altair package exploration

January 8, 2021

[2]:

```
# Explore Altair package and hopefully produce some interactive charts to play␣
 ↪with.

!pip install altair
!pip install vega
!pip install vega_datasets
!pip install altair_viewer

import pandas as pd
import numpy as np
import plotly as plt

import datascience
from datascience import *

import altair as alt
from vega_datasets import data
import vega
```

```
Requirement already satisfied: altair in /opt/conda/lib/python3.8/site-packages
(4.1.0)
Requirement already satisfied: jsonschema in /opt/conda/lib/python3.8/site-
packages (from altair) (3.2.0)
Requirement already satisfied: pandas>=0.18 in /opt/conda/lib/python3.8/site-
packages (from altair) (1.1.0)
Requirement already satisfied: numpy in /opt/conda/lib/python3.8/site-packages
(from altair) (1.18.5)
Requirement already satisfied: jinja2 in /opt/conda/lib/python3.8/site-packages
(from altair) (2.11.2)
Requirement already satisfied: toolz in /opt/conda/lib/python3.8/site-packages
(from altair) (0.11.1)
Requirement already satisfied: entrypoints in /opt/conda/lib/python3.8/site-
packages (from altair) (0.3)
Requirement already satisfied: pyrsistent>=0.14.0 in
/opt/conda/lib/python3.8/site-packages (from jsonschema->altair) (0.17.3)
Requirement already satisfied: setuptools in /opt/conda/lib/python3.8/site-
packages (from jsonschema->altair) (49.6.0.post20201009)
Requirement already satisfied: six>=1.11.0 in /opt/conda/lib/python3.8/site-
```

```
packages (from jsonschema->altair) (1.15.0)
Requirement already satisfied: attrs>=17.4.0 in /opt/conda/lib/python3.8/site-
packages (from jsonschema->altair) (19.3.0)
Requirement already satisfied: pytz>=2017.2 in /opt/conda/lib/python3.8/site-
packages (from pandas>=0.18->altair) (2020.5)
Requirement already satisfied: python-dateutil>=2.7.3 in
/opt/conda/lib/python3.8/site-packages (from pandas>=0.18->altair) (2.8.1)
Requirement already satisfied: MarkupSafe>=0.23 in
/opt/conda/lib/python3.8/site-packages (from jinja2->altair) (1.1.1)
Requirement already satisfied: vega in /opt/conda/lib/python3.8/site-packages
(3.4.0)
Requirement already satisfied: jupyter<2.0.0,>=1.0.0 in
/opt/conda/lib/python3.8/site-packages (from vega) (1.0.0)
Requirement already satisfied: pandas<2.0.0,>=1.0.0 in
/opt/conda/lib/python3.8/site-packages (from vega) (1.1.0)
Requirement already satisfied: notebook in /opt/conda/lib/python3.8/site-
packages (from jupyter<2.0.0,>=1.0.0->vega) (7.0.0.dev0)
Requirement already satisfied: ipywidgets in /opt/conda/lib/python3.8/site-
packages (from jupyter<2.0.0,>=1.0.0->vega) (7.5.1)
Requirement already satisfied: jupyter-console in /opt/conda/lib/python3.8/site-
packages (from jupyter<2.0.0,>=1.0.0->vega) (6.2.0)
Requirement already satisfied: nbconvert in /opt/conda/lib/python3.8/site-
packages (from jupyter<2.0.0,>=1.0.0->vega) (5.6.1)
Requirement already satisfied: ipykernel in /opt/conda/lib/python3.8/site-
packages (from jupyter<2.0.0,>=1.0.0->vega) (5.4.2)
Requirement already satisfied: qtconsole in /opt/conda/lib/python3.8/site-
packages (from jupyter<2.0.0,>=1.0.0->vega) (5.0.1)
Requirement already satisfied: python-dateutil>=2.7.3 in
/opt/conda/lib/python3.8/site-packages (from pandas<2.0.0,>=1.0.0->vega) (2.8.1)
Requirement already satisfied: numpy>=1.15.4 in /opt/conda/lib/python3.8/site-
packages (from pandas<2.0.0,>=1.0.0->vega) (1.18.5)
Requirement already satisfied: pytz>=2017.2 in /opt/conda/lib/python3.8/site-
packages (from pandas<2.0.0,>=1.0.0->vega) (2020.5)
Requirement already satisfied: ipython-genutils in
/opt/conda/lib/python3.8/site-packages (from
notebook->jupyter<2.0.0,>=1.0.0->vega) (0.2.0)
Requirement already satisfied: tornado>=5.0 in /opt/conda/lib/python3.8/site-
packages (from notebook->jupyter<2.0.0,>=1.0.0->vega) (6.1)
Requirement already satisfied: Send2Trash in /opt/conda/lib/python3.8/site-
packages (from notebook->jupyter<2.0.0,>=1.0.0->vega) (1.5.0)
Requirement already satisfied: jupyter-client>=5.3.4 in
/opt/conda/lib/python3.8/site-packages (from
notebook->jupyter<2.0.0,>=1.0.0->vega) (6.1.7)
Requirement already satisfied: terminado>=0.8.3 in
/opt/conda/lib/python3.8/site-packages (from
notebook->jupyter<2.0.0,>=1.0.0->vega) (0.9.1)
Requirement already satisfied: prometheus-client in
/opt/conda/lib/python3.8/site-packages (from
notebook->jupyter<2.0.0,>=1.0.0->vega) (0.9.0)
Requirement already satisfied: pyzmq>=17 in /opt/conda/lib/python3.8/site-
packages (from notebook->jupyter<2.0.0,>=1.0.0->vega) (20.0.0)
Requirement already satisfied: traitlets>=4.2.1 in
/opt/conda/lib/python3.8/site-packages (from
notebook->jupyter<2.0.0,>=1.0.0->vega) (5.0.5)
Requirement already satisfied: nbformat in /opt/conda/lib/python3.8/site-
packages (from notebook->jupyter<2.0.0,>=1.0.0->vega) (5.0.7)
Requirement already satisfied: argon2-cffi in /opt/conda/lib/python3.8/site-
packages (from notebook->jupyter<2.0.0,>=1.0.0->vega) (20.1.0)
Requirement already satisfied: jinja2 in /opt/conda/lib/python3.8/site-packages
(from notebook->jupyter<2.0.0,>=1.0.0->vega) (2.11.2)
Requirement already satisfied: jupyter-core>=4.6.1 in
/opt/conda/lib/python3.8/site-packages (from
notebook->jupyter<2.0.0,>=1.0.0->vega) (4.7.0)
Requirement already satisfied: widgetsnbextension~=3.5.0 in
/opt/conda/lib/python3.8/site-packages (from
ipywidgets->jupyter<2.0.0,>=1.0.0->vega) (3.5.1)
Requirement already satisfied: ipython>=4.0.0; python_version >= "3.3" in
/opt/conda/lib/python3.8/site-packages (from
ipywidgets->jupyter<2.0.0,>=1.0.0->vega) (7.19.0)
Requirement already satisfied: pygments in /opt/conda/lib/python3.8/site-
packages (from jupyter-console->jupyter<2.0.0,>=1.0.0->vega) (2.7.3)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in
/opt/conda/lib/python3.8/site-packages (from jupyter-
console->jupyter<2.0.0,>=1.0.0->vega) (3.0.8)
Requirement already satisfied: testpath in /opt/conda/lib/python3.8/site-
packages (from nbconvert->jupyter<2.0.0,>=1.0.0->vega) (0.4.4)
Requirement already satisfied: defusedxml in /opt/conda/lib/python3.8/site-
packages (from nbconvert->jupyter<2.0.0,>=1.0.0->vega) (0.6.0)
Requirement already satisfied: bleach in /opt/conda/lib/python3.8/site-packages
(from nbconvert->jupyter<2.0.0,>=1.0.0->vega) (3.2.1)
Requirement already satisfied: entrypoints>=0.2.2 in
/opt/conda/lib/python3.8/site-packages (from
nbconvert->jupyter<2.0.0,>=1.0.0->vega) (0.3)
Requirement already satisfied: pandocfilters>=1.4.1 in
/opt/conda/lib/python3.8/site-packages (from
nbconvert->jupyter<2.0.0,>=1.0.0->vega) (1.4.3)
Requirement already satisfied: mistune<2,>=0.8.1 in
/opt/conda/lib/python3.8/site-packages (from
nbconvert->jupyter<2.0.0,>=1.0.0->vega) (0.8.4)
Requirement already satisfied: qtpy in /opt/conda/lib/python3.8/site-packages
(from qtconsole->jupyter<2.0.0,>=1.0.0->vega) (1.9.0)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.8/site-
packages (from python-dateutil>=2.7.3->pandas<2.0.0,>=1.0.0->vega) (1.15.0)
Requirement already satisfied: ptyprocess; os_name != "nt" in
/opt/conda/lib/python3.8/site-packages (from
terminado>=0.8.3->notebook->jupyter<2.0.0,>=1.0.0->vega) (0.7.0)
```

```
Requirement already satisfied: jsonschema!=2.5.0,>=2.4 in
/opt/conda/lib/python3.8/site-packages (from
nbformat->notebook->jupyter<2.0.0,>=1.0.0->vega) (3.2.0)
Requirement already satisfied: cffi>=1.0.0 in /opt/conda/lib/python3.8/site-
packages (from argon2-cffi->notebook->jupyter<2.0.0,>=1.0.0->vega) (1.14.4)
Requirement already satisfied: MarkupSafe>=0.23 in
/opt/conda/lib/python3.8/site-packages (from
jinja2->notebook->jupyter<2.0.0,>=1.0.0->vega) (1.1.1)
Requirement already satisfied: decorator in /opt/conda/lib/python3.8/site-
packages (from ipython>=4.0.0; python_version >=
"3.3"->ipywidgets->jupyter<2.0.0,>=1.0.0->vega) (4.4.2)
Requirement already satisfied: pexpect>4.3; sys_platform != "win32" in
/opt/conda/lib/python3.8/site-packages (from ipython>=4.0.0; python_version >=
"3.3"->ipywidgets->jupyter<2.0.0,>=1.0.0->vega) (4.8.0)
Requirement already satisfied: backcall in /opt/conda/lib/python3.8/site-
packages (from ipython>=4.0.0; python_version >=
"3.3"->ipywidgets->jupyter<2.0.0,>=1.0.0->vega) (0.2.0)
Requirement already satisfied: pickleshare in /opt/conda/lib/python3.8/site-
packages (from ipython>=4.0.0; python_version >=
"3.3"->ipywidgets->jupyter<2.0.0,>=1.0.0->vega) (0.7.5)
Requirement already satisfied: setuptools>=18.5 in
/opt/conda/lib/python3.8/site-packages (from ipython>=4.0.0; python_version >=
"3.3"->ipywidgets->jupyter<2.0.0,>=1.0.0->vega) (49.6.0.post20201009)
Requirement already satisfied: jedi>=0.10 in /opt/conda/lib/python3.8/site-
packages (from ipython>=4.0.0; python_version >=
"3.3"->ipywidgets->jupyter<2.0.0,>=1.0.0->vega) (0.18.0)
Requirement already satisfied: wcwidth in /opt/conda/lib/python3.8/site-packages
(from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->jupyter-
console->jupyter<2.0.0,>=1.0.0->vega) (0.2.5)
Requirement already satisfied: webencodings in /opt/conda/lib/python3.8/site-
packages (from bleach->nbconvert->jupyter<2.0.0,>=1.0.0->vega) (0.5.1)
Requirement already satisfied: packaging in /opt/conda/lib/python3.8/site-
packages (from bleach->nbconvert->jupyter<2.0.0,>=1.0.0->vega) (20.8)
Requirement already satisfied: pyrsistent>=0.14.0 in
/opt/conda/lib/python3.8/site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat->notebook->jupyter<2.0.0,>=1.0.0->vega)
(0.17.3)
Requirement already satisfied: attrs>=17.4.0 in /opt/conda/lib/python3.8/site-
packages (from
jsonschema!=2.5.0,>=2.4->nbformat->notebook->jupyter<2.0.0,>=1.0.0->vega)
(19.3.0)
Requirement already satisfied: pycparser in /opt/conda/lib/python3.8/site-
packages (from cffi>=1.0.0->argon2-cffi->notebook->jupyter<2.0.0,>=1.0.0->vega)
(2.20)
Requirement already satisfied: parso<0.9.0,>=0.8.0 in
/opt/conda/lib/python3.8/site-packages (from jedi>=0.10->ipython>=4.0.0;
python_version >= "3.3"->ipywidgets->jupyter<2.0.0,>=1.0.0->vega) (0.8.1)
Requirement already satisfied: pyparsing>=2.0.2 in
```

```
/opt/conda/lib/python3.8/site-packages (from
packaging->bleach->nbconvert->jupyter<2.0.0,>=1.0.0->vega) (2.4.7)
Requirement already satisfied: vega_datasets in /opt/conda/lib/python3.8/site-
packages (0.9.0)
Requirement already satisfied: pandas in /opt/conda/lib/python3.8/site-packages
(from vega_datasets) (1.1.0)
Requirement already satisfied: python-dateutil>=2.7.3 in
/opt/conda/lib/python3.8/site-packages (from pandas->vega_datasets) (2.8.1)
Requirement already satisfied: numpy>=1.15.4 in /opt/conda/lib/python3.8/site-
packages (from pandas->vega_datasets) (1.18.5)
Requirement already satisfied: pytz>=2017.2 in /opt/conda/lib/python3.8/site-
packages (from pandas->vega_datasets) (2020.5)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.8/site-
packages (from python-dateutil>=2.7.3->pandas->vega_datasets) (1.15.0)
Requirement already satisfied: altair_viewer in /opt/conda/lib/python3.8/site-
packages (0.3.0)
Requirement already satisfied: altair-data-server>=0.4.0 in
/opt/conda/lib/python3.8/site-packages (from altair_viewer) (0.4.1)
Requirement already satisfied: altair in /opt/conda/lib/python3.8/site-packages
(from altair_viewer) (4.1.0)
Requirement already satisfied: portpicker in /opt/conda/lib/python3.8/site-
packages (from altair-data-server>=0.4.0->altair_viewer) (1.3.1)
Requirement already satisfied: tornado in /opt/conda/lib/python3.8/site-packages
(from altair-data-server>=0.4.0->altair_viewer) (6.1)
Requirement already satisfied: jsonschema in /opt/conda/lib/python3.8/site-
packages (from altair->altair_viewer) (3.2.0)
Requirement already satisfied: entrypoints in /opt/conda/lib/python3.8/site-
packages (from altair->altair_viewer) (0.3)
Requirement already satisfied: toolz in /opt/conda/lib/python3.8/site-packages
(from altair->altair_viewer) (0.11.1)
Requirement already satisfied: jinja2 in /opt/conda/lib/python3.8/site-packages
(from altair->altair_viewer) (2.11.2)
Requirement already satisfied: pandas>=0.18 in /opt/conda/lib/python3.8/site-
packages (from altair->altair_viewer) (1.1.0)
Requirement already satisfied: numpy in /opt/conda/lib/python3.8/site-packages
(from altair->altair_viewer) (1.18.5)
Requirement already satisfied: attrs>=17.4.0 in /opt/conda/lib/python3.8/site-
packages (from jsonschema->altair->altair_viewer) (19.3.0)
Requirement already satisfied: setuptools in /opt/conda/lib/python3.8/site-
packages (from jsonschema->altair->altair_viewer) (49.6.0.post20201009)
Requirement already satisfied: pyrsistent>=0.14.0 in
/opt/conda/lib/python3.8/site-packages (from jsonschema->altair->altair_viewer)
(0.17.3)
Requirement already satisfied: six>=1.11.0 in /opt/conda/lib/python3.8/site-
packages (from jsonschema->altair->altair_viewer) (1.15.0)
Requirement already satisfied: MarkupSafe>=0.23 in
/opt/conda/lib/python3.8/site-packages (from jinja2->altair->altair_viewer)
(1.1.1)
```

```
Requirement already satisfied: python-dateutil>=2.7.3 in
/opt/conda/lib/python3.8/site-packages (from
pandas>=0.18->altair->altair_viewer) (2.8.1)
Requirement already satisfied: pytz>=2017.2 in /opt/conda/lib/python3.8/site-
packages (from pandas>=0.18->altair->altair_viewer) (2020.5)
```

[14]:

```
from vega_datasets import data
```

[3]:

```
data = pd.DataFrame({'a': list('CCCDDDEEE'),
                     'b': [2, 7, 4, 1, 2, 6, 8, 4, 7]})
data
```

[3]:

```
   a  b
0  C  2
1  C  7
2  C  4
3  D  1
4  D  2
5  D  6
6  E  8
7  E  4
8  E  7
```

[4]:

```
chart = alt.Chart(data)
```

[5]:

```
alt.Chart(data).mark_point()
```

[5]: alt.Chart(…)

[6]:

```
chart.mark_point().encode(
    x = 'a',
    y = 'average(b)'
)
```

[6]: alt.Chart(…)

[7]:

```
chart.mark_bar(color='firebrick').encode(
x = 'average(b)',
y = 'a')
```

[7]: alt.Chart(…)

[8]:

```
y = alt.Y('average(b):Q')
print(y.to_json())
```

```
{
  "aggregate": "average",
  "field": "b",
  "type": "quantitative"
}
```

```
[9]: y = alt.Y(field='b', type='quantitative', aggregate='average')
     print(y.to_json())
```

```
{
  "aggregate": "average",
  "field": "b",
  "type": "quantitative"
}
```

```
[11]: red_chart = chart.mark_bar(color = 'firebrick').encode(
      alt.Y('a', title= 'category'),
      alt.X('average(b)', title='avg(b) by category'))

      red_chart
```

```
[11]: alt.Chart(…)
```

```
[10]: red_chart.to_json()
      red_chart.save('red_chart.html')
```

```
[15]: cars = data.cars()
      cars
```

```
[15]:                       Name  Miles_per_Gallon  Cylinders  Displacement  \
      0    chevrolet chevelle malibu              18.0          8         307.0
      1            buick skylark 320              15.0          8         350.0
      2           plymouth satellite              18.0          8         318.0
      3                amc rebel sst              16.0          8         304.0
      4                  ford torino              17.0          8         302.0
      ..                         …                …        …           …
      401            ford mustang gl              27.0          4         140.0
      402                  vw pickup              44.0          4          97.0
      403              dodge rampage              32.0          4         135.0
      404                ford ranger              28.0          4         120.0
      405                 chevy s-10              31.0          4         119.0

           Horsepower  Weight_in_lbs  Acceleration       Year  Origin
      0         130.0           3504          12.0 1970-01-01     USA
      1         165.0           3693          11.5 1970-01-01     USA
      2         150.0           3436          11.0 1970-01-01     USA
      3         150.0           3433          12.0 1970-01-01     USA
      4         140.0           3449          10.5 1970-01-01     USA
      ..          …            …           …        …     …
      401        86.0           2790          15.6 1982-01-01     USA
```

```
402         52.0            2130            24.6 1982-01-01  Europe
403         84.0            2295            11.6 1982-01-01     USA
404         79.0            2625            18.6 1982-01-01     USA
405         82.0            2720            19.4 1982-01-01     USA

[406 rows x 9 columns]
```

[16]:

```
# Sample - cars dataset

alt.Chart(cars).mark_point().encode(
    x = 'Miles_per_Gallon',
    y = 'Horsepower',
    color = 'Origin'
).interactive()
```

[16]: alt.Chart(...)

[17]:

```
# Place data into bins. Curious as to why count() is a string, compared to alt.
 ↪X()...

alt.Chart(cars).mark_bar().encode(
    x = alt.X('Miles_per_Gallon', bin=True),
    y = 'count()',
    color = 'Origin'
).interactive()
```

[17]: alt.Chart(...)

[18]:

```
# No 'heat_map', graph by bins

alt.Chart(cars).mark_bar().encode(
    x = alt.X('Miles_per_Gallon', bin=True),
    y = alt.Y('Horsepower', bin=True),
    color = 'count()'
).interactive()
```

[18]: alt.Chart(...)

[19]:

```
# Include intervals: click and drag to create a box where data points included␣
 ↪are in color; points excluded are gray

interval = alt.selection_interval()

chart = alt.Chart(cars).mark_point().encode(
    x = 'Miles_per_Gallon',
    y = 'Horsepower',
    color = alt.condition(interval,'Origin', alt.value('lightgray')),
    tooltip = 'Name' # Add hover for Names
).properties(
    selection=interval
)

# create two charts that are linked, swapping the x-axis for Acceleration to␣
 ↪see how the data is related

chart | chart.encode(x='Acceleration')
```

[19]: alt.HConcatChart(…)

[20]:
```
# Click & drag to create a box on the scatterplot to filter the barchart

interval = alt.selection_interval()

chart = alt.Chart(cars).mark_point().encode(
    x = 'Miles_per_Gallon',
    y = 'Horsepower',
    color = alt.condition(interval,'Origin', alt.value('lightgray')),
    tooltip = 'Name' # Add hover for Names
).properties(selection=interval)

hist = alt.Chart(cars).mark_bar().encode(
    x = 'count()',
    y = 'Origin',
    color = 'Origin'
).transform_filter(interval)

chart & hist
```

[20]: alt.VConcatChart(…)

[21]:
```
# CLick on the bar chart to filter the data

click = alt.selection_multi(encodings = ['color'])

hist = alt.Chart(cars).mark_bar().encode(
    x = 'count()',
    y = 'Origin:N',
    color = alt.condition(click, 'Origin', alt.value('lightgray'))
).properties(selection = click)

scatter = alt.Chart(cars).mark_point().encode(
    x = 'Horsepower:Q',
    y = 'Miles_per_Gallon:Q',
    color = 'Origin:N'
).transform_filter(click).interactive()

hist & scatter
```

```
[21]: alt.VConcatChart(...)
```

```
[22]: # CLick on the legend to filter the data

      # Emily - focus on recreating stuff like this using MTF data, add in a dropdown␣
       ↪menu and a couple variables.
      # focus on aesthetic pieces - i.e. fonts, labels, etc.

      click = alt.selection_multi(encodings = ['color'])

      hist = alt.Chart(cars).mark_point().encode(
          y = 'Origin',
          color = alt.condition(click, 'Origin', alt.value('lightgray'), legend =␣
       ↪None)
      ).properties(selection = click)

      scatter = alt.Chart(cars).mark_point().encode(
          x = 'Horsepower:Q',
          y = 'Miles_per_Gallon:Q',
          color = 'Origin:N'
      ).transform_filter(click).interactive()

      scatter | hist
```

```
[22]: alt.HConcatChart(...)
```

```
[23]: weather = data.seattle_weather()
      weather.head()
```

```
[23]:          date  precipitation  temp_max  temp_min  wind  weather
      0 2012-01-01            0.0      12.8       5.0   4.7  drizzle
      1 2012-01-02           10.9      10.6       2.8   4.5     rain
      2 2012-01-03            0.8      11.7       7.2   2.3     rain
      3 2012-01-04           20.3      12.2       5.6   4.7     rain
      4 2012-01-05            1.3       8.9       2.8   6.1     rain
```

```
[24]: # Interactive weather graph

      interval = alt.selection_interval(encodings = ['x'])

      base = alt.Chart(weather).mark_rule(size = 2).encode(
          x = 'date:T',
          y = 'temp_min:Q',
          y2 = 'temp_max:Q',
          color = 'weather:N'
      )

      chart = base.properties(
          width = 800,
          height = 300).encode(
          x = alt.X('date:T', scale = alt.Scale(domain=interval.ref()))
      )

      view = chart.properties(
          width = 800,
          height = 50,
          selection = interval # Add interval
      ).interactive()

      chart & view
```

[24]: alt.VConcatChart(…)

[ ]:

```
```

[ ]:

```
```

[ ]:

```
```

[25]:

```
# Sample pulled from documentation

movies = alt.UrlData(
    data.movies.url,
    format=alt.DataFormat(parse={"Release_Date":"date"})
)
ratings = ['G', 'NC-17', 'PG', 'PG-13', 'R']
genres = ['Action', 'Adventure', 'Black Comedy', 'Comedy',
       'Concert/Performance', 'Documentary', 'Drama', 'Horror', 'Musical',
       'Romantic Comedy', 'Thriller/Suspense', 'Western']

base = alt.Chart(movies, width=200, height=200).mark_point(filled=True).
 ↪transform_calculate(
    Rounded_IMDB_Rating = "floor(datum.IMDB_Rating)",
    Hundred_Million_Production =  "datum.Production_Budget > 100000000.0 ? 100 :
 ↪ 10",
    Release_Year = "year(datum.Release_Date)"
).transform_filter(
    alt.datum.IMDB_Rating > 0
).transform_filter(
    alt.FieldOneOfPredicate(field='MPAA_Rating', oneOf=ratings)
).encode(
    x=alt.X('Worldwide_Gross:Q', scale=alt.Scale(domain=(100000,10**9),␣
 ↪clamp=True)),
    y='IMDB_Rating:Q',
    tooltip="Title:N"
)

# A slider filter
year_slider = alt.binding_range(min=1969, max=2018, step=1)
slider_selection = alt.selection_single(bind=year_slider,␣
 ↪fields=['Release_Year'], name="Release Year_")


filter_year = base.add_selection(
    slider_selection
).transform_filter(
    slider_selection
).properties(title="Slider Filtering")

# A dropdown filter
genre_dropdown = alt.binding_select(options=genres)
genre_select = alt.selection_single(fields=['Major_Genre'],␣
 ↪bind=genre_dropdown, name="Genre")

filter_genres = base.add_selection(
    genre_select
).transform_filter(
    genre_select
).properties(title="Dropdown Filtering")

#color changing marks
rating_radio = alt.binding_radio(options=ratings)

rating_select = alt.selection_single(fields=['MPAA_Rating'], bind=rating_radio,␣
 ↪name="Rating")
rating_color_condition = alt.condition(rating_select,
                      alt.Color('MPAA_Rating:N', legend=None),
                      alt.value('lightgray'))

highlight_ratings = base.add_selection(
    rating_select
).encode(
    color=rating_color_condition
).properties(title="Radio Button Highlighting")

# Boolean selection for format changes
input_checkbox = alt.binding_checkbox()
```

```
checkbox_selection = alt.selection_single(bind=input_checkbox, name="Big Budget␣
 ↪Films")

size_checkbox_condition = alt.condition(checkbox_selection,
                                        alt.SizeValue(25),
                                        alt.Size('Hundred_Million_Production:Q')
                                       )

budget_sizing = base.add_selection(
    checkbox_selection
).encode(
    size=size_checkbox_condition
).properties(title="Checkbox Formatting")

( filter_year | filter_genres) &  (highlight_ratings | budget_sizing  )
```

[25]: alt.VConcatChart(…)

[ ]:

```
click = alt.selection_multi(encodings = ['color'])

tiers = alt.Chart(kenya_mtf2_mini).mark_bar().encode(
    x = alt.X('elc_aggr_tier', axis = alt.Axis(title = "Tier")),
    y = 'count(elc_aggr_tier)',
    color = 'c_c_159',
    tooltip = 'elc_aggr_tier'
).properties(width = 400, selection = click).interactive()

lighting_source = alt.Chart(kenya_mtf2_mini).mark_bar(color='green').encode(
    x = alt.X('c_c_159', axis = alt.Axis(title='Of all the electricity sources␣
 ↪you mentioned above, which is the source that you use most of the time in␣
 ↪the household?')),
    y = 'count(c_c_159)',
    tooltip = 'c_c_159'
).properties(width = 400, selection = click).transform_filter(click).
 ↪interactive()

tiers & lighting_source
```

[ ]:

```
```

[ ]:

```
```

[ ]:

```
```

[3]:

```
# Load Kenya MTF Data

chunksize = 10
```

```
mtf_list = []
chunksize = 10
for chunk in pd.read_csv("kenya_mtf2_mini.csv", encoding='latin-1', chunksize =␣
 ↪chunksize):
    mtf_list.append(chunk)

kenya_mtf2_mini = pd.concat(mtf_list, axis=0)
kenya_mtf2_mini = pd.DataFrame(data = kenya_mtf2_mini)
```

[4]:

```
kenya_mtf2_mini.head()
```

[4]:

```
                                  parent_key elc_aggr_tier locality_ur  \
0  uuid:0006ae15-e9cf-419e-ac14-0c66a739366e        Tier 0       Urban
1  uuid:001b24c5-30b9-41fe-ac90-9e9e3e476935        Tier 5       Urban
2  uuid:0031732a-2efc-43e7-ba34-9447ddc31b32        Tier 2       Rural
3  uuid:0042ae86-e063-4b44-a859-9c253d82bd38        Tier 0       Rural
4  uuid:006014f9-c69e-48b2-8198-60cbbc894b59        Tier 0       Rural

   c_c_25bii  c_c_27b         c_c_30             c_c_31 c_c_119  c_c_123  \
0        NaN      NaN            NaN                NaN     NaN      NaN
1       24.0     18.0  Kerosene lamp  No back-up source     NaN      NaN
2        NaN      NaN            NaN                NaN     NaN      0.0
3        NaN      NaN            NaN                NaN     NaN      NaN
4        NaN      NaN            NaN                NaN     NaN      NaN

   c_c_149b_typicalmonth  … g_g_radio_4 g_g_radio_5  g_g_radio_6  \
0                    NaN  …         NaN         NaN          NaN
1                    NaN  …         NaN         NaN          NaN
2                   24.0  …         NaN         NaN          NaN
3                    NaN  …         NaN         NaN          NaN
4                    NaN  …         NaN         NaN          NaN

  g_g_radio_7 g_g_3_other_other_r_count g_g_3_othr_other_r g_g_9 g_g_10  \
0         NaN                       NaN                NaN   NaN    NaN
1         NaN                       NaN                NaN   NaN    NaN
2         NaN                       NaN                NaN   NaN    NaN
3         NaN                       NaN                NaN   NaN    NaN
4         NaN                       NaN                NaN   NaN    NaN

  hh_grid solar
0      No    No
1     Yes    No
2      No   Yes
3      No    No
4      No    No
```

```
[5 rows x 120 columns]
```

[7]:

```
click = alt.selection_multi(encodings = ['color'])

tiers = alt.Chart(kenya_mtf2_mini).mark_bar().encode(
    x = alt.X('elc_aggr_tier', axis = alt.Axis(title = "Tier")),
    y = 'count(elc_aggr_tier)',
    #scale = alt.Scale(domain = [0, 2100])),
    color = 'c_c_159',
    tooltip = 'c_c_159'
).transform_filter(click).properties(width = 600, height = 600).interactive()

hours_slider = alt.binding_range(min=0, max=24, step=1)
slider_selection = alt.selection_single(bind=hours_slider,
                                        fields=['c_c_27b'],
                                        name="Hours of Electricity Used per␣
 ↪Day")
filter_hours = tiers.add_selection(
    slider_selection
).transform_filter(
    slider_selection
).properties(title="Tier/Locality Count & Breakdown of Electricity Sources")

tier_scatter = alt.Chart(kenya_mtf2_mini).mark_point().encode(
    y = alt.Y('locality_ur', axis = alt.Axis(title = "")),
    color = alt.condition(click, 'locality_ur',
                          alt.value('lightgray'),
                          #alt.Color('locality_ur',
                          #          scale = alt.Scale(
                          #          domain = ['Urban', 'Rural'],
                          #          range = ['blue','green'])),
                          legend = None)
).properties(selection = click)

filter_hours | tier_scatter
```

[7]:

```
alt.HConcatChart(…)
```

[33]:

```
# Heat map for locality / sources of electricity

locality = alt.Chart(kenya_mtf2_mini).mark_bar().encode(
    y = alt.Y('locality_ur', axis = alt.Axis(title = "National Locality")),
    x = 'c_c_159',
    color = 'count()'
).properties(width = 400, height = 200)

locality
```

```
[33]: alt.Chart(…)
```

```
[34]: # Emily's comments

      # As of December 2020, you can't make any pie charts in Altair, which is a bit␣
       ↪concerning:
      # https://github.com/altair-viz/altair/issues/2148

      # However, we can use matplotlib, plotly or dash as a workaround.

      # I'm not sure if there are any things Altair can do that Tableau can't. To␣
       ↪create these charts in Altair, I'm working in
      # Jupyter Notebooks. As such, there is a memory limit on the notebook (1GB). If␣
       ↪I were to try and load all 778 columns and
      # 4,590 rows of data from the MTF, the notebook crashes. The charts made above␣
       ↪are used with select columns (120 in total)
      # from the MTF. The smaller dataframe is workable in the notebook. I chose␣
       ↪those columns based on the AIP Module Mock-Ups.
      # However, I am worried that users would not be able to fully explore the␣
       ↪breadth of the MTF (and the other variables) due
      # to the memory issues of the notebook.

      # From what I understand based on the Jam Board and team meetings, we want a␣
       ↪tool that allows the user to select variables and
      # the type of visualization they'd like to see in a single screen -- not a␣
       ↪long, scrolling website. As I think more about the
      # AIP project, it seems like we are trying to create something like Tableau␣
       ↪from scratch using MTF Data? I will have to explore
      # Altair more, but I am unsure if the package can do something like that; based␣
       ↪on the documentation available, most of the
      # visualizations are pre-made charts in which the user can interact with, but␣
       ↪does not provide the "pick your own visualization"
      # functionality...
```