AMES HOUSING DATA ANALYSIS

PROBLEM STATEMENT

The aim of this coursework is to utilize Google Colab to generate a Jupyter notebook for analysing the Ames Housing dataset. The dataset describes the sale of individual residential property from 2006 to 2010 in Ames (a small town in Iowa, USA).

The data was subsequently read and analysed, for the purpose of making an exploratory analysis of the House prices against other factors in the data.

SUMMARY

The Ames housing data was imported from my google drive. The data cleaning and analysis tool utilized for the cleaning and formatting of the data is Pandas, which is is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language and NumPy which offers comprehensive mathematical functions, random number generators, linear algebra routines, Fourier transforms, and more. For the exploration and visualization of the data, two python libraries are imported, matplotlib which provides an object-oriented API for embedding plots into applications and seaborn which is a python data visualization library based on matplotlib.

DATA CLEANING AND FORMATTING

The first step in Data formatting is importing the data to be analyzed and cleaned. The code above shows how data can be imported into Pandas by first mounting the drive on which it is saved on and calling the file from the directory location.

```
# Loading the data
from google.colab import drive
drive.mount("/content/drive")
%ls
%cd drive/
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.m
MyDrive/
[Errno 2] No such file or directory: 'drive/'
/content/drive
```

The drive has been successfully mounted and Pandas now has access to the file stored on the drive.

```
%ls
```

```
MyDrive/
```

```
%ls
```

```
MyDrive/
```

We must now call the functions that will be utilized for the formatting and cleaning of the data.

```
import pandas as pd
import numpy as np
```

```
Ames = pd.read_csv('/content/drive/MyDrive/Ames Housing.csv')
```

We then proceed to read the imported file and call the head of the data. by doing this, we know our file has been successfully imported and can be read easily.

```
Ames.head()
```

	Order	PID	MS SubClass	MS Zoning	Lot Frontage	Lot Area	Street	Alley	Lot Shape	Lanc Contour
0	1	526301100	20	RL	141.0	31770	Pave	NaN	IR1	Lv
1	2	526350040	20	RH	80.0	11622	Pave	NaN	Reg	Lv
2	3	526351010	20	RL	81.0	14267	Pave	NaN	IR1	Lv
3	4	526353030	20	RL	93.0	11160	Pave	NaN	Reg	Lv
4	5	527105010	60	RL	74.0	13830	Pave	NaN	IR1	Lv

5 rows × 82 columns

It is quite crucial and neccessary to know the shape, that is how many rows and columns we are dealing with. The .info tag tells us everything we need to know about the data. It lists the columns, tells us the type of data in each column and list the number of non-null entities in the data frame, as this will enable us clean the data.

```
Ames.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2930 entries, 0 to 2929
```

```
Data columns (total 82 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Order            2930 non-null   int64  
 1   PID              2930 non-null   int64  
 2   MS SubClass      2930 non-null   int64  
 3   MS Zoning        2930 non-null   object 
 4   Lot Frontage     2440 non-null   float64
 5   Lot Area         2930 non-null   int64  
 6   Street           2930 non-null   object 
 7   Alley            198 non-null    object 
 8   Lot Shape        2930 non-null   object 
 9   Land Contour     2930 non-null   object 
 10  Utilities        2930 non-null   object 
 11  Lot Config       2930 non-null   object 
 12  Land Slope       2930 non-null   object 
 13  Neighborhood     2930 non-null   object 
 14  Condition 1      2930 non-null   object 
 15  Condition 2      2930 non-null   object 
 16  Bldg Type        2930 non-null   object 
 17  House Style      2930 non-null   object 
 18  Overall Qual     2930 non-null   int64  
 19  Overall Cond     2930 non-null   int64  
 20  Year Built       2930 non-null   int64  
 21  Year Remod/Add   2930 non-null   int64  
 22  Roof Style       2930 non-null   object 
 23  Roof Matl        2930 non-null   object 
 24  Exterior 1st     2930 non-null   object 
 25  Exterior 2nd     2930 non-null   object 
 26  Mas Vnr Type     2907 non-null   object 
 27  Mas Vnr Area     2907 non-null   float64
 28  Exter Qual       2930 non-null   object 
 29  Exter Cond       2930 non-null   object 
 30  Foundation       2930 non-null   object 
 31  Bsmt Qual        2850 non-null   object 
 32  Bsmt Cond        2850 non-null   object 
 33  Bsmt Exposure    2847 non-null   object 
 34  BsmtFin Type 1   2850 non-null   object 
 35  BsmtFin SF 1     2929 non-null   float64
 36  BsmtFin Type 2   2849 non-null   object 
 37  BsmtFin SF 2     2929 non-null   float64
 38  Bsmt Unf SF      2929 non-null   float64
 39  Total Bsmt SF    2929 non-null   float64
 40  Heating          2930 non-null   object 
 41  Heating QC       2930 non-null   object 
 42  Central Air      2930 non-null   object 
 43  Electrical       2929 non-null   object 
 44  1st Flr SF       2930 non-null   int64  
 45  2nd Flr SF       2930 non-null   int64  
 46  Low Qual Fin SF  2930 non-null   int64  
 47  Gr Liv Area      2930 non-null   int64  
 48  Bsmt Full Bath   2928 non-null   float64
 49  Bsmt Half Bath   2928 non-null   float64
 50  Full Bath        2930 non-null   int64  
 51  Half Bath        2930 non-null   int64  
 52  Bedroom AbvGr    2930 non-null   int64
```

There are 2930 rows in the data, with 82 different columns. However, four columns seem to possess a lot of empty data, the number of null data contained in their rows are too significant in

relation to the number of rows. Due to this fact, they will be dropped as part of our effort to ensure a clean data. Pool Qc is the most notable, as it only has 13 non-null values, meaning 2913 values are empty. Misc Feature has 106 values, Fence has 572 value and Alley only has 198 values.

```
Ames=Ames.drop(["Alley", "Pool QC", "Misc Feature", "Fence"], axis = 1)
```

The .info of the data is run again, to confirm that the columns have been successfully eliminated from the dataframe.

```
Ames.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2930 entries, 0 to 2929
Data columns (total 78 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Order            2930 non-null   int64  
 1   PID              2930 non-null   int64  
 2   MS SubClass      2930 non-null   int64  
 3   MS Zoning        2930 non-null   object 
 4   Lot Frontage     2440 non-null   float64
 5   Lot Area         2930 non-null   int64  
 6   Street           2930 non-null   object 
 7   Lot Shape        2930 non-null   object 
 8   Land Contour     2930 non-null   object 
 9   Utilities        2930 non-null   object 
 10  Lot Config       2930 non-null   object 
 11  Land Slope       2930 non-null   object 
 12  Neighborhood     2930 non-null   object 
 13  Condition 1      2930 non-null   object 
 14  Condition 2      2930 non-null   object 
 15  Bldg Type        2930 non-null   object 
 16  House Style      2930 non-null   object 
 17  Overall Qual     2930 non-null   int64  
 18  Overall Cond     2930 non-null   int64  
 19  Year Built       2930 non-null   int64  
 20  Year Remod/Add   2930 non-null   int64  
 21  Roof Style       2930 non-null   object 
 22  Roof Matl        2930 non-null   object 
 23  Exterior 1st     2930 non-null   object 
 24  Exterior 2nd     2930 non-null   object 
 25  Mas Vnr Type     2907 non-null   object 
 26  Mas Vnr Area     2907 non-null   float64
 27  Exter Qual       2930 non-null   object 
 28  Exter Cond       2930 non-null   object 
 29  Foundation       2930 non-null   object 
 30  Bsmt Qual        2850 non-null   object 
 31  Bsmt Cond        2850 non-null   object 
 32  Bsmt Exposure    2847 non-null   object 
 33  BsmtFin Type 1   2850 non-null   object 
 34  BsmtFin SF 1     2929 non-null   float64
 35  BsmtFin Type 2   2849 non-null   object 
 36  BsmtFin SF 2     2929 non-null   float64
 37  Bsmt Unf SF      2929 non-null   float64
```

```
38  Total Bsmt SF    2929 non-null   float64
39  Heating          2930 non-null   object
40  Heating QC       2930 non-null   object
41  Central Air      2930 non-null   object
42  Electrical       2929 non-null   object
43  1st Flr SF       2930 non-null   int64
44  2nd Flr SF       2930 non-null   int64
45  Low Qual Fin SF  2930 non-null   int64
46  Gr Liv Area      2930 non-null   int64
47  Bsmt Full Bath   2928 non-null   float64
48  Bsmt Half Bath   2928 non-null   float64
49  Full Bath        2930 non-null   int64
50  Half Bath        2930 non-null   int64
51  Bedroom AbvGr    2930 non-null   int64
52  Kitchen AbvGr    2930 non-null   int64
```

In a further attempt to clean the data, we dropped the PID to give the data a better shape. PID is the Parcel ID and we do not need it for the processing of the data. It is dropped to give the data a better shape.

```
Ames=Ames.drop(["PID"], axis = 1)
```

The shape of the data is yet to be perfect, as there are some columns still missing some values. A forward fill is done to fill out the missing values in the different columns with empty values, to fill out the shape of the dataframe. What forward fill does is take the value before the empty slot and fill that in.

```
Ames["Garage Type"].fillna( method ='ffill', inplace = True)

Ames["Garage Yr Blt"].fillna( method ='ffill', inplace = True)

Ames["Garage Finish"].fillna( method ='ffill', inplace = True)

Ames["Garage Cars"].fillna( method ='ffill', inplace = True)

Ames["Garage Area"].fillna( method ='ffill', inplace = True)

Ames["Garage Qual"].fillna( method ='ffill', inplace = True)

Ames["Garage Cond"].fillna( method ='ffill', inplace = True)

Ames["Fireplace Qu"].fillna( method ='ffill', inplace = True)

Ames["Lot Frontage"].fillna( method ='ffill', inplace = True)

Ames["Mas Vnr Type"].fillna( method ='ffill', inplace = True)
```

```
Ames["Mas Vnr Area"].fillna( method ='ffill', inplace = True)

Ames["Bsmt Qual"].fillna( method ='ffill', inplace = True)

Ames["Bsmt Cond"].fillna( method ='ffill', inplace = True)

Ames["Bsmt Exposure"].fillna( method ='ffill', inplace = True)

Ames["BsmtFin Type 1"].fillna( method ='ffill', inplace = True)

Ames["BsmtFin Type 2"].fillna( method ='ffill', inplace = True)

Ames["BsmtFin SF 1"].fillna( method ='ffill', inplace = True)

Ames["BsmtFin SF 2"].fillna( method ='ffill', inplace = True)

Ames["Bsmt Unf SF"].fillna( method ='ffill', inplace = True)

Ames["Total Bsmt SF"].fillna( method ='ffill', inplace = True)

Ames["Electrical"].fillna( method ='ffill', inplace = True)

Ames["Bsmt Full Bath"].fillna( method ='ffill', inplace = True)

Ames["Bsmt Half Bath"].fillna( method ='ffill', inplace = True)
```

before any exploration can be done, we must confirm that every row is unique in itself and has no duplicated values across board. This is confirmed easily with the .duplicated function, which returns a false across all rows.

```
Ames.duplicated()
```

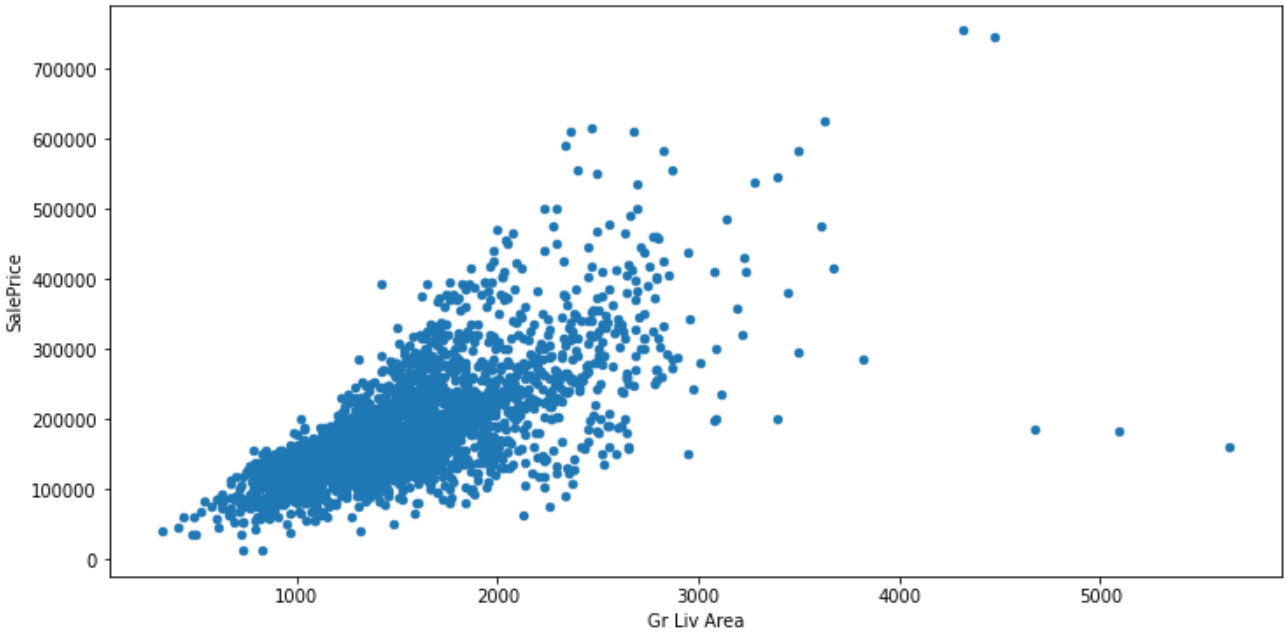
```
0       False
1       False
2       False
3       False
4       False
        ...
2925    False
2926    False
2927    False
2928    False
```

```
2929    False
Length: 2930, dtype: bool
```

According to the documentation attached to the file, there are five outliers in the dataset that must be eliminated and this can be made visible after running a scatter plot of Sale Price versus Gr Liv Area. We use the .loc function to call the specific row, column and remove houses with grade Living Area greater than 4000 square feet.

```
Ames.plot(kind = 'scatter', y = 'SalePrice', x = 'Gr Liv Area', figsize = (12,6))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f260d696910>
```




```
Ames.drop(index = Ames.loc[Ames.loc[:,"Gr Liv Area"]>4000].index, inplace=True)
```

We now have 2925 rows and 76 columns. our dataframe has been formatted and cleaned. We can now start the next part. Which is the Exploratory Data Analysis and Visualization of the data.

```
Ames.info()
```

```
 21  Roof Matl        2925 non-null   object
 22  Exterior 1st     2925 non-null   object
 23  Exterior 2nd     2925 non-null   object
 24  Mas Vnr Type     2925 non-null   object
 25  Mas Vnr Area     2925 non-null   float64
 26  Exter Qual       2925 non-null   object
 27  Exter Cond       2925 non-null   object
 28  Foundation       2925 non-null   object
 29  Bsmt Qual        2925 non-null   object
 30  Bsmt Cond        2925 non-null   object
 31  Bsmt Exposure    2925 non-null   object
 32  BsmtFin Type 1   2925 non-null   object
 33  BsmtFin SF 1     2925 non-null   float64
 34  BsmtFin Type 2   2925 non-null   object
 35  BsmtFin SF 2     2925 non-null   float64
```

```
 36  Bsmt Unf SF      2925 non-null   float64
 37  Total Bsmt SF    2925 non-null   float64
 38  Heating          2925 non-null   object
 39  Heating QC       2925 non-null   object
 40  Central Air      2925 non-null   object
 41  Electrical       2925 non-null   object
 42  1st Flr SF       2925 non-null   int64
 43  2nd Flr SF       2925 non-null   int64
 44  Low Qual Fin SF  2925 non-null   int64
 45  Gr Liv Area      2925 non-null   int64
 46  Bsmt Full Bath   2925 non-null   float64
 47  Bsmt Half Bath   2925 non-null   float64
 48  Full Bath        2925 non-null   int64
 49  Half Bath        2925 non-null   int64
 50  Bedroom AbvGr    2925 non-null   int64
 51  Kitchen AbvGr    2925 non-null   int64
 52  Kitchen Qual     2925 non-null   object
 53  TotRms AbvGrd    2925 non-null   int64
 54  Functional       2925 non-null   object
 55  Fireplaces       2925 non-null   int64
 56  Fireplace Qu     2925 non-null   object
 57  Garage Type      2925 non-null   object
 58  Garage Yr Blt    2925 non-null   float64
 59  Garage Finish    2925 non-null   object
 60  Garage Cars      2925 non-null   float64
 61  Garage Area      2925 non-null   float64
 62  Garage Qual      2925 non-null   object
 63  Garage Cond      2925 non-null   object
 64  Paved Drive      2925 non-null   object
 65  Wood Deck SF     2925 non-null   int64
 66  Open Porch SF    2925 non-null   int64
 67  Enclosed Porch   2925 non-null   int64
 68  3Ssn Porch       2925 non-null   int64
 69  Screen Porch     2925 non-null   int64
 70  Pool Area        2925 non-null   int64
 71  Misc Val         2925 non-null   int64
 72  Mo Sold          2925 non-null   int64
 73  Yr Sold          2925 non-null   int64
 74  Sale Type        2925 non-null   object
 75  Sale Condition   2925 non-null   object
 76  SalePrice        2925 non-null   int64
dtypes: float64(11), int64(27), object(39)
memory usage: 1.7+ MB
```

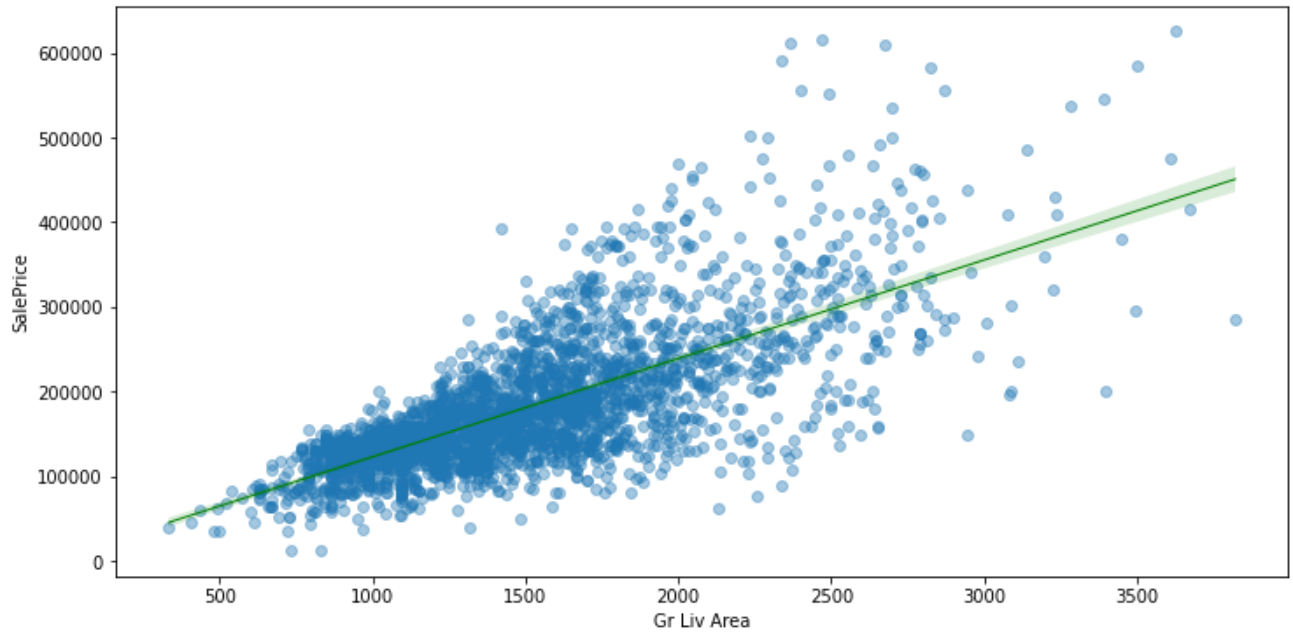
EXPLORATORY DATA ANALYSIS AND DATA VISUALIZATION

```
import matplotlib.pyplot as plt
import seaborn as sns
```

To explore the data and visualize it we import two libraries. Matplotlib and Seaborn.

```
plt.figure(figsize = (12,6))
g = sns.regplot(data=Ames, x = "Gr Liv Area" , y = "SalePrice" ,
                scatter_kws={"alpha" : 0.4}, line_kws={"color": "green", "linewidth":  1.0
```

)



Obviously our code earlier to remove houses with grade living area greater than 4000 was a successful one. Here we can clearly see that houses with grade living area close to 1000 square feet up to 2000 square feet sold the most. They also ranged from over 100000 to the range between 200000 and 300000 in sale price. However a few houses in the 2500 square feet sold for 500000 to over 600000. Another few with square feet 3000 to 3500 sold between 200000 and 0ver 600000

DATA CORRELATION

We must do a correlation of the integer sets in the dataframe, as this makes it easy to know what columns have a positive relationship with one another within our dataset. The SalePrice is a key column in this dataset and will be considered more against other columns. The year sold is also a factor that will be considered.

```
ames_corr=Ames.corr(method = "kendall")
```

```
ames_corr
```

	Order	MS SubClass	Lot Frontage	Lot Area	Overall Qual	Overall Cond	Year Built	I
Order	1.000000	0.011427	-0.017868	0.006821	-0.038073	-0.011288	-0.043244	
MS SubClass	0.011427	1.000000	-0.248512	-0.239909	0.080513	-0.052423	0.008489	
Lot Frontage	-0.017868	-0.248512	1.000000	0.455255	0.135735	-0.055645	0.106615	
Lot Area	0.006821	-0.239909	0.455255	1.000000	0.143334	-0.060239	0.083883	
Overall Qual	-0.038073	0.080513	0.135735	0.143334	1.000000	-0.162979	0.517550	
Overall Cond	-0.011288	-0.052423	-0.055645	-0.060239	-0.162979	1.000000	-0.336228	
Year Built	-0.043244	0.008489	0.106615	0.083883	0.517550	-0.336228	1.000000	
Year Remod/Add	-0.060882	0.001525	0.068547	0.070697	0.452952	-0.057413	0.650065	
Mas Vnr Area	-0.029574	-0.010880	0.186821	0.153611	0.346527	-0.154243	0.296009	
BsmtFin SF 1	-0.025016	-0.071826	0.100707	0.117193	0.136350	-0.019780	0.167443	
BsmtFin SF 2	-0.013960	-0.079726	0.051718	0.046746	-0.078848	0.086079	-0.075497	
Bsmt Unf SF	0.004438	-0.083387	0.051352	0.045093	0.178172	-0.092640	0.076792	
Total Bsmt SF	-0.018147	-0.223188	0.238622	0.242277	0.357499	-0.174947	0.305921	
1st Flr SF	-0.007063	-0.202065	0.287688	0.304670	0.310855	-0.145046	0.222685	
2nd Flr SF	-0.002424	0.396985	0.007369	0.044322	0.200023	-0.009770	0.006113	
Low Qual Fin SF	0.021713	0.052297	-0.032910	-0.013493	-0.044570	0.017025	-0.101083	
Gr Liv Area	-0.017363	0.128919	0.223992	0.281167	0.443112	-0.146199	0.208902	
Bsmt Full Bath	-0.034574	-0.037267	0.070751	0.084996	0.129414	-0.039431	0.151279	
Bsmt Half Bath	0.014680	0.005259	0.001289	0.004161	-0.045960	0.082574	-0.047214	
Full Bath	-0.040631	0.172534	0.139694	0.179809	0.492366	-0.260669	0.435472	
Half Bath	-0.030589	0.236181	0.058089	0.101456	0.259055	-0.098597	0.215269	
Bedroom AbvGr	0.010673	0.055306	0.220108	0.234145	0.066222	-0.010976	-0.024358	
Kitchen AbvGr	-0.012134	0.238014	0.009138	-0.020531	-0.151927	-0.078553	-0.106242	

TotRms AbvGrd	-0.000152	0.113387	0.254300	0.283795	0.308294	-0.105147	0.130407
Fireplaces	-0.018687	0.008857	0.180604	0.246860	0.360817	-0.050372	0.155389
Garage Yr Blt	-0.037067	0.034641	0.067940	0.060568	0.478072	-0.287154	0.818145
Garage Cars	-0.030706	0.015386	0.244024	0.264964	0.542453	-0.231000	0.489074
Garage Area	-0.025316	-0.037121	0.239706	0.248937	0.428013	-0.169558	0.384500
Wood Deck SF	-0.019731	0.015086	0.071737	0.126913	0.229866	-0.029019	0.208741
Open Porch SF	-0.000105	0.023603	0.110602	0.120609	0.346769	-0.142187	0.288734
Enclosed Porch	0.020052	-0.003745	-0.061635	-0.032037	-0.163560	0.119847	-0.343986
3Ssn Porch	-0.012257	-0.029240	-0.002750	0.023918	0.017650	0.041012	0.009409
Screen Porch	0.004982	-0.034232	0.069459	0.073984	0.023851	0.045568	-0.052916
Pool Area	0.040834	-0.006122	0.052628	0.059332	0.014763	-0.014218	-0.006695
Misc Val	-0.032233	-0.026226	0.027454	0.058180	-0.070726	0.057944	-0.067336
Mo Sold	0.099804	0.011895	0.019242	0.003409	0.022331	-0.004435	0.011333
Yr Sold	-0.889506	-0.017301	0.003643	-0.014937	-0.012545	0.033112	-0.005396

```
plt.figure(figsize=(8,6))
sns.heatmap(ames_corr)
```

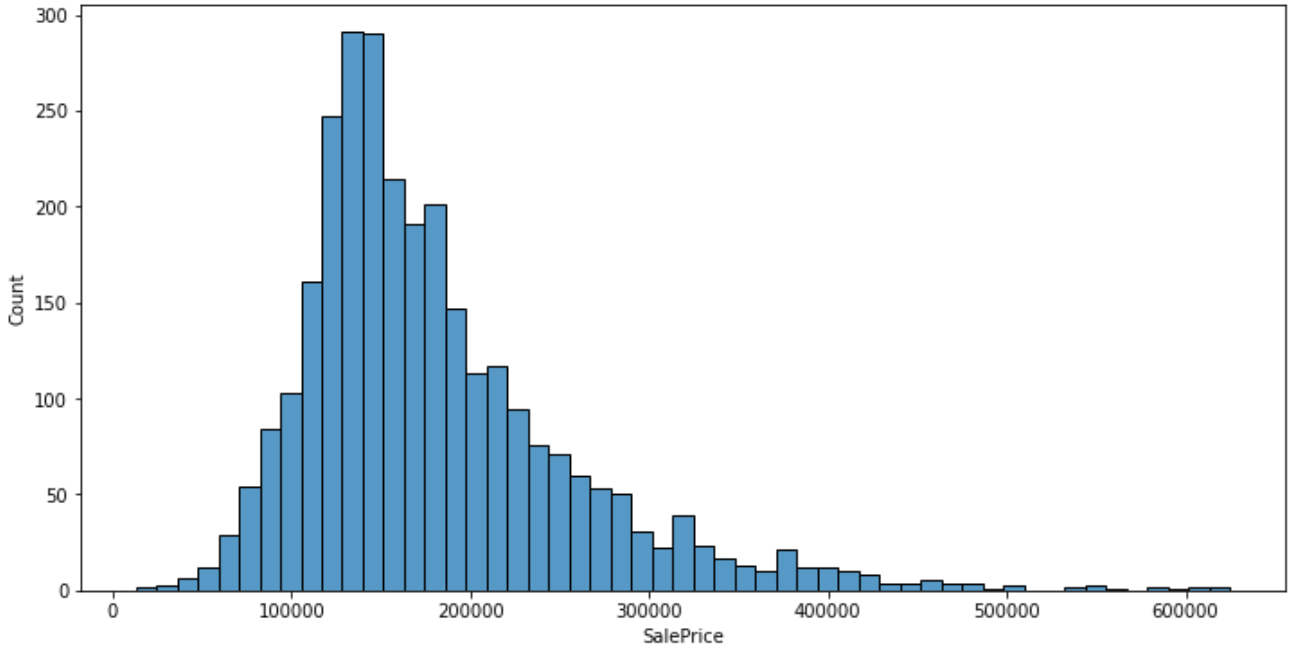
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f260c4cde10>
```

A histogram shows the frequency of a particular entity, in this case we are checking for what appears the most in SalePrice. According to the histogram plotted we can say that houses sold more between the range of 100000 to 200000. With houses costing roughly 150000 selling the most, almost 300 houses sold within this range. Houses that cost almost 500000 and houses greater than 600000 sold less. With only one house selling for over 60000.

SCATTER PLOT, HISTPLOT, DISTPLOT, COUNTPLOT AND BOXPLOT


```
plt.figure(figsize = (12,6))
sns.histplot(Ames["SalePrice"])
```

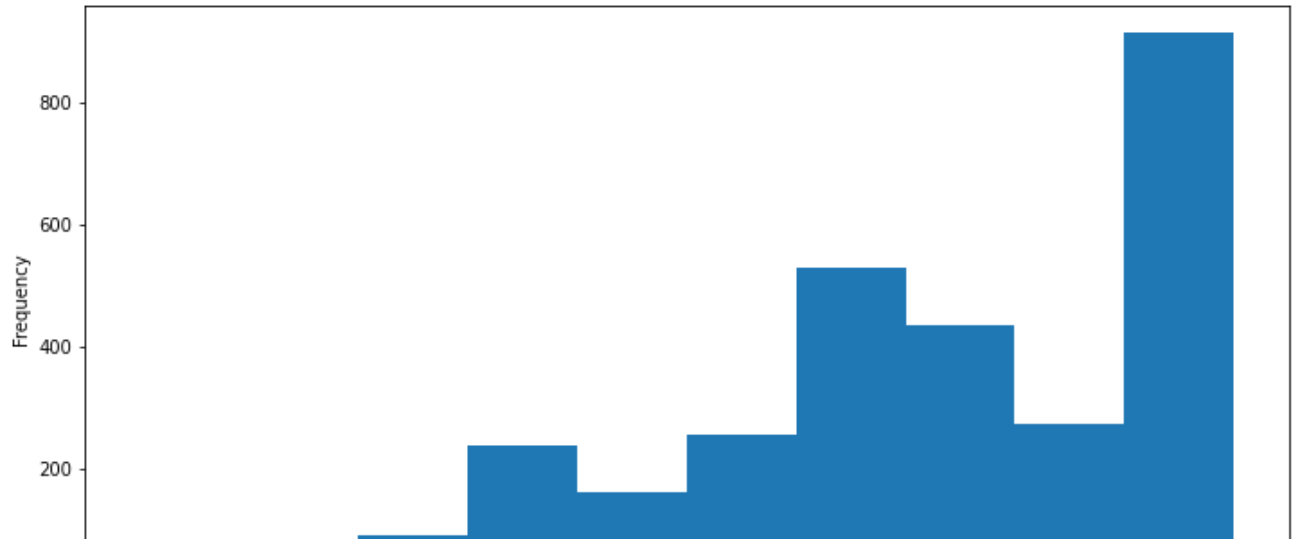
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f260c42d390>
```



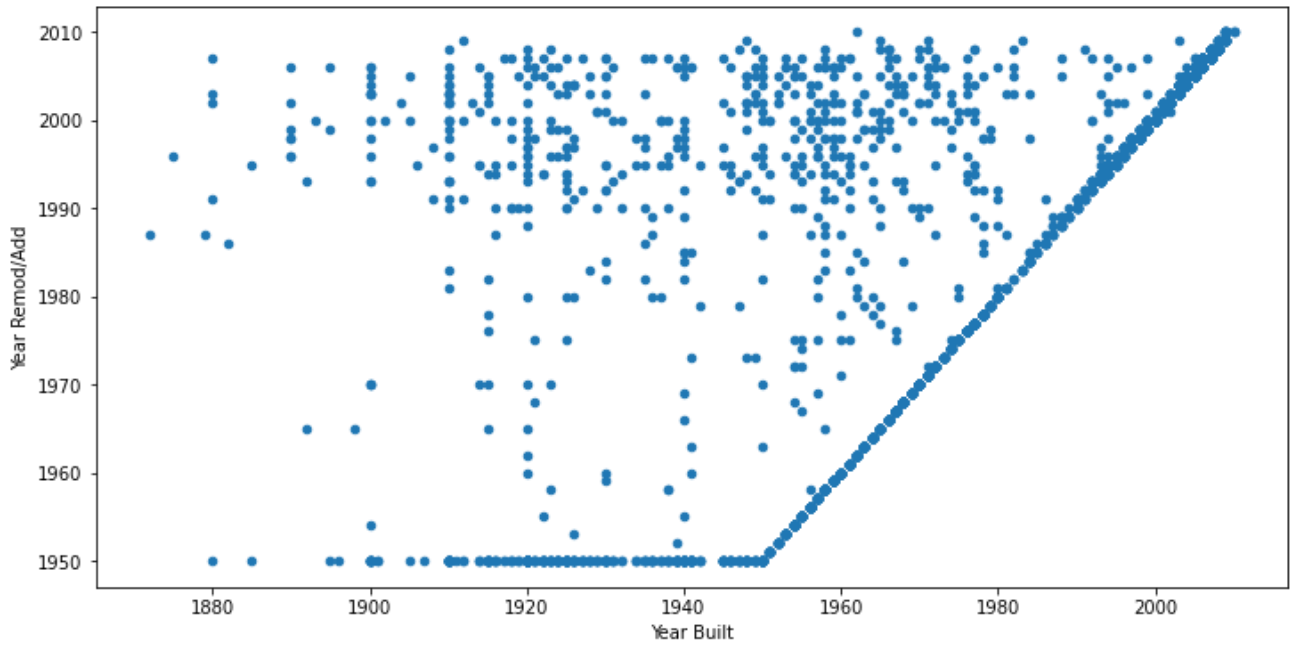
Now we visualize the year built using histogram also, to see a frequency of the year houses were built. Over 800 houses were built from year 2000 upwards.

```
plt.figure(figsize = (12,6))
Ames["Year Built"].plot(kind = 'hist')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f260c23d390>
```

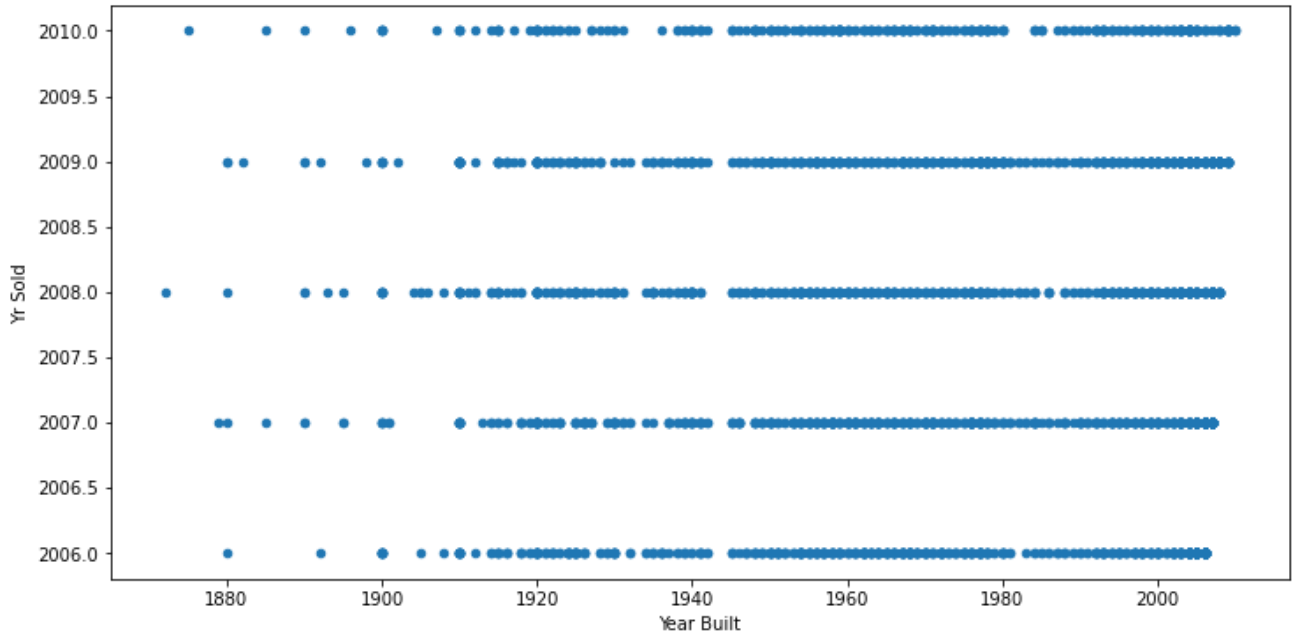



```
Ames.plot(kind = 'scatter', x = 'Year Built', y = 'Year Remod/Add', figsize = (12,6))

plt.show()
```



Scatter plot is used to check the relationship between two entities. I am interested in the relationship between when a house was built and when it was remodelled. Houses built from the 1950 to 1980 were remodelled in the 2000s. A few houses built in 2000s have also been remodelled.

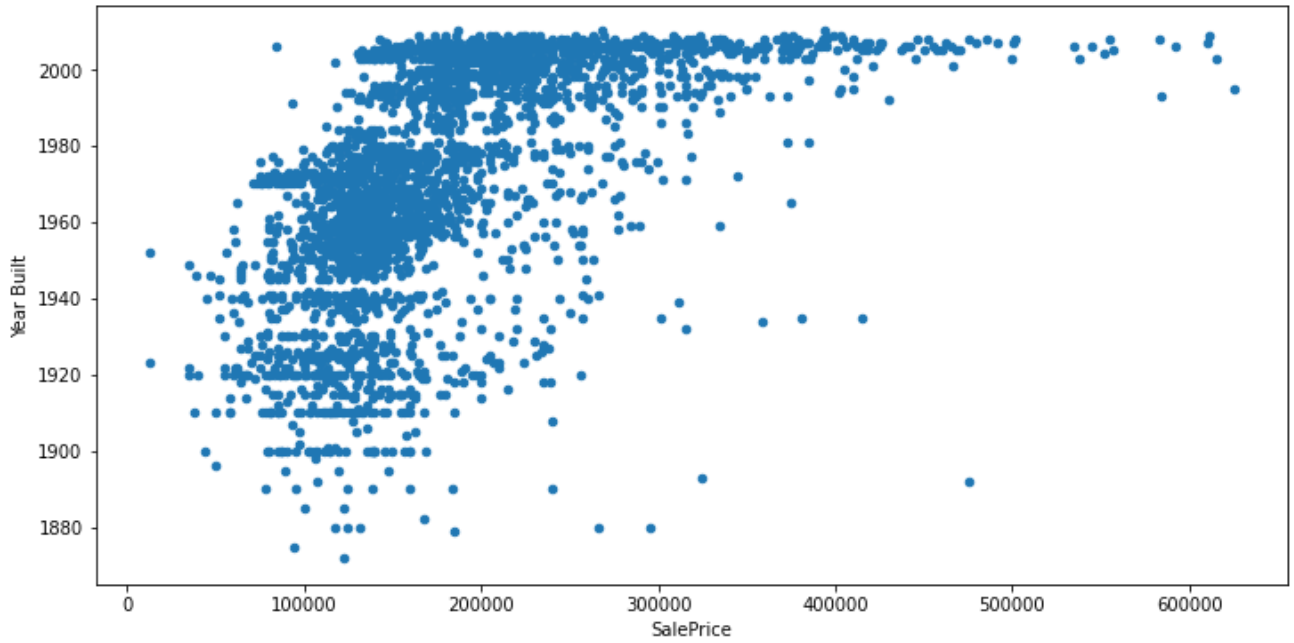
```
Ames.plot(kind = 'scatter', x = 'Year Built', y = 'Yr Sold', figsize = (12,6))

plt.show()
```



The year built and year sold is also something key to be compared. We can see a lateral movement in the year built and year sold. Five houses built in 1880 were sold across the 2006 to 2010. More houses built in the 2000s were sold than houses built earlier.

```
plt.figure(figsize=(12,6))
Ames.plot(kind = 'scatter', x = 'SalePrice', y = 'Year Built', figsize = (12,6))

plt.show()
```

```
<Figure size 864x432 with 0 Axes>
```



The relationship between Sale price and Year Built is a very key component to be checked, as the relationship will show us how expensive houses are based on the year they are built. Between the year 1880 and 1980, houses built during this period sold for 100000 and close to 200000.

Houses built in the year 2000 and above cost 200000 and higher, with a few houses selling for over half a million and a few costing over 600000.

```
Ames.loc[:,["SalePrice","Yr Sold","Mo Sold"]]
```

	SalePrice	Yr Sold	Mo Sold
0	215000	2010	5
1	105000	2010	6
2	172000	2010	6
3	244000	2010	4
4	189900	2010	3
...	...	...	...
2925	142500	2006	3
2926	131000	2006	6
2927	132000	2006	7
2928	170000	2006	4
2929	188000	2006	11

2925 rows × 3 columns

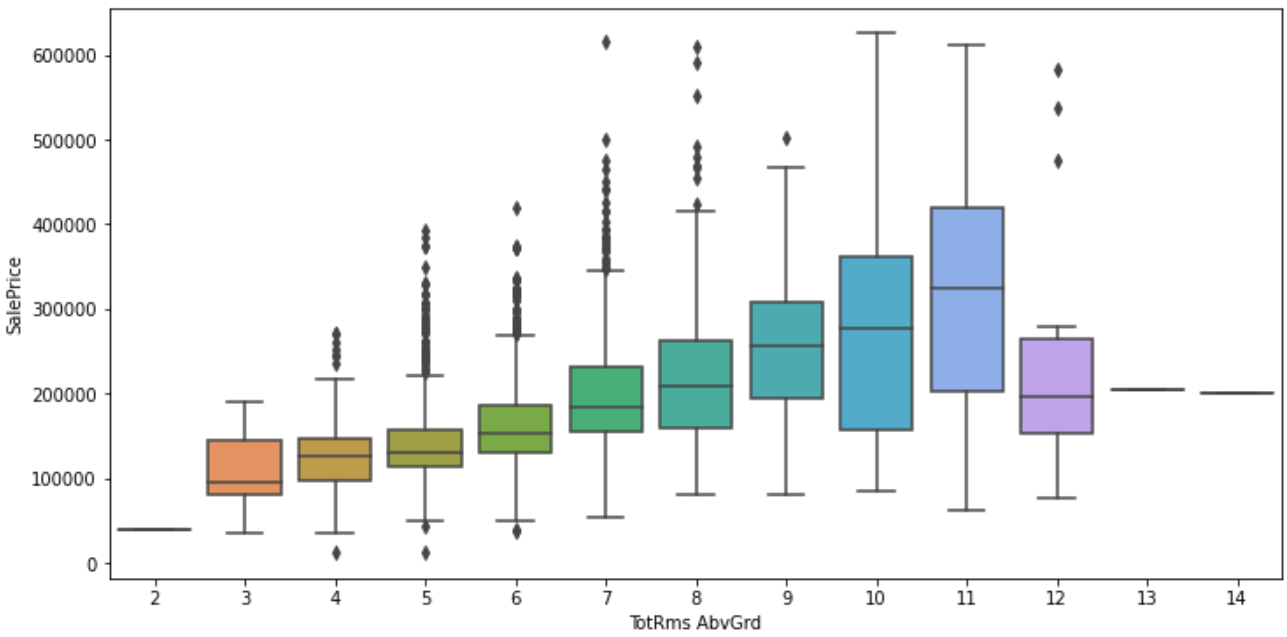
A distribution plot is a key tool that shows the density of the entity being considered. It also shows the skewness of the entity. From the graph we can see that sale price of 150000 came out at the highest, which means more houses were sold for that price. The plot is negatively skewed, with the sale price between 100000 and 200000 heavily densed.

```
plt.figure(figsize = (12,6))
sns.distplot(Ames["SalePrice"])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning:
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f260c2a1990>
```

A box plot is meant to show the median, the minimum, maximum, first quartile and third quartile. The houses with total rooms above grade was plotted against sale price. It also shows the skewness. Houses with three rooms above grade are positively skewed, with the first quartile just a little below 100000 and the minimum at 50000, with a maximum at 200000 and the third quartile at 150000. Houses with 4,5,6,7,8,9 and 12 rooms above grade have outliers. Houses with 4 rooms above grade are negatively skewed, with the median a little close to 150000 and the minimum at 50000, the maximum at a little 200000. Houses with 8 rooms above grade are normally skewed, with the minimum a little above 50000 and the maximum a little close to 400000. There are a number of outliers going as far up as 600000 in sale price, with a median around 150000. Houses with 2, 13 and 14 rooms above grade do not have enough data to form a median or any of the quartile ranges.

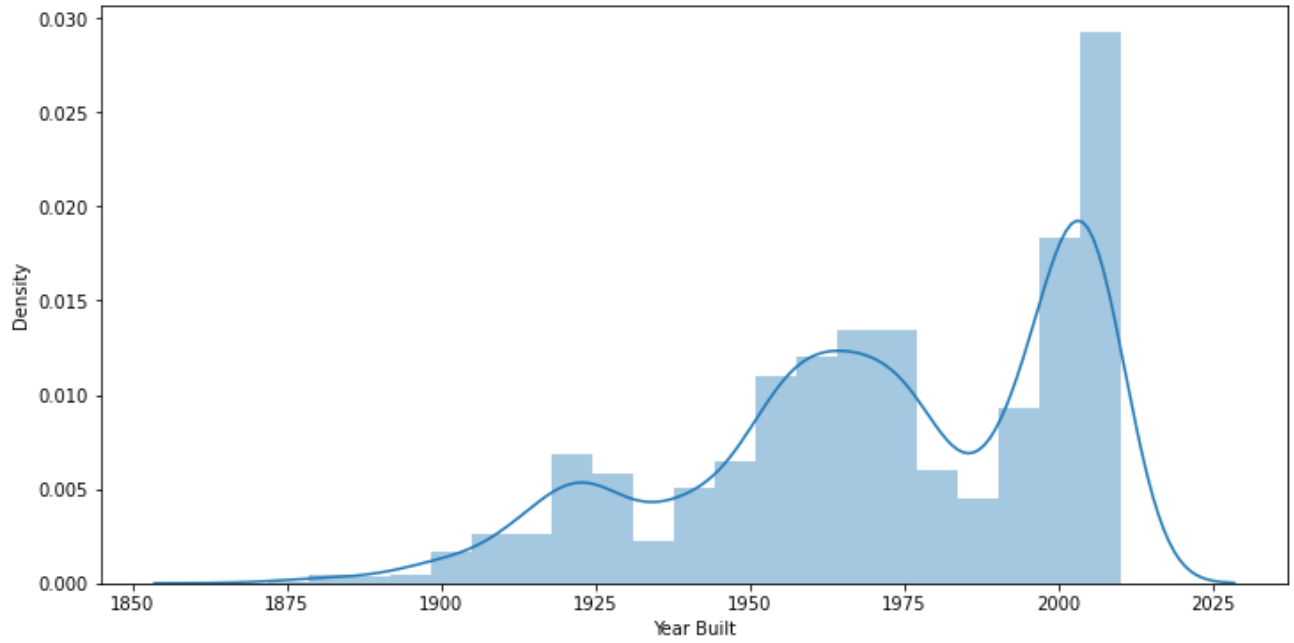
```
plt.figure(figsize=(12,6))
sns.boxplot(y = "SalePrice", x = "TotRms AbvGrd", data=Ames )
plt.show()
```



In this cell, we check the density of the year houses were built and most houses in the data were built from the year 2000 upwards. the skewness is distributed across the plot. A lot of houses were built between 1950 and 1975, there was a dip between 1975 and 1990. But we can see the amount of houses built over the years increased geometrically

```
plt.figure(figsize = (12,6))
sns.distplot(Ames["Year Built"])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning:
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f260d1d1850>
```




```
plt.figure(figsize=(20,6))
sns.boxplot(y = "SalePrice", x = "Year Built", data=Ames )
plt.xticks(rotation=90)
plt.show()
```

This is a boxplot of the sale price and year built. There are a lot of plots in this graph, however, there are some notable plots to be considered, like the year 1892, 1935 and year 2008. The plot shows the trend of a geometric increase in sale price over the year built of houses in the data. However, 1892 stands out in the 1800s, with the maximum showing houses selling for almost 500000. The year 2008 shows houses selling for almost 600000.

Because of the many data in year built and the corresponding sale price. A data frame must be created, to check the trend on a bar chat. A data frame for year built called YearBuilt was created with only the 2000s in focus, from year 2000 to 2010. Also a data frame for sale price was created called, sp, taking into factor a range of prices from 205000 to 625000. Then a barchart was plotted and the result coincides with the box plot of a rise in saleprice over the years.

```
  YearBuilt = np.array(Ames["Year Built"])
YearBuilt
np.unique(YearBuilt)
```

```
     array([1872, 1875, 1879, 1880, 1882, 1885, 1890, 1892, 1893, 1895, 1896,
            1898, 1900, 1901, 1902, 1904, 1905, 1906, 1907, 1908, 1910, 1911,
            1912, 1913, 1914, 1915, 1916, 1917, 1918, 1919, 1920, 1921, 1922,
            1923, 1924, 1925, 1926, 1927, 1928, 1929, 1930, 1931, 1932, 1934,
            1935, 1936, 1937, 1938, 1939, 1940, 1941, 1942, 1945, 1946, 1947,
            1948, 1949, 1950, 1951, 1952, 1953, 1954, 1955, 1956, 1957, 1958,
            1959, 1960, 1961, 1962, 1963, 1964, 1965, 1966, 1967, 1968, 1969,
            1970, 1971, 1972, 1973, 1974, 1975, 1976, 1977, 1978, 1979, 1980,
            1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988, 1989, 1990, 1991,
            1992, 1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002,
            2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010])
```

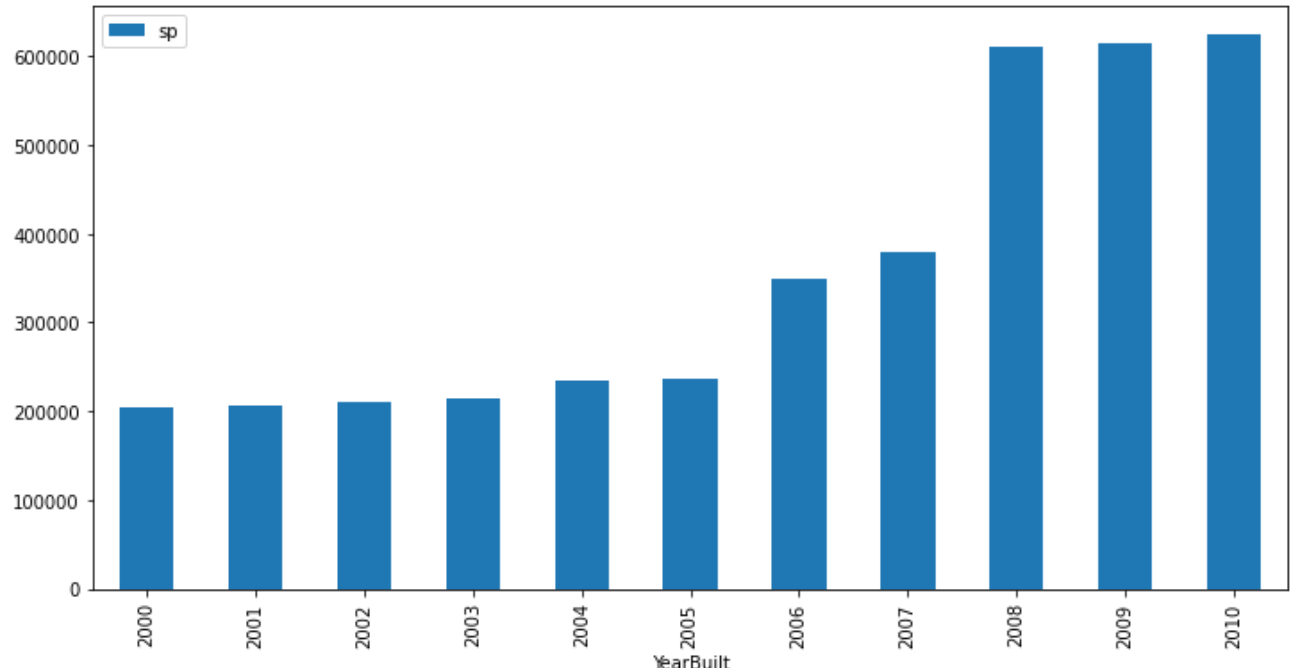
```
  sp = np.array(Ames["SalePrice"])
sp
np.unique(sp)
```

```
     array([ 12789,  13100,  34900, ..., 611657, 615000, 625000])
```

```
spYearBuilt = pd.DataFrame({
    'YearBuilt': [2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010],
    'sp': [205000, 206000, 210000, 215000, 235000, 236500, 349000, 378500,  611657, 615000
```

```
spYearBuilt.plot(y="sp", x="YearBuilt", kind="bar", figsize=(12,6))
```

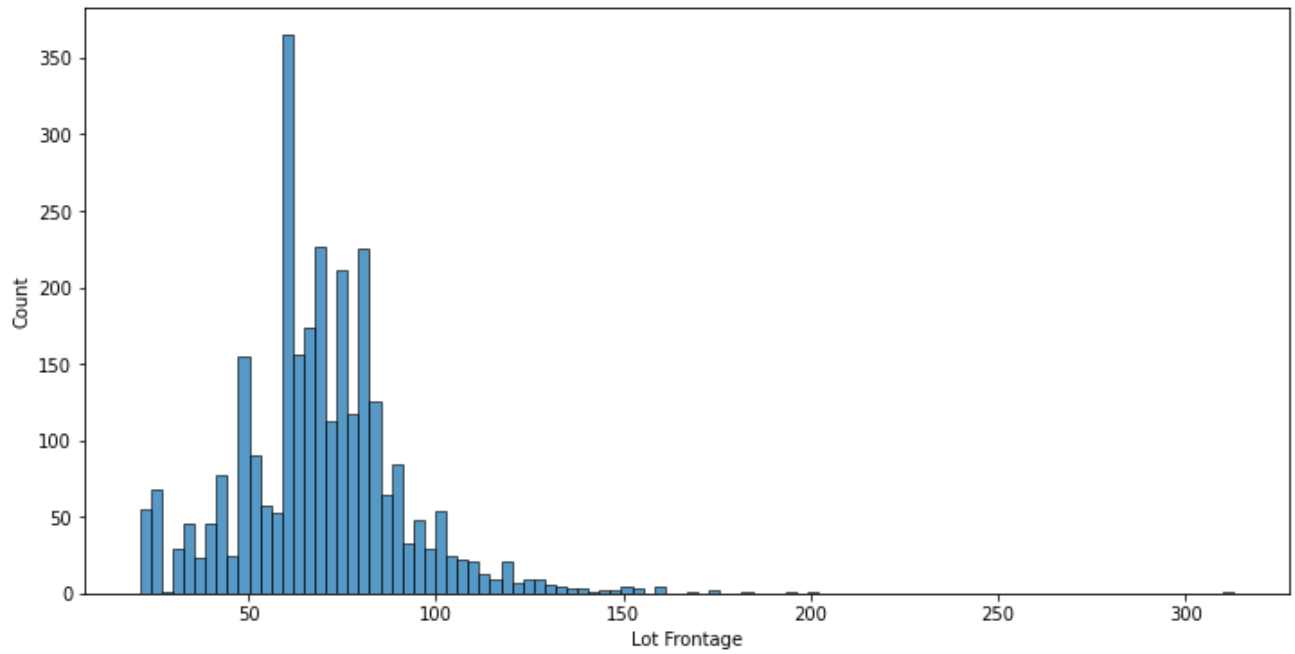
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f260bc0a5d0>
```



This is a histplot of the lot frontage and a number of houses that boast of them. Houses with 58 square feet lot frontage are more prominent on the graph. The graph is skewed towards the highest count.

```
plt.figure(figsize = (12,6))
sns.histplot(Ames["Lot Frontage"])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f260bb4ddd0>
```

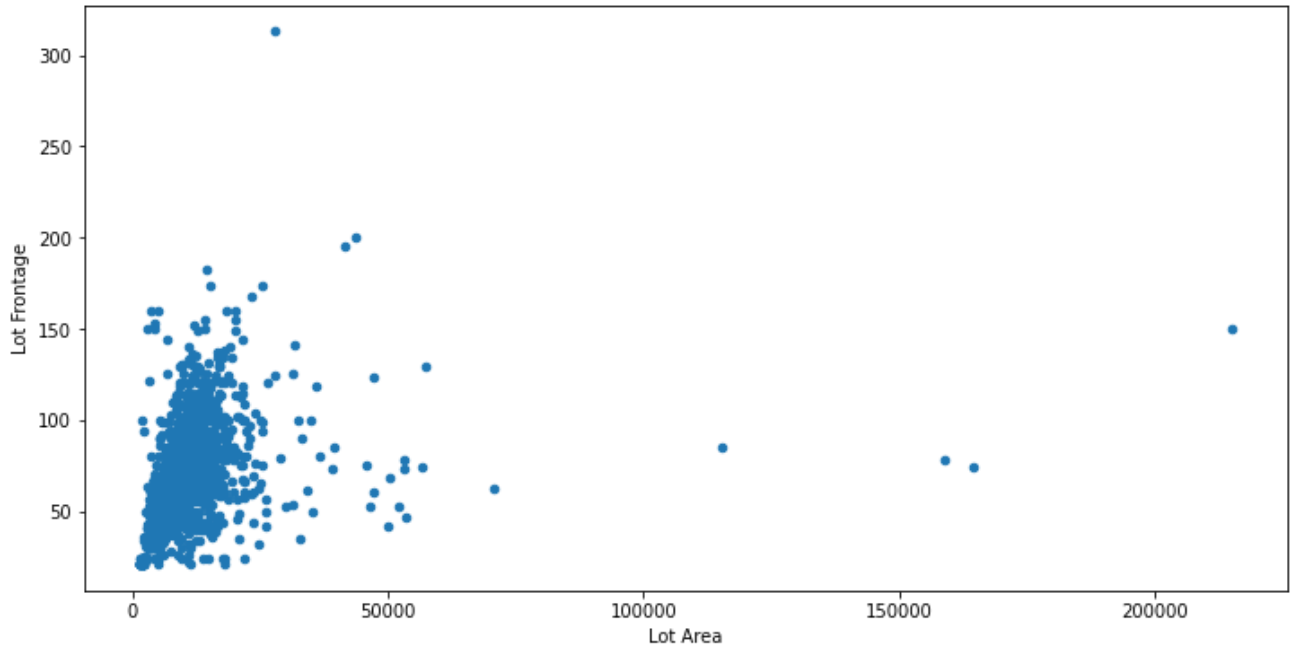


A scatter plot of the lot frontage and the lot area shows that a great many number of lot frontage and lot area are between 50 to 150 in lot frontage and less than 5000 lot area. There

were a few outliers from greater than 50000 lot area to greater than 200000 lot area.

```
Ames.plot (kind = "scatter", y ="Lot Frontage" , x = "Lot Area",figsize=(12,6))
```

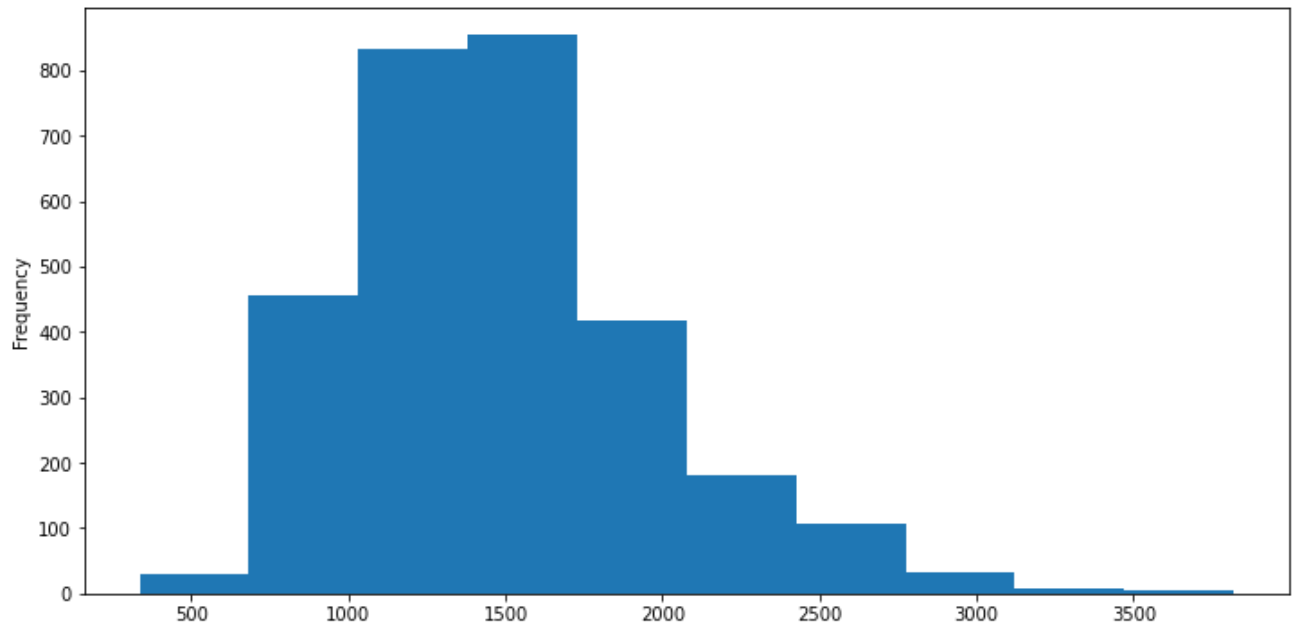
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f261e2cd210>
```



We plot a histogram of the ground living area, to check which one comes out the most. The ground living area between 1250 and 1750 square feet has the most frequency in the entire data.

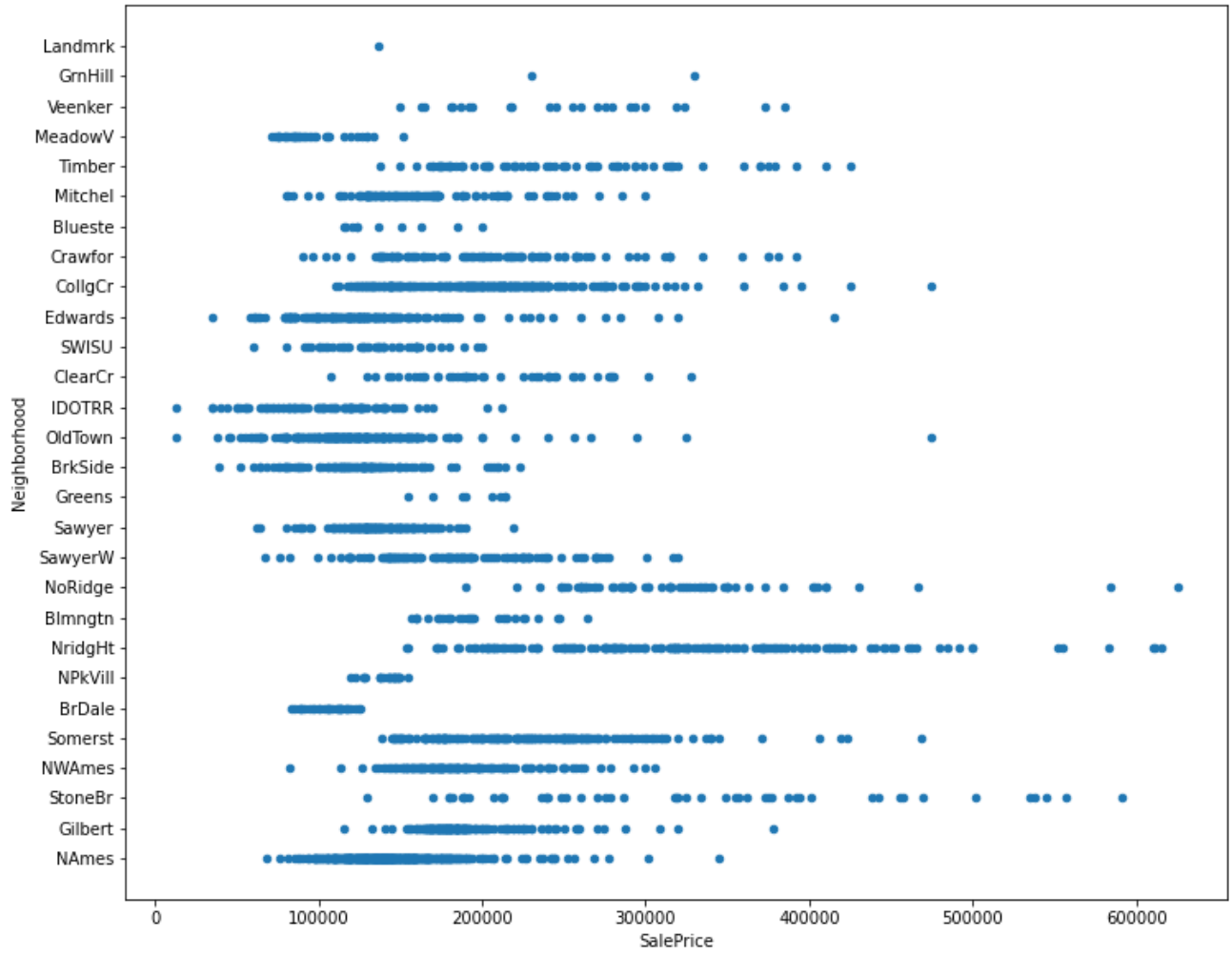
```
plt.figure(figsize = (12,6))
Ames["Gr Liv Area"].plot(kind = 'hist')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f260d44ead0>
```



```
Ames.plot(kind = "scatter", y = "Neighborhood", x = "SalePrice", figsize = (12,10))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f260bc0ae90>
```

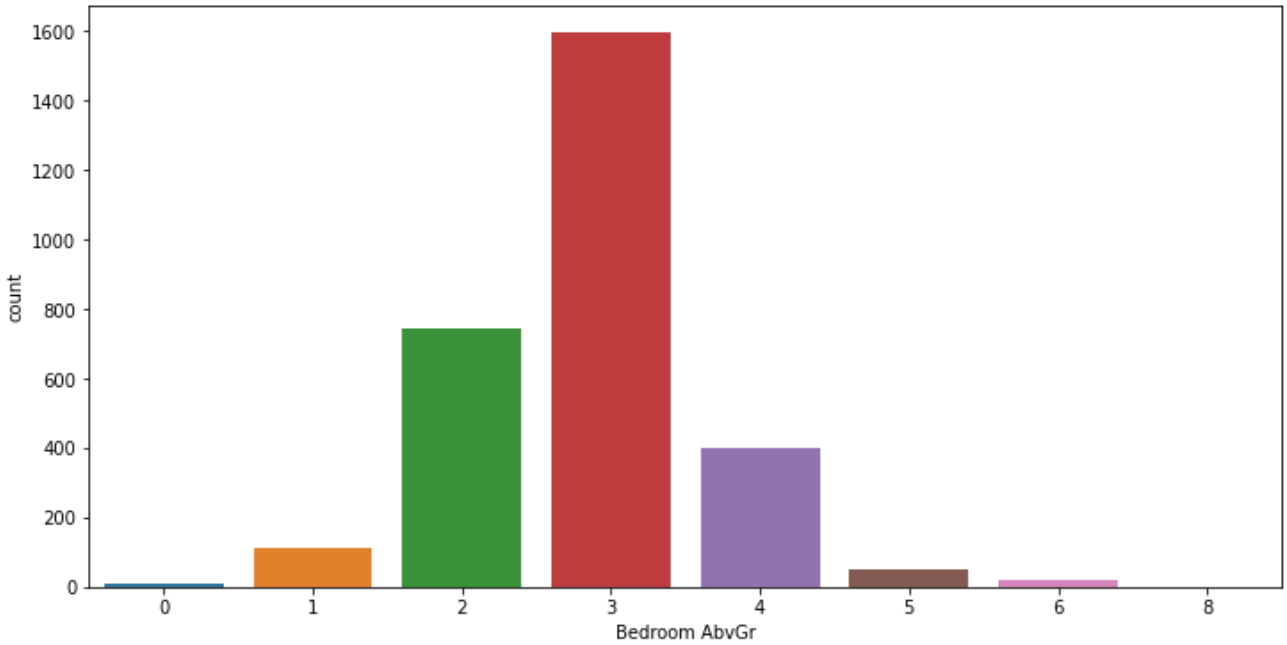


The scatter plot of Neighborhood and sale price shows the most expensive neighborhood in the data. The North Ames neighborhood has houses more concentrated in the 100000 to 200000 region. The Northern Heights neighborhood seems to be more concentrated with more houses in the region 200000 to 500000 and a number of houses going for over 500000 to a little over 600000. Northpark Villa, Greens and Briar Dale neighborhoods have the lowest sale price between 90000 and 150000. Landmark has only one house showing on the scatter plot close to 150000 in sale price.

A countplot of the bedroom above grade clearly shows that houses with 3 bedrooms above grade are way more than others, with 2 bedroom houses coming second and 4 bedroom houses coming third. Houses with 0 bedrooms above grade, could be studio flats, with only one house having 6 bedrooms above grade.

```
plt.figure(figsize = (12,6))
sns.countplot(Ames["Bedroom AbvGr"])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f260bd0bed0>
```



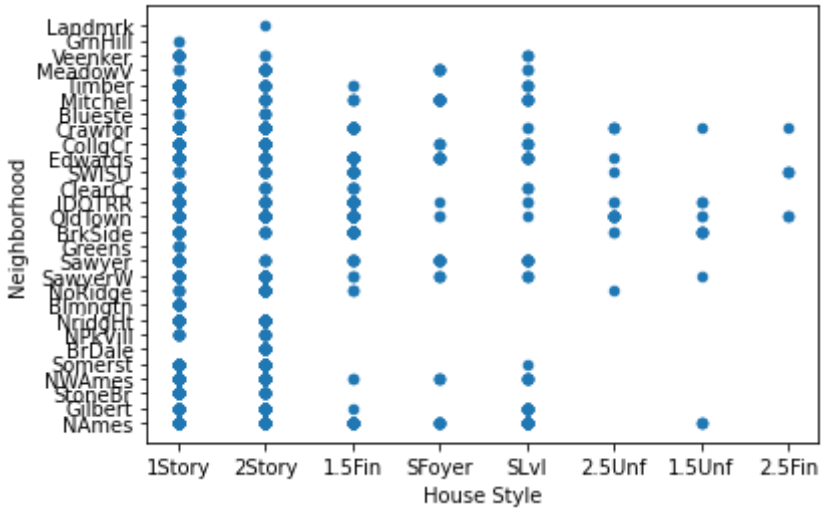

```
plt.figure(figsize = (12,6))
plt.xticks(rotation=45)
sns.histplot(Ames["Neighborhood"])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f260bd769d0>
```

A histogram plot of the neighborhood shows North Ames coming out on top of the other neighborboods with over 400 houses and Landmark having just the one house as shown in the scatter plot plotted against sale price. College Creek comes second to North Ames having over 250 houses and Old Town a close third with 250 houses.

```
Ames.plot (kind = "scatter", y = "Neighborhood", x ="House Style")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f260cd6e0d0>
```



There are a variety of house style in the dataset and this plot shows which appears the most. There are more one story houses in the dataset, with two story houses coming second with over 800. The lowest in the plot are two and one-half story houses, with second level unfinished, one and one-half story, with second level unfinished and two and one-half story, with second level finished.

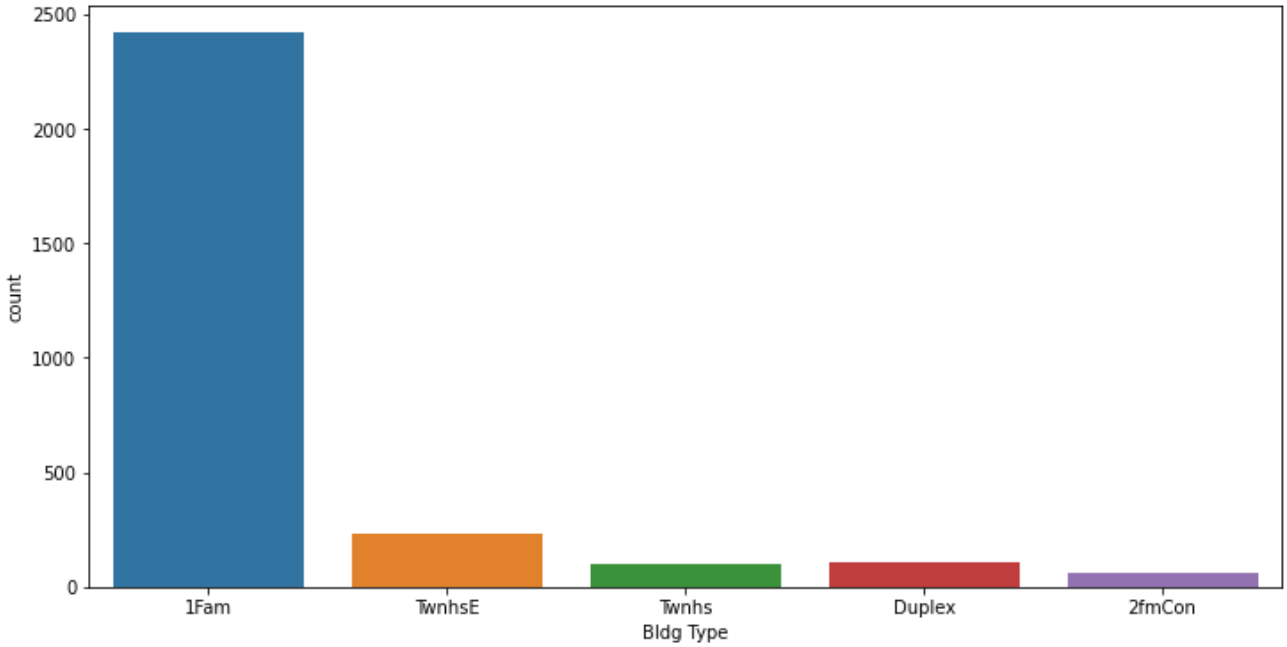
```
plt.figure(figsize = (12,6))
sns.countplot(Ames["House Style"])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f260bccfdd0>
```

We also consider the building types in the dataset. Single family detached is king out of all the types of building in the dataset. Two family conversion and duplex come lowest on the rank.

```
plt.figure(figsize=(12,6))
sns.countplot(Ames["Bldg Type"])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f260bccffd0>
```

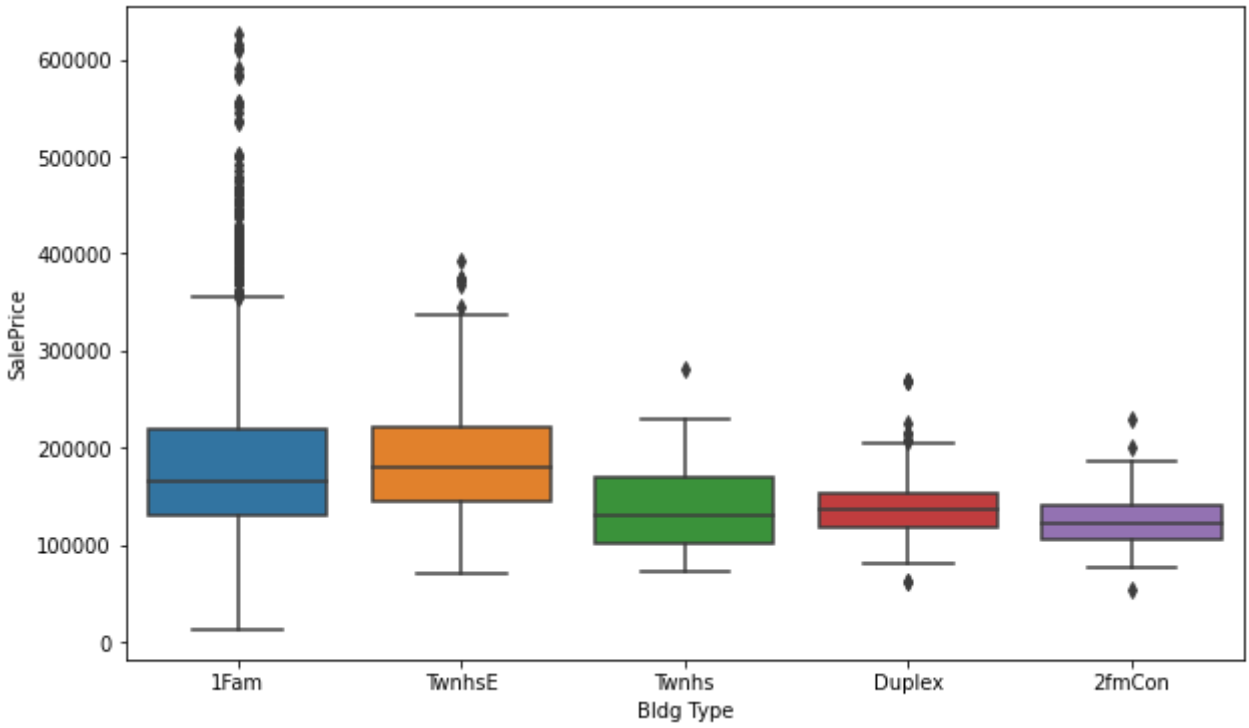


I did a boxplot of Sale Price against Building type. Positive skewness is noticeable in one family and townhouse inside unit building types, with duplex building type showing negative skewness, while two family conversions and town house end units showing normal skewness.

The one family building type, has its maximum between 300000 and 400000 and outliers going as far ass over 600000, the minimum is at 10000 and the median at above 150000. Town house end unit has its maximum at above 300000, median at above 150000, minimum at above 50000, and median at close to 200000. It also has a few outliers close to 400000. Townhouse inside unit has its minimum at the same level as townhouse end unit and its maximum at the 75th percentile of townhouse end unit, with a single outlier at almost 300000. Duplex and two family

conversion building types have outliers at both ends, with the maximum of two family

```
plt.figure(figsize=(10,6))
sns.boxplot(y = "SalePrice", x = "Bldg Type", data=Ames )
plt.show()
```



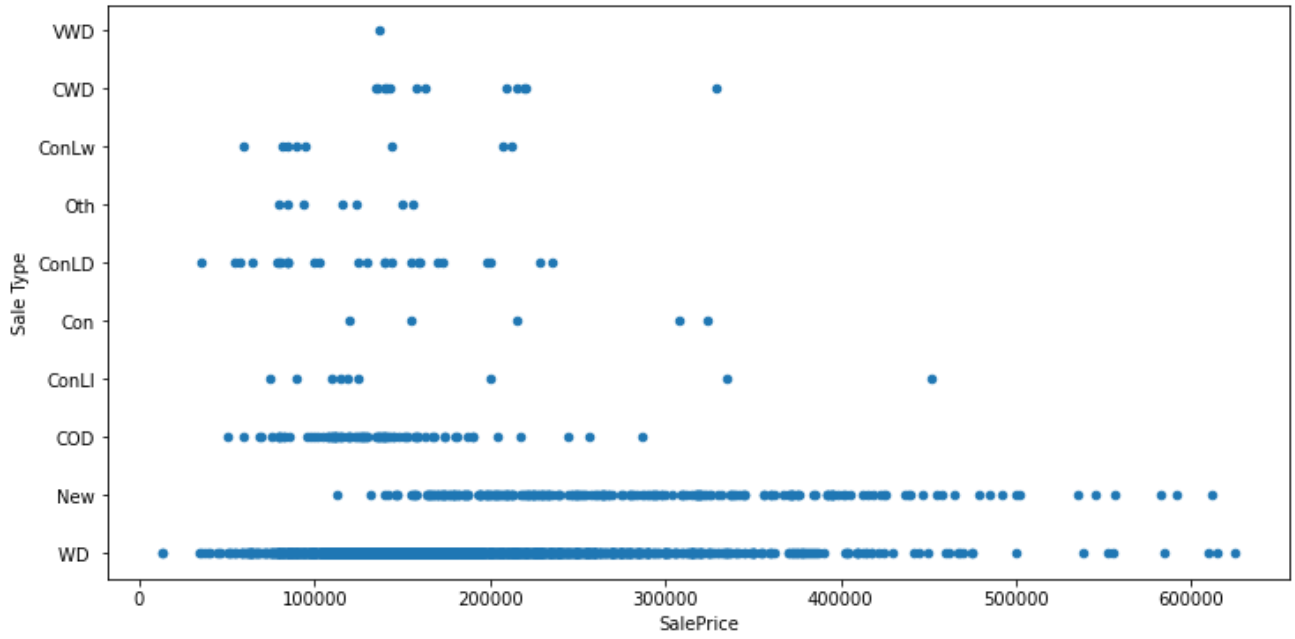
The relationship between Sale Price and house style of dwelling is plotted in a boxplot. The one story dwelling and two story dwelling are more prominent in the plot, as they command the highest range of sale price in the dataset. The outliers of the one story stretch from the maximum to over 600000, same for the outliers of the two story. However, the two and one-half story, with second floor finished has an outlier between 400000 and 500000, with a normal skewness. The one and one-half story, second floor unfinished has a negative skewness, same with the split level. One story and two story both have positive skewness.

```
plt.figure(figsize=(10,6))
sns.boxplot(y = "SalePrice", x = "House Style", data=Ames )
plt.show()
```

The sale price and sale type are entities that are of interest to me. What sale type was most favored? We can see that the Warranty deed, which is a conventional mode was the most in the sale type. The warranty deed with a VA loan appears just once. The contract 15% down payment option was used for five houses. And houses just constructed and sold is a close second to conventional warranty deed sale type.

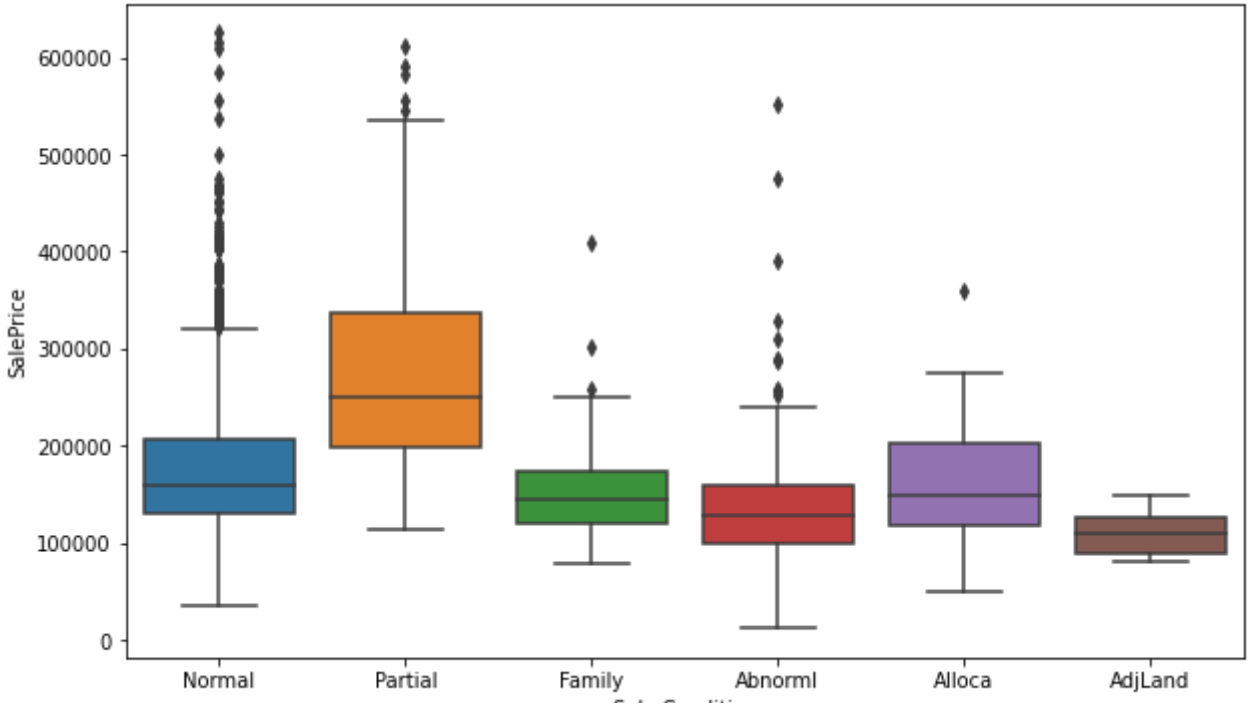
```
Ames.plot(kind ="scatter", x = "SalePrice", y ="Sale Type", figsize = (12,6))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f260c1e93d0>
```



The condition of sale of the dwellings is worthy of note. Normal sale condition appears the most, with partial condition sale coming next. A partial condition is a home that was not completed when last assessed.

```
plt.figure(figsize=(10,6))
sns.boxplot(y = "SalePrice", x = "Sale Condition", data=Ames )
plt.show()
```

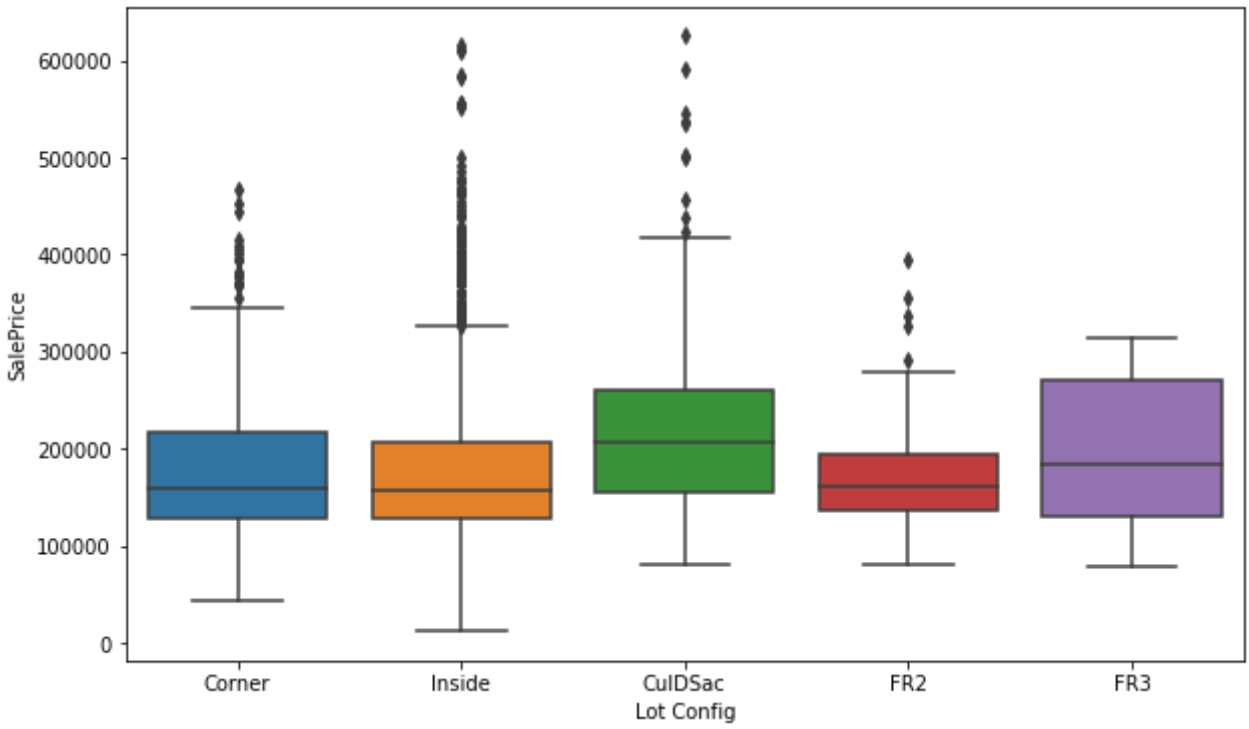


The scatter plot earlier of Sale Price against Neighborhood does not give us a clear representation of the relationship between the two columns, however, a box plot will show a clearer picture. We can see the Stone Brook neighborhood is the most expensive neighborhood in the set, with Northridge Heights coming a close second. Landmark has only one house in the dataset provided. Fifteen neighborhoods have outliers.

```
plt.figure(figsize=(12,6))
plt.xticks(rotation=45)
sns.boxplot(y = "SalePrice", x = "Neighborhood", data=Ames )
plt.show()
```

How does the lot configuration affect the sale price? That will be obvious in a boxplot. Houses with inside lots sold a lot more and boast outliers going over the 600000 sale price, with positive skewness. Corner lots also have a positive skewness. The Cul-de-sac is quite prominent, as it is heads and shoulders above all other lot configuration, with a minimum of almost 100000, maximum of over 400000 and median of 200000. It is also perfectly skewed. Frontage on three sides of the property have no outliers and the skewness is positive.

```
plt.figure(figsize=(10,6))
sns.boxplot(y = "SalePrice", x = "Lot Config", data=Ames )
plt.show()
```

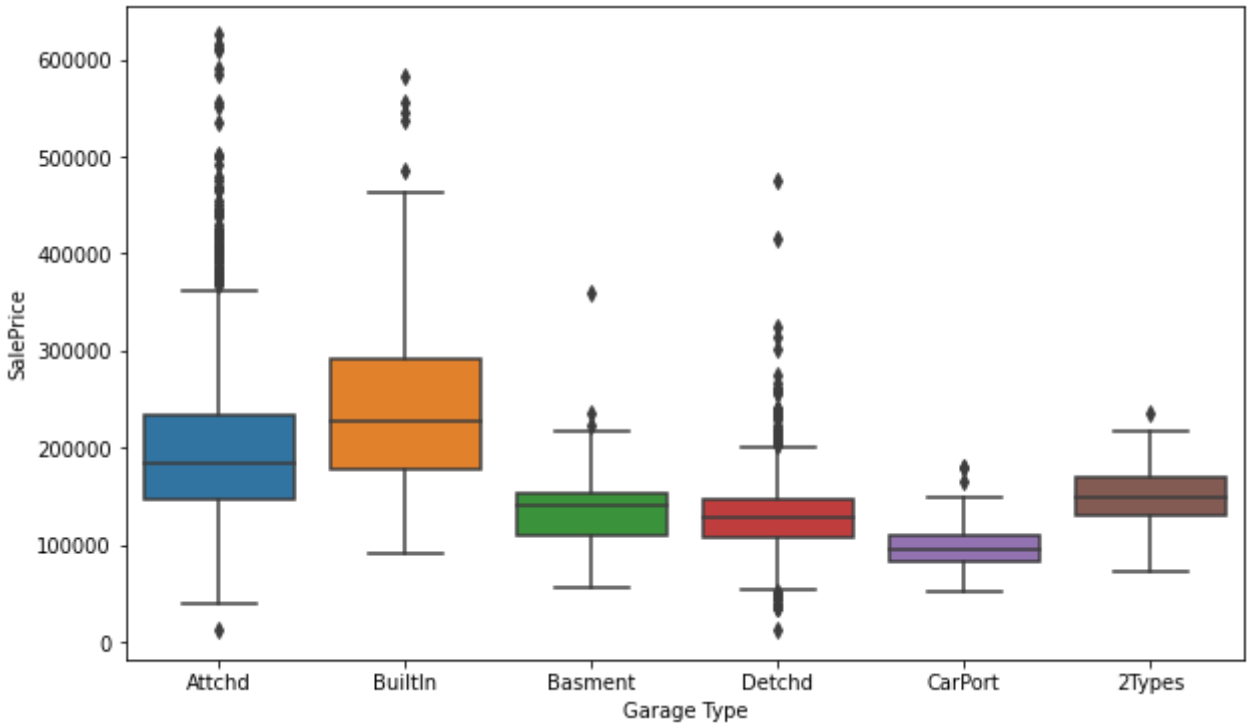


The garage type is also a key component to be checked against Sale Price. houses with the garage attached to the home appears to have more value than the others, with the outliers stretching over 600000 and one outlier at 10000, maximum at almost 400000 and minimum at 50000. A built-in garage also gives the house value, as that means there is a room above the garage, with outliers almost reaching 600000. Both Built-in and attached garage types are positively skewed.

Basement garage is negatively skewed, with the median close to the 75th percentile. Garages detached from home are normally skewed, with outliers below the minimum and above the maximum stretching to almost 500000. Houses with more than one type of garage and car port are normally skewed and have only a couple of outliers.

```
plt.figure(figsize=(10,6))
sns.boxplot(y = "SalePrice", x = "Garage Type", data=Ames )
```

```
plt.show()
```

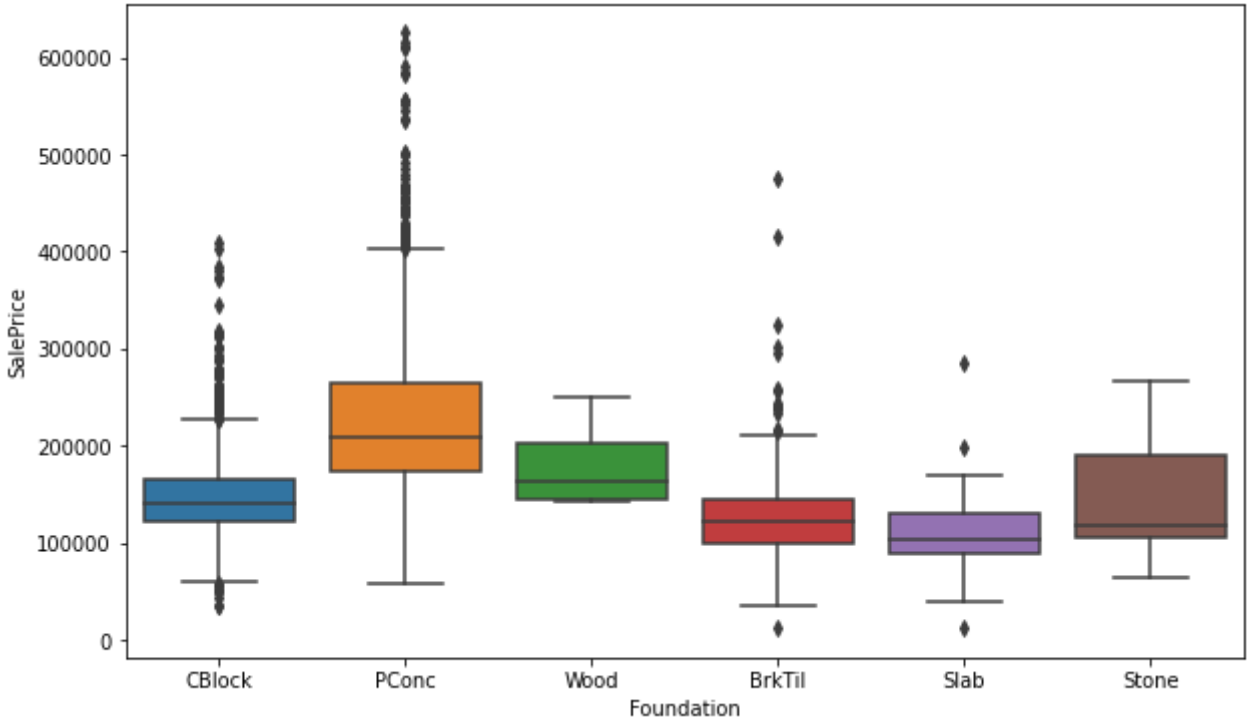


The zoning classification of the sale of an house, will let us know for what purpose a property was purchased. The Residential low density zoning is chief amongst this, a positively skewed entity, with outliers in the maximum region going as high as over 600000. Residential medium density is also a prominent entity, as it has outliers at both the minimum and the maximum, with outliers ranging from 10000 to almost 500000. Residential high density, commercial, Industrial and Agriculture zoning have no outliers.

```
plt.figure(figsize=(10,6))
sns.boxplot(y = "SalePrice", x = "MS Zoning", data=Ames )
plt.show()
```

One of the things home buyers want to know is how strong the foundation of a building is. Will this have an effect on the price? I think so. Houses with poured concrete foundation sold for the highest price, with outliers in the maximum going as high as over 600000. Cinder block foundation and broken tile foundation houses also seem to cost a lot.


```
plt.figure(figsize=(10,6))
sns.boxplot(y = "SalePrice", x = "Foundation", data=Ames )
plt.show()
```



CONCLUSION

Summary of transformations performed on datasets

- Removed five rows in Ground Living Area exceeding 4000, as advised in the data documentation
- Removed PID as it is no longer need after data cleaning aspect is completed.
- Removed Alley, Pool QC, Fence and Misc Feature columns for having a low number of values.

No removals were done during exploratory data analysis.

The original Ames Housing dataset had 2930 rows and 82 columns, a total of 5 columns and 5 rows were removed based on the justification above. We were left with 2925 rows and 77 columns.

Summary on datasets

The analysis of the dataset has shown that a number of features determine the sale price of a property/dwelling. The ground living area, the neighbourhood, the number of bedrooms above ground, down to the type of garage, type of foundation, the house style and the building type. The year a house was built also plays a certain role in how much it will be sold, with old houses commanding a price, for their historical value and modern houses commanding a high price, for their modernity. We believe that this dataset will proviable a useful set of features that can be used to create a model for the forecasting of price of houses over the coming years.