Stat158_FinalProject

June 11, 2023

[2]:
```
import pandas as pd
import numpy as np
import statsmodels.api as sm
import statsmodels.formula.api as smf;
```

[3]:
```
data = pd.read_stata("EnosFowler_PivotalityExperiment.dta")
```

[4]:
```
data.head(1)
```

[4]:
```
    id  s11   treatment  pivotal  call_attempt  ind_contact  informed  \
0  1.0  0.0  No Contact      NaN           0.0          0.0       NaN

   hh_contact  stratum_id  phone_id  …      town hispanic  dem  rep   age  \
0         0.0        40.0       1.0  …  CHARLTON      0.0  0.0  0.0  36.0

   g10  s09  g08  s11_absentee  unmatched
0  1.0  0.0  1.0             0        0.0

[1 rows x 21 columns]
```

0.1 Part 1 : Replicating the Original Analysis

[5]:
```
data["pivotal"].isna()
```

[5]:
```
0         True
1         True
2         True
3         True
4        False
         …
17035    False
17036    False
17037    False
17038    False
17039     True
Name: pivotal, Length: 17040, dtype: bool
```

[6]:
```
data_no_missing_values = data.dropna(subset=["pivotal"])
data_no_missing_values[data_no_missing_values["stratum_id"] == 1]
```

[6]:

	id	s11	treatment	pivotal	call_attempt	ind_contact	informed	\
12	13.0	0.0	Pivotal	1.0	1.0	0.0	NaN	
110	111.0	0.0	Pivotal	1.0	0.0	0.0	NaN	
113	114.0	0.0	Pivotal	1.0	1.0	0.0	NaN	
117	118.0	0.0	Pivotal	1.0	1.0	0.0	NaN	
136	137.0	0.0	Pivotal	1.0	1.0	0.0	NaN	
…	…	…	…	…	…	…	…	
16948	16949.0	0.0	Pivotal	1.0	0.0	0.0	NaN	
16972	16973.0	0.0	Reminder	0.0	1.0	0.0	NaN	
16998	16999.0	0.0	Reminder	0.0	1.0	0.0	NaN	
17028	17029.0	0.0	Pivotal	1.0	1.0	0.0	NaN	
17034	17035.0	0.0	Pivotal	1.0	1.0	0.0	NaN	

	hh_contact	stratum_id	phone_id	…	town	hispanic	dem	rep	\
12	0.0	1.0	6.0	…	CHARLTON	0.0	0.0	1.0	
110	0.0	1.0	83.0	…	CHARLTON	0.0	0.0	0.0	
113	0.0	1.0	86.0	…	CHARLTON	0.0	0.0	0.0	
117	0.0	1.0	89.0	…	CHARLTON	0.0	1.0	0.0	
136	0.0	1.0	103.0	…	CHARLTON	0.0	1.0	0.0	
…	…	…	…	…	…	…	…	…	
16948	0.0	1.0	8793.0	…	CHARLTON	0.0	0.0	0.0	
16972	0.0	1.0	8812.0	…	CHARLTON	1.0	0.0	0.0	
16998	0.0	1.0	8832.0	…	CHARLTON	0.0	1.0	0.0	
17028	0.0	1.0	8857.0	…	CHARLTON	0.0	0.0	0.0	
17034	0.0	1.0	8861.0	…	CHARLTON	0.0	0.0	1.0	

	age	g10	s09	g08	s11_absentee	unmatched
12	83.0	0.0	0.0	1.0	0	0.0
110	23.0	0.0	0.0	0.0	0	0.0
113	42.0	0.0	0.0	0.0	0	0.0
117	29.0	0.0	0.0	0.0	0	0.0
136	36.0	0.0	0.0	1.0	0	0.0
…	…	…	…	…	…	
16948	81.0	0.0	0.0	0.0	0	0.0
16972	26.0	0.0	0.0	0.0	0	0.0
16998	24.0	0.0	0.0	1.0	0	0.0
17028	36.0	0.0	0.0	1.0	0	0.0
17034	33.0	0.0	0.0	0.0	0	0.0

[172 rows x 21 columns]

[7]:
```
md = smf.ols("s11 ~ pivotal ", data_no_missing_values,␣
 ↪groups=data_no_missing_values["stratum_id"])
```

```
mdf = md.fit(cov_type = "cluster", cov_kwds =  {'groups':␣
 ↪data_no_missing_values['phone_id']})
print(mdf.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    s11   R-squared:                       0.000
Model:                            OLS   Adj. R-squared:                 -0.000
Method:                 Least Squares   F-statistic:                    0.3754
Date:                Sat, 13 May 2023   Prob (F-statistic):              0.540
Time:                        04:54:44   Log-Likelihood:                -7437.4
No. Observations:               11361   AIC:                         1.488e+04
Df Residuals:                   11359   BIC:                         1.489e+04
Df Model:                           1
Covariance Type:            cluster
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.3146      0.008     40.986      0.000       0.300       0.330
pivotal        0.0067      0.011      0.613      0.540      -0.015       0.028
==============================================================================
Omnibus:                   119772.571   Durbin-Watson:                   1.414
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             2070.125
Skew:                           0.782   Prob(JB):                         0.00
Kurtosis:                       1.612   Cond. No.                         2.62
==============================================================================

Notes:
[1] Standard Errors are robust to cluster correlation (cluster)

/opt/conda/lib/python3.9/site-packages/statsmodels/base/model.py:127:
ValueWarning: unknown kwargs ['groups']
  warnings.warn(msg, ValueWarning)
```

[8]:
```
md_2 = smf.ols("s11 ~ pivotal ",␣
 ↪data_no_missing_values[data_no_missing_values["ind_contact"]==1],␣
 ↪groups=data_no_missing_values[data_no_missing_values["ind_contact"]==1]["stratum_id"])
mdf_2 = md_2.fit()
print(mdf_2.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    s11   R-squared:                       0.000
Model:                            OLS   Adj. R-squared:                 -0.001
Method:                 Least Squares   F-statistic:                    0.1462
Date:                Sat, 13 May 2023   Prob (F-statistic):              0.702
Time:                        04:54:44   Log-Likelihood:                -677.07
No. Observations:                 936   AIC:                             1358.
```

```
Df Residuals:                      934   BIC:                             1368.
Df Model:                            1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.4598      0.023     20.279      0.000       0.415       0.504
pivotal        0.0125      0.033      0.382      0.702      -0.052       0.077
==============================================================================
Omnibus:                     3590.952   Durbin-Watson:                   1.934
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              155.917
Skew:                           0.137   Prob(JB):                     1.39e-34
Kurtosis:                       1.019   Cond. No.                         2.58
==============================================================================
```

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

```
/opt/conda/lib/python3.9/site-packages/statsmodels/base/model.py:127:
ValueWarning: unknown kwargs ['groups']
  warnings.warn(msg, ValueWarning)
```

0.2 Part 2: FRT under the Sharp Null with Difference-In-Means Test Statistic

[9]:
```
dat = data_no_missing_values
```

[10]:
```
# number of n1 & n0 in the whole dataset
n1 = sum(dat["pivotal"] == 1)
n0 = sum(dat["pivotal"] == 0)
print(f"The overall number of treated units in this dataset is {n1} and the␣
 ↪overall number of control units is {n0}")
```

```
The overall number of treated units in this dataset is 5674 and the overall
number of control units is 5687
```

[11]:
```
#calculating the prerequisite to run FRT under sharp null --> Is the proportion␣
 ↪of control & treatment same across strata?
expectation = 0
for i in np.arange(44):
    expectation += (sum(dat[dat["stratum_id"] == (i+1)]["pivotal"] == 1) / n1)␣
 ↪- (sum(dat[dat["stratum_id"] == (i+1)]["pivotal"] == 0) / n0)

round(expectation,3)
```

[11]:
```
0.0
```

Prerequisite Satisfied!

[12]:
```
#propensity score example
len(dat[dat["stratum_id"] == 1])/ len(dat)
```

[12]:
```
0.015139512366869114
```

[13]:
```
#per stratum treatment effect example
id_1 = dat[dat["stratum_id"] == 1]
np.mean(id_1[id_1["pivotal"] == 1]["s11"]) - np.mean(id_1[id_1["pivotal"] ==␣
 ↪0]["s11"])
```

[13]:
```
0.0
```

[14]:
```
#estimated_ate with difference-in-means as test statistic for intent to treat
estimated_ate = 0
for i in np.arange(44):
    propensity_score = len(dat[dat["stratum_id"] == (i+1)])/ len(dat)
    k = dat[dat["stratum_id"] == (i+1)]
    estimated_ate += propensity_score * (np.mean(k[k["pivotal"] == 1]["s11"]) -␣
 ↪np.mean(k[k["pivotal"] == 0]["s11"]))
estimated_ate
```

[14]:
```
0.00643550290747193
```

This estimated ATE is consistent with the regression coefficient estimate (the first one).

[15]:
```
permuted_effects = []
for i in np.arange(200):
    permuted_ate = 0
    for i in np.arange(44):
        dat_stratum = dat[dat["stratum_id"] == (i + 1)]
        permuted_treatment = np.random.permutation(dat_stratum["pivotal"])
        dat_stratum["permuted_pivotal"] = permuted_treatment
        propensity_score = len(dat_stratum )/ len(dat)
        permuted_ate += propensity_score * (np.
 ↪mean(dat_stratum[dat_stratum["permuted_pivotal"] == 1]["s11"]) - np.
 ↪mean(dat_stratum[dat_stratum["permuted_pivotal"] == 0]["s11"]))
    permuted_effects.append(permuted_ate)

permuted_effects
```

```
/tmp/ipykernel_133/1147578683.py:7: 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-view-versus-a-copy
  dat_stratum["permuted_pivotal"] = permuted_treatment
```

```
[15]: [0.0025629902394758474,
       9.80916379670336e-05,
       0.0036204632097210383,
       0.00502646572172416,
       -0.006238306929027727,
       -0.011520834829687455,
       -0.004126271931393061,
       0.003971493108878934,
       -0.005183378920639791,
       0.003972423308062601,
       0.0018582202031147352,
       0.012421567730448866,
       -0.0016623572221383383,
       -0.0023684341143385525,
       -0.006943380639833174,
       -0.007648545123316036,
       -0.002719894275016774,
       0.015942421118688564,
       -0.0006061829580653992,
       -0.007294328407668993,
       -0.005886527978624216,
       -0.021378959554760154,
       -0.0016625453150326927,
       0.0008035647193370451,
       -0.008703765738648898,
       -0.013632514778563662,
       -0.013984166089654883,
       0.004675313043083885,
       -0.0002524486061089444,
       0.008900211810516114,
       0.0018595909564866112,
       9.792089992169319e-05,
       0.004674861575018173,
       0.0057317886848142685,
       -0.006239242364800256,
       -0.002013539022377245,
       -0.003071442305131464,
       -0.0044801836439383996,
       0.0004504454868035116,
       -0.0009578775259382554,
       -0.0002540188413036293 6,
       -0.004829396782204379,
       0.009603106253712794,
       0.011367384402050058,
       -0.0009581926882518818,
       -0.009760376481420816,
       0.0025635357110336213,
```

0.011365141276584608,
0.006435832933893862,
0.011014843779139702,
9.973203566133104e-05,
-0.006591700188288984,
-0.005535006594286976,
0.010661674260597352,
-0.007646175001553378,
0.004323674389323072,
-0.0020156334506884078,
0.0015059039636104614,
-0.020675045975131708,
0.014886467066129403,
-0.005887211333468771,
0.011014356728173951,
0.0036200491817901387,
-0.000605654048162619,
-0.012929854356949311,
-0.013633033693369278,
0.007845474911225574,
0.0022108746010011037,
0.00291615012000589 5,
0.0022109946680812925,
-0.006592804074337259,
0.004322931294738275,
0.007140500685607514,
0.005028830158984344,
0.00221137652951913,
-0.0027178392452328407,
0.006083410941526698,
0.0043244015872373545,
-0.0009560186801850922,
-0.004831306558620782,
0.009603907145548637,
0.0057323874312039565,
0.006788198309455809,
-0.007645479507462969,
-0.008000023028554343,
0.007492098431018087,
0.0015065234832235454,
0.009603887277005813,
-0.00659178683038969,
0.009251942812029757,
0.0015082568256155745,
0.013126938712435825,
-0.012225124504275856,
0.0036180429691795566,

0.0025630249350741025,
-0.004477962107515579,
-0.004478919159218666,
-0.0016615655312219365,
0.003268627433966882,
0.014887084950661062,
0.0036190804758076903,
0.003618798198747483,
-0.0002537779090307136,
0.0025638057150641188,
-0.008703134152584122,
-0.004125154189697904,
-0.005535446403482707,
-0.005181176634993146,
0.016294551587825657,
0.003972206058484148,
-0.002719143751328935,
0.003971028196027073,
0.004677345597015258,
-0.016098166515332756,
-0.0037739872220887557,
0.0215754045302883,
-0.006943052830518043,
0.00714138001216653,
-0.0009585146896095836,
0.0036187890791476436,
0.0025641489221474487,
0.006436849762717978,
-0.014337177332052933,
0.008196365255126336,
-0.0037747124630861757,
0.005731310926908317,
0.015588988069713445,
0.001859720985873802,
0.0011554832194233633,
0.025097171459653594,
0.001508034488315598,
0.0025643295034715047,
-0.0118732418349832,
-0.0006033986201563115,
0.0008038420234412933,
-0.005886515223907023,
0.0011545462602199081,
0.0018587749352700464,
0.006787902769102994,
0.002915790384713535,
0.0015080222598805012,

0.011365741172269473,
-0.00835178329643518,
0.0018589748736104023,
0.0022103339718944452,
-0.004477552337387208,
-0.009408429254857327,
0.010309916964816793,
-0.0009580719856983566,
0.003268000058978651,
0.0022107698177433006,
0.005381251048618361,
-0.00271722543569856,
0.010660492911398407,
0.0029154702448533276,
-0.0009570767987090267,
0.00854851624631418,
-0.007295705370841737,
-0.027012950881178475,
0.010310683502399966,
0.014534924597718036,
-0.006591744742249425,
-0.010111644729314771,
-0.0034221465891279123,
0.0004510578224198546,
-0.003421753955466069,
0.01488716834637179,
-0.0002550200567642308,
0.006788103621797015,
-0.00975978277258371,
0.011365132762455113,
-0.0009584398303256678,
-0.007295499182696515 5,
0.00502700202580846,
-0.015042542742335577,
-0.00025364887384123546,
0.005028103659828407,
0.002914260466256073,
0.0015064459305557283,
-0.003774389366864973,
-0.008352562213616098,
-0.004478646634280516,
0.008196679821314813,
-0.001310542188129022,
0.007140617936340692,
0.003971270165288861,
0.004675923790672843,
0.0008028117211396383,

```
 -0.012223523871538852,
 0.004323393059570642,
 -0.012224902219112233,
 -0.007999196865996493,
 -0.01011354259732806,
 -0.002367176679503633,
 -0.00764701336959321,
 0.012774857614133972,
 0.006084287495677523,
 0.0036190253693173333,
 -0.008000958022643391,
 -0.0016612771135481015]
```

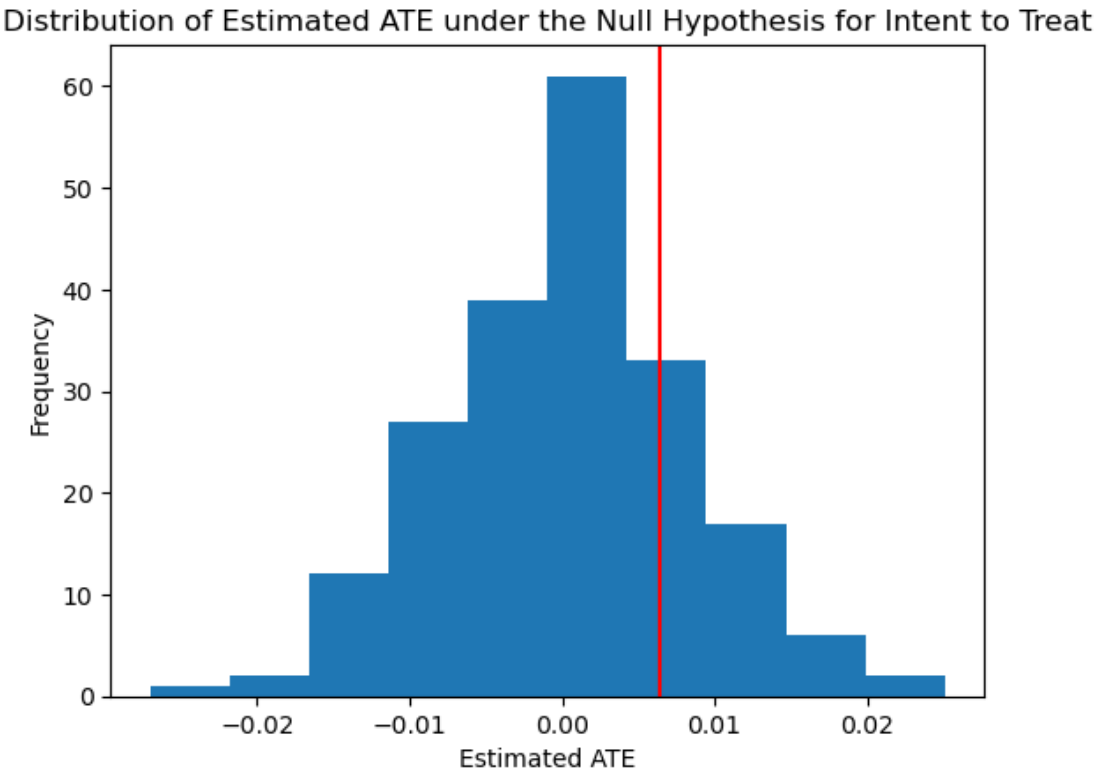
Calculate the p-value

[16]:
```
n = 0
for i in permuted_effects:
    if estimated_ate <= i:
        n+=1
n/200
```

[16]:
```
0.2
```

This p value is consistent with the authors' findings.

[17]:
```
import matplotlib.pyplot as plt
plt.hist(permuted_effects, bins=10);
plt.axvline(x=estimated_ate, color='red')
plt.xlabel('Estimated ATE')
plt.ylabel('Frequency')
plt.title('Distribution of Estimated ATE under the Null Hypothesis for Intent␣
 ↪to Treat');
```



[18]:
```
#check prereq for contacted individual
contacted_ind = dat[dat["ind_contact"] == 1]

n1 = sum(contacted_ind["pivotal"] == 1)
n0 = sum(contacted_ind["pivotal"] == 0)

print(f"The overall number of treated units in this dataset is {n1} and the␣
 ↪overall number of control units is {n0}")

expectation = 0
for i in np.arange(44):
    expectation += (sum(contacted_ind[contacted_ind["stratum_id"] ==␣
 ↪(i+1)]["pivotal"] == 1) / n1) -␣
 ↪(sum(contacted_ind[contacted_ind["stratum_id"] == (i+1)]["pivotal"] == 0) /␣
 ↪n0)

round(expectation,15)
```

```
The overall number of treated units in this dataset is 451 and the overall
number of control units is 485
```

[18]:
```
0.0
```

[19]:
```
#estimated_ate with difference-in-means as test statistic for actually␣
 ↪contacted individuals
estimated_ate_2 = 0
for i in np.arange(44):
    propensity_score = len(contacted_ind[contacted_ind["stratum_id"] == (i+1)])/
 ↪ len(contacted_ind)
    k = contacted_ind[contacted_ind["stratum_id"] == (i+1)]
    estimated_ate_2 += propensity_score * (np.mean(k[k["pivotal"] == 1]["s11"])␣
 ↪- np.mean(k[k["pivotal"] == 0]["s11"]))
estimated_ate_2
```

[19]:
```
0.012951408664130758
```

Consistent with the authors' results!

[20]:
```
permuted_effects_2 = []
for i in np.arange(100):
    permuted_ate_2 = 0
    for i in np.arange(44):
        dat_stratum = contacted_ind[contacted_ind["stratum_id"] == (i + 1)]
        permuted_treatment = np.random.permutation(dat_stratum["pivotal"])
        dat_stratum["permuted_pivotal"] = permuted_treatment
        propensity_score = len(dat_stratum )/ len(contacted_ind)
        permuted_ate_2 += propensity_score * (np.
 ↪mean(dat_stratum[dat_stratum["permuted_pivotal"] == 1]["s11"]) - np.
 ↪mean(dat_stratum[dat_stratum["permuted_pivotal"] == 0]["s11"]))
    permuted_effects_2.append(permuted_ate_2)

permuted_effects_2
```

```
/tmp/ipykernel_133/1136022369.py:7: 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-view-versus-a-copy
  dat_stratum["permuted_pivotal"] = permuted_treatment
```

[20]:
```
[0.04041518892647109,
 0.009748784323724415,
 -0.04834176505255139,
 -0.03330170437215996,
 -0.07229441335886462,
 0.016938951829623468,
 0.008044926250655938,
 0.017372636347372305,
 -0.039794674549156264,
```

```
-0.043653480899639614,
-0.032895032013965486,
0.036705882097475044,
-0.0246430623671438,
-0.002001793585462954,
-0.03909851160919309,
-0.025673794392973938,
-0.028554169620331537,
0.007856928146420375,
0.004834620902935669,
0.008816024575095904,
0.04401823276510605,
0.024756990221894197,
-0.0020709659776881042,
0.0046400660410141365,
0.0023581681884506786,
0.05516132562715783,
-0.005233698620054961,
0.005035953174353153,
-0.030430619652645714,
-0.024631461701714076,
-0.0066557236127236935,
0.016761722090916763,
0.012521912908961631,
-0.005867706858513194,
-0.013242016036986801,
0.040511721522252804,
0.009053119362737888,
-0.017849319702030242,
-0.016390335141147815,
0.0034471845620463078,
0.016719786689067498,
0.008446963607437082,
-0.02129249160106366,
-0.026384983100315437,
-0.00037539685861422405,
0.061455240226390526,
-0.00933967800572133,
-0.009441395393676225,
0.010564654380974613,
0.0226356137671277,
0.013892910268125882,
0.03128151567732422,
-0.039567454558852896,
0.02225158744865758,
0.06994647421261185,
-0.028301356049875416,
```

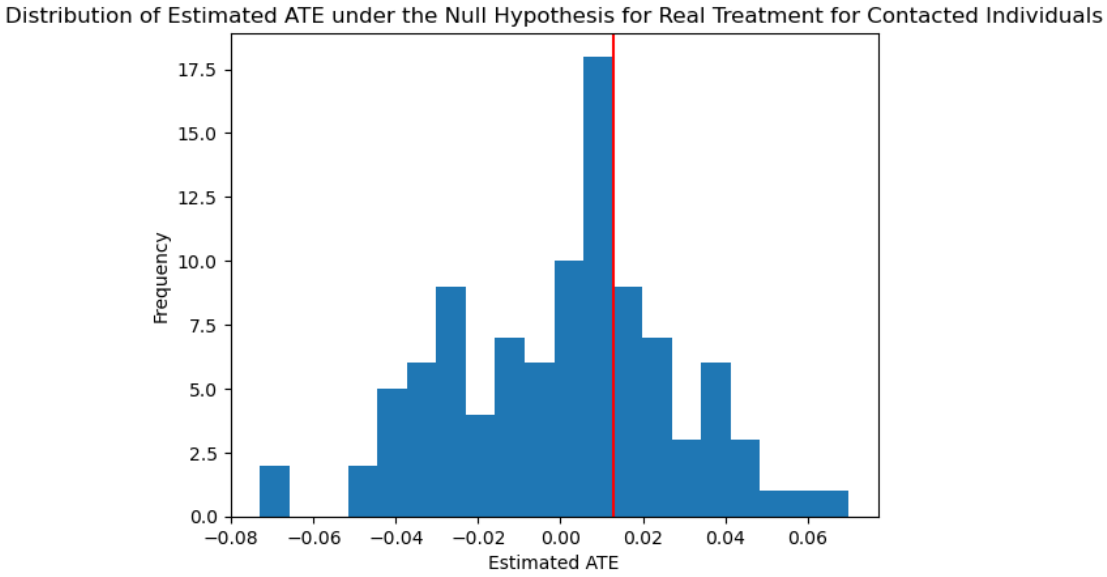
```
 0.04282600331542073,
 -0.00834494517940996,
 0.04310373846505189,
 0.03269546057901576,
 -0.023472504037567694,
 -0.009222120396856569,
 0.006254604157920067,
 0.017041228401164208,
 -0.024059948535301752,
 0.014038382949801084,
 -0.009229068922945581,
 0.00879617208917427,
 0.016117273280635863,
 0.037997258007208,
 -0.03351630299933191,
 0.0011628098053555235,
 0.010712375625585895,
 0.017234457322420217,
 0.006148172423052478,
 0.007921679733464347,
 0.012631340875712212,
 0.026819303655662596,
 -0.02194812769691149,
 -0.01387910721584773,
 -0.015173235565869721,
 -0.026653001857045878,
 -0.0010702192433114776,
 0.037416715214713514,
 0.035989077634408936,
 0.007464637231623009,
 0.0008262663872705576,
 -0.03665140133478447,
 0.011242338297013033,
 0.021095962717364997,
 -0.0364602358900329,
 0.028003219467325084,
 -0.04407564198009224,
 0.011283745384241773,
 0.00959533425923596,
 0.021533435210585598,
 -0.04707416257637944,
 -0.07307827092993718,
 0.02598398879297778,
 0.002576093245138463]
```

[21]:
```
n = 0
for i in permuted_effects_2:
```

```
    if estimated_ate_2 <= i:
        n+=1
n/100
```

[21]: 0.31

[22]:
```
plt.hist(permuted_effects_2, bins=20)
plt.axvline(x=estimated_ate_2, color='red')
plt.xlabel('Estimated ATE')
plt.ylabel('Frequency')
plt.title('Distribution of Estimated ATE under the Null Hypothesis for Real␣
 ↪Treatment for Contacted Individuals');
```



This finding is consistent with the coefficient estimate of the authors.

0.3 Part 3

0.4 Post Stratification : Matched Pair Experiment

Inspired by George Box "Block what you can and randomize what you cannot" First analyze the data within strata. Maybe do exploratory data analysis for the data overall with hispanic population and also with the other variables that were not blocked Also whats up with the whole didn't actually answer my call thing. So, maybe actually evaluate the data with that.

I will first match the data using the k nearest algorithm in python. While I do this I will make sure that the town names remain the same within each match considering that towns have an important role in voting outcomes.

[23]:
```
#mapping town names to floats so that
np.unique(contacted_ind["town"].values)

contacted_ind["town_id"] = contacted_ind["town"].map({"CHARLTON": 100, "EAST␣
 ↪BROOKFIELD": 200, "OXFORD": 300,
                                                      "SOUTHBRIDGE": 400,␣
 ↪"SPENCER": 500})
```

```
/tmp/ipykernel_133/2357145659.py:4: 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-view-versus-a-copy
  contacted_ind["town_id"] = contacted_ind["town"].map({"CHARLTON": 100, "EAST
BROOKFIELD": 200, "OXFORD": 300,
```

[24]:
```
from sklearn.neighbors import NearestNeighbors
```

[25]:
```
df = contacted_ind.drop("town", axis = 1)
df = contacted_ind.drop("treatment", axis = 1)
df = df[["age", "g08", "town_id", "s09", "g10", "dem", "hispanic", "s11",␣
 ↪"pivotal"]]
df
```

[25]:

	age	g08	town_id	s09	g10	dem	hispanic	s11	pivotal
18	41.0	0.0	100	0.0	0.0	1.0	0.0	0.0	0.0
93	83.0	0.0	400	0.0	0.0	0.0	1.0	0.0	1.0
127	63.0	0.0	400	0.0	0.0	0.0	0.0	0.0	1.0
226	59.0	0.0	100	0.0	1.0	0.0	0.0	0.0	1.0
328	51.0	0.0	100	0.0	0.0	0.0	0.0	0.0	0.0
…	…	…	…	…	…	…	…	…	…
16771	59.0	0.0	500	1.0	0.0	0.0	0.0	0.0	0.0
16782	34.0	1.0	200	0.0	1.0	0.0	0.0	0.0	1.0
16845	67.0	1.0	400	0.0	0.0	1.0	0.0	0.0	0.0
16908	33.0	1.0	100	0.0	1.0	0.0	0.0	0.0	0.0
16991	52.0	0.0	100	0.0	0.0	1.0	0.0	0.0	0.0

[936 rows x 9 columns]

[26]:
```
treatment = df[df['pivotal'] == 1]
control = df[df['pivotal'] == 0]

control_filtered = control[control["town_id"] == treatment["town_id"].iloc[0]]


# Fit KNN model on control group
```

```
knn = NearestNeighbors(n_neighbors=1, algorithm='auto').fit(control_filtered.
 ↪drop(['pivotal', "town_id"], axis=1))

# Find nearest neighbors in control group for each observation in treatment␣
 ↪group
distances, indices = knn.kneighbors(treatment.drop(['pivotal', "town_id"],␣
 ↪axis=1))

# Create matched pairs dataframe
matches = pd.concat([
    treatment.reset_index(drop=True),
    control_filtered.iloc[indices.flatten()].reset_index(drop=True),
    pd.Series(distances.flatten(), name='distance')
], axis=1)
treated = matches["s11"].iloc[:, 0]
controlled = matches["s11"].iloc[:, 1]
matches.head()
#thefirsts11ispivotal_thesecondoneisnot
```

[26]:

	age	g08	town_id	s09	g10	dem	hispanic	s11	pivotal	age	g08	\
0	83.0	0.0	400	0.0	0.0	0.0	1.0	0.0	1.0	82.0	0.0	
1	63.0	0.0	400	0.0	0.0	0.0	0.0	0.0	1.0	63.0	1.0	
2	59.0	0.0	100	0.0	1.0	0.0	0.0	0.0	1.0	58.0	1.0	
3	42.0	1.0	100	0.0	0.0	1.0	0.0	0.0	1.0	42.0	1.0	
4	85.0	1.0	100	1.0	1.0	1.0	0.0	1.0	1.0	86.0	1.0	

	town_id	s09	g10	dem	hispanic	s11	pivotal	distance
0	400	0.0	0.0	0.0	0.0	0.0	0.0	1.414214
1	400	1.0	1.0	0.0	0.0	1.0	0.0	2.000000
2	400	0.0	1.0	0.0	0.0	0.0	0.0	1.414214
3	400	0.0	0.0	1.0	1.0	0.0	0.0	1.000000
4	400	1.0	1.0	1.0	0.0	1.0	0.0	1.000000

[36]:
```
ate_matched = 0
for i in np.arange(len(matches)):
    ate_matched += treated[i] - controlled[i]
ate_matched/len(matches)
```

[36]:
```
-0.004434589800443459
```

[37]:
```
diff = treated - controlled
diff = np.array(diff)
```

[38]:
```
#paired_t_statistic
import scipy.stats as stats
from scipy.stats import ttest_rel
```

```
t, p = ttest_rel(np.array(controlled), np.array(treated))

(t, p)
```

[38]: (0.25373665032330317, 0.7998149062560772)

[41]:
```
# but we can use McNemar's statistic for binary outcome
from statsmodels.stats.contingency_tables import mcnemar


table =[[sum(treated), n- sum(treated)], [sum(controlled), n - sum(controlled) ␣
 ↪ ]]
result = mcnemar(table, exact = True, correction =True)


statistic = result.statistic
p_value = result.pvalue

# Print the results
print("McNemar's test statistic:", statistic)
print("p-value:", p_value)
```

```
McNemar's test statistic: -182.0
p-value: 0.0
```

[31]:
```
((n-sum(treated)) - sum(controlled))/(np.sqrt((n-sum(treated)) +␣
 ↪sum(controlled)))
```

[31]: -69.10882941441204

[32]:
```
#compare with normal
from scipy.stats import ttest_1samp
normal_dist = np.random.normal(0, 1, size=1000)
sdf = 0
for i in np.arange(1000):
    if 1.080634267190361 <= normal_dist[i]:
        sdf += 1
sdf/1000
#the p value is not statistically significant, again! Even in matched pair␣
 ↪experiment, nevertheless we can see that
#this estimator gives us a higher value for the treatment effects in the paper.␣
 ↪--> 14%
```

[32]: 0.142

[35]:
```
len(matches)
```

[35]: 451

[ ]:

0.5 Part 5: Regression Readjustment

Let's see how are matched pair adjustment compares to regression adjustment

[33]:
```
contacted_ind
```

[33]:
```
             id  s11 treatment  pivotal  call_attempt  ind_contact  informed  \
18         19.0  0.0  Reminder      0.0           1.0          1.0       0.0
93         94.0  0.0   Pivotal      1.0           1.0          1.0       0.0
127       128.0  0.0   Pivotal      1.0           1.0          1.0       0.0
226       227.0  0.0   Pivotal      1.0           1.0          1.0       0.0
328       329.0  0.0  Reminder      0.0           1.0          1.0       0.0
…           …  …        …          …             …            …
16771   16772.0  0.0  Reminder      0.0           1.0          1.0       0.0
16782   16783.0  0.0   Pivotal      1.0           1.0          1.0       0.0
16845   16846.0  0.0  Reminder      0.0           1.0          1.0       0.0
16908   16909.0  0.0  Reminder      0.0           1.0          1.0       0.0
16991   16992.0  0.0  Reminder      0.0           1.0          1.0       0.0

       hh_contact  stratum_id  phone_id  …  hispanic  dem  rep   age  g10  \
18            1.0         2.0      11.0  …       0.0  1.0  0.0  41.0  0.0
93            1.0         9.0      69.0  …       1.0  0.0  0.0  83.0  0.0
127           1.0        10.0      96.0  …       0.0  0.0  0.0  63.0  0.0
226           1.0        21.0     162.0  …       0.0  0.0  0.0  59.0  1.0
328           1.0        13.0     206.0  …       0.0  0.0  0.0  51.0  0.0
…           …          …        …  …       …  …  …    …  …
16771         1.0        44.0    8666.0  …       0.0  0.0  0.0  59.0  0.0
16782         1.0        24.0    8672.0  …       0.0  0.0  0.0  34.0  1.0
16845         1.0         3.0    8718.0  …       0.0  1.0  0.0  67.0  0.0
16908         1.0        21.0    8763.0  …       0.0  0.0  1.0  33.0  1.0
16991         1.0         8.0    8826.0  …       0.0  1.0  0.0  52.0  0.0

       s09  g08  s11_absentee  unmatched  town_id
18     0.0  0.0             0        0.0      100
93     0.0  0.0             0        0.0      400
127    0.0  0.0             0        0.0      400
226    0.0  0.0             0        0.0      100
328    0.0  0.0             0        0.0      100
…    …  …  …          …        …
16771  1.0  0.0             0        0.0      500
16782  0.0  1.0             0        0.0      200
16845  0.0  1.0             0        0.0      400
16908  0.0  1.0             0        0.0      100
```

```
16991  0.0  0.0              0         0.0       100

[936 rows x 22 columns]
```

[46]:
```
model = smf.ols("s11 ~ pivotal + dem + g10 + age + hispanic + hispanic*pivotal␣
↪+ g10*pivotal + pivotal*age + s09 +s09*pivotal + g08*pivotal + g08",␣
↪contacted_ind, groups=contacted_ind["stratum_id"])
model = model.fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    s11   R-squared:                       0.351
Model:                            OLS   Adj. R-squared:                  0.343
Method:                 Least Squares   F-statistic:                     41.65
Date:                Sat, 13 May 2023   Prob (F-statistic):           1.80e-78
Time:                        05:54:16   Log-Likelihood:                -474.61
No. Observations:                 936   AIC:                             975.2
Df Residuals:                     923   BIC:                             1038.
Df Model:                          12
Covariance Type:            nonrobust
================================================================================
====
                     coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
----
Intercept          -0.1741      0.070     -2.502      0.013      -0.311
-0.038
pivotal             0.1149      0.100      1.144      0.253      -0.082
0.312
dem                -0.0581      0.029     -1.988      0.047      -0.115
-0.001
g10                 0.3054      0.047      6.524      0.000       0.214
0.397
age                 0.0044      0.001      4.093      0.000       0.002
0.006
hispanic            0.1824      0.105      1.736      0.083      -0.024
0.389
hispanic:pivotal   -0.1775      0.153     -1.157      0.248      -0.479
0.124
g10:pivotal         0.0215      0.067      0.320      0.749      -0.110
0.153
pivotal:age        -0.0020      0.002     -1.297      0.195      -0.005
0.001
s09                 0.2549      0.044      5.774      0.000       0.168
0.342
```

```
s09:pivotal          0.1019      0.064      1.599      0.110      -0.023
0.227
g08                  0.1204      0.053      2.265      0.024       0.016
0.225
g08:pivotal         -0.0315      0.076     -0.414      0.679      -0.181
0.118
==============================================================================
Omnibus:                       11.434   Durbin-Watson:                   1.963
Prob(Omnibus):                  0.003   Jarque-Bera (JB):                7.889
Skew:                          -0.086   Prob(JB):                       0.0194
Kurtosis:                       2.585   Cond. No.                         923.
==============================================================================

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

/opt/conda/lib/python3.9/site-packages/statsmodels/base/model.py:127:
ValueWarning: unknown kwargs ['groups']
  warnings.warn(msg, ValueWarning)
```

[48]:
```
model = smf.ols("s11 ~ pivotal + dem + rep + age + hispanic + hispanic*pivotal␣
↪+ dem*pivotal + pivotal*age  ", contacted_ind,␣
↪groups=contacted_ind["stratum_id"])
model = model.fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    s11   R-squared:                       0.073
Model:                            OLS   Adj. R-squared:                  0.065
Method:                 Least Squares   F-statistic:                     9.127
Date:                Sat, 13 May 2023   Prob (F-statistic):           3.89e-12
Time:                        05:55:41   Log-Likelihood:                -641.67
No. Observations:                 936   AIC:                             1301.
Df Residuals:                     927   BIC:                             1345.
Df Model:                           8
Covariance Type:            nonrobust
==============================================================================
====
                     coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
----
Intercept          -0.0061      0.075     -0.081      0.936      -0.154
0.142
pivotal             0.0678      0.108      0.627      0.531      -0.144
0.280
```

```
dem                -0.0520      0.049     -1.067      0.286      -0.148
0.044
rep                 0.0212      0.051      0.417      0.677      -0.079
0.121
age                 0.0080      0.001      6.457      0.000       0.006
0.010
hispanic            0.0517      0.125      0.414      0.679      -0.193
0.297
hispanic:pivotal    0.0157      0.181      0.087      0.931      -0.340
0.371
dem:pivotal         0.0307      0.070      0.441      0.660      -0.106
0.167
pivotal:age        -0.0010      0.002     -0.532      0.595      -0.005
0.003
==============================================================================
Omnibus:                     4544.245   Durbin-Watson:                   1.934
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              114.242
Skew:                           0.079   Prob(JB):                     1.56e-25
Kurtosis:                       1.296   Cond. No.                         911.
==============================================================================

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

/opt/conda/lib/python3.9/site-packages/statsmodels/base/model.py:127:
ValueWarning: unknown kwargs ['groups']
  warnings.warn(msg, ValueWarning)
```

[42]:
```
matches["diff"] = treated - controlled
```

[42]:

	age	g08	town_id	s09	g10	dem	hispanic	s11	pivotal	age	g08	\
0	83.0	0.0	400	0.0	0.0	0.0	1.0	0.0	1.0	82.0	0.0	
1	63.0	0.0	400	0.0	0.0	0.0	0.0	0.0	1.0	63.0	1.0	
2	59.0	0.0	100	0.0	1.0	0.0	0.0	0.0	1.0	58.0	1.0	
3	42.0	1.0	100	0.0	0.0	1.0	0.0	0.0	1.0	42.0	1.0	
4	85.0	1.0	100	1.0	1.0	1.0	0.0	1.0	1.0	86.0	1.0	
..	…	…	…	…	…	…	…	…	…	…	…	
446	44.0	1.0	500	0.0	0.0	1.0	1.0	0.0	1.0	44.0	1.0	
447	60.0	1.0	500	0.0	1.0	0.0	0.0	1.0	1.0	60.0	1.0	
448	42.0	1.0	500	0.0	1.0	0.0	0.0	0.0	1.0	42.0	1.0	
449	64.0	1.0	500	0.0	1.0	0.0	0.0	0.0	1.0	65.0	1.0	
450	34.0	1.0	200	0.0	1.0	0.0	0.0	0.0	1.0	34.0	0.0	

	town_id	s09	g10	dem	hispanic	s11	pivotal	distance	diff
0	400	0.0	0.0	0.0	0.0	0.0	0.0	1.414214	0.0
1	400	1.0	1.0	0.0	0.0	1.0	0.0	2.000000	-1.0
2	400	0.0	1.0	0.0	0.0	0.0	0.0	1.414214	0.0

```
3        400  0.0  0.0  1.0       1.0  0.0      0.0  1.000000   0.0
4        400  1.0  1.0  1.0       0.0  1.0      0.0  1.000000   0.0
..       …    …    …    …         …    …        …        …    …
446      400  0.0  1.0  0.0       0.0  0.0      0.0  1.732051   0.0
447      400  0.0  0.0  0.0       0.0  1.0      0.0  1.000000   0.0
448      400  0.0  1.0  0.0       0.0  1.0      0.0  1.000000  -1.0
449      400  0.0  1.0  0.0       0.0  0.0      0.0  1.000000   0.0
450      400  0.0  0.0  1.0       1.0  0.0      0.0  2.000000   0.0

[451 rows x 20 columns]
```

[44]:
```
model = sm.OLS(matches['diff'], sm.add_constant(matches[['dem', 'age',␣
 ↪"hispanic", "s09", "g10", "town_id", "g08"]]))
# Get regression results
results = model.fit()

# Print summary of regression results
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   diff   R-squared:                       0.065
Model:                            OLS   Adj. R-squared:                  0.037
Method:                 Least Squares   F-statistic:                     2.323
Date:                Sat, 13 May 2023   Prob (F-statistic):            0.00544
Time:                        05:45:46   Log-Likelihood:                -177.37
No. Observations:                 451   AIC:                             382.7
Df Residuals:                     437   BIC:                             440.3
Df Model:                          13
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
dem           -0.0443      0.049     -0.902      0.368      -0.141       0.052
dem            0.0243      0.054      0.449      0.654      -0.082       0.131
age           -0.0099      0.025     -0.392      0.696      -0.060       0.040
age            0.0089      0.025      0.350      0.727      -0.041       0.059
hispanic       0.0587      0.101      0.579      0.563      -0.140       0.258
hispanic      -0.0181      0.083     -0.218      0.828      -0.181       0.145
s09            0.0612      0.051      1.199      0.231      -0.039       0.162
s09           -0.0546      0.057     -0.960      0.338      -0.166       0.057
g10            0.0801      0.050      1.591      0.112      -0.019       0.179
g10           -0.1829      0.058     -3.163      0.002      -0.297      -0.069
town_id    -6.179e-05      0.000     -0.557      0.578      -0.000       0.000
town_id        0.0004      0.000      1.979      0.048    2.77e-06       0.001
g08            0.0335      0.053      0.629      0.530      -0.071       0.138
g08           -0.0592      0.062     -0.959      0.338      -0.181       0.062
```

```
==============================================================================
Omnibus:                       44.821   Durbin-Watson:                   2.111
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              257.993
Skew:                           0.019   Prob(JB):                     9.50e-57
Kurtosis:                       6.705   Cond. No.                     3.07e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 3.07e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[ ]: