Project in Python Data Products Specialization

Dataset - House Prices - Advanced Regression Techniques data from Kaggle Website: [https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/data (https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/data)](https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/data)

Dataset Description: This data has 79 explanatory variables describing almost every aspect of residential homes in Ames, Iowa. There is a competition on Kaggle to predict the final price of each home.

Variables: Some of the variables have missing values. For such variables statistics such as correlation were computed if at least a 1000 values were common for the two variables. For mean etc also, the missing values are as of now ignored.

Dataset author: The Ames housing dataset was compiled by Dean De Cock for use in data science education.

In [3]:

```
# Load the dataset:
import pandas as pd
import zipfile
df_zip = zipfile.ZipFile('D:\TRL\OneDrive\CourseraPython\ProjectCourse1\KaggleHousePr
df = pd.read_csv(df_zip.open('train.csv'))
type(df)
```

Out[3]:

```
pandas.core.frame.DataFrame
```

In [4]:

```
df
```

Out[4]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandCon
0	1	60	RL	65.0	8450	Pave	NaN	Reg	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	
...	...	...	...	...	...	...	...	...	
1455	1456	60	RL	62.0	7917	Pave	NaN	Reg	
1456	1457	20	RL	85.0	13175	Pave	NaN	Reg	
1457	1458	70	RL	66.0	9042	Pave	NaN	Reg	
1458	1459	20	RL	68.0	9717	Pave	NaN	Reg	
1459	1460	20	RL	75.0	9937	Pave	NaN	Reg	

1460 rows × 81 columns

df[10:]

In [5]:

```
# To select all the rows in the dataframe where Alley value is not NaN
alley_notnan = df[~df['Alley'].isnull()]
alley_notnan
```

Out[5]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandCon
21	22	45	RM	57.0	7449	Pave	Grvl	Reg	
30	31	70	C (all)	50.0	8500	Pave	Pave	Reg	
56	57	160	FV	24.0	2645	Pave	Pave	Reg	
79	80	50	RM	60.0	10440	Pave	Grvl	Reg	
87	88	160	FV	40.0	3951	Pave	Pave	Reg	
...	...	...	...	...	...	...	...	...	
1404	1405	50	RL	60.0	10410	Pave	Grvl	Reg	
1414	1415	50	RL	64.0	13053	Pave	Pave	Reg	
1427	1428	50	RL	60.0	10930	Pave	Grvl	Reg	
1432	1433	30	RL	60.0	10800	Pave	Grvl	Reg	
1454	1455	20	FV	62.0	7500	Pave	Pave	Reg	

91 rows × 81 columns

In [6]:

```
df['MSSubClass'].isna()
```

Out[6]:

```
0       False
1       False
2       False
3       False
4       False
        ...
1455    False
1456    False
1457    False
1458    False
1459    False
Name: MSSubClass, Length: 1460, dtype: bool
```

There are so many variables associated with the house price. Let us see which of them most effect the sale price of the house the most.

df.corr() - to find the correlation

is used to find Pearson's correlation among the columns in the dataframe df. Any NaN value is automatically excluded. Any non- numeric data type or colummns are excluded. We can specify min_periods, which is the min number of observations required per pair of columns to have a valid result. Ex: at least 100 obs needed per column to arrive at a valid correlation value.

In [7]:

```
df['SalePrice']
```

Out[7]:

```
0       208500
1       181500
2       223500
3       140000
4       250000
         ...
1455    175000
1456    210000
1457    266500
1458    142125
1459    147500
Name: SalePrice, Length: 1460, dtype: int64
```

In [8]:

```
# First let us examine the missing data patterns:
df_new = df.drop(['Id'], axis = 1)
correlation_mat = df_new.corr(method = 'pearson', min_periods = 1000)
# min_periods = 1000 means that there are at least 1000 observations
# required per pair of columns to have a valid result.
mat = round(correlation_mat,2)
#mat[35:].iloc[:,31:40]
mat
```

Out[8]:

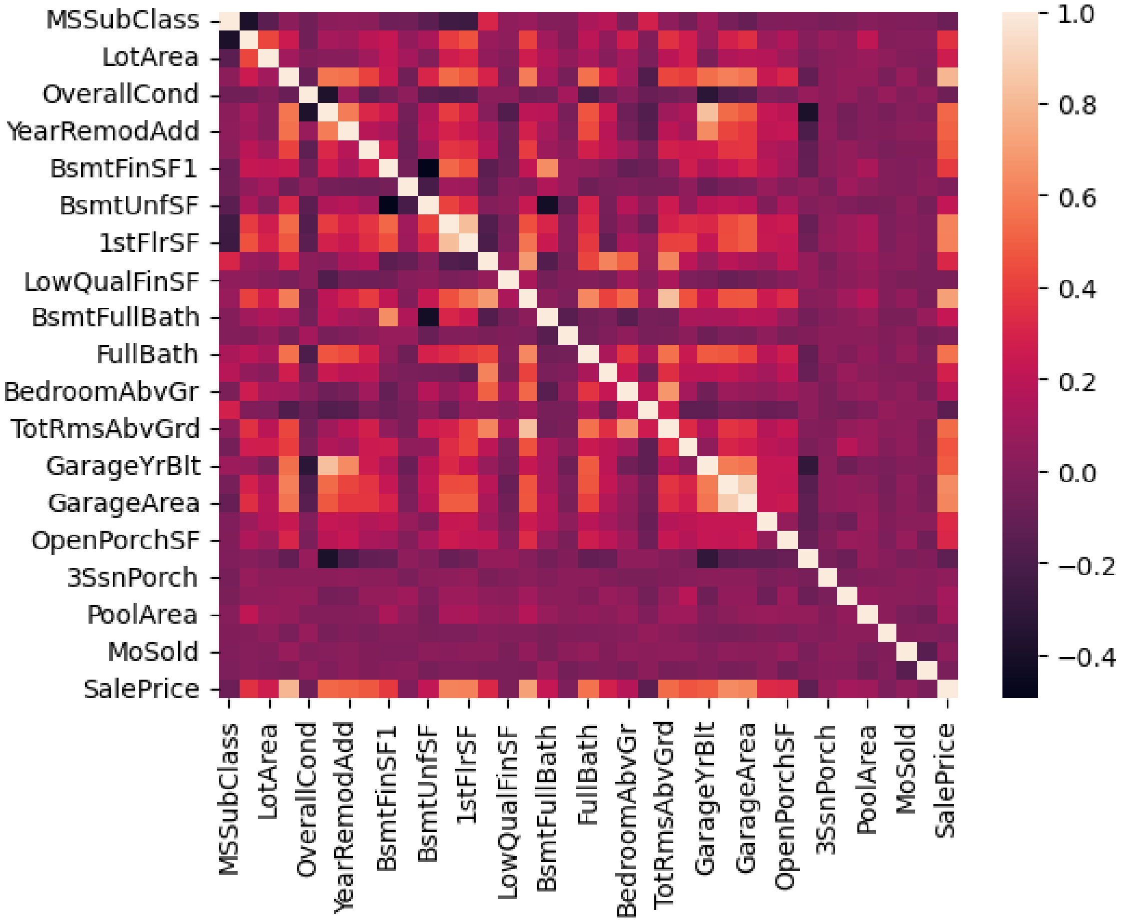
	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	M
MSSubClass	1.00	-0.39	-0.14	0.03	-0.06	0.03	0.04	
LotFrontage	-0.39	1.00	0.43	0.25	-0.06	0.12	0.09	
LotArea	-0.14	0.43	1.00	0.11	-0.01	0.01	0.01	
OverallQual	0.03	0.25	0.11	1.00	-0.09	0.57	0.55	
OverallCond	-0.06	-0.06	-0.01	-0.09	1.00	-0.38	0.07	
YearBuilt	0.03	0.12	0.01	0.57	-0.38	1.00	0.59	
YearRemodAdd	0.04	0.09	0.01	0.55	0.07	0.59	1.00	
MasVnrArea	0.02	0.19	0.10	0.41	-0.13	0.32	0.18	
BsmtFinSF1	-0.07	0.23	0.21	0.24	-0.05	0.25	0.13	
BsmtFinSF2	-0.07	0.05	0.11	-0.06	0.04	-0.05	-0.07	

In [9]:

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

sns.heatmap(correlation_mat)

```

Out[9]:

<AxesSubplot:>



From here, we see that the sale price is more correlated with the following variables:

1.OverallQual (Overall Quality of the house)

2.YearRemodAdd (Year of remodelling/construction)

3. GrLivArea (Above grade ground living area in sq feet)
4. MasVnrArea (Masonry Veneer Area in sq feet),
5. TotalBsmtSF (Total basement area in sq feet),
6. 1stFlrSF (1st floor area in sq feet),
7. FullBath (Full bathrooms above grade).

There is a greater than or equal to 0.5 correlation between these variables and SalesPrice (paired correlation).

In [1]:

```
# Let us calculate a new correlation coefficient for these variables
import pandas as pd
df_new = pd.DataFrame()
df_new = pd.concat([df.SalePrice, df.OverallQual,df.YearRemodAdd,df.GrLivArea,df.Mas
correlation_matnew = df_new.corr(method = 'pearson', min_periods = 1000)
mat_new = round(correlation_matnew,2)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_18448\2312607258.py in <module>
      2 import pandas as pd
      3 df_new = pd.DataFrame()
----> 4 df_new = pd.concat([df.SalePrice, df.OverallQual,df.YearRemodAdd,df.GrLivArea,df.MasVnrArea,df['1stFlrSF'],df.TotalBsmtSF,df.FullBath],axis=1)
      5 correlation_matnew = df_new.corr(method = 'pearson', min_periods = 1000)
      6 mat_new = round(correlation_matnew,2)

NameError: name 'df' is not defined
```

In [11]:

```
mat_new
```

Out[11]:

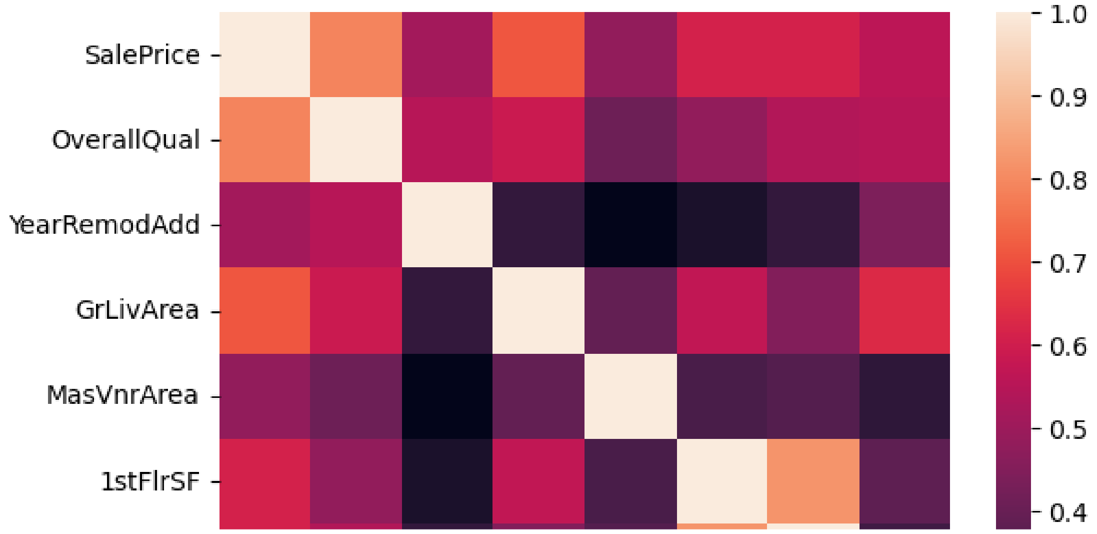
	SalePrice	OverallQual	YearRemodAdd	GrLivArea	MasVnrArea	1stFlrSF	To
SalePrice	1.00	0.79	0.51	0.71	0.48	0.61	
OverallQual	0.79	1.00	0.55	0.59	0.41	0.48	
YearRemodAdd	0.51	0.55	1.00	0.29	0.18	0.24	
GrLivArea	0.71	0.59	0.29	1.00	0.39	0.57	
MasVnrArea	0.48	0.41	0.18	0.39	1.00	0.34	
1stFlrSF	0.61	0.48	0.24	0.57	0.34	1.00	
TotalBsmtSF	0.61	0.54	0.29	0.45	0.36	0.82	
FullBath	0.56	0.55	0.44	0.63	0.28	0.38	

In [12]:

```
sns.heatmap(mat_new)
```

Out[12]:

```
<AxesSubplot:>
```



In [13]:

```
import numpy as np
GrLivArea_mean = np.mean(df['GrLivArea'])
GrLivArea_mean
```

Out[13]:

```
1515.463698630137
```

In [14]:

```
# Now, let us compare the distribution of data for sales price less than
# and greater than the mean.
Greater = []
Lesser = []
for i in range(len(df['SalePrice'])):
    if df['GrLivArea'].iloc[i] >= GrLivArea_mean:
        Greater.append(df.iloc[i])
    else:
        Lesser.append(df.iloc[i])
#The following doesn't work
#Greater = [df['SalePrice'] for d in df['SalePrice'] if df['SalePrice'] >= GrLivArea_
```

In [15]:

```
len(df['SalePrice']), df['SalePrice'].iloc[0]
```

Out[15]:

```
(1460, 208500)
```

In [16]:

```
df.iloc[0]
```

Out[16]:

```
Id                    1
MSSubClass           60
MSZoning             RL
LotFrontage        65.0
LotArea            8450
                  ...
MoSold                2
YrSold             2008
SaleType             WD
SaleCondition    Normal
SalePrice        208500
Name: 0, Length: 81, dtype: object
```

In [17]:

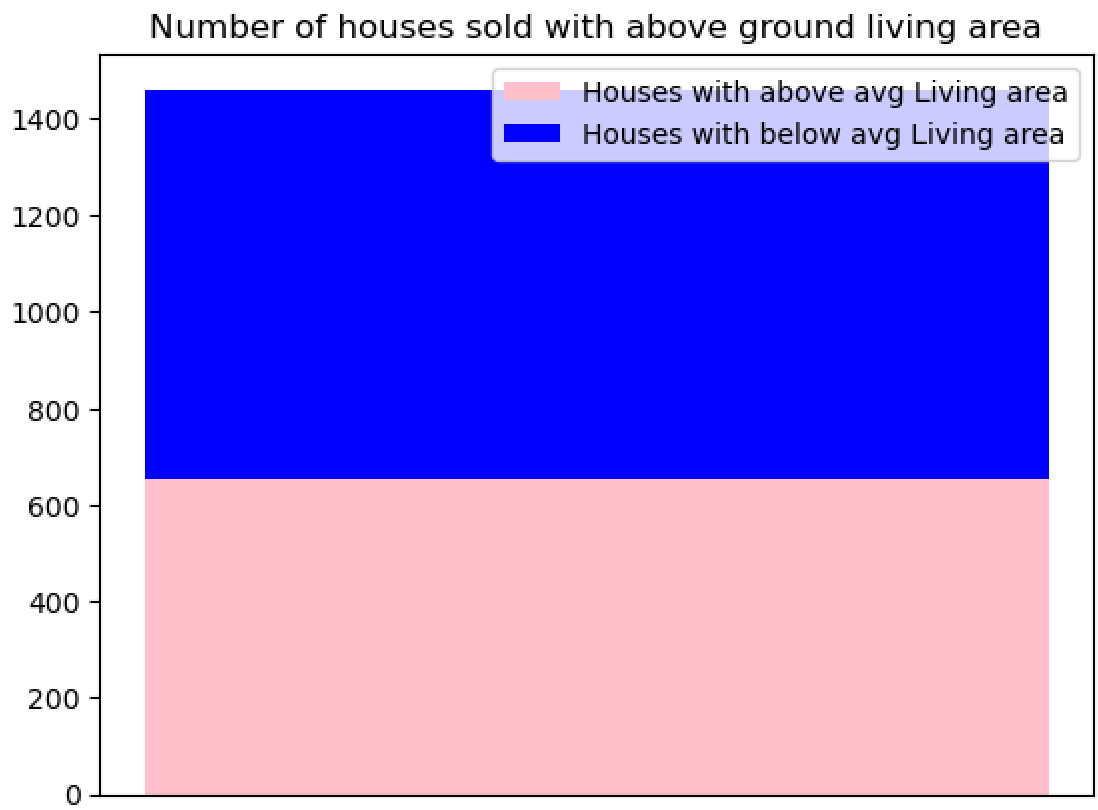
```
len(Greater), len(Lesser), len(df['SalePrice']), type(Greater)
```

Out[17]:

```
(653, 807, 1460, list)
```

In [18]:

```
# Stacked bar chart
import matplotlib.pyplot as plt
index = [1]
p1 = plt.bar(index, len(Greater), color = 'pink')
p2 = plt.bar(index, len(Lesser), bottom = len(Greater), color = 'blue')
plt.gca().set(title = "Number of houses sold with above ground living area")
plt.xticks([])
plt.legend((p1[0],p2[0]),('Houses with above avg Living area','Houses with below avg
plt.show()

```



Is there a relationship between Overall Quality of the house and the SalePrice?

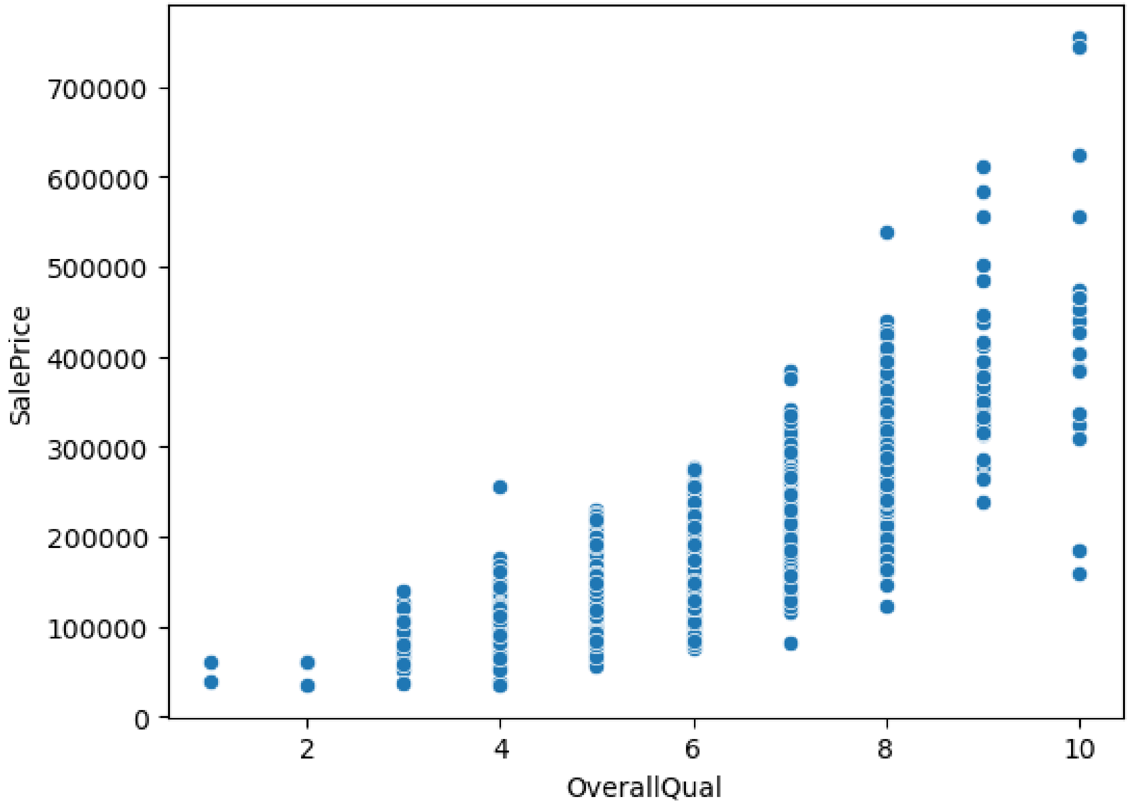
This is our first goal, from the correlation matrix above, there may be a relationship between this variable and SalePrice. Our next question: What is the expected saleprice of the house if we consider only the most correlated variable? The most correlated variable is the OverallQual - i.e the overall material and finish of the house. Let us first look at the scatterplot.

In [19]:

```
import seaborn as sns
sns.scatterplot(x = df['OverallQual'], y = df['SalePrice'])
```

Out[19]:

```
<AxesSubplot:xlabel='OverallQual', ylabel='SalePrice'>
```



In [20]:

```
data1 = []
data1 = pd.concat([df['OverallQual'],df['SalePrice']],axis = 1)
#sns.lmplot(x = df['OverallQual'], y = df['SalePrice'], data = data1)
#sns.lmplot(x = data1['OverallQual'], y = data1['SalePrice'], data = data1);
sns.lmplot(x = 'OverallQual', y = 'SalePrice', data = data1);
ax = plt.gca()
ax.set_title("Regression Line Predicting the SalePrice dependant on the Overall Qual
```

Out[20]:

```
Text(0.5, 1.0, 'Regression Line Predicting the SalePrice dependant on the
Overall Quality of the house ')
```

In [21]:

```
data1['OverallQual']
```

Out[21]:

```
0       7
1       6
2       7
3       7
4       8
       ..
1455    6
1456    6
1457    7
1458    5
1459    5
Name: OverallQual, Length: 1460, dtype: int64
```

In [22]:

```
# Now, let us add the line of best fit:
data1 = []
data1 = pd.concat([df['GrLivArea'],df['SalePrice']],axis =1)
data1
```

Out[22]:

	GrLivArea	SalePrice
0	1710	208500
1	1262	181500
2	1786	223500
3	1717	140000
4	2198	250000
...	...	...
1455	1647	175000
1456	2073	210000
1457	2340	266500
1458	1078	142125
1459	1256	147500

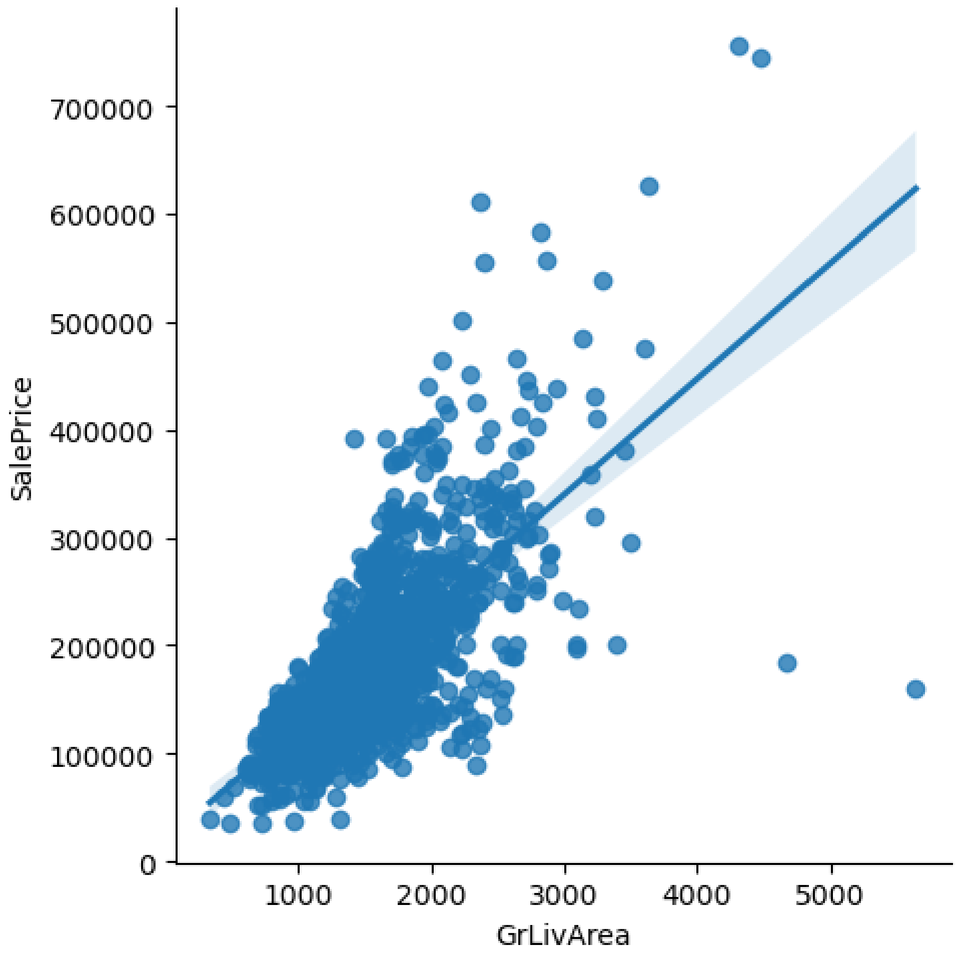
1460 rows × 2 columns

In [29]:

```
sns.lmplot(x = 'GrLivArea', y = 'SalePrice', data = data1)
```

Out[29]:

```
<seaborn.axisgrid.FacetGrid at 0x2b9cac4dfd0>
```



In [32]:

```
# to calculate the regression line:
from scipy import stats
slope, intercept, r_value, p_value, std_error = stats.linregress(data1["GrLivArea"],
slope, intercept, r_value, p_value, std_error
```

Out[32]:

```
(107.1303589658252,
 18569.02585648722,
 0.7086244776126522,
 4.518033646776791e-223,
 2.7936210388899063)
```

The Regression line:

A regression line has been predicted and plotted above. This regression line is given by: SalePrice(i) = 107.13* GrLivArea(i) + 18569.025

p- value = 0 => there is a significant relationship between X and Y. We reject the hypothesis that beta1/ the slope value = 0.

The correlation between X and Y is: r_value = 0.7086 std error = 2.79, this is the stad error for beta1, ie. the slope, this shows that the 95% confidence interval for beta1 is slope +- the standard error.

We conclude that the Sale price of a house is dependent on the GrLivArea (i.e. above grade (ground) living area in square feet).

In [ ]:

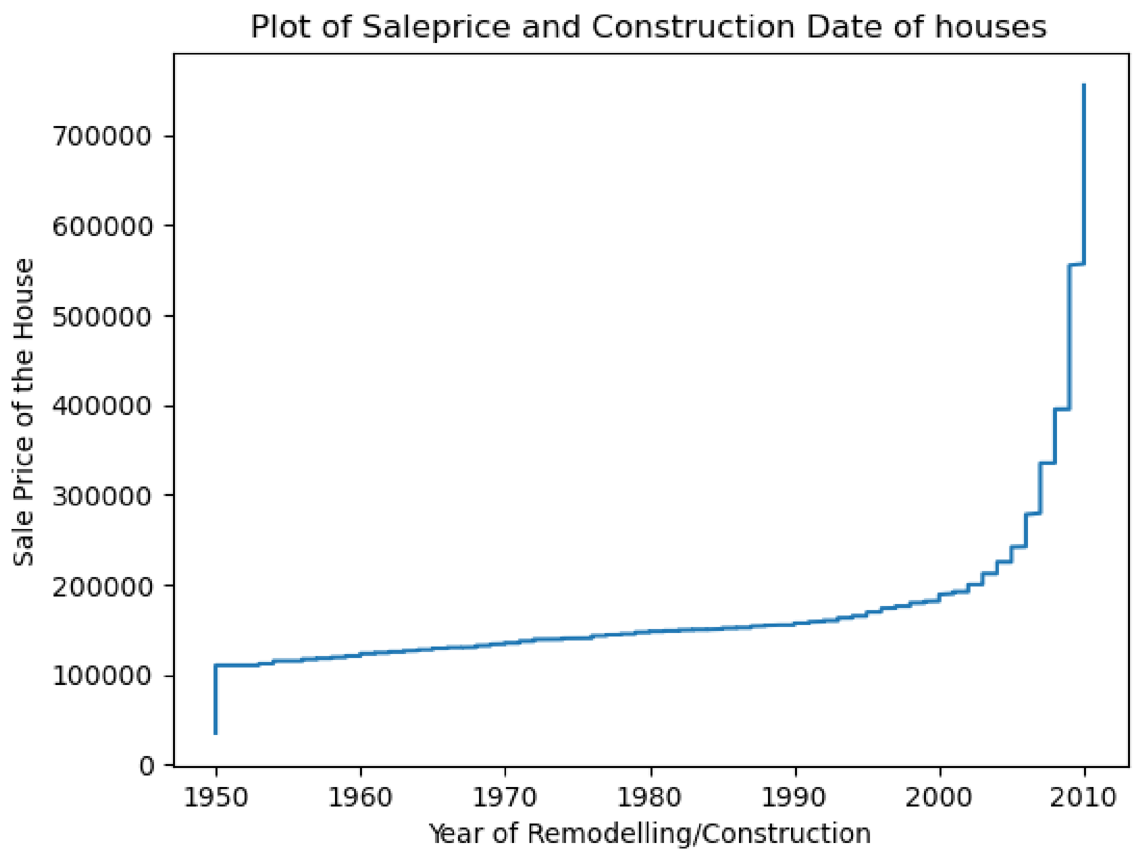
```
#### From here, we see that the sale price is more correlated with
##OverallQual, YearRemodAdd, GrLivArea, MasVnrArea, TotalBsmtSF, 1stFlrSF, FullBath.

```

In [42]:

```
# Let us plot the variable: YearRemodAdd and see whether the relationship
# between YearRemodAdd and SalePrice is strong and whether it is linear.
# YearRemodAdd is the remodel date (it is the same as construction date if
# there has been no remodelling)
data2 = pd.concat([df["YearRemodAdd"],df["SalePrice"]], axis = 1)
X = list(df["YearRemodAdd"])
X.sort()
Y = list(df["SalePrice"])
Y.sort()
plt.gca().set(xlabel = 'Year of Remodelling/Construction', ylabel =
'Sale Price of the House', title = 'Plot of Saleprice and Construction Date of houses
plt.plot(X,Y)
```

Out[42]:

```
[<matplotlib.lines.Line2D at 0x2b9cb366df0>]
```

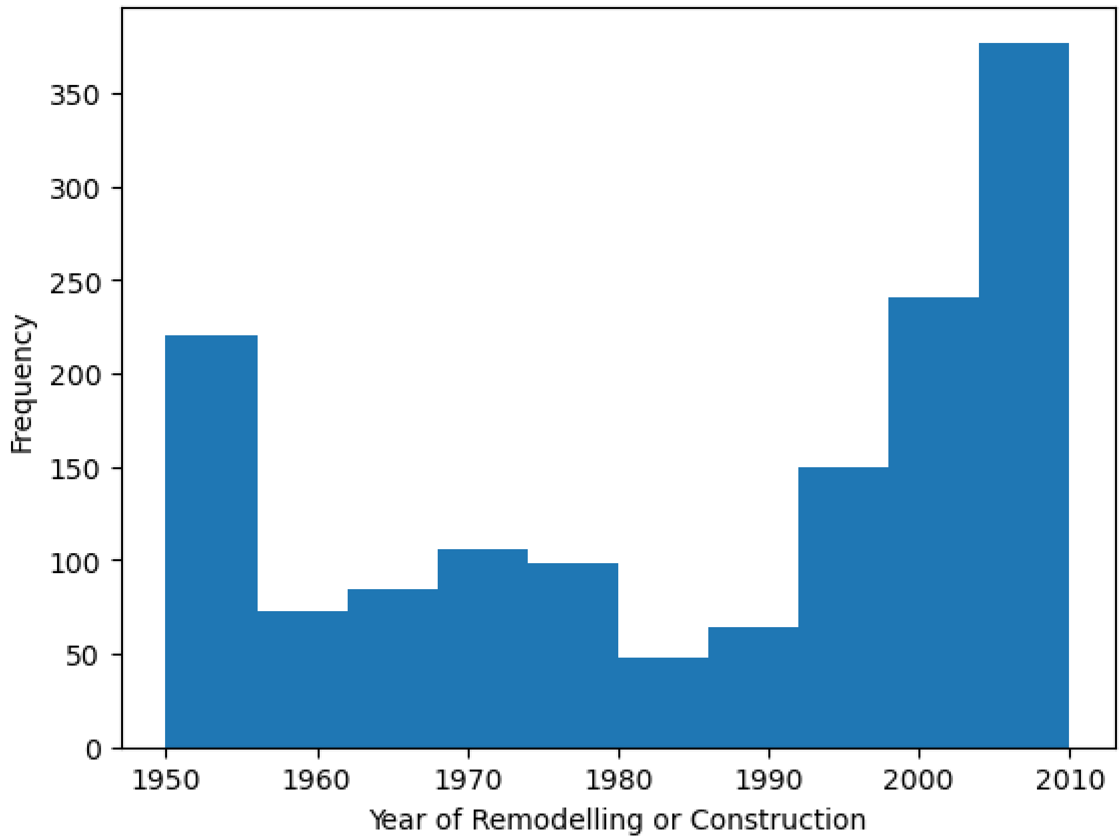


In [43]:

```
# Histogram of the Year of Remodelling or construction
plt.gca().set(xlabel = 'Year of Remodelling or Construction', ylabel = 'Frequency')
plt.hist(X)
```

Out[43]:

```
(array([220.,  72.,  84., 106.,  98.,  48.,  64., 150., 241., 377.]),
 array([1950., 1956., 1962., 1968., 1974., 1980., 1986., 1992., 1998.,
        2004., 2010.]),
 <BarContainer object of 10 artists>)
```

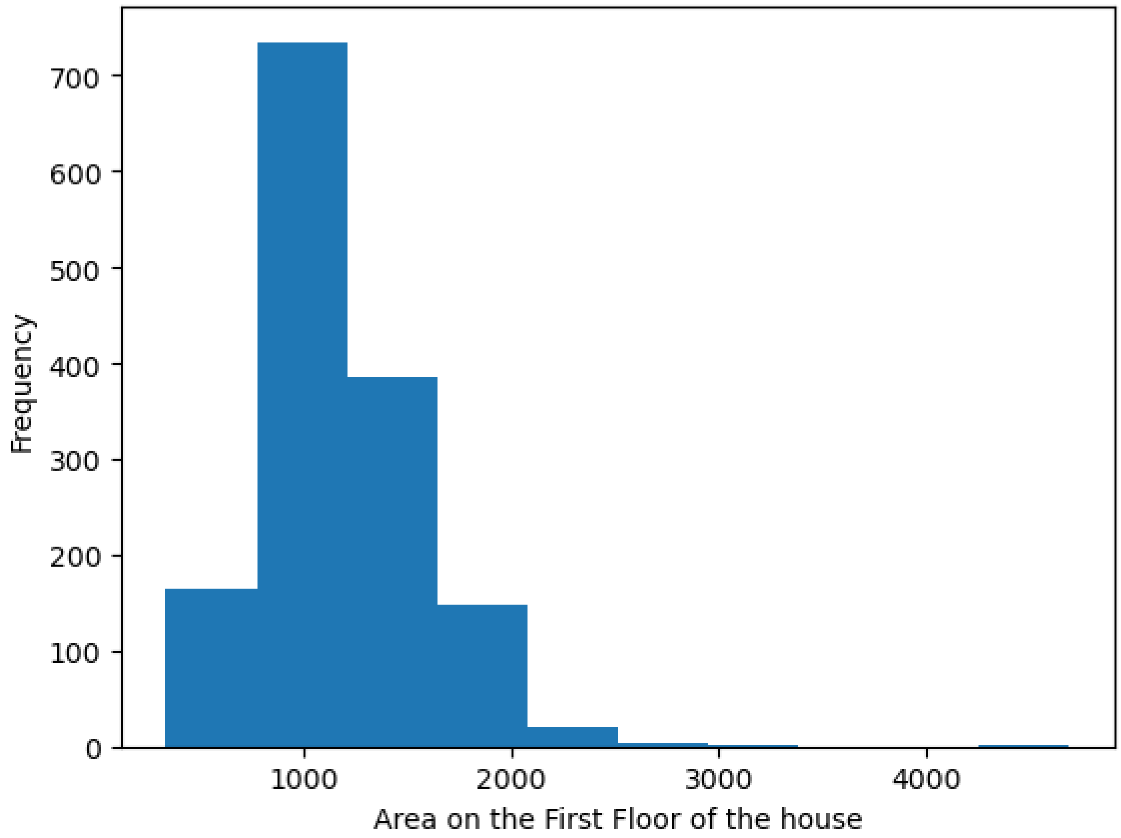


In [49]:

```
# Histogram of the first floor square feet (1stFlrSF)
X = list(df['1stFlrSF'])
plt.gca().set(xlabel = 'Area on the First Floor of the house', ylabel = 'Frequency')
plt.hist(X)
```

Out[49]:

```
(array([165., 735., 385., 147.,  21.,   4.,   2.,   0.,   0.,   1.]),
 array([ 334. ,  769.8, 1205.6, 1641.4, 2077.2, 2513. , 2948.8, 3384.6,
        3820.4, 4256.2, 4692. ]),
 <BarContainer object of 10 artists>)
```



In [48]:

```
import numpy
numpy.var(X)**(0.5), numpy.mean(X)
# This shows that the average first floor area is 1160 sq feet.
```

Out[48]:

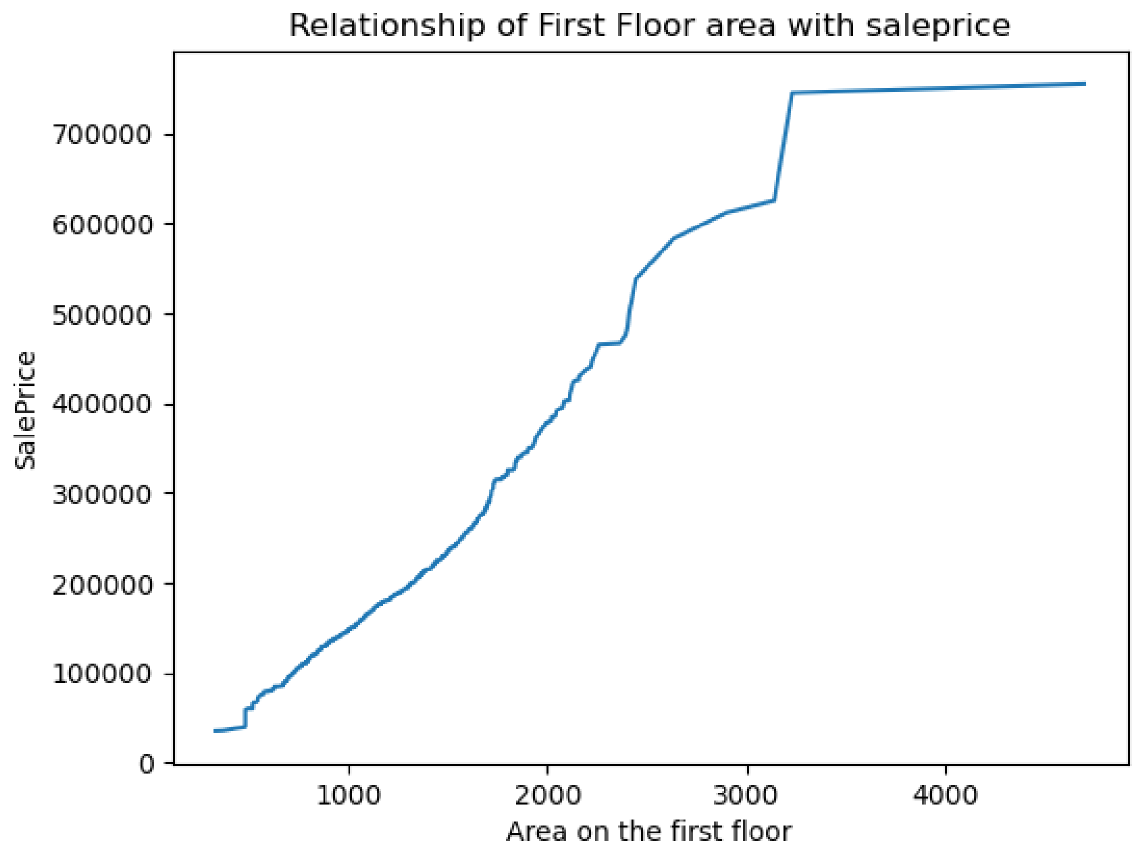
```
(386.45532230228963, 1162.626712328767)
```

In [52]:

```
# let us plot this data with SalePrice, to see whether the number of
# square feet in the first floor influences the sale price.
X = list(df['1stFlrSF'])
X.sort()
plt.gca().set(xlabel = 'Area on the first floor', ylabel = 'SalePrice',
              title = 'Relationship of First Floor area with saleprice')
plt.plot(X,Y)
```

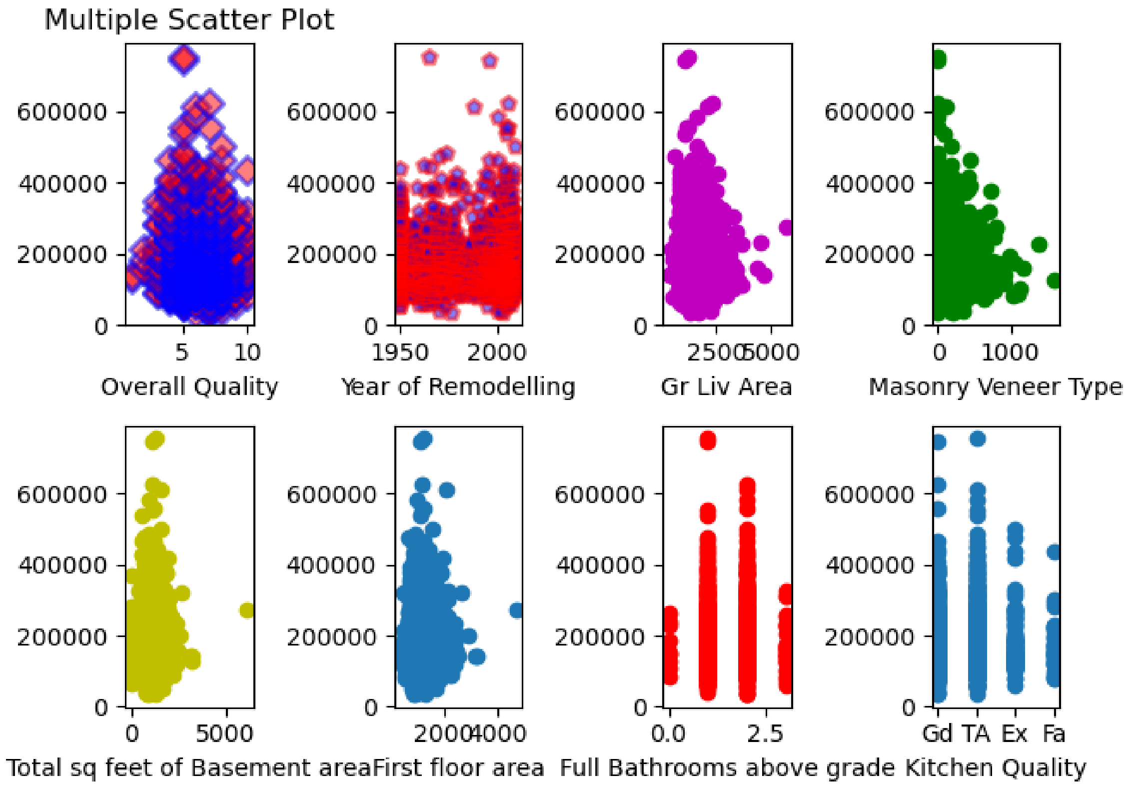
Out[52]:

```
[<matplotlib.lines.Line2D at 0x2b9ccec1a30>]
```



In [153]:

