Assignment: Pranav Maniyar

Real World Scenarios - This project is broken down in two parts. The first part of the project is analysing UK Housing Market. Further down at a later stage it will focus on "Slough" area as this is my residential area. I am interested in ivestigating at how the Sales Volume and Average Sale Price has changed over the years in this area for different types of properties. I was also intersted in understanding the difference in the sale price of Cash Vs Mortgage. For this part of the exercise various different data sources have been used to source different set of information by reading CSV and Webscrapping methods.

For Machine Learning, this project is looking at the the set of data for Students Grading and trying to predict one of the 3 gadings using liner regression.

In [1]:
```
#importing the required modules.  The modules for the machine learning will be impor
import pandas as pd
import numpy as np
import re
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
import requests
from bs4 import BeautifulSoup as bs
```

In [2]:
```
#Reading CSV File for UK Housing Market sourced from Gov.UK
df_house=pd.read_csv('UK-HPI-full-file-2022-07.csv')
```

In [3]:
```
df_house.head(5)
```

Out[3]:

	Date	RegionName	AreaCode	AveragePrice	Index	IndexSA	1m%Change	12m%Cha
0	01/01/2004	Aberdeenshire	S12000034	81693.66964	40.864214	NaN	NaN	I
1	01/02/2004	Aberdeenshire	S12000034	81678.76231	40.856757	NaN	-0.018248	I
2	01/03/2004	Aberdeenshire	S12000034	83525.09702	41.780317	NaN	2.260483	I
3	01/04/2004	Aberdeenshire	S12000034	84333.67900	42.184780	NaN	0.968071	I
4	01/05/2004	Aberdeenshire	S12000034	86379.95396	43.208353	NaN	2.426403	I

5 rows × 54 columns

In [4]:
```
df_house.tail(5)
```

Out[4]:

	Date	RegionName	AreaCode	AveragePrice	Index	IndexSA	1m%Change	1
137771	01/03/2022	Yorkshire and The Humber	E12000003	196887.00370	144.590495	143.380845	-0.161327	
137772	01/04/2022	Yorkshire and The Humber	E12000003	199389.28479	146.428128	144.457866	1.270922	

	Date	RegionName	AreaCode	AveragePrice	Index	IndexSA	1m%Change	1
137773	01/05/2022	Yorkshire and The Humber	E12000003	204076.87702	149.870616	147.641720	2.350975	
137774	01/06/2022	Yorkshire and The Humber	E12000003	205647.53948	151.024084	148.074041	0.769643	
137775	01/07/2022	Yorkshire and The Humber	E12000003	211960.15107	155.659960	152.076749	3.069627	

5 rows × 54 columns

In [5]:

```
#Examining the data to remove the unnecesary variables
df_house.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 137776 entries, 0 to 137775
Data columns (total 54 columns):
 #   Column                  Non-Null Count   Dtype  
---  ------                  --------------   -----  
 0   Date                    137776 non-null  object 
 1   RegionName              137776 non-null  object 
 2   AreaCode                137776 non-null  object 
 3   AveragePrice            137776 non-null  float64
 4   Index                   137776 non-null  float64
 5   IndexSA                 4629 non-null    float64
 6   1m%Change               137333 non-null  float64
 7   12m%Change              132736 non-null  float64
 8   AveragePriceSA          4629 non-null    float64
 9   SalesVolume             133277 non-null  float64
 10  DetachedPrice           131455 non-null  float64
 11  DetachedIndex           131455 non-null  float64
 12  Detached1m%Change       131046 non-null  float64
 13  Detached12m%Change      126571 non-null  float64
 14  SemiDetachedPrice       131467 non-null  float64
 15  SemiDetachedIndex       131467 non-null  float64
 16  SemiDetached1m%Change   131058 non-null  float64
 17  SemiDetached12m%Change  126583 non-null  float64
 18  TerracedPrice           131494 non-null  float64
 19  TerracedIndex           131494 non-null  float64
 20  Terraced1m%Change       131085 non-null  float64
 21  Terraced12m%Change      126610 non-null  float64
 22  FlatPrice               131798 non-null  float64
 23  FlatIndex               131798 non-null  float64
 24  Flat1m%Change           131388 non-null  float64
 25  Flat12m%Change          126902 non-null  float64
 26  CashPrice               51816 non-null   float64
 27  CashIndex               51816 non-null   float64
 28  Cash1m%Change           51410 non-null   float64
 29  Cash12m%Change          46944 non-null   float64
 30  CashSalesVolume         50999 non-null   float64
 31  MortgagePrice           51816 non-null   float64
 32  MortgageIndex           51816 non-null   float64
 33  Mortgage1m%Change       51410 non-null   float64
 34  Mortgage12m%Change      46944 non-null   float64
 35  MortgageSalesVolume     51000 non-null   float64
 36  FTBPrice                52212 non-null   float64
 37  FTBIndex                52212 non-null   float64
 38  FTB1m%Change            51806 non-null   float64
 39  FTB12m%Change           47340 non-null   float64
```

```
 40  FOOPrice                51816 non-null   float64
 41  FOOIndex                51816 non-null   float64
 42  FOO1m%Change            51410 non-null   float64
 43  FOO12m%Change           46944 non-null   float64
 44  NewPrice                130965 non-null  float64
 45  NewIndex                130965 non-null  float64
 46  New1m%Change            130555 non-null  float64
 47  New12m%Change           126069 non-null  float64
 48  NewSalesVolume          129486 non-null  float64
 49  OldPrice                130978 non-null  float64
 50  OldIndex                130978 non-null  float64
 51  Old1m%Change            130568 non-null  float64
 52  Old12m%Change           126082 non-null  float64
 53  OldSalesVolume          130966 non-null  float64
dtypes: float64(51), object(3)
memory usage: 56.8+ MB
```

In [6]:

```
#Keeping only required variables for this project.
df_house_clean = pd.DataFrame(df_house,  columns=['Date','RegionName','AveragePrice'
                                                  'Detached12m%Change','SemiDetachedP
                                                  'FlatPrice','Flat12m%Change','CashP
                                                   'NewPrice','New1m%Change','New12m%
                                                  ])
```

In [7]:

```
#changing the date type from object to datetime so that it can be merged with Crime

df_house_clean['Date'] = pd.to_datetime(df_house_clean['Date'], format='%d/%m/%Y')
```

In [8]:

```
df_house_clean.head(5)
```

Out[8]:

	Date	RegionName	AveragePrice	AveragePriceSA	SalesVolume	DetachedPrice	Detached12m%
0	2004-01-01	Aberdeenshire	81693.66964	NaN	388.0	122490.0641	
1	2004-02-01	Aberdeenshire	81678.76231	NaN	326.0	121280.8840	
2	2004-03-01	Aberdeenshire	83525.09702	NaN	453.0	123395.4269	
3	2004-04-01	Aberdeenshire	84333.67900	NaN	571.0	122334.0258	
4	2004-05-01	Aberdeenshire	86379.95396	NaN	502.0	124498.8747	

5 rows × 26 columns

In [9]:

```
df_house_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 137776 entries, 0 to 137775
Data columns (total 26 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   Date                    137776 non-null  datetime64[ns]
```

```
 1   RegionName              137776 non-null  object
 2   AveragePrice            137776 non-null  float64
 3   AveragePriceSA          4629 non-null    float64
 4   SalesVolume             133277 non-null  float64
 5   DetachedPrice           131455 non-null  float64
 6   Detached12m%Change      126571 non-null  float64
 7   SemiDetachedPrice       131467 non-null  float64
 8   SemiDetached12m%Change  126583 non-null  float64
 9   TerracedPrice           131494 non-null  float64
 10  Terraced12m%Change      126610 non-null  float64
 11  FlatPrice               131798 non-null  float64
 12  Flat12m%Change          126902 non-null  float64
 13  CashPrice               51816 non-null   float64
 14  Cash12m%Change          46944 non-null   float64
 15  CashSalesVolume         50999 non-null   float64
 16  MortgagePrice           51816 non-null   float64
 17  MortgageSalesVolume     51000 non-null   float64
 18  NewPrice                130965 non-null  float64
 19  New1m%Change            130555 non-null  float64
 20  New12m%Change           126069 non-null  float64
 21  NewSalesVolume          129486 non-null  float64
 22  OldPrice                130978 non-null  float64
 23  Old1m%Change            130568 non-null  float64
 24  Old12m%Change           126082 non-null  float64
 25  OldSalesVolume          130966 non-null  float64
dtypes: datetime64[ns](1), float64(24), object(1)
memory usage: 27.3+ MB
```

In [10]:

```
#checking if the dates are working as intended
df_house_clean[(df_house_clean['Date'] >= '2005-01-01')]
```

Out[10]:

	Date	RegionName	AveragePrice	AveragePriceSA	SalesVolume	DetachedPrice	Detached
12	2005-01-01	Aberdeenshire	101818.93310	NaN	400.0	147874.76830	
13	2005-02-01	Aberdeenshire	97626.00076	NaN	289.0	141906.63220	
14	2005-03-01	Aberdeenshire	94779.34241	NaN	435.0	137477.95560	
15	2005-04-01	Aberdeenshire	95847.70590	NaN	499.0	136933.00080	
16	2005-05-01	Aberdeenshire	101744.46620	NaN	515.0	144287.69480	
...	...	...	...	...	...	...	
137771	2022-03-01	Yorkshire and The Humber	196887.00370	195239.83920	5381.0	326581.90590	
137772	2022-04-01	Yorkshire and The Humber	199389.28479	196706.40443	4823.0	329420.75611	
137773	2022-05-01	Yorkshire and The Humber	204076.87702	201041.81756	4020.0	335548.54265	
137774	2022-06-01	Yorkshire and The Humber	205647.53948	201630.50399	NaN	335641.76202	
137775	2022-07-01	Yorkshire and The Humber	211960.15107	207080.93940	NaN	347175.33547	

88831 rows × 26 columns

In [11]:
```
#converting date to mmmyy to merge later with Crime data.  To this I have created 2

df_house_clean['Year'] = pd.DatetimeIndex(df_house_clean['Date']).year
df_house_clean['Month'] = pd.DatetimeIndex(df_house_clean['Date']).month
```

In [12]:
```
df_house_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 137776 entries, 0 to 137775
Data columns (total 28 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   Date                    137776 non-null  datetime64[ns]
 1   RegionName              137776 non-null  object
 2   AveragePrice            137776 non-null  float64
 3   AveragePriceSA          4629 non-null    float64
 4   SalesVolume             133277 non-null  float64
 5   DetachedPrice           131455 non-null  float64
 6   Detached12m%Change      126571 non-null  float64
 7   SemiDetachedPrice       131467 non-null  float64
 8   SemiDetached12m%Change  126583 non-null  float64
 9   TerracedPrice           131494 non-null  float64
 10  Terraced12m%Change      126610 non-null  float64
 11  FlatPrice               131798 non-null  float64
 12  Flat12m%Change          126902 non-null  float64
 13  CashPrice               51816 non-null   float64
 14  Cash12m%Change          46944 non-null   float64
 15  CashSalesVolume         50999 non-null   float64
 16  MortgagePrice           51816 non-null   float64
 17  MortgageSalesVolume     51000 non-null   float64
 18  NewPrice                130965 non-null  float64
 19  New1m%Change            130555 non-null  float64
 20  New12m%Change           126069 non-null  float64
 21  NewSalesVolume          129486 non-null  float64
 22  OldPrice                130978 non-null  float64
 23  Old1m%Change            130568 non-null  float64
 24  Old12m%Change           126082 non-null  float64
 25  OldSalesVolume          130966 non-null  float64
 26  Year                    137776 non-null  int64
 27  Month                   137776 non-null  int64
dtypes: datetime64[ns](1), float64(24), int64(2), object(1)
memory usage: 29.4+ MB
```

In [13]:
```
df_house_clean.head()
```

Out[13]:

	Date	RegionName	AveragePrice	AveragePriceSA	SalesVolume	DetachedPrice	Detached12m%
0	2004-01-01	Aberdeenshire	81693.66964	NaN	388.0	122490.0641	
1	2004-02-01	Aberdeenshire	81678.76231	NaN	326.0	121280.8840	
2	2004-03-01	Aberdeenshire	83525.09702	NaN	453.0	123395.4269	
3	2004-04-01	Aberdeenshire	84333.67900	NaN	571.0	122334.0258	

	Date	RegionName	AveragePrice	AveragePriceSA	SalesVolume	DetachedPrice	Detached12m%
4	2004-05-01	Aberdeenshire	86379.95396	NaN	502.0	124498.8747	

5 rows × 28 columns

In [14]:
```
#Reading a CSV file to merge and create mmm-yy column
MonthFile=pd.read_csv('MonthFile.csv')
```

In [15]:
```
MonthFile.head(12)
```

Out[15]:

	Month	Month_Txt
0	1	Jan
1	2	Feb
2	3	Mar
3	4	Apr
4	5	May
5	6	Jun
6	7	Jul
7	8	Aug
8	9	Sep
9	10	Oct
10	11	Nov
11	12	Dec

In [16]:
```
df_house_clean=pd.merge(df_house_clean, MonthFile, left_on='Month', right_on='Month'
```

In [17]:
```
df_house_clean.head()
```

Out[17]:

	Date	RegionName	AveragePrice	AveragePriceSA	SalesVolume	DetachedPrice	Detached12m%
0	2004-01-01	Aberdeenshire	81693.66964	NaN	388.0	122490.0641	
1	2005-01-01	Aberdeenshire	101818.93310	NaN	400.0	147874.7683	2(
2	2006-01-01	Aberdeenshire	111351.61130	NaN	395.0	156980.8357	6
3	2007-01-01	Aberdeenshire	145371.47530	NaN	361.0	201628.8122	28
4	2008-01-01	Aberdeenshire	173220.42830	NaN	304.0	237001.4713	17

5 rows × 29 columns

In [18]:
```
df_house_clean["Date"] = df_house_clean["Month_Txt"] + " "+df_house_clean["Year"].as
```

In [19]:
```
df_house_clean.head()
```

Out[19]:

	Date	RegionName	AveragePrice	AveragePriceSA	SalesVolume	DetachedPrice	Detached12m%(
0	Jan 2004	Aberdeenshire	81693.66964	NaN	388.0	122490.0641	
1	Jan 2005	Aberdeenshire	101818.93310	NaN	400.0	147874.7683	20.
2	Jan 2006	Aberdeenshire	111351.61130	NaN	395.0	156980.8357	6.
3	Jan 2007	Aberdeenshire	145371.47530	NaN	361.0	201628.8122	28.
4	Jan 2008	Aberdeenshire	173220.42830	NaN	304.0	237001.4713	17.

5 rows × 29 columns

In [20]:
```
#Investigating different regions
print(df_house_clean['RegionName'])
```

```
0                    Aberdeenshire
1                    Aberdeenshire
2                    Aberdeenshire
3                    Aberdeenshire
4                    Aberdeenshire
                     ...
137771    Yorkshire and The Humber
137772    Yorkshire and The Humber
137773    Yorkshire and The Humber
137774    Yorkshire and The Humber
137775    Yorkshire and The Humber
Name: RegionName, Length: 137776, dtype: object
```

Investigating Sales and Prices

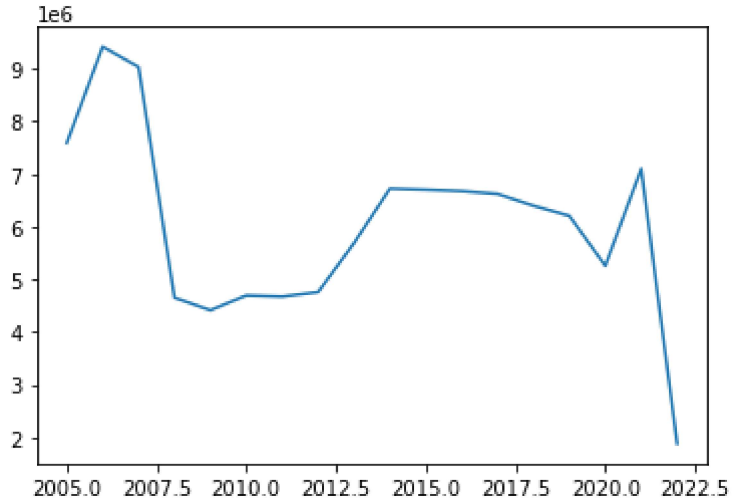
In [21]:
```
Salesbyyear=df_house_clean[["SalesVolume","RegionName","NewSalesVolume","OldSalesVol
Salesbyyear_Clean=Salesbyyear[(Salesbyyear['Year'] >= 2005)]
Salesbyyear.info()
Salesbyyear_Clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55 entries, 0 to 54
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Year            55 non-null     int64
 1   SalesVolume     55 non-null     float64
 2   NewSalesVolume  55 non-null     float64
 3   OldSalesVolume  55 non-null     float64
```

```
dtypes: float64(3), int64(1)
memory usage: 1.8 KB
<class 'pandas.core.frame.DataFrame'>
Int64Index: 18 entries, 37 to 54
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Year            18 non-null     int64
 1   SalesVolume     18 non-null     float64
 2   NewSalesVolume  18 non-null     float64
 3   OldSalesVolume  18 non-null     float64
dtypes: float64(3), int64(1)
memory usage: 720.0 bytes
```

In [22]:

```
plt.plot(Salesbyyear_Clean["Year"],Salesbyyear_Clean["SalesVolume"])
plt.show()
```



In [23]:

```
Pricebyyear=df_house_clean[["AveragePrice", "DetachedPrice","SemiDetachedPrice","Ter
Pricebyyear_Clean=Pricebyyear[(Pricebyyear['Year'] >= 2005)]
Pricebyyear_Clean.info()
Pricebyyear_Clean
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 18 entries, 37 to 54
Data columns (total 6 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Year               18 non-null     int64
 1   AveragePrice       18 non-null     float64
 2   DetachedPrice      18 non-null     float64
 3   SemiDetachedPrice  18 non-null     float64
 4   TerracedPrice      18 non-null     float64
 5   FlatPrice          18 non-null     float64
dtypes: float64(5), int64(1)
memory usage: 1008.0 bytes
```

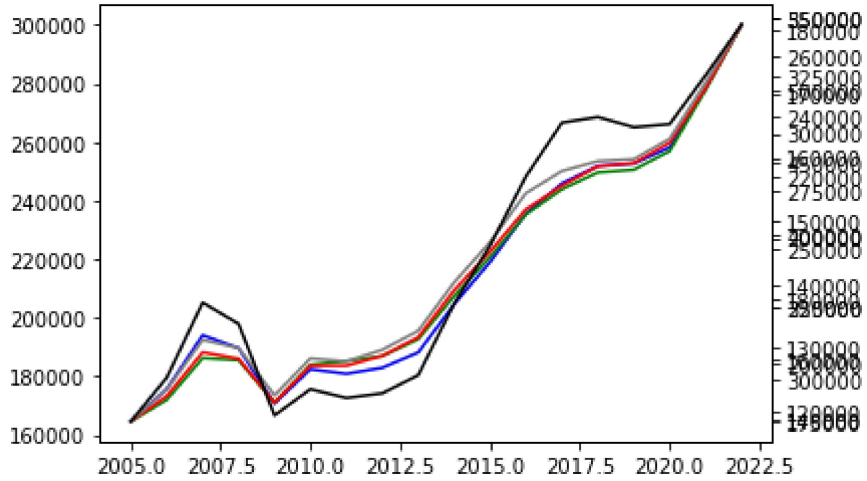
Out[23]:

	Year	AveragePrice	DetachedPrice	SemiDetachedPrice	TerracedPrice	FlatPrice
37	2005	164595.497797	270485.767599	175201.730557	139628.423406	118448.872993
38	2006	175684.772076	285535.962637	186236.033979	150191.822163	125359.498493
39	2007	194099.302111	314555.585396	205236.137749	166508.820040	137197.021681
40	2008	189679.638569	313109.887035	202450.670110	163841.058048	133845.216481
41	2009	170751.439991	283617.892249	183431.892544	148285.543074	119415.507686

	Year	AveragePrice	DetachedPrice	SemiDetachedPrice	TerracedPrice	FlatPrice
42	2010	182463.227661	309889.760116	199431.708942	160429.186582	123546.821117
43	2011	180920.438032	312557.554908	199431.285042	159455.219264	122159.130460
44	2012	182974.697777	315907.238953	203710.375478	163280.718683	122897.351349
45	2013	188199.476137	327669.337025	211758.896451	169527.687130	125763.118240
46	2014	204663.569855	357620.910403	232101.453692	185464.431043	137009.228343
47	2015	219077.713291	384706.040294	248856.799541	198748.012689	146170.995490
48	2016	235668.232093	414371.214226	267275.851373	214877.203824	157083.420391
49	2017	245929.271388	431855.308928	277418.437406	222140.987755	165528.899063
50	2018	251895.253259	443466.275476	285874.247884	225378.565554	166424.740302
51	2019	252696.516991	445333.130569	287190.199772	226021.078137	164819.936587
52	2020	258260.336871	458274.181672	296162.151760	232780.287583	165299.852983
53	2021	278490.460934	500828.994242	319828.352662	251285.573903	173018.919322
54	2022	300171.737252	546131.990473	347316.028380	270352.547432	180976.061644

In [24]:

```
fig,ax1 = plt.subplots()
Years=Pricebyyear_Clean["Year"]
AveragePrice=Pricebyyear_Clean["AveragePrice"]
DetachedPrice=Pricebyyear_Clean["DetachedPrice"]
SemiDetachedPrice=Pricebyyear_Clean["SemiDetachedPrice"]
TerracedPrice=Pricebyyear_Clean["TerracedPrice"]
FlatPrice=Pricebyyear_Clean["FlatPrice"]

ax1.plot(Years,AveragePrice,color="blue")

ax2=ax1.twinx()
ax2.plot(Years,DetachedPrice,color="green")

ax3=ax1.twinx()
ax3.plot(Years,SemiDetachedPrice,color="red")

ax4=ax1.twinx()
ax4.plot(Years,TerracedPrice,color="grey")

ax5=ax1.twinx()
ax5.plot(Years,FlatPrice,color="black")

plt.show()
```



The above 2 charts are clarly showing the fact that UK housing market was booming around mid 2005 to 2006 in terms of sales volum and increase in average sales prices. This clearly took a massive hit at the time of 2008 financial crisis where you can see a deep dive in the graph. From 2010 both Sales volume and prieces have started going up. The Sales volume is still to hit prior to 2008 financial crisis mark but average sale price has gone up significantly. Interstingly all different types of houses have followed the similar pattern in terms of incrase in their prices but Flats have seen some massive growth in the price around 2016 to 2018 which has then slowed down and now inline with other types of properties.

Now looking at Slough Specific Numbers

In [25]:
```
#Using Regex to keep only slough Region
df_house_clean['Region'] = df_house_clean['RegionName'].str.contains(r'\Slou\w\w$',
```

In [26]:
```
print(df_house_clean.loc[df_house_clean['Region'] == False])
```

```
            Date                RegionName  AveragePrice  AveragePriceSA  \
0       Jan 2004               Aberdeenshire   81693.66964             NaN
1       Jan 2005               Aberdeenshire  101818.93310             NaN
2       Jan 2006               Aberdeenshire  111351.61130             NaN
3       Jan 2007               Aberdeenshire  145371.47530             NaN
4       Jan 2008               Aberdeenshire  173220.42830             NaN
...          ...                         ...           ...             ...
137771  Dec 2017  Yorkshire and The Humber  158435.41610    154899.57060
137772  Dec 2018  Yorkshire and The Humber  162130.21910    159036.66050
137773  Dec 2019  Yorkshire and The Humber  165654.31670    162123.35810
137774  Dec 2020  Yorkshire and The Humber  179315.58060    175927.14760
137775  Dec 2021  Yorkshire and The Humber  193480.81558    190445.13239

        SalesVolume  DetachedPrice  Detached12m%Change  SemiDetachedPrice  \
0             388.0    122490.0641                 NaN        70563.15784
1             400.0    147874.7683           20.723888        89180.82074
2             395.0    156980.8357            6.157959        95681.84218
3             361.0    201628.8122           28.441673       124947.09450
4             304.0    237001.4713           17.543455       147817.63650
...             ...            ...                 ...               ...
137771       7722.0    254799.5279            3.375240       156584.55530
137772       7433.0    261207.3817            2.514861       160966.47980
137773       7313.0    266424.9779            1.997492       165042.34860
137774       8328.0    290412.8028            9.003595       178203.23150
137775       6123.0    319486.0447           10.011006       193037.08288
```

```
        SemiDetached12m%Change  TerracedPrice  ...  New12m%Change  \
0                          NaN    55319.63336  ...            NaN
1                    26.384396    71919.37068  ...      24.741351
2                     7.289708    79306.80150  ...      10.593246
3                    30.586004   104516.79190  ...      27.332037
4                    18.304181   124860.56390  ...      18.206515
...                        ...            ...  ...            ...
137771                4.318597   125624.72960  ...       3.027151
137772                2.798440   128214.98540  ...       3.374218
137773                2.532123   131347.17780  ...      -0.215119
137774                7.974246   143020.82080  ...       8.736456
137775                8.324120   152854.79816  ...      15.641178

        NewSalesVolume      OldPrice  Old1m%Change  Old12m%Change  \
0                103.0   81043.95084           NaN            NaN
1                 42.0  100978.67930      0.934250      24.597429
2                 44.0  109985.04010     -3.889775       8.919072
3                 39.0  144281.99980      4.333466      31.183295
4                 45.0  172276.83400      0.971876      19.402860
...                ...           ...           ...            ...
137771          1176.0  155964.37420      0.798295       3.849487
137772          1283.0  159644.93400      0.243895       2.359872
137773          1250.0  163454.57700      0.463247       2.386323
137774           659.0  177184.23030      0.604865       8.399675
137775           269.0  189614.71400      1.224259       7.015570

        OldSalesVolume  Year  Month  Month_Txt  Region
0                285.0  2004      1        Jan   False
1                358.0  2005      1        Jan   False
2                351.0  2006      1        Jan   False
3                322.0  2007      1        Jan   False
4                259.0  2008      1        Jan   False
...                ...   ...    ...        ...     ...
137771          6546.0  2017     12        Dec   False
137772          6150.0  2018     12        Dec   False
137773          6063.0  2019     12        Dec   False
137774          7669.0  2020     12        Dec   False
137775          5854.0  2021     12        Dec   False

[137445 rows x 30 columns]
```

In [27]:

```
print(df_house_clean.loc[df_house_clean['Region'] == True])
```

```
            Date RegionName  AveragePrice  AveragePriceSA  SalesVolume  \
8452    Jan 1995     Slough   54250.70947             NaN         92.0
8453    Jan 1996     Slough   53765.05970             NaN        145.0
8454    Jan 1997     Slough   57060.07900             NaN        182.0
8455    Jan 1998     Slough   66903.97400             NaN        168.0
8456    Jan 1999     Slough   73845.26834             NaN        191.0
...          ...        ...           ...             ...          ...
134708  Dec 2017     Slough  303048.49620             NaN        169.0
134709  Dec 2018     Slough  305263.77430             NaN        110.0
134710  Dec 2019     Slough  287422.74910             NaN        110.0
134711  Dec 2020     Slough  297700.25250             NaN         86.0
134712  Dec 2021     Slough  305842.60389             NaN         97.0

        DetachedPrice  Detached12m%Change  SemiDetachedPrice  \
8452     101753.14280                 NaN        69287.18989
8453     100394.29790           -1.335433        69440.11418
8454     107923.34080            7.499473        74198.42836
8455     129047.76080           19.573542        87988.43234
8456     142742.16030           10.611885        97319.39919
```

```
...              ...                ...                ...
134708   562016.22260           1.819609       404805.99910
134709   572433.90620           1.853627       411475.72340
134710   540593.64090          -5.562261       391328.01490
134711   570262.12990           5.488131       409354.94210
134712   602586.09212           5.668264       428875.53609

        SemiDetached12m%Change  TerracedPrice  ...  New12m%Change  \
8452                       NaN    54264.86941  ...            NaN
8453                  0.220711    53758.31612  ...       2.304725
8454                  6.852400    57509.00695  ...       5.027748
8455                 18.585305    67656.18517  ...      17.128569
8456                 10.604765    74282.86678  ...      12.744703
...                        ...            ...  ...            ...
134708                2.427170   318434.06300  ...       1.116608
134709                1.647635   321880.71100  ...       1.059711
134710               -4.896451   305856.81080  ...      -6.278376
134711                4.606603   322704.41790  ...       1.808489
134712                4.768623   331114.12122  ...      10.252749

        NewSalesVolume      OldPrice  Old1m%Change  Old12m%Change  \
8452               8.0   53506.48269           NaN            NaN
8453               6.0   52937.00556      0.062285      -1.064314
8454              19.0   56273.00942      1.923260       6.301837
8455              24.0   65990.47116      0.187745      17.268424
8456              27.0   72672.06000     -0.775779      10.125081
...                ...           ...           ...            ...
134708            25.0  298310.88550     -0.622240       2.247240
134709            30.0  299853.11870     -0.125380       0.516989
134710             2.0  282689.11490     -1.880421      -5.724137
134711             5.0  294503.51140     -0.409694       4.179290
134712             3.0  299778.38243     -0.395338       1.791106

        OldSalesVolume  Year  Month  Month_Txt  Region
8452              84.0  1995      1        Jan    True
8453             139.0  1996      1        Jan    True
8454             163.0  1997      1        Jan    True
8455             144.0  1998      1        Jan    True
8456             164.0  1999      1        Jan    True
...                ...   ...    ...        ...     ...
134708           144.0  2017     12        Dec    True
134709            80.0  2018     12        Dec    True
134710           108.0  2019     12        Dec    True
134711            81.0  2020     12        Dec    True
134712            94.0  2021     12        Dec    True

[331 rows x 30 columns]
```

In [28]:
```
#Creating Dataframe for Only Slough Region
df_house_Slough = pd.DataFrame(df_house_clean[(df_house_clean['Region']==True)])
```

In [29]:
```
print(df_house_Slough)
```

```
            Date RegionName  AveragePrice  AveragePriceSA  SalesVolume  \
8452    Jan 1995     Slough   54250.70947             NaN         92.0
8453    Jan 1996     Slough   53765.05970             NaN        145.0
8454    Jan 1997     Slough   57060.07900             NaN        182.0
8455    Jan 1998     Slough   66903.97400             NaN        168.0
8456    Jan 1999     Slough   73845.26834             NaN        191.0
...          ...        ...           ...             ...          ...
134708  Dec 2017     Slough  303048.49620             NaN        169.0
134709  Dec 2018     Slough  305263.77430             NaN        110.0
```

```
134710  Dec 2019     Slough  287422.74910             NaN        110.0
134711  Dec 2020     Slough  297700.25250             NaN         86.0
134712  Dec 2021     Slough  305842.60389             NaN         97.0

        DetachedPrice  Detached12m%Change  SemiDetachedPrice  \
8452     101753.14280                 NaN        69287.18989
8453     100394.29790           -1.335433        69440.11418
8454     107923.34080            7.499473        74198.42836
8455     129047.76080           19.573542        87988.43234
8456     142742.16030           10.611885        97319.39919
...               ...                 ...               ...
134708   562016.22260            1.819609       404805.99910
134709   572433.90620            1.853627       411475.72340
134710   540593.64090           -5.562261       391328.01490
134711   570262.12990            5.488131       409354.94210
134712   602586.09212            5.668264       428875.53609

        SemiDetached12m%Change  TerracedPrice  ...  New12m%Change  \
8452                       NaN    54264.86941  ...            NaN
8453                  0.220711    53758.31612  ...       2.304725
8454                  6.852400    57509.00695  ...       5.027748
8455                 18.585305    67656.18517  ...      17.128569
8456                 10.604765    74282.86678  ...      12.744703
...                        ...            ...  ...            ...
134708                2.427170   318434.06300  ...       1.116608
134709                1.647635   321880.71100  ...       1.059711
134710               -4.896451   305856.81080  ...      -6.278376
134711                4.606603   322704.41790  ...       1.808489
134712                4.768623   331114.12122  ...      10.252749

        NewSalesVolume      OldPrice  Old1m%Change  Old12m%Change  \
8452               8.0   53506.48269           NaN            NaN
8453               6.0   52937.00556      0.062285      -1.064314
8454              19.0   56273.00942      1.923260       6.301837
8455              24.0   65990.47116      0.187745      17.268424
8456              27.0   72672.06000     -0.775779      10.125081
...                ...           ...           ...            ...
134708            25.0  298310.88550     -0.622240       2.247240
134709            30.0  299853.11870     -0.125380       0.516989
134710             2.0  282689.11490     -1.880421      -5.724137
134711             5.0  294503.51140     -0.409694       4.179290
134712             3.0  299778.38243     -0.395338       1.791106

        OldSalesVolume  Year  Month  Month_Txt  Region
8452              84.0  1995      1        Jan    True
8453             139.0  1996      1        Jan    True
8454             163.0  1997      1        Jan    True
8455             144.0  1998      1        Jan    True
8456             164.0  1999      1        Jan    True
...                ...   ...    ...        ...     ...
134708           144.0  2017     12        Dec    True
134709            80.0  2018     12        Dec    True
134710           108.0  2019     12        Dec    True
134711            81.0  2020     12        Dec    True
134712            94.0  2021     12        Dec    True

[331 rows x 30 columns]
```

In [30]:

```
#checking if the regex worked fine or not by matching the count of rows.
df_house_clean[(df_house_clean['RegionName'] == 'Slough')]
```

Out[30]:

	Date	RegionName	AveragePrice	AveragePriceSA	SalesVolume	DetachedPrice	Detached1

	Date	RegionName	AveragePrice	AveragePriceSA	SalesVolume	DetachedPrice	Detached1
8452	Jan 1995	Slough	54250.70947	NaN	92.0	101753.14280	
8453	Jan 1996	Slough	53765.05970	NaN	145.0	100394.29790	
8454	Jan 1997	Slough	57060.07900	NaN	182.0	107923.34080	
8455	Jan 1998	Slough	66903.97400	NaN	168.0	129047.76080	
8456	Jan 1999	Slough	73845.26834	NaN	191.0	142742.16030	
...	...	...	...	...	...	...	
134708	Dec 2017	Slough	303048.49620	NaN	169.0	562016.22260	
134709	Dec 2018	Slough	305263.77430	NaN	110.0	572433.90620	
134710	Dec 2019	Slough	287422.74910	NaN	110.0	540593.64090	
134711	Dec 2020	Slough	297700.25250	NaN	86.0	570262.12990	
134712	Dec 2021	Slough	305842.60389	NaN	97.0	602586.09212	

331 rows × 30 columns

In [31]:
```
#Invetigating the null and missing values
df_house_Slough.isna().sum()
```

Out[31]:
```
Date                       0
RegionName                 0
AveragePrice               0
AveragePriceSA           331
SalesVolume                2
DetachedPrice              0
Detached12m%Change        12
SemiDetachedPrice          0
SemiDetached12m%Change    12
TerracedPrice              0
Terraced12m%Change        12
FlatPrice                  0
Flat12m%Change            12
CashPrice                204
Cash12m%Change           216
CashSalesVolume          206
MortgagePrice            204
MortgageSalesVolume      206
NewPrice                   2
New1m%Change               3
New12m%Change             14
NewSalesVolume             3
OldPrice                   2
Old1m%Change               3
Old12m%Change             14
```

```
OldSalesVolume              2
Year                        0
Month                       0
Month_Txt                   0
Region                      0
dtype: int64
```

In [32]:
```
df_house_Slough_1=df_house_Slough.drop(['AveragePriceSA','Region','Month','Month_Txt
```

In [33]:
```
df_house_Slough_1.shape,df_house_Slough.shape
```

Out[33]:
```
((331, 26), (331, 30))
```

In [34]:
```
#checking the type of variables in the file
df_house_Slough_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 331 entries, 8452 to 134712
Data columns (total 26 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Date                    331 non-null    object
 1   RegionName              331 non-null    object
 2   AveragePrice            331 non-null    float64
 3   SalesVolume             329 non-null    float64
 4   DetachedPrice           331 non-null    float64
 5   Detached12m%Change      319 non-null    float64
 6   SemiDetachedPrice       331 non-null    float64
 7   SemiDetached12m%Change  319 non-null    float64
 8   TerracedPrice           331 non-null    float64
 9   Terraced12m%Change      319 non-null    float64
 10  FlatPrice               331 non-null    float64
 11  Flat12m%Change          319 non-null    float64
 12  CashPrice               127 non-null    float64
 13  Cash12m%Change          115 non-null    float64
 14  CashSalesVolume         125 non-null    float64
 15  MortgagePrice           127 non-null    float64
 16  MortgageSalesVolume     125 non-null    float64
 17  NewPrice                329 non-null    float64
 18  New1m%Change            328 non-null    float64
 19  New12m%Change           317 non-null    float64
 20  NewSalesVolume          328 non-null    float64
 21  OldPrice                329 non-null    float64
 22  Old1m%Change            328 non-null    float64
 23  Old12m%Change           317 non-null    float64
 24  OldSalesVolume          329 non-null    float64
 25  Year                    331 non-null    int64
dtypes: float64(23), int64(1), object(2)
memory usage: 69.8+ KB
```

In [35]:
```
df_house_Slough_1.head()
```

Out[35]:

	Date	RegionName	AveragePrice	SalesVolume	DetachedPrice	Detached12m%Change	SemiD
8452	Jan 1995	Slough	54250.70947	92.0	101753.1428	NaN	
8453	Jan 1996	Slough	53765.05970	145.0	100394.2979	-1.335433	

	Date	RegionName	AveragePrice	SalesVolume	DetachedPrice	Detached12m%Change	SemiD
8454	Jan 1997	Slough	57060.07900	182.0	107923.3408	7.499473	
8455	Jan 1998	Slough	66903.97400	168.0	129047.7608	19.573542	
8456	Jan 1999	Slough	73845.26834	191.0	142742.1603	10.611885	

5 rows × 26 columns

In [36]:
```
df_house_Slough_1.describe().T
```

Out[36]:

	count	mean	std	min	25%	
AveragePrice	331.0	181989.897475	81400.282126	53258.093120	119821.260350	179
SalesVolume	329.0	174.227964	77.130549	40.000000	109.000000	
DetachedPrice	331.0	335727.167973	152805.491454	98811.443850	230572.337800	321
Detached12m%Change	319.0	7.229892	8.027284	-16.096452	2.091006	
SemiDetachedPrice	331.0	240321.533046	112018.705449	68094.285670	155569.560900	232
SemiDetached12m%Change	319.0	7.373547	7.963146	-16.282333	2.126288	
TerracedPrice	331.0	189271.528967	88586.489083	53378.385400	118279.921700	185
Terraced12m%Change	319.0	7.349829	7.860908	-16.486292	2.213014	
FlatPrice	331.0	140444.394400	58146.509624	43917.169080	97327.107590	140
Flat12m%Change	319.0	6.454327	8.683234	-17.517264	0.532022	
CashPrice	127.0	246417.668484	42969.600041	170606.107500	201487.096850	271
Cash12m%Change	115.0	5.452897	7.424383	-6.546104	0.515812	
CashSalesVolume	125.0	17.896000	9.557824	3.000000	11.000000	
MortgagePrice	127.0	267993.831641	47564.886520	183887.940800	218760.253600	296
MortgageSalesVolume	125.0	105.752000	40.777060	34.000000	76.000000	
NewPrice	329.0	199794.869499	92989.103747	55738.036900	131118.780200	194
New1m%Change	328.0	0.590615	1.376442	-5.486124	-0.248116	
New12m%Change	317.0	7.460384	8.139943	-17.508592	2.561208	
NewSalesVolume	328.0	20.106707	19.149761	0.000000	7.000000	
OldPrice	329.0	178157.578890	79439.207240	52479.341260	116771.642300	176
Old1m%Change	328.0	0.536772	1.303483	-4.325033	-0.214859	
Old12m%Change	317.0	7.008668	8.175294	-16.623203	1.658571	
OldSalesVolume	329.0	154.182371	67.430747	37.000000	98.000000	
Year	331.0	2008.296073	7.977048	1995.000000	2001.000000	2

In [37]:
```
df_house_Slough_1.isna().sum()
```

Out[37]:
```
Date                      0
RegionName                0
AveragePrice              0
SalesVolume               2
DetachedPrice             0
Detached12m%Change       12
SemiDetachedPrice         0
SemiDetached12m%Change   12
TerracedPrice             0
Terraced12m%Change       12
FlatPrice                 0
Flat12m%Change           12
CashPrice               204
Cash12m%Change          216
CashSalesVolume         206
MortgagePrice           204
MortgageSalesVolume     206
NewPrice                  2
New1m%Change              3
New12m%Change            14
NewSalesVolume            3
OldPrice                  2
Old1m%Change              3
Old12m%Change            14
OldSalesVolume            2
Year                      0
dtype: int64
```

In [38]:
```
df_house_Slough_1.fillna({'Detached12m%Change': 0, 'SemiDetached12m%Change': 0, 'Ter
df_house_Slough_1.isna().sum()
```

Out[38]:
```
Date                      0
RegionName                0
AveragePrice              0
SalesVolume               2
DetachedPrice             0
Detached12m%Change        0
SemiDetachedPrice         0
SemiDetached12m%Change    0
TerracedPrice             0
Terraced12m%Change        0
FlatPrice                 0
Flat12m%Change            0
CashPrice               204
Cash12m%Change          216
CashSalesVolume         206
MortgagePrice           204
MortgageSalesVolume     206
NewPrice                  2
New1m%Change              0
New12m%Change             0
NewSalesVolume            3
OldPrice                  2
Old1m%Change              0
Old12m%Change             0
OldSalesVolume            2
Year                      0
dtype: int64
```

In [39]:
```
s_array = df_house_Slough_1[["AveragePrice", "CashPrice", "MortgagePrice"]].to_numpy
s_array
```

Out[39]:

```
array([[ 54250.70947,          nan,          nan],
       [ 53765.0597 ,          nan,          nan],
       [ 57060.079  ,          nan,          nan],
       [ 66903.974  ,          nan,          nan],
       [ 73845.26834,          nan,          nan],
       [ 86457.45844,          nan,          nan],
       [100376.844  ,          nan,          nan],
       [121934.8468 ,          nan,          nan],
       [143993.5085 ,          nan,          nan],
       [155370.024  ,          nan,          nan],
       [169837.1584 ,          nan,          nan],
       [169853.1337 ,          nan,          nan],
       [187587.5981 ,          nan,          nan],
       [195013.6054 ,          nan,          nan],
       [167831.5487 ,          nan,          nan],
       [178832.1468 ,          nan,          nan],
       [177253.5995 ,          nan,          nan],
       [182958.7854 , 175236.3163 , 189072.8595 ],
       [188416.1473 , 176220.0646 , 190848.3856 ],
       [199731.7954 , 186854.8797 , 202297.8342 ],
       [228621.9191 , 213835.4074 , 231563.8108 ],
       [275182.8108 , 257413.6473 , 278716.4709 ],
       [292226.8089 , 274281.2956 , 295788.4486 ],
       [304627.3558 , 285016.8141 , 308503.2868 ],
       [297426.9419 , 278451.7785 , 301188.8297 ],
       [290350.4888 , 269929.2826 , 294339.3944 ],
       [299359.7142 , 275209.2348 , 303978.4203 ],
       [305225.01433, 279536.87328, 310057.2809 ],
       [ 53258.09312,          nan,          nan],
       [ 54063.10442,          nan,          nan],
       [ 58812.95117,          nan,          nan],
       [ 66665.59662,          nan,          nan],
       [ 74079.93485,          nan,          nan],
       [ 87789.69922,          nan,          nan],
       [101409.7773 ,          nan,          nan],
       [120899.6805 ,          nan,          nan],
       [144986.0458 ,          nan,          nan],
       [154196.4036 ,          nan,          nan],
       [170879.1485 ,          nan,          nan],
       [171896.518  ,          nan,          nan],
       [187419.9534 ,          nan,          nan],
       [194106.7477 ,          nan,          nan],
       [161507.1116 ,          nan,          nan],
       [179388.0384 ,          nan,          nan],
       [179365.8128 ,          nan,          nan],
       [184319.2381 , 174349.2299 , 188067.0546 ],
       [189673.0187 , 177498.6052 , 192096.9096 ],
       [203065.3068 , 190100.8332 , 205643.8763 ],
       [230580.42   , 215821.1948 , 233514.6474 ],
       [280608.4466 , 262589.7737 , 284190.9425 ],
       [295693.5433 , 277432.568  , 299321.219  ],
       [303599.9971 , 284249.6494 , 307433.4533 ],
       [293336.1092 , 274636.5595 , 297044.3747 ],
       [292414.5872 , 271551.8362 , 296478.9878 ],
       [299791.338  , 276412.6286 , 304292.7997 ],
       [303691.29572, 278319.55442, 308472.4297 ],
       [ 53713.57446,          nan,          nan],
       [ 54105.22293,          nan,          nan],
       [ 59839.01966,          nan,          nan],
       [ 68281.84013,          nan,          nan],
       [ 74270.3872 ,          nan,          nan],
       [ 89174.53107,          nan,          nan],
       [101716.5159 ,          nan,          nan],
```

```
[121469.2618 ,          nan,          nan],
[144698.3235 ,          nan,          nan],
[154342.6285 ,          nan,          nan],
[168705.6091 ,          nan,          nan],
[173128.3612 ,          nan,          nan],
[185291.0844 ,          nan,          nan],
[192110.6769 ,          nan,          nan],
[160774.0193 ,          nan,          nan],
[177881.3043 ,          nan,          nan],
[177781.5245 ,          nan,          nan],
[184355.0635 , 173095.0873 , 186574.6561 ],
[191025.0223 , 178945.5479 , 193422.8537 ],
[201925.2734 , 189058.8529 , 204482.6726 ],
[229916.2476 , 215173.3251 , 232847.2503 ],
[277158.3004 , 259366.1922 , 280695.7107 ],
[294945.7024 , 276623.1472 , 298591.9611 ],
[305831.4868 , 286330.7661 , 309697.8207 ],
[289925.3108 , 270723.193  , 293711.4166 ],
[290518.4389 , 269159.5343 , 294660.2762 ],
[299171.4998 , 276445.2917 , 303569.9674 ],
[302189.03573, 277161.41153, 306915.60852],
[ 53929.51043,          nan,          nan],
[ 53964.28354,          nan,          nan],
[ 59931.31551,          nan,          nan],
[ 70220.88324,          nan,          nan],
[ 75188.29233,          nan,          nan],
[ 92202.81589,          nan,          nan],
[104470.1783 ,          nan,          nan],
[122278.8992 ,          nan,          nan],
[144696.947  ,          nan,          nan],
[154893.1185 ,          nan,          nan],
[167004.4538 ,          nan,          nan],
[174141.8104 ,          nan,          nan],
[186909.8411 ,          nan,          nan],
[192269.6241 ,          nan,          nan],
[164416.4216 ,          nan,          nan],
[178210.8758 ,          nan,          nan],
[177660.2639 ,          nan,          nan],
[183850.9344 , 172794.17   , 186023.5509 ],
[192115.9054 , 179932.2764 , 194535.8268 ],
[203525.2586 , 190495.2521 , 206115.9596 ],
[229401.3073 , 214628.9526 , 232338.6615 ],
[281482.7162 , 263613.2933 , 285033.8667 ],
[295422.8926 , 277152.6973 , 299061.934  ],
[300191.1012 , 280790.9516 , 304031.4484 ],
[291423.5139 , 272663.0696 , 295140.3264 ],
[287136.4886 , 265967.3777 , 291239.6805 ],
[292546.9643 , 270640.7767 , 296804.5939 ],
[304184.6841 , 279113.88507, 308925.32942],
[ 54988.17518,          nan,          nan],
[ 54091.03174,          nan,          nan],
[ 61031.76894,          nan,          nan],
[ 73215.69758,          nan,          nan],
[ 77063.18197,          nan,          nan],
[ 95298.77359,          nan,          nan],
[106514.5285 ,          nan,          nan],
[124624.3286 ,          nan,          nan],
[145222.1746 ,          nan,          nan],
[159014.1007 ,          nan,          nan],
[166024.2553 ,          nan,          nan],
[174900.8274 ,          nan,          nan],
[189062.7857 ,          nan,          nan],
[193515.1677 ,          nan,          nan],
[171030.7347 ,          nan,          nan],
```

```
[177932.7468 ,          nan,          nan],
[176520.8756 ,          nan,          nan],
[182229.2658 , 171393.9838 , 184353.3114 ],
[192843.8198 , 180595.6767 , 195277.2868 ],
[207143.0937 , 193723.1388 , 209813.3633 ],
[233070.5081 , 218150.0696 , 236036.6101 ],
[289191.5535 , 270829.1274 , 292840.7045 ],
[292992.5713 , 274902.3213 , 296596.936  ],
[305023.7937 , 285149.6294 , 308954.2746 ],
[292628.103  , 273191.9323 , 296459.1589 ],
[281995.8732 , 261841.5862 , 285921.8404 ],
[293344.3255 , 270903.2247 , 297678.7846 ],
[307816.9549 , 282010.71371, 312675.16914],
[ 55063.83939,          nan,          nan],
[ 54729.68376,          nan,          nan],
[ 61726.96206,          nan,          nan],
[ 73628.64172,          nan,          nan],
[ 77718.07823,          nan,          nan],
[ 98573.2915 ,          nan,          nan],
[109470.2274 ,          nan,          nan],
[127188.5856 ,          nan,          nan],
[146780.5185 ,          nan,          nan],
[161141.4114 ,          nan,          nan],
[168387.3608 ,          nan,          nan],
[177299.3338 ,          nan,          nan],
[193650.3254 ,          nan,          nan],
[195372.8344 ,          nan,          nan],
[170200.25   ,          nan,          nan],
[179974.4533 ,          nan,          nan],
[176783.2476 ,          nan,          nan],
[181741.7151 , 170818.0441 , 183887.9408 ],
[193746.9263 , 181253.7435 , 196236.5804 ],
[212500.2458 , 198559.3004 , 215276.4194 ],
[238478.4978 , 223168.9151 , 241522.3117 ],
[296611.4428 , 277944.529  , 300319.7851 ],
[296634.1528 , 278297.7327 , 300286.7243 ],
[306233.5845 , 286355.8392 , 310166.4823 ],
[295789.8706 , 276377.6555 , 299623.5446 ],
[287755.3078 , 266719.0584 , 291838.3594 ],
[295653.4782 , 272600.1289 , 300081.9131 ],
[317033.67801, 290318.28065, 322056.4448 ],
[ 55018.22507,          nan,          nan],
[ 55378.24733,          nan,          nan],
[ 63806.0449 ,          nan,          nan],
[ 73717.16907,          nan,          nan],
[ 78971.83311,          nan,          nan],
[100175.8462 ,          nan,          nan],
[111385.0033 ,          nan,          nan],
[129405.2346 ,          nan,          nan],
[147248.5254 ,          nan,          nan],
[163239.4883 ,          nan,          nan],
[169442.0917 ,          nan,          nan],
[179076.8257 ,          nan,          nan],
[195919.5558 ,          nan,          nan],
[195386.5075 ,          nan,          nan],
[168778.063  ,          nan,          nan],
[180201.6369 ,          nan,          nan],
[179232.6903 ,          nan,          nan],
[181807.1799 , 170771.1082 , 183979.8947 ],
[195263.4912 , 182637.836  , 197780.9185 ],
[214166.0951 , 199944.6839 , 217000.3182 ],
[247493.0489 , 231588.0614 , 250655.3747 ],
[295356.9807 , 276735.5347 , 299056.5609 ],
[297244.6727 , 278798.5758 , 300916.2673 ],
```

```
[312088.4675 , 291894.7351 , 316085.3395 ],
[305912.5712 , 285329.4767 , 309960.6141 ],
[295396.2307 , 273427.9618 , 299648.5543 ],
[295022.77744, 272111.87331, 299429.00954],
[323859.47843, 296498.50537, 329000.2615 ],
[ 54760.54237,          nan,          nan],
[ 55383.49308,          nan,          nan],
[ 64564.156  ,          nan,          nan],
[ 73323.12988,          nan,          nan],
[ 79592.70104,          nan,          nan],
[100949.2963 ,          nan,          nan],
[113267.7774 ,          nan,          nan],
[132110.826  ,          nan,          nan],
[149103.3607 ,          nan,          nan],
[163621.4953 ,          nan,          nan],
[171120.8129 ,          nan,          nan],
[180516.1838 ,          nan,          nan],
[197279.9391 ,          nan,          nan],
[189568.7688 ,          nan,          nan],
[169786.6842 ,          nan,          nan],
[182500.5938 ,          nan,          nan],
[177912.0514 ,          nan,          nan],
[182438.857  , 171081.6177 , 184686.334  ],
[194454.4672 , 181793.0197 , 196982.4991 ],
[215155.3532 , 200749.7806 , 218027.8085 ],
[252520.7485 , 236146.4785 , 255777.5511 ],
[294966.4868 , 276454.1352 , 298643.7408 ],
[302693.5406 , 283713.5852 , 306463.7646 ],
[312025.4294 , 291969.6425 , 315998.0532 ],
[306112.7381 , 285941.8576 , 310092.9868 ],
[300881.1007 , 277574.5736 , 305364.4981 ],
[295376.33047, 271926.67035, 299858.44732],
[ 54018.42742,          nan,          nan],
[ 55429.52205,          nan,          nan],
[ 65517.72877,          nan,          nan],
[ 73187.5135 ,          nan,          nan],
[ 82519.28884,          nan,          nan],
[100582.4249 ,          nan,          nan],
[114133.2727 ,          nan,          nan],
[134176.4076 ,          nan,          nan],
[149767.2254 ,          nan,          nan],
[166032.1828 ,          nan,          nan],
[170597.2324 ,          nan,          nan],
[179339.3156 ,          nan,          nan],
[195730.4468 ,          nan,          nan],
[183566.9732 ,          nan,          nan],
[169882.7317 ,          nan,          nan],
[181848.257  ,          nan,          nan],
[178019.1397 ,          nan,          nan],
[182818.8274 , 171435.218  , 185071.6307 ],
[195362.2439 , 182675.106  , 197894.0897 ],
[216622.9684 , 202224.4131 , 219492.6987 ],
[254702.1213 , 238287.9108 , 257966.0544 ],
[296229.1702 , 277622.5044 , 299925.2761 ],
[304570.6215 , 285433.3074 , 308370.5659 ],
[307051.3764 , 287183.3427 , 310983.7813 ],
[309763.8288 , 288895.6629 , 313867.3089 ],
[301683.5157 , 278212.11   , 306195.725  ],
[296353.60518, 272863.18932, 300845.47681],
[ 53688.91619,          nan,          nan],
[ 55444.84915,          nan,          nan],
[ 65758.869  ,          nan,          nan],
[ 73877.58193,          nan,          nan],
[ 84652.80352,          nan,          nan],
```

```
[101835.0368 ,          nan,          nan],
[116722.451  ,          nan,          nan],
[137709.2168 ,          nan,          nan],
[152927.113  ,          nan,          nan],
[165999.8042 ,          nan,          nan],
[170119.1807 ,          nan,          nan],
[178623.3156 ,          nan,          nan],
[195955.1161 ,          nan,          nan],
[179127.2106 ,          nan,          nan],
[170040.999  ,          nan,          nan],
[182732.811  ,          nan,          nan],
[179724.1979 ,          nan,          nan],
[182097.0978 , 170606.1075 , 184377.2205 ],
[195788.0815 , 183026.0728 , 198336.7353 ],
[221581.8034 , 206894.2508 , 224508.6074 ],
[258097.886  , 241339.8483 , 261431.2264 ],
[301382.9432 , 282510.817  , 305131.3114 ],
[307597.7173 , 288187.3265 , 311448.696  ],
[308333.9974 , 288340.9024 , 312290.1956 ],
[295841.0659 , 275157.56   , 299884.8199 ],
[300450.6803 , 276451.4594 , 305046.7368 ],
[302391.29423, 277649.13863, 307080.88406],
[ 53545.85941,          nan,          nan],
[ 55237.21734,          nan,          nan],
[ 66162.91132,          nan,          nan],
[ 73777.82492,          nan,          nan],
[ 86098.64414,          nan,          nan],
[100902.1888 ,          nan,          nan],
[119039.8826 ,          nan,          nan],
[140598.8517 ,          nan,          nan],
[153943.9281 ,          nan,          nan],
[166929.1681 ,          nan,          nan],
[167688.3439 ,          nan,          nan],
[181122.7551 ,          nan,          nan],
[196333.5904 ,          nan,          nan],
[175482.1532 ,          nan,          nan],
[170803.1092 ,          nan,          nan],
[180504.4692 ,          nan,          nan],
[181401.7348 ,          nan,          nan],
[184557.5564 , 172858.512  , 186881.077  ],
[196774.8257 , 183914.4091 , 199344.4732 ],
[222751.8561 , 208063.35   , 225677.8528 ],
[261102.954  , 244258.1523 , 264452.6855 ],
[298002.5859 , 279421.4484 , 301692.5596 ],
[305067.379  , 285620.03   , 308918.2244 ],
[305619.1957 , 285797.7534 , 309541.3275 ],
[293043.8031 , 271724.5795 , 297187.2405 ],
[299319.5102 , 275158.3916 , 303939.8188 ],
[307040.65215, 282148.37399, 311770.48821],
[ 53705.56445,          nan,          nan],
[ 56008.05478,          nan,          nan],
[ 66719.55848,          nan,          nan],
[ 74361.28592,          nan,          nan],
[ 86611.64808,          nan,          nan],
[101956.1997 ,          nan,          nan],
[120602.6381 ,          nan,          nan],
[142851.4067 ,          nan,          nan],
[156213.1693 ,          nan,          nan],
[167219.5205 ,          nan,          nan],
[168179.6596 ,          nan,          nan],
[185615.0915 ,          nan,          nan],
[197668.9632 ,          nan,          nan],
[172529.4568 ,          nan,          nan],
[175069.2692 ,          nan,          nan],
```

```
       [180675.1296 ,          nan,          nan],
       [183099.3617 ,          nan,          nan],
       [186407.2562 , 174378.6668 , 188804.639  ],
       [198210.5164 , 185291.8603 , 200790.4105 ],
       [227587.9126 , 212655.7405 , 230561.5091 ],
       [270131.7621 , 252660.8769 , 273606.3415 ],
       [297204.7663 , 278893.8424 , 300839.409  ],
       [303048.4962 , 283454.0943 , 306918.0269 ],
       [305263.7743 , 285747.9509 , 309131.9072 ],
       [287422.7491 , 267042.5936 , 291398.8291 ],
       [297700.2525 , 273367.8749 , 302345.24   ],
       [305842.60389, 280571.72532, 310619.68703]])
```

In [40]:
```
cashpricedifference=(s_array[:,1]-s_array[:,0])/s_array[:,0]
mortgagePricedifference=(s_array[:,2]-s_array[:,0])/s_array[:,0]
cashpricedifferencemean=np.nanmean(cashpricedifference)
mortgagepricedifferencemean=np.nanmean(mortgagePricedifference)
cashpricedifferencemean, mortgagepricedifferencemean
```

Out[40]:
```
(-0.06745991750187741, 0.013521606160163145)
```

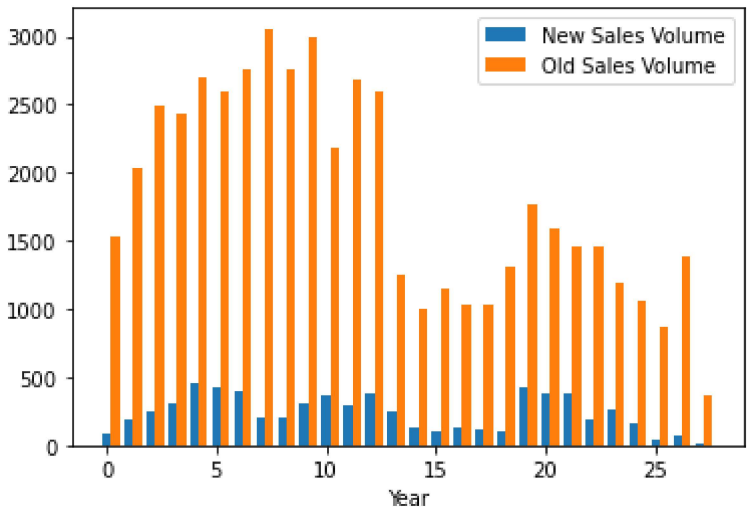
In [41]:
```
Salesbyyear=df_house_Slough_1[["SalesVolume", "NewSalesVolume","OldSalesVolume","Yea
Salesbyyear.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 28 entries, 0 to 27
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Year            28 non-null     int64  
 1   SalesVolume     28 non-null     float64
 2   NewSalesVolume  28 non-null     float64
 3   OldSalesVolume  28 non-null     float64
dtypes: float64(3), int64(1)
memory usage: 1.0 KB
```

In [42]:
```
plt.plot(Salesbyyear["Year"],Salesbyyear["SalesVolume"])
plt.show()
```

In [43]:
```
w=.4

x=Salesbyyear["Year"]
bar1=np.arange(len(x))
bar2= [i+w for i in bar1]
```

```
#bar3= [i+w for i in bar1]

plt.bar(bar1,Salesbyyear["NewSalesVolume"],w,label="New Sales Volume")
plt.bar(bar2,Salesbyyear["OldSalesVolume"],w,label="Old Sales Volume")
#plt.bar(bar3,Salesbyyear["OldSalesVolume"],w,label="Old Sales Volume")

plt.xlabel("Year")
#plt.xticks(bar1,x)
plt.legend()
plt.show()
```



In [44]:

```
Pricebyyear=df_house_Slough_1[["AveragePrice", "DetachedPrice","SemiDetachedPrice","
Pricebyyear.info()
Pricebyyear
```

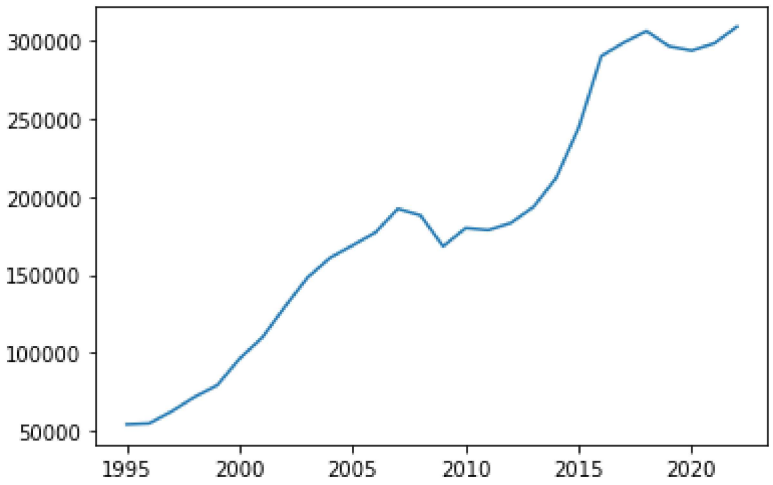
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 28 entries, 0 to 27
Data columns (total 6 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Year               28 non-null     int64
 1   AveragePrice       28 non-null     float64
 2   DetachedPrice      28 non-null     float64
 3   SemiDetachedPrice  28 non-null     float64
 4   TerracedPrice      28 non-null     float64
 5   FlatPrice          28 non-null     float64
dtypes: float64(5), int64(1)
memory usage: 1.4 KB
```

Out[44]:

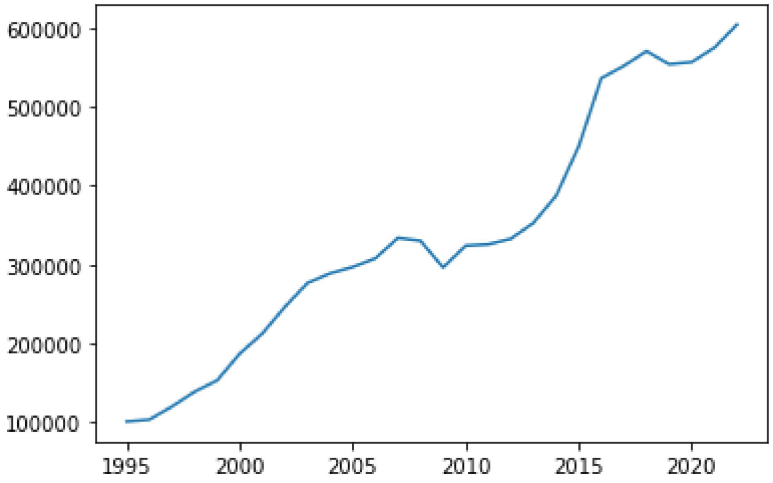
	Year	AveragePrice	DetachedPrice	SemiDetachedPrice	TerracedPrice	FlatPrice
0	1995	54161.786413	100538.118998	69474.983440	54219.500076	44585.957496
1	1996	54799.980818	102940.343492	70888.995454	55028.761208	44499.721500
2	1997	62577.613734	119528.087733	81881.157897	63211.500882	49832.256721
3	1998	71763.428209	138324.702242	94371.693320	72607.313592	56712.303383
4	1999	79217.671804	152757.317433	103830.643342	79915.948555	62929.691358
5	2000	96324.796868	186602.137292	126332.162917	96426.310022	76861.101442
6	2001	109925.758042	212331.906692	143278.867692	109376.793931	88532.746107
7	2002	129603.962158	246253.450025	167406.897325	127349.891025	106311.346652
8	2003	148298.403317	276464.335242	191232.577442	145542.523567	122363.333658

	Year	AveragePrice	DetachedPrice	SemiDetachedPrice	TerracedPrice	FlatPrice
9	2004	160999.945492	288706.654617	205977.275025	160075.703667	133387.907133
10	2005	168998.775592	296762.710192	214843.999525	170174.991875	139938.316442
11	2006	177126.122650	307689.086850	224683.739050	180308.510958	146003.844792
12	2007	192400.766625	333757.574558	244071.422767	197343.872492	157749.418058
13	2008	188170.810525	330077.357192	239362.994758	193431.366708	153378.821800
14	2009	168343.411850	296220.029875	215690.077908	173699.665567	135528.973800
15	2010	180056.871908	323929.702733	234648.709033	187803.783800	140183.140625
16	2011	178729.541642	325570.978250	234128.406050	186511.196067	137851.293033
17	2012	183298.481417	332618.208100	241113.447042	192558.642600	139834.242808
18	2013	193639.538808	352938.218075	256018.517442	204013.175600	146190.341975
19	2014	212146.413542	387413.037367	281124.051467	223958.477892	159765.518525
20	2015	244509.785058	450025.052267	325649.127692	257485.687983	183721.886658
21	2016	290281.516950	536551.343217	386711.677908	305066.857475	218288.073575
22	2017	299011.508217	552290.869333	396889.306317	312270.723417	226400.059400
23	2018	306324.129983	570838.321017	411192.337142	322944.773383	228815.747008
24	2019	296552.217133	554361.605217	400436.274667	314042.434733	220171.696558
25	2020	293800.206217	557063.906517	402079.935642	315398.186425	214019.854550
26	2021	298491.215280	575221.184817	412599.203471	323391.405785	213591.818576
27	2022	309142.877317	604625.389586	433629.379640	336997.156646	216814.511814

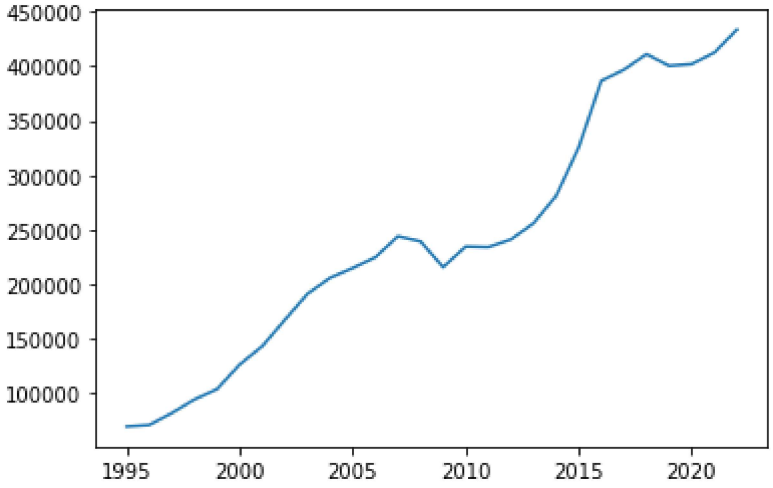
In [45]:
```
plt.plot(Pricebyyear["Year"],Pricebyyear["AveragePrice"])
plt.show()
```



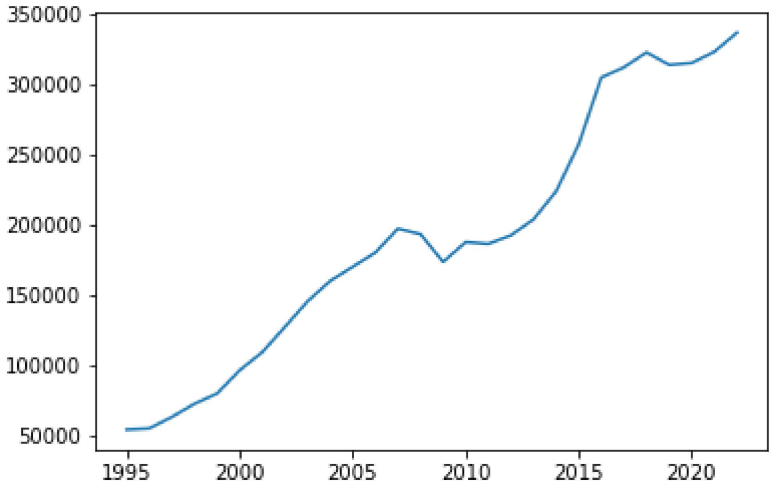
In [46]:
```
plt.plot(Pricebyyear["Year"],Pricebyyear["DetachedPrice"])
plt.show()
```



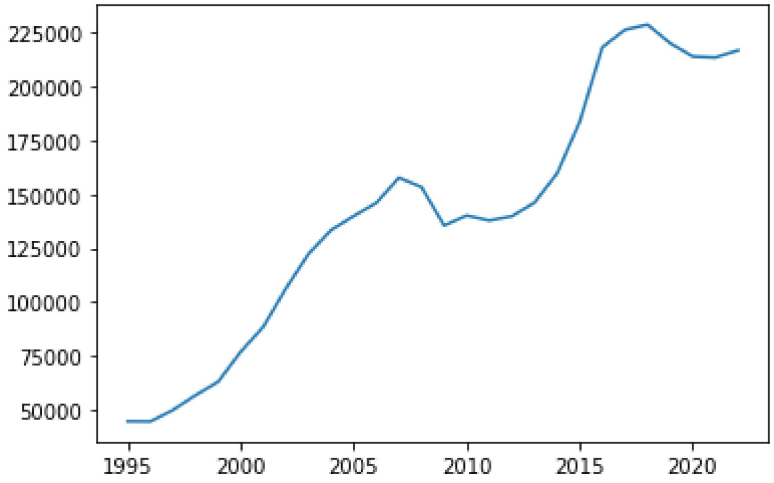
In [47]:
```
plt.plot(Pricebyyear["Year"],Pricebyyear["SemiDetachedPrice"])
plt.show()
```



In [48]:
```
plt.plot(Pricebyyear["Year"],Pricebyyear["TerracedPrice"])
plt.show()
```



In [49]:
```
plt.plot(Pricebyyear["Year"],Pricebyyear["FlatPrice"])
plt.show()
```



In [50]:

```
#keeping the Cashprice, cash12m%change, CashSalesVolume,MortgagePrice, MortgageSales
df_house_Slough_1['SalesVolume']=df_house_Slough_1['SalesVolume'].fillna(method="bfi
#df_house_Slough_1['NewPrice']=df_house_Slough_1['NewPrice'].fillna(df_house_Slough_
df_house_Slough_1['NewPrice']=df_house_Slough_1['NewPrice'].fillna(method="bfill")
df_house_Slough_1['NewSalesVolume']=df_house_Slough_1['NewSalesVolume'].fillna(metho
df_house_Slough_1['OldPrice']=df_house_Slough_1['OldPrice'].fillna(method="bfill")
df_house_Slough_1['OldSalesVolume']=df_house_Slough_1['OldSalesVolume'].fillna(metho
df_house_Slough_1.isna().sum()
```

Out[50]:

```
Date                     0
RegionName               0
AveragePrice             0
SalesVolume              0
DetachedPrice            0
Detached12m%Change       0
SemiDetachedPrice        0
SemiDetached12m%Change   0
TerracedPrice            0
Terraced12m%Change       0
FlatPrice                0
Flat12m%Change           0
CashPrice              204
Cash12m%Change         216
CashSalesVolume        206
MortgagePrice          204
MortgageSalesVolume    206
NewPrice                 0
New1m%Change             0
New12m%Change            0
NewSalesVolume           0
OldPrice                 0
Old1m%Change             0
Old12m%Change            0
OldSalesVolume           0
Year                     0
dtype: int64
```

In [51]:

```
# Slough Crime Rate from UKCrimeStates.com by webscrpaing
url = 'https://www.ukcrimestats.com/Constituency/65680'
page = requests.get(url)
Crime = pd.read_html(url)
Crime[0]
```

Out[51]:

	Unnamed: 0	ASB	Burglary	Robbery	Vehicle	Violent	Shoplifting	CD&A	Other Theft	Drugs	Bik Thef
0	Jul 2022	193	49	14	129	700	41	122	115	27	3
1	Jun 2022	158	56	12	94	612	41	127	101	26	2
2	May 2022	184	53	12	127	630	61	133	103	44	2
3	Apr 2022	163	41	18	128	534	40	128	99	20	2
4	Mar 2022	127	66	12	153	592	39	152	113	27	2
...	...	...	...	...	...	...	...	...	...	...	.
135	Apr 2011	613	118	24	157	213	580	580	580	580	58
136	Mar 2011	535	139	25	171	224	585	585	585	585	58
137	Feb 2011	514	157	30	141	195	563	563	563	563	56
138	Jan 2011	206	180	35	156	210	486	486	486	486	48
139	Dec 2010	437	153	36	162	242	479	479	479	479	47

140 rows × 16 columns

In [52]:
```
Slough_Crime=pd.DataFrame(Crime[0])
```

In [53]:
```
Slough_Crime.head()
```

Out[53]:

	Unnamed: 0	ASB	Burglary	Robbery	Vehicle	Violent	Shoplifting	CD&A	Other Theft	Drugs	Bike Theft
0	Jul 2022	193	49	14	129	700	41	122	115	27	33
1	Jun 2022	158	56	12	94	612	41	127	101	26	29
2	May 2022	184	53	12	127	630	61	133	103	44	26
3	Apr 2022	163	41	18	128	534	40	128	99	20	23
4	Mar 2022	127	66	12	153	592	39	152	113	27	23

In [54]:
```
Slough_Crime.describe()
```

Out[54]:

	ASB	Burglary	Robbery	Vehicle	Violent	Shoplifting	CD&A	Oth Th
count	140.000000	140.000000	140.000000	140.000000	140.000000	140.000000	140.000000	140.000
mean	264.507143	84.714286	16.628571	126.528571	324.507143	101.257143	145.442857	138.542
std	99.140225	36.552571	6.759318	33.808271	149.263684	121.987107	110.335086	113.934
min	93.000000	28.000000	3.000000	48.000000	113.000000	24.000000	47.000000	46.000

	ASB	Burglary	Robbery	Vehicle	Violent	Shoplifting	CD&A	Oth Th
25%	193.750000	57.000000	12.000000	107.000000	204.000000	52.000000	106.750000	95.000
50%	247.500000	76.000000	16.000000	124.500000	269.000000	77.000000	120.000000	104.000
75%	304.250000	104.000000	20.000000	146.000000	445.750000	88.000000	133.250000	126.500
max	613.000000	201.000000	36.000000	256.000000	713.000000	625.000000	625.000000	625.000

In [55]:
```
Slough_Crime.describe().T
```

Out[55]:

	count	mean	std	min	25%	50%	75%	max
ASB	140.0	264.507143	99.140225	93.0	193.75	247.5	304.25	613.0
Burglary	140.0	84.714286	36.552571	28.0	57.00	76.0	104.00	201.0
Robbery	140.0	16.628571	6.759318	3.0	12.00	16.0	20.00	36.0
Vehicle	140.0	126.528571	33.808271	48.0	107.00	124.5	146.00	256.0
Violent	140.0	324.507143	149.263684	113.0	204.00	269.0	445.75	713.0
Shoplifting	140.0	101.257143	121.987107	24.0	52.00	77.0	88.00	625.0
CD&A	140.0	145.442857	110.335086	47.0	106.75	120.0	133.25	625.0
Other Theft	140.0	138.542857	113.934562	46.0	95.00	104.0	126.50	625.0
Drugs	140.0	88.121429	132.214915	15.0	30.00	38.5	53.00	625.0
Bike Theft	140.0	83.521429	134.177830	9.0	23.00	32.0	53.25	625.0
Theft From the Person	140.0	56.057143	132.067335	8.0	17.75	22.0	27.25	625.0
Weapons	140.0	50.200000	134.027174	0.0	8.00	11.5	18.00	625.0
Public Order	140.0	98.221429	132.440190	15.0	32.00	47.5	91.50	625.0
Other	140.0	53.778571	132.967355	4.0	12.75	17.0	24.25	625.0
Total	140.0	1292.607143	207.881008	728.0	1133.75	1294.0	1443.25	1760.0

In [56]:
```
Slough_Crime.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 140 entries, 0 to 139
Data columns (total 16 columns):
 #   Column                Non-Null Count  Dtype 
---  ------                --------------  ----- 
 0   Unnamed: 0            140 non-null    object
 1   ASB                   140 non-null    int64 
 2   Burglary              140 non-null    int64 
 3   Robbery               140 non-null    int64 
 4   Vehicle               140 non-null    int64 
 5   Violent               140 non-null    int64 
 6   Shoplifting           140 non-null    int64 
 7   CD&A                  140 non-null    int64 
 8   Other Theft           140 non-null    int64 
 9   Drugs                 140 non-null    int64 
 10  Bike Theft            140 non-null    int64 
```

```
 11  Theft From the Person  140 non-null    int64
 12  Weapons                140 non-null    int64
 13  Public Order           140 non-null    int64
 14  Other                  140 non-null    int64
 15  Total                  140 non-null    int64
dtypes: int64(15), object(1)
memory usage: 17.6+ KB
```

In [57]:
```
Slough_Crime.isna().sum()
```

Out[57]:
```
Unnamed: 0              0
ASB                     0
Burglary                0
Robbery                 0
Vehicle                 0
Violent                 0
Shoplifting             0
CD&A                    0
Other Theft             0
Drugs                   0
Bike Theft              0
Theft From the Person   0
Weapons                 0
Public Order            0
Other                   0
Total                   0
dtype: int64
```

In [58]:
```
Slough_Crime_1 = Slough_Crime.rename(columns={'Unnamed: 0': 'Date','Total': 'Total C
```

In [59]:
```
Slough_Crime_1.head()
```

Out[59]:

	Date	ASB	Burglary	Robbery	Vehicle	Violent	Shoplifting	CD&A	Other Theft	Drugs	Bike Theft	The Fro tl Persc
0	Jul 2022	193	49	14	129	700	41	122	115	27	33	
1	Jun 2022	158	56	12	94	612	41	127	101	26	29	
2	May 2022	184	53	12	127	630	61	133	103	44	26	
3	Apr 2022	163	41	18	128	534	40	128	99	20	23	
4	Mar 2022	127	66	12	153	592	39	152	113	27	23	

In [60]:
```
#Keeping only required variables.
Slough_Crime_Clean = pd.DataFrame(Slough_Crime_1,  columns=['Date','Total Crime'])
Slough_Crime_Clean
```

Out[60]:

	Date	Total Crime
0	Jul 2022	1675

	Date	Total Crime
1	Jun 2022	1527
2	May 2022	1625
3	Apr 2022	1412
4	Mar 2022	1537
...	...	...
135	Apr 2011	1705
136	Mar 2011	1679
137	Feb 2011	1600
138	Jan 2011	1273
139	Dec 2010	1509

140 rows × 2 columns

In [61]:
```
#reading in UK Salary Data
Salary=pd.read_csv('UK Average Salary.csv')
Salary.head()
```

Out[61]:

	Date	Average Salary
0	Jan 2000	1332
1	Feb 2000	1302
2	Mar 2000	1343
3	Apr 2000	1343
4	May 2000	1353

In [62]:
```
#reading in CPI Data
CPI=pd.read_csv('CPI Index.csv')
CPI.head()
```

Out[62]:

	Date	Average Households CPI
0	Jan 2006	2.302169
1	Feb 2006	2.263099
2	Mar 2006	1.985953
3	Apr 2006	2.335195
4	May 2006	2.351207

In [63]:
```
df_house_Slough_2=pd.DataFrame(df_house_Slough_1,  columns=['Date','AveragePrice','S
df_house_Slough_2.head()
df_house_Slough_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 331 entries, 8452 to 134712
Data columns (total 6 columns):
```

```
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Date            331 non-null    object
 1   AveragePrice    331 non-null    float64
 2   SalesVolume     331 non-null    float64
 3   NewSalesVolume  331 non-null    float64
 4   OldSalesVolume  331 non-null    float64
 5   Year            331 non-null    int64
dtypes: float64(4), int64(1), object(1)
memory usage: 18.1+ KB
```

In [64]:
```
House_Final_1=pd.merge(df_house_Slough_2, Slough_Crime_Clean, how='left',left_on='Da
House_Final_1.head()
House_Final_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 331 entries, 0 to 330
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Date            331 non-null    object
 1   AveragePrice    331 non-null    float64
 2   SalesVolume     331 non-null    float64
 3   NewSalesVolume  331 non-null    float64
 4   OldSalesVolume  331 non-null    float64
 5   Year            331 non-null    int64
 6   Total Crime     140 non-null    float64
dtypes: float64(5), int64(1), object(1)
memory usage: 20.7+ KB
```

In [65]:
```
House_Final_2=pd.merge(House_Final_1, CPI, how='left',left_on='Date', right_on='Date
House_Final_2.head()
House_Final_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 331 entries, 0 to 330
Data columns (total 8 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Date                    331 non-null    object
 1   AveragePrice            331 non-null    float64
 2   SalesVolume             331 non-null    float64
 3   NewSalesVolume          331 non-null    float64
 4   OldSalesVolume          331 non-null    float64
 5   Year                    331 non-null    int64
 6   Total Crime             140 non-null    float64
 7   Average Households CPI  199 non-null    float64
dtypes: float64(6), int64(1), object(1)
memory usage: 23.3+ KB
```

In [66]:
```
House_Final_3=pd.merge(House_Final_2, Salary, how='left',left_on='Date', right_on='D
```

In [67]:
```
House_Final_3.head(10)
```

Out[67]:

	Date	AveragePrice	SalesVolume	NewSalesVolume	OldSalesVolume	Year	Total Crime	Average Households CPI
0	Jan 1995	54250.70947	92.0	8.0	84.0	1995	NaN	NaN

	Date	AveragePrice	SalesVolume	NewSalesVolume	OldSalesVolume	Year	Total Crime	Average Households CPI
1	Jan 1996	53765.05970	145.0	6.0	139.0	1996	NaN	NaN
2	Jan 1997	57060.07900	182.0	19.0	163.0	1997	NaN	NaN
3	Jan 1998	66903.97400	168.0	24.0	144.0	1998	NaN	NaN
4	Jan 1999	73845.26834	191.0	27.0	164.0	1999	NaN	NaN
5	Jan 2000	86457.45844	207.0	15.0	192.0	2000	NaN	NaN
6	Jan 2001	100376.84400	187.0	9.0	178.0	2001	NaN	NaN
7	Jan 2002	121934.84680	181.0	5.0	176.0	2002	NaN	NaN
8	Jan 2003	143993.50850	218.0	1.0	217.0	2003	NaN	NaN
9	Jan 2004	155370.02400	228.0	11.0	217.0	2004	NaN	NaN

In [68]:
```
House_Final_3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 331 entries, 0 to 330
Data columns (total 9 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Date                    331 non-null    object 
 1   AveragePrice            331 non-null    float64
 2   SalesVolume             331 non-null    float64
 3   NewSalesVolume          331 non-null    float64
 4   OldSalesVolume          331 non-null    float64
 5   Year                    331 non-null    int64  
 6   Total Crime             140 non-null    float64
 7   Average Households CPI  199 non-null    float64
 8   Average Salary          271 non-null    float64
dtypes: float64(7), int64(1), object(1)
memory usage: 25.9+ KB
```

In [69]:
```
House_Final_3.isna().sum()
```

Out[69]:
```
Date                      0
AveragePrice              0
SalesVolume               0
NewSalesVolume            0
OldSalesVolume            0
Year                      0
Total Crime             191
Average Households CPI  132
Average Salary           60
dtype: int64
```

In [70]:
```
House_Final_3['Average Salary']=House_Final_3['Average Salary'].fillna(House_Final_3
#House_Final_4=House_Final_3.dropna()
#House_Final_4 = pd.DataFrame(House_Final_3[(House_Final_3['Date'] >= Jan 2006)])

#df_house_Slough = pd.DataFrame(df_house_clean[(df_house_clean['Region']==True)])
```

In [71]:
```
House_Final_4=House_Final_3[["AveragePrice","SalesVolume","Total Crime","Average Hou
House_Final_4_Clean=House_Final_4[(House_Final_4['Year'] >= 2005)]
House_Final_4_Clean.info()
House_Final_4_Clean
```

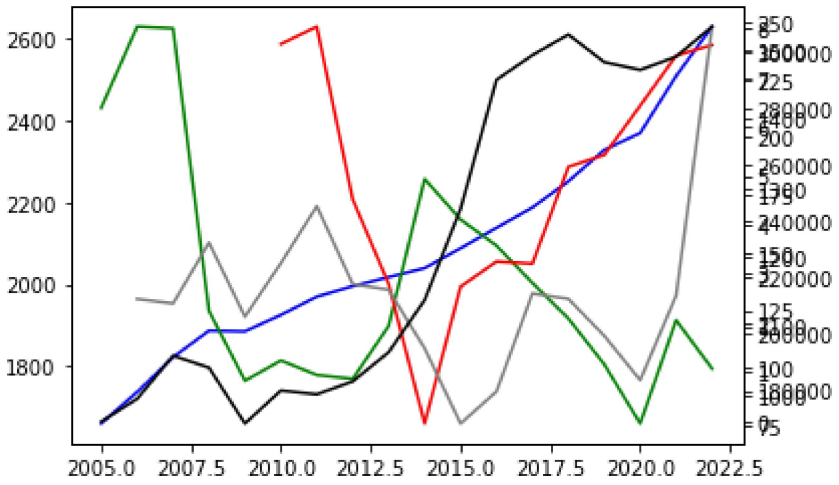
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 18 entries, 10 to 27
Data columns (total 6 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Year                    18 non-null     int64  
 1   AveragePrice            18 non-null     float64
 2   SalesVolume             18 non-null     float64
 3   Total Crime             13 non-null     float64
 4   Average Households CPI  17 non-null     float64
 5   Average Salary          18 non-null     float64
dtypes: float64(5), int64(1)
memory usage: 1008.0 bytes
```

Out[71]:

	Year	AveragePrice	SalesVolume	Total Crime	Average Households CPI	Average Salary
10	2005	168998.775592	212.833333	NaN	NaN	1659.750000
11	2006	177126.122650	248.083333	NaN	2.491458	1737.000000
12	2007	192400.766625	247.333333	NaN	2.399556	1822.000000
13	2008	188170.810525	124.750000	NaN	3.641056	1886.916667
14	2009	168343.411850	94.500000	NaN	2.130365	1885.166667
15	2010	180056.871908	103.250000	1509.000000	3.223202	1925.500000
16	2011	178729.541642	97.000000	1534.250000	4.384076	1970.000000
17	2012	183298.481417	95.333333	1284.166667	2.794082	1995.666667
18	2013	193639.538808	118.250000	1161.833333	2.680253	2017.750000
19	2014	212146.413542	182.000000	959.750000	1.485698	2039.833333
20	2015	244509.785058	164.416667	1157.833333	-0.046990	2088.500000
21	2016	290281.516950	153.083333	1194.250000	0.606583	2138.333333
22	2017	299011.508217	136.916667	1191.166667	2.603825	2188.166667
23	2018	306324.129983	121.500000	1331.333333	2.495083	2252.166667
24	2019	296552.217133	101.583333	1348.500000	1.740821	2328.916667
25	2020	293800.206217	75.916667	1419.833333	0.837116	2370.333333
26	2021	298491.215280	120.750000	1492.333333	2.561035	2509.750000
27	2022	309142.877317	99.714286	1507.571429	8.042857	2630.428571

In [72]:
```
fig,ax1 = plt.subplots()
Years=House_Final_4_Clean["Year"]
AverageSalary=House_Final_4_Clean["Average Salary"]
```

```
SalesVolume=House_Final_4_Clean["SalesVolume"]
TotalCrime=House_Final_4_Clean["Total Crime"]
AverageHouseholdsCPI=House_Final_4_Clean["Average Households CPI"]
AveragePrice=House_Final_4_Clean["AveragePrice"]

ax1.plot(Years,AverageSalary,color="blue")

ax2=ax1.twinx()
ax2.plot(Years,SalesVolume,color="green")

ax3=ax1.twinx()
ax3.plot(Years,TotalCrime,color="red")

ax4=ax1.twinx()
ax4.plot(Years,AverageHouseholdsCPI,color="grey")

ax5=ax1.twinx()
ax5.plot(Years,AveragePrice,color="black")

plt.show()
```



Machine Learning

In [73]:
```
import pandas as pd
import numpy as np
import sklearn
from sklearn import linear_model
from sklearn.utils import shuffle
```

In [74]:
```
#Reading the base data file
df_student=pd.read_csv('student-mat.csv',sep=";")
```

In [75]:
```
df_student.head()
```

Out[75]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	freetim
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	freetim
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	

5 rows × 33 columns

In [76]:
```
df_student.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 395 entries, 0 to 394
Data columns (total 33 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   school      395 non-null    object
 1   sex         395 non-null    object
 2   age         395 non-null    int64 
 3   address     395 non-null    object
 4   famsize     395 non-null    object
 5   Pstatus     395 non-null    object
 6   Medu        395 non-null    int64 
 7   Fedu        395 non-null    int64 
 8   Mjob        395 non-null    object
 9   Fjob        395 non-null    object
 10  reason      395 non-null    object
 11  guardian    395 non-null    object
 12  traveltime  395 non-null    int64 
 13  studytime   395 non-null    int64 
 14  failures    395 non-null    int64 
 15  schoolsup   395 non-null    object
 16  famsup      395 non-null    object
 17  paid        395 non-null    object
 18  activities  395 non-null    object
 19  nursery     395 non-null    object
 20  higher      395 non-null    object
 21  internet    395 non-null    object
 22  romantic    395 non-null    object
 23  famrel      395 non-null    int64 
 24  freetime    395 non-null    int64 
 25  goout       395 non-null    int64 
 26  Dalc        395 non-null    int64 
 27  Walc        395 non-null    int64 
 28  health      395 non-null    int64 
 29  absences    395 non-null    int64 
 30  G1          395 non-null    int64 
 31  G2          395 non-null    int64 
 32  G3          395 non-null    int64 
dtypes: int64(16), object(17)
memory usage: 102.0+ KB
```

In [77]:
```
df_student.describe()
```

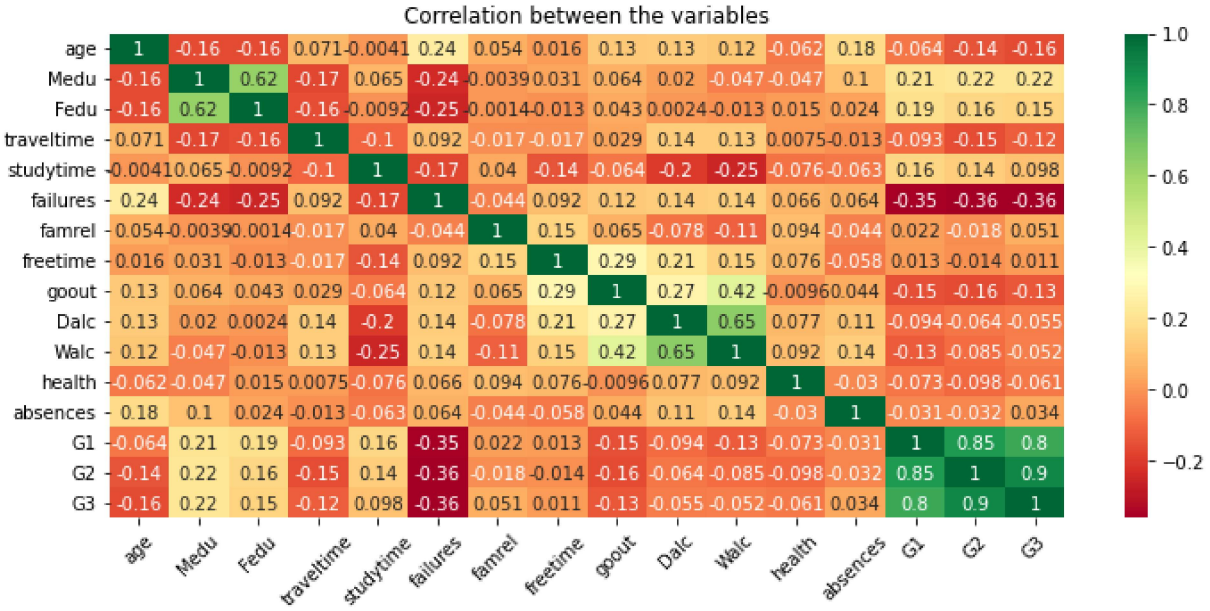
Out[77]:

	age	Medu	Fedu	traveltime	studytime	failures	famrel	freetim
count	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.0000
mean	16.696203	2.749367	2.521519	1.448101	2.035443	0.334177	3.944304	3.2354
std	1.276043	1.094735	1.088201	0.697505	0.839240	0.743651	0.896659	0.9988

	age	Medu	Fedu	traveltime	studytime	failures	famrel	freetin
min	15.000000	0.000000	0.000000	1.000000	1.000000	0.000000	1.000000	1.0000
25%	16.000000	2.000000	2.000000	1.000000	1.000000	0.000000	4.000000	3.0000
50%	17.000000	3.000000	2.000000	1.000000	2.000000	0.000000	4.000000	3.0000
75%	18.000000	4.000000	3.000000	2.000000	2.000000	0.000000	5.000000	4.0000
max	22.000000	4.000000	4.000000	4.000000	4.000000	3.000000	5.000000	5.0000

In [78]:

```
plt.subplots(figsize=(12,5))
table_correlation=df_student.corr()
sns.heatmap(table_correlation,annot=True,cmap='RdYlGn')
plt.title('Correlation between the variables')
plt.xticks(rotation=45)
plt.savefig('Grading Corelation')
```

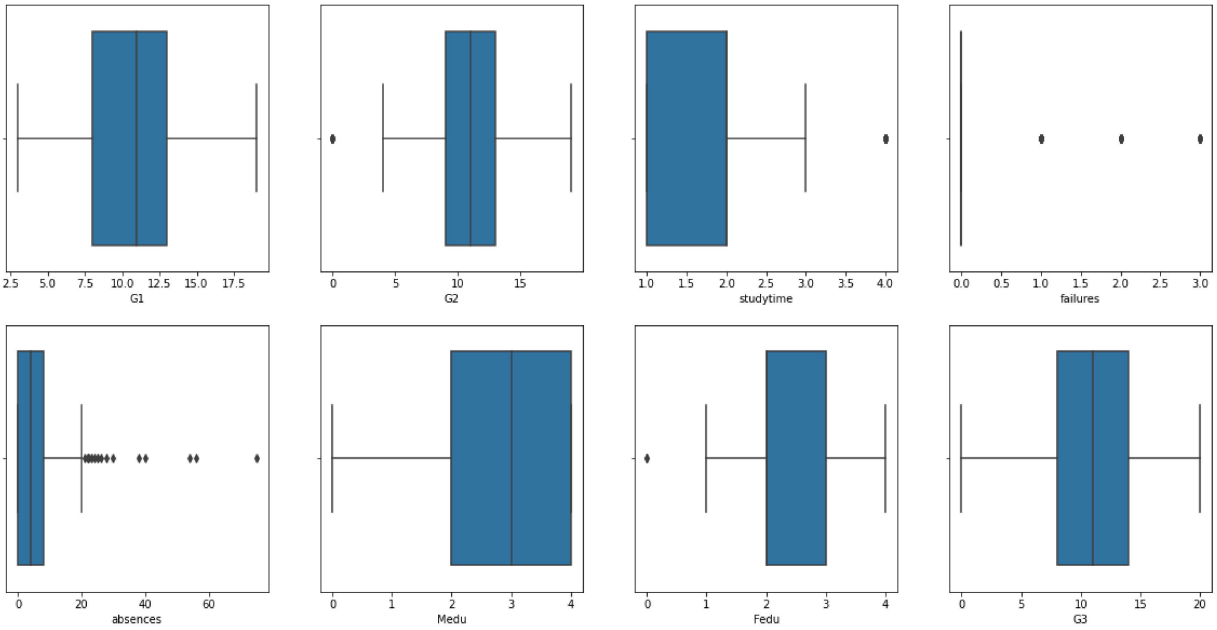


In [97]:

```
#keeping the only varibales that we would use
Student= df_student[["G1", "G2", "studytime", "failures", "absences","Medu","Fedu","
```

In [98]:

```
#Urivariate Analysis

columns = Student.columns
counter = 1

f = plt.figure(figsize=(20,10))

for col in Student[0:-1]:
    f.add_subplot(2,4,counter)
    sns.boxplot(x=Student[col])
    counter = counter + 1
```



In [99]:

```
outlier_column_name = ["G1", "G2", "studytime", "failures", "absences","Medu","Fedu"
outlier_column_upper_limit = []
outlier_column_lower_limit = []

#Calculate upper and lower limits for outliers
for c in outlier_column_name:
    upper_limit = Student[c].quantile(0.995)
    outlier_column_upper_limit.append(upper_limit)
    lower_limit = Student[c].quantile(0.005)
    outlier_column_lower_limit.append(lower_limit)
    print(c, "upper limit: ",upper_limit)
    print(c, "lower limit: ", lower_limit)
    print(" # upper outliers:" , Student[Student[c] > upper_limit].shape[0] )
    print(" # lower outliers:" ,Student[Student[c] < lower_limit].shape[0] )
    print("")

#Remove outliers
for x in range(len(outlier_column_name)):
    Student_1 = Student.loc[(Student[outlier_column_name[x]] <= outlier_column_upper
                (Student[outlier_column_name[x]] >= outlier_column_lower_limit[x]) ]
```

```
G1 upper limit:  19.0
G1 lower limit:  4.97
 # upper outliers: 0
 # lower outliers: 2

G2 upper limit:  19.0
G2 lower limit:  0.0
 # upper outliers: 0
 # lower outliers: 0

studytime upper limit:  4.0
studytime lower limit:  1.0
 # upper outliers: 0
 # lower outliers: 0

failures upper limit:  3.0
failures lower limit:  0.0
 # upper outliers: 0
 # lower outliers: 0

absences upper limit:  54.059999999999945
```

```
absences lower limit:  0.0
 # upper outliers: 2
 # lower outliers: 0

Medu upper limit:  4.0
Medu lower limit:  0.0
 # upper outliers: 0
 # lower outliers: 0

Fedu upper limit:  4.0
Fedu lower limit:  0.97
 # upper outliers: 0
 # lower outliers: 2

G3 upper limit:  19.0
G3 lower limit:  0.0
 # upper outliers: 1
 # lower outliers: 0
```

In [100…

```
Student.head(10)
```

Out[100…

	G1	G2	studytime	failures	absences	Medu	Fedu	G3
0	5	6	2	0	6	4	4	6
1	5	5	2	0	4	1	1	6
2	7	8	2	3	10	1	1	10
3	15	14	3	0	2	4	2	15
4	6	10	2	0	4	3	3	10
5	15	15	2	0	10	4	3	15
6	12	12	2	0	0	2	2	11
7	6	5	2	0	6	4	4	6
8	16	18	2	0	0	3	2	19
9	14	15	2	0	0	3	4	15

In [101…

```
Student.describe()
```

Out[101…

	G1	G2	studytime	failures	absences	Medu	Fedu	G
count	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.0000
mean	10.908861	10.713924	2.035443	0.334177	5.708861	2.749367	2.521519	10.4151
std	3.319195	3.761505	0.839240	0.743651	8.003096	1.094735	1.088201	4.5814
min	3.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.0000
25%	8.000000	9.000000	1.000000	0.000000	0.000000	2.000000	2.000000	8.0000
50%	11.000000	11.000000	2.000000	0.000000	4.000000	3.000000	2.000000	11.0000
75%	13.000000	13.000000	2.000000	0.000000	8.000000	4.000000	3.000000	14.0000
max	19.000000	19.000000	4.000000	3.000000	75.000000	4.000000	4.000000	20.0000

In [102…
```
Student_1.head(10)
```

Out[102…

	G1	G2	studytime	failures	absences	Medu	Fedu	G3
0	5	6	2	0	6	4	4	6
1	5	5	2	0	4	1	1	6
2	7	8	2	3	10	1	1	10
3	15	14	3	0	2	4	2	15
4	6	10	2	0	4	3	3	10
5	15	15	2	0	10	4	3	15
6	12	12	2	0	0	2	2	11
7	6	5	2	0	6	4	4	6
8	16	18	2	0	0	3	2	19
9	14	15	2	0	0	3	4	15

In [103…
```
Student_1.describe()
```

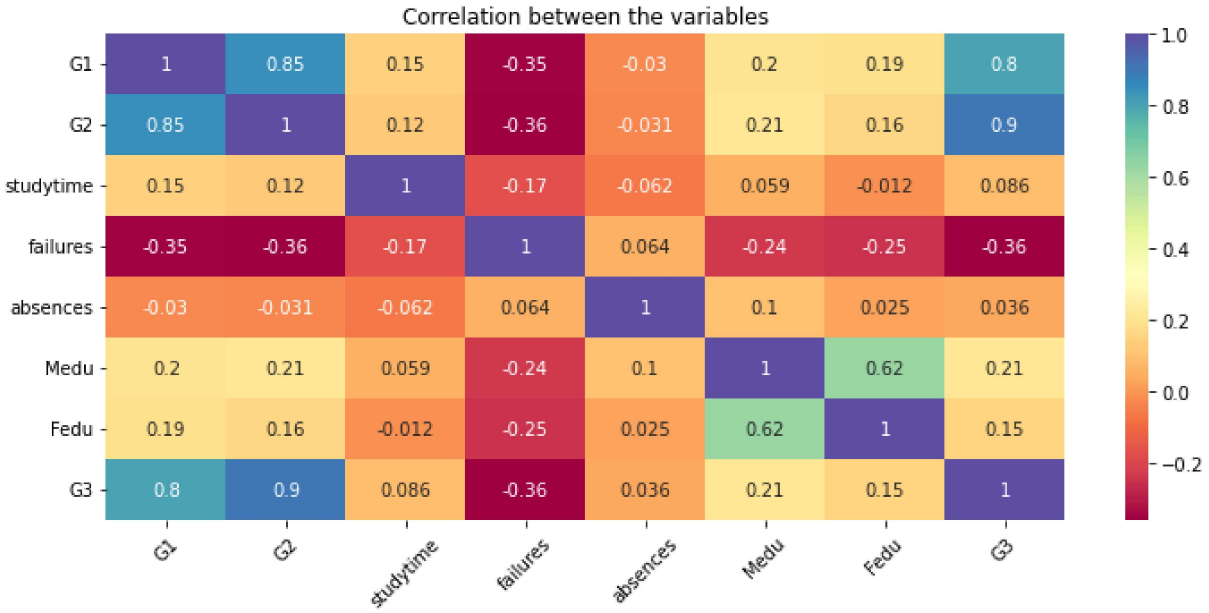
Out[103…

	G1	G2	studytime	failures	absences	Medu	Fedu	G
count	394.000000	394.000000	394.000000	394.000000	394.000000	394.000000	394.000000	394.0000
mean	10.888325	10.692893	2.030457	0.335025	5.713198	2.746193	2.520305	10.3908
std	3.298194	3.742963	0.834429	0.744405	8.012806	1.094306	1.089316	4.5616
min	3.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.0000
25%	8.000000	9.000000	1.000000	0.000000	0.000000	2.000000	2.000000	8.0000
50%	11.000000	11.000000	2.000000	0.000000	4.000000	3.000000	2.000000	11.0000
75%	13.000000	13.000000	2.000000	0.000000	8.000000	4.000000	3.000000	13.7500
max	19.000000	19.000000	4.000000	3.000000	75.000000	4.000000	4.000000	19.0000

In [104…
```
Student_1.isna().sum()
```

Out[104…
```
G1           0
G2           0
studytime    0
failures     0
absences     0
Medu         0
Fedu         0
G3           0
dtype: int64
```

In [105…
```
plt.subplots(figsize=(12,5))
table_correlation=Student_1.corr()
sns.heatmap(table_correlation,annot=True,cmap='Spectral')
plt.title('Correlation between the variables')
plt.xticks(rotation=45)
plt.savefig('Grading Corelation')
```



In [106…

```
predict = "G3"

X = np.array(Student_1.drop([predict], 1)) # Features
y = np.array(Student_1[predict]) # Labels
```

```
C:\Users\prana\AppData\Local\Temp/ipykernel_33688/2680968852.py:3: FutureWarning: In
a future version of pandas all arguments of DataFrame.drop except for the argument
'labels' will be keyword-only
  X = np.array(Student_1.drop([predict], 1)) # Features
```

In [107…

```
x_train, x_test, y_train, y_test = sklearn.model_selection.train_test_split(X, y, te
```

In [108…

```
linear = linear_model.LinearRegression()
```

In [109…

```
linear.fit(x_train, y_train)
accuracy = linear.score(x_test, y_test) # acc stands for accuracy
```

In [110…

```
print(accuracy)
```

```
0.857531260646293
```

In [111…

```
print('Coefficient: \n', linear.coef_) # These are each slope value
print('Intercept: \n', linear.intercept_) # This is the intercept
```

```
Coefficient:
 [ 0.15377774  0.97134109 -0.2524329  -0.36458232  0.04092874  0.12347398
 -0.12790557]
Intercept:
 -1.3202158607011523
```

In [112…

```
predictions = linear.predict(x_test) # Gets a list of all predictions

for x in range(len(predictions)):
    print(predictions[x], x_test[x], y_test[x])
```

```
5.59991129627569 [7 7 2 2 4 3 2] 9
```

```
13.312045901477743 [14 13  1  0  6  3  4] 13
15.154998909254216 [16 15  2  0  2  1  2] 15
8.306493461776569 [8 9 2 0 4 1 1] 10
13.71942354954303 [11 12  2  0 54  3  3] 11
4.182136304761699 [6 5 2 0 6 4 4] 6
9.35439771348837 [11 10  3  0  4  2  3] 10
11.13267516019023 [13 11  2  0  4  4  3] 11
13.388420460955663 [12 14  3  0  7  2  4] 14
12.905656452271147 [11 13  1  1 10  3  2] 13
5.314115629465499 [7 6 1 0 0 4 4] 0
9.500154319545603 [10 10  1  0  2  1  3] 10
12.72067974133665 [12 13  2  0  2  2  2] 13
15.431583534053301 [16 15  3  0  9  4  3] 16
-0.7833728210033541 [6 0 2 0 0 2 1] 0
12.413314878265965 [13 13  2  0  0  1  4] 13
15.187752243380109 [16 15  2  0  0  4  4] 15
13.820137706917857 [14 14  3  0  4  4  4] 14
10.537930752061573 [11 11  2  0  0  3  3] 10
14.97381917771978 [16 15  4  0  7  3  3] 17
12.005922278650615 [10 12  2  0 16  4  4] 11
9.618143372167118 [10 10  2  0  2  4  3] 11
8.17858788717593 [8 9 2 0 4 1 2] 10
8.51868540658591 [8 9 1 0 0 2 1] 8
7.465698139530696 [10  8  2  0  0  4  4] 9
12.001771548786504 [12 12  1  0  2  2  2] 11
15.684683536330075 [17 15  1  0  2  2  2] 15
12.618646333385767 [13 13  2  0  2  2  4] 13
14.949476834704308 [13 15  2  0  2  1  0] 16
16.911683534320332 [17 17  4  0  0  3  2] 18
6.72947353396647 [9 8 1 3 6 1 1] 10
16.368856382252673 [16 16  2  0  2  4  3] 16
11.47109558398952 [12 11  1  0 16  3  4] 11
9.49843387648121 [ 9 10  3  0  9  4  3] 9
5.603307245381613 [ 7  6  2  0 10  3  2] 6
18.64032333172534 [17 18  1  0  0  3  2] 18
8.29319869180988 [8 9 2 0 4 4 4] 10
10.288938103115626 [12 10  2  0 14  4  4] 11
14.791478800681718 [14 15  3  0  4  4  4] 16
12.63882227082859 [12 13  2  0  0  2  2] 13
```

In [ ]: