Lab 8: Estimating Causal Effects Using Unconfoundedness

Welcome to the eighth DS102 lab!

The goals of this lab is to implement and better understand causal inference in observational studies using the unconfoundedness assumption.

The code you need to write is commented out with a message "TODO: fill in".

Collaboration Policy

Data science is a collaborative activity. While you may talk with others about the labs, we ask that you **write your solutions individually**. If you do discuss the assignments with others please **include their names** in the cell below.

Gradescope Submission

To submit this assignment, rerun the notebook from scratch (by selecting Kernel > Restart & Run all), and then print as a pdf (File > download as > pdf) and submit it to Gradescope.

This assignment should be completed and submitted before Wednesday November 3rd, 2021, at 11:59 PM PST.

Collaborators

Write the names of your collaborators in this cell.

```
<Collaborator Name> <Collaborator e-mail>
```

In [6]:

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import statsmodels.api as sm
import seaborn as sns
import itertools


import hashlib

sns.set(style="dark")
plt.style.use("ggplot")
%matplotlib inline
```

Causal Inference Background and Review

In the last lab, you saw how we could use instrumental variables to identify a causal effect from observational study data. But in many cases, it may not be so easy to find a good instrumental variable. In this lab, we'll explore other ways to identify causal effects.

Potential Outcomes and Average Treatment Effect

In general, we can measure the causal effect of a binary treatment Z on an outcome Y by considering the potential outcomes $Y(0)$ and $Y(1)$. Recall that these are *potential* outcomes: they represent thought experiments about what would happen if the treatment was or wasn't applied. In the real world, we only ever get to observe one of them for any individual, depending on whether that unit received the treatment or not.

We defined the average treatment effect (ATE, represented by the Greek letter τ) as:

$$\tau = E[Y(1) - Y(0)]$$

This represents the causal effect of a treatment Z on an outcome Y. We saw that in general, we were unable to compute this without making assumptions. If our data come from a randomized experiment, then we saw that the difference in group means (SDO) was an unbiased estimate of the ATE:

$$\hat{\tau} = \underbrace{\frac{1}{n_1}\sum_{i:Z_i=1} Y_i}_{\text{mean of treatment group}} - \underbrace{\frac{1}{n_0}\sum_{i:Z_i=0} Y_i}_{\text{mean of control group}}$$

Independence and Unconfoundedness

Recall that in a randomized experiment, we make treatment decisions completely at random. This prevents the treatment from being confounded by any external factors. Unfortunately, in an observational study, we often must deal with confounders: variables that have a causal effect on both the treatment and the outcome. Mathematically, in a randomized experiment, we say that

$$Z_i \perp\!\!\!\perp \big(Y_i(0), Y_i(1)\big) \ \forall i$$

meaning that knowing any unit's treatment status doesn't give us any additional information about the distribution of their potential outcomes ("what-ifs"). For example, in a drug trial, because of randomization, the people who receive the drug have the same (distribution of) potential outcomes as the people who receive a placebo, since there are no systematic differences between the treatment and control groups.

In an observational study, this usually isn't true. For example, suppose we are interested in the effect of a job training program on income. People who receive the job training program might be poorer than people who don't, and so whether they receive the training or not, their incomes might be lower. In this case, the treatment variable (job training program) gives us information about both potential outcomes (income with the program, and income without the program), because of the confounding effect of socioeconomic status (and other variables which we'll explore in this lab).

Throughout this lab, we'll need to make the assumption of **unconfoundedness**, which says that the treatment and potential outcomes are *conditionally* independent given a set of known confounding variables X. Mathematically,

$$Z_i \perp\!\!\!\perp \big(Y_i(0), Y_i(1)\big) \mid X_i \ \forall i$$

If we make this assumption, we can use a few different approaches to estimate the average treatment effect.

Problem Setup and Data

For this lab, you'll be working with data from a job training program in the mid-1970s called the National

Supported Work Demonstration. The data (and the results we'll reproduce) come from a famous 1986 paper by Robert LaLonde, Evaluating the Econometric Evaluations of Training Programs. Here's a description of the program from the original paper (emphasis added):

> The National Supported Work Demonstration (NSW) was a temporary employment program designed to help disadvantaged workers lacking basic job skills move into the labor market by giving them work experience and counseling in a sheltered environment. Unlike other federally sponsored employment and training programs, the NSW program **assigned qualified applicants to training positions randomly**. Those assigned to the treatment group received all the benefits of the NSW program, while those assigned to the control group were left to fend for themselves.

Here are a few more important excerpts from the paper, describing the participants and data collected (emphasis and links added):

> The MDRC admitted into the program AFDC women, ex-drug addicts, ex-criminal offenders, and high school dropouts of both sexes. For those assigned to the treatment group, the program guaranteed a job for 9 to 18 months, depending on the target group and site. The treatment group was divided into crews of three to five participants who worked together and met frequently with an NSW counselor to discuss grievances and performance...
>
> The type of work even varied within sites. In particular, **male and female participants frequently performed different sorts of work**. The female participants usually worked in service occupations, whereas the male participants tended to work in construction occupations.
>
> The MDRC collected earnings and demographic information from both the treatment and the control group at baseline and every nine months thereafter. MDRC also conducted up to four post-baseline interviews.

Our goal will be to estimate the causal effect of the training program on income. Specifically, we will compare the income of people in 1974, 1975 (before the training program) with their income in 1978 (after the program).

Just like LaLonde did, we'll start by evaluating the randomized experiment. Then, we'll look at what would happen if we didn't have a control group, and instead had to use data from an observational study.

Part I: Randomized Experiment

Let's begin by looking at the data from the NSW experiment. It contains the following columns:

- `data_id` : always `NSW` , incidcating that the data are from the NSW randomized experiment
- `treat` : binary variable indicating treatment (1 for job training, 0 for control)
- `age` : age in years
- `educ` : number of years of education
- `black` : whether the worker was Black (1) or not (0).
- `hisp` : whether the worker was Hispanic (1) or not (0).
- `marr` : whether the worker was married (1) or not (0).
- `nodegree` : whether the worker had a high school diploma (0) or not (1).
- `re74` , `re75` : earnings in 1974 and 1975, before the program
- `re78` : earnings in 1978, after the program.
- `outcome` : difference in earnings from 1974 to 1978

In [7]:
```
nsw = pd.read_csv('nsw_dw.csv')
```

```
nsw
```

Out[7]:

	data_id	treat	age	educ	black	hisp	marr	nodegree	re74	re75	re78	outcome
0	NSW	1.0	37.0	11.0	1.0	0.0	1.0	1.0	0.000	0.00	9930.0460	9930.0460
1	NSW	1.0	22.0	9.0	0.0	1.0	0.0	1.0	0.000	0.00	3595.8940	3595.8940
2	NSW	1.0	30.0	12.0	1.0	0.0	0.0	0.0	0.000	0.00	24909.4500	24909.4500
3	NSW	1.0	27.0	11.0	1.0	0.0	0.0	1.0	0.000	0.00	7506.1460	7506.1460
4	NSW	1.0	33.0	8.0	1.0	0.0	0.0	1.0	0.000	0.00	289.7899	289.7899
...	...	...	...	...	...	...	...	...	...	...	...	...
440	NSW	0.0	21.0	9.0	1.0	0.0	0.0	1.0	31886.430	12357.22	0.0000	-31886.4300
441	NSW	0.0	28.0	11.0	1.0	0.0	0.0	1.0	17491.450	13371.25	0.0000	-17491.4500
442	NSW	0.0	29.0	9.0	0.0	1.0	0.0	1.0	9594.308	16341.16	16900.3000	7305.9930
443	NSW	0.0	25.0	9.0	1.0	0.0	1.0	1.0	24731.620	16946.63	7343.9640	-17387.6560
444	NSW	0.0	22.0	10.0	0.0	0.0	1.0	1.0	25720.920	23031.98	5448.8010	-20272.1200

445 rows × 12 columns

In Part I, we assume the participants are randomly assigned to the treatment group, i.e attending the training program (`treat = 1`) and the control group, i.e. not attending the training program (`treat = 0`). Hence, we can compute the causal effect using the following expression:

$$\hat{\tau} = \underbrace{\frac{1}{n_1}\sum_{i:Z_i=1} Y_i}_{\text{mean of treatment group}} - \underbrace{\frac{1}{n_0}\sum_{i:Z_i=0} Y_i}_{\text{mean of control group}}$$

Question 1a Compute causal effect in randomized experiments

Complete the code below to output the causal effect of training program on participants' income using the expression above.

In [10]:
```
# Question 1a, TODO here
causal_effect_nsw = np.mean(nsw.loc[nsw['treat']==1]['re78']) - np.mean(nsw.loc[nsw['treat
causal_effect_nsw
```

Out[10]:
```
1794.342404270271
```

In [11]:
```
# Validation tests: Do not modify
assert np.abs(causal_effect_nsw - 1794.3424) < 0.1
print("Test passed!")
```

```
Test passed!
```

Question 1b Interpret the result

Based on your answer above, what is the causal effect of attending the training program on income? In other words, does attending the training program lead to higher income?

It appears that attending the program will lead to a increase of income of 1794.3424

Part II: Using an Observational Study

Now, suppose instead that (like many programs) this hadn't been a randomized experiment. In that case, we would need to find a separate population to use as our "control group". LaLonde used the Current Population Survey (CPS), a publicly available dataset, as a control group. Let's now look at this data: for your convenience, it has the same columns as the NSW data above. Note that it's much larger!

In [12]:
```
cps = pd.read_csv('cps.csv')
cps
```

Out[12]:

	data_id	treat	age	educ	black	hisp	marr	nodegree	re74	re75	re78	outcome
0	CPS	0	45	11	0	0	1	1	21516.670	25243.550	25564.670	4048.0000
1	CPS	0	21	14	0	0	0	0	3175.971	5852.565	13496.080	10320.1090
2	CPS	0	38	12	0	0	1	0	23039.020	25130.760	25564.670	2525.6504
3	CPS	0	48	6	0	0	1	1	24994.370	25243.550	25564.670	570.3008
4	CPS	0	18	8	0	0	1	1	1669.295	10727.610	9860.869	8191.5740
...	...	...	...	...	...	...	...	...	...	...	...	...
15987	CPS	0	22	12	1	0	0	0	3975.352	6801.435	2757.438	-1217.9141
15988	CPS	0	20	12	1	0	1	0	1445.939	11832.240	6895.072	5449.1330
15989	CPS	0	37	12	0	0	0	0	1733.951	1559.371	4221.865	2487.9140
15990	CPS	0	47	9	0	0	1	1	16914.350	11384.660	13671.930	-3242.4200
15991	CPS	0	40	10	0	0	0	1	13628.660	13144.550	7979.724	-5648.9360

15992 rows × 12 columns

For the rest of the lab, we'll work with a modified version of the data that doesn't have any randomized controls, only the ones from the general population. In the cell below, we creat a new dataframe called `obs` by concatenating the `cps` dataframe with rows of the `nsw` dataframe corresponding to the people who attended the training program.

Your answers to all remaining questions should only use the `obs` table, not the `nsw` table!*

In [13]:
```
treated = nsw[nsw['treat'] == 1]
obs = pd.concat([treated, cps], ignore_index=True)
obs
```

Out[13]:

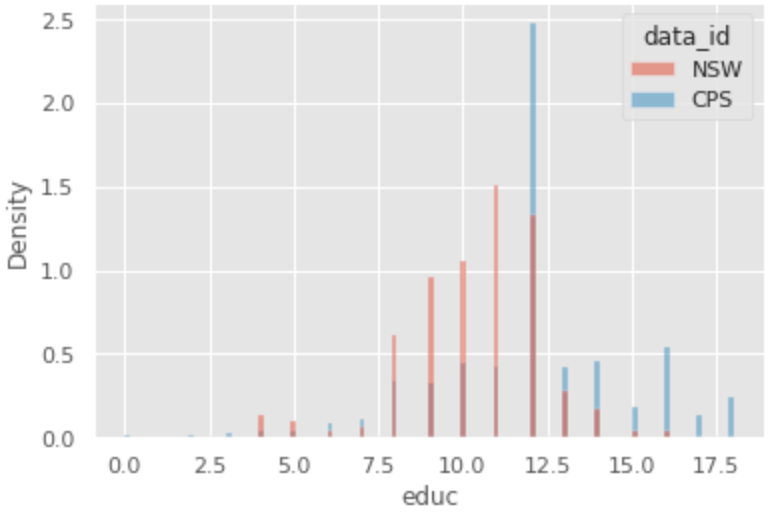
	data_id	treat	age	educ	black	hisp	marr	nodegree	re74	re75	re78	outcome
0	NSW	1.0	37.0	11.0	1.0	0.0	1.0	1.0	0.000	0.000	9930.0460	9930.0460
1	NSW	1.0	22.0	9.0	0.0	1.0	0.0	1.0	0.000	0.000	3595.8940	3595.8940
2	NSW	1.0	30.0	12.0	1.0	0.0	0.0	0.0	0.000	0.000	24909.4500	24909.4500
3	NSW	1.0	27.0	11.0	1.0	0.0	0.0	1.0	0.000	0.000	7506.1460	7506.1460
4	NSW	1.0	33.0	8.0	1.0	0.0	0.0	1.0	0.000	0.000	289.7899	289.7899
...	...	...	...	...	...	...	...	...	...	...	...	...
16172	CPS	0.0	22.0	12.0	1.0	0.0	0.0	0.0	3975.352	6801.435	2757.4380	-1217.9141
16173	CPS	0.0	20.0	12.0	1.0	0.0	1.0	0.0	1445.939	11832.240	6895.0720	5449.1330
16174	CPS	0.0	37.0	12.0	0.0	0.0	0.0	0.0	1733.951	1559.371	4221.8650	2487.9140

	data_id	treat	age	educ	black	hisp	marr	nodegree	re74	re75	re78	outcome
16175	CPS	0.0	47.0	9.0	0.0	0.0	1.0	1.0	16914.350	11384.660	13671.9300	-3242.4200
16176	CPS	0.0	40.0	10.0	0.0	0.0	0.0	1.0	13628.660	13144.550	7979.7240	-5648.9360

16177 rows × 12 columns

The following histogram compares the distribution of education between the NSW treatment group and the CPS group:

In [14]:
```
sns.histplot(data=obs, x='educ', hue='data_id', stat='density', common_norm=False);
```



Question 2a

Based on the histogram above, we can say that education is a confounding variable. How can you justify this claim? In other words, why is education a confounding variable?

Hint: What kind of association do you expect between education and income?

We expect if the education time is longer, then the income would be higher, however from the histogram above we can clearly see education from CPS is overall higher than the education from NSW.

Question 2b

As our first attempt to estimate the causal effect, we decide to try what we did in Question 1. In other words, we compute the Simple Differenence in Observed group means (SDO) for this observational data.

Complete the code below to output compute the SDO using dataset `obs` .

Hint: The code is very similar to the code in question 1a.

In [15]:
```
# Question 2b, TODO here
sdo = np.mean(nsw.loc[nsw['treat']==1]['re78'])-np.mean(obs.loc[obs['treat']==0]['re78'])
sdo
```

Out[15]:
```
-8497.516142636992
```

In [16]:
```
# Validation tests: Do not modify
assert np.abs(sdo + 8497.51614) < 0.1
```

```
print("Test passed!")
```

```
Test passed!
```

You should have found a negative result. This is because of confounding: even though the actual effect of the program is positive (as we saw from the randomized experiment), the treatment group and our CPS group are very different. In particular, individuals in the treatment group face many disadvantages that cause their earnings to be lower, and also cause them to be more likely to end up in the treatment group.

Part III: Unconfoundedness Techniques

Technique 1. Outcome Regression

Suppose the provided variables (age, years of education, Black/Hispanic race, marriage, and diploma) are the only confounders in this problem. In that case, we can make the unconfoundedness assumption, where X represents the collection of all 6 confounding variables listed above.

Suppose we fit a linear model of the following form:

Earnings = $\tau * Z$ + a*age + b*years of education + c*isBlack + d*isHispanic + e*isMarried + f*hasDiploma.

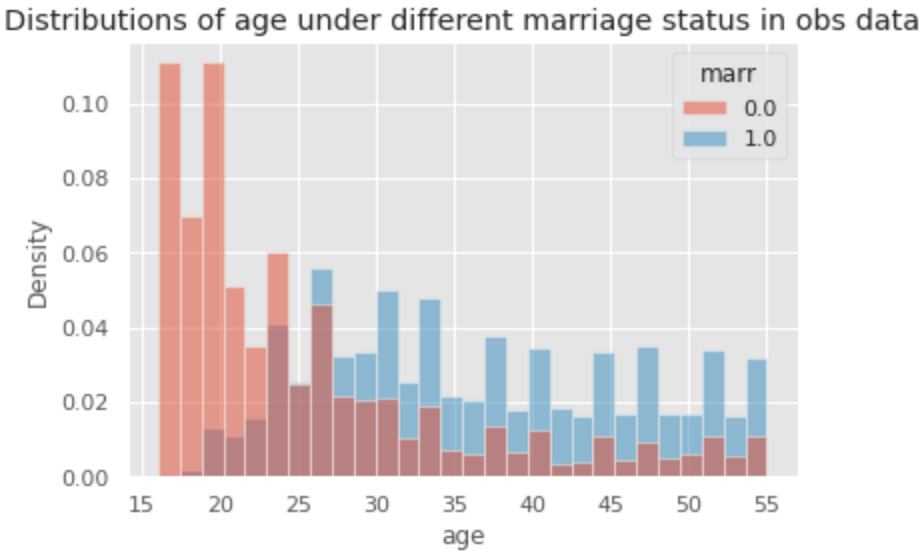
We saw in lecture that if we make two assumptions, then the estimated coefficient of treament from OLS, $\hat{\tau}$, will be an unbiased estimate of the ATE. The two assumptions are:

1. Assume unconfoundedness (described above).
2. Assume this linear model correctly describes the interaction between the variables.

We'll take assumption 1 for granted for now. Assumption 2, however, is much more questionable: it's not clear that the confounding variables would all have a linear effect on earnings. Much worse than that, though, is the fact that the linear model above does not model any interactions between the variables. In particular, it assumes that the effect of each confounder is the same for both treatment and control. This is probably unrealistic.

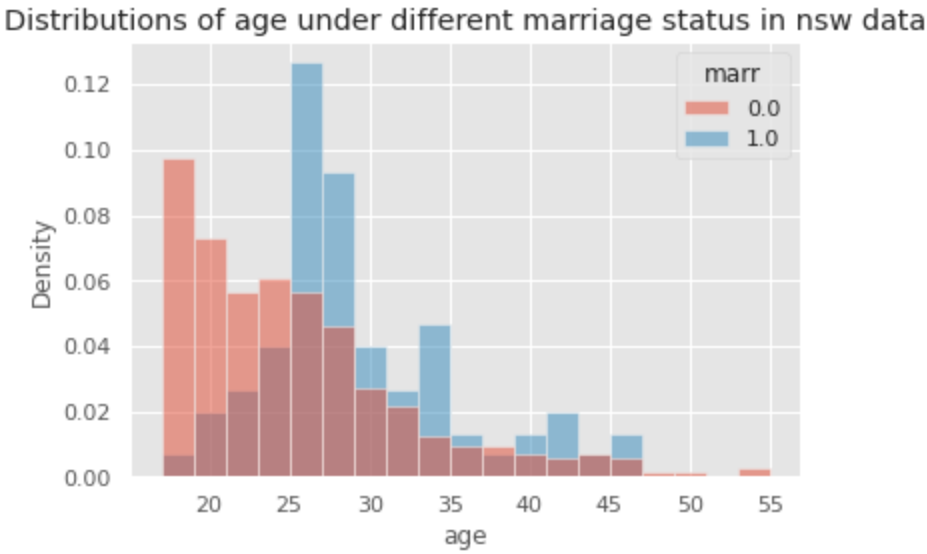
For example, married individuals in the CPS sample might have more financial stability (since they may wait for financial stability to get married), which might not be true in the NSW sample (where individuals have much lower financial stability overall). But, the model above only uses one coefficient, e, for the effect of marriage on income, regardless of whether an individual is from the treatment or control. See the histograms below.

In [17]:
```
sns.histplot(data=obs, x='age', hue='marr', stat='density', common_norm=False);
plt.title("Distributions of age under different marriage status in obs data");
```



In [18]:

```
sns.histplot(data=nsw, x='age', hue='marr', stat='density', common_norm=False);
plt.title("Distributions of age under different marriage status in nsw data");
```



Neverthless, let's try to fit a linear model and see how well it performs. The code below are taken from previous labs.

In [19]:

```
# No TODOs here: Just examine the code
def fit_OLS_model(df, target_variable, explanatory_variables, intercept = False):
    """
    Fits an OLS model from data.

    Inputs:
        df: pandas DataFrame
        target_variable: string, name of the target variable
        explanatory_variables: list of strings, names of the explanatory variables
        intercept: bool, if True add intercept term
    Outputs:
        fitted_model: model containing OLS regression results
    """

    target = df[target_variable]
    inputs = df[explanatory_variables]
    if intercept:
        inputs = sm.add_constant(inputs)

    fitted_model = sm.OLS(target, inputs).fit()
```

```
    return(fitted_model)

def mean_squared_error(true_vals, predicted_vals):
    """
    Return the mean squared error

    Inputs:
        true_vals: array of true labels
        predicted_vals: array labels predicted from the data
    Output:
        float, mean squared error of the predicted values
    """
    return np.mean((true_vals - predicted_vals) ** 2)
```

As a reminder, in previous labs, we used it like this: `fit_OLS_model(student_data, 'NumBooks', ['ReadathonDuration', 'Income'])`

Question 3a

Complete the code below by using the functions above to fit a model to predict 1978 income from the treatment and the confounding variables.

In [20]:
```
# Question 3a, TODO here
linear_model = fit_OLS_model(obs, 're78', ['treat', 'age', 'educ', 'black', 'hisp', 'marr'
#print(linear_model.summary())
```

In [21]:
```
# Compute the mean square error of the values predicted by model. No need to modify here,
predicted = linear_model.predict(obs[['treat', 'age', 'educ', 'black', 'hisp', 'marr', 'no
err = mean_squared_error(obs['re78'].values, predicted)
err
```

Out[21]:
```
84944525.405008
```

In [22]:
```
# Validation tests: Do not modify
assert np.abs(err - 84944525) < 100
print("Test passed!")
```

```
Test passed!
```

Question 3b

Explain, in your own words, why linear regression produces a very incorrect result for this question.

Hint: we've mostly answered this question for you above; you just have to understand and explain in your own words here.

Assumption 2 states: "Assume this linear model correctly describes the interaction between the variables". But the biggest problem is that the linear midel does not correctly describe the interaction between variables. For example 'black', 'hisp', 'marr', 'nodegree' are all binary [0,1] variables,which is non-continous and those are not very helpful variables to plugin in a linear model. So the assumption we rely on is false.

Technique 2: Matching

We have seen above that a simple linear regression model is not ideal. Now, we consider a technique introduced in lecture called matching.

Consider two individuals, one treated and one untreated, with the exact same values of all confounding variables X. Here's an example of someone from the NSW study and someone from the CPS data with the exact same set of confounding variables:

In [26]:
```
nsw.iloc[50:51, :]
```

Out[26]:

	data_id	treat	age	educ	black	hisp	marr	nodegree	re74	re75	re78	outcome
50	NSW	1.0	28.0	8.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0

In [27]:
```
cps.iloc[2363:2364, :]
```

Out[27]:

	data_id	treat	age	educ	black	hisp	marr	nodegree	re74	re75	re78	outcome
2363	CPS	0	28	8	1	0	0	1	15286.2	3863.516	0.0	-15286.2

If we assume unconfoundedness, then for these two people, there should be no other variables that have an effect on both the treatment and the outcome. So, by subtracting their outcomes, we should be able to estimate the causal effect of the job training program for this particular X (specifically, 28-year old, unmarried, Black, non-Hispanic people without high school diplomas who've completed 8 years of schooling).

If we do this for every possible set of values for the confounders X, then we can take all of them and compute the expectation (weighting each by the probability of seeing that corresponding value of X). Empirically, this corresponds to just taking the average of all the data points.

Here is the matching algorithm in English:

1. For each treated row:
    - Find all untreated rows that have the exact same values of all confounders.
    - Take those untreated rows and average their outcome
    - Subtract the average above from the treated row's outcome
1. For each *untreated* row:
    - Find all *treated* rows that have the exact same values of all confounders.
    - Take those *treated* rows and average their outcome
    - Subtract the *untreated* row's outcome from the average above
2. Average all the results from steps 1 and 2.

Question 3c

Explain why this exact maching algorithm will not work for the dataset provided.

Hint: What if there are no matches for a person?

The goal of exact matching is to mimic the procedure of a randomlized experiment assuming we are taking accunt of all confounding factors. However if a person has no matching, then it means we cannot perform any experiment on that person. For example we know the edu of the NSW data set is generally lower than the edu of the CPS data set, this will create a peoblem that a lot of people from the NSW set cannot find a match due to their low edu level, so we are essentialy neglecting a large chunck from the NSW set.

There are solutions such as approximate matching which matches people if they have similar features (not

necessarily identical), but we'll instead turn to using propensity scores instead.

Technique 3: Inverse Propensity Weighting

Recall the definition of the propensity score: it is the probability that a unit was treated, conditioned on a particular set of confounders x:

$$e(x) = P(Z = 1|X = x)$$

We've already seen that for this dataset, we can't use the simple difference in observed group means (SDO) to estimate the causal ATE. In this section, we'll try inverse propensity weighting instead.

The simplest and most common way to compute propensity scores is using logistic regression: you'll get practice with this on HW4. In particular, in this example, we would use the `treat` column as our target variable and the confounders as our predictors.

In this lab, we have computed the propensity scores for you using a slightly more complex model that also includes income before the program (`re74`) and includes some nonlinear interactions.

In [29]:
```
# Import obs data with propensity scores computed
obs_prop = pd.read_csv('obs_with_propensity_scores.csv')
obs_prop.drop('Unnamed: 0', axis = 1, inplace = True)
obs_prop
```

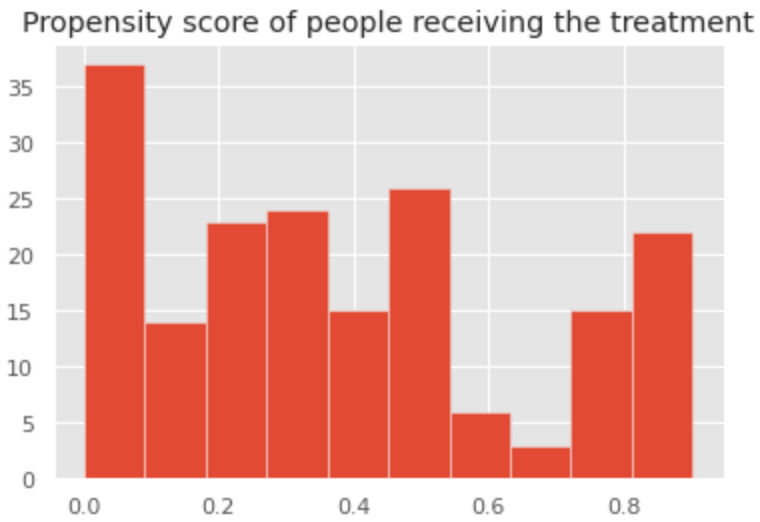
Out[29]:

	data_id	treat	age	educ	black	hisp	marr	nodegree	re74	re75	re78	outcome	ps
0	NSW	1.0	37.0	11.0	1.0	0.0	1.0	1.0	0.000	0.000	9930.0460	9930.0460	
1	NSW	1.0	22.0	9.0	0.0	1.0	0.0	1.0	0.000	0.000	3595.8940	3595.8940	
2	NSW	1.0	30.0	12.0	1.0	0.0	0.0	0.0	0.000	0.000	24909.4500	24909.4500	
3	NSW	1.0	27.0	11.0	1.0	0.0	0.0	1.0	0.000	0.000	7506.1460	7506.1460	
4	NSW	1.0	33.0	8.0	1.0	0.0	0.0	1.0	0.000	0.000	289.7899	289.7899	
...	...	...	...	...	...	...	...	...	...	...	...	...	
16172	CPS	0.0	22.0	12.0	1.0	0.0	0.0	0.0	3975.352	6801.435	2757.4380	-1217.9141	
16173	CPS	0.0	20.0	12.0	1.0	0.0	1.0	0.0	1445.939	11832.240	6895.0720	5449.1330	
16174	CPS	0.0	37.0	12.0	0.0	0.0	0.0	0.0	1733.951	1559.371	4221.8650	2487.9140	
16175	CPS	0.0	47.0	9.0	0.0	0.0	1.0	1.0	16914.350	11384.660	13671.9300	-3242.4200	
16176	CPS	0.0	40.0	10.0	0.0	0.0	0.0	1.0	13628.660	13144.550	7979.7240	-5648.9360	

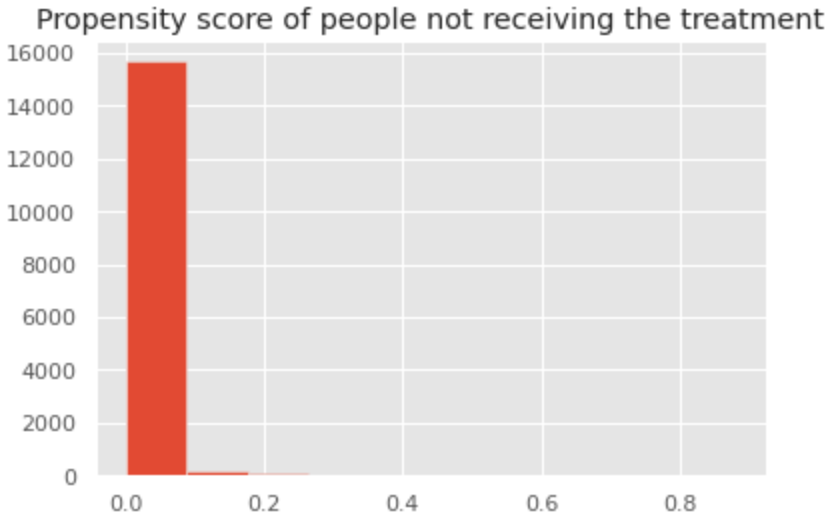
16177 rows × 13 columns

Examine the following histogram of propensity scores, grouped by dataset:

In [30]:
```
plt.hist(obs_prop[obs_prop['treat'] == 1]['pscore']);
plt.title("Propensity score of people receiving the treatment");
```



In [43]:
```
plt.hist(obs_prop[obs_prop['treat'] == 0]['pscore']);
plt.title("Propensity score of people not receiving the treatment");
```



Question 3d

Explain why the two histograms are so different.

Hint: Think about the characteristics of the people participating in the training program (see Problem Setup and Data section).

The people who participated in the program are: disadvantaged workers lacking basic job skills.The first graph represent those who receives treatment that's why their Propensity score is evenly distributed, the second graph represents those who receives no treatment at all that's why their Propensity score is concentrated at 0

Question 3e

We could use the propensity scores for a number of things, including matching (as described in Discussion 7), but in this lab we'll focus on inverse propensity weighting (IPW). Recall from lecture that the IPW estimator of the ATE is:

$$\hat{\tau}_{IPW} = \underbrace{\frac{1}{n_1}\sum_{i:Z_i=1}\frac{Y_i}{e(X_i)}}_{\text{weighted mean of treated rows}} - \underbrace{\frac{1}{n_0}\sum_{i:Z_i=0}\frac{Y_i}{1-e(X_i)}}_{\text{weighted mean of untreated rows}}$$

Note that the weights are different for the two groups. Intuitively, the weights decrease the importance of points that have a high probability of being in the group that they're in.

For example, consider two individuals from the CPS data: person A, who looks very different from the treatment (NSW) population, and person B, who looks much more similar to the treatment (NSW) population. Person A's propensity score will be much closer to 1, and so the denominator $1 - e(X_A)$ will be small, increasing our weight of their outcome. On the other hand, person B's propensity score will be closer to 0, decreasing our weight of their outcome. This way, we give less weight to person B, who doesn't look like someone from the treatment group anyway.

Complete the cell below to compute the IPW estimate for the ATE.

In [81]:
```
def w_avg(df, values, weights):
    d = df[values]
    w = df[weights]
    return (d * w).sum() / w.sum()
```

In [85]:
```
# Question 3e, TODO here
df1= obs_prop.loc[obs_prop['treat']==1]
df2= obs_prop.loc[obs_prop['treat']==0]
ipw_estimate = w_avg(df1,'re78','pscore')-w_avg(df2,'re78','pscore')
ipw_estimate
```

Out[85]:
```
249147.38510221028
```

In [86]:
```
# Validation tests: Do not modify
assert np.abs(ipw_estimate - 248589) < 1000
print("Test passed!")
```

```
Test passed!
```

Question 3f

You might find a surprisingly large result in 3e. Recent work in IPW suggests that a good rule of thumb is to only include points with propensity scores between 0.1 and 0.9

In the cell below, remove any data points with propensity scores that are too low or too high, and repeat the computation in 3e.

In [94]:
```
# Question 3f, TODO here
cleaned_obs_prop = obs_prop.loc[(obs_prop['pscore'] < 0.9) & obs_prop['pscore']>0.1]
df11= cleaned_obs_prop.loc[cleaned_obs_prop['treat']==1]
df22= cleaned_obs_prop.loc[cleaned_obs_prop['treat']==0]
ipw_estimate =  w_avg(df11,'re78','pscore')-w_avg(df22,'re78','pscore')
ipw_estimate
```

Out[94]:
```
9290.443905334563
```

In [95]:
```
# Validation tests: Do not modify
assert np.abs(ipw_estimate - 9298) < 100
print("Test passed!")
```

```
Test passed!
```

This estimate is much closer to the true causal effect we obtained from the randomized experiment in part I, even if it is quite a bit larger.

Question 3g

Now let's interpret the result of IPW. Fill in the blanks below with the appropriate phrases:

If we assume that ___, then the estimated effect of the program using IPW is that the program causes people to earn ___ more than they would have.

TODO: Your answer here

Blank 1: unconfoundedness and Outcome Regression

Blank 2: a lot

Question 3h

Give at least one reason why the IPW estimate doesn't match the true estimate, using what you know about the assumptons we've made.

Hint: there is more than one right answer.

The NSW and the CPS are data sets coming from different time periods, we are assuming unconfoundedness but in reality treatment and potential outcomes may actually be dependent given a set of confounding variables

In [96]:
```
import matplotlib.image as mpimg
img = mpimg.imread('baby_duckling.jpg')
imgplot = plt.imshow(img)
imgplot.axes.get_xaxis().set_visible(False)
imgplot.axes.get_yaxis().set_visible(False)
print("Yay, you've made it to the end of Lab 8!")
plt.show()
```

```
Yay, you've made it to the end of Lab 8!
```

In [ ]: