AML-2203 Advanced Python AI and ML Tools

Assignment Report

Instructor: Vahid Hadavi, PhD, P. Eng

Group Members:

Bhanu Prakash Mahadevuni C0850515

Deeksha Naikap C0835440

Pramod Reddy Gurrala C0850493

Sai Varun Kollipara C0828403

Date: 07-07-2022

Problem Statement:

- In this assignment, we are tasked to identify some scenarios like when the recession starts, when it ends and when it hits the bottom. For which some definitions are provided, along with we need to prove some hypothesis based on the functions provided.
- We need to calculate the house prices in the university located towns, in the time of the recession starts and ends.
- All the information regarding the definitions and hypothesis are mentioned below:

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.

Provided Definition

Hypothesis: University towns have their mean housing prices less effected by recessions.Perfome the following tasks:

- Download the data and run a completed set of exploratory data analysis including best possible tasks on that. This may include but not limited to observing the shape, describe the data set, checking the missing values, drawing the profile of the data, checking the distribution type and data types and several other tasks.

- 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`)

Hypothesis to be Calculated

- To perform all the above tasks three datasets are provided, that are listed here:
    - Zillow house pricing data for USA
    - University location Data with respect to the states in USA
    - GDP data Quarterly wise for the USA

Steps for Implementation:

- Using the dataset information, definitions, and hypothesis to be calculated we have divided the assignment into multiple phases.
    1. Data Collection
        a) Zillow CSV data (using with a data frame directly)
        b) University Location Text data (Cleaning the data and converting the text to data frame)

c) GDP XLS data (cleaning the dataset and utilizing the required information from year 2000)

2. Exploratory Data Analysis: Here are some basic functions that have been used:
    a) Viewing the data – head(), tail() and sample()
    b) Shape of the data frame
    c) Central Tendency – describe()
    d) Information – info() and dtypes()
    e) Checking null values – isnull() and isna()
3. Definitions:
    a) Recession Start
    b) Recession End
    c) Recession Bottom
    d) Converting the Zillow housing data from yearly to quarterly basics
    e) Calculate the price ratio
    f) Differentiate weather University Town or not
    g) Perform TTest
4. Data Visualization
    a) Plotting graph to compare the results
5. TTest

Methodology:

- For the first step we need to collect the data and store in the form of data frames, for all the three datasets, we need to perform the same task.
- Since the Zillow data is originally CSV data, we can use pandas and label it as data frame. But for the other datasets, we need to build some methodologies that will convert them into data frames.
- For the University Location Data, which is a text file, we can convert them into lines first. Then use the key variables and divide them into States and Regions. Using those as the base to the data frame, we can build a university location data frame.

```
for line in lines:
    if '[edit]' in line:
        state_column = line.replace('[edit]', '')
    else:
        region_column = re.sub(r' \(.*', '', line)
        df_uni_result.loc[i] = [state_column, region_column]
        i = i + 1
```

Cleaning Method for University Locations Data

- Now for the GDP data, the given condition is to consider the data form 2000 onwards and only consider the GDP calculation based on the 2009-dollar value. We have built some conditions that can meet the required data frame.

```
df_gdp_upd = pd.read_excel('gdplev.xls', skiprows=220, usecols='E,G', header=None)
df_gdp_upd.columns = ['Quarter', 'GDP']
```

Cleaning Method for GDP Data

- Since all the data frames are ready, we have performed some basic exploratory data analysis.
- After that we have worked on building the functions based on the definitions provided. All the functions and the working are provided below.
- Recession Start: As per the definition when two continuous quarter has fall in the GDP it is considered as a recession. But the quarter at which the fall happen is called the recession start. We build a function accordingly that can compare the GDP of all the rows with the continuous ones. Then we have added used the indicator to identify the quarter and return its value.

```
def recession_start():
    i=2
    l = len(df_gdp_upd['GDP'])
    ress_s = ''

    while i<l:
        if df_gdp_upd.GDP[i]<df_gdp_upd.GDP[i-1] and df_gdp_upd.GDP[i-1]<df_gdp_upd.GDP[i-2]:
            ress_s = df_gdp_upd.Quarter[i-1]
            break
        i = i+1

    return ress_s
```

```
recession_start()
```

```
'2008q3'
```

Recession Start Calculation

- Recession End: We have implemented the same logic where two future continuous quarters have increasing value and two pasts continuous quarters have decreasing value. We build a function accordingly that can compare the GDP of all the rows with the continuous ones. Then we have added used the indicator to identify the quarter and return its value.

```
def recession_end():

    ress_e = ''
    l = len(df_gdp_upd['GDP'])

    for i in range(4, l):
        if (df_gdp_upd.GDP[i - 4] > df_gdp_upd.GDP[i - 3]) and (
                df_gdp_upd.GDP[i - 3] > df_gdp_upd.GDP[i - 2]) and (
                df_gdp_upd.GDP[i - 2] < df_gdp_upd.GDP[i - 1]) and (
                df_gdp_upd.GDP[i - 1] < df_gdp_upd.GDP[i]):
            recess_ends = i
    ress_e = df_gdp_upd.Quarter[recess_ends]

    return ress_e
```

```
recession_end()
```

```
'2009q4'
```

Recession End Calculation

- Recession Bottom: We have implemented the same logic where two future continuous quarters have increasing value and two pasts continuous quarters have decreasing value. The logic is same as the previous one, but while considering the indicator we have taken the one where the GDP has the lowest, since we need the Quarter where it hit the bottom. Then returned the quarter where the indicator is pointed to.

```
def recession_bottom():

    l = len(df_gdp_upd['GDP'])
    ress_b = ''

    for i in range(4,l):
        if (df_gdp_upd.GDP[i-4] > df_gdp_upd.GDP[i-3]) and (
                df_gdp_upd.GDP[i-3] > df_gdp_upd.GDP[i-2]) and (
                df_gdp_upd.GDP[i-2] < df_gdp_upd.GDP[i-1]) and (
                df_gdp_upd.GDP[i-1] < df_gdp_upd.GDP[i]):
            recess_bottom = i-2
            ress_b = df_gdp_upd.Quarter[recess_bottom]

    return ress_b
```

```
recession_bottom()
```

```
'2009q2'
```

Recession Bottom Calculation

- Zillow Housing data Function: In this function we can take the original Zillow data frame as input and clean the data and perform some analysis on that. Here is the list of operations that we will be performing:
    - Dropping the Unwanted Columns
    - Using the university data frame's State list
    - Only using the data from 2000
    - Converting the year - month format to year-month-date
    - Converting the Date to Quarter based using the mean function

```
def house_info_to_quarters(zillo_df):

    # Adding the zillow data frame to a temperary data frame
    df = zillo_df
    print("Original Zillow Data")
    print(df.head())

    # drop the unnecessary columns
    df = df.drop(['RegionID', 'Metro', 'CountyName', 'SizeRank'], axis=1)
    print("After dropping the Unnecessory Colomns")
    print(df.head())

    # replacing the State column values with respective unoversity states locations dictionary values.
    df['State'] = df['State'].replace(uni_states)

    df = df.set_index(['State', 'RegionName'])


    # delete the columns from the year 2000
    for col in df.columns:
        if '199' in col:
            del df[col]


    # year to datetime with format YY-mm
    df.columns = pd.to_datetime(df.columns, format='%Y-%m')
    print("Conversion on time  format for future processing and using the data from 2000 onwards")
    print(df.head())


    # Adding the quater-wise frequency
    df = df.resample('Q', axis=1).mean()

    # rename the columns as per quarter numbers in lowercase
    df = df.rename(columns=lambda x: str(x.to_period('Q')).lower())
    print("After aAdding the Quaters with respect to the months")
    print(df.head())

    return df
```

Method for Housing yearly data into Quarters based data

- As per the hypothesis to perform the TTest, we need the price ratio, we have just implemented the formula.

(price_ratio=quarter_before_recession/recession_bottom)

```
housedata['ratio'] =  (housedata[bef_ress] - housedata[bottom])/ housedata[bef_ress]
housedata['ratio']
```

Price Ratio Calculation

- Now we are merging the university towns information and housing towns information, we also created a new label to represent whether the town has university or not(Boolean Value).

```
unitowns_hdata['uni_tows'] = True

house_data_upd = pd.merge(housedata, unitowns_hdata, how='outer', on=['State','RegionName',bottom, bef_ress, 'ratio'])
house_data_upd['uni_tows'] = house_data_upd['uni_tows'].fillna(False)

uni_tow = house_data_upd[house_data_upd['uni_tows'] == True]
nonuni_tow = house_data_upd[house_data_upd['uni_tows'] == False]
```

University Town identification method

- Now we are performing the TTest on the sorted data frame, with quarters from recession and price ratio of the location and university town info.

```
def ttest():

    t_val,p_val = ttest_ind(uni_tow['ratio'].dropna(), nonuni_tow['ratio'].dropna())
    print('P_value and T_value')
    print(t_val,p_val)

    different = True if p_val<0.01 else False
    better = "university town" if uni_tow['ratio'].mean() < nonuni_tow['ratio'].mean() else "non-university town"

    return (p_val,different,better)
```

TTest Method

Results:

- First the results related to the data frame are showcased here.

```
zillo_df
```

	RegionID	RegionName	State	Metro	CountyName	SizeRank	1996-04	1996-05	1996-06	1996-07	...	2015-11	2015-12	2016-01	2016-02	2016-03	2016-04	2016-05	2016-06	2016-07	2016-08
0	6181	New York	NY	New York	Queens	1	NaN	NaN	NaN	NaN	...	573600	576200	578400	582200	588000	592200	592500	590200	588000	586400
1	12447	Los Angeles	CA	Los Angeles-Long Beach-Anaheim	Los Angeles	2	155000.0	154600.0	154400.0	154200.0	...	558200	560800	562800	565600	569700	574000	577800	580600	583000	585100
2	17426	Chicago	IL	Chicago	Cook	3	109700.0	109400.0	109300.0	109300.0	...	207800	206900	206200	205800	206200	207300	208200	209100	211000	213000
3	13271	Philadelphia	PA	Philadelphia	Philadelphia	4	50000.0	49900.0	49600.0	49400.0	...	122300	121600	121800	123300	125200	126400	127000	127400	128300	129100
4	40326	Phoenix	AZ	Phoenix	Maricopa	5	87200.0	87700.0	88200.0	88400.0	...	183800	185300	186600	188000	189100	190200	191300	192800	194500	195900
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
10725	398292	Town of Wrightstown	WI	Green Bay	Brown	10726	NaN	NaN	NaN	NaN	...	149900	150100	150300	150000	149200	149900	151400	152500	154100	155900
10726	398343	Urbana	NY	Corning	Steuben	10727	66900.0	65800.0	65500.0	65100.0	...	135700	136400	137700	138700	140500	143600	145000	144000	143000	143000
10727	398496	New Denmark	WI	Green Bay	Brown	10728	NaN	NaN	NaN	NaN	...	188700	189800	190800	191200	191200	191700	192800	194000	196300	198900
10728	398839	Angels	CA	NaN	Calaveras	10729	115600.0	116400.0	118000.0	119000.0	...	280400	279600	278000	276600	275000	273700	272000	269100	269000	270900
10729	399114	Holland	WI	Sheboygan	Sheboygan	10730	129900.0	130200.0	130300.0	129100.0	...	217800	219400	221100	222000	222800	224900	228000	231200	233900	236000

Original Zillow Data frame

```
df_uni_result
```

	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

Original University Data Frame

```
df_gdp_upd
```

	Quarter	GDP
0	2000q1	12359.1
1	2000q2	12592.5
2	2000q3	12607.7
3	2000q4	12679.3
4	2001q1	12643.3
...	...	...
61	2015q2	16374.2
62	2015q3	16454.9
63	2015q4	16490.7
64	2016q1	16525.0
65	2016q2	16583.1

Original GDP Data frame

- Here is the result from the Zillow Housing data, after converting from Year to Quarter based.

```
final_uni_zillow_df
```

State	RegionName	2000q1	2000q2	2000q3	2000q4	2001q1	2001q2	2001q3	2001q4	2002q1	2002q2	...	2014q2	2014q3	2014q4
New York	New York	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	515466.666667	522800.000000	528066.666667
California	Los Angeles	207066.666667	214466.666667	220966.666667	226166.666667	233000.000000	239100.000000	245066.666667	253033.333333	261966.666667	272700.000000	...	498033.333333	509066.666667	518866.666667
Illinois	Chicago	138400.000000	143633.333333	147866.666667	152133.333333	156933.333333	161800.000000	166400.000000	170433.333333	175500.000000	177566.666667	...	192633.333333	195766.666667	201266.666667
Pennsylvania	Philadelphia	53000.000000	53633.333333	54133.333333	54700.000000	55333.333333	55533.333333	56266.666667	57533.333333	59133.333333	60733.333333	...	113733.333333	115300.000000	115666.666667
Arizona	Phoenix	111833.333333	114366.666667	116000.000000	117400.000000	119600.000000	121566.666667	122700.000000	124300.000000	126533.333333	128366.666667	...	164266.666667	165366.666667	168500.000000
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
Wisconsin	Town of Wrightstown	101766.666667	105400.000000	111366.666667	114866.666667	125966.666667	129900.000000	129900.000000	129433.333333	131900.000000	134200.000000	...	144866.666667	146866.666667	149233.333333
New York	Urbana	79200.000000	81666.666667	91700.000000	98366.666667	94866.666667	98533.333333	102966.666667	98033.333333	93966.666667	94600.000000	...	132133.333333	137033.333333	140066.666667
Wisconsin	New Denmark	114566.666667	119266.666667	126066.666667	131966.666667	143800.000000	146966.666667	148366.666667	149166.666667	153133.333333	156733.333333	...	174566.666667	181166.666667	186166.666667
California	Angels	151000.000000	155900.000000	158100.000000	167466.666667	176833.333333	183766.666667	190233.333333	184566.666667	184033.333333	186133.333333	...	244466.666667	254066.666667	259933.333333
Wisconsin	Holland	151033.333333	150500.000000	153233.333333	155833.333333	161866.666667	165733.333333	168033.333333	167400.000000	165766.666667	161966.666667	...	201266.666667	201566.666667	201266.666667

10730 rows × 67 columns

Finalized Zillow data wr.t Quarters

- Here is the result of the data frame after filtering it with price ratio info, recession start and bottom info.

	State	RegionName	2009q2	2008q2	ratio
0	Nevada	Las Vegas	164333.333333	232300.000000	0.292581
1	California	San Diego	389500.000000	441400.000000	0.117580
2	Texas	Dallas	105100.000000	115366.666667	0.088992
3	Texas	Austin	204000.000000	213733.333333	0.045540
4	Ohio	Columbus	109766.666667	113500.000000	0.032893
5	Tennessee	Memphis	76000.000000	84400.000000	0.099526
6	Massachusetts	Boston	325200.000000	347966.666667	0.065428
7	Tennessee	Nashville	150133.333333	154600.000000	0.028892

Data Frame with the university locations and price ratio

- Now, the finalized data frame that can be used for the performing TTest is showed below.

```
house_data_upd.head(12)
```

	State	RegionName	2009q2	2008q2	ratio	uni_tows
0	New York	New York	465833.333333	503933.333333	0.075605	False
1	California	Los Angeles	413900.000000	502266.666667	0.175936	False
2	Illinois	Chicago	219700.000000	237900.000000	0.076503	False
3	Pennsylvania	Philadelphia	116166.666667	118133.333333	0.016648	False
4	Arizona	Phoenix	168233.333333	205266.666667	0.180416	False
5	Nevada	Las Vegas	164333.333333	232300.000000	0.292581	True
6	California	San Diego	389500.000000	441400.000000	0.117580	True
7	Texas	Dallas	105100.000000	115366.666667	0.088992	True
8	California	San Jose	530300.000000	611733.333333	0.133119	False
9	Florida	Jacksonville	140833.333333	160433.333333	0.122169	False
10	California	San Francisco	695000.000000	783633.333333	0.113106	False
11	Texas	Austin	204000.000000	213733.333333	0.045540	True

Finalized University Townhouse Data Frame

- Here are the results from the TTest function.

```
ttest()
```

```
P_value and T_value
-2.933170587337901 0.0033629228768551505
(0.0033629228768551505, True, 'university town')
```

TTest Results

What have we Learnt?

- We learnt how to handle data that are not only CSV files. In this assignment we have learnt how to handle the text files and excel files.
- We have identified that only performing Exploratory Analysis won't provide any insights, As the data will vary and cannot correlate multiple datasets.
- We have identified that basic numpy and pandas operations like splitting lines, dropping the columns, data frames merging and slicing the data frames can be more useful for gathering the insights.
- We have identified how to implement a regular formula in the code format while calculating the price ratio.
- We have also learnt how to build functions effectively and use them in multiple cases.

References:

1. https://www.scribbr.com/statistics/t-test/#:~:text=What%20is%20a%20t%2Dtest,means%20is%20different%20from%20zero.
2. https://www.scribbr.com/statistics/p-value/
3. https://medium.com/mlearning-ai/forecasting-recessions-with-scikit-learn-df58e1ea695f
4. https://towardsdatascience.com/exploratory-data-analysis-in-python-c9a77dfa39ce