

Did elimination of copayment increase visits to physicians?

In [18]:

```
# Import a package with tools (pandas) and call it pd
import pandas as pd
```

In [19]:

```
# Read the data into a dataframe called df
df = pd.read_csv('gp data.csv')
```

In [20]:

```
# Have a look at the data
df.head()
```

Out[20]:

	Unnamed: 0	age	gender	year	visits	population
0	86	2	1	2006	49814	28361
1	87	2	1	2007	49077	28510
2	88	2	1	2008	50816	28592
3	89	2	1	2009	49637	29375
4	90	2	1	2010	52343	29333

In [21]:

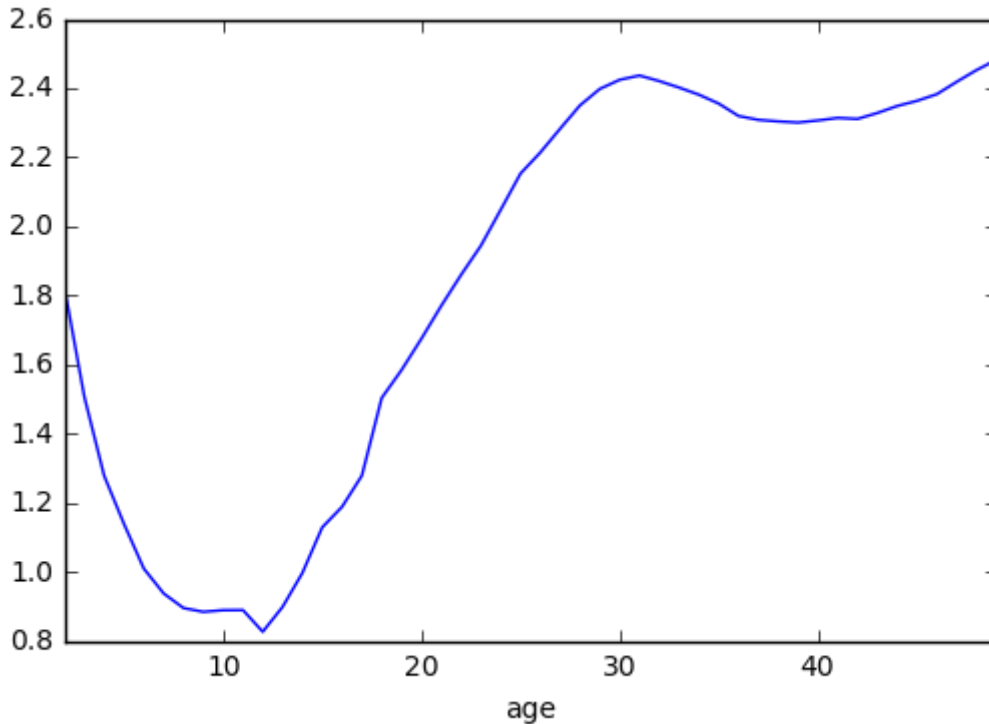
```
# create a variable showing visits per capita per year (in the different age groups)
df['visits_per_capita'] = df.visits/df.population
```

In [22]:

```
# plot the average number of visits in different age groups
%matplotlib inline
df.groupby('age')['visits_per_capita'].mean().plot()
```

Out[22]:

<matplotlib.axes._subplots.AxesSubplot at 0x1dc33d2a048>



In [23]:

```
#disable warnings (you should not do this when you work)
#import warnings
#warnings.filterwarnings('ignore')
```

Interrupted time design

In [24]:

```
# Did the number of visits to the physician change after 2010?  
# simplify and select only one age and one gender  
# put this data into a new dataframe called df14f  
# (dataframe with information about 14 year old females)  
  
df14f = df.query('(age==14) & (gender==1)')  
  
# create a variable to indicate whether it was before or after the reform  
df14f['copay']=0  
df14f.copay[df14f.year>2009] = 1
```

In [25]:

```
# run the interrupted time design model

import statsmodels.formula.api as smf

model = 'visits_per_capita ~ copay + year'

mod = smf.ols(model, data=df14f)
res = mod.fit()
print (res.summary())
```

OLS Regression Results

```

=====
====
Dep. Variable:      visits_per_capita    R-squared:
0.984
Model:              OLS    Adj. R-squared:
0.977
Method:             Least Squares    F-statistic:          1
52.2
Date:               Thu, 02 Feb 2017    Prob (F-statistic):    3.32
e-05
Time:               01:12:04    Log-Likelihood:        2
0.899
No. Observations:      8    AIC:                    -3
5.80
Df Residuals:          5    BIC:                    -3
5.56
Df Model:              2

```

Covariance Type: nonrobust

```

=====
====
               coef    std err          t      P>|t|      [95.0% Conf. I
nt.]
-----
----
Intercept    -56.8936     14.254     -3.991     0.010     -93.535    -2
0.252
copay         0.1541      0.033      4.737     0.005       0.070
0.238
year          0.0288      0.007      4.060     0.010       0.011
0.047

```

```

=====
====
Omnibus:          0.263    Durbin-Watson:
2.806
Prob(Omnibus):    0.877    Jarque-Bera (JB):
0.072
Skew:             0.080    Prob(JB):
0.965
Kurtosis:         2.563    Cond. No.          3.61
e+06

```

Warnings:

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

[2] The condition number is large, 3.61e+06. This might indicate that there are strong multicollinearity or other numerical problems.

Difference-in-Difference

The aim is now to compare the average number of visits before and after the reform for one group that was affected by the reform and one group who is as similar as possible, but unaffected. For instance: Compare 14 year olds (who are affected since they do no longer have to pay to visits the doctor after 2010) and 10 year olds (who are not affected since they never had to pay. In order to do so we first reshape the dataframe so it only includes 10 and 14 year olds (of one gender). Moreover we focus only on visits per capita in these age groups, and it would be convenient if each age group had a separate column (as opposed to being stacked on top of each other). So, before doing the calculations we will reshape and fix this.

In [26]:

```
#create a new dataframe (copies the old one)
df2 = df.set_index(['age', 'year'])
df2 = df2[df2.gender==1]
```

In [27]:

```
# focus on only one variable and unstack age
df3 = df2['visits_per_capita'].copy()
df3 = df3.unstack('age')
```

In [28]:

```
# change the column names so there is a "age" in front of each number/age
cols = df3.columns.tolist()
cols2 = ['age'+str(x) for x in cols]
df3.columns = cols2
```

In [29]:

```
# create a new dataframe with only 10 and 14 year olds
dfsmall = df3[['age10', 'age14']]
```

In [30]:

```
# create a variable indicating whether something is before or after the reform
dfsmall['copay']='before'
dfsmall.copay[dfsmall.index>2009] = 'after'
```

In [31]:

```
# get the average number of visits to the doctor before and after the reform for each age group
avg = dfsmall.groupby('copay').mean()
avg
```

Out[31]:

	age10	age14
copay		
after	0.944511	1.246069
before	0.865694	0.976641

In [32]:

```
# calculate the differences between the age groups  
# and the difference between the difference before and after the reform  
avg['difference'] = avg.age14 - avg.age10  
avg.difference.after - avg.difference.before
```

Out[32]:

0.19061068722838814

Synthetic control

In the difference-in-difference model we compared the 14 year olds to the 10 year olds. We could have compared against 8,9, 10, 11, 17 or many other age groups. The idea behind synthetic control is that instead of just selecting one group, we allow the computer to select a combination of the other possible comparisons. We would like to select a comparison with a good fit between our group (the 14 year olds) so we first run a regression to determine what combination of the other groups that best predict visits among 14 year olds.

In [33]:

```
import statsmodels.formula.api as smf
model = 'age14 ~ age8 + age9'
mod = smf.ols(model, data=df3[df3.index<2010])
res = mod.fit()
print (res.summary())

# after determinig the importance of the other groups,
# we use the results of the model to predic what the pattern
# should be after the reform (if it follows the same pattern as previously)

df3['control14'] = res.predict(exog = df3)

# the difference between the predicted visits and observed visits (for 14
# year olds) is the impact of the reform. We plot this.

df3[['age14', 'control14']].plot()
```

OLS Regression Results

```

=====
====
Dep. Variable:          age14    R-squared:
0.805
Model:                  OLS      Adj. R-squared:
0.414
Method:                 Least Squares    F-statistic:
2.062
Date:                   Thu, 02 Feb 2017    Prob (F-statistic):
0.442
Time:                   01:12:09    Log-Likelihood:          1
2.339
No. Observations:      4    AIC:          -1
8.68
Df Residuals:          1    BIC:          -2
0.52
Df Model:              2

```

Covariance Type: nonrobust

```

=====
====
               coef    std err          t      P>|t|      [95.0% Conf. I
nt.]
-----
----
Intercept      2.7562      1.995      1.381      0.399      -22.594      2
8.106
age8           -4.6958      3.753     -1.251      0.429      -52.384      4
2.993
age9           2.6915      1.615      1.667      0.344      -17.823      2
3.206

```

```

=====
====
Omnibus:          nan    Durbin-Watson:
1.649
Prob(Omnibus):    nan    Jarque-Bera (JB):
0.502
Skew:            -0.653    Prob(JB):
0.778
Kurtosis:        1.858    Cond. No.
647.

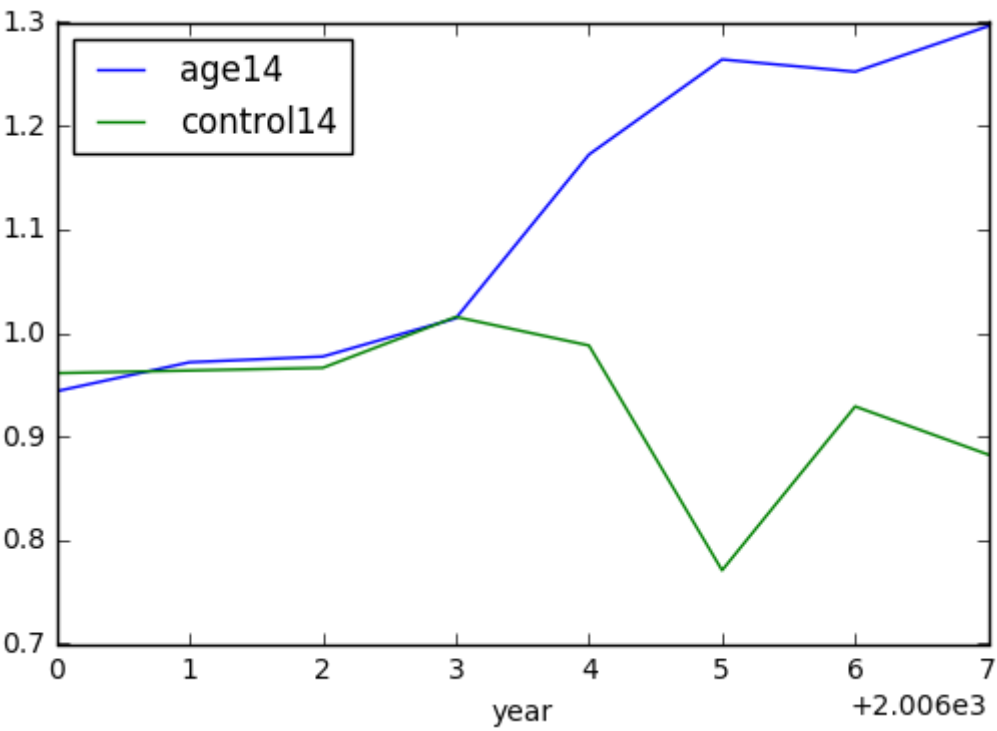
```

Warnings:

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

Out[33]:

<matplotlib.axes._subplots.AxesSubplot at 0x1dc33d33e80>



In []:

In []: