Hypothesis Testing using Housing, Universities and GDP of USA Dataset

Definitions:

- A *quarter* is a specific three month period, Q1 is January through March, Q2 is April through June, Q3 is July through September, Q4 is October through December.
- A *recession* is defined as starting with two consecutive quarters of GDP decline, and ending with two consecutive quarters of GDP growth.
- A *recession bottom* is the quarter within a recession which had the lowest GDP.
- A *university town* is a city which has a high percentage of university students compared to the total population of the city.

Hypothesis: University towns have their mean housing prices less effected by recessions. Run a t-test to compare the ratio of the mean price of houses in university towns the quarter before the recession starts compared to the recession bottom. (`price_ratio=quarter_before_recession/recession_bottom`)

The following data files are using:

- From the Zillow research data site there is housing data for the United States. In particular the datafile for all homes at a city level, `City_Zhvi_AllHomes.csv`, has median home sale prices at a fine grained level.
- From the Wikipedia page on college towns is a list of university towns in the United States which has been copy and pasted into the file `university_towns.txt`.
- From Bureau of Economic Analysis, US Department of Commerce, the GDP over time of the United States in current dollars (use the chained value in 2009 dollars), in quarterly intervals, in the file `gdplev.xls`. For this assignment, only look at GDP data from the first quarter of 2000 onward.

Each function in this assignment below is worth 10%, with the exception of `run_ttest()`, which is worth 50%.

In [8]:

```python
import pandas as pd
import numpy as np
from scipy.stats import ttest_ind
```

In [9]:

```python
# Use this dictionary to map state names to two letter acronyms
states = {'OH': 'Ohio', 'KY': 'Kentucky', 'AS': 'American Samoa', 'NV': 'Nevada', 'WY': 'Wyoming', 'NA': 'National', 'AL': 'Alabama', 'MD': 'Maryland', 'AK': 'Alaska', 'UT': 'Utah', 'OR': 'Oregon', 'MT': 'Montana', 'IL': 'Illinois', 'TN': 'Tennessee', 'DC': 'District of Columbia', 'VT': 'Vermont', 'ID': 'Idaho', 'AR': 'Arkansas', 'ME': 'Maine', 'WA': 'Washington', 'HI': 'Hawaii', 'WI': 'Wisconsin', 'MI': 'Michigan', 'IN': 'Indiana', 'NJ': 'New Jersey', 'AZ': 'Arizona', 'GU': 'Guam', 'MS': 'Mississippi', 'PR': 'Puerto Rico', 'NC': 'North Carolina', 'TX': 'Texas', 'SD': 'South Dakota', 'MP': 'Northern Mariana Islands', 'IA': 'Iowa', 'MO': 'Missouri', 'CT': 'Connecticut', 'WV': 'West Virginia', 'SC': 'South Carolina', 'LA': 'Louisiana', 'KS': 'Kansas', 'NY': 'New York', 'NE': 'Nebraska', 'OK': 'Oklahoma', 'FL': 'Florida', 'CA': 'California', 'CO': 'Colorado', 'PA': 'Pennsylvania', 'DE': 'Delaware', 'NM': 'New Mexico', 'RI': 'Rhode Island', 'MN': 'Minnesota', 'VI': 'Virgin Islands', 'NH': 'New Hampshire', 'MA': 'Massachusetts', 'GA': 'Georgia', 'ND': 'North Dakota', 'VA': 'Virginia'}
```

In [19]:

```python
def get_list_of_university_towns():
    '''Returns a DataFrame of towns and the states they are in from the
    university_towns.txt list. The format of the DataFrame should be:
    DataFrame( [ ["Michigan", "Ann Arbor"], ["Michigan", "Yipsilanti"] ],
    columns=["State", "RegionName"]  )

    The following cleaning needs to be done:

    1. For "State", removing characters from "[" to the end.
    2. For "RegionName", when applicable, removing every character from " (" to the end.
    3. Depending on how you read the data, you may need to remove newline character '\n'. '''
    data = []
    state = None
    state_towns = []
    with open('E:\Protfolio\Hypothesis Testing using Housing, Universities and GDP of USA Dataset/university_towns.txt') as file:
        for line in file:
            thisLine = line[:-1]
            if thisLine[-6:] == '[edit]':
                state = thisLine[:-6]
                continue
            if '(' in line:
                town = thisLine[:thisLine.index('(')-1]
                state_towns.append([state,town])
            else:
                town = thisLine
                state_towns.append([state,town])
            data.append(thisLine)
    df = pd.DataFrame(state_towns,columns = ['State','RegionName'])
    return df
get_list_of_university_towns()
```

Out[19]:

	State	RegionName
0	Alabama	Auburn
1	Alabama	Florence
2	Alabama	Jacksonville
3	Alabama	Livingston
4	Alabama	Montevallo
...	...	...
512	Wisconsin	River Falls
513	Wisconsin	Stevens Point
514	Wisconsin	Waukesha
515	Wisconsin	Whitewater
516	Wyoming	Laramie

517 rows × 2 columns

In [11]:

```python
def get_recession_start():
    '''Returns the year and quarter of the recession start time as a
    string value in a format such as 2005q3'''
    gdplev = pd.ExcelFile('E:\Protfolio\Hypothesis Testing using Housing, Universities and GDP of USA Dataset/gdplev.xls')
    gdplev = gdplev.parse("Sheet1", skiprows=219)
    gdplev = gdplev[['1999q4', 9926.1]]
    gdplev.columns = ['Quarter','GDP']
    for i in range(2, len(gdplev)):
        if (gdplev.iloc[i-2][1] > gdplev.iloc[i-1][1]) and (gdplev.iloc[i-1][1] > gdplev.iloc[i][1]):
            return gdplev.iloc[i-2][0]
get_recession_start()
```

Out[11]: '2008q3'

In [17]:

```python
def get_recession_end():
    '''Returns the year and quarter of the recession end time as a
    string value in a format such as 2005q3'''
    gdplev = pd.ExcelFile('E:\Protfolio\Hypothesis Testing using Housing, Universities and GDP of USA Dataset/gdplev.xls')
    gdplev = gdplev.parse("Sheet1", skiprows=219)
    gdplev = gdplev[['1999q4', 9926.1]]
    gdplev.columns = ['Quarter','GDP']
    start = get_recession_start()
    start_index = gdplev[gdplev['Quarter'] == start].index.tolist()[0]
    gdplev=gdplev.iloc[start_index:]
    for i in range(2, len(gdplev)):
        if (gdplev.iloc[i-2][1] < gdplev.iloc[i-1][1]) and (gdplev.iloc[i-1][1] < gdplev.iloc[i][1]):
            return gdplev.iloc[i][0]

get_recession_end()
```

Out[17]: '2009q4'

In [20]:

```python
def get_recession_bottom():
    '''Returns the year and quarter of the recession bottom time as a
    string value in a format such as 2005q3'''
    gdplev = pd.ExcelFile('E:\Protfolio\Hypothesis Testing using Housing, Universities and GDP of USA Dataset/gdplev.xls')
    gdplev = gdplev.parse("Sheet1", skiprows=219)
    gdplev = gdplev[['1999q4', 9926.1]]
    gdplev.columns = ['Quarter','GDP']
    start = get_recession_start()
    start_index = gdplev[gdplev['Quarter'] == start].index.tolist()[0]
    end = get_recession_end()
    end_index = gdplev[gdplev['Quarter'] == end].index.tolist()[0]
    gdplev=gdplev.iloc[start_index:end_index+1]
    bottom = gdplev['GDP'].min()
    bottom_index = gdplev[gdplev['GDP'] == bottom].index.tolist()[0]-start_index
    return gdplev.iloc[bottom_index]['Quarter']

get_recession_bottom()
```

Out[20]: '2009q2'

In [23]:

```python
def convert_housing_data_to_quarters():
    '''Converts the housing data to quarters and returns it as mean
    values in a dataframe. This dataframe should be a dataframe with
    columns for 2000q1 through 2016q3, and should have a multi-index
    in the shape of ["State","RegionName"].

    Note: Quarters are defined in the assignment description, they are
    not arbitrary three month periods.

    The resulting dataframe should have 67 columns, and 10,730 rows.
    '''
    hdata = pd.read_csv('E:\Protfolio\Hypothesis Testing using Housing, Universities and GDP of USA Dataset/City_Zhvi_AllHomes.csv')
    hdata = hdata.drop(hdata.columns[[0]+list(range(3,51))],axis=1)
    hdata2 = pd.DataFrame(hdata[['State','RegionName']])
    for year in range(2000,2016):
        hdata2[str(year)+'q1'] = hdata[[str(year)+'-01',str(year)+'-02',str(year)+'-03']].mean(axis=1)
        hdata2[str(year)+'q2'] = hdata[[str(year)+'-04',str(year)+'-05',str(year)+'-06']].mean(axis=1)
        hdata2[str(year)+'q3'] = hdata[[str(year)+'-07',str(year)+'-08',str(year)+'-09']].mean(axis=1)
        hdata2[str(year)+'q4'] = hdata[[str(year)+'-10',str(year)+'-11',str(year)+'-12']].mean(axis=1)
    year = 2016
    hdata2[str(year)+'q1'] = hdata[[str(year)+'-01',str(year)+'-02',str(year)+'-03']].mean(axis=1)
    hdata2[str(year)+'q2'] = hdata[[str(year)+'-04',str(year)+'-05',str(year)+'-06']].mean(axis=1)
    hdata2[str(year)+'q3'] = hdata[[str(year)+'-07',str(year)+'-08']].mean(axis=1)
    hdata2 = hdata2.replace({'State':states})
    hdata2 = hdata2.set_index(['State','RegionName'])
    return hdata2

convert_housing_data_to_quarters()
```

Out[23]:

State	RegionName	2000q1	2000q2	2000q3	2000q4	2001q1	2001q2	2001q3
New York	**New York**	NaN	NaN	NaN	NaN	NaN	NaN	NaN
California	**Los Angeles**	207066.666667	214466.666667	220966.666667	226166.666667	233000.000000	239100.000000	245066.666667
Illinois	**Chicago**	138400.000000	143633.333333	147866.666667	152133.333333	156933.333333	161800.000000	166400.000000
Pennsylvania	**Philadelphia**	53000.000000	53633.333333	54133.333333	54700.000000	55333.333333	55533.333333	56266.666667
Arizona	**Phoenix**	111833.333333	114366.666667	116000.000000	117400.000000	119600.000000	121566.666667	122700.000000
...	**...**	...	...	...	...	...	...	...
Wisconsin	**Town of Wrightstown**	101766.666667	105400.000000	111366.666667	114866.666667	125966.666667	129900.000000	129900.000000
New York	**Urbana**	79200.000000	81666.666667	91700.000000	98366.666667	94866.666667	98533.333333	102966.666667
Wisconsin	**New Denmark**	114566.666667	119266.666667	126066.666667	131966.666667	143800.000000	146966.666667	148366.666667
California	**Angels**	151000.000000	155900.000000	158100.000000	167466.666667	176833.333333	183766.666667	190233.333333
Wisconsin	**Holland**	151033.333333	150500.000000	153233.333333	155833.333333	161866.666667	165733.333333	168033.333333

10730 rows × 67 columns

In [24]:

```python
def run_ttest():
    '''First creates new data showing the decline or growth of housing prices
    between the recession start and the recession bottom. Then runs a ttest
    comparing the university town values to the non-university towns values,
    return whether the alternative hypothesis (that the two groups are the same)
    is true or not as well as the p-value of the confidence.

    Return the tuple (different, p, better) where different=True if the t-test is
    True at a p<0.01 (we reject the null hypothesis), or different=False if
    otherwise (we cannot reject the null hypothesis). The variable p should
    be equal to the exact p value returned from scipy.stats.ttest_ind(). The
    value for better should be either "university town" or "non-university town"
    depending on which has a lower mean price ratio (which is equivilent to a
    reduced market loss).'''

    unitowns = get_list_of_university_towns()
    bottom = get_recession_bottom()
    start = get_recession_start()
    hdata = convert_housing_data_to_quarters()
    bstart = hdata.columns[hdata.columns.get_loc(start) -1]

    hdata['ratio'] = hdata[bottom] - hdata[bstart]
    hdata = hdata[[bottom,bstart,'ratio']]
    hdata = hdata.reset_index()
    unitowns_hdata = pd.merge(hdata,unitowns,how='inner',on=['State','RegionName'])
    unitowns_hdata['uni'] = True
    hdata2 = pd.merge(hdata,unitowns_hdata,how='outer',on=['State','RegionName',bottom,bstart,'ratio'])
    hdata2['uni'] = hdata2['uni'].fillna(False)

    ut = hdata2[hdata2['uni'] == True]
    nut = hdata2[hdata2['uni'] == False]

    t,p = ttest_ind(ut['ratio'].dropna(),nut['ratio'].dropna())

    different = True if p < 0.01 else False

    better = "non-university town" if ut['ratio'].mean() < nut['ratio'].mean() else "university town"

    return different, p, better

run_ttest()
```

Out[24]: (True, 0.002099659657952052, 'university town')

In [ ]: