

Assignment 4

May 3, 2020

You are currently looking at **version 1.1** of this notebook. To download notebooks and datafiles, as well as get help on Jupyter notebooks in the Coursera platform, visit the [Jupyter Notebook FAQ](#) course resource.

```
In [8]: import pandas as pd
import numpy as np
from scipy.stats import ttest_ind
```

1 Assignment 4 - Hypothesis Testing

This assignment requires more individual learning than previous assignments - you are encouraged to check out the [pandas documentation](#) to find functions or methods you might not have used yet, or ask questions on [Stack Overflow](#) and tag them as pandas and python related. And of course, the discussion forums are open for interaction with your peers and the course staff.

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 available for this assignment: * 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 gdp1ev.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 [9]: # Use this dictionary to map state names to two letter acronyms
        states = {'OH': 'Ohio', 'KY': 'Kentucky', 'AS': 'American Samoa', 'NV': 'Nevada', 'WY':

In [11]: 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'

import re

# Open the file, read the lines, and close the file
fo = open('university_towns.txt', "r")
lines = fo.readlines()
fo.close()

# remove empty lines
new_lines = []
for line in lines:
    if not re.match(r'^\s*$', line):
        new_lines.append(line)

lines = new_lines.copy()

# Strip the white space at the beginning and end of each line
for index, line in enumerate(lines):
    lines[index] = line.strip()

    # Loop through the lines to form a dataframe
df_result = pd.DataFrame(columns=('State', 'RegionName'))
i = 0 # counter for each new line in the dataframe
state_string = "" # Empty initial state string
region_string = "" # Empty initial region string
for line in lines:
    if '[edit]' in line:
        state_string = line.replace('[edit]', "")
    else:
        region_string = re.sub(r' \(.+', "", line)
        df_result.loc[i] = [state_string, region_string]
        i += 1

return df_result

```

```
get_list_of_university_towns()
```

```
Out[11]:
```

	State	RegionName
0	Alabama	Auburn
1	Alabama	Florence
2	Alabama	Jacksonville
3	Alabama	Livingston
4	Alabama	Montevallo
5	Alabama	Troy
6	Alabama	Tuscaloosa
7	Alabama	Tuskegee
8	Alaska	Fairbanks
9	Arizona	Flagstaff
10	Arizona	Tempe
11	Arizona	Tucson
12	Arkansas	Arkadelphia
13	Arkansas	Conway
14	Arkansas	Fayetteville
15	Arkansas	Jonesboro
16	Arkansas	Magnolia
17	Arkansas	Monticello
18	Arkansas	Russellville
19	Arkansas	Searcy
20	California	Angwin
21	California	Arcata
22	California	Berkeley
23	California	Chico
24	California	Claremont
25	California	Cotati
26	California	Davis
27	California	Irvine
28	California	Isla Vista
29	California	University Park, Los Angeles
...
487	Virginia	Wise
488	Virginia	Chesapeake
489	Washington	Bellingham
490	Washington	Cheney
491	Washington	Ellensburg
492	Washington	Pullman
493	Washington	University District, Seattle
494	West Virginia	Athens
495	West Virginia	Buckhannon
496	West Virginia	Fairmont
497	West Virginia	Glenville
498	West Virginia	Huntington
499	West Virginia	Montgomery

500	West Virginia	Morgantown
501	West Virginia	Shepherdstown
502	West Virginia	West Liberty
503	Wisconsin	Appleton
504	Wisconsin	Eau Claire
505	Wisconsin	Green Bay
506	Wisconsin	La Crosse
507	Wisconsin	Madison
508	Wisconsin	Menomonie
509	Wisconsin	Milwaukee
510	Wisconsin	Oshkosh
511	Wisconsin	Platteville
512	Wisconsin	River Falls
513	Wisconsin	Stevens Point
514	Wisconsin	Waukesha
515	Wisconsin	Whitewater
516	Wyoming	Laramie

[517 rows x 2 columns]

```
In [12]: def get_recession_start():
    '''Returns the year and quarter of the recession start time as a
    string value in a format such as 2008q3'''

    import pandas as pd
    df_gdp = pd.read_excel('gdplev.xls', skiprows=220, parse_cols='E,G', header=None)
    df_gdp.columns = ['quarter', 'GDP']

    for i in range(4, len(df_gdp)):
        if (df_gdp.loc[i-4, 'GDP'] > df_gdp.loc[i-3, 'GDP']) \
            and (df_gdp.loc[i-3, 'GDP'] > df_gdp.loc[i-2, 'GDP']) \
            and (df_gdp.loc[i-2, 'GDP'] < df_gdp.loc[i-1, 'GDP']) \
            and (df_gdp.loc[i-1, 'GDP'] < df_gdp.loc[i, 'GDP']):
            recession_base_idx = i-4

    result = df_gdp.loc[recession_base_idx, 'quarter']
    j = recession_base_idx
    while True:
        if (df_gdp.loc[j-1, 'GDP'] > df_gdp.loc[j, 'GDP']):
            j -= 1
        else:
            result = df_gdp.loc[j+1, 'quarter']
            break

    return result

get_recession_start()

Out[12]: '2008q3'
```

```
In [13]: def get_recession_end():
    '''Returns the year and quarter of the recession end time as a
    string value in a format such as 2005q3'''

    import pandas as pd
    df_gdp = pd.read_excel('gdplev.xls', skiprows=220, parse_cols='E,G', header=None)
    df_gdp.columns = ['quarter', 'GDP']

    for i in range(4, len(df_gdp)):
        if (df_gdp.loc[i-4, 'GDP'] > df_gdp.loc[i-3, 'GDP']) \
            and (df_gdp.loc[i-3, 'GDP'] > df_gdp.loc[i-2, 'GDP']) \
            and (df_gdp.loc[i-2, 'GDP'] < df_gdp.loc[i-1, 'GDP']) \
            and (df_gdp.loc[i-1, 'GDP'] < df_gdp.loc[i, 'GDP']):
            result = df_gdp.loc[i, 'quarter']

    return result

get_recession_end()

Out[13]: '2009q4'
```

```
In [14]: def get_recession_bottom():
    '''Returns the year and quarter of the recession bottom time as a
    string value in a format such as 2005q3'''

    import pandas as pd
    df_gdp = pd.read_excel('gdplev.xls', skiprows=220, parse_cols='E,G', header=None)
    df_gdp.columns = ['quarter', 'GDP']

    for i in range(4, len(df_gdp)):
        if (df_gdp.loc[i-4, 'GDP'] > df_gdp.loc[i-3, 'GDP']) \
            and (df_gdp.loc[i-3, 'GDP'] > df_gdp.loc[i-2, 'GDP']) \
            and (df_gdp.loc[i-2, 'GDP'] < df_gdp.loc[i-1, 'GDP']) \
            and (df_gdp.loc[i-1, 'GDP'] < df_gdp.loc[i, 'GDP']):
            result = df_gdp.loc[i-2, 'quarter']

    return result

get_recession_bottom()

Out[14]: '2009q2'
```

```
In [17]: 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.

'''

```
import pandas as pd
import numpy as np
df = pd.read_csv('City_Zhvi_AllHomes.csv', header=0)

# Create columns to keep
cols_to_keep = ['RegionID', 'RegionName', 'State']
for i in range(2000, 2017):
    for j in range(1, 13):
        if j <= 9:
            if i == 2016 and j == 9:
                pass
            else:
                month_str = '0' + str(j)
        else:
            if i == 2016:
                pass
            else:
                month_str = str(j)
        cols_to_keep.append(str(i) + '-' + month_str)
df = df[cols_to_keep]

# Convert two letter state abbreviations to state names
df['State'] = df['State'].replace(states)

def convert_to_qtr(ym):
    year, month = ym.split('-')
    if month == '01' or month == '02' or month == '03':
        result = year + 'q1'
    elif month == '04' or month == '05' or month == '06':
        result = year + 'q2'
    elif month == '07' or month == '08' or month == '09':
        result = year + 'q3'
    else:
        result = year + 'q4'
    return result

# Stack the columns of GDP values and add a quarter column
df_compiled = df.copy().set_index(['State', 'RegionName', 'RegionID']).stack(dropna=False)
df_compiled = df_compiled.reset_index().rename(columns={'level_3': 'year_month', 0: 'gdp'})
df_compiled.drop_duplicates(inplace=True)
df_compiled['quarter'] = df_compiled['year_month'].apply(convert_to_qtr)
df_compiled = df_compiled.drop('year_month', axis=1)
result = df_compiled.pivot_table(values='gdp', index=['State', 'RegionName', 'RegionID'], columns='quarter')
```

```

result = result.reset_index()
result = result.drop('RegionID', axis=1)
#del result.index.name
result = result.set_index(['State', 'RegionName'])
return result

```

```
convert_housing_data_to_quarters()
```

```

Out[17]: quarter          2000q1          2000q2          2000q3  \
State  RegionName
Alabama Adamsville      69033.333333      69166.666667      69800.000000
        Alabaster       122133.333333      123066.666667      123166.666667
        Albertville     73966.666667       72600.000000       72833.333333
        Arab            83766.666667       81566.666667       81333.333333
        Ardmore          NaN              NaN              NaN
        Axis             NaN              NaN              NaN
        Baileyton        NaN              NaN              NaN
        Bay Minette      81700.000000       78533.333333       79133.333333
        Bayou La Batre   44066.666667       44500.000000       44266.666667
        Bessemer         NaN              NaN              NaN
        Birmingham      54033.333333       54400.000000       54966.666667
        Boaz            70866.666667       70266.666667       70300.000000
        Brent           92933.333333       94333.333333       96166.666667
        Brighton         NaN              NaN              NaN
        Brookwood       92566.666667       95100.000000       98866.666667
        Buhl            90800.000000       94600.000000       96500.000000
        Calera          108933.333333      110366.666667      108000.000000
        Center Point     80966.666667       81233.333333       81500.000000
        Centreville      95300.000000       96566.666667       98000.000000
        Chalkville       94100.000000       94433.333333       94433.333333
        Chancellor        NaN              NaN              NaN
        Chelsea         162066.666667      167033.333333      166900.000000
        Chickasaw        51200.000000       53666.666667       54933.333333
        Chunchula        80266.666667       81766.666667       82200.000000
        Citronelle       64833.333333       66633.333333       68066.666667
        Clay            120900.000000      122266.666667      123966.666667
        Coden            62600.000000       64800.000000       66866.666667
        Coker           118100.000000      120766.666667      118166.666667
        Concord          78600.000000       78700.000000       80133.333333
        Cottondale      100833.333333      102633.333333      104766.666667
        ...
Wisconsin Vernon        183200.000000      178200.000000      174300.000000
        Vienna          178033.333333      181533.333333      182433.333333
        Vinland         119800.000000      126766.666667      134933.333333
        Wales            NaN              NaN              NaN
        Waterford       121200.000000      119433.333333      120200.000000
        Waukesha        141266.666667      138433.333333      136733.333333
        Waunakee        187400.000000      191433.333333      192666.666667

```

	Waupun	84000.000000	84000.000000	84400.000000
	Wausau	69566.666667	69466.666667	71033.333333
	Wayne	NaN	NaN	NaN
	West Allis	80933.333333	81933.333333	84066.666667
	Weston	80166.666667	82700.000000	84166.666667
	Wheatland	129800.000000	132133.333333	134933.333333
	Whitelaw	96033.333333	101433.333333	108033.333333
	Williams Bay	122900.000000	115300.000000	110766.666667
	Wilson	NaN	NaN	NaN
	Wilson	129033.333333	129366.666667	132433.333333
	Wind Lake	124366.666667	123666.666667	124133.333333
	Wind Point	149233.333333	145266.666667	140866.666667
	Wisconsin Dells	100000.000000	103400.000000	105000.000000
	Wolf River	95166.666667	99333.333333	103800.000000
	Woodruff	NaN	NaN	NaN
	Woodville	NaN	NaN	NaN
	Yorkville	139066.666667	134900.000000	135033.333333
Wyoming	Bar Nunn	93033.333333	89666.666667	88900.000000
	Burns	101533.333333	104566.666667	108366.666667
	Casper	89233.333333	89600.000000	89733.333333
	Cheyenne	116866.666667	120033.333333	121533.333333
	Evansville	128033.333333	128766.666667	130833.333333
	Pine Bluffs	93733.333333	95066.666667	94633.333333

quarter		2000q4	2001q1	2001q2 \
State	RegionName			
Alabama	Adamsville	71966.666667	73466.666667	74000.000000
	Alabaster	123700.000000	123233.333333	125133.333333
	Albertville	74200.000000	75900.000000	76000.000000
	Arab	82966.666667	84200.000000	84533.333333
	Ardmore	NaN	NaN	NaN
	Axis	NaN	NaN	NaN
	Baileyton	NaN	NaN	NaN
	Bay Minette	81300.000000	85700.000000	87266.666667
	Bayou La Batre	43666.666667	42500.000000	43333.333333
	Bessemer	NaN	NaN	NaN
	Birmingham	56066.666667	56833.333333	57600.000000
	Boaz	71466.666667	72833.333333	71900.000000
	Brent	98333.333333	96533.333333	98500.000000
	Brighton	NaN	NaN	NaN
	Brookwood	99966.666667	101666.666667	103666.666667
	Buhl	96566.666667	99266.666667	101966.666667
	Calera	107933.333333	109333.333333	112200.000000
	Center Point	82233.333333	83366.666667	84333.333333
	Centreville	99366.666667	98100.000000	99266.666667
	Chalkville	94433.333333	96066.666667	97433.333333
	Chancellor	NaN	NaN	NaN
	Chelsea	167400.000000	170633.333333	174300.000000

	Chickasaw	56066.666667	55133.333333	53566.666667	
	Churchula	83400.000000	85400.000000	86100.000000	
	Citronelle	67766.666667	67866.666667	70000.000000	
	Clay	126500.000000	128033.333333	128300.000000	
	Coden	68566.666667	68933.333333	71300.000000	
	Coker	115333.333333	114066.666667	115700.000000	
	Concord	82800.000000	85133.333333	85800.000000	
	Cottondale	105300.000000	106733.333333	109233.333333	
...		
Wisconsin	Vernon	177466.666667	186233.333333	190500.000000	
	Vienna	186300.000000	190600.000000	191566.666667	
	Vinland	137833.333333	144300.000000	148200.000000	
	Wales	NaN	NaN	NaN	
	Waterford	121966.666667	127000.000000	133000.000000	
	Waukesha	139833.333333	147600.000000	151066.666667	
	Waunakee	193733.333333	197333.333333	198500.000000	
	Waupun	84533.333333	85100.000000	86600.000000	
	Wausau	69833.333333	69866.666667	71366.666667	
	Wayne	NaN	NaN	NaN	
	West Allis	80666.666667	79100.000000	80333.333333	
	Weston	85800.000000	87266.666667	88500.000000	
	Wheatland	137833.333333	142566.666667	143000.000000	
	Whitelaw	114000.000000	116333.333333	114966.666667	
	Williams Bay	110700.000000	112600.000000	117100.000000	
	Wilson	NaN	NaN	NaN	
	Wilson	137200.000000	143533.333333	143833.333333	
	Wind Lake	126066.666667	132133.333333	139466.666667	
	Wind Point	147866.666667	158666.666667	163966.666667	
	Wisconsin Dells	105433.333333	105533.333333	107100.000000	
	Wolf River	107633.333333	111400.000000	115033.333333	
	Woodruff	NaN	NaN	NaN	
	Woodville	NaN	NaN	NaN	
	Yorkville	140300.000000	148800.000000	155233.333333	
Wyoming	Bar Nunn	93500.000000	98833.333333	99900.000000	
	Burns	113000.000000	115833.333333	117200.000000	
	Casper	93166.666667	95500.000000	97633.333333	
	Cheyenne	123633.333333	125533.333333	126300.000000	
	Evansville	132066.666667	130566.666667	131433.333333	
	Pine Bluffs	98066.666667	103233.333333	104600.000000	
quarter		2001q3	2001q4	2002q1	\
State	RegionName				
Alabama	Adamsville	73333.333333	73100.000000	73333.333333	
	Alabaster	127766.666667	127200.000000	127300.000000	
	Albertville	72066.666667	73566.666667	76533.333333	
	Arab	81666.666667	83900.000000	87266.666667	
	Ardmore	NaN	NaN	NaN	
	Axis	NaN	NaN	NaN	

Baileyton	NaN	NaN	NaN
Bay Minette	85900.000000	85000.000000	84066.666667
Bayou La Batre	45433.333333	45400.000000	45400.000000
Bessemer	NaN	NaN	NaN
Birmingham	58433.333333	58700.000000	59500.000000
Boaz	68733.333333	69833.333333	72766.666667
Brent	99366.666667	104100.000000	103800.000000
Brighton	NaN	NaN	NaN
Brookwood	101833.333333	99900.000000	99633.333333
Buhl	102000.000000	102733.333333	102533.333333
Calera	114333.333333	112133.333333	111500.000000
Center Point	84466.666667	85333.333333	86100.000000
Centreville	100400.000000	104633.333333	105233.333333
Chalkville	98400.000000	99366.666667	99866.666667
Chancellor	NaN	NaN	NaN
Chelsea	174900.000000	173300.000000	170566.666667
Chickasaw	53533.333333	53400.000000	54266.666667
Churchula	86733.333333	86566.666667	86166.666667
Citronelle	71466.666667	70933.333333	69833.333333
Clay	129233.333333	130466.666667	130900.000000
Coden	72733.333333	73600.000000	72933.333333
Coker	118833.333333	122566.666667	122366.666667
Concord	84866.666667	84766.666667	86366.666667
Cottondale	109200.000000	109900.000000	109333.333333
...
Wisconsin Vernon	193100.000000	197466.666667	201800.000000
Vienna	193733.333333	194833.333333	198466.666667
Vinland	147733.333333	151200.000000	149233.333333
Wales	NaN	NaN	NaN
Waterford	139233.333333	143100.000000	145733.333333
Waukesha	153100.000000	155033.333333	158200.000000
Waunakee	201266.666667	202100.000000	206100.000000
Waupun	87666.666667	87100.000000	87533.333333
Wausau	73100.000000	74033.333333	76400.000000
Wayne	NaN	NaN	NaN
West Allis	81433.333333	82400.000000	84966.666667
Weston	90066.666667	91766.666667	94833.333333
Wheatland	143966.666667	142633.333333	143933.333333
Whitelaw	115666.666667	115866.666667	117766.666667
Williams Bay	119533.333333	121400.000000	126566.666667
Wilson	NaN	NaN	NaN
Wilson	143233.333333	141966.666667	142466.666667
Wind Lake	144200.000000	147333.333333	150933.333333
Wind Point	170833.333333	169566.666667	170533.333333
Wisconsin Dells	109866.666667	111633.333333	115833.333333
Wolf River	116900.000000	118233.333333	118600.000000
Woodruff	NaN	NaN	NaN
Woodville	NaN	NaN	NaN

	Yorkville	161133.333333	162800.000000	164866.666667
Wyoming	Bar Nunn	99100.000000	98333.333333	98633.333333
	Burns	117800.000000	117633.333333	117333.333333
	Casper	99433.333333	100633.333333	101733.333333
	Cheyenne	126466.666667	128133.333333	128466.666667
	Evansville	132400.000000	133466.666667	133300.000000
	Pine Bluffs	106500.000000	104066.666667	102233.333333
quarter		2002q2	...	2014q2 \
State	RegionName		...	
Alabama	Adamsville	73133.333333	...	77066.666667
	Alabaster	128000.000000	...	147133.333333
	Albertville	76366.666667	...	84033.333333
	Arab	87700.000000	...	113366.666667
	Ardmore	NaN	...	140533.333333
	Axis	NaN	...	111066.666667
	Baileyton	NaN	...	87666.666667
	Bay Minette	84566.666667	...	113333.333333
	Bayou La Batre	45566.666667	...	49800.000000
	Bessemer	NaN	...	54600.000000
	Birmingham	59866.666667	...	61733.333333
	Boaz	72100.000000	...	80933.333333
	Brent	102533.333333	...	93866.666667
	Brighton	NaN	...	45366.666667
	Brookwood	100366.666667	...	106633.333333
	Buhl	100100.000000	...	119233.333333
	Calera	111233.333333	...	125533.333333
	Center Point	86500.000000	...	78833.333333
	Centreville	105066.666667	...	95566.666667
	Chalkville	100733.333333	...	105233.333333
	Chancellor	NaN	...	116400.000000
	Chelsea	171100.000000	...	197133.333333
	Chickasaw	55000.000000	...	59000.000000
	Chunchula	86533.333333	...	120666.666667
	Citronelle	68133.333333	...	88600.000000
	Clay	131100.000000	...	134100.000000
	Coden	69100.000000	...	77566.666667
	Coker	119200.000000	...	141700.000000
	Concord	90366.666667	...	104733.333333
	Cottondale	107966.666667	...	128500.000000
...	
Wisconsin	Vernon	201966.666667	...	257733.333333
	Vienna	202466.666667	...	265700.000000
	Vinland	146433.333333	...	181766.666667
	Wales	NaN	...	262133.333333
	Waterford	151266.666667	...	213133.333333
	Waukesha	157833.333333	...	181166.666667
	Waunakee	208533.333333	...	278100.000000

	Waupun	87466.666667	...	113166.666667
	Wausau	78700.000000	...	91600.000000
	Wayne	NaN	...	238633.333333
	West Allis	87566.666667	...	90433.333333
	Weston	97600.000000	...	133233.333333
	Wheatland	146433.333333	...	169266.666667
	Whitelaw	116800.000000	...	120433.333333
	Williams Bay	131700.000000	...	223000.000000
	Wilson	NaN	...	170300.000000
	Wilson	139200.000000	...	159933.333333
	Wind Lake	156166.666667	...	228200.000000
	Wind Point	179733.333333	...	261233.333333
	Wisconsin Dells	117333.333333	...	137500.000000
	Wolf River	116800.000000	...	162666.666667
	Woodruff	NaN	...	172133.333333
	Woodville	NaN	...	162600.000000
	Yorkville	172466.666667	...	231500.000000
Wyoming	Bar Nunn	100866.666667	...	199233.333333
	Burns	117233.333333	...	168866.666667
	Casper	101533.333333	...	175766.666667
	Cheyenne	129633.333333	...	177466.666667
	Evansville	131066.666667	...	296733.333333
	Pine Bluffs	103566.666667	...	148666.666667

quarter		2014q3	2014q4	2015q1 \
State	RegionName			
Alabama	Adamsville	75966.666667	71900.000000	71666.666667
	Alabaster	147633.333333	148700.000000	148900.000000
	Albertville	84766.666667	86800.000000	88466.666667
	Arab	111700.000000	111600.000000	110166.666667
	Ardmore	139566.666667	140900.000000	143233.333333
	Axis	111833.333333	111800.000000	109533.333333
	Baileyton	90033.333333	94100.000000	94600.000000
	Bay Minette	110600.000000	109900.000000	111100.000000
	Bayou La Batre	51000.000000	51766.666667	50733.333333
	Bessemer	55333.333333	54733.333333	55200.000000
	Birmingham	60900.000000	59533.333333	59900.000000
	Boaz	81866.666667	82866.666667	83966.666667
	Brent	97100.000000	100100.000000	102433.333333
	Brighton	45133.333333	43766.666667	42500.000000
	Brookwood	110233.333333	114366.666667	117700.000000
	Buhl	132033.333333	138400.000000	141766.666667
	Calera	126733.333333	127633.333333	128366.666667
	Center Point	78433.333333	80200.000000	80100.000000
	Centreville	98866.666667	101866.666667	103933.333333
	Chalkville	104333.333333	105866.666667	106966.666667
	Chancellor	117000.000000	117133.333333	114733.333333
	Chelsea	199066.666667	199933.333333	202333.333333

	Chickasaw	60400.000000	59733.333333	58066.666667	
	Churchula	121600.000000	122733.333333	123666.666667	
	Citronelle	89633.333333	90766.666667	91300.000000	
	Clay	134300.000000	135466.666667	138466.666667	
	Coden	76866.666667	77800.000000	79300.000000	
	Coker	146300.000000	154333.333333	157566.666667	
	Concord	100233.333333	94800.000000	96200.000000	
	Cottondale	130566.666667	133433.333333	134500.000000	
...		
Wisconsin	Vernon	256200.000000	254566.666667	253633.333333	
	Vienna	267466.666667	267766.666667	268300.000000	
	Vinland	183500.000000	184233.333333	184033.333333	
	Wales	262233.333333	258666.666667	256600.000000	
	Waterford	213166.666667	213166.666667	218600.000000	
	Waukesha	180433.333333	180066.666667	180433.333333	
	Waunakee	283866.666667	283800.000000	283366.666667	
	Waupun	113066.666667	112666.666667	111666.666667	
	Wausau	93533.333333	92900.000000	94066.666667	
	Wayne	238966.666667	236000.000000	235500.000000	
	West Allis	90933.333333	88333.333333	90633.333333	
	Weston	133033.333333	131866.666667	133733.333333	
	Wheatland	170166.666667	173600.000000	174800.000000	
	Whitelaw	121700.000000	122766.666667	121933.333333	
	Williams Bay	230366.666667	232866.666667	230433.333333	
	Wilson	174433.333333	177733.333333	181833.333333	
	Wilson	157633.333333	154233.333333	154766.666667	
	Wind Lake	226133.333333	225966.666667	230800.000000	
	Wind Point	254666.666667	248000.000000	241866.666667	
	Wisconsin Dells	138933.333333	141433.333333	140500.000000	
	Wolf River	164733.333333	170433.333333	170533.333333	
	Woodruff	171600.000000	168600.000000	168166.666667	
	Woodville	169566.666667	171666.666667	172866.666667	
	Yorkville	238133.333333	237766.666667	239133.333333	
Wyoming	Bar Nunn	200933.333333	201133.333333	206666.666667	
	Burns	161933.333333	160433.333333	162600.000000	
	Casper	177300.000000	181000.000000	182066.666667	
	Cheyenne	176733.333333	178766.666667	181700.000000	
	Evansville	305666.666667	309500.000000	307300.000000	
	Pine Bluffs	154366.666667	158100.000000	163900.000000	
quarter		2015q2	2015q3	2015q4	\
State	RegionName				
Alabama	Adamsville	73033.333333	73933.333333	73866.666667	
	Alabaster	149566.666667	150366.666667	151733.333333	
	Albertville	89500.000000	90233.333333	91366.666667	
	Arab	109433.333333	110900.000000	112233.333333	
	Ardmore	143000.000000	144600.000000	143966.666667	
	Axis	109666.666667	110033.333333	109600.000000	

Baileyton	95666.666667	96833.333333	97233.333333
Bay Minette	114300.000000	118533.333333	121433.333333
Bayou La Batre	50500.000000	50133.333333	48933.333333
Bessemer	57200.000000	58633.333333	59433.333333
Birmingham	61966.666667	62666.666667	63100.000000
Boaz	85266.666667	86033.333333	86566.666667
Brent	101800.000000	97900.000000	95600.000000
Brighton	44233.333333	46266.666667	45766.666667
Brookwood	118700.000000	120066.666667	118533.333333
Buhl	141200.000000	141066.666667	145500.000000
Calera	128033.333333	128200.000000	128533.333333
Center Point	80233.333333	81000.000000	80900.000000
Centreville	104600.000000	100200.000000	101700.000000
Chalkville	107666.666667	109666.666667	111133.333333
Chancellor	109733.333333	109233.333333	110233.333333
Chelsea	205633.333333	207333.333333	208100.000000
Chickasaw	57633.333333	57466.666667	58300.000000
Churchula	124333.333333	123966.666667	123733.333333
Citronelle	92400.000000	94433.333333	94033.333333
Clay	139633.333333	139433.333333	141500.000000
Coden	81900.000000	83400.000000	80700.000000
Coker	157166.666667	156400.000000	155066.666667
Concord	103466.666667	106400.000000	110800.000000
Cottondale	135500.000000	137200.000000	137900.000000
...
Wisconsin Vernon	255600.000000	259100.000000	262766.666667
Vienna	268433.333333	272233.333333	280966.666667
Vinland	186266.666667	182933.333333	182233.333333
Wales	258666.666667	261400.000000	266900.000000
Waterford	227633.333333	230933.333333	230600.000000
Waukesha	183766.666667	187033.333333	190266.666667
Waunakee	289966.666667	290833.333333	296933.333333
Waupun	111366.666667	110733.333333	109700.000000
Wausau	94766.666667	95200.000000	94700.000000
Wayne	237900.000000	245800.000000	256100.000000
West Allis	91866.666667	92500.000000	99700.000000
Weston	137733.333333	139300.000000	139100.000000
Wheatland	176100.000000	174866.666667	176166.666667
Whitelaw	120033.333333	117266.666667	117766.666667
Williams Bay	223500.000000	224966.666667	226366.666667
Wilson	181066.666667	185300.000000	188466.666667
Wilson	157333.333333	161733.333333	166366.666667
Wind Lake	238466.666667	240200.000000	241833.333333
Wind Point	244333.333333	246566.666667	249833.333333
Wisconsin Dells	139500.000000	141066.666667	140566.666667
Wolf River	169400.000000	167400.000000	163266.666667
Woodruff	168200.000000	169266.666667	165600.000000
Woodville	171100.000000	176966.666667	181800.000000

Wyoming	Yorkville	243066.666667	243066.666667	244800.000000
	Bar Nunn	208866.666667	211800.000000	211800.000000
	Burns	163066.666667	164600.000000	164300.000000
	Casper	182633.333333	183300.000000	182700.000000
	Cheyenne	183266.666667	186766.666667	190666.666667
	Evansville	303166.666667	300966.666667	304200.000000
	Pine Bluffs	167433.333333	167466.666667	169200.000000
quarter		2016q1	2016q2	2016q3
State	RegionName			
Alabama	Adamsville	74166.666667	74933.333333	74700.0
	Alabaster	153466.666667	155100.000000	155850.0
	Albertville	92000.000000	92466.666667	92200.0
	Arab	110033.333333	110100.000000	112000.0
	Ardmore	142566.666667	143233.333333	141950.0
	Axis	110266.666667	112200.000000	112750.0
	Baileyton	96766.666667	98900.000000	102200.0
	Bay Minette	120266.666667	118333.333333	118500.0
	Bayou La Batre	48566.666667	47833.333333	47400.0
	Bessemer	59766.666667	59866.666667	59800.0
	Birmingham	62033.333333	61633.333333	61250.0
	Boaz	85966.666667	87366.666667	88600.0
	Brent	94433.333333	91366.666667	87650.0
	Brighton	45033.333333	44433.333333	44000.0
	Brookwood	115233.333333	111966.666667	108650.0
	Buhl	139800.000000	135566.666667	128350.0
	Calera	129633.333333	129933.333333	129600.0
	Center Point	82033.333333	83033.333333	84000.0
	Centreville	103233.333333	100300.000000	98500.0
	Chalkville	111833.333333	114400.000000	115700.0
	Chancellor	113300.000000	120333.333333	121500.0
	Chelsea	209200.000000	211400.000000	214200.0
	Chickasaw	57766.666667	57066.666667	56600.0
	Chunchula	122500.000000	123333.333333	124350.0
	Citronelle	93366.666667	93966.666667	93750.0
	Clay	143033.333333	144500.000000	145850.0
	Coden	81100.000000	82033.333333	80550.0
	Coker	150966.666667	147200.000000	143550.0
	Concord	112333.333333	111933.333333	113850.0
	Cottondale	134500.000000	134033.333333	133550.0

Wisconsin	Vernon	264000.000000	268166.666667	274100.0
	Vienna	284133.333333	285800.000000	289550.0
	Vinland	183133.333333	186500.000000	189750.0
	Wales	268433.333333	275666.666667	282400.0
	Waterford	230533.333333	234833.333333	237600.0
	Waukesha	191133.333333	193966.666667	199000.0
	Waunakee	299366.666667	303133.333333	308100.0

	Waupun	110766.666667	112600.000000	112750.0
	Wausau	93933.333333	96833.333333	98200.0
	Wayne	261866.666667	267133.333333	272250.0
	West Allis	100900.000000	97900.000000	95850.0
	Weston	140166.666667	142233.333333	144800.0
	Wheatland	182233.333333	189000.000000	194500.0
	Whitelaw	117233.333333	117400.000000	117600.0
	Williams Bay	231566.666667	248400.000000	255150.0
	Wilson	189066.666667	197266.666667	206200.0
	Wilson	170466.666667	174866.666667	178350.0
	Wind Lake	240833.333333	243866.666667	248850.0
	Wind Point	254400.000000	259500.000000	258450.0
	Wisconsin Dells	139766.666667	141000.000000	143000.0
	Wolf River	159500.000000	160933.333333	168050.0
	Woodruff	162533.333333	161966.666667	164200.0
	Woodville	182633.333333	186666.666667	193100.0
	Yorkville	248466.666667	255200.000000	265950.0
Wyoming	Bar Nunn	210200.000000	208833.333333	207450.0
	Burns	168266.666667	171600.000000	170500.0
	Casper	184333.333333	185166.666667	184350.0
	Cheyenne	194433.333333	196500.000000	199100.0
	Evansville	309433.333333	309200.000000	309050.0
	Pine Bluffs	166833.333333	163800.000000	157650.0

[10730 rows x 67 columns]

```
In [18]: 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 equivalent to a
    reduced market loss).'''
```

```
    univ_towns = get_list_of_university_towns()
    univ_towns['univ_town'] = True
```

```
    # merge the housing data with university town DataFrames
```

```
    df = pd.merge(convert_housing_data_to_quarters(), univ_towns, how='outer', left_ind
    df['univ_town'] = df['univ_town'].replace({np.NaN: False})
```



```

# Get the recession quarters
recession_start = get_recession_start()
recession_bottom = get_recession_bottom()

# Parse the year and quarter of the recession quarters
year_recession_start = int(recession_start[0:4])
qtr_recession_start = int(recession_start[-1])
year_recession_bottom = int(recession_bottom[0:4])
qtr_recession_bottom = int(recession_bottom[-1])

# get the columns to keep in the merged DataFrame
cols_to_keep = ['State', 'RegionName', 'univ_town']
qtrs_to_keep = []
for i in range(year_recession_start, year_recession_bottom+1):
    for j in range(1, 5):
        if (i == year_recession_start and j < qtr_recession_start)\
            or (i == year_recession_bottom and j > qtr_recession_bottom):
            pass
        else:
            qtrs_to_keep.append(str(i) + 'q' + str(j))
df = df[cols_to_keep + qtrs_to_keep]

# Compute the price_ratio
df['price_ratio'] = df[recession_bottom] - df[recession_start]

# t-test to determine if there is a difference between university and non-university towns
univ_town_price_ratio = df[df['univ_town'] == True]['price_ratio']
non_univ_town_price_ratio = df[df['univ_town'] == False]['price_ratio']
st, p = ttest_ind(univ_town_price_ratio, non_univ_town_price_ratio, nan_policy='omit')

# get different and better values
different = False
if p < 0.01:
    different = True

# determine which type of town is better
better = ""
if univ_town_price_ratio.mean() > non_univ_town_price_ratio.mean():
    better = "university town"
else:
    better = "non-university town"

return (different, p, better)

```

```
run_ttest()
```

```
Out[18]: (True, 0.0043252148534599624, 'university town')
```

```
In [ ]:
```