Problem Set 4

Creator: Junjie Lei

CreationDate: Feb_23/2020

Title: Causal Inference Assignment04 -- Practical Sources of bias

Note: My *Colab*

In-class foundation

1.
    - the $ATE = 0.03$
    - selection bias = $E[Y^0|d=1] - E[Y^0|d=0] = -0.039$
    - Differential effect bias = $E[Y^1 - Y^0|D=1] - E[Y^1 - Y^0|D=0] = -0.001198$
    - $NATE = ATE + Selection\ Bias = +Differential\ Effect\ Bias = -0.01092719$
    - Calculate $NATE$ using the `nate` function = -0.010903325
    - We have 2 kinds of biases $Selection\ Bias$ & $Differential\ Bias$

      however, the $Differential\ Bias$ seems very trivial compared to the NATE, hence we can say that $Selection\ Bias$ is the main bias here.

```
df['delta'] = df['y1'] - df['y0']
df.head()
```

	date	day_of_week	unit	y0	y1	treatment_percent_by_day	d	y	hour_of_day	delta
0	0	0	0	0	0	0.05	0	0	16	0
1	0	0	1	0	1	0.05	0	0	16	1
2	0	0	2	1	0	0.05	0	1	10	-1
3	0	0	3	0	1	0.05	0	0	5	1
4	0	0	4	0	0	0.05	0	0	2	0

```
#selection bias:
sb = df.query("d==1")['y0'].mean() - df.query("d==0")['y0'].mean()
print(sb)

#differential effect bias:
#deb = df.query("d==1")['y1'].mean()-df.query("d==1")['y0'].mean() -
(df.query("d==0")['y1'].mean()-df.query("d==0")['y0'].mean())
deb = df.query("d==1")['delta'].mean()-df.query("d==0")['delta'].mean()
print(deb)

#NATE by decomposition:
print(0.03+sb+deb)

#NATE using function:
```

```
print(estimate_nate(df))
```

```
#selection bias:
sb = df.query("d==1")['y0'].mean() - df.query("d==0")['y0'].mean()
print(sb)
```

```
-0.039728686746445974
```

```
#differential effect bias:
#deb = df.query("d==1")['y1'].mean()-df.query("d==1")['y0'].mean() - (df.query("d==0")['y1'].mean()-df.query("d==0")['y0'].mean()
deb = df.query("d==1")['delta'].mean()-df.query("d==0")['delta'].mean()
print(deb)
```

```
-0.001198507716721798
```

```
print(0.03+sb+deb)
```

```
-0.010927194463167773
```

```
print(estimate_nate(df))
```

```
-0.010903325481920106
```

```
ind_1 = df.query("d==1")['y0'].mean() - df.query("d==0")['y0'].mean()
ind_2 = df.query("d==1")['y1'].mean() - df.query("d==0")['y1'].mean()
print(ind_1)
print(ind_2)
```

```
-0.039728686746445974
-0.040927194463167754
```

```
ate = df['y1'].mean() - df['y0'].mean()
print(ate)
```

```
0.029617165394838385
```

2.

```
ind_1 = df.query("d==1")['y0'].mean() - df.query("d==0")['y0'].mean()
ind_2 = df.query("d==1")['y1'].mean() - df.query("d==0")['y1'].mean()
print(ind_1)
print(ind_2)
```

```
-0.039728686746445974
-0.040927194463167754
```

Independence assumption holds true when the following conditions are met:

(1) E[Y0|D=1] = E[Y0|D=0];

(2) E[Y1|D=1] = E[Y1|D=0]

Condition (1) is not met: E[Y0|D=1] - E[Y0|D=0] = -0.03973

Condition (2) is not met: E[Y1|D=1] - E[Y1|D=0] = = - 0.040927

=> Independence assumption does not hold.

3.

```
ate = df['y1'].mean() - df['y0'].mean()
print(ate)
```

```
0.029617165394838385
```

ATE calculated from the data = E[Y1] - E[Y0] = 0.029617 which is approximate to the ATE set in generate experiment function.

4.

```
#selection bias:
sb_ps = conversion_rates_by_strata_y0.eval("difference * counts").sum() / conversion_rates_by_strata_y0['counts'].sum()
print(sb_ps)

#differential effect bias:
deb_ps = conversion_rates_by_strata_delta.eval("difference * counts").sum() / conversion_rates_by_strata_delta['counts'].sum()
print(deb_ps)

#NATE by decomposition:
print(0.03+sb_ps+deb_ps)
```

```
-0.00041967518384157905
0.000543332160684533
0.030123656976842952
```

ATE = 0.030351581467128186

Selection bias = E[Y0|D=1] - E[Y0|D=0] = -0.00041967518384157905

Differential effect bias = E[Y1-Y0|D=1] - E[Y1-Y0|D=0] = 0.000543332160684533

NATE = ATE + selection bias + differential effect bias = 0.030123656976842952

NATE without perfect stratification: -0.010903325481920106

We can have unbiased ATE this time since NATE is very close to ATE. This work because we somehow eliminate the variation during the week by stratify on the basis of date. By doing so, the variation will get smaller.

5.

```
print(sfa.ols("y ~ d", df).fit().summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.000
Model:                            OLS   Adj. R-squared:                  0.000
Method:                 Least Squares   F-statistic:                     197.5
Date:                Mon, 24 Feb 2020   Prob (F-statistic):           7.30e-45
Time:                        20:04:06   Log-Likelihood:            -8.1947e+05
No. Observations:             1399560   AIC:                         1.639e+06
Df Residuals:                 1399558   BIC:                         1.639e+06
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.2565      0.000    567.473      0.000       0.256       0.257
d             -0.0109      0.001    -14.054      0.000      -0.012      -0.009
==============================================================================
Omnibus:                   275861.278   Durbin-Watson:                   1.920
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           330856.191
Skew:                           1.138   Prob(JB):                         0.00
Kurtosis:                       2.295   Cond. No.                         2.41
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

```
1   print(sfa.ols("y ~ d + C(day_of_week)", df).fit().summary())
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:                      y   R-squared:                       0.041
Model:                            OLS   Adj. R-squared:                  0.041
Method:                 Least Squares   F-statistic:                     8595.
Date:                Mon, 24 Feb 2020   Prob (F-statistic):               0.00
Time:                        20:04:18   Log-Likelihood:            -7.9012e+05
No. Observations:             1399560   AIC:                         1.580e+06
Df Residuals:                 1399552   BIC:                         1.580e+06
Df Model:                           7                                         
Covariance Type:            nonrobust                                         
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
Intercept                 0.2996      0.001    307.191      0.000       0.298       0.302
C(day_of_week)[T.1]       0.0008      0.001      0.583      0.560      -0.002       0.003
C(day_of_week)[T.2]      -0.0012      0.001     -0.879      0.380      -0.004       0.001
C(day_of_week)[T.3]       0.0006      0.001      0.467      0.641      -0.002       0.003
C(day_of_week)[T.4] -2.692e-05      0.001     -0.020      0.984      -0.003       0.003
C(day_of_week)[T.5]      -0.1988      0.001   -146.506      0.000      -0.202      -0.196
C(day_of_week)[T.6]      -0.2002      0.001   -147.559      0.000      -0.203      -0.198
d                         0.0299      0.001     38.415      0.000       0.028       0.031
==============================================================================
Omnibus:                   256211.678   Durbin-Watson:                   1.999
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           284841.651
Skew:                           1.046   Prob(JB):                         0.00
Kurtosis:                       2.286   Cond. No.                         8.35
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Homework Extension

1.

we got

```
Estimate of ATE (perfect stratification) =  -0.010997400989847719
```

from the dataset.

Doing a perfect stratification analysis by the hour of the day can't give an unbiased estimate (the result is -0.01099 while the ATE set in DGP is 0.03). Hour of the day is not a pattern like day of the week as showed in graph above - there's hardly any trend of conversion rate based on hour. The experiment last for 14 days and data is generated based on days. While hour is only randomly added to units. If our data is generated based on hours during day, then the stratification may work.

2.

true ATE

```python
ate_day = df_day['y1'].mean() - df_day['y0'].mean()
print(ate_day)
```

```
0.07271803948205777
```

estimate the ate using a regression --> 0.0680

```python
print(sfa.ols("y ~ d + C(day_of_week)", df_day).fit().summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.058
Model:                            OLS   Adj. R-squared:                  0.058
Method:                 Least Squares   F-statistic:                 1.241e+04
Date:                Mon, 24 Feb 2020   Prob (F-statistic):               0.00
Time:                        20:12:06   Log-Likelihood:            -7.9658e+05
No. Observations:             1400079   AIC:                         1.593e+06
Df Residuals:                 1400071   BIC:                         1.593e+06
Df Model:                           7
Covariance Type:            nonrobust
=============================================================================================
                                 coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------------
Intercept                      0.3096      0.001    316.105      0.000       0.308       0.311
C(day_of_week)[T.1.0]         -0.0004      0.001     -0.282      0.778      -0.003       0.002
C(day_of_week)[T.2.0]          0.0001      0.001      0.097      0.922      -0.003       0.003
C(day_of_week)[T.3.0]         -0.0007      0.001     -0.554      0.579      -0.003       0.002
C(day_of_week)[T.4.0]         -0.0013      0.001     -0.982      0.326      -0.004       0.001
C(day_of_week)[T.5.0]         -0.2412      0.001   -176.951      0.000      -0.244      -0.239
C(day_of_week)[T.6.0]         -0.2406      0.001   -176.597      0.000      -0.243      -0.238
d                              0.0680      0.001     87.075      0.000       0.066       0.070
==============================================================================
Omnibus:                   296290.092   Durbin-Watson:                   1.999
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           251449.240
Skew:                           0.950   Prob(JB):                         0.00
Kurtosis:                       2.161   Cond. No.                         8.35
==============================================================================
```

estimate the ATE using a perfect stratification on `day_of_week`

day_of_week	control	treatment	difference	counts
0.0	0.301142	0.399956	0.098814	200028
1.0	0.299420	0.402889	0.103469	200340
2.0	0.300947	0.400784	0.099836	200262
3.0	0.300017	0.400055	0.100037	199210
4.0	0.300057	0.397984	0.097927	199612
5.0	0.100219	0.104605	0.004386	200053
6.0	0.100897	0.105318	0.004421	200574

```
Estimate of ATE (perfect stratification) =  0.07265873336988789
```

from all above, both give unbiased estimate since both results are closed to the true ATE.

Bonus

Regression with day_of_week as control variable: this estimator gives unbiased ATE estimate since it takes into account day of week as a factor that could influence the estimate.

Stratification on day_of_week help reduces the variation in ATE between days during the week. These 2 estimators work since the true ATE is set to vary by day. If ATE varies by week or hour during the day, these might not work.

One-sided noncompliance in a web experiment

1.

assignment is the Z which is the `coin flips`

`saw the page is` the treatment D

2.

- the `saw_treatment_page == coin_flips` indicates the individuals who complies to the assignments meaning that $D_i(Z_i) = Z_i$
- `viewed_page == 1` indicates those who see the treatment page. it contains either the treatment compliers $(D(Z=1)=1$ and the control defiers $(D(Z=0)=0)$. always taker

group

- the main differences for these 2 is that `saw_treatment_page == coin_flip` has the $D(Z=0)=0$ but the `viewed_page == 1` only contains the $D(Z=0)=1$ group.

3.

Estimation methods

	Depe
	As-treated
	(1)
Intercept	0.264***
	(0.001)
coin_flip	
saw_treatment_page	0.281***
	(0.001)
Observations	1000000.0

the as-treated estimates is 0.281

the `as-treated` estimate ignores that people who were given the assignments can actually defy or act in the totally opposite way. In this case, we can no longer say that $E(Y^0)\&E(Y^1)$ in independent because now the besides treatment and assignments there are other factors influences individuals' decisions; i.e charitability.

4.

```
r_itt_1 = sfa.ols("y ~ coin_flip + charitability", df).fit(co
v_type='HC1')
r_itt_2 = sfa.ols("y ~ coin_flip + viewed_page", df).fit(cov_
type='HC1')
```

$Coef_{viewpage}$ = 0.22

$Coef_{coinflip}$ = 0.1

using the ITT regression but including different covariates did not help us improve and identify the ATE estimates.

5.

```
complier_ratio = df.query('viewed_page ==
coin_flip').unit.count()/df.unit.count()
IIT = r_itt.params['coin_flip']
CACE = IIT / complier_ratio
```

ITT = 0.01

CACE is 0.12

- ITT tend to underestimate the ATE, because it assumes E(Y) is the same as Z.
- CACE can be used to restore/improve ATE by using the P(D = 1)

Heterogeneous treatment effects

1.

```
df.y1.mean() - df.y0.mean()
```

output --> 0.15

2.

Estimation methods

	Dependent variable: P(Donation)			
	As-treated	Per-protocol	ITT	CACE
	(1)	(2)	(3)	(4)
Intercept	0.264***	0.284***	0.284***	0.344***
	(0.001)	(0.001)	(0.001)	(0.001)
coin_flip			0.087***	0.175***
			(0.001)	(0.001)
saw_treatment_page	0.256***	0.235***		
	(0.001)	(0.001)		
Observations	1000000.0	749736.0	1000000.0	499848.0

the ATE and CACE in this case were hard to identify.

they are pretty much far away from the ATE -> 0.15

3.

```
cace_1 = sfa.ols("y ~ coin_flip + charitability", df4.query("viewed_page ==
1")).fit(cov_type='HC1')
print(r_cace_1.summary(yname="Conversion"))
```

in the previous problem, the `CACE` can help to find the ATE estimate;

but now it is hard for us to identify, because in the previous problem we can stratify, but in this case, after we adding the covariate of charitability, we can no longer control for each individual decision.

4.

if we still have the `coin_flip` and `saw_treatment_page` it is possible for us to restore the missing data with some deviation, but now we are expecting a lower estimation. The accuracy of estimation will be affected because we still need to identify the individuals who `treatment (saw_page)== assingment(coin_flip)`
