

VEGA-LITE

A Grammar of Interactive Graphics

Kanit "Ham" Wongsuphasawat @kanitw

Dominik Moritz @domoritz

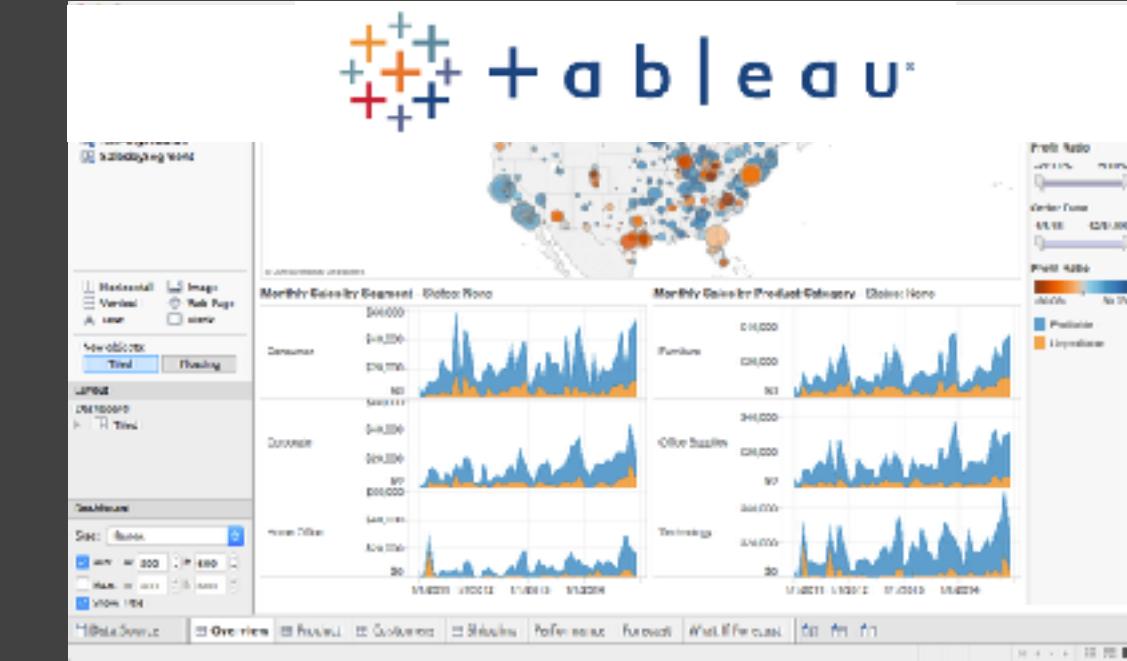
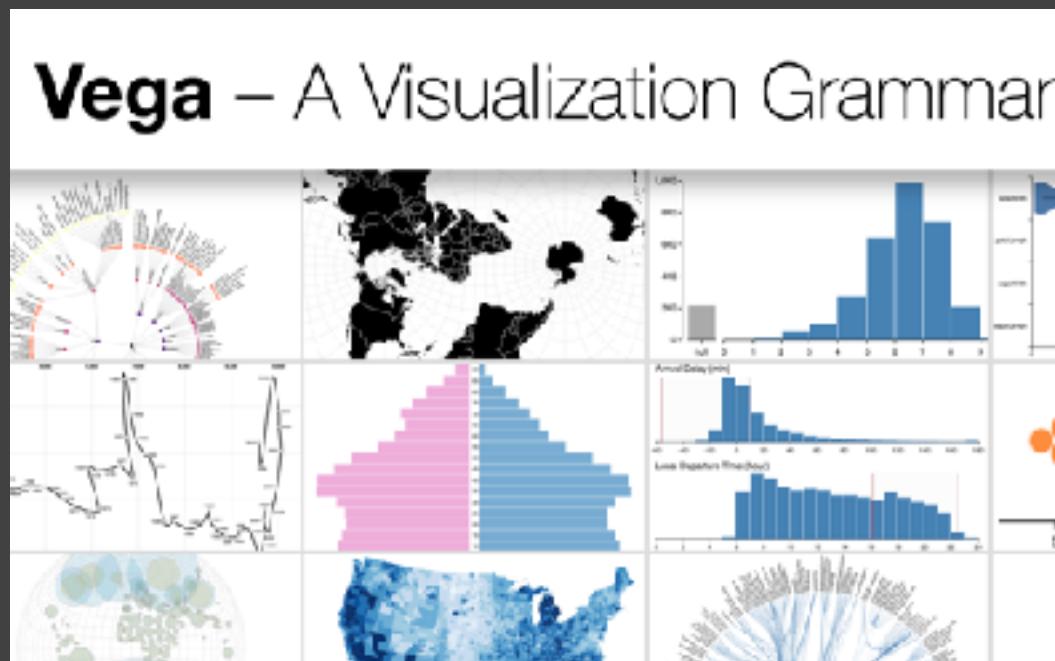
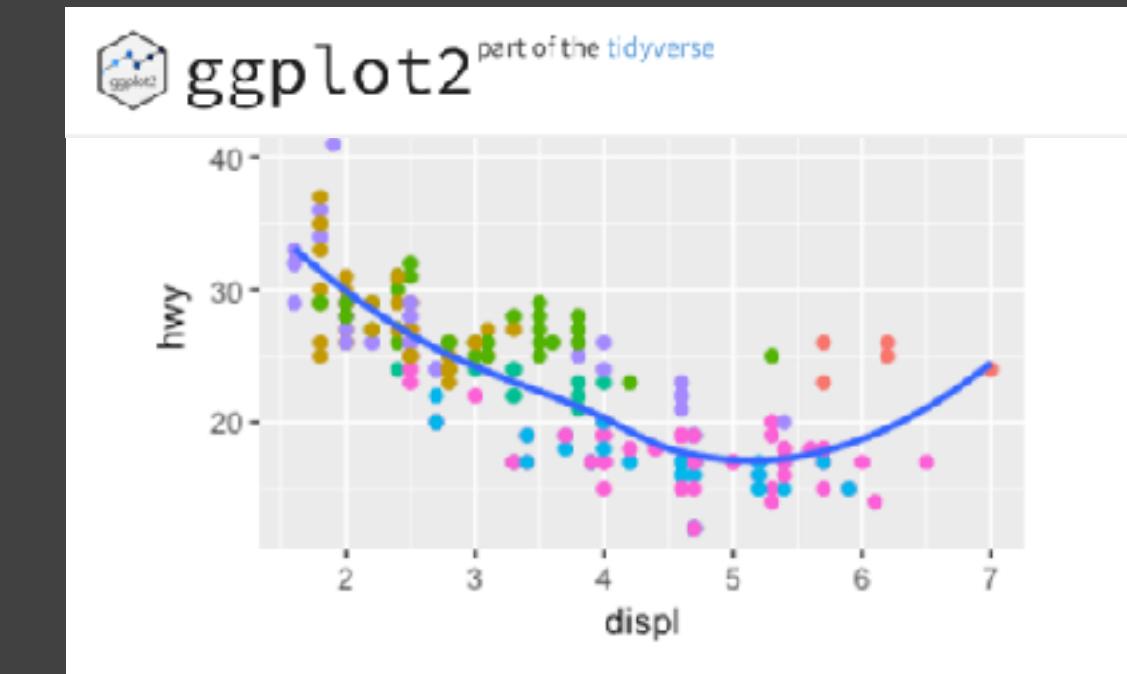
Arvind Satyanarayan @arvindsatya1

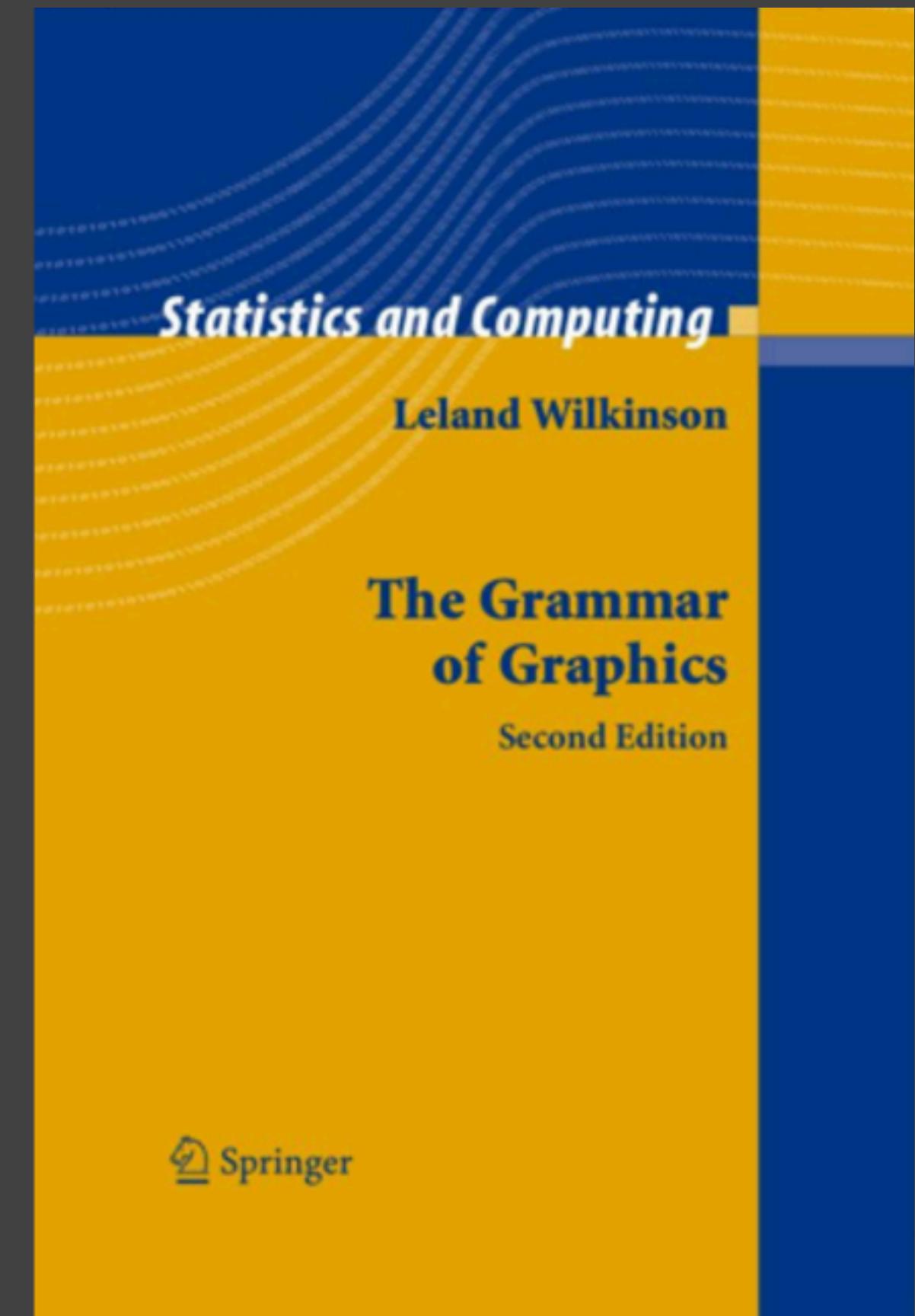
Jeffrey Heer @jeffrey_heer

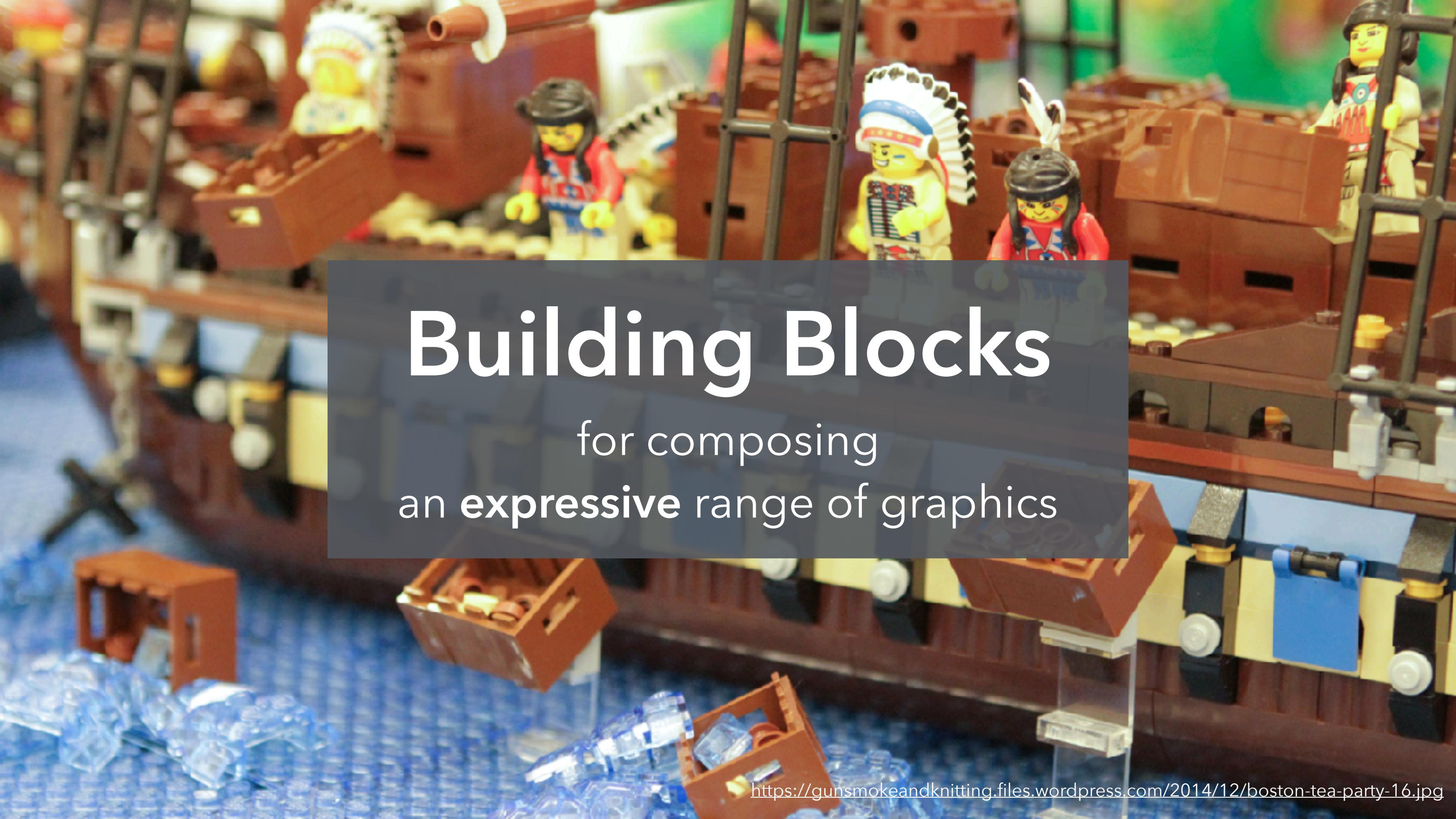


Interactive Data Lab @uwdata
University of Washington







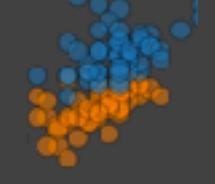
A detailed LEGO model of the Boston Tea Party scene. In the foreground, a small boat is filled with tea crates, heading towards a larger ship. On the ship, several LEGO minifigures dressed as colonists and Native Americans are gathered. One figure in the center wears a large white feathered headdress. The background shows a colorful sky and more LEGO structures.

Building Blocks
for composing
an **expressive** range of graphics

Grammar of Graphics

Data

Input data source to visualize.

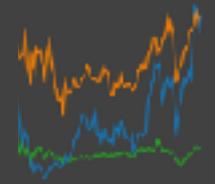
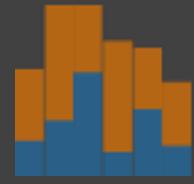


Area

Point/Symbol

Transform

Filter, aggregation, binning, etc.



Bar

Line

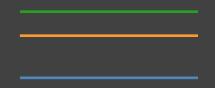
Mark

Data-representative graphics.

Encoding

Mapping between data and mark properties.

Abc



Text

Rule

Scale

Functions that map data values to visual values.



Tick

Rect

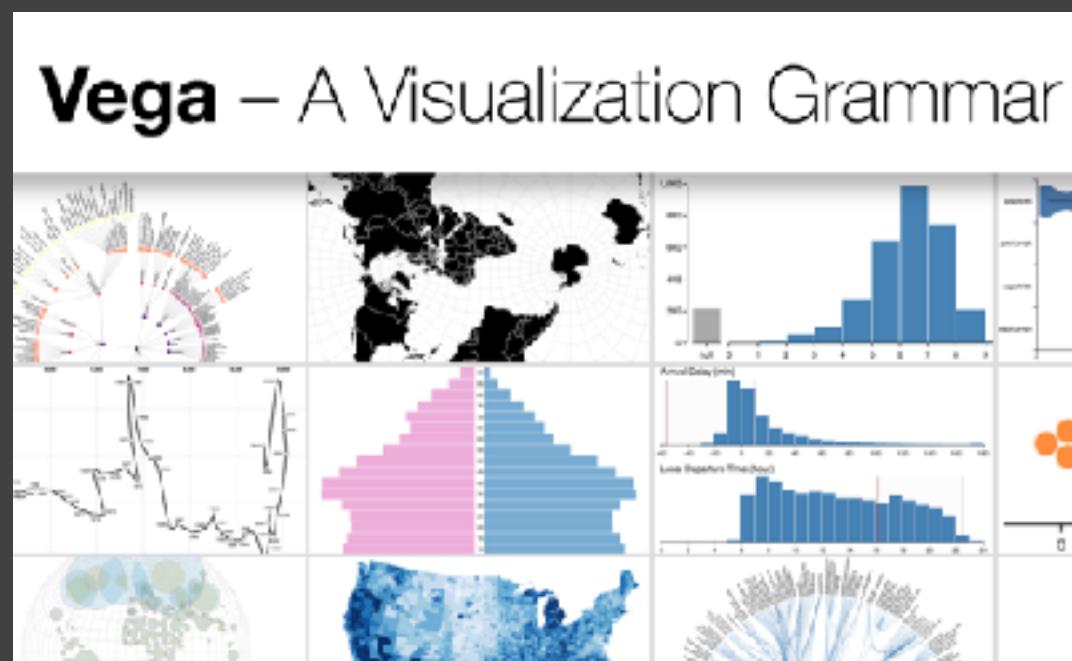
Guides

Axes & legends that visualize scales.

Grammar of Graphics for Customized Designs

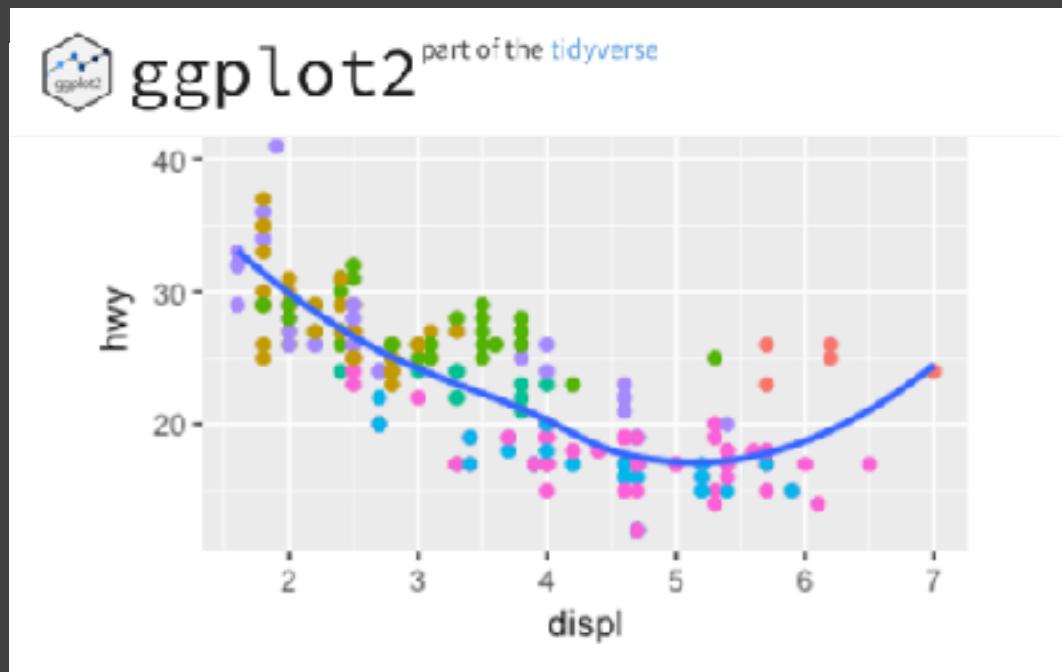


Offer **fine-grained control** for composing interactive graphics.

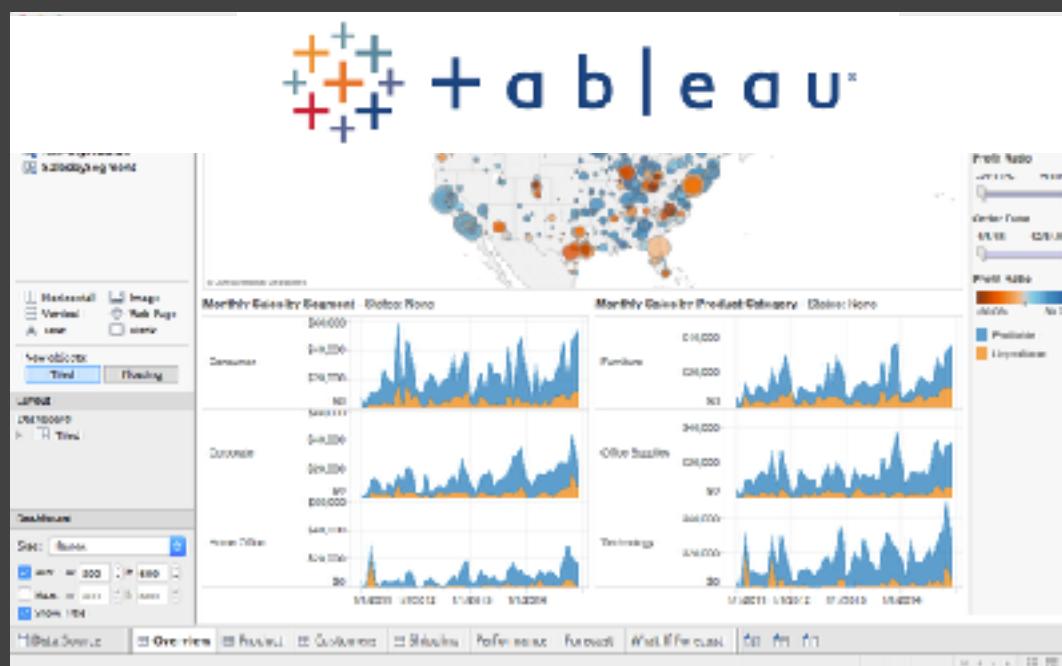


But require **verbose** specifications and technical expertise.

Grammar of Graphics for Exploration



Facilitate **rapid exploration** with **concise** specifications by omitting low-level details.



Infer **sensible defaults** and allow customization by overriding defaults.

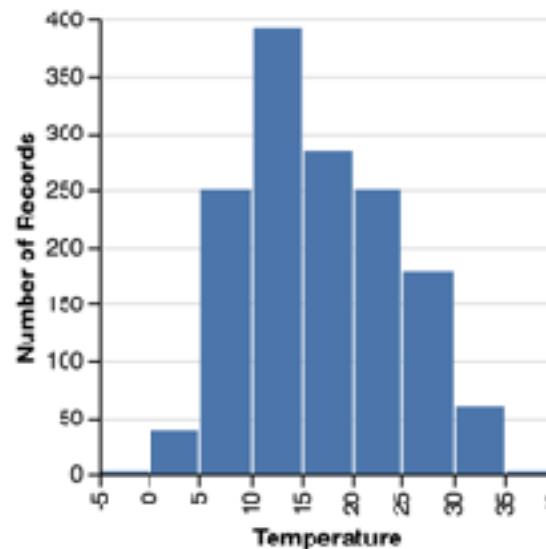
But **limited support for interactions**.

Vega-Lite's Mission

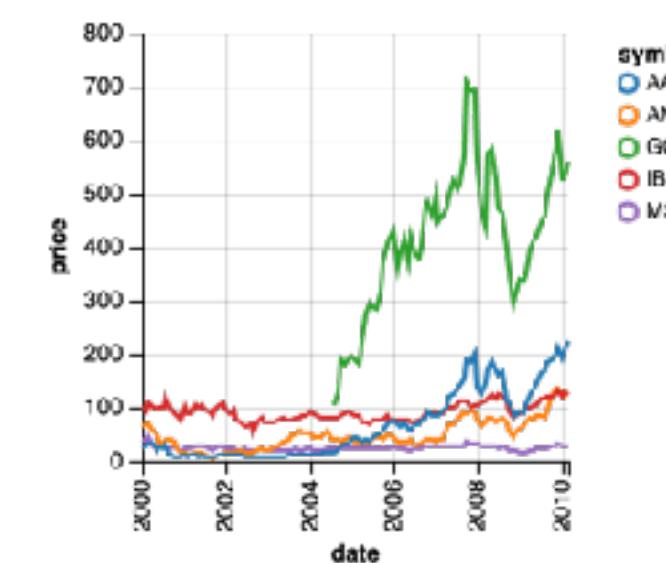
Facilitate exploratory data analysis
with an **expressive** yet **concise** language
to specify ***interactive multi-view graphics***

Vega-Lite: a Grammar of Graphics

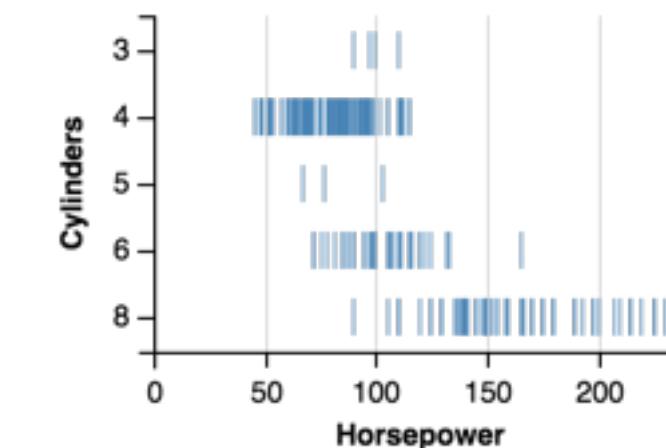
Histogram



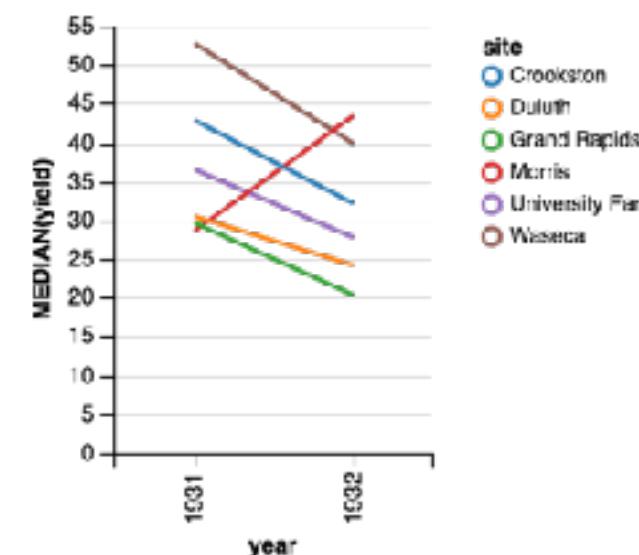
Multi-series Line Chart



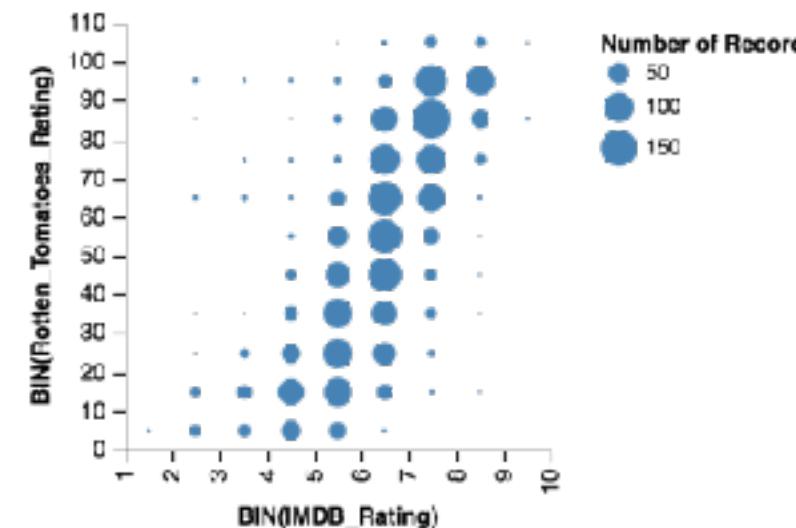
Stripplot



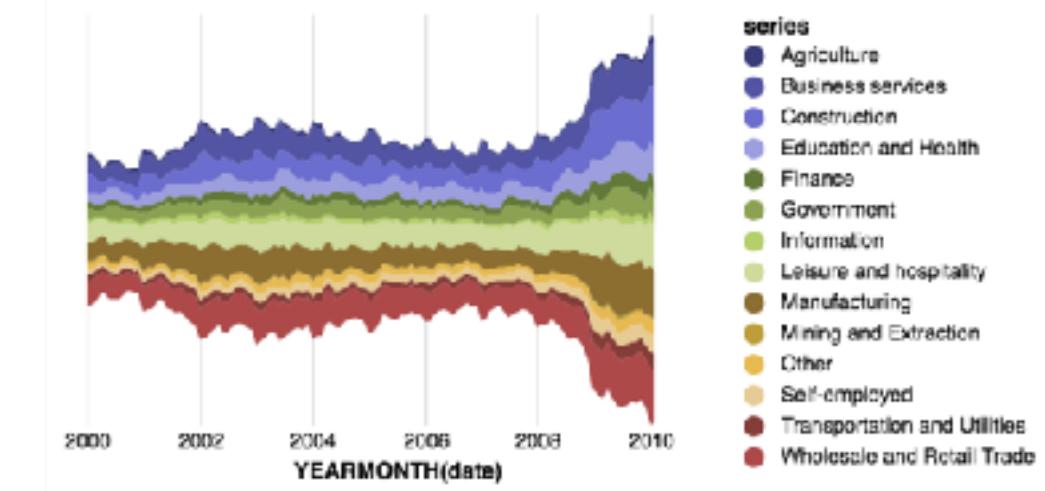
Slope Graph



Binned Scatterplot

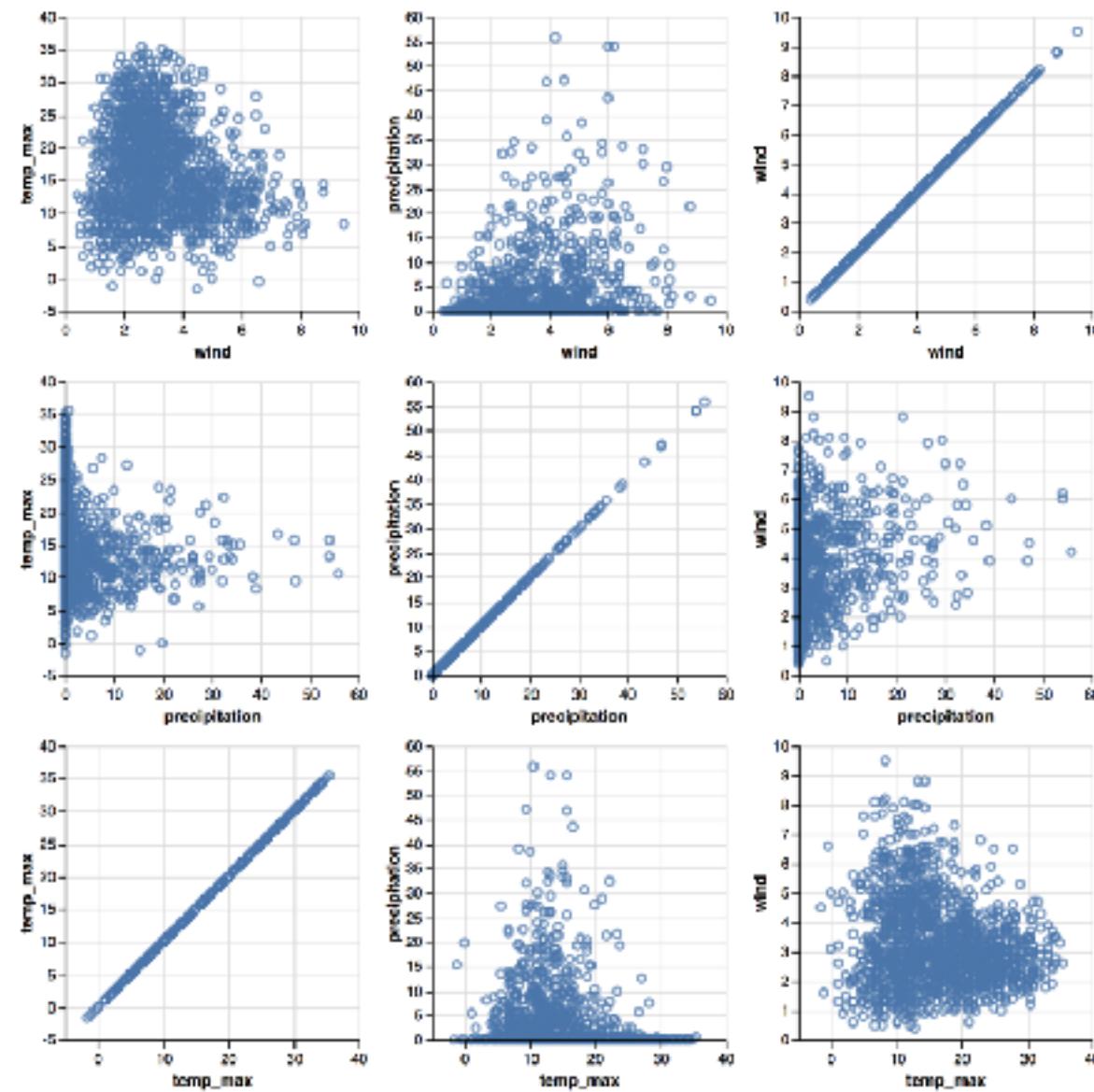


Area Chart

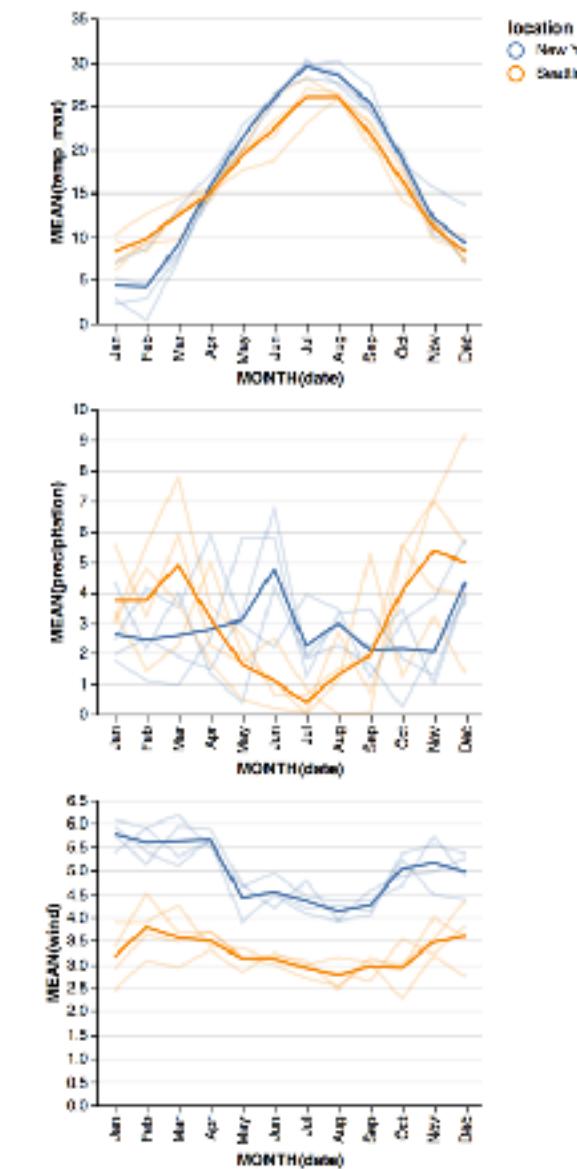


Vega-Lite: a Grammar of Multi-View Graphics

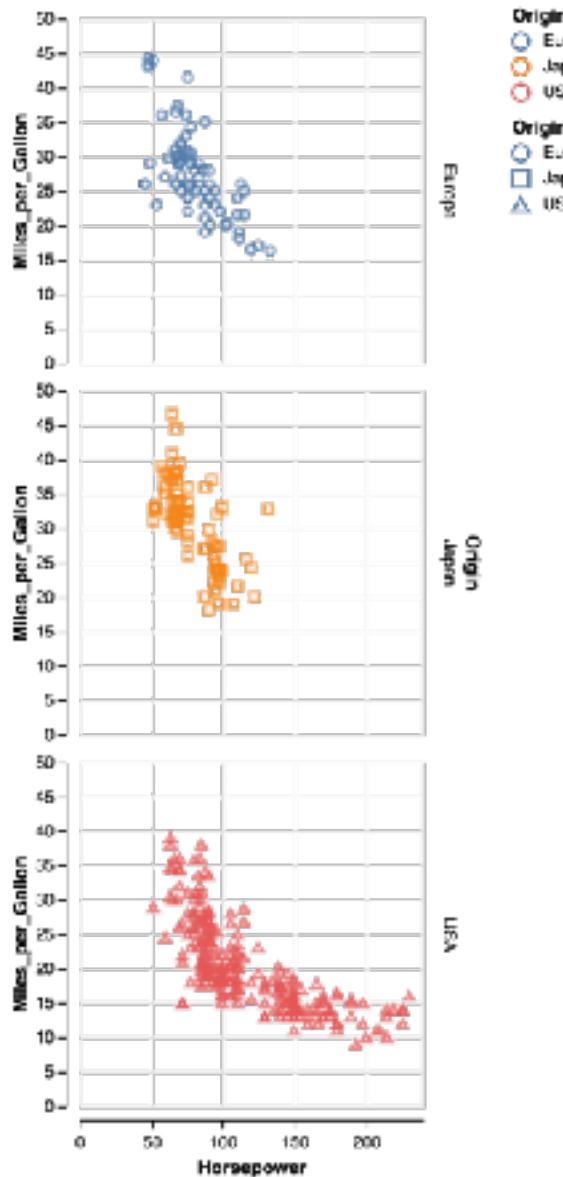
Scatterplot Matrix



Concatenated & Layered View

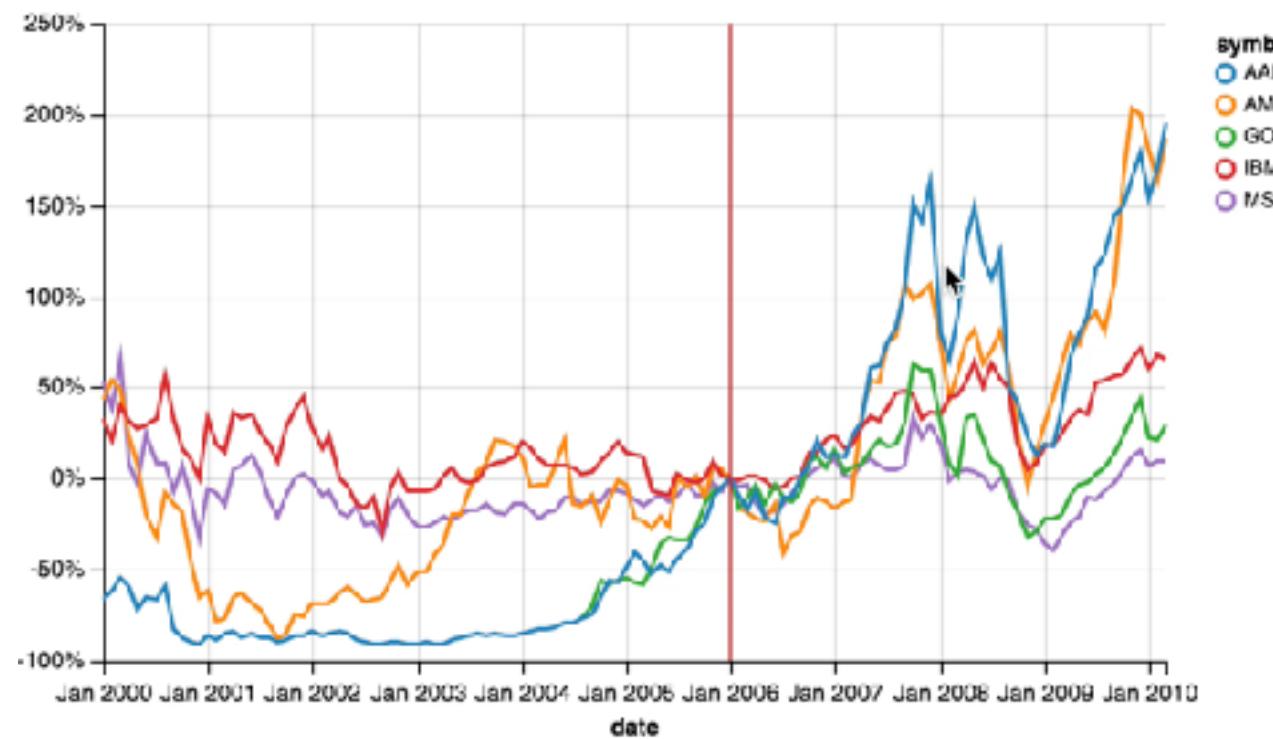


Faceted View

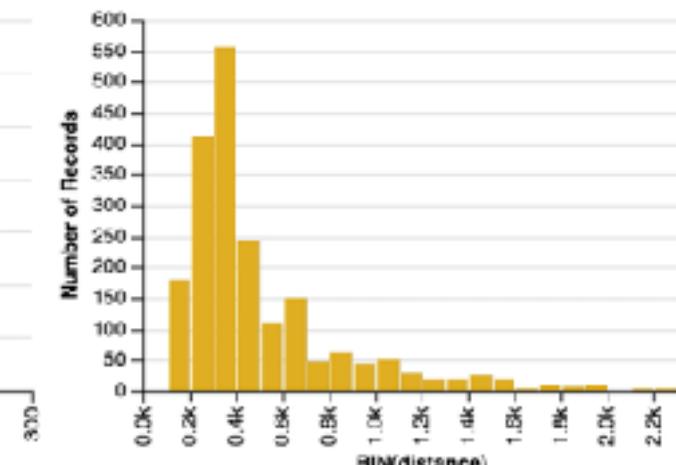
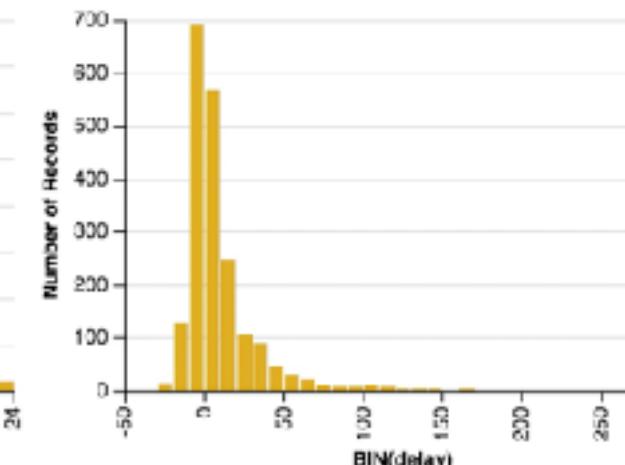
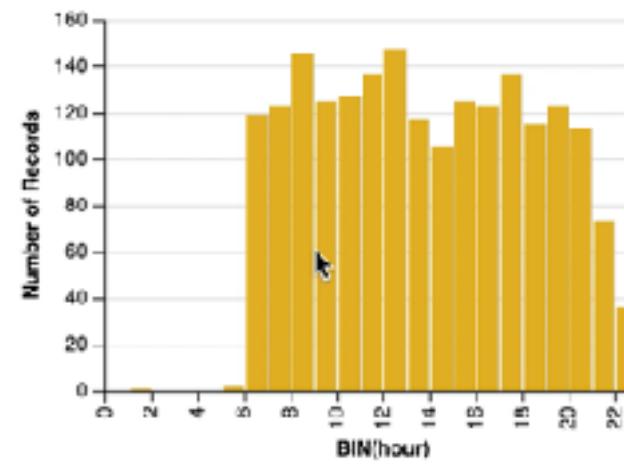
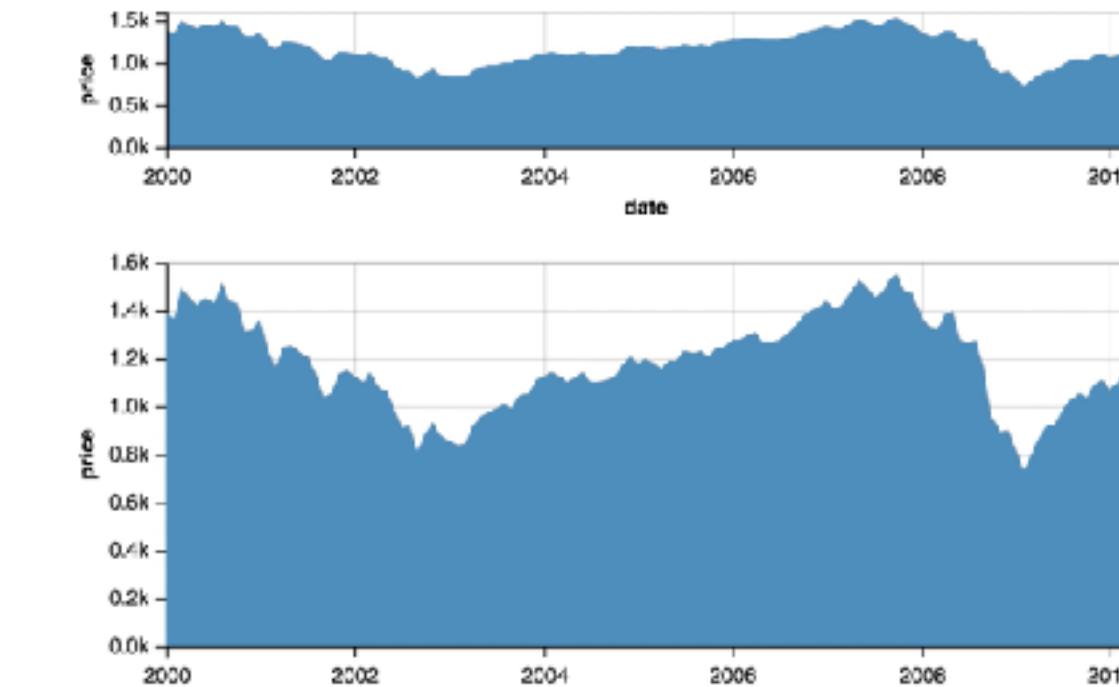


Vega-Lite: a Grammar of **Interactive** Multi-View Graphics

Indexed Chart



Focus+Context



Cross-filtering

Vega-Lite: a Grammar of Interactive Graphics

The Design of Vega-Lite

Single View Specification

Layered and **Multi-view** Composition

Interactions with Selections

Using Vega-Lite

Programming with Vega-Lite

Higher-level Tools and **Recommendations**

Vega-Lite: a Grammar of Interactive Graphics

The Design of Vega-Lite

Single View Specification

Layered and **Multi-view** Composition

Interactions with Selections

Using Vega-Lite

Programming with Vega-Lite

Higher-level Tools and Recommendations

Specifying Single Views

Abstract Data



Visual Representation

Specifying Single Views

Abstract Data

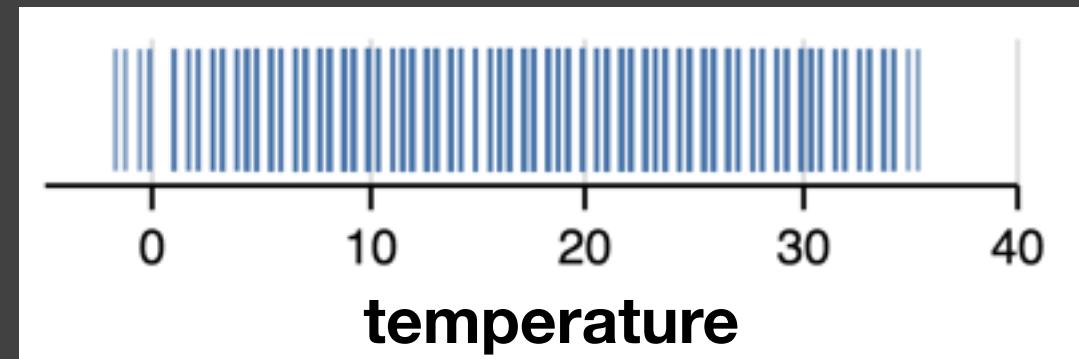


Visual Representation

Weather Data for Seattle

date	temperature	precipitation	weather
1/1	10.6	10.9	"rain"
1/2	11.7	0.8	"drizzle"
1/3	12.2	10.2	"rain"
...

Strip Plot of Temperature

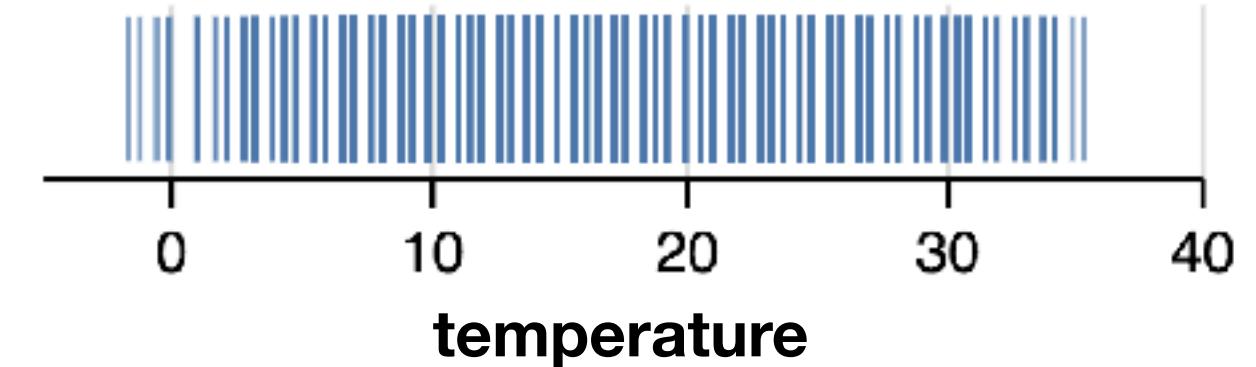


Strip Plot = (Tick with $x=$ field)

Tick Mark



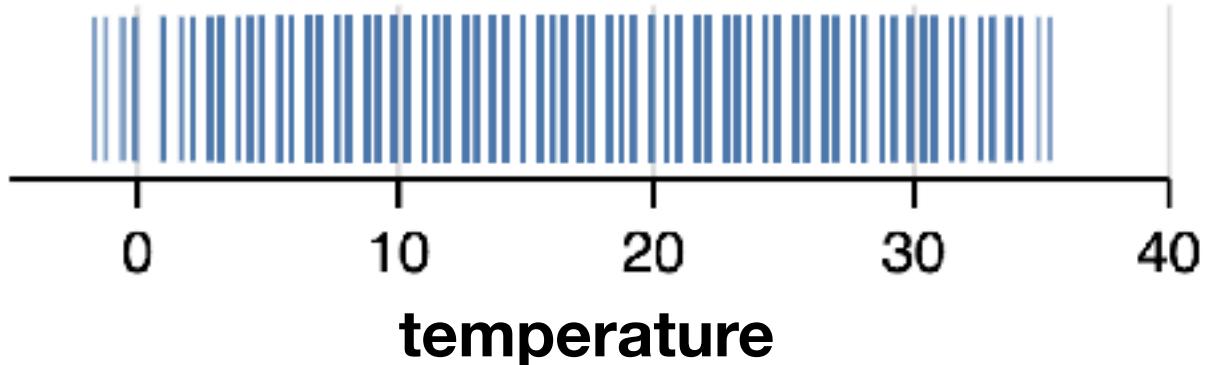
```
{  
  data: {url: "weather-seattle.json"},  
  mark: "tick",  
  encoding: {  
    x: {  
      field: "temperature",  
      type: "quantitative"  
    }  
  }  
}
```



Vega-Lite is portable JSON specification

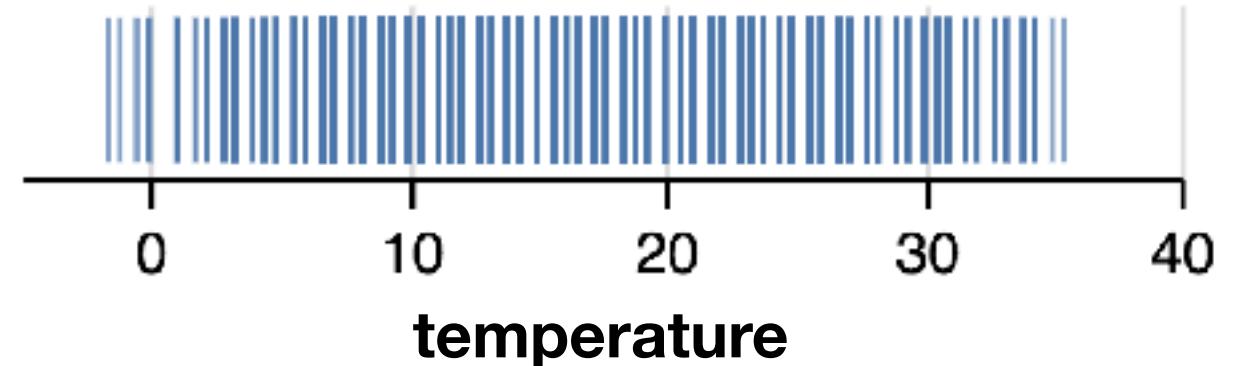
Strip Plot = (**Tick** with **x=field**)

```
{
  data: {url: "weather-seattle.json"},
  mark: "tick",
  encoding: {
    x: {
      field: "temperature",
      type: "quantitative"
    }
  }
}
```



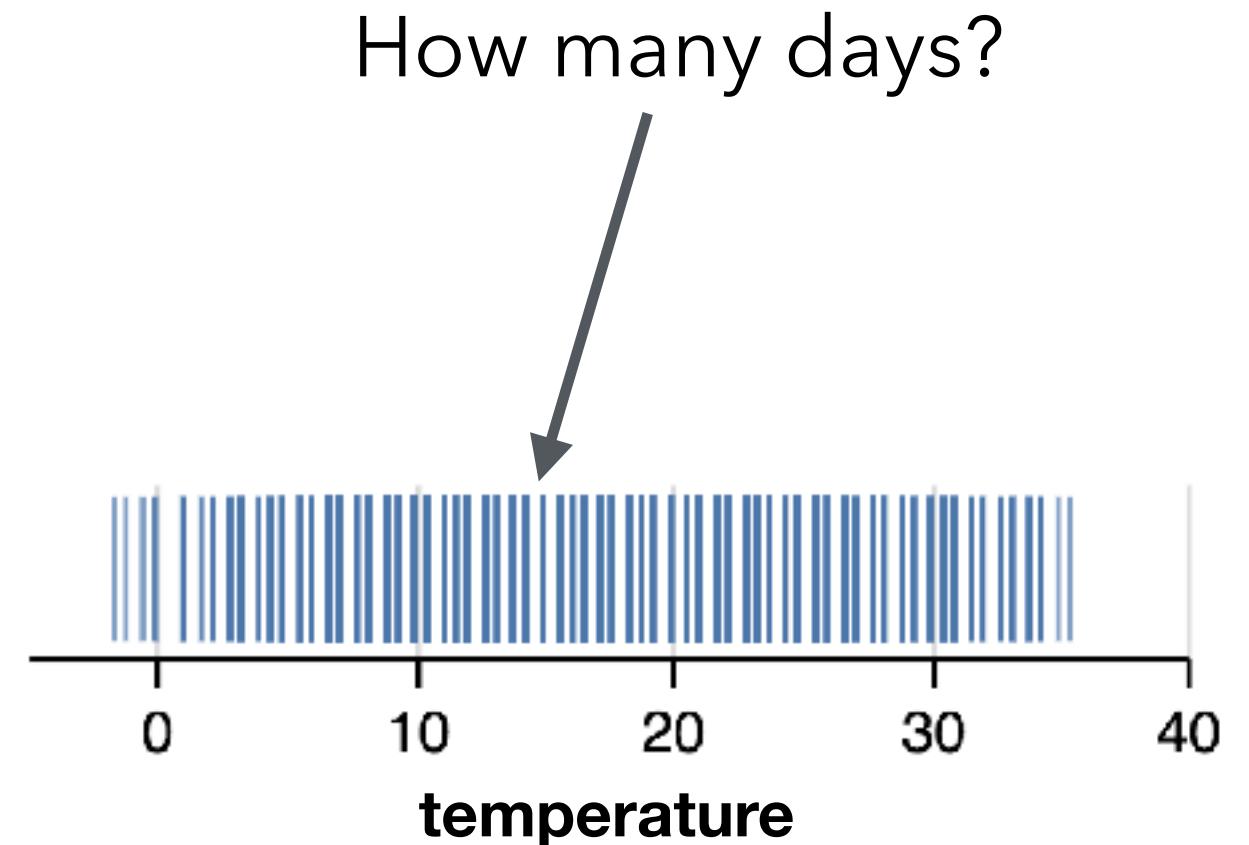
Strip Plot: Default Scales and Axes

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "tick",  
  encoding: {  
    x: {  
      field: "temperature",  
      type: "quantitative",  
      scale: {type: "linear", domain: [0, 8], ...}  
      axis: {title: "temp", grid: true, ...}  
    }  
  }  
}
```



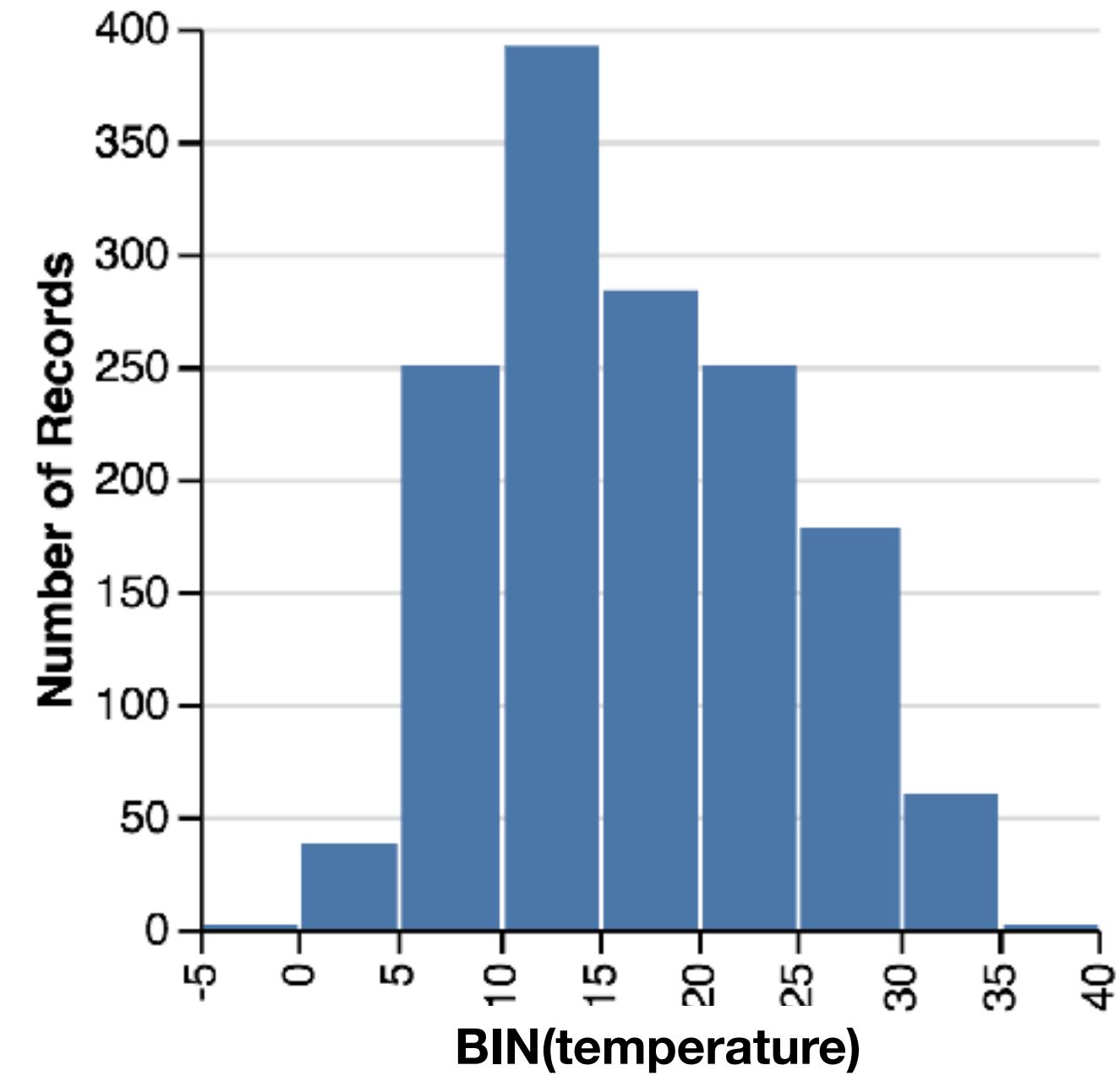
Strip Plot

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "tick",  
  encoding: {  
    x: {  
      field: "temperature",  
      type: "quantitative"  
    }  
  }  
}
```



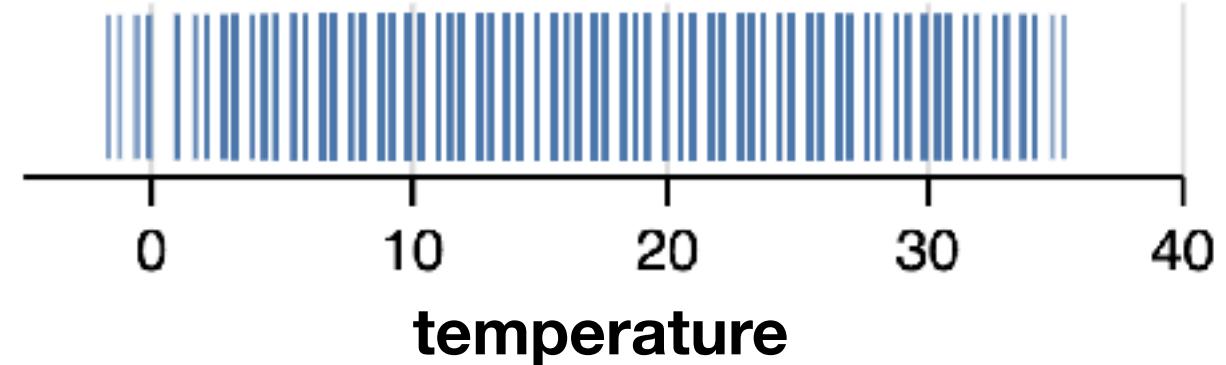
Histogram = (**Bar** with **x=binned field**, **y=count**)

Goal



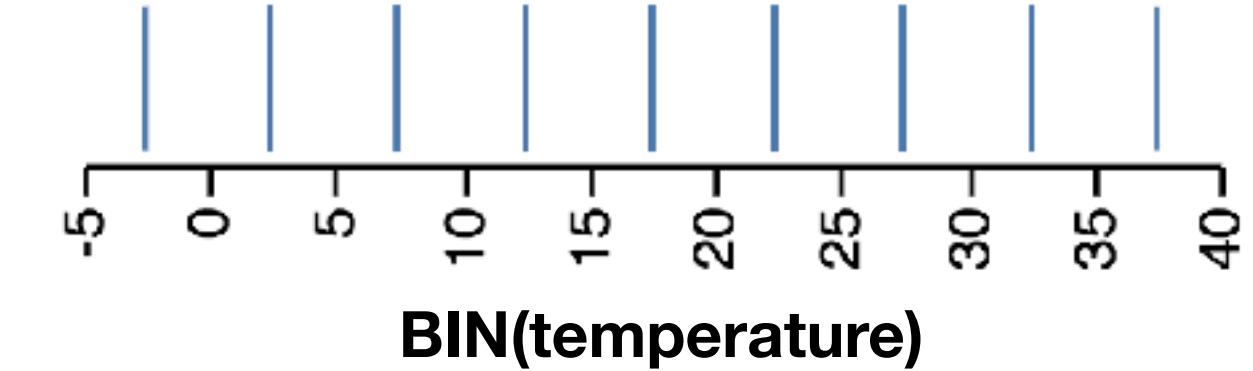
Histogram = (Bar with x=binned field, y=count)

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "tick",  
  encoding: {  
    x: {  
      field: "temperature",  
      type: "quantitative"  
    }  
  }  
}
```



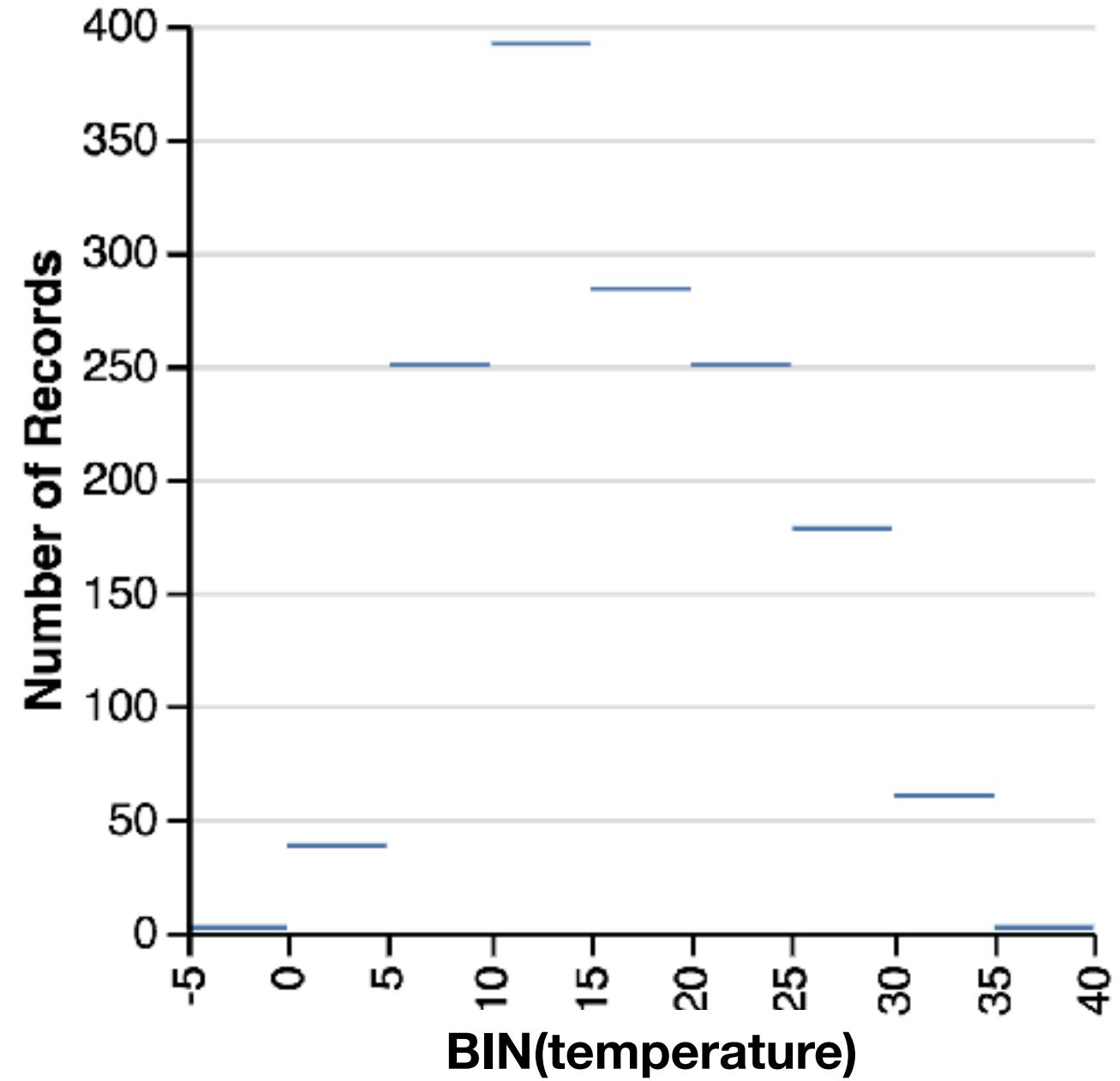
Histogram = (Bar with x=binned field, y=count)

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "tick",  
  encoding: {  
    x: {  
      bin: true,  
      field: "temperature",  
      type: "quantitative"  
    }  
  }  
}
```



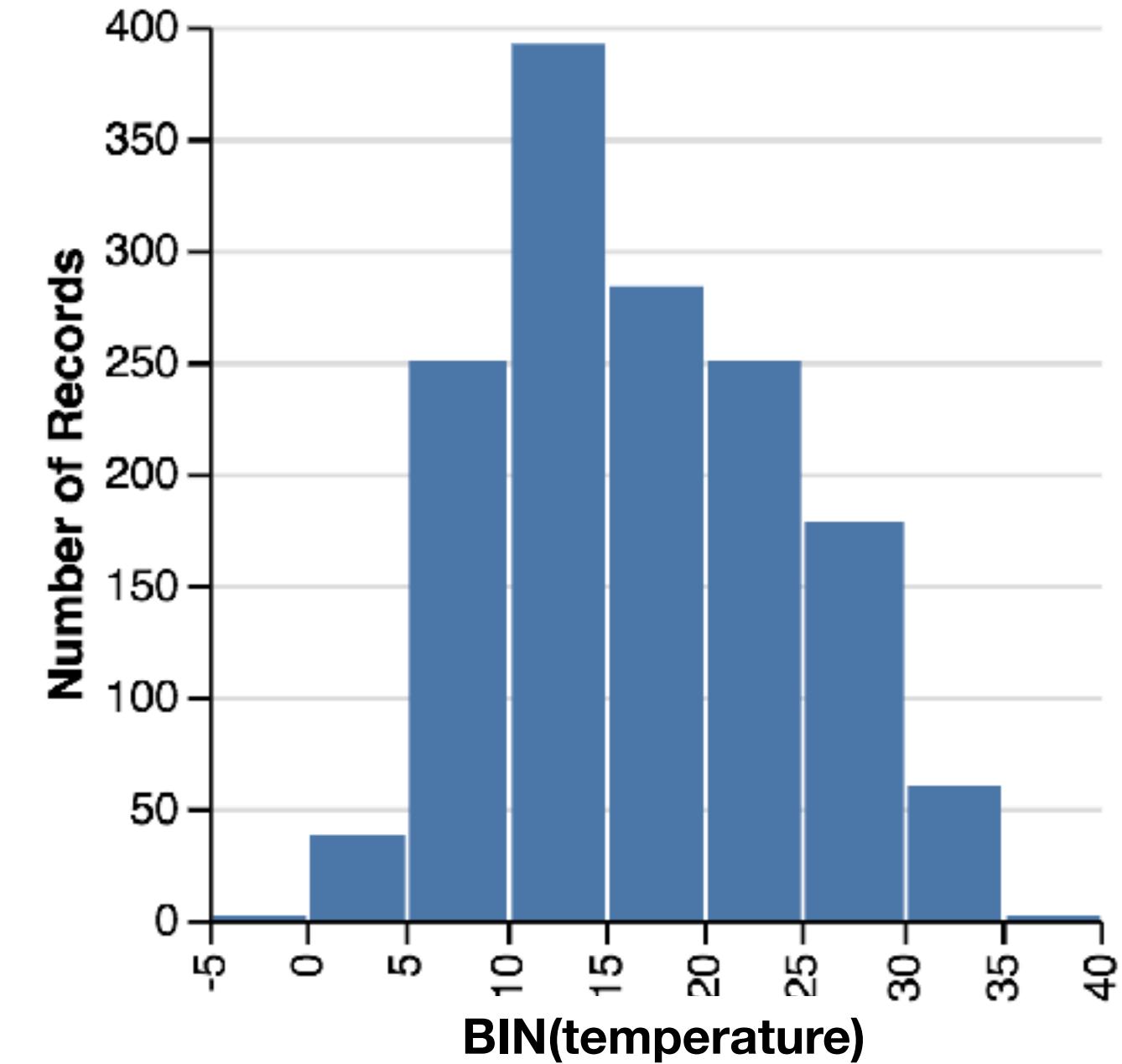
Histogram = (Bar with $x=binned$ field, $y=count$)

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "tick",  
  encoding: {  
    x: {  
      bin: true,  
      field: "temperature",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "count",  
      type: "quantitative"  
    }  
  }  
}
```



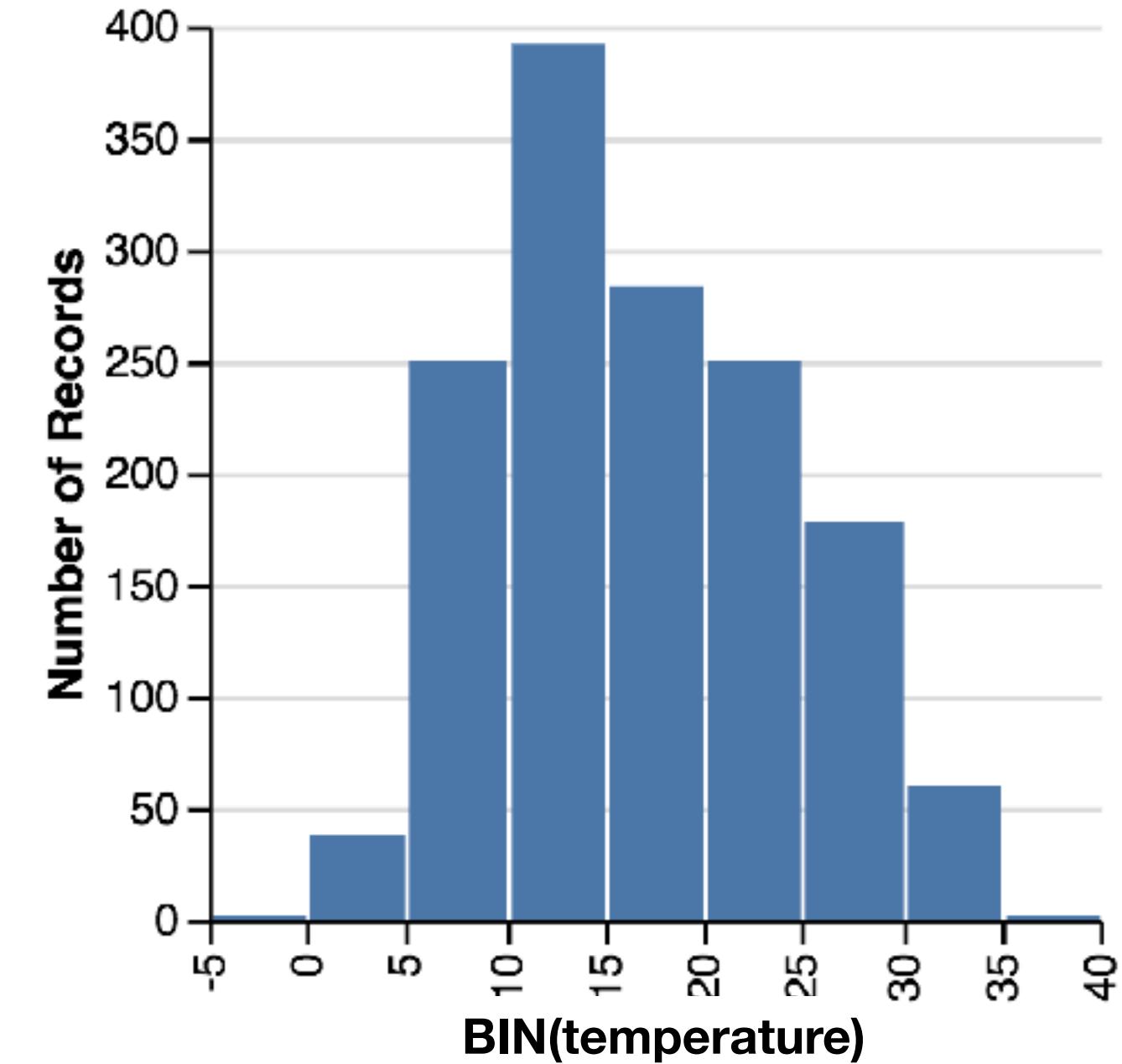
Histogram = (**Bar** with **x=binned field**, **y=count**)

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      bin: true,  
      field: "temperature",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "count",  
      type: "quantitative"  
    }  
  }  
}
```



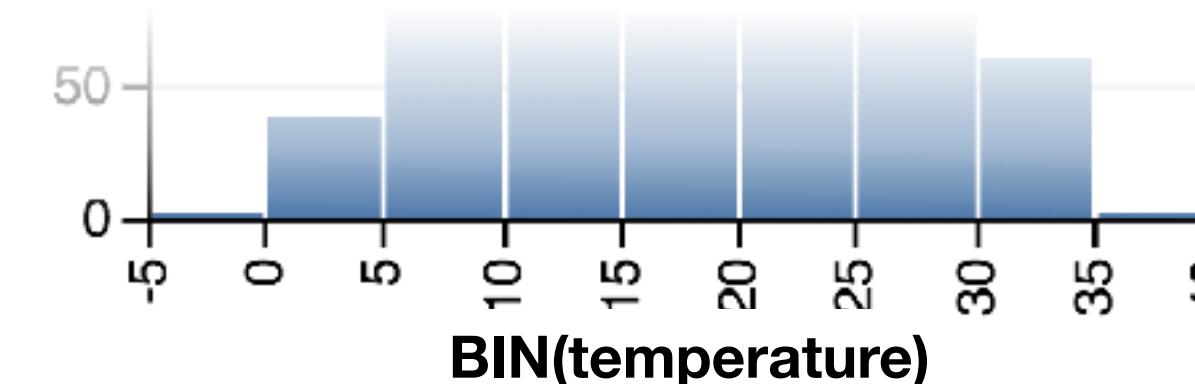
Histogram = (Bar with x=binned field, y=count)

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      bin: true,  
      field: "temperature",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "count",  
      type: "quantitative"  
    }  
  }  
}
```



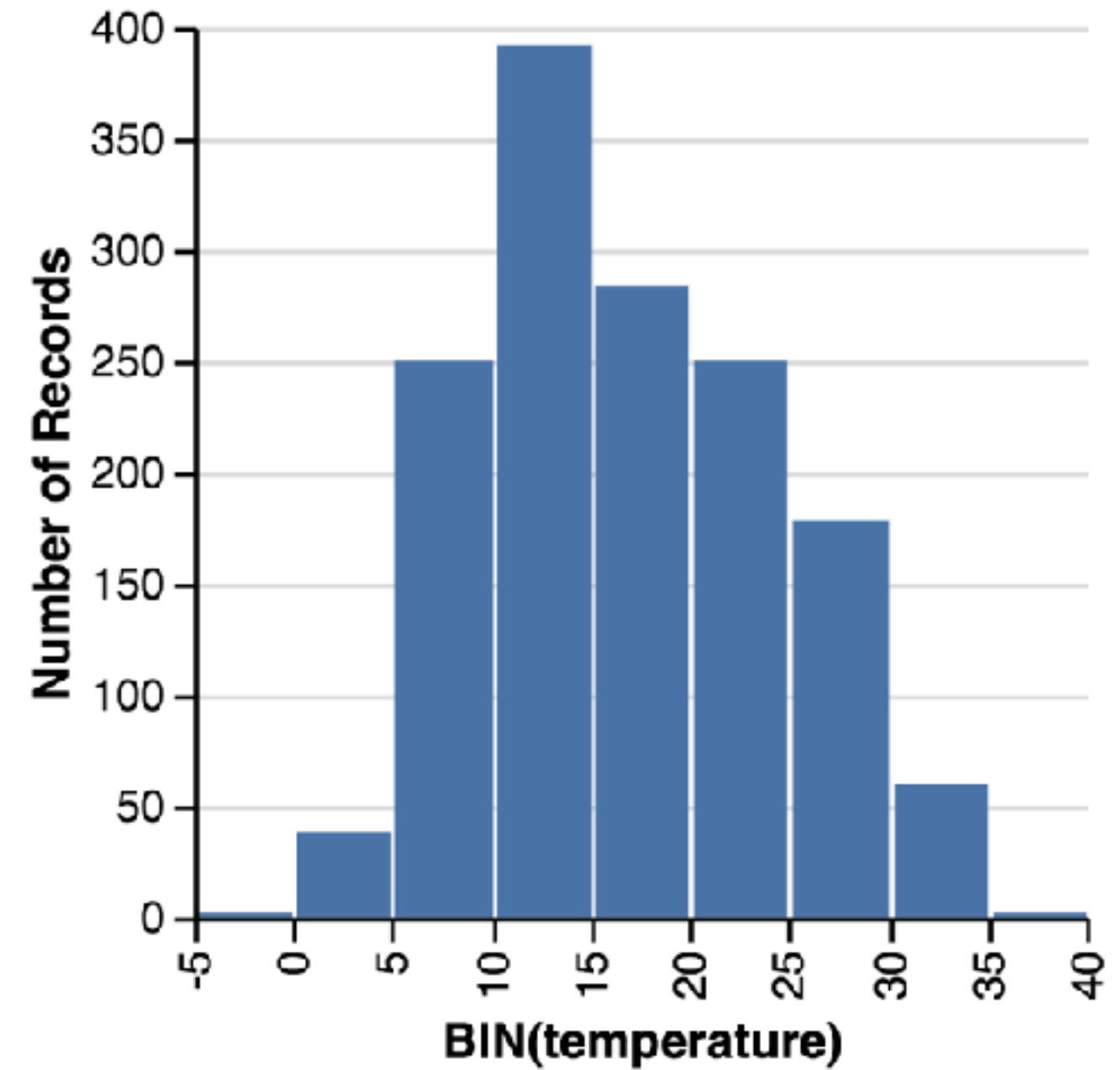
Sensible Defaults for Binning

Channel determines guide and bin parameters

	Color/Opacity/Shape	Position
Guide	Legend with range labels	Quantitative axis
# of Bins	Fewer bins	More bins
	<p>Hottest Temperature</p> <ul style="list-style-type: none">-10–400–1010–2020–3030–40	

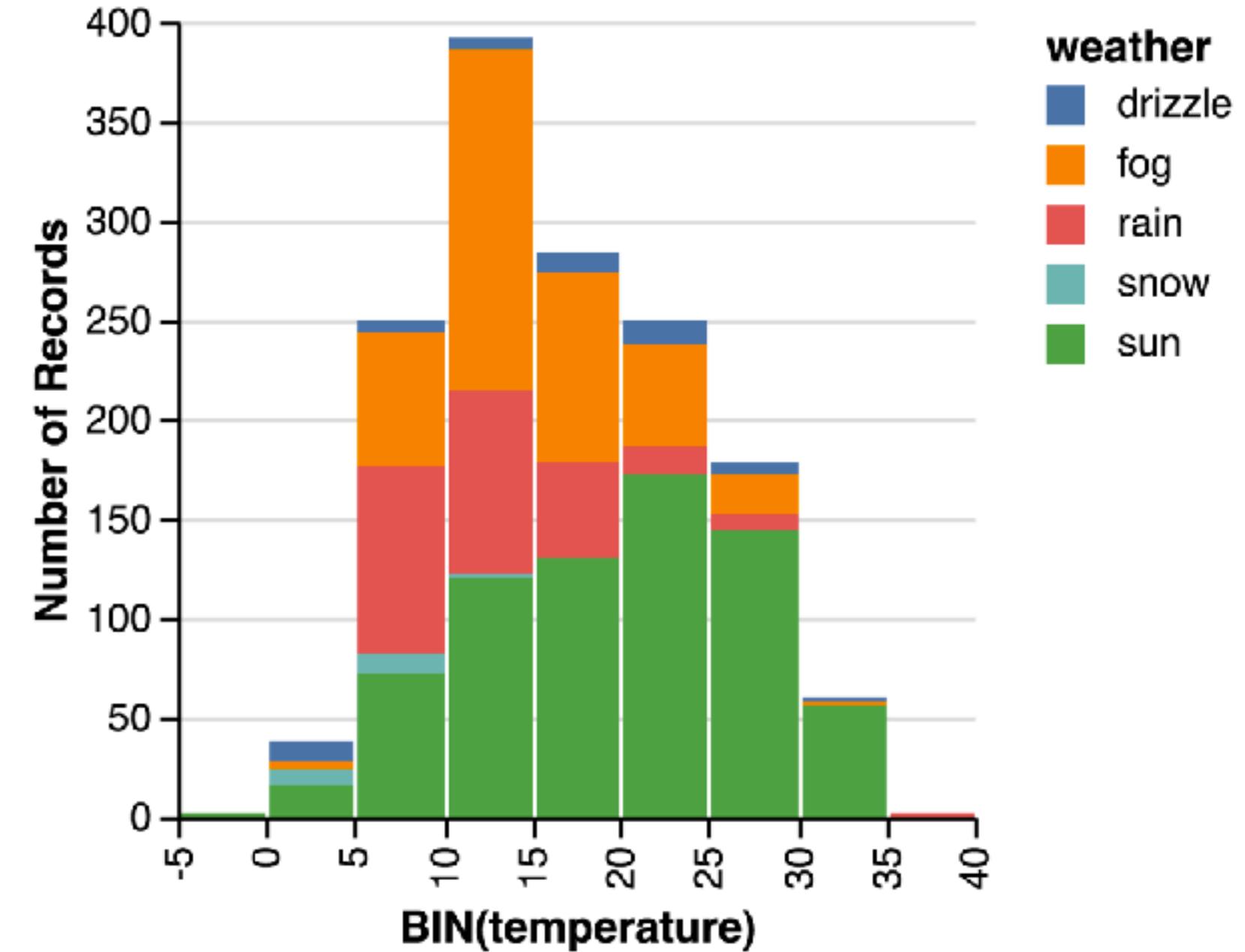
Histogram

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      bin: true,  
      field: "temperature",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "count",  
      type: "quantitative"  
    }  
  }  
}
```



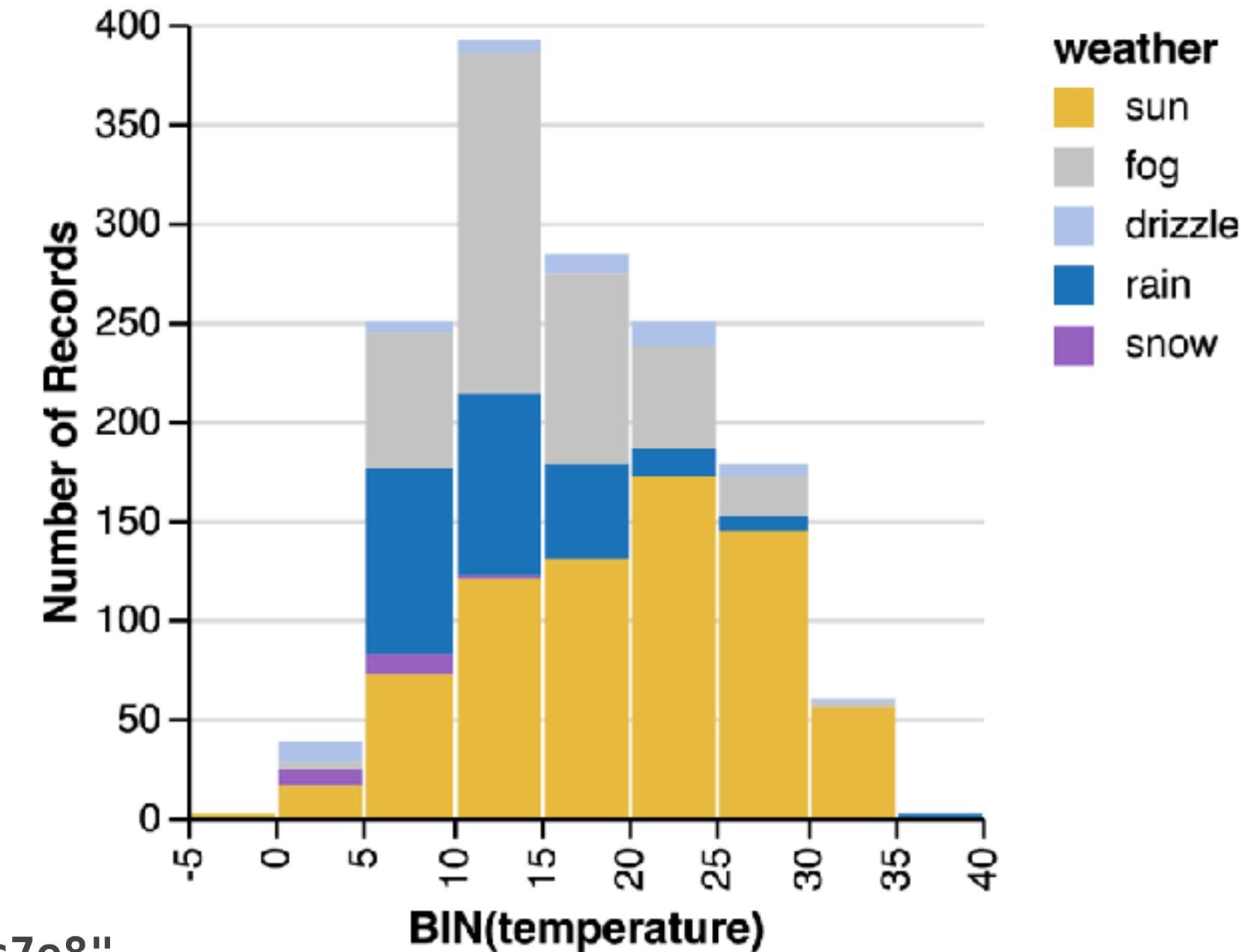
Histogram + Color

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      bin: true,  
      field: "temperature",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "count",  
      type: "quantitative"  
    },  
    color: {  
      field: "weather",  
      type: "nominal"  
    }  
  }  
}
```



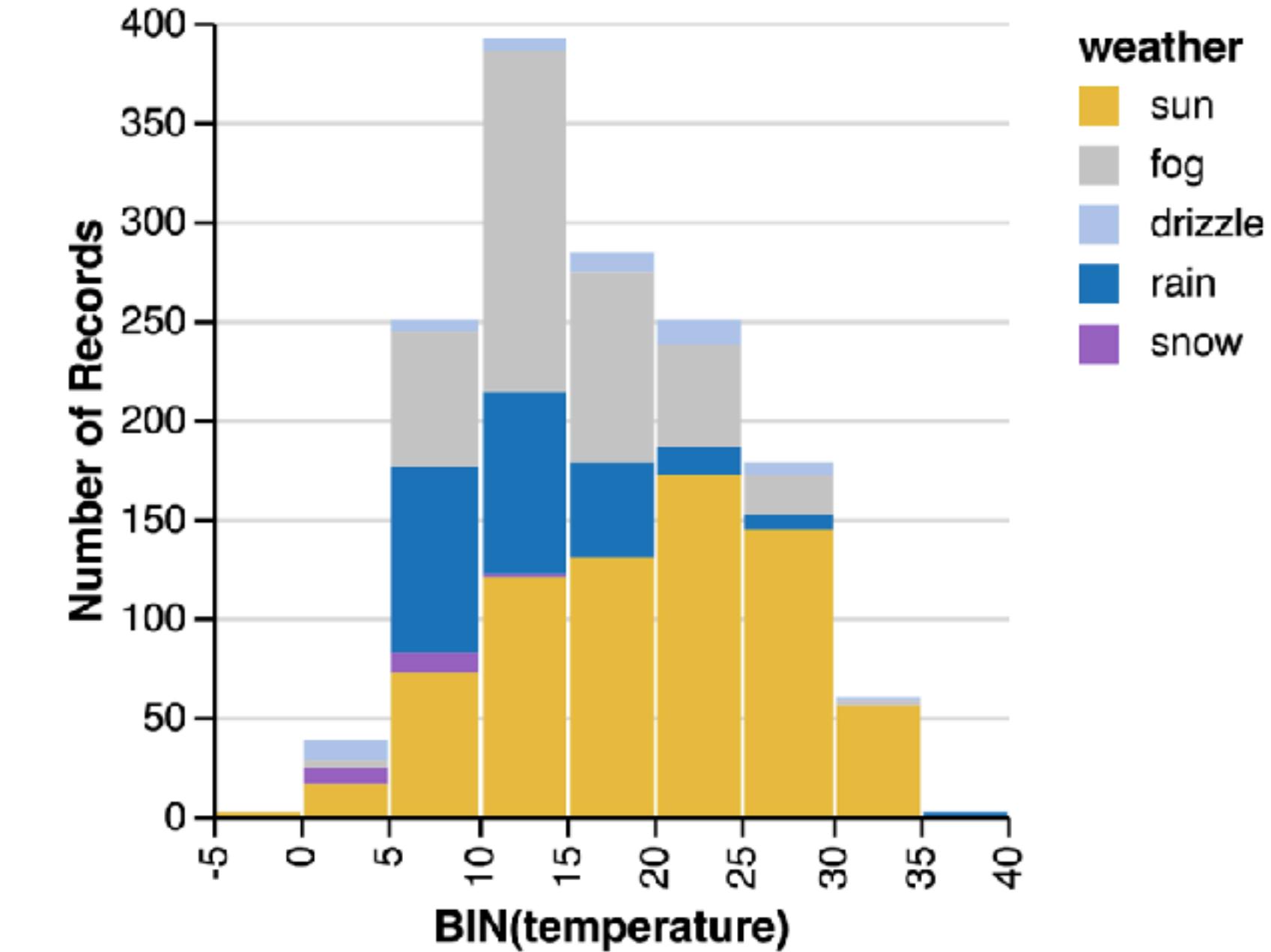
Histogram + Color

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      bin: true,  
      field: "temperature",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "count",  
      type: "quantitative"  
    },  
    color: {  
      field: "weather",  
      type: "nominal",  
      "scale": {  
        "domain": ["sun", "fog", "drizzle",  
                  "rain", "snow"],  
        "range": ["#e7ba52", "#c7c7c7", "#aec7e8",  
                  "#1f77b4", "#9467bd"]  
      }  
    }  
  }  
}
```



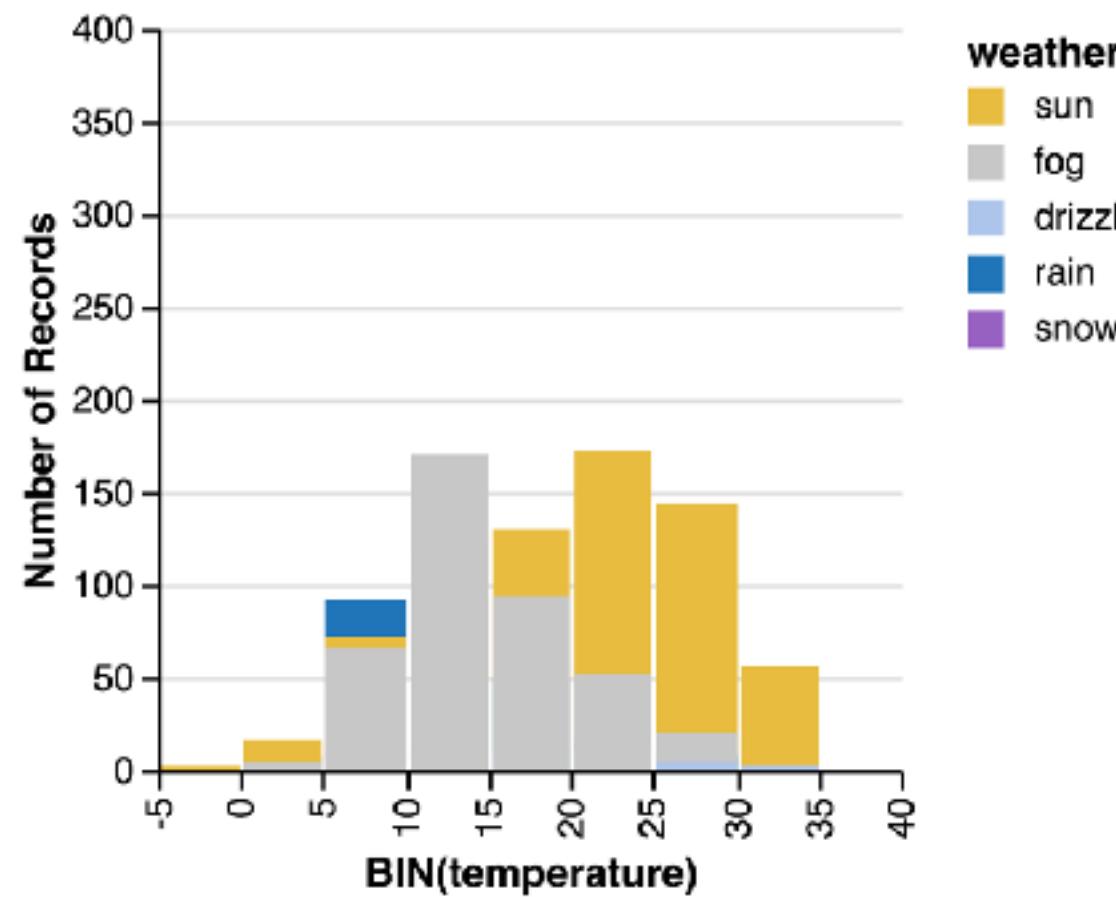
Histogram + Color = Stacked Histogram

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      bin: true,  
      field: "temperature",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "count",  
      type: "quantitative"  
    },  
    color: {  
      field: "weather",  
      type: "nominal",  
      ...  
    }  
  }  
}
```

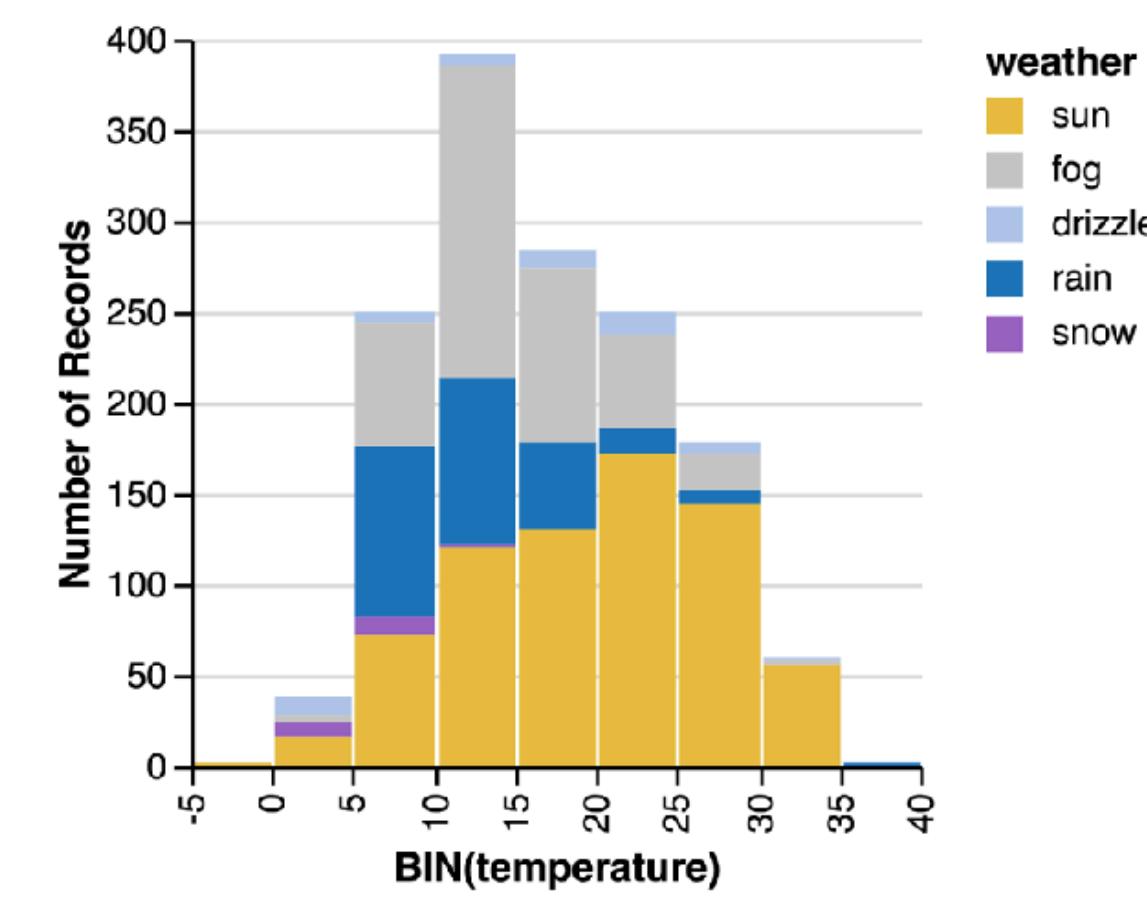


Stacked Histogram: Sensible Defaults

no stack



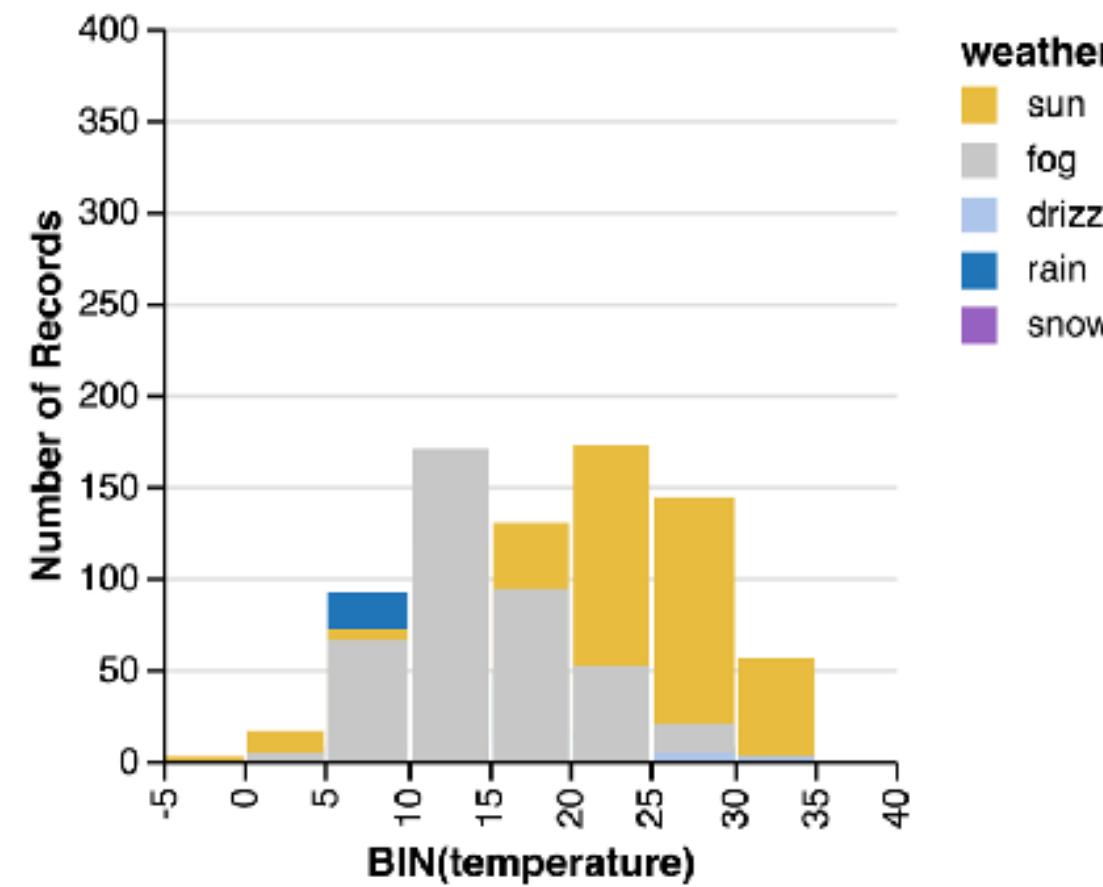
stack (default)



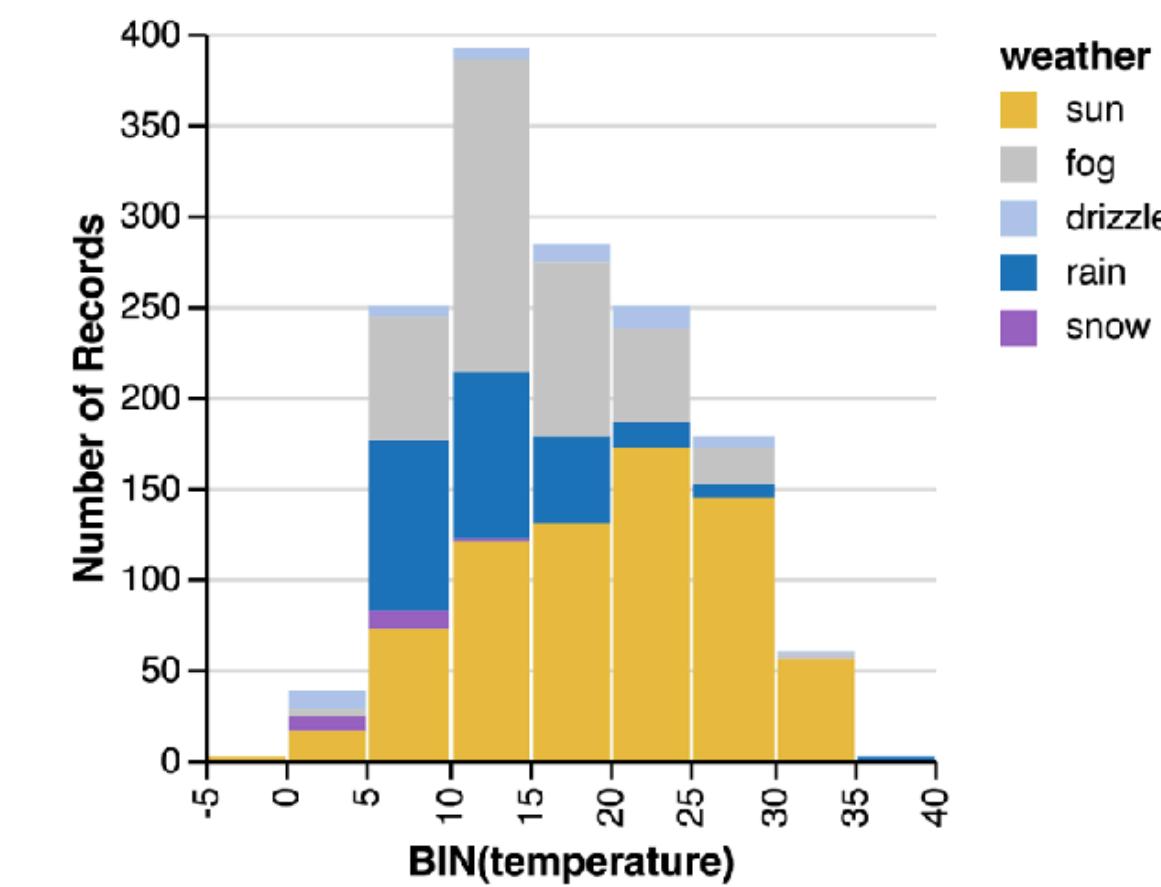
Stacked Histogram: Sensible Defaults

Channel (color) + Mark (bar) automatically enables stacking: a layout transform.

no stack



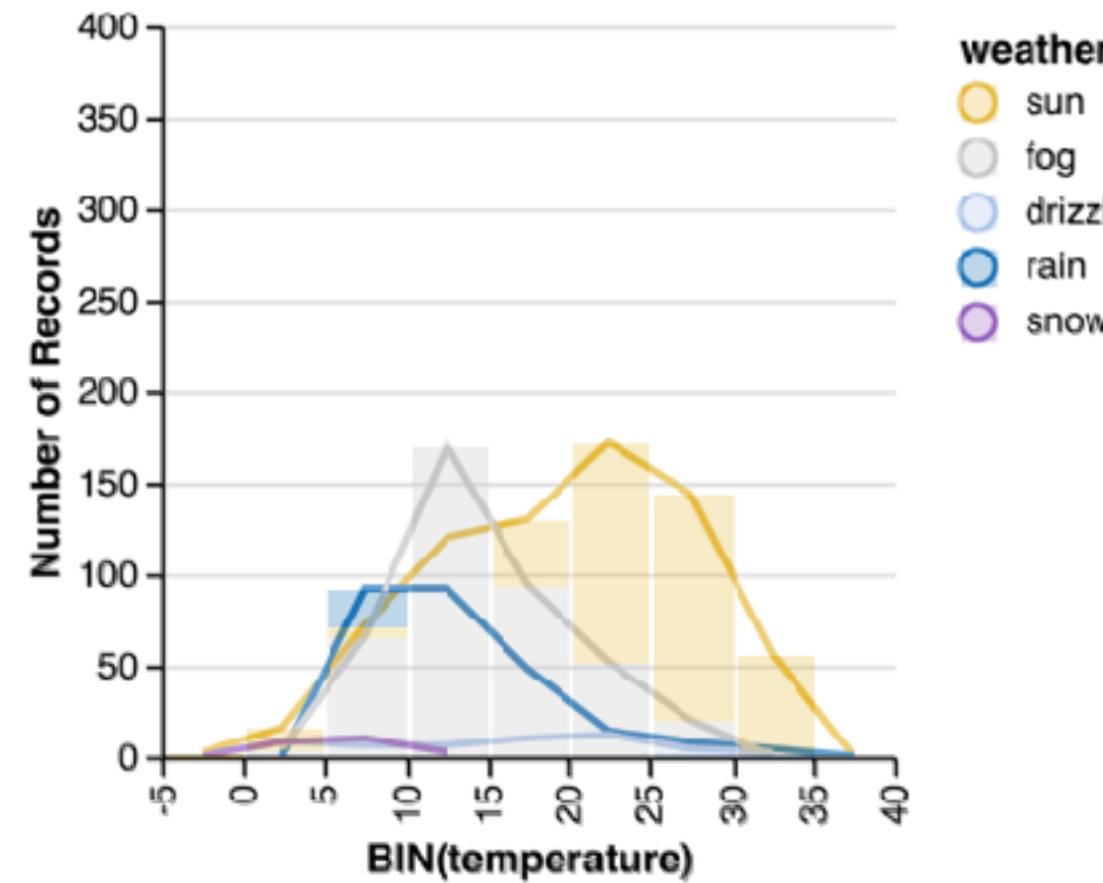
stack (default)



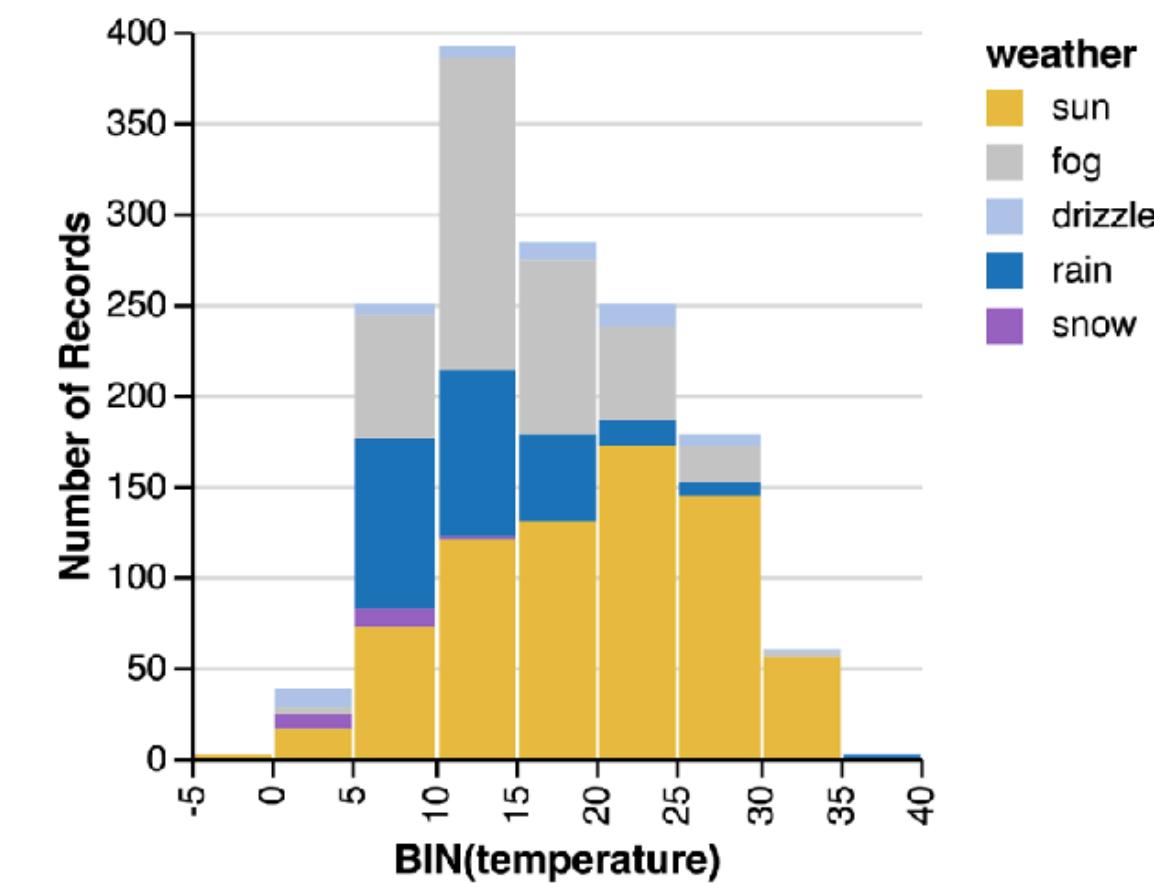
Stacked Histogram: Sensible Defaults

Channel (color) + Mark (bar) automatically enables stacking: a layout transform.

no stack → overlap

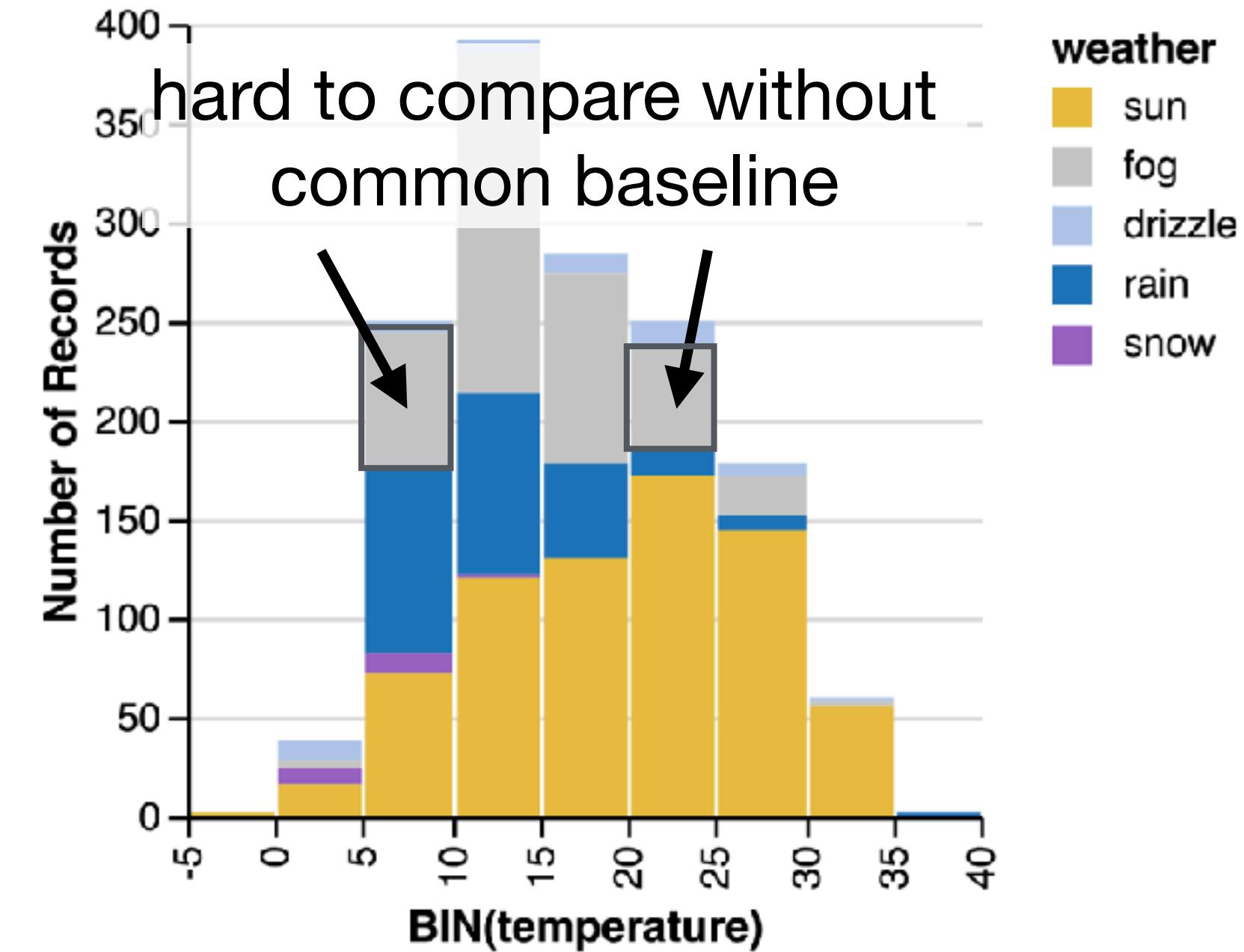


stack (default)



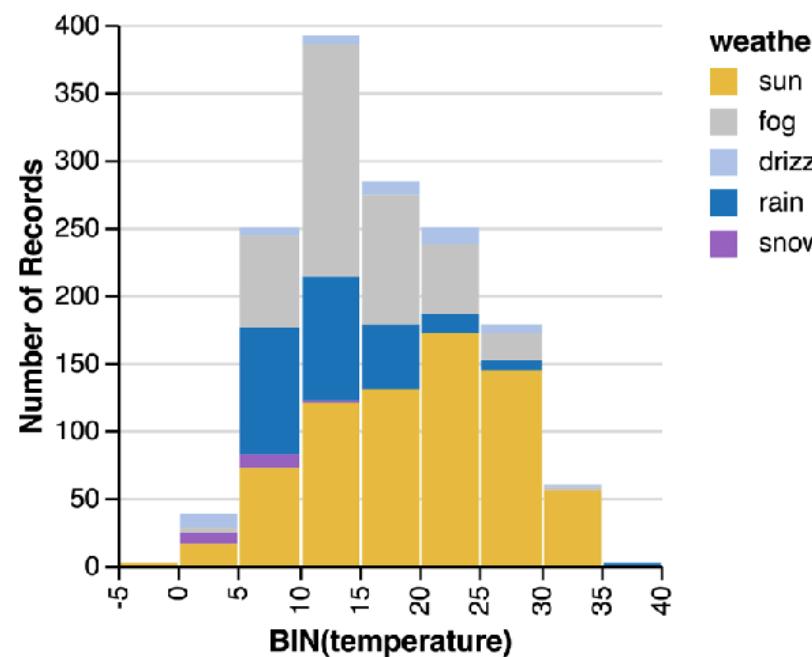
Histogram + Color = Stacked Histogram

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      bin: true,  
      field: "temperature",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "count",  
      type: "quantitative"  
    },  
    color: {  
      field: "weather",  
      type: "nominal"  
    }  
  }  
}
```



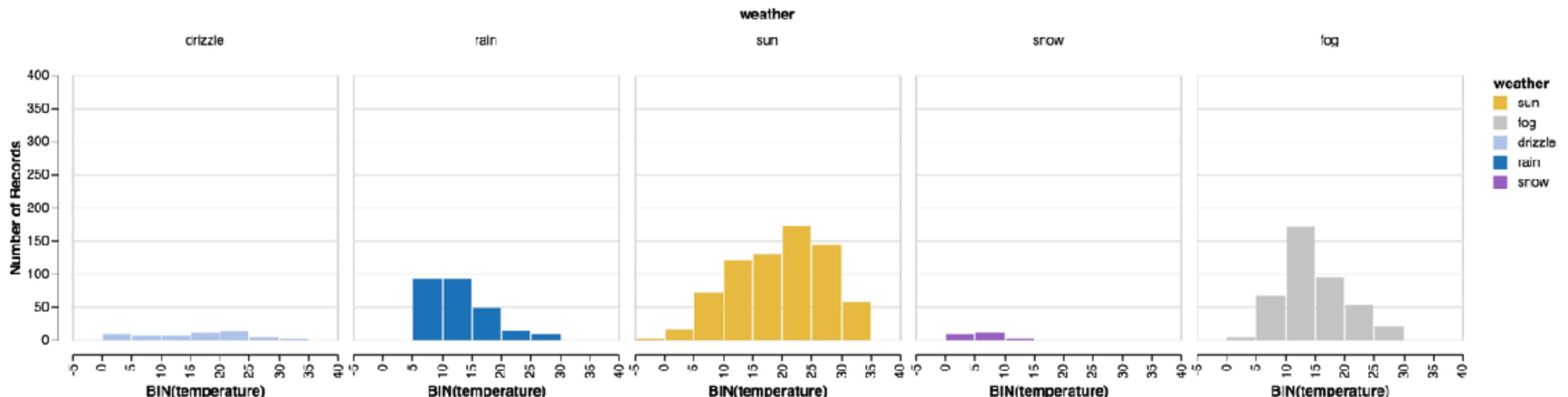
Histogram + Color = Stacked Histogram

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {bin: true, field: "temperature", type: "quantitative"},  
    y: {aggregate: "count", type: "quantitative"},  
    color: {field: "weather", type: "nominal"}  
}
```



Histogram + Column = Trellis Histogram

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {bin: true, field: "temperature", type: "quantitative"},  
    y: {aggregate: "count", type: "quantitative"},  
    column: {field: "weather", type: "nominal"}  
  }  
}
```



Vega-Lite: a Grammar of Interactive Graphics

The Design of Vega-Lite

Single View Specification

Layered and Multi-view Composition

Interactions with Selections

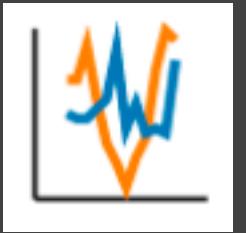
Using Vega-Lite

Programming with Vega-Lite

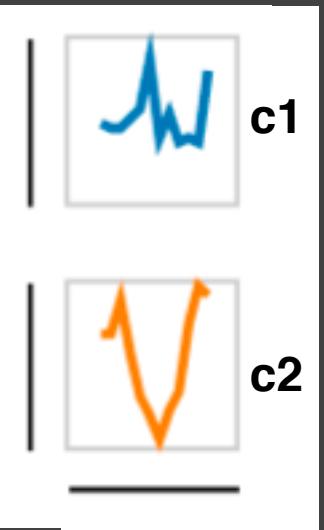
Higher-level Tools and Recommendations

View Composition Operators

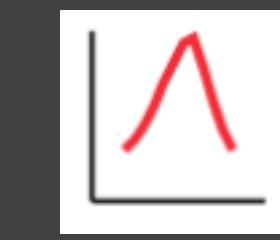
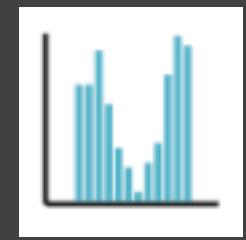
facet row: C



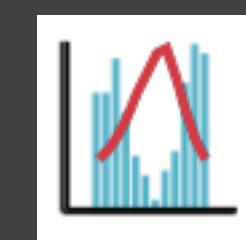
=



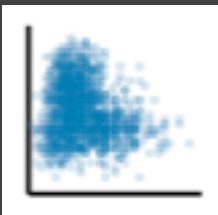
layer: [



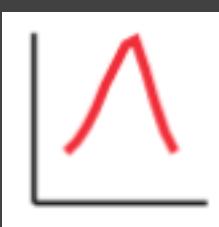
] =



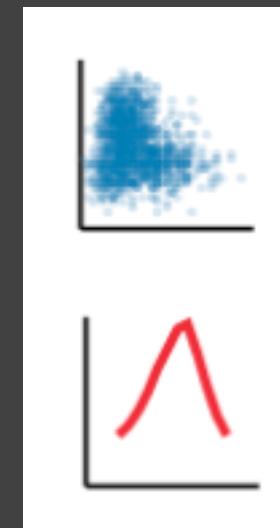
vconcat: [



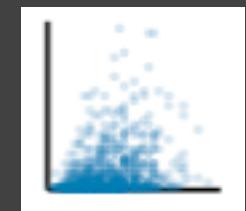
,



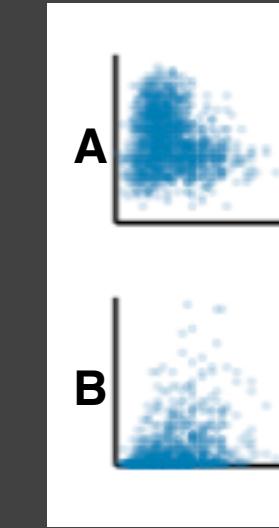
=



repeat row: [A,B]

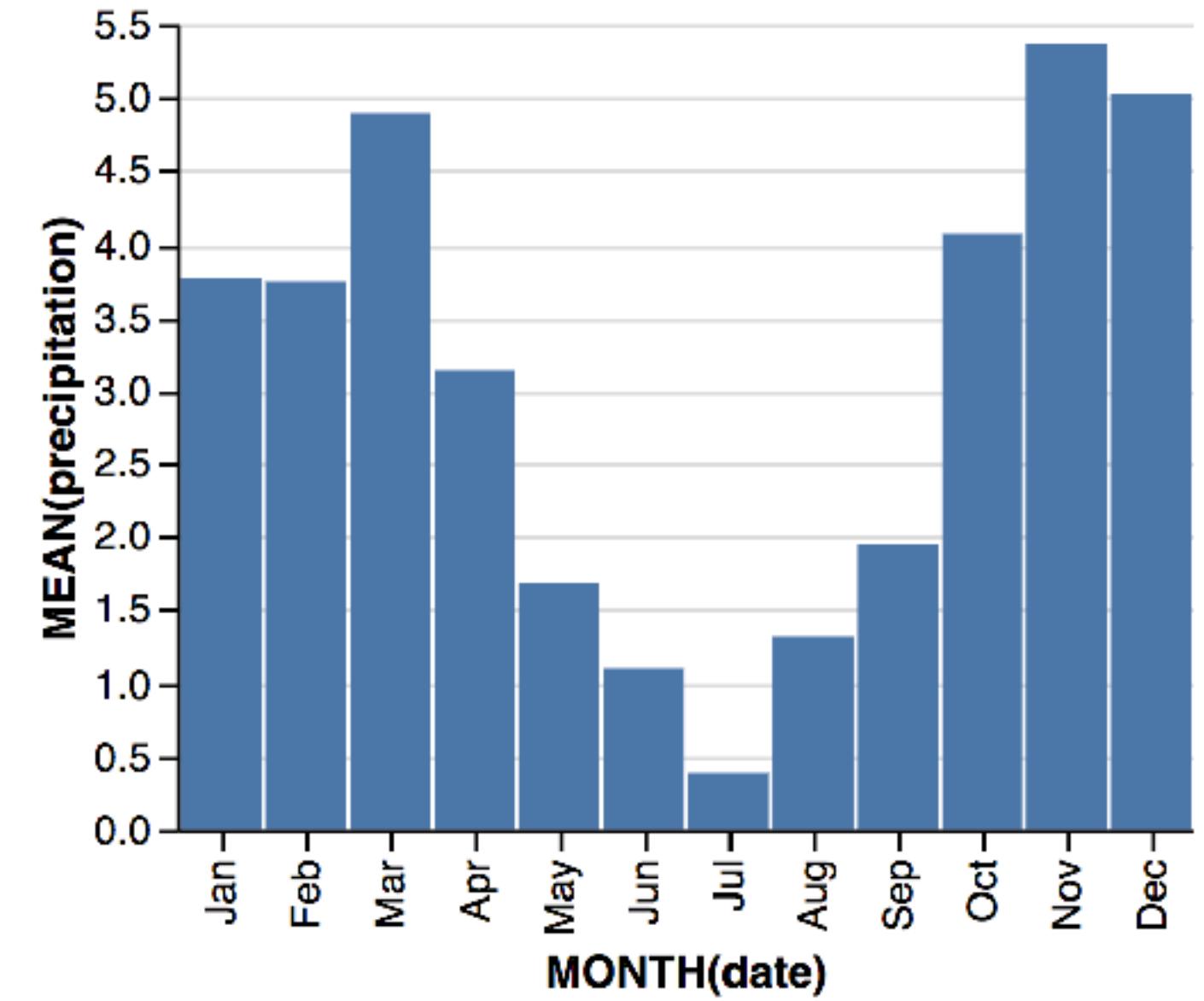


=



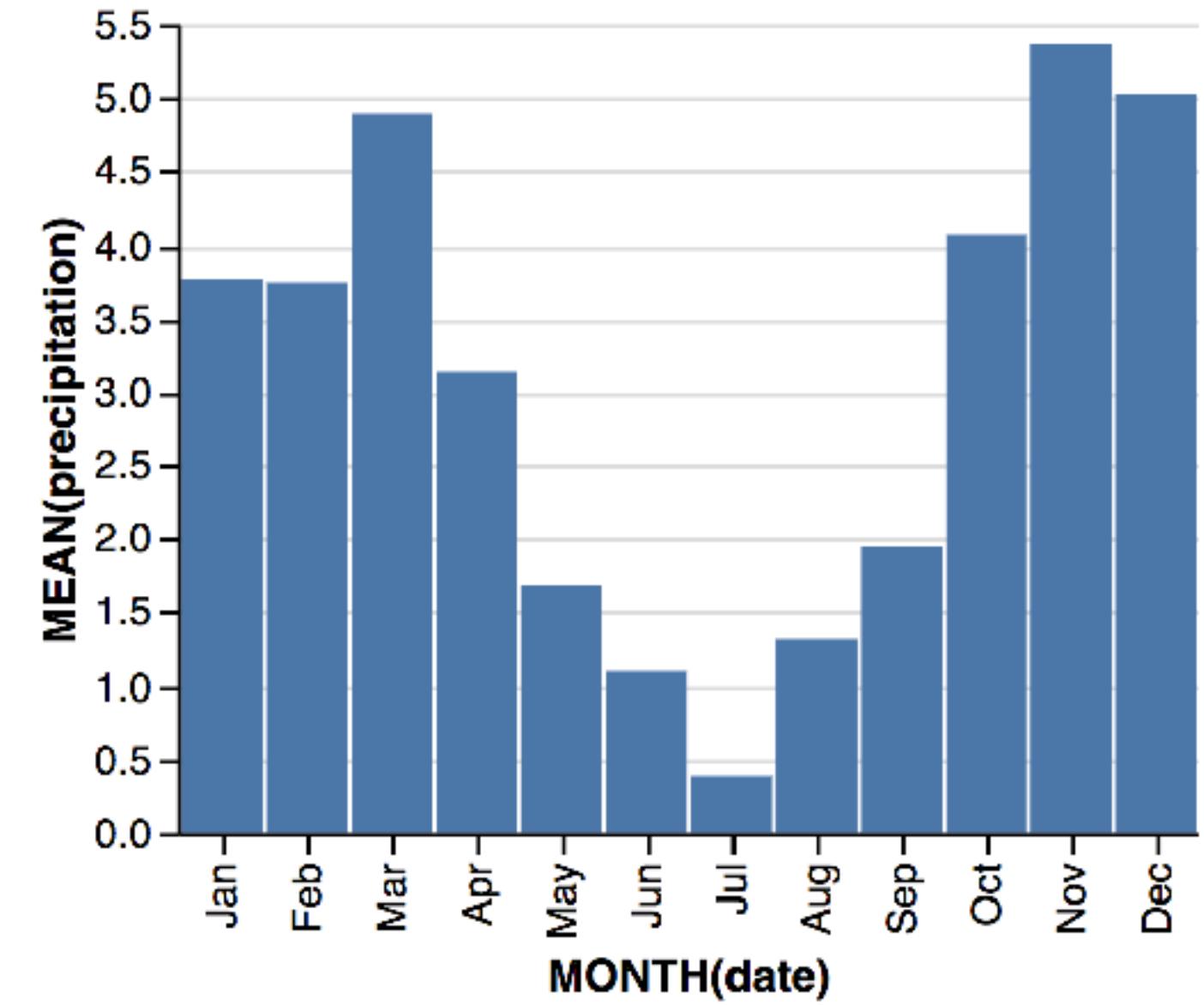
Monthly Precipitation

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      timeUnit: "month",  
      field: "date",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "mean",  
      field: "precipitation",  
      type: "quantitative"  
    }  
  }  
}
```



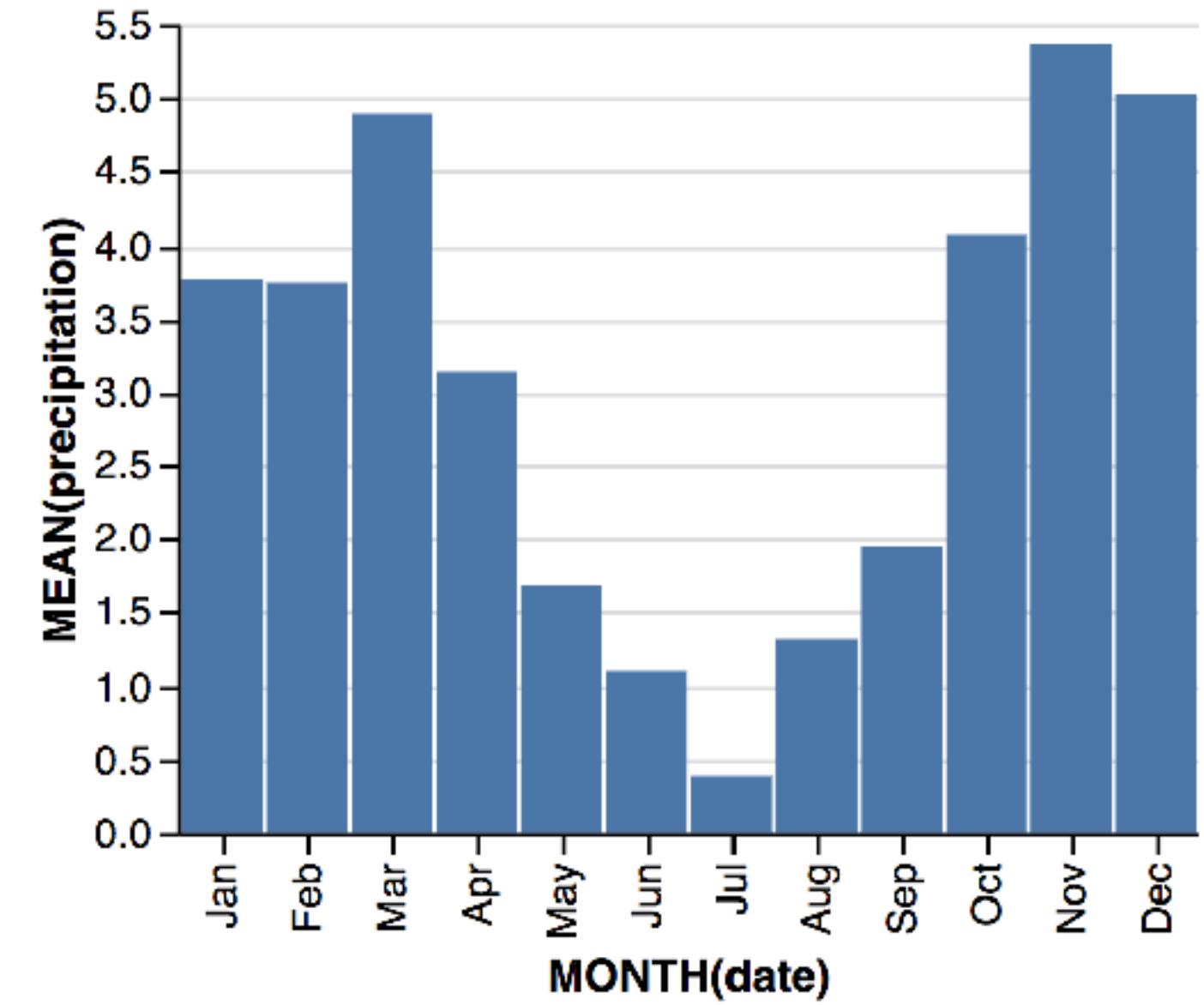
Monthly Precipitation

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      timeUnit: "month",  
      field: "date",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "mean",  
      field: "precipitation",  
      type: "quantitative"  
    }  
  }  
}
```



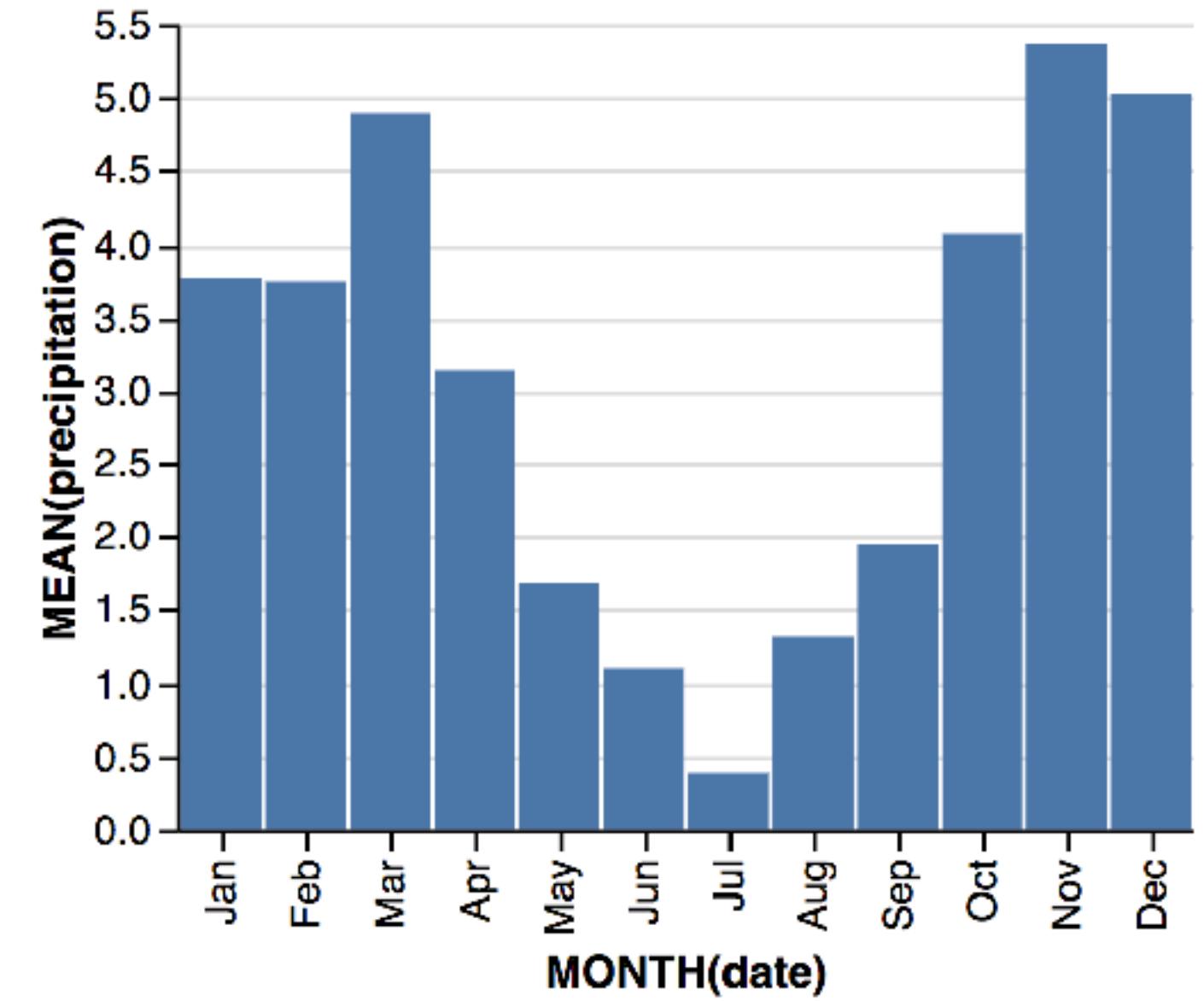
Monthly Precipitation

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      timeUnit: "month",  
      field: "date",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "mean",  
      field: "precipitation",  
      type: "quantitative"  
    }  
  }  
}
```



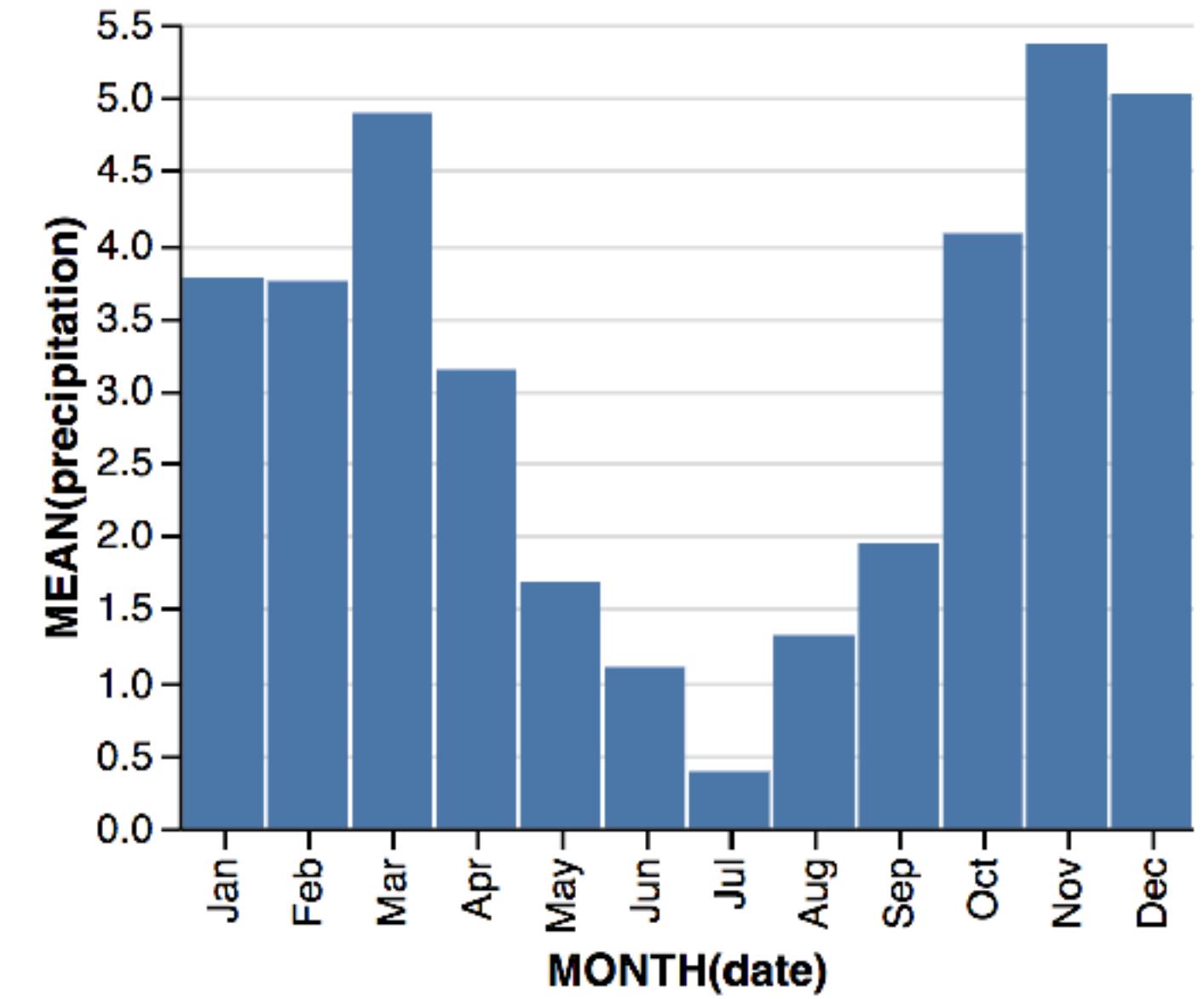
Layering

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      timeUnit: "month",  
      field: "date",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "mean",  
      field: "precipitation",  
      type: "quantitative"  
    }  
  }  
}
```



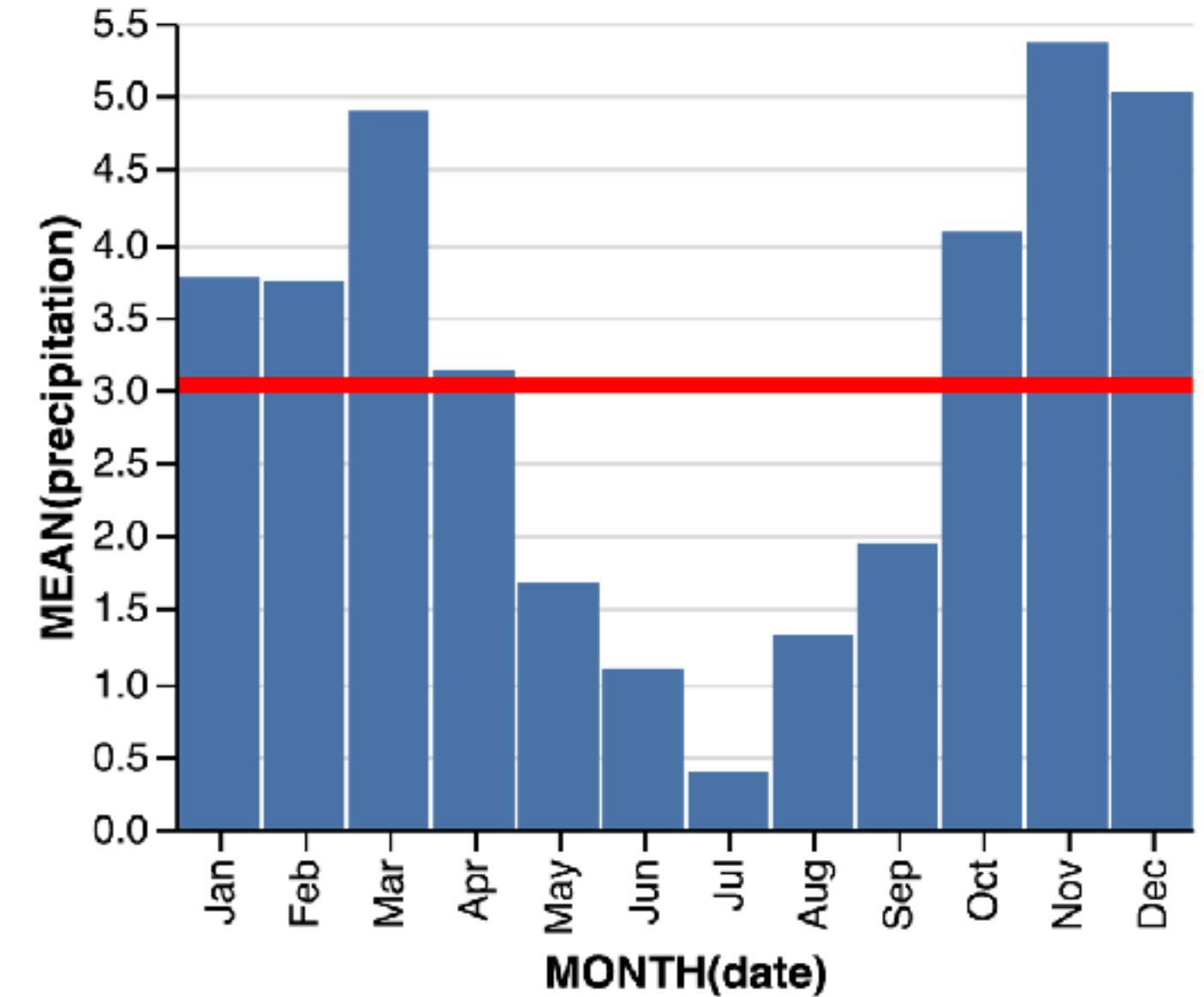
Layering

```
{  
  data: {url: "weather-seattle.json"},  
  layer: [{  
    mark: "bar",  
    encoding: {  
      x: {  
        timeUnit: "month",  
        field: "date",  
        type: "quantitative"  
      },  
      y: {  
        aggregate: "mean",  
        field: "precipitation",  
        type: "quantitative"  
      }  
    }  
  }  
}
```



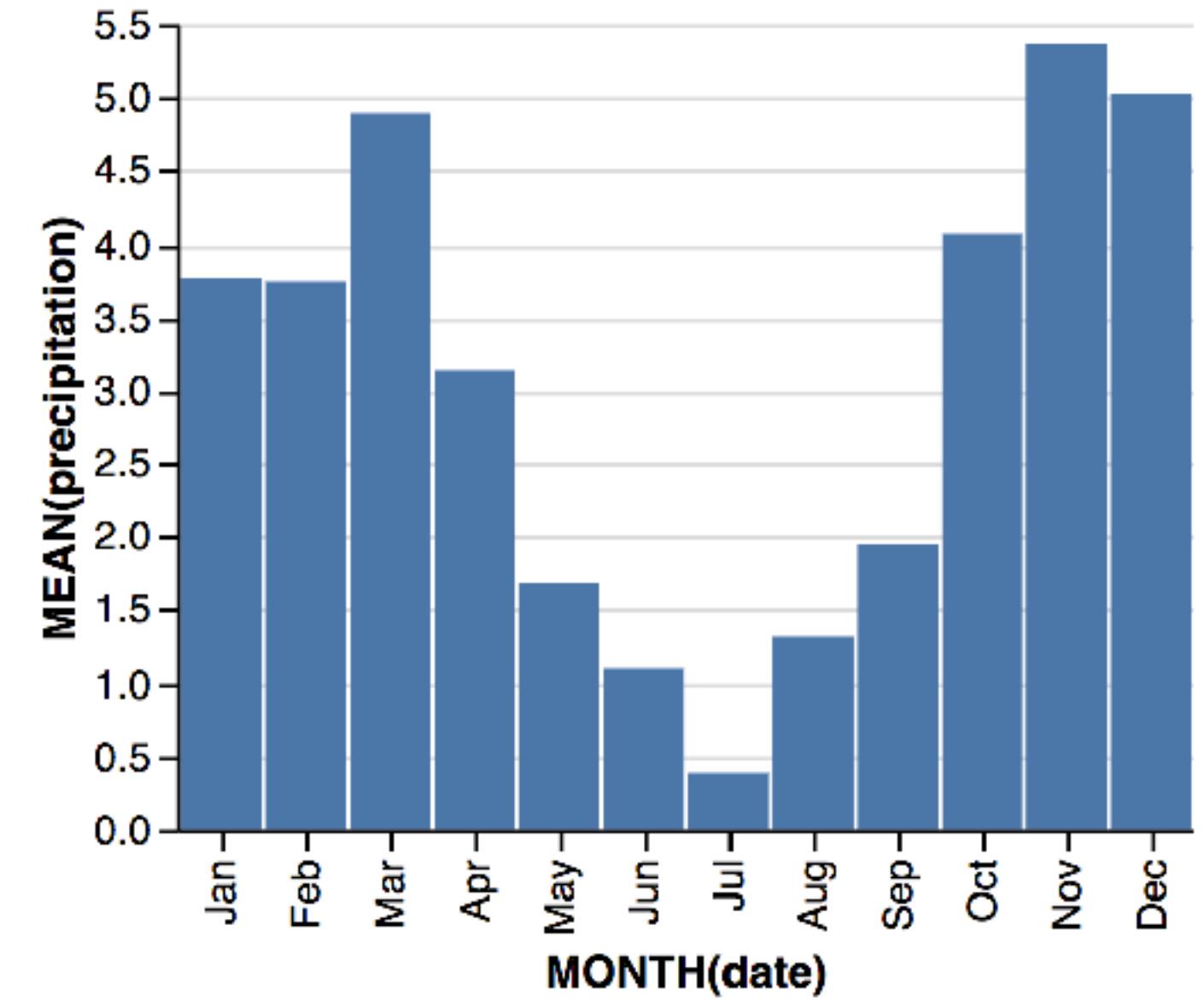
Layering

```
{  
  data: {url: "weather-seattle.json"},  
  layer: [  
    {  
      data: {  
        url: "weather-seattle.json"  
      },  
      layer: [  
        {  
          mark: "rule",  
          encoding: {  
            y: {  
              aggregate: "mean",  
              field: "precipitation",  
              type: "quantitative"  
            }  
          }  
        }  
      ]  
    }  
  ]  
}
```



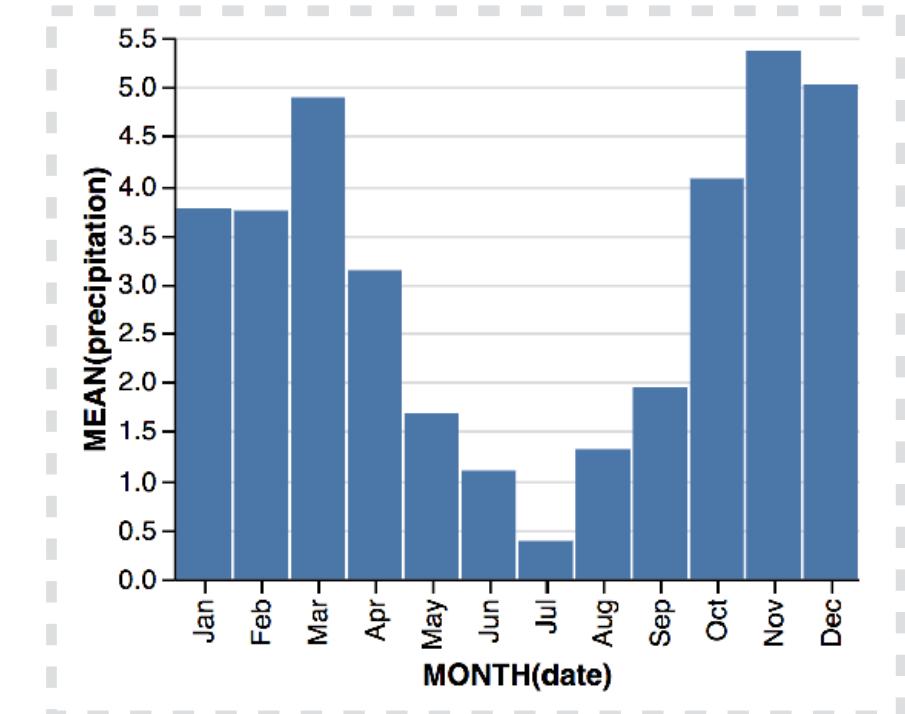
Concat

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      timeUnit: "month",  
      field: "date",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "mean",  
      field: "precipitation",  
      type: "quantitative"  
    }  
  }  
}
```



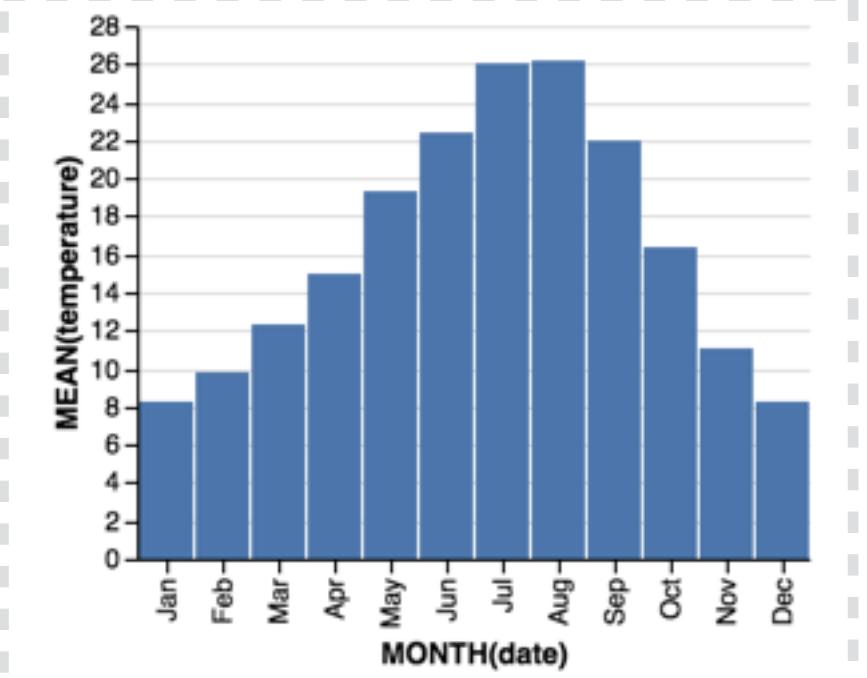
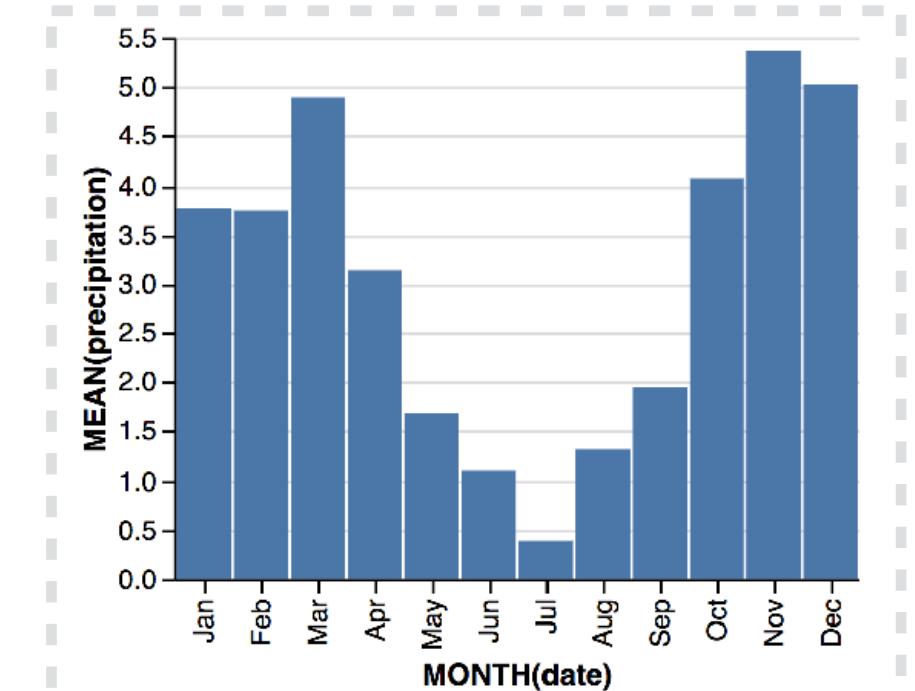
Concat

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      timeUnit: "month",  
      field: "date",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "mean",  
      field: "precipitation",  
      type: "quantitative"  
    }  
  }  
}
```



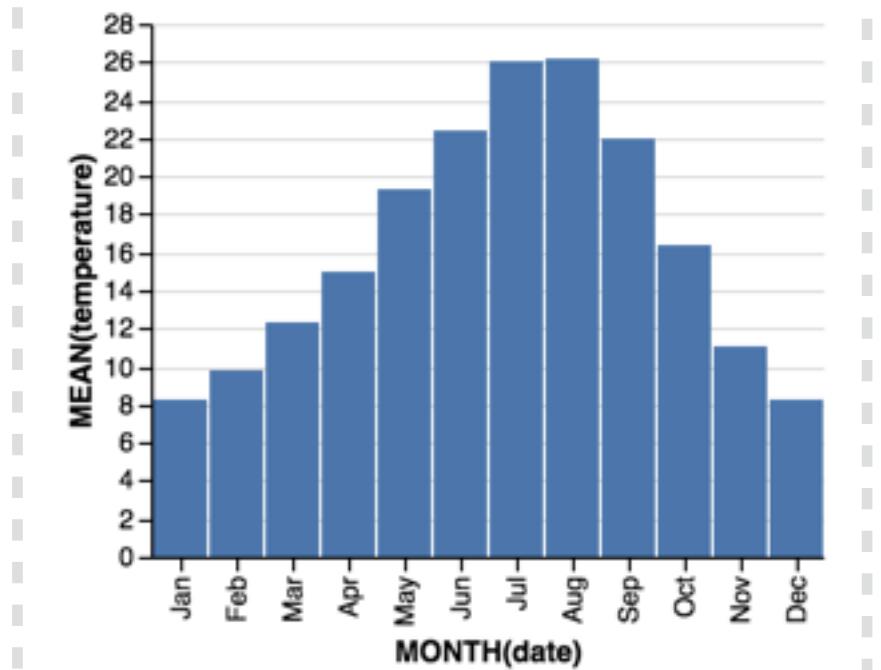
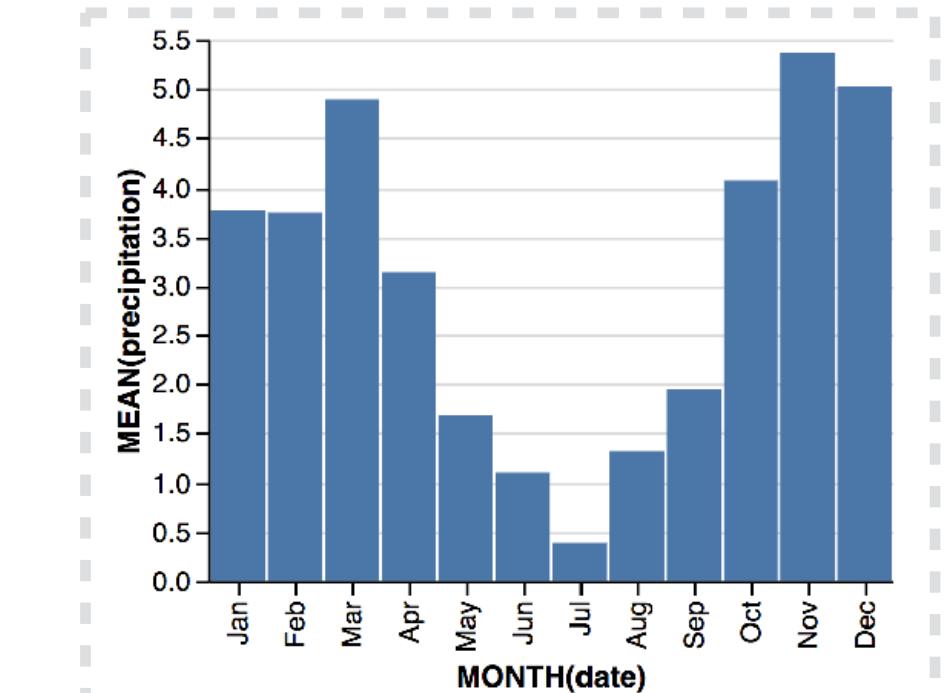
Concat

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      timeUnit: "month",  
      field: "date",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "mean",  
      field: "precipitation",  
      type: "quantitative"  
    }  
  }  
}  
  
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      timeUnit: "month",  
      field: "date",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "mean",  
      field: "temperature",  
      type: "quantitative"  
    }  
  }  
}
```



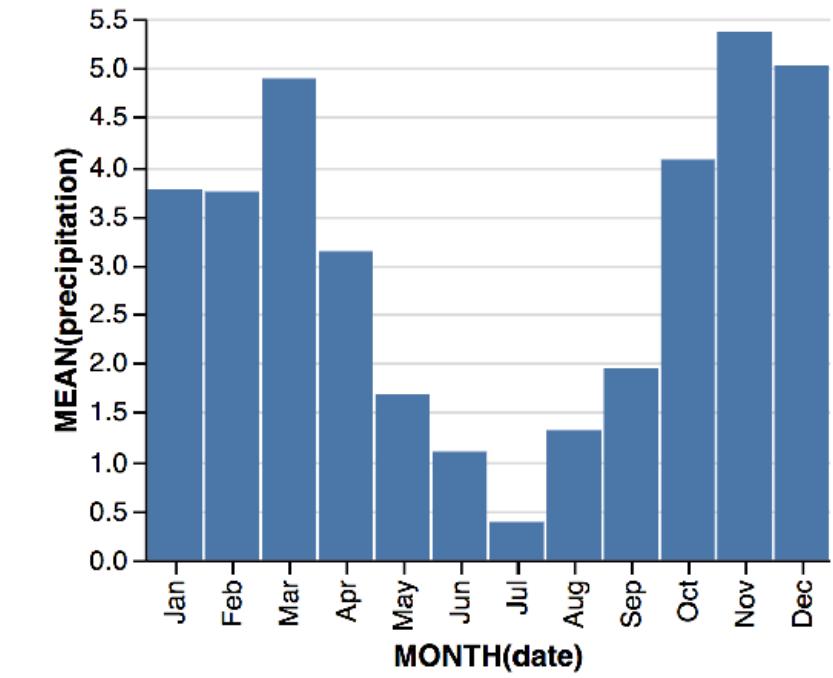
Concat

```
{  
  vconcat: [{  
    data: {url: "weather-seattle.json"},  
    mark: "bar",  
    encoding: {  
      x: {  
        timeUnit: "month",  
        field: "date",  
        type: "quantitative"  
      },  
      y: {  
        aggregate: "mean",  
        field: "precipitation",  
        type: "quantitative"  
      }  
    }  
  },  
  {  
    data: {url: "weather-seattle.json"},  
    mark: "bar",  
    encoding: {  
      x: {  
        timeUnit: "month",  
        field: "date",  
        type: "quantitative"  
      },  
      y: {  
        aggregate: "mean",  
        field: "temperature",  
        type: "quantitative"  
      }  
    }  
  }]  
}
```



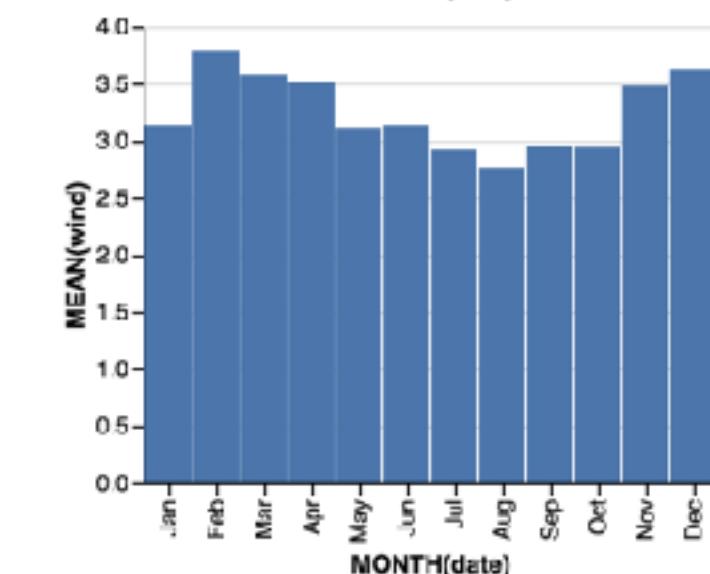
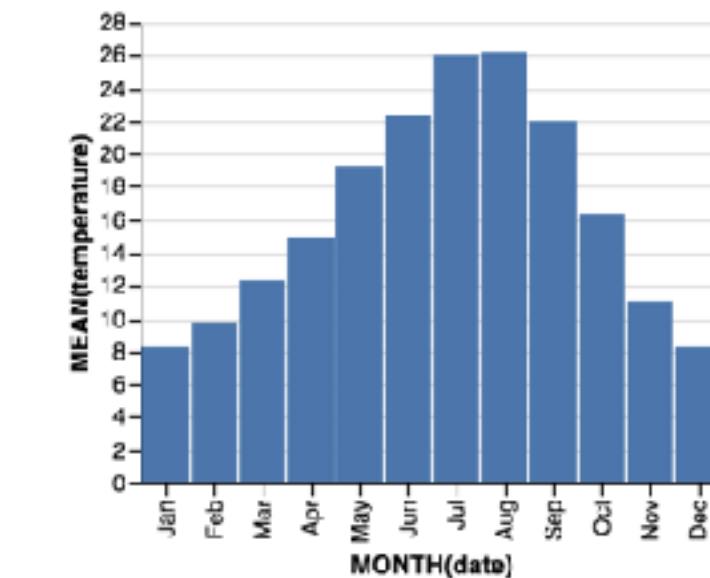
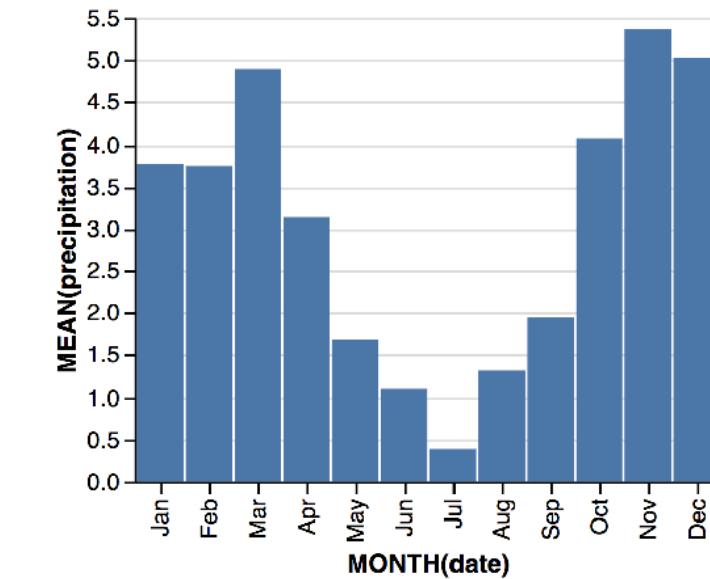
Repeat

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      timeUnit: "month",  
      field: "date",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "mean",  
      field: "precipitation",  
      type: "quantitative"  
    }  
  }  
}
```



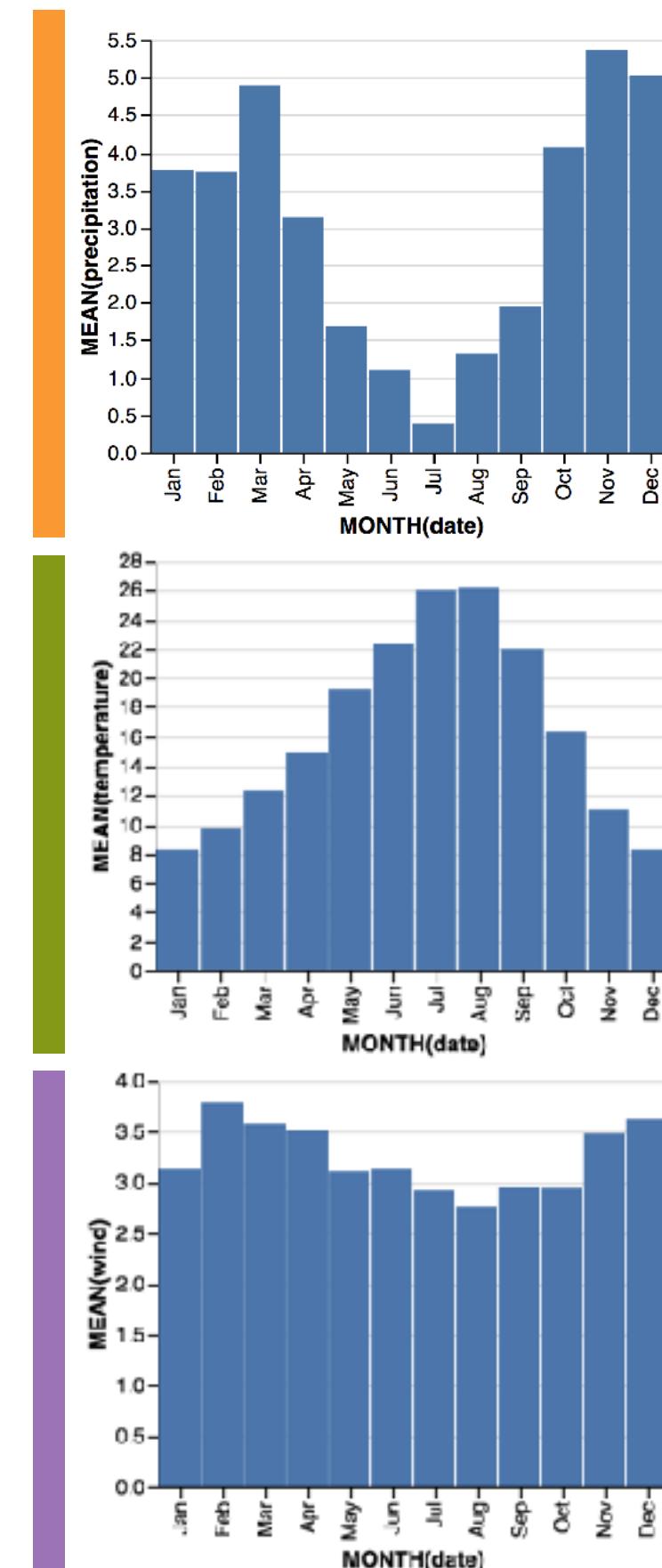
Repeat

```
{  
  repeat: {  
    column: ["precipitation", "temperature", "wind"]  
  },  
  spec: {  
    data: {url: "weather-seattle.json"},  
    mark: "bar",  
    encoding: {  
      x: {  
        timeUnit: "month",  
        field: "date",  
        type: "quantitative"  
      },  
      y: {  
        aggregate: "mean",  
        field: {repeat: "column"},  
        type: "quantitative"  
      }  
    }  
  }  
}
```



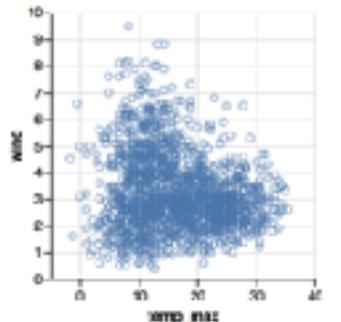
Concat

```
{  
  repeat: {  
    column: ["precipitation", "temperature", "wind"]  
  },  
  spec: {  
    data: {url: "weather-seattle.json"},  
    mark: "bar",  
    encoding: {  
      x: {  
        timeUnit: "month",  
        field: "date",  
        type: "quantitative"  
      },  
      y: {  
        aggregate: "mean",  
        field: {repeat: "column"},  
        type: "quantitative"  
      }  
    }  
  }  
}
```

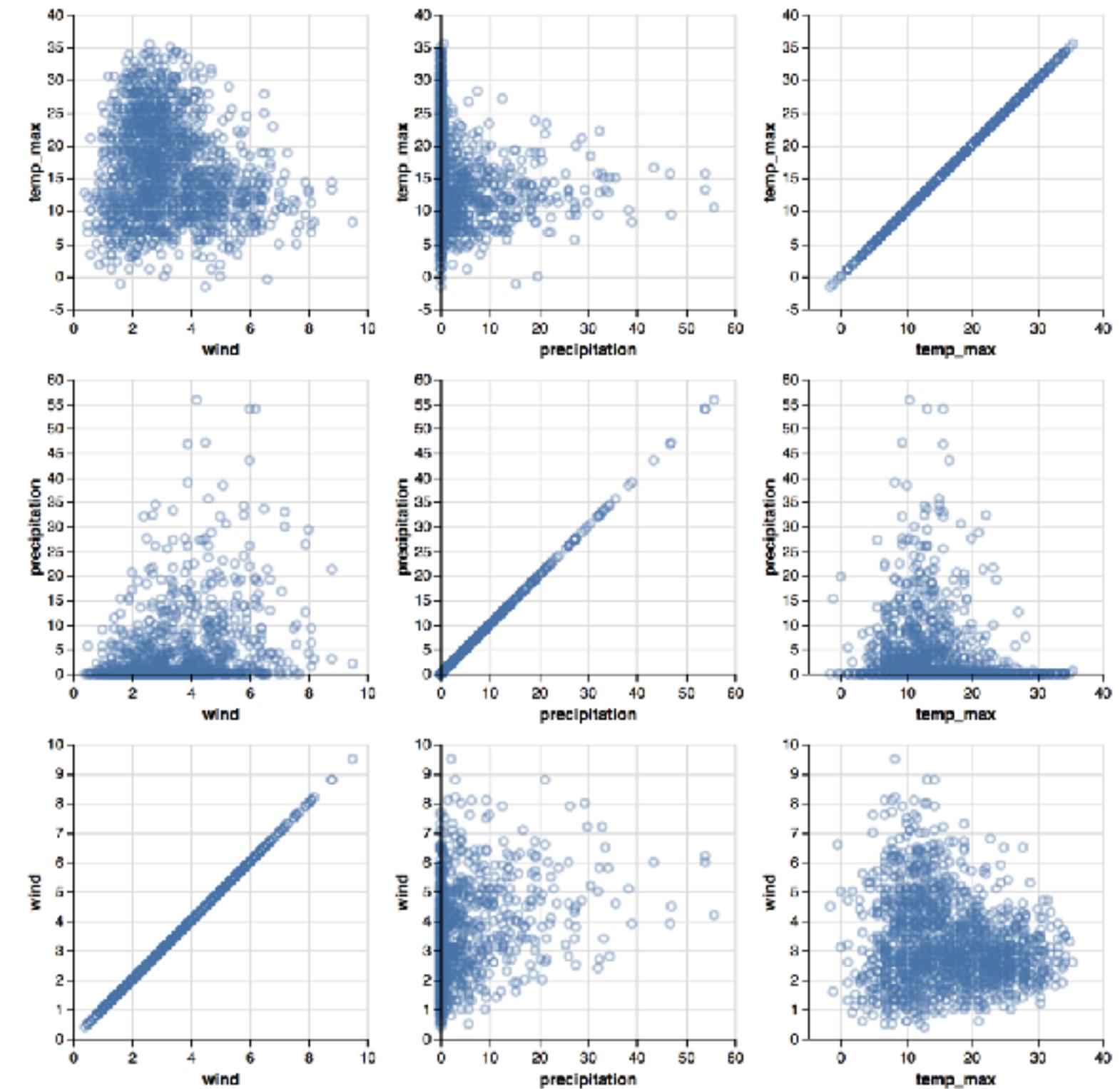


Repeat: SPLOM

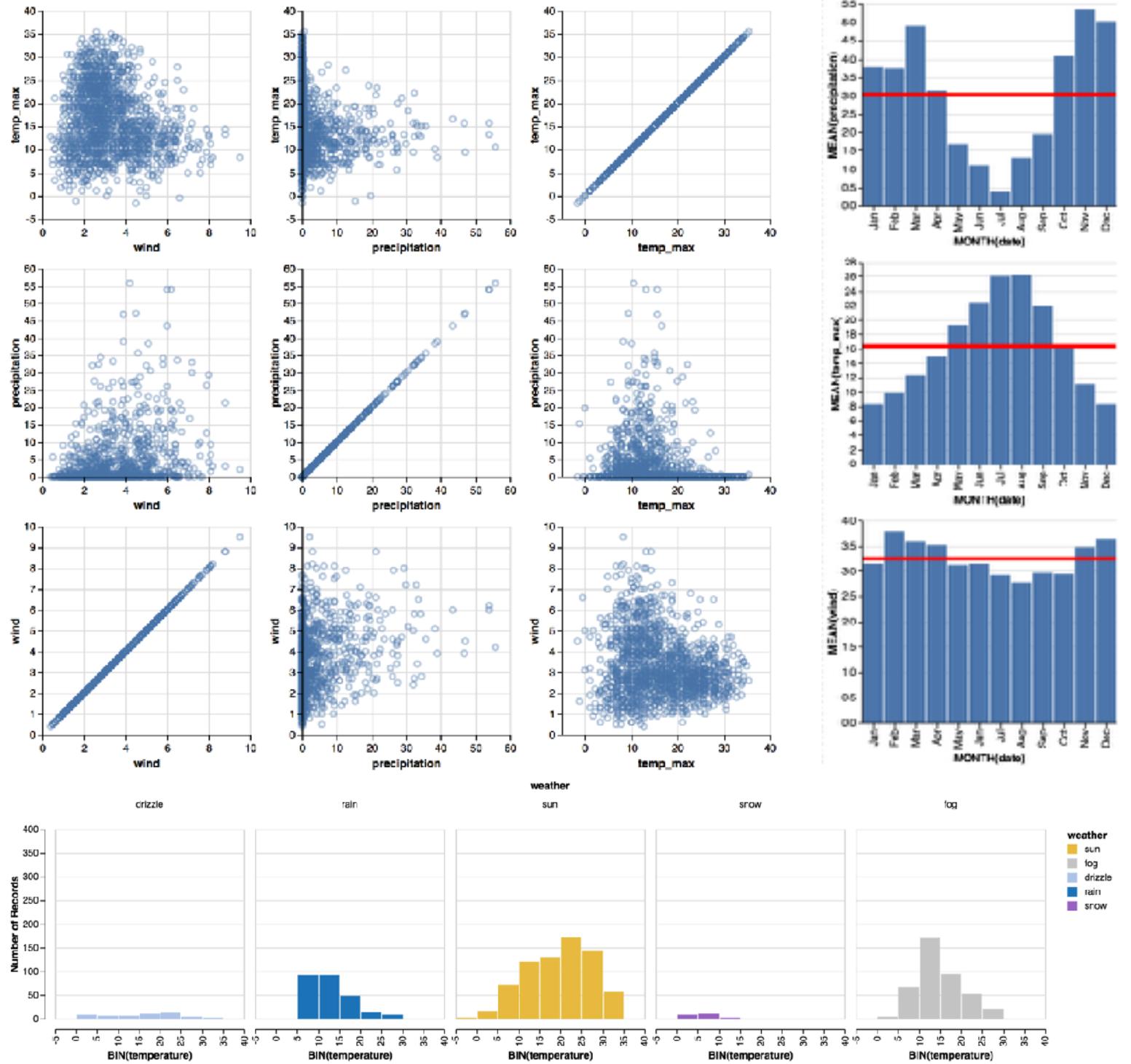
```
{  
  repeat: {  
    column: [  
      "temperature"  
      "precipitation",  
      "wind"],  
    row: [  
      "wind",  
      "precipitation",  
      "temperature",  
    ],  
  },  
  spec: {  
    }
```



```
}
```



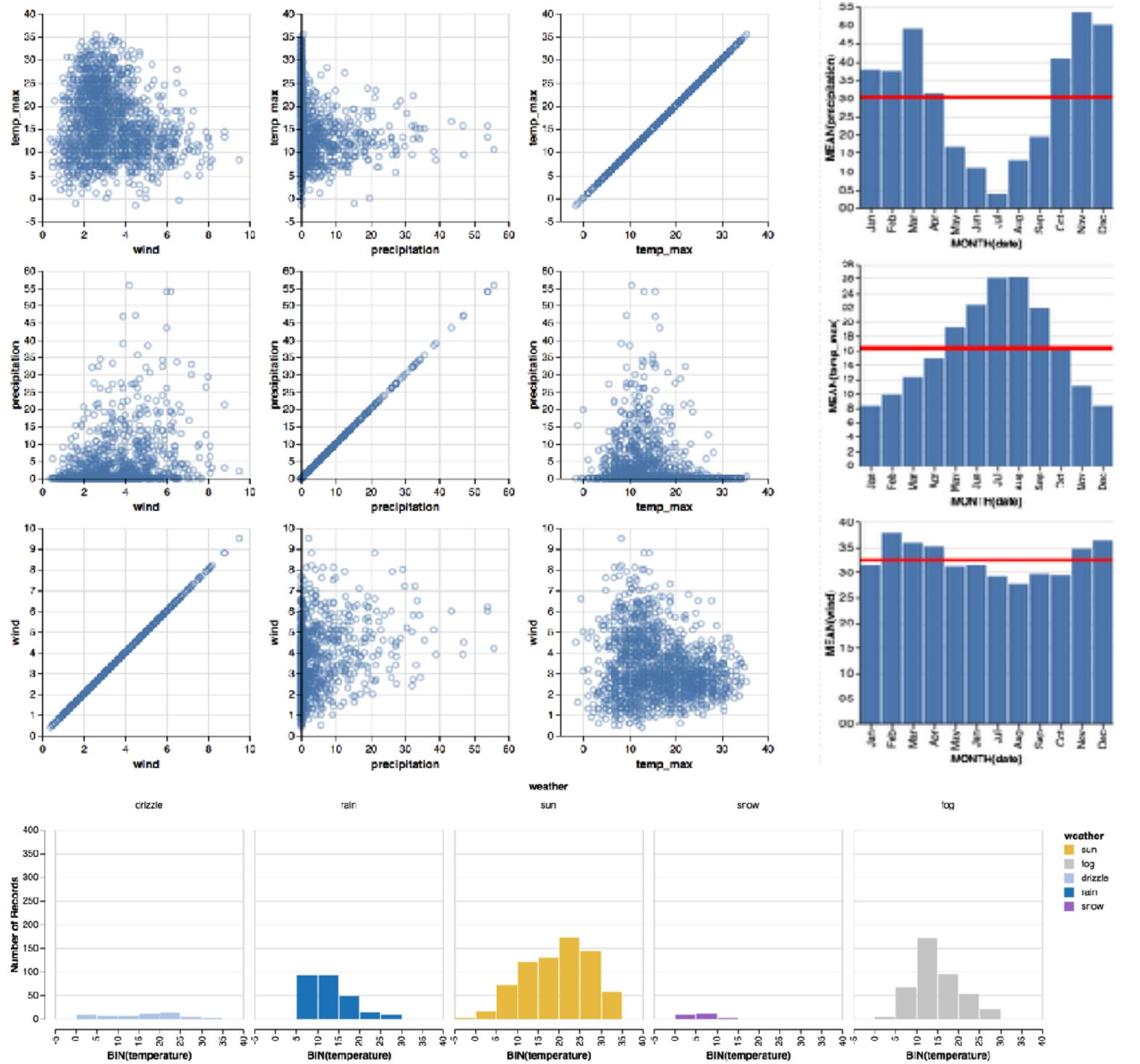
Dashboards



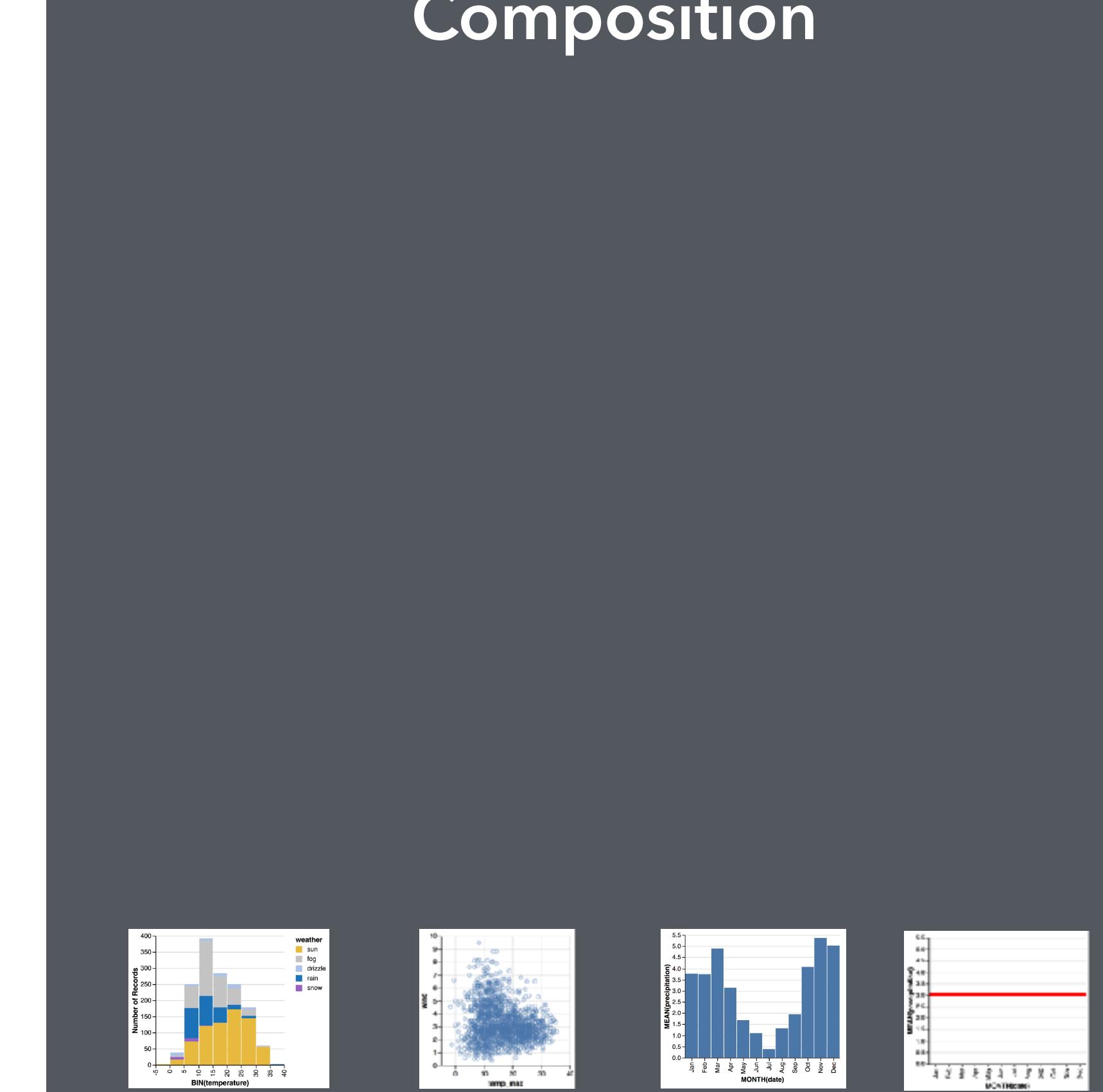
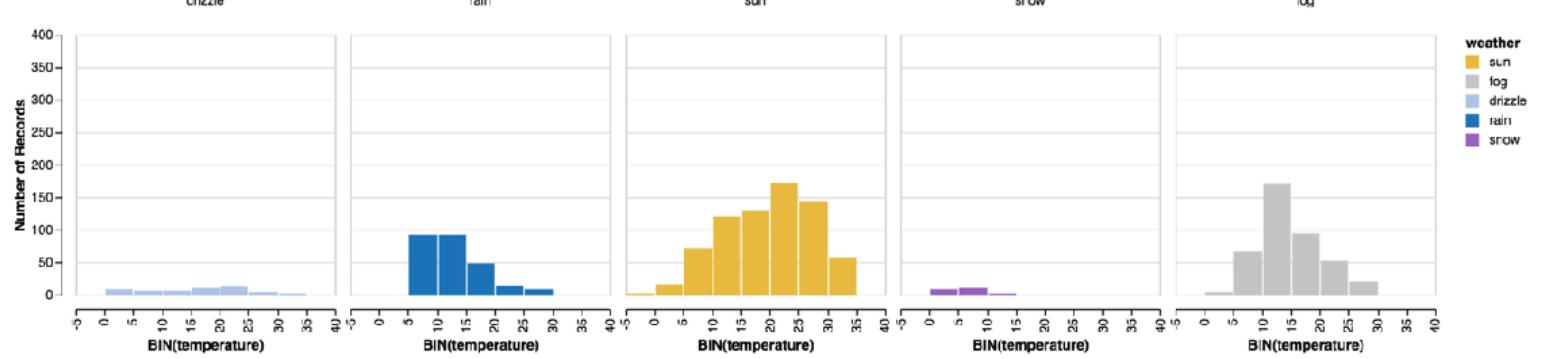
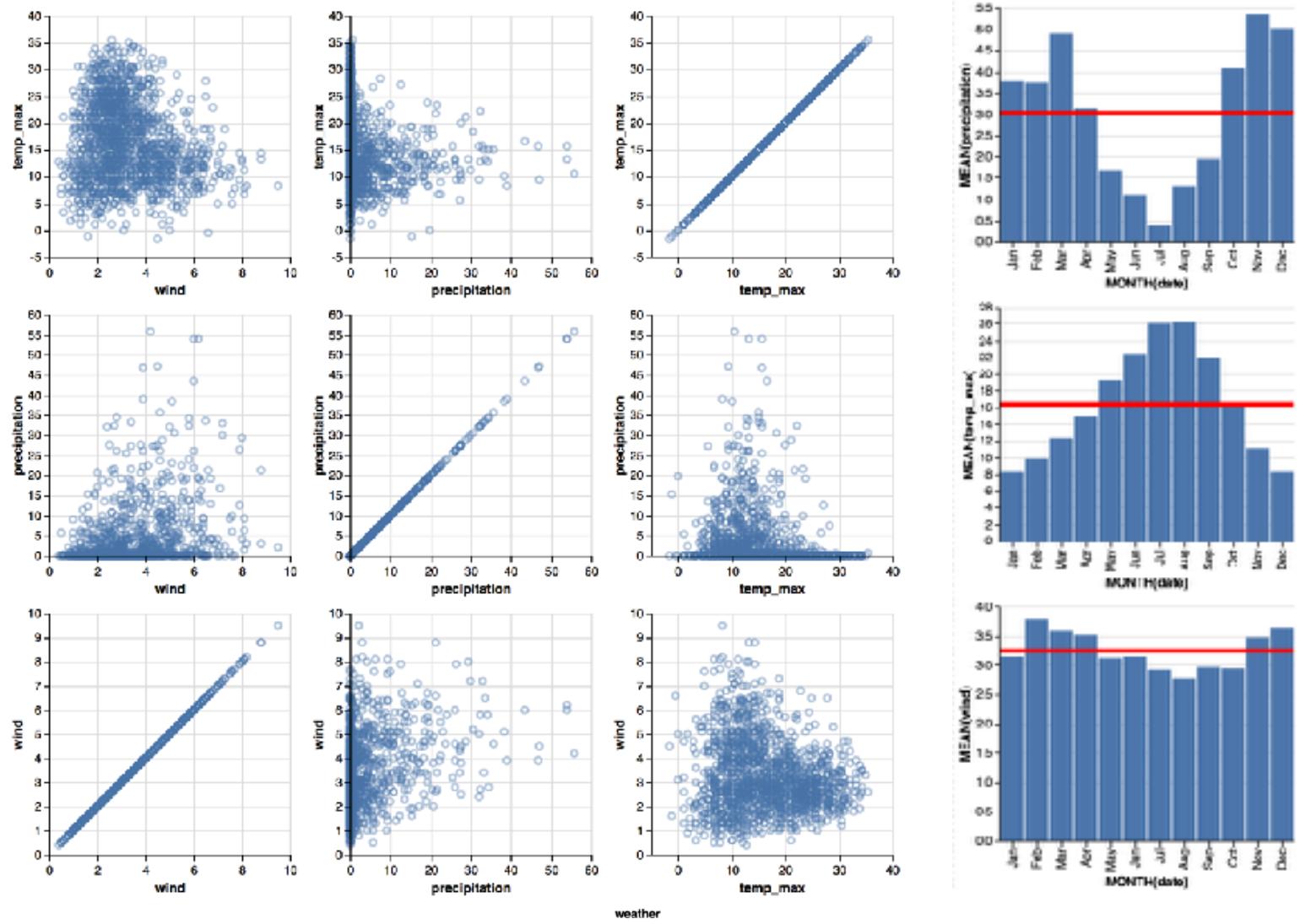
Many views, some of them composed

Manual data, and view management

Composition

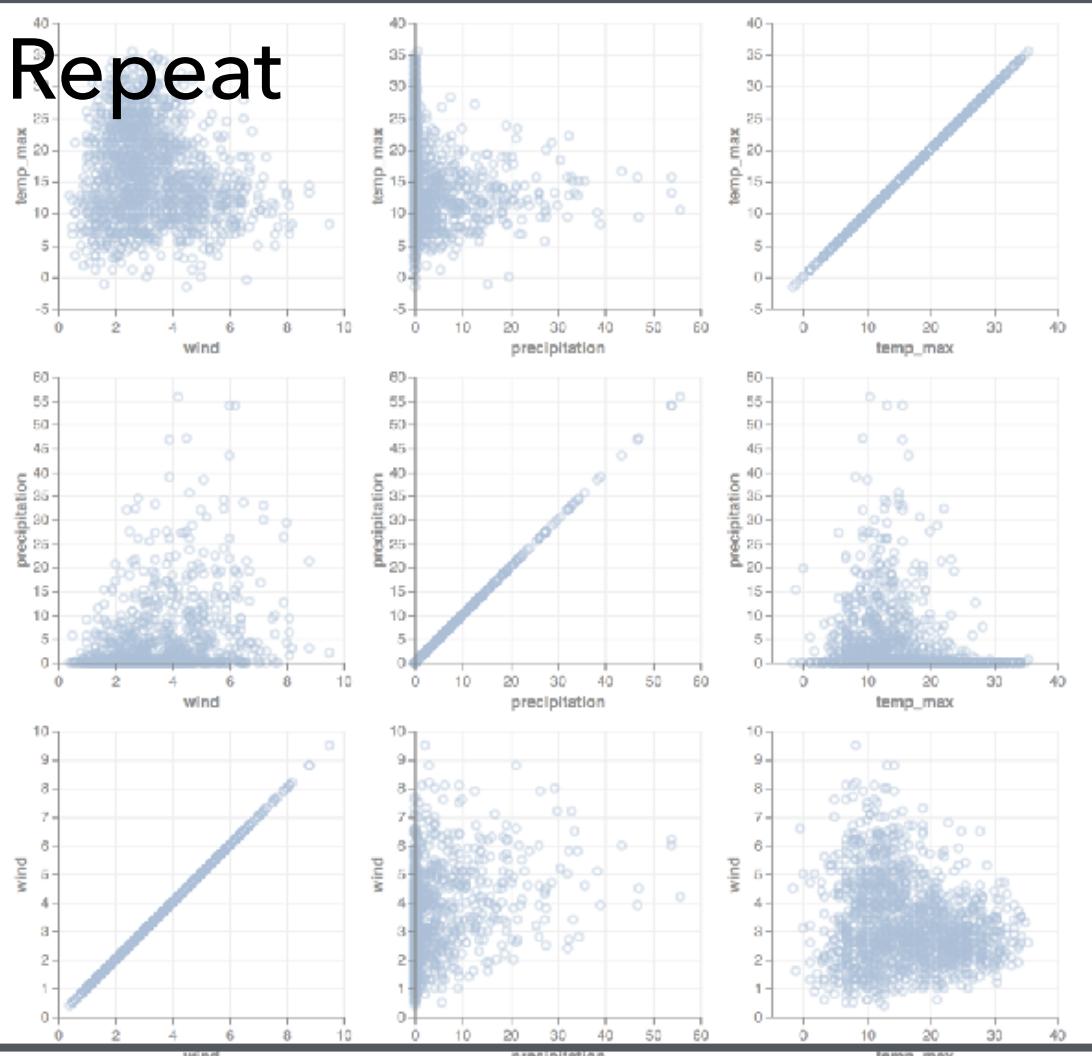


Composition

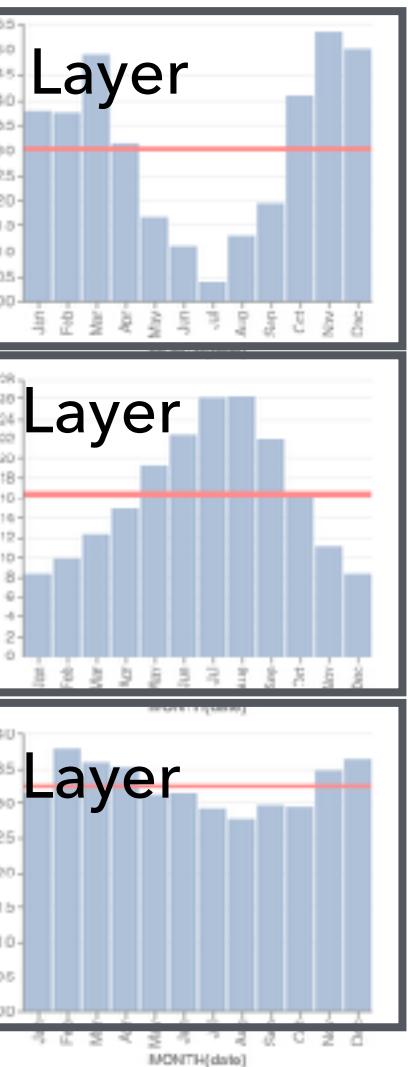


Composition

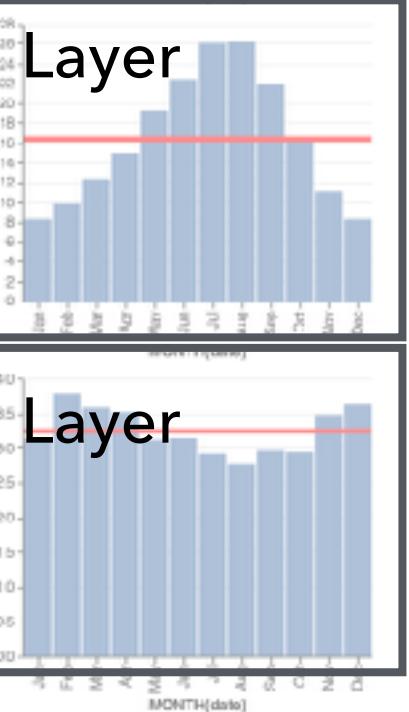
Repeat



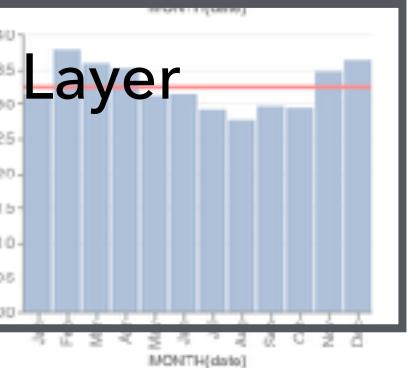
Layer



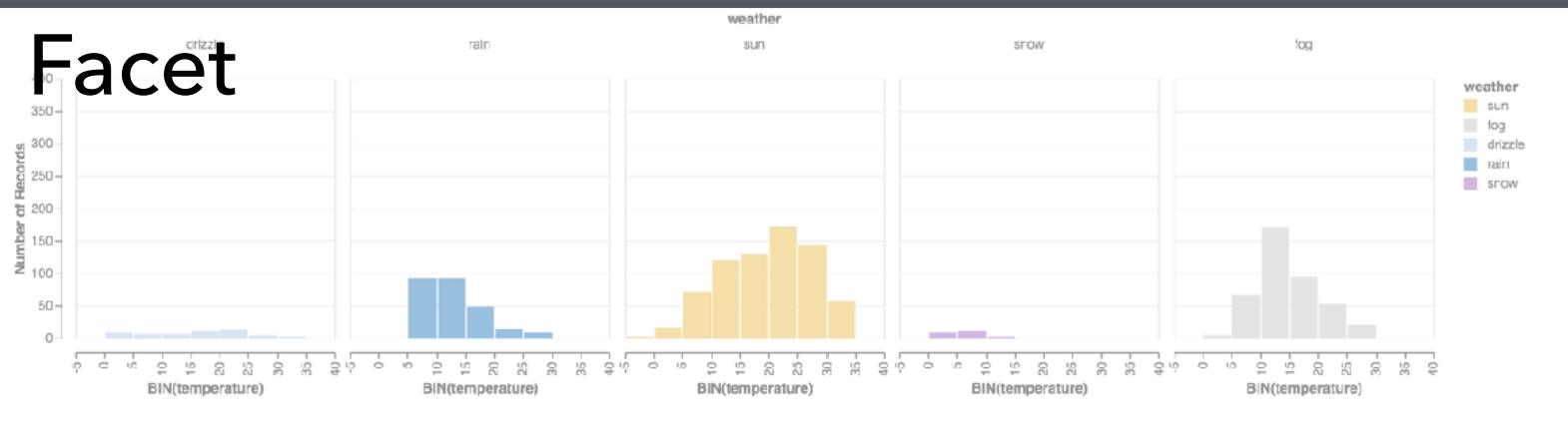
Layer



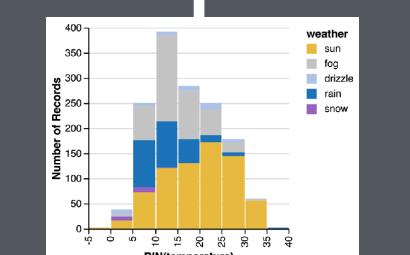
Layer



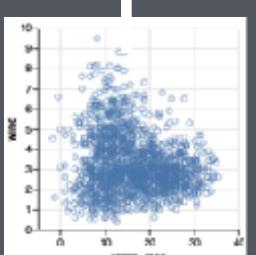
Facet



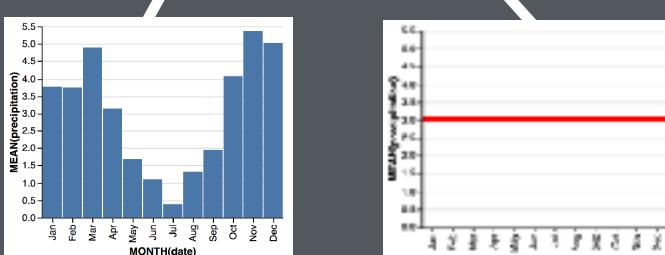
Facet
Weather



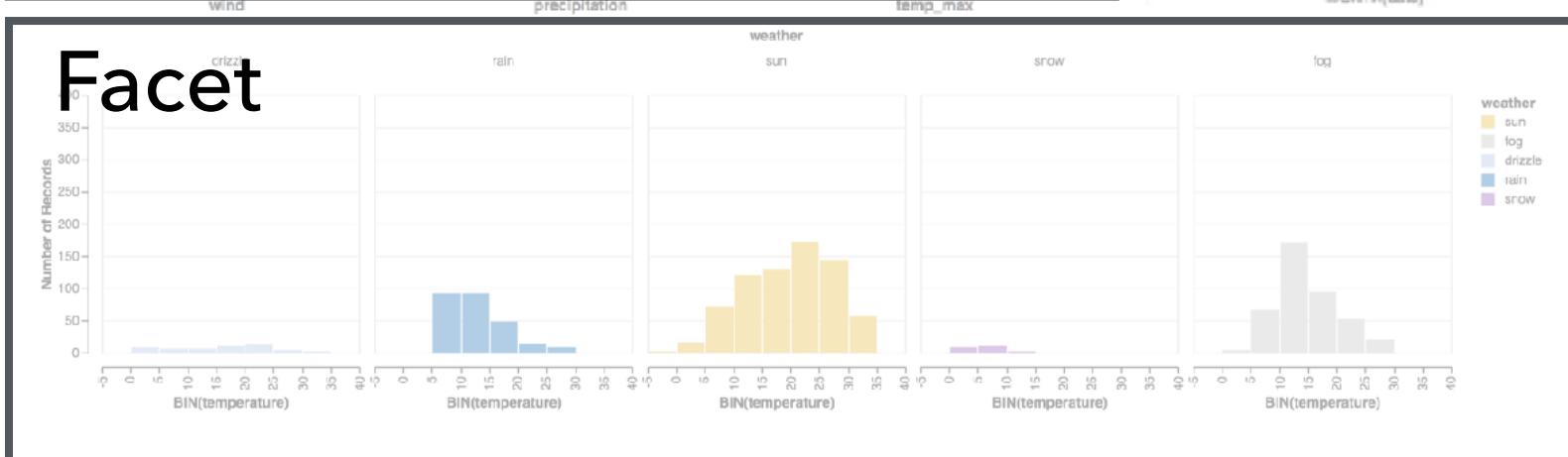
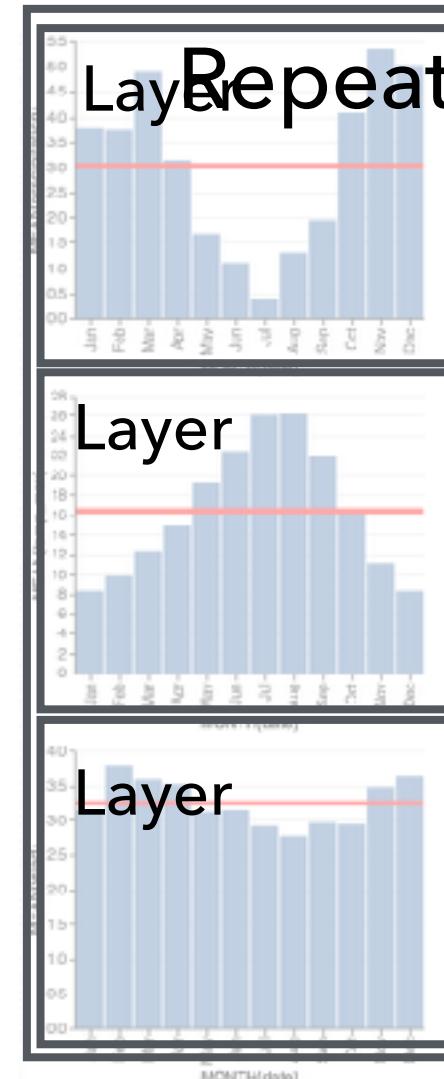
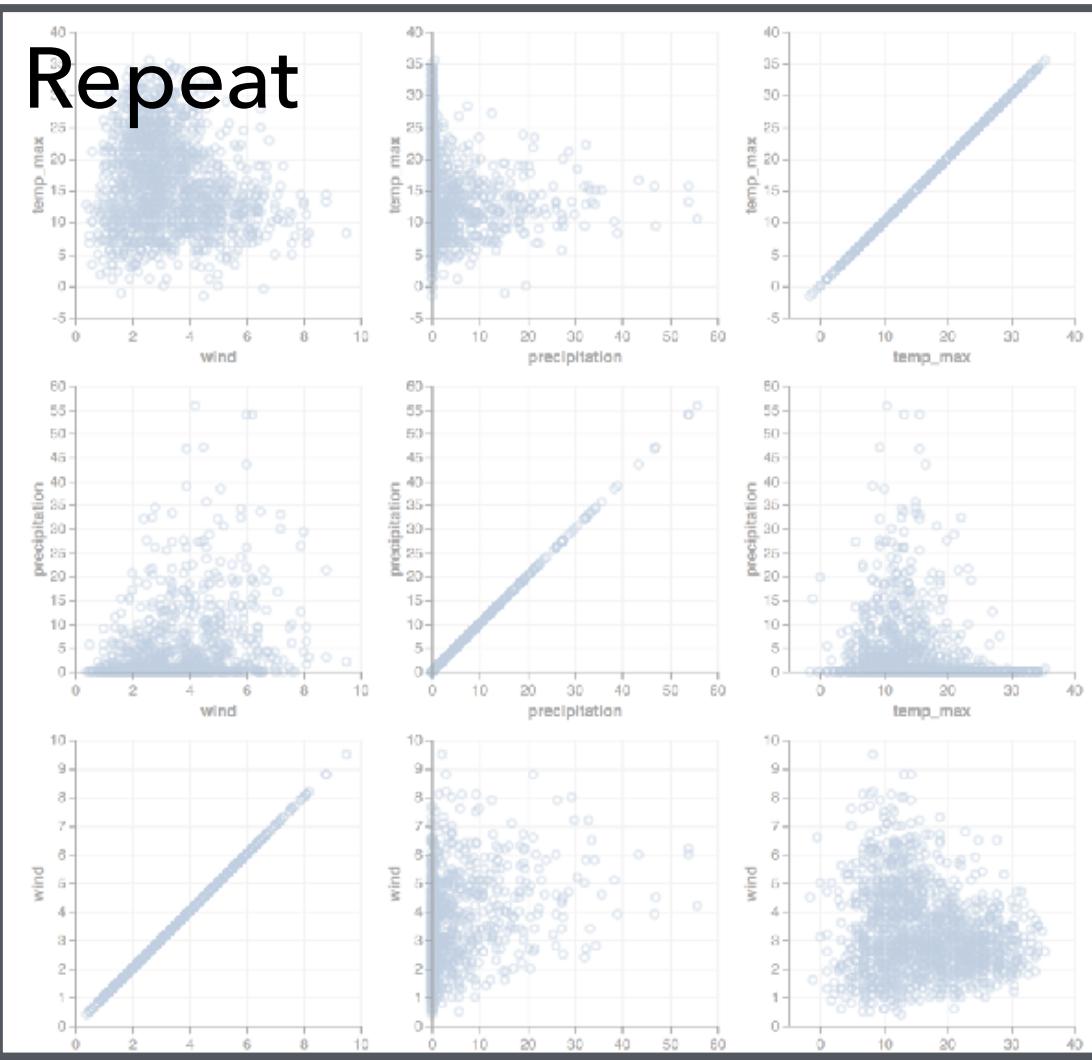
Repeat
Temp, Prec, Wind



Layer

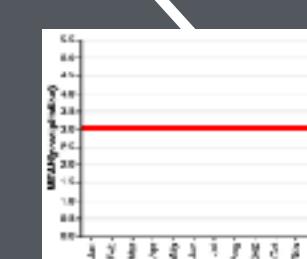
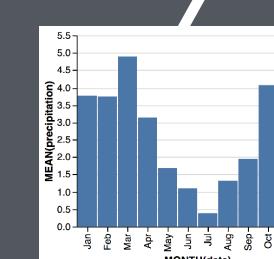
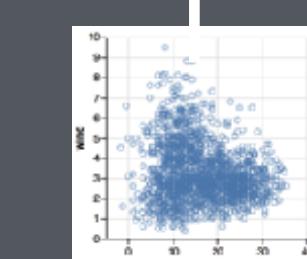
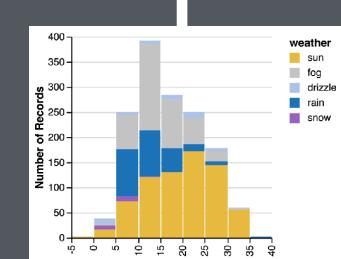


Hierarchical View Composition

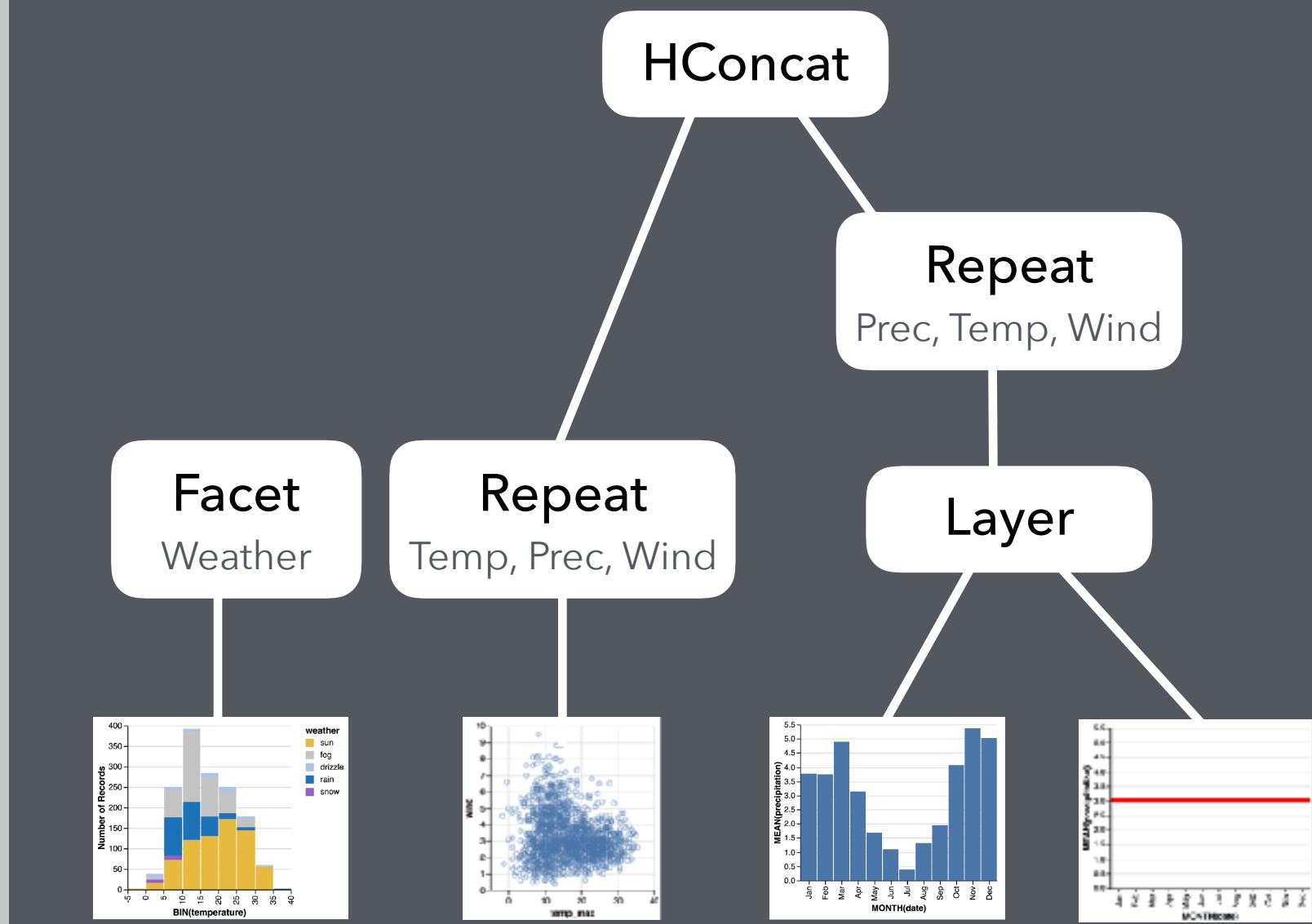
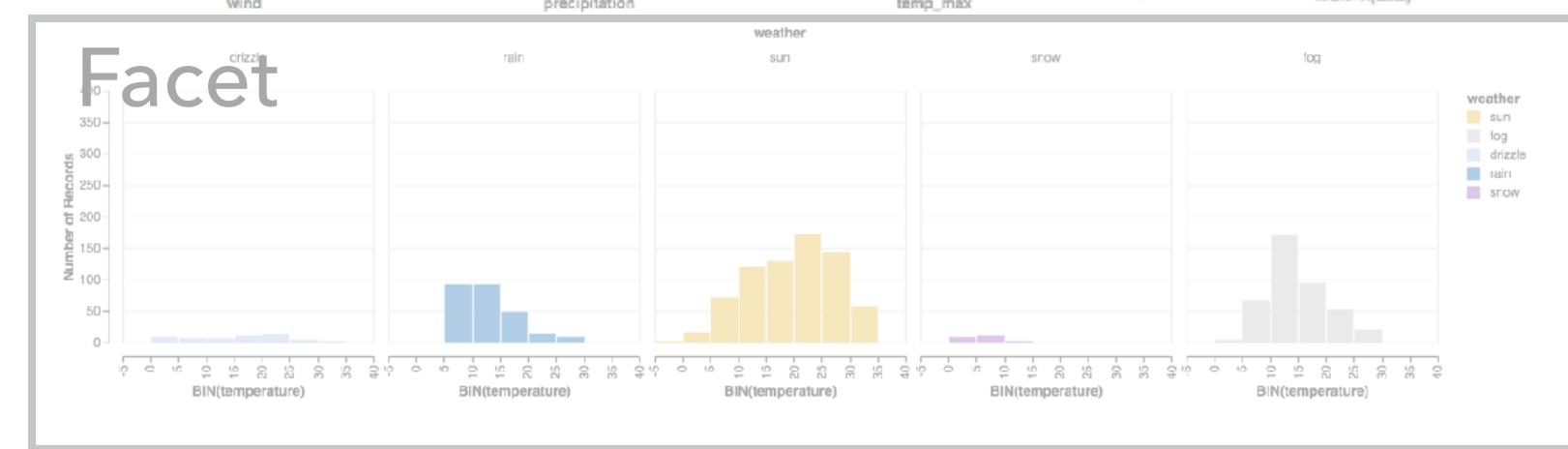
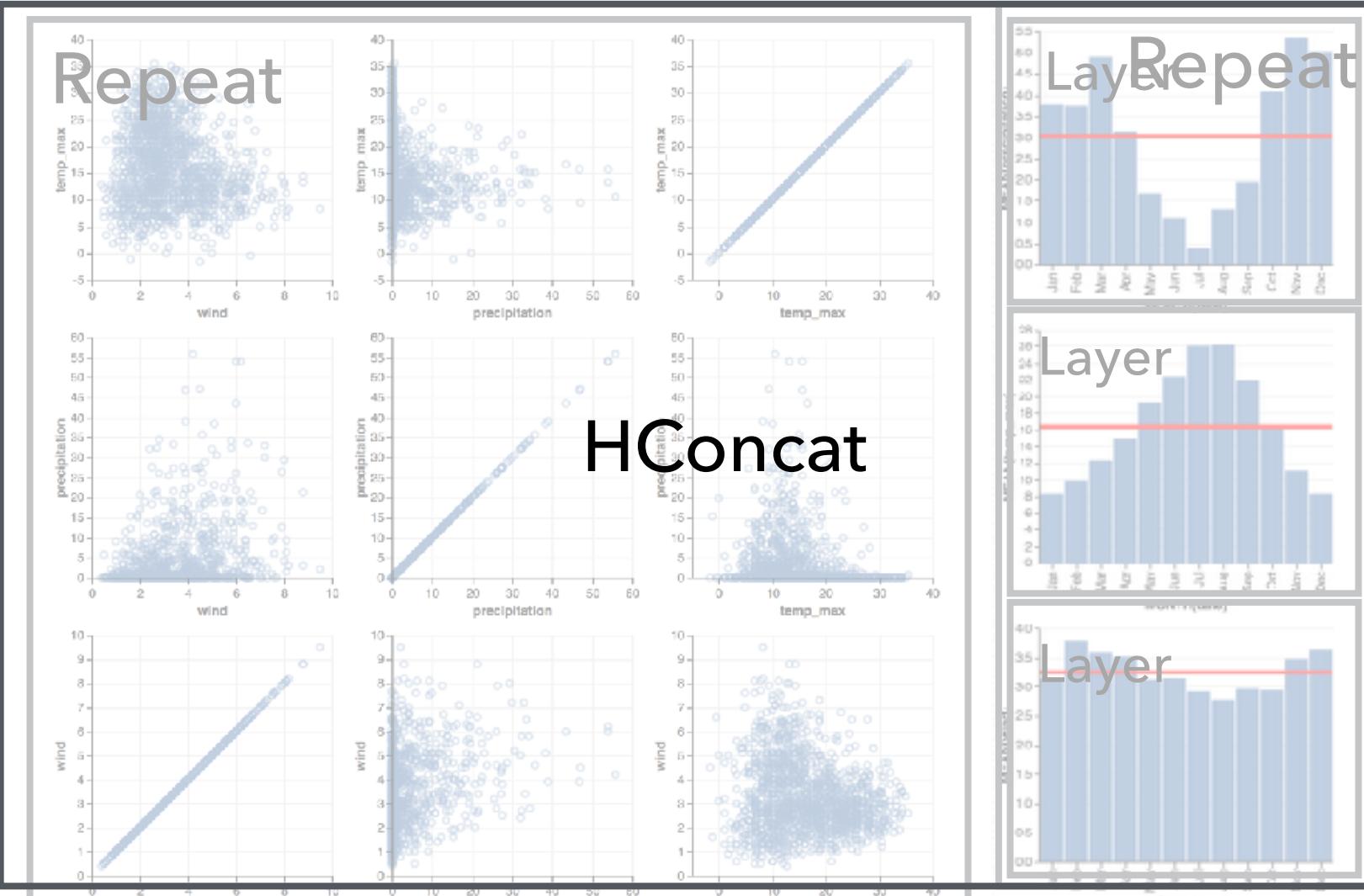


Facet
Weather

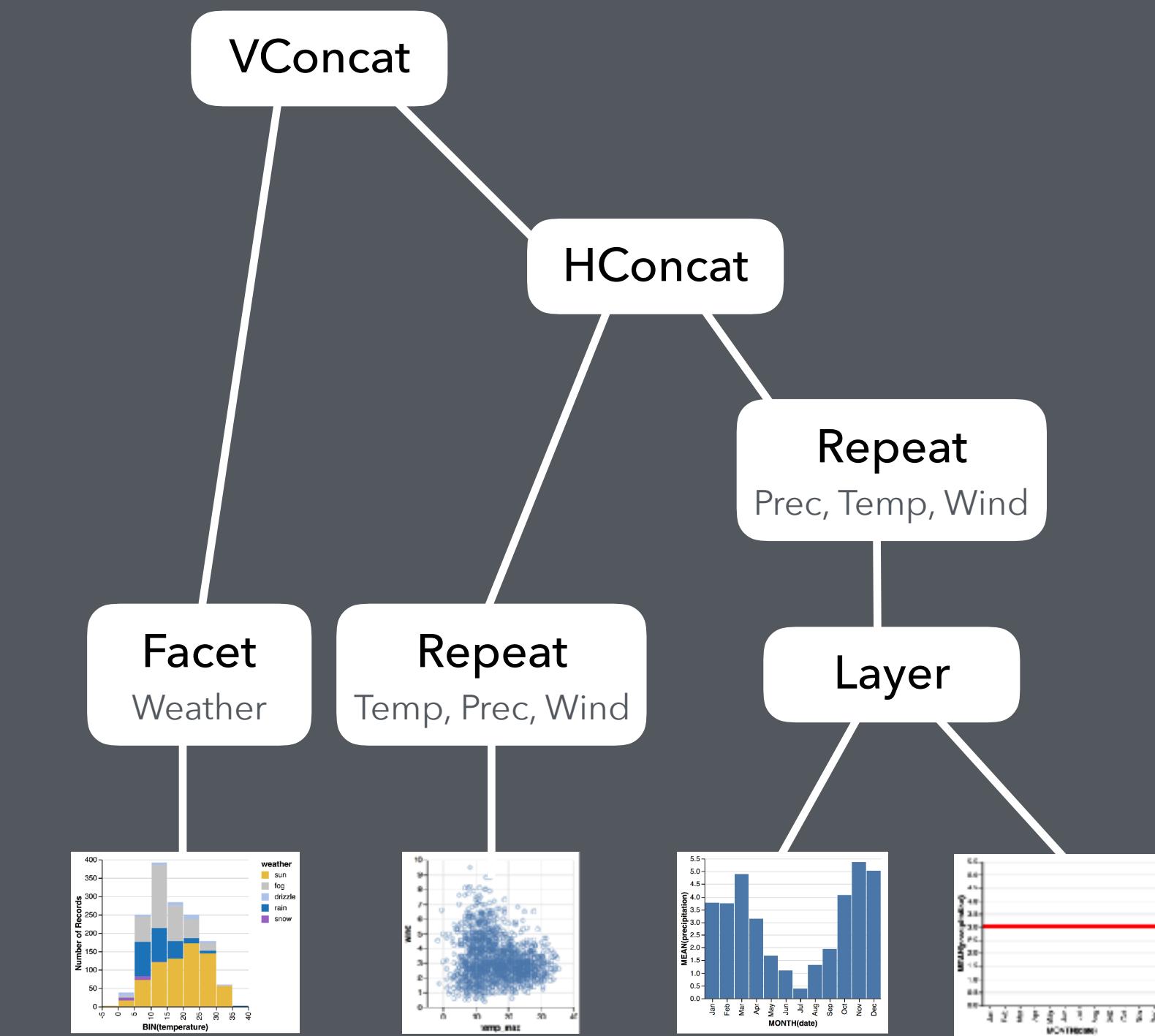
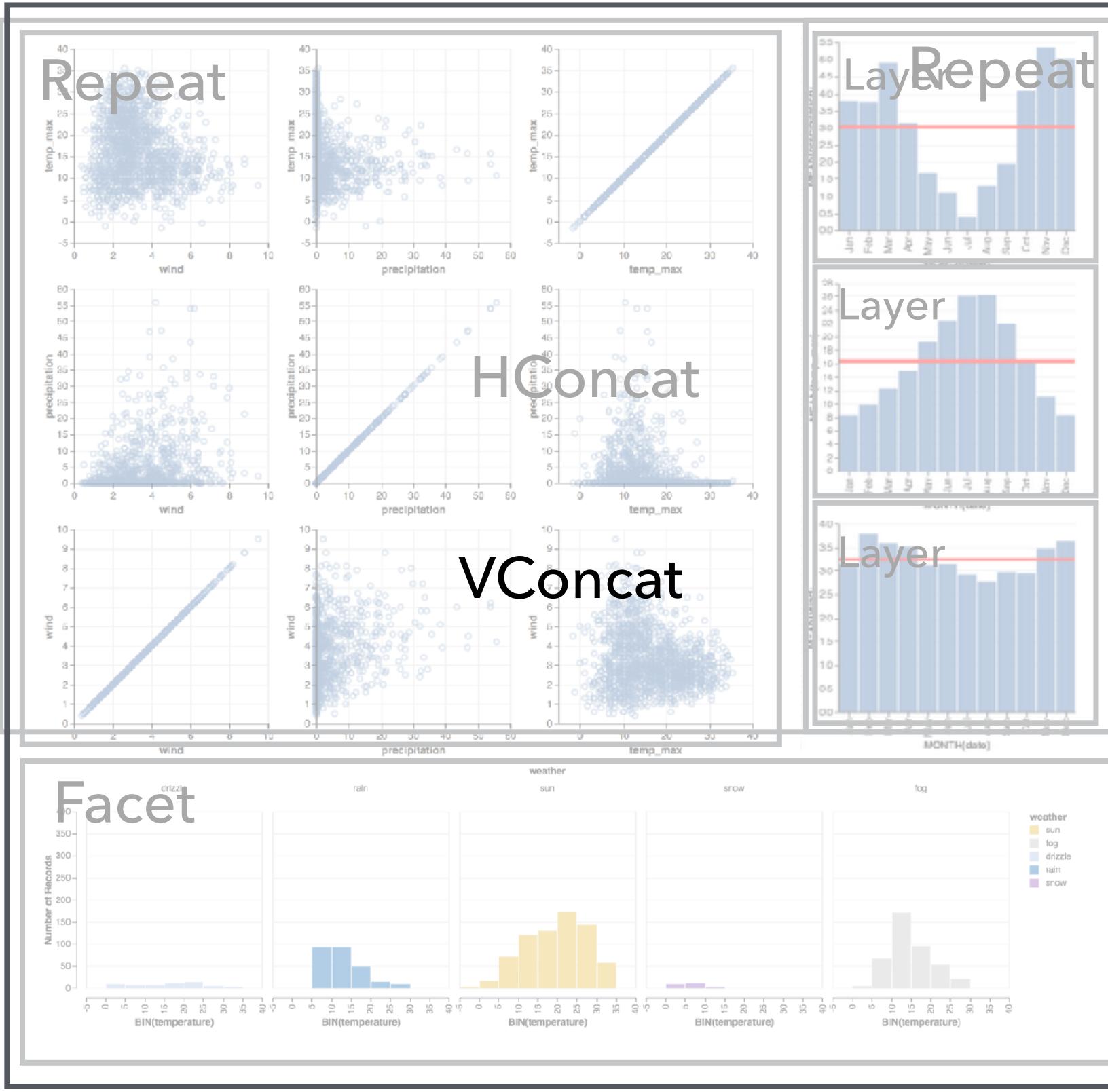
Repeat
Temp, Prec, Wind



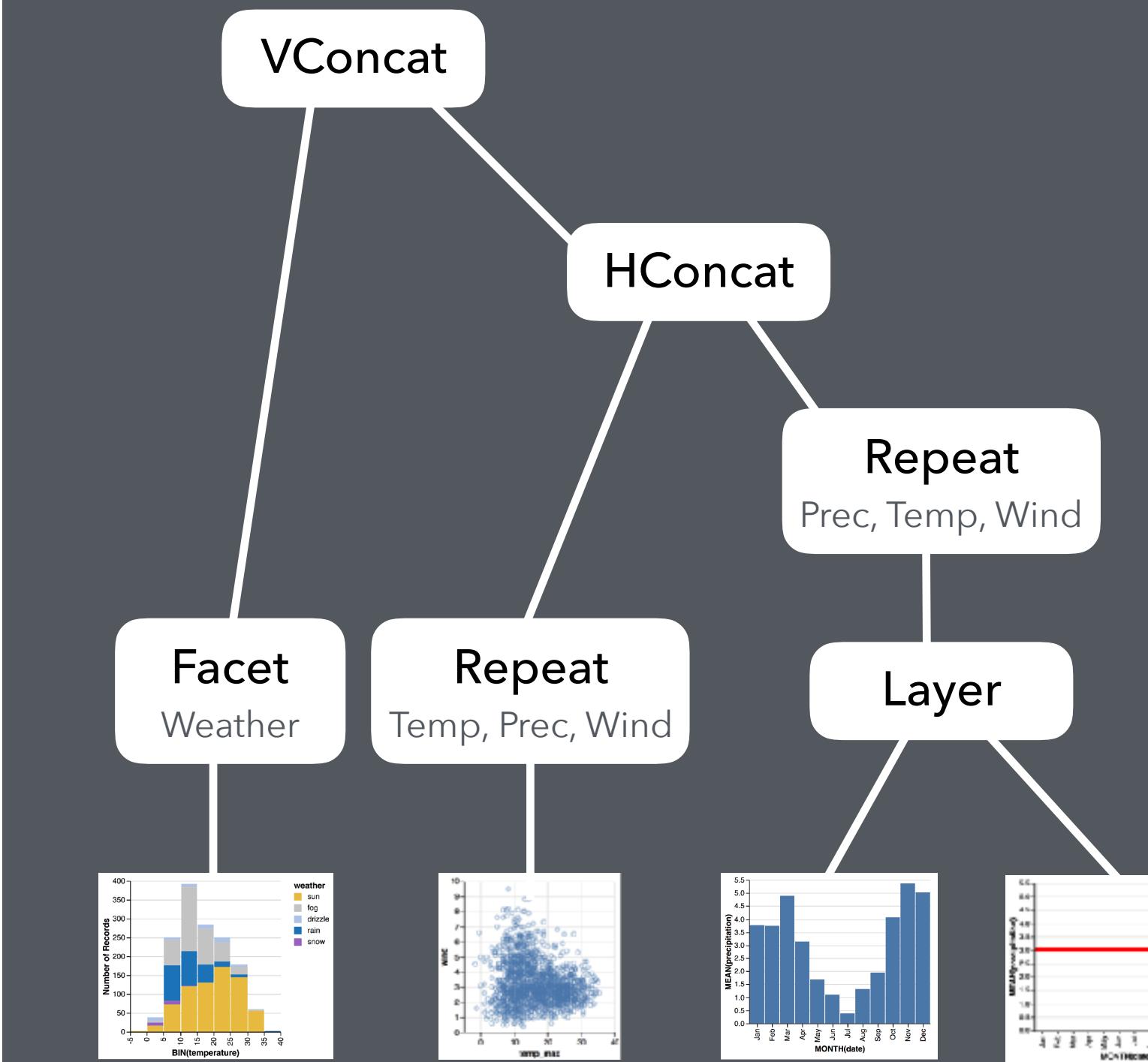
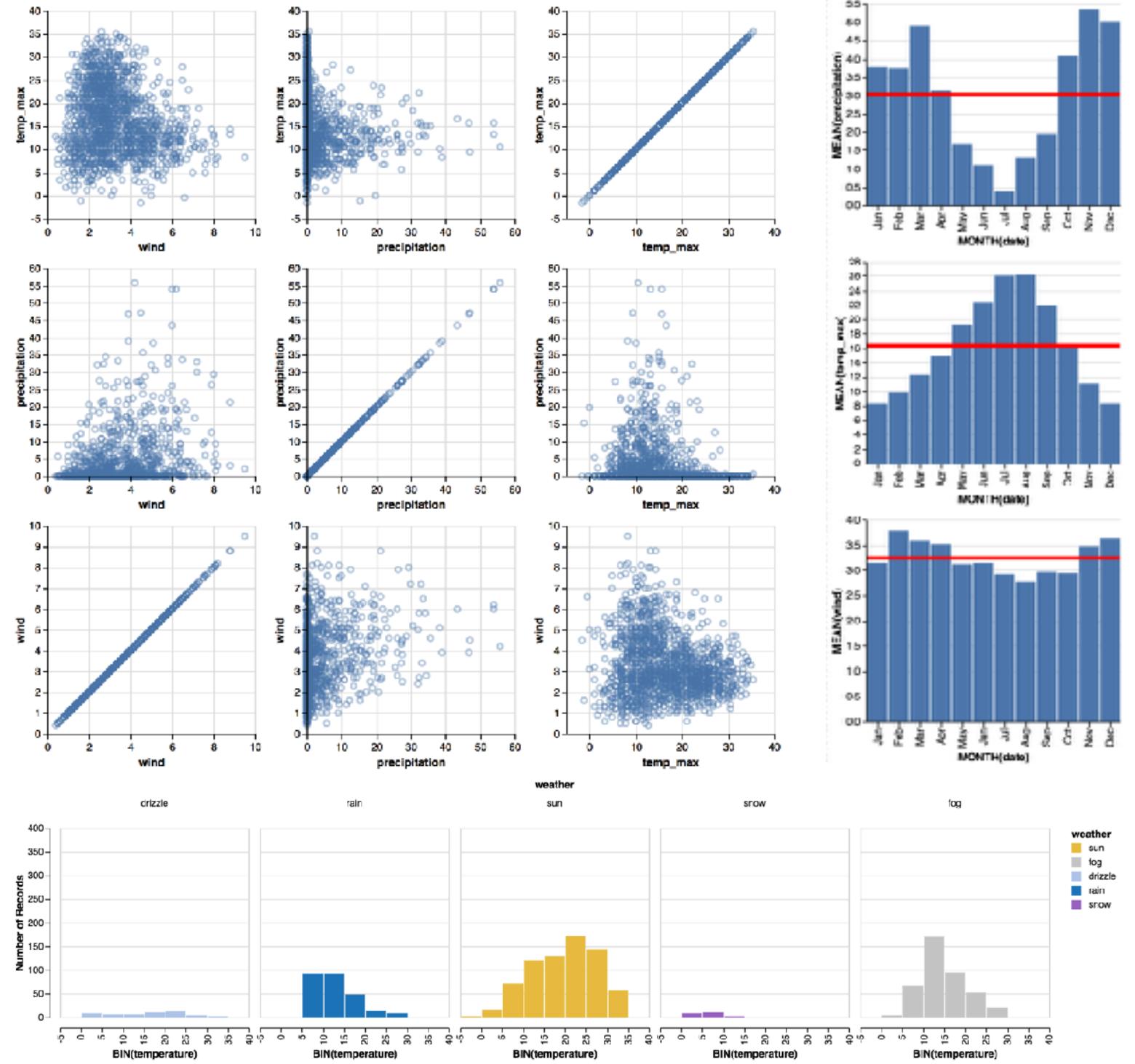
Hierarchical View Composition



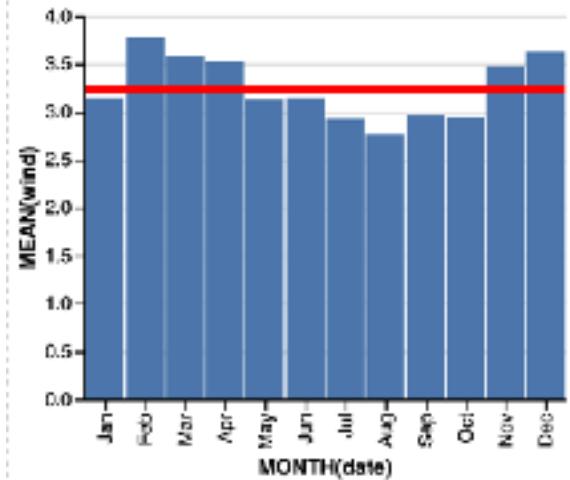
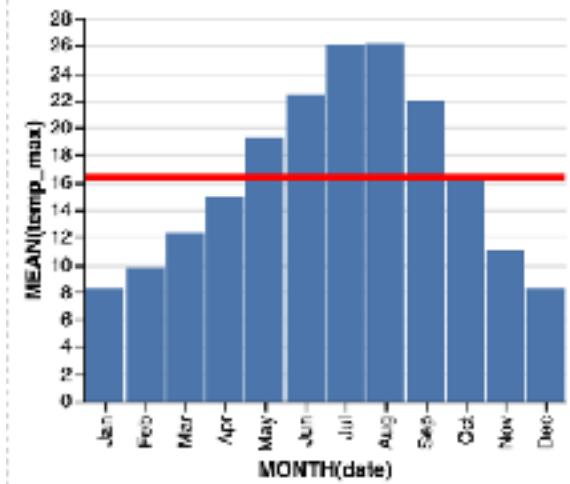
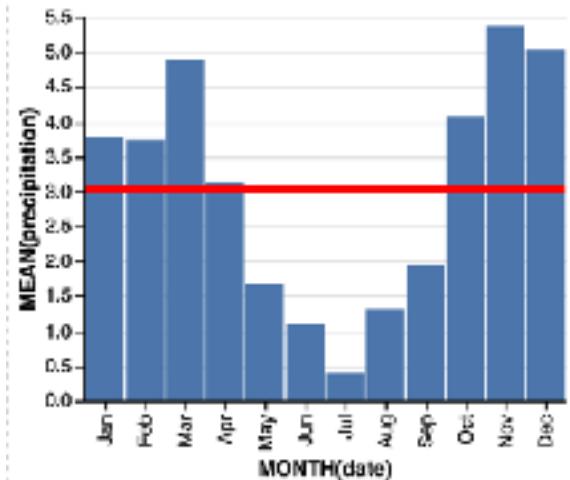
Hierarchical View Composition



Hierarchical View Composition



Layer + Repeat



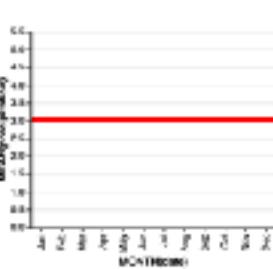
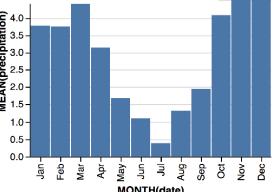
{

Repeat

Prec,Temp,Wind 'precipitation", "temperature", "wind"]

,
spec: {

Layer



},

]

}

Vega-Lite: a Grammar of Interactive Graphics

The Design of Vega-Lite

Single View Specification

Layered and **Multi-view** Composition

Interactions with Selections

Using Vega-Lite

Programming with Vega-Lite

Higher-level Tools and **Recommendations**

Selections

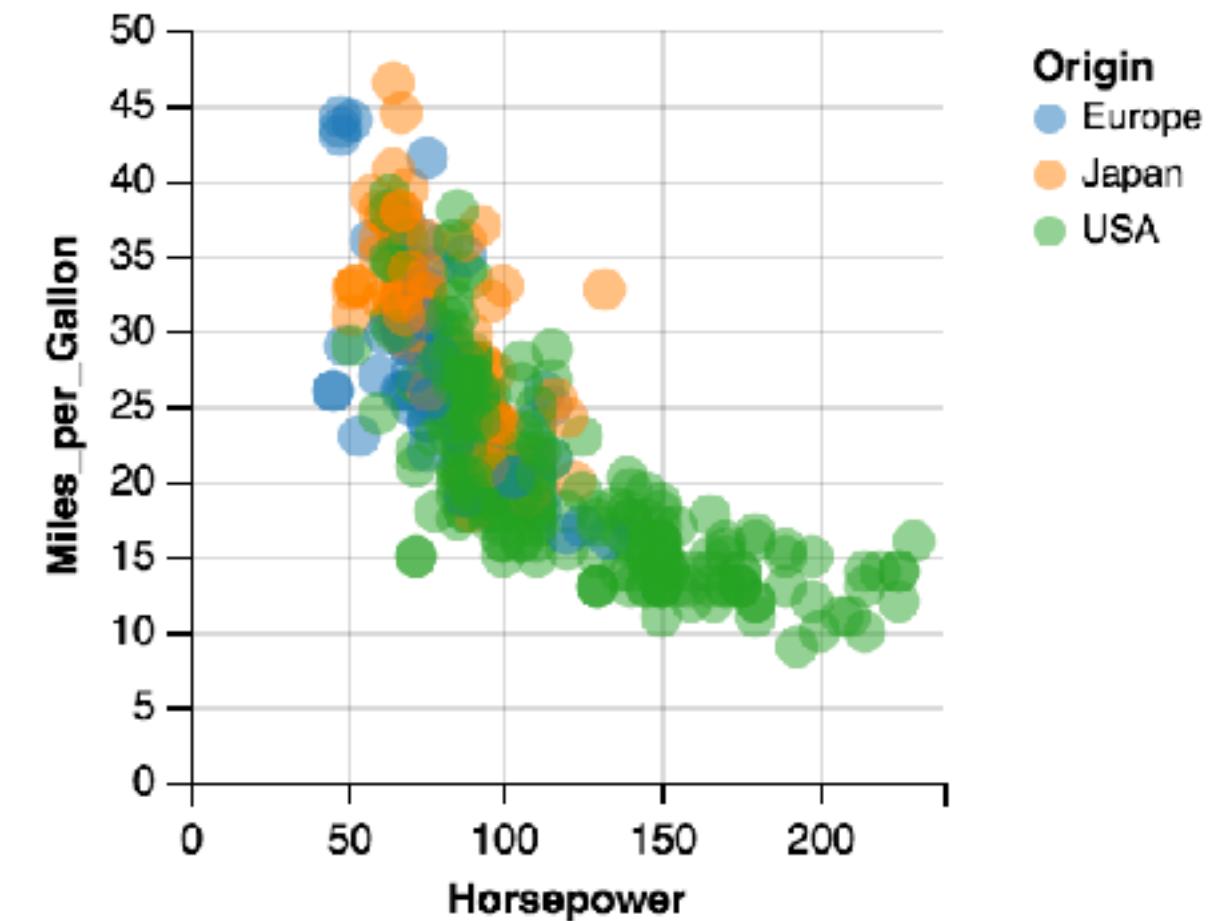
Selections

The core interactive building blocks. Define three components:

1. Event processing – how does the interaction occur?
2. Points of interest – which marks/data points were interacted with?
3. Predicate function – what is the full set of selected marks/data points?

Vega-Lite Selections

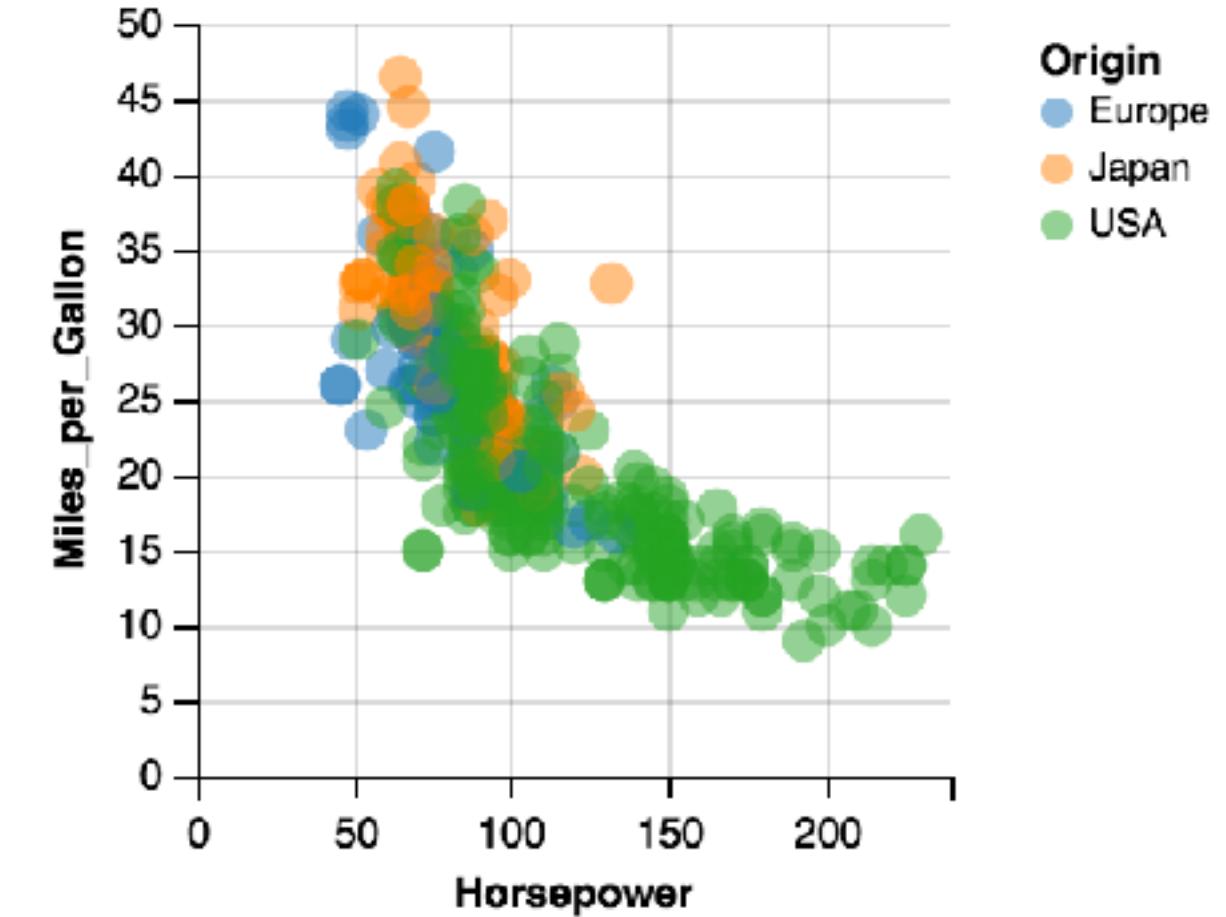
```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {field: "Origin", type: "N"}  
}
```



Selections define event processing, points of interest, and a predicate function.

Vega-Lite Selections: A Single Point

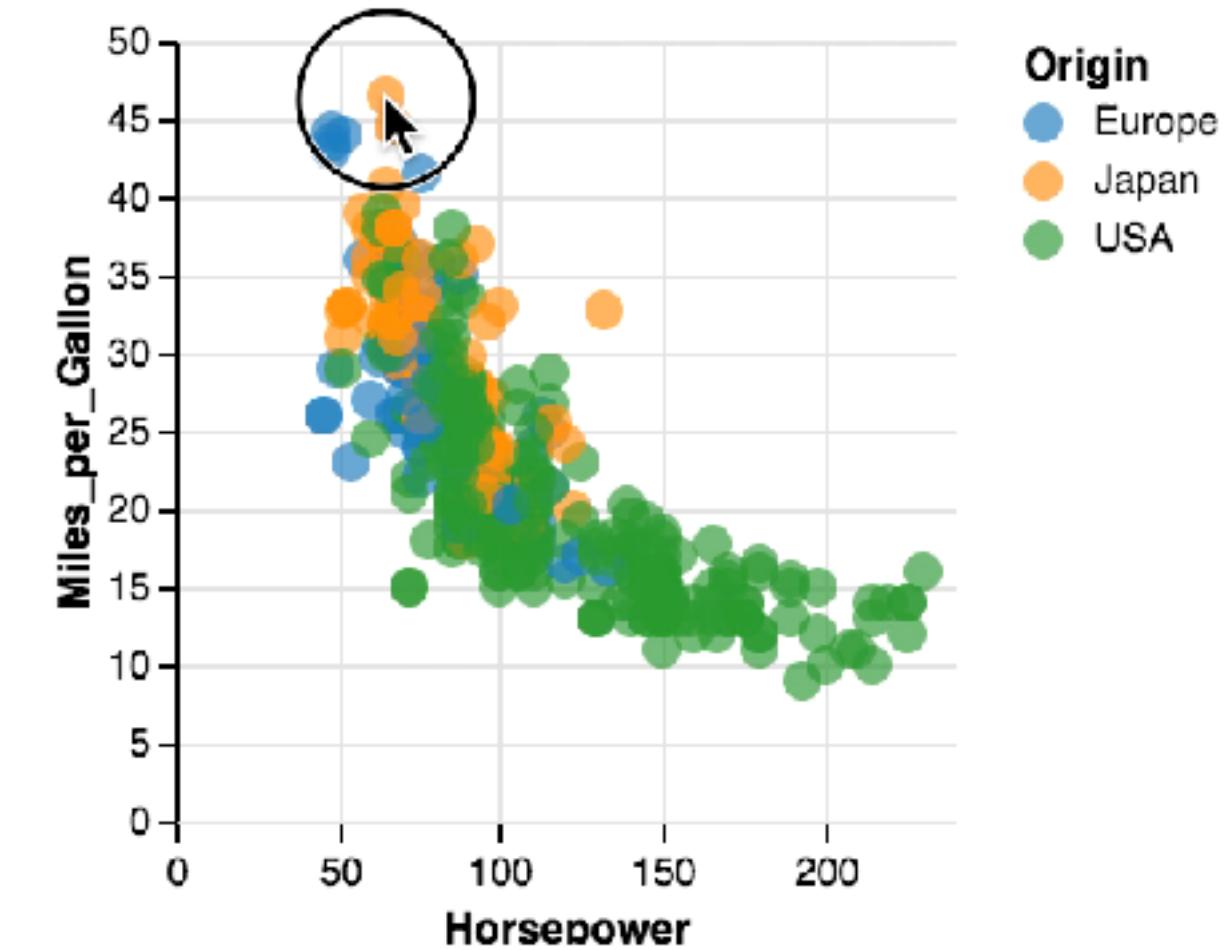
```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    picked: {type: "single"}  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {field: "Origin", type: "N"}  
  }  
}
```



Selections define event processing, points of interest, and a predicate function.

Vega-Lite Selections: A Single Point

```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    picked: {type: "single"}  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {  
      condition: {  
        selection: "!picked", value: "grey"  
      },  
      field: "Origin", type: "N"  
    }  
  }  
}
```

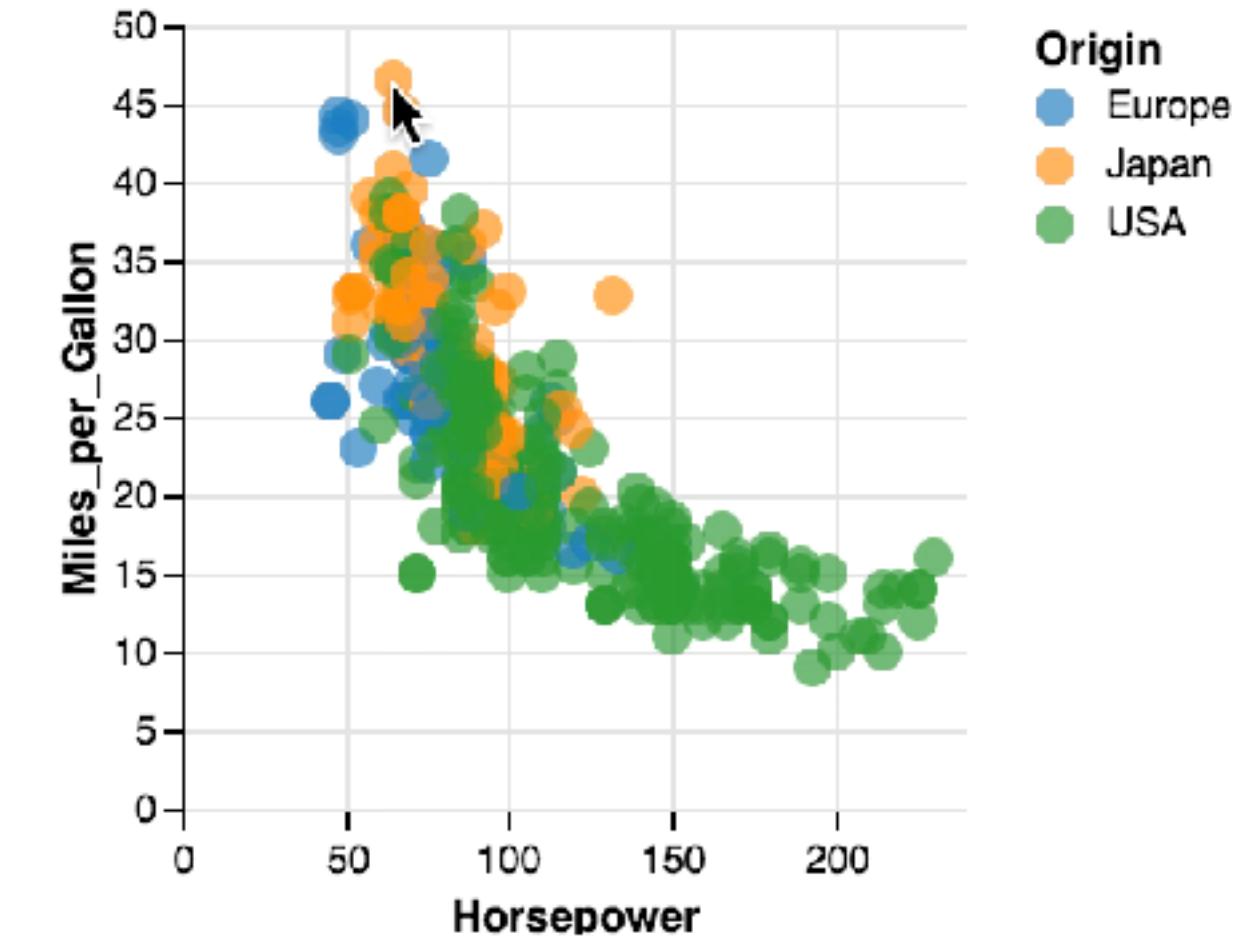


Selections define event processing, points of interest, and a predicate function.

Selection **types** provide defaults values for these three components.

Vega-Lite Selections: Multiple Points

```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    picked: {type: "multi"}  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {  
      condition: {  
        selection: "!picked", value: "grey"  
      },  
      field: "Origin", type: "N"  
    }  
  }  
}
```

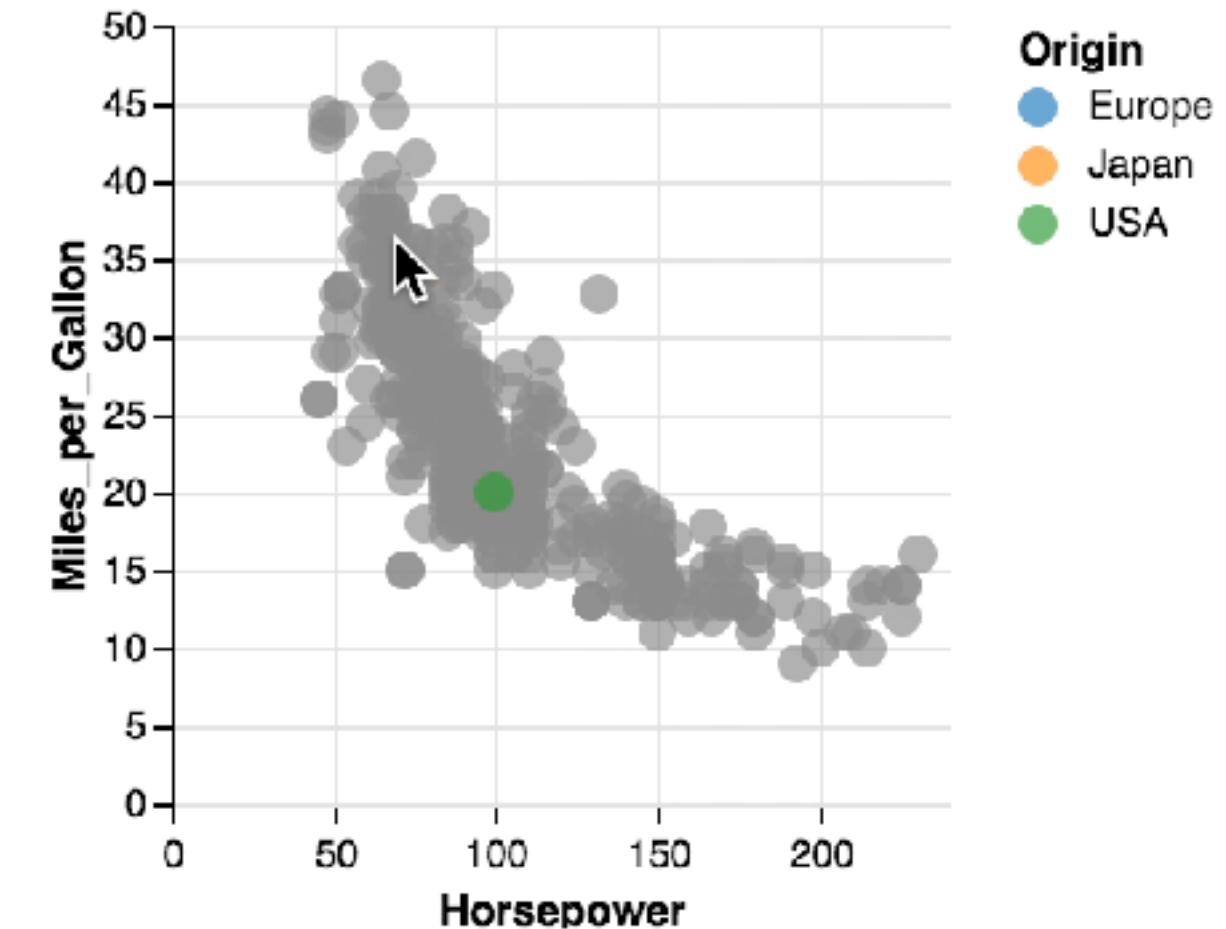


Selections define event processing, points of interest, and a predicate function.

Selection **types** provide defaults values for these three components.

Vega-Lite Selections: Multiple Points on hover

```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    picked: {type: "multi", on: "mouseover"}  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {  
      condition: {  
        selection: "!picked", value: "grey"  
      },  
      field: "Origin", type: "N"  
    }  
  }  
}
```



Selections define event processing, points of interest, and a predicate function.

Selection **types** provide defaults values for these three components.

Selections

The core interactive building blocks. Define three components:

1. Event processing – how does the interaction occur?
2. Points of interest – which marks/data points were interacted with?
3. Predicate function – what is the full set of selected marks/data points?

Selections

The core interactive building blocks. Define three components:

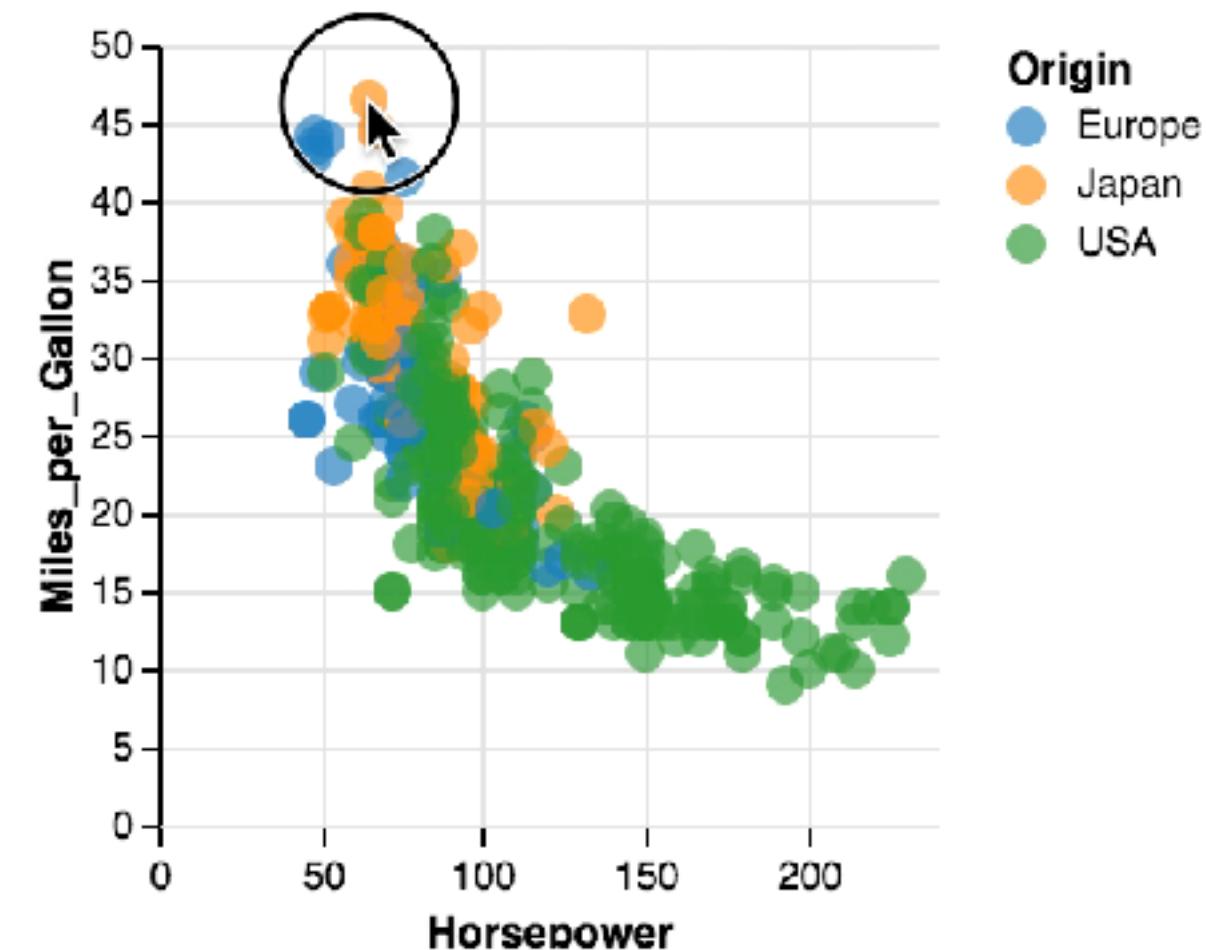
1. Event processing – how does the interaction occur?
2. Points of interest – which marks/data points were interacted with?
3. Predicate function – what is the full set of selected marks/data points?

Selection Transforms

Composable operators that modify a selection's components.

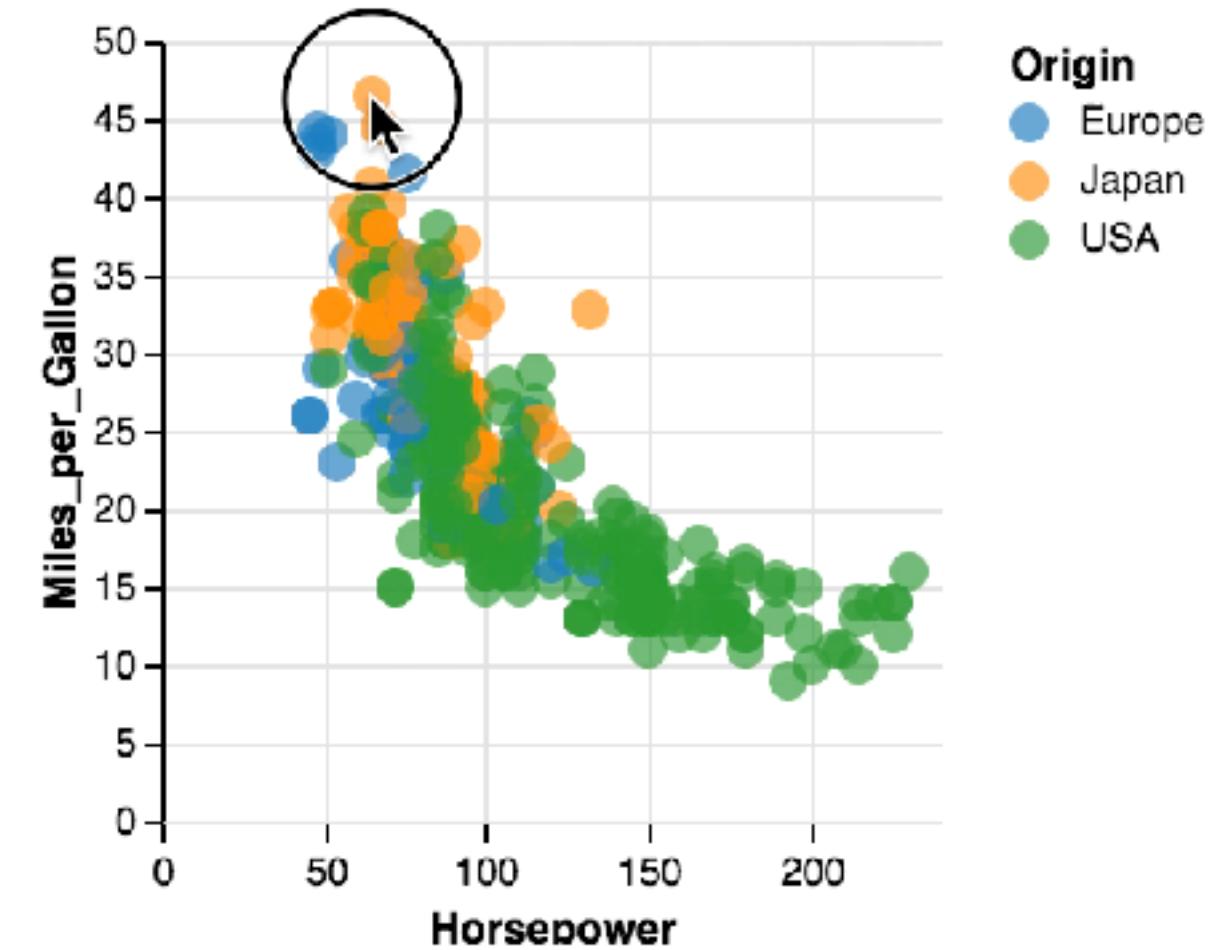
Vega-Lite Selections: A Single Point

```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    picked: {type: "single"}  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {  
      condition: {  
        selection: "!picked", value: "grey"  
      },  
      field: "Origin", type: "N"  
    }  
  }  
}
```



Vega-Lite Selections: A Single Cylinder

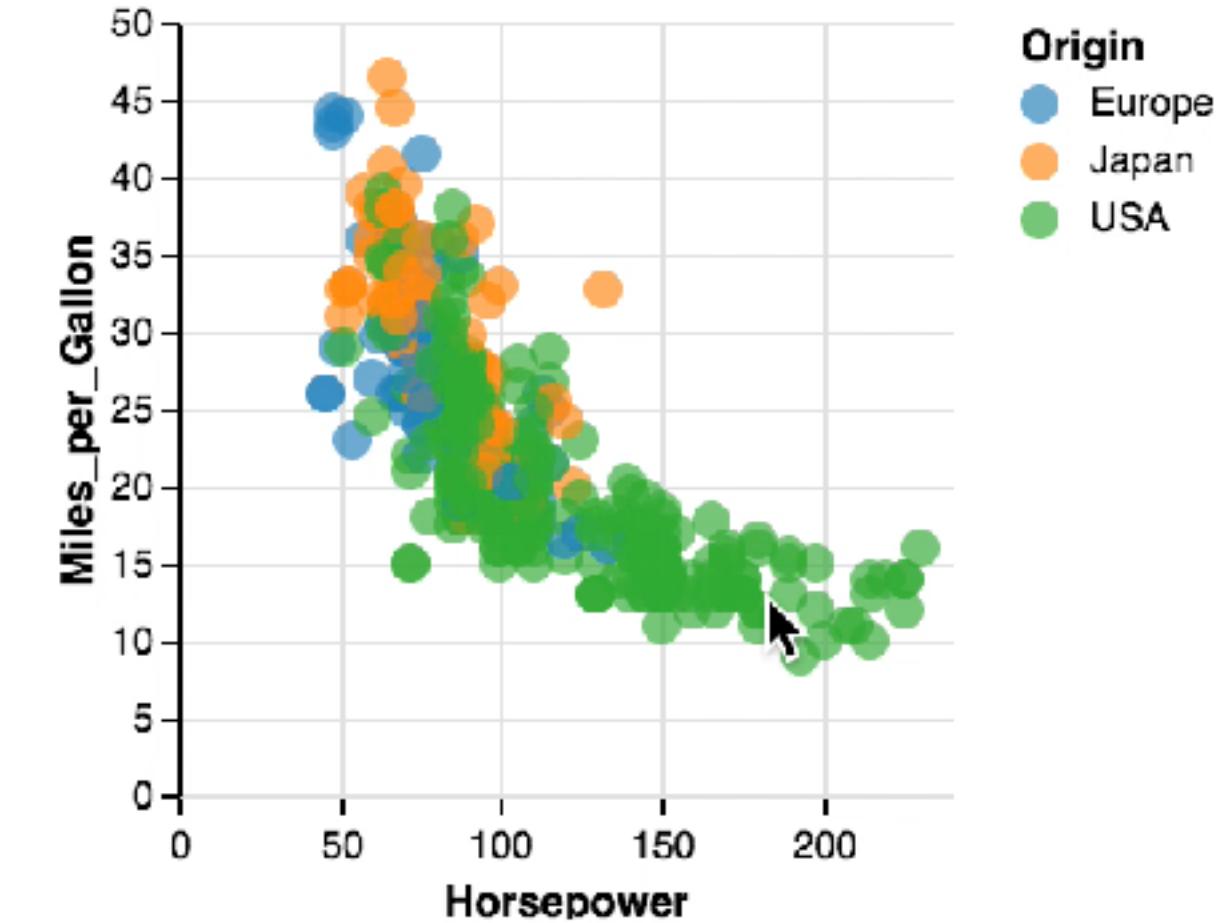
```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    picked: {type: "single"}  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {  
      condition: {  
        selection: "!picked", value: "grey"  
      },  
      field: "Origin", type: "N"  
    }  
  }  
}
```



The **project transform** rewrites the predicate to match on **fields** or encodings.

Vega-Lite Selections: A Single Cylinder

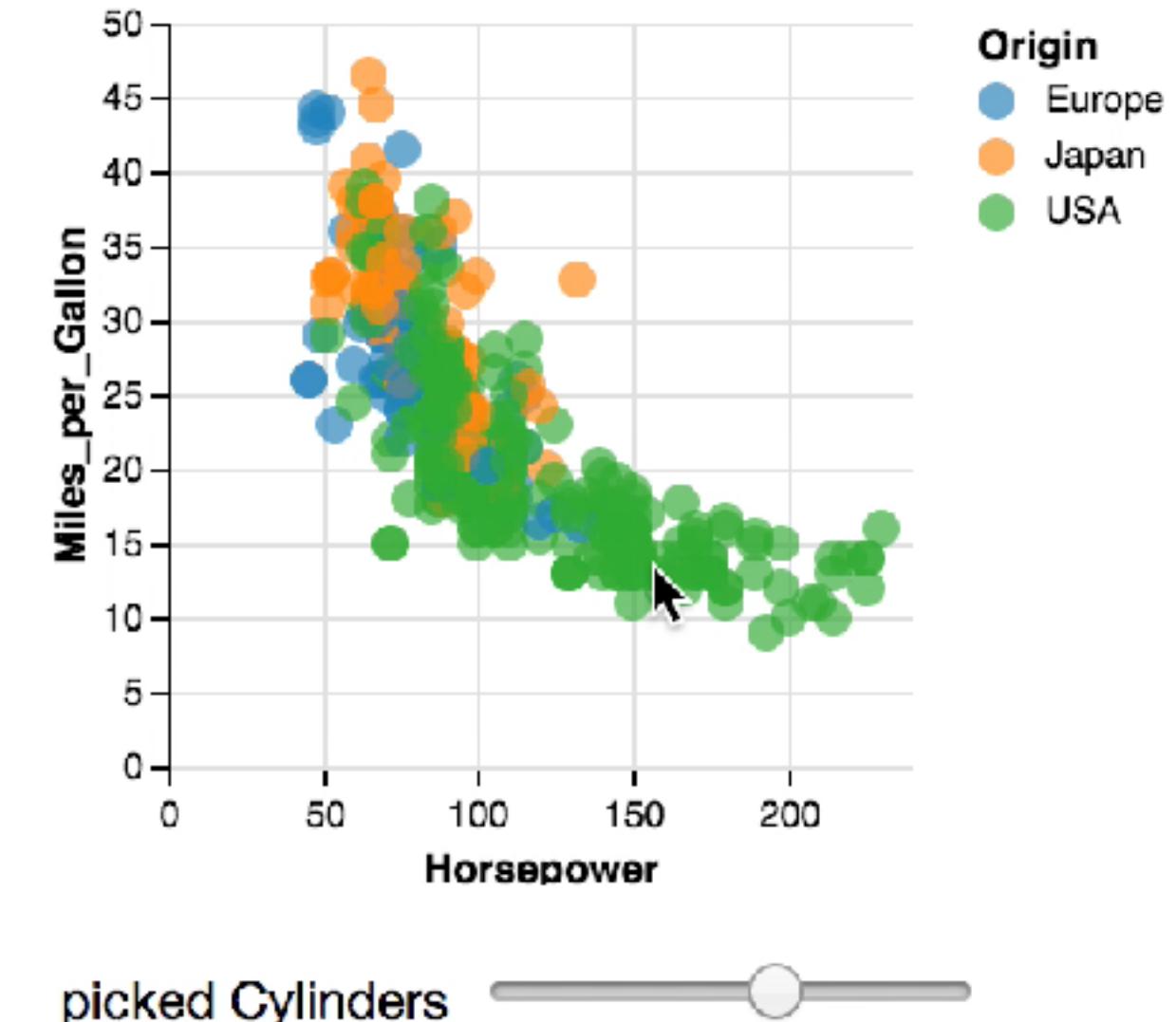
```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    picked: {  
      type: "single", fields: ["Cylinders"]  
    }  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {  
      condition: {  
        selection: "!picked", value: "grey"  
      },  
      field: "Origin", type: "N"  
    }  
  }  
}
```



The **project transform** rewrites the predicate to match on **fields** or encodings.

Vega-Lite Selections: A Single Cylinder & Year

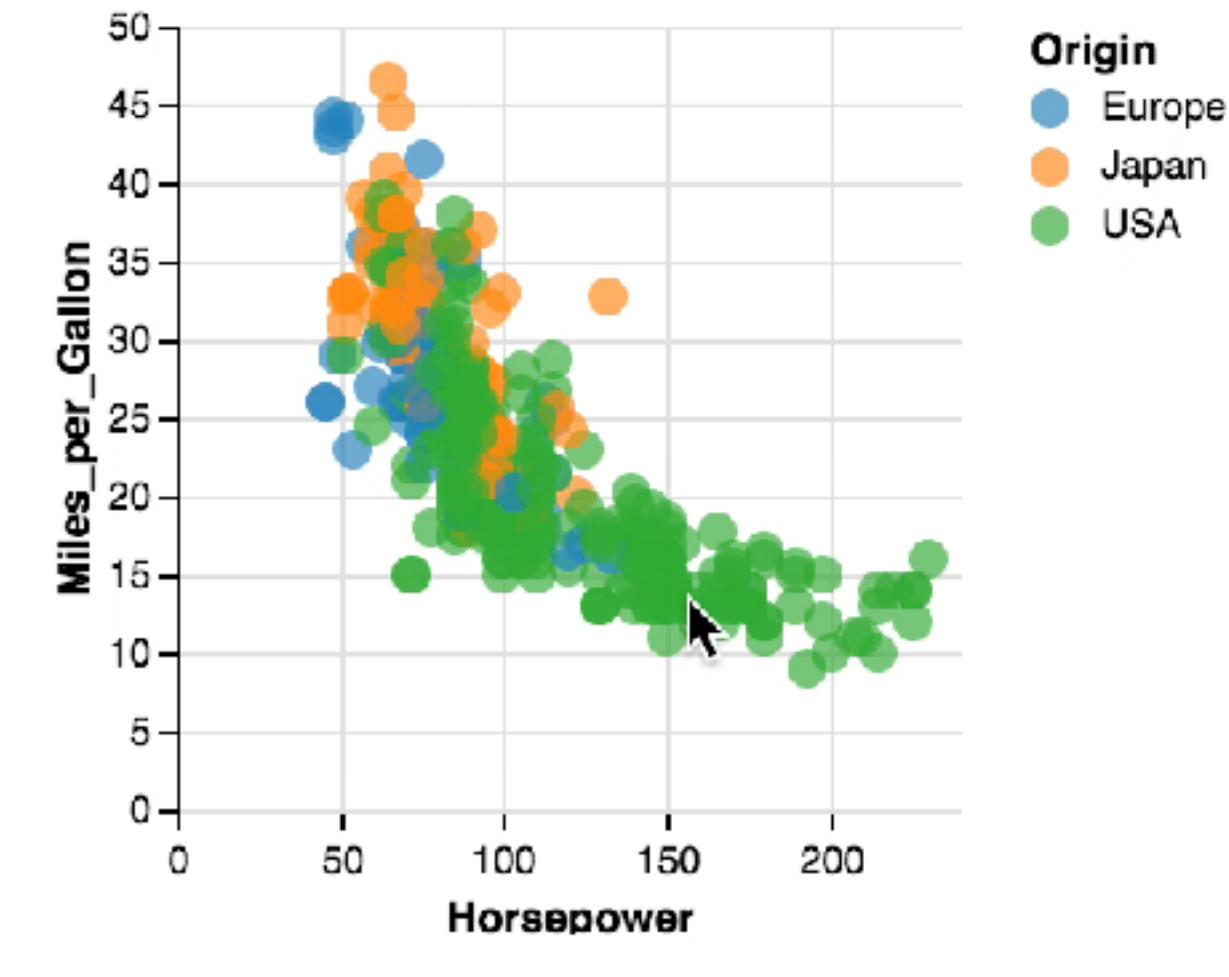
```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    picked: {  
      type: "single", fields: ["Cylinders"],  
      bind: {input: "range", min: 3, ...}  
    }  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {  
      condition: {  
        selection: "!picked", value: "grey"  
      },  
      field: "Origin", type: "N"  
    }  
  }  
}
```



The **bind transform** drives selections via **query widgets** and scale functions.

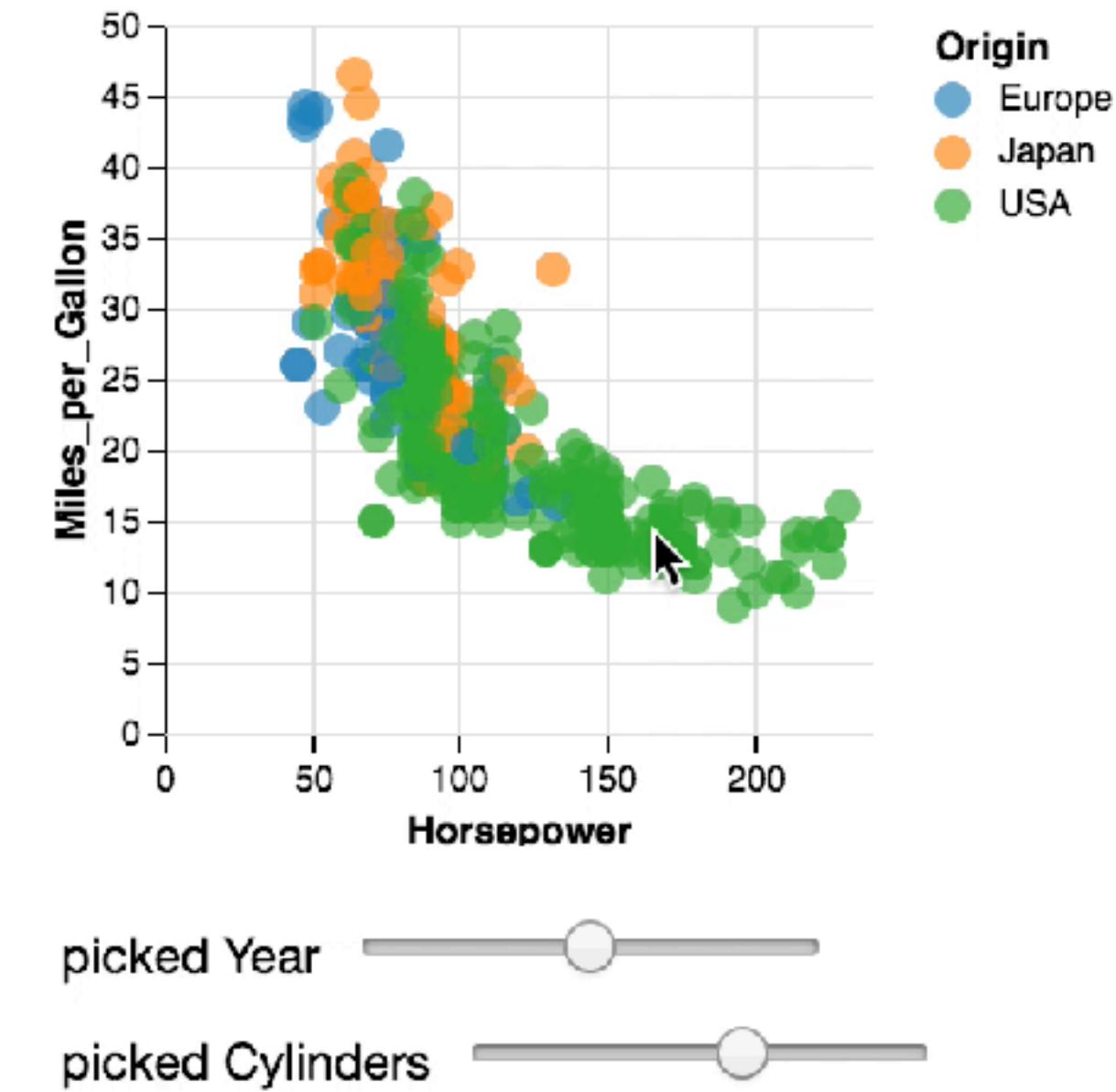
Vega-Lite Selections: A Single Cylinder & Year

```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    picked: {  
      type: "single",  
      fields: ["Cylinders"],  
      bind: {  
        input: "range", min: 3, ...  
      }  
    }  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {  
      condition: {  
        selection: "!picked", value: "grey"  
      },  
      field: "Origin", type: "N"  
    }  
  }  
}
```



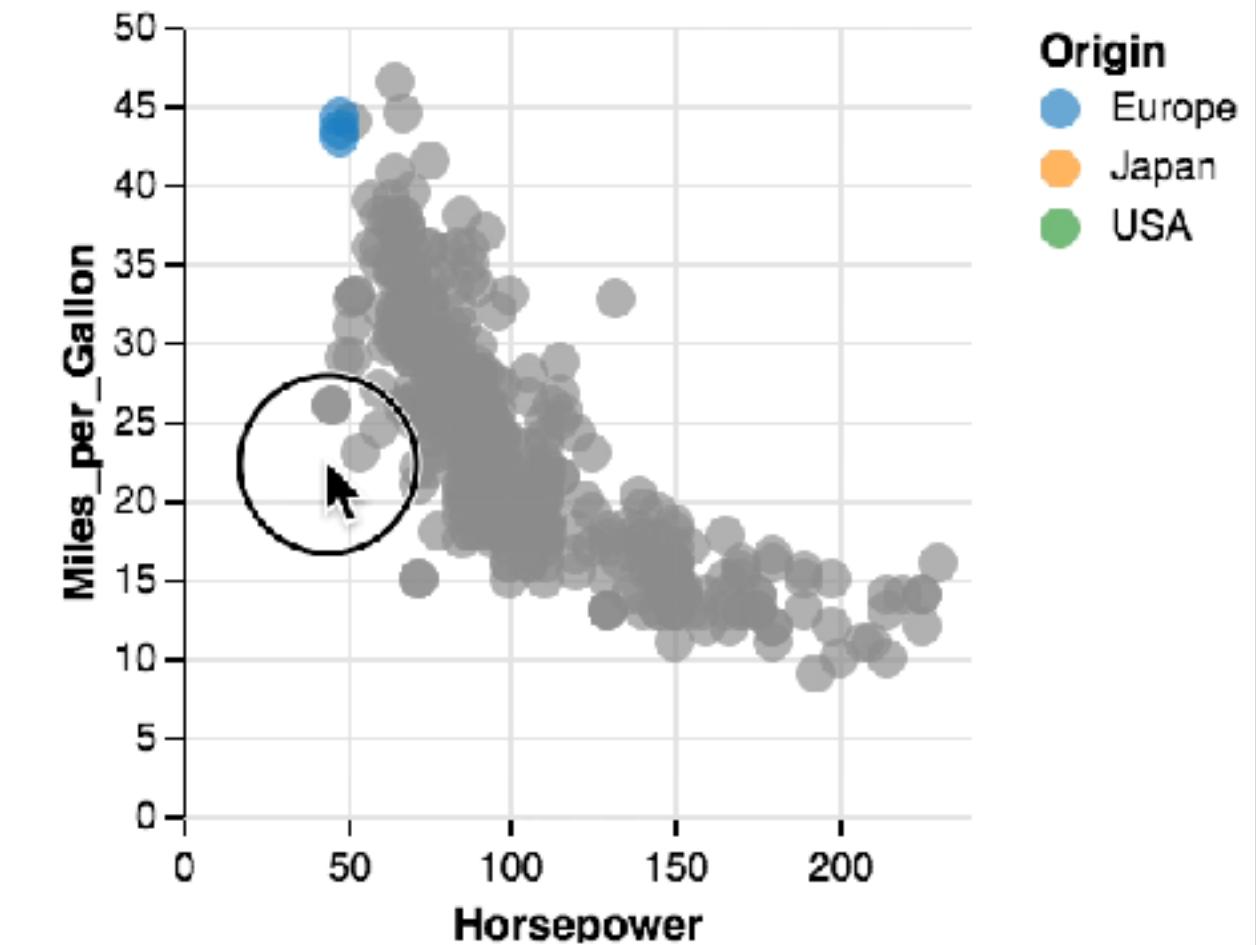
Vega-Lite Selections: A Single Cylinder & Year

```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    picked: {  
      type: "single",  
      fields: ["Cylinders", "Year"],  
      bind: {  
        Cylinders: {input: "range", min: 3},  
        Year: {input: "range", min: 1967}  
      }  
    }  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {  
      condition: {  
        selection: "!picked", value: "grey"  
      },  
      field: "Origin", type: "N"  
    }  
  }  
}
```



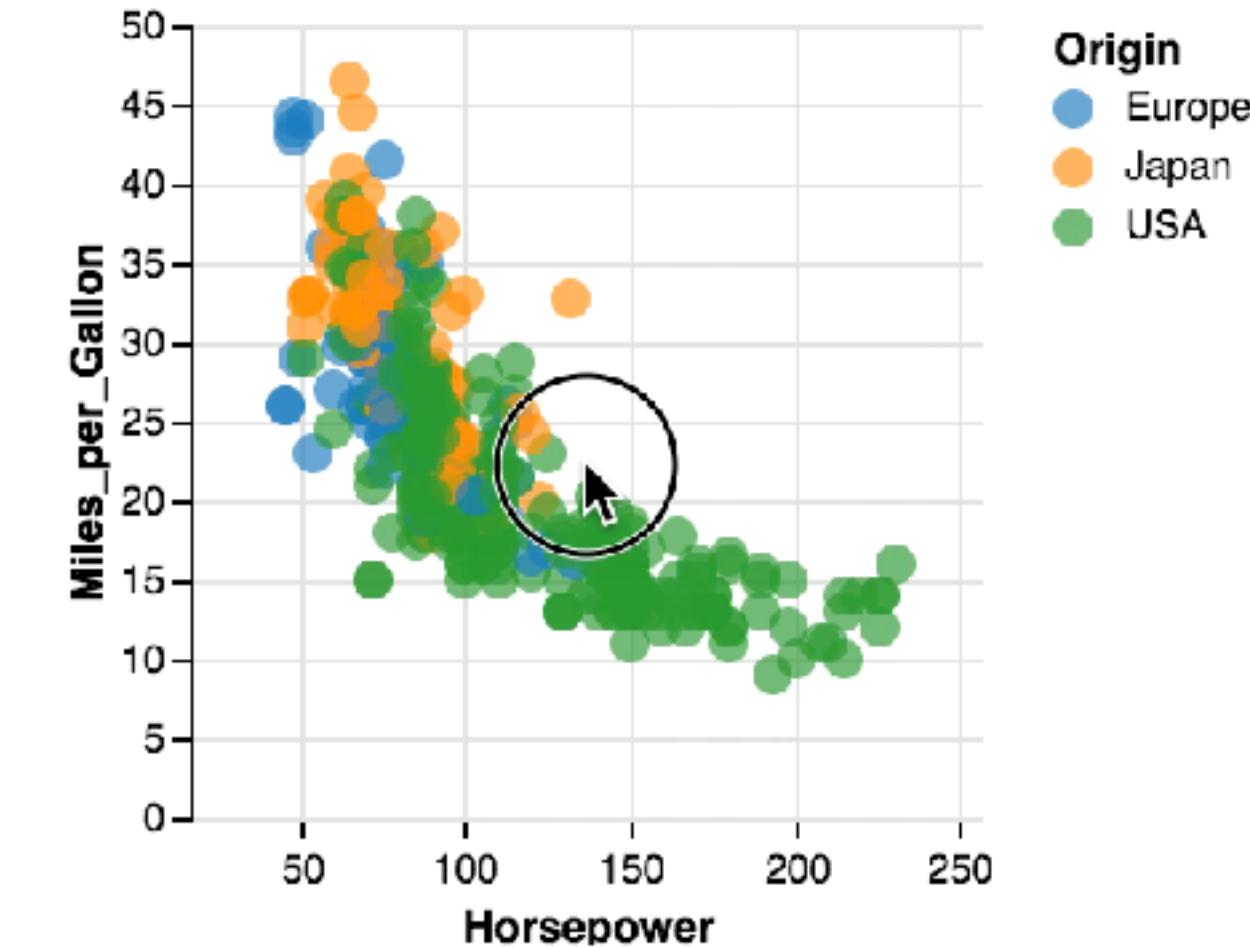
Vega-Lite Selections: Continuous Region

```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    grid: {  
      type: "interval",  
      encodings: ["x"]  
    }  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {  
      condition: {  
        selection: "!grid", value: "grey"  
      },  
      field: "Origin", type: "N"  
    }  
  }  
}
```



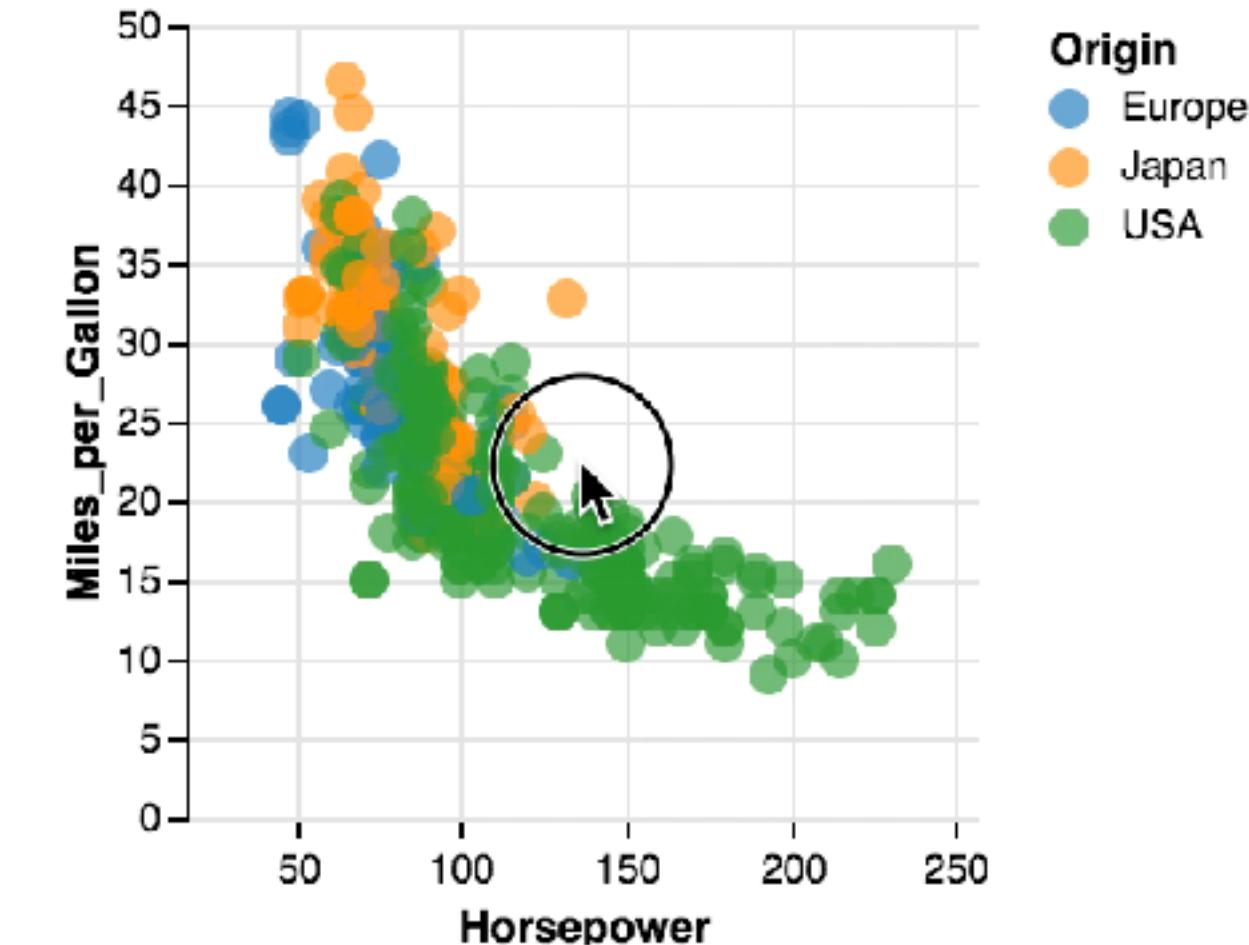
Vega-Lite Selections: Continuous Region

```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    grid: {  
      type: "interval",  
      encodings: ["x"],  
      bind: "scales"  
    }  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {  
      condition: {  
        selection: "!grid", value: "grey"  
      },  
      field: "Origin", type: "N"  
    }  
  }  
}
```



Vega-Lite Selections: Continuous Region

```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    grid: {  
      type: "interval",  
      encodings: ["x"],  
      bind: "scales"  
    }  
  },  
  encoding: {  
    x: {  
      field: "Horsepower", type: "Q",  
      scale: {domain: {selection: "grid"}}  
    },  
    y: {field: "Miles_per_Gallon", type: "Q"},  
  }  
}
```



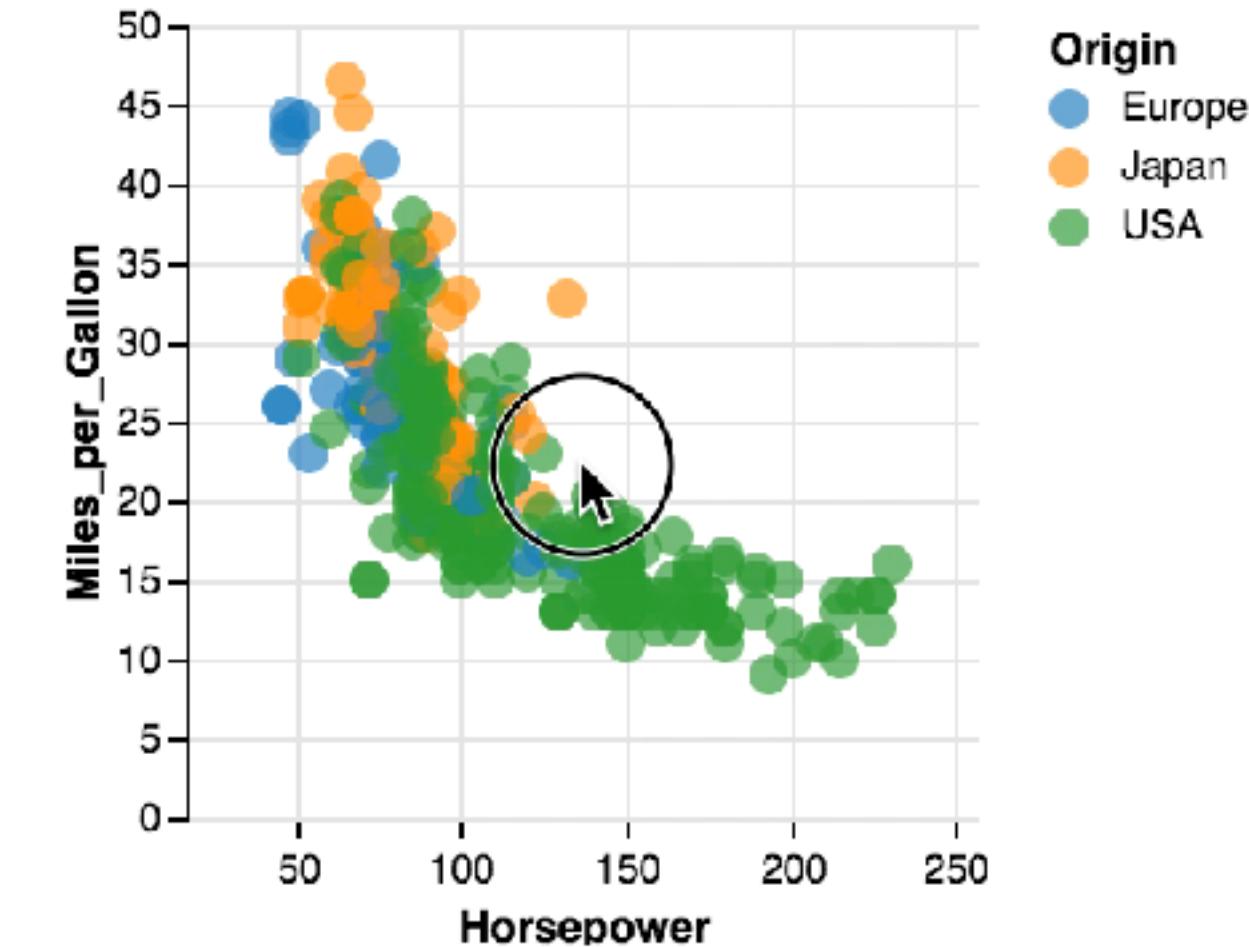
Two-way binding between a selection and scales:

Selection is **populated** with scale domains.

Selection now **drives** scale domains.

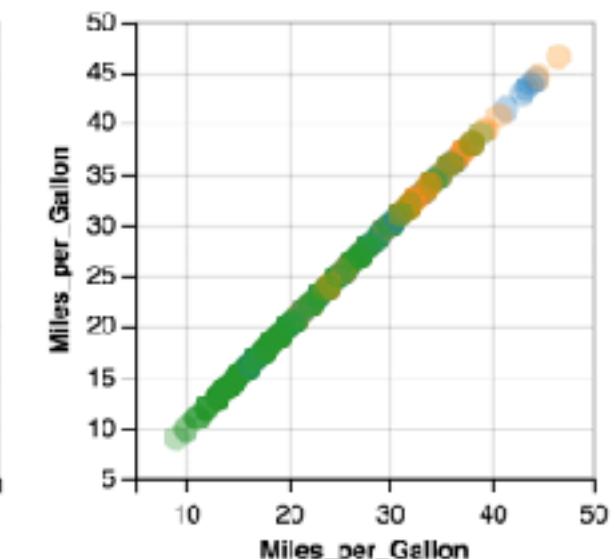
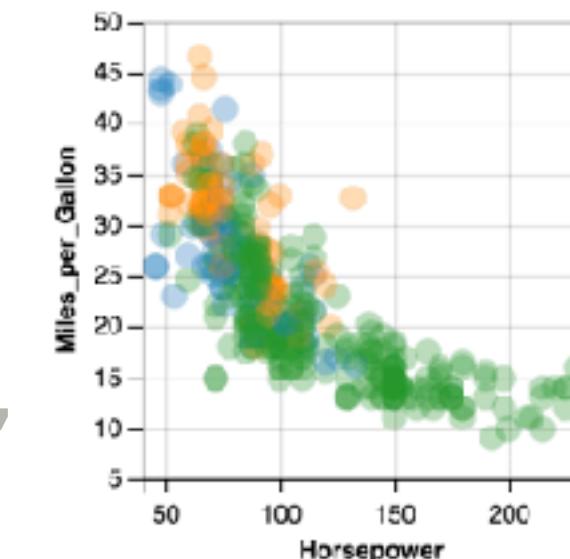
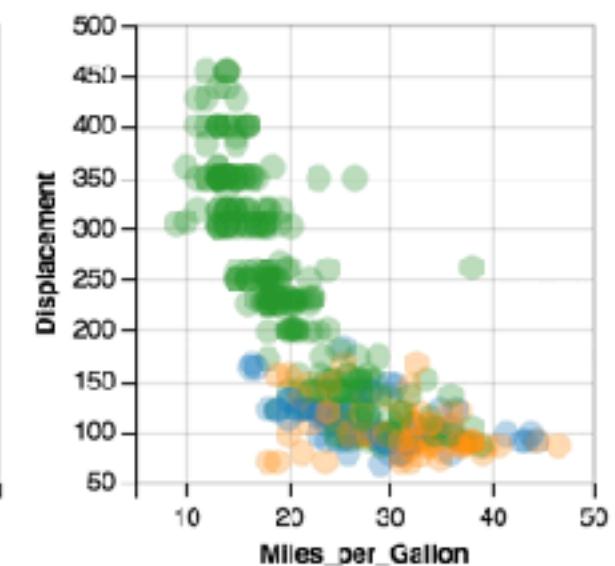
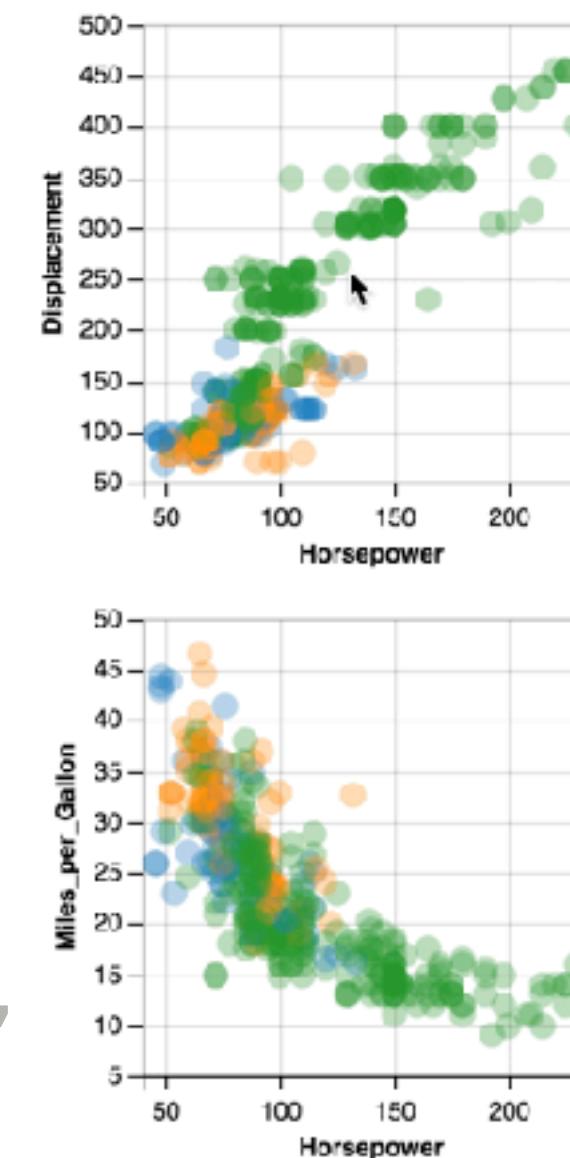
Vega-Lite Selections: Continuous Region

```
{  
  data: {url: "data/cars.json"},  
  mark: "circle",  
  selection: {  
    grid: {  
      type: "interval",  
      encodings: ["x"],  
      bind: "scales"  
    }  
  },  
  encoding: {  
    x: {field: "Horsepower", type: "Q"},  
    y: {field: "Miles_per_Gallon", type: "Q"},  
    color: {  
      condition: {  
        selection: "!grid", value: "grey"  
      },  
      field: "Origin", type: "N"  
    }  
  }  
}
```



Vega-Lite Brushing & Linking

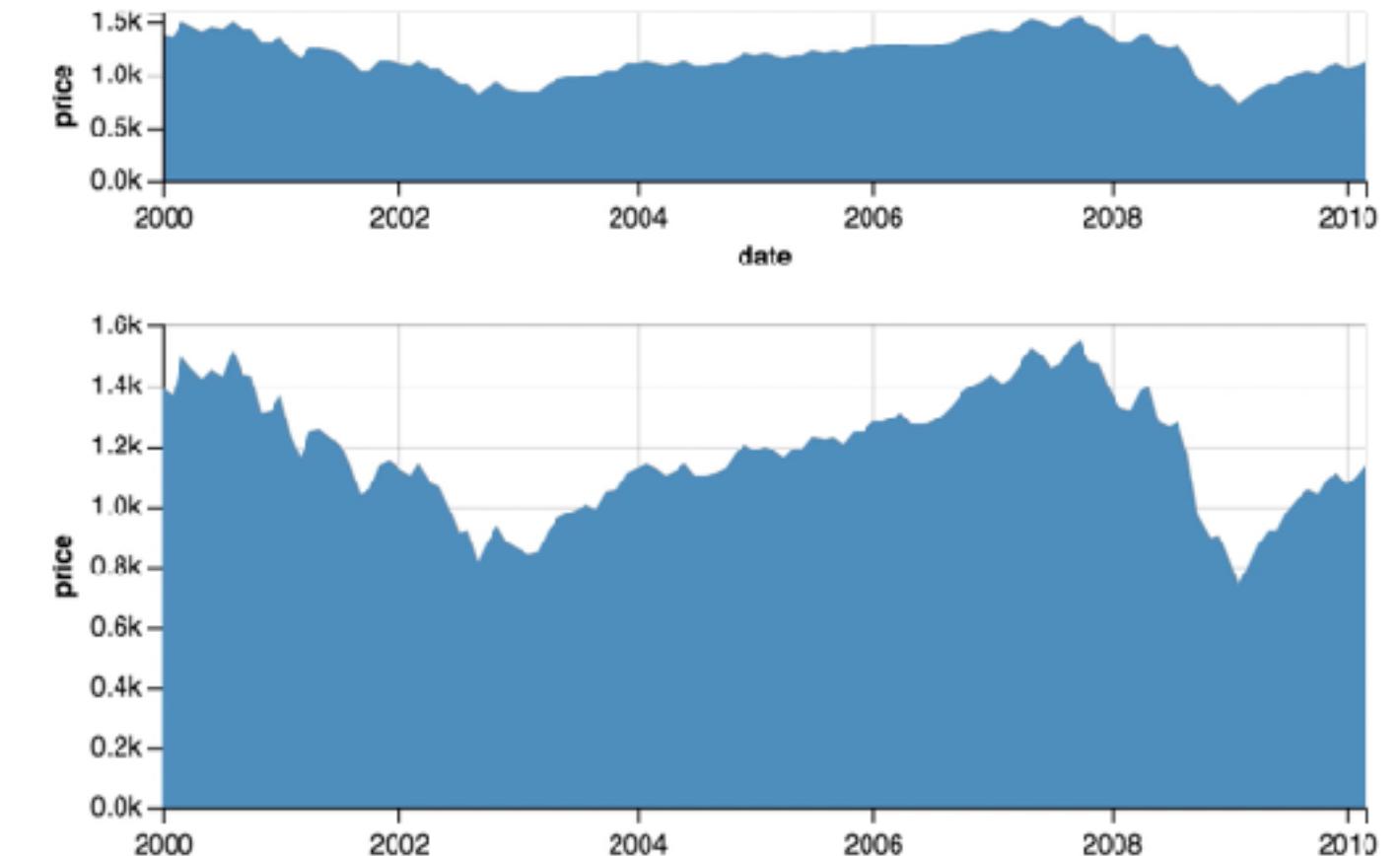
```
{  
  data: {url: "data/cars.json"},  
  repeat: {  
    row: ["Displacement", "Miles_per_Gallon"],  
    column: ["Horsepower", "Miles_per_Gallon"]  
  },  
  spec: {  
    mark: "circle",  
    selection: {  
      grid: {  
        type: "interval",  
        encodings: ["x"],  
        bind: "scales"  
      }  
    },  
    encoding: {  
      x: {field: "Horsepower", type: "Q"},  
      y: {field: "Miles_per_Gallon", type: "Q"},  
      color: {  
        condition: {  
          selection: "!grid", value: "grey"  
        },  
        field: "Origin", type: "N"  
      }  
    }  
  }  
}
```



Origin
Europe
Japan
USA

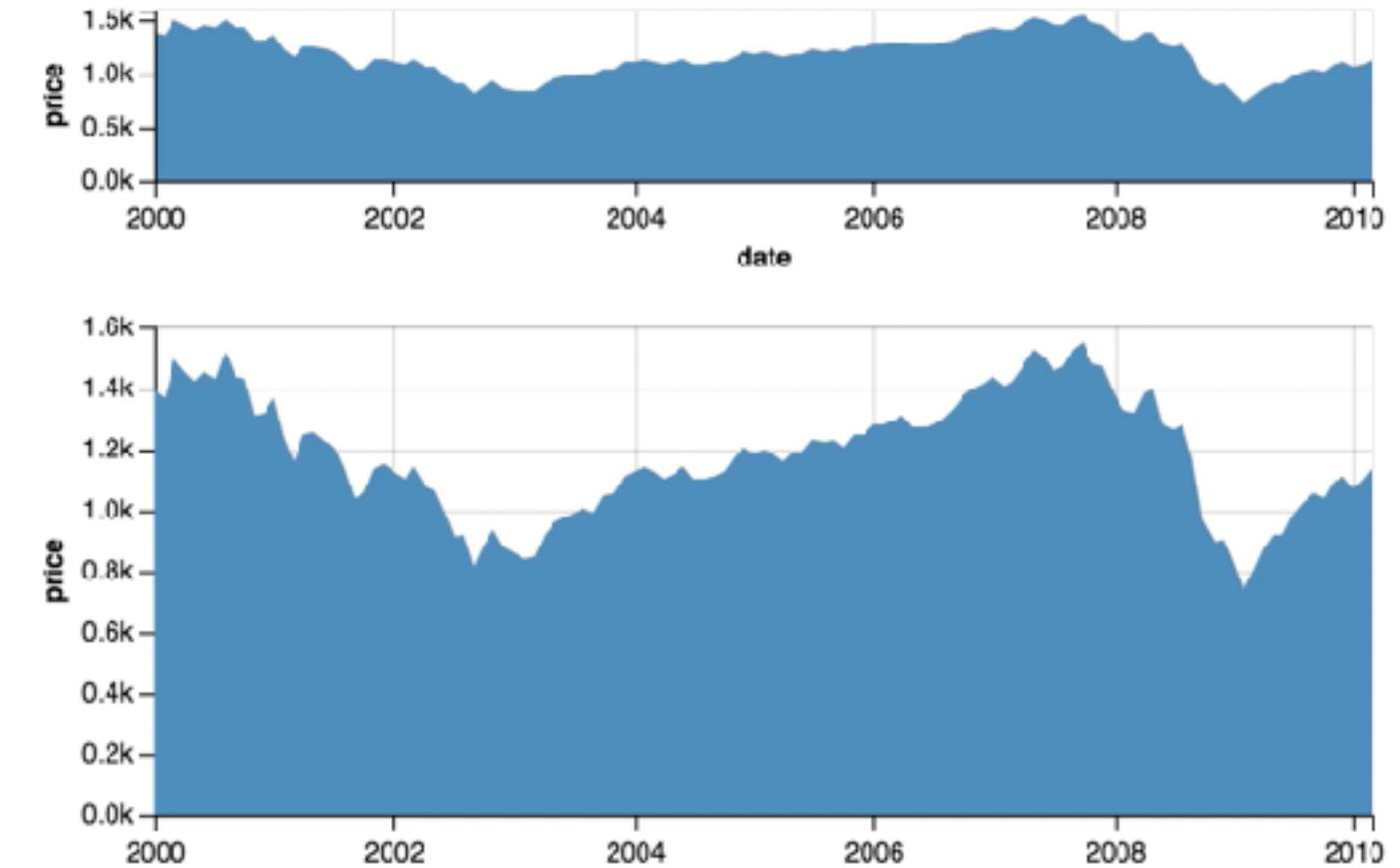
Vega-Lite Overview+Detail

```
{  
  data: {url: "data/sp500.csv", ...},  
  vconcat: [{  
    mark: "area",  
    encoding: {  
      x: {field: "date", type: "T", ...},  
      y: {field: "price", type: "Q", ...}  
    }  
  }, {  
    mark: "area",  
    encoding: {  
      x: {field: "date", type: "T", ...},  
      y: {field: "price", type: "Q", ...}  
    }  
  }]  
}
```



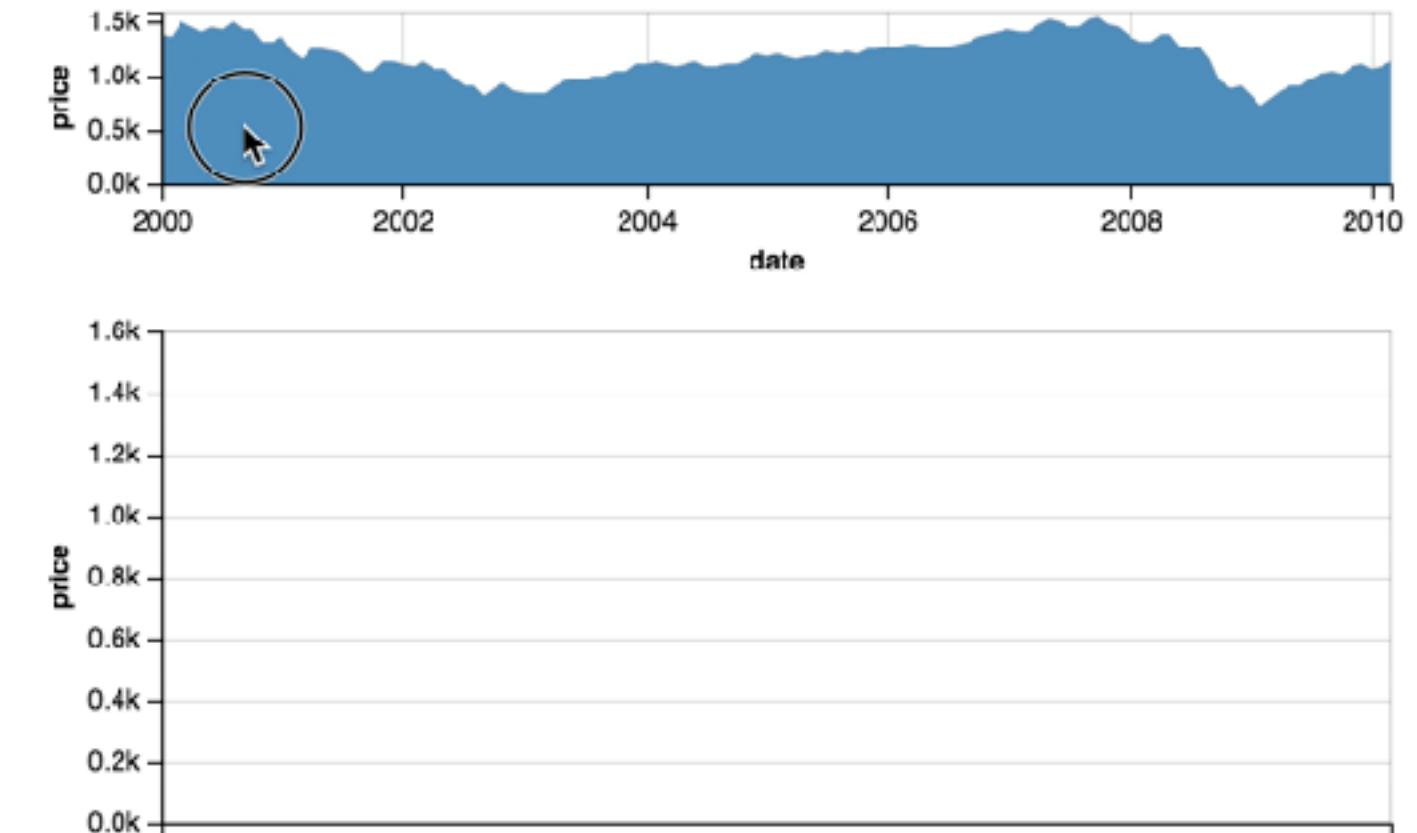
Vega-Lite Overview+Detail

```
{  
  data: {url: "data/sp500.csv", ...},  
  vconcat: [{  
    mark: "area",  
    selection: {  
      region: {  
        type: "interval", encodings: ["x"]  
      }  
    },  
    ...  
  }, {  
    mark: "area",  
    encoding: {  
      x: {  
        field: "date", type: "T", ...  
      },  
      y: {field: "price", type: "Q", ...}  
    }  
  }]
```



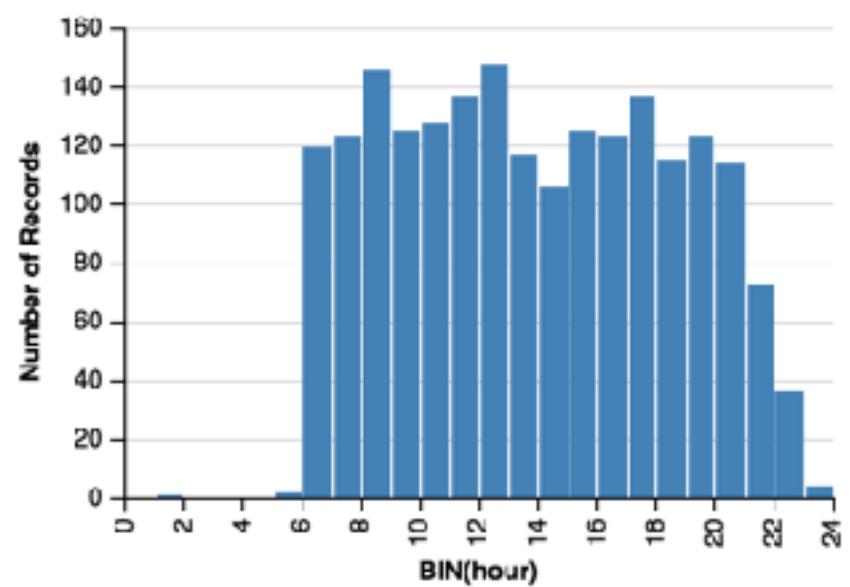
Vega-Lite Overview+Detail

```
{  
  data: {url: "data/sp500.csv", ...},  
  vconcat: [{  
    mark: "area",  
    selection: {  
      region: {  
        type: "interval", encodings: ["x"]  
      }  
    },  
    ...  
  }, {  
    mark: "area",  
    encoding: {  
      x: {  
        field: "date", type: "T",  
        scale: {domain: {selection: "region"}}  
      },  
      ...  
    },  
    y: {field: "price", type: "Q", ...}  
  }]
```



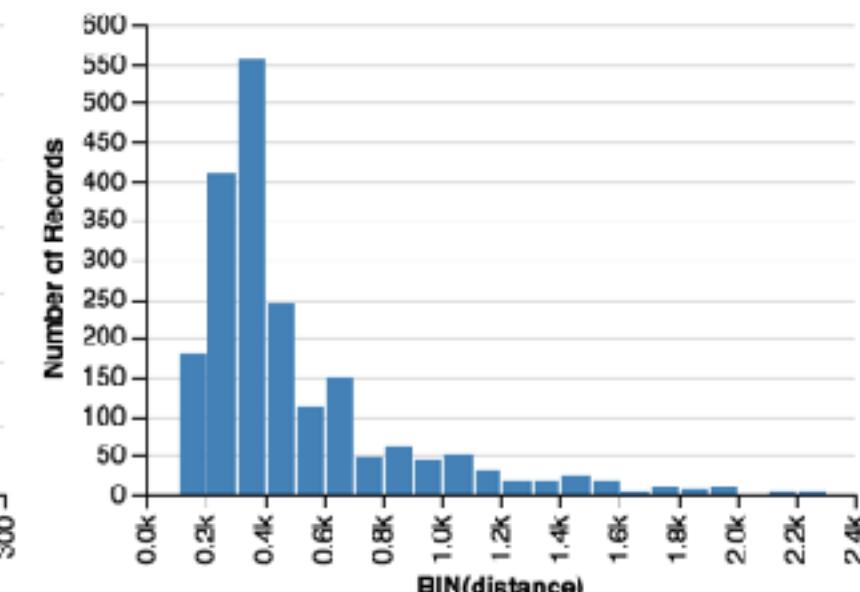
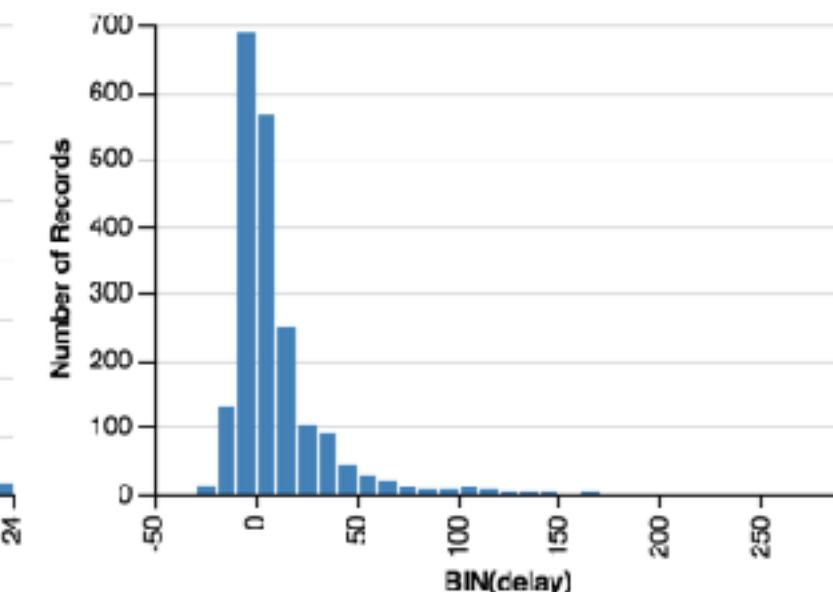
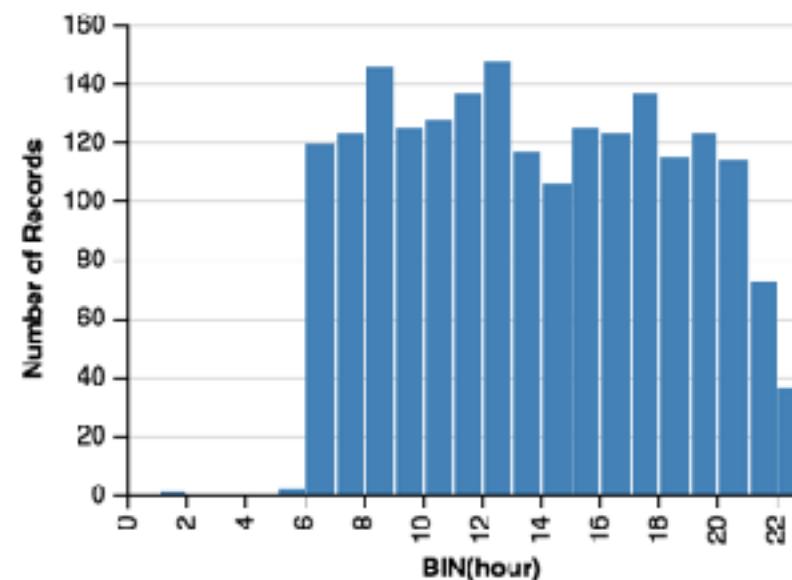
Vega-Lite Layered Cross Filtering

```
{  
  data: {url: "data/flights.json"},  
  mark: "bar",  
  encoding: {  
    x: {field: "hour", type: "Q", bin: true},  
    y: {aggregate: "count", type: "Q"}  
}
```



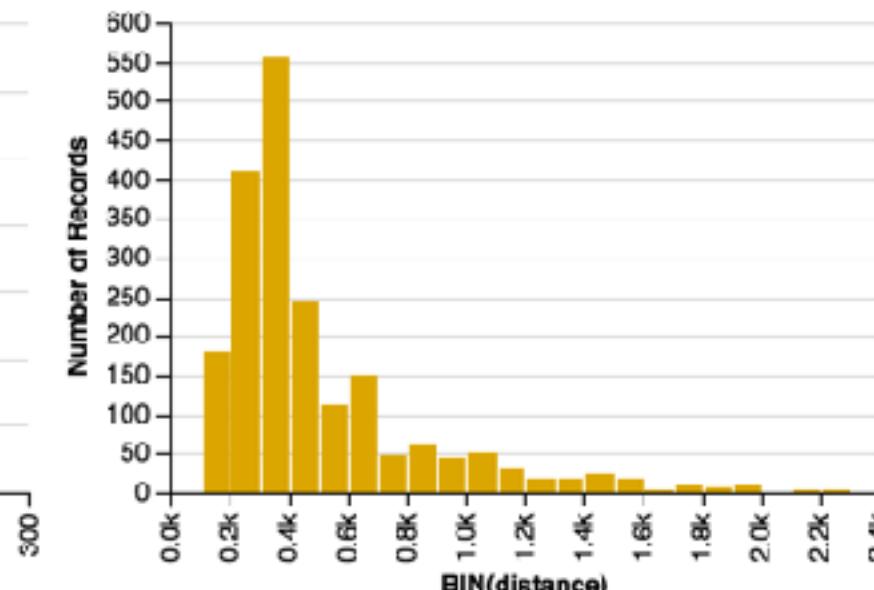
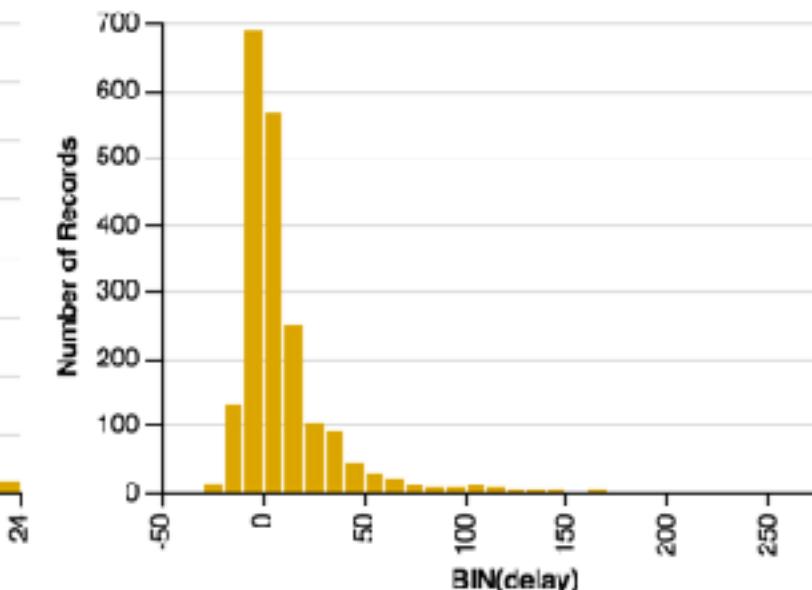
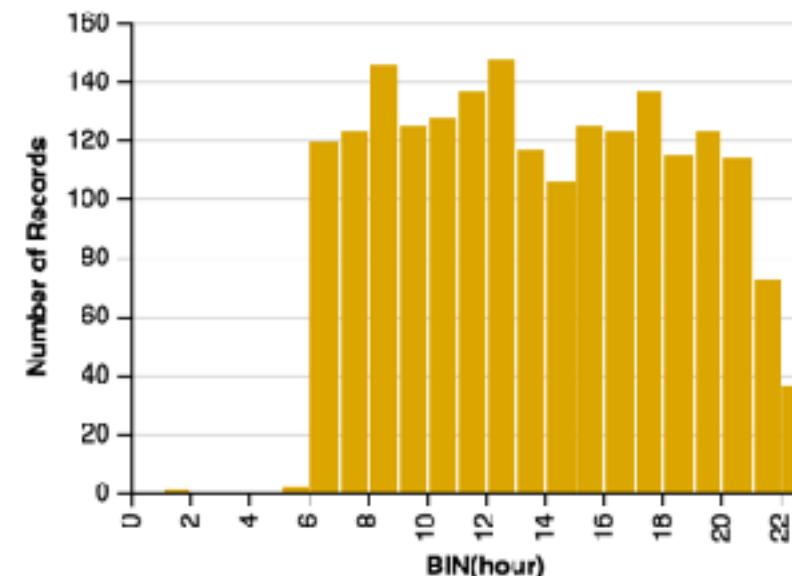
Vega-Lite Layered Cross Filtering

```
{  
  repeat: {column: ["hour", "delay", "distance"]},  
  spec: {  
    data: {url: "data/flights.json"},  
    mark: "bar",  
    encoding: {  
      x: {field: {repeat: "column"}, type: "Q", bin: true},  
      y: {aggregate: "count", field: "*", type: "Q"}  
    }  
  }  
}
```



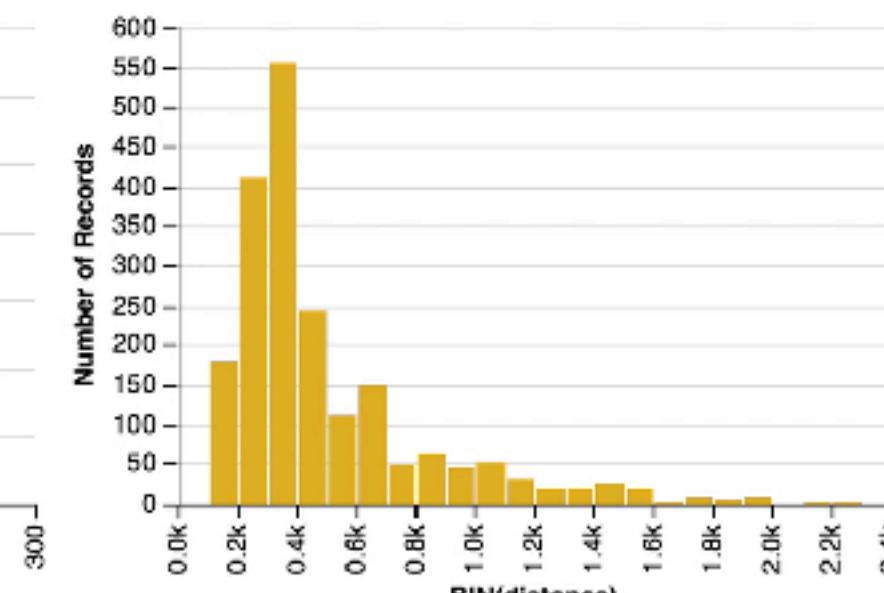
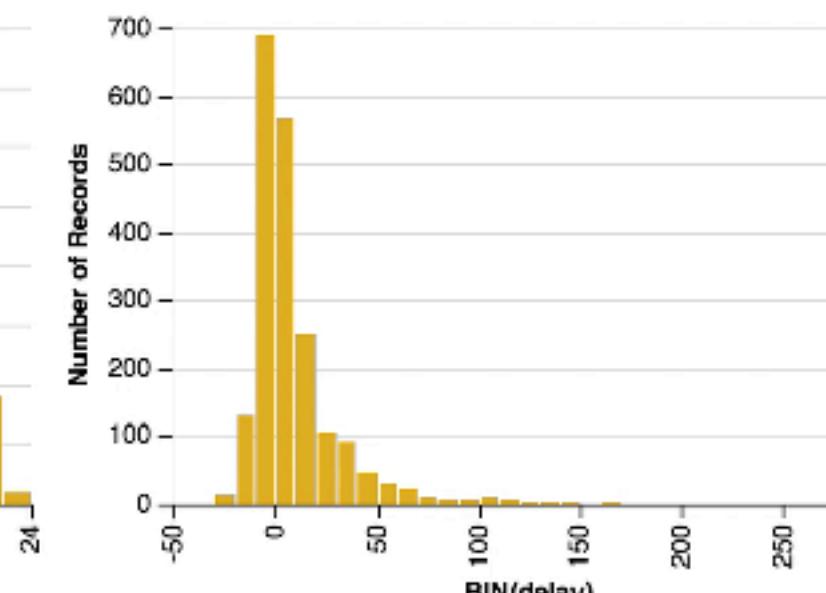
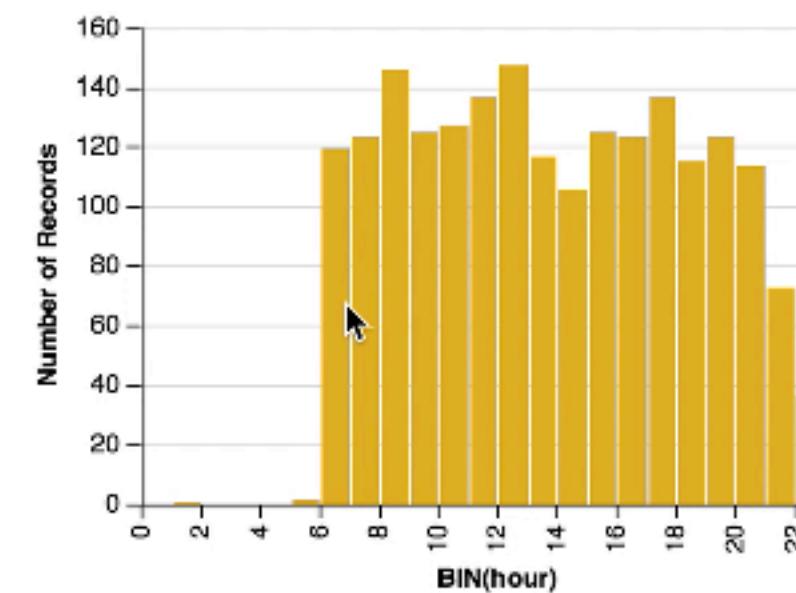
Vega-Lite Layered Cross Filtering

```
{  
  repeat: {column: ["hour", "delay", "distance"]},  
  spec: {  
    layer: [{  
      data: {url: "data/flights.json"},  
      mark: "bar",  
      encoding: {  
        x: {field: {repeat: "column"}, type: "Q", bin: true},  
        y: {aggregate: "count", type: "Q"}  
      }  
    }, {  
      ...,  
      color: {value: "goldenrod"}  
    }]  
  }  
}
```



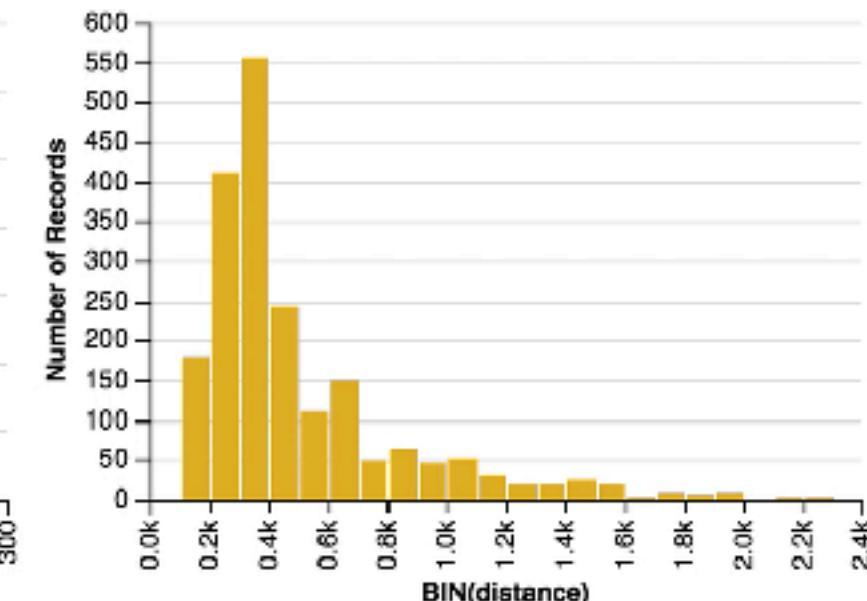
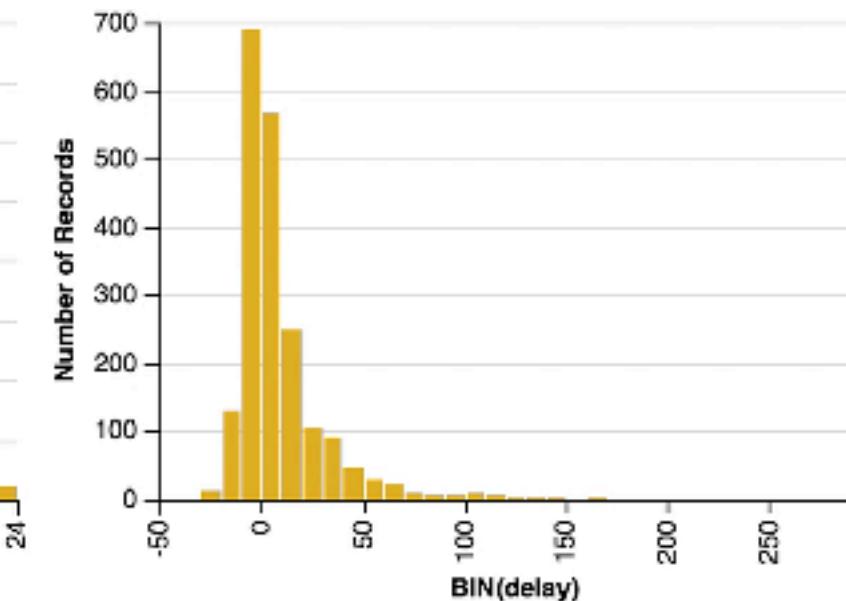
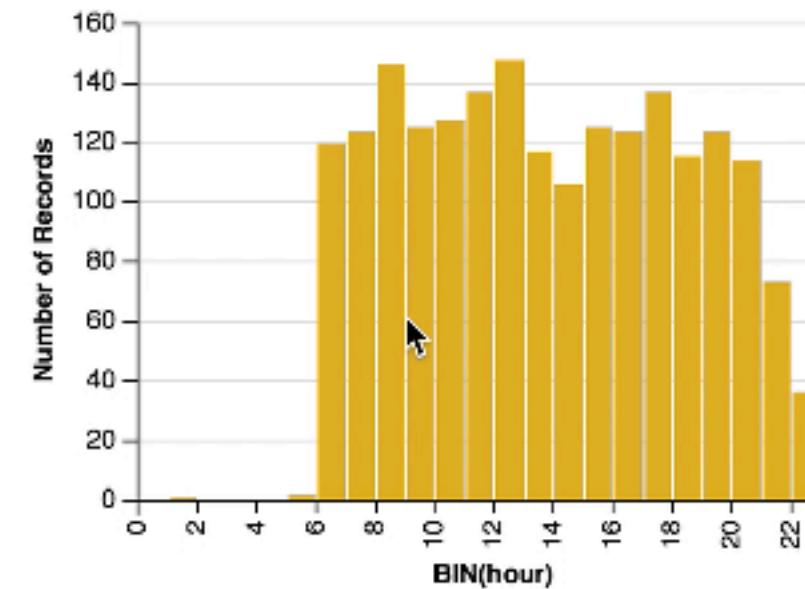
Vega-Lite Layered Cross Filtering

```
{  
  repeat: {column: ["hour", "delay", "distance"]},  
  spec: {  
    layer: [{  
      ...  
      selection: {  
        region: {type: "interval", encodings: ["x"]}  
      }  
    }, {  
      ...  
    }]  
  }  
}
```



Vega-Lite Layered Cross Filtering

```
{  
  repeat: {column: ["hour", "delay", "distance"]},  
  spec: {  
    layer: [{  
      ...,  
      selection: {  
        region: {type: "interval", encodings: ["x"]}  
      },  
      ...  
    }, {  
      ...,  
      transform: [{filter: {selection: "region"} }]  
    }]  
  }  
}
```



35 Lines
of JSON!

Vega-Lite: a Grammar of Interactive Graphics

The Design of Vega-Lite

Single View Specification

Layered and Multi-view Composition

Interactions with Selections

Using Vega-Lite

Programming with Vega-Lite

Higher-level Tools and Recommendations

Vega-Lite: a Grammar of Interactive Graphics

The Design of Vega-Lite

Single View Specification

Layered and Multi-view Composition

Interactions with Selections

Using Vega-Lite

Programming with Vega-Lite

Higher-level Tools and Recommendations

Using Vega-Lite

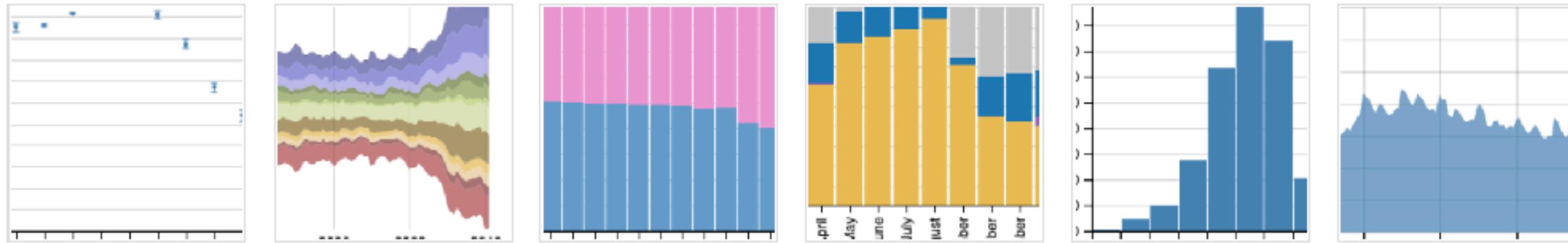
Compile to Vega and
use Vega's runtime

{ Retarget to different renderers
(Web-based/Server, Canvas/SVG)
Support streaming data

Declarative
JSON Syntax

{ Serve as file format
Bindings for different languages

Declarative Visualization in Python



Altair is a declarative statistical visualization library for Python, based on [Vega-Lite](#).

With Altair, you can spend more time understanding your data and its meaning. Altair's API is simple, friendly and consistent and built on top of the powerful [Vega-Lite](#) visualization grammar. This elegant simplicity produces beautiful and effective visualizations with a minimal amount of code.

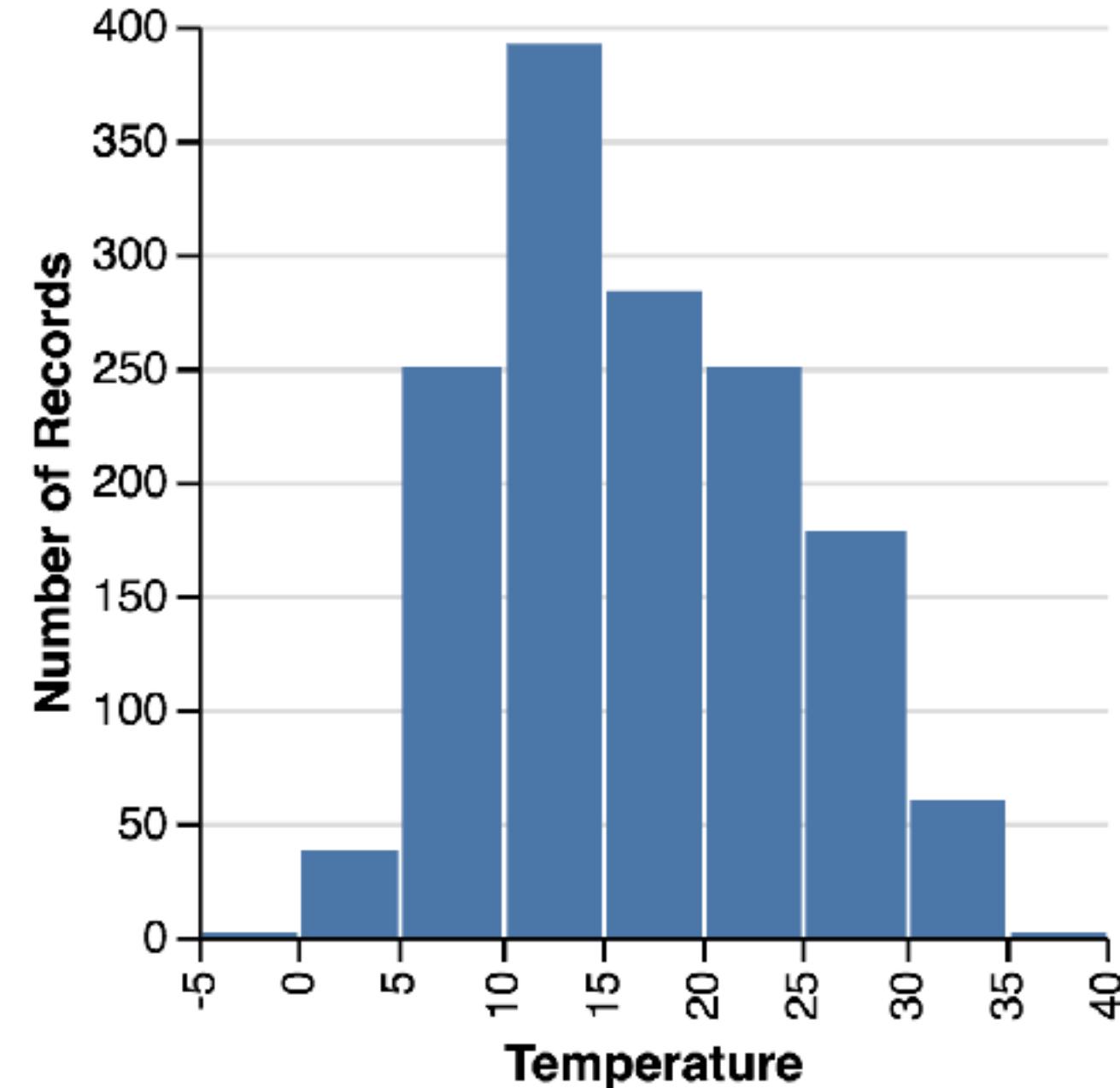
Altair: Vega-Lite in Python

Led by Brian Granger and Jake VanderPlas.



Histogram in Altair

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      bin: true,  
      field: "Temperature",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "count",  
      type: "quantitative"  
    }  
  }  
}
```



Histogram in Altair

```
{  
  data: {url: "weather-seattle.json"},  
  mark: "bar",  
  encoding: {  
    x: {  
      bin: true,  
      field: "Temperature",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "count",  
      type: "quantitative"  
    }  
  }  
}
```

```
from altair import Chart, expr  
  
data_weather = expr.DataFrame('data/weather-  
seattle.json')  
  
Chart(data_weather)  
  .mark_bar()  
  .encode(  
    x=X(bin=True, field='Temperature'),  
    y=Y(aggregate='count'))  
)
```

**Altair's API is automatically generated
from the Vega-Lite JSON schema.**

“It is this type of 1:1:1 mapping between thinking, code, and visualization that is my favorite thing about [Altair]”

– Dan Saber.

<https://dansaber.wordpress.com/2016/10/02/a-dramatic-tour-through-pythons-data-visualization-landscape-including-ggplot-and-altair/>

“We see this portion of the effort as much bigger than Altair itself: the Vega and Vega-Lite specifications are perhaps the best existing candidates for a principled lingua franca of data visualization” – Altair Team.

Vega-Lite: a Grammar of Interactive Graphics

The Design of Vega-Lite

Single View Specification

Layered and Multi-view Composition

Interactions with Selections

Using Vega-Lite

Programming with Vega-Lite

Higher-level Tools and Recommendations

Voyager

Augment manual specification with recommendation
to **promote breadth & reduce tedium** in exploration.

Use Vega-Lite to **recommend** data and visual encodings.

<https://github.com/vega/voyager>

Data

Cars Change

Fields

- A Cylinders T +
- A Name T +
- A Origin T +
- Year T +
- # Acceleration T +
- # Displacement T +
- # Horsepower T +
- # Miles per Gallon T +
- # Weight in lbs T +
- # COUNT +

Wildcards

- A Categorical Fields +
- Temporal Fields +
- # Quantitative Fields +

Filter Filter invalid numbers

Encoding

Clear

x drop a field here

y drop a field here

column drop a field here

row drop a field here

Marks

auto

size drop a field here

color drop a field here

shape drop a field here

detail drop a field here

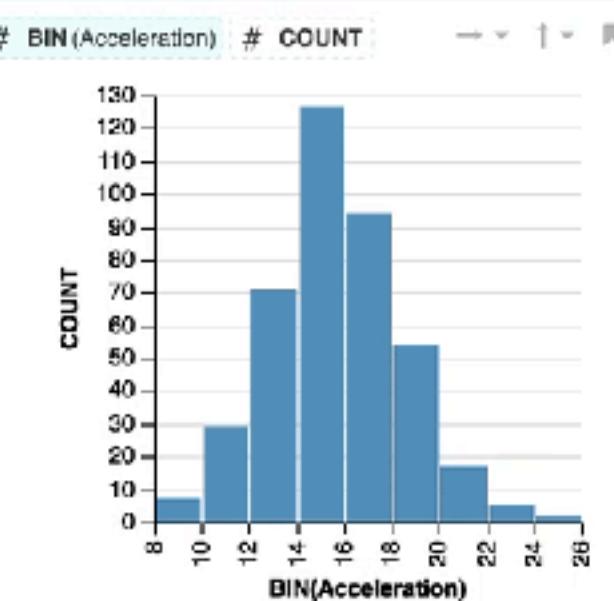
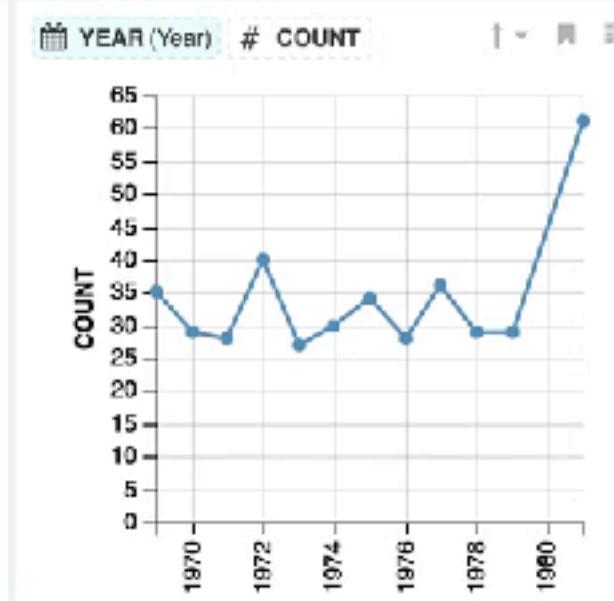
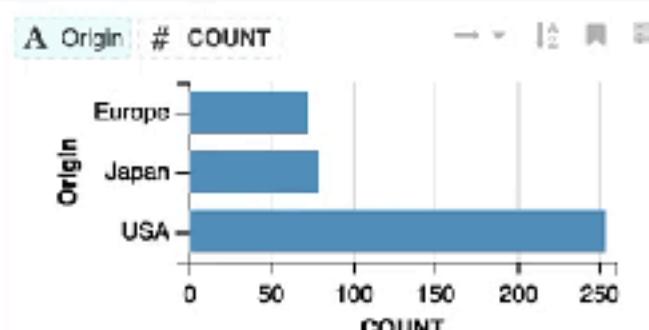
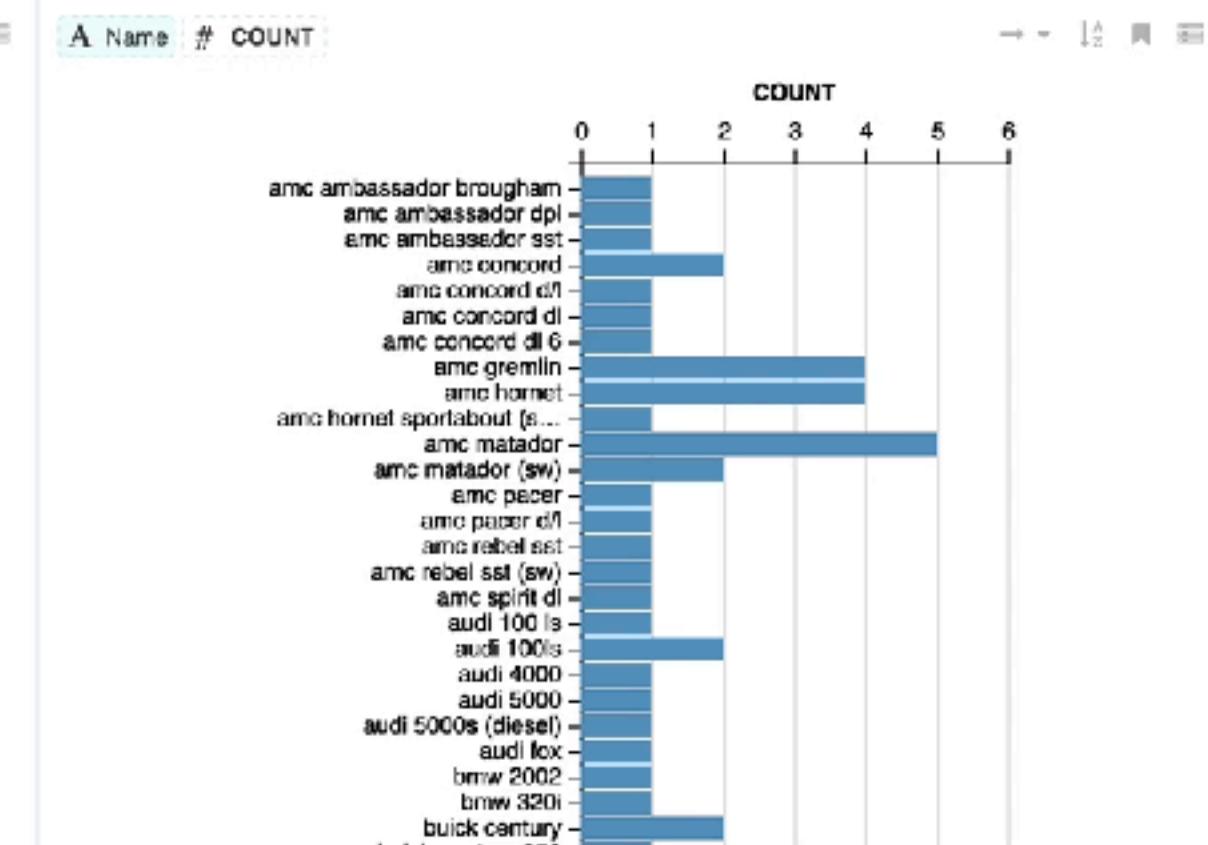
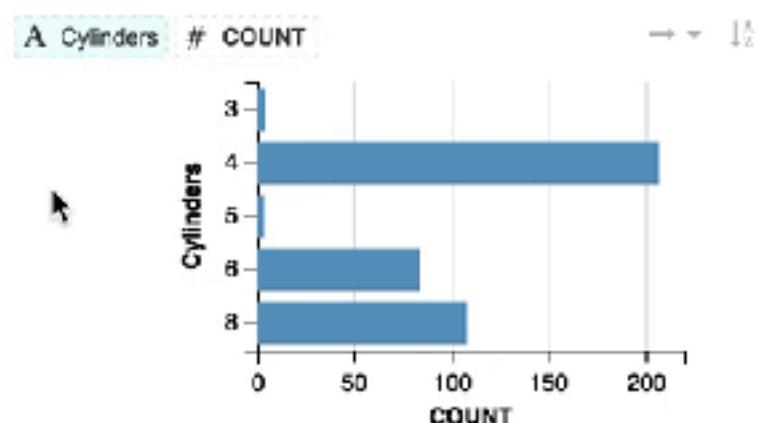
text drop a field here

Specified View

No specified visualization yet. Start exploring by dragging a field to encoding pane on the left or examining univariate summaries below.

Related Views

Univariate Summaries Hide

Univariate Summaries

Data

Cars Change

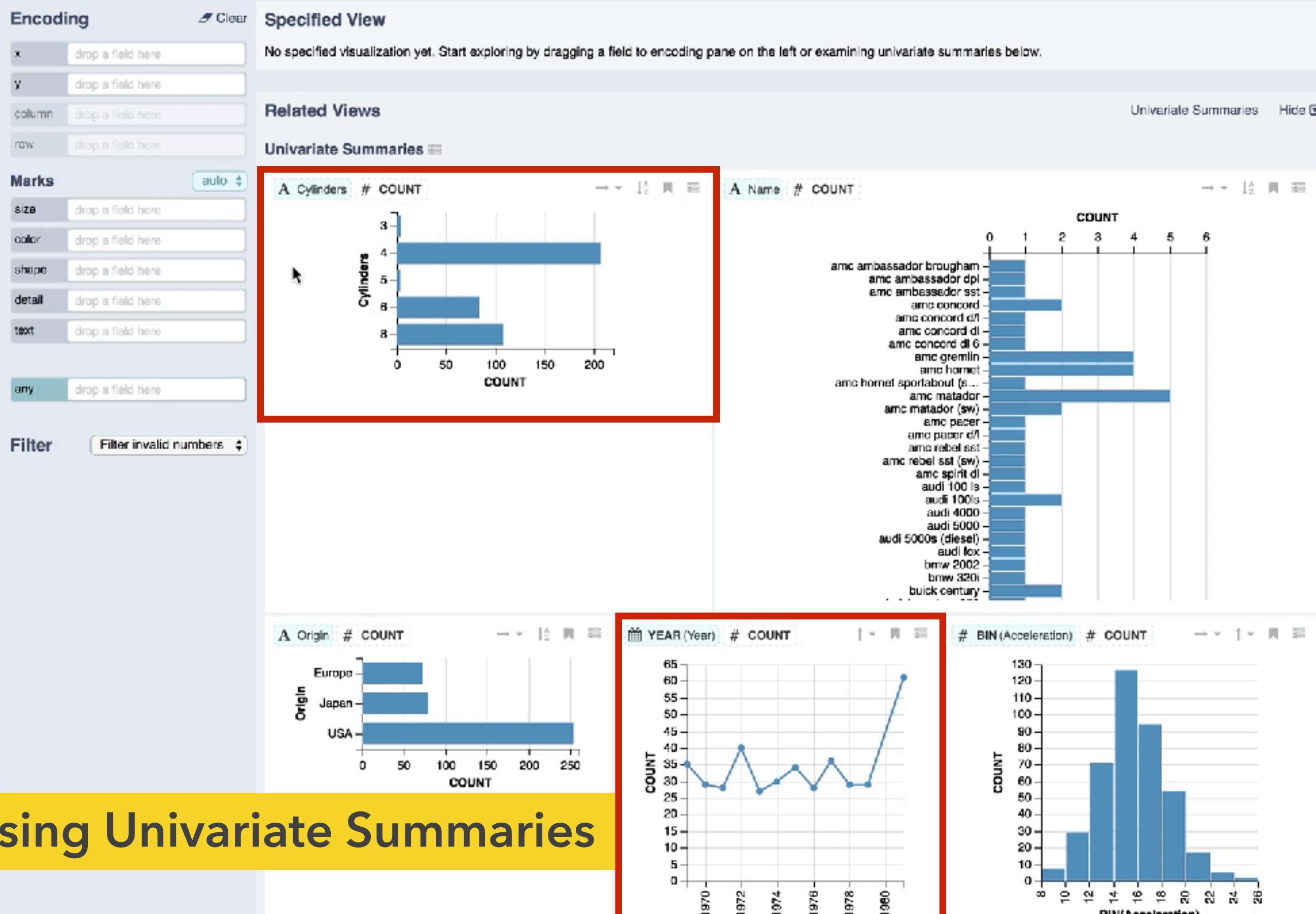
Fields

- A Cylinders T +
- A Name T +
- A Origin T +
- # Year T +
- # Acceleration T +
- # Displacement T +
- # Horsepower T +
- # Miles per Gallon T +
- # Weight in lbs T +
- # COUNT +

Wildcards

- A Categorical Fields +
- Temporal Fields +
- # Quantitative Fields +

Filter Filter invalid numbers



Data

Cars Change

Fields

- A Cylinders T +
- A Name T +
- A Origin T +
- Year T +
- # Acceleration T +
- # Displacement T +
- # Horsepower T +
- # Miles per Gallon T +
- # Weight in lbs T +
- # COUNT +

Wildcards

- A Categorical Fields +
- Temporal Fields +
- # Quantitative Fields +

Filter Filter invalid numbers

Encoding Clear

x drop a field here
y drop a field here
column drop a field here
row drop a field here

Marks auto

size drop a field here
color drop a field here
shape drop a field here
detail drop a field here
text drop a field here

Specified View

No specified visualization yet. Start exploring by dragging a field to encoding pane on the left or examining univariate summaries below.

Related Views

Univariate Summaries Hide

Univariate Summaries



2. Manual View Specification

Undo Redo

```
mark: "tick",
encoding: {
  x: {
    field: "Horsepower",
    type: "quantitative"
  }
}
```

```
}
```

```
}
```

```
}
```

```
}
```

```
}
```

```
# Miles per Gallon +
```

```
# Weight in lbs +
```

```
# COUNT +
```

```
Wildcards
```

```
A Categorical Fields +
```

```
Temporal Fields +
```

```
# Quantitative Fields +
```

detail drop a field here

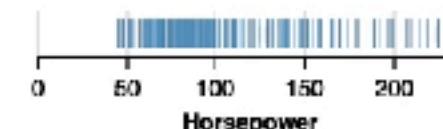
text drop a field here

any drop a field here

Filter Filter invalid numbers

Specified View

X-AXIS SWAP X/Y BOOKMARK

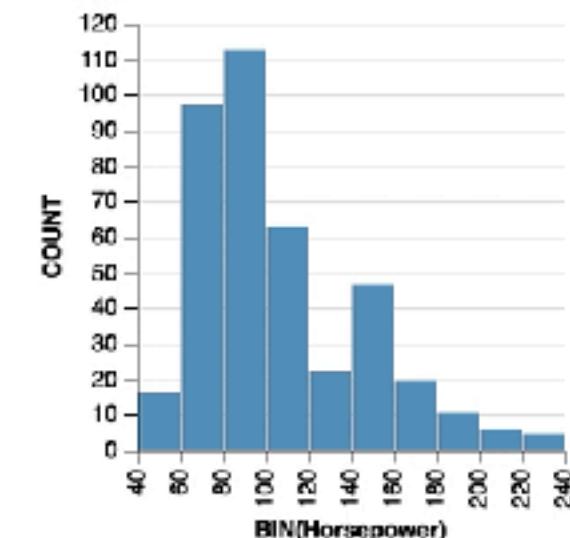


Related Views

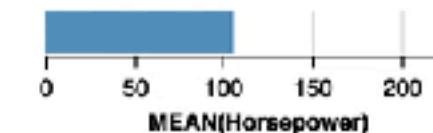
All Summaries Add Quantitative Field Add Categorical Field Add Temporal Field Hide

Summaries

BIN(Horsepower) # COUNT



MEAN(Horsepower)

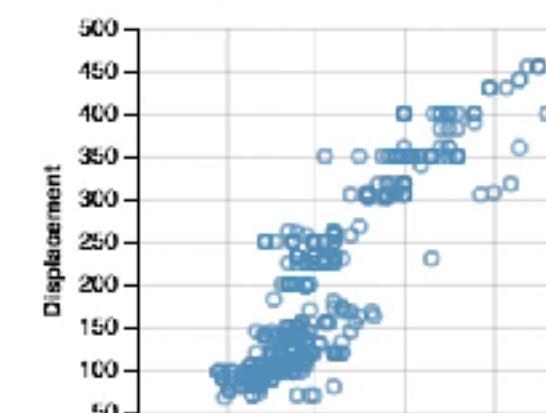


Add Quantitative Field

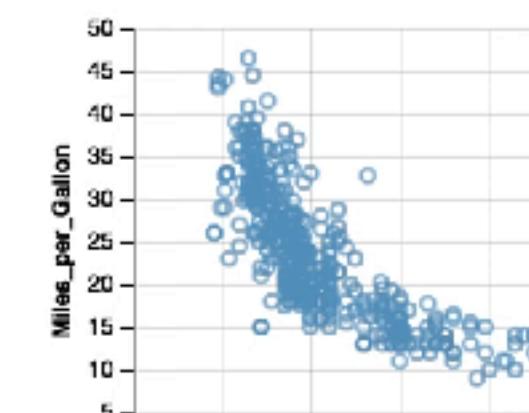
Horsepower # Acceleration



Horsepower # Displacement



Horsepower # Miles per Gallon



2. Manual View Specification

Data

Cars Change

Fields

- A Cylinders T +
- A Name T +
- A Origin T +
- # Year T +
- # Acceleration T +
- # Displacement T +
- # Horsepower T +
- # Miles per Gallon T +
- # Weight in lbs T +
- # COUNT +

Wildcards

- A Categorical Fields +
- Temporal Fields +
- # Quantitative Fields +

Filter Filter invalid numbers

Encoding Clear

x # Horsepower

y drop a field here

column drop a field here

row drop a field here

Marks auto

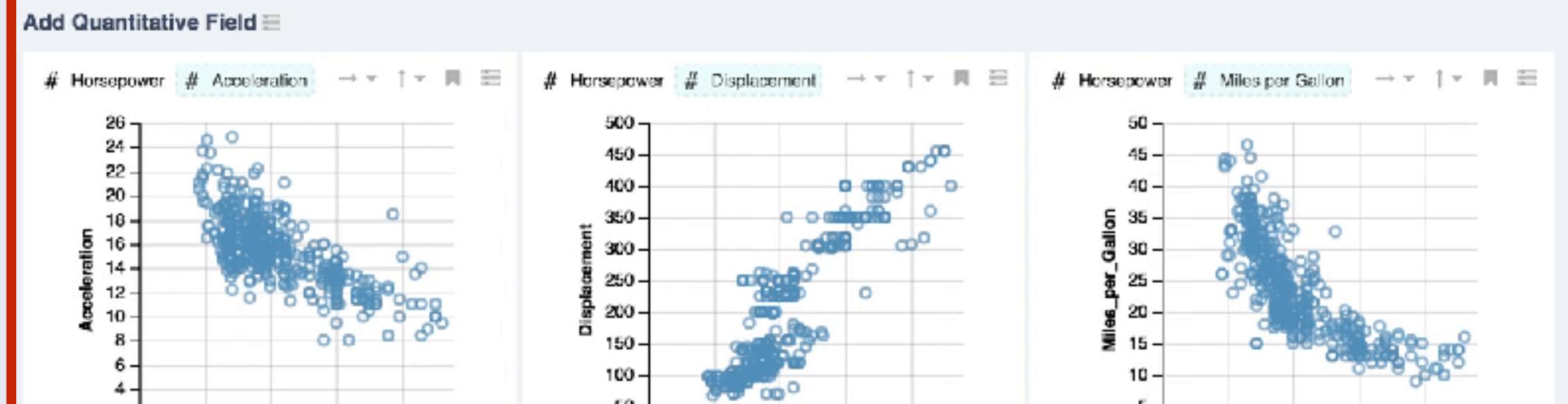
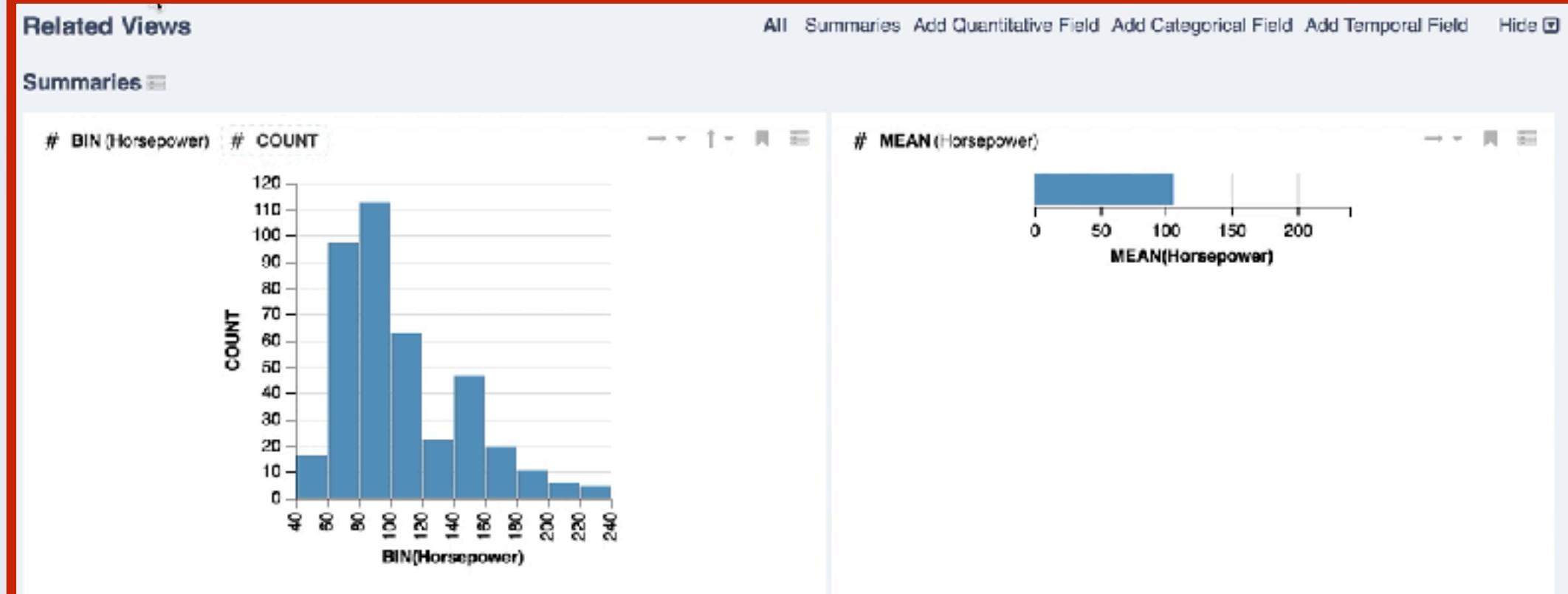
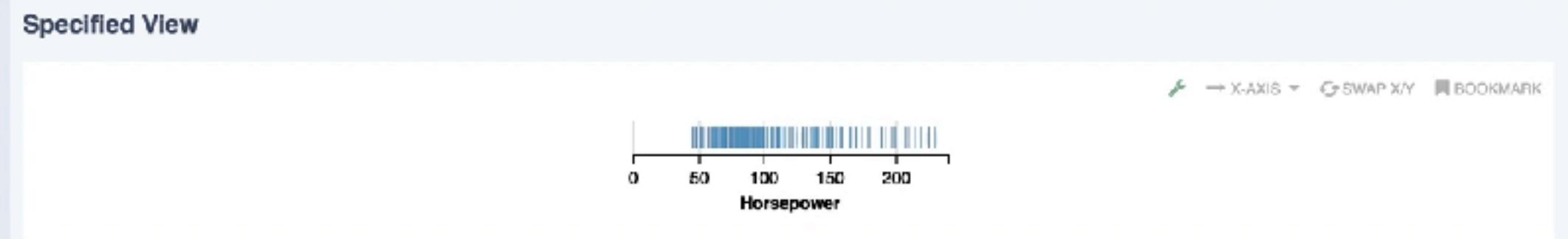
size drop a field here

color drop a field here

shape drop a field here

detail drop a field here

text drop a field here



3. Related Views

Data

Cars Change

Fields

- A Cylinders T +
- A Name T +
- A Origin T +
- # Year T +
- # Acceleration T +
- # Displacement T +
- # Horsepower T +
- # Miles per Gallon T +
- # Weight in lbs T +
- # COUNT +

Wildcards

- A Categorical Fields +
- Temporal Fields +
- # Quantitative Fields +

Encoding Clear

x # Horsepower

y drop a field here

column drop a field here

row drop a field here

Marks auto

size drop a field here

color drop a field here

shape drop a field here

detail drop a field here

text drop a field here

any drop a field here



```

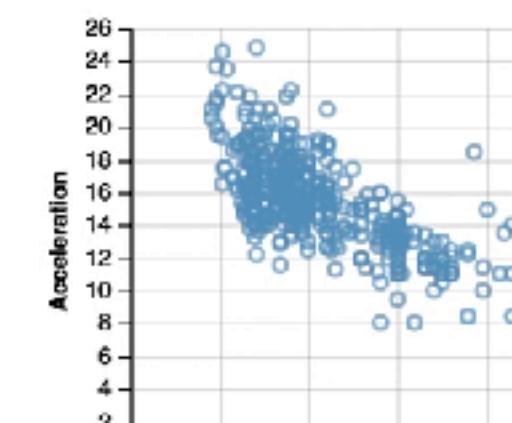
mark: "tick",
encoding: {
  x: {
    field: "Horsepower",
    type: "quantitative"
  }
}
}

```

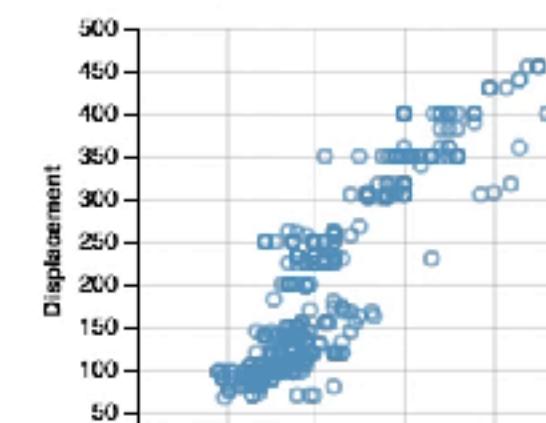


Add Quantitative Field

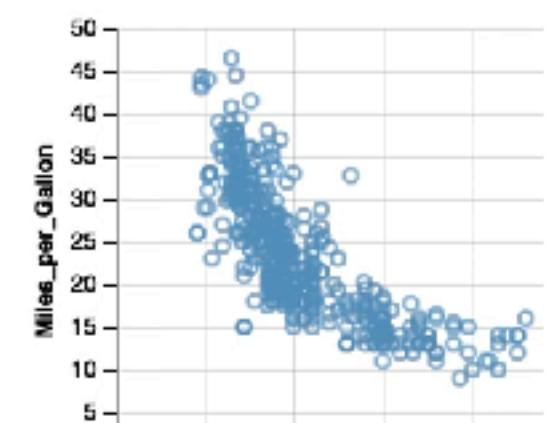
Horsepower # Acceleration



Horsepower # Displacement



Horsepower # Miles_per_Gallon



3. Related Views

Data

Cars Change

Fields

- A Cylinders T +
- A Name T +
- A Origin T +
- Y Year T +
- # Acceleration T +
- # Displacement T +
- # Horsepower T +

Encoding

x # Horsepower

y drop a field here

column drop a field here

row drop a field here

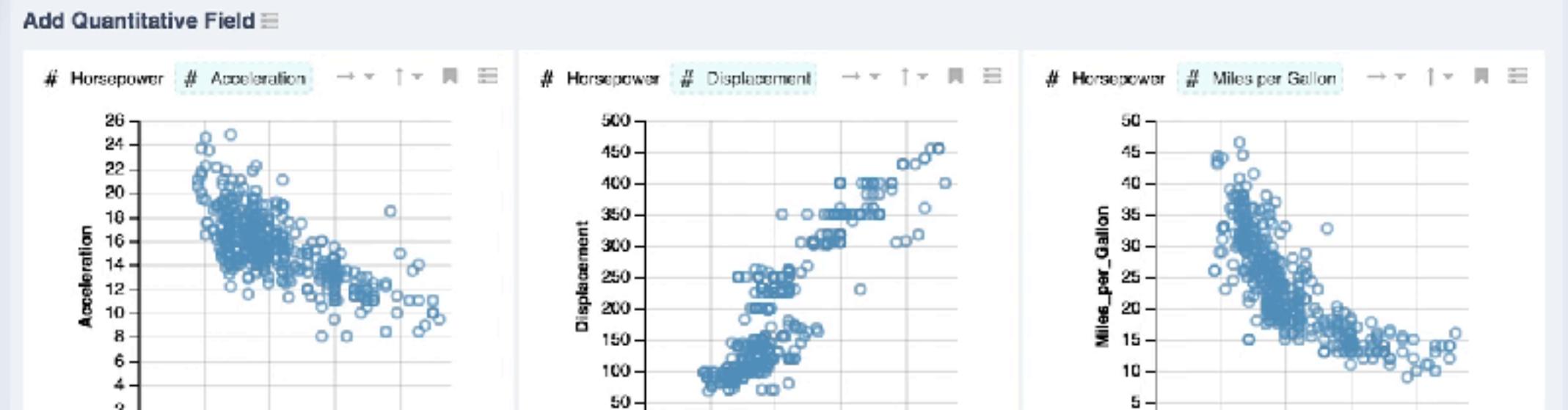
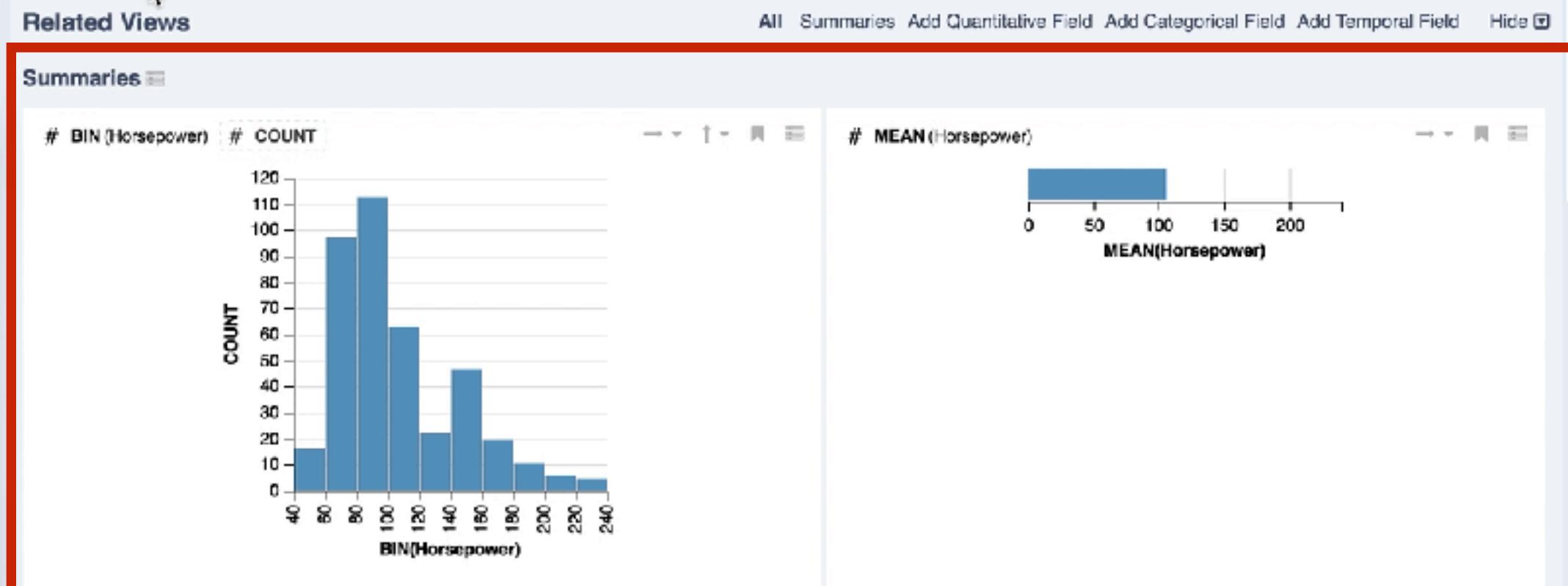
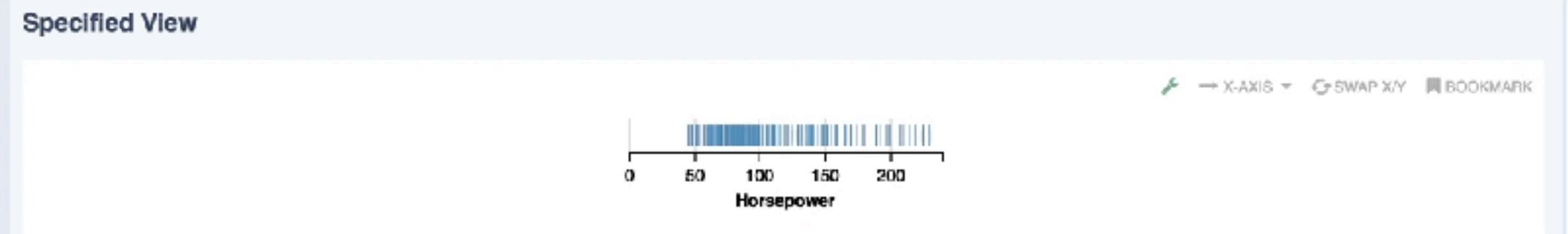
Marks

auto

size drop a field here

color drop a field here

shape drop a field here



3. Related Views

Data

- Cars

Fields

- A Cylinders
- A Name
- A Origin
- # Year
- # Acceleration
- # Displacement
- # Horsepower

Encoding

x: # Horsepower

y: drop a field here

column: drop a field here

row: drop a field here

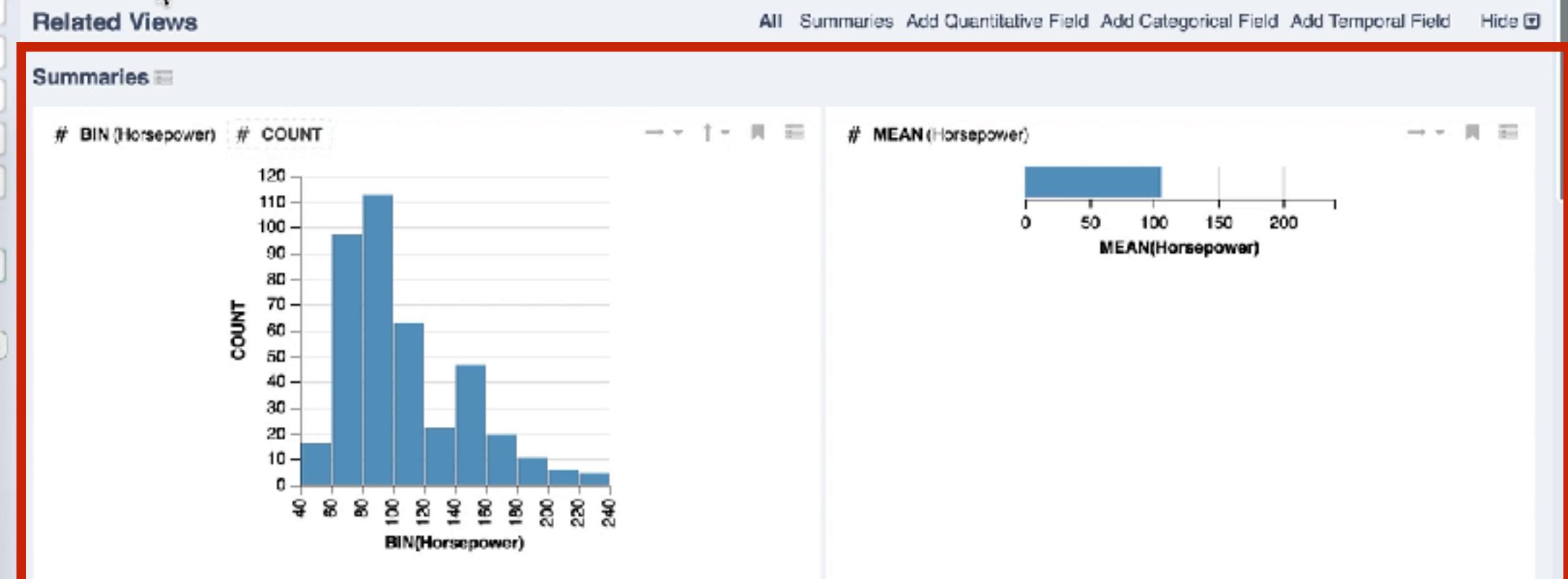
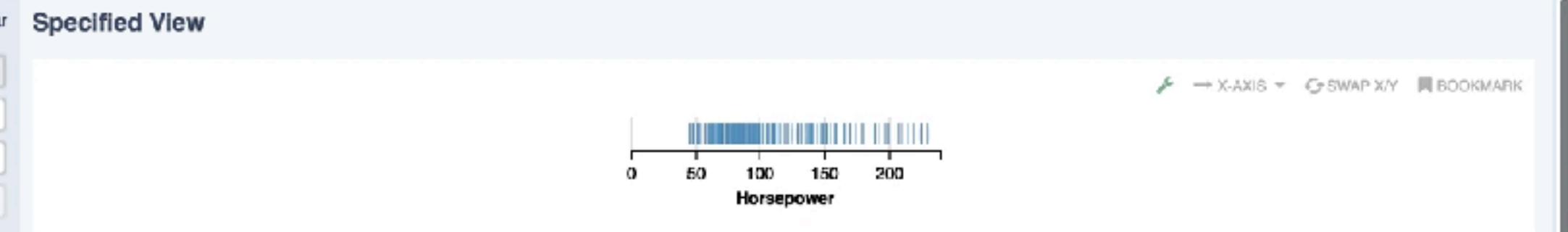
Marks

auto

size: drop a field here

color: drop a field here

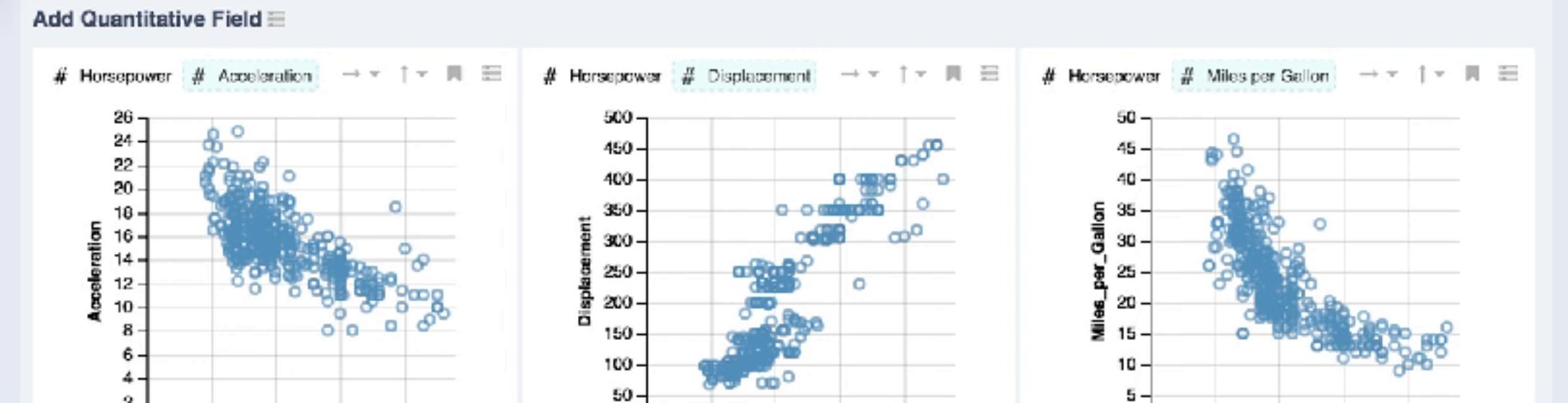
shape: drop a field here



```
{
  mark: "tick",
  encoding: {
    x: {
      fn: bin/mean?,
      field: "Horsepower",
      type: "quantitative"
    }
  }
}
```

Enumerate Transform

3. Related Views



Data

Cars Change

Fields

- A Cylinders T +
- A Name T +
- A Origin T +
- Y Year T +
- # Acceleration T +
- # Displacement T +
- # Horsepower T +

Encoding

x # Horsepower

y drop a field here

column drop a field here

row drop a field here

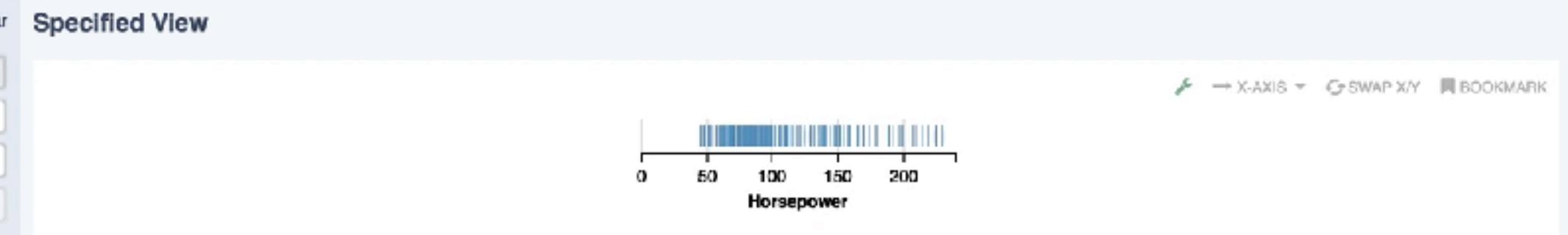
Marks

auto

size drop a field here

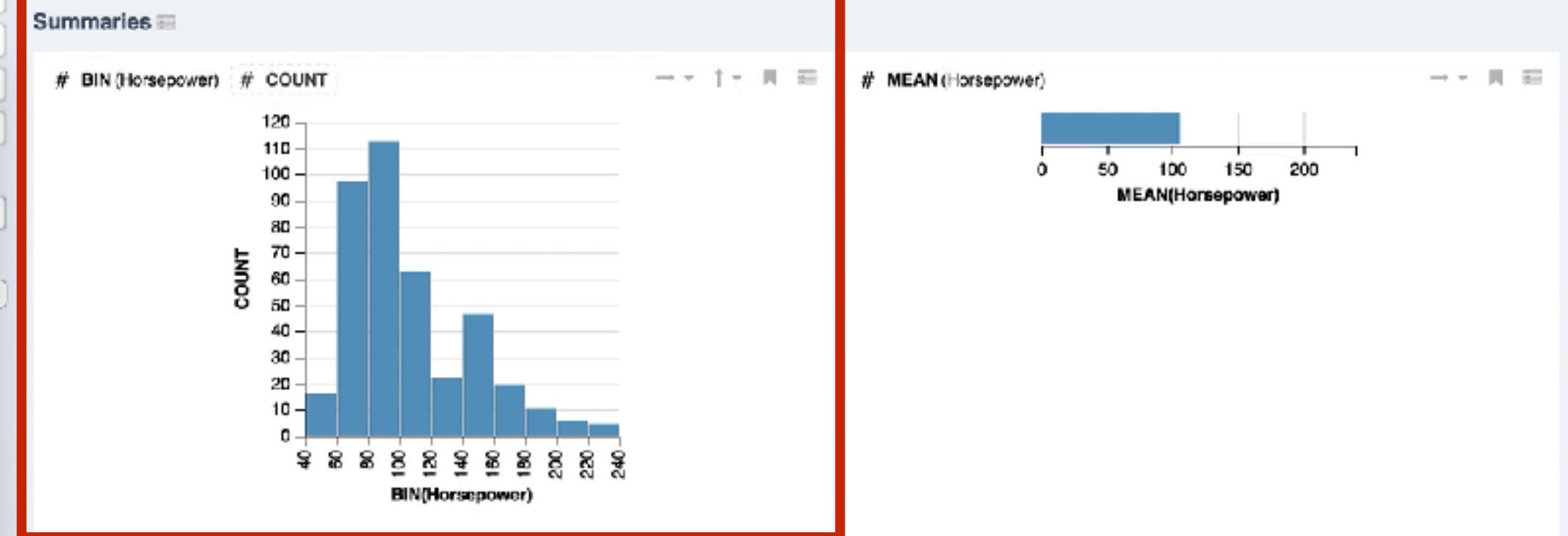
color drop a field here

shape drop a field here



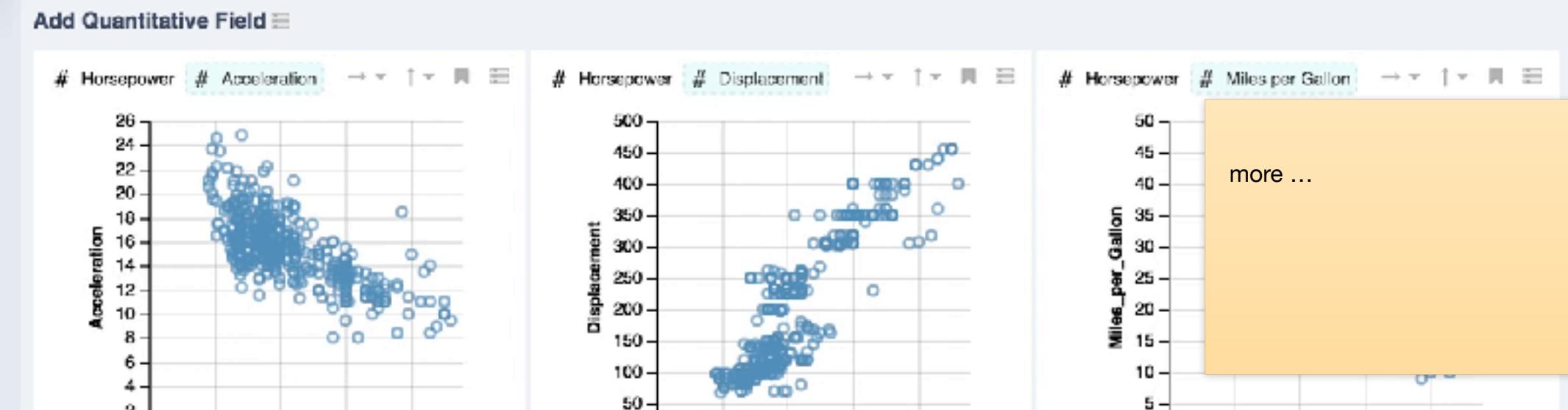
Related Views

All Summaries Add Quantitative Field Add Categorical Field Add Temporal Field Hide



```
{  
  mark: "tick",  
  encoding: {  
    x: {  
      fn: "bin",  
      field: "Horsepower",  
      type: "quantitative"  
    }  
  }  
}
```

3. Related Views



Data

Cars

Change

Fields

A. Cylinders

T +

A. Name

T +

A. Origin

T +

Year

T +

Acceleration

T +

Displacement

T +

Horsepower

T +

Encoding

Clear

Specified View

```
{  
  mark: "tick",  
  encoding: {  
    x: {  
      fn: "bin",  
      field: "Horsepower",  
      type: "quantitative"  
    }  
  }  
}
```

x

Horsepower

x

Y

drop a field here

y

column

drop a field here

c

row

drop a field here

r

Marks

auto

size

drop a field here

s

color

drop a field here

c

shape

drop a field here

s



Related Views

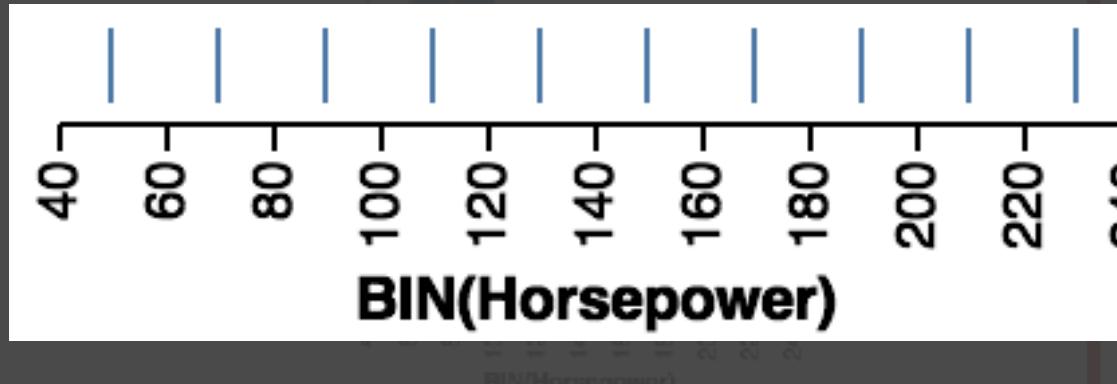
All Summaries Add Quantitative Field Add Categorical Field Add Temporal Field Hide

Summaries

BIN(Horsepower) # COUNT



MEAN(Horsepower)



Add Quantitative Field

Horsepower

Acceleration

Horsepower

Displacement

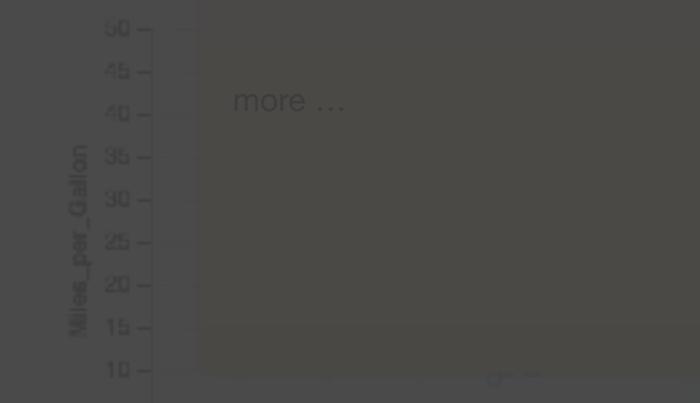
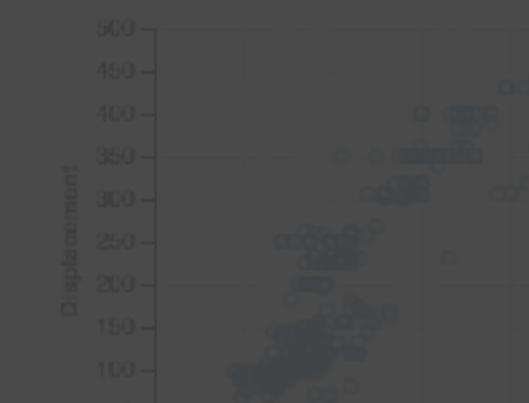
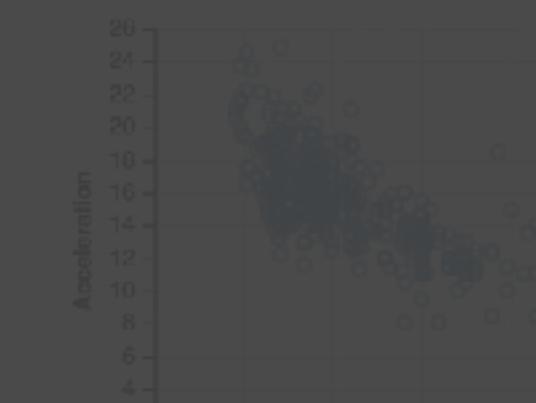
Miles_per_Gallon

Horsepower

Acceleration

Displacement

Miles_per_Gallon



3. Related Views

Data

Cars

Change

Fields

A_Cylinders

T +

A_Name

T +

A_Oigin

T +

Year

T +

#_Acceleration

T +

#_Displacement

T +

#_Horsepower

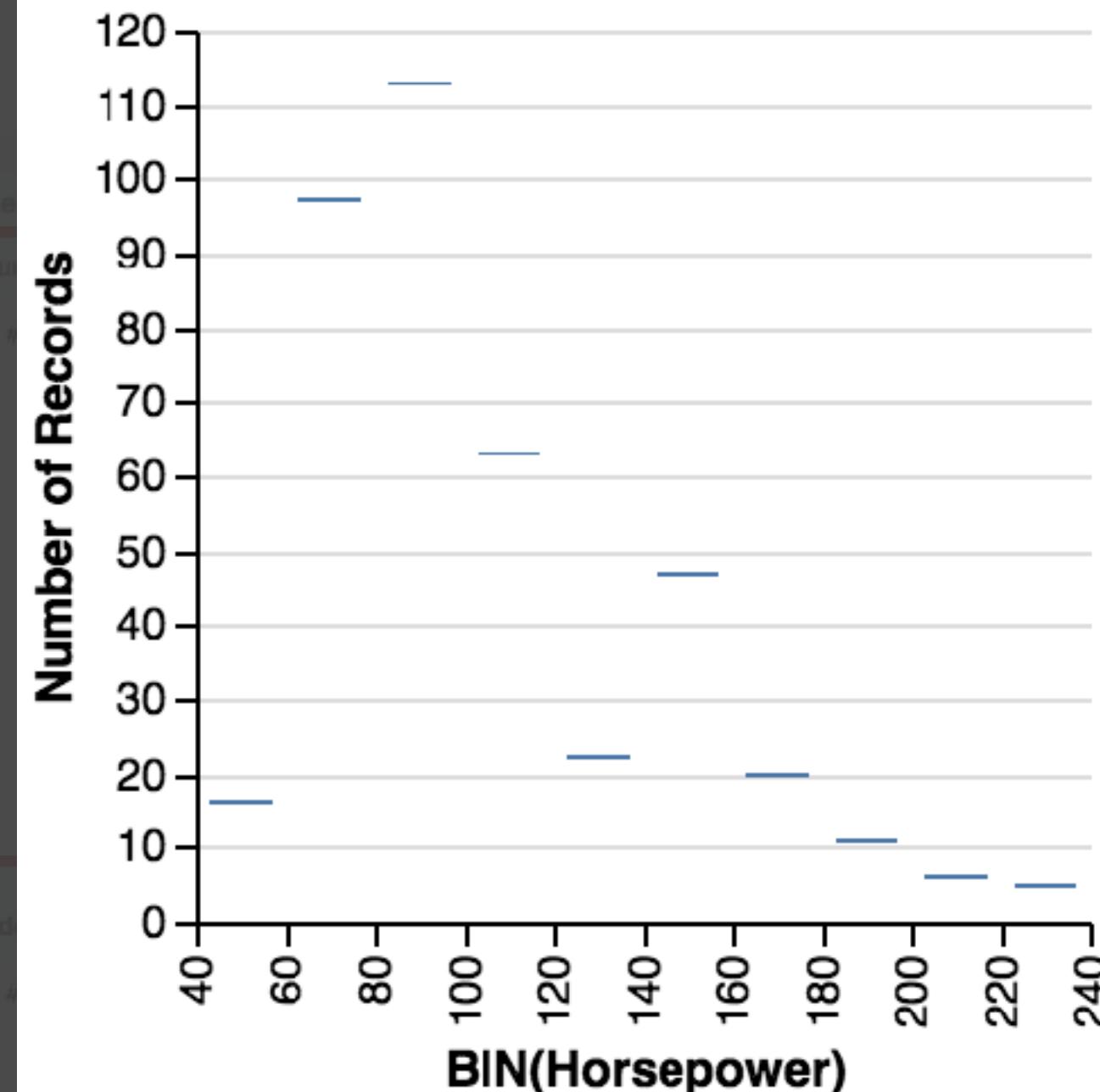
T +

Encoding

Clear

Specified View

X — X-AXIS Y — Y-AXIS SWAP XY BOOKMARK



```
{  
  mark: "tick",  
  encoding: {  
    x: {  
      fn: "bin",  
      field: "Horsepower",  
      type: "quantitative"  
    },  
    y: {  
      aggregate: "count",  
      type: "quantitative"  
    }  
  }  
}
```

3. Related Views

Data

Cars

Change

Encoding

Clear

Specified View

Fields

A_Cylinders

Horsepower

A_Name

drop a field here

A_Origin

drop a field here

Year

drop a field here

Acceleration

drop a field here

Displacement

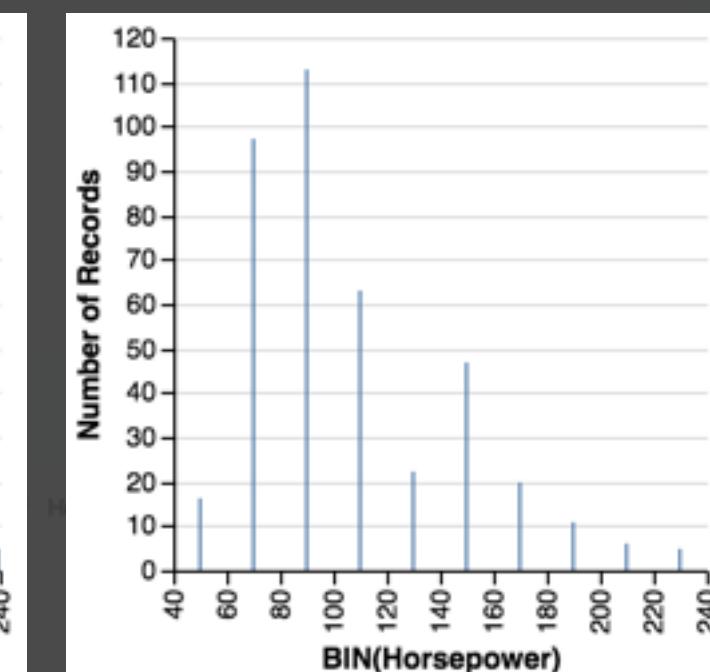
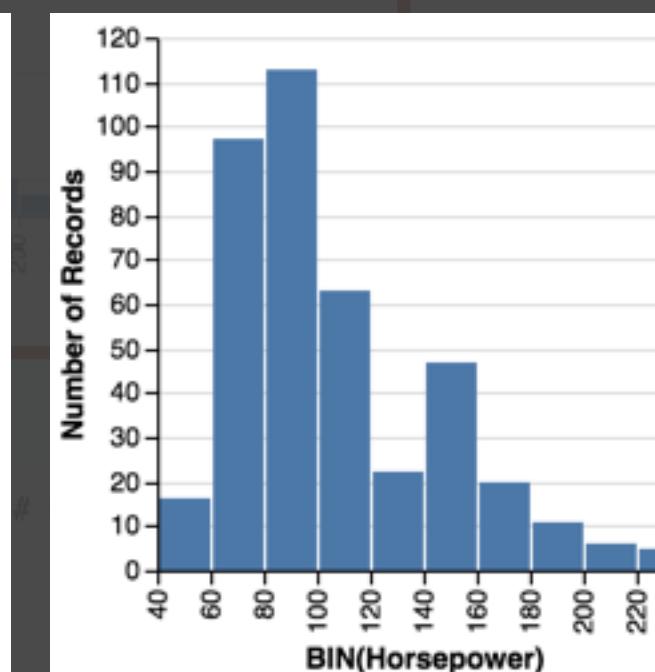
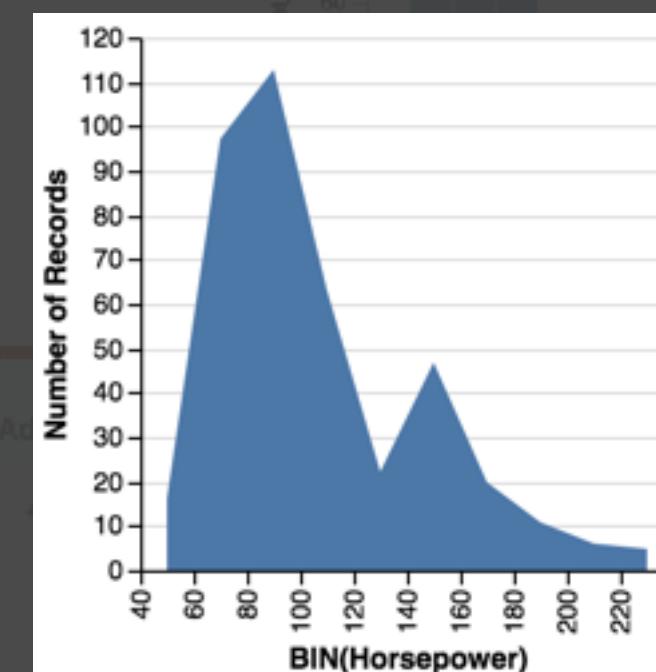
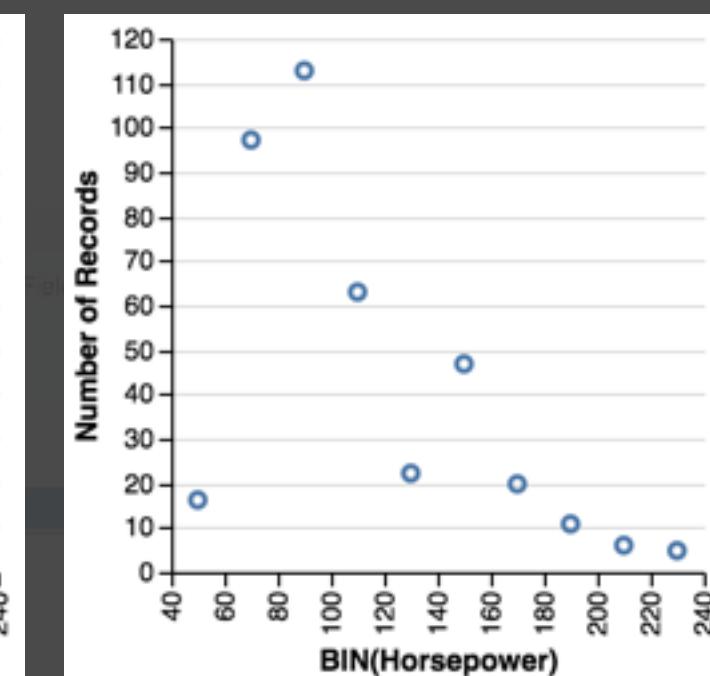
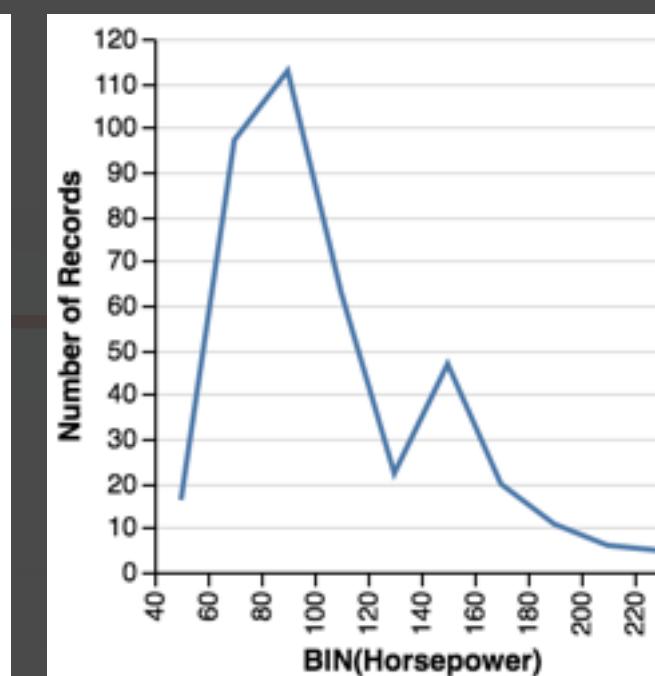
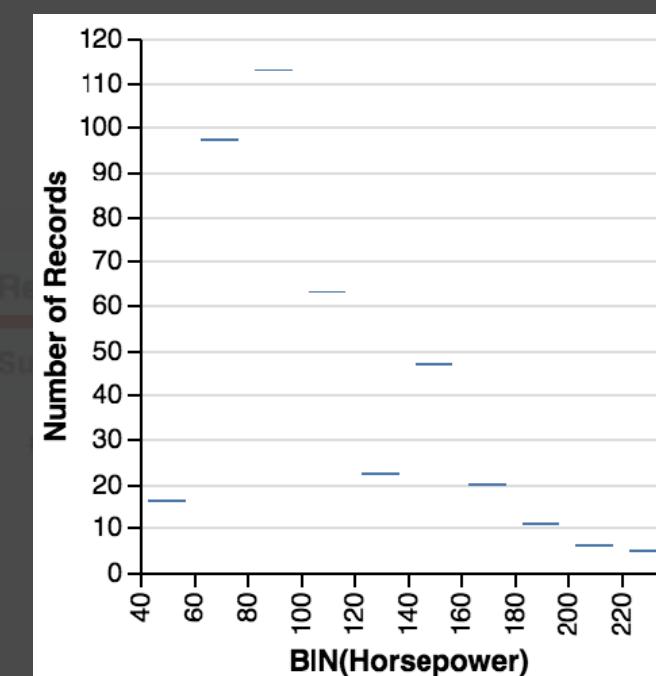
drop a field here

Horsepower

drop a field here

Enumerate & Rank Encodings

```
{
  mark: "?",
  encoding: {
    x: {
      fn: "bin",
      field: "Horsepower",
      type: "quantitative"
    },
    y: {
      aggregate: "count",
      type: "quantitative"
    }
  }
}
```



3. Related Views

Data

Cars

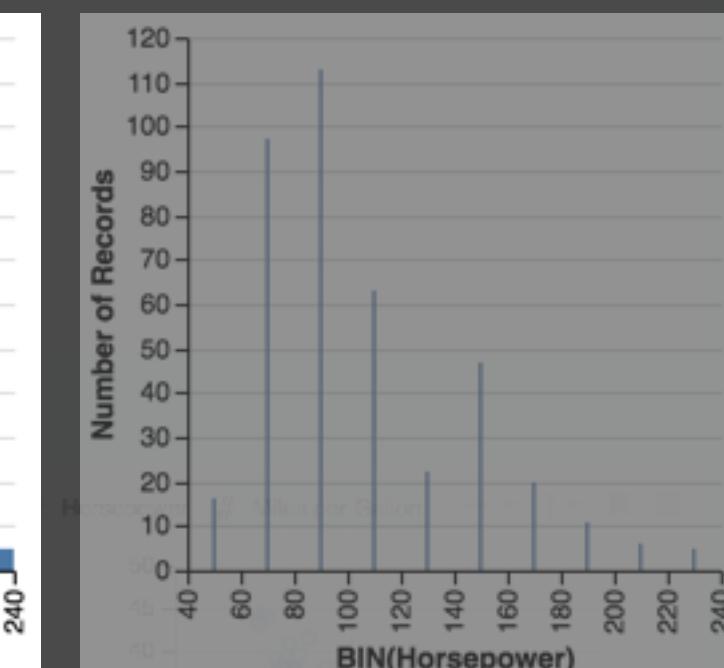
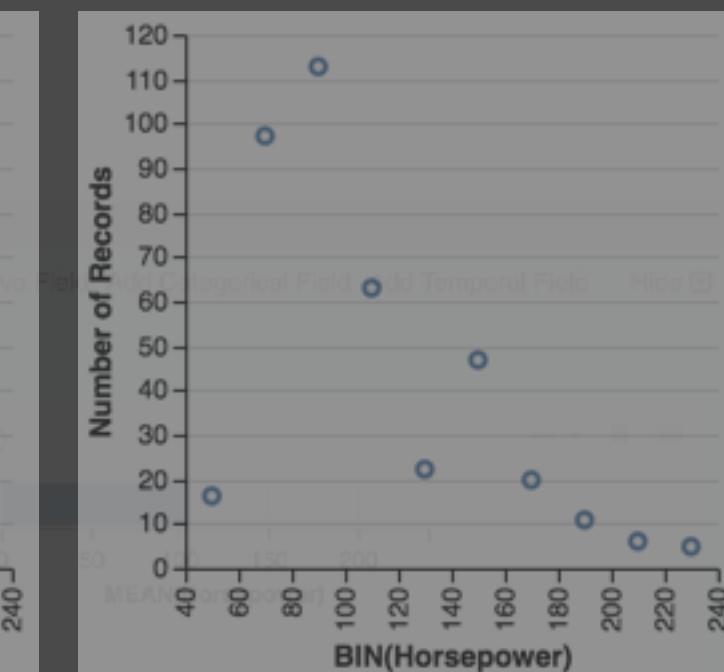
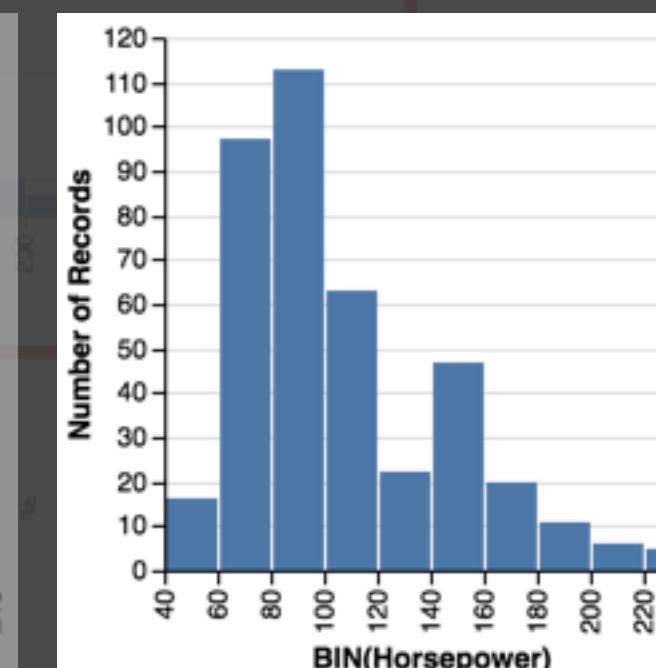
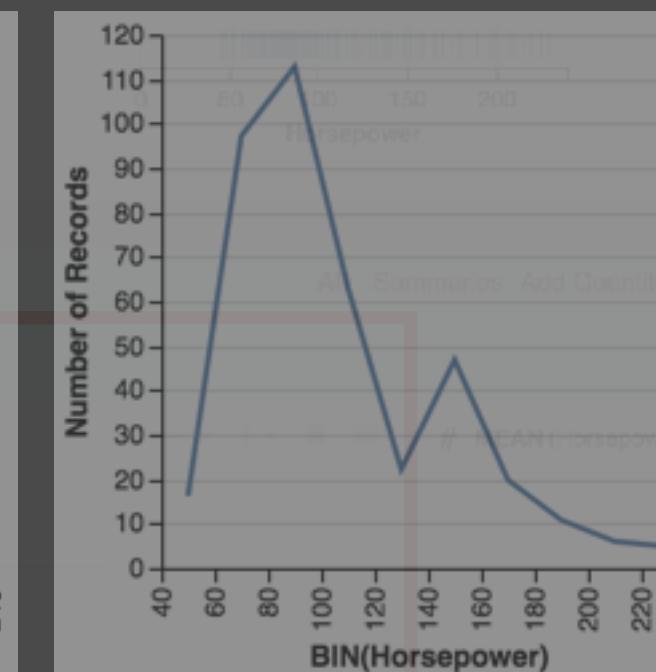
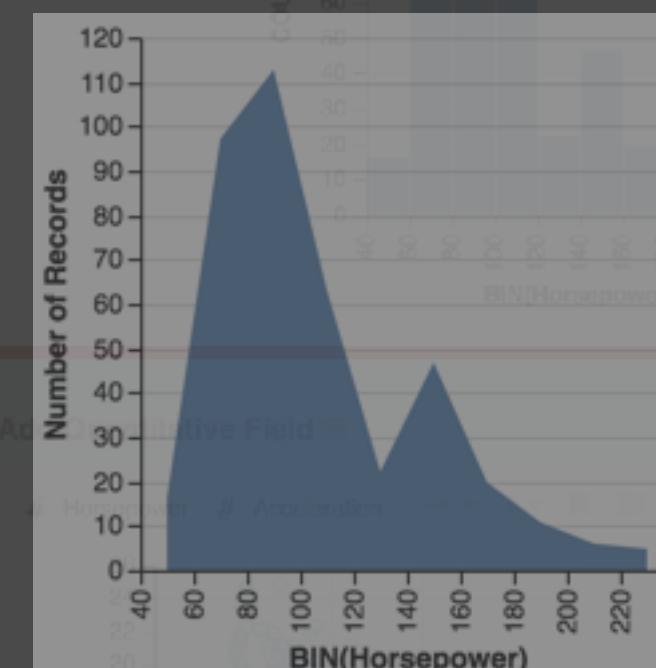
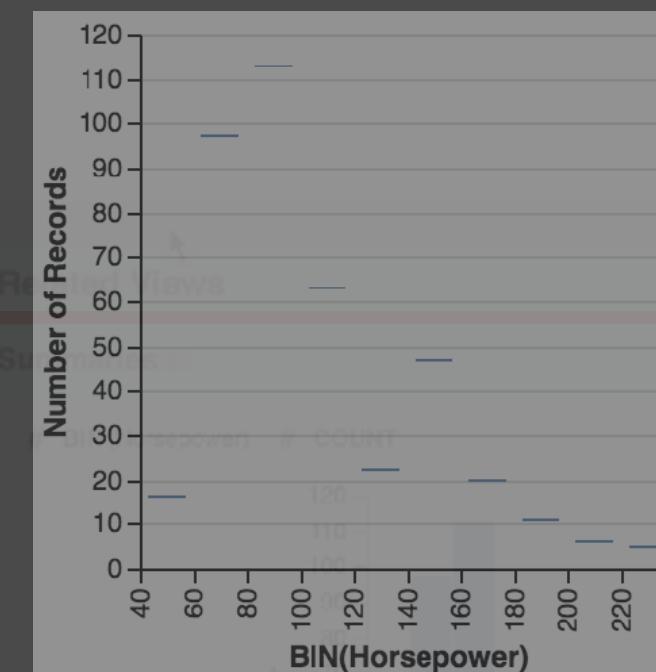
Change

Encoding

Clear

Specified View

```
{
  mark: "?",
  encoding: {
    x: {
      fn: "bin",
      field: "Horsepower",
      type: "quantitative"
    },
    y: {
      aggregate: "count",
      type: "quantitative"
    }
  }
}
```



3. Related Views

Data

Cars Change

Fields

- A Cylinders T +
- A Name T +
- A Origin T +
- # Year T +
- # Acceleration T +
- # Displacement T +
- # Horsepower T +
- # Miles per Gallon T +
- # Weight in lbs T +
- # COUNT +

Wildcards

- A Categorical Fields +
- Temporal Fields +
- # Quantitative Fields +

Filter Filter invalid numbers

Encoding Clear

x # Horsepower

y drop a field here

column drop a field here

row drop a field here

Marks auto

size drop a field here

color drop a field here

shape drop a field here

detail drop a field here

text drop a field here

any drop a field here



3. Related Views

Data

Cars Change

Fields

- A Cylinders T +
- A Name T +
- A Origin T +
- # Year T +
- # Acceleration T +
- # Displacement T +
- # Horsepower T +
- # Miles per Gallon T +
- # Weight in lbs T +
- # COUNT +

Wildcards

- A Categorical Fields +
- Temporal Fields +
- # Quantitative Fields +

Encoding Clear

x # Horsepower

y drop a field here

column drop a field here

row drop a field here

Marks auto

size drop a field here

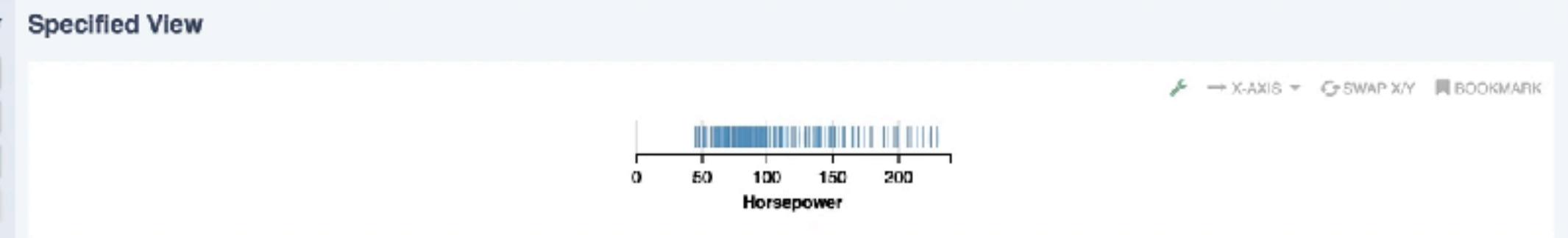
color drop a field here

shape drop a field here

detail drop a field here

text drop a field here

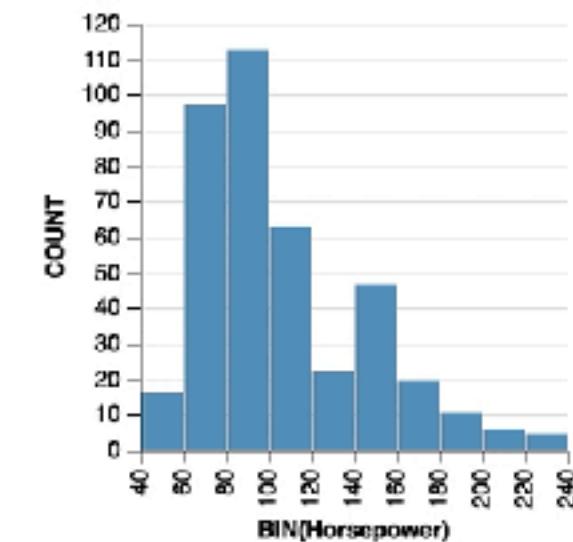
any drop a field here



All Summaries Add Quantitative Field Add Categorical Field Add Temporal Field Hide

Summaries

BIN(Horsepower) # COUNT



MEAN(Horsepower)

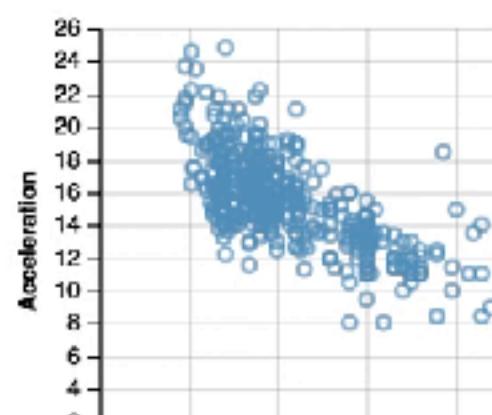


Filter

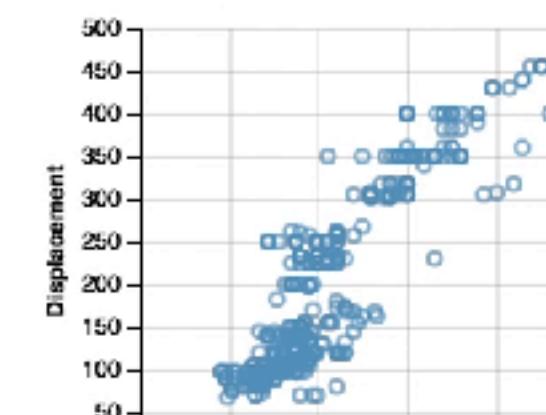
Filter invalid numbers

Add Quantitative Field

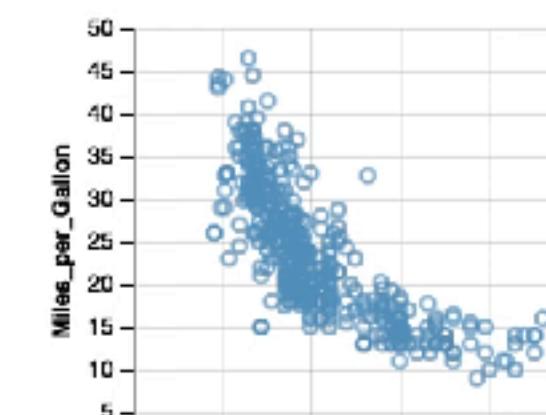
Horsepower # Acceleration



Horsepower # Displacement



Horsepower # Miles per Gallon



3. Related Views

Data

Cars Change

Fields

- A Cylinders T +
- A Name T +
- A Origin T +
- # Year T +
- # Acceleration T +
- # Displacement T +
- # Horsepower T +
- # Miles per Gallon T +
- # Weight in lbs T +
- # COUNT +

Wildcards

- A Categorical Fields +
- Temporal Fields +
- # Quantitative Fields +

Filter Filter invalid numbers

Encoding Clear

x # Horsepower

y drop a field here

column drop a field here

row drop a field here

Marks auto

size drop a field here

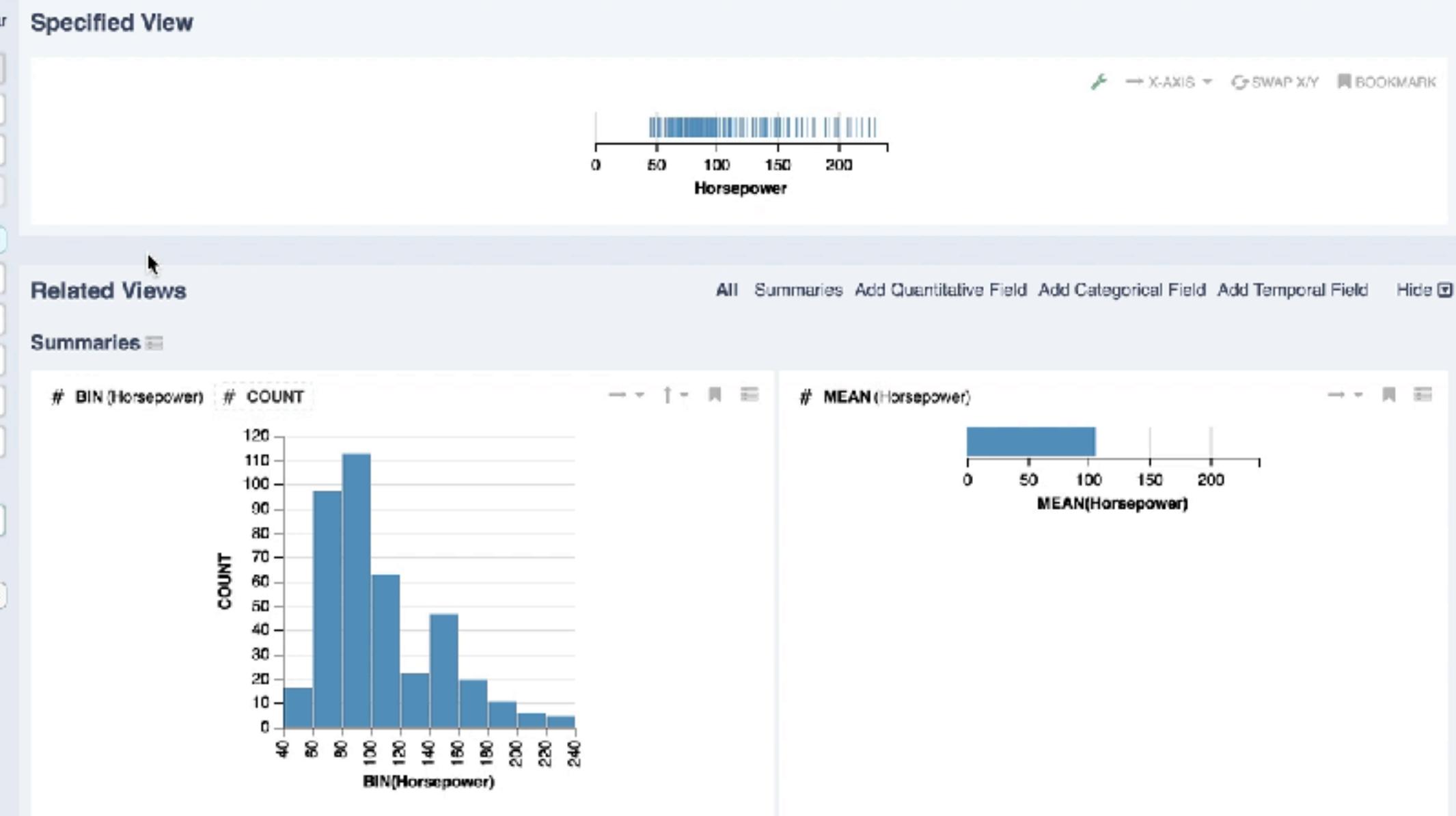
color drop a field here

shape drop a field here

detail drop a field here

text drop a field here

any drop a field here



4. Wildcard Specification

Data

Cars

Change

Fields

A_Cylinders

T +

A_Name

T +

```
mark: "tick",
encoding: {
  x: {
    field: "Horsepower",
    type: "quantitative"
  },
  y: {
    field: "?",
    type: "quantitative"
  }
}
```

Enumerate Variable

Encoding

Clear

x # Horsepower

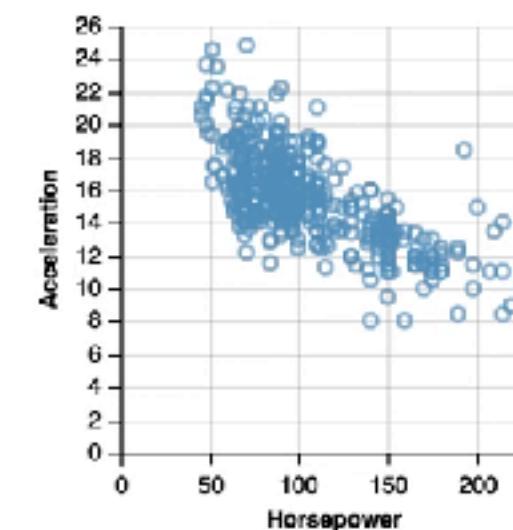
Y # Quantitative Fields

column drop a field here

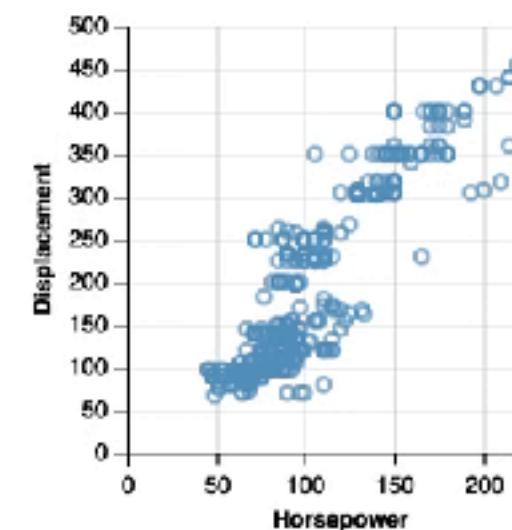
Specified Views

(Showing views with different fields) Auto-Add Count

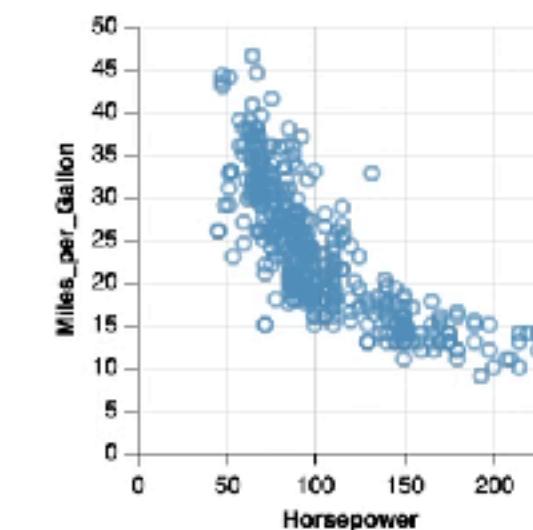
Horsepower # Acceleration



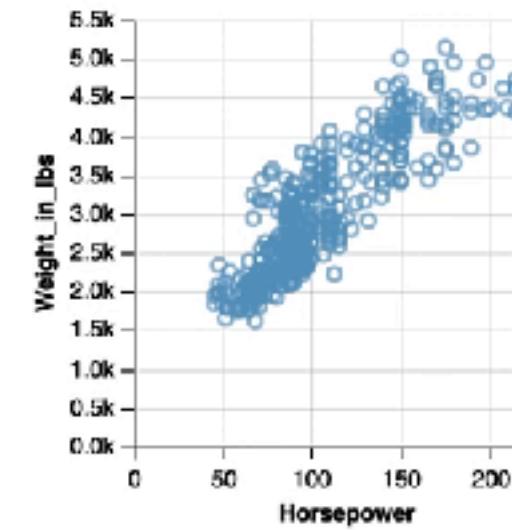
Horsepower # Displacement



Horsepower # Miles_per_Gallon



Horsepower # Weight_in_lbs



4. Wildcard Specification

Data

Cars Change

Fields

- A Cylinders T +
- A Name T +
- A Origin T +
- # Year T +
- # Acceleration T +
- # Displacement T +
- # Horsepower T +
- # Miles per Gallon T +
- # Weight in lbs T +
- # COUNT +

Wildcards

- A Categorical Fields +
- Temporal Fields +
- # Quantitative Fields +

Filter Filter invalid numbers

Encoding Clear

X # Horsepower

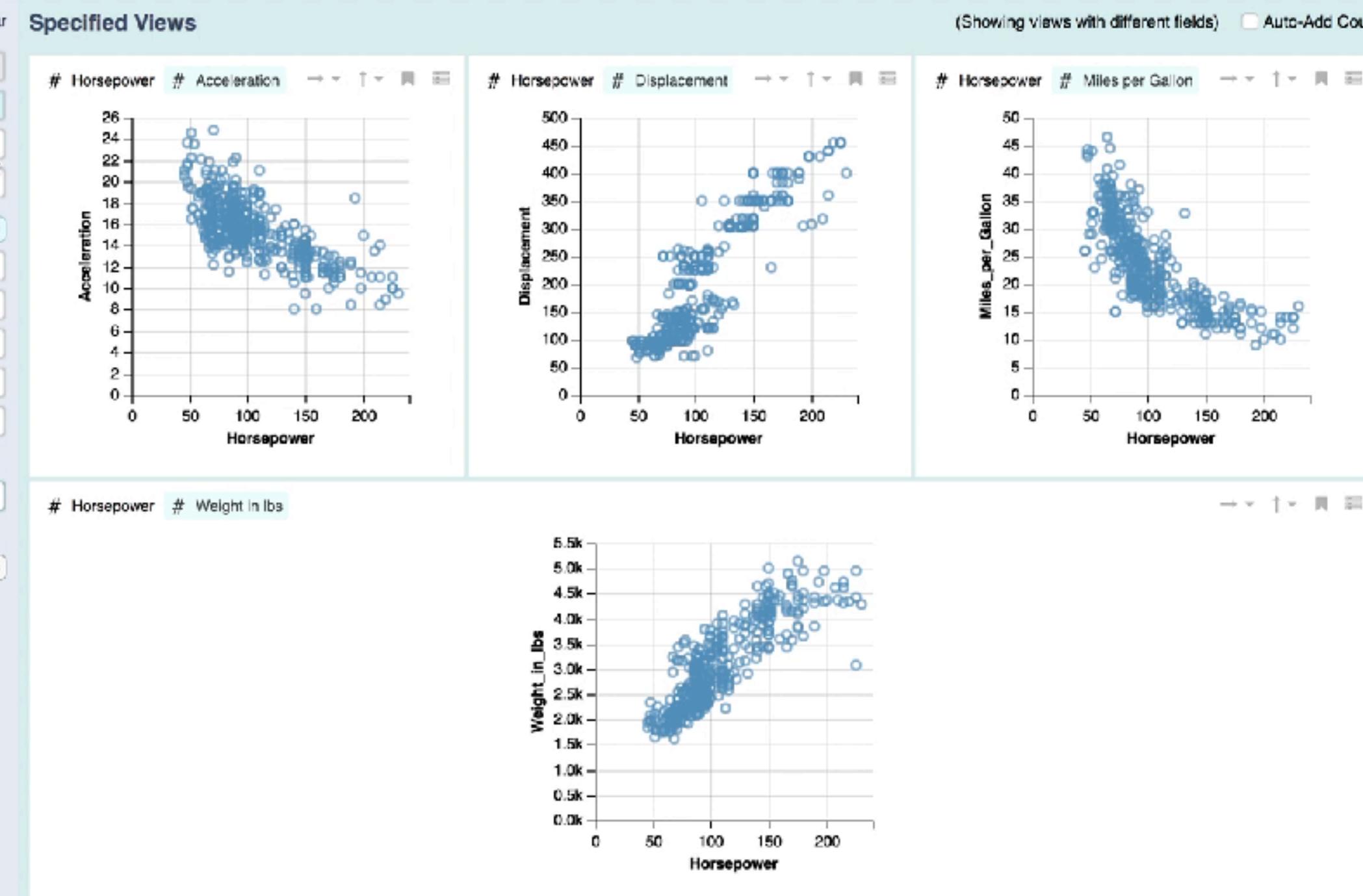
Y # Quantitative Fields

column drop a field here

row drop a field here

Marks

- size drop a field here
- color drop a field here
- shape drop a field here
- detail drop a field here
- text drop a field here



4. Wildcard Specification

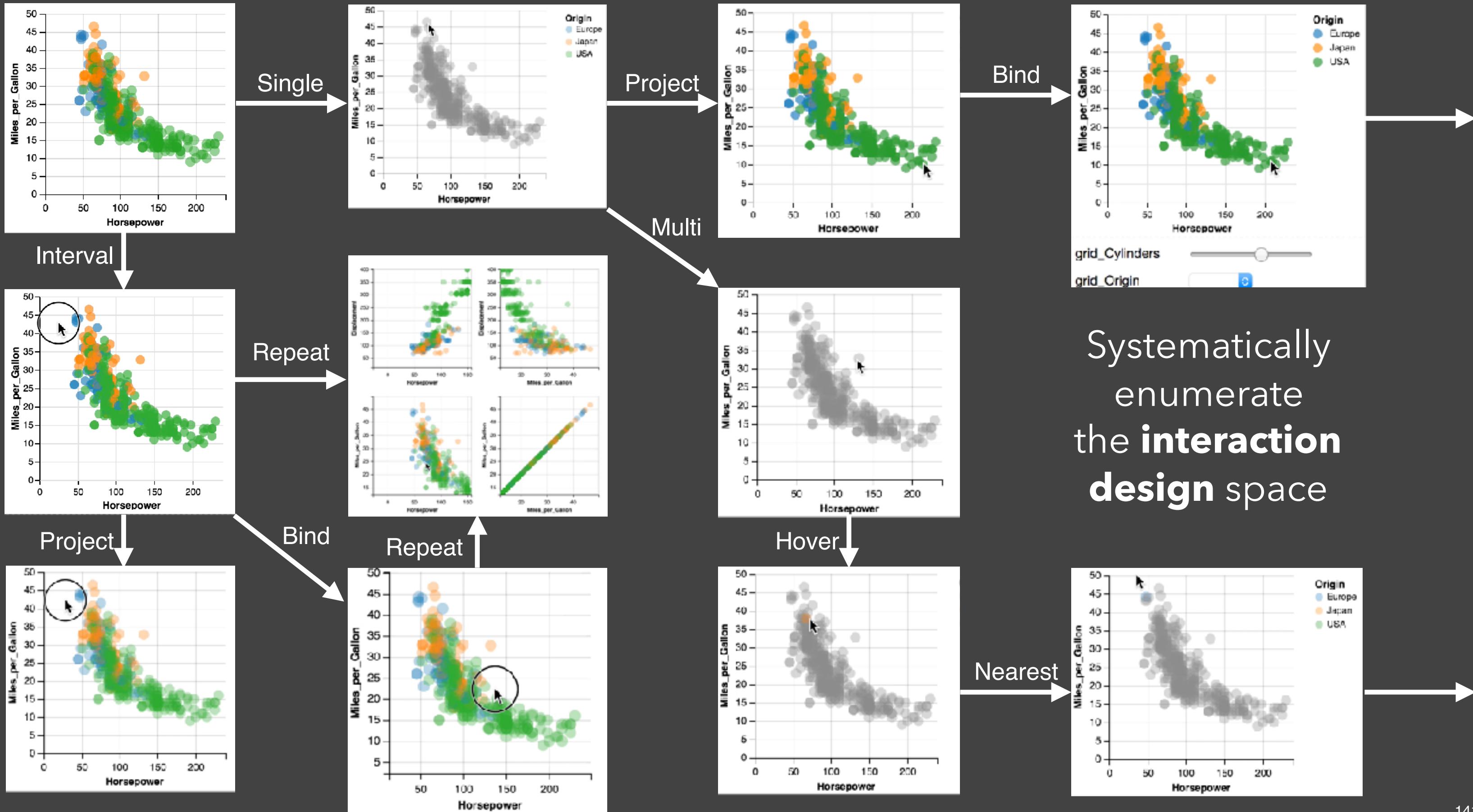
Voyager

Augment manual specification with recommendation
to **promote breadth & reduce tedium** in exploration

Use Vega-Lite to **recommend** data and visual encodings.

interactions?

<https://github.com/vega/voyager>



Vega-Lite: a Grammar of Interactive Graphics

The Design of Vega-Lite

Single View Specification

Layered and **Multi-view** Composition

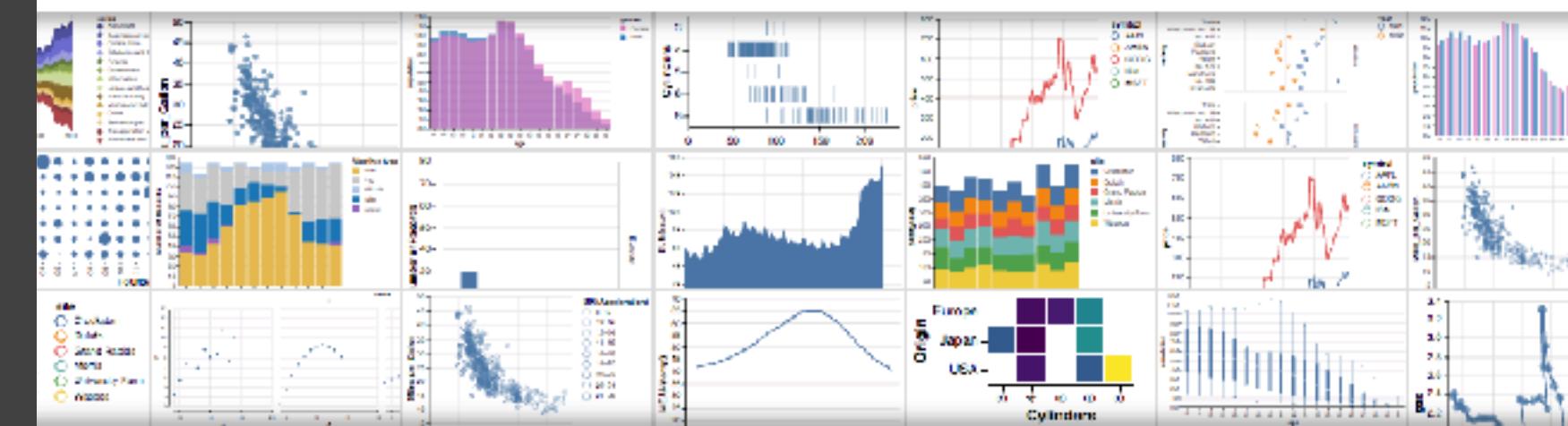
Interactions with Selections

Using Vega-Lite

Programming with Vega-Lite

Higher-level Tools and **Recommendations**

Vega-Lite – A High-Level Visualization Grammar



Vega-Lite is a high-level visualization grammar. It provides a concise JSON syntax for supporting rapid generation of visualizations to support analysis. Vega-Lite specifications can be compiled to [Vega](#) specifications.

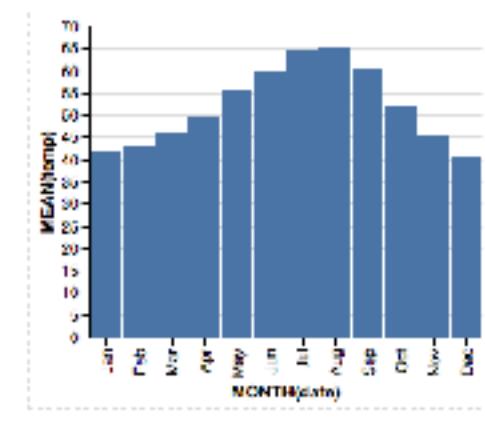
Vega-Lite specifications describe visualizations as mappings from data to properties of graphical marks (e.g., points or bars). It automatically produces visualization components including axes, legends, and scales. It then determines properties of these components based on a set of carefully designed rules. This approach allows Vega-Lite specifications to be succinct and expressive, but also provide user control. As Vega-Lite is designed for analysis, it supports data transformations such as aggregation, binning, filtering, sorting, and visual transformations including stacking and faceting.

[Get started](#)[Try online](#)

Read our [Introduction article on Medium](#), check out the [documentation](#) and take a look at our [example gallery](#).

Example

This is a Vega-Lite specification to create a bar chart that shows the average temperature in Seattle for each month.

[Open In Vega Editor](#)

```
{  
  "schema": "https://vega.github.io/schemas/vega-lite/v2.json",  
  "data": {"url": "data/seattle-temps.csv"},  
  "mark": "bar",  
  "encoding": {  
    "x": {  
      "binwidth": "month",  
      "field": "date",  
      "type": "temporal"  
    },  
    "y": {  
      "aggregate": "mean",  
      "field": "temp",  
      "type": "quantitative"  
    }  
  }  
}
```

Exploring Data

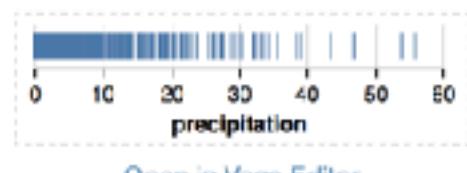
In this tutorial, you'll learn a few more techniques for creating visualizations in Vega-Lite. If you are not familiar with Vega-Lite, please read the [getting started tutorial](#) first.

For this tutorial, we will create visualizations to explore weather data for Seattle, taken from [NOAA](#). The data is available as a CSV file with columns for the temperature (in Celsius), precipitation (in centimeter), wind (in meter/second), and humidity. We have one row for each day from January 1st, 2012 to December 31st, 2015.

To load the CSV file with Vega-Lite, we need to provide a URL and set the format type in the data section of the specification.

```
"data": {"url": "data/seattle-weather.csv"}
```

Let's start by looking at the precipitation. Precipitation is a quantitative variable. Let's use a tick mark to show the distribution of precipitation.


[Open in Vega Editor](#)

```
{
  "$schema": "https://vega.github.io/schema/vega-lite/v2.json",
  "data": {"url": "data/seattle-weather.csv"},
  "mark": "tick",
  "encoding": {
    "x": {"field": "precipitation", "type": "quantitative"}
  }
}
```

It looks as though precipitation is skewed towards lower values; that is, when it rains, it usually doesn't rain very much. To better see this, we can create a histogram of the precipitation data. For this, we have to add an encoding that uses a special field `*` that is aggregated with `count`. It is difficult to see patterns across continuous data, so we can therefore discretize the data by binning it into bins of width 10 mm.

Tutorials & Documentations

```
{
  "$schema": "https://vega.github.io/schema/vega-lite/v2.json",
  "data": {"url": "data/seattle-weather.csv"},
  "mark": "bar",
  "encoding": {
    "x": {"field": "precipitation", "type": "quantitative", "bin": true}
  }
}
```

Overview

[Edit this page](#)

Vega-Lite is a high-level visualization grammar. It provides a concise JSON syntax for supporting rapid generation of visualizations to support analysis. Vega-Lite can serve as a declarative format for describing and creating data visualizations. Vega-Lite specifications can be compiled to a lower-level, more detailed [Vega](#) specifications and rendered using Vega's compiler.

This documentation describes the [JSON specification language](#) and how to use Vega-Lite visualizations in a web application.

Table of Contents

Below is an overview of the documentation for Vega-Lite properties. See [below](#) for an overview of Vega-Lite specifications.

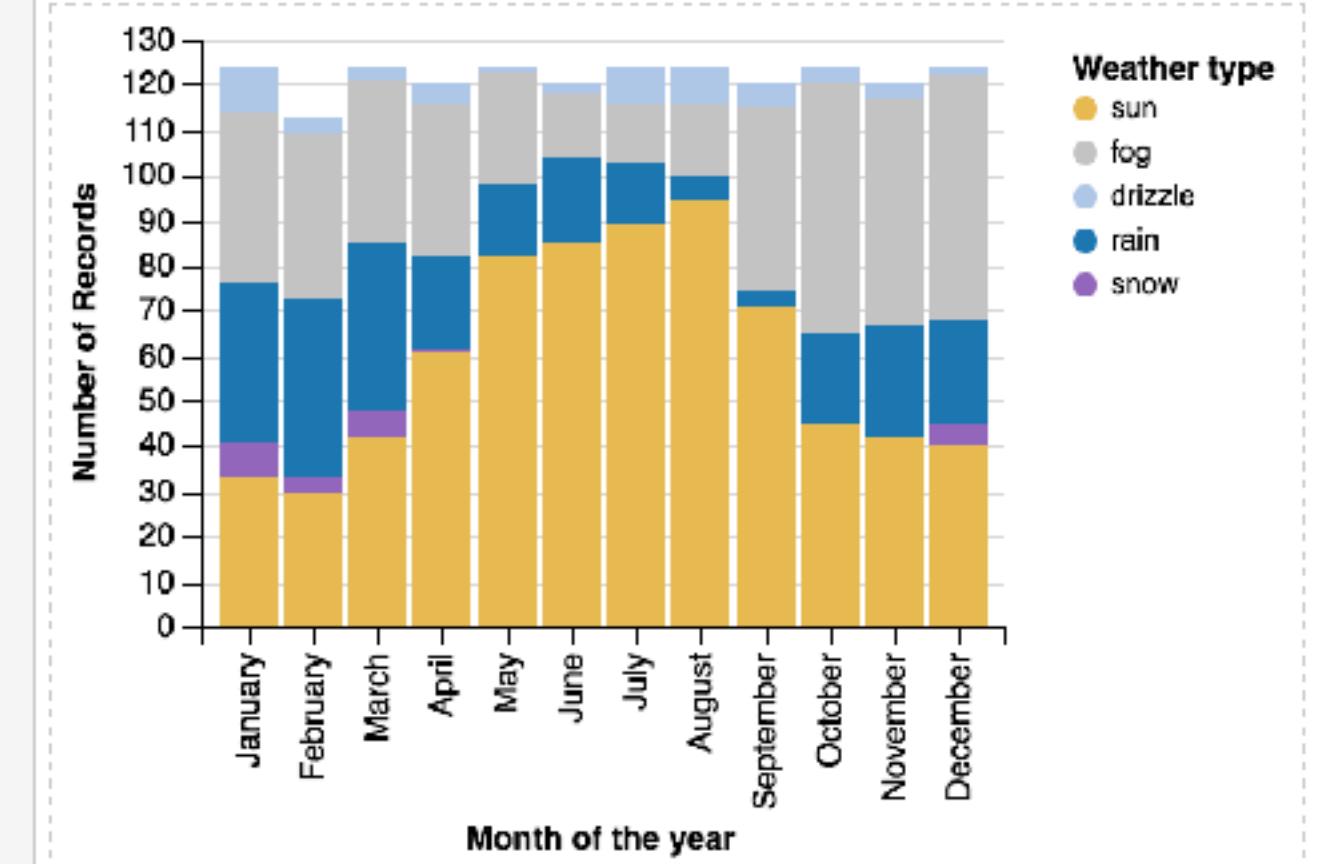
- [Overview](#)
 - [Table of Contents](#)
- [Spec](#)
 - [Vega-Lite Specifications](#)
- [Data](#)
 - [Inline Data](#)
 - [Data from URL](#)
 - [Format](#)
- [Transform](#)
 - [Top-level Transform Property](#)
- [Mark](#)
 - [Point](#)
 - [Text](#)
 - [Tick](#)
 - [Bar](#)
 - [Rect](#)
 - [Line](#)

vega.github.io/vega-lite

```

1  {
2    "data": {"url": "data/seattle-weather.csv", "format": {"type": "csv"}},
3    "mark": "bar",
4    "encoding": {
5      "x": {
6        "field": "date",
7        "type": "temporal",
8        "timeUnit": "month",
9        "axis": {"title": "Month of the year"}
10     },
11     "y": {
12       "field": "*",
13       "type": "quantitative",
14       "aggregate": "count"
15     },
16     "color": {
17       "field": "weather",
18       "type": "nominal",
19       "scale": {
20         "domain": ["sun", "fog", "drizzle", "rain", "snow"],
21         "range": ["#e7ba52", "#c7c7c7", "#aec7e8", "#1f77b4", "#9467bd"]
22       },
23     "legend": {
24       "title": "Weather type"
25     }
26   }
27 }
28 }
29 }
```

Visualization



Vega compiled from Vega-Lite (read-only)

Edit Vega spec

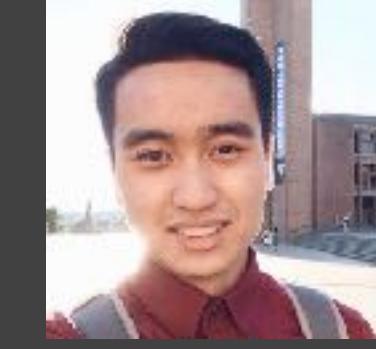
```

1  {
2    "width": 1,
3    "height": 1,
4    "padding": "auto",
5    "data": [
6      {"date": "2010-01-01", "weather": "sun", "count": 35}, {"date": "2010-01-01", "weather": "fog", "count": 5}, {"date": "2010-01-01", "weather": "drizzle", "count": 10}, {"date": "2010-01-01", "weather": "rain", "count": 40}, {"date": "2010-01-01", "weather": "snow", "count": 2}, {"date": "2010-02-01", "weather": "sun", "count": 30}, {"date": "2010-02-01", "weather": "fog", "count": 25}, {"date": "2010-02-01", "weather": "drizzle", "count": 10}, {"date": "2010-02-01", "weather": "rain", "count": 15}, {"date": "2010-02-01", "weather": "snow", "count": 0}, {"date": "2010-03-01", "weather": "sun", "count": 45}, {"date": "2010-03-01", "weather": "fog", "count": 30}, {"date": "2010-03-01", "weather": "drizzle", "count": 10}, {"date": "2010-03-01", "weather": "rain", "count": 15}, {"date": "2010-03-01", "weather": "snow", "count": 0}, {"date": "2010-04-01", "weather": "sun", "count": 60}, {"date": "2010-04-01", "weather": "fog", "count": 20}, {"date": "2010-04-01", "weather": "drizzle", "count": 10}, {"date": "2010-04-01", "weather": "rain", "count": 15}, {"date": "2010-04-01", "weather": "snow", "count": 0}, {"date": "2010-05-01", "weather": "sun", "count": 80}, {"date": "2010-05-01", "weather": "fog", "count": 20}, {"date": "2010-05-01", "weather": "drizzle", "count": 10}, {"date": "2010-05-01", "weather": "rain", "count": 15}, {"date": "2010-05-01", "weather": "snow", "count": 0}, {"date": "2010-06-01", "weather": "sun", "count": 90}, {"date": "2010-06-01", "weather": "fog", "count": 20}, {"date": "2010-06-01", "weather": "drizzle", "count": 10}, {"date": "2010-06-01", "weather": "rain", "count": 15}, {"date": "2010-06-01", "weather": "snow", "count": 0}, {"date": "2010-07-01", "weather": "sun", "count": 95}, {"date": "2010-07-01", "weather": "fog", "count": 20}, {"date": "2010-07-01", "weather": "drizzle", "count": 10}, {"date": "2010-07-01", "weather": "rain", "count": 15}, {"date": "2010-07-01", "weather": "snow", "count": 0}, {"date": "2010-08-01", "weather": "sun", "count": 98}, {"date": "2010-08-01", "weather": "fog", "count": 20}, {"date": "2010-08-01", "weather": "drizzle", "count": 10}, {"date": "2010-08-01", "weather": "rain", "count": 15}, {"date": "2010-08-01", "weather": "snow", "count": 0}, {"date": "2010-09-01", "weather": "sun", "count": 70}, {"date": "2010-09-01", "weather": "fog", "count": 20}, {"date": "2010-09-01", "weather": "drizzle", "count": 10}, {"date": "2010-09-01", "weather": "rain", "count": 15}, {"date": "2010-09-01", "weather": "snow", "count": 0}, {"date": "2010-10-01", "weather": "sun", "count": 55}, {"date": "2010-10-01", "weather": "fog", "count": 20}, {"date": "2010-10-01", "weather": "drizzle", "count": 10}, {"date": "2010-10-01", "weather": "rain", "count": 15}, {"date": "2010-10-01", "weather": "snow", "count": 0}, {"date": "2010-11-01", "weather": "sun", "count": 45}, {"date": "2010-11-01", "weather": "fog", "count": 20}, {"date": "2010-11-01", "weather": "drizzle", "count": 10}, {"date": "2010-11-01", "weather": "rain", "count": 15}, {"date": "2010-11-01", "weather": "snow", "count": 0}, {"date": "2010-12-01", "weather": "sun", "count": 40}, {"date": "2010-12-01", "weather": "fog", "count": 20}, {"date": "2010-12-01", "weather": "drizzle", "count": 10}, {"date": "2010-12-01", "weather": "rain", "count": 15}, {"date": "2010-12-01", "weather": "snow", "count": 0}
```

Online Editor

vega.github.io/new-editor

Contributors : Vega-Lite, Altair & Voyager



and many more...

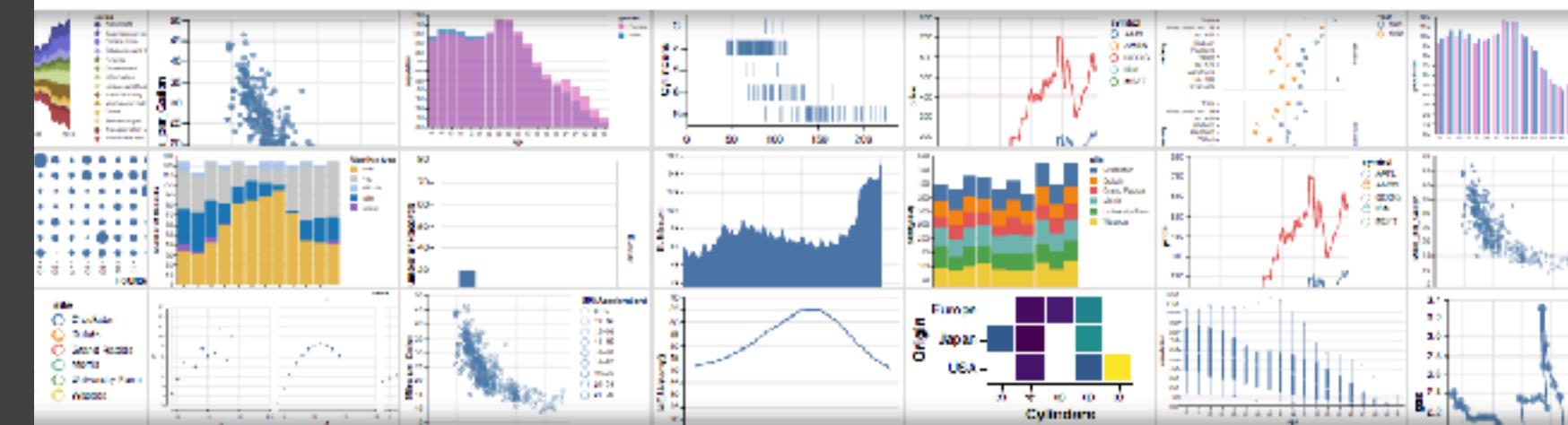


Kanit "Ham" Wongsuphasawat @kanitw
 Dominik Moritz @domoritz
 Arvind Satyanarayan @arvindsatya1
 Jeffrey Heer @jeffrey_heer

Interactive Data Lab @uwdata
 University of Washington



Vega-Lite – A High-Level Visualization Grammar



Vega-Lite is a high-level visualization grammar. It provides a concise JSON syntax for supporting rapid generation of visualizations to support analysis. Vega-Lite specifications can be compiled to [Vega](#) specifications.

Vega-Lite specifications describe visualizations as mappings from data to properties of graphical marks (e.g., points or bars). It automatically produces visualization components including axes, legends, and scales. It then determines properties of these components based on a set of carefully designed rules. This approach allows Vega-Lite specifications to be succinct and expressive, but also provide user control. As Vega-Lite is designed for analysis, it supports data transformations such as aggregation, binning, filtering, sorting, and visual transformations including stacking and faceting.

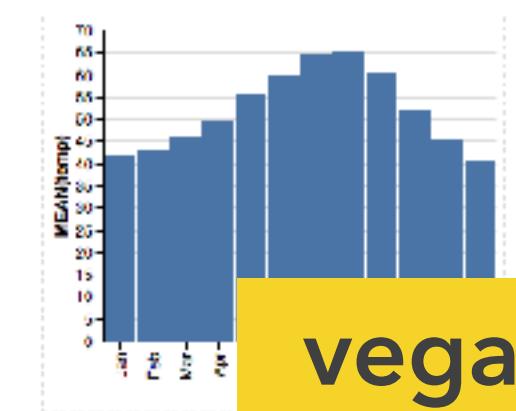
Read our [Introduction article on Medium](#), check out the [documentation](#) and take a look at our [example gallery](#).

[Get started](#)

[Try online](#)

Example

This is a Vega-Lite specification to create a bar chart that shows the average temperature in Seattle for each month.



```
{
  "schema": "https://vega.github.io/schemas/vega-lite/v2.json",
  "data": {"url": "data/seattle-temps.csv"},
  "mark": "bar",
  "encoding": {
    "x": {
      "binwidth": "month",
      "field": "date",
      "type": "temporal"
    }
  }
}
```

vega.github.io/vega-lite

[Open In Vega Editor](#)