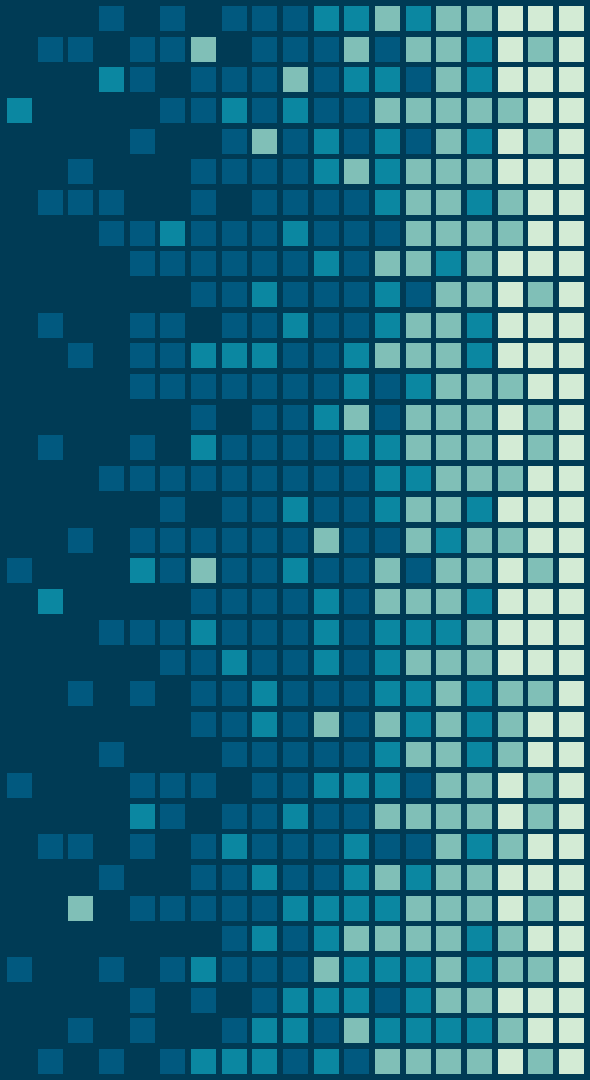
Going Beyond Matplotlib and Seaborn:

A survey of Python data visualization tools

Stephanie Kirmer
@data_stephanie
https://github.com/skirmer/new-py-dataviz



Our Contenders

The Old Standards

The New Generation

Criteria

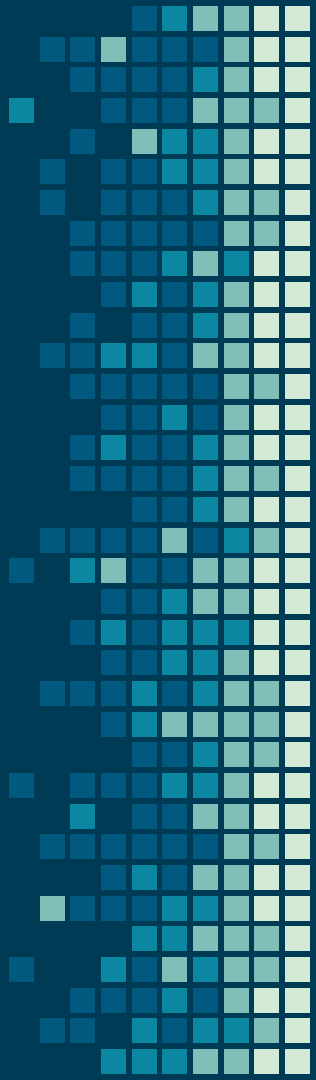
Easy learning curve

Sensible, consistent grammar

Flexibility

Beautiful, readable output

Nice to have: Interactivity



Perspectives

New user (not power user)

Busy person (not unlimited time to waste)

Care how it looks (Want to be visually compelling)

The Tests

- Histogram
- Scatterplot
- Faceted Scatterplot
- Grouped Bar
- Time Series Line
- BONUS: 3D Scatterplot

Dataset: Spotify Tracks

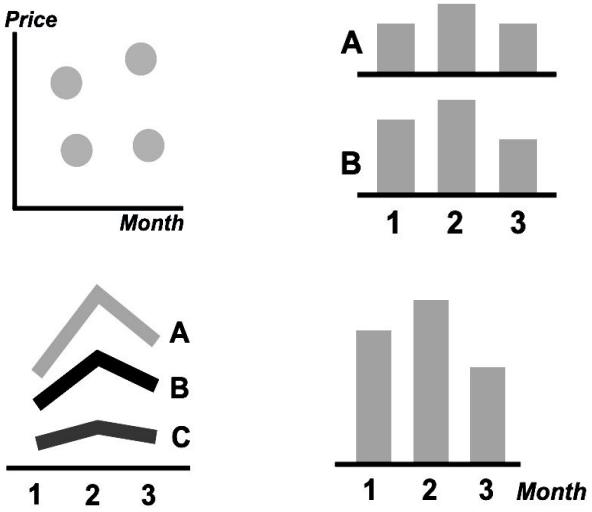
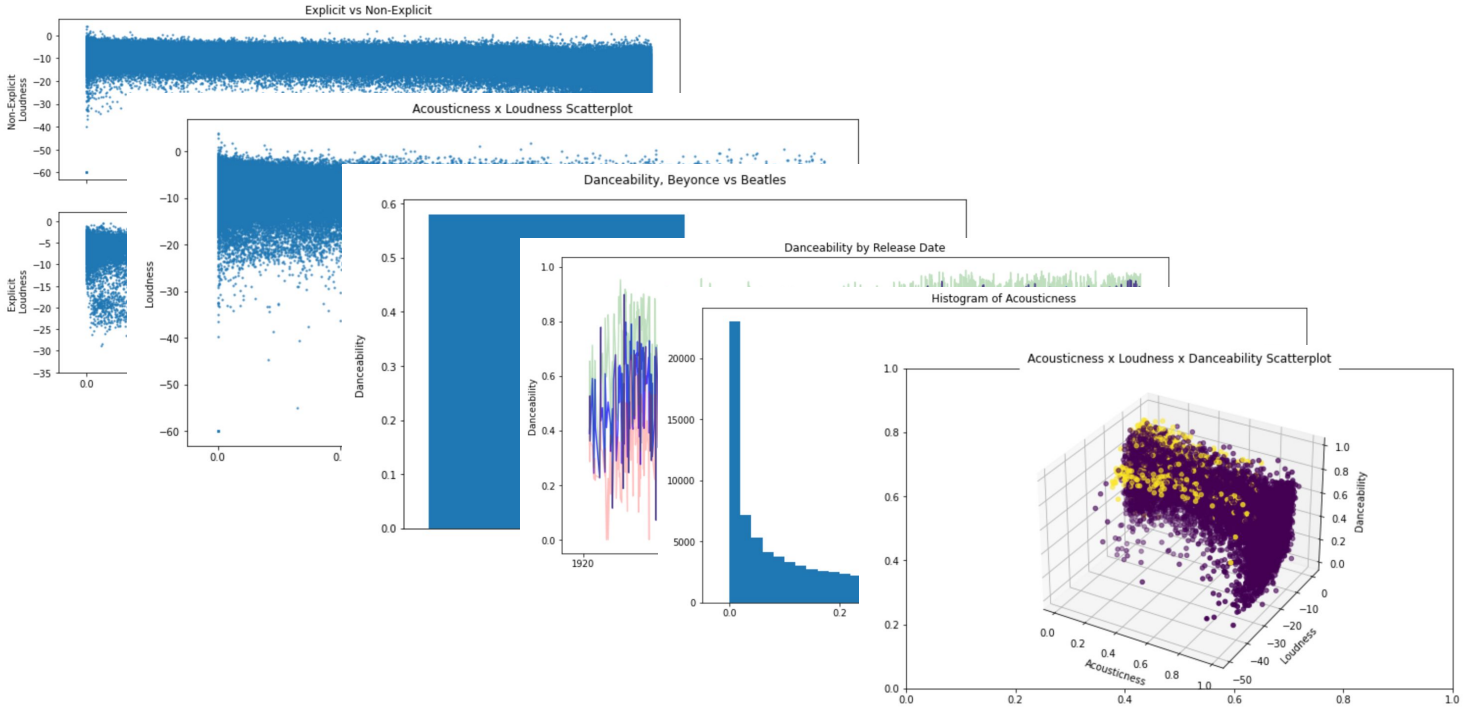


