Lab Assignment 11: Data Visualizations

DS 6001: Practice and Application of Data Science

H. Diana McSpadden (hdm5s)

Instructions

Please answer the following questions as completely as possible using text, code, and the results of code as needed. Format your answers in a Jupyter notebook. To receive full credit, make sure you address every part of the problem, and make sure your document is formatted in a clean and professional way.

Problem 0

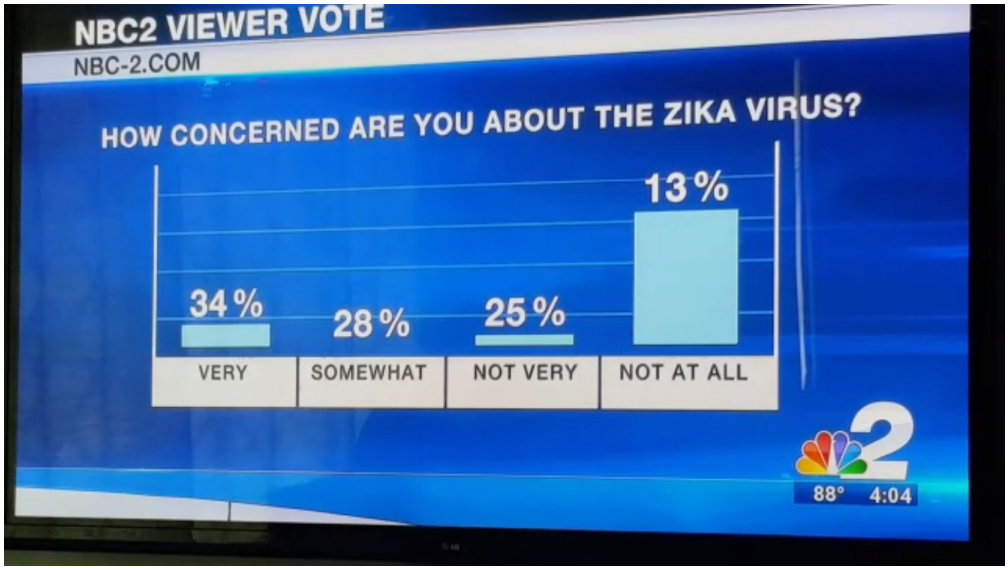
Import the following libraries:

In [ ]:
```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
```

Problem 1

Write a short paragraph that provides a critique of the following data visualizations. What's good about each figure, and what's not good? Pay particular attention to how well the figure communicates information to a general audience and tells a complete story. Make specific references to the ideas discussed in the first section of the Module 11 Jupyter notebook.

Part a

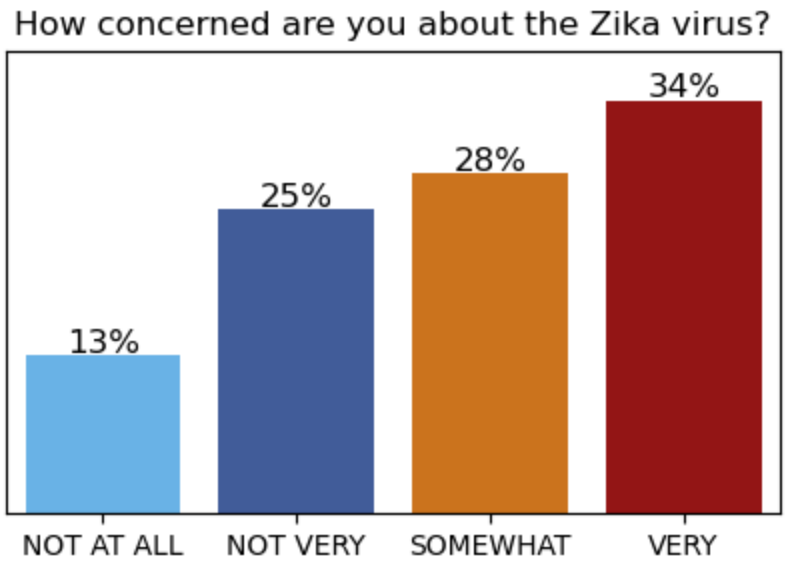


[1 point]

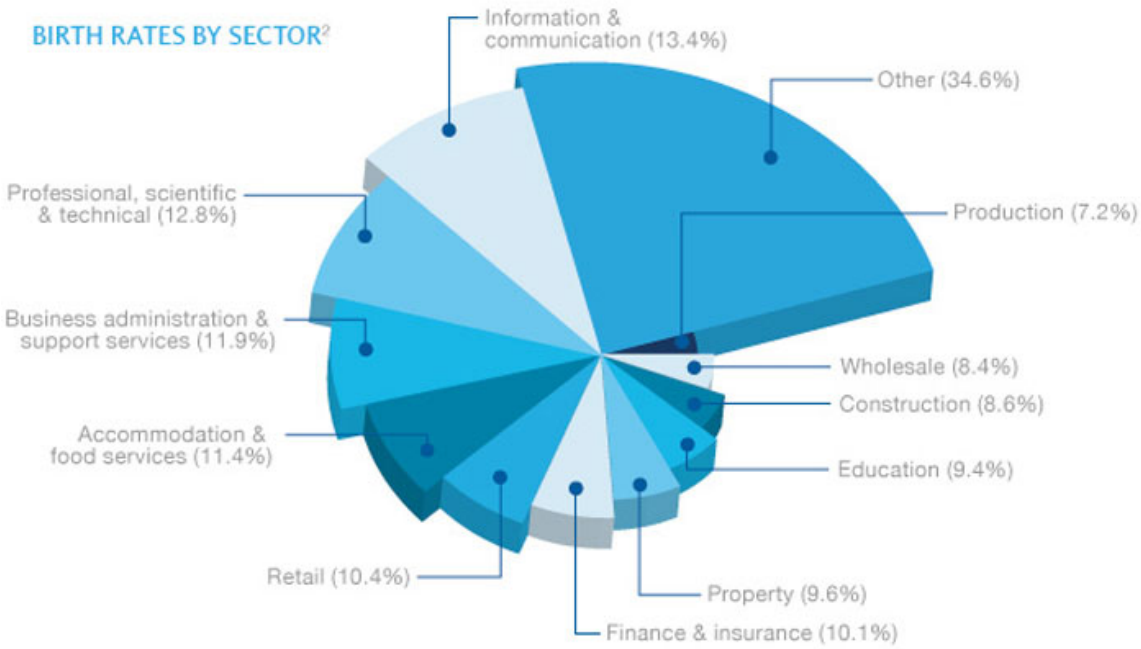
Answer Problem 1 Part a

What is good is that the order, from left to right, is in an ordinal order from very concerned to not at all concerned. Everything else about this plot bothers me. The relative bar heights do not represent realtive differences in the values. In other words, this breaks the **principle of proportional ink**. I don't believe the heights of the bars and the overall height of the bar area are in any way related. There are no counts for how many people were surveyed and no information is provided about whether the sample is representative of a particular population. The plot below corrects one of these issues, the relative bar height.

In [ ]:
```
zika_df = pd.DataFrame([['13','25','28','34']],columns=['NOT AT ALL','NOT VERY','SO

plt.figure(figsize=(5,3))
plt.title('How concerned are you about the Zika virus?')
# make a bar plot with the height being the value of the column and the x labels be
sns.barplot(data=[['13'],['25'],['28'],['34']], palette=['#54B5FB','#3257A8','#E872
# set the x tick labels to the column names
plt.xticks([0,1,2,3],['NOT AT ALL','NOT VERY','SOMEWHAT','VERY'])
# put the label on top of the bar for the measure
for i in range(4):
    plt.text(i, int(zika_df.iloc[0,i]) + 0.2, str(zika_df.iloc[0,i]) + '%', ha='cen

# don't show the y axis ticks
plt.yticks([])
# set the y range to 0 to 38
plt.ylim(0,38)
plt.show()
```



Part b

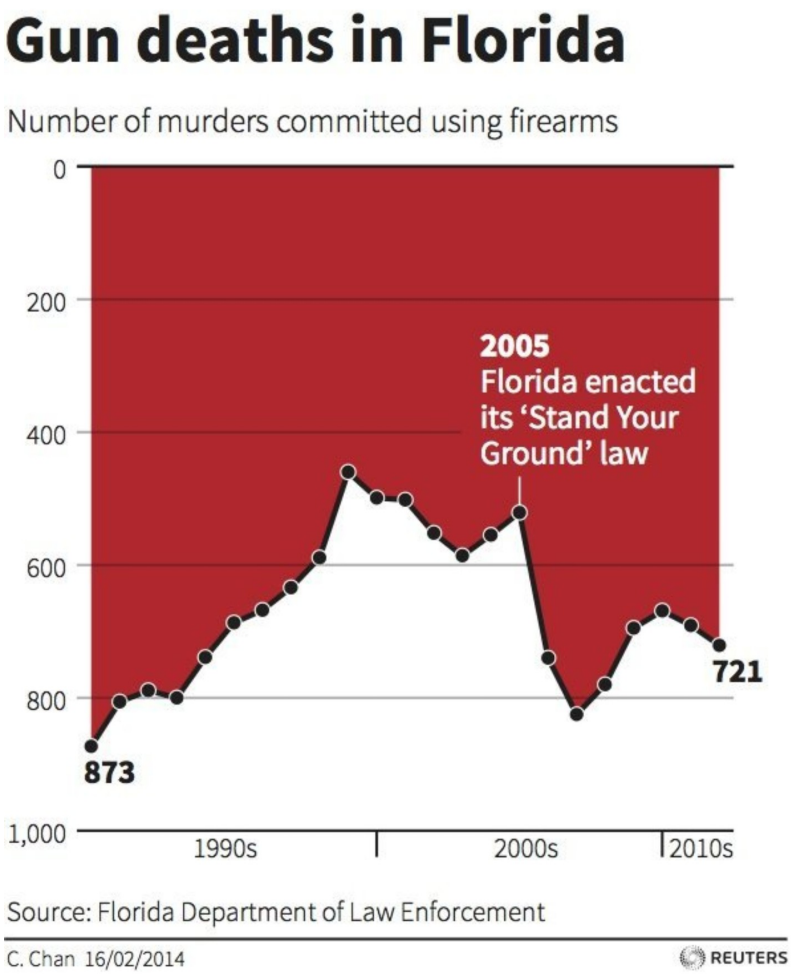


[1 point]

Answer Problem 1 Part b

What is good is that the sections of the pie chart are well labeled. Everything else bothers me. This plot has the Darrell Hull "One Dimensional graph" problem, or it breaks the **principle of proportional ink**. The 3D pie chart distorts the proportionality of the areas. This is very obvious with the difference in the Production vs. Other. Production is 7.2% and Other is 34.6%. This should only be a difference of 1:4.8 or 1:5 ink. That sliver of Production does not look like one-fifth of Other.

Part c



[1 point]

Answer Problem 1 Part c

This plot took a minute for me to even comprehend. The y axis is flipped bottom-to-top from increased to decreasing. I think this was an aesthetic choice to evoke dripping blood; however it makes the **increase** in murders committed using firearms harder to correlated with the enactment with Florida's 'Stand Your Ground' law, which is what the infographic is attempting to convey. I also find the x axis labels with the 1990s and 2000s decades and then the 2010s, that include two data points for 2011 and 2012 (if I am counting the data point correctly), confusing.

Problem 2

For the rest of this lab, we will once again be working with the 2019 General Social Survey.

In [ ]:
```
%%capture
gss = pd.read_csv("https://github.com/jkropko/DS-6001/raw/master/localdata/gss2018.
```

```
                encoding='cp1252', na_values=['IAP','IAP,DK,NA,uncodeable', 'NOT S
                                              'DK', 'IAP, DK, NA, uncodeable', '.a
```

Here is code that cleans the data and gets it ready to be used for data visualizations:

In [ ]:
```
mycols = ['id', 'wtss', 'sex', 'educ', 'region', 'age', 'coninc',
          'prestg10', 'mapres10', 'papres10', 'sei10', 'satjob',
          'fechld', 'fefam', 'fepol', 'fepresch', 'meovrwrk']
gss_clean = gss[mycols]
gss_clean = gss_clean.rename({'wtss':'weight',
                              'educ':'education',
                              'coninc':'income',
                              'prestg10':'job_prestige',
                              'mapres10':'mother_job_prestige',
                              'papres10':'father_job_prestige',
                              'sei10':'socioeconomic_index',
                              'fechld':'relationship',
                              'fefam':'male_breadwinner',
                              'fehire':'hire_women',
                              'fejobaff':'preference_hire_women',
                              'fepol':'men_bettersuited',
                              'fepresch':'child_suffer',
                              'meovrwrk':'men_overwork'},axis=1)
gss_clean.age = gss_clean.age.replace({'89 or older':'89'})
gss_clean.age = gss_clean.age.astype('float')
```

The `gss_clean` dataframe now contains the following features:

- `id` - a numeric unique ID for each person who responded to the survey
- `weight` - survey sample weights
- `sex` - male or female
- `education` - years of formal education
- `region` - region of the country where the respondent lives
- `age` - age
- `income` - the respondent's personal annual income
- `job_prestige` - the respondent's occupational prestige score, as measured by the GSS using the methodology described above
- `mother_job_prestige` - the respondent's mother's occupational prestige score, as measured by the GSS using the methodology described above
- `father_job_prestige` -the respondent's father's occupational prestige score, as measured by the GSS using the methodology described above
- `socioeconomic_index` - an index measuring the respondent's socioeconomic status
- `satjob` - responses to "On the whole, how satisfied are you with the work you do?"
- `relationship` - agree or disagree with: "A working mother can establish just as warm and secure a relationship with her children as a mother who does not work."
- `male_breadwinner` - agree or disagree with: "It is much better for everyone involved if the man is the achiever outside the home and the woman takes care of the home and family."

- `men_bettersuited` - agree or disagree with: "Most men are better suited emotionally for politics than are most women."
- `child_suffer` - agree or disagree with: "A preschool child is likely to suffer if his or her mother works."
- `men_overwork` - agree or disagree with: "Family life often suffers because men concentrate too much on their work."

Part a

Reorder the categories of `relationship` to "strongly agree", "agree", "disagree", and "strongly disagree".

Then create a simple barplot that shows the frequencies of the categories of `relationship` three times:

- once using `matplotlib` alone,
- once using `seaborn`,
- and once using the `.plot()` method from `pandas`.

Make sure each barplot has descriptive axis labels and a title, and set a good size for each figure displayed in the Jupyter notebook. [2 points]

In [ ]:
```
#gss_clean['relationship'].unique()
gss_clean['relationship'] = gss_clean.relationship.astype('category')


gss_clean['relationship'] = pd.Categorical(gss_clean.relationship.astype('category'
                      categories=["strongly agree","agree","disagree","strongly dis
                      ordered=True)
```

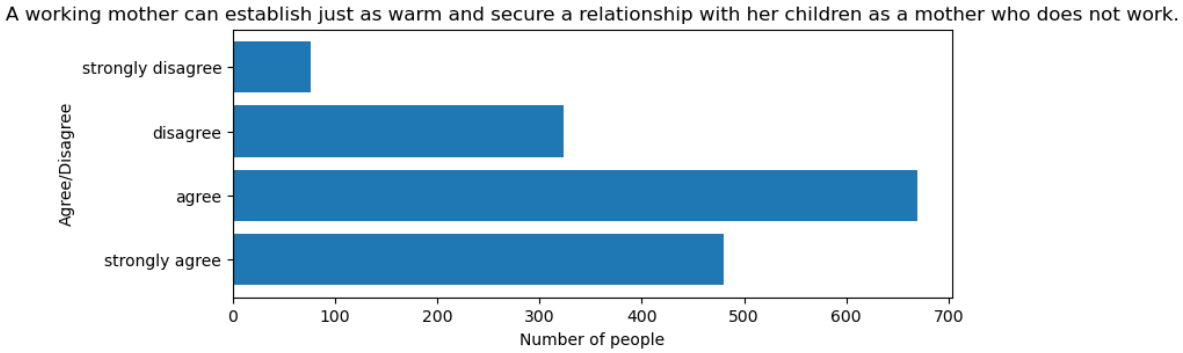
In [ ]:
```
mybars_relationship = gss_clean['relationship'].value_counts().sort_index()
mybars_relationship
```

Out[ ]:
```
strongly agree       480
agree                670
disagree             324
strongly disagree     76
Name: relationship, dtype: int64
```

First, Matplotlib

In [ ]:
```
plt.figure(figsize=(8, 3))
plt.barh(mybars_relationship.index, mybars_relationship.values)
plt.ylabel('Agree/Disagree')
plt.xlabel('Number of people')
plt.title('A working mother can establish just as warm and secure a relationship wi
```

Out[ ]:
```
Text(0.5, 1.0, 'A working mother can establish just as warm and secure a relations
hip with her children as a mother who does not work.')
```



Second, Seaborn

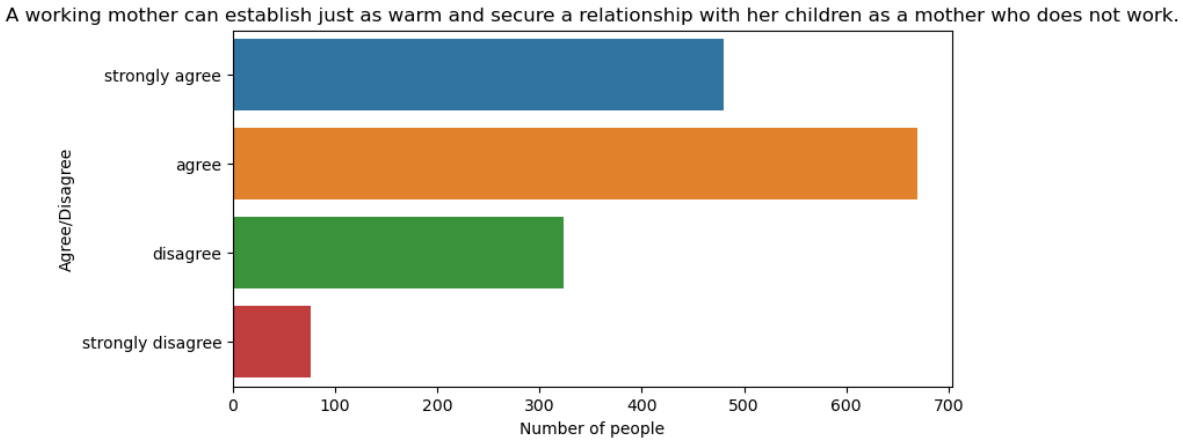
In [ ]:
```
mybars_relationship_df = mybars_relationship.reset_index()
mybars_relationship_df
```

Out[ ]:

	index	relationship
0	strongly agree	480
1	agree	670
2	disagree	324
3	strongly disagree	76

In [ ]:
```
plt.figure(figsize=(8, 4))
sns.barplot(x='relationship', y='index', data=mybars_relationship_df)
plt.ylabel('Agree/Disagree')
plt.xlabel('Number of people')
plt.title('A working mother can establish just as warm and secure a relationship wi
```

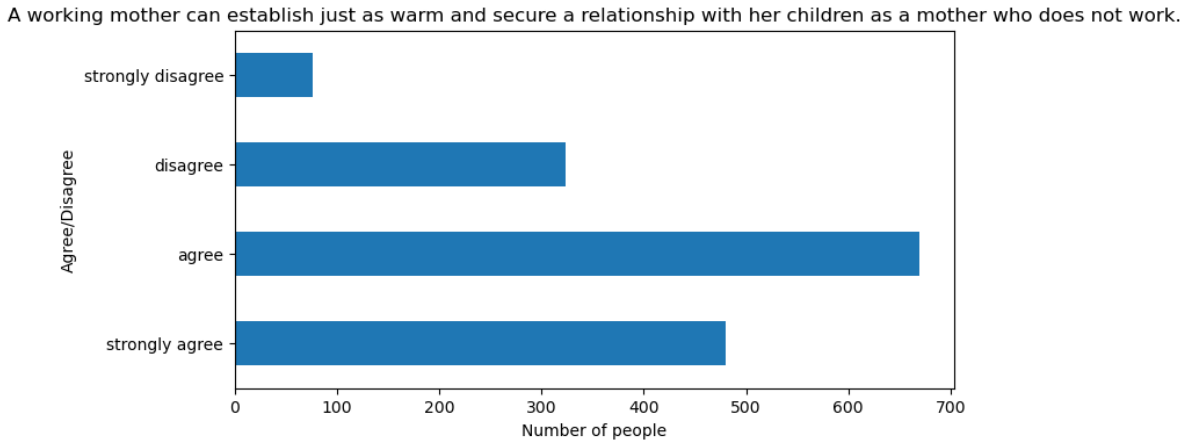
Out[ ]:
```
Text(0.5, 1.0, 'A working mother can establish just as warm and secure a relations
hip with her children as a mother who does not work.')
```



Third, with `plot()`

In [ ]:
```
mybars_relationship_df.plot(kind='barh', x='index', y='relationship', figsize=[8,4]
plt.ylabel('Agree/Disagree')
plt.xlabel('Number of people')
```

```
plt.title('A working mother can establish just as warm and secure a relationship wi
plt.legend().remove() #remove the legend because we don't want it
```



Part b

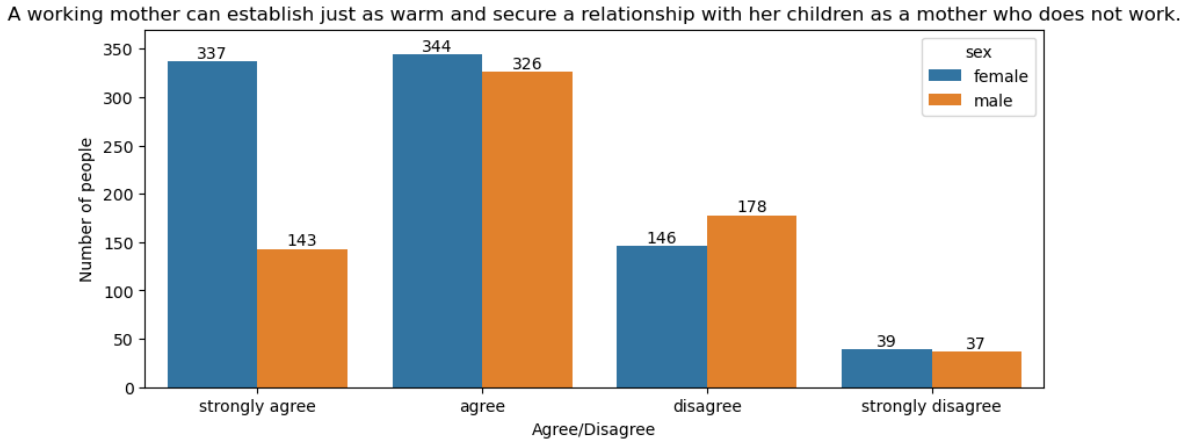
Create two barplots that show

- the frequency of the different levels of agreement for `relationship` for men and for women on the same plot,
- with bars for men and bars for women side-by-side,
- using different colors for the bars for men and the bars for women,
- listing these colors and the sex they refer to in a legend,
- and labeling each bar with the number the bar represents.

Create the first barplot using `seaborn` with the bars oriented vertically, and create the second barplot using the `.plot()` method with the bars oriented horizontally. [2 points]

First, the seaborn plot with vertical bars

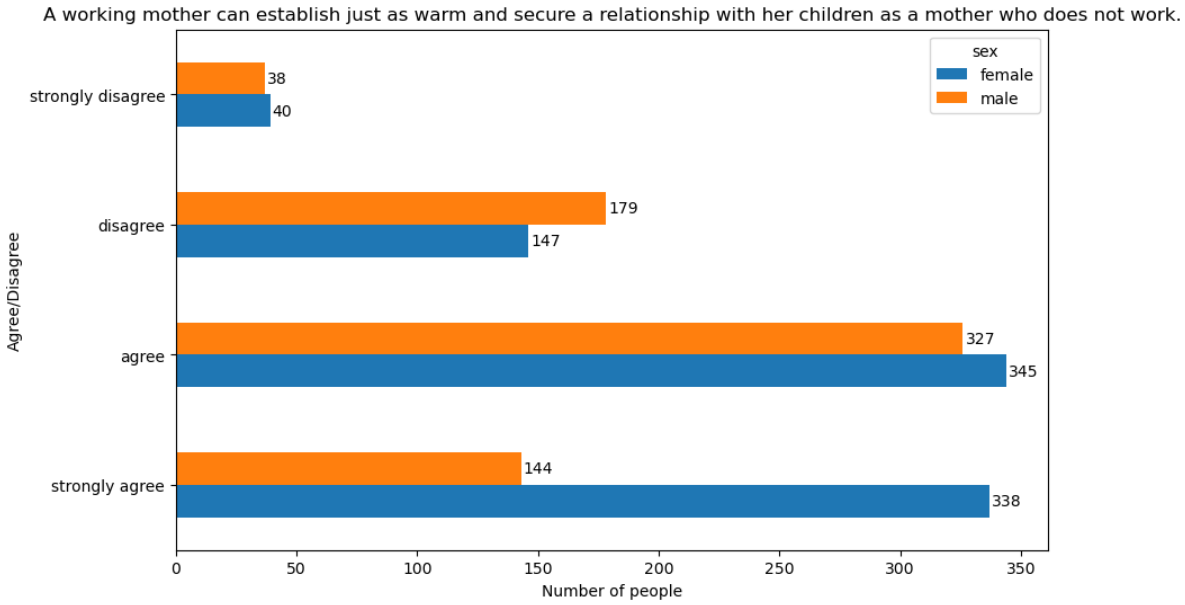
In [ ]:
```
gss_clean_plot = gss_clean.groupby(['sex', 'relationship']).size()
gss_clean_plot = gss_clean_plot.reset_index()
gss_clean_plot = gss_clean_plot.rename({0:'count'}, axis=1)

plt.figure(figsize=(10, 4))
myplt = sns.barplot(x='relationship', y='count', hue='sex', data=gss_clean_plot)
# set the y limit
plt.ylim([0,370])
plt.ylabel('Number of people')
plt.xlabel('Agree/Disagree')
plt.title('A working mother can establish just as warm and secure a relationship wi
for rect in myplt.patches:
    xcoor = rect.get_x() + .5*rect.get_width()
    ycoor = rect.get_height()
    plt.text(xcoor, ycoor, str(int(ycoor)),
             horizontalalignment='center',
             verticalalignment='bottom',
             fontsize=10)
```



Now with the `plot()` method with horizontal bars

In [ ]:
```
xtab = pd.crosstab(gss_clean.relationship, gss_clean.sex)
```

In [ ]:
```
myplot = xtab.plot(kind='barh', figsize = [10,6])
plt.ylabel('Agree/Disagree')
plt.xlabel('Number of people')
plt.title('A working mother can establish just as warm and secure a relationship wi
for rect in myplot.patches:
    ycoor = rect.get_y() + .5*rect.get_height()
    xcoor = rect.get_width() + 1
    plt.text(xcoor, ycoor, str(int(xcoor)),
             horizontalalignment='left',
             verticalalignment='center',
             fontsize=10)
```



Part c

Create a visualization with

- nine barplots, arranged in a 3x3 grid.

- The barplots should refer to each of the nine categories of `region`,
- and each barplot should be given a label that contains the name of the region.
- Within each barplot, list the categories of `relationship`,
- and display horizontal bars.

Only one figure is required. Use `seaborn`, `matplotlib`, and `.plot()` as you see fit. [2 points]

I think the `seaborn FacetGrid` should work nicely for this problem

In [ ]:
```
# gss_clean turn region into a categorical variable
gss_clean['region'] = pd.Categorical(gss_clean.region.astype('category'))
```

In [ ]:
```
gss_clean['region'].unique().to_list()
```

Out[ ]:
```
['new england',
 'middle atlantic',
 'pacific',
 'e. nor. central',
 'south atlantic',
 'w. sou. central',
 'mountain',
 'w. nor. central',
 'e. sou. central']
```

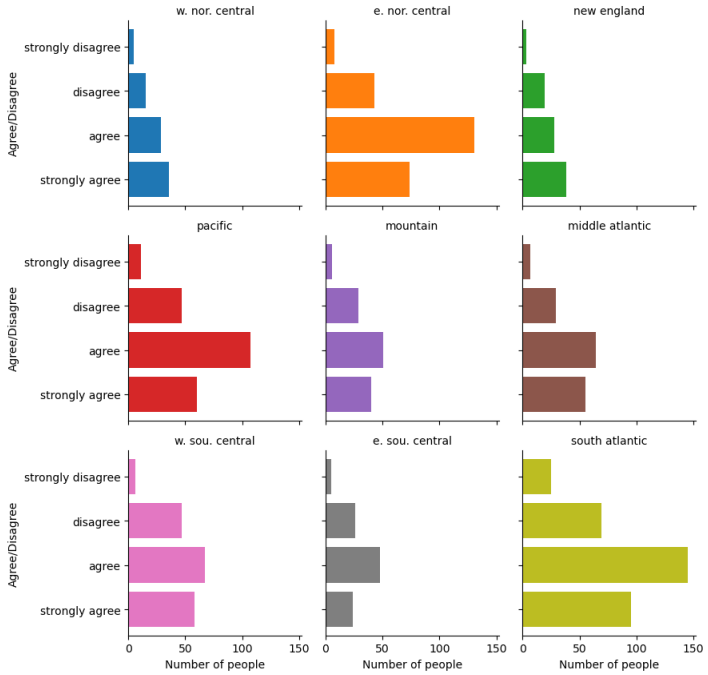
In [ ]:
```
gss_clean['region'] = pd.Categorical(gss_clean.region.astype('category'),
                     categories=["w. nor. central","e. nor. central","new england"
                     ordered=True)
```

In [ ]:
```
gss_plot = gss_clean.groupby(['region','relationship']).size().reset_index()
gss_plot = gss_plot.rename({0:'count'}, axis=1)
```

In [ ]:
```
g = sns.FacetGrid(gss_plot, col = 'region', hue = 'region', col_wrap=3, height=3, a
g.map(plt.barh, 'relationship', 'count')
g.set_titles('{col_name}')
g.set_axis_labels('Number of people', 'Agree/Disagree')
g.fig.subplots_adjust(top=.92)
g.fig.suptitle('A working mother can establish just as warm and secure a relationsh
```

Out[ ]:
```
Text(0.5, 0.98, 'A working mother can establish just as warm and secure a relation
ship with her children as a mother who does not work. By Region.')
```

A working mother can establish just as warm and secure a relationship with her children as a mother who does not work. By Region.

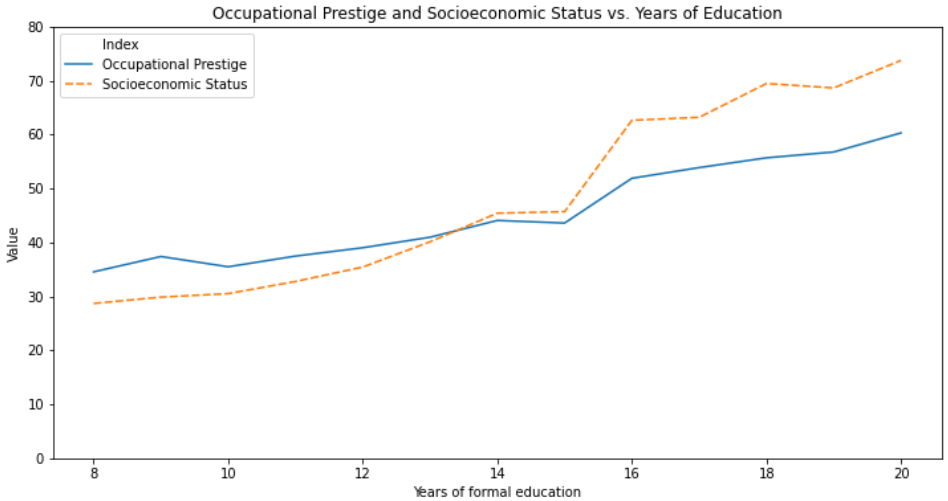


Problem 3

Write code that exactly replicates the following figures, including all aesthetic choices. **Don't worry, however, about making the size of the figures exactly the same as that varies from browser to browser.** All of the following figures are generated by a primary graphing function from `seaborn`.

Part a

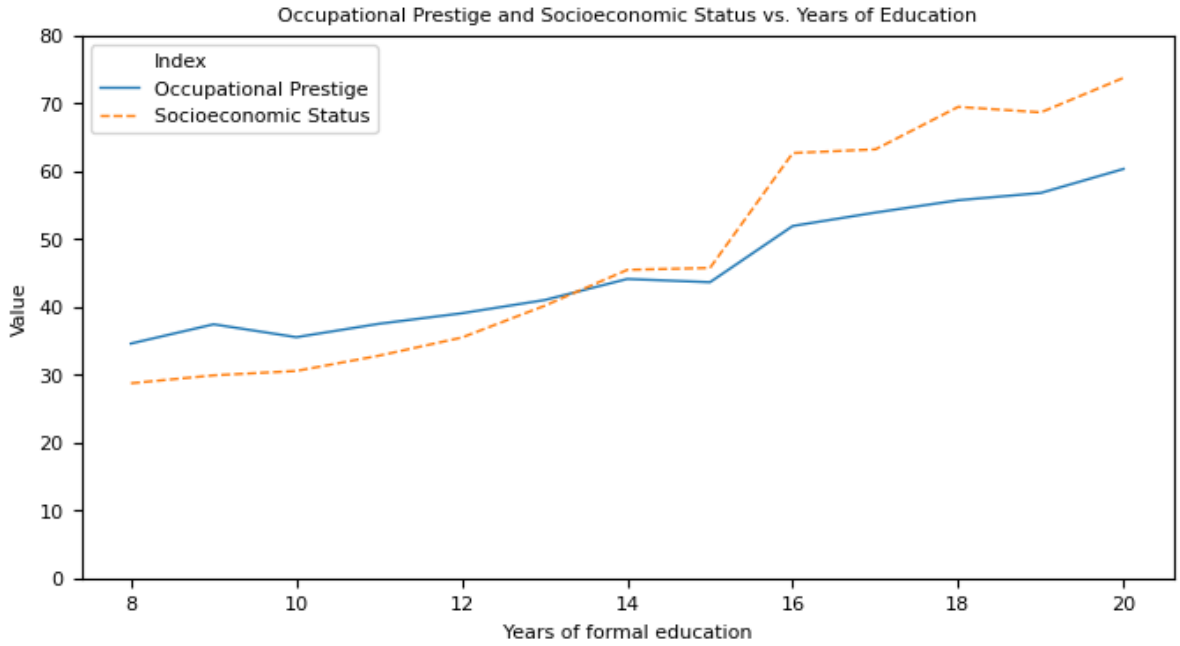
Replicate the following figure:



[Hint: the values of occuptational prestige and socioeconomic status are the means of `job_prestige` and `socioeconomic_index` within years of `education`. Note that values of `education` less than 8 are excluded.] [2 points]

In [ ]:
```
gss_p3a = gss_clean.query('education > = 8')

plt.figure(figsize=(8, 4))
sns.lineplot(data=gss_p3a, x='education', y='job_prestige', linewidth=1., errorbar=
sns.lineplot(data=gss_p3a, x='education', y='socioeconomic_index', linewidth=1., er
plt.ylim([0,80])
plt.ylabel('Value', fontsize=8)
plt.xlabel('Years of formal education', fontsize=8)
plt.xticks(fontsize=8)
plt.yticks(fontsize=8)
l = plt.legend(fontsize=8, loc='upper left', title='Index', title_fontsize=8)
l.get_title().set_position((-30, 0))
plt.title('Occupational Prestige and Socioeconomic Status vs. Years of Education',
```

Out[ ]:
```
Text(0.5, 1.0, 'Occupational Prestige and Socioeconomic Status vs. Years of Educat
ion')
```



Part b

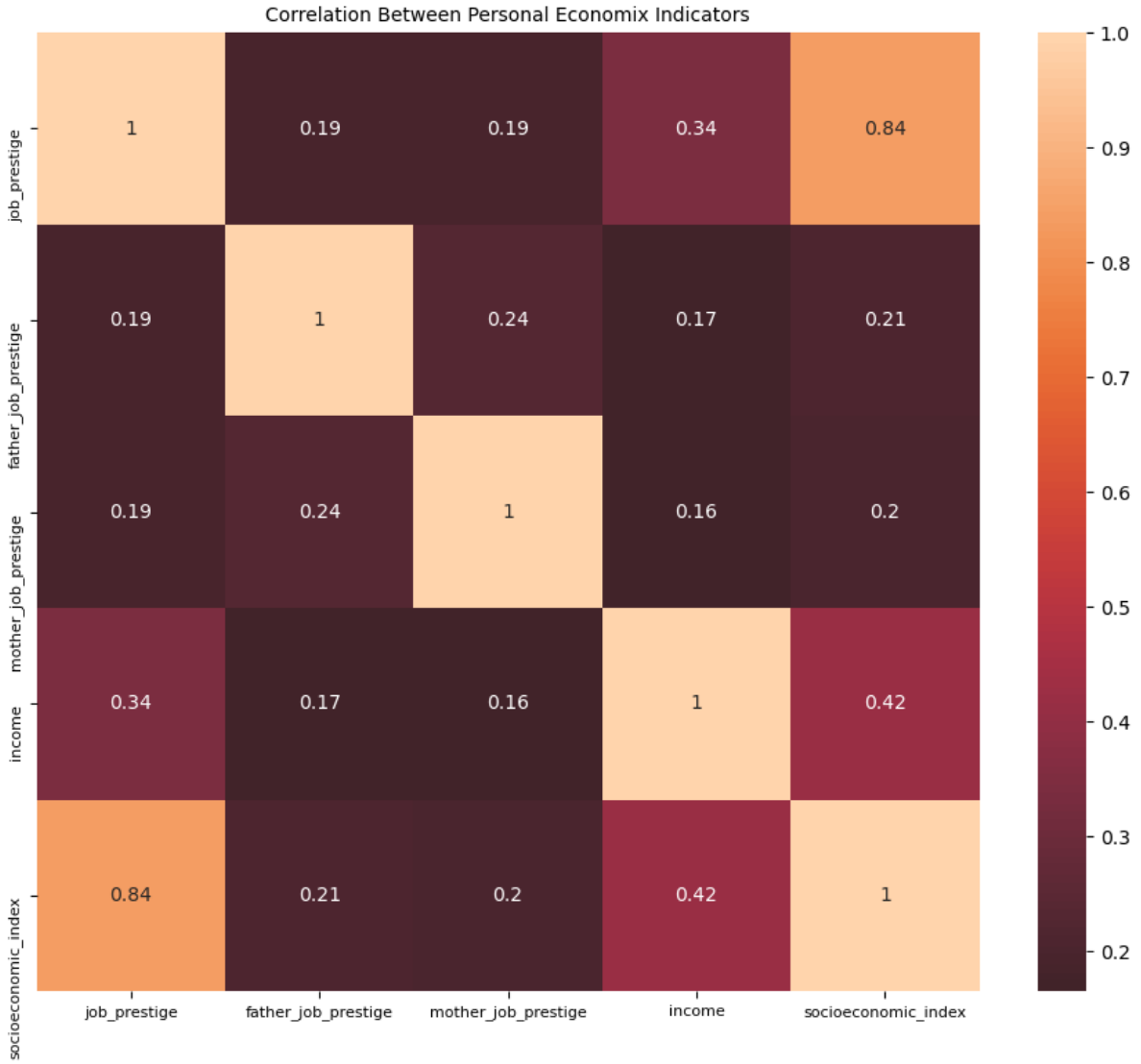
Replicate the following figure:

