Postgraduate Certificate in Cloud Native Computing
Postgraduate Certificate in Software Design with Artificial Intelligence

Applied Scripting Languages

Assignment 1

Student ID: A00277179

Brief Description: Analysis of Properties for Sale in Ireland on daft.ie

Contents

Introduction

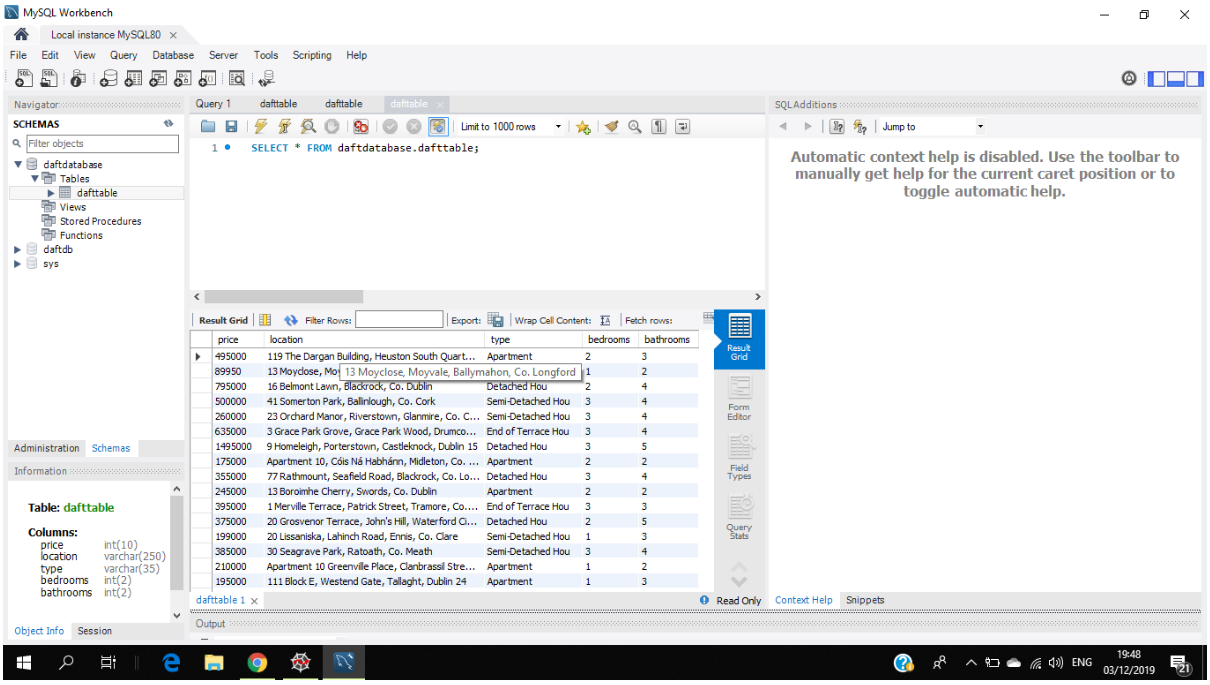
In this project the goal was to gather data and analyse, specifically I wanted to gather data on house prices in Ireland and compare them. The data on houses was obtained by web-scraping data from a website called daft.ie. The data that was gathered included house price, house type, house location, number of bedrooms and number of bathrooms. Once this data is gathered it will be put into a database, MySQL, and stored in a table.

The next step after gathering the data will be to analyse it. The mean, mode and median of different variables such as house price and number of bedrooms will be calculated. Correlations between different variables will be calculated and finally visualisations will be created. Charts and Graphs of house price and location will be created. Scatter graphs of house price and number of bedrooms will be created. Pie charts showing the number of houses in each county will be created.

Data

The data from this project was sourced from daft.ie. The data was web scraped using the BeautifulSoup module and compiled into separate lists. While the data was being webscraped variables such as price had the '€' symbol removed and made into an integer. The data that was webscraped was on houses for sale on daft.ie. It included house price, location of the house, the type of house it was, how many bedrooms and how many bathrooms it contained. This data was then put into a csv file and finally into a database in MySQL in the form of a table with column names : Price, Location, Type, Bedrooms, Bathrooms. It contains 931 rows of information on house prices on daft.ie and can be extended to as many webpages as are on the website.

I chose this data because I want to see if there was any correlation between house prices and house location within in Ireland. I also wanted to see if the number of bedrooms and the number of bathrooms had any correlation within house in Ireland. This data would be useful for any real estate company looking for houses below a certain price in a certain area within Ireland. For the fact that it can scrape through the data on the webpage everyday and be altered to give notification on any houses below a certain price in certain areas when updated.



```
SELECT * FROM daftdatabase.dafttable;
```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

price	location	type	bedrooms	bathrooms
495000	119 The Dargan Building, Heuston South Quart...	Apartment	2	3
89950	13 Moyclose, Moy 13 Moyclose, Moyvale, Ballymahon, Co. Longford		1	2
795000	16 Belmont Lawn, Blackrock, Co. Dublin	Detached Hou	2	4
500000	41 Somerton Park, Ballinlough, Co. Cork	Semi-Detached Hou	3	4
260000	23 Orchard Manor, Riverstown, Glanmire, Co. C...	Semi-Detached Hou	3	4
635000	3 Grace Park Grove, Grace Park Wood, Drumco...	End of Terrace Hou	3	4
1495000	9 Homeleigh, Porterstown, Castleknock, Dublin 15	Detached Hou	3	5
175000	Apartment 10, Cóis Ná Habhánn, Midleton, Co. ...	Apartment	2	2
355000	77 Rathmount, Seafield Road, Blackrock, Co. Lo...	Detached Hou	3	4
245000	13 Boroimhe Cherry, Swords, Co. Dublin	Apartment	2	2
395000	1 Merville Terrace, Patrick Street, Tramore, Co....	End of Terrace Hou	3	3
375000	20 Grosvenor Terrace, John's Hill, Waterford Ci...	Detached Hou	2	5
199000	20 Lissaniska, Lahinch Road, Ennis, Co. Clare	Semi-Detached Hou	1	3
385000	30 Seagrave Park, Ratoath, Co. Meath	Semi-Detached Hou	3	4
210000	Apartment 10 Greenville Place, Clanbrassil Stre...	Apartment	1	2
195000	111 Block E, Westend Gate, Tallaght, Dublin 24	Apartment	1	3

Table: dafttable

Columns:

price	int(10)
location	varchar(250)
type	varchar(35)
bedrooms	int(2)
bathrooms	int(2)

Design

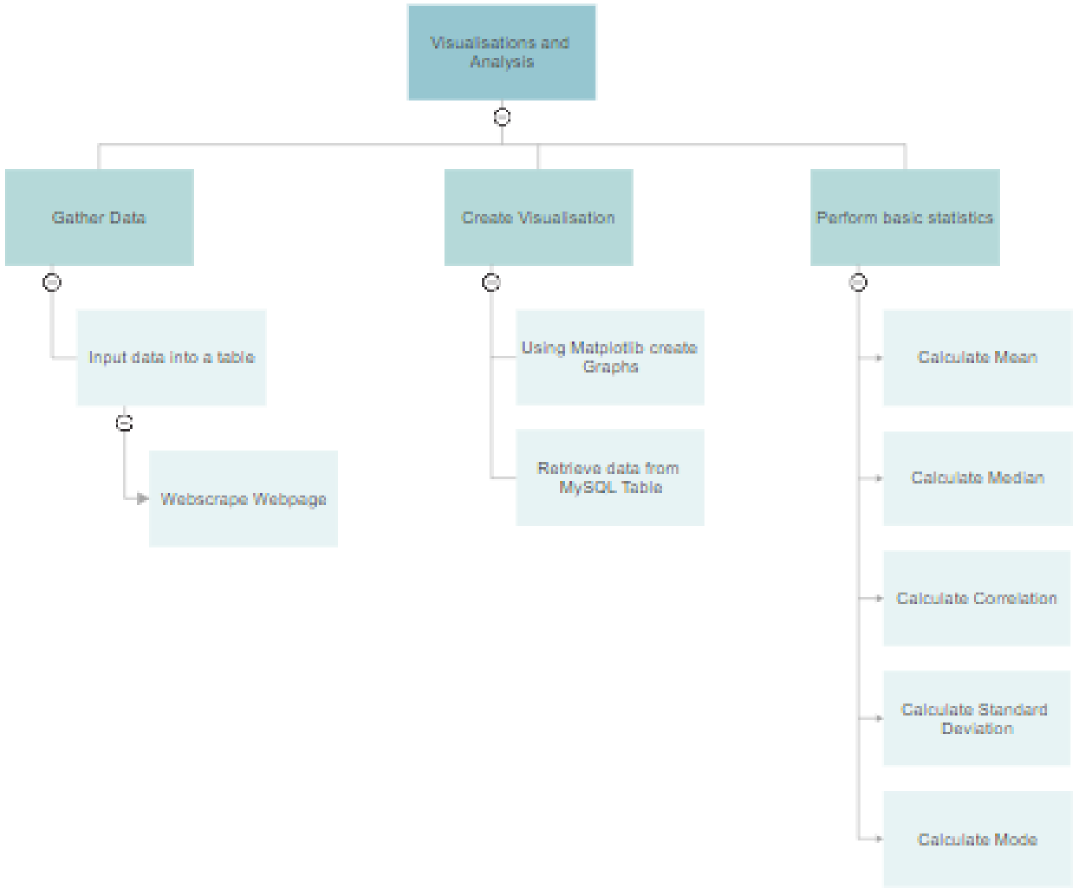
Analyse Problem:

Webscrape data from a website, input it into a MySQL table and perform analysis on the data gathered.

Specificiation Table:

<u>**Input**</u>	<u>**Process**</u>	<u>**Output**</u>
Website URL	Gather Data and input it into lists Store Data in a MySQL database Retrieve data from MySQL database Calculate mean, mode, median, standard deviation, correlation. Graph data.	Visualisations and Basic Statistics

<u>Top Down Design</u>



Modular Development:

Module: Create Statistics

Pseudo Code:

```
Retrieve data from my SQL
Perform Calculations
Store Results
```

Input	**Process**	**Output**
Input List	Input list Try return sum of list/length of list Otherwise return none	Average of values in list
Input List	Input list Try to return mode of values in list	mode of values of list
Input List	Input list Try to return median of values in list	Median of values in lists
Input List	Input list Try to Return standard deviations of list	Standard deviation of values of list
Input List	Input two lists Try to calculate mean of values in list. Return correlation between two values in list	Correlation of two different lists

Module: Create Visulisations

PseudoCode:

```
Retrieve data from MySQL
Create Visualisation
Save Visualisation
```

Input	**Process**	**Output**

Retrieve data from my SQL	Input data into list fig, ax = plt.subplots() fig.savefig("barhousetype.png", bbox_inches = "tight")	Create Visualisation

Module: Gather data

Pseudo Code:

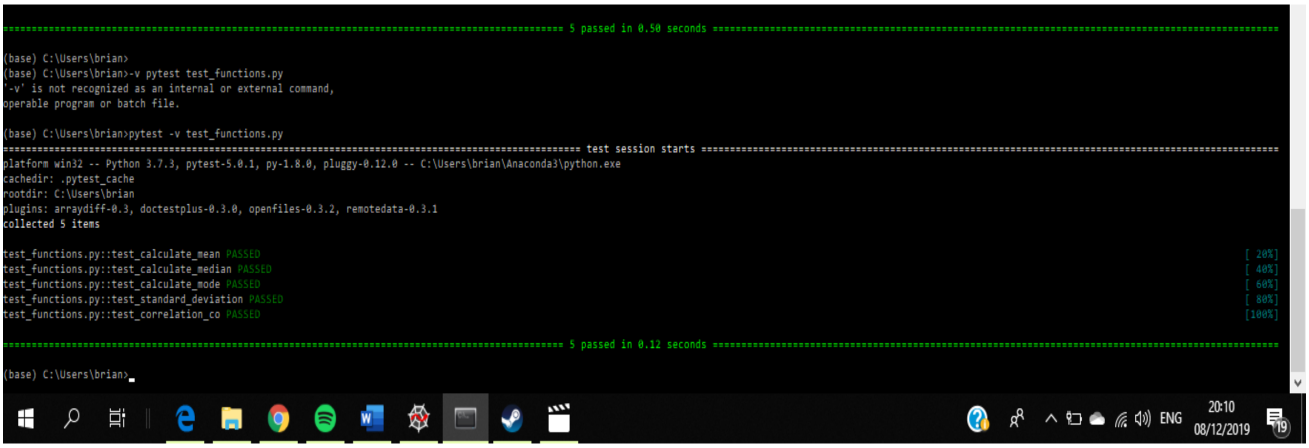
```
Insert webpage URL
Open up website to webscrape
Gather desired data
Input data in database
```

Input	**Process**	**Output**
Webpage url	Open MySQL database Gather Chosen data Input data in database table	Database Table

Mainline Algorithm:

```
Input Webpage URL
Gather Data
Create Visualisation
Create Statistics
```

Testing



```
=============================================== 5 passed in 0.50 seconds ===============================================

(base) C:\Users\brian>
(base) C:\Users\brian>-v pytest test_functions.py
'-v' is not recognized as an internal or external command,
operable program or batch file.

(base) C:\Users\brian>pytest -v test_functions.py
================================================ test session starts ================================================
platform win32 -- Python 3.7.3, pytest-5.0.1, py-1.8.0, pluggy-0.12.0 -- C:\Users\brian\Anaconda3\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\brian
plugins: arraydiff-0.3, doctestplus-0.3.0, openfiles-0.3.2, remotedata-0.3.1
collected 5 items

test_functions.py::test_calculate_mean PASSED                                                                    [ 20%]
test_functions.py::test_calculate_median PASSED                                                                  [ 40%]
test_functions.py::test_calculate_mode PASSED                                                                    [ 60%]
test_functions.py::test_standard_deviation PASSED                                                                [ 80%]
test_functions.py::test_correlation_co PASSED                                                                    [100%]

=============================================== 5 passed in 0.12 seconds ===============================================

(base) C:\Users\brian>_
```

The functions created to calculate mean, mode, median, standard deviation and correlation were all tested and passed.

Analysis and Visualisation

For the analysis of my data I did some basic statistics first calculating the mean, mode, median and standard deviation.

	House Price	Number of Bedrooms	Number of Bathrooms
Mean	561052.63	3.8	2.8
Mode	575000	5	3
Median	240000	4	2
Standard Deviation	338477.0102595598	1.1513138087136294	1.1962022530578604

Calculations were performed for the correlation between house price and number of bedrooms using the peason's product moment correlation which returned a value of: 0.6563097397968524. Suggesting there is a strong positive correlation between these two variables.

Using Matplotlib, several different visualisations were created. As seen in figure 1, the number of detached houses is far greater than the number of apartments for sale. Looking at the graph, there are over 500 detached houses for sale in the data that was gathered in comparison to only over 100 semi-detached houses and less than 100 townhouses, apartments, bungalows and terraced houses.

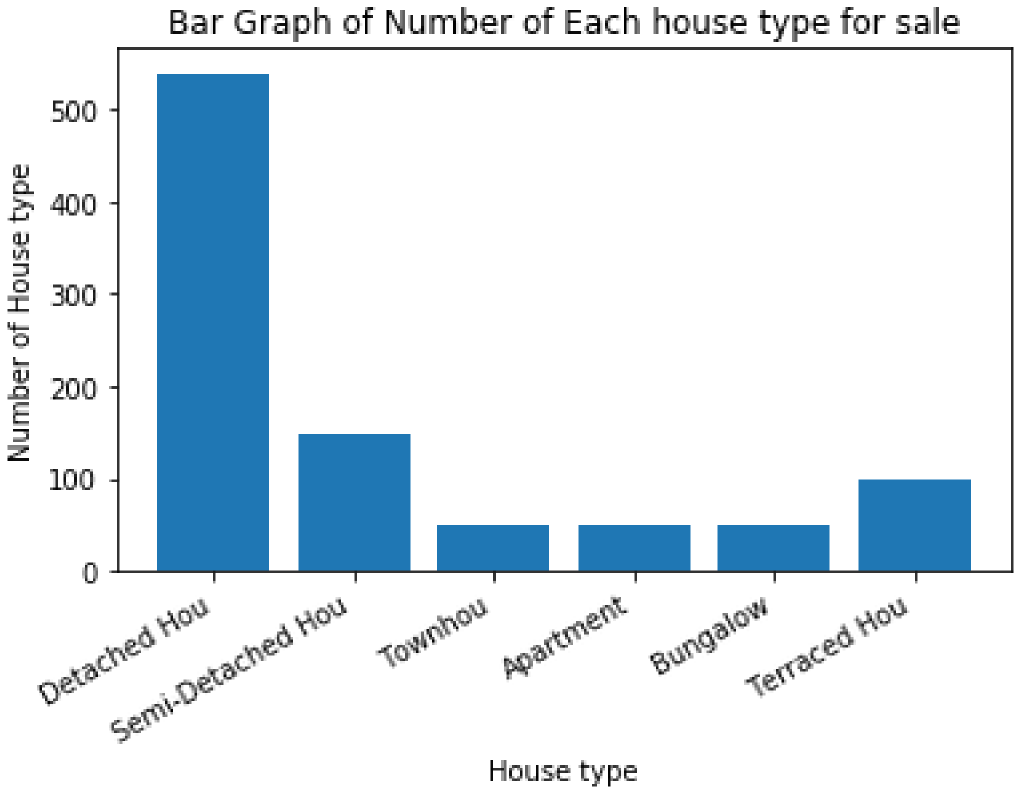


Figure 1 Bar Graph Of Number of Each House Type For Sale On Daft.ie

In figure 2 there is a relative correlation seen in the scatter graph between the price of houses and number of bedrooms. What is suggested is that as the number of bedrooms increase the house price increases. With the few exceptions on the lower side graph, but what is interesting is there are no outliars above where a line could be drawn from point (1,250000) to point (5,1400000). Suggesting that you the house price is effected by the number of bedrooms.

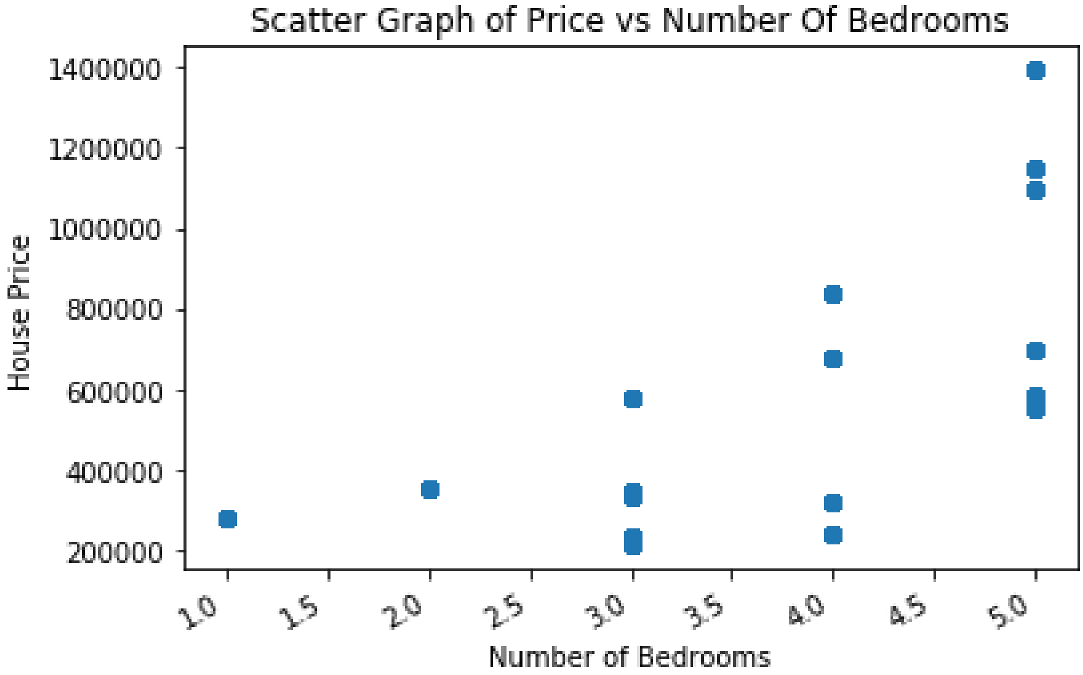


Figure 2 Scatter Graph of Price and Number of Bedrooms

Looking at the Pie chart below, figure 3, we can see that Cork, Galway and Dublin have the most houses for sale. If we combined the Dublin post codes it would have the greatest number of houses for sale. It can also be noted that not every county has a house for sale, this could be due to multiple factors, not webscraping enough pages to generate a match, smaller counties may not have houses for sale and the house for may not be for sale on Daft.ie.

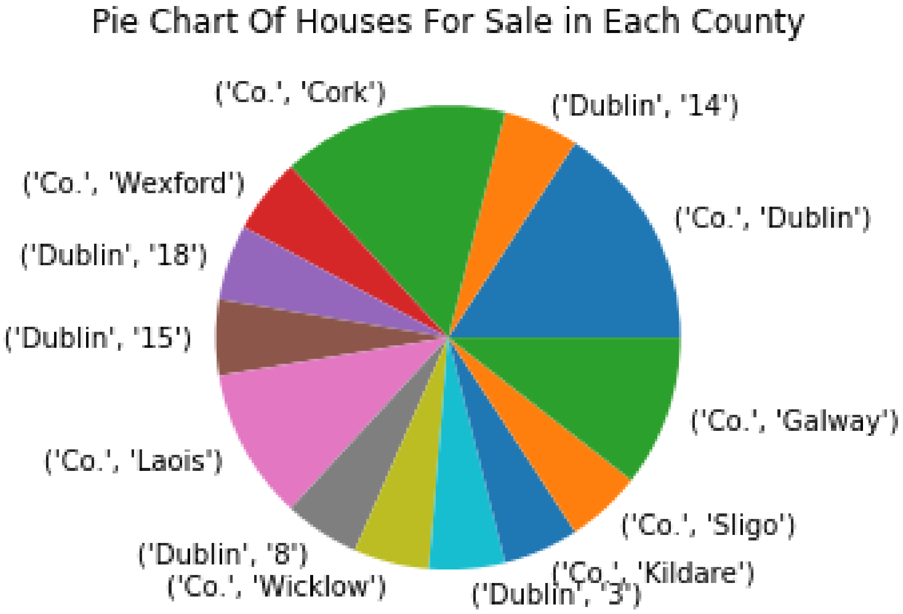


Figure 3Pie Chart Of House Location Based On County and Dublin Post Code vs Number of Houses

Conclusion

In conclusion with the analyse of this project, it can be clearly seen that Dublin has a higher proportion of houses for sale. The average house price in Ireland for houses on sale on Daft.ie is 561052.63. The average number of bedrooms and bathrooms were 3.8 and 2.8 respectively. The median stands at 240000, 4 and 2 for house price, bedrooms and bathrooms respectively. While the mode for house price, number of bathrooms and number of bedrooms was 575000, 3 and 5 resepectively. The Standard deviation’s were roughly 338477, 1.15 and 1.196 for house price, number of bedrooms and number of bathrooms. It can also be seen that the price of houses and the number of bedrooms in each house are correlated, with a strong positive correlation between the two. Looking at our graphs it can be seen that that there is mostly detached houses for sale in Ireland with in excess of 500. From the scatter graph we can see that house price and number of bedrooms are related. The pie chart shows that Co.Dublin, Co. Cork and Co.Galway all have the highest amount of houses for sale. While this data does give us insight into the trends in house sales within Ireland on Daft.ie, it does not give us any conclusive evidence as there are many more pages to be webscraped.

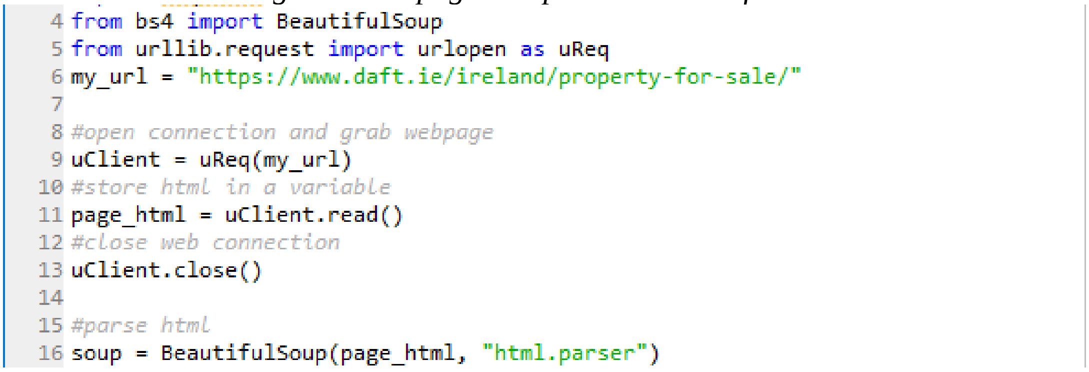
In Terms of my own experience in carrying out this project I found it highly beneficial. If time permitted I would have enjoyed learning more material then what was covered in this course. The scope for self-learning is really motivating and I find that it allows you to gather a deeper understanding of the functioning behind the code. In terms of the project outline, it is broad enough to allow you to explore your own interests within the code and really forces you to become more comfortable with the code. I feel that my coding has dramatically improved and my skills wit python has really developed by doing this project. I hope to in the future learn more analysis skills in terms of statistics and improve my modular development.

Appendix 1: Reflective Learning Log

Date: 12/11/2019

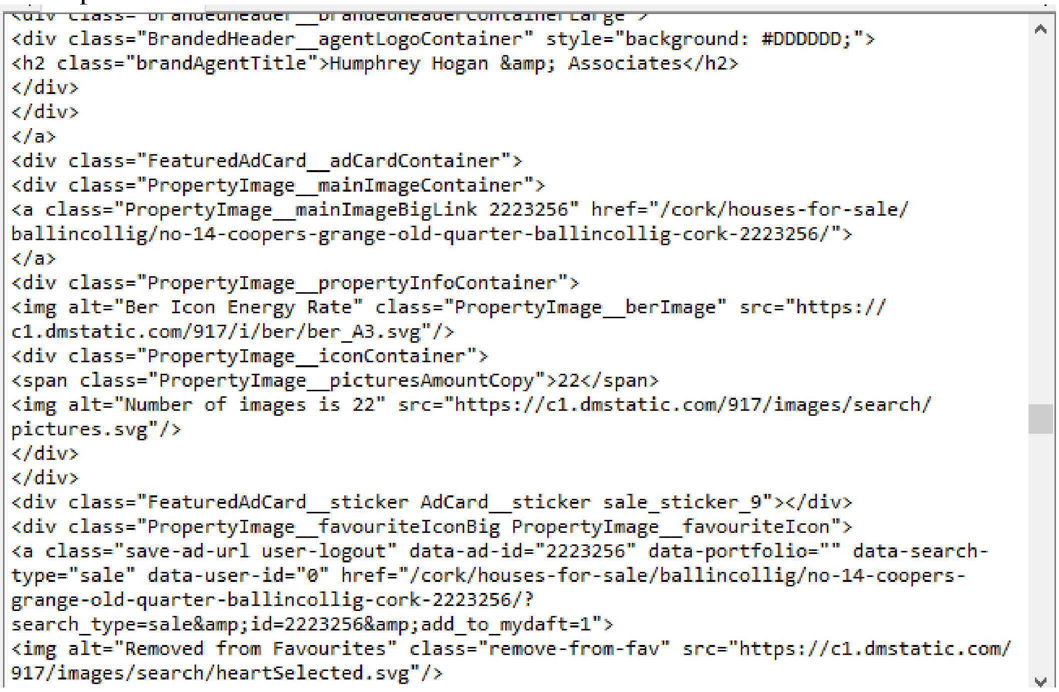
Work Completed
Webscraped data from daft.ie

Understanding Achieved:
Before starting any of this module I had virtually zero understanding of what code actually does and how it runs. The first step of the project was to find data to analyse. While there is many datasets out there and websites dedicated to presenting them. I heard about webscraping during one of the lectures and it sounded like a challenge. I've spent the previous few days(maybe weeks) trying to figure out how it works. Web scraping is the process of taking data from a website. This particularly useful for any business from finance to real estate. I decided I would take information from the website daft on house prices location and sizes. To begin I had to import functions called urllopen and beautiful soup. These allowed me to grab a webpage and pulled the data from the html.



```python
from bs4 import BeautifulSoup
from urllib.request import urlopen as uReq
my_url = "https://www.daft.ie/ireland/property-for-sale/"

#open connection and grab webpage
uClient = uReq(my_url)
#store html in a variable
page_html = uClient.read()
#close web connection
uClient.close()

#parse html
soup = BeautifulSoup(page_html, "html.parser")
```

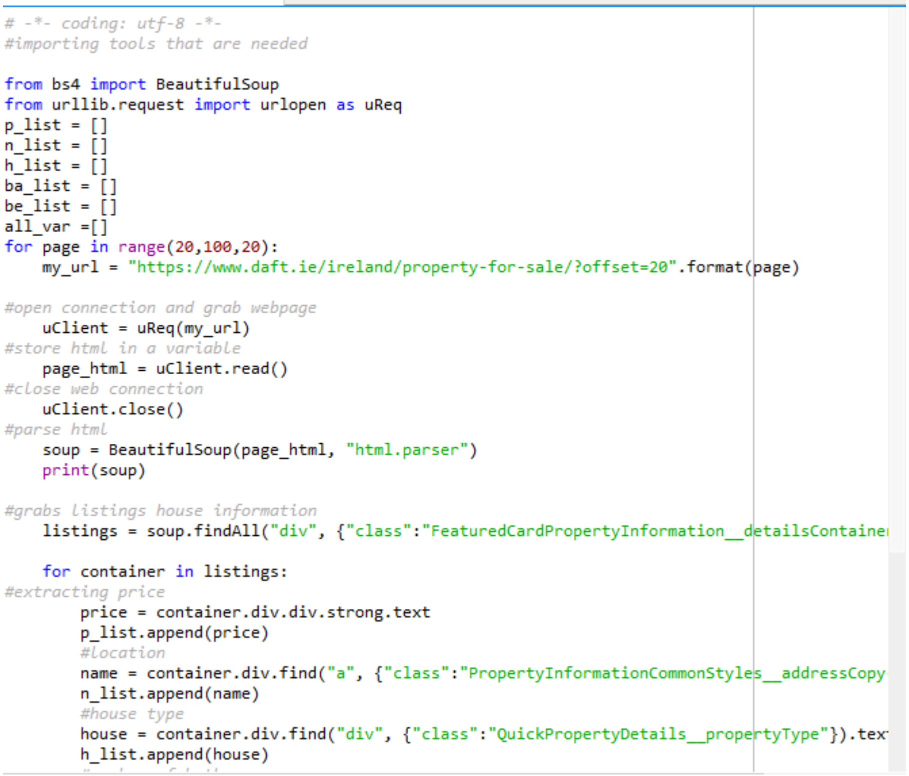
I then parsed the data to make the text more readable



```
<div class="BrandedHeader__brandedHeaderContainerLarge">
<div class="BrandedHeader__agentLogoContainer" style="background: #DDDDDD;">
<h2 class="brandAgentTitle">Humphrey Hogan &amp; Associates</h2>
</div>
</div>
</a>
<div class="FeaturedAdCard__adCardContainer">
<div class="PropertyImage__mainImageContainer">
<a class="PropertyImage__mainImageBigLink 2223256" href="/cork/houses-for-sale/
ballincollig/no-14-coopers-grange-old-quarter-ballincollig-cork-2223256/">
</a>
<div class="PropertyImage__propertyInfoContainer">
<img alt="Ber Icon Energy Rate" class="PropertyImage__berImage" src="https://
c1.dmstatic.com/917/i/ber/ber_A3.svg"/>
<div class="PropertyImage__iconContainer">
<span class="PropertyImage__picturesAmountCopy">22</span>
<img alt="Number of images is 22" src="https://c1.dmstatic.com/917/images/search/
pictures.svg"/>
</div>
</div>
<div class="FeaturedAdCard__sticker AdCard__sticker sale_sticker_9"></div>
<div class="PropertyImage__favouriteIconBig PropertyImage__favouriteIcon">
<a class="save-ad-url user-logout" data-ad-id="2223256" data-portfolio="" data-search-
type="sale" data-user-id="0" href="/cork/houses-for-sale/ballincollig/no-14-coopers-
grange-old-quarter-ballincollig-cork-2223256/?
search_type=sale&amp;id=2223256&amp;add_to_mydaft=1">
<img alt="Removed from Favourites" class="remove-from-fav" src="https://c1.dmstatic.com/
917/images/search/heartSelected.svg"/>
```

Since I had no experience in html I learned the basics in reading the text. Using the inspect key on the website I was able to identify which parts I wanted to scrape.

I then wrote the script ,with some help from tutorials , to find out the data that I wanted these included the house price, house type, location and number of bedrooms and bathrooms. Due to wanting to collect data on multiple different properties I had to implement for loops.



```
# -*- coding: utf-8 -*-
#importing tools that are needed

from bs4 import BeautifulSoup
from urllib.request import urlopen as uReq
p_list = []
n_list = []
h_list = []
ba_list = []
be_list = []
all_var =[]
for page in range(20,100,20):
    my_url = "https://www.daft.ie/ireland/property-for-sale/?offset=20".format(page)

#open connection and grab webpage
    uClient = uReq(my_url)
#store html in a variable
    page_html = uClient.read()
#close web connection
    uClient.close()
#parse html
    soup = BeautifulSoup(page_html, "html.parser")
    print(soup)

#grabs listings house information
    listings = soup.findAll("div", {"class":"FeaturedCardPropertyInformation__detailsContaine

    for container in listings:
#extracting price
        price = container.div.div.strong.text
        p_list.append(price)
        #location
        name = container.div.find("a", {"class":"PropertyInformationCommonStyles__addressCopy
        n_list.append(name)
        #house type
        house = container.div.find("div", {"class":"QuickPropertyDetails__propertyType"}).tex
        h_list.append(house)
```

The for loops here allow the url html to be grabbed from any url between 20 to 600 with a 20 step. The following for loop then allows the html to be scrapped multiple times for each of the data points I require.

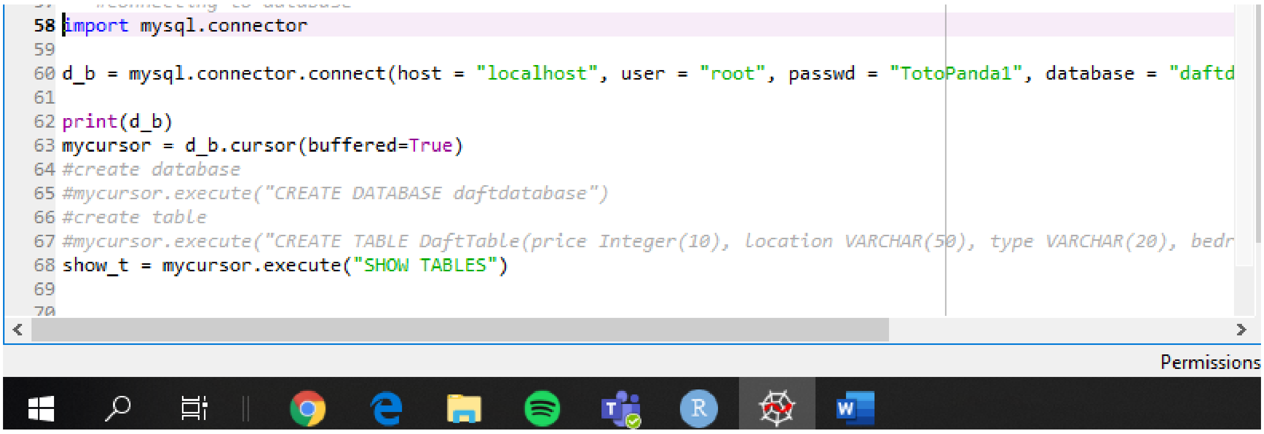
After this I know I would be performing different calculations on the values and creating different graphs, therefore I converted all my numerical values into integers by using the .strip and .replace to first take away symbols then the int() to convert from strings to integers. The next step I must take is to create a database which will store all of this webscraped data.

Date: 13/11/2019

Work Completed
Creating a database of the webscraped data.

Understanding Achieved:

Without even understanding a database before the start of this module I now see the importance of databases to hold structured sets of data in one place. There are many different sources where you can build a database, from which I chose SQL. These databases can be stored locally or in a cloud. My goal was to try to just webscrape my data and input into a data base. I first started by watching tutorials on youtube and reading different articles on building databases. I then downloaded the applications for mysql and set-up my account and host

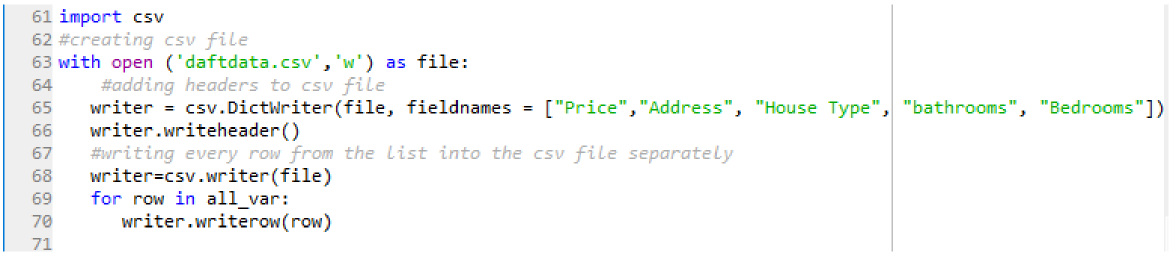


```
import mysql.connector

d_b = mysql.connector.connect(host = "localhost", user = "root", passwd = "TotoPanda1", database = "daftd

print(d_b)
mycursor = d_b.cursor(buffered=True)
#create database
#mycursor.execute("CREATE DATABASE daftdatabase")
#create table
#mycursor.execute("CREATE TABLE DaftTable(price Integer(10), location VARCHAR(50), type VARCHAR(20), bedr
show_t = mycursor.execute("SHOW TABLES")

```

Firstly I had to import the module before setting up a connection with the database, which was going to be stored locally in my computer. I then created my database and a table in MySQL using the above code before commenting it out when running the next stage. I could check and see that the table with the correct headings was now stored in my database on MySQL workbench. Then I began to run into problems as I tried to insert the data from the webscraping errors began to occur. While I could create a database row by row individually that would take too long using all the data I had collected. I tried using the following code to input the data but it would not work.

```
for var in a_v:
   data = mycursor.execute("INSERT INTO DaftTable(price, location, type, bathrooms, bedrooms)
VALUES"+var)
   mycursor.commit()
```

I then decided to try another route. I had to learn to create a csv file. A csv file, short for comma separated values file, it is used to hold store data separated with commas and can be viewed in excel to see the tabular format clearly.



```
import csv
#creating csv file
with open ('daftdata.csv','w') as file:
    #adding headers to csv file
   writer = csv.DictWriter(file, fieldnames = ["Price","Address", "House Type", "bathrooms", "Bedrooms"])
   writer.writeheader()
   #writing every row from the list into the csv file separately
   writer=csv.writer(file)
   for row in all_var:
      writer.writerow(row)

```

To create the csv file I had to first import the module and then using the with and open function I created a new file. I then added headers and wrote each row from the list into rows on the csv file. After this I inserted the csv file into the MySQL database table dafttable.

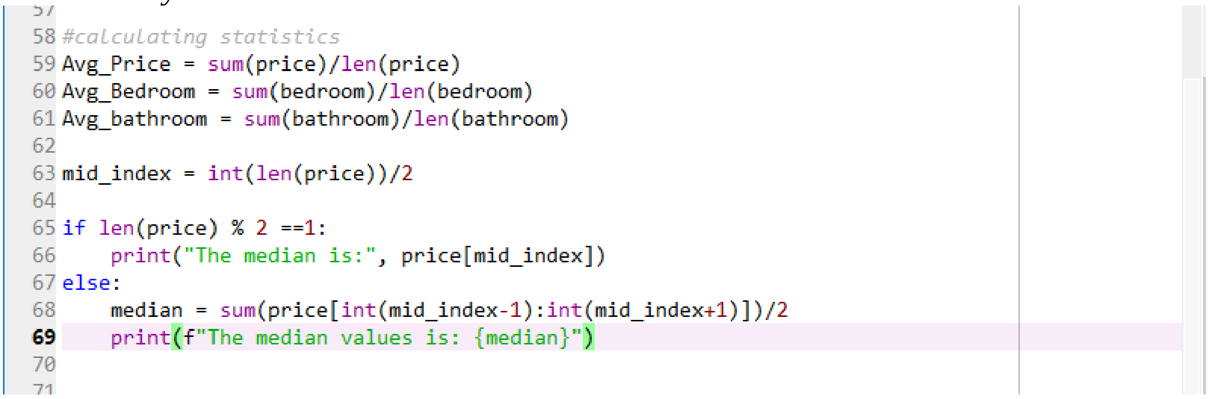
Date: 27/11/2019

Work Completed

Analysing data

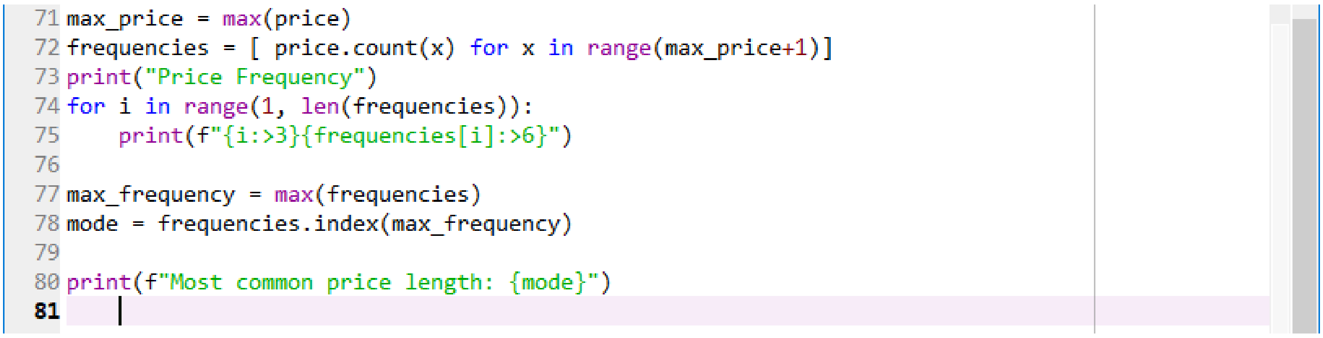
Understanding Achieved:

The whole purpose of getting this data is to analyse and make some conclusion from it. To start I had to learn how to obtain the basic statistics such as mean, mode, median etc… Since this is a python scripting course we are expected to do this in python. While in the project I used python, I also gathered a few bits of information about MySQL. In MySQL you must submit a query to the database, after opening a connection, requesting the AVG() of whichever column you wish to get the average of. The first step I had to make was to create a new list of each of the variables. I then had to append the information from the tables into the list. Calculating mean was fine, it could be done with the following code but calculating mode and median required a bit more work. To calculate the median you first had to create a variable called mid_index. This mid_index divided the length of the list by two creating a number which should be close to the middle. I then checked to see if it was an odd number by checking if the modulo of the length was 1 which meant that then mid_index number was our median. Otherwise we got the sum of the two values either side of the mid_index and divided this values by 2.



```
#calculating statistics
Avg_Price = sum(price)/len(price)
Avg_Bedroom = sum(bedroom)/len(bedroom)
Avg_bathroom = sum(bathroom)/len(bathroom)

mid_index = int(len(price))/2

if len(price) % 2 ==1:
    print("The median is:", price[mid_index])
else:
    median = sum(price[int(mid_index-1):int(mid_index+1)])/2
    print(f"The median values is: {median}")
```

The next part of the analysis was to calculate the mode of the price, the mode is the value that occurs most often. To obtain the mode it was first required to find the maximum value price then for every value in the range between the maximum value and 1 we obtained the values frequency. Then we obtained the maximum frequency value and printed it.



```
max_price = max(price)
frequencies = [ price.count(x) for x in range(max_price+1)]
print("Price Frequency")
for i in range(1, len(frequencies)):
    print(f"{i:>3}{frequencies[i]:>6}")

max_frequency = max(frequencies)
mode = frequencies.index(max_frequency)

print(f"Most common price length: {mode}")
```

To obtain the standard deviation we use the following formula

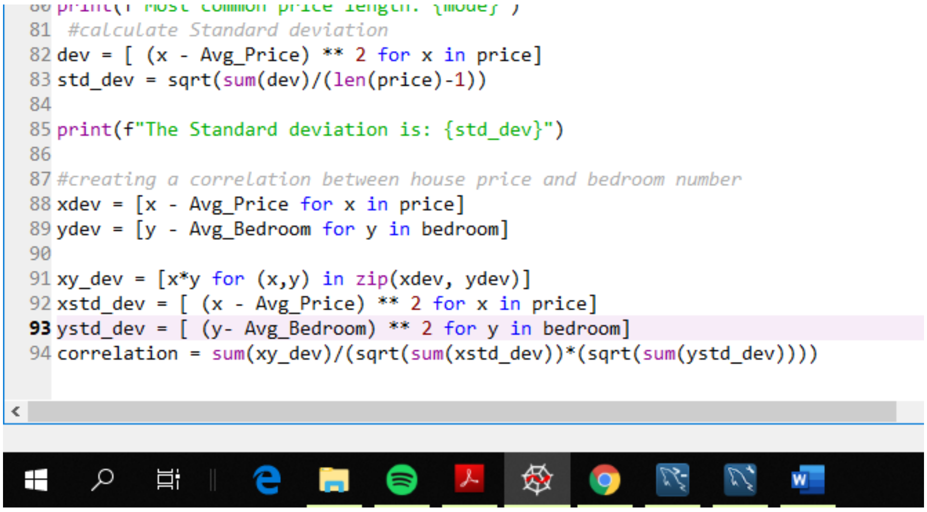
$$s = \sqrt{\frac{\sum(x - \bar{x}\ \ )^2}{n - 1}}$$

Where x bar is the mean. N is the number of values. The standard deviation tells us the variation in the data.

To calculate the r coefficient between two different values you can use the following formula:

$$\mathrm{r} = \frac{\sum(\mathrm{x} - \bar{\mathrm{x}})(\ \mathrm{y} - \bar{\mathrm{y}})}{\sqrt{\sum(\mathrm{x} - \bar{\mathrm{x}})^2}\ \sqrt{\sum(\ \mathrm{y} - \bar{\mathrm{y}})^2}}$$

The r coefficient tells us how closely related two different variables are ranging from -1 to 1



```
print(f"Most common price length: {mode}")
 #calculate Standard deviation
dev = [ (x - Avg_Price) ** 2 for x in price]
std_dev = sqrt(sum(dev)/(len(price)-1))

print(f"The Standard deviation is: {std_dev}")

#creating a correlation between house price and bedroom number
xdev = [x - Avg_Price for x in price]
ydev = [y - Avg_Bedroom for y in bedroom]

xy_dev = [x*y for (x,y) in zip(xdev, ydev)]
xstd_dev = [ (x - Avg_Price) ** 2 for x in price]
ystd_dev = [ (y- Avg_Bedroom) ** 2 for y in bedroom]
correlation = sum(xy_dev)/(sqrt(sum(xstd_dev))*(sqrt(sum(ystd_dev))))
```

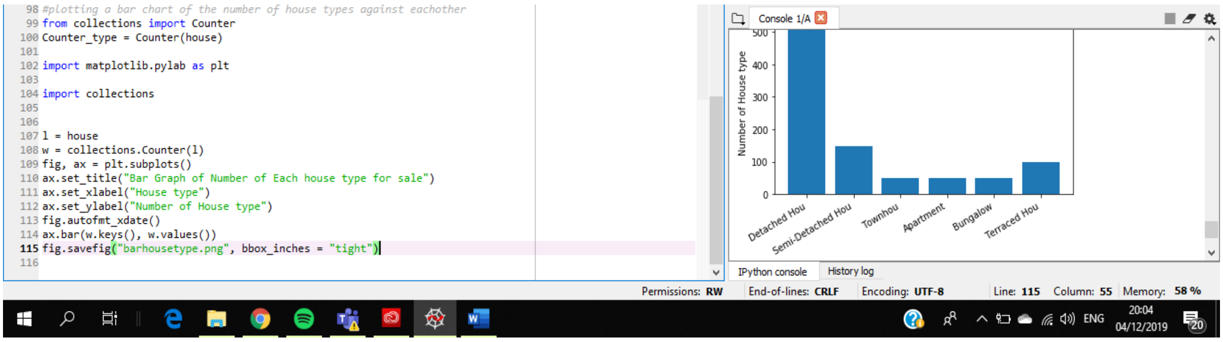
Date: 29/11/2019

Work Completed
Creating visualisations
Understanding Achieved:
There are many ways to create visualisations in python using matplotlib or plotly. For my project is decided to use ggplot2. In ggplot2

To install we use:

pip install plotnine.

After installation I discovered that this function requires pandas which aren't meant to be used as part of the project. Therefore I decided to start using Bokeh. I tried using bokeh and figured out how to make scatter plots. Due to time constraints I focused instead on matplotlib as that is what was covered on the course. I created a bar chart of the number of house types vs the house type. Firstly I had to use the counter function to count the number of each house type. After doing this I used the collection function and also made a dictionary with each house type being the key and the number of that house type being assigned a value. I then used matplotlib to plot a bar chart of these. I ran into the problem of the x labels all overlapping eachother whenever I plotted the graph. This was easily solved using using the .autofmt_ function which added a space and rotated the labels. This was done using the following code



```python
#plotting a bar chart of the number of house types against eachother
from collections import Counter
Counter_type = Counter(house)

import matplotlib.pylab as plt

import collections


l = house
w = collections.Counter(l)
fig, ax = plt.subplots()
ax.set_title("Bar Graph of Number of Each house type for sale")
ax.set_xlabel("House type")
ax.set_ylabel("Number of House type")
fig.autofmt_xdate()
ax.bar(w.keys(), w.values())
fig.savefig("barhousetype.png", bbox_inches = "tight")

```

Permissions: RW   End-of-lines: CRLF   Encoding: UTF-8   Line: 115   Column: 55   Memory: 58 %

I then proceeded to complete a pie chart and scatter chart using matplotlib.

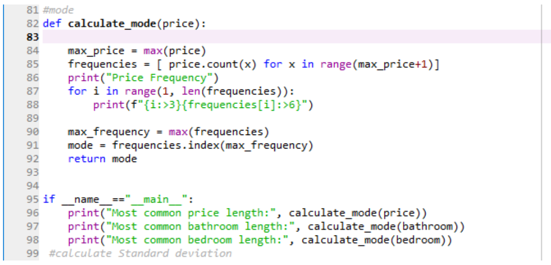
Date: 01/12/2019

Work Completed
Modular Development
Understanding Achieved:
Due to a lot of my code being repetitive in terms of calculating statistics. I decided the implement functions, functions are pieces of code which help simplify code and stop it being reused. Using the help of the notes I created functions for mode, mean and median. Instead of having to type the same code multiple times I could instead implement in over again using:

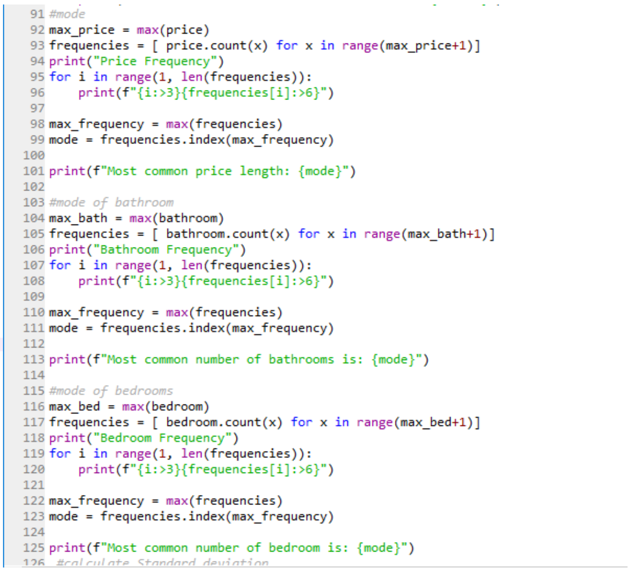
Def calculate_mean():

Then I could call the function to be used by using

if __name__ ==”__main__”:
This is a side by side example of the time and space you could save by implementing functions



```
#mode
def calculate_mode(price):

    max_price = max(price)
    frequencies = [ price.count(x) for x in range(max_price+1)]
    print("Price Frequency")
    for i in range(1, len(frequencies)):
        print(f"{i:>3}{frequencies[i]:>6}")

    max_frequency = max(frequencies)
    mode = frequencies.index(max_frequency)
    return mode


if __name__=="__main__":
    print("Most common price length:", calculate_mode(price))
    print("Most common bathroom length:", calculate_mode(bathroom))
    print("Most common bedroom length:", calculate_mode(bedroom))
 #calculate Standard deviation
```



```
#mode
max_price = max(price)
frequencies = [ price.count(x) for x in range(max_price+1)]
print("Price Frequency")
for i in range(1, len(frequencies)):
    print(f"{i:>3}{frequencies[i]:>6}")

max_frequency = max(frequencies)
mode = frequencies.index(max_frequency)

print(f"Most common price length: {mode}")

#mode of bathroom
max_bath = max(bathroom)
frequencies = [ bathroom.count(x) for x in range(max_bath+1)]
print("Bathroom Frequency")
for i in range(1, len(frequencies)):
    print(f"{i:>3}{frequencies[i]:>6}")

max_frequency = max(frequencies)
mode = frequencies.index(max_frequency)

print(f"Most common number of bathrooms is: {mode}")

#mode of bedrooms
max_bed = max(bedroom)
frequencies = [ bedroom.count(x) for x in range(max_bed+1)]
print("Bedroom Frequency")
for i in range(1, len(frequencies)):
    print(f"{i:>3}{frequencies[i]:>6}")

max_frequency = max(frequencies)
mode = frequencies.index(max_frequency)

print(f"Most common number of bedroom is: {mode}")
 #calculate Standard deviation
```

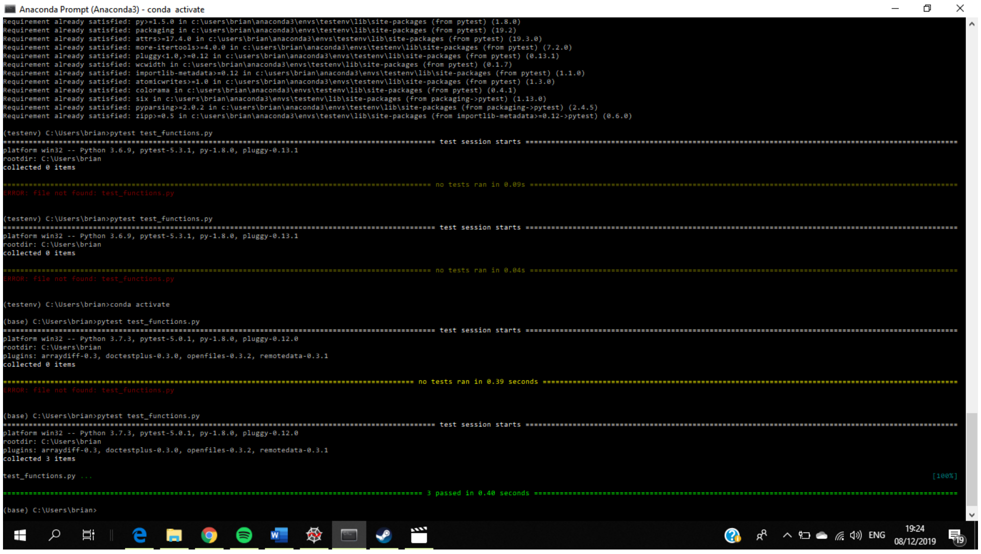
When you create any functions, or any code for that matter, it should be tested. Testing should be done regularly and if you test early it could be possible to identify important mistakes and help to remedy them. There are many different forms of software testing techniques. From my reading there is various different forms of testing examples include performance testing, which evaluates if the code performs its specific requirement, correctness testing, which checks if the code works correctly, and unit testing, which tests specific parts of the code individually.

Date: 01/12/2019

Work Completed
Testing
Understanding Achieved:
For the project I decided to do unit testing on all of my functions. The problem was I couldn't install it, loading environments wouldn't work even when I left it for hours. After doing some research, I found a suggestion to create a new environment. Once I created a new environment and tried to install unit testing on this environment, it worked perfectly. In order to test a function it must be stored in a file starting with test_. The next step was to install the a spyder command line prompt in order to run the tests, this was done using the

pip install spyder spyder-terminal

command in the console. After creating the functions file to be tested, I tried testing the file multiple times but the following error kept appearing, ERROR: file not found: test_functions.py. Searching for answers to this error was difficult as there was not much information about it online. Eventually I noticed the root directory which said it was in C:\Users\brian. My file was in this directory but it was in a folder within this. I moved the function outside of this folder and the test ran perfectly. After all my installing and changing environments, it turns out it was just changing folders was all that was required.



```
Requirement already satisfied: py>=1.5.0 in c:\users\brian\anaconda3\envs\testenv\lib\site-packages (from pytest) (1.8.0)
Requirement already satisfied: packaging in c:\users\brian\anaconda3\envs\testenv\lib\site-packages (from pytest) (19.2)
Requirement already satisfied: attrs>=17.4.0 in c:\users\brian\anaconda3\envs\testenv\lib\site-packages (from pytest) (19.3.0)
Requirement already satisfied: more-itertools>=4.0.0 in c:\users\brian\anaconda3\envs\testenv\lib\site-packages (from pytest) (7.2.0)
Requirement already satisfied: pluggy<1.0,>=0.12 in c:\users\brian\anaconda3\envs\testenv\lib\site-packages (from pytest) (0.13.1)
Requirement already satisfied: wcwidth in c:\users\brian\anaconda3\envs\testenv\lib\site-packages (from pytest) (0.1.7)
Requirement already satisfied: importlib-metadata>=0.12 in c:\users\brian\anaconda3\envs\testenv\lib\site-packages (from pytest) (1.1.0)
Requirement already satisfied: atomicwrites>=1.0 in c:\users\brian\anaconda3\envs\testenv\lib\site-packages (from pytest) (1.3.0)
Requirement already satisfied: colorama in c:\users\brian\anaconda3\envs\testenv\lib\site-packages (from pytest) (0.4.1)
Requirement already satisfied: six in c:\users\brian\anaconda3\envs\testenv\lib\site-packages (from packaging->pytest) (1.13.0)
Requirement already satisfied: pyparsing>=2.0.2 in c:\users\brian\anaconda3\envs\testenv\lib\site-packages (from packaging->pytest) (2.4.5)
Requirement already satisfied: zipp>=0.5 in c:\users\brian\anaconda3\envs\testenv\lib\site-packages (from importlib-metadata>=0.12->pytest) (0.6.0)

(testenv) C:\Users\brian>pytest test_functions.py
================================================================ test session starts ================================================================
platform win32 -- Python 3.6.9, pytest-5.3.1, py-1.8.0, pluggy-0.13.1
rootdir: C:\Users\brian
collected 0 items

================================================================ no tests ran in 0.09s ================================================================
ERROR: file not found: test_functions.py


(testenv) C:\Users\brian>pytest test_functions.py
================================================================ test session starts ================================================================
platform win32 -- Python 3.6.9, pytest-5.3.1, py-1.8.0, pluggy-0.13.1
rootdir: C:\Users\brian
collected 0 items

================================================================ no tests ran in 0.04s ================================================================
ERROR: file not found: test_functions.py


(testenv) C:\Users\brian>conda activate

(base) C:\Users\brian>pytest test_functions.py
================================================================ test session starts ================================================================
platform win32 -- Python 3.7.3, pytest-5.0.1, py-1.8.0, pluggy-0.12.0
rootdir: C:\Users\brian
plugins: arraydiff-0.3, doctestplus-0.3.0, openfiles-0.3.2, remotedata-0.3.1
collected 0 items

============================================================== no tests ran in 0.39 seconds ==============================================================
ERROR: file not found: test_functions.py


(base) C:\Users\brian>pytest test_functions.py
================================================================ test session starts ================================================================
platform win32 -- Python 3.7.3, pytest-5.0.1, py-1.8.0, pluggy-0.12.0
rootdir: C:\Users\brian
plugins: arraydiff-0.3, doctestplus-0.3.0, openfiles-0.3.2, remotedata-0.3.1
collected 3 items

test_functions.py ...                                                                                                                          [100%]

=============================================================== 3 passed in 0.40 seconds ===============================================================

(base) C:\Users\brian>
```

Appendix 2: References

“Understanding how to use MySQL” https://dev.mysql.com/doc/mysql-getting-started/en/
“Installing Unit Tests” https://www.spyder-ide.org/blog/introducing-unittest-plugin/
“Changing Environments” https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html