Visualization (Exploring Co-variation)

AUTHOR
Peter Ganong and Maggie Shi

PUBLISHED
October 14, 2024

```
DataTransformerRegistry.enable('default')
```

Table of contents

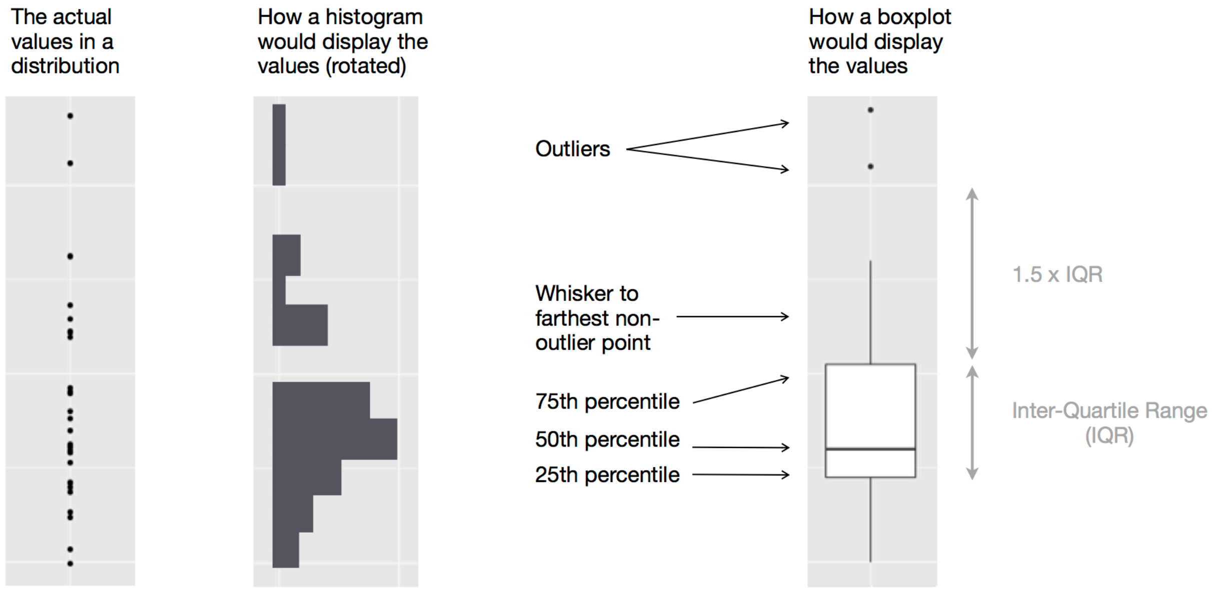
Categorical variable and continuous variable

```
from palmerpenguins import load_penguins
penguins = load_penguins()
display(penguins)
```

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex	year
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	male	2007
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	female	2007
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	female	2007
3	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN	2007
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0	female	2007
...	...	...	...	...	...	...	...	...
339	Chinstrap	Dream	55.8	19.8	207.0	4000.0	male	2009
340	Chinstrap	Dream	43.5	18.1	202.0	3400.0	female	2009
341	Chinstrap	Dream	49.6	18.2	193.0	3775.0	male	2009
342	Chinstrap	Dream	50.8	19.0	210.0	4100.0	male	2009
343	Chinstrap	Dream	50.2	18.7	198.0	3775.0	female	2009

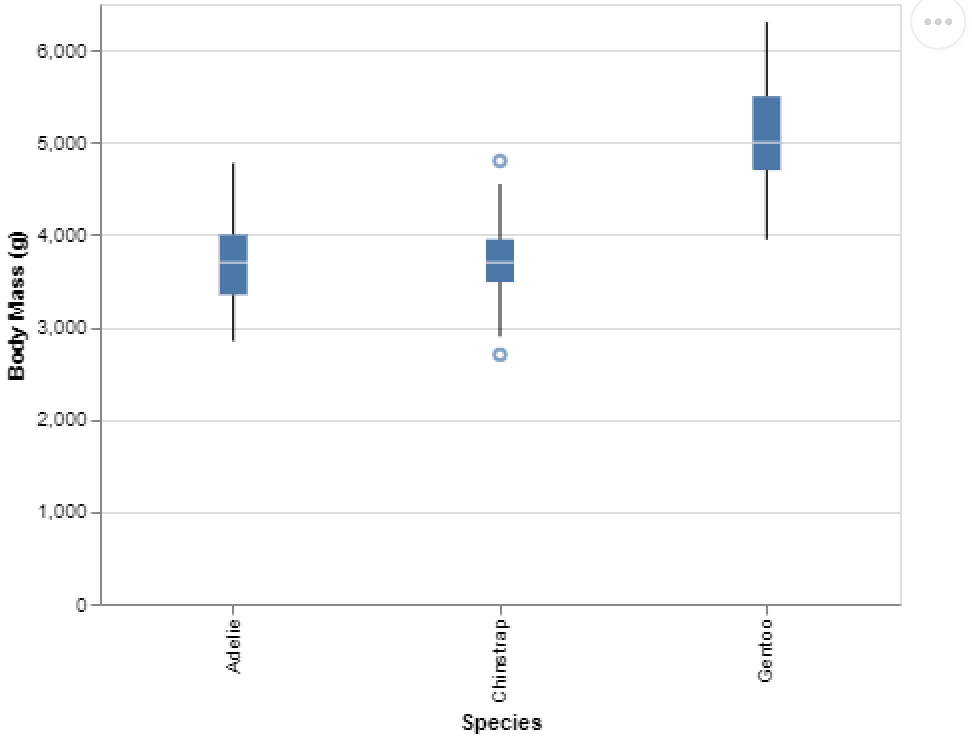
344 rows × 8 columns

numeric & categorical: box plot


The actual values in a distribution
How a histogram would display the values (rotated)
How a boxplot would display the values
Outliers
Whisker to farthest non-outlier point
75th percentile
50th percentile
25th percentile
1.5 x IQR
Inter-Quartile Range (IQR)

numeric & categorical: `mark_boxplot()`

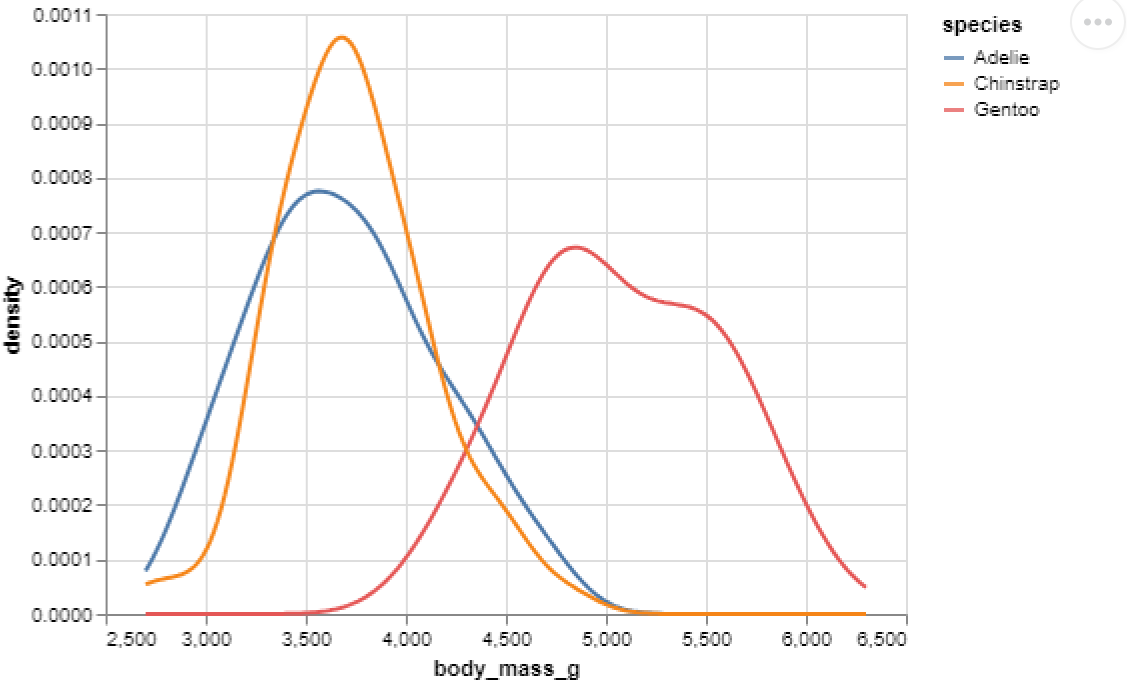
```
alt.Chart(penguins).mark_boxplot().encode(
    x=alt.X('species:N', title="Species"),
    y=alt.Y('body_mass_g:Q', title="Body Mass (g)"),
).properties(
    width=400,
    height=300
)
```



Discussion question: what do you notice from this graph?

numeric & categorical: `transform_density()`

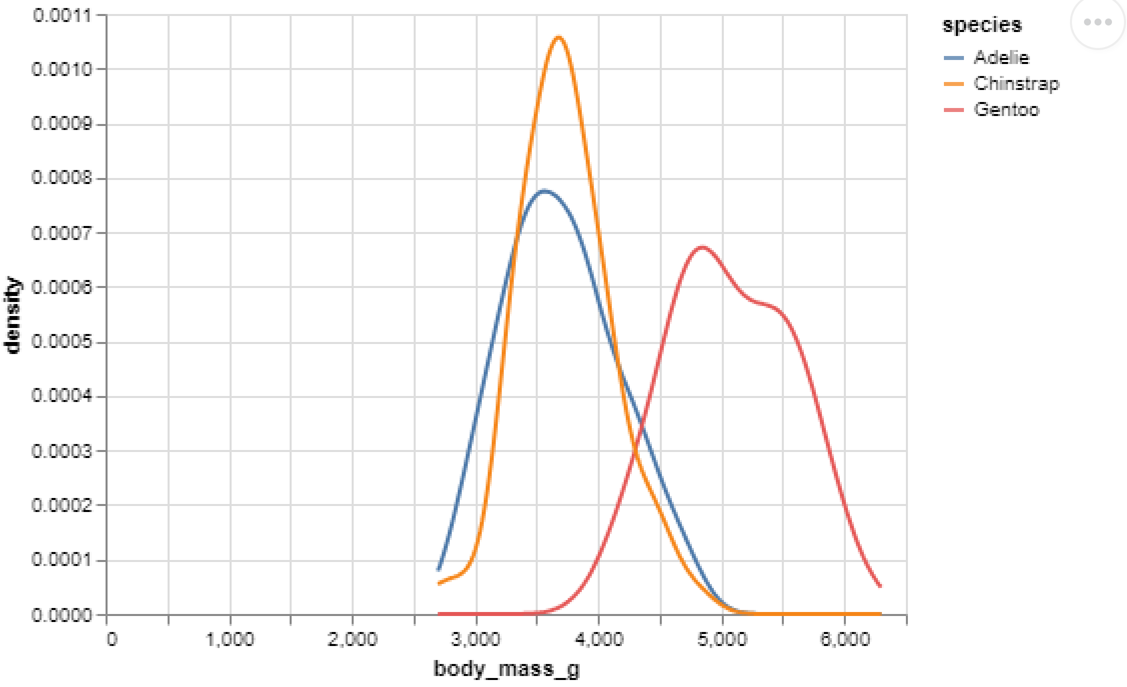
```
alt.Chart(penguins).transform_density(
    'body_mass_g',
        groupby=['species'],
        as_=['body_mass_g', 'density']
    ).mark_line().encode(
        alt.X('body_mass_g:Q'),
        alt.Y('density:Q', stack=None),
        alt.Color('species:N')
    ).properties(width=400,height=300)
```



numeric & categorical: `transform_density()`

Discussion q – What if we required the x-axis range to include zero? Would that improve or reduce clarity? How come?

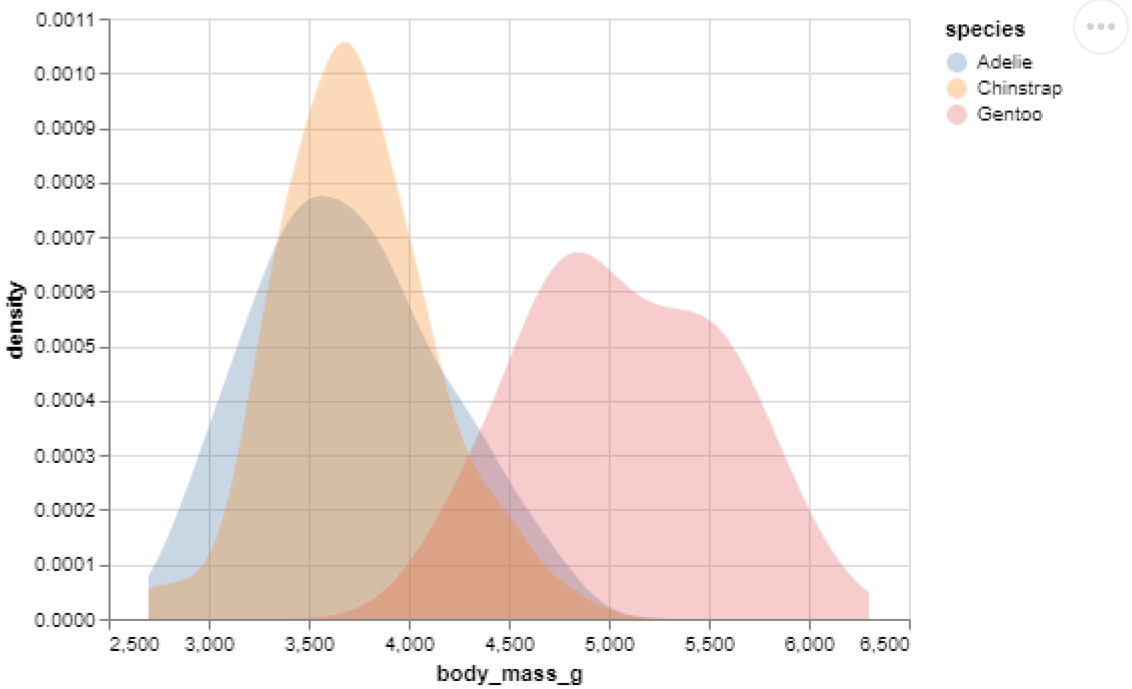
```
alt.Chart(penguins).transform_density(
    'body_mass_g',
        groupby=['species'],
        as_=['body_mass_g', 'density']
    ).mark_line().encode(
        alt.X('body_mass_g:Q', scale=alt.Scale(zero=True)),
        alt.Y('density:Q', stack=None),
        alt.Color('species:N')
    ).properties(width=400,height=300)
```



numeric & categorical: `transform_density()` filled in

`opacity=0.3` makes no difference in content; maybe a bit more elegant

```
alt.Chart(penguins).transform_density(
    'body_mass_g',
        groupby=['species'],  # Group by species for different density curves
        as_=['body_mass_g', 'density']
    ).mark_area(opacity=0.3).encode(
        alt.X('body_mass_g:Q'),
        alt.Y('density:Q', stack=None),
        alt.Color('species:N')
    ).properties(width=400,height=300)
```



Two categorical variables

Question: How is cut related to color? 2 categorical vars

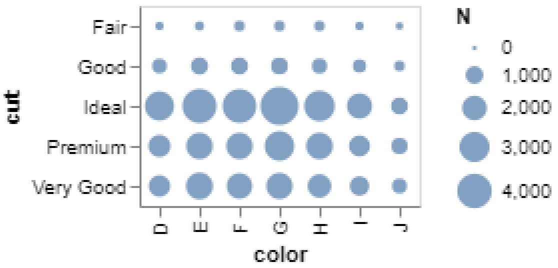
```
diamonds_grouped = diamonds.groupby(['color','cut']).size().reset_index().rename(columns={0:'N'})
diamonds_grouped
```

	color	cut	N
0	D	Fair	163
1	D	Good	662
2	D	Very Good	1513
3	D	Premium	1603
4	D	Ideal	2834
5	E	Fair	224
6	E	Good	933
7	E	Very Good	2400
8	E	Premium	2337
9	E	Ideal	3903
10	F	Fair	312
11	F	Good	909
12	F	Very Good	2164
13	F	Premium	2331

	color	cut	N
14	F	Ideal	3826
15	G	Fair	314
16	G	Good	871
17	G	Very Good	2299
18	G	Premium	2924
19	G	Ideal	4884
20	H	Fair	303
21	H	Good	702
22	H	Very Good	1824
23	H	Premium	2360
24	H	Ideal	3115
25	I	Fair	175
26	I	Good	522
27	I	Very Good	1204
28	I	Premium	1428
29	I	Ideal	2093
30	J	Fair	119
31	J	Good	307
32	J	Very Good	678
33	J	Premium	808
34	J	Ideal	896

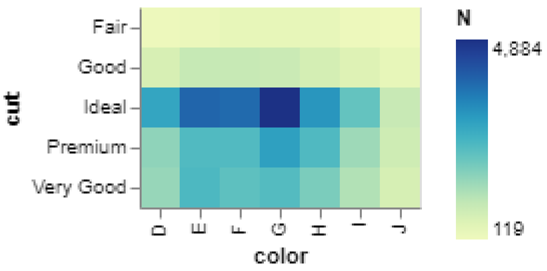
Question: How is cut related to color? 2 categorical vars

```
alt.Chart(diamonds_grouped).mark_circle().encode(
    x = 'color:N',
    y = 'cut:N',
    size='N:Q')
```



Question: How is cut related to color? 2 categorical vars

```
alt.Chart(diamonds_grouped).mark_rect().encode(
    x = 'color:N',
    y = 'cut:N',
    color='N:Q')
```



Discussion question: what diamond types are most common?

Two continuous variables

Two continuous variables: roadmap

- `movies` ratings from Rotten Tomatoes and IMDB
- `diamonds`: `carat` vs `price`

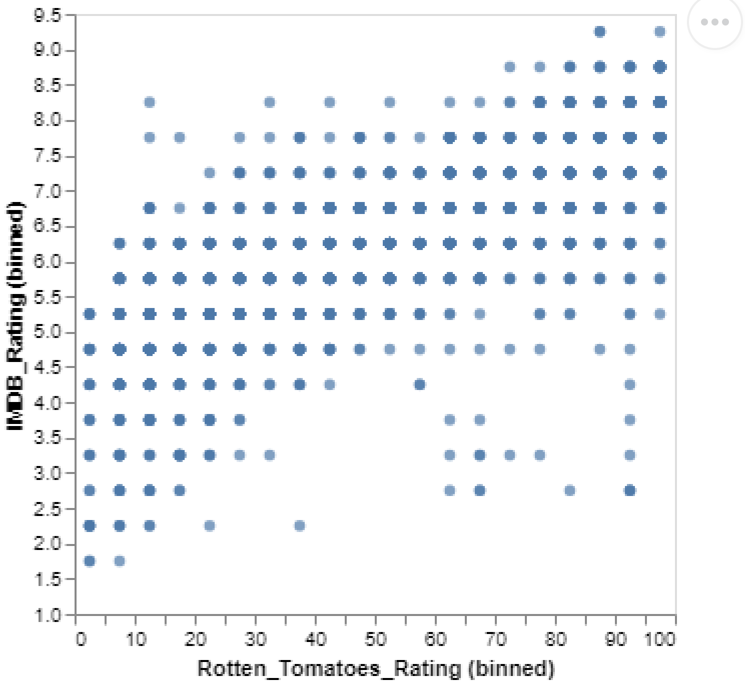
movies dataset

```
movies_url = 'https://cdn.jsdelivr.net/npm/vega-datasets@1/data/movies.json'
```

```
movies = pd.read_json(movies_url)
```

Covariation: a first binned scatter plot

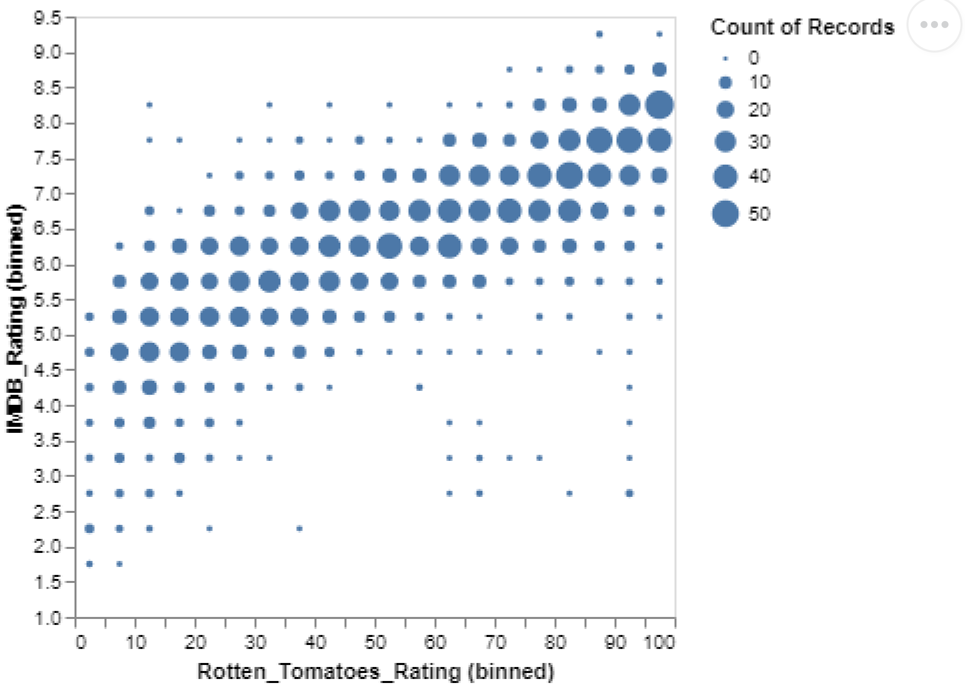
```
alt.Chart(movies_url).mark_circle().encode(
    alt.X('Rotten_Tomatoes_Rating:Q', bin=alt.BinParams(maxbins=20)),
    alt.Y('IMDB_Rating:Q', bin=alt.BinParams(maxbins=20)),
)
```



Suffers from overplotting!

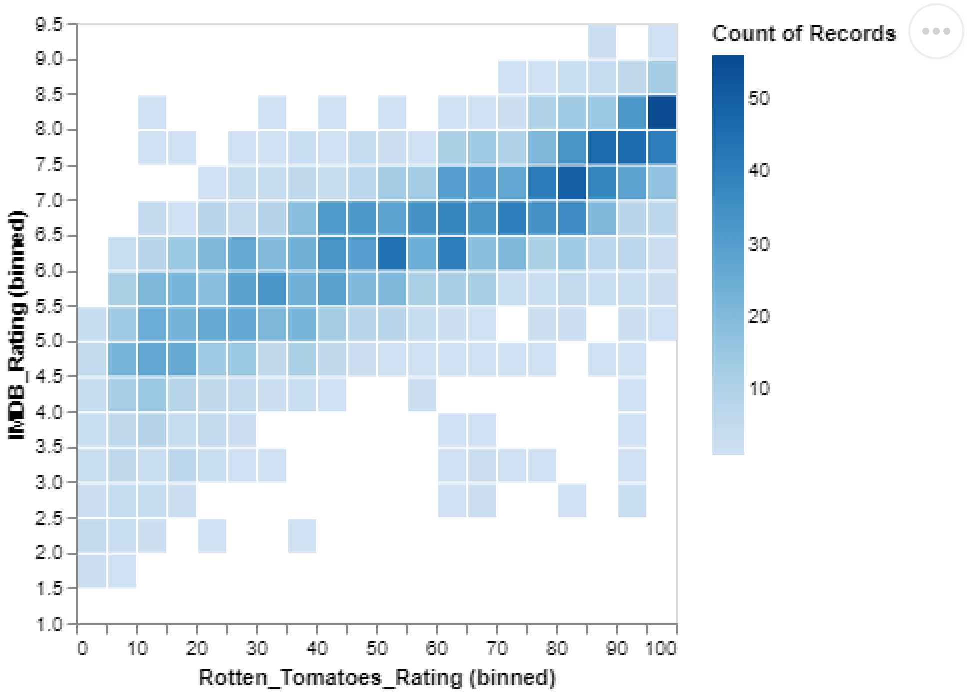
use `alt.Size('count()')` to address overplotting

```
xy_size = alt.Chart(movies_url).mark_circle().encode(
    alt.X('Rotten_Tomatoes_Rating:Q', bin=alt.BinParams(maxbins=20)),
    alt.Y('IMDB_Rating:Q', bin=alt.BinParams(maxbins=20)),
    alt.Size('count()')
)
xy_size
```



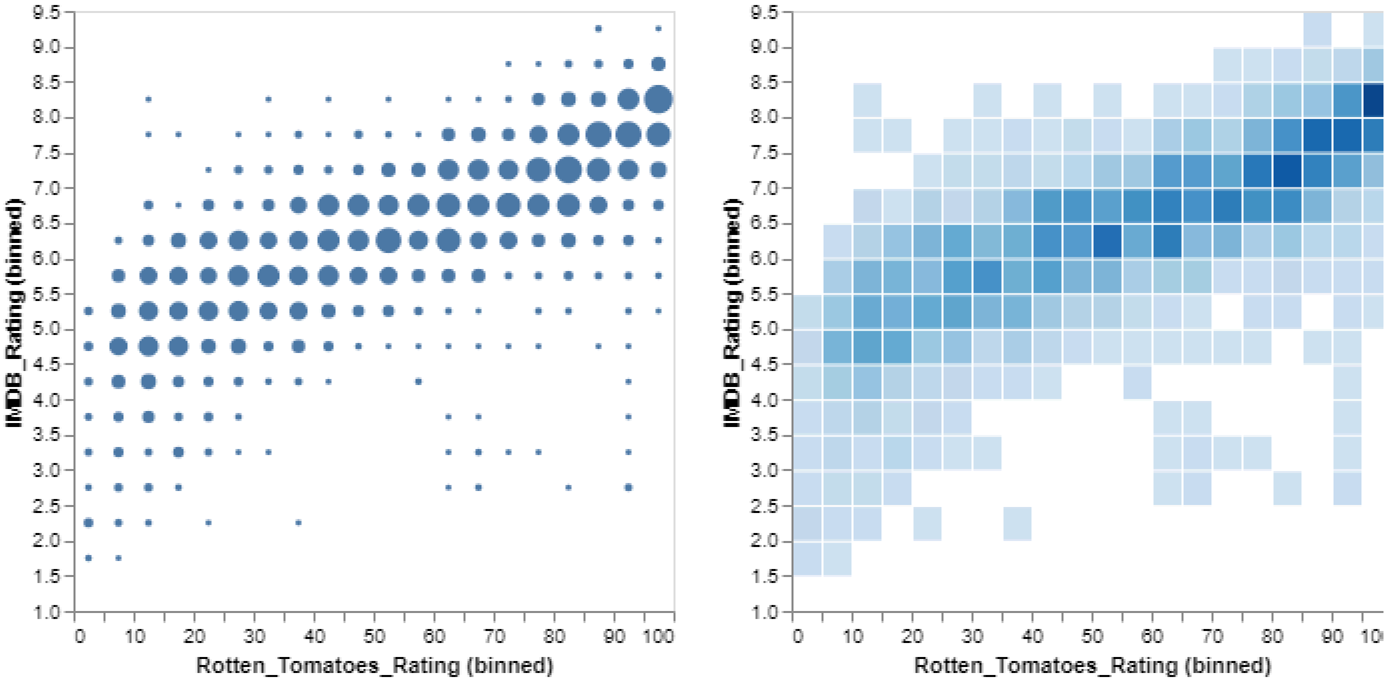
use `alt.Color('count()')` to address overplotting

```
xy_color = alt.Chart(movies_url).mark_bar().encode(
    alt.X('Rotten_Tomatoes_Rating:Q', bin=alt.BinParams(maxbins=20)),
    alt.Y('IMDB_Rating:Q', bin=alt.BinParams(maxbins=20)),
    alt.Color('count()')
)
xy_color
```



Discussion question

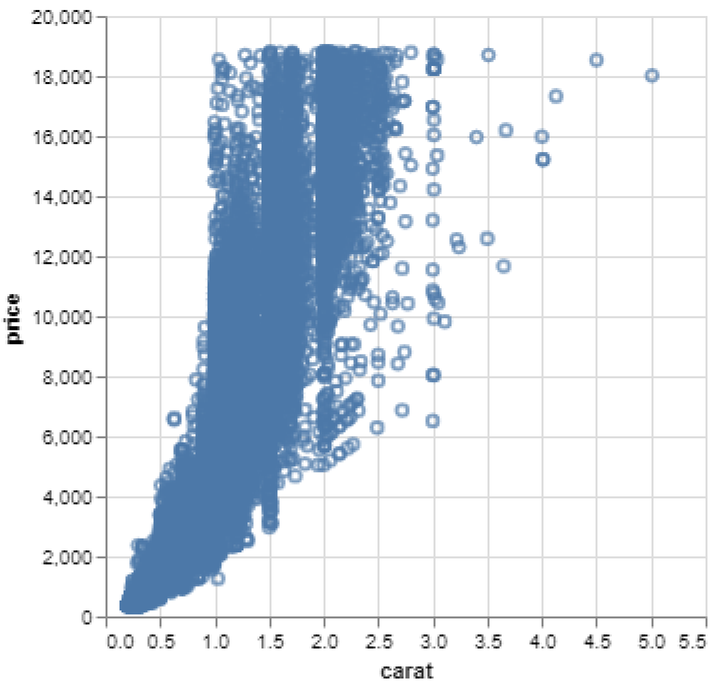
```
xy_size | xy_color
```



Compare the *size* and *color*-based 2D histograms above. Which encoding do you think should be preferred? Why?

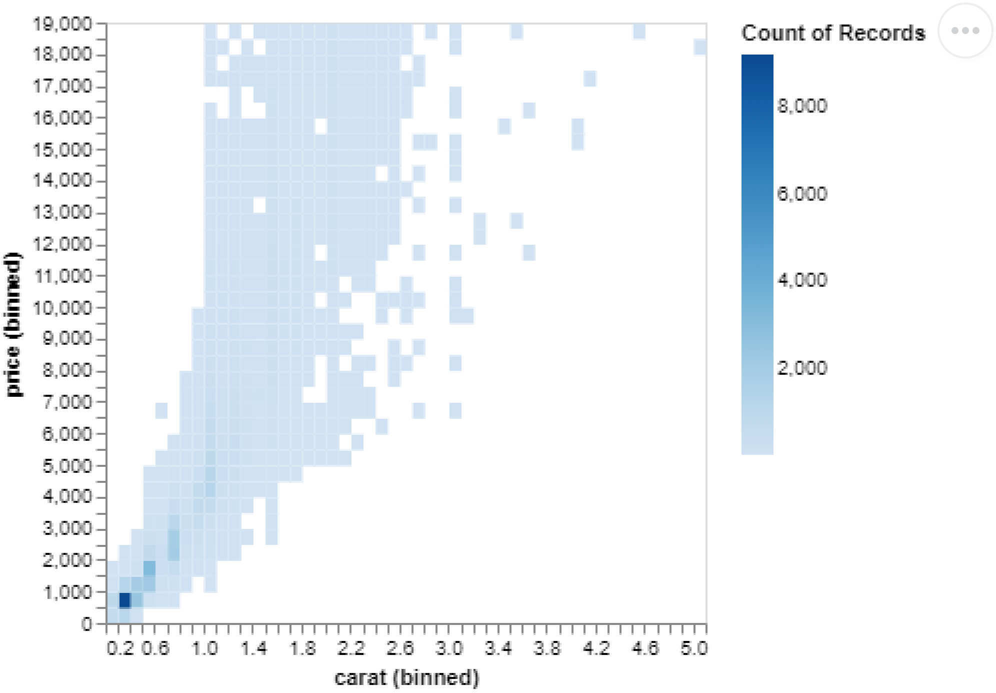
Question: How is carat related to price? 2 continuous vars

```
alt.Chart(diamonds).mark_point().encode(
    x = 'carat:Q',
    y = 'price:Q'
)
```



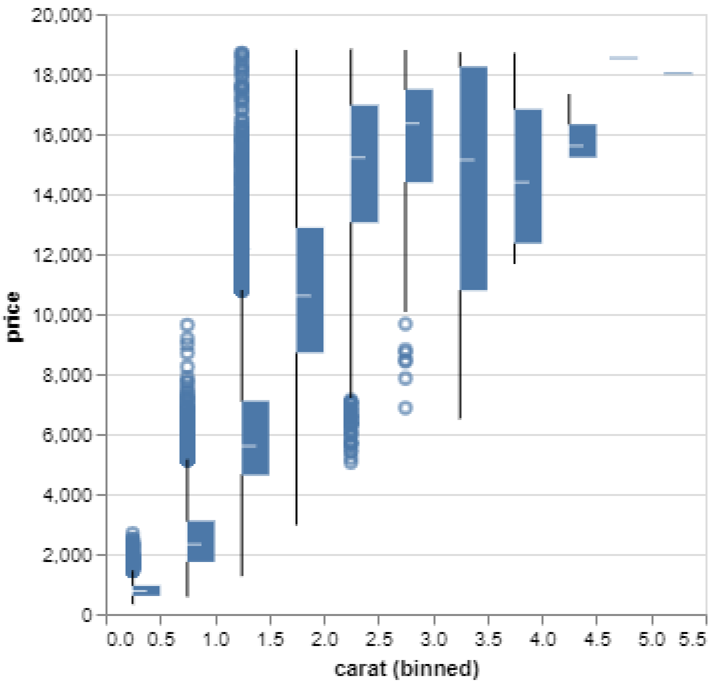
Question: How is carat related to price? 2 continuous vars

```
alt.Chart(diamonds).mark_rect().encode(
    alt.X('carat:Q', bin=alt.Bin(maxbins=70)),
    alt.Y('price:Q', bin=alt.Bin(maxbins=70)),
    alt.Color('count()', scale=alt.Scale(scheme='blues')))
```



Question: How is carat related to price? 2 continuous vars

```
alt.Chart(diamonds).mark_boxplot().encode(
    alt.X('carat:Q', bin=alt.Bin(maxbins=10)),
    alt.Y('price:Q'))
```



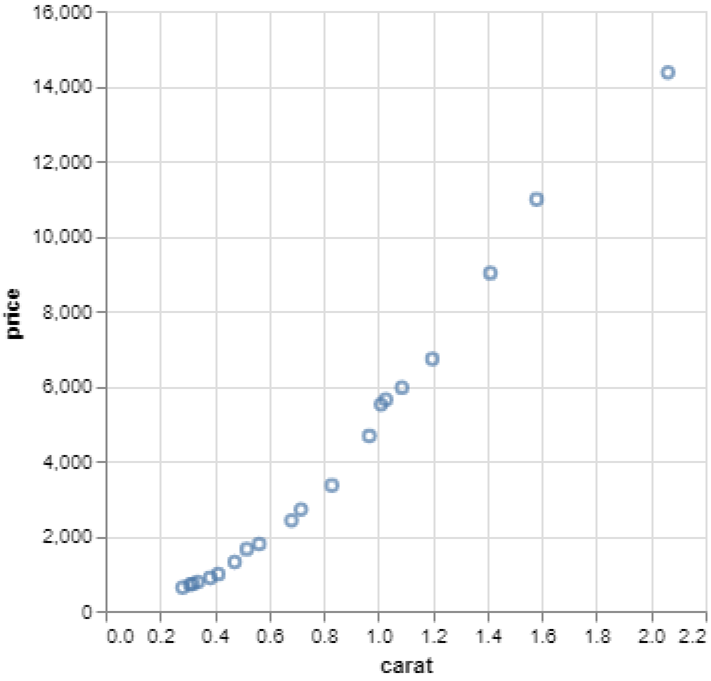
Question: How is carat related to price? 2 continuous vars

```
df = diamonds
df['carat_bin'] = pd.qcut(df['carat'], q=20, labels=(np.arange(1, 21, 1)))

df = df.groupby('carat_bin').agg(
    carat = ('carat', 'mean'),
    price = ('price', 'mean')).reset_index()

alt.Chart(df).mark_point().encode(
   x = 'carat:Q',
   y = 'price:Q'
)
```

```
/var/folders/9k/556bcdln0hsc_tw0rlx916_00000gn/T/ipykernel_37573/2590590619.py:4: FutureWarning:
The default of observed=False is deprecated and will be changed to True in a future version of
pandas. Pass observed=False to retain current behavior or observed=True to adopt the future
default and silence this warning.
```



- What it does:
    1. Computes bins using quantiles of x
    2. Computes means of y within each bin

- Called `binscatter` in stata and `binsreg` in R. Doesn't exist yet for Altair, but easy to code up yourself

Discussion question – "How is carat related to price?"

Review the `mark_rect()`, `mark_boxplot()`, and `binscatter` plots

- headline? (aka the main message)
- sub-messages? (other information one can learn beyond the main message)

Summary: Exploring covariation

Scenario	Functions
Categorical and continuous variable	`mark_boxplot()`
	`transform_density()`
Two categorical variables	`size`
	`color`
Two continuous variables	`alt.Size('count()')`

Scenario	Functions
	`alt.Color('count()')`
	`mark_boxplot()`
	binscatter

Do-pair-share

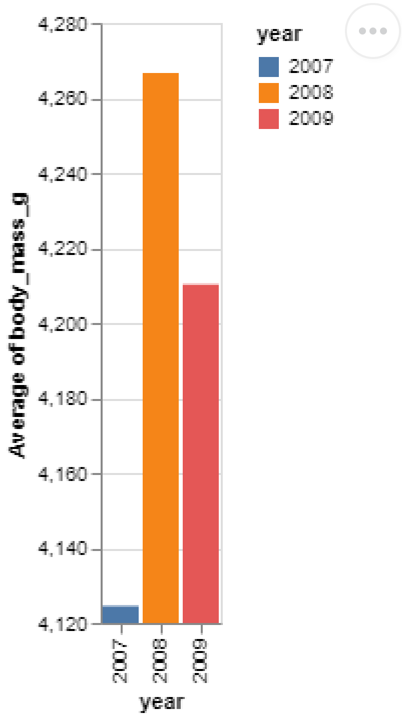
We are now going to transition from making plots to teach **ourselves** to making plots for an audience.

Are penguins getting heavier (`body_mass_g`) over time?

Bonus: what is the headline of your plot and what are the sub-messages?

Do-pair-share solution I

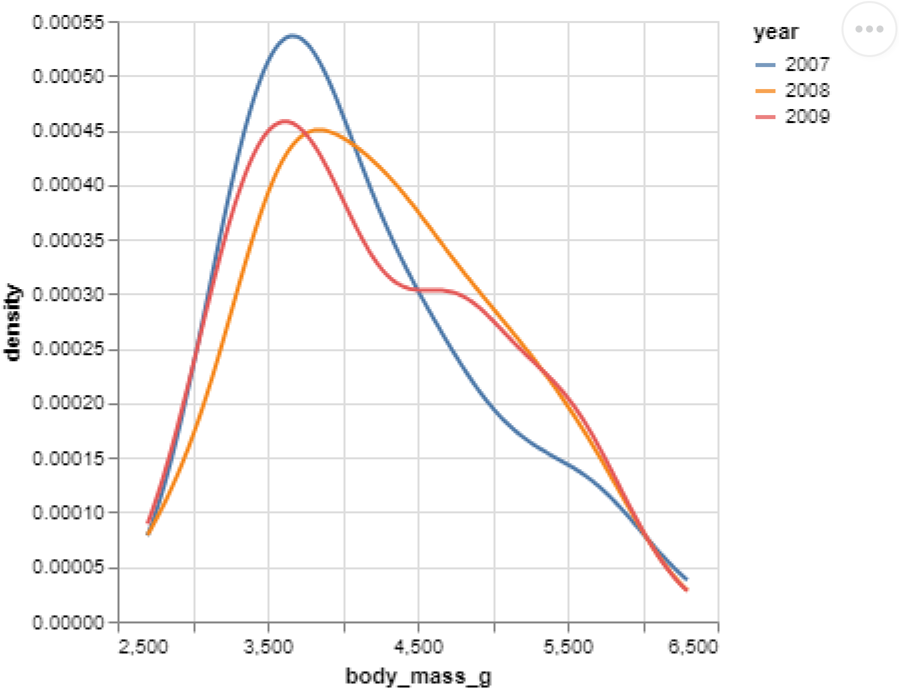
```
alt.Chart(penguins).mark_bar().encode(
  alt.Y('average(body_mass_g):Q',  scale=alt.Scale(zero=False)),
  alt.X('year:N'),
  alt.Color('year:N')
)
```



This does answers the question, albeit in the most simple/boring way possible.

Do-pair-share solution II

```
alt.Chart(penguins).transform_density(
   'body_mass_g',
    groupby=['year'],
    as_= ['body_mass_g', 'density']
).mark_line().encode(
    x = 'body_mass_g:Q',
    y = 'density:Q',
    color='year:N'
)
```



- Headline: 2007 is lightest, 2008 is heaviest

- Sub-messages

  1. Similar shares of penguins above 5,000 grams in 2008 and 2009
  2. Average weight is higher in 2008 because 2009 has more lightweight penguins

Meta comment: iterating on plot design

“Make dozens of plots” – Quoctrung Bui, former 30535 guest lecturer and former Harris data viz instructor

What does he mean?

- The first plot you make will never be the one you should show
- As a rule of thumb, you should try out at least three different plotting concepts (`mark` s)
- Within each concept, you will need to try out several different encodings