[Hint: to match the color scheme, you will need to set `center=0`.] [2 points]

In [ ]:
```
gss_p3b = gss_clean[['job_prestige','father_job_prestige','mother_job_prestige','in
# Draw a heatmap with the numeric values in each cell
f, ax = plt.subplots(figsize=(11, 9))
sns.heatmap(gss_p3b.corr(), annot=True, linewidths=.0, ax=ax, center=0.)
plt.yticks(rotation = 90, fontsize=8)

for label in ax.get_yticklabels(minor=False):
    print(label) # printing the labels to make sure I understand what I am doing
    label.set_verticalalignment('top')

plt.xticks(fontsize=8)
ax.tick_params(axis='x', labelrotation = 0)
plt.title('Correlation Between Personal Economix Indicators', fontsize=10)
```

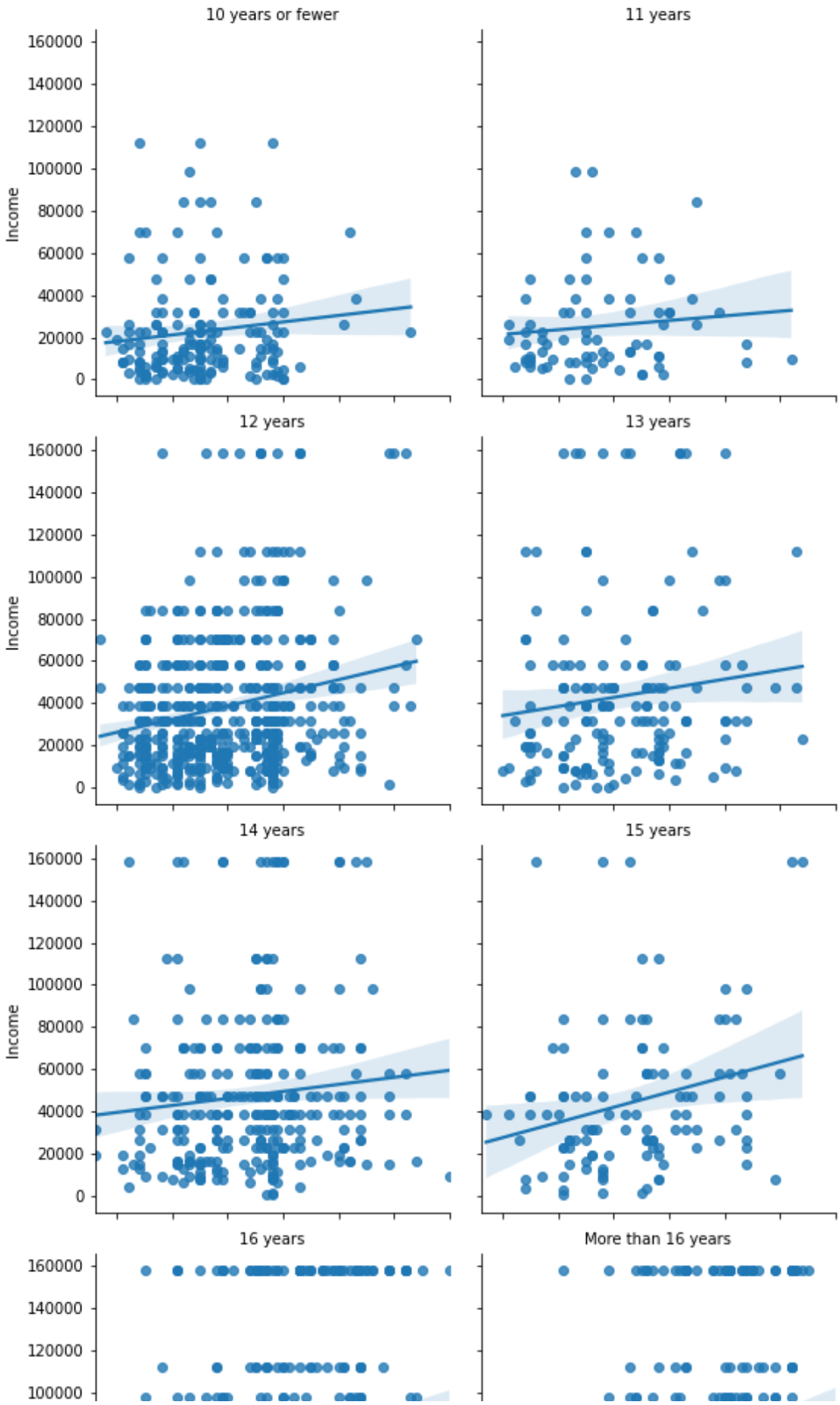
```
Text(0, 0.5, 'job_prestige')
Text(0, 1.5, 'father_job_prestige')
Text(0, 2.5, 'mother_job_prestige')
Text(0, 3.5, 'income')
Text(0, 4.5, 'socioeconomic_index')
```

Out[ ]:
```
Text(0.5, 1.0, 'Correlation Between Personal Economix Indicators')
```



Part c

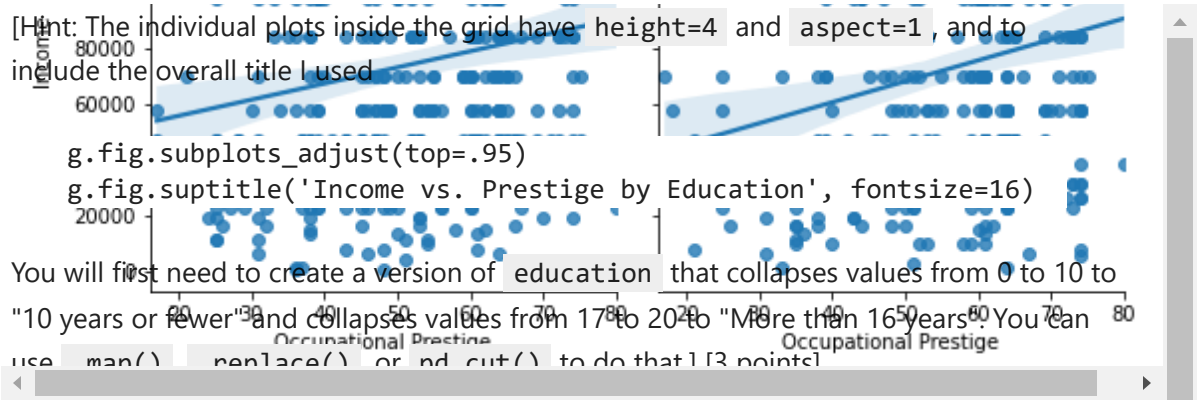
Replicate the following figure:


Income vs. Prestige by Education
10 years or fewer
11 years
12 years
13 years
14 years
15 years
16 years
More than 16 years
Income
160000
140000
120000
100000
80000
60000
40000
20000
0

[Hint: The individual plots inside the grid have `height=4` and `aspect=1`, and to include the overall title I used

```
g.fig.subplots_adjust(top=.95)
g.fig.suptitle('Income vs. Prestige by Education', fontsize=16)
```

You will first need to create a version of `education` that collapses values from 0 to 10 to "10 years or fewer" and collapses values from 17 to 20 to "More than 16 years". You can use `map()` `replace()` or `pd.cut()` to do that.] [3 points]

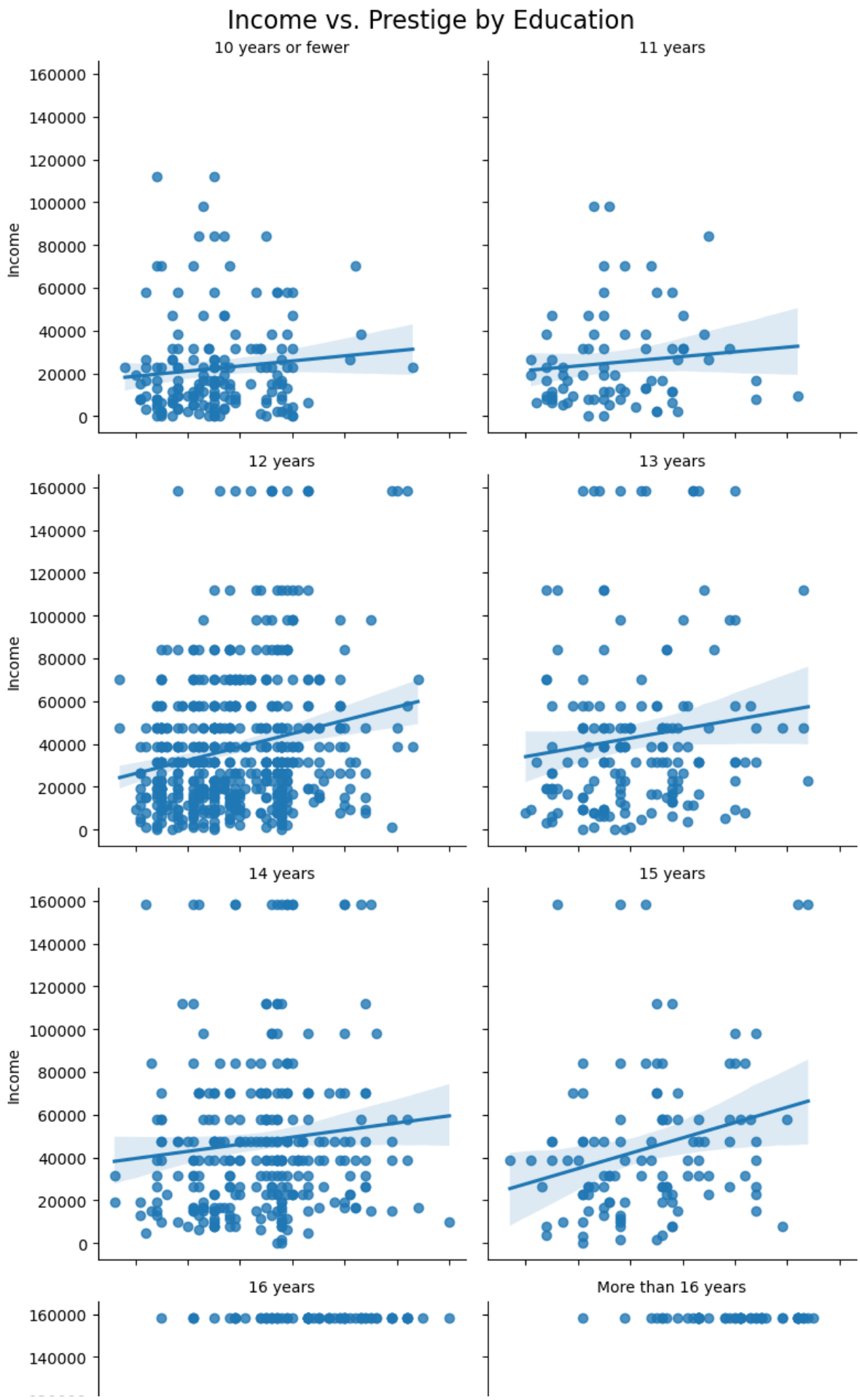


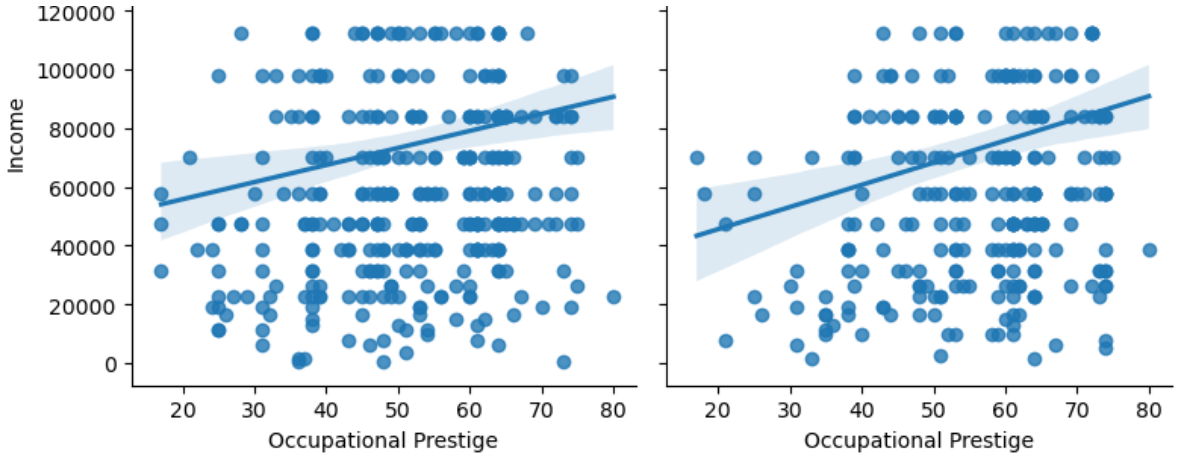
In [ ]:
```
df_p3_c  = gss_clean.copy()
df_p3_c['education_cut'] = pd.cut(df_p3_c['education'], bins=[0, 10, 11, 12,13,14,1
# turn education_cut into a categorical variable
df_p3_c['education_cut'] = pd.Categorical(df_p3_c.education_cut.astype('category'),
                                          categories=['10 years or fewer',"11 years
                      ordered=True)
```

In [ ]:
```
df_p3_c['education_cut'].unique()
```

Out[ ]:
```
['14 years', '10 years or fewer', '16 years', 'More than 16 years', '13 years', '1
2 years', '15 years', '11 years', NaN]
Categories (8, object): ['10 years or fewer' < '11 years' < '12 years' < '13 year
s' < '14 years' < '15 years' < '16 years' < 'More than 16 years']
```

In [ ]:
```
# set the figure size
g = sns.FacetGrid(df_p3_c, col = 'education_cut', col_wrap=2,
                  height=4, aspect=1)
g.map(sns.regplot, 'job_prestige', 'income')
g.set_titles('{col_name}')
g.set_axis_labels('Occupational Prestige', 'Income')
g.fig.subplots_adjust(top=.95)
g.fig.suptitle('Income vs. Prestige by Education', fontsize=16)
#make the figure size 90% as big as the default
g.fig.set_size_inches(8, 16)
```


Income vs. Prestige by Education
10 years or fewer
11 years
12 years
13 years
14 years
15 years
16 years
More than 16 years
Income
160000
140000
120000
100000
80000
60000
40000
20000
0



Problem 4

There is a consistent finding that in the United States that women get paid only 80% of what men get paid. Other research however finds that the gap is much smaller when comparing men and women who hold the same job. In this problem you will use the GSS data to investigate the following questions:

1. Do men have higher incomes than women?
2. If there is a difference, is this difference due to the fact that men have jobs with higher occupational prestige than women?

You may use any kind of data visualization and you may use multiple visualizations to find an answer to these questions. In order to receive credit for this problem, you must write in text what parts of your visualizations are important and what we should learn from the visualizations to answer the questions. Please consider the entire distributions of income and occupational prestige, not just the means or medians. [4 points]

In [ ]:
```
# set the figure size
g = sns.FacetGrid(df_p3_c, col = 'education_cut', hue='sex', col_wrap=2,
                  height=4, aspect=1)
g.map(sns.regplot, 'job_prestige', 'income')
# adjust the alpha of the points in the subplots
g.set_titles('{col_name}')
g.set_axis_labels('Occupational Prestige', 'Income')
g.fig.subplots_adjust(top=.95)
g.fig.suptitle('Figure 1: Income vs. Prestige by Education and Sex', fontsize=16)
#make the figure size 90% as big as the default
g.fig.set_size_inches(8, 16)
# add the legend to the upper right corner
g.add_legend(loc='upper right')
```

Out[ ]:
```
<seaborn.axisgrid.FacetGrid at 0x2ae87993040>
```

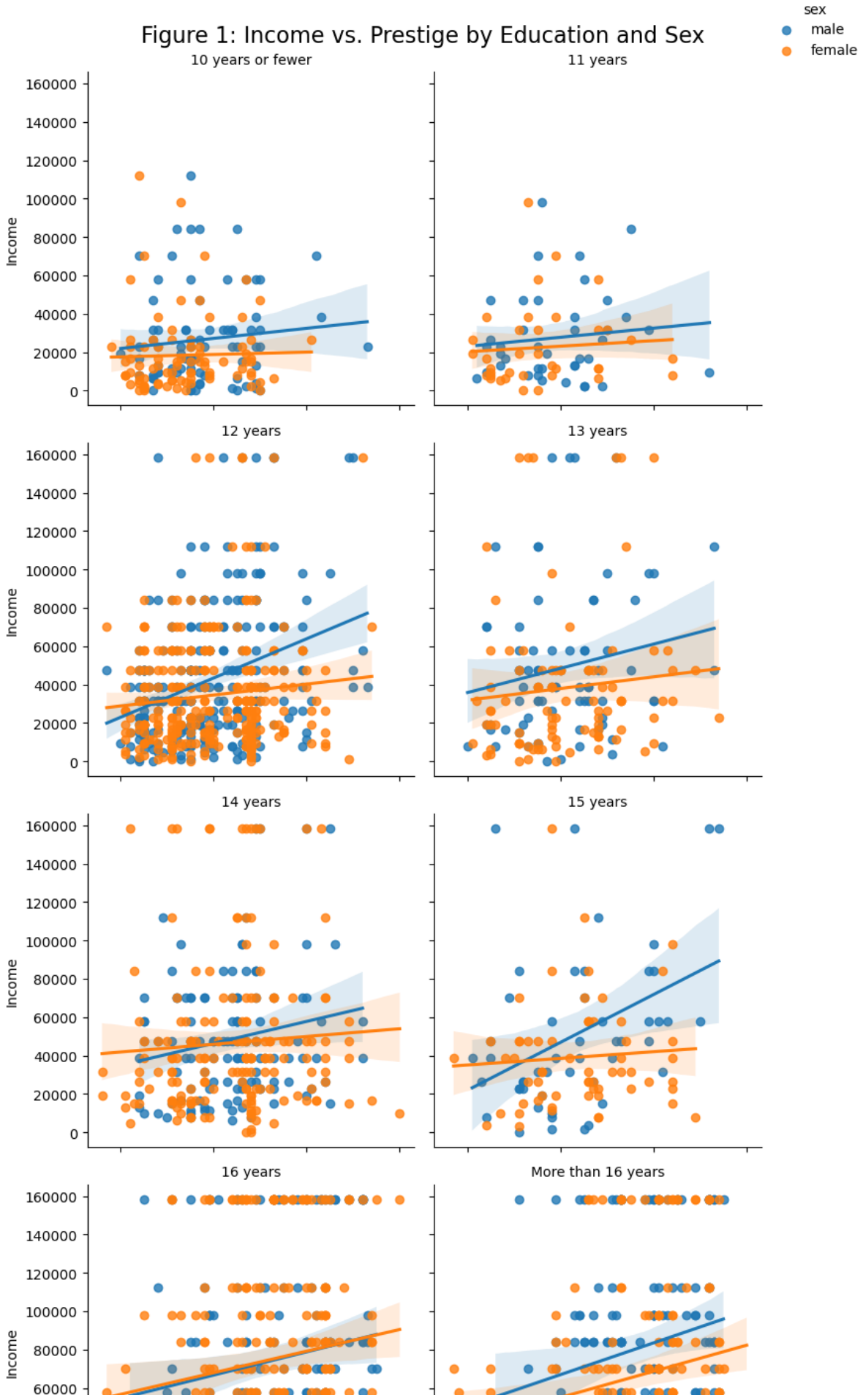


Figure 1: Income vs. Prestige by Education and Sex
sex
male
female
10 years or fewer
11 years
12 years
13 years
14 years
15 years
16 years
More than 16 years
Income
160000
140000
120000
100000
80000
60000
40000
20000
0

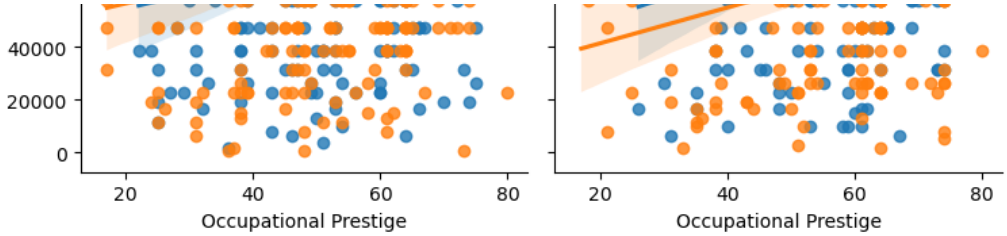


Figure 1: Income vs. Prestige by Education and Sex

I used the same plot from Problem 3, but also added the "Sex" dimension with color. This shows that the line of best fit for women generally has a lower intercept at the lowest occupational prestige and either a flatter trajectory (slope), or near equal slope for all education levels except "16 years of education" where the intercept and slope are equal, so only for a 4-year college degree are salaries equal for men and women with equal occupational prestige. Based on this p[lot and these best linear fits, even with similar occupational prestige and education, women's salaries are lower than men's salaries.

In [ ]: