PSTAT 100 Homework 2

In [1]:

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

Background

Gender achievement gaps in education have been well-documented over the years -- studies consistently find boys outperforming girls on math tests and girls outperforming boys on reading and language tests. A particularly controversial article (https://www.jstor.org/stable/1684489) was published in Science in 1980 arguing that this pattern was due to an 'innate' difference in ability (focusing, of course, on mathematics rather than on reading and language). Such views persisted in part because studying systematic patterns in achievement nationwide was a challenge due to differential testing standards across school districts and the general lack of availability of large-scale data.

It is only recently that data-driven research has begun to reveal socioeconomic drivers of achievement gaps. The Standford Educational Data Archive (https://edopportunity.org/) (SEDA), a publicly available database on academic achievement and educational opportunity in U.S. schools, has supported this effort. The database is part of a broader initiave aiming to improve educational opportunity by enabling researchers and policymakers to identify systemic drivers of disparity.

> SEDA includes a range of detailed data on educational conditions, contexts, and outcomes in school districts and counties across the United States. It includes measures of academic achievement and achievement gaps for school districts and counties, as well as district-level measures of racial and socioeconomic composition, racial and socioeconomic segregation patterns, and other features of the schooling system.

The database standardizes average test scores for schools 10,000 U.S. school districts relative to national standards to allow comparability between school districts and across grade levels and years. The test score data come from the U.S. Department of Education. In addition, multiple data sources (American Community Survey and Common Core of Data) are integrated to provide district-level socioeconomic and demographic information.

A study of the SEDA data published in 2018 (https://cepa.stanford.edu/content/gender-achievement-gaps-us-school-districts) identified the following persistent patterns across grade levels 3 - 8 and school ears from 2008 through 2015:

- a consistent reading and language achievement gap favoring girls;
- *no* national math achievement gap on average; and
- local math achievement gaps that depend on the socioeconomic conditions of school districts. You can read about the main findings of the study in this brief NY Times article (https://www.nytimes.com/interactive/2018/06/13/upshot/boys-girls-math-reading-tests.html).

Below, we'll work with selected portions of the database. The full datasets can be downloaded here (https://edopportunity.org/get-the-data/seda-archive-downloads/).

Assignment objectives

In this assignment, you'll explore achievement gaps in California school districts in 2018, reproducing the findings described in the article above (https://www.nytimes.com/interactive/2018/06/13/upshot/boys-girls-math-reading-tests.html) on a more local scale and with the most recent SEDA data. This will afford you an opportunity to practice the first several stages of the data science lifecycle: collect, acquaint, tidy, and explore.

Collect/acquiant

- review data documentation
- identify population, sampling frame, sample
- assess scope of inference

Tidy

- data import
- slicing and filtering
- merging multiple data frames
- pivoting tables
- renaming and reordering variables

Explore

- scatterplots
- basic plotting aesthetics
- faceted plots
- visualizing trends
- aggregation and tabulation

Communicate

- narrative summary of exploratory analysis

0. Getting acquainted with the SEDA data

The cell below imports the district-level SEDA data from California in 2018. The test score data is stored in a separate file (`ca-main.csv`) from the socioeconomic and demographic covariate data (`ca-cov.csv`).

In [2]:

```
# import seda data
ca_main = pd.read_csv('data/ca-main.csv')
ca_cov = pd.read_csv('data/ca-cov.csv')
```

Test score data

The first few rows of the test data are shown below. The columns are:

Column name	Meaning
`sedalea`	District ID
`grade`	Grade level
`stateabb`	State abbreviation
`sedaleaname`	District name
`subject`	Test subject
`cs_mn_...`	Estimated mean test score
`cs_mnse_...`	Standard error for estimated mean test score
`totgyb_...`	Number of individual tests used to estimate the mean score

In [3]:

```
ca_main.head(3)
```

Out[3]:

	sedalea	grade	stateabb	sedaleaname	subject	cs_mn_all	cs_mnse_all	totgyb_all	cs_mn_asn	cs_mnse_asn	...	totgyb_whg	cs_
0	600001	4	CA	ACTON-AGUA DULCE UNIFIED ...	mth	-0.367007	0.108543	86.0	NaN	NaN	...	79.0	-(
1	600001	4	CA	ACTON-AGUA DULCE UNIFIED ...	rla	0.005685	0.117471	85.0	NaN	NaN	...	78.0	(
2	600001	6	CA	ACTON-AGUA DULCE UNIFIED ...	rla	-0.000040	0.092172	114.0	NaN	NaN	...	NaN	

3 rows × 59 columns

The test score means for each district are named `cs_mn_...` with an abbreviation indicating subgroup (such as mean score for all `cs_mean_all` , for boys `cs_mean_mal` , for white students `cs_mn_wht` , and so on). Notice that these are generally small-ish: decimal numbers between -0.5 and 0.5.

These means are *estimated* from a number of individual student tests and *standardized* relative to national averages. They represent the number of standard deviations by which a district mean differs from the national average. So, for instance, the value `cs_mn_all = 0.1` indicates that the district average is estimated to be 0.1 standard deviations greater than the national average on the corresponding test and at the corresponding grade level.

Q0 (a). Interpreting test score values

Interpret the average math test score for all 4th grade students in Acton-Agua Dulce Unified School District (the first row of the dataset shown above).

Answer

These students' average math score is a little bit lower than all students' average math score in the school.

Covariate data

The first few rows of the covariate data are shown below. The column information is as follows:

Column name	Meaning
`sedalea`	District ID
`grade`	Grade level
`sedaleanm`	District name
`urban`	Indicator: is the district in an urban locale?
`suburb`	Indicator: is the district in a suburban locale?
`town`	Indicator: is the district in a town locale?
`rural`	Indicator: is the district in a rural locale?
`locale`	Description of district locale
Remaining variables	Demographic and socioeconomic measures

In [4]:

```
ca_cov.head(3)
```

Out[4]:

	sedalea	grade	sedaleanm	urban	suburb	town	rural	locale	perind	perasn	...	snapall	snapblk	snaphsp	snapwht	sing
0	600001	4.0	ACTON-AGUA DULCE UNIFIED ...	0.0	0.0	0.0	1.0	Rural, Distant	0.003893	0.045901	...	0.035165	0.20293	0.0819	0.032362	
1	600001	5.0	ACTON-AGUA DULCE UNIFIED ...	0.0	0.0	0.0	1.0	Rural, Distant	0.003788	0.046652	...	0.035165	0.20293	0.0819	0.032362	
2	600001	6.0	ACTON-AGUA DULCE UNIFIED ...	0.0	0.0	0.0	1.0	Rural, Distant	0.003218	0.043657	...	0.035165	0.20293	0.0819	0.032362	

3 rows × 60 columns

You will only be working with a handful of the demographic and socioeconomic measures, so you can put off getting acquainted with those until selecting a subset of variables.

Q0 (b). Data semantics

In the non-public data, observational units are students -- test scores are measured for each student. However, in the SEDA data you've imported, scores are *aggregated* to the district level by grade. Let's regard estimated test score means for each grade as distinct variables, so that an observation consists in a set of estimated means for different grade levels and groups. In this view, what are the observational units in the test score dataset? Are they the same or different for the covariate dataset?

Answer

In the test score dataset, observational units are districts by grade levels. They are not the same for the covariate dataset.

Q0 (c). Sample sizes

How many observational units are in each dataset? Count the number of units in the test dataset and the number of units in the covariate dataset separately.

(*Hint*: use `.nunique()` .)

In [5]:

```
# solution
print(ca_main.nunique().sedalea)
print(ca_main.nunique().grade)# 872 * 5 = 4360
print(ca_cov.size) # 260400
```

```
872
5
260400
```

Q0 (d). Sample characteristics

Answer the questions below about the sampling design. You do not need to dig through any data documentation in order to resolve these questions.

(i) What is the relevant population for the datasets you've imported?

The population is 260400.

(ii) About what proportion (to within 0.1) of the population is captured in the sample?

(*Hint*: have a look at this website (https://www.cde.ca.gov/ds/sd/cb/ceffingertipfacts.asp).)

About 0.06.

(iii) Considering that the sampling frame is not identified clearly, what kind of dataset do you suspect this is (*e.g.*, administrative, data from a 'typical sample', census, etc.)?

I suspect that the dataset is data from a typical sampling.

Q0 (d). Scope of inference

In light of your description of the sample characteristics, what is the scope of inference for this dataset?

Answer

The scope is the studying status of each grade from 4 to 8 on two specific subjects in the same district, and the performance of these students among districts in California.

---

1. Tidy

Your goal will be to examine the relationship between gender achievement gaps and socioeconomic measures for school districts in California in 2018. In order to do this, the following manipulations of the imported data are needed:

- selecting columns of interest;
- filtering out non-urban districts;
- merging the covariate data with the test data; and
- putting the result in tidy format.

Since you've already had some guided practice doing this in previous assignments, you'll be left to fill in a little bit more of the details on your own in this assignment.

You'll work with the following variables from each dataset:

- **Test score data**
  - District ID
  - District name
  - Grade
  - Test subject
  - Estimated male-female gap
- **Covariate data**
  - District ID
  - Locale
  - Grade
  - Socioeconomic status (all demographic groups)
  - Log median income (all demographic groups)
  - Poverty rate (all demographic groups)
  - Unemployment rate (all demographic groups)
  - SNAP benefit receipt rate (all demographic groups)

Q1 (a). Variable names of interest

Download the codebooks by opening the 'data' directory from your Jupyter Lab file navigator (pstat100-s21-content > hw > hw2 > data), right-click the codebook .xlsx files, and select 'Download'. Identify the variables listed above, and store the column names in lists `main_vars` and `cov_vars`.

In [6]:

```
# store variable names of interest
main_vars = ['sedalea', 'sedaleaname', 'grade',
             'subject', 'cs_mn_mfg']
cov_vars = ['sedalea', 'locale', 'grade',
            'sesall', 'lninc50all', 'povertyall',
             'unempall', 'snapall']
```

Q1 (b). Slice columns

Use your result from Q1 (a) to slice the columns of interest from the covariate and test score data. Store the results as `main_sub` and `cov_sub`.

In [7]:

```
# slice columns to select variables of interest
cov_sub = ca_cov[cov_vars]
main_sub = ca_main[main_vars]
```

In the next step you'll merge the covariate data with the test score data. In order to do this, you can use the `pd.merge(A, B, how = ..., on = SHARED_COLS)` function, which will match the rows of `A` and `B` based on the shared columns `SHARED_COLS`. If `how = 'left'`, then only rows in `A` will be retained in the output (so `B` will be merged *to* `A`); conversely, if `how = 'right'`, then only rows in `B` will be retained in the output (so `A` will be merged *to* `B`).

A simple example of the use of `pd.merge` is illustrated below:

In [8]:

```
# toy data frames
A = pd.DataFrame(
    {'shared_col': ['a', 'b', 'c'],
    'x1': [1, 2, 3],
    'x2': [4, 5, 6]}
)

B = pd.DataFrame(
    {'shared_col': ['a', 'b'],
    'y1': [7, 8]}
)
```

In [9]:

```
A
```

Out[9]:

	shared_col	x1	x2
0	a	1	4
1	b	2	5
2	c	3	6

In [10]:

```
B
```

Out[10]:

	shared_col	y1
0	a	7
1	b	8

Below, if `A` and `B` are merged retaining the rows in `A`, notice that a missing value is input because `B` has no row where the shared column (on which the merging is done) has value `c`. In other words, the third row of `A` has no match in `B`.

In [11]:

```
# left join
pd.merge(A, B, how = 'left', on = 'shared_col')
```

Out[11]:

	shared_col	x1	x2	y1
0	a	1	4	7.0
1	b	2	5	8.0
2	c	3	6	NaN

If the direction of merging is reversed, and the row structure of `B` is dominant, then the third row of `A` is dropped altogether because it has no match in `B`.

In [12]:

```
# right join
pd.merge(A, B, how = 'right', on = 'shared_col')
```

Out[12]:

	shared_col	x1	x2	y1
0	a	1	4	7
1	b	2	5	8

Q1 (c). Merge

Follow the example above and merge the covariate and test score data on district ID and grade level, retaining only the columns from the test score data (meaning, treat the test score data as primary and merge the covariate data *to* the test score data). Store the result as `rawdata` and print the first four rows.

In [13]:

```
# merge covariates with gap data
rawdata = pd.merge(main_sub, cov_sub, how = 'left', on = ['sedalea', 'grade'])

# print first four rows
rawdata.head(4)
```

Out[13]:

	sedalea	sedaleaname	grade	subject	cs_mn_mfg	locale	sesall	lninc50all	povertyall	unempall	snapall
0	600001	ACTON-AGUA DULCE UNIFIED ...	4	mth	NaN	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165
1	600001	ACTON-AGUA DULCE UNIFIED ...	4	rla	NaN	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165
2	600001	ACTON-AGUA DULCE UNIFIED ...	6	rla	NaN	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165
3	600001	ACTON-AGUA DULCE UNIFIED ...	8	mth	-0.562855	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165

Q1 (d). Rename and reorder columns

Now rename and rearrange the columns of `rawdata` so that they appear in the following order and with the following names:

- District ID, District, Locale, Socioeconomic index, log(Median income), Poverty rate, Unemployment rate, SNAP rate, Grade, Gender gap, Subject

Store the result as `rawdata_mod1` and print the first four rows.

(*Hint*: first define a dictionary to map the old names to the new ones; then create a list of the new names specified in the desired order; then use `.rename()` and `.loc[]`. You can follow the renaming steps in HW1 as an example if needed.)

In [14]:

```
# define dictionary mapping for renaming columns
rename_map = {'sedalea': 'District ID', 'sedaleaname': 'District', 'locale': 'Locale',
              'lninc50all': 'log(Median income)', 'povertyall': 'Poverty rate',
              'unempall': 'Unemployment rate', 'snapall': 'SNAP rate', 'sesall': 'Socioeconomic index',
              'grade': 'Grade', 'cs_mn_mfg': 'Gender gap', 'subject': 'Subject'}
# specify order of columns
order_name = ['District ID', 'District', 'Locale', 'Socioeconomic index',
             'log(Median income)', 'Poverty rate', 'Unemployment rate',
             'SNAP rate', 'Grade', 'Gender gap', 'Subject']
# rename and reorder
rawdata_mod1 = rawdata.rename(columns=rename_map)
rawdata_mod1 = rawdata_mod1[order_name]
# print first four rows
rawdata_mod1.head(4)
```

Out[14]:

	District ID	District	Locale	Socioeconomic index	log(Median income)	Poverty rate	Unemployment rate	SNAP rate	Grade	Gender gap	Subject
0	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165	4	NaN	mth
1	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165	4	NaN	rla
2	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165	6	NaN	rla
3	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165	8	-0.562855	mth

Q1 (e). Pivot

Notice that the Gender gap column contains the values of two variables: the gap in estimated mean test scores for math tests, and the gap in estimated mean test scores for reading and language tests. To put the data in tidier format, use `.pivot` to pivot the table so that the gender gap column is spread into two columns corresponding to the entries of `Subject`. Name the resulting columns `Math gap` and `Reading gap`, and store the result as `rawdata_mod2` and print the first four rows.

In [15]:

```
# pivot to unstack gender gap (fixing tidy issue: multiple variables in one column)
rawdata_mod3 = rawdata_mod1[['Gender gap', 'Subject']].pivot(columns=['Subject'])
rawdata_mod2 = rawdata_mod1.copy()
rawdata_mod2[['Math gap', 'Reading gap']] = rawdata_mod3['Gender gap']
rawdata_mod2.drop(columns='Gender gap')
order_mod2 = ['District ID', 'District', 'Locale', 'Socioeconomic index',
              'log(Median income)', 'Poverty rate', 'Unemployment rate',
              'SNAP rate', 'Grade', 'Math gap', 'Reading gap', 'Subject']
rawdata_mod2 = rawdata_mod2[order_mod2]

rawdata_mod2['Math gap'] = rawdata_mod2['Math gap'].replace(np.nan, 0)
rawdata_mod2['Reading gap'] = rawdata_mod2['Reading gap'].replace(np.nan, 0)

i = 0
while i < 7513:
    if((rawdata_mod2.at[i, 'Grade'] == rawdata_mod2.at[i+1, 'Grade']) & (rawdata_mod2.at[i, 'District ID'] == raw
data_mod2.at[i+1, 'District ID'])):
        rawdata_mod2.at[i, 'Math gap'] = rawdata_mod2.at[i, 'Math gap'] + rawdata_mod2.at[i+1, 'Math gap']
        rawdata_mod2.at[i, 'Reading gap'] = rawdata_mod2.at[i, 'Reading gap'] + rawdata_mod2.at[i+1, 'Reading gap
']
        rawdata_mod2 = rawdata_mod2.drop([i+1])
        i += 2
    else:
        i += 1

rawdata_mod2['Math gap'] = rawdata_mod2['Math gap'].replace(0, np.nan)
rawdata_mod2['Reading gap'] = rawdata_mod2['Reading gap'].replace(0, np.nan)


# print first four rows
rawdata_mod2.head(4)
```

Out[15]:

	District ID	District	Locale	Socioeconomic index	log(Median income)	Poverty rate	Unemployment rate	SNAP rate	Grade	Math gap	Reading gap	Subject
0	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165	4	NaN	NaN	mth
2	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165	6	NaN	NaN	rla
3	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165	8	-0.562855	-0.785321	mth
5	600006	ROSS VALLEY ELEMENTARY ...	Suburb, Large	1.912972	11.607236	0.041418	0.048269	0.028006	4	-0.025131	-0.521408	mth

Q1 (f). Indexing

Finally, remove the name of the column index ('Subject') that was induced by the pivot step using `.rename_axis()`. Store the result as `data`, and print the first four rows.

In [16]:

```
# drop the name of column index induced by pivoting
data = rawdata_mod2.reset_index(drop=True).drop(columns='Subject')

# print first four rows
data.head(4)
```

Out[16]:

	District ID	District	Locale	Socioeconomic index	log(Median income)	Poverty rate	Unemployment rate	SNAP rate	Grade	Math gap	Reading gap
0	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165	4	NaN	NaN
1	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165	6	NaN	NaN
2	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165	8	-0.562855	-0.785321
3	600006	ROSS VALLEY ELEMENTARY ...	Suburb, Large	1.912972	11.607236	0.041418	0.048269	0.028006	4	-0.025131	-0.521408

Your final dataset should match the dataframe below. You can use this to check your answer and revise any portions above that lead to different results.

In [17]:

```
# intended result
data_reference = pd.read_csv('data/tidy-seda.csv')
data_reference.head(4)
```

Out[17]:

	District ID	District	Locale	log(Median income)	Poverty rate	Unemployment rate	SNAP rate	Socioeconomic index	Grade	Math gap	Reading gap
0	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	11.392048	0.091894	0.048886	0.035165	1.237209	4.0	NaN	NaN
1	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	11.392048	0.091894	0.048886	0.035165	1.237209	6.0	NaN	NaN
2	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	11.392048	0.091894	0.048886	0.035165	1.237209	8.0	-0.562855	-0.785321
3	600006	ROSS VALLEY ELEMENTARY ...	Suburb, Large	11.607236	0.041418	0.048269	0.028006	1.912972	4.0	-0.025131	-0.521408

Q1 (g). Sanity check

Ensure that your tidying did not inadvertently drop any observations: count the number of units in `data`. Does this match the number of units represented in the original test score data `ca_main`?

(*Hint*: use `.nunique()`.)

In [18]:

```
# number of districts in tidied data compared with raw data
data.nunique()['District ID'] # It is the same
```

Out[18]:

```
872
```

Q1 (h). Missing values

Gap estimates were not calculated for certain grades in certain districts due to small sample sizes (not enough individual tests recorded).

(i) What proportion of rows are missing for each of the reading and math gap variables?

In [19]:

```
# proportion of missing values
print(np.isnan(data['Math gap']).sum() / data['Math gap'].shape[0])
print(np.isnan(data['Reading gap']).sum() / data['Reading gap'].shape[0])
```

```
0.2830729166666667
0.2877604166666667
```

(ii) What proportion of *districts* have missing gap estimates for one or both test subjects for at least one grade level?

In [20]:

```
# proportion of districts with missing values
data_mod1 = data.copy()
data_mod1 = data_mod1[np.isnan(data_mod1['Math gap']) | np.isnan(data_mod1['Reading gap'])]
data_mod1.groupby(['District ID']).sum().shape[0] / data.nunique()['District ID']
```

Out[20]:

```
0.5091743119266054
```

(iii) Do you expect that this missingness is related to any particular district attribute(s)?

It seems like the locale of districts influence this missingness. The more away from urban the district is, the easier for it to miss some math gaps and reading gaps.

2. Explore

For the purpose of visualizing the relationship between estimated gender gaps and socioeconomic variables, you'll find it more helpful to store a non-tidy version of the data. The cell below rearranges the dataset so that one column contains an estimated gap, one column contains the value of a socioeconomic variable, and the remaining columns record the gap type and variable identity.

Ensure that your results from part 1 match the reference dataset before running this cell.

In [21]:

```
# format data for plotting
plot_df = data.melt(
    id_vars = order_name[0:9],
    value_vars = ['Math gap', 'Reading gap'],
    var_name = 'Gap type',
    value_name = 'Gap'
).melt(
    id_vars = ['District ID', 'District', 'Locale', 'Gap type', 'Gap', 'Grade'],
    value_vars = order_name[3:8],
    var_name = 'Socioeconomic variable',
    value_name = 'Measure'
)

# preview
plot_df.head()
```

Out[21]:

	District ID	District	Locale	Gap type	Gap	Grade	Socioeconomic variable	Measure
0	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	Math gap	NaN	4	Socioeconomic index	1.237209
1	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	Math gap	NaN	6	Socioeconomic index	1.237209
2	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	Math gap	-0.562855	8	Socioeconomic index	1.237209
3	600006	ROSS VALLEY ELEMENTARY ...	Suburb, Large	Math gap	-0.025131	4	Socioeconomic index	1.912972
4	600006	ROSS VALLEY ELEMENTARY ...	Suburb, Large	Math gap	0.143163	5	Socioeconomic index	1.912972

Altair, by default, limits the number of rows for input dataframes. We will need to disable this behavior in order to generate plots of this dataset.

In [22]:

```
# disable row limit for plotting
alt.data_transformers.disable_max_rows()
```

Out[22]:

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

Relationship between gender gaps and socioeconomic factors

The cell below generates a panel of scatterplots showing the relationship between estimated gender gap and socioeconomic factors for all grade levels by test subject. The plot suggests that the reading gap favors girls consistently across the socioeconomic spectrum -- in a typical district girls seem to outperform boys by 0.25 standard deviations of the national average. By contrast, the math gap appears to depend on socioeconomic factors -- boys only seem to outperform girls under *better* socioeconomic conditions.

In [23]:

```
# plot gap against socioeconomic variables by subject for all grades
fig1 = alt.Chart(plot_df).mark_circle(opacity = 0.1).encode(
    y = 'Gap',
    x = alt.X('Measure', scale = alt.Scale(zero = False), title = ''),
    color = 'Gap type'
).properties(
    width = 200,
    height = 200
).facet(
    column = alt.Column('Socioeconomic variable')
).resolve_scale(x = 'independent')

fig1
```

Out[23]:

Q2 (a). Relationships by grade level

Does the pattern shown in the plot above persist within each grade level? Modify the plot above to show these relationships by grade level: generate a panel of scatterplots of gap against socioeconomic measures by subject, where each column of the panel corresponds to one socioeconomic variable and each row corresponds to one grade level; the result should by a 5x5 panel. Resize the width and height of each facet so that the panel is of reasonable size.

(*Hint*: you may find it useful to have a look at the altair documentation on compound charts (https://altair-viz.github.io/user_guide/compound_charts.html), and lab 2, for examples to follow.)

In [24]:

```
# plotting codes here
fig2a = alt.Chart(plot_df).mark_circle(opacity = 0.1).encode(
    y = 'Gap',
    x = alt.X('Measure', scale = alt.Scale(zero = False), title = ''),
    color = 'Gap type'
).properties(
    width = 200,
    height = 200
).facet(
    column = alt.Column('Socioeconomic variable'),
    row = 'Grade'
).resolve_scale(x = 'independent')

# display
fig2a
```

Out[24]:

Answer: is the pattern consistent across grade level?

Yes, they are very likely for each socioeconomic variable.

Q2 (b). Do gaps shift across grade levels?

Construct a 2x5 panel of scatterplots showing estimated achievement gap against each of the 5 socioeconomic variables, with one row per test subject. Display grade level using a color gradient. Do the gaps seem to shift with grade level?

(*Hint:* plot gap against measure, facet by gap type (rows) and socioeconomic variable (columns), and color by grade.)

In [25]:

```
# plotting codes here
fig2b = alt.Chart(plot_df).mark_circle(opacity = 0.1).encode(
    y = 'Gap',
    x = alt.X('Measure', scale = alt.Scale(zero = False), title = ''),
    color = 'Grade'
).properties(
    width = 200,
    height = 200
).facet(
    column = alt.Column('Socioeconomic variable'),
    row = 'Gap type'
).resolve_scale(x = 'independent')

# display
fig2b
```

Out[25]:

Answer: Do the gaps seem to shift with grade level?

Yes -- the scatter shifts from dark to light as the estimated gap decreases for both subjects and all socioeconomic variables, indicating that as grade level increases, the gap increasingly favors girls in both math and reading and language.

Aggregating by grade

While the magnitude of the achievement gaps seems to depend very slightly on grade level (figure 2b), the *relationship* between achievement gap and socioeconomic factors does not differ from grade to grade (figure 2a). In what follows, you'll look at the average relationship between estimated achievement gap and median income after aggregating across grade. The cell below computes the mean of each variable across grade levels for each district.

In [26]:

```
# aggregate across grades
data_agg = data.groupby(['District ID', 'District', 'Locale']).mean().reset_index().drop(columns = 'Grade')
data_agg.head()
```

Out[26]:

	District ID	District	Locale	Socioeconomic index	log(Median income)	Poverty rate	Unemployment rate	SNAP rate	Math gap	Reading gap
0	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	1.237209	11.392048	0.091894	0.048886	0.035165	-0.562855	-0.785321
1	600006	ROSS VALLEY ELEMENTARY ...	Suburb, Large	1.912972	11.607236	0.041418	0.048269	0.028006	0.061163	-0.242572
2	600011	FORT SAGE UNIFIED ...	Rural, Distant	-0.478127	10.704570	0.159981	0.066333	0.102054	-0.015417	-0.191400
3	600012	TWIN RIDGES ELEMENTARY ...	Rural, Distant	-0.096379	10.589787	0.179102	0.059158	0.074903	NaN	NaN
4	600013	ROCKLIN UNIFIED ...	Suburb, Large	1.398133	11.399662	0.060338	0.045533	0.035016	0.054454	-0.312638

Similar to working with the disaggregated data, it will be helpful for plotting to melt the two gap variables into a single column.

In [27]:

```
# format for plotting
agg_plot_df = data_agg.melt(
    id_vars = order_name[0:7],
    value_vars = ['Math gap', 'Reading gap'],
    var_name = 'Subject',
    value_name = 'Average estimated gap'
)

agg_plot_df.head()
```

Out[27]:

	District ID	District	Locale	Socioeconomic index	log(Median income)	Poverty rate	Unemployment rate	Subject	Average estimated gap
0	600001	ACTON-AGUA DULCE UNIFIED ...	Rural, Distant	1.237209	11.392048	0.091894	0.048886	Math gap	-0.562855
1	600006	ROSS VALLEY ELEMENTARY ...	Suburb, Large	1.912972	11.607236	0.041418	0.048269	Math gap	0.061163
2	600011	FORT SAGE UNIFIED ...	Rural, Distant	-0.478127	10.704570	0.159981	0.066333	Math gap	-0.015417
3	600012	TWIN RIDGES ELEMENTARY ...	Rural, Distant	-0.096379	10.589787	0.179102	0.059158	Math gap	NaN
4	600013	ROCKLIN UNIFIED ...	Suburb, Large	1.398133	11.399662	0.060338	0.045533	Math gap	0.054454

Q2 (c). District average gaps

Construct a scatterplot of the average estimated gap against log(Median income) by subject for each district and add trend lines.

In [28]:

```
# scatterplot
base = alt.Chart(agg_plot_df).mark_circle(opacity = 0.1).encode(
    y = 'Average estimated gap',
    x = 'log(Median income)',
    color = 'Subject'
).properties(
    width = 200,
    height = 200
)

# trend line
trend = base.transform_loess('log(Median income)', 'Average estimated gap').mark_line()

# combine layers
fig2c = base + trend

# display
fig2c
```

Out[28]:

Now let's try to capture this pattern in *tabular* form. The cell below adds an `Income bracket` variable by cutting the median income into 8 contiguous intervals using `pd.cut()`, and tabulates the average socioeconomic measures and estimated gaps across districts by income bracket. Notice that with respect to the gaps, this displays the pattern that is shown visually in the figures above.

In [29]:

```
data_agg['Income bracket'] = pd.cut(np.e**data_agg['log(Median income)'], 8)
data_agg.groupby('Income bracket').mean().drop(columns = ['District ID', 'log(Median income)'])
```

Out[29]:

Income bracket	Socioeconomic index	Poverty rate	Unemployment rate	SNAP rate	Math gap	Reading gap
(21980.176, 46455.372]	-0.651999	0.194870	0.072689	0.155061	-0.070284	-0.309743
(46455.372, 70736.321]	0.291085	0.134078	0.063788	0.095303	-0.034061	-0.315545
(70736.321, 95017.269]	1.110433	0.088713	0.052785	0.048242	0.004239	-0.302114
(95017.269, 119298.218]	1.640159	0.064131	0.046848	0.030548	0.050006	-0.287117
(119298.218, 143579.167]	2.167272	0.050315	0.044343	0.011023	0.090138	-0.289529
(143579.167, 167860.115]	2.382258	0.043896	0.042379	0.008451	0.084683	-0.335975
(167860.115, 192141.064]	2.652906	0.040552	0.040120	0.010159	0.175793	-0.232306
(192141.064, 216422.013]	2.588499	0.047097	0.054055	0.002555	0.267301	-0.299798

Q2 (d). Proportion of districts with a math gap.

What proportion of districts in each income bracket have an average estimated math achievement gap favoring boys? Answer this question by performing the following steps:

- Append an indicator variable `Math gap favoring boys` to `data_agg` that records whether the average estimated math gap favors boys by more than 0.1 standard deviations relative to the national average.
- Compute the proportion of districts in each income bracket for which the indicator is true: group by bracket and take the mean.

In [30]:

```
# define indicator
Mathgap_favorboys = data_agg['Math gap'] > 0.1
data_agg['Math gap favoring boys'] = Mathgap_favorboys
data_agg

# proportion of districts with gap favoring boys, by income bracket
data_agg.groupby('Income bracket').mean()
```

Out[30]:

Income bracket	District ID	Socioeconomic index	log(Median income)	Poverty rate	Unemployment rate	SNAP rate	Math gap	Reading gap	Math gap favoring boys
(21980.176, 46455.372]	620805.849593	-0.651999	10.562801	0.194870	0.072689	0.155061	-0.070284	-0.309743	0.036585
(46455.372, 70736.321]	621709.390671	0.291085	10.954123	0.134078	0.063788	0.095303	-0.034061	-0.315545	0.061224
(70736.321, 95017.269]	622209.692771	1.110433	11.304530	0.088713	0.052785	0.048242	0.004239	-0.302114	0.084337
(95017.269, 119298.218]	619816.642857	1.640159	11.565446	0.064131	0.046848	0.030548	0.050006	-0.287117	0.232143
(119298.218, 143579.167]	619998.555556	2.167272	11.792310	0.050315	0.044343	0.011023	0.090138	-0.289529	0.388889
(143579.167, 167860.115]	626333.333333	2.382258	11.945790	0.043896	0.042379	0.008451	0.084683	-0.335975	0.444444
(167860.115, 192141.064]	631245.000000	2.652906	12.126221	0.040552	0.040120	0.010159	0.175793	-0.232306	0.500000
(192141.064, 216422.013]	625425.000000	2.588499	12.271721	0.047097	0.054055	0.002555	0.267301	-0.299798	1.000000

Q2 (e). Statewide averages

To wrap up the exploration, calculate a few statewide averages to get a sense of how some of the patterns above compare with the state as a whole.

(i) Compute the statewide average estimated achievement gaps.

In [31]:

```
# statewide average
print(data_agg['Math gap'].mean())
print(data_agg['Reading gap'].mean())
```

```
-0.020746235617636184
-0.3080140087905015
```

(ii) Compute the proportion of districts in the state with a math gap favoring boys.

In [32]:

```
# proportion of districts in the state with a math gap favoring boys
math_favor_boys = data_agg['Math gap'] > 0.1
math_favor_boys.mean()
```

Out[32]:

```
0.08314087759815242
```

(iii) Compute the proportion of districts in the state with a math gap favoring girls.

You will need to define a new indicator to perform this calculation.

In [33]:

```
# new indicator
math_favor_girls = data_agg['Math gap'] < 0.1

# proportion of districts in the state with a math gap favoring boys
math_favor_girls.mean()
```

Out[33]:

```
0.6524249422632794
```

3. Communicating results

Take a moment to review and reflect on your findings, and then answer the questions below.

Q3 (a). Summary

Write a brief summary of your exploratory analysis. What have you discovered about educational achievement gaps in California school districts? Aim to answer in 3-5 sentences or less.

Answer

In general, Girls perform better on math and reading than boys do in California. But when the income bracket becomes higher, the proportion of districts that boys perform better on math than girls grows bigger. And it only influences the math achievement gap.

Q3 (b). Hypothesize!

It's a cliche in statistics that 'correlation is not causation'. In your exploratory analysis, you identified a correlation between socioeconomic factors and achievement gaps. But clearly, affluence does not directly cause a math achievement gap favoring boys. What factors do you think might explain this association?

Answer

Boys living in richer homes(places) tend to get better education at home. Parents care more about their children's study so boys in these areas will focus more on school. And since it is always believed that boys are more talented in math than girls are, so it may be a reson to explain that there is a correlation between affluence and math achievement gap.

Submission

1. Save file to confirm all changes are on disk
2. Run *Kernel* > *Restart & Run All* to execute all code from top to bottom
3. Save file again to write any new output to disk
4. Generate PDF copy
5. Submit to Gradescope