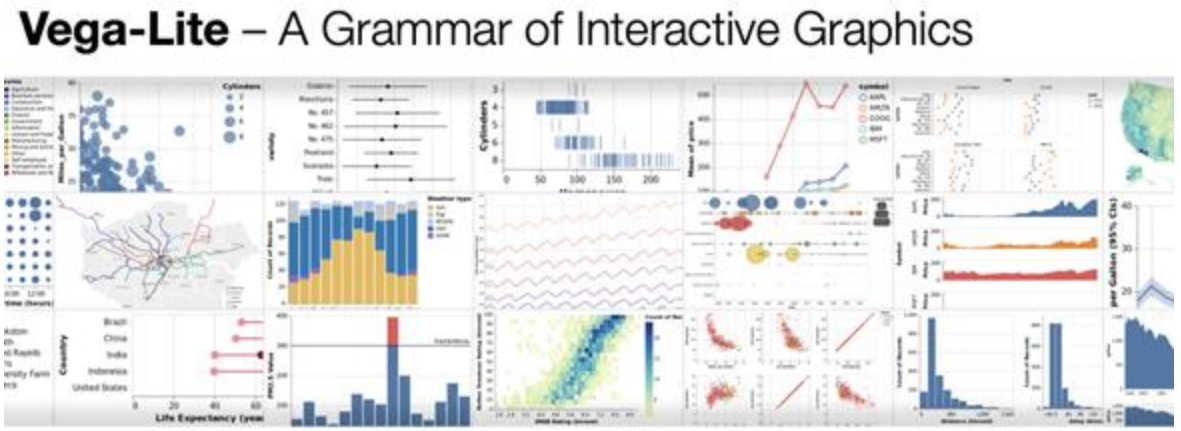
SI649 Altair Demo

Gabriel Grill

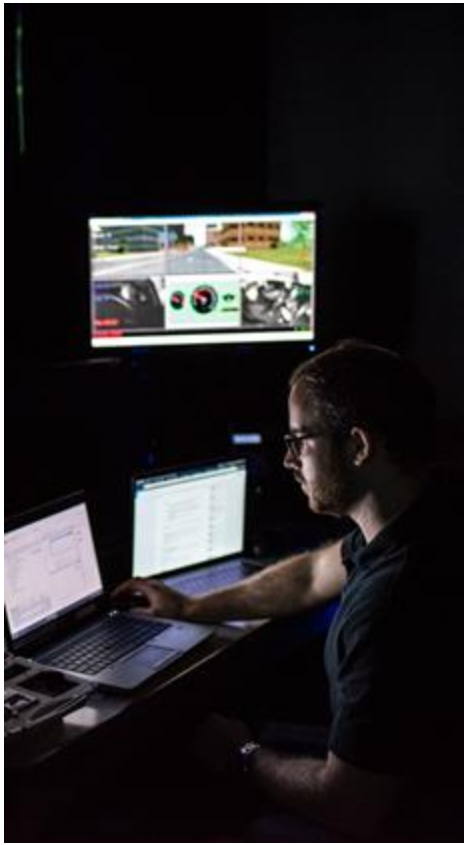
Originally Designed by Liang Sie

What is **Altair**

- Altair is a Python API to Vega-Lite



Section 1
Scatterplot



Encoding

Translates your code to the JSON format that Vega-Lite understands (and can render).

Color

- Input

```
alt.Color('transmission',type='nominal')
```

- Output

```
Color({
  shorthand: 'transmission',
  type: 'nominal'
})
```

- "encode the nominal transmission variable using color."

Load Data

- Input

```
mtcars = pd.read_csv("https://raw.githubusercontent.com/…/mtcars.csv")
mtcars.sample(5)
```

- Output

	model	MPG	cylinders	displacement	HP	rear_axle_ratio	weight	qsec	vs	transmission	gears	carb
29	Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.5	0	1	5	6
25	Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.9	1	1	4	1
28	Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.5	0	1	5	4
27	Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.9	1	1	5	2
26	Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.7	0	1	5	2

Marks

Specify how exactly those attributes should be represented on the plot.

Altair provides a number of basic mark properties:

Mark Name	Method	Description	Example
arc	`mark_arc()`	A pie chart.	Pie Chart
area	`mark_area()`	A filled area plot.	Simple Stacked Area Chart
bar	`mark_bar()`	A bar plot.	Simple Bar Chart
circle	`mark_circle()`	A scatter plot with filled circles.	One Dot Per Zipcode
geoshape	`mark_geoshape()`	A geographic shape	Choropleth Map
image	`mark_image()`	A scatter plot with image markers.	Image Mark
line	`mark_line()`	A line plot.	Simple Line Chart
point	`mark_point()`	A scatter plot with configurable point shapes.	Multi-panel Scatter Plot with Linked Brushing
rect	`mark_rect()`	A filled rectangle, used for heatmaps	Simple Heatmap
rule	`mark_rule()`	A vertical or horizontal line spanning the axis.	Candlestick Chart
square	`mark_square()`	A scatter plot with filled squares.	N/A
text	`mark_text()`	A scatter plot with points represented by text.	Bar Chart with Labels
tick	`mark_tick()`	A vertical or horizontal tick mark.	Simple Strip Plot
trail	`mark_trail()`	A line with variable widths.	Line Chart with Varying Size

Scatterplot

- `alt.Chart(data).mark_point()` >> only gives you a single dot
- Data Encoding (Vega Lite Data Type)
  - `alt.X('MPG',type='quantitative')`

    OR
  - `alt.X('MPG:Q')`

    OR
  - `x='MPG'`
-
```
alt.Chart(data).mark_point().encode(
    x='???',
    y='???'
)
```

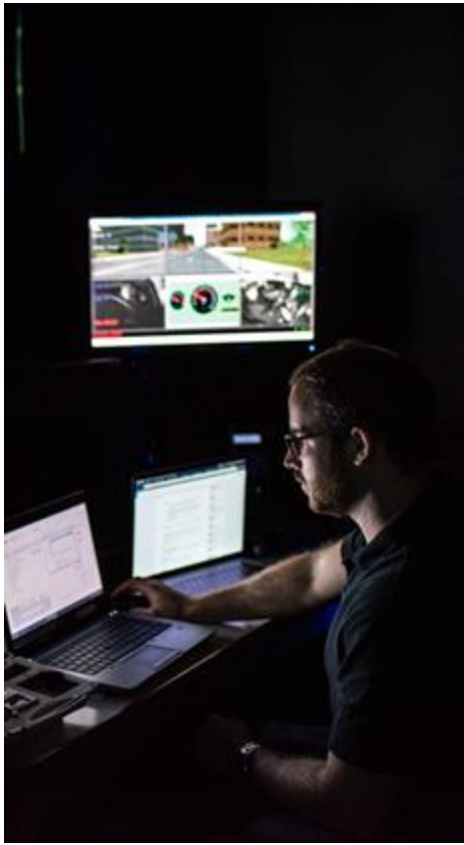
Scatterplot

X-axis = MPG
Y-axis = weight

- `alt.Chart(mtcars).mark_point()` >> only gives you a single dot

- ```
  alt.Chart(mtcars).mark_point().encode(
      x='MPG',
      y='weight'
  )
  ```
  ```

Section 2
Variation



Scatterplot

Change color
Color = transmission

- ```
  alt.Chart(mtcars).mark_point().encode(
      x='MPG',
      y="weight",
      color="transmission"
  )
  ```

- ```
  alt.Chart(mtcars).mark_point().encode(
      x='MPG',
      y="weight",
      color="transmission:N"
  )
  ```

- What's the difference? Why?

Scatterplot

Fill the mark

```
alt.Chart(mtcars).mark_point(filled=True).encode(
    x='MPG',
    y="weight",
    color="transmission"
)
```

Scatterplot **Practice**

- Use the shape to display the transmission variable
- Change the size of points

Scatterplot Answer

```
alt.Chart(mtcars).mark_point(filled=True, size=100).encode(
  x='MPG',
  y="weight",
  color="transmission:N",
  shape="transmission:N"
)
```

Assignment Hint

- Axis title

```
x=alt.X('field', title='name')
```

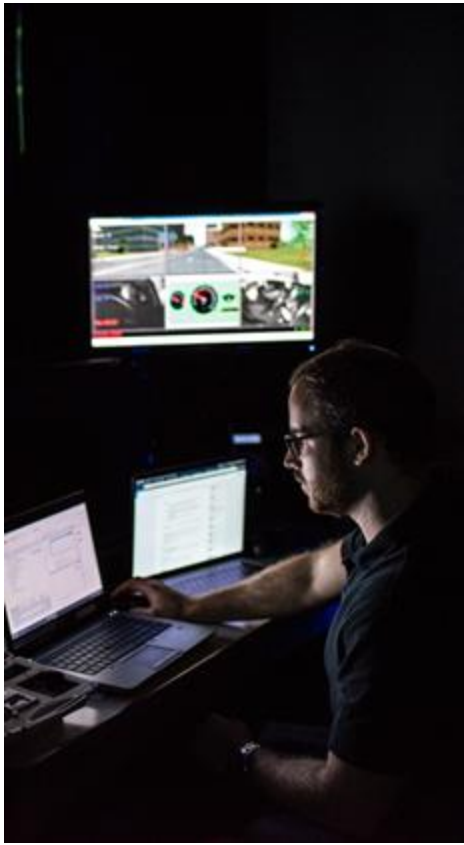
- Color or Opacity Scale

```
scale=alt.Scale(domain=['value', 'value'], range=['color', 'color']
```

- Opacity

```
opacity=alt.Opacity('field', scale…
```

Section 3
Bars and Binning



Marks

Specify how exactly those attributes should be represented on the plot.

Altair provides a number of basic mark properties:

Mark Name	Method	Description	Example
arc	`mark_arc()`	A pie chart.	Pie Chart
area	`mark_area()`	A filled area plot.	Simple Stacked Area Chart
bar	`mark_bar()`	A bar plot.	Simple Bar Chart
circle	`mark_circle()`	A scatter plot with filled circles.	One Dot Per Zipcode
geoshape	`mark_geoshape()`	A geographic shape	Choropleth Map
image	`mark_image()`	A scatter plot with image markers.	Image Mark
line	`mark_line()`	A line plot.	Simple Line Chart
point	`mark_point()`	A scatter plot with configurable point shapes.	Multi-panel Scatter Plot with Linked Brushing
rect	`mark_rect()`	A filled rectangle, used for heatmaps	Simple Heatmap
rule	`mark_rule()`	A vertical or horizontal line spanning the axis.	Candlestick Chart
square	`mark_square()`	A scatter plot with filled squares.	N/A
text	`mark_text()`	A scatter plot with points represented by text.	Bar Chart with Labels
tick	`mark_tick()`	A vertical or horizontal tick mark.	Simple Strip Plot
trail	`mark_trail()`	A line with variable widths.	Line Chart with Varying Size

Bar Chart

X-axis = model
Y-axis = MPG
OR
X-axis = MPG
Y-axis = model

Vertical

```
alt.Chart(mtcars).mark_bar().encode(
    x='model',
    y='MPG'
)
```

Horizontal

```
alt.Chart(mtcars).mark_bar().encode(
    y='model',
    x='MPG'
)
```

Bins

Based on transmission

Pandas

- `binned = mtcars.groupby("transmission").count().reset_index()[['transmission','model']]`
- `binned = binned.rename(columns={'model':'count'})`

	transmission	count
0	0	19
1	1	13

Altair

-
```
alt.Chart(binned).mark_bar().encode(
    x='transmission:N',
    y='count'
)
```

Bins

Based on transmission

Aggregate data

- ```
  alt.Chart(mtcars).mark_bar().encode(
      x=alt.X('transmission:N',bin=True),
      y=alt.Y('count()'), # tell altair how we aggregate the data
  )
  ```
- ```
  alt.Chart(mtcars).mark_bar().encode(
      x=alt.X('transmission:N',bin=True),
      y=alt.Y('max(HP)') # see the max of horsepower
  )
  ```

Bins

Based on weight

Numeric variable as bins

```
alt.Chart(mtcars).mark_bar().encode(
    x=alt.X('weight',bin=True),
    y=alt.Y('count()')
)
```

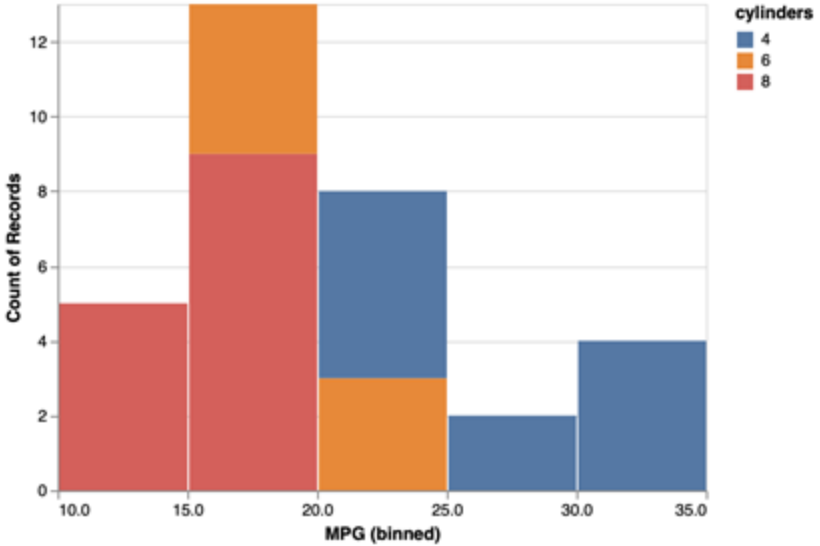
Modify bins

```
alt.Chart(mtcars).mark_bar().encode(
    x=alt.X('weight',bin=alt.BinParams(maxbins=20)),
    y=alt.Y('count()')
)
```

Bars and Bins
Practice

Create a stacked bar chart

- For each MPG bins, how many cars are there? How many 2-cylinder, 4-cylinder, 6-cylinder cars are there in each bin?

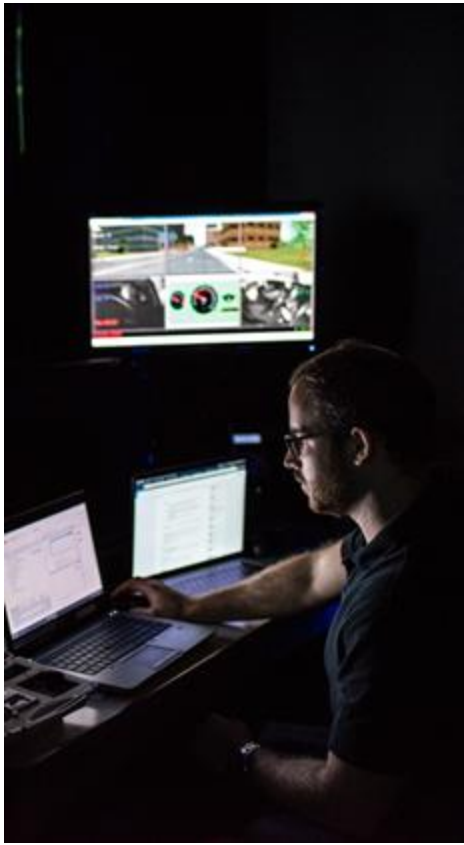


Bars and Bins
Answer

Create a stacked bar chart

```
alt.Chart(mtcars).mark_bar().encode(
    alt.X('MPG', bin=True),
    alt.Y('count()'),
    alt.Color('cylinders:N')
)
```

Section 4
Sorting



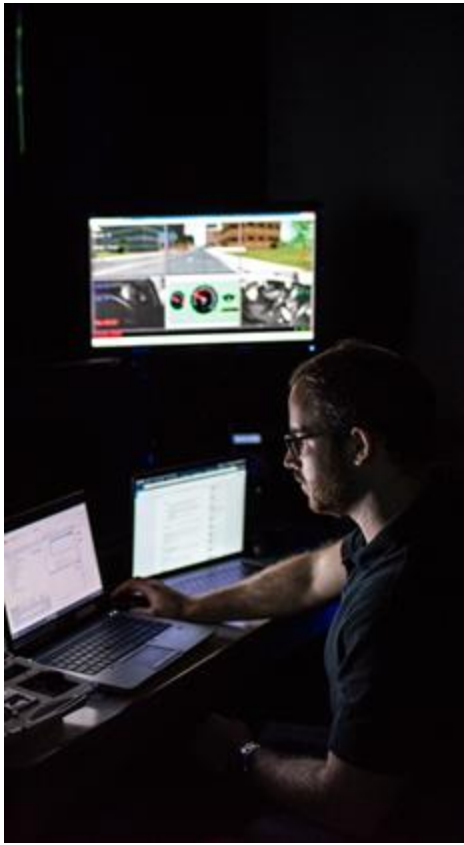
Sorting

Sort by MPG

EncodingSortField

```
alt.Chart(mtcars).mark_bar().encode(
    x='MPG',
    y=alt.Y('model',
        sort=alt.EncodingSortField(
            field="MPG",  # The field to use for the sort
            order="descending"  # The order to sort in
        )
    )
)
```

Section 5
Other Marks



Marks

Specify how exactly those attributes should be represented on the plot.

Altair provides a number of basic mark properties:

Mark Name	Method	Description	Example
arc	`mark_arc()`	A pie chart.	Pie Chart
area	`mark_area()`	A filled area plot.	Simple Stacked Area Chart
bar	`mark_bar()`	A bar plot.	Simple Bar Chart
circle	`mark_circle()`	A scatter plot with filled circles.	One Dot Per Zipcode
geoshape	`mark_geoshape()`	A geographic shape	Choropleth Map
image	`mark_image()`	A scatter plot with image markers.	Image Mark
line	`mark_line()`	A line plot.	Simple Line Chart
point	`mark_point()`	A scatter plot with configurable point shapes.	Multi-panel Scatter Plot with Linked Brushing
rect	`mark_rect()`	A filled rectangle, used for heatmaps	Simple Heatmap
rule	`mark_rule()`	A vertical or horizontal line spanning the axis.	Candlestick Chart
square	`mark_square()`	A scatter plot with filled squares.	N/A
text	`mark_text()`	A scatter plot with points represented by text.	Bar Chart with Labels
tick	`mark_tick()`	A vertical or horizontal tick mark.	Simple Strip Plot
trail	`mark_trail()`	A line with variable widths.	Line Chart with Varying Size

Heatmap

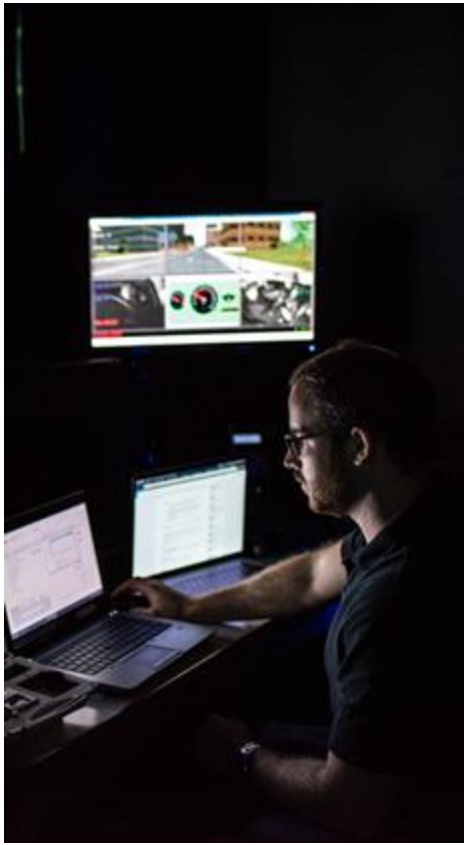
X-axis = MPG
Y-axis = HP

mark_rect()

```
alt.Chart(mtcars).mark_rect().encode(
    x=alt.X('MPG', bin=True),   # Create bins for MPG
    y=alt.Y('HP', bin=True),   # Create bins for HP
    color='count()'    # tell altair how we aggregate the data
)
```

rect	`mark_rect()`	A filled rectangle, used for heatmaps	Simple Heatmap

Section 6
Compound Chart



Compound

Side by side
Bar 1: Model vs. MPG
Bar 2: Model vs. HP

Use "|" operation to plot charts side by side

```
model_mpg= alt.Chart(mtcars).mark_bar().encode(
    x=alt.X('MPG'),
    y=alt.Y("model")
)
model_hp= alt.Chart(mtcars).mark_bar().encode(
    x=alt.X('HP'),
    y=alt.Y("model")
)
Model_mpg|model_hp
```

Use alt.hconcat() >> horizontal

```
alt.hconcat(model_mpg,model_hp)
```

Compound

One above the other
Bar 1: Model vs. MPG
Bar 2: Model vs. HP

Use "&" operation to plot charts one above the other

- `model_mpg & model_hp`

Use alt.vconcat() >> vertical

- `alt.vconcat(model_mpg,model_hp)`

Compound **Practice**

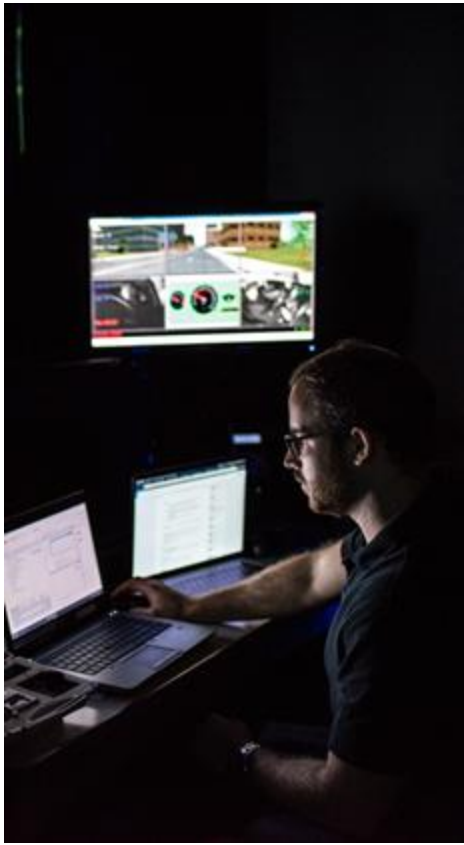
Share axis

Shared Y-axis

- `(model_mpg|model_hp).resolve_scale(y="shared")`

How about X-axis in a one of above chart?

Section 7
Styling



Styling

Change bar height

```
model_mpg= alt.Chart(mtcars,title="Mile Per
Gallon").mark_bar(height=5).encode(
    x=alt.X('MPG'),
    y=alt.Y("model", sort=alt.EncodingSortField(
            field="model",  # The field to use for the sort
            order="descending"  # The order to sort in
        ))
)
```

Styling

Change bar color

```
model_hp= alt.Chart(mtcars,title="Horse Power")
.mark_bar(color="pink").encode(
   x=alt.X('HP'),
   y=alt.Y("model", axis=None, sort=alt.EncodingSortField(
           field="model",  # The field to use for the sort
           order="descending"  # The order to sort in
       ))
)
```

Styling

Plot charts side by side and add chart title

```
(model_mpg|model_hp).resolve_scale(y='shared').properties(
    title="MPG and HP"
)
```

Assignment Hint

- Properties

```
.properties(
   width=???,
   height=???,
   title=alt.TitleParams(
       text='name',
       subtitle="name",
       subtitleFontSize = ??
   )
```

Themes

Set the theme to 'fivethirtyeight'

```
alt.themes.enable('fivethirtyeight')    # try vox, dark, etc.
```

```
alt.Chart(mtcars).mark_point(filled=True).encode(
    x='MPG',
    y="weight",
    color="transmission:N"
)
```

Information
changes
everything.

Thank You

SCHOOL OF INFORMATION
UNIVERSITY OF MICHIGAN