Lab Assignment 10: Exploratory Data Analysis, Part 1

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.

In this lab, you will be working with the 2018 General Social Survey (GSS). The GSS is a sociological survey created and regularly collected since 1972 by the National Opinion Research Center at the University of Chicago. It is funded by the National Science Foundation. The GSS collects information and keeps a historical record of the concerns, experiences, attitudes, and practices of residents of the United States, and it is one of the most important data sources for the social sciences.

The data includes features that measure concepts that are notoriously difficult to ask about directly, such as religion, racism, and sexism. The data also include many different metrics of how successful a person is in his or her profession, including income, socioeconomic status, and occupational prestige. These occupational prestige scores are coded separately by the GSS. The full description of their methodology for measuring prestige is available here: http://gss.norc.org/Documents/reports/methodological-reports/MR122%20Occupational%20Prestige.pdf Here's a quote to give you an idea about how these scores are calculated:

> Respondents then were given small cards which each had a single occupational titles listed on it. Cards were in English or Spanish. They were given one card at a time in the preordained order. The interviewer then asked the respondent to "please put the card in the box at the top of the ladder if you think that occupation has the highest possible social standing. Put it in the box of the bottom of the ladder if you think it has the lowest possible social standing. If it belongs somewhere in between, just put it in the box that matches the social standing of the occupation."

The prestige scores are calculated from the aggregated rankings according to the method described above.

Problem 0

Import the following packages:

```
In [ ]: #pip install pandas-profiling
```

```
In [ ]: #pip install prince==0.8.3
```

```
In [ ]: import numpy as np
        import pandas as pd
        import sidetable
        import weighted # this is a module of wquantiles, so type pip install wquantiles or cc
        from scipy import stats
        from sklearn import manifold
        from sklearn import metrics
        import prince
        from pandas_profiling import ProfileReport
        pd.options.display.max_columns = None
```

```
C:\Users\dianam\AppData\Local\Temp\ipykernel_23976\3521480416.py:9: DeprecationWarnin
g: `import pandas_profiling` is going to be deprecated by April 1st. Please use `impo
rt ydata_profiling` instead.
  from pandas_profiling import ProfileReport
```

Then load the GSS data with the following code:

```
In [ ]: %%capture
        gss = pd.read_csv("https://github.com/jkropko/DS-6001/raw/master/localdata/gss2018.csv
                          encoding='cp1252', na_values=['IAP','IAP,DK,NA,uncodeable', 'NOT SURE
                                                        'DK', 'IAP, DK, NA, uncodeable', '.a',
```

Problem 1

Drop all columns except for the following:

1. `id` - a numeric unique ID for each person who responded to the survey
2. `wtss` - survey sample weights
3. `sex` - male or female
4. `educ` - years of formal education
5. `region` - region of the country where the respondent lives
6. `age` - age
7. `coninc` - the respondent's personal annual income
8. `prestg10` - the respondent's occupational prestige score, as measured by the GSS using the methodology described above
9. `mapres10` - the respondent's mother's occupational prestige score, as measured by the GSS using the methodology described above
10. `papres10` -the respondent's father's occupational prestige score, as measured by the GSS using the methodology described above
11. `sei10` - an index measuring the respondent's socioeconomic status
12. `satjob` - responses to "On the whole, how satisfied are you with the work you do?"

13. `fechld` - 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."
14. `fefam` - 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."
15. `fepol` - agree or disagree with: "Most men are better suited emotionally for politics than are most women."
16. `fepresch` - agree or disagree with: "A preschool child is likely to suffer if his or her mother works."
17. `meovrwrk` - agree or disagree with: "Family life often suffers because men concentrate too much on their work."

Then rename any columns with names that are non-intuitive to you to more intuitive and descriptive ones. Finally, replace the "89 or older" values of `age` with 89, and convert `age` to a float data type. [1 point]

In [ ]:
```
cols = ['id','wtss','sex','educ','region','age','coninc','prestg10','mapres10','papres
gss_p1 = gss[cols]

new_cols = ['id','wtss','sex','educ','region','age','ann_income','prestg10','mat_pres1
gss_p1.columns = new_cols
print(gss_p1.shape)
gss_p1.head()
```

```
(2348, 17)
```

Out[ ]:

	id	wtss	sex	educ	region	age	ann_income	prestg10	mat_pres10	pat_pres10	ses_10
0	1	2.357493	male	14.0	new england	43	NaN	47.0	31.0	45.0	65.3
1	2	0.942997	female	10.0	new england	74	22782.5000	22.0	32.0	39.0	14.8
2	3	0.942997	male	16.0	new england	42	112160.0000	61.0	32.0	72.0	83.4
3	4	0.942997	female	16.0	new england	63	158201.8412	59.0	NaN	39.0	69.3
4	5	0.942997	male	18.0	new england	71	158201.8412	53.0	35.0	45.0	68.6

Problem 2

Part a

Use the `ProfileReport()` function to generate and embed an HTML formatted exploratory data analysis report in your notebook. Make sure that it includes a "Correlations" report along with "Overview" and "Variables". [1 point]

In [ ]:
```
ProfileReport(gss_p1,
              title="GSS 2018 Profile Report",
              html={'style':{'full_width':True}},
              minimal=False).to_notebook_iframe()
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

Overview

Dataset statistics

Number of variables	17
Number of observations	2348
Missing cells	6276
Missing cells (%)	15.7%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	312.0 KiB
Average record size in memory	136.1 B

Variable types

Numeric	8
Categorical	9

Alerts

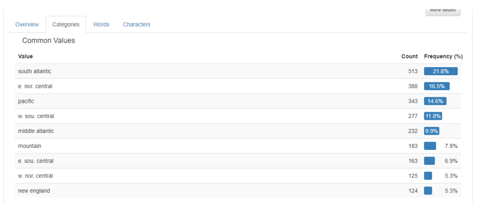
`age` has a high cardinality: 72 distinct values	**High cardinality**
`educ` is highly overall correlated with `ses_10`	**High correlation**
`prestg10` is highly overall correlated with `ses_10`	**High correlation**

Part b

Looking through the HTML report you displayed in part a, how many people in the data are from New England? [1 point]

Answer Part b

124 people in the data are in the New England region.


More details

Overview | Categories | Words | Characters

Common Values

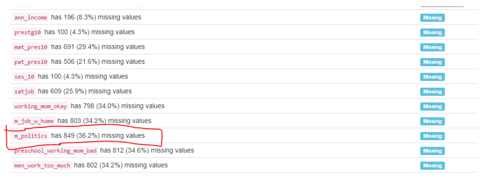
Value	Count	Frequency (%)
south atlantic	513	21.8%
e. nor. central	388	16.5%
pacific	343	14.6%
w. sou. central	277	11.8%
middle atlantic	232	9.9%
mountain	183	7.8%
e. sou. central	163	6.9%
w. nor. central	125	5.3%
new england	124	5.3%

Part c

Looking through the HTML report you displayed in part a, which feature in the data has the highest number of missing values, and what percent of the values are missing for this feature? [1 point]

Answer Part c

`m_politics` has the highest number of missing values. (This was `fepol` in the original csv.). 849 missing values, or 36.2%.

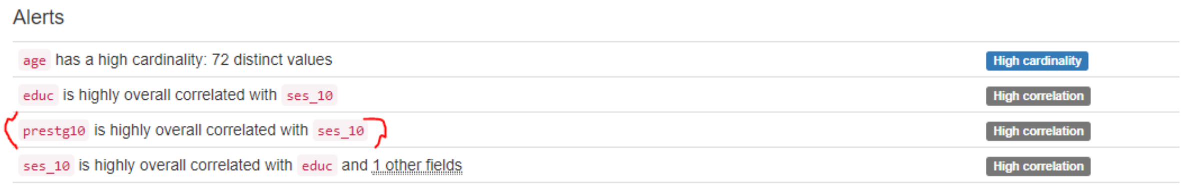

`ann_income` has 196 (8.3%) missing values	Missing
`prestg10` has 100 (4.3%) missing values	Missing
`mat_pres10` has 691 (29.4%) missing values	Missing
`pat_pres10` has 506 (21.6%) missing values	Missing
`ses_10` has 100 (4.3%) missing values	Missing
`satjob` has 609 (25.9%) missing values	Missing
`working_mom_okay` has 798 (34.0%) missing values	Missing
`m_job_w_home` has 803 (34.2%) missing values	Missing
`m_politics` has 849 (36.2%) missing values	Missing
`preschool_working_mom_bad` has 812 (34.6%) missing values	Missing
`men_work_too_much` has 802 (34.2%) missing values	Missing

Part d

Looking through the HTML report you displayed in part a, which two distinct features in the data have the highest correlation? [1 point]

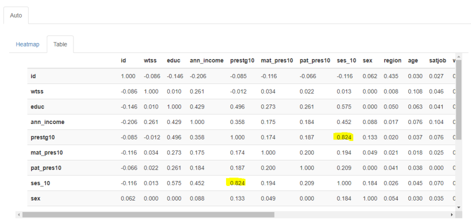
Answer Part d

`prestg10` and `ses_10` have the highest correlation with each other.



Alerts

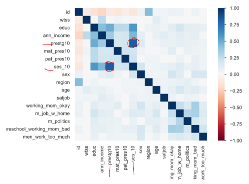
`age` has a high cardinality: 72 distinct values	High cardinality
`educ` is highly overall correlated with `ses_10`	High correlation
`prestg10` is highly overall correlated with `ses_10`	High correlation
`ses_10` is highly overall correlated with `educ` and 1 other fields	High correlation



Auto

Heatmap Table

	id	wtss	educ	ann_income	prestg10	mat_pres10	pat_pres10	ses_10	sex	region	age	satjob
id	1.000	-0.086	-0.146	-0.206	-0.085	-0.116	-0.066	-0.116	0.062	0.435	0.030	0.027
wtss	-0.086	1.000	0.010	0.261	-0.012	0.034	0.022	0.013	0.000	0.008	0.108	0.046
educ	-0.146	0.010	1.000	0.429	0.496	0.273	0.261	0.575	0.000	0.050	0.063	0.041
ann_income	-0.206	0.261	0.429	1.000	0.358	0.175	0.184	0.452	0.088	0.017	0.076	0.104
prestg10	-0.085	-0.012	0.496	0.358	1.000	0.174	0.187	0.824	0.133	0.020	0.037	0.076
mat_pres10	-0.116	0.034	0.273	0.175	0.174	1.000	0.200	0.194	0.049	0.021	0.018	0.025
pat_pres10	-0.066	0.022	0.261	0.184	0.187	0.200	1.000	0.209	0.000	0.041	0.038	0.000
ses_10	-0.116	0.013	0.575	0.452	0.824	0.194	0.209	1.000	0.184	0.026	0.045	0.070
sex	0.062	0.000	0.000	0.088	0.133	0.049	0.000	0.184	1.000	0.054	0.030	0.035



Problem 3

On a primetime show on a 24-hour cable news network, two unpleasant-looking men in suits sit across a table from each other, scowling. One says "This economy is failing the middle-class. The average American today is making less than $48,000 a year." The other screams "Fake news! The typical American makes more than $55,000 a year!" Explain, using words and code, how the data can support both of their arguments. Use the sample weights to calculate descriptive statistics that are more representative of the American adult population as a whole. [1 point]

Answer Problem 3

With the dataset one could get a weighted average of greater than $55,000 a year, but that excludes the NaN values which could have been low or high values, and averages are sensitive to low and high outliers..

One could also get less than $48,000 with the weighted median, which is not sensitive to outliers and is less than$48,000. I would place more more trust in the weighted median descriptive statistic than the weighted mean. Both are calculated below, as are unweighted versions, and trimmed versions of the descriptive statistics.

```
In [ ]: gss_p1.describe()
```

Out[ ]:

	id	wtss	educ	ann_income	prestg10	mat_pres10	pat_pres10
count	2348.000000	2348.000000	2345.000000	2152.000000	2248.000000	1657.000000	1842.000000
mean	1174.500000	1.000000	13.731770	49973.960778	44.682384	41.853953	44.393051
std	677.953538	0.611458	2.974313	42407.211338	13.644987	12.966155	12.971230
min	1.000000	0.471499	0.000000	350.500000	16.000000	16.000000	16.000000
25%	587.750000	0.471499	12.000000	19277.500000	35.000000	31.000000	35.000000
50%	1174.500000	0.942997	14.000000	38555.000000	45.000000	39.000000	44.000000
75%	1761.250000	0.942997	16.000000	70100.000000	53.000000	48.000000	50.000000
max	2348.000000	5.897420	20.000000	158201.841200	80.000000	80.000000	80.000000

In [ ]:
```
gss_p1['ann_income'].mean()
```
Out[ ]:
```
49973.96077843866
```

In [ ]:
```
stats.trim_mean(gss_p1['ann_income'], .15)
```
Out[ ]:
```
49029.57884306569
```

In [ ]:
```
stats.trim_mean(gss_p1['ann_income'], .1)
```
Out[ ]:
```
53441.17147851064
```

In [ ]:
```
gss_p3 = gss_p1.loc[~gss_p1['ann_income'].isna()]
np.average(gss_p3['ann_income'], weights=gss_p3.wtss)
```
Out[ ]:
```
55158.96280421564
```

Medians are not sensitive to outliers in the values, and here I use the weighted median to get a statistic more representative of the American population: **$47,317.50**

In [ ]:
```
weighted.median(gss_p1['ann_income'], weights=gss_p1['wtss'])
```
Out[ ]:
```
47317.5
```

Problem 4

For each of the following parts,

- generate a table that provides evidence about the relationship between the two features in the data that are relevant to each question,
- interpret the table in words,
- use a hypothesis test to assess the strength of the evidence in the table,
- and provide a **specific and accurate** intepretation of the p-value associated with this hypothesis test beyond "significant or not".

Part a

Is there a gender wage gap? That is, is there a difference between the average incomes of men and women? [2 points]

In [ ]:
```
# group by sex and calculate the weighted median of annual income, the max of annual i
def q25(x): return x.quantile(0.25)
def q75(x): return x.quantile(0.75)
def q85(x): return x.quantile(0.85)
gss_p1_f = gss_p1[['sex','ann_income']].query("sex == 'female'").dropna()
print(gss_p1_f.shape)
gss_p1_m = gss_p1[['sex','ann_income']].query("sex == 'male'").dropna()
print(gss_p1_m.shape)

# format like dollars
pd.DataFrame(gss_p1.groupby(['sex']).agg({'ann_income': [q25, 'mean', 'median', q75, q
```

```
(1174, 2)
(978, 2)
```

Out[ ]:

	ann_income						
	q25	mean	median	q75	q85	std	count
sex							
female	16648.75	47191.02	38555.0	57832.5	84120.0	41637.98	1174
male	22782.50	53314.63	38555.0	70100.0	98140.0	43097.03	978

In [ ]:
```
1174 - 978
```

Out[ ]:
```
196
```

In [ ]:
```
53314.63 - 47191.02
```

Out[ ]:
```
6123.610000000001
```

In [ ]:
```
22782.50 - 16648.75
```

Out[ ]:
```
6133.75
```

In [ ]:
```
70100.00 - 57832.50
```

Out[ ]:
```
12267.5
```

In [ ]:
```
98140.00 - 84120.00
```

Out[ ]:
```
14020.0
```

The unweighted average annual income for women in the survery was $6,123.61 less than for men in the survey. Women in the 25th quantile of female respondents ha$ 6,133.75 less than men in the 25th quantile of male respondents. Female respondents in the 75th quantile of salary had salaries $12,267.50 less than men in the 75th quantile of male respondents. The pattern continues forw$

14,020 less than male respondents in the 85th quantile of male responents. There were 196 more female respondents than male respondents. The variability of male salaries was slightly higher than female salaries, but both were highly variable.

It appears that the survey was constructed, or the data provided has equal quantile results for male and female results above the 90th quantile.

"A hypothesis test provides a clear statement about whether enough evidence exists in the data for us to conclude that a relationship is real and not simply a byproduct of randomness." - textbook

Hypothesis: **Women and Men Make the Same Salaries**

This is an **independent samples t-test**.

```
In [ ]: stats.ttest_ind(gss_p1_m['ann_income'], gss_p1_f['ann_income'], equal_var=False)
```

```
Out[ ]: Ttest_indResult(statistic=3.332824087618215, pvalue=0.0008749557881530089)
```

This is under the assumption, or hypothesis, that men and women have the same mean salary. **The p-value is 0.00053**. We reject the null hypothesis and conclude that there is a statisitically significant difference between the annual salaries of men and women in this survey's population.

Part b

Are there different average values of occupational prestige for different levels of job satisfaction? [2 points]

Answer Part b

```
In [ ]: # create these dataframes to use later for t-tests
gss_p1_vs = gss_p1[['satjob','prestg10']].query("satjob == 'very satisfied'").dropna()
print(gss_p1_vs.shape)
gss_p1_ms = gss_p1[['satjob','prestg10']].query("satjob == 'mod. satisfied'").dropna()
print(gss_p1_ms.shape)
gss_p1_ld = gss_p1[['satjob','prestg10']].query("satjob == 'a little dissat'").dropna(
print(gss_p1_ld.shape)
gss_p1_vd = gss_p1[['satjob','prestg10']].query("satjob == 'very dissatisfied'").dropr
print(gss_p1_vd.shape)

print("The overall unweighted mean prestige is: ", gss_p1['prestg10'].mean())
gss_p1b = gss_p1.loc[~gss_p1['prestg10'].isna()]
print("The overall weighted mean prestige is: ", np.average(gss_p1b['prestg10'], weigh

print("The overall weighted median prestige is: ", weighted.median(gss_p1['prestg10'],

df_pres_jobsat = pd.DataFrame(gss_p1.groupby(['satjob']).agg({'prestg10': ['mean', 'me
# explcitly state which order I want the indices to show in
df_pres_jobsat = df_pres_jobsat.reindex(['very satisfied', 'mod. satisfied', 'a little
df_pres_jobsat
```

```
(824, 2)
(639, 2)
(168, 2)
(62, 2)
The overall unweighted mean prestige is:  44.68238434163701
The overall weighted mean prestige is:  44.688924030057294
The overall weighted median prestige is:  46.0
```

Out[ ]:

	prestg10			
	mean	median	std	count
satjob				
very satisfied	46	47	14	824
mod. satisfied	43	42	13	639
a little dissat	41	38	13	168
very dissatisfied	43	39	14	62

The unweighted mean prestige for participants is not correlated with job satisfaction, although participants that reported being 'very satisfied' at their jobs had the highest levels of average occupational presige (46). This is the overall weighted median prestige, which is expected, as the category with the greatest number of survey participants is the 'very satisfied' category.

The mean presitige is lowest (41) for participants who reported being a 'little dissatisfied' with their jobs. Mean prestige for both the 'very dissatisfied' and 'moderately satisfied' was 43.

The median prestige by reported job satisfaction follows a similar pattern, however respondents reporting they are 'moderately satisfied' at their jobs has greater median prestige (42) than very dissatisfied participants (39).

There was similar standard deviation/variance in prestige scores across the job satisfaction categories.

The majority of respondents reported being very satisfied at their workplaces (824), followed by 624 respondents reporting they were moderately satisfied, 168 reporting they were a little dissatisfied, and 62 reporting they were very dissatisfied.

Hypothesis: **Prestige is the same for all levels of job satisfaction**

This is an **Analysis of Variance Test (ANOVA)** and uses **F-test**.

In [ ]:

```python
stats.f_oneway(gss_p1_vs['prestg10'],
               gss_p1_ms['prestg10'],
               gss_p1_ld['prestg10'],
                gss_p1_vd['prestg10'])
```

Out[ ]:

```
F_onewayResult(statistic=12.205403153509735, pvalue=6.676686425029878e-08)
```

In [ ]:

```python
stats.f_oneway(gss_p1_ms['prestg10'],
               gss_p1_ld['prestg10'],
                gss_p1_vd['prestg10'])
```

```
Out[ ]: F_onewayResult(statistic=1.0953326262156533, pvalue=0.33489120488676083)
```

```
In [ ]: gss_p1['satjob'].unique()
```

```
Out[ ]: array(['very satisfied', nan, 'mod. satisfied', 'a little dissat',
               'very dissatisfied'], dtype=object)
```

I ran two ANOVA tests. The first ANOVA test was with the prestige scores for the four job satisfaction categories: 'very satisfied', 'mod. satisfied', 'a little dissat', and 'very dissatisfied'. The second ANOVA test was with only the 'mod. satisfied', 'a little dissat', and 'very dissatisfied', and without the 'very satisfied' category.

The p-value for the first test is **6.68e-8** and so we reject the null hypothesis that job prestige is the same for repondents reporting job satisfaction levels of 'very satisfied', 'mod. satisfied', 'a little dissat', or 'very dissatisfied.' We conclude there is a difference between job prestige for workers with different job satisfaction levels.

The p-vale for the second test is **0.33**, and so we fail to reject a null hypothesis that prestige is the same for respondents reporting that their job satisfaction is 'mod. satisfied', 'a little dissat', or 'very dissatisfied.'

Problem 5

Report the Pearson's correlation between :

1. years of education,
2. socioeconomic status,
3. income,
4. occupational prestige,
5. and a person's mother's
6. and father's occupational prestige?

Then perform a hypothesis test for the correlation between years of education and socioeconomic status and provide a **specific and accurate** intepretation of the p-value associated with this hypothesis test beyond "significant or not". [2 points]

Answer Problem 5

I can use the `.corr()` method. I also do the correlations with p-value for all the features requested, but I only discus the hypothesis test for the correlation between the `educ` and `ses_10` features.

```
In [ ]: gss_p1.columns # remind myself of the column names
```

```
Out[ ]: Index(['id', 'wtss', 'sex', 'educ', 'region', 'age', 'ann_income', 'prestg10',
               'mat_pres10', 'pat_pres10', 'ses_10', 'satjob', 'working_mom_okay',
               'm_job_w_home', 'm_politics', 'preschool_working_mom_bad',
               'men_work_too_much'],
              dtype='object')
```

In [ ]:
```
cols_to_corr = ['educ','ses_10','ann_income','prestg10','mat_pres10','pat_pres10']
```

In [ ]:
```
gss_p1[cols_to_corr].corr()
```

Out[ ]:

	educ	ses_10	ann_income	prestg10	mat_pres10	pat_pres10
educ	1.000000	0.558169	0.389245	0.479933	0.269115	0.261417
ses_10	0.558169	1.000000	0.417210	0.835515	0.203486	0.210451
ann_income	0.389245	0.417210	1.000000	0.340995	0.164881	0.171048
prestg10	0.479933	0.835515	0.340995	1.000000	0.189262	0.192180
mat_pres10	0.269115	0.203486	0.164881	0.189262	1.000000	0.235750
pat_pres10	0.261417	0.210451	0.171048	0.192180	0.235750	1.000000

In [ ]:
```
for col1 in cols_to_corr:
    for col2 in cols_to_corr:
        if col1 != col2:
            # don't wwant to do a col1 col2 combination twice
            if cols_to_corr.index(col1) < cols_to_corr.index(col2):
                print(f'Variable 1: {col1} - Variable 2: {col2}')
                # get a version with no na's in both columns
                gss_corr = gss_p1[[col1, col2]].dropna()
                print(stats.pearsonr(gss_corr[col1], gss_corr[col2]))
                print("-----" * 20)
```

```
Variable 1: educ - Variable 2: ses_10
(0.5581686004626782, 3.719448810018995e-184)
--------------------------------------------------------------------------------
---------------
Variable 1: educ - Variable 2: ann_income
(0.3892454444707866, 9.423778849689854e-79)
--------------------------------------------------------------------------------
---------------
Variable 1: educ - Variable 2: prestg10
(0.4799333122156877, 9.429361735799834e-130)
--------------------------------------------------------------------------------
---------------
Variable 1: educ - Variable 2: mat_pres10
(0.2691149100164562, 7.2087978091475e-29)
--------------------------------------------------------------------------------
---------------
Variable 1: educ - Variable 2: pat_pres10
(0.26141654416322213, 3.9573721192221336e-30)
--------------------------------------------------------------------------------
---------------
Variable 1: ses_10 - Variable 2: ann_income
(0.4172103007077431, 3.4430866124302764e-88)
--------------------------------------------------------------------------------
---------------
Variable 1: ses_10 - Variable 2: prestg10
(0.8355149892995795, 0.0)
--------------------------------------------------------------------------------
---------------
Variable 1: ses_10 - Variable 2: mat_pres10
(0.20348584400028333, 2.1846230684022438e-16)
--------------------------------------------------------------------------------
---------------
Variable 1: ses_10 - Variable 2: pat_pres10
(0.2104509082884091, 3.0106338738339515e-19)
--------------------------------------------------------------------------------
---------------
Variable 1: ann_income - Variable 2: prestg10
(0.34099456629193237, 1.1636229235539331e-57)
--------------------------------------------------------------------------------
---------------
Variable 1: ann_income - Variable 2: mat_pres10
(0.1648814429626584
7, 6.630395980972316e-11)
--------------------------------------------------------------------------------
---------------
Variable 1: ann_income - Variable 2: pat_pres10
(0.17104831301323348, 1.6170955801636064e-12)
--------------------------------------------------------------------------------
---------------
Variable 1: prestg10 - Variable 2: mat_pres10
(0.18926221482010577, 2.4122687791734108e-14)
--------------------------------------------------------------------------------
---------------
Variable 1: prestg10 - Variable 2: pat_pres10
(0.1921796108539416, 2.9803769062466836e-16)
--------------------------------------------------------------------------------
---------------
Variable 1: mat_pres10 - Variable 2: pat_pres10
(0.23575002402833126, 6.043857539301134e-17)
--------------------------------------------------------------------------------
---------------
```

Discussion of the correlation and p-value

The correlation coefficient is **0.558** and the corresponding pvalue is **3.72e-184**. This indicates that there is moderate correlation between years of education and socioeconomic status and that a randomly drawn sample would have almost no probability of producing a correlation as extreme as 0.558 (assuming the correlation in the general population is 0). A p-value of **3.72e-184** means we reject the null hypothesis that education and socioeconomic status are uncorrelatied, and we conclude that there is a moderate correlation between the two features.

Problem 6

Create a new categorical feature for age groups, with categories for 18-35, 36-49, 50-69, and 70 and older (see the module 8 notebook for an example of how to do this).

Then create a cross-tabulation in which the rows represent age groups and the columns represent responses to the statement that "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." Rearrange the columns so that they are in the following order: strongly agree, agree, disagree, strongly disagree. Place row percents in the cells of this table.

Finally, use a hypothesis test that can tell use whether there is enough evidence to conclude that these two features have a relationship, and provide a specific and accurate intepretation of the p-value. [2 points]

```
In [ ]: gss_p6 = gss_p1.copy()
        gss_p6.columns
```

```
Out[ ]: Index(['id', 'wtss', 'sex', 'educ', 'region', 'age', 'ann_income', 'prestg10',
               'mat_pres10', 'pat_pres10', 'ses_10', 'satjob', 'working_mom_okay',
               'm_job_w_home', 'm_politics', 'preschool_working_mom_bad',
               'men_work_too_much'],
              dtype='object')
```

```
In [ ]: gss_p6.age.unique()
```

```
Out[ ]: array(['43', '74', '42', '63', '71', '67', '59', '62', '55', '34', '61',
               '44', '41', '75', '30', '40', '29', '37', '56', '82', '68', '20',
               '89 or older', '60', '65', '45', '50', '52', '46', '53', '22',
               '33', '23', '28', '27', '64', '79', '32', '35', '21', '47', '70',
               '77', '69', '48', '81', '78', '54', '58', '76', '39', '38', '25',
               '49', '18', '19', '26', '57', '51', '36', '72', '24', '88', '66',
               '84', '80', '31', '83', '73', '86', nan, '85', '87'], dtype=object)
```

```
In [ ]: # need to change the '89 or older to '89'
        gss_p6['age'] = gss_p6['age'].str.replace('89 or older', '89')
        # can the column dtype to int
        gss_p6['age'] = gss_p6['age'].astype(float)
```

```
In [ ]: gss_p6.age.unique()
```

Out[ ]:
```
array([43., 74., 42., 63., 71., 67., 59., 62., 55., 34., 61., 44., 41.,
       75., 30., 40., 29., 37., 56., 82., 68., 20., 89., 60., 65., 45.,
       50., 52., 46., 53., 22., 33., 23., 28., 27., 64., 79., 32., 35.,
       21., 47., 70., 77., 69., 48., 81., 78., 54., 58., 76., 39., 38.,
       25., 49., 18., 19., 26., 57., 51., 36., 72., 24., 88., 66., 84.,
       80., 31., 83., 73., 86., nan, 85., 87.])
```

In [ ]:
```
# pd.cut() does what we are looking for - creates categories from break points in a co
#  list of the breakpoints (inclusive for the upper bound but not the lower bound),
gss_p6['age_grp'] = pd.cut(gss_p6['age'], bins=[17,35,49,69,90], labels=("18-35", "36-
```

In [ ]:
```
# show the distribution of the ages in the new age_grp column
gss_p6['age_grp'].value_counts()
```

Out[ ]:
```
50-69           771
18-35           672
36-49           541
70 and older    357
Name: age_grp, dtype: int64
```

In [ ]:
```
gss_p6.query("age_grp == '18-35'")['age'].unique()
```

Out[ ]:
```
array([34., 30., 29., 20., 22., 33., 23., 28., 27., 32., 35., 21., 25.,
       18., 19., 26., 24., 31.])
```

Create the cross tabulation

In [ ]:
```
gss_p6['m_job_w_home'] = gss_p6['m_job_w_home'].astype('category')
```

In [ ]:
```
gss_p6['m_job_w_home'] = gss_p6['m_job_w_home'].cat.reorder_categories(['strongly agre
                                                                        'agree',
                                                                        'disagree',
                                                                        'strongly disag

(pd.crosstab(gss_p6['age_grp'], gss_p6['m_job_w_home'], normalize='index')*100).round(
```

Out[ ]:

m_job_w_home	strongly agree	agree	disagree	strongly disagree
age_grp				
18-35	3.94	14.04	47.54	34.48
36-49	4.79	17.46	46.48	31.27
50-69	4.63	20.85	48.07	26.45
70 and older	11.97	31.66	39.00	17.37

Determine if these two categorical features have a relationship ...

Here I do a groupby just to determine if I have at least 5 samples in each "bucket" for the chi^2 test.

In [ ]:
```
gss_p6.groupby(['age_grp','m_job_w_home']).agg({'wtss': 'count'})
```

Out[ ]:

age_grp	m_job_w_home	wtss
18-35	strongly agree	16
	agree	57
	disagree	193
	strongly disagree	140
36-49	strongly agree	17
	agree	62
	disagree	165
	strongly disagree	111
50-69	strongly agree	24
	agree	108
	disagree	249
	strongly disagree	137
70 and older	strongly agree	31
	agree	82
	disagree	101
	strongly disagree	45

In [ ]:
```
crosstab = pd.crosstab(gss_p6['age_grp'], gss_p6['m_job_w_home'])
stats.chi2_contingency(crosstab.values)
```

Out[ ]:
```
(69.24381761791811,
 2.1419004733989943e-11,
 9,
 array([[ 23.23016905,  81.56957087, 186.89726918, 114.3029909 ],
        [ 20.31209363,  71.32314694, 163.42002601,  99.94473342],
        [ 29.63849155, 104.07152146, 238.45513654, 145.83485046],
        [ 14.81924577,  52.03576073, 119.22756827,  72.91742523]]))
```

The chi^2 is large (69), and there were greater than 5 samples (people) in each of the age groups. The p-value is small, **2.14e-11**, thus state there is a statistically significant relationship between the age groups and the survery question, and we reject the null hypothesis that all age groups equally agree/disagree with the statement, "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." We conclude there is a relationship between the age group and the opinion of women working.

Problem 7

For this problem, you will conduct and interpret a correspondence analysis on the categorical features that ask respondents to state the extent to which they agree or disagree with the statements:

- "A working mother can establish just as warm and secure a relationship with her children as a mother who does not work."
- "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."
- "Most men are better suited emotionally for politics than are most women."
- "A preschool child is likely to suffer if his or her mother works."
- "Family life often suffers because men concentrate too much on their work."

Part a

Conduct a correspondence analysis using the observed features listed above that measures two latent features. Plot the two latent categories for each category in each of the features used in the analysis. [2 points]

In [ ]:
```
gss_p6.columns
```

Out[ ]:
```
Index(['id', 'wtss', 'sex', 'educ', 'region', 'age', 'ann_income', 'prestg10',
       'mat_pres10', 'pat_pres10', 'ses_10', 'satjob', 'working_mom_okay',
       'm_job_w_home', 'm_politics', 'preschool_working_mom_bad',
       'men_work_too_much', 'age_grp'],
      dtype='object')
```

In [ ]:
```
gss_cat = gss_p6[['working_mom_okay','m_job_w_home','m_politics','preschool_working_mo
print(gss_cat.shape)
gss_cat.head()
```

```
(1454, 5)
```

Out[ ]:

	working_mom_okay	m_job_w_home	m_politics	preschool_working_mom_bad	men_work_too_much
0	strongly agree	disagree	agree	strongly disagree	agree
2	strongly agree	disagree	disagree	disagree	disagree
3	agree	disagree	disagree	disagree	neither agree nor disagree
5	strongly agree	disagree	disagree	disagree	agree
8	disagree	strongly disagree	disagree	agree	agree

In [ ]:
```
#pip install prince==0.8.3
```

In [ ]:
```
prince_mca = prince.MCA(n_components=2)
prince_mca = prince_mca.fit(gss_cat)
#gss_mca = prince_mca.transform(gss_cat)
prince_mca.row_coordinates(gss_cat)
```

Out[ ]:

	0	1
0	-0.202209	0.338287
2	-0.423361	-0.316906
3	-0.195576	-0.648698
5	-0.240092	-0.298094
8	0.341540	0.091189
...	...	...
2341	1.219019	0.567439
2343	-0.521777	0.384969
2344	-0.423361	-0.316906
2346	1.076900	0.642127
2347	1.440615	2.529640

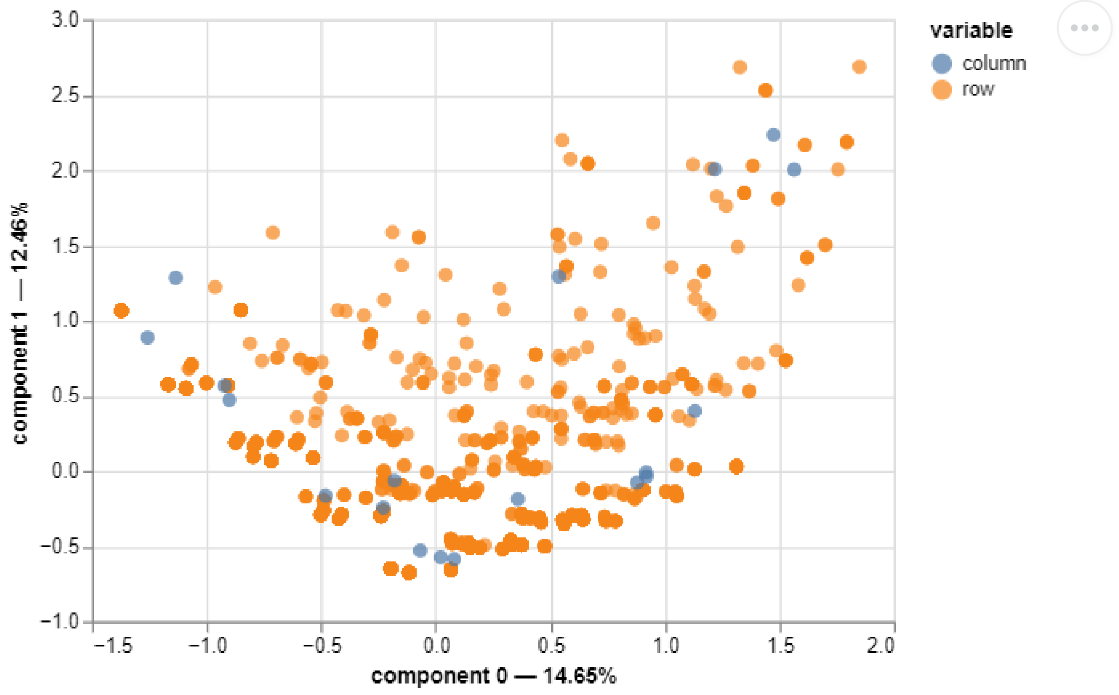
1454 rows × 2 columns

In [ ]:
```
import matplotlib.pyplot as plt
```

In [ ]:
```
plt = prince_mca.plot(
    gss_cat,
    x_component=0,
    y_component=1
)

plt
```

Out[ ]:



Part b

Display the latent features for every category in the observed features, sorted by the first latent feature. Describe in words what concept this feature is attempting to measure, and give the feature a name. [2 points]

```
In [ ]: prince_mca.column_coordinates(gss_cat).sort_values(0)
```

Out[ ]:

	0	1
preschool_working_mom_bad_strongly disagree	-1.258059	0.886694
men_work_too_much_strongly disagree	-1.135403	1.283829
m_job_w_home_strongly disagree	-0.922034	0.566808
working_mom_okay_strongly agree	-0.901119	0.472183
men_work_too_much_neither agree nor disagree	-0.480746	-0.163826
men_work_too_much_disagree	-0.228690	-0.242582
m_politics_disagree	-0.180400	-0.063736
preschool_working_mom_bad_disagree	-0.067886	-0.529257
m_job_w_home_disagree	0.022160	-0.572470
working_mom_okay_agree	0.080484	-0.586395
men_work_too_much_agree	0.358280	-0.187027
men_work_too_much_strongly agree	0.536779	1.292006
m_job_w_home_agree	0.878984	-0.076584
working_mom_okay_disagree	0.918041	-0.010326
preschool_working_mom_bad_agree	0.919993	-0.036429
m_politics_agree	1.131107	0.399623
working_mom_okay_strongly disagree	1.218706	2.005406
preschool_working_mom_bad_strongly agree	1.474183	2.233948
m_job_w_home_strongly agree	1.564722	2.002702

Answer Problem 7 Partb

The first latest feature appears to be a scale from negative values strongly disagreeing that men should be the only people in the work force and women should be home, and also some disagreement that men are better suited to politics, and the positive values at the opposite end of the spectrum. This appears to be a latent feature representing the degree of conservative thought on women in the work force. I will name this feature **working_women**.

Part c

We can use the results of the MCA model to conduct some cool EDA. For one example, follow these steps:

1. Use the `.row_coordinates()` method to calculate values of the latent feature for every row in the data you passed to the MCA in part a. Extract the first column and store it in its own dataframe.
2. To join it with the full, cleaned GSS data based on row numbers (instead of on a primary key), use the `.join()` method. For example, if we named the cleaned GSS data `gss_clean` and if we named the dataframe in step 1 `latentfeature`, we can type

   ```
   gss_clean = gss_clean.join(latentfeature, how="outer")
   ```
3. Create a cross-tabuation with age categories (that you constructed in problem 5) in the rows and sex in the columns. Instead of a frequency, place the mean value of the latent feature in the cells.

What does this table tell you about the relationship between sex, age, and the latent feature? [2 points]

In [ ]:
```
#create the new dataframe with the latent feature from the first MCA component
df_workingwomen = prince_mca.row_coordinates(gss_cat)[[0]]
df_workingwomen.columns = ['working_women']
df_workingwomen
```

Out[ ]:

	working_women
0	-0.202209
2	-0.423361
3	-0.195576
5	-0.240092
8	0.341540
...	...
2341	1.219019
2343	-0.521777
2344	-0.423361
2346	1.076900
2347	1.440615

1454 rows × 1 columns

In [ ]:
```
# concatenate the column to to the original dataframe to a new dataframe and take a lo
# DID NOT USE CONCAT METHOD
#df_gss_p7c = pd.concat([gss_p6, df_workingwomen], axis=1)

# DID USE THE JOIN METHOD
# oops, you specified that you wanted us to use the join method, here is the join meth
df_gss_p7_join = gss_p6.join(df_workingwomen, how="outer")
df_gss_p7_join
```

Out[ ]:

	id	wtss	sex	educ	region	age	ann_income	prestg10	mat_pres10	pat_pres10	s
0	1	2.357493	male	14.0	new england	43.0	NaN	47.0	31.0	45.0	
1	2	0.942997	female	10.0	new england	74.0	22782.5000	22.0	32.0	39.0	
2	3	0.942997	male	16.0	new england	42.0	112160.0000	61.0	32.0	72.0	
3	4	0.942997	female	16.0	new england	63.0	158201.8412	59.0	NaN	39.0	
4	5	0.942997	male	18.0	new england	71.0	158201.8412	53.0	35.0	45.0	
...	...	...	...	...	...	...	...	...	...	...	
2343	2344	0.471499	female	12.0	new england	37.0	NaN	47.0	31.0	72.0	
2344	2345	0.942997	female	12.0	new england	75.0	22782.5000	28.0	NaN	27.0	
2345	2346	0.942997	female	12.0	new england	67.0	70100.0000	40.0	45.0	53.0	
2346	2347	0.942997	male	16.0	new england	72.0	38555.0000	47.0	53.0	50.0	
2347	2348	0.471499	female	12.0	new england	79.0	NaN	33.0	NaN	46.0	

2348 rows × 19 columns

Now, create the cross-tab:

1. with age categories (that you constructed in problem 5) in the rows
2. and sex in the columns.
3. Instead of a frequency, place the mean value of the latent feature in the cells.

In [ ]:

```
df_p7_ct = df_gss_p7_join.groupby(['age_grp','sex']).agg({'working_women': 'mean'})
df_p7_ct
```

Out[ ]:

		working_women
age_grp	**sex**	
18-35	**female**	-0.241140
	male	-0.003774
36-49	**female**	-0.137001
	male	-0.000686
50-69	**female**	-0.125059
	male	0.222532
70 and older	**female**	0.129256
	male	0.473005

In [ ]:
```python
df_p7_ct_neg = df_p7_ct.copy()
df_p7_ct_neg['working_women'] = -df_p7_ct_neg['working_women']
df_p7_ct_neg.reset_index(inplace=True)
df_p7_ct_neg
```

Out[ ]:

	age_grp	sex	working_women
0	18-35	female	0.241140
1	18-35	male	0.003774
2	36-49	female	0.137001
3	36-49	male	0.000686
4	50-69	female	0.125059
5	50-69	male	-0.222532
6	70 and older	female	-0.129256
7	70 and older	male	-0.473005

Interpretation

I interpret this cross tab as saying that younger survery respondents had stronger agreement that women belonged in the workplace, epsecially female respondents. It is interesting that men, aged 18 - 50 were almost ambivalent about whether women were in the workforce according to this latent feature.

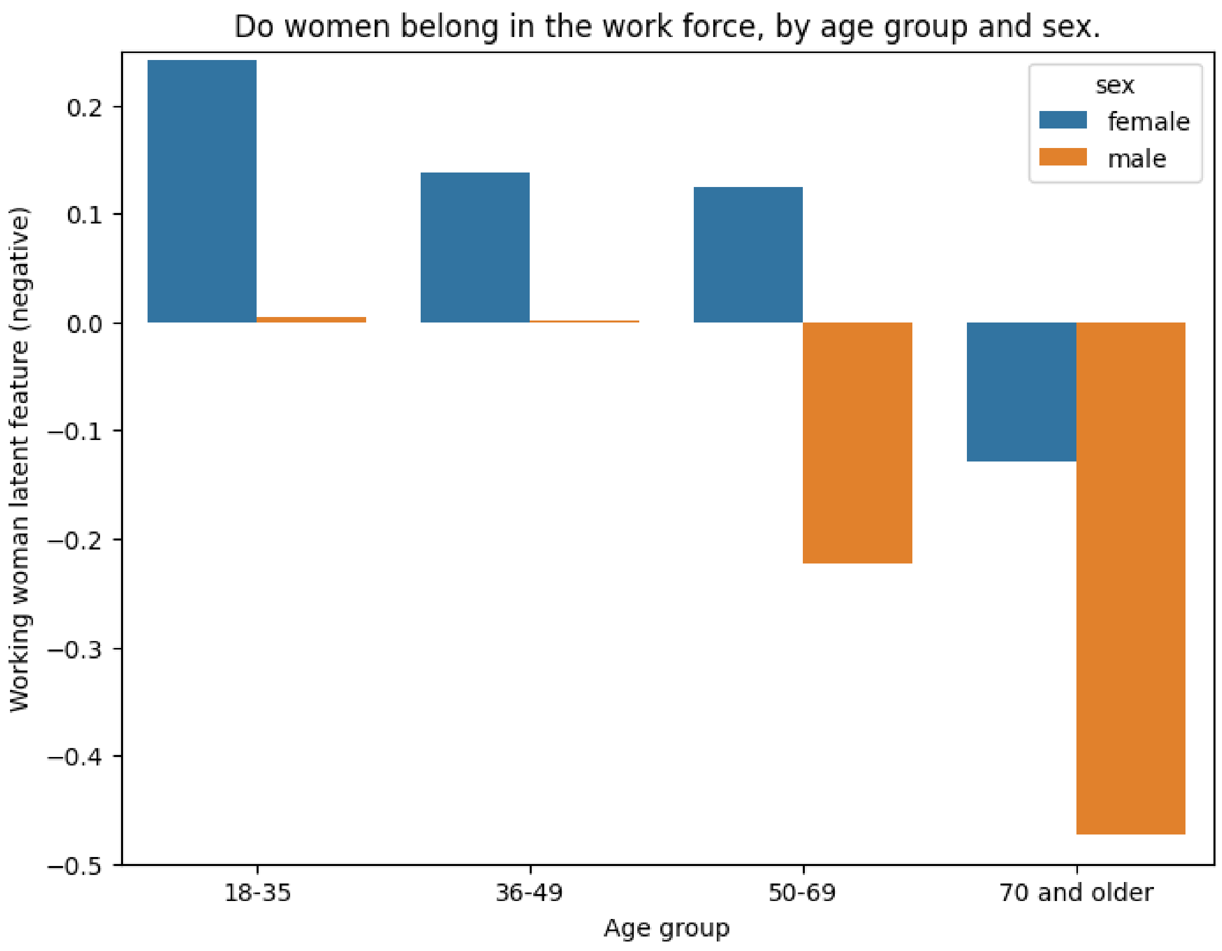
I also created a visual, by taking the negative of the latent feature, and plotting the values.

In [ ]:
```python
import seaborn as sns
import matplotlib.pyplot as plt
```

In [ ]:
```python
plt.figure(figsize=(8, 6))
myplt = sns.barplot(x='age_grp', y='working_women', hue='sex', data=df_p7_ct_neg)
# set the y limit
plt.ylim([-.5,.25])
plt.ylabel('Working woman latent feature (negative)')
plt.xlabel('Age group')
plt.title('Do women belong in the work force, by age group and sex.')
```

Out[ ]:

```
Text(0.5, 1.0, 'Do women belong in the work force, by age group and sex.')
```



In [ ]: