code

February 23, 2025

[1]:

```
import pandas as pd
import numpy as np
import altair as alt

alt.themes.enable('fivethirtyeight')
```

[1]:

```
ThemeRegistry.enable('fivethirtyeight')
```

0.1 Graph 1

[2]:

```
private = pd.read_csv('NCES_private_clean.csv', index_col=0)
```

[3]:

```
private_1 = private[['State Name [Private School] Latest available␣
 ↪year','Percentage of Black Students [Private School] 2015-16','Percentage of␣
 ↪White Students [Private School] 2015-16']]
private_1 = private_1.dropna(subset=["State Name [Private School] Latest␣
 ↪available year",
                         "Percentage of Black Students [Private School] 2015-16",
                         "Percentage of White Students [Private School] 2015-16"])
private_1["Percentage of Black Students [Private School] 2015-16"] = pd.
 ↪to_numeric(private_1["Percentage of Black Students [Private School]␣
 ↪2015-16"], errors="coerce")
private_1["Percentage of White Students [Private School] 2015-16"] = pd.
 ↪to_numeric(private_1["Percentage of White Students [Private School]␣
 ↪2015-16"], errors="coerce")
private_1["Highly Segregated"] = (private_1["Percentage of Black Students␣
 ↪[Private School] 2015-16"] >= 80) | (private_1["Percentage of White Students␣
 ↪[Private School] 2015-16"] >= 80)
df_state = private_1.groupby("State Name [Private School] Latest available␣
 ↪year").agg(
    total_schools=("Highly Segregated", "count"),
    segregated_schools=("Highly Segregated", "sum")
).reset_index()
df_state["segregated_percentage"] = (df_state["segregated_schools"] /␣
 ↪df_state["total_schools"]) * 100
```

[4]:

```python
from vega_datasets import data

data_df = df_state.rename(columns={
    "State Name [Private School] Latest available year": "state",
    "segregated_percentage": "percentage"
})

state_id_map = {
    "ALABAMA": "01",
    "ALASKA": "02",
    "ARIZONA": "04",
    "ARKANSAS": "05",
    "CALIFORNIA": "06",
    "COLORADO": "08",
    "CONNECTICUT": "09",
    "DELAWARE": "10",
    "DISTRICT OF COLUMBIA": "11",
    "FLORIDA": "12",
    "GEORGIA": "13",
    "HAWAII": "15",
    "IDAHO": "16",
    "ILLINOIS": "17",
    "INDIANA": "18",
    "IOWA": "19",
    "KANSAS": "20",
    "KENTUCKY": "21",
    "LOUISIANA": "22",
    "MAINE": "23",
    "MARYLAND": "24",
    "MASSACHUSETTS": "25",
    "MICHIGAN": "26",
    "MINNESOTA": "27",
    "MISSISSIPPI": "28",
    "MISSOURI": "29",
    "MONTANA": "30",
    "NEBRASKA": "31",
    "NEVADA": "32",
    "NEW HAMPSHIRE": "33",
    "NEW JERSEY": "34",
    "NEW MEXICO": "35",
    "NEW YORK": "36",
    "NORTH CAROLINA": "37",
    "NORTH DAKOTA": "38",
    "OHIO": "39",
    "OKLAHOMA": "40",
    "OREGON": "41",
    "PENNSYLVANIA": "42",
    "RHODE ISLAND": "44",
    "SOUTH CAROLINA": "45",
    "SOUTH DAKOTA": "46",
    "TENNESSEE": "47",
    "TEXAS": "48",
    "UTAH": "49",
    "VERMONT": "50",
    "VIRGINIA": "51",
    "WASHINGTON": "53",
    "WEST VIRGINIA": "54",
    "WISCONSIN": "55",
    "WYOMING": "56"
}

data_df["id"] = data_df["state"].map(state_id_map)
data_df["id"] = data_df["id"].astype(int)
us_states = alt.topo_feature(data.us_10m.url, feature='states')
```

[5]:

```
data_map = (
    alt.Chart(us_states)
    .mark_geoshape(
        stroke='white',
        strokeWidth=0.5
    )
    .transform_lookup(
        lookup='id',
        from_=alt.LookupData(
            data_df,
            key='id',
            fields=["state", "percentage"]
        )
    )
    .encode(
        color=alt.Color("percentage:Q", title="Percentage"),
        tooltip=[
            alt.Tooltip("state:N", title="State"),
            alt.Tooltip("percentage:Q", title="Segregated %", format=".2f")
        ]
    )
    .project(type='albersUsa')
    .properties(
        width=700,
        height=450,
        title="Segregated Percentage by State (Private Schools)"
    )
)
annot_df = pd.DataFrame({
    'state': ['NEW YORK'],
    'latitude': [43],
    'longitude': [-76.0],
    'label': ['NEW YORK']
})
label_layer = (
    alt.Chart(annot_df)
    .mark_text(
    )
    .encode(
        longitude='longitude:Q',
        latitude='latitude:Q',
        text='label:N'
    )
    .project(type='albersUsa')
)

final_chart_1 = data_map + label_layer
final_chart_1.save('p1.pdf')
final_chart_1.save('p1.png')
final_chart_1
```

```
[5]: alt.LayerChart(…)
```

This is a U.S map which shows the segregated extent of each states, the deeper the color is, the higher percentage of school with specific race is. The standard of a segregated school is whether the percentage of white of black students is over 80 percentage, and the data comes from `National-level data on all public and private schools(https://nces.ed.gov/datatools/)`. Also, New York State is specifically noted in the graph, corresponding to the author's discussion about education situation in New York State.

0.2 Graph2

```
[6]: public = pd.read_csv('NCES_public_clean.csv', index_col=0)
     public_2 = public[(public['School Name'] == 'PS 307 DANIEL HALE WILLIAMS' )␣
       ↪|(public['School Name'] == 'PS 8 ROBERT FULTON')]
     public_2 = public_2[['School Name','Total Students All Grades (Includes AE)␣
       ↪[Public School] 2015-16','Free and Reduced Lunch Students [Public School]␣
       ↪2015-16','Black or African American Students [Public School] 2015-16','White␣
       ↪Students [Public School] 2015-16','Full-Time Equivalent (FTE) Teachers␣
       ↪[Public School] 2015-16','Pupil/Teacher Ratio [Public School] 2015-16']]
```

```
[7]: public_2['Black or African American Students [Public School] 2015-16'] = pd.
       ↪to_numeric(public_2['Black or African American Students [Public School]␣
       ↪2015-16'], errors='coerce')
```

```
public_2['White Students [Public School] 2015-16'] = pd.
 ↪to_numeric(public_2['White Students [Public School] 2015-16'],␣
 ↪errors='coerce')
public_2['b/w'] = public_2['Black or African American Students [Public School]␣
 ↪2015-16'] / public_2['White Students [Public School] 2015-16']
```

[8]:

```
public_2 = public_2.rename(columns={
    "Free and Reduced Lunch Students [Public School] 2015-16": "Free and␣
 ↪Reduced Lunch Students",
    "Full-Time Equivalent (FTE) Teachers [Public School] 2015-16": "FTE␣
 ↪Teachers",
    "Pupil/Teacher Ratio [Public School] 2015-16": "Pupil/Teacher Ratio",
    "b/w": "Black/White Ratio"
})
public_2['Free and Reduced Lunch Students'] = pd.to_numeric(public_2['Free and␣
 ↪Reduced Lunch Students'], errors='coerce')
public_2['FTE Teachers'] = pd.to_numeric(public_2['FTE Teachers'],␣
 ↪errors='coerce')
public_2['Pupil/Teacher Ratio'] = pd.to_numeric(public_2['Pupil/Teacher␣
 ↪Ratio'], errors='coerce')
public_2['School Name'] = ['P.S 307', 'P.S 8']
```

[9]:

```
p2 = (alt.Chart(public_2).mark_bar(size=70).encode(
    x = alt.X('School Name', title=None, axis=alt.Axis(labelAngle=0)),
    y = alt.Y(alt.repeat('column'),type='quantitative', axis=alt.
 ↪Axis(titleAngle=0,titleX=60, titleY=-15)),
    color = 'School Name'
)+
alt.Chart(public_2)
.mark_text(
    align='center',
    baseline='bottom',
    dy=-2,
    color='black'
)
.encode(
    x=alt.X('School Name', title=None),
    y=alt.Y(alt.repeat('column'), type='quantitative'),
    text=alt.Text(
        alt.repeat('column'),
        format=".2f"
    )
)
).properties(width=150, height=300).properties(
    width=150,
    height=330,
).repeat(
    column = ['Free and Reduced Lunch Students','FTE Teachers','Pupil/Teacher␣
  ↪Ratio','Black/White Ratio'],
    columns= 4
).properties(
    title = 'Comparison of PS 307 and PS 8'
)
p2.save('p2.pdf')
p2.save('p2.png')
p2
```

```
[9]: alt.RepeatChart(…)
```

This graph compares the basic statistics of two public school 8 and 307 mentioned in the article. It shows the ratio of black and white students counts as well as students receive free or reduced students in the aspect of students situation, and represent education resources distribution by teacher counts and pupil-teacher ratio. The data comes from `National-level data on all public and private schools (https://nces.ed.gov/datatools/)`.

0.3 Graph3

```
[10]: math = pd.read_csv('ny-math-results-2013-2019-public-all.csv',index_col=0)
math = math[(math['School Name'] == 'P.S. 307 DANIEL HALE WILLIAMS' )␣
  ↪|(math['School Name'] == 'P.S. 008 ROBERT FULTON')]
math = math[math['Grade']=='All Grades']
math = math[['School Name','Year','Mean Scale Score','% Level 1','% Level 2','%␣
  ↪Level 3+4']]
math = math.rename(columns={
    'Mean Scale Score':'Mean Scale Score Math',
    '% Level 1':'L1 Math',
    '% Level 2':'L2 Math',
    '% Level 3+4':'L3,4 Math'
})
math['Year'] = pd.to_datetime(math['Year'], format='%Y')
math['Mean Scale Score Math'] = pd.to_numeric(math['Mean Scale Score Math'],␣
  ↪errors='coerce')
math['L1 Math'] = pd.to_numeric(math['L1 Math'], errors='coerce')
math['L2 Math'] = pd.to_numeric(math['L2 Math'], errors='coerce')
math['L3,4 Math'] = pd.to_numeric(math['L3,4 Math'], errors='coerce')
math = math[math['Year'].dt.year <2018]
math['School Name'] = ['P.S 307']*5 + ['P.S 8']*5

ela = pd.read_csv('ny-ela-results-2013-2019-public-all.csv',index_col=0)
ela = ela[(ela['School Name'] == 'P.S. 307 DANIEL HALE WILLIAMS' )␣
  ↪|(ela['School Name'] == 'P.S. 008 ROBERT FULTON')]
ela = ela[ela['Grade']=='All Grades']
ela = ela[['School Name','Year','Mean Scale Score','% Level 1','% Level 2','%␣
 ↪Level 3+4']]
ela = ela.rename(columns={
    'Mean Scale Score':'Mean Scale Score ela',
    '% Level 1':'L1 ela',
    '% Level 2':'L2 ela',
    '% Level 3+4':'L3,4 ela'
})
ela['Year'] = pd.to_datetime(ela['Year'], format='%Y')
ela['Mean Scale Score ela'] = pd.to_numeric(ela['Mean Scale Score ela'],␣
 ↪errors='coerce')
ela['L1 ela'] = pd.to_numeric(ela['L1 ela'], errors='coerce')
ela['L2 ela'] = pd.to_numeric(ela['L2 ela'], errors='coerce')
ela['L3,4 ela'] = pd.to_numeric(ela['L3,4 ela'], errors='coerce')
ela = ela[ela['Year'].dt.year <2018]
ela['School Name'] = ['P.S 307']*5 + ['P.S 8']*5
```

```
/var/folders/gp/bnf8n57s1nlgccs8xl1kk3mc0000gn/T/ipykernel_77833/1678129904.py:1
9: DtypeWarning: Columns (8,9,10,11,12,13,14,15,16,17,18) have mixed types.
Specify dtype option on import or set low_memory=False.
  ela = pd.read_csv('ny-ela-results-2013-2019-public-all.csv',index_col=0)
```

[11]:

```
math
```

[11]:

```
      School Name       Year  Mean Scale Score Math    L1 Math    L2 Math  \
13067     P.S 307 2013-01-01             318.876679  13.000000  27.666666
13068     P.S 307 2014-01-01             322.076538  12.755102  26.785715
13069     P.S 307 2015-01-01             324.705414  10.540541  26.486486
13070     P.S 307 2016-01-01             324.625580  11.627907  24.883720
13071     P.S 307 2017-01-01             324.139587  10.526316  26.773455
13648       P.S 8 2013-01-01             287.357147  50.793652  30.158730
13649       P.S 8 2014-01-01             294.410858  38.759689  37.209301
13650       P.S 8 2015-01-01             287.568634  48.366013  31.372549
13651       P.S 8 2016-01-01             280.368408  53.383457  30.075188
13652       P.S 8 2017-01-01             286.697662  48.062016  27.906977

       L3,4 Math
13067  59.333332
13068  60.459183
13069  62.972973
13070  63.488373
13071  62.700230
13648  19.047619
13649  24.031008
13650  20.261438
13651  16.541353
13652  24.031008
```

[12]:

```python
chart_math = (
    alt.Chart(math)
    .mark_line(point=True)
    .encode(
        alt.X('Year', title=None, axis=alt.Axis(labelAngle=0)),
        y=alt.Y('Mean Scale Score Math:Q',
                axis=alt.Axis(titleAngle=0,titleX=60, titleY=-15),
                scale=alt.Scale(domain=[250, 350])),
        color=alt.Color('School Name:N', title='School')
    )
    .properties(
        width=300,
        height=300,
    )
)

chart_ela = (
    alt.Chart(ela)
    .mark_line(point=True)
    .encode(
        alt.X('Year', title=None, axis=alt.Axis(labelAngle=0)),
        y=alt.Y('Mean Scale Score ela:Q',
                axis=alt.Axis(titleAngle=0,titleX=60, titleY=-15),
                scale=alt.Scale(domain=[250, 350])),
        color=alt.Color('School Name:N', title='School')
    )
    .properties(
        width=300,
        height=300,
    )
)
p3 = (chart_math | chart_ela) .properties(
    title = 'Mean Score of Math and ELA for PS 307 and PS 8'
)
p3.save('p3.pdf')
p3.save('p3.png')
p3
```

[12]:

```
alt.HConcatChart(…)
```

This line graph depicts the trend of math and ela score of both public school 8 and 307 in the artical, it shows the average score from 2013 to 2017 while helping compare between the two school. The data comes from `Historical New York school test scores (math and language) https://infohub.nyced.org/reports/academics/test-results`

0.4 Graph4

[13]:

```
math = math.sort_values('Year', ascending=True)
math = math.melt(
    id_vars=["School Name", "Year", "Mean Scale Score Math"],
    value_vars=["L1 Math", "L2 Math", "L3,4 Math"],
    var_name="Level",
    value_name="Percentage"
)
math['School_Year'] = math['School Name'] + '_' + math['Year'].dt.year.
 ↪astype(str)
ela = ela.sort_values('Year', ascending=True)
ela = ela.melt(
    id_vars=["School Name", "Year", "Mean Scale Score ela"],
    value_vars=["L1 ela", "L2 ela", "L3,4 ela"],
    var_name="Level",
    value_name="Percentage"
)
ela['School_Year'] = ela['School Name'] + '_' + ela['Year'].dt.year.astype(str)
```

[14]:

```
import altair as alt
import pandas as pd

# Load and preprocess math data
math = pd.read_csv('ny-math-results-2013-2019-public-all.csv', index_col=0)
math = math[(math['School Name'] == 'P.S. 307 DANIEL HALE WILLIAMS') |␣
 ↪(math['School Name'] == 'P.S. 008 ROBERT FULTON')]
math = math[math['Grade'] == 'All Grades']
math = math[['School Name', 'Year', '% Level 1', '% Level 2', '% Level 3+4']]
math = math.rename(columns={'% Level 1': 'L1 Math', '% Level 2': 'L2 Math', '%␣
 ↪Level 3+4': 'L3,4 Math'})
math['Year'] = pd.to_datetime(math['Year'], format='%Y').dt.year
math = math.melt(id_vars=["School Name", "Year"], value_vars=["L1 Math", "L2␣
 ↪Math", "L3,4 Math"],
                 var_name="Level", value_name="Percentage")
math['School_Year'] = math['School Name'] + ' ' + math['Year'].astype(str)

# Load and preprocess ELA data
ela = pd.read_csv('ny-ela-results-2013-2019-public-all.csv', index_col=0)
ela = ela[(ela['School Name'] == 'P.S. 307 DANIEL HALE WILLIAMS') |␣
 ↪(ela['School Name'] == 'P.S. 008 ROBERT FULTON')]
ela = ela[ela['Grade'] == 'All Grades']
ela = ela[['School Name', 'Year', '% Level 1', '% Level 2', '% Level 3+4']]
ela = ela.rename(columns={'% Level 1': 'L1 ELA', '% Level 2': 'L2 ELA', '%␣
 ↪Level 3+4': 'L3,4 ELA'})
ela['Year'] = pd.to_datetime(ela['Year'], format='%Y').dt.year
```

```
ela = ela.melt(id_vars=["School Name", "Year"], value_vars=["L1 ELA", "L2 ELA",␣
 ↪"L3,4 ELA"],
               var_name="Level", value_name="Percentage")
ela['School_Year'] = ela['School Name'] + ' ' + ela['Year'].astype(str)

# Create math bar chart
math_chart = alt.Chart(math).mark_bar().encode(
    y=alt.Y("School_Year:N", title=None),
    x=alt.X("Percentage:Q", axis=alt.Axis(format='%', title='Math Level␣
 ↪Percentage')),
    color=alt.Color("Level:N", title="Math Levels"),
    tooltip=["School Name:N", "Year:O", "Level:N", alt.Tooltip("Percentage:Q",␣
 ↪format=".2f")]
).properties(width=300, height=300)

# Create ELA bar chart
ela_chart = alt.Chart(ela).mark_bar().encode(
    y=alt.Y("School_Year:N", title=None),
    x=alt.X("Percentage:Q" , axis=alt.Axis(format='%', title='ELA Level␣
 ↪Percentage')),
    color=alt.Color("Level:N", title="ELA Levels"),
    tooltip=["School Name:N", "Year:O", "Level:N", alt.Tooltip("Percentage:Q",␣
 ↪format=".2f")]
).properties(width=300, height=300)

# Combine both charts
final_chart = (math_chart | ela_chart).properties(title='')

# Save and display
final_chart.save('final_chart.pdf')
final_chart.save('final_chart.png')
final_chart
```

```
/var/folders/gp/bnf8n57s1nlgccs8xl1kk3mc0000gn/T/ipykernel_77833/4280645688.py:1
6: DtypeWarning: Columns (8,9,10,11,12,13,14,15,16,17,18) have mixed types.
Specify dtype option on import or set low_memory=False.
  ela = pd.read_csv('ny-ela-results-2013-2019-public-all.csv', index_col=0)
```

[14]:

```
alt.HConcatChart(...)
```

[15]:

```
math_p = alt.Chart(math).mark_bar(orient='horizontal').encode(
    y=alt.Y(
        "School_Year:N",
        title=None,
    ),
    x=alt.X(
        "Percentage:Q",
        stack='normalize',
        axis=alt.Axis(format='%', title='Math Level Percentage')
    ),
    color=alt.Color(
        "Level:N",
        title="Math Levels"
    ),
    tooltip=[
        alt.Tooltip("School Name:N"),
        alt.Tooltip("Year:O"),
        alt.Tooltip("Level:N"),
        alt.Tooltip("Percentage:Q", format=".2f")
    ]
).properties(
    width=300,
    height=300
)
ela_p = alt.Chart(ela).mark_bar(orient='horizontal').encode(
    y=alt.Y(
        "School_Year:N",
        title=None,
    ),
    x=alt.X(
        "Percentage:Q",
        stack='normalize',
        axis=alt.Axis(format='%', title='ELA Level Percentage')
    ),
    color=alt.Color(
        "Level:N",
        title="Ela Levels"
    ),
    tooltip=[
        alt.Tooltip("School Name:N"),
        alt.Tooltip("Year:O"),
        alt.Tooltip("Level:N"),
        alt.Tooltip("Percentage:Q", format=".2f")
    ]
).properties(
    width=300,
    height=300
)
p4 = (math_p | ela_p).properties(
    title = 'Percentage of Math and ELA Levels for PS 307 and PS 8'
)
p4.save('p4.pdf')
p4.save('p4.png')
p4
```

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

This is a stack bar chart showing the student score level percentages of two public school 8 and 307 from 2013 to 2017, where Level 1 represent the highest score and L3,4 are the lower ones. We can compare the student level distribution between two school as well as different years. The data comes from `Historical New York school test scores (math and language)` `https://infohub.nyced.org/reports/academics/test-results`.