Midterm Part 2 - Fides Regina Schwartz

Question 1 - import data

In [ ]:
```
import pandas as pd
from scipy import stats
import statsmodels.api as sm

#Load data
acs = pd.read_stata("C:/Users/dm93/Downloads/US_ACS_2017/US_ACS_2017.dta")
```

In [ ]:
```
# Look at data
acs.head()
```

Out[ ]:

	year	datanum	serial	cbserial	numprec	subsamp	hhwt	hhtype	repwt	cluster	...	repwtp71	repwtp72	repwtp73	rep
0	2017	1	1	2.017000e+12	1 person record	26	206	male householder, living alone	1	2.017000e+12	...	60	46	260	
1	2017	1	2	2.017000e+12	1 person record	76	45	female householder, living alone	1	2.017000e+12	...	55	31	11	
2	2017	1	3	2.017000e+12	3	2	136	married-couple family household	1	2.017000e+12	...	101	47	160	
3	2017	1	3	2.017000e+12	3	2	136	married-couple family household	1	2.017000e+12	...	147	49	213	
4	2017	1	3	2.017000e+12	3	2	136	married-couple family household	1	2.017000e+12	...	62	36	128	

5 rows × 266 columns

Question 2 - Thin your data to these variables: age, empstat, inctot, educd, statefip, countyfip, sex, race, hispan.

In [ ]:
```
#create a copy of acs with only the needed variables
acs_copy = acs[['age', 'empstat', 'inctot', 'educd', 'statefip', 'countyfip', 'sex', 'race', 'hispan']]
```

In [ ]:
```
#Check that this worked
acs_copy.head()
```

Out[ ]:

	age	empstat	inctot	educd	statefip	countyfip	sex	race	hispan
0	73	not in labor force	10000	grade 7	alabama	0	male	black/african american/negro	not hispanic
1	31	employed	38500	bachelor's degree	alabama	0	female	white	not hispanic
2	41	employed	82000	ged or alternative credential	alabama	0	male	white	mexican
3	48	not in labor force	8700	regular high school diploma	alabama	0	female	white	not hispanic
4	16	not in labor force	0	grade 10	alabama	0	male	white	mexican

Question 3 - Now create an indicator variable for whether a respondent identifies as a Person of Color, which means anyone who is not both White (according to race) and non-Hispanic (according to hispan).

In [ ]:
```
#check data type of race column
acs_copy["race"].dtype
```

Out[ ]:
```
CategoricalDtype(categories=['white', 'black/african american/negro',
                  'american indian or alaska native', 'chinese', 'japanese',
                  'other asian or pacific islander', 'other race, nec',
                  'two major races', 'three or more major races'],
, ordered=True)
```

In [ ]:
```
#check data type of hispan column
acs_copy['hispan'].dtype
```

Out[ ]:
```
CategoricalDtype(categories=['not hispanic', 'mexican', 'puerto rican', 'cuban', 'other'], ordered=True)
```

In [ ]:
```
# create indicator variable for person of colour
import numpy as np

acs_copy["poc"] = (acs_copy["race"] != "white") | (acs_copy["hispan"] != "not hispanic")
acs_copy.head()
```

```
C:\Users\dm93\AppData\Local\Temp/ipykernel_24892/334483827.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-v
iew-versus-a-copy
  acs_copy["poc"] = (acs_copy['race'] != 'white' ) | (acs_copy['hispan'] != 'not hispanic')
```

Out[ ]:

	age	empstat	inctot	educd	statefip	countyfip	sex	race	hispan	poc
0	73	not in labor force	10000	grade 7	alabama	0	male	black/african american/negro	not hispanic	True
1	31	employed	38500	bachelor's degree	alabama	0	female	white	not hispanic	False
2	41	employed	82000	ged or alternative credential	alabama	0	male	white	mexican	True
3	48	not in labor force	8700	regular high school diploma	alabama	0	female	white	not hispanic	False
4	16	not in labor force	0	grade 10	alabama	0	male	white	mexican	True

Question 4 The ACS surveys everyone, but we're only interested in people who are "in the labor force", which means they are either employed or seeking employment. Use the empstat variable to restrict your sample to people who are employed or unemployed, excluding anyone who doesn't answer or who aren't in the labor force.

In [ ]:
```
acs_copy['empstat'].dtype
```

Out[ ]:
```
CategoricalDtype(categories=['n/a', 'employed', 'unemployed', 'not in labor force'], ordered=True)
```

In [ ]:
```
acs_copy = acs_copy.loc[
    ~((acs_copy["empstat"] == "n/a") | (acs_copy["empstat"] == "not in labor force")), :
]
# df_new = df.drop(df[(df['col_1'] == 1.0) & (df['col_2'] == 0.0)].index)
acs_copy.head()
```

Out[ ]:

	age	empstat	inctot	educd	statefip	countyfip	sex	race	hispan	poc
1	31	employed	38500	bachelor's degree	alabama	0	female	white	not hispanic	False
2	41	employed	82000	ged or alternative credential	alabama	0	male	white	mexican	True
5	37	employed	18300	regular high school diploma	alabama	0	female	black/african american/negro	not hispanic	True
10	32	employed	65000	regular high school diploma	alabama	0	male	white	mexican	True
11	54	employed	57000	associate's degree, type not specified	alabama	0	female	black/african american/negro	not hispanic	True

In [ ]:

```
acs_copy['empstat'].dtype
```

Out[ ]:

```
CategoricalDtype(categories=['n/a', 'employed', 'unemployed', 'not in labor force'], ordered=True)
```

In [ ]:

```
# Check that we don't have n/a or not in labor force anymore
acs_copy['empstat'].value_counts()
```

Out[ ]:

```
employed              1488986
unemployed              76387
n/a                         0
not in labor force          0
Name: empstat, dtype: int64
```

In [ ]:

```
acs_copy['empstat'].dtype
```

Out[ ]:

```
CategoricalDtype(categories=['n/a', 'employed', 'unemployed', 'not in labor force'], ordered=True)
```

In [ ]:

```
#acs_copy = acs_copy.assign(employment=acs_copy['empstat'].replace({'employed': '0'}))
#acs_copy.sample(10)
```

In [ ]:

```
# acs_copy = acs_copy.assign(employment1=acs_copy['empstat'].replace({'unemployed': 1}))
# acs_copy.sample(25)
```

In [ ]:

```
#acs_copy['employment'].dtype
```

In [ ]:
```
#acs_copy['employment'] = acs_copy['employment'].astype(int)
```

Question 5 - Also restrict attention to people who are at least 25. Note this may require some cleaning of the “age” variable.

In [ ]:
```
#check category of age
acs_copy["age"].dtype
```

Out[ ]:
```
CategoricalDtype(categories=['less than 1 year old', '1', '2', '3', '4', '5', '6', '7',
                  '8', '9', '10', '11', '12', '13', '14', '15', '16', '17',
                  '18', '19', '20', '21', '22', '23', '24', '25', '26', '27',
                  '28', '29', '30', '31', '32', '33', '34', '35', '36', '37',
                  '38', '39', '40', '41', '42', '43', '44', '45', '46', '47',
                  '48', '49', '50', '51', '52', '53', '54', '55', '56', '57',
                  '58', '59', '60', '61', '62', '63', '64', '65', '66', '67',
                  '68', '69', '70', '71', '72', '73', '74', '75', '76', '77',
                  '78', '79', '80', '81', '82', '83', '84', '85', '86', '87',
                  '88', '89', '90 (90+ in 1980 and 1990)', '91', '92', '93',
                  '94', '95', '96'],
, ordered=True)
```

In [ ]:
```
# turn string into variables
acs_copy["age"] = acs_copy["age"].str.replace("less than 1 year old", "0")
acs_copy["age"] = acs_copy["age"].str.replace("90 \(90\+ in 1980 and 1990\)", "90")
acs_copy.sample(10)
```

```
C:\Users\dm93\AppData\Local\Temp/ipykernel_24892/2816797171.py:3: FutureWarning: The default value of regex will change f
rom True to False in a future version.
  acs_copy['age'] = acs_copy['age'].str.replace("90 \(90\+ in 1980 and 1990\)", '90')
```

Out[ ]:

	age	empstat	inctot	educd	statefip	countyfip	sex	race	hispan	poc
8506	41	employed	60000	master's degree	alabama	0	female	black/african american/negro	not hispanic	True
2524941	53	employed	40000	bachelor's degree	south carolina	0	female	other asian or pacific islander	not hispanic	True
1964766	67	employed	98700	master's degree	new york	0	male	white	not hispanic	False
1480082	38	employed	52000	bachelor's degree	michigan	125	female	other race, nec	other	True

	age	empstat	inctot	educd	statefip	countyfip	sex	race	hispan	poc
836128	22	employed	6400	regular high school diploma	florida	33	female	other asian or pacific islander	not hispanic	True
1643535	46	employed	20000	1 or more years of college credit, no degree	missouri	0	male	white	not hispanic	False
512461	54	employed	46000	regular high school diploma	california	85	female	other asian or pacific islander	not hispanic	True
393451	29	employed	7000	ged or alternative credential	california	37	female	other race, nec	other	True
1497503	55	employed	190000	bachelor's degree	michigan	0	male	white	not hispanic	False
3089593	60	employed	30000	bachelor's degree	washington	11	male	white	not hispanic	False

In [ ]:
```
acs_copy['age'].value_counts()
```

Out[ ]:
```
54    36059
55    35903
53    35460
56    35096
52    34730
      ...  
92       81
95       59
93       57
91       30
96        3
Name: age, Length: 81, dtype: int64
```

In [ ]:
```
acs_copy["age"].dtype
```

Out[ ]:
```
dtype('O')
```

In [ ]:
```
# Check if there is missing data in the age variable
acs_copy['age'].isna().values.any()
```

```
False
```

```
Out[ ]:
```

```
In [ ]: acs_copy['age'].astype(int).dtype
```

```
Out[ ]: dtype('int32')
```

```
In [ ]: # Keep only people over the age of 25
        acs_copy = acs_copy.loc[acs["age"] >= 25]
        acs_copy.head()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_24892/542161771.py in <module>
      1 # Keep only people over the age of 25
----> 2 acs_copy = acs_copy.loc[acs['age'] >= 25]
      3 acs_copy.head()

~\miniconda3\lib\site-packages\pandas\core\ops\common.py in new_method(self, other)
     67         other = item_from_zerodim(other)
     68 
---> 69         return method(self, other)
     70 
     71     return new_method

~\miniconda3\lib\site-packages\pandas\core\arraylike.py in __ge__(self, other)
     50     @unpack_zerodim_and_defer("__ge__")
     51     def __ge__(self, other):
---> 52         return self._cmp_method(other, operator.ge)
     53 
     54     # -------------------------------------------------------------

~\miniconda3\lib\site-packages\pandas\core\series.py in _cmp_method(self, other, op)
   5500 
   5501         with np.errstate(all="ignore"):
-> 5502             res_values = ops.comparison_op(lvalues, rvalues, op)
   5503 
   5504         return self._construct_result(res_values, name=res_name)

~\miniconda3\lib\site-packages\pandas\core\ops\array_ops.py in comparison_op(left, right, op)
    268     ):
    269         # Call the method on lvalues
--> 270         res_values = op(lvalues, rvalues)
```

```
    271
    272     elif is_scalar(rvalues) and isna(rvalues):

~\miniconda3\lib\site-packages\pandas\core\ops\common.py in new_method(self, other)
     67         other = item_from_zerodim(other)
     68 
---> 69         return method(self, other)
     70 
     71     return new_method

~\miniconda3\lib\site-packages\pandas\core\arrays\categorical.py in func(self, other)
    173                 return ret
    174             else:
--> 175                 return ops.invalid_comparison(self, other, op)
    176         else:
    177             # allow categorical vs object dtype array comparisons for equality

~\miniconda3\lib\site-packages\pandas\core\ops\invalid.py in invalid_comparison(left, right, op)
     32     else:
     33         typ = type(right).__name__
---> 34         raise TypeError(f"Invalid comparison between dtype={left.dtype} and {typ}")
     35     return res_values
     36 

TypeError: Invalid comparison between dtype=category and int
```

In [ ]:

```python
import numpy as np

# Create column with squared ages
acs_copy["age-squared"] = acs_copy["age"]
acs_copy["age-squared"] = np.square(acs_copy["age-squared"])
# acs_copy.value_counts(dropna=False)
acs_copy.head()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_24892/4223549050.py in <module>
      2 # Create column with squared ages
      3 acs_copy['age-squared'] = acs_copy['age']
----> 4 acs_copy['age-squared'] = np.square(acs_copy['age-squared'])
      5 #acs_copy.value_counts(dropna=False)
      6 acs_copy.head()

~\miniconda3\lib\site-packages\pandas\core\generic.py in __array_ufunc__(self, ufunc, method, *inputs, **kwargs)
```

```
   2030         self, ufunc: np.ufunc, method: str, *inputs: Any, **kwargs: Any
   2031     ):
-> 2032         return arraylike.array_ufunc(self, ufunc, method, *inputs, **kwargs)
   2033 
   2034     # ideally we would define this to avoid the getattr checks, but

~\miniconda3\lib\site-packages\pandas\core\arraylike.py in array_ufunc(self, ufunc, method, *inputs, **kwargs)
    362         # ufunc(series, ...)
    363         inputs = tuple(extract_array(x, extract_numpy=True) for x in inputs)
--> 364         result = getattr(ufunc, method)(*inputs, **kwargs)
    365     else:
    366         # ufunc(dataframe)

TypeError: can't multiply sequence by non-int of type 'str'
```

Question 6 - Create a categorical variable that identifies whether a person has (a) high school diploma or equivalent (a ged or alternative credential), (b) an undergraduate degree or better, or (c) neither using the information in educd. Note that a college degree in the US may be called either a Bachelors Degree or an Associates Degree. You may assume anyone in college has a high school diploma, and anyone with an advanced degree (masters degree or doctoral degree) has a college degree.

In [ ]:

```
acs_copy["educd"].dtype
```

Out[ ]:

```
CategoricalDtype(categories=['n/a', 'no schooling completed', 'nursery school, preschool',
                  'kindergarten', 'grade 1', 'grade 2', 'grade 3', 'grade 4',
                  'grade 5', 'grade 6', 'grade 7', 'grade 8', 'grade 9',
                  'grade 10', 'grade 11', '12th grade, no diploma',
                  'regular high school diploma',
                  'ged or alternative credential',
                  'some college, but less than 1 year',
                  '1 or more years of college credit, no degree',
                  'associate's degree, type not specified',
                  'bachelor's degree', 'master's degree',
                  'professional degree beyond a bachelor's degree',
                  'doctoral degree'],
, ordered=True)
```

In [ ]:

```
# Add educational indicators
# a) high school diploma or equivalent (a ged or alternative credential counts),
# b) an undergraduate degree or better, or
# c) the person has neither a high school diploma, nor a high school diploma equivalent, nor an undergraduate degree.
```

```
acs_copy = acs_copy.assign(
    education=acs_copy["educd"].replace({"regular high school diploma": 1})
)
acs_copy = acs_copy.assign(
    education=acs_copy["educd"].replace({"ged or alternative credential": 1})
)
acs_copy = acs_copy.assign(
    education=acs_copy["educd"].replace({"some college, but less than 1 year": 1})
)
acs_copy = acs_copy.assign(
    education=acs_copy["educd"].replace(
        {"1 or more years of college credit, no degree": 1}
    )
)
acs_copy = acs_copy.assign(
    education=acs_copy["educd"].replace({"associate's degree, type not specified": 2})
)
acs_copy = acs_copy.assign(
    education=acs_copy["educd"].replace({"bachelor's degree": 2})
)
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"master's degree": 2}))
acs_copy = acs_copy.assign(
    education=acs_copy["educd"].replace(
        {"professional degree beyond a bachelor's degree": 2}
    )
)
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"doctoral degree": 2}))
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"kindergarten": 3}))
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"grade 1": 3}))
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"grade 2": 3}))
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"grade 3": 3}))
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"grade 4": 3}))
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"grade 5": 3}))
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"grade 6": 3}))
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"grade 7": 3}))
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"grade 8": 3}))
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"grade 9": 3}))
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"grade 10": 3}))
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"grade 11": 3}))
acs_copy = acs_copy.assign(
    education=acs_copy["educd"].replace({"12th grade, no diploma": 3})
)
acs_copy = acs_copy.assign(
    education=acs_copy["educd"].replace({"no schooling completed": 3})
)
```

```
acs_copy = acs_copy.assign(
    education=acs_copy["educd"].replace({"nursery school, preschool": 3})
)
acs_copy = acs_copy.assign(education=acs_copy["educd"].replace({"n/a": 3}))

acs_copy.sample(10)
```

