Cap14Python

May 14, 2018

0.1 Ejercicio 7

i) Se esperaría que el número de ejecuciones tenga un efecto negativo en la tasa de asesinatos y que la tasa de desempleo tenga un efecto positivo por que un aumento en la tasa de desempleo puede generar más violencia en general.

In [24]:
```
from rpy2.robjects import r, pandas2ri
from linearmodels import PooledOLS, FirstDifferenceOLS
pandas2ri.activate()
r.load('data/murder.RData')
murder = r['data'][(r.data.year > 87)].set_index(["id", "year"])
#7.2
model14 = PooledOLS.from_formula("mrdrte ~ 1 + d93 + exec + unem", murder)
model14.fit().summary
```

Out[24]:
```
<class 'linearmodels.compat.statsmodels.Summary'>
"""
                          PooledOLS Estimation Summary
================================================================================
Dep. Variable:                  mrdrte   R-squared:                        0.1016
Estimator:                   PooledOLS   R-squared (Between):              0.1167
No. Observations:                  102   R-squared (Within):              -5.0117
Date:                Mon, May 14 2018   R-squared (Overall):              0.1016
Time:                        05:17:32   Log-likelihood                   -379.81
Cov. Estimator:            Unadjusted
                                        F-statistic:                      3.6947
Entities:                          51   P-value                           0.0144
Avg Obs:                       2.0000   Distribution:                    F(3,98)
Min Obs:                       2.0000
Max Obs:                       2.0000   F-statistic (robust):             3.6947
                                        P-value                           0.0144
Time periods:                       2   Distribution:                    F(3,98)
Avg Obs:                       51.000
Min Obs:                       51.000
Max Obs:                       51.000

                             Parameter Estimates
================================================================================
```

```
             Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper CI
------------------------------------------------------------------------------
Intercept     -5.2780     4.4278    -1.1920     0.2361     -14.065      3.5088
d93           -2.0674     2.1446    -0.9640     0.3374     -6.3234      2.1885
exec           0.1277     0.2632     0.4852     0.6286     -0.3947      0.6501
unem           2.5289     0.7817     3.2350     0.0017      0.9776      4.0802
==============================================================================
"""
```

ii) El efecto de *exec* es positivo pero estadísticamente insignificante.

In [2]:
```
#7.3
model14_3 = FirstDifferenceOLS.from_formula("mrdrte ~ (d93 - d90)\
            + exec + unem ", murder)
model14_3.fit().summary
```

Out[2]:
```
<class 'linearmodels.compat.statsmodels.Summary'>
"""
                     FirstDifferenceOLS Estimation Summary
==============================================================================
Dep. Variable:                  mrdrte   R-squared:                      0.1653
Estimator:         FirstDifferenceOLS   R-squared (Between):            -0.0370
No. Observations:                  51   R-squared (Within):             0.1653
Date:                Mon, May 14 2018   R-squared (Overall):            -0.0366
Time:                        04:53:12   Log-likelihood                 -74.693
Cov. Estimator:            Unadjusted
                                        F-statistic:                    3.1694
Entities:                          51   P-value                         0.0326
Avg Obs:                       2.0000   Distribution:                  F(3,48)
Min Obs:                       2.0000
Max Obs:                       2.0000   F-statistic (robust):           3.1694
                                        P-value                         0.0326
Time periods:                       2   Distribution:                  F(3,48)
Avg Obs:                       51.000
Min Obs:                       51.000
Max Obs:                       51.000

                             Parameter Estimates
==============================================================================
             Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper CI
------------------------------------------------------------------------------
d93             0.4133     0.2094     1.9737     0.0542     -0.0077      0.8343
exec           -0.1038     0.0434    -2.3918     0.0207     -0.1911     -0.0165
unem           -0.0666     0.1587    -0.4196     0.6766     -0.3857      0.2525
==============================================================================
"""
```

iii) el número de ejecuciones ahora es significativo, y cada que se incrementan las ejecuciones, la tasa de homicidios se reduce en 0.104.

```
In [3]: model14.fit(cov_type='robust').summary

Out[3]: <class 'linearmodels.compat.statsmodels.Summary'>
        """
                                PooledOLS Estimation Summary
        ================================================================================
        Dep. Variable:                  mrdrte   R-squared:                        0.1016
        Estimator:                   PooledOLS   R-squared (Between):              0.1167
        No. Observations:                  102   R-squared (Within):              -5.0117
        Date:                Mon, May 14 2018   R-squared (Overall):              0.1016
        Time:                        04:53:12   Log-likelihood                   -379.81
        Cov. Estimator:                Robust
                                                F-statistic:                      3.6947
        Entities:                          51   P-value                           0.0144
        Avg Obs:                       2.0000   Distribution:                    F(3,98)
        Min Obs:                       2.0000
        Max Obs:                       2.0000   F-statistic (robust):             10.744
                                                P-value                           0.0000
        Time periods:                       2   Distribution:                    F(3,98)
        Avg Obs:                       51.000
        Min Obs:                       51.000
        Max Obs:                       51.000

                                    Parameter Estimates
        ==============================================================================
                    Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper CI
        ------------------------------------------------------------------------------
        Intercept     -5.2780     5.3868    -0.9798     0.3296     -15.968      5.4119
        d93           -2.0674     1.9981    -1.0347     0.3034     -6.0326      1.8978
        exec           0.1277     0.1342     0.9515     0.3437     -0.1387      0.3941
        unem           2.5289     1.1076     2.2833     0.0246      0.3310      4.7268
        ==============================================================================
        """
```

iv) Los errores estándar robustos a heterocedasticidad para son 2 para d93, 0.13 para exec y 1.11 para unem. usando estos errores se pierde la significancia de los estimadores.

```
In [25]: # 7.5
         murder[(murder.exec == murder.exec.max())]
         murder.sort_values(by = ['exec'], ascending = False).head(2)

Out[25]:         state  mrdrte  exec  unem  d90  d93   cmrdrte  cexec  cunem  cexec_1  \
         id year
         44 93      TX    11.9    34   7.0    0    1 -2.200001     23    0.8      -11
         47 93      VA     8.3    11   5.0    0    1 -0.500000      8    0.7       -1

                  cunem_1
         id year
```

```
44 93       -2.2
47 93        0.1
```

v) El estado que más veces aplicó la pena de muerte fue Texas con un total de 34 y le sigue Virginia con 11.

```
In [27]: # 7.6
         murder_wo_tx = murder[(murder.state != 'TX')]
         formula = 'mrdrte ~ (d93 - d90) + exec + unem + \
                    EntityEffects + TimeEffects'
         model14_6 = FirstDifferenceOLS.from_formula(formula, murder_wo_tx)
         model14_6.fit().summary.tables[1]

Out[27]: <class 'statsmodels.iolib.table.SimpleTable'>

In [5]: #Errores Robustos a Heterocedasticidad
        model14_6.fit(cov_type='robust').summary.tables[1]

Out[5]: <class 'statsmodels.iolib.table.SimpleTable'>
```

vi) sin el estado de Texas, se obtiene que la tasa de ejecuciones no es significativa cuando se usan estimaciones robustas. Al tener menos datos puede que no se hagan bien los cálculos y exista más varianza.

```
In [28]: import pandas as pd
         from linearmodels import BetweenOLS, PanelOLS
         murder = r['data'].set_index(["id", "year"])
         murder['year'] = pd.Categorical(r['data'].year)
         formula = 'mrdrte ~ exec + unem + TimeEffects + EntityEffects'
         model14_7 = PanelOLS.from_formula(formula, murder)

         model14_7.fit().summary

Out[28]: <class 'linearmodels.compat.statsmodels.Summary'>
         """
                                   PanelOLS Estimation Summary
         ================================================================================
         Dep. Variable:                  mrdrte   R-squared:                        0.0109
         Estimator:                    PanelOLS   R-squared (Between):              0.1249
         No. Observations:                 153   R-squared (Within):               0.0026
         Date:                Mon, May 14 2018   R-squared (Overall):              0.1178
         Time:                        05:27:40   Log-likelihood                   -375.63
         Cov. Estimator:            Unadjusted
                                                 F-statistic:                      0.5391
         Entities:                          51   P-value                           0.5850
         Avg Obs:                       3.0000   Distribution:                   F(2,98)
         Min Obs:                       3.0000
         Max Obs:                       3.0000   F-statistic (robust):             0.5391
                                                 P-value                           0.5850
```

```
Time periods:                        3   Distribution:                  F(2,98)
Avg Obs:                        51.000
Min Obs:                        51.000
Max Obs:                        51.000

                             Parameter Estimates
==============================================================================
            Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper CI
------------------------------------------------------------------------------
exec          -0.1383     0.1770    -0.7815     0.4364     -0.4896      0.2129
unem           0.2213     0.2964     0.7467     0.4570     -0.3668      0.8095
==============================================================================

F-test for Poolability: 16.796
P-value: 0.0000
Distribution: F(52,98)

Included effects: Entity, Time
"""
```

vii) Los resultados siguen sin tener significancia estadística, ahora el valor de exex es de -0.14 con un T de -0.78 por lo que no se puede aceptar que el número de ejecuciones tenga algún efecto en la tasa de homicidios

0.2 Ejercicio 8

```
In [36]: import pandas as pd
         import numpy as np
         from statsmodels.api import OLS
         from rpy2.robjects import r, pandas2ri
         from linearmodels import PooledOLS, PanelOLS
         from math import log
         pandas2ri.activate()
         r.load('data/mathpnl.RData')
         params =['year','lrexpp', 'lrexpp_1', 'lenrol', 'lunch', 'math4']
         math = r['data'].set_index(["distid", "year"])
         math['year'] = pd.Categorical(r['data'].year)
         math = math.dropna(0, how = "any", subset=params)
         params.remove('math4')

         model8_1 = PooledOLS(math.math4, math[params])
         model8_1.fit().summary

Out[36]: <class 'linearmodels.compat.statsmodels.Summary'>
         """
                                  PooledOLS Estimation Summary
         ==============================================================================
         Dep. Variable:                  math4   R-squared:                      0.5053
```

```
Estimator:                  PooledOLS   R-squared (Between):              0.3884
No. Observations:                3300   R-squared (Within):               0.5903
Date:                Mon, May 14 2018   R-squared (Overall):              0.5053
Time:                        05:52:23   Log-likelihood                -1.289e+04
Cov. Estimator:            Unadjusted
                                        F-statistic:                     373.34
Entities:                         550   P-value                          0.0000
Avg Obs:                       6.0000   Distribution:                 F(9,3290)
Min Obs:                       6.0000
Max Obs:                       6.0000   F-statistic (robust):            7993.7
                                        P-value                          0.0000
Time periods:                     7     Distribution:                 F(9,3290)
Avg Obs:                     471.43
Min Obs:                     0.0000
Max Obs:                     550.00

                            Parameter Estimates
==============================================================================
            Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper CI
------------------------------------------------------------------------------
year.1993     -31.662     10.301    -3.0736     0.0021     -51.859     -11.464
year.1994     -25.284     10.358    -2.4409     0.0147     -45.594     -4.9746
year.1995     -13.011     10.486    -1.2408     0.2148     -33.571      7.5483
year.1996     -13.628     10.545    -1.2924     0.1963     -34.304      7.0475
year.1997     -16.321     10.573    -1.5437     0.1228     -37.052      4.4086
year.1998     -1.2637     10.590    -0.1193     0.9050     -22.027      19.500
lrexpp         0.5339     2.4281     0.2199     0.8260     -4.2268      5.2947
lrexpp_1       9.0492     2.3053     3.9253     0.0001      4.5292      13.569
lenrol         0.5927     0.2050     2.8905     0.0039      0.1906      0.9947
lunch         -0.4067     0.0138    -29.396     0.0000     -0.4338     -0.3796
==============================================================================
"""
```

i) Estimando por Sección Cruzada, Cuando el gasto real por alumno incrementa en 100%, la tasa de aprobados aumenta en $100 * [exp(0.5339) - 1] = 70.56\%$ pero no es significativo.

ii) La variable lunch es el porcentaje de elegibles para almuerzo gratis, lo que se analiza aquí es que para un aumento del 1% en lunch, el número de aprobados en matemáticas se reduce en 0.41%, Se puede explicar por que a mayor porcentaje de inscritos en el programa para almuerzos gratis más pobre es el estado en observación.

In [37]:
```
def lag(array, periods = 1):
    arr = np.empty_like(array)
    arr[:periods] = [np.NaN]
    arr[periods:]=array[:-periods]
    return arr
```

```
resids = model8_1.fit()._resids
years = ['y94', 'y95', 'y96', 'y97', 'y98']
params = math[years]
params['lresid'] = lag(resids)
params['resid'] = resids
formula = 'resid ~ lresid + 1 + y94 + y95 + y96 + y97 + y98'
ARtest = PooledOLS.from_formula(formula, params ).fit()
ARtest.summary
```

```
/home/hudson94/.local/lib/python3.5/site-packages/ipykernel_launcher.py:11: SettingWithCopyWarni
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: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
  # This is added back by InteractiveShellApp.init_path()
/home/hudson94/.local/lib/python3.5/site-packages/ipykernel_launcher.py:12: SettingWithCopyWarni
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: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
  if sys.path[0] == '':
/home/hudson94/.local/lib/python3.5/site-packages/linearmodels/utility.py:492: MissingValueWarni
Inputs contain missing values. Dropping rows with missing observations.
  warnings.warn(missing_value_warning_msg, MissingValueWarning)
```

Out[37]:
```
<class 'linearmodels.compat.statsmodels.Summary'>
"""
                            PooledOLS Estimation Summary
================================================================================
Dep. Variable:                   resid   R-squared:                        0.1931
Estimator:                   PooledOLS   R-squared (Between):              0.6108
No. Observations:                 3299   R-squared (Within):              -0.2603
Date:                Mon, May 14 2018   R-squared (Overall):              0.1931
Time:                        05:52:37   Log-likelihood                -1.253e+04
Cov. Estimator:            Unadjusted
                                        F-statistic:                      131.34
Entities:                         550   P-value                           0.0000
Avg Obs:                       5.9982   Distribution:                  F(6,3292)
Min Obs:                       5.0000
Max Obs:                       6.0000   F-statistic (robust):             131.34
                                        P-value                           0.0000
Time periods:                       6   Distribution:                  F(6,3292)
Avg Obs:                       549.83
Min Obs:                       549.00
Max Obs:                       550.00
```

```
                              Parameter Estimates
==============================================================================
            Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper CI
------------------------------------------------------------------------------
Intercept      0.0188     0.4615     0.0408     0.9674     -0.8860      0.9237
lresid         0.4395     0.0157     28.072     0.0000      0.4088      0.4702
y94           -0.0188     0.6524    -0.0289     0.9770     -1.2979      1.2602
y95           -0.0188     0.6524    -0.0289     0.9770     -1.2979      1.2602
y96           -0.0188     0.6524    -0.0289     0.9770     -1.2979      1.2602
y97           -0.0188     0.6524    -0.0289     0.9770     -1.2979      1.2602
y98           -0.0188     0.6524    -0.0289     0.9770     -1.2979      1.2602
==============================================================================
"""
```

In [38]:
```
formula = 'math4 ~ lrexpp + lrexpp_1 + lenrol + lunch + \
           TimeEffects + EntityEffects'
model8_4 = PanelOLS.from_formula(formula, math)
model8_4.fit().summary
```

Out[38]:
```
<class 'linearmodels.compat.statsmodels.Summary'>
"""
                           PanelOLS Estimation Summary
================================================================================
Dep. Variable:                    math4   R-squared:                        0.0042
Estimator:                   PanelOLS   R-squared (Between):              0.9622
No. Observations:                3300   R-squared (Within):               0.0694
Date:                Mon, May 14 2018   R-squared (Overall):              0.9215
Time:                        05:52:46   Log-likelihood                -1.163e+04
Cov. Estimator:            Unadjusted
                                        F-statistic:                      2.9056
Entities:                         550   P-value                           0.0206
Avg Obs:                       6.0000   Distribution:                  F(4,2741)
Min Obs:                       6.0000
Max Obs:                       6.0000   F-statistic (robust):             2.9056
                                        P-value                           0.0206
Time periods:                       7   Distribution:                  F(4,2741)
Avg Obs:                       471.43
Min Obs:                       0.0000
Max Obs:                       550.00

                              Parameter Estimates
==============================================================================
            Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper CI
------------------------------------------------------------------------------
lrexpp        -0.4112     2.4577    -0.1673     0.8671     -5.2302      4.4079
lrexpp_1       7.0030     2.3692     2.9559     0.0031      2.3574      11.649
lenrol         0.2451     1.1004     0.2227     0.8238     -1.9126      2.4027
```

```
lunch          0.0615     0.0515     1.1955     0.2320     -0.0394      0.1624
================================================================================

F-test for Poolability: 11.346
P-value: 0.0000
Distribution: F(554,2741)

Included effects: Entity, Time
"""
```

iv) El gasto rezagado sigue siendo altamente significativo.

v) Se pierde total significación en *lenrol* y *lunch* por que estas variables no presentan cambios notables en los periodos de la muestra, y su efecto se va al error idiosincrático.

```
In [39]: formula = 'math4 ~ lrexpp + I(lrexpp_1 - lrexpp) + \
                    TimeEffects + EntityEffects '
         model8_5 = PanelOLS.from_formula(formula, math)
         model8_5.fit().summary
```

```
Out[39]: <class 'linearmodels.compat.statsmodels.Summary'>
"""
                           PanelOLS Estimation Summary
================================================================================
Dep. Variable:                   math4   R-squared:                        0.0037
Estimator:                    PanelOLS   R-squared (Between):              0.9640
No. Observations:                3300   R-squared (Within):               0.0629
Date:                Mon, May 14 2018   R-squared (Overall):              0.9229
Time:                        05:52:54   Log-likelihood                -1.163e+04
Cov. Estimator:            Unadjusted
                                        F-statistic:                      5.0817
Entities:                         550   P-value                           0.0063
Avg Obs:                       6.0000   Distribution:                  F(2,2743)
Min Obs:                       6.0000
Max Obs:                       6.0000   F-statistic (robust):             5.0817
                                        P-value                           0.0063
Time periods:                     7     Distribution:                  F(2,2743)
Avg Obs:                       471.43
Min Obs:                       0.0000
Max Obs:                       550.00

                                Parameter Estimates
================================================================================
                      Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper C
--------------------------------------------------------------------------------
lrexpp                   6.4927     2.6362     2.4629     0.0138      1.3236      11.66
I(lrexpp_1 - lrexpp)     6.5983     2.1455     3.0753     0.0021      2.3912      10.80
================================================================================
```

```
F-test for Poolability: 13.974
P-value: 0.0000
Distribution: F(554,2743)

Included effects: Entity, Time
"""
```

v) El efecto de largo plazo $\hat{\theta} = 6.5983$ y su $sd(2.15)$

0.3 Ejercicio 9

```
In [14]: from rpy2.robjects import r, pandas2ri
         from statsmodels.api import OLS
         pandas2ri.activate()
         r.load('data/pension.RData')
         pension = r['data']
         formula = """pctstck ~ 1 + choice + prftshr + female + age +
                      educ + finc25 + finc35 + finc50 + finc75 + finc100
                      + finc101 + wealth89 + stckin89 + irain89"""
         model9_1 = OLS.from_formula(formula, pension)
         model9_1.fit().summary()
```

```
Out[14]: <class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 pctstck   R-squared:                       0.108
Model:                             OLS   Adj. R-squared:                  0.038
Method:                  Least Squares   F-statistic:                     1.542
Date:                 Mon, 14 May 2018   Prob (F-statistic):              0.100
Time:                         04:54:28   Log-Likelihood:                -978.02
No. Observations:                 194   AIC:                             1986.
Df Residuals:                     179   BIC:                             2035.
Df Model:                          14
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    128.5442     55.170      2.330      0.021      19.677     237.411
choice        11.7443      6.232      1.884      0.061      -0.553      24.042
prftshr       14.3361      7.231      1.982      0.049       0.066      28.606
female         1.4522      6.766      0.215      0.830     -11.898      14.803
age           -1.5006      0.777     -1.932      0.055      -3.033       0.032
educ           0.7036      1.197      0.588      0.557      -1.658       3.065
finc25       -15.2887     14.229     -1.074      0.284     -43.368      12.790
finc35         0.1880     14.693      0.013      0.990     -28.806      29.182
finc50        -3.8617     14.551     -0.265      0.791     -32.576      24.852
```

```
finc75       -13.7481     16.022     -0.858      0.392     -45.364      17.868
finc100       -2.6861     15.719     -0.171      0.865     -33.704      28.331
finc101      -25.0504     17.800     -1.407      0.161     -60.176      10.075
wealth89      -0.0026      0.013     -0.200      0.841      -0.028       0.023
stckin89       6.6742      6.683      0.999      0.319      -6.513      19.862
irain89       -7.4978      6.378     -1.176      0.241     -20.083       5.088
==============================================================================
Omnibus:                       52.568   Durbin-Watson:                   1.856
Prob(Omnibus):                  0.000   Jarque-Bera (JB):                9.876
Skew:                           0.055   Prob(JB):                      0.00717
Kurtosis:                       1.900   Cond. No.                     6.74e+03
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specif
[2] The condition number is large, 6.74e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

i) El stock de pensiones aumenta en 11.74 cuando se puede elegir como hacer la inversión, y es significativa al 96%.

```
In [15]: hypotheses = """(finc25 = finc35 = finc50 = finc75 = finc100 =
                          finc101 = wealth89 = stckin89 = irain89 = 0)"""
         print(model9_1.fit().f_test(hypotheses))
```

```
<F test: F=array([[1.03087375]]), p=0.4171606761558133, df_denom=179, df_num=9>
```

ii) La prueba F de significancia conjunta tiene un p-valor de 0.42 por lo tanto no son conjuntamente significativas estas variables.

```
In [16]: len(pension.id.unique())
```

```
Out[16]: 171
```

iii) Hay 171 familias en la muestra

```
In [17]: model9_1.fit().get_robustcov_results(cov_type='cluster', \
                          groups = pension.id).summary().tables[1]
```

```
Out[17]: <class 'statsmodels.iolib.table.SimpleTable'>
```

iv) no son tan diferentes, en la tabla anterior se presentan los errores robustos. no sorprende este resultado porque con 171 familias en una muestra de 194 se espera que sean independientes.

```
In [18]: from linearmodels import FirstDifferenceOLS
         couples = pd.concat(id for _, id in pension.groupby('id') if len(id)>1)
         i = (1,2)*(int(len(couples)/2))
```

```
couples['i'] = i
couples = couples.set_index(['id', 'i'])
couples['i'] = pd.Categorical(i)
formula = """pctstck ~ i + choice + prftshr + female  + age + educ
                + EntityEffects"""
model9_5 = FirstDifferenceOLS.from_formula(formula, couples)
model9_5.fit().summary
```

Out[18]:

```
<class 'linearmodels.compat.statsmodels.Summary'>
"""
                    FirstDifferenceOLS Estimation Summary
================================================================================
Dep. Variable:                 pctstck   R-squared:                        0.2060
Estimator:         FirstDifferenceOLS   R-squared (Between):             -16.307
No. Observations:                  23   R-squared (Within):               0.2060
Date:                Mon, May 14 2018   R-squared (Overall):             -15.160
Time:                        04:54:28   Log-likelihood                   -110.38
Cov. Estimator:            Unadjusted
                                        F-statistic:                      0.7351
Entities:                          23   P-value                           0.6284
Avg Obs:                       2.0000   Distribution:                    F(6,17)
Min Obs:                       2.0000
Max Obs:                       2.0000   F-statistic (robust):             0.7351
                                        P-value                           0.6284
Time periods:                       2   Distribution:                    F(6,17)
Avg Obs:                       23.000
Min Obs:                       23.000
Max Obs:                       23.000

                             Parameter Estimates
================================================================================
            Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper CI
--------------------------------------------------------------------------------
i              15.927     10.938     1.4562     0.1636     -7.1495      39.003
choice         2.2757     15.000     0.1517     0.8812     -29.372      33.923
prftshr       -9.2671     16.924    -0.5476     0.5911     -44.973      26.439
female         21.551     21.485     1.0031     0.3299     -23.779      66.881
age           -3.5727     8.9992    -0.3970     0.6963     -22.559      15.414
educ          -1.2203     3.4290    -0.3559     0.7263     -8.4550      6.0143
================================================================================
"""
```

v) En la primera diferencia se borran todas las variables que eran iguales para el mismo hogar.

vi) En este modelo ninguna variable es significativa. puede deberse a que solo se tienen 23 observaciones.

0.4 Ejercicio 11

```
In [19]: from rpy2.robjects import r, pandas2ri
         from linearmodels import PooledOLS, PanelOLS
         pandas2ri.activate()
         r.load('data/wagepan.RData')
         wagepan = r.data.set_index(['year','nr'])
         wagepan['year'] = pd.Categorical(r.data['year'])

         formula = "lwage ~ 1+educ + black + hisp \
                     + exper + expersq + married + union \
                     + EntityEffects + TimeEffects"

         model_i = PooledOLS.from_formula(formula, wagepan)
         model_i.fit(cov_type='kernel').summary.tables[1]

Out[19]: <class 'statsmodels.iolib.table.SimpleTable'>
```

i) Como es de esperarse, los estimadores robustos a heterocedasticidad y autocorrelación son más grandes, para el caso de educ el EE es 0.009, casi el doble del estimado por OLS clásico. para hisp el EE es 0.04 y antes era 0.024, y married ahora tiene EE 0.026 a diferencia de 0.016.

```
In [20]: model_ii = PanelOLS.\
                 from_formula("lwage ~  expersq + married + \
                                union + EntityEffects + TimeEffects",
                              wagepan)
         model_ii.fit(cov_type='kernel').summary.tables[1]

Out[20]: <class 'statsmodels.iolib.table.SimpleTable'>
```

ii) Estos Errores estándar son mas grandes que los normales pero la diferencia es muy pequeña, en el caso de $Exper^2$ el error es solo 0.0002 mas grande. esto puede deberse a que se han eliminado las variables constantes en el tiempo, al igual que parte del error *ai*

iii) Es más importante hacer robustos los errores a autocorrelación en el modelo de sección agregada. puesto que es más probable que el error se correlacione serialmente.

0.5 Ejercicio 12

```
In [21]: import numpy as np
         from rpy2.robjects import r, pandas2ri
         pandas2ri.activate()
         r.load('data/elem94_95.RData')
         elem = r["data"]
         i = elem.groupby(["distid"])["schid"]
         print("Menor: ", min(i.count()), "\nMayor: ", max(i.count()),\
```

```
        "\nMedia: ", round(sum(i.count())/len(i), 2))

    del i
```

```
Menor:  1
Mayor:  162
Media:  3.44
```

```
In [14]: from statsmodels.api import OLS
         formula = "lavgsal ~ bs + lenrol + lstaff + lunch"
         modii = OLS.from_formula(formula, elem)
         modii.fit().summary()
```

```
Out[14]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                  OLS Regression Results
         ==============================================================================
         Dep. Variable:                lavgsal   R-squared:                       0.488
         Model:                            OLS   Adj. R-squared:                  0.487
         Method:                 Least Squares   F-statistic:                     439.4
         Date:                Mon, 14 May 2018   Prob (F-statistic):          4.22e-266
         Time:                        04:59:15   Log-Likelihood:                 689.98
         No. Observations:                1848   AIC:                            -1370.
         Df Residuals:                    1843   BIC:                            -1342.
         Df Model:                           4
         Covariance Type:            nonrobust
         ==============================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
         ------------------------------------------------------------------------------
         Intercept     13.8315      0.110    126.055      0.000      13.616      14.047
         bs            -0.5161      0.110     -4.702      0.000      -0.731      -0.301
         lenrol        -0.0284      0.008     -3.360      0.001      -0.045      -0.012
         lstaff        -0.6906      0.018    -37.615      0.000      -0.727      -0.655
         lunch         -0.0008      0.000     -4.695      0.000      -0.001      -0.000
         ==============================================================================
         Omnibus:                       46.324   Durbin-Watson:                   0.921
         Prob(Omnibus):                  0.000   Jarque-Bera (JB):              105.674
         Skew:                           0.009   Prob(JB):                     1.13e-23
         Kurtosis:                       4.171   Cond. No.                     1.46e+03
         ==============================================================================

         Warnings:
         [1] Standard Errors assume that the covariance matrix of the errors is correctly specif
         [2] The condition number is large, 1.46e+03. This might indicate that there are
         strong multicollinearity or other numerical problems.
         """
```

ii) El estimador para *bs* es -0.52 con un error estándar de 0.11

```
In [7]: modii.fit().get_robustcov_results(cov_type='cluster', \
                        groups = r.data.distid).summary().tables[1]

Out[7]: <class 'statsmodels.iolib.table.SimpleTable'>
```

iii) El error estándar robusto a cluster por distrito es 0.25 y el valor de t es -2.04 el cual se redujo bastante y pierde significatividad pero aun es aceptable.

```
In [8]: modii = OLS.from_formula(formula, elem[(elem.bs < 0.5)])
        modii.fit().get_robustcov_results(cov_type='cluster', \
                        groups = elem.distid[(elem.bs < 0.5)]).summary().tables[1]

Out[8]: <class 'statsmodels.iolib.table.SimpleTable'>
```

iV) Al omitir las observaciones donde bs es > a .5, el estimador ahora pierde significancia y toma el valor de -0.19 con un error de 0.273 por lo que el efecto que tiene es estadísticamente cero.

```
In [22]: from linearmodels import PanelOLS
         elem = elem[(elem.bs < 0.5)].set_index(["distid","schid"])
         modeliv = PanelOLS.from_formula("lavgsal ~ bs + lenrol + lstaff + lunch + \
                                         EntityEffects + TimeEffects", elem)
         modeliv.fit(cov_type = "clustered").summary.tables[1]


         ---------------------------------------------------------------------------

         MemoryError                               Traceback (most recent call last)

         <ipython-input-22-b9db816f8182> in <module>()
           2 elem = elem[(elem.bs < 0.5)].set_index(["distid","schid"])
           3 modeliv = PanelOLS.from_formula("lavgsal ~ bs + lenrol + lstaff + lunch +
       ----> 4 modeliv.fit(cov_type = "clustered").summary.tables[1]


         ~/.local/lib/python3.5/site-packages/linearmodels/panel/model.py in fit(self, use_lsdv,
        1149             y, x, ybar, y_effects, x_effects = self._slow_path()
        1150         elif not weighted:
       -> 1151             y, x, ybar = self._fast_path()
        1152         else:
        1153             y, x, ybar, y_effects, x_effects = self._weighted_fast_path()


         ~/.local/lib/python3.5/site-packages/linearmodels/panel/model.py in _fast_path(self)
         946             x = x.general_demean(groups)
         947         elif self.entity_effects and self.time_effects:
       --> 948             y = y.demean('both')
         949             x = x.demean('both')
         950         elif self.entity_effects:
```

```
    ~/.local/lib/python3.5/site-packages/linearmodels/panel/data.py in demean(self, group, w
    456             raise ValueError
    457         if group == 'both':
--> 458             return self._demean_both(weights)
    459
    460         level = 0 if group == 'entity' else 1


    ~/.local/lib/python3.5/site-packages/linearmodels/panel/data.py in _demean_both(self, we
    340         d = self.dummies(dummy, drop_first=True)
    341         d.index = e.index
--> 342         d = PanelData(d).demean(group, weights=weights)
    343         d = d.values2d
    344         e = e.values2d


    ~/.local/lib/python3.5/site-packages/linearmodels/panel/data.py in __init__(self, x, var
    205             raise ValueError('The index on the time dimension must be either '
    206                              'numeric or date-like')
--> 207         self._k, self._t, self._n = self.panel.shape
    208         self._frame.index.levels[0].name = 'entity'
    209         self._frame.index.levels[1].name = 'time'


    ~/.local/lib/python3.5/site-packages/linearmodels/panel/data.py in panel(self)
    213         """pandas Panel view of data"""
    214         if self._panel is None:
--> 215             self._panel = _Panel(self._frame)
    216         return self._panel
    217


    ~/.local/lib/python3.5/site-packages/linearmodels/panel/data.py in __init__(self, df)
     38         self._full_index = pd.MultiIndex.from_product([self._minor_axis,
     39                                                        self._major_axis])
---> 40         new_df = df.reindex(self._full_index)
     41         self._frame = new_df
     42         i, j, k = len(self._items), len(self._major_axis), len(self.minor_axis)


    ~/.local/lib/python3.5/site-packages/pandas/util/_decorators.py in wrapper(*args, **kwar
    125         @wraps(func)
    126         def wrapper(*args, **kwargs):
--> 127             return func(*args, **kwargs)
    128
    129         if not PY2:
```

```
   ~/.local/lib/python3.5/site-packages/pandas/core/frame.py in reindex(self, *args, **kwar
   2933         kwargs.pop('axis', None)
   2934         kwargs.pop('labels', None)
-> 2935         return super(DataFrame, self).reindex(**kwargs)
   2936
   2937     @Appender(_shared_docs['reindex_axis'] % _shared_doc_kwargs)


   ~/.local/lib/python3.5/site-packages/pandas/core/generic.py in reindex(self, *args, **kw
   3021         # perform the reindex on the axes
   3022         return self._reindex_axes(axes, level, limit, tolerance, method,
-> 3023                                   fill_value, copy).__finalize__(self)
   3024
   3025     def _reindex_axes(self, axes, level, limit, tolerance, method, fill_value,


   ~/.local/lib/python3.5/site-packages/pandas/core/frame.py in _reindex_axes(self, axes, l
   2868         if index is not None:
   2869             frame = frame._reindex_index(index, method, copy, level,
-> 2870                                          fill_value, limit, tolerance)
   2871
   2872         return frame


   ~/.local/lib/python3.5/site-packages/pandas/core/frame.py in _reindex_index(self, new_in
   2879         return self._reindex_with_indexers({0: [new_index, indexer]},
   2880                                            copy=copy, fill_value=fill_value,
-> 2881                                            allow_dups=False)
   2882
   2883     def _reindex_columns(self, new_columns, method, copy, level,


   ~/.local/lib/python3.5/site-packages/pandas/core/generic.py in _reindex_with_indexers(se
   3143                                                 fill_value=fill_value,
   3144                                                 allow_dups=allow_dups,
-> 3145                                                 copy=copy)
   3146
   3147         if copy and new_data is self._data:


   ~/.local/lib/python3.5/site-packages/pandas/core/internals.py in reindex_indexer(self, n
   4148             new_blocks = [blk.take_nd(indexer, axis=axis, fill_tuple=(
   4149                 fill_value if fill_value is not None else blk.fill_value,))
-> 4150                 for blk in self.blocks]
   4151
   4152         new_axes = list(self.axes)
```

```
    ~/.local/lib/python3.5/site-packages/pandas/core/internals.py in <listcomp>(.0)
   4148             new_blocks = [blk.take_nd(indexer, axis=axis, fill_tuple=(
   4149                 fill_value if fill_value is not None else blk.fill_value,))
-> 4150                 for blk in self.blocks]
   4151
   4152         new_axes = list(self.axes)


    ~/.local/lib/python3.5/site-packages/pandas/core/internals.py in take_nd(self, indexer,
   1219             fill_value = fill_tuple[0]
   1220             new_values = algos.take_nd(values, indexer, axis=axis,
-> 1221                                        allow_fill=True, fill_value=fill_value)
   1222
   1223         if new_mgr_locs is None:


    ~/.local/lib/python3.5/site-packages/pandas/core/algorithms.py in take_nd(arr, indexer,
   1377             out = np.empty(out_shape, dtype=dtype, order='F')
   1378         else:
-> 1379             out = np.empty(out_shape, dtype=dtype)
   1380
   1381     func = _get_take_nd_function(arr.ndim, arr.dtype, out.dtype, axis=axis,


    MemoryError:
```

MemoryError: el software no es capaz de hacer este cálculo con 3.5 gb de memoria :/