Vega-Lite

TAs Lucas and Jong

Agenda

- Concepts Recap
  - Data, Mark, Transform, Encoding
- Live Demo
  - Distribution of different kinds of weather in Seattle
- Exercises
  - Exercise 1: Aggregate mean

Vega-Lite recap

- Data inline
  - JSON Array
  - Each row is an object in the array.

```
{
  "data": {
    "values": [
      {"a": "C", "b": 2},
      {"a": "C", "b": 7},
      {"a": "C", "b": 4},
      {"a": "D", "b": 1},
      {"a": "D", "b": 2},
      {"a": "D", "b": 6},
      {"a": "E", "b": 8},
      {"a": "E", "b": 4},
      {"a": "E", "b": 7}
    ]
  }
}
```

Vega-Lite recap

- Data from source
  - Can be imported through URL
  - Runtime datasource API (not addressed here)

```
{
  "data": {"url": "https://vega.github.io/editor/data/seattle-weather.csv" },

}
```

Vega-Lite recap

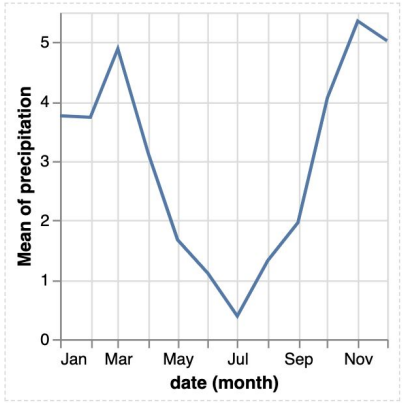
- Marks
  - Shapes to visually encode data

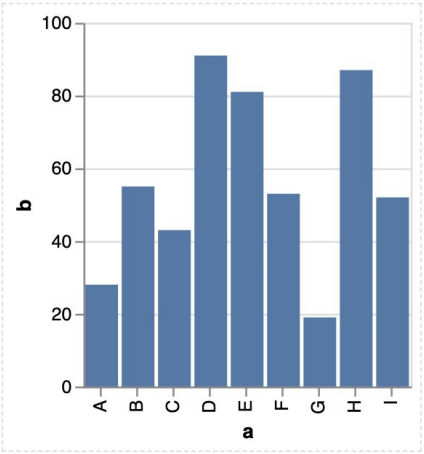
```
{ ...
  "mark": {
    "type": ...,// mark Object
    ...
  },
 ...
}
```

```
{ ...
  "mark": "..." ,// mark type
 ...
}
```

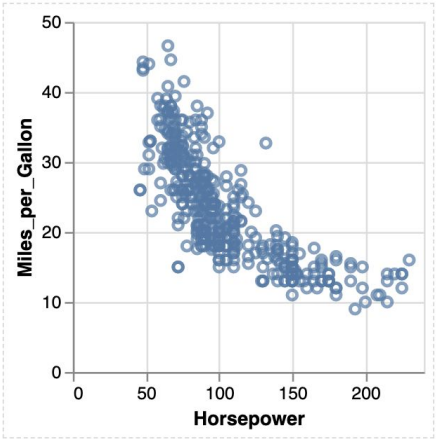
Vega-Lite recap

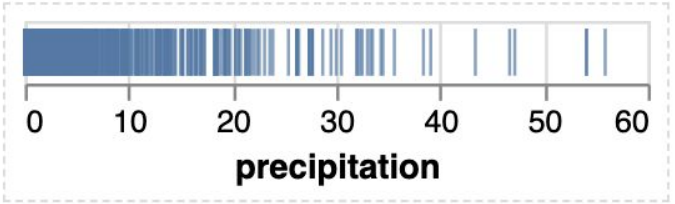
- Marks
  - Shapes to visually encode data
  - Primitive types: area, bar, circle, line, point, rect, rule, square, text, tick, and geoshape





Open in Vega Editor





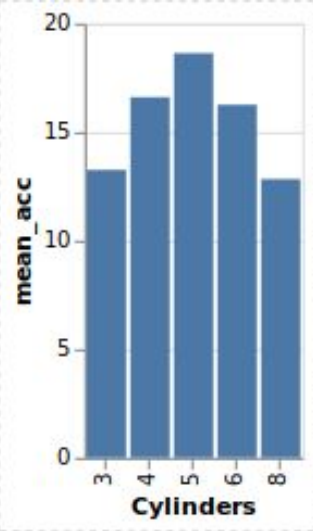
Vega-Lite Recap

- Transform
  - Describe transformations on the data
  - **view-level** or **field transforms** inside “encoding” (more on that later)
  - View-level are executed **in order**
  - Inline transforms’ execution order:
    - bin, timeUnit, aggregate, sort, stack
- Some examples
  - Filter, Aggregate, Bin

```
{    // view-level transforms
  ...
  "transform": [
    { ... : { ... }},
    { ... : { ... }}
  ],
  ...
}
```

Vega-Lite Recap

- Transform example - view level
  - data/cars.json
  - Aggregate mean of acceleration, group by number of cylinders



Name:	"chevrolet chevelle
Miles_per_Gallon:	18
Cylinders:	8
Displacement:	307
Horsepower:	130
Weight_in_lbs:	3504
Acceleration:	12
Year:	"1970-01-01"
Origin:	"USA"

```
{
  "data": { "url": "data/cars.json" },
  "transform": [
    {
      "aggregate": [{
       "op": "mean",   // Operation
       "field": "Acceleration",    // Data
       "as": "mean_acc" // Output
      }],
      "groupby": ["Cylinders"] // group
    }
  ],
  "mark": "bar",
  "encoding": {
    "x": { "field": "Cylinders", "type":
"ordinal" },
    "y": { "field": "mean_acc", "type":
"quantitative" }
  }
}
```

Vega-Lite Recap

- Encoding
  - Maps **encoding channels** to **data fields** or **constant values**
- Encoding channels
  - Position channels
    - x, y, x2, y2
  - Mark property channels
    - color, opacity, shape, size
  - Tooltip, Hyperlink channels
    - Text, tooltip, href (explained later)

```
// Specification of a Single View
{
    ...,
    "encoding": {
      "x": { // Encoding channel
        "field": ...,
        "type": ...,
        ...
      },
      ...
    },
    ...
  }
```

Vega-Lite Recap

- Channel definition
  - Either a **field** definition or a **value** definition
- Field definition
  - Encodes a particular field in the dataset with an encoding channel
  - “field” : String defining the name of the field from which to pull data from
  - “type” : Type of measurement
    - quantitative, ordinal, nominal, temporal

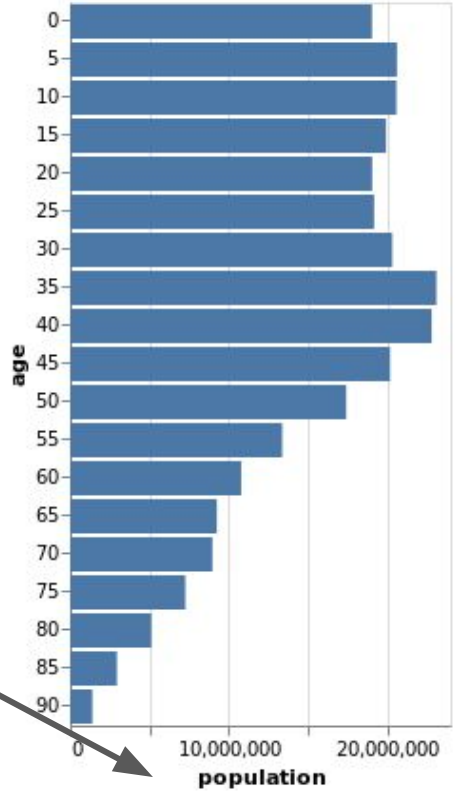
```
{ // Specification of a Single View
  ...,
  "encoding": {
    "x": { // Field definition
      "field": ..., // Required
      "type": ...,  // Required
      ...
    },
    ...
  },
  ...
}
```

```
{ // Specification of a Single View
  ...,
  "encoding": {
    "x": {  // Constant value definition
      "value": ...
    },
    ...
  },
}
```

Vega-Lite Recap

- Encoding types
  - Quantitative: for data that expresses **some kind of quantity** (e.g., population number)
  - Temporal: for **dates** and **times**
  - Ordinal: for discrete **ranked data** that can be sorted
  - Nominal: for **categorical data**, doesn't determine magnitude or ordering

"y" is typed as ordinal in the class population demo



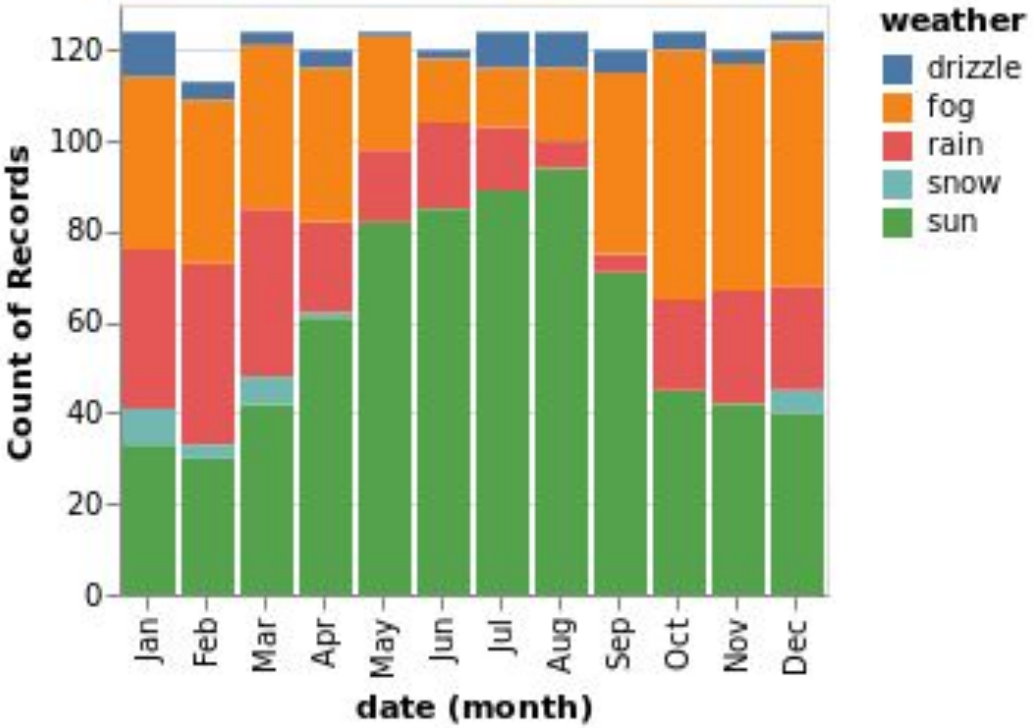
"x" is typed as quantitative in the class population demo

Vega-Lite Recap

- Marks as clickable data poi
  - Add properties:
    - Transform
    - Tooltip
    - href

```
{...
  "transform": [{
    "calculate": "'https://www.google.com/search?q=' +
datum.Name", "as": "url"
  }],
  "encoding": {
    ...
    "tooltip": {"field": "Name", "type": "nominal"},
    "href": {"field": "url", "type": "nominal"}
  }
}
```

Vega-Lite Live Demo

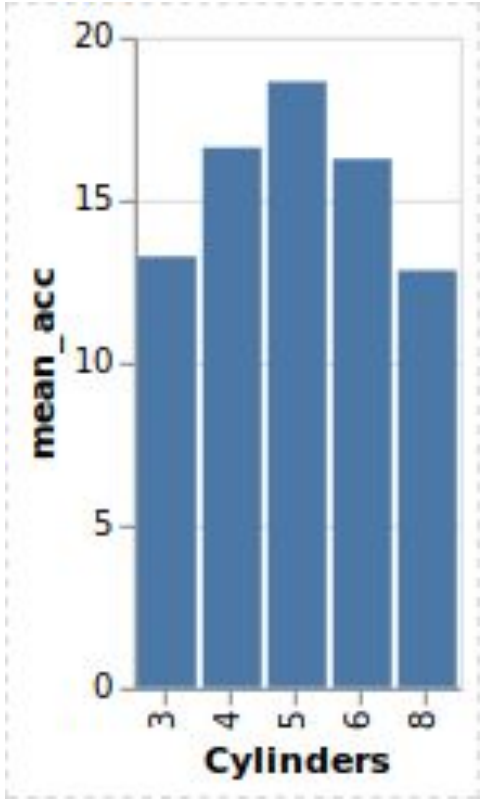


Vega-Lite Exercise 1

- Vega-Lite Docs: https://vega.github.io/vega-lite/docs/
- Using data from this URL: https://vega.github.io/editor/data/cars.json,
- Produce a bar chart that aggregates mean of Acceleration grouped by the number of cylinders using **encoding field definition**

  (No transform)

```
Name:           "buick skylark 320"
Miles_per_Gallon:  15
Cylinders:      8
Displacement:   350
Horsepower:     165
Weight_in_lbs:  3693
Acceleration:   11.5
Year:           "1970-01-01"
Origin:         "USA"
```



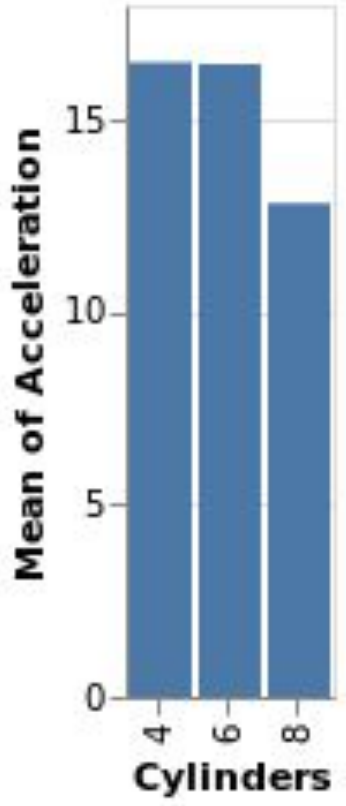
Vega-Lite Exercise 1 Solution

```
{
  "data": {"url": "https://vega.github.io/editor/data/cars.json"},
  "mark": "bar",
  "encoding": {
    "x": {"field": "Cylinders", "type": "ordinal"},
    "y": {"aggregate": "mean", "field": "Acceleration", "type": "quantitative"}
  }
}
```

Vega-Lite Exercise 1-1

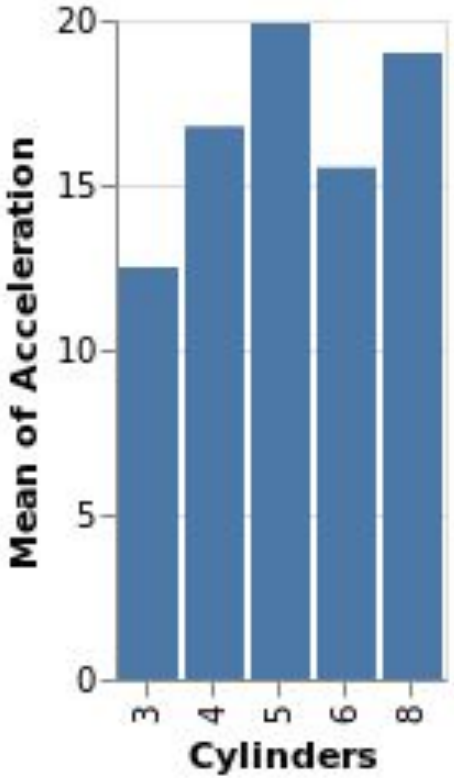
- From the previous exercise 1,
  - Data: https://vega.github.io/editor/data/cars.json
- Produce a bar chart that aggregates mean of Acceleration grouped by the number of cylinders using encoding field definition **only for cars produced in the USA**


```
Name:           "buick skylark 320"
Miles_per_Gallon:  15
Cylinders:      8
Displacement:   350
Horsepower:     165
Weight_in_lbs:  3693
Acceleration:   11.5
Year:           "1970-01-01"
Origin:         "USA"
```

Vega-Lite Exercise 1-2

- From the previous exercise 1,
- Produce a bar chart that aggregates mean of Acceleration grouped by the number of cylinders using encoding field definition **only for cars produced at or after 1979**



Vega-Lite Exercise Solutions 1-1, 1-2

```
{
  "data": {"url": "https://vega.github.io/editor/data/cars.json"},
  "mark": "bar",
  "transform": [
      //{"filter": {"field": "Origin", "equal": "USA"}}  1-1 Solution
      {"filter": {"timeUnit": "year", "field": "Year", "gte": "1979"}}
  ],
  "encoding": {
    "x": {"field": "Cylinders", "type": "ordinal"},
    "y": {"aggregate": "mean", "field": "Acceleration", "type": "quantitative"}
  }
}
```