Out[ ]:

	age	empstat	inctot	educd	statefip	countyfip	sex	race	hispan	poc	age-squared	education
1385102	69	employed	52000	regular high school diploma	massachusetts	0	male	white	not hispanic	False	69	regular high school diploma
1082651	33	employed	38600	1 or more years of college credit, no degree	illinois	0	female	white	not hispanic	False	33	1 or more years of college credit, no degree
1854488	61	employed	25500	bachelor's degree	new jersey	3	female	other asian or pacific islander	not hispanic	True	61	bachelor's degree
1987210	38	employed	35000	bachelor's degree	new york	91	male	white	not hispanic	False	38	bachelor's degree
2258019	29	unemployed	0	regular high school diploma	ohio	0	female	white	not hispanic	False	29	regular high school diploma
243726	43	employed	115000	master's degree	california	29	male	other asian or pacific islander	not hispanic	True	43	master's degree
2133743	52	employed	45000	bachelor's degree	north carolina	119	female	other asian or pacific islander	not hispanic	True	52	bachelor's degree
992331	23	employed	11400	regular high school diploma	illinois	119	male	black/african american/negro	not hispanic	True	23	regular high school diploma
1449438	57	unemployed	0	regular high school diploma	massachusetts	25	male	black/african american/negro	not hispanic	True	57	regular high school diploma

	age	empstat	inctot	educd	statefip	countyfip	sex	race	hispan	poc	age-squared	education
2116611	53	employed	34500	associate's degree, type not specified	north carolina	0	male	white	not hispanic	False	53	associate's degree, type not specified

In [ ]:

```
acs_copy["education"] = acs_copy["education"].astype(str).astype(int)
```

In [ ]:

```
acs_copy["education"].dtype
```

Question 7 - Now look at the simple relationship between these three educational attainment levels and the likelihood of being employed in a linear probability model (i.e. use a linear regression). Also include age and age-squared as controls. How much more likely is someone to be employed if they have a high school degree (as opposed to no degrees or diplomas)? Is that difference statistically significant? What is the t-value associated with that difference?

In [ ]:

```
import statsmodels.formula.api as smf

model = smf.ols("C(employment) ~ C(education) + age + age-squared", acs_copy).fit()
# model.get_robustcov_results("HC3").summary()
model.summary()
```

Question 8 - Now add an indicator for whether the respondent is a person of color. What effect does that have on the apparent treatment effect of getting a high school diploma (as compared to not having any degree).

In [ ]:

```
model2 = smf.ols(
    "C(employment) ~ C(education) + age + age-squared + C(poc)", acs_copy
).fit()
model2.get_robustcov_results("HC3").summary()
```

Question 9 - Given our formula for omitted variable bias in a simple regression like this, and given that the coefficient on PoC is negative and the coefficient on having a high school

diploma has changed, what does that tell you about the correlation between being a PoC and getting a high school degree in the US?

This tells me that being a person of color in the US reduces the likelihood of attaining a high-school diploma.

Question 10 - Using the language of potential outcomes, explain what you have learned about why the estimate of the effect of having a high school diploma in your simple regression (without a variable for whether the respondent was a Person of Color) was wrong? i.e. what assumption required for estimating a causal effect do we know was violated, and how was it violated?

The assumption of "no baseline differences" was violated in this model that did not include the factor of being a person of color. Since there are differences in the likelihood of attaining a high school diploma based on whether someone is a person of color or not, this baseline difference needs to be taken into account in our model.

Question 11 - Now, using our regression of employment status on our three income categories, age, age-squared, and whether the respondent is a PoC, much more likely is someone to be employed if they have a college degree as opposed to a high school diploma or equivalent? Is that difference statistically significant? Please provide the difference and t-value if you can; if not, at least provide the difference.

In [ ]:

```
model3 = smf.ols(
    "empstat ~ C(employment) + poc + age + age_squared + C(poc)", acs_copy
).fit()
model3.get_robustcov_results("HC3").summary()
```

I don't know what happened between yesterday and today with both the age variable and the employment variable. I cannot figure out why it's not treating age as an integer and what the problem is with replacing the strings with numbers (it worked yesterday). I will submit this and try and follow-up later, what I did wrong.