Telling stories with data

Éléonore Mayola
@EleonoreMayola
Github.com/eleonore9

💻 🐍 📊

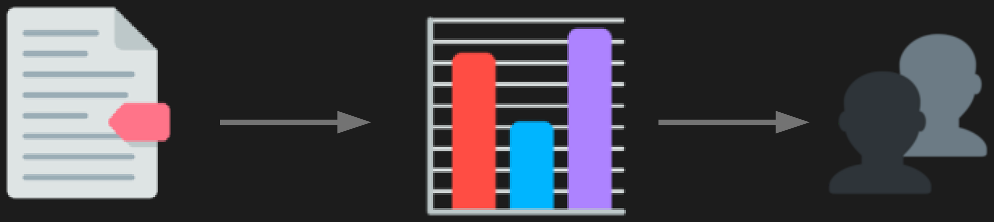
Who am I?

Web + Data
Python, Clojure, JS

👩🏾‍💻

Who is this talk for?

Anyone - all devs work with data



github.com/Eleonore9/pyconse17

Python data viz libraries

“10 Useful Python Data Visualization Libraries...”

matplotlib

seaborn ggplot Pygal

Plotly

Folium

geoplotlib

Meanwhile in the land of **Javascript**

What should we make of it?

Different libraries for different uses

Here: dataset → quick viz → tell someone a story

📑 📊 👥

What’s Altair ?

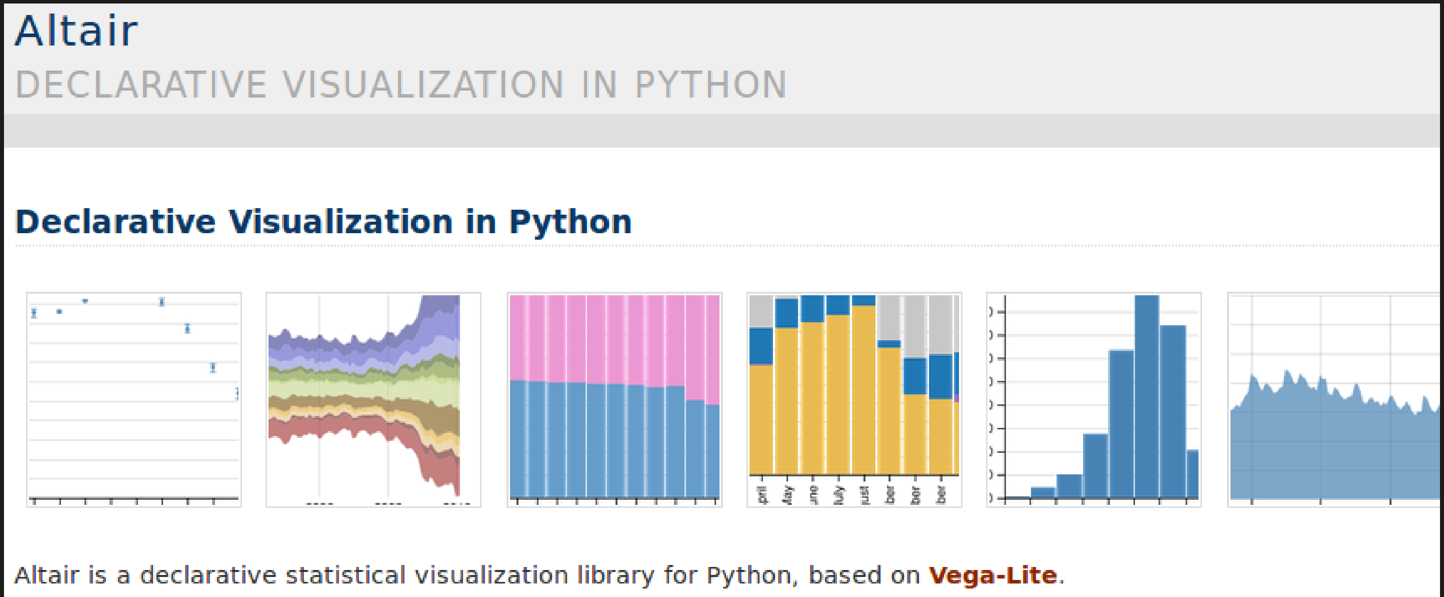
Library: https://github.com/altair-viz/altair

BSD-3 License

Main contributers: Jake Vanderplas (@jakevdp) & Brian E Granger (@ellisonbg)

What’s different with Altair?

Doc: https://altair-viz.github.io/



Altair

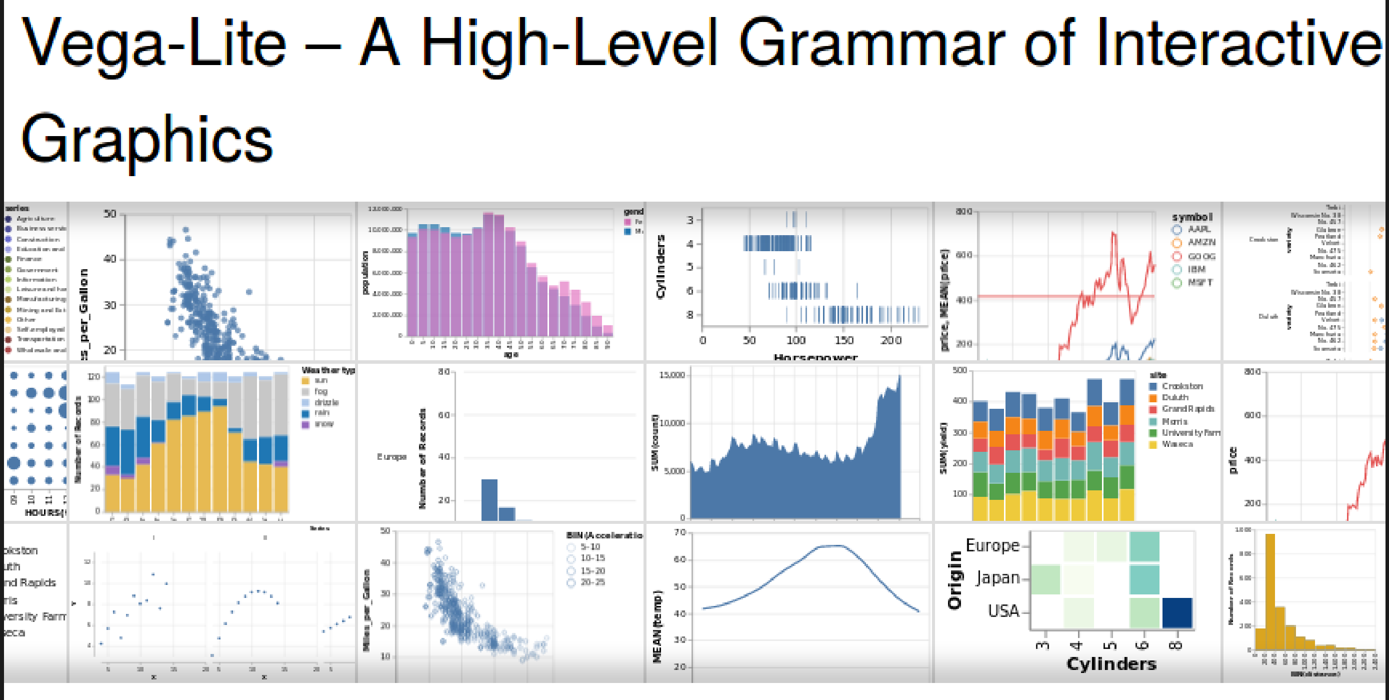
DECLARATIVE VISUALIZATION IN PYTHON

Declarative Visualization in Python

Altair is a declarative statistical visualization library for Python, based on **Vega-Lite**.

Wait, what’s Vega-Lite ?

Vega-Lite is a concise **JSON syntax** for supporting rapid generation of visualizations to support analysis.



Vega-Lite – A High-Level Grammar of Interactive Graphics

So Altair is

- Simple **API** built on top of Vega-Life
- Aimed to produce beautiful **visualizations** with a **minimal amount of code**

Note: Still at an early stage (v1.2.0) → more docs and plots to come

Let’s have a closer look at Altair

Installation (Python 3)

```
$ pip install altair
$ pip install --upgrade notebook
$ jupyter nbextension install --sys-prefix --py vega
```

For Conda users:

```
$ conda install altair --channel conda-forge
```

Note: To render Vega-Lite in a notebook → extension IPython Vega needed

Let’s have a closer look at Altair

Installation (Python 2)

```
$ mkvirtualenv --python=/usr/bin/python3.4 altair
$ pip3 altair
$ pip3 install --upgrade notebook
$ jupyter nbextension install --sys-prefix --py vega
```

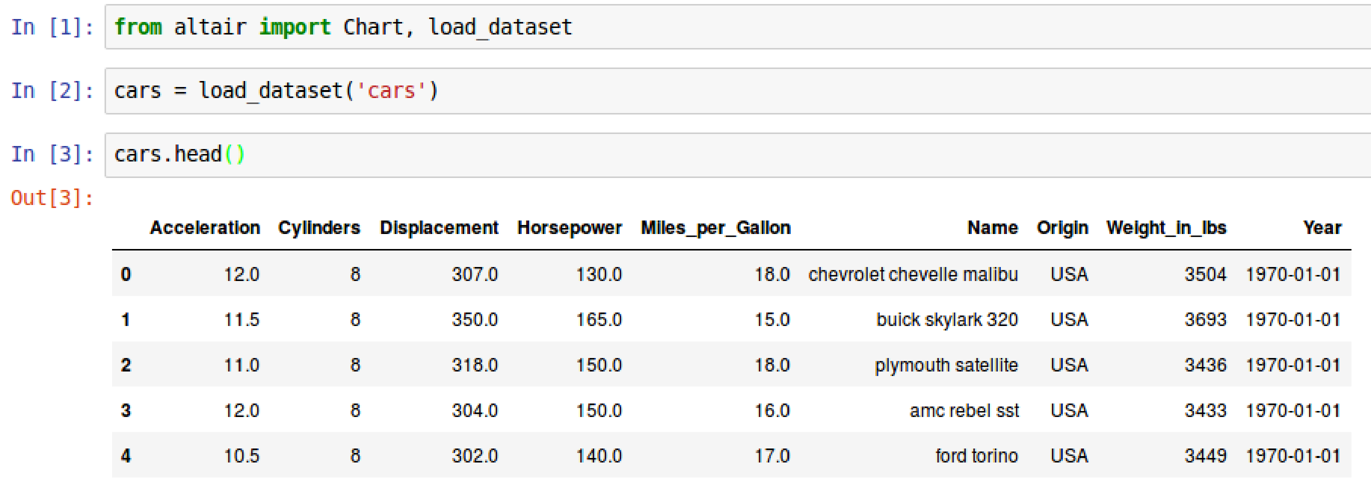
CLI output

```
To initialize this nbextension in the browser every
time the notebook (or other app) loads:

    jupyter nbextension enable vega --py --sysprefix
```

Let’s have a closer look at Altair

Doc example – import Altair, get dataset

```
$ jupyter nbextension enable vega --py --sys-prefix
$ jupyter notebook
```

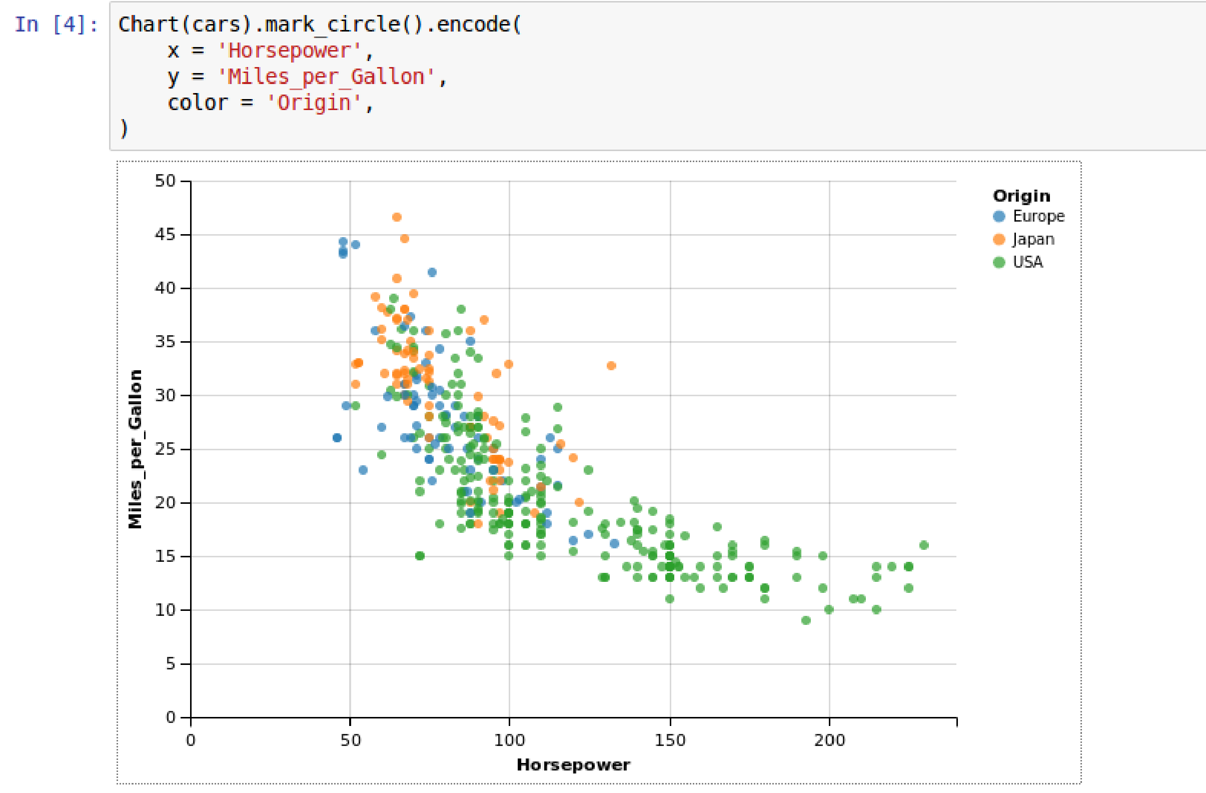


```
In [1]: from altair import Chart, load_dataset

In [2]: cars = load_dataset('cars')

In [3]: cars.head()
Out[3]:
```

	Acceleration	Cylinders	Displacement	Horsepower	Miles_per_Gallon	Name	Origin	Weight_in_lbs	Year
0	12.0	8	307.0	130.0	18.0	chevrolet chevelle malibu	USA	3504	1970-01-01
1	11.5	8	350.0	165.0	15.0	buick skylark 320	USA	3693	1970-01-01
2	11.0	8	318.0	150.0	18.0	plymouth satellite	USA	3436	1970-01-01
3	12.0	8	304.0	150.0	16.0	amc rebel sst	USA	3433	1970-01-01
4	10.5	8	302.0	140.0	17.0	ford torino	USA	3449	1970-01-01

Let’s have a closer look at Altair

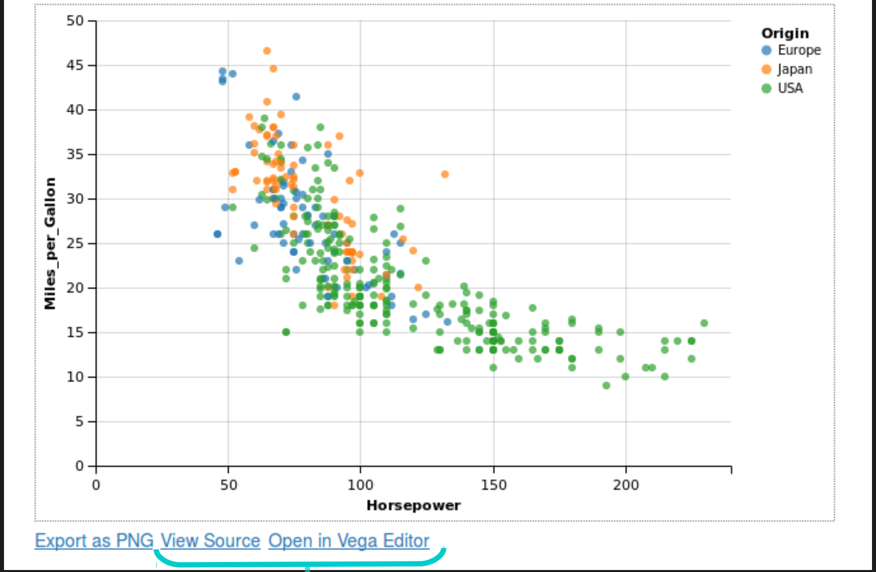
Doc example – create a plot

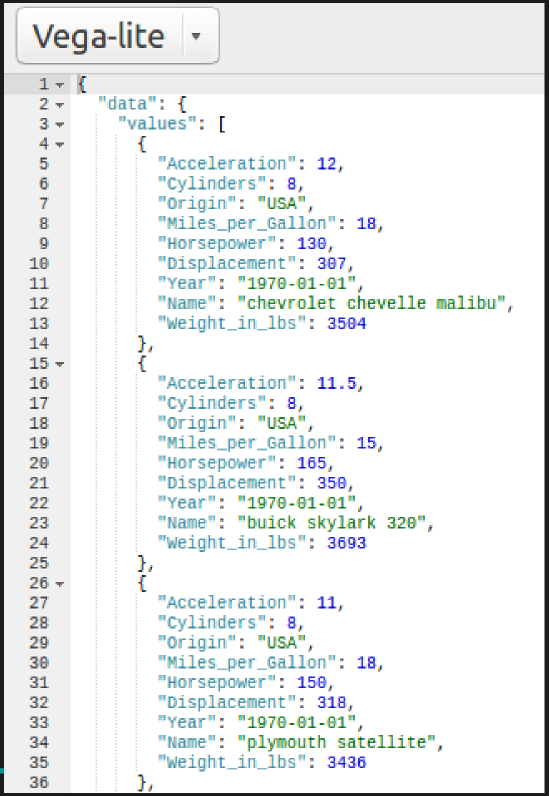


```
In [4]: Chart(cars).mark_circle().encode(
            x = 'Horsepower',
            y = 'Miles_per_Gallon',
            color = 'Origin',
        )
```

Let’s have a closer look at Altair

Doc example – more on the plot





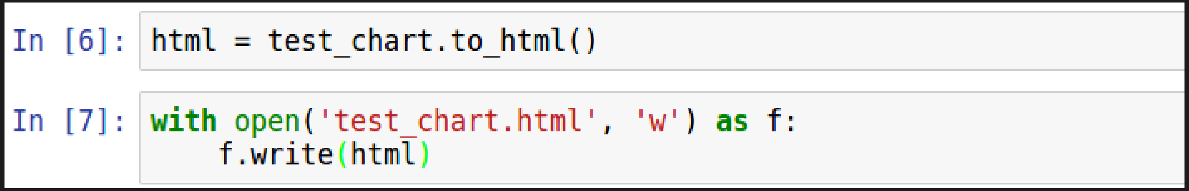
```
{
  "data": {
    "values": [
      {
        "Acceleration": 12,
        "Cylinders": 8,
        "Origin": "USA",
        "Miles_per_Gallon": 18,
        "Horsepower": 130,
        "Displacement": 307,
        "Year": "1970-01-01",
        "Name": "chevrolet chevelle malibu",
        "Weight_in_lbs": 3504
      },
      {
        "Acceleration": 11.5,
        "Cylinders": 8,
        "Origin": "USA",
        "Miles_per_Gallon": 15,
        "Horsepower": 165,
        "Displacement": 350,
        "Year": "1970-01-01",
        "Name": "buick skylark 320",
        "Weight_in_lbs": 3693
      },
      {
        "Acceleration": 11,
        "Cylinders": 8,
        "Origin": "USA",
        "Miles_per_Gallon": 18,
        "Horsepower": 150,
        "Displacement": 318,
        "Year": "1970-01-01",
        "Name": "plymouth satellite",
        "Weight_in_lbs": 3436
      },
```

Alternative:

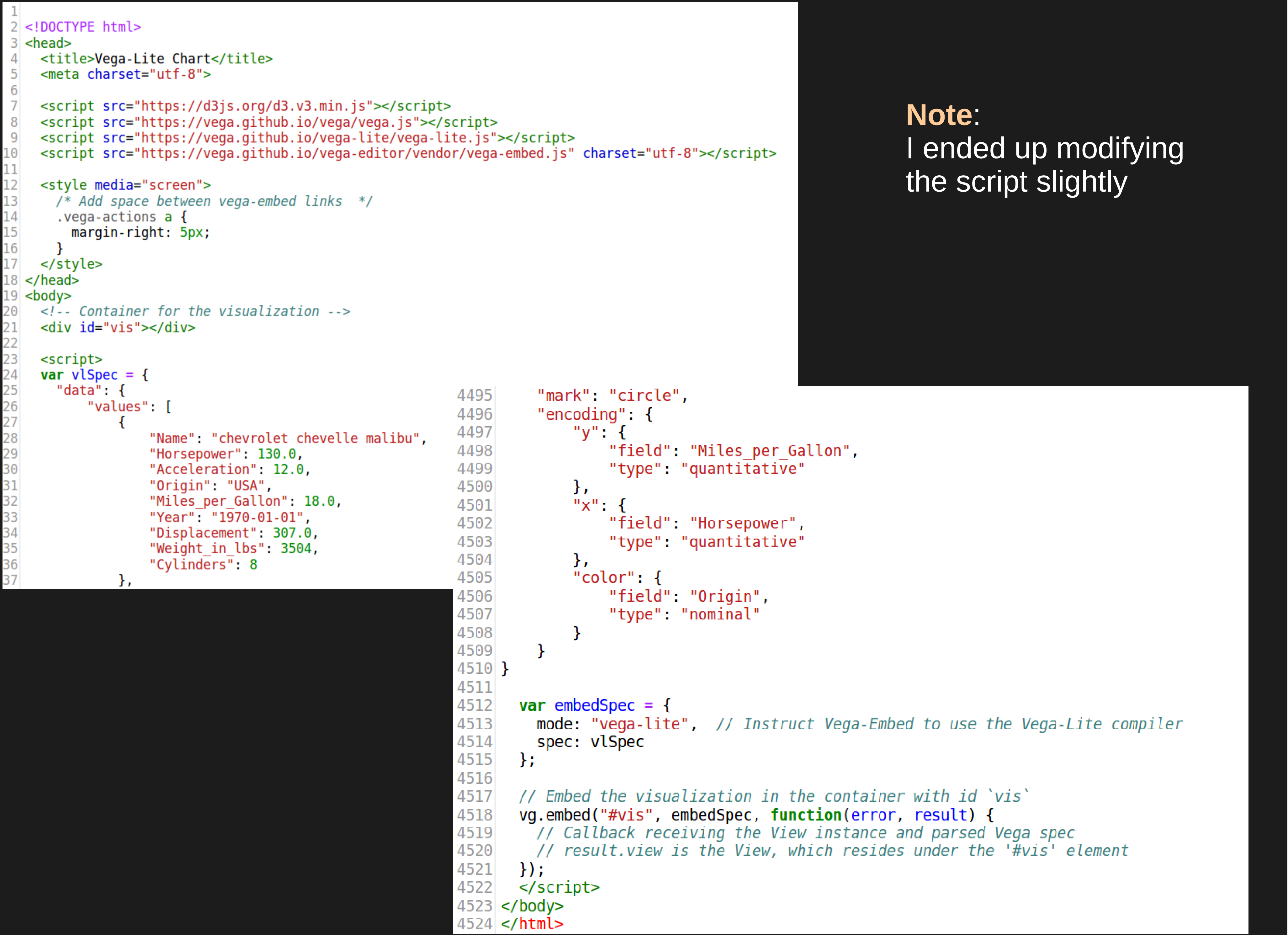
```
print(test_chart.to_json(indent=2))
```

How to share the results?

Generate html



```
In [6]: html = test_chart.to_html()

In [7]: with open('test_chart.html', 'w') as f:
            f.write(html)
```



```html

<!DOCTYPE html>
<head>
  <title>Vega-Lite Chart</title>
  <meta charset="utf-8">

  <script src="https://d3js.org/d3.v3.min.js"></script>
  <script src="https://vega.github.io/vega/vega.js"></script>
  <script src="https://vega.github.io/vega-lite/vega-lite.js"></script>
  <script src="https://vega.github.io/vega-editor/vendor/vega-embed.js" charset="utf-8"></script>

  <style media="screen">
    /* Add space between vega-embed links  */
    .vega-actions a {
      margin-right: 5px;
    }
  </style>
</head>
<body>
  <!-- Container for the visualization -->
  <div id="vis"></div>

  <script>
  var vlSpec = {
    "data": {
        "values": [
            {
                "Name": "chevrolet chevelle malibu",
                "Horsepower": 130.0,
                "Acceleration": 12.0,
                "Origin": "USA",
                "Miles_per_Gallon": 18.0,
                "Year": "1970-01-01",
                "Displacement": 307.0,
                "Weight_in_lbs": 3504,
                "Cylinders": 8
            },
```

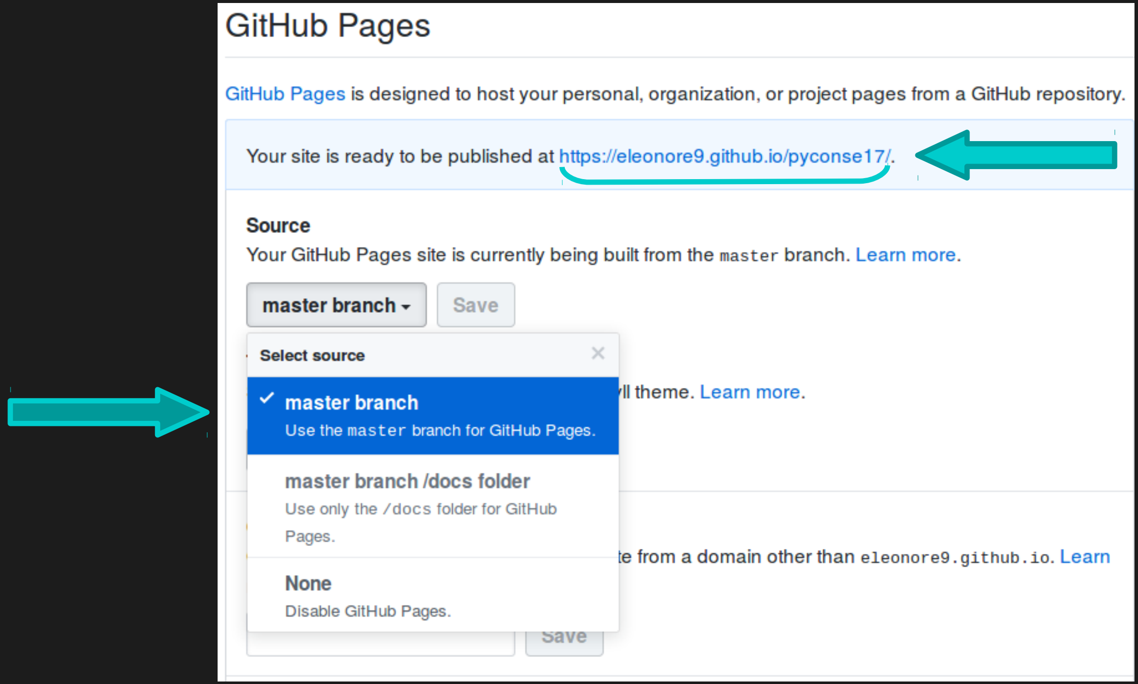
```html
    "mark": "circle",
    "encoding": {
        "y": {
            "field": "Miles_per_Gallon",
            "type": "quantitative"
        },
        "x": {
            "field": "Horsepower",
            "type": "quantitative"
        },
        "color": {
            "field": "Origin",
            "type": "nominal"
        }
    }
}

  var embedSpec = {
    mode: "vega-lite",  // Instruct Vega-Embed to use the Vega-Lite compiler
    spec: vlSpec
  };

  // Embed the visualization in the container with id `vis`
  vg.embed("#vis", embedSpec, function(error, result) {
    // Callback receiving the View instance and parsed Vega spec
    // result.view is the View, which resides under the '#vis' element
  });
  </script>
</body>
</html>
```

Note:
I ended up modifying
the script slightly

Produce shareable plots

Easy way → use Github pages

Note: rename your html file to “**index.html**”



GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Your site is ready to be published at https://eleonore9.github.io/pyconse17/.

Source

Your GitHub Pages site is currently being built from the `master` branch. Learn more.

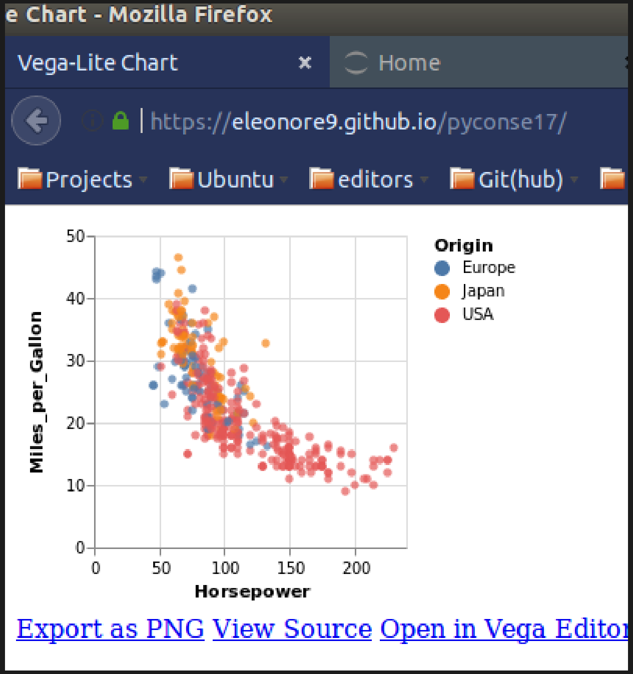
master branch ▾
Save
Select source ×
✓ master branch
Use the master branch for GitHub Pages.
master branch /docs folder
Use only the /docs folder for GitHub Pages.
None
Disable GitHub Pages.
ll theme. Learn more.
te from a domain other than eleonore9.github.io. Learn
Save

Produce shareable plots

Workflow

1. Explore dataset and create plot(s)
2. Commit changes and push to Github
3. Share the URL

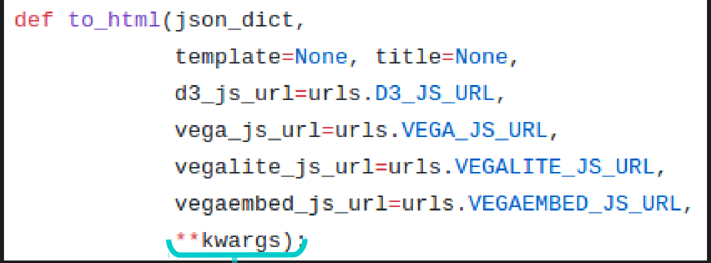
https://<gh username>.github.io/<project name>



How to add to our page

Let’s look at Altair source code

to_html function

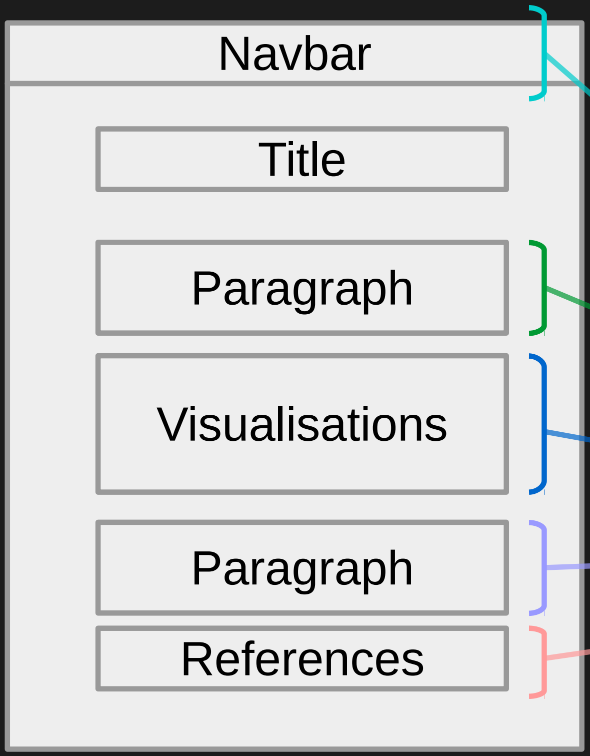


```
def to_html(json_dict,
            template=None, title=None,
            d3_js_url=urls.D3_JS_URL,
            vega_js_url=urls.VEGA_JS_URL,
            vegalite_js_url=urls.VEGALITE_JS_URL,
            vegaembed_js_url=urls.VEGAEMBED_JS_URL,
            **kwargs):
```

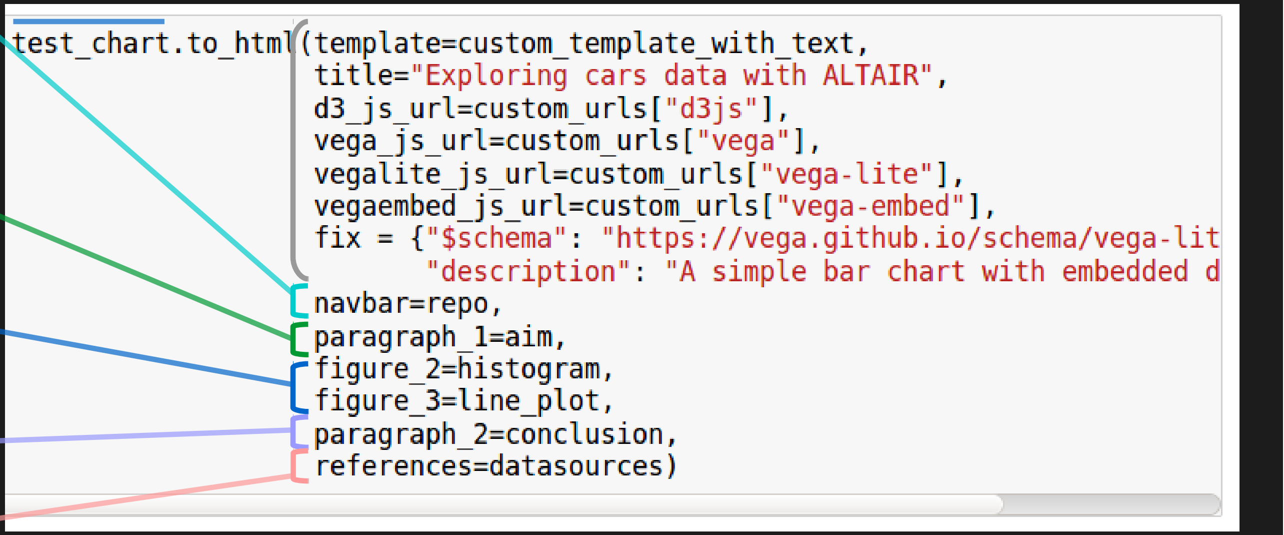
We can pass keyworded arguments 🤩

How to add to our page

Page wireframe

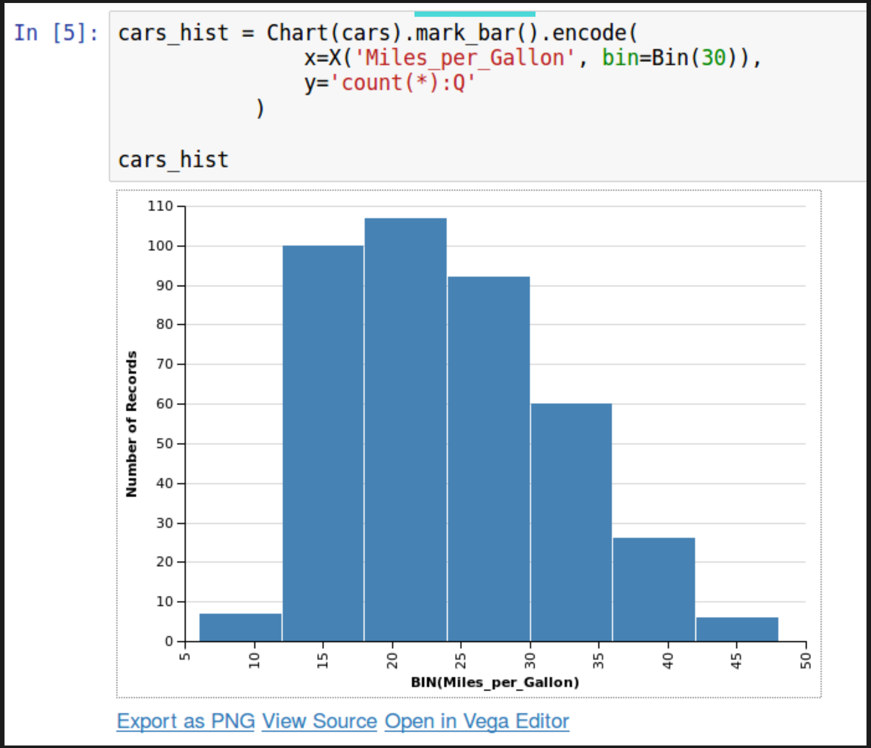


New function arguments

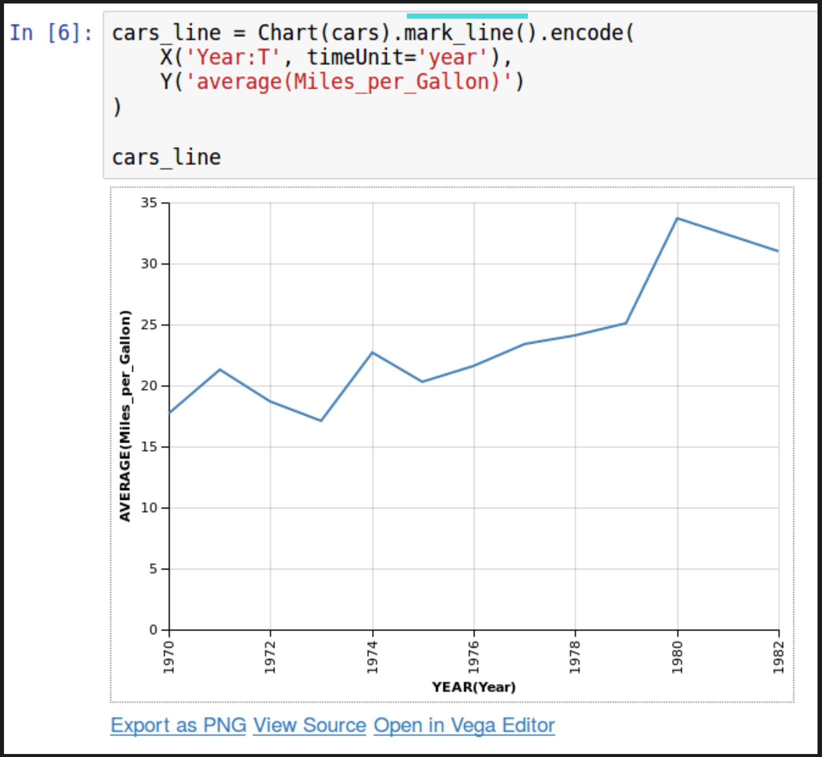


```
test_chart.to_html(template=custom_template_with_text,
                   title="Exploring cars data with ALTAIR",
                   d3_js_url=custom_urls["d3js"],
                   vega_js_url=custom_urls["vega"],
                   vegalite_js_url=custom_urls["vega-lite"],
                   vegaembed_js_url=custom_urls["vega-embed"],
                   fix = {"$schema": "https://vega.github.io/schema/vega-lit
                          "description": "A simple bar chart with embedded d
                   navbar=repo,
                   paragraph_1=aim,
                   figure_2=histogram,
                   figure_3=line_plot,
                   paragraph_2=conclusion,
                   references=datasources)
```

How to add to our page

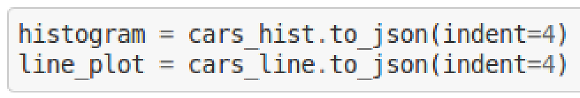


```
In [5]: cars_hist = Chart(cars).mark_bar().encode(
                     x=X('Miles_per_Gallon', bin=Bin(30)),
                     y='count(*):Q'
                )

        cars_hist
```



```
In [6]: cars_line = Chart(cars).mark_line().encode(
            X('Year:T', timeUnit='year'),
            Y('average(Miles_per_Gallon)')
        )

        cars_line
```

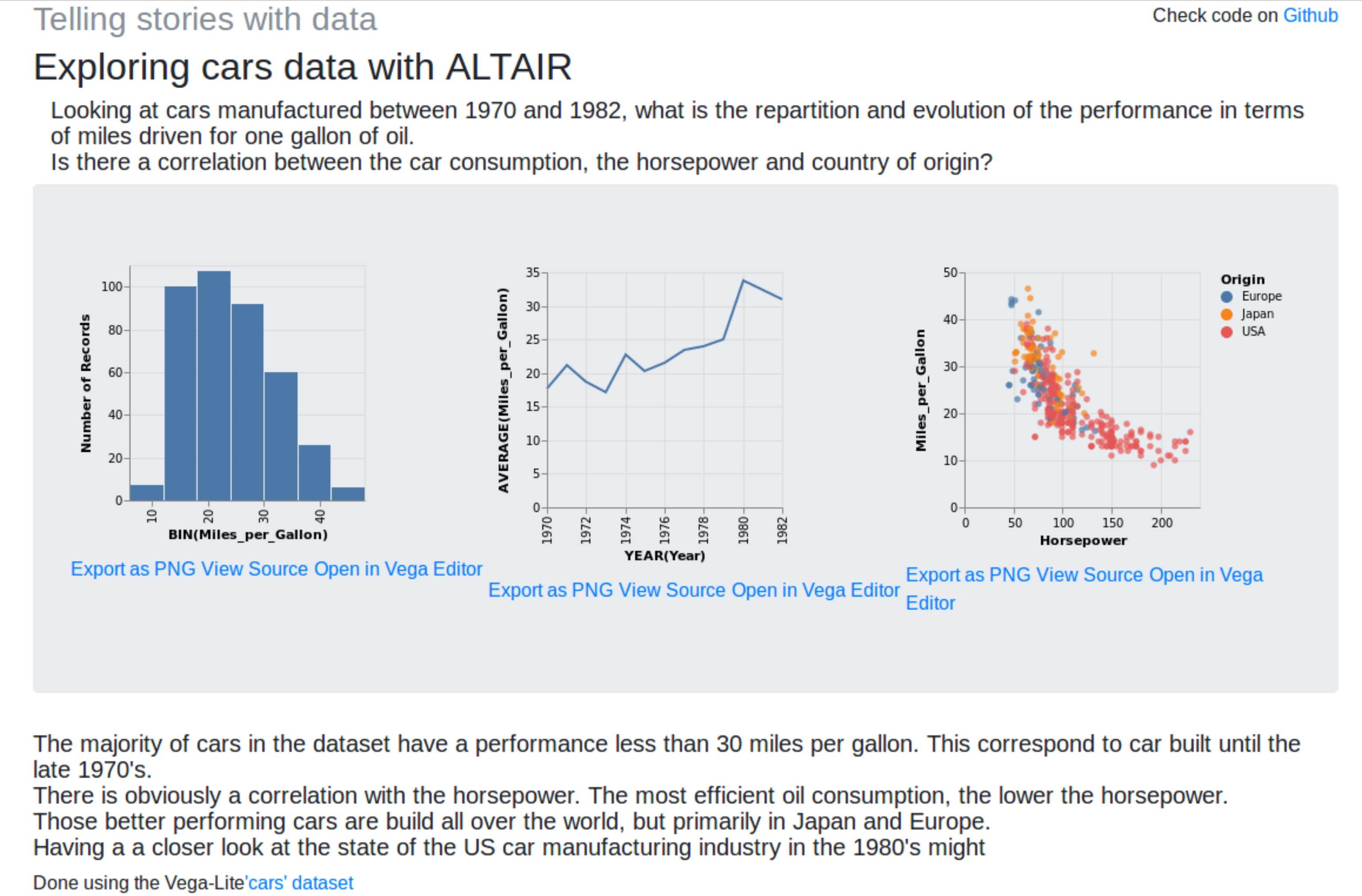
Note: I converted extras Altair Chart objects to JSON



```
histogram = cars_hist.to_json(indent=4)
line_plot = cars_line.to_json(indent=4)
```

Tada 🎉

https://eleonore9.github.io/pyconse17/



Telling stories with data

Check code on Github

Exploring cars data with ALTAIR

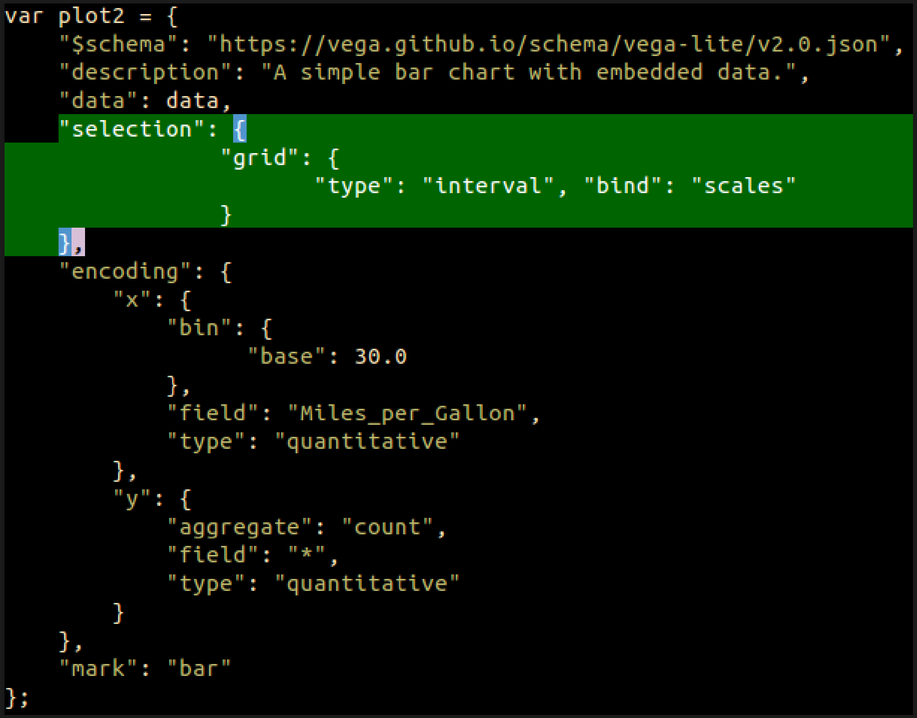
Looking at cars manufactured between 1970 and 1982, what is the repartition and evolution of the performance in terms of miles driven for one gallon of oil.
Is there a correlation between the car consumption, the horsepower and country of origin?

The majority of cars in the dataset have a performance less than 30 miles per gallon. This correspond to car built until the late 1970's.
There is obviously a correlation with the horsepower. The most efficient oil consumption, the lower the horsepower.
Those better performing cars are build all over the world, but primarily in Japan and Europe.
Having a a closer look at the state of the US car manufacturing industry in the 1980's might

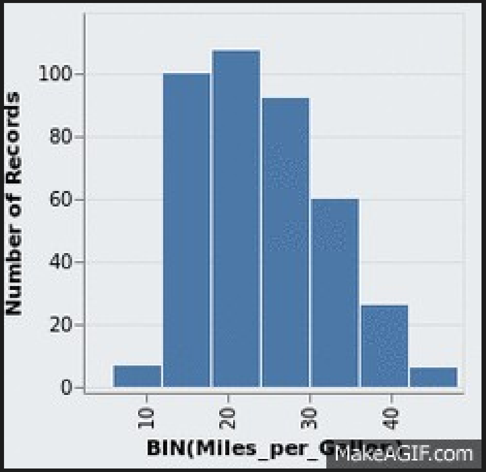
Done using the Vega-Lite'cars' dataset

Let’s add interactivity

Plot zoom

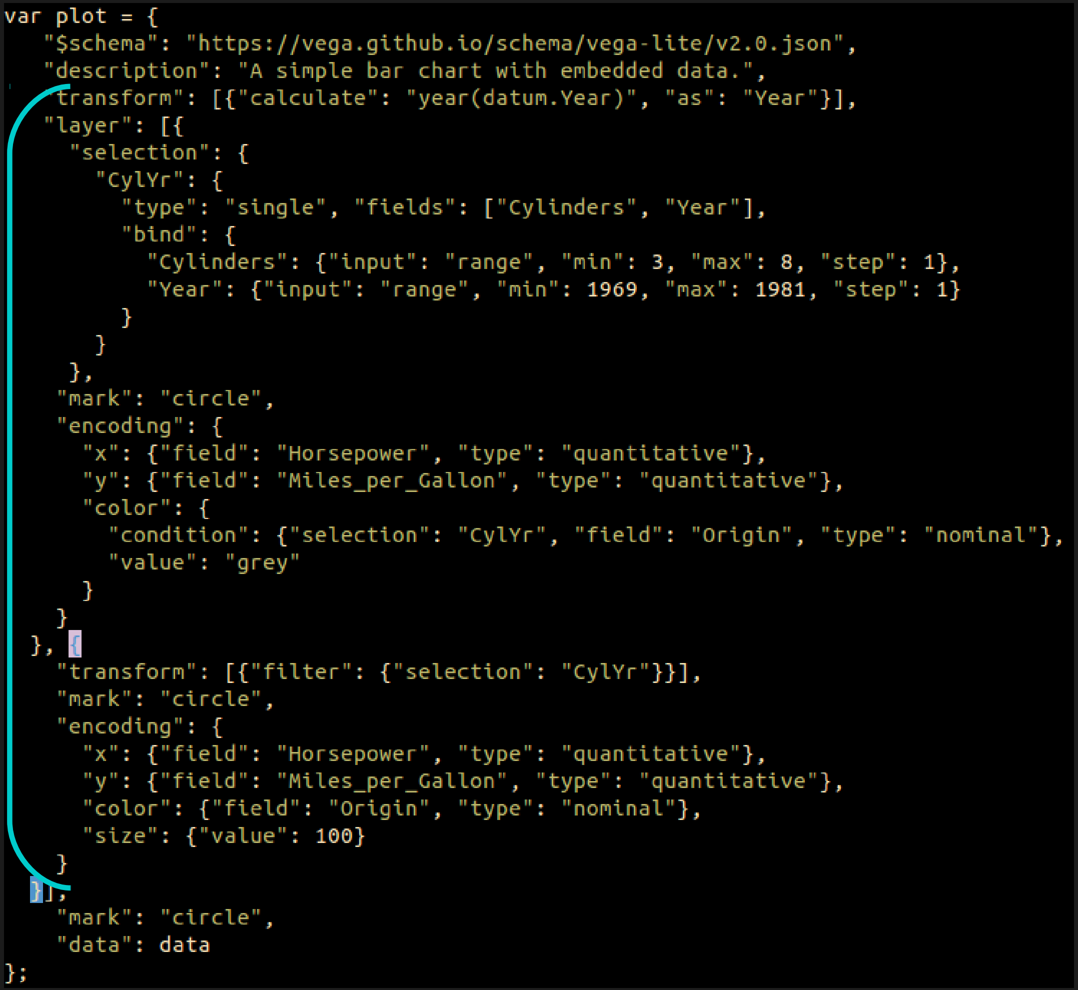


```
var plot2 = {
    "$schema": "https://vega.github.io/schema/vega-lite/v2.0.json",
    "description": "A simple bar chart with embedded data.",
    "data": data,
    "selection": {
                "grid": {
                        "type": "interval", "bind": "scales"
                }
    },
    "encoding": {
        "x": {
            "bin": {
                  "base": 30.0
            },
            "field": "Miles_per_Gallon",
            "type": "quantitative"
        },
        "y": {
            "aggregate": "count",
            "field": "*",
            "type": "quantitative"
        }
    },
    "mark": "bar"
};
```

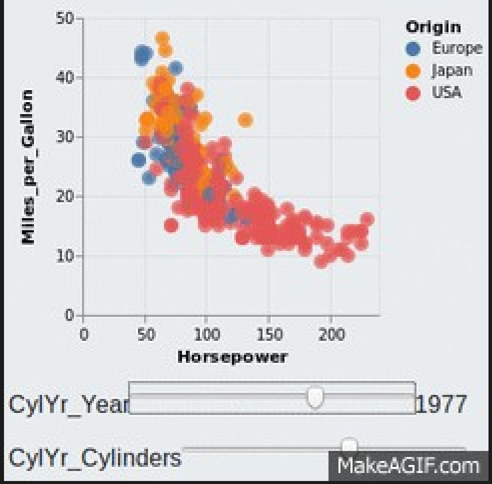


Let’s add interactivity

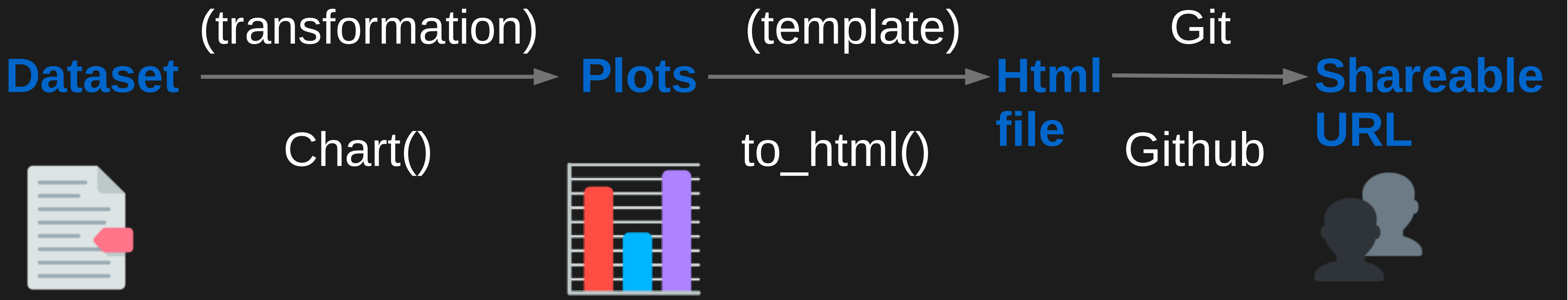
Query widgets



```
var plot = {
   "$schema": "https://vega.github.io/schema/vega-lite/v2.0.json",
   "description": "A simple bar chart with embedded data.",
    "transform": [{"calculate": "year(datum.Year)", "as": "Year"}],
   "layer": [{
     "selection": {
       "CylYr": {
         "type": "single", "fields": ["Cylinders", "Year"],
         "bind": {
           "Cylinders": {"input": "range", "min": 3, "max": 8, "step": 1},
           "Year": {"input": "range", "min": 1969, "max": 1981, "step": 1}
         }
       }
     },
    "mark": "circle",
    "encoding": {
      "x": {"field": "Horsepower", "type": "quantitative"},
      "y": {"field": "Miles_per_Gallon", "type": "quantitative"},
      "color": {
        "condition": {"selection": "CylYr", "field": "Origin", "type": "nominal"},
        "value": "grey"
      }
    }
  }, {
    "transform": [{"filter": {"selection": "CylYr"}}],
    "mark": "circle",
    "encoding": {
      "x": {"field": "Horsepower", "type": "quantitative"},
      "y": {"field": "Miles_per_Gallon", "type": "quantitative"},
      "color": {"field": "Origin", "type": "nominal"},
      "size": {"value": 100}
    }
  }],
    "mark": "circle",
    "data": data
};
```



Let’s look back at the steps



Dataset

(transformation)

Chart()

Plots

(template)

to_html()

Html file

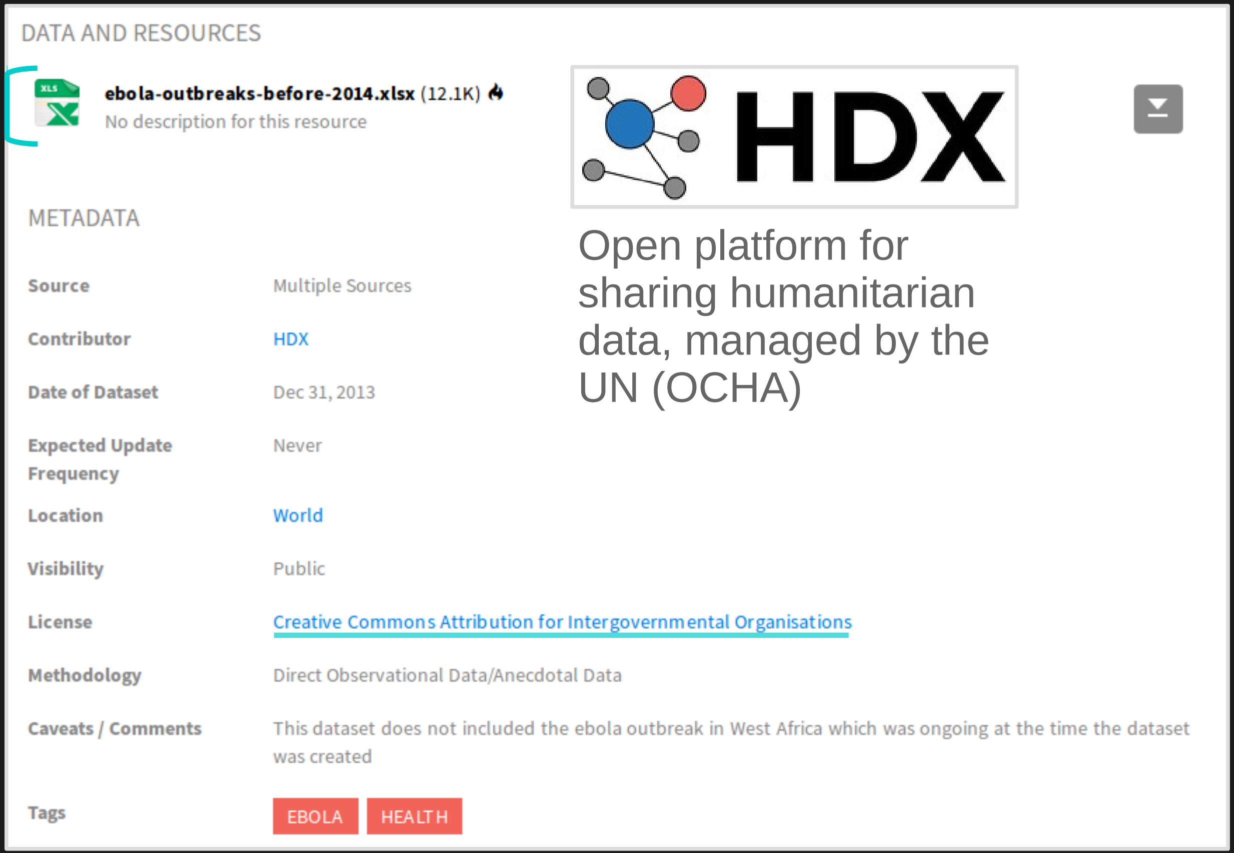
Git

Github

Shareable URL

Looking at a dataset with Altair

Ebola outbreaks before 2014



DATA AND RESOURCES

ebola-outbreaks-before-2014.xlsx (12.1K)
No description for this resource

HDX

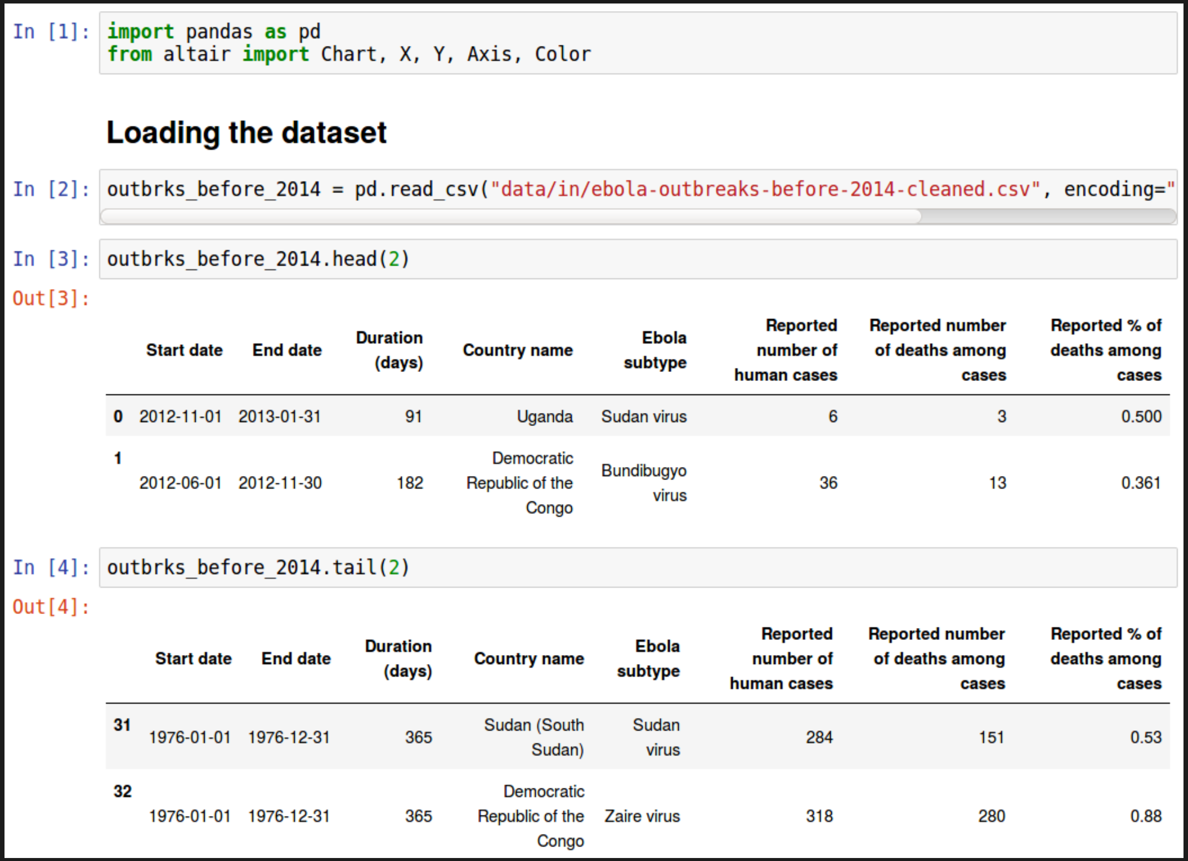
Open platform for sharing humanitarian data, managed by the UN (OCHA)

METADATA

Source	Multiple Sources
Contributor	HDX
Date of Dataset	Dec 31, 2013
Expected Update Frequency	Never
Location	World
Visibility	Public
License	Creative Commons Attribution for Intergovernmental Organisations
Methodology	Direct Observational Data/Anecdotal Data
Caveats / Comments	This dataset does not included the ebola outbreak in West Africa which was ongoing at the time the dataset was created
Tags	EBOLA HEALTH

Source: https://data.humdata.org/dataset/ebola-outbreaks-before-2014

Looking at a dataset with Altair



```
In [1]: import pandas as pd
        from altair import Chart, X, Y, Axis, Color
```

Loading the dataset

```
In [2]: outbrks_before_2014 = pd.read_csv("data/in/ebola-outbreaks-before-2014-cleaned.csv", encoding="
```

```
In [3]: outbrks_before_2014.head(2)
```

Out[3]:

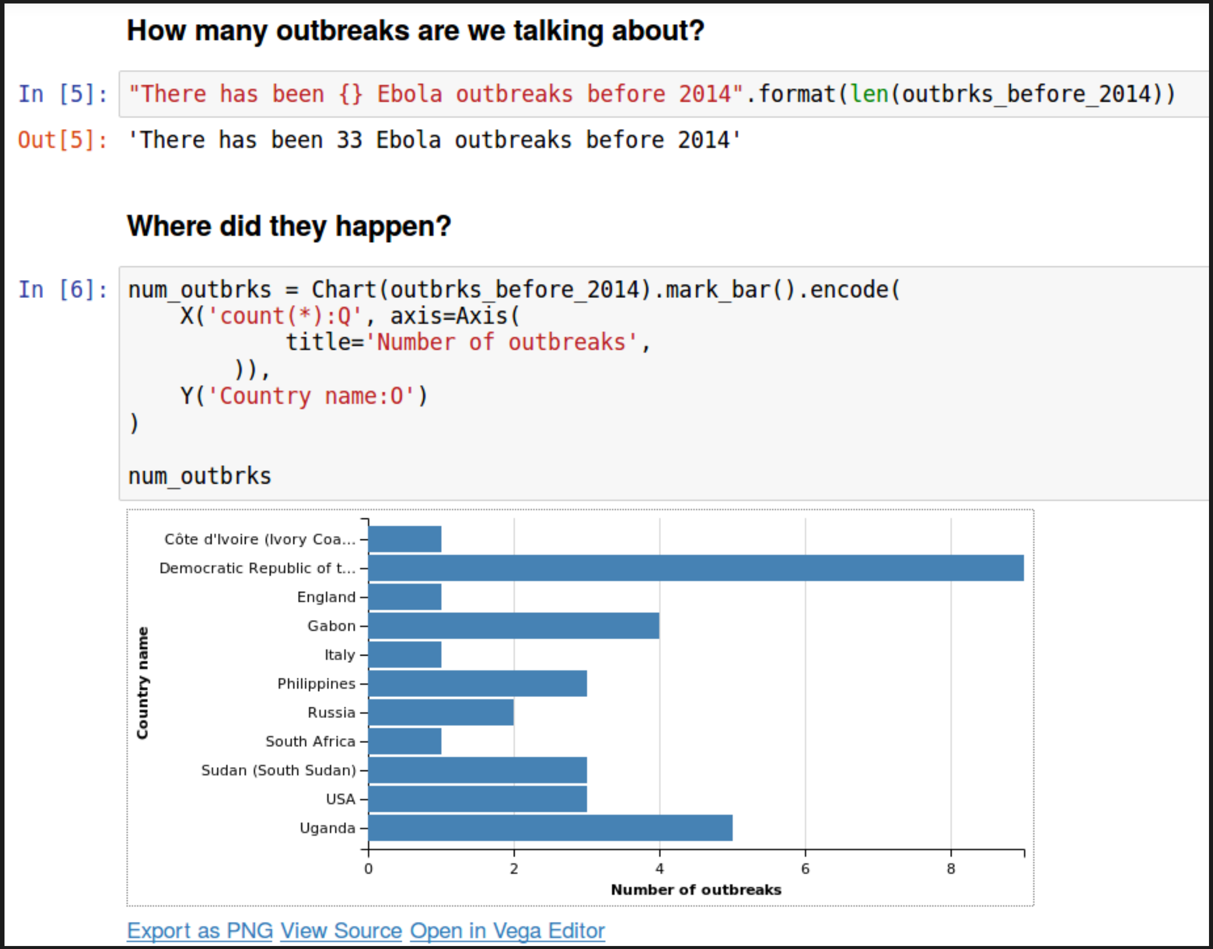
	Start date	End date	Duration (days)	Country name	Ebola subtype	Reported number of human cases	Reported number of deaths among cases	Reported % of deaths among cases
0	2012-11-01	2013-01-31	91	Uganda	Sudan virus	6	3	0.500
1	2012-06-01	2012-11-30	182	Democratic Republic of the Congo	Bundibugyo virus	36	13	0.361

```
In [4]: outbrks_before_2014.tail(2)
```

Out[4]:

	Start date	End date	Duration (days)	Country name	Ebola subtype	Reported number of human cases	Reported number of deaths among cases	Reported % of deaths among cases
31	1976-01-01	1976-12-31	365	Sudan (South Sudan)	Sudan virus	284	151	0.53
32	1976-01-01	1976-12-31	365	Democratic Republic of the Congo	Zaire virus	318	280	0.88

Example use of Altair



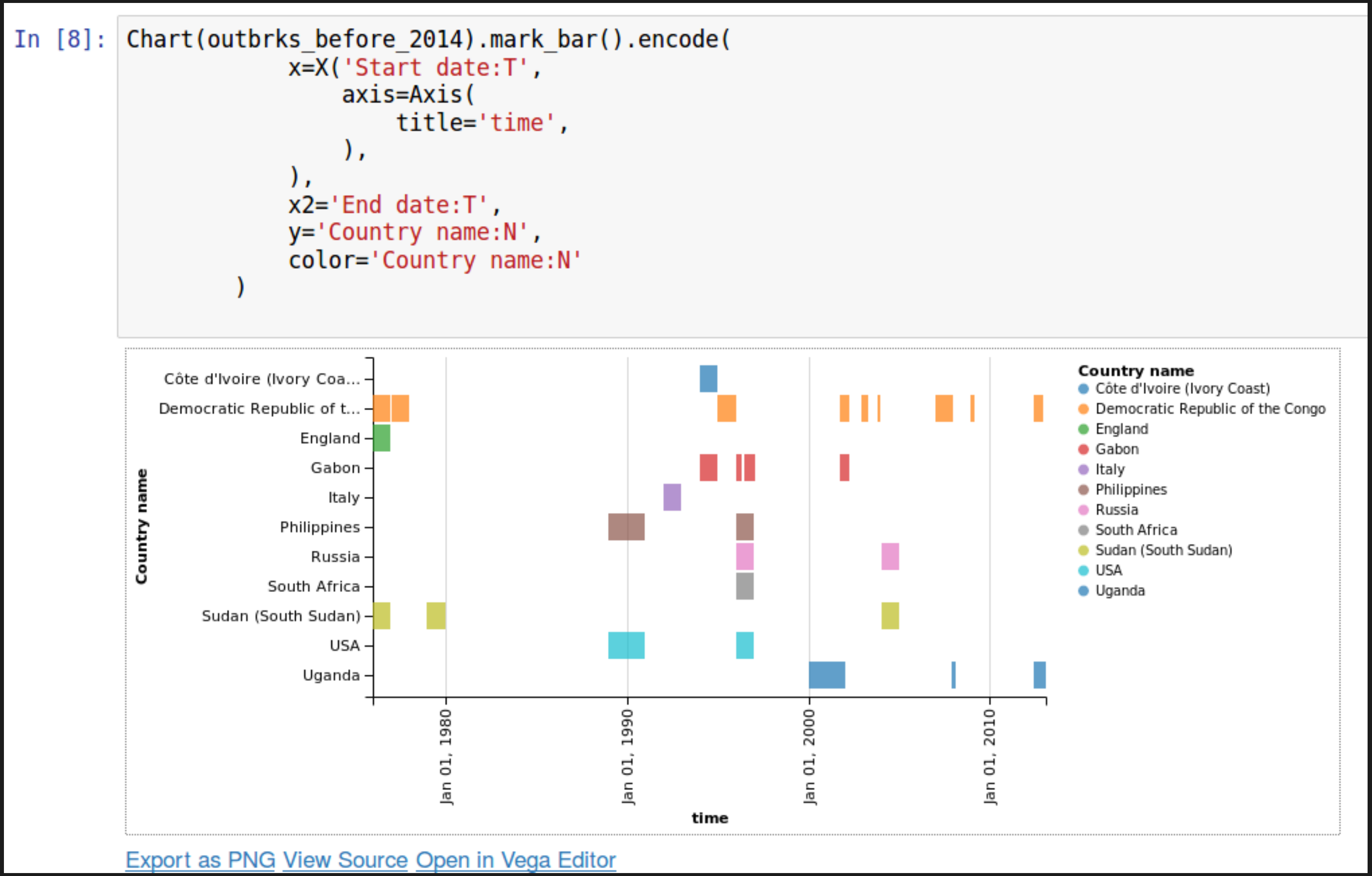
How many outbreaks are we talking about?

```
In [5]: "There has been {} Ebola outbreaks before 2014".format(len(outbrks_before_2014))

Out[5]: 'There has been 33 Ebola outbreaks before 2014'
```

Where did they happen?

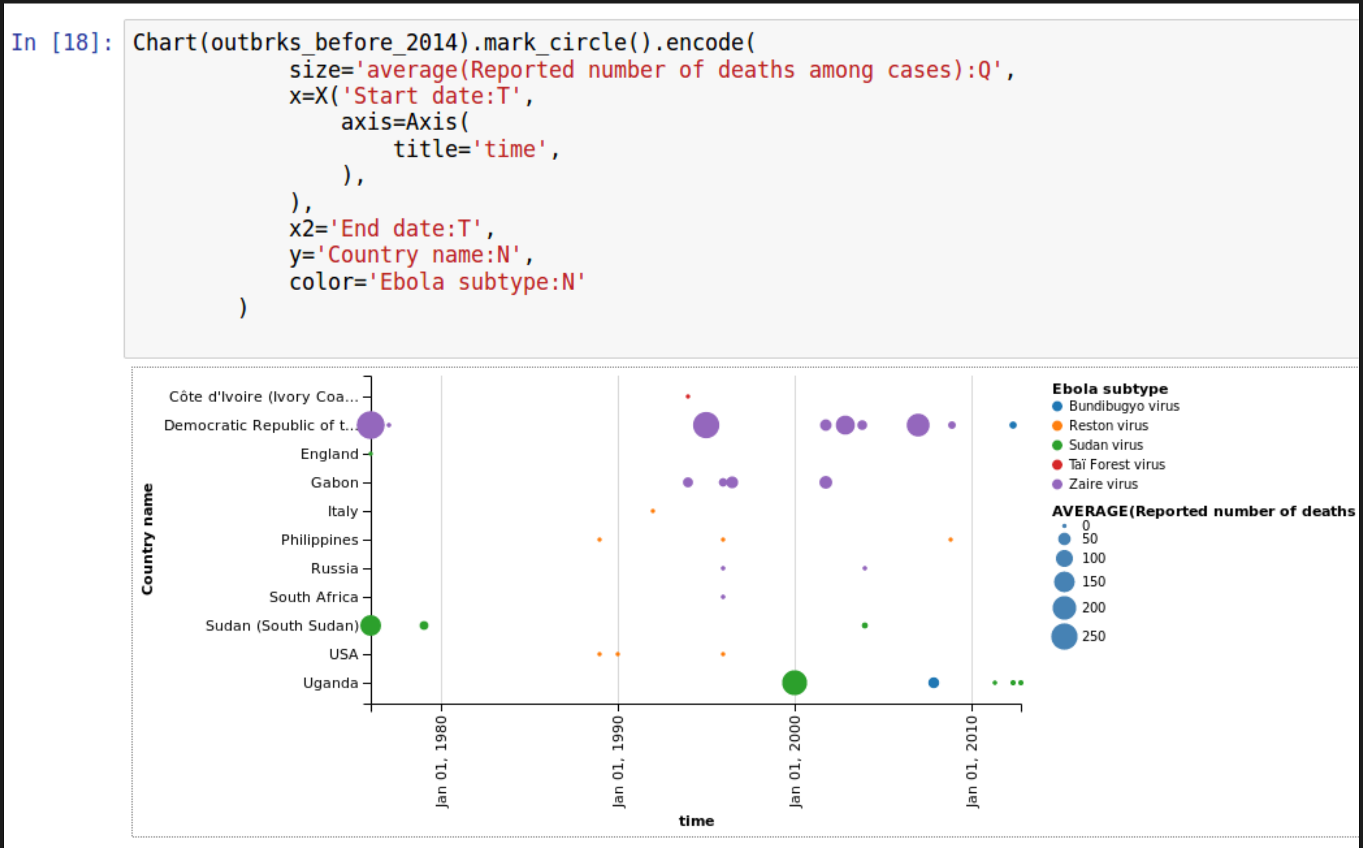
```
In [6]: num_outbrks = Chart(outbrks_before_2014).mark_bar().encode(
            X('count(*):Q', axis=Axis(
                    title='Number of outbreaks',
                )),
            Y('Country name:O')
        )

        num_outbrks
```

Example use of Altair



```
In [8]: Chart(outbrks_before_2014).mark_bar().encode(
                    x=X('Start date:T',
                        axis=Axis(
                            title='time',
                        ),
                    ),
                    x2='End date:T',
                    y='Country name:N',
                    color='Country name:N'
                )
```

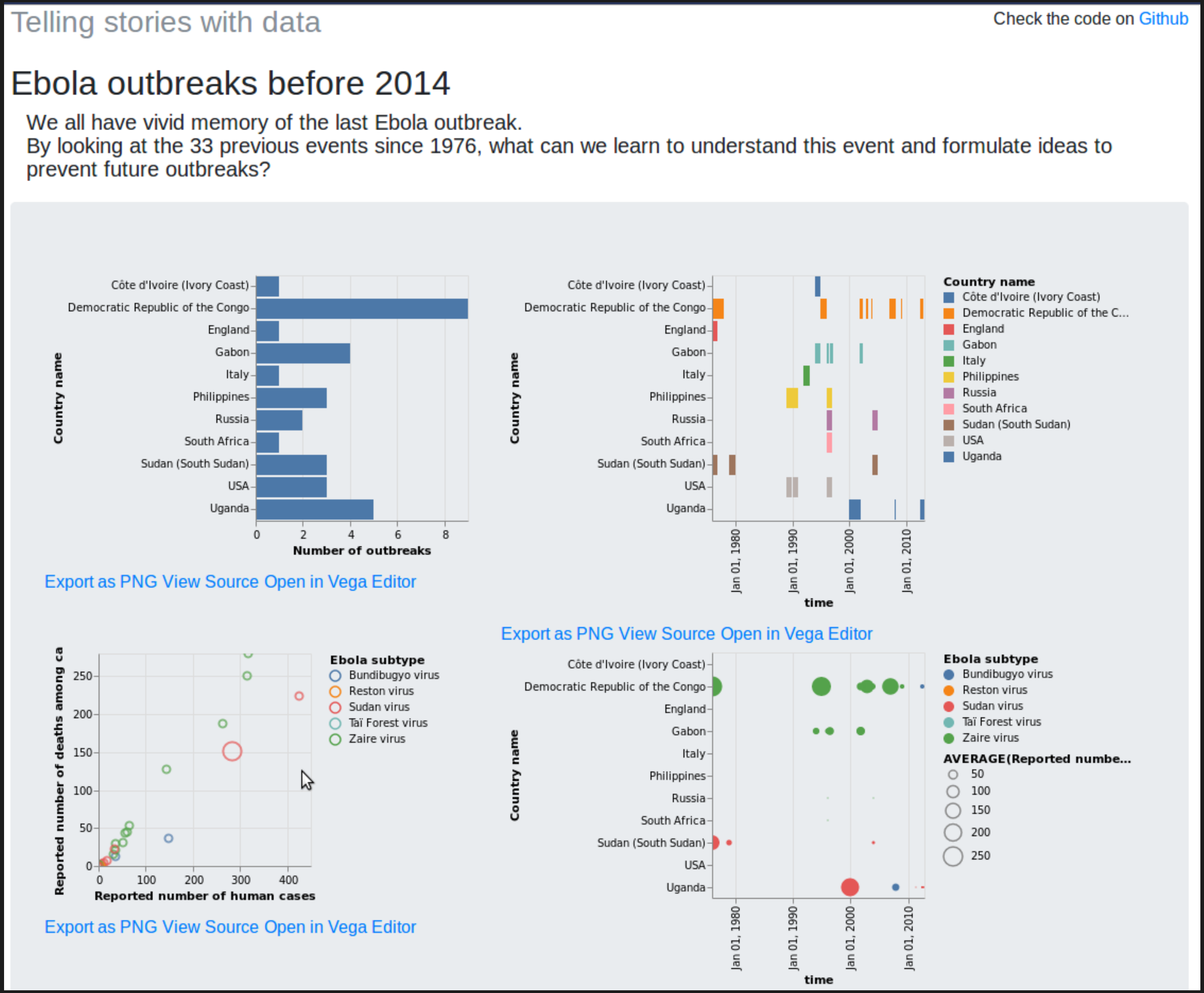
Example use of Altair



```
In [18]: Chart(outbrks_before_2014).mark_circle().encode(
                     size='average(Reported number of deaths among cases):Q',
                     x=X('Start date:T',
                         axis=Axis(
                             title='time',
                         ),
                     ),
                     x2='End date:T',
                     y='Country name:N',
                     color='Ebola subtype:N'
                 )
```

My shareable story

http://eleonore9.github.io/ebola_outbreaks/



Telling stories with data

Check the code on Github

Ebola outbreaks before 2014

We all have vivid memory of the last Ebola outbreak.
By looking at the 33 previous events since 1976, what can we learn to understand this event and formulate ideas to prevent future outbreaks?

Export as PNG View Source Open in Vega Editor

Export as PNG View Source Open in Vega Editor

Export as PNG View Source Open in Vega Editor

Tips

- Organise your repository: input datasets /notebooks / helpers / images or html outputs
- Watch out for the next release supporting Vega-Lite 2.0

Pros & Cons

- **Pros**: simple, work cooperatively (Github), reuse your code and html templates
- **Cons**: no handling of geo data, depending on Github
- **Improvements**: interactivity (next release)

🙏🏾

Thank you

🙋🏾‍♀️

@EleonoreMayola

github.com/Eleonore9/pyconse17

Resources

Docs:
https://altair-viz.github.io/
https://altair-viz.github.io/documentation/displaying.html
https://vega.github.io/vega-lite/usage/embed.html
https://vega.github.io/vega-lite/examples/#interactive
https://vega.github.io/vega-lite/usage/applications.html

Data source:
https://data.humdata.org/

Blogposts:
https://blog.modeanalytics.com/python-data-visualization-libraries/
http://pbpython.com/altair-intro.html

Emojis: emojipedia.org