Tarea3_Jara_Retamal

December 10, 2022

0.0.1 juan.caro@uni.lu el dia 28/11 hasta las 21:00.

1 Tarea 3: Experimentos y pseudo-experimentos

##Laboratorio de métodos aplicados avanzados

Integrantes:

- Francisca Carolina Jara Yévenes
- Luis Fernando Retamal Fuentes

Fecha: 28 de noviembre de 2022

#Bibliotecas requeridas y lectura de archivos

[ ]:
```
%pip install linearmodels
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
import statsmodels.formula.api as smf
import sklearn
import scipy
from scipy.linalg import eigh, cholesky
from scipy.stats import norm
import linearmodels.panel as lmp
from pylab import plot, show, axis, subplot, xlabel, ylabel, grid

%matplotlib inline
np.random.seed(123) #set seed
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
wheels/public/simple/
Requirement already satisfied: linearmodels in /usr/local/lib/python3.7/dist-
packages (4.25)
Requirement already satisfied: Cython>=0.29.21 in /usr/local/lib/python3.7/dist-
packages (from linearmodels) (0.29.32)
Requirement already satisfied: mypy-extensions>=0.4 in
/usr/local/lib/python3.7/dist-packages (from linearmodels) (0.4.3)
```

```
Requirement already satisfied: scipy>=1.2 in /usr/local/lib/python3.7/dist-
packages (from linearmodels) (1.7.3)
Requirement already satisfied: pandas>=0.24 in /usr/local/lib/python3.7/dist-
packages (from linearmodels) (1.3.5)
Requirement already satisfied: formulaic in /usr/local/lib/python3.7/dist-
packages (from linearmodels) (0.5.2)
Requirement already satisfied: property-cached>=1.6.3 in
/usr/local/lib/python3.7/dist-packages (from linearmodels) (1.6.4)
Requirement already satisfied: pyhdfe>=0.1 in /usr/local/lib/python3.7/dist-
packages (from linearmodels) (0.1.1)
Requirement already satisfied: numpy>=1.16 in /usr/local/lib/python3.7/dist-
packages (from linearmodels) (1.21.6)
Requirement already satisfied: patsy in /usr/local/lib/python3.7/dist-packages
(from linearmodels) (0.5.3)
Requirement already satisfied: statsmodels>=0.11 in
/usr/local/lib/python3.7/dist-packages (from linearmodels) (0.12.2)
Requirement already satisfied: python-dateutil>=2.7.3 in
/usr/local/lib/python3.7/dist-packages (from pandas>=0.24->linearmodels) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-
packages (from pandas>=0.24->linearmodels) (2022.6)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-
packages (from python-dateutil>=2.7.3->pandas>=0.24->linearmodels) (1.15.0)
Requirement already satisfied: wrapt>=1.0 in /usr/local/lib/python3.7/dist-
packages (from formulaic->linearmodels) (1.14.1)
Requirement already satisfied: typing-extensions>=4.2.0 in
/usr/local/lib/python3.7/dist-packages (from formulaic->linearmodels) (4.4.0)
Requirement already satisfied: graphlib-backport>=1.0.0 in
/usr/local/lib/python3.7/dist-packages (from formulaic->linearmodels) (1.0.3)
Requirement already satisfied: cached-property>=1.3.0 in
/usr/local/lib/python3.7/dist-packages (from formulaic->linearmodels) (1.5.2)
Requirement already satisfied: astor>=0.8 in /usr/local/lib/python3.7/dist-
packages (from formulaic->linearmodels) (0.8.1)
Requirement already satisfied: interface-meta>=1.2.0 in
/usr/local/lib/python3.7/dist-packages (from formulaic->linearmodels) (1.3.0)
```

#Parte 1 - Experimentos

Deben conceptualizar un experimento con el objetivo de estudiar posibles incentivos o estrategias para incrementar la asistencia a clases en estudiantes universitarios de la UdeC. El outcome del tratamiento es la proporcion promedio de estudiantes que asisten a clases. Todos los elementos del experimento deben ser definidos, respondiendo a las siguientes preguntas:

1.1 Pregunta 1

Asumiendo la existencia de recursos disponibles e implementacion a nivel de estudiante, sugiera un tratamiento que pueda ser testeado a traves de un experimento

aleatorizado controlado. Sea especifico en cuanto a los detalles del tratamiento (costos, materiales, duracion, etcetera).

El experimento consiste en resolver un par de preguntas estilo certamen al final de la clase. Una vez revisada la materia, el profesor o profesora procederá a exhibir el enunciado de uno o dos ejercicios relacionados con la materia vista y resolverá en conjunto con los alumnos. Todo esto será realizado al final de la clase, durando aproximadamente 10 minutos. La duración total del experimento corresponde a un semestre académico. El tratamiento se implementa en la segunda mitad del semestre.

Supuestos:

- La cantidad de estudiantes inscritos en el curso es suficientemente grande para que ambos grupos se parezcan en las caracteristicas observables y no observables. Por lo tanto, el tamaño total se estima en aproximadamente 200 alumnos.
- El calendario académico se mantiene sin cambios desde el incio al fin del semestre académico.

1.2 Pregunta 2

Defina los grupos de tratamiento y control para implementar su experimento. Describa en detalle el mecanismo de asignación aleatorio que permite la comparacion entre grupos.

Se forman dos secciones. Cada sección tiene la misma cantidad de alumnos, el mismo horario y se les pasa el mismo contenido por clase. Obviamente, cada sección tendrá un profesor y ambos deben que cumplir con las mismas aptitudes para ser docente del ramo. Dichas aptitudes son las impuestas por el departamento encargado de impartir la asignatura. Como ambas secciones tienen el mismo horario de inicio y de término, el grupo de tratamiento debe pasar la materia considerando que los últimos 10 minutos de la clase son para el desarrollo de ejercicios. El tratamiento se implementa a las dos semanas de iniciado el semestre.

Los alumnos que tomarán la asignatura son asignados al azar en alguna de las secciones, procurando que se tengan el mismo número de alumnos en cada grupo al finalizar el proceso de asignación. A través de un sistema computacional, se les asignará a cada uno un número entre 0 y el 1 de forma aleatoria. Los alumnos que tengan entre menor o igual a 0.5, serán parte del grupo 1: el grupo de control. Los alumnos que tengan un número mayor a 0.5, serán asignados al grupo 2: el grupo de tratamiento. En caso de que se tengan cantidades de alumnos diferentes, elegir una muestra aleatoria de tamaño de esa diferencia del grupo con mayor número de estudiantes y repetir el proceso antes mencionado hasta que se iguale la cantidad de estudiantes.

1.3 Pregunta 3

Que metodo considera el mas apropiado para la estimacion del efecto promedio? (pre-test, pre-post test, Salomon 4 group). Justifique su respuesta en base a las ventajas y desventajas de cada metodo.

El método utilizado es el de ***Post test***, el cual mide las diferencias después del programa entre aquellos que participaron en el programa y aquellos que no participaron. El grupo de comparación corresponde a los individuos que no participaron en el programa (por alguna razón), y para los cuales tenemos datos después del programa. * ***Supuestos clave:*** Los no participantes son idénticos a los participantes excepto por la intervención del programa. No hay ninguna selección en el tipo

de persona que entró al programa. * ***Ventajas:*** Muchas veces ya existen datos administrativos que se pueden analizar retrospectivamente. No requiere datos de la situación anterior al programa. * ***Desventajas:*** Necesita un grupo de control. Si los grupos tratados y no tratados (control) son distintos antes del programa, el método puede sub estimar o sobre estimar el impacto verdadero de la política; es decir se introduce un sesgo de selección en la estimación.

Otros métodos:

Pre-post test : Mide el cambio en los resultados de los participantes de un programa en el tiempo. Es la diferencia entre la situación anterior y posterior a un tratamiento. El grupo de comparación consiste en los mismos participantes del programa antes de su inicio.

- ***Supuestos clave:*** El programa es el único factor que influyó en el cambio del resultado. Sin el programa el resultado se hubiera mantenido igual.

- ***Ventajas:*** Muchas veces ya existen datos administrativos que se pueden analizar retrospectivamente. No requiere datos de personas que no participaron al programa.

- ***Desventajas:*** Muchos factores cambian con el tiempo y pueden afectar el resultado, lo que va en contra del supuesto clave. En particular, la comparación pre-post no controla por el efecto de la tendencia secular (tendencia natural del resultado a través del tiempo) o de choques que cambien el resultado pero no tienen que ver con el programa.

Solomon 4 group: Tiene como finalidad controlar la interacción de la medida pretratamiento con la variable independiente. Consta de 4 grupos: dos son de tratamiento, dos de control, dos con medidas pre y post tratamiento, y dos sólo con medidas post tratamiento. * ***Ventajas:*** Verifica los posibles efectos de la pre tratamiento sobre el post tratamiento. Comprueba explícitamente la posible interacción entre la medida pre y el tratamiento. * ***Desventajas:*** No existe la manipulación ni grupo con el cual se establezca una comparación.

1.4 Pregunta 4

Ahora suponga que no es posible implementar un experimento a nivel de estudiante, sino a nivel de clase. Como ajustaria los elementos de su experimento para poder ser implementado a nivel de cluster? Sea especifico respecto tanto del tratamiento como del metodo de asignacion aleatorio y potencial comparacion entre grupos de tratamiento y control.

Se utilizaría el método de ***RCT de cluster***. Consiste en comparar el cambio en los resultados de los participantes con el cambio en los resultados de los que no participaron en el programa. Para este experimento, se imita el proceso descrito anteriormente con la diferencia de inlcuir otras asignaturas. Deben ser ramos impartidos por diversas facultades y departamentos, en las que se pueda medir lo aprendido a través evaluaciones escritas. Se incluyen dos grupos por asignatura, uno de control y otro de tratamiento, donde ambos tengan el mismo tamaño, mismo horario y mismos contenidos por clase. Estas características pueden variar entre clusters, pero no entre los individuos de un mismo grupo.

El mecanismo de asignación aleatoria es el mismo definido en la Pregunta 2. Se utiliza para separar por secciones a los alumnos de una misma clase. El tamaño de las secciones de una misma clase deben ser iguales, pero pueden variar entre asignaturas.

Ventajas: Tiene implicancias prácticas en la estimación de los esrrores estándar (correlación intra grupos).

Desventajas: Mayor riesgo de contaminación.

1.5 Pregunta 5

Suponga que en vez de un experimento, se planifica que sea un programa implementado a nivel de toda la Universidad. Como ajustaria los elementos descritos anteriormente para poder comparar el efecto de la intervencion.

De la misma forma definida en el inciso anterior, el experimento se realiza utilizando el método RTC cluster. Para las clases donde se evalúa el contenido de forma oral, el tratamiento consiste en responder una pregunta simulando este tipo de evalución.

Referencias: * Pomeranz, D. (2011). MÉTODOS DE EVALUACIÓN. https://www.hbs.edu/ris/Supplemental%20Files/Metodos-de-Evaluacion-de-Impacto_50067.pdf

- Rondón, X. & Buitrago, N. (s. f.). Diseño de cuatro grupos de Solomon. prezi.com. https://prezi.com/oz8bpp7blv8_/diseno-de-cuatro-grupos-de-solomon/

#Parte 2 - Estimacion de efectos promedio de tratamiento (data simulada)

1.6 Pregunta 6

A partir de sus respuestas en Parte 1, genere data para 40 grupos (considere cada grupo como una clase) con 50 estudiantes cada uno (asuma que los estudiantes son asignados aleatoriamente a cada clase). Cada estudiante debe tener data de asistencia en un periodo, generando una variable binaria aleatoria tal que la asistencia promedio a traves de todos los grupos es de 80%.

[ ]:
```
from pandas.core.groupby.groupby import T
# experiment parameters
np.random.seed(1293) #set seed
nsize = 2000  #sample size = 40*50

# we create simulated data starting from a given variance-covariance matrix

# variance-covariance matrix (simetric)
cov = np.array([
        [  3.40, -2.75, -2.00],
        [ -2.75,  5.50,  1.50],
        [ -2.00,  1.50,  1.25]
    ])

X = norm.rvs(size=(3, nsize)) #vector of variables N(0,1)
evals, evecs = eigh(cov) #eigenvalues and eigenvector from var-covar matrix
c = np.dot(evecs, np.diag(np.sqrt(evals))) #cholesky decomposition
Xc = np.dot(c, X) #introduce correlation to matrix X
```

```
Xc = Xc.transpose()
Xc = pd.DataFrame(Xc, columns=['X1','X2','X3'])

#time periods and treatment asignment
Xc['p'] = 1
Xc.loc[0:999,'p'] = 1
tr = np.random.binomial(1, 0.5, size=1000) #treatment status
Xc.loc[0:999,'T'] = tr
Xc.loc[1000:1999,'T'] = tr


#grouping variable: 40 groups of 50
Xc['cl']=1
t = 2
for i in range(50, 1999, 50):
  j = i + 49
  Xc.loc[i : j, 'cl'] = t
  Xc.loc[i+1000: j+1000, 'cl'] = t
  t = t + 1

#outcome variable: Asistencia del alumno.
Xc['y'] = 10*(1+Xc['X1']) + 10*(Xc['X2']+Xc['T']*Xc['p']) + Xc['X3']

#Para que la asistencia promedio a traves de todos los grupos sea de 80%
for i in range(0,1999):
  if Xc['y'][i] <= 31:
    Xc.loc[i, 'y'] = 1
  else:
    Xc.loc[i, 'y']=0

#data description
Xc.describe()
```

[ ]:

	X1	X2	X3	p	T	cl	y
count	2000.000000	2000.000000	2000.000000	2000.0	2000.000000	2000.000000	2000.000000
mean	-0.001575	0.016372	-0.001424	1.0	0.492000	20.500000	0.804996
std	1.853244	2.365134	1.123714	0.0	0.500061	11.546283	0.511851
min	-5.529312	-6.903029	-4.132867	1.0	0.000000		
25%	-1.296389	-1.597809	-0.769206	1.0	0.000000		
50%	0.025012	-0.064607	-0.014518	1.0	0.000000		
75%	1.269114	1.667424	0.753173	1.0	1.000000		
max	6.385734	8.889169	3.435354	1.0	1.000000		

```
min        1.000000     0.000000
25%       10.750000     1.000000
50%       20.500000     1.000000
75%       30.250000     1.000000
max       40.000000    14.991904
```

1.7 Pregunta 7

Genere un mecanismo de asignacion aleatorio a nivel de estudiante y muestre que en la data generada permite que ambos grupos (tratamiento y control) tienen una asistencia promedio comparable.

[ ]:
```
from statsmodels.stats.power import TTestIndPower

xc=Xc['y'][0:999] # control group
xt=Xc['y'][1000:1999] # treatment group

# The estimated difference of the means
diff = np.abs(xc.mean()-xt.mean())

# Polled estimated standard deviations
d = np.sqrt((xc.std()**2 + xt.std()**2)/2)

# parameters for power analysis
effect = diff/d
alpha = 0.05
power = 0.8

# perform power analysis
analysis = TTestIndPower()
result = analysis.solve_power(effect, power = power, nobs1= None, ratio = 1.0,␣
 ↪alpha = alpha)
print('Sample Size: %.3f' % round(result))
```

```
Sample Size: 8734.000
```

1.8 Pregunta 8

Genere un tratamiento que imcrementa la participacion en el grupo de tratamiento en 10 puntos porcentuales. Ademas en la data posterior al experimento, asuma que la participacion promedio cayo a 75%. Estime el efecto promedio del tratamiento usando solo post-test.

[ ]:
```
#post-test
y = Xc['y']
X = Xc['T']
X = sm.add_constant(X)
model = sm.OLS(y, X)
```

```
results = model.fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.021
Model:                            OLS   Adj. R-squared:                  0.020
Method:                 Least Squares   F-statistic:                     42.82
Date:                Mon, 28 Nov 2022   Prob (F-statistic):           7.60e-11
Time:                        23:14:31   Log-Likelihood:                -1476.7
No. Observations:                2000   AIC:                             2957.
Df Residuals:                    1998   BIC:                             2969.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.8779      0.016     55.242      0.000       0.847       0.909
T             -0.1483      0.023     -6.544      0.000      -0.193      -0.104
==============================================================================
Omnibus:                     3297.617   Durbin-Watson:                   1.640
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          7540041.981
Skew:                          10.133   Prob(JB):                         0.00
Kurtosis:                     303.116   Cond. No.                         2.60
==============================================================================

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

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142:
FutureWarning: In a future version of pandas all arguments of concat except for
the argument 'objs' will be keyword-only
  x = pd.concat(x[::order], 1)
```

1.9 Pregunta 9

Estime el efecto promedio del tratamiento usando pre-post test con la data generada. Muestre que el efecto es equivalente usando ambos metodos.

[ ]:
```
#pre-post test
y=Xc['y']
Xc['dd']= Xc['p']*Xc['T']
X=Xc[['p','T','dd']]
X = sm.add_constant(X)
model = sm.OLS(y, X)
results2 = model.fit()
print(results2.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.019
Model:                            OLS   Adj. R-squared:                  0.018
Method:                 Least Squares   F-statistic:                     19.27
Date:                Mon, 28 Nov 2022   Prob (F-statistic):           5.14e-09
Time:                        23:15:05   Log-Likelihood:                -1478.8
No. Observations:                2000   AIC:                             2964.
Df Residuals:                    1997   BIC:                             2980.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
p              0.8779      0.016     55.170      0.000       0.847       0.909
T          -4.058e+12    2.8e+12     -1.448      0.148   -9.55e+12    1.44e+12
dd          4.058e+12    2.8e+12      1.448      0.148   -1.44e+12    9.55e+12
==============================================================================
Omnibus:                     3298.793   Durbin-Watson:                   1.641
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          7577647.516
Skew:                          10.139   Prob(JB):                         0.00
Kurtosis:                     303.866   Cond. No.                     4.54e+14
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The smallest eigenvalue is 1.64e-26. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142:
FutureWarning: In a future version of pandas all arguments of concat except for
the argument 'objs' will be keyword-only
  x = pd.concat(x[::order], 1)
```

1.10 Pregunta 10

Estime el efecto ajustando los errores estandar por cluster (la variable grupo representa cada clase). Cual es la diferencia entre ambas estimaciones? Explique porque es esperable (o no) encontrar diferencias entre ambos metodos.

[ ]:
```
#clustered standard errors
results3 = model.fit(cov_type="cluster", cov_kwds={'groups': Xc['cl']})
print(results3.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.019
Model:                            OLS   Adj. R-squared:                  0.018
```

```
Method:                 Least Squares   F-statistic:                     3204.
Date:                Mon, 28 Nov 2022   Prob (F-statistic):           5.53e-44
Time:                        23:15:46   Log-Likelihood:                -1478.8
No. Observations:                2000   AIC:                             2964.
Df Residuals:                    1997   BIC:                             2980.
Df Model:                           2
Covariance Type:            cluster
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
p              0.8779      0.017     52.354      0.000       0.845       0.911
T          -4.058e+12   6.77e+10    -59.961      0.000   -4.19e+12   -3.93e+12
dd          4.058e+12   5.11e+10     79.341      0.000    3.96e+12    4.16e+12
==============================================================================
Omnibus:                     3298.793   Durbin-Watson:                   1.641
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          7577647.516
Skew:                          10.139   Prob(JB):                         0.00
Kurtosis:                     303.866   Cond. No.                     4.54e+14
==============================================================================

Notes:
[1] Standard Errors are robust to cluster correlation (cluster)
[2] The smallest eigenvalue is 1.64e-26. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

#Parte 3 - Experimentos naturales

Usando la data charls.csv, responda las siguientes preguntas relativas a experimentos naturales.

[90]:
```
charls = pd.read_csv('charls.csv')
charls.dropna(inplace=True) ##remueve faltantes
charls.reset_index(drop=True, inplace=True) ##resetea indices
charls.describe() ##muestra principales estadisticas
```

[90]:
```
                age         bnrps          cesd         child         dnrps  \
count  21045.000000  21045.000000  21045.000000  21045.000000  21045.000000
mean      59.386553     59.610683      8.656878      2.825232      0.259111
std        9.016106     51.905928      6.307677      1.372179      0.438157
min       20.000000      0.000000      0.000000      0.000000      0.000000
25%       52.000000      0.000000      4.000000      2.000000      0.000000
50%       59.000000     60.000000      7.000000      3.000000      0.000000
75%       65.000000     74.875404     12.000000      4.000000      1.000000
max       95.000000    300.000000     30.000000     10.000000      1.000000

             female        hrsusu         hsize      intmonth       married  \
count  21045.000000  21045.000000  21045.000000  21045.000000  21045.000000
mean       0.521026      2.548166      3.585222      7.511143      0.907674
```

```
std        0.499570      1.757182      1.720136      0.865851      0.289492
min        0.000000      0.000000      1.000000      1.000000      0.000000
25%        0.000000      0.000000      2.000000      7.000000      1.000000
50%        1.000000      3.401197      3.000000      7.000000      1.000000
75%        1.000000      4.025352      5.000000      8.000000      1.000000
max        1.000000      5.123964     16.000000     12.000000      1.000000

              nrps        retage       retired        schadj         urban  \
count  21045.000000  21045.000000  21045.000000  21045.000000  21045.000000
mean       0.519078      1.280969      0.204942      4.162414      0.206652
std        0.499648      3.830963      0.403669      3.540039      0.404914
min        0.000000      0.000000      0.000000      0.000000      0.000000
25%        0.000000      0.000000      0.000000      0.000000      0.000000
50%        1.000000      0.000000      0.000000      4.000000      0.000000
75%        1.000000      0.000000      0.000000      8.000000      0.000000
max        1.000000     51.000000      1.000000     16.000000      1.000000

              wave        wealth          inid
count  21045.000000  2.104500e+04  21045.000000
mean       1.909385  6.783959e+03  12747.082870
std        0.817975  5.453065e+04   7769.025809
min        1.000000 -1.648450e+06      1.000000
25%        1.000000  1.000000e+02   5176.000000
50%        2.000000  1.000000e+03  13314.000000
75%        3.000000  6.800000e+03  19650.000000
max        3.000000  1.040000e+06  25403.000000
```

[91]:
```
cantidad = charls.age.count() ##cantidad de elementos en el dataframe
```

1.11 Pregunta 11

Simule un experimento natural (e.g. intervencion de politica publica) tal que se reduce la proporcion de individuos con 3 hijos o mas que declaran beber alcohol en el tercer periodo a la mitad. Para ello, genere una variable de tratamiento (todos los individuos con mas de 2 hijos son parte de la intervencion), y una nueva variable llamada sdrinlky, talque es identica a drinkly en los periodos 1 y 2 , pero sustituya los valores aleatoriamente en el periodo 3 para generar el efecto esperado.

Se introduce una variable para el tratamiento, en este caso una variable binaria ("tres_o_mas") que representa si el individuo tiene 3 o más hijos. Luego se introduce la variable "sdrinlky" que representa los valores aleatorios en el periodo 3.

[92]:
```
list1=[]
for i in range (cantidad):
  if (charls.child[i]>2):
    list1.append(1)
  else:
    list1.append(0)
charls.insert(4, "tres_o_mas", list1,False)
```

```
list2=[]
for i in range (cantidad):
  if (charls.wave[i]==1 or charls.wave[i]==2):
    if (charls.drinkly[i]=='0'):
      list2.append(0)
    else: list2.append(1)
  else:
    a=np.random.randint(2)
    list2.append(a)


charls.insert(6, "sdrinlky", list2,False)
```

[93]:
```
charls.describe()
```

[93]:

	age	bnrps	cesd	child	tres_o_mas
count	21045.000000	21045.000000	21045.000000	21045.000000	21045.000000
mean	59.386553	59.610683	8.656878	2.825232	0.518983
std	9.016106	51.905928	6.307677	1.372179	0.499651
min	20.000000	0.000000	0.000000	0.000000	0.000000
25%	52.000000	0.000000	4.000000	2.000000	0.000000
50%	59.000000	60.000000	7.000000	3.000000	1.000000
75%	65.000000	74.875404	12.000000	4.000000	1.000000
max	95.000000	300.000000	30.000000	10.000000	1.000000

	dnrps	sdrinlky	female	hrsusu	hsize
count	21045.000000	21045.000000	21045.000000	21045.000000	21045.000000
mean	0.259111	0.380375	0.521026	2.548166	3.585222
std	0.438157	0.485491	0.499570	1.757182	1.720136
min	0.000000	0.000000	0.000000	0.000000	1.000000
25%	0.000000	0.000000	0.000000	0.000000	2.000000
50%	0.000000	0.000000	1.000000	3.401197	3.000000
75%	1.000000	1.000000	1.000000	4.025352	5.000000
max	1.000000	1.000000	1.000000	5.123964	16.000000

	intmonth	married	nrps	retage	retired
count	21045.000000	21045.000000	21045.000000	21045.000000	21045.000000
mean	7.511143	0.907674	0.519078	1.280969	0.204942
std	0.865851	0.289492	0.499648	3.830963	0.403669
min	1.000000	0.000000	0.000000	0.000000	0.000000
25%	7.000000	1.000000	0.000000	0.000000	0.000000
50%	7.000000	1.000000	1.000000	0.000000	0.000000
75%	8.000000	1.000000	1.000000	0.000000	0.000000
max	12.000000	1.000000	1.000000	51.000000	1.000000

	schadj	urban	wave	wealth	inid

count	21045.000000	21045.000000	21045.000000	2.104500e+04	21045.000000
mean	4.162414	0.206652	1.909385	6.783959e+03	12747.082870
std	3.540039	0.404914	0.817975	5.453065e+04	7769.025809
min	0.000000	0.000000	1.000000	-1.648450e+06	1.000000
25%	0.000000	0.000000	1.000000	1.000000e+02	5176.000000
50%	4.000000	0.000000	2.000000	1.000000e+03	13314.000000
75%	8.000000	0.000000	3.000000	6.800000e+03	19650.000000
max	16.000000	1.000000	3.000000	1.040000e+06	25403.000000

[94]:
```
Xa=charls[['age','bnrps','cesd','child','tres_o_mas','dnrps','female','hrsusu','intmonth','mar
Xa = sm.add_constant(Xa)
ya=charls['sdrinlky']
model = sm.OLS(ya, Xa)
results = model.fit(cov_type="HC1")
print(results.summary())
```

OLS Regression Results

Dep. Variable:	sdrinlky	R-squared:	0.124
Model:	OLS	Adj. R-squared:	0.123
Method:	Least Squares	F-statistic:	193.1
Date:	Mon, 28 Nov 2022	Prob (F-statistic):	0.00
Time:	23:55:52	Log-Likelihood:	-13260.
No. Observations:	21045	AIC:	2.656e+04
Df Residuals:	21027	BIC:	2.670e+04
Df Model:	17		
Covariance Type:	HC1		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.3870	0.045	8.655	0.000	0.299	0.475
age	-0.0008	0.000	-1.844	0.065	-0.002	5.3e-05
bnrps	9.791e-05	8.79e-05	1.114	0.265	-7.44e-05	0.000
cesd	0.0006	0.001	1.208	0.227	-0.000	0.002
child	0.0036	0.004	0.948	0.343	-0.004	0.011
tres_o_mas	-0.0030	0.010	-0.306	0.760	-0.022	0.016
dnrps	0.0062	0.012	0.499	0.618	-0.018	0.030
female	-0.2891	0.007	-40.398	0.000	-0.303	-0.275
hrsusu	0.0151	0.003	5.843	0.000	0.010	0.020
intmonth	-0.0069	0.004	-1.967	0.049	-0.014	-2.4e-05
married	0.0347	0.011	3.301	0.001	0.014	0.055
nrps	-0.0157	0.008	-1.858	0.063	-0.032	0.001
retage	0.0029	0.001	3.399	0.001	0.001	0.005
retired	-0.0138	0.011	-1.243	0.214	-0.036	0.008
schadj	0.0022	0.001	2.091	0.037	0.000	0.004
urban	-0.0003	0.008	-0.042	0.966	-0.016	0.016
wave	0.0803	0.005	16.646	0.000	0.071	0.090

```
wealth     -1.026e-08   5.91e-08     -0.174      0.862   -1.26e-07    1.06e-07
==============================================================================
Omnibus:                   1122928.449   Durbin-Watson:                   1.751
Prob(Omnibus):                   0.000   Jarque-Bera (JB):             1970.382
Skew:                            0.338   Prob(JB):                         0.00
Kurtosis:                        1.662   Cond. No.                     8.02e+05
==============================================================================

Notes:
[1] Standard Errors are heteroscedasticity robust (HC1)
[2] The condition number is large, 8.02e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142:
FutureWarning: In a future version of pandas all arguments of concat except for
the argument 'objs' will be keyword-only
  x = pd.concat(x[::order], 1)
```

1.12 Pregunta 12

Estime el efecto del tratamiento usando diferencias en diferencias, comparando entre los periodos 2 y 3.

Para la estimación del efecto de la intervención mediante el método de diferencias en diferencias se considera el grupo de control y el grupo de tratamiento. Este es un potente método que nos permite examinar el efecto de una intervención teniendo en cuenta cómo cambia la media de un grupo antes y después de dicha intervención (grupo de tratamiento) y comparar este cambio con la media a lo largo del tiempo de un grupo similar que no se sometió al tratamiento (grupo de control).

La estimación viene dada por la ecuación: **Yit = 0 + 1*Treat + 2(Post) + 3(Treat)(Post) + eit**

[99]:
```
df1 = charls.loc[charls["wave"] >1 ]
```

[100]:
```
df1.head()
```

[100]:

	age	bnrps	cesd	child	tres_o_mas	dnrps	sdrinlky	drinkly	female	\
1	48	58.964134	7.0	2	0	0	0	0	1	
2	50	60.000130	5.0	2	0	0	1	0	1	
4	50	58.964134	5.0	2	0	0	1	1	0	
5	52	60.000130	6.0	2	0	0	1	1	0	
7	60	60.000130	6.0	2	0	0	0	0	1	

	hrsusu	…	intmonth	married	nrps	retage	retired	schadj	urban	\
1	3.891820	…	7	1	1	17	0	0	0	
2	4.025352	…	8	1	1	10	0	0	0	
4	3.891820	…	7	1	1	0	0	4	0	
5	4.025352	…	8	1	1	0	0	4	0	

```
7  3.178054  …         8        1     1       0        0       0      0

   wave   wealth  inid
1     2    100.0     1
2     3 -59970.0     1
4     2    100.0     2
5     3 -59970.0     2
7     3   1400.0     3

[5 rows x 21 columns]
```

[101]:
```
cantidad2 = df1.age.count() ##cantidad de elementos en el dataframe
```

[102]:
```
list0=[]
for i in range(cantidad2):
  if charls.wave[i]==2:
    list0.append(1)
  else: list0.append(0)

df1.insert(18, "post", list0)
list1=[]
for i in range (cantidad2):
  if (charls.child[i]>2):
    list1.append(1)
  else:
    list1.append(0)
df1.insert(4, "Treat", list1,False)
```

[103]:
```
df1['did'] = df1['post'] * df1['Treat']
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1:
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
  """Entry point for launching an IPython kernel.
```

1.13 Pregunta 13

Compare el efecto del tratamiento generando grupos pseudo-equivalentes, en particular entre individuos solo con 3 hijos (tratamiento) y 2 hijos (control).

[ ]:
```

```

1.14 Pregunta 14

Estime el efecto anterior usando la variable married como instrumento para determinar el efecto del tratamiento en la pregunta 12. Como se interpreta el efecto en este caso?

[ ]:

1.15 Pregunta 15

Finalmente, asuma que la intervencion se implementa en todos los individuos. Genere una nueva variable de tratamiento un nueva variable llamada tdrinkly donde el efecto es una reduccion de 50% en la prevalencia de consumo de alcohol en toda la poblacion en el tercer periodo (identica a drinkly en los periodos 1 y 2). Genere una variable cdrinkly que es identica a drinkly en los periodos 1 y 2 y use la informacion de ambos periodos para predicir el valor esperado de drinkly en el tercer periodo, estos seran los valores de cdrinkly en el periodo 3 (contrafactual). Finalmente, estime el efecto de la intervencion en toda la poblacion comparando entre tdrinkly (datos reales) versus cdrinkly contrafactual.

En primer lugar se lee nuevamente el archivo charls, esto para no tener cruce de informacion de los ejercicios anteriores

[ ]:
```
charls = pd.read_csv('charls.csv')
charls.dropna(inplace=True) ##remueve faltantes
charls.reset_index(drop=True, inplace=True) ##resetea indices
cantidad = charls.age.count() ##cantidad de elementos en el dataframe
print(cantidad)
```

[ ]:
```
list3=[]
for i in range (cantidad):
  if (charls.wave[i]==1 or charls.wave[i]==2):
    if (charls.drinkly[i]=='0'):
      list3.append(0)
    else: list3.append(1)
  else:
    a=np.random.randint(2)
    list3.append(a)

charls.insert(6, "tdrinlky", list3,False)
```

[ ]:
```
%pip install scikit-learn
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
import statsmodels.formula.api as smf
import sklearn
import scipy
from sklearn import metrics
from sklearn.tree import DecisionTreeRegressor
```

```
x=[0.1,0.2,0.3,0.4]
y=[1,2,3,4]
model = sm.OLS(y,x)
results = model.fit(cov_type="HC1")
y_predic=results.predict(x)
plt.plot(x,y_predic)
```

[ ]:
```
charls.head(6)
```

[ ]:
```
list1=[]
for i in range (cantidad):
  if (charls.child[i]>2):
    list1.append(1)
  else:
    list1.append(0)
charls.insert(4, "tres_o_mas", list1,False)
```

[ ]:
```
Xf=charls[['age','bnrps','cesd','child','dnrps','female','hrsusu','intmonth','married','nrps',
Xf = sm.add_constant(Xf)
yf=charls['drinkly']
model = sm.OLS(yf, Xf)
first = model.fit(cov_type="HC1")
charls['cdrinkly']=first.predict(Xf)
print(first.summary())
```