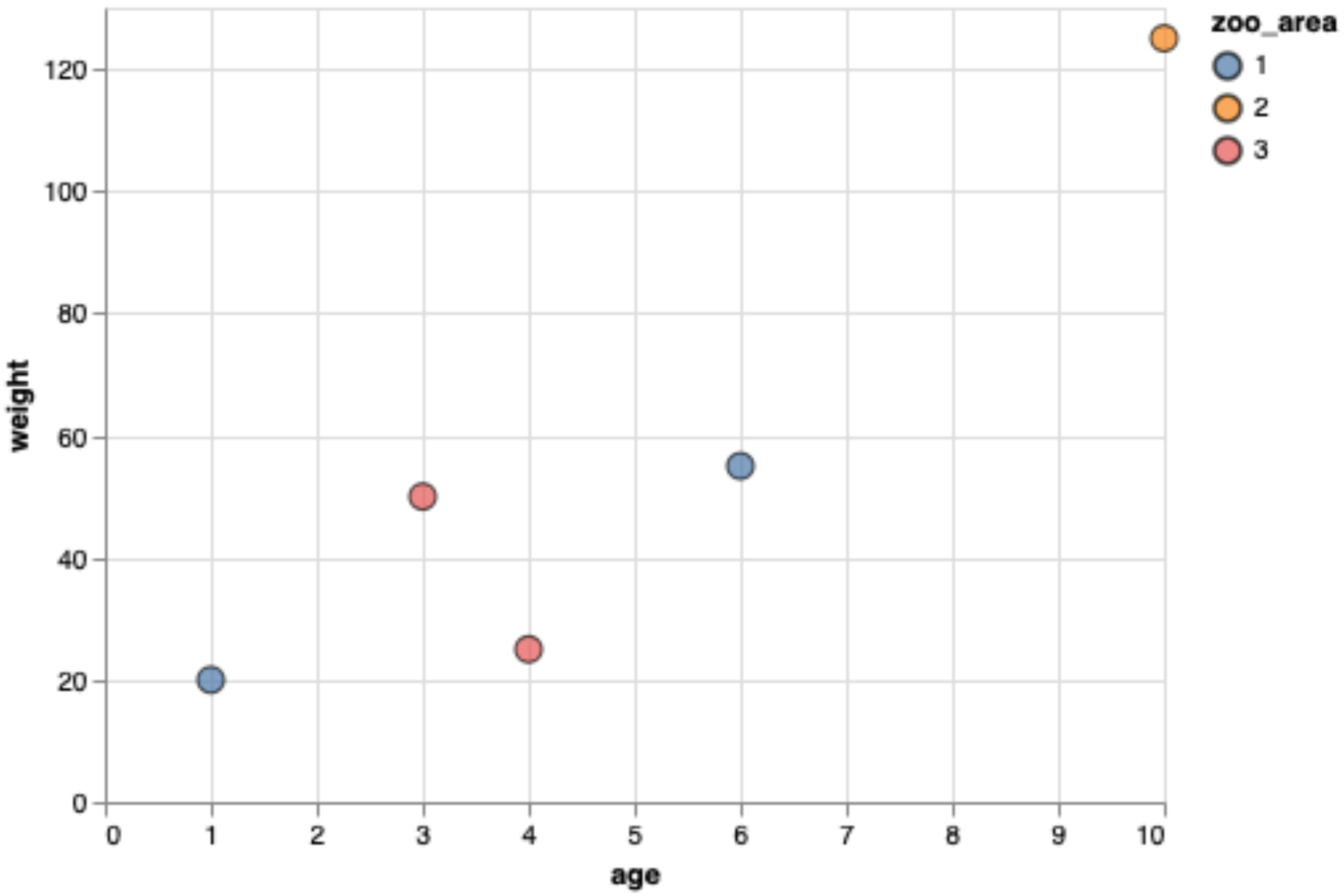
- Categorical data = Nominal (N)
- Ordinal data = Ordinal (O)
- Quantitative data = Quantitative (Q)
- Temporal data = Temporal (T)

[Munzner (ill. Maguire), 2014]

Specifying the Type

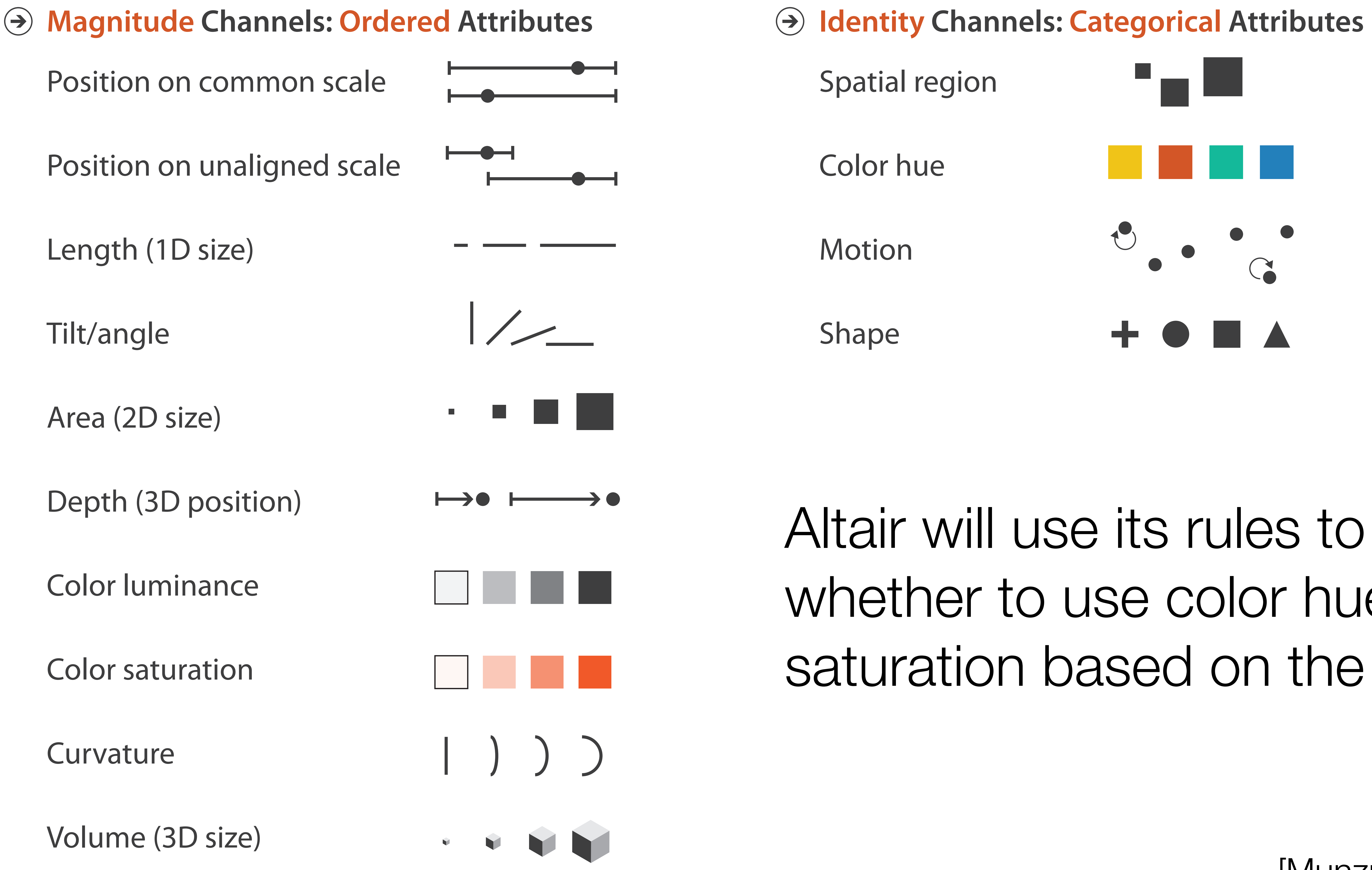


zoo_area:O



zoo_area:N

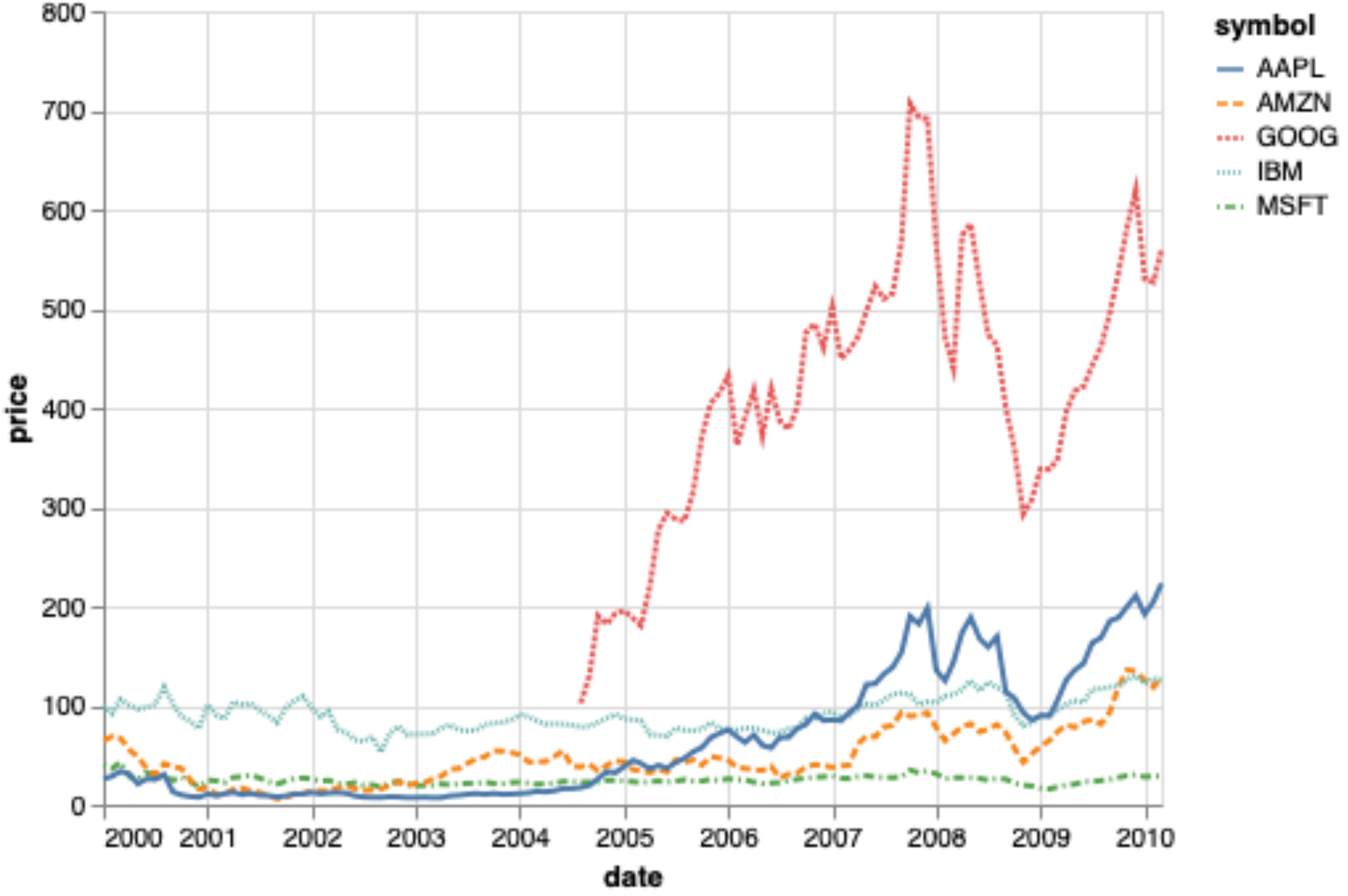
Different Channels for Different Attribute Types



Altair will use its rules to pick whether to use color hue or saturation based on the type

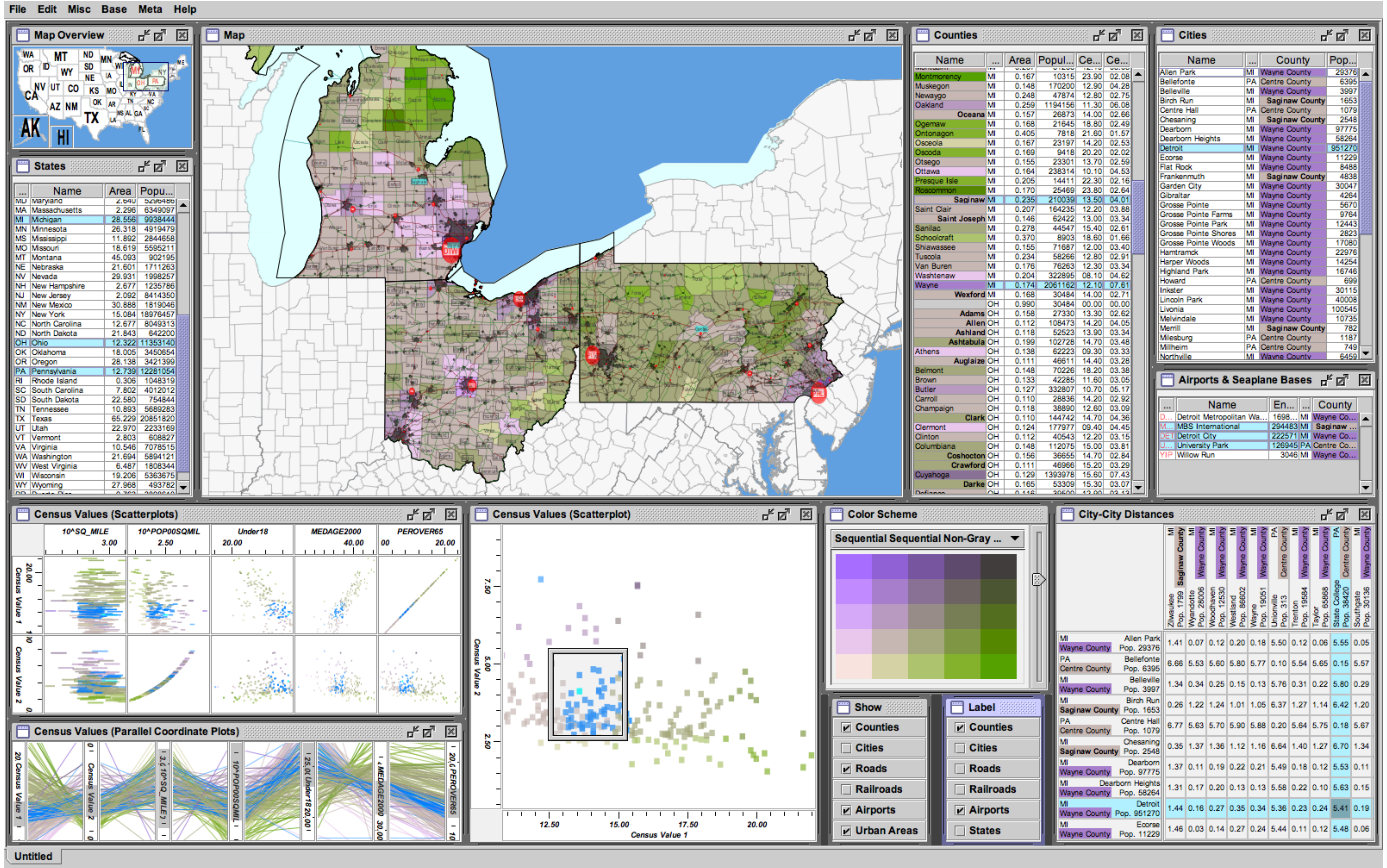
[Munzner (ill. Maguire), 2014]

Multiple Views in Visualization



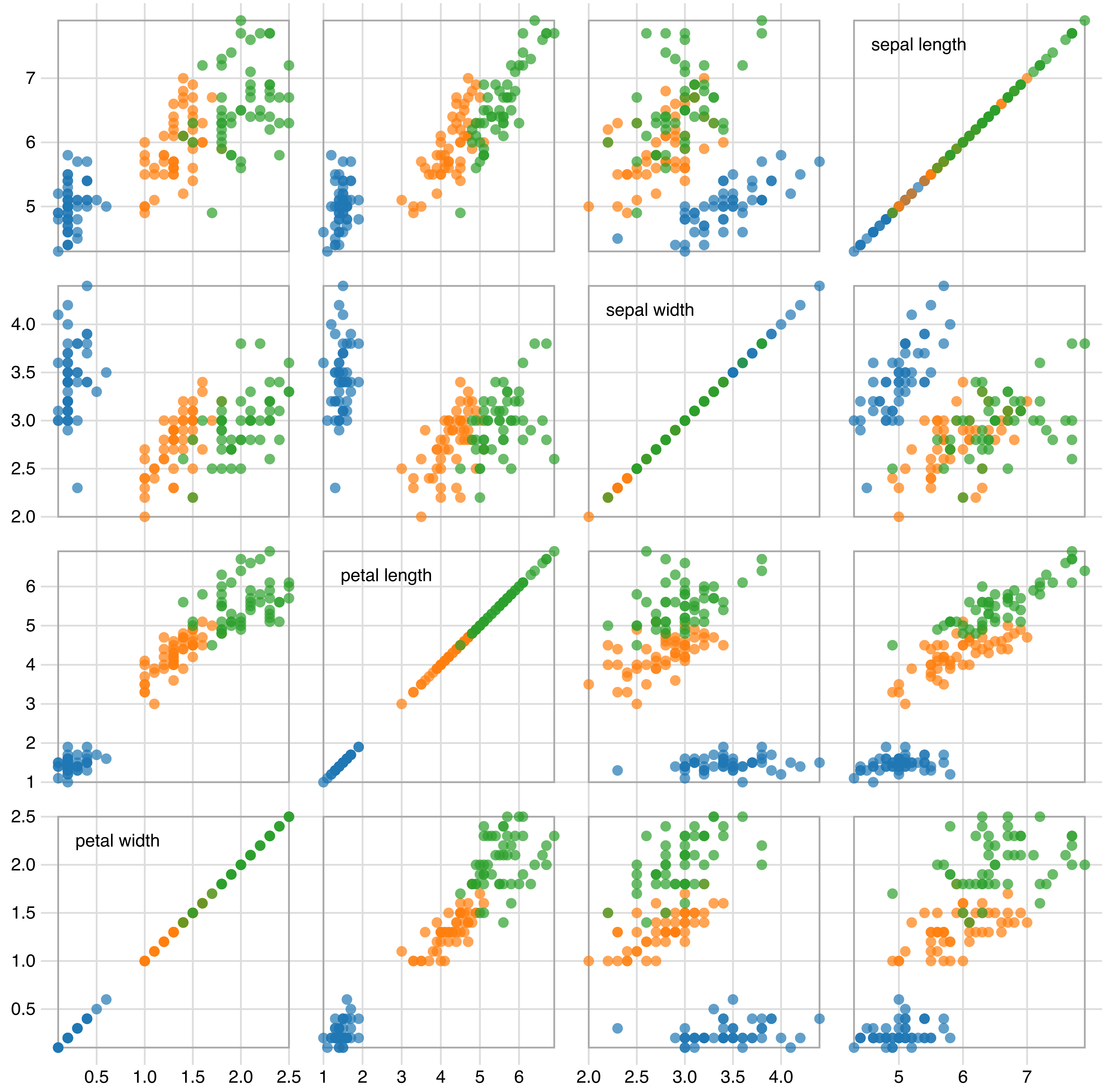
[Altair]

Multiple Views in Visualization



[Improvise, Weaver, 2004]

Multiple Views in Visualization

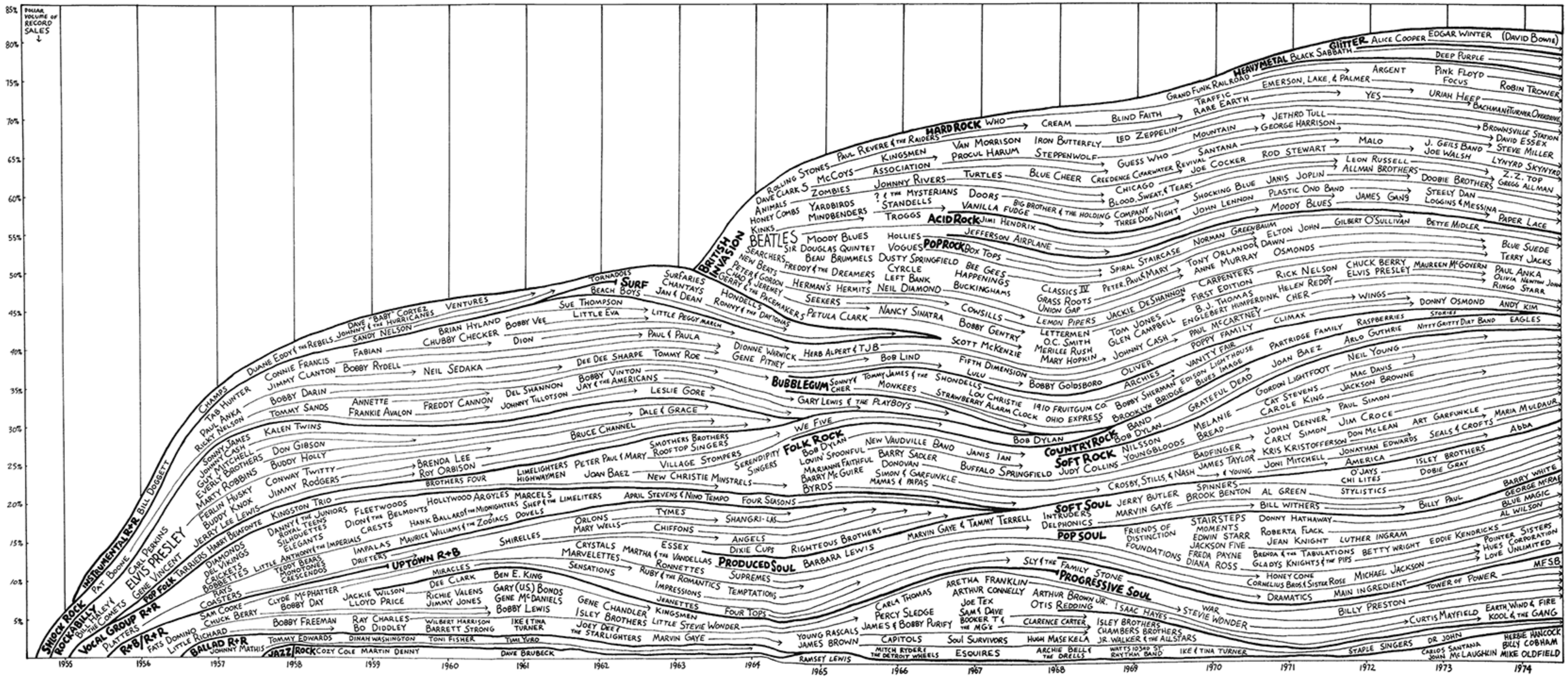


[M. Bostock]

Altair Supports Concatenation, Layering, & Repetition

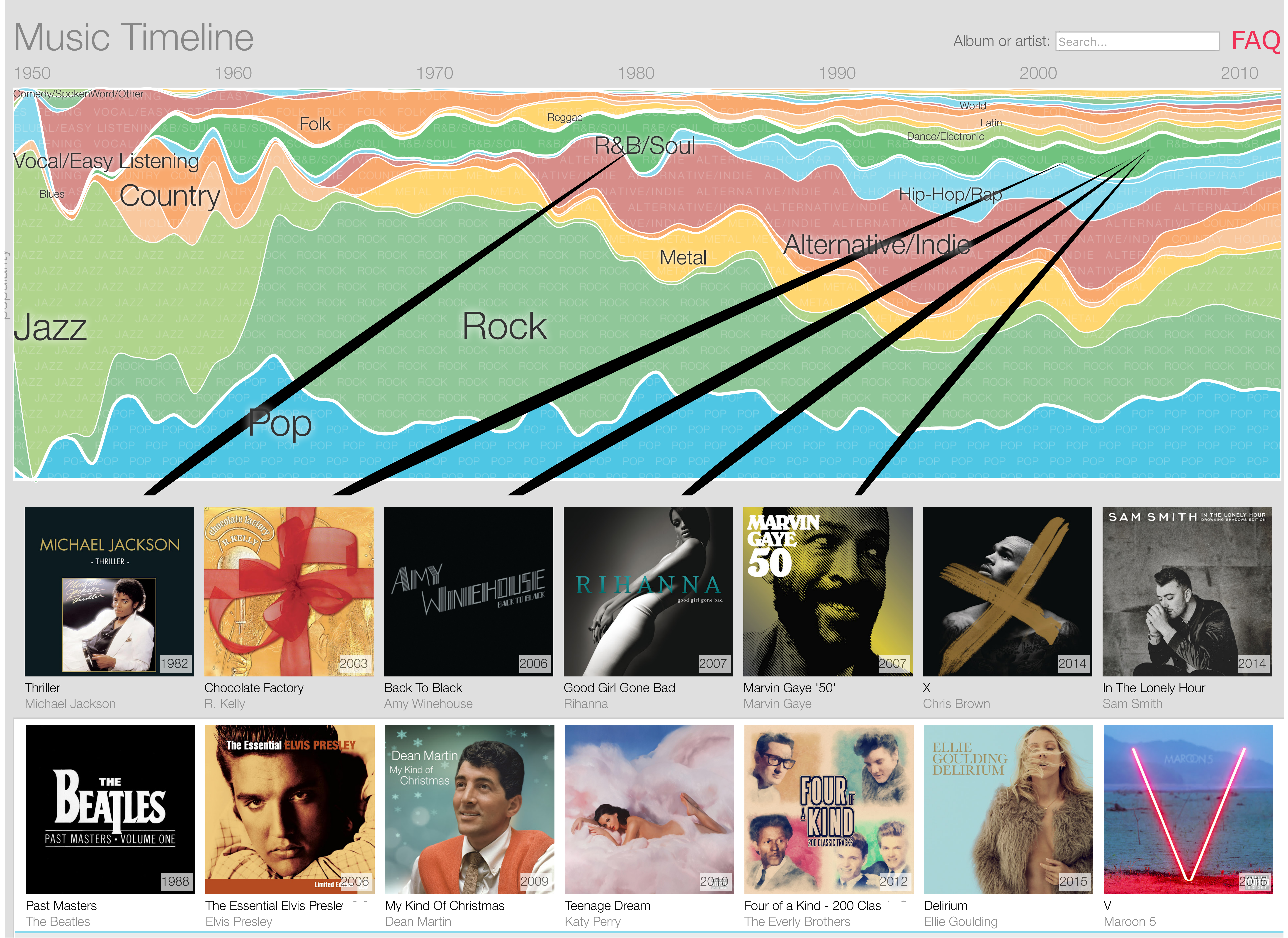
- Layering:
  - \+ Operator
- Concatenation:
  - Horizontal: | operator
  - Vertical: & operator
- Repetition
  - Use of .repeat for layout
  - Reference repeated variables in the encoding

Visualization



[Rock 'N' Roll is Here to Pay, R. Garofalo, 1977 (via Tufte)]

Also Visualization, but with Interaction



[Music Timeline, Google Research (no working version)]

Interaction

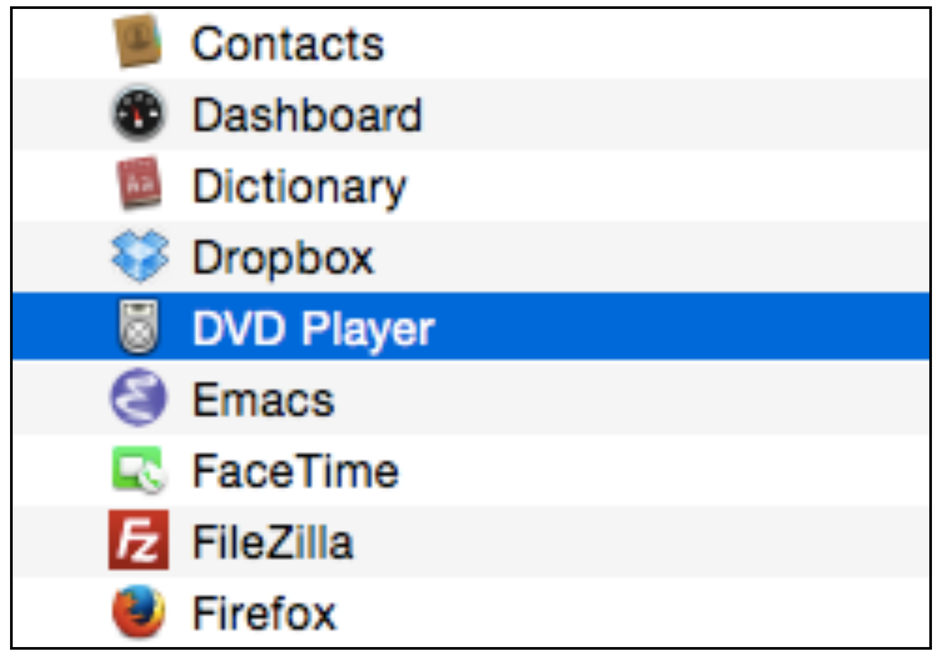
- Grammar of Graphics, why not Grammar of Interaction?
- Vega-Lite/Altair is about **interactive** graphics
- Types of Interactions:
  - Selection
  - Zoom
  - Brushing

Selection

- Selection is often used to initiate other changes
- User needs to select something to drive the next change
- What can be a selection target?
  - Items, links, attributes, (views)
- How?
  - mouse click, mouse hover, touch
  - keyboard modifiers, right/left mouse click, force
- Selection modes:
  - Single, multiple
  - Contiguous?

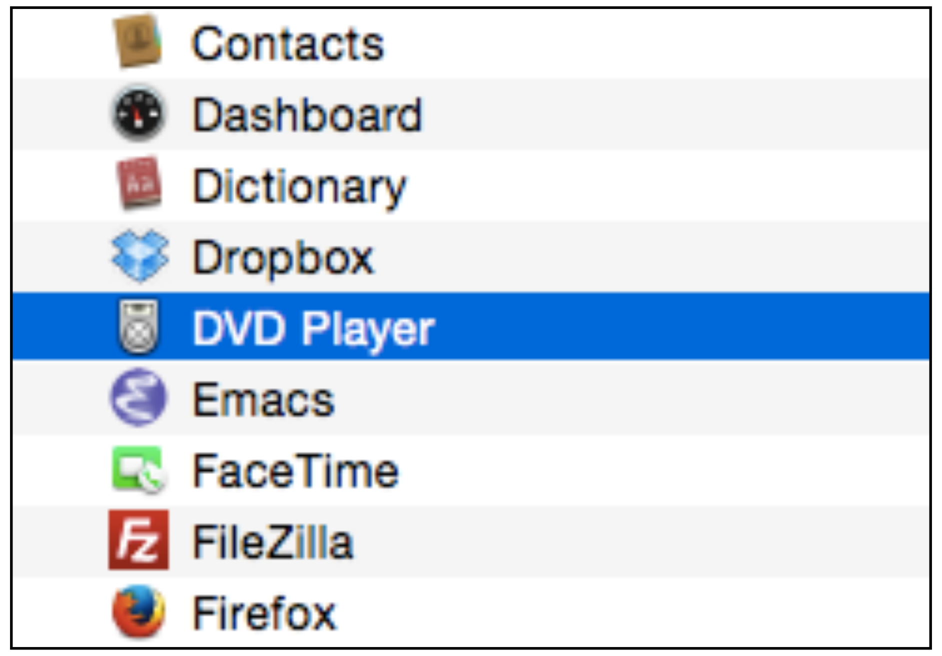
Highlighting

- Selection is the user action
- Feedback is important!
- How? Change selected item's visual encoding
  - Change color: want to achieve visual popout
  - Add outline mark: allows original color to be preserved
  - Change size (line width)
  - Add motion: marching ants



Highlighting

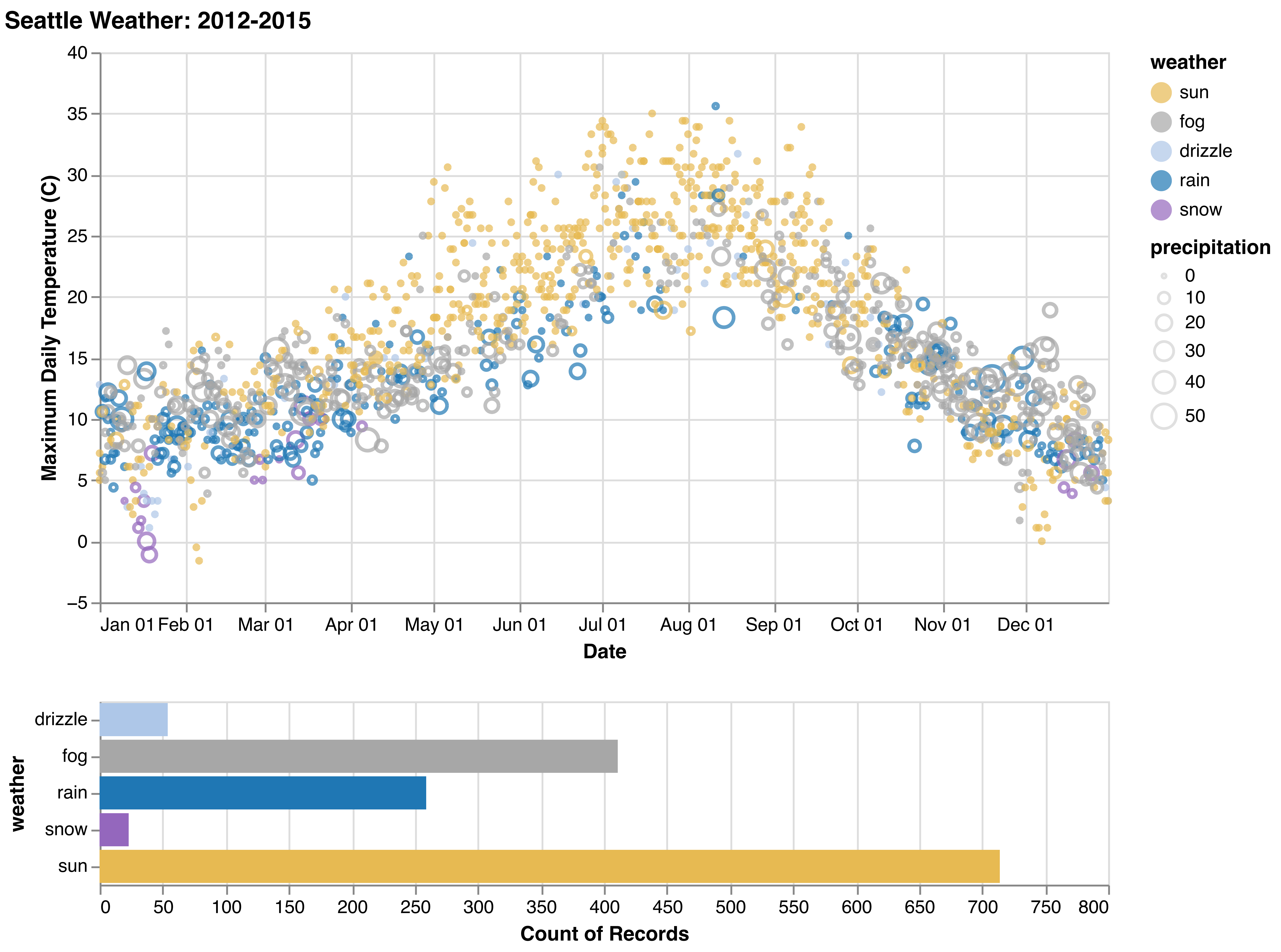
- Selection is the user action
- Feedback is important!
- How? Change selected item's visual encoding
  - Change color: want to achieve visual popout
  - Add outline mark: allows original color to be preserved
  - Change size (line width)
  - Add motion: marching ants



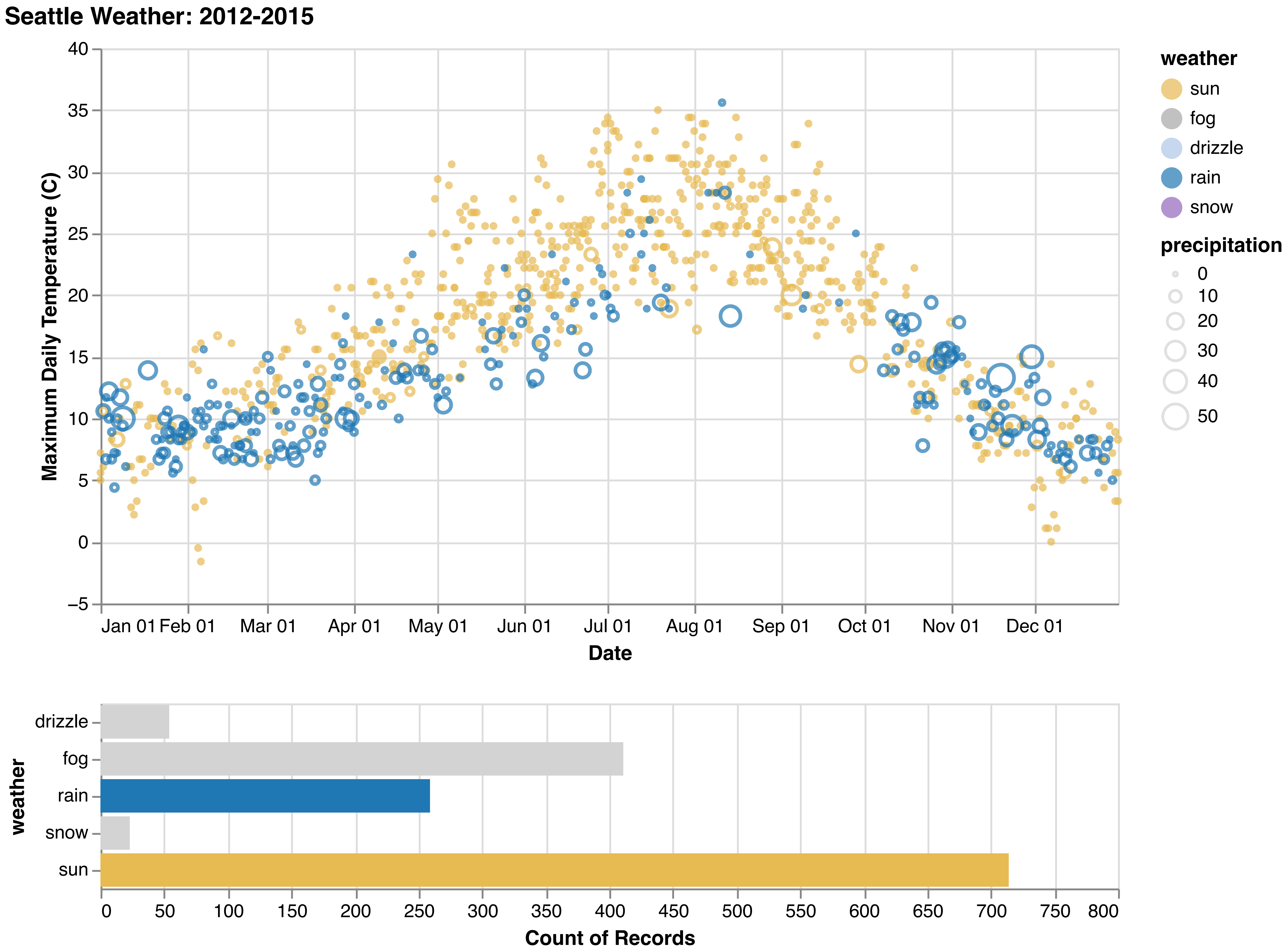
Altair's Interactive Charts

- https://altair-viz.github.io/gallery/index.html#interactive-charts

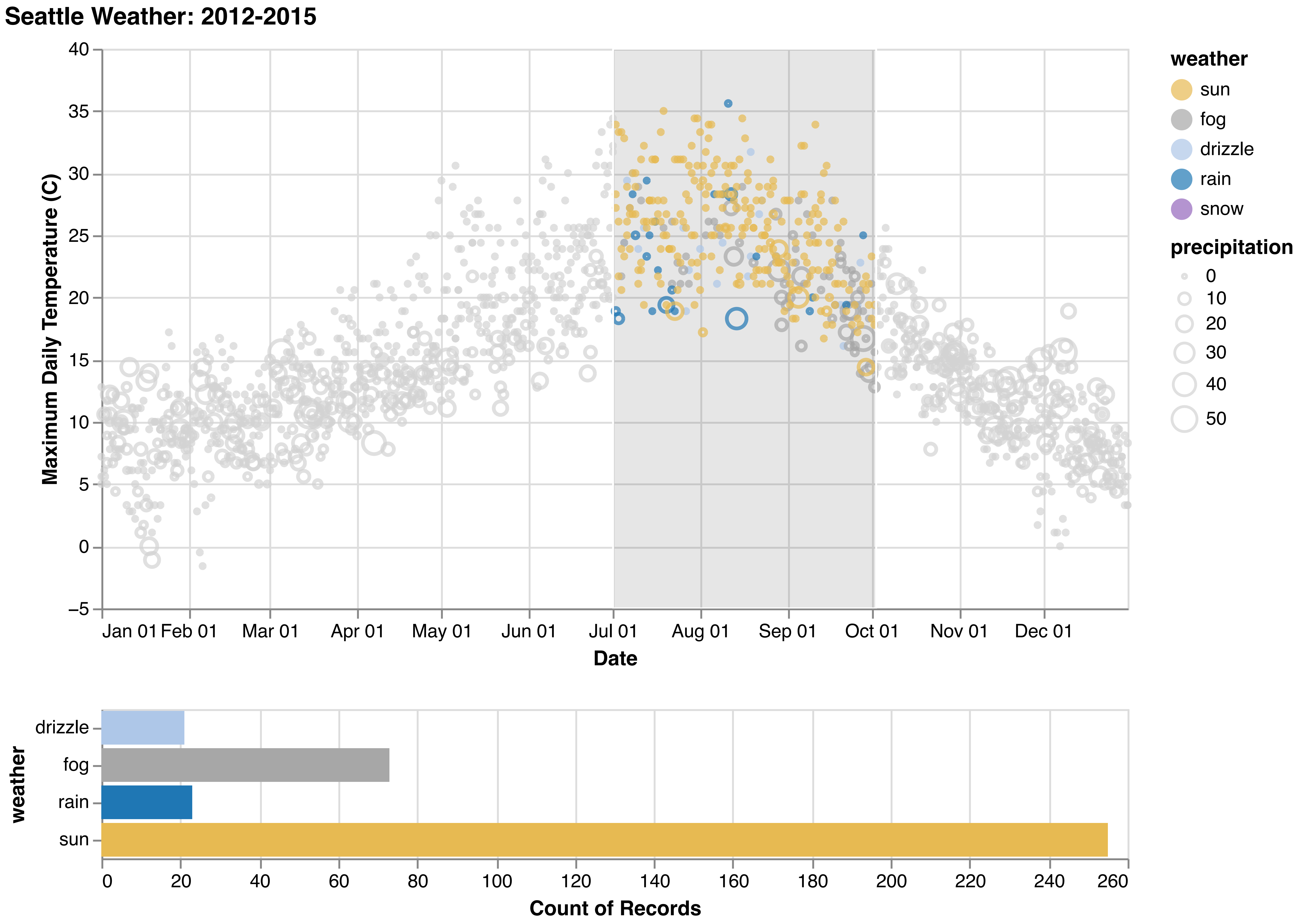
Interaction



Weather Selection: Rain vs. Sun



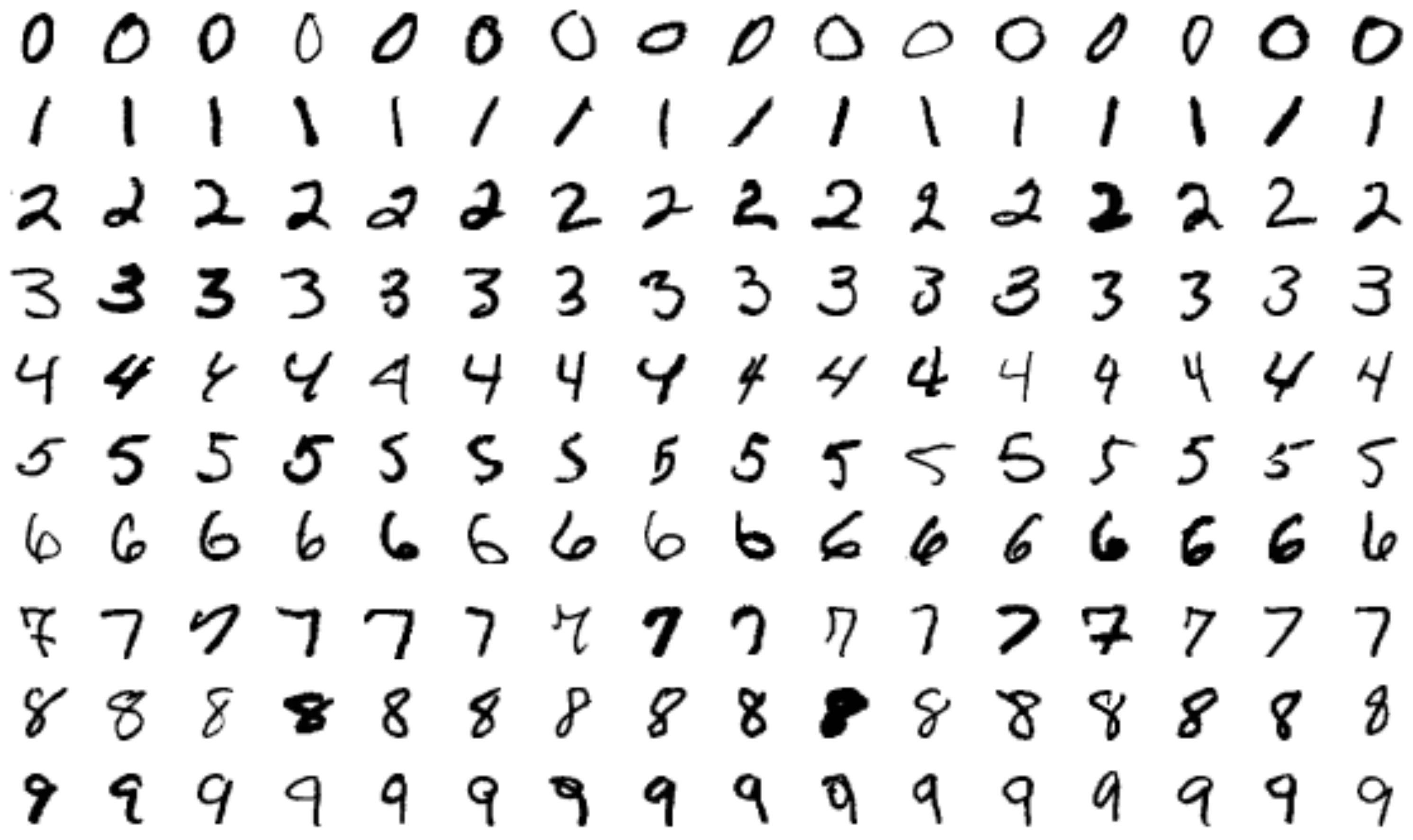
Date Selection: July-September Sun



Machine Learning in Python

Tasks Machine Learning can Help With

- Identifying the zip code from handwritten digits on an envelope



- Detecting fraudulent activity in credit card transactions
- Identifying topics in a set of blog posts
- Grouping customers with similar preferences

[A. Müller & S. Guido, Introduction to Machine Learning with Python, J. Steppan (MNIST image)]

When to Use Machine Learning?

- ML is used when:
  - Human expertise does not exist (navigating on Mars)
  - Humans can't explain their expertise (speech recognition)
  - Models must be customized (personalized medicine)
  - Models are based on huge amounts of data (genomics)
- ML isn't always useful:
  - Calculating payroll…

[E. Alpaydin via E. Eaton]

Questions when building a machine learning solution

- What question(s) am I trying to answer? Do I think the data collected can answer that question?
- What is the best way to phrase my question(s) as a machine learning problem?
- Have I collected enough data to represent the problem I want to solve?
- What features of the data did I extract, and will these enable the right predictions?
- How will I measure success in my application?

[A. Müller & S. Guido]

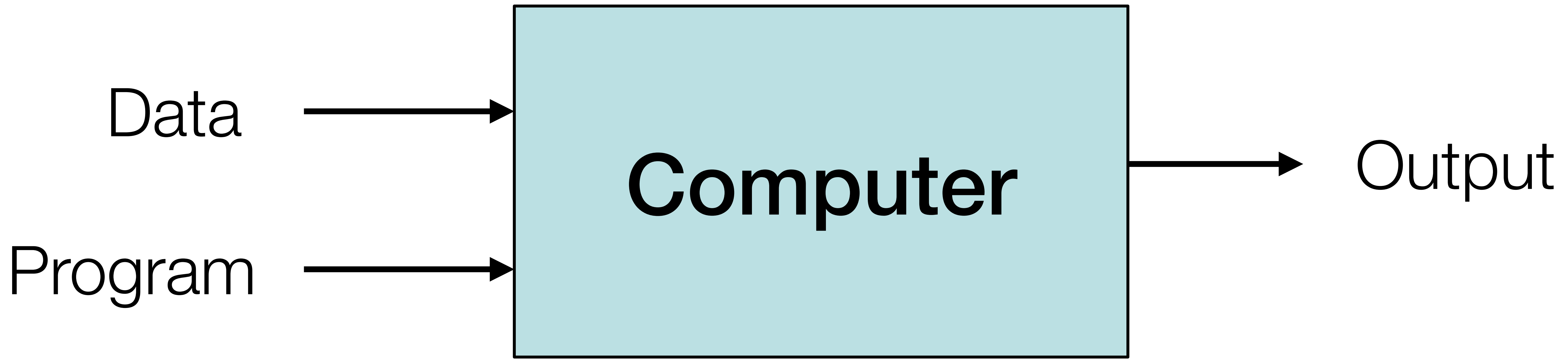
Machine Learning Workflow Overview

1. Should I use ML on this problem?
   - Is there a pattern to detect? Can I solve it analytically? Do I have data?
2. Gather and organize data.
   - Preprocessing, cleaning, visualizing.
3. Establishing a baseline.
4. Choosing a model, loss, regularization, …
5. Optimization (could be simple, could be a Phd…).
6. Hyperparameter search.
7. Analyze performance & mistakes, and iterate back to step 4 (or 2).

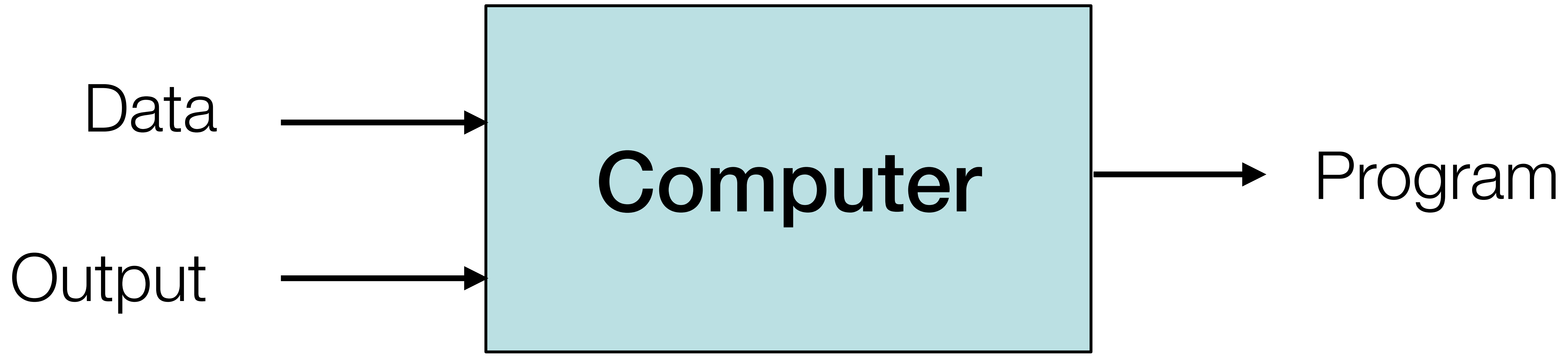
[R. Grosse et al.]

Machine Learning

- Traditional Programming



- Machine Learning



[P. Domingos]

Machine Learning

- Every machine learning algorithm has three components:
  - Representation
  - Evaluation
  - Optimization

Representation

- Decision trees
- Sets of rules / Logic programs
- Instances
- Graphical models (Bayes/Markov nets)
- Neural networks
- Support vector machines
- Model ensembles
- Etc.

[P. Domingos]

Evaluation

- Accuracy
- Precision and recall
- Squared error
- Likelihood
- Posterior probability
- Cost / Utility
- Margin
- Entropy
- K-L divergence
- Etc.

[P. Domingos]

Optimization

- Combinatorial optimization
  - E.g.: Greedy search
- Convex optimization
  - E.g.: Gradient descent
- Constrained optimization
  - E.g.: Linear programming

[P. Domingos]

Final Exam

- Wednesday, Dec. 11, **8:00**-9:50am in PM 103
- **More comprehensive** than Test 2
- Expect questions from topics covered on Test 1 and 2
- Expect questions from the last few weeks of class (data, visualization, machine learning)
- Similar format

- Bring Questions on Wednesday!