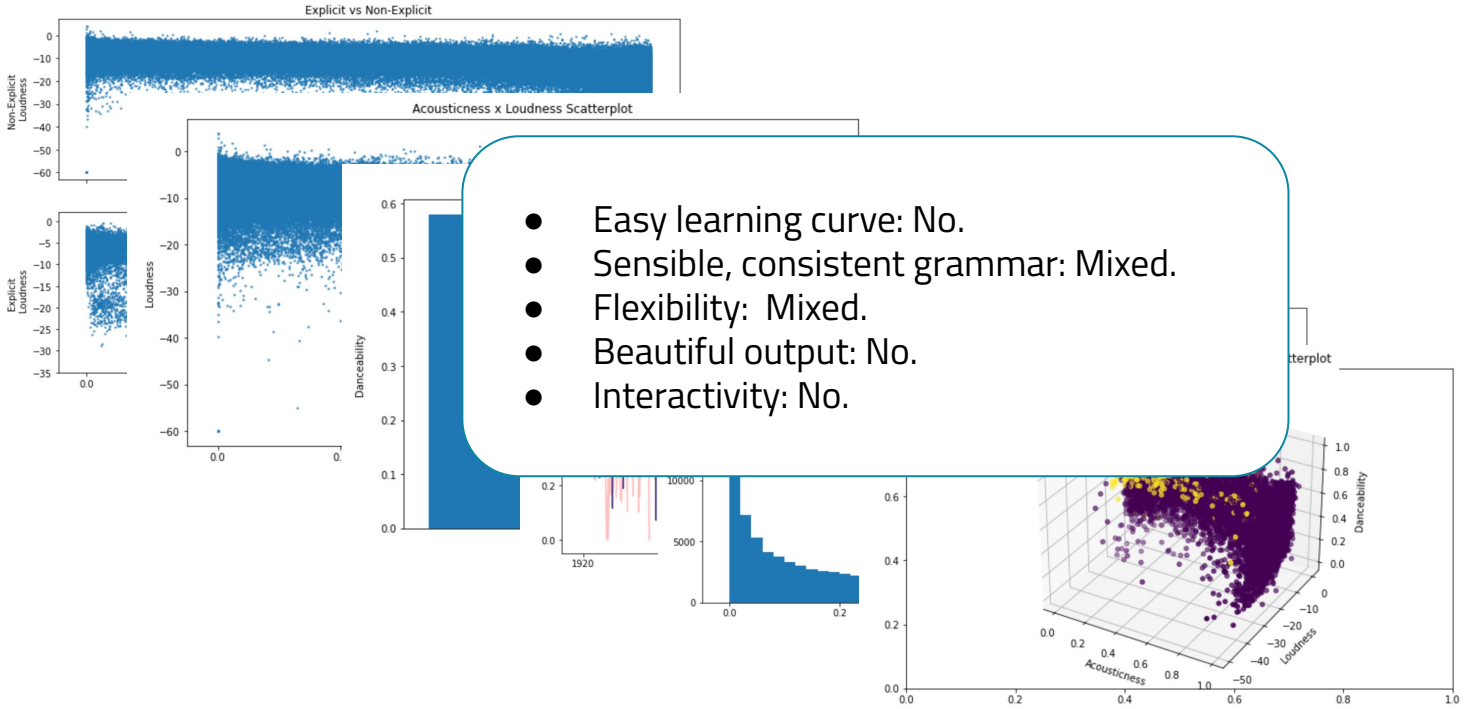
Image credits: Steve Franconeri, 2019

matplotlib

2003, John D. Hunter
https://matplotlib.org/



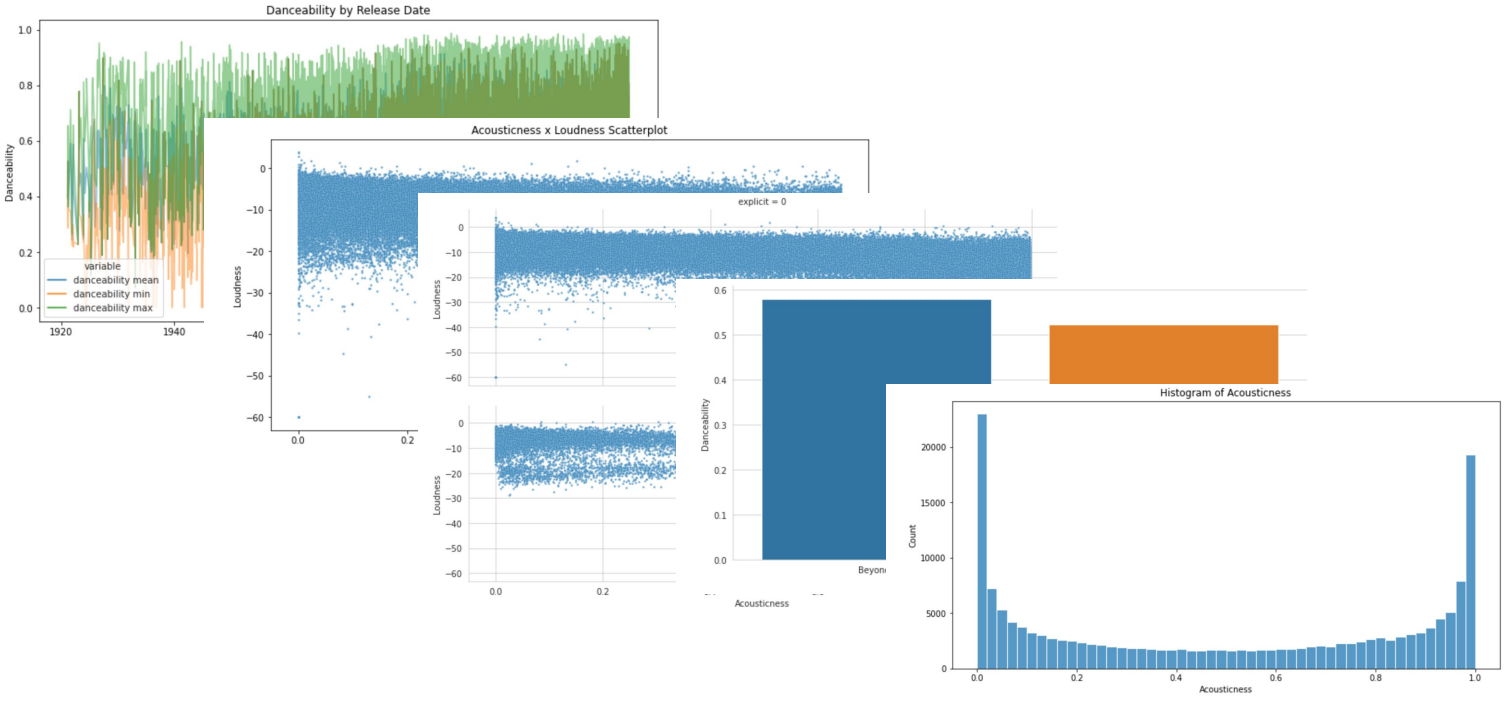
matplotlib



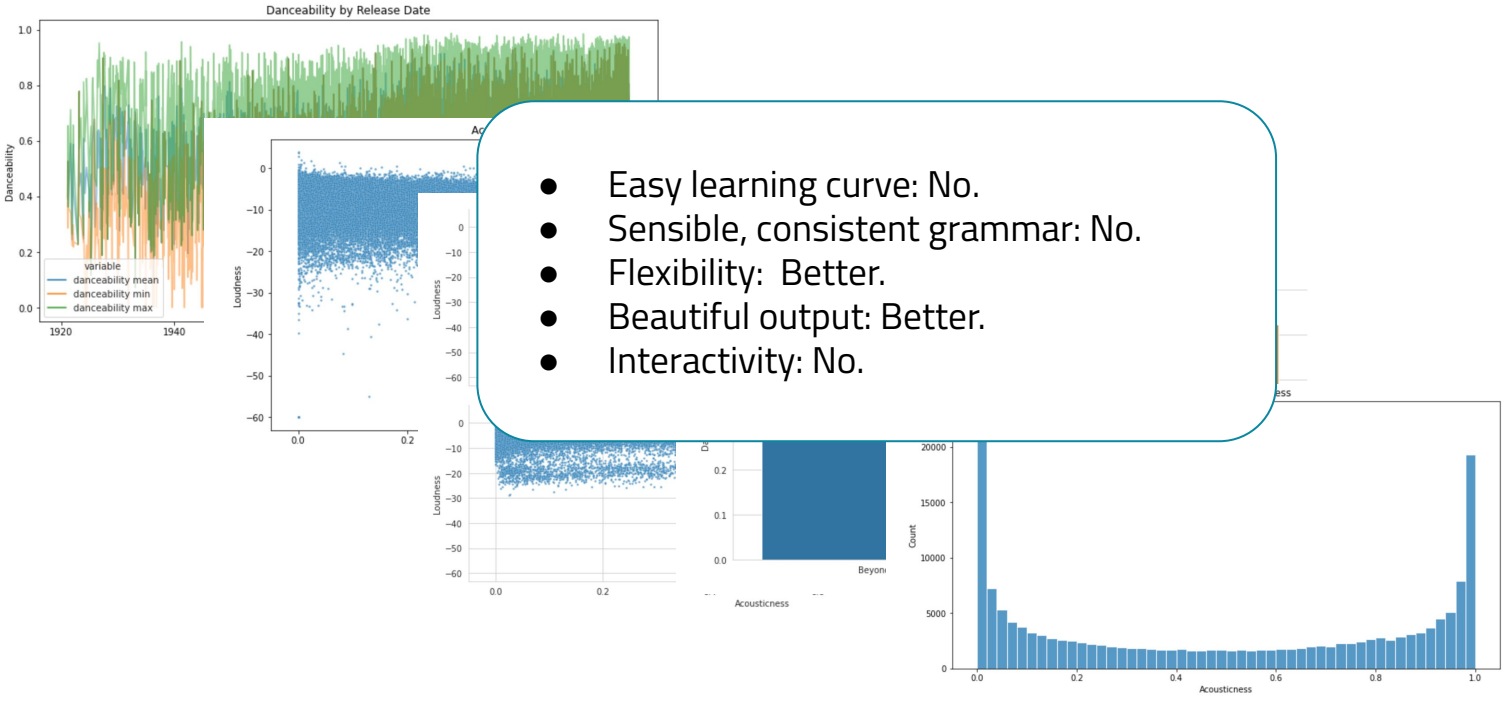
- Easy learning curve: No.
- Sensible, consistent grammar: Mixed.
- Flexibility: Mixed.
- Beautiful output: No.
- Interactivity: No.

seaborn

2012, Michael Waskom
https://seaborn.pydata.org/



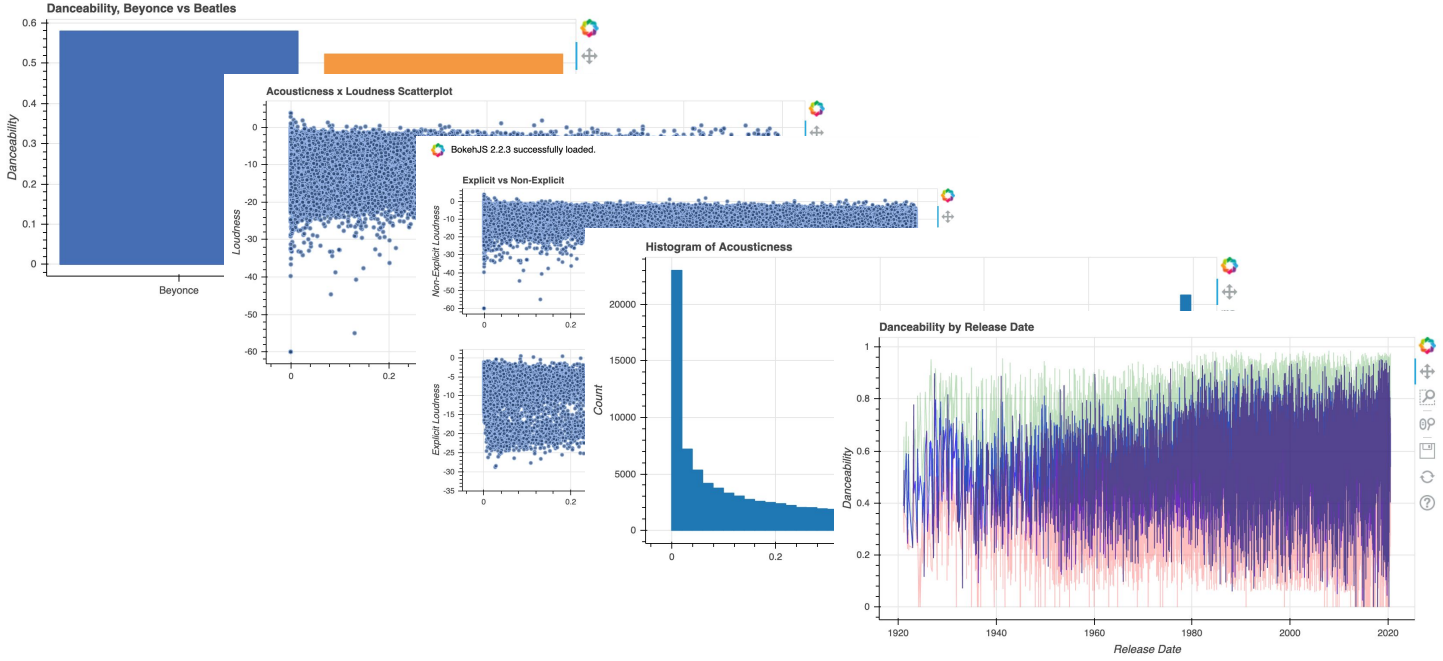
seaborn



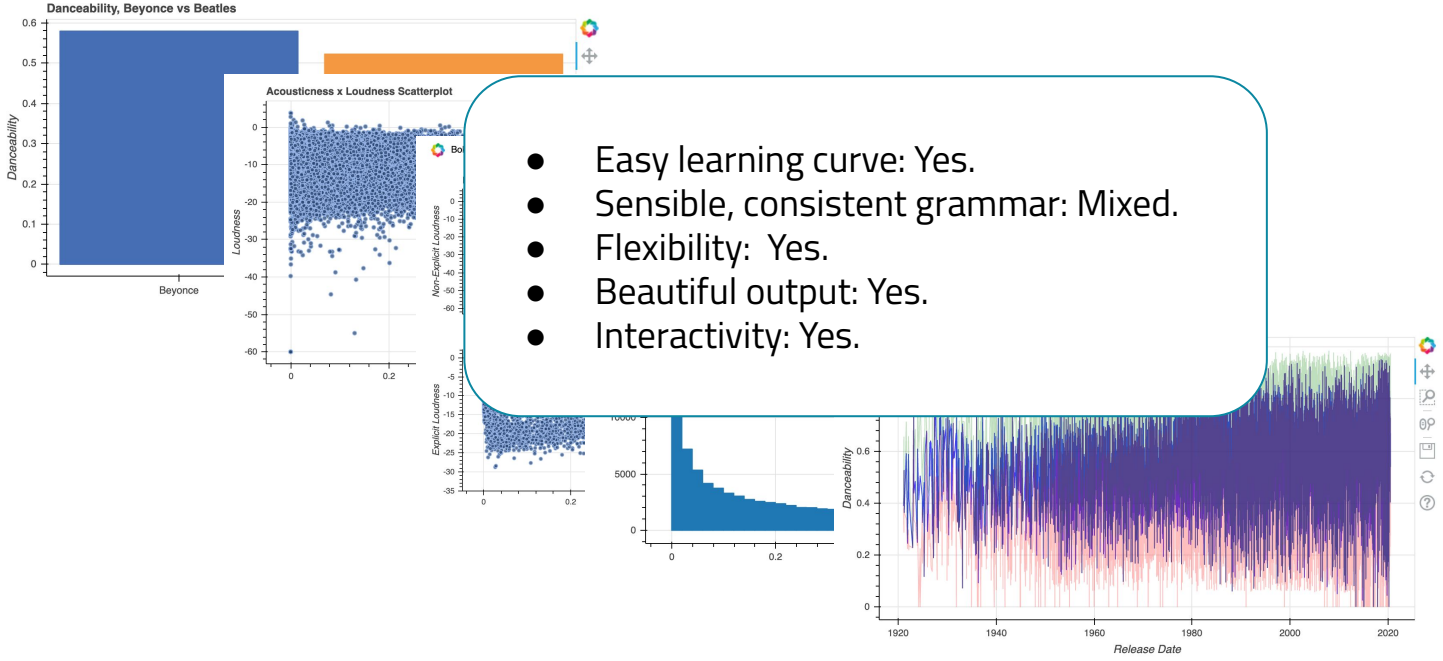
- Easy learning curve: No.
- Sensible, consistent grammar: No.
- Flexibility: Better.
- Beautiful output: Better.
- Interactivity: No.

bokeh

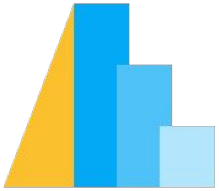
2012, NumFOCUS
http://bokeh.pydata.org/



bokeh

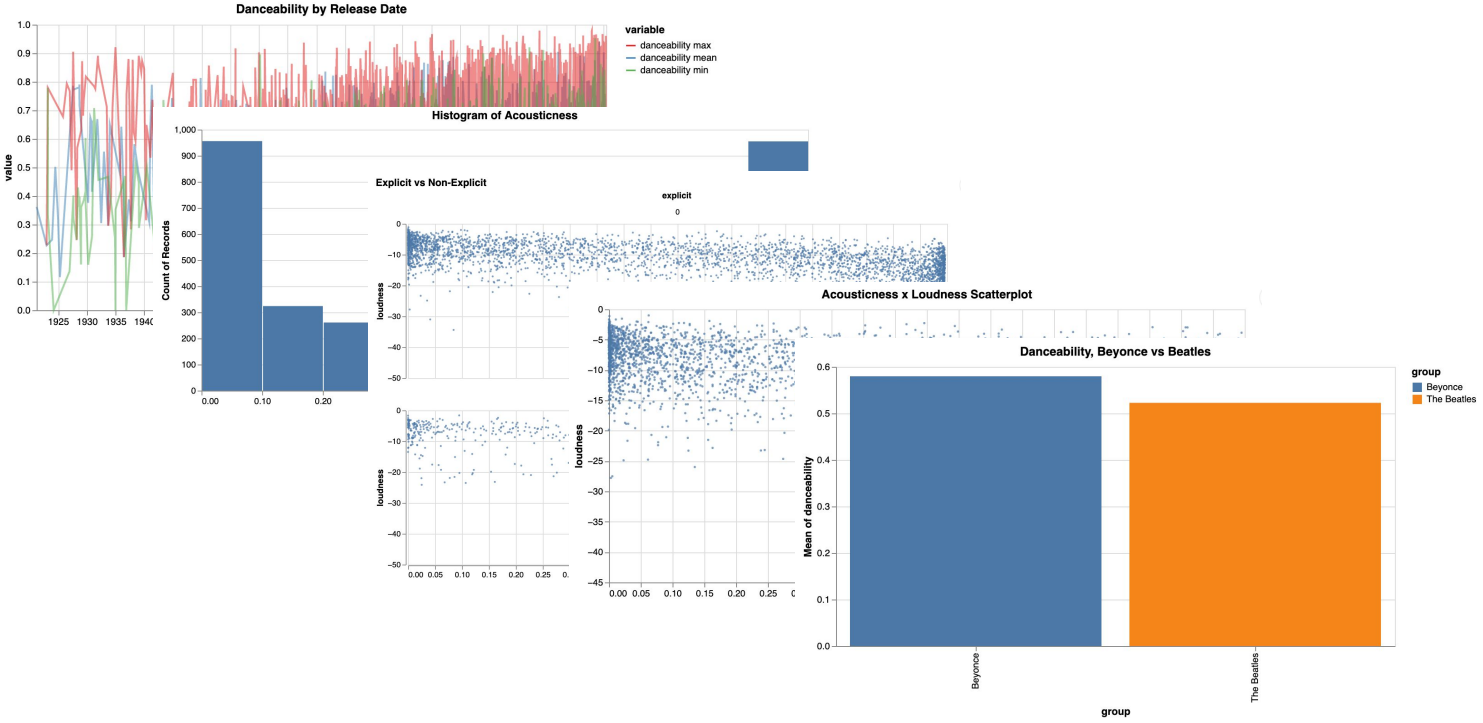


- Easy learning curve: Yes.
- Sensible, consistent grammar: Mixed.
- Flexibility: Yes.
- Beautiful output: Yes.
- Interactivity: Yes.

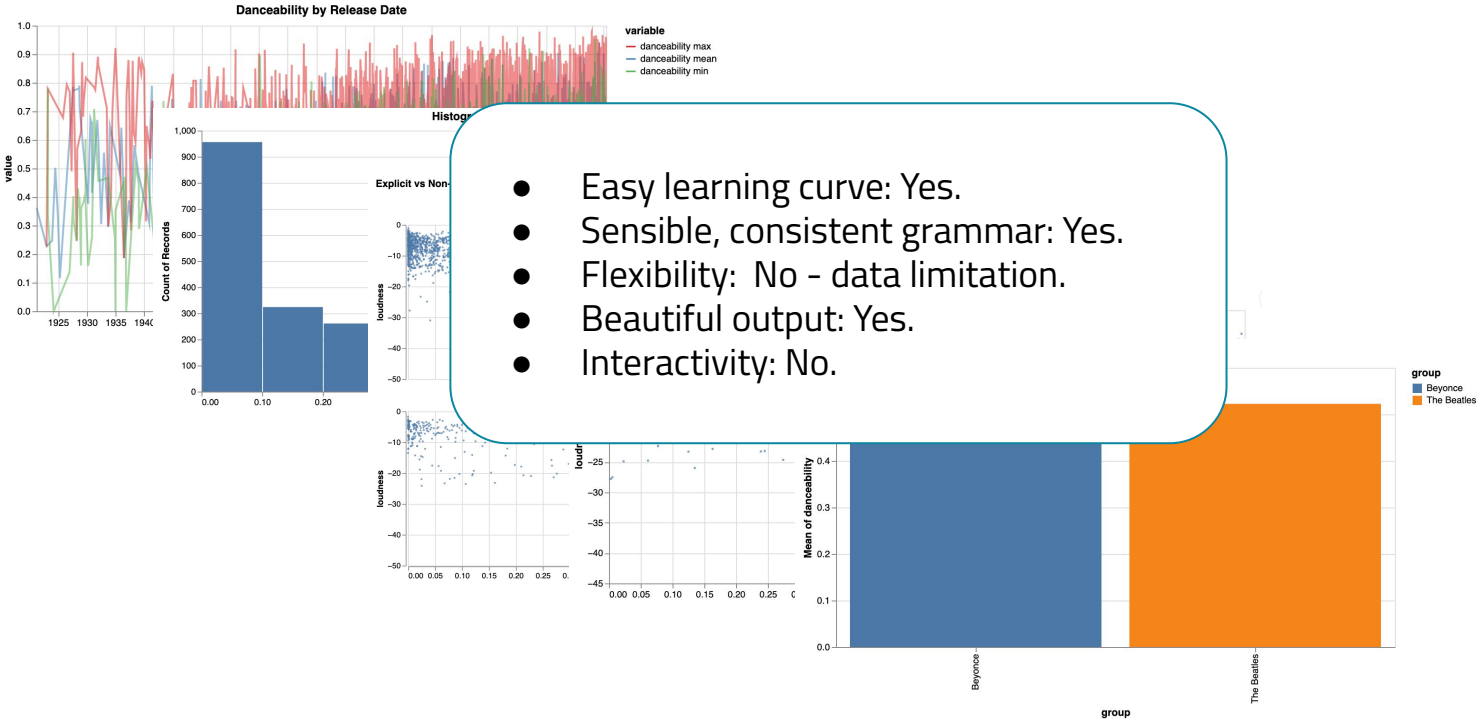


Altair

2016, Jake VanderPlas et al.
https://altair-viz.github.io/

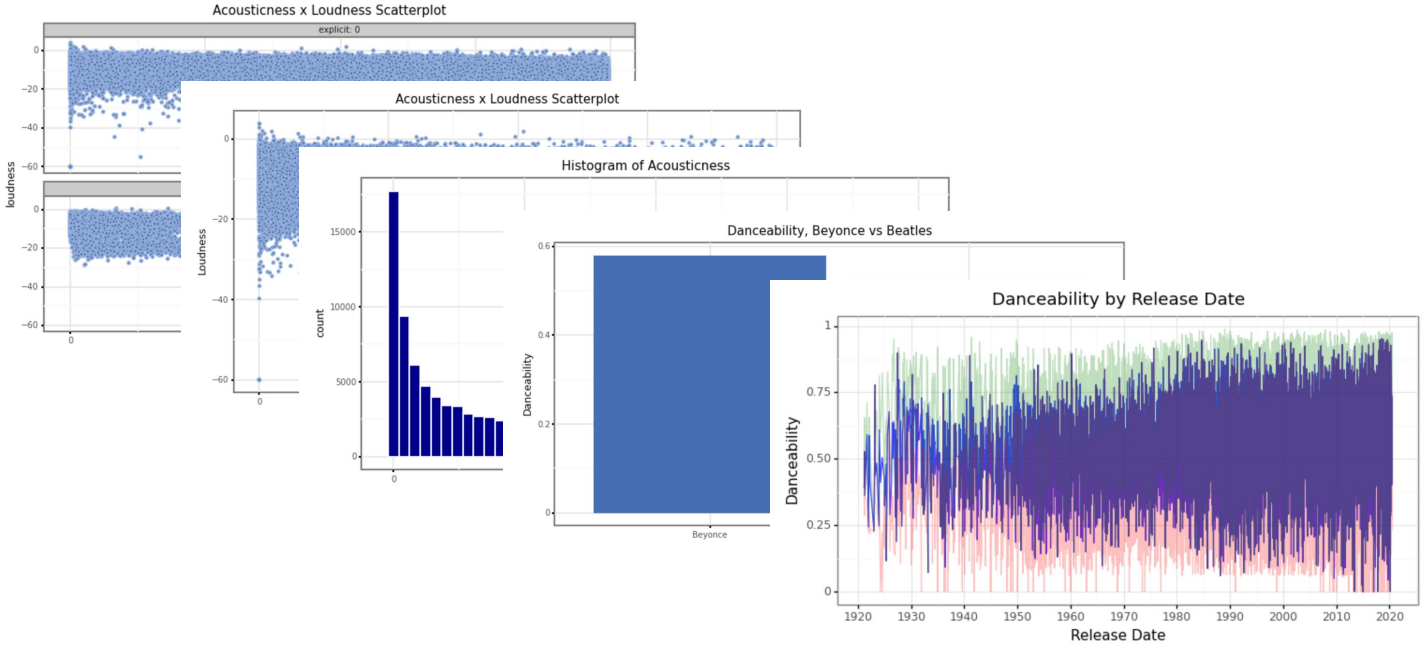


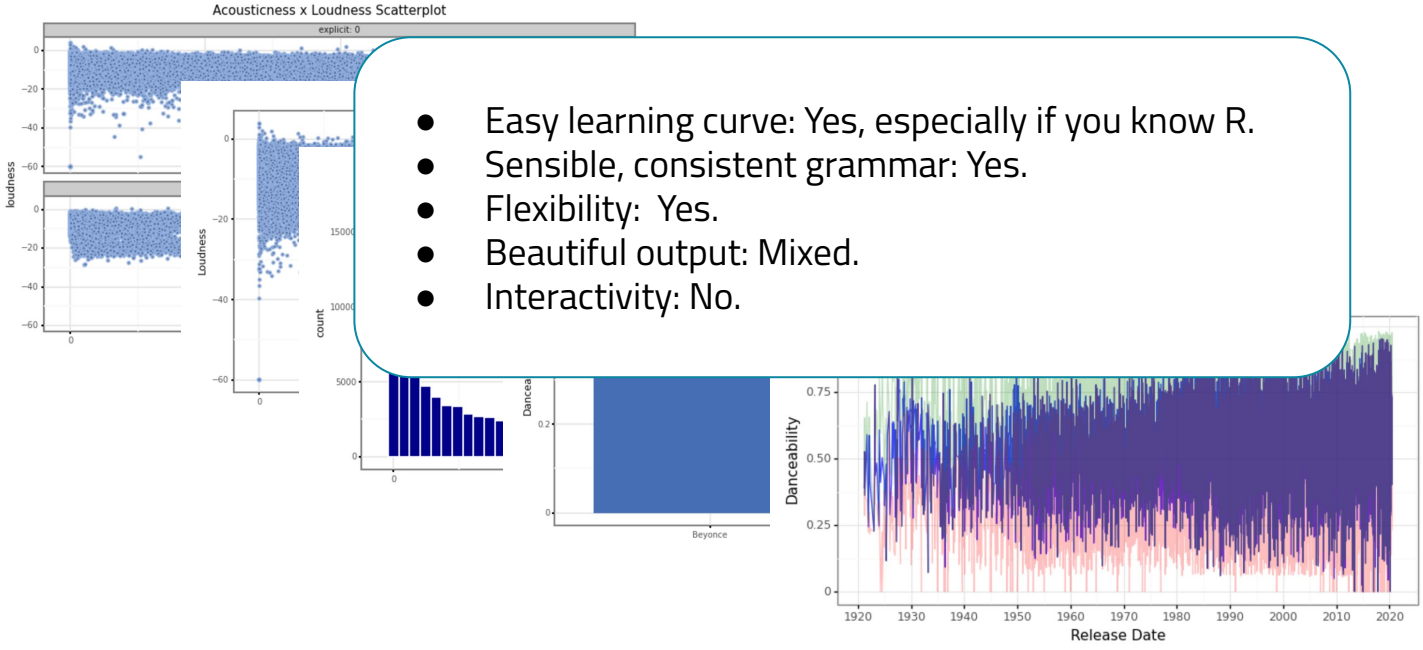
Altair



- Easy learning curve: Yes.
- Sensible, consistent grammar: Yes.
- Flexibility: No - data limitation.
- Beautiful output: Yes.
- Interactivity: No.

2017, Hassan Kibirige
https://plotnine.readthedocs.io/

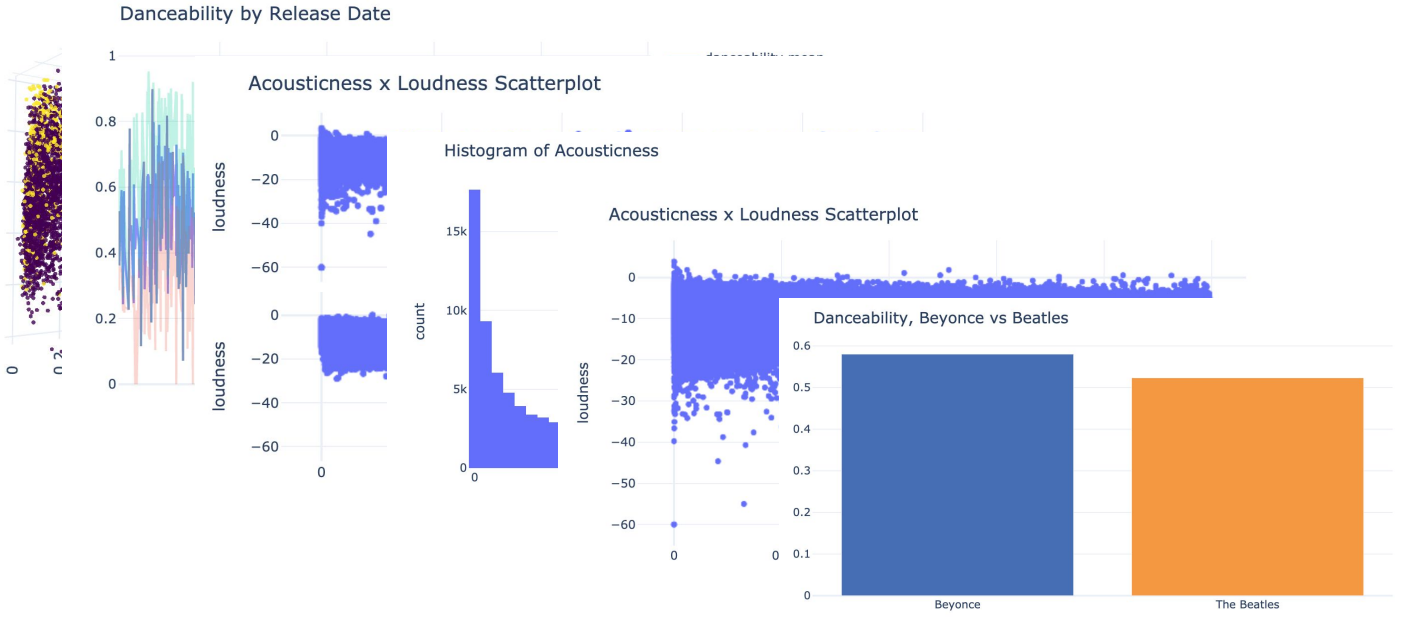




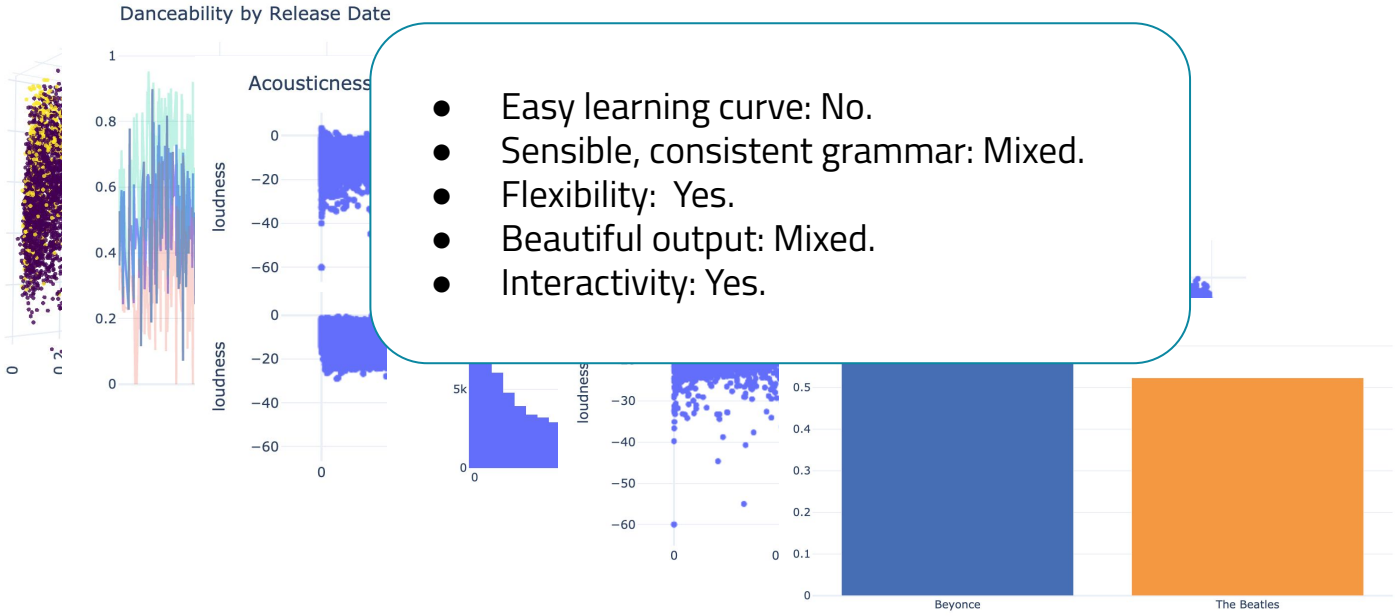
- Easy learning curve: Yes, especially if you know R.
- Sensible, consistent grammar: Yes.
- Flexibility: Yes.
- Beautiful output: Mixed.
- Interactivity: No.

2013, Plotly
https://plotly.com/python/

Acousticness x Loudness x Danceability Scatterplot



Acousticness x Loudness x Danceability Scatterplot



- Easy learning curve: No.
- Sensible, consistent grammar: Mixed.
- Flexibility: Yes.
- Beautiful output: Mixed.
- Interactivity: Yes.

Choose Your Fighter

Sensible grammar
Plotnine or Altair*

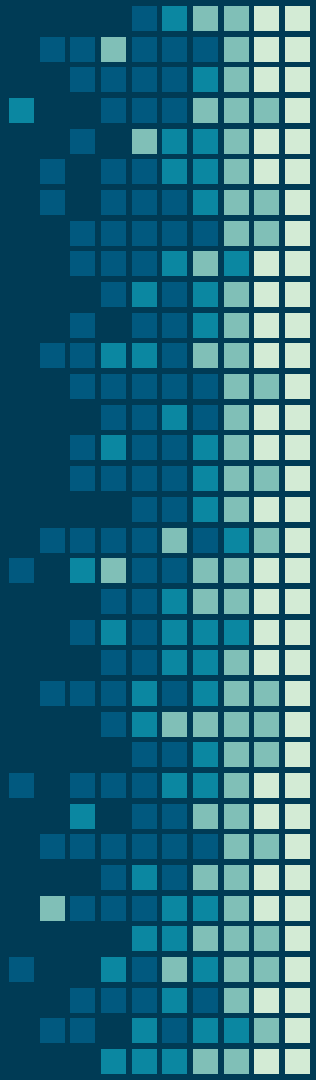
Easy learning curve
Plotnine or Altair*

Interactivity
Plotly or Bokeh

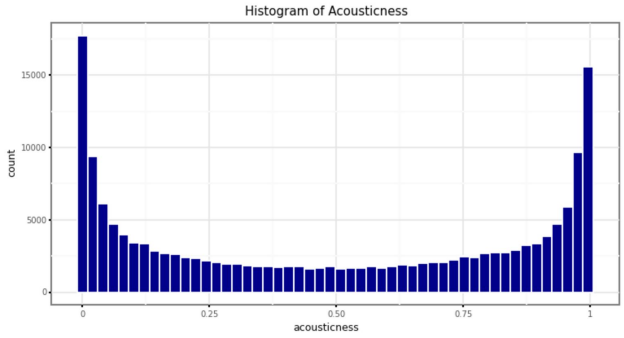
Beautiful images
Altair* or Bokeh

Flexibility
Plotnine or Bokeh

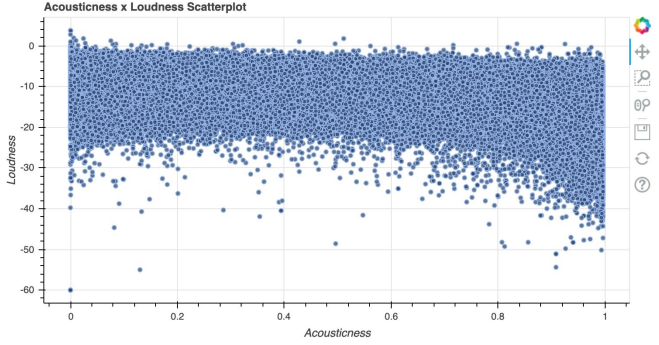
*provided you don't have big datasets.



Best-Of Awards

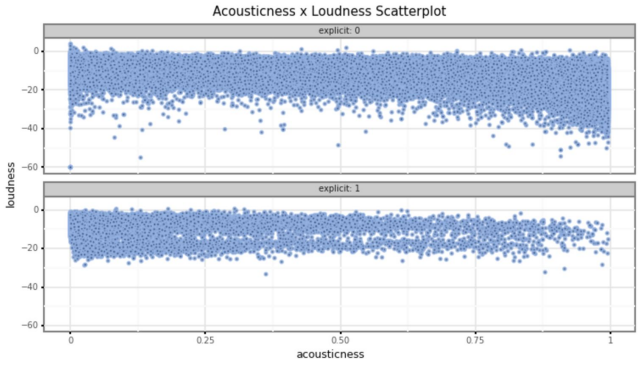


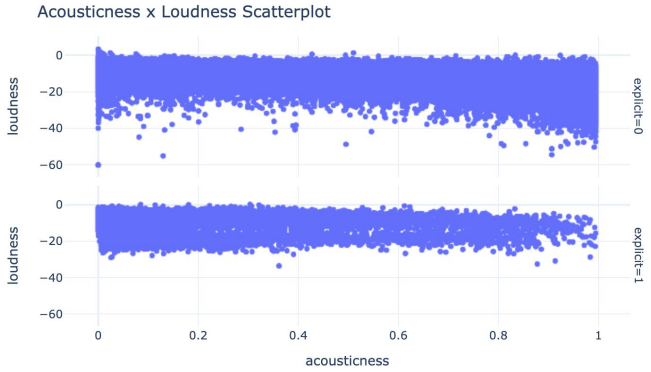
Histogram:
Plotnine



Plain Scatter:
Bokeh

Best-Of Awards

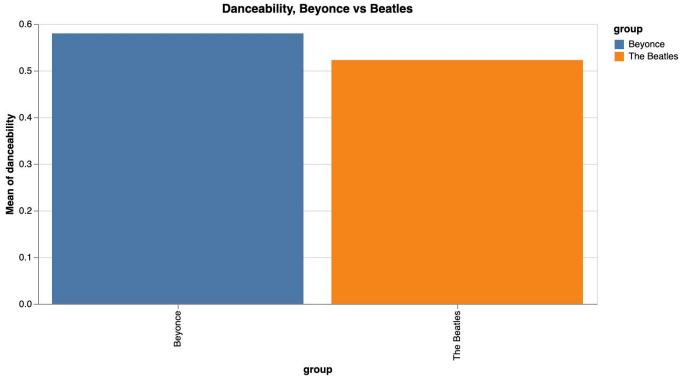


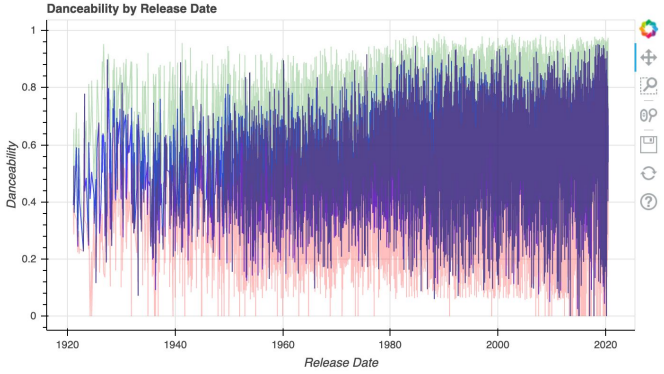


Faceted Scatter:

Plotnine/Plotly tie

Best-Of Awards





Grouped Bar:
Altair

Timeline:
Bokeh

Thank You!

There is not a perfect Python dataviz library.

You have lots of wonderful choices.

Choose what works for you!

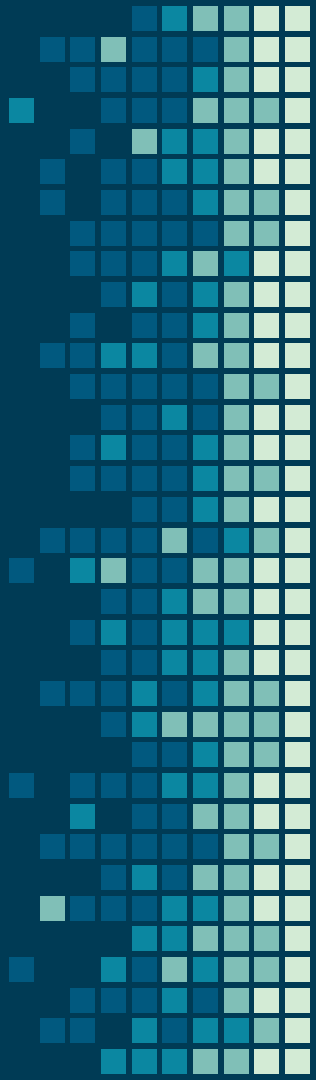
github.com/skirmer/new-py-dataviz

www.stephaniekirmer.com

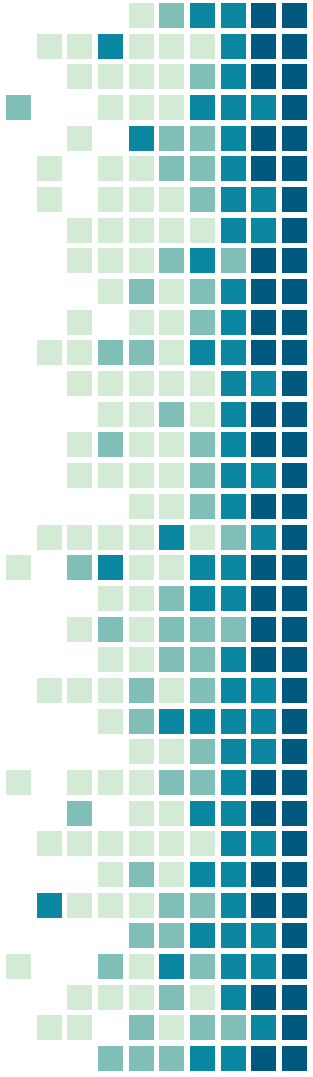
@data_stephanie

Saturn Cloud

saturncloud.io

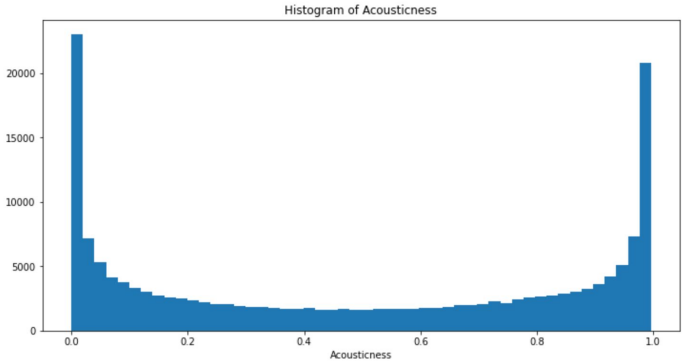


Appendix: Code Examples



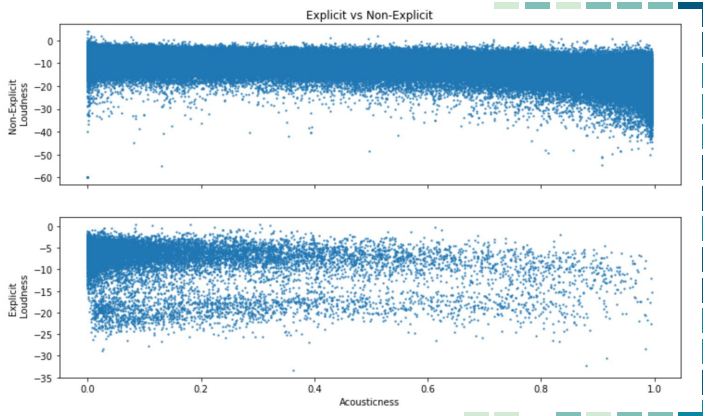
matplotlib

```
fig, ax = plt.subplots(figsize=(12, 6))
n, bins, patches = ax.hist(dataset.acousticness, 50)
ax.set_xlabel('Acousticness')
ax.set_title('Histogram of Acousticness')
ax.grid(False)
plt.show()
```



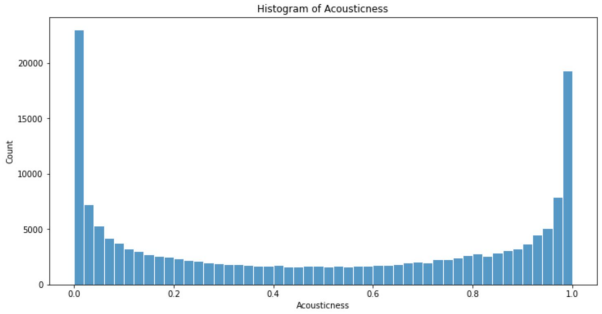
matplotlib

```
d0 = dataset[dataset['explicit'] == 0]
d1 = dataset[dataset['explicit'] == 1]

f, (ax1, ax2) = plt.subplots(2, sharex=True,
figsize=(12, 7))
ax1.scatter(x=d0.acousticness, y=d0.loudness,
alpha=0.75, s=2)
ax1.set_title('Explicit vs Non-Explicit')
ax1.set_ylabel('Non-Explicit \n Loudness')

ax2.scatter(x=d1.acousticness, y=d1.loudness,
alpha=0.75, s=2)
ax2.set_xlabel('Acousticness')
ax2.set_ylabel('Explicit \n Loudness')
```



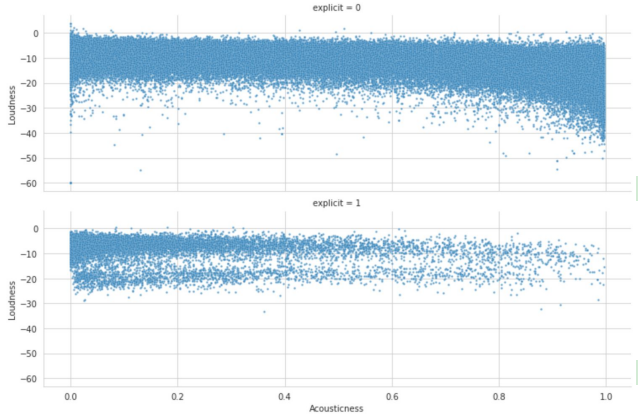
seaborn

```
fig, ax = plt.subplots(figsize=(12, 6))

with sns.axes_style("whitegrid"):
    viz = sns.histplot(data=dataset, x="acousticness",
binwidth=.02, ax=ax)
    viz.set_title("Histogram of Acousticness")
    viz.set_xlabel('Acousticness')
    viz
```

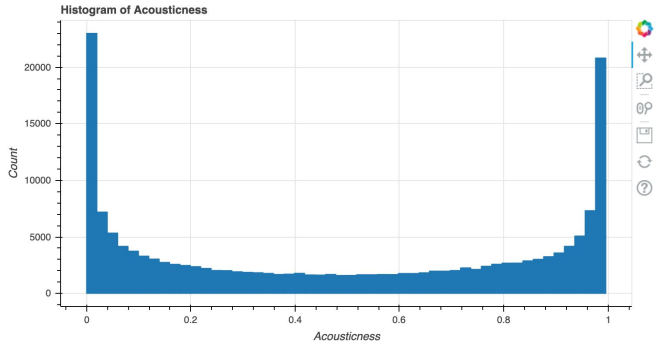


seaborn

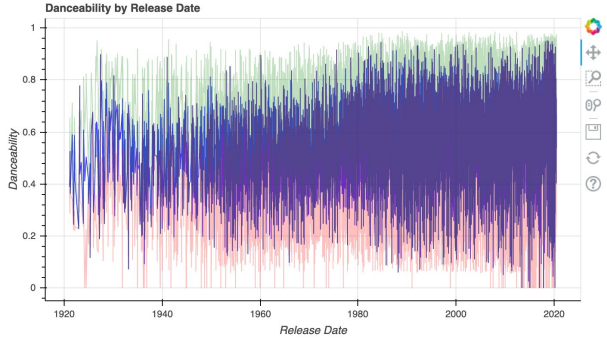
```
with sns.axes_style("whitegrid"):
    g = sns.FacetGrid(dataset, row="explicit",
                      aspect = 3, sharex=True, height=3.5)
    g.map(sns.scatterplot, "acousticness", 'loudness',
          alpha = .75, s = 6)
    g.set_ylabels('Loudness')
    g.set_xlabels('Acousticness')
    g
```

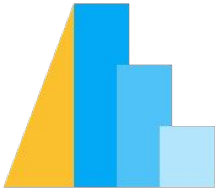



```
output_notebook()
hist, edges = np.histogram(dataset.acousticness,
bins=50)

p = figure(title="Histogram of Acousticness",
           y_axis_label='Count',
           x_axis_label='Acousticness',
           width=750,
           height = 400)

p.quad(top=hist, bottom=0, left=edges[:-1],
right=edges[1:])
show(p)
```



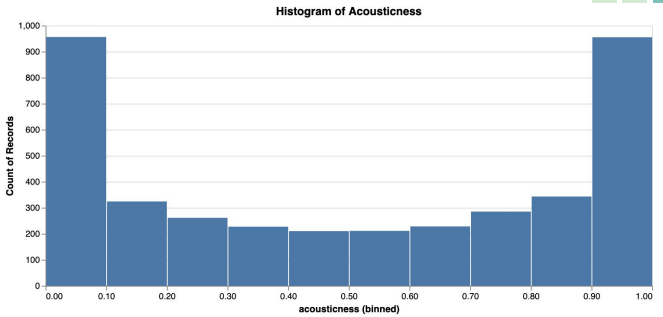
```
output_notebook()

p = figure(title="Danceability by Release Date",
           y_axis_label='Danceability',
           x_axis_label='Release Date',
           width=700,
           x_axis_type='datetime',
           height = 400)

p.line(x=grouped_sample['release_date'],
    y=grouped_sample['danceability mean'], color = 'blue', alpha = .75)

p.line(x=grouped_sample['release_date'],
    y=grouped_sample['danceability min'], color = 'red', alpha = .25)

p.line(x=grouped_sample['release_date'],
    y=grouped_sample['danceability max'], color = 'green',alpha = .25)

show(p)
```

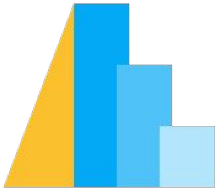




Altair

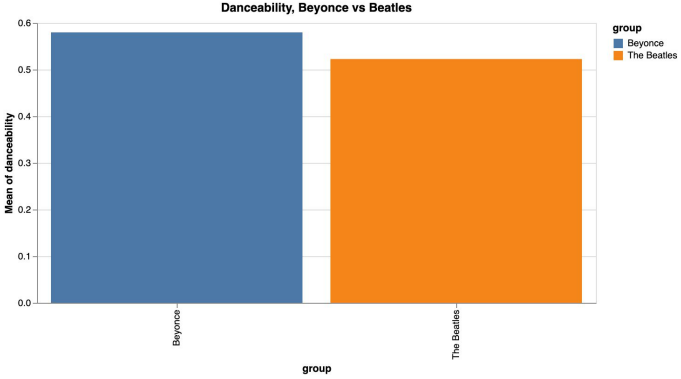
```
source = dataset.sample(axis = 0, n=4000)

viz = alt.Chart(source)
viz = viz.mark_bar()
viz = viz.encode(alt.X("acousticness",
                       bin=True), y='count()')
viz = viz.properties(title='Histogram of
Acousticness').properties(width=700, height=300)

viz
```

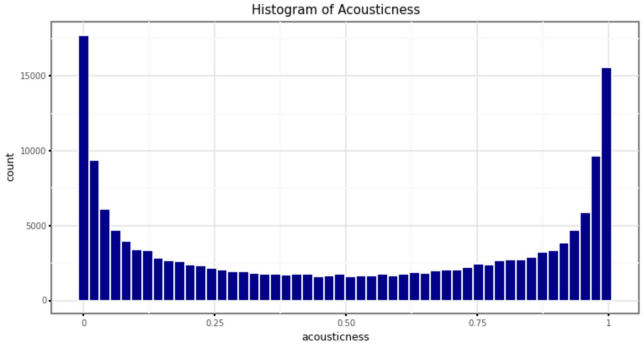




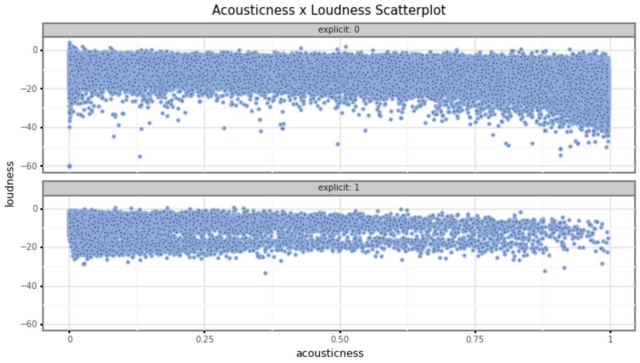
Altair

```
source = sample

viz = alt.Chart(source)
viz = viz.mark_bar()
viz = viz.encode(
    x='group:O',
    y='mean(danceability):Q',
    color='group:N'
)
viz = viz.properties(title='Danceability, Beyonce vs
Beatles').properties(width=600, height=300)

viz
```

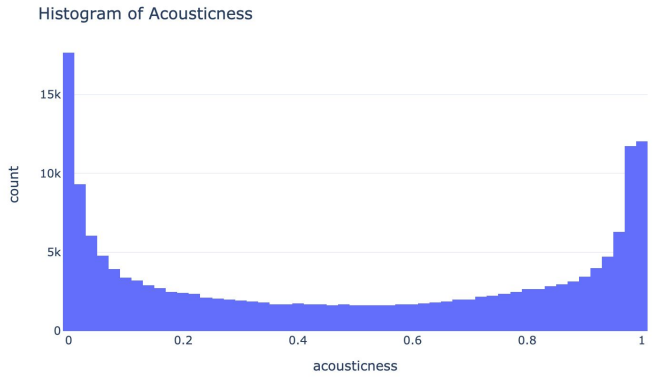





```
pno.dpi = (150)
pno.figure_size = (6,3)
ggplot(data=dataset, mapping=aes(x='acousticness')) + \
    theme_bw(base_size = 6) + \
    geom_histogram(color='white', fill = 'darkblue', bins=50) + \
    labs(title = "Histogram of Acousticness")
```




```
pno.dpi = (150)
pno.figure_size = (6,3)
ggplot(data=dataset, mapping=aes(x='acousticness', y='loudness')) + \
    facet_wrap('explicit', ncol = 1, labeller='label_both') + \
    theme_bw(base_size=6) + \
    geom_point(size = .5, fill = '#2b4570', alpha = .75, color =
"#97b5e6") + \
    labs(title = "Acousticness x Loudness Scatterplot")
```




```
fig = px.histogram(dataset, \
                   x="acousticness", \
                   nbins=50, \
                   title="Histogram of Acousticness",\
                   template='plotly_white')

fig.update_layout(
    width=700,height=400,
    margin=dict(l=15,r=25,b=15,t=40,pad=1))
fig.show()
```


```
from plotly.graph_objects import layout, XAxis, YAxis

x=dataset_sm.acousticness
y=dataset_sm.loudness
z=dataset_sm.danceability
col=dataset_sm.explicit

fig = go.Figure(
    data=[go.Scatter3d(x=x,y=y,z=z,mode='markers',
    marker=dict(size=2, color = np.array(col), colorscale = 'Viridis', opacity=0.8))])

fig.update_layout(
    title="Acousticness x Loudness x Danceability Scatterplot",
    template='plotly_white',
    autosize=True,
    width=500,
    height=500,
    scene=layout.Scene(
        xaxis=XAxis(title='Acousticness'),
        yaxis=YAxis(title='Loudness'),
        zaxis=layout.scene.ZAxis(title='Danceability')
    ),
    margin=dict(l=1,r=1,b=40,t=45,pad=1))
fig.show()
```