```
# Scatterplot between SalePrice and OverallQual, YearRemodAdd, GrLivArea,
# MasVnrArea, TotalBsmtSF, 1stFlrSF, FullBath. There is a greater than or
# equal to 0.5 correlation between these variables and SalesPrice (paired correlatio
x1 = list(df["OverallQual"])
x2 = list(df["YearRemodAdd"])
x3 = list(df["GrLivArea"])
x4 = list(df["MasVnrArea"])
x5 = list(df["TotalBsmtSF"])
x6 = list(df["1stFlrSF"])
x7 = list(df["FullBath"])
x8 = list(df["KitchenQual"])
plt.subplot(2,4,1) # rows =2, columns = 4, count =1
plt.title("Multiple Scatter Plot")
plt.scatter(x1,Y,c ='r',linewidths = 2, marker = "D", edgecolor = "b",
            s = 70, alpha = 0.5)
plt.xlabel('Overall Quality')
plt.subplot(2,4,2)
plt.scatter(x2,Y, c = "b", linewidths = 2, marker = "p", edgecolor = "red", alpha = 
plt.xlabel("Year of Remodelling")
plt.subplot(2,4,3)
plt.scatter(x3,Y, c = 'm')
plt.xlabel("Gr Liv Area")
plt.subplot(2,4,4)
plt.scatter(x4,Y,c = 'g')
plt.xlabel("Masonry Veneer Type")
plt.subplot(2,4,5)
plt.scatter(x5,Y,c = 'y')
plt.xlabel("Total sq feet of Basement area")
plt.subplot(2,4,6)
plt.scatter(x6,Y)
plt.xlabel("First floor area")
plt.subplot(2,4,7)
plt.scatter(x7,Y, c = 'r')
plt.xlabel("Full Bathrooms above grade")
plt.subplot(2,4,8)
plt.scatter(x8,Y)
plt.xlabel("Kitchen Quality")
plt.tight_layout()
```


Multiple Scatter Plot
600000
400000
200000
0
5
10
Overall Quality
1950
2000
Year of Remodelling
2500
5000
Gr Liv Area
0
1000
Masonry Veneer Type
0
5000
Total sq feet of Basement area
2000
4000
First floor area
0.0
2.5
Full Bathrooms above grade
Gd
TA
Ex
Fa
Kitchen Quality

In [154]:

```
category1 = []
# this is equal to 1 if the Overall quality is greater than 5,
# this is equal to 0, if the overall quality is less than or equal to 5.
for i in range(len(x1)):
    if x1[i] > 5:
        category1.append(1)
    else:
        category1.append(0)
print(sum(category1)) # number of houses with above 5 category
category1 = pd.DataFrame(category1, columns =['Category_OverallQual'])
data3 = pd.concat([df["GrLivArea"],df["SalePrice"],category1], axis = 1)
data3
```

922

Out[154]:

	GrLivArea	SalePrice	Category_OverallQual
0	1710	208500	1
1	1262	181500	1
2	1786	223500	1
3	1717	140000	1
4	2198	250000	1
...	...	...	...
1455	1647	175000	1
1456	2073	210000	1
1457	2340	266500	1
1458	1078	142125	0
1459	1256	147500	0

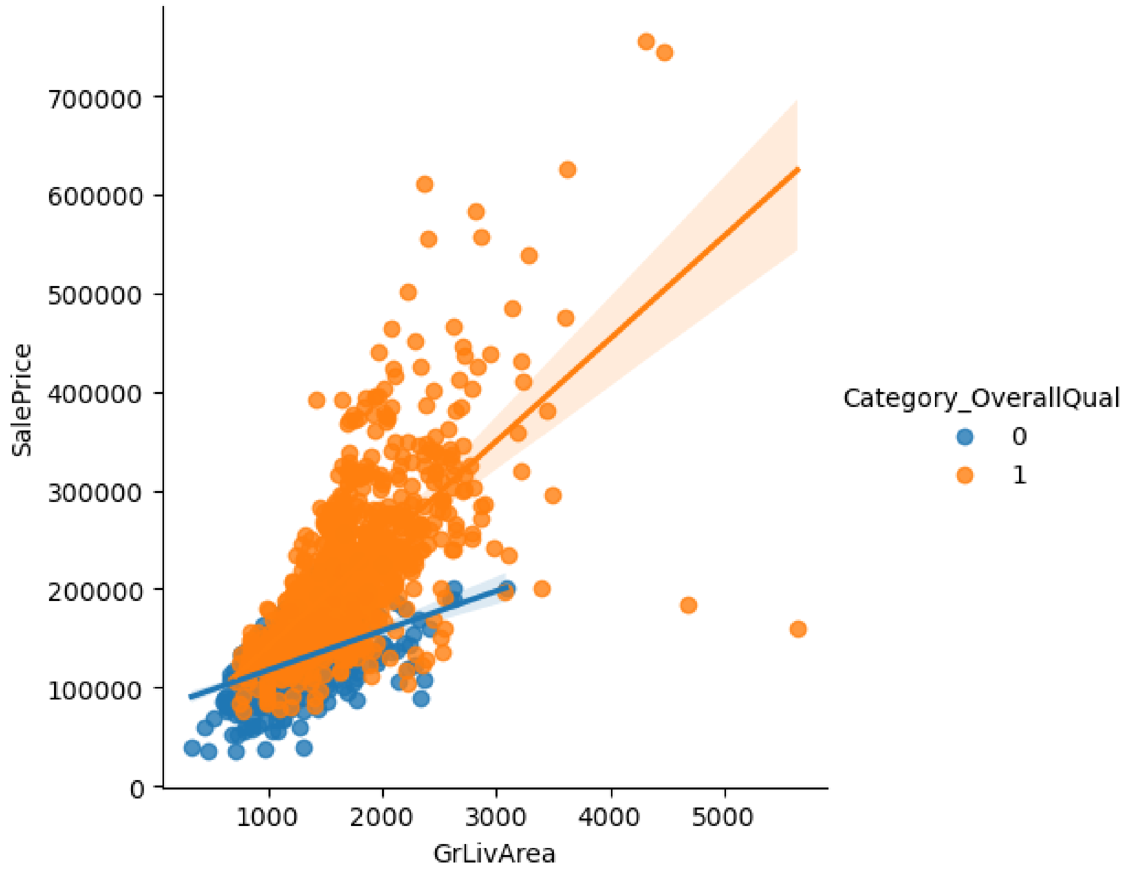
1460 rows × 3 columns

In [155]:

```
1 sns.lmplot(x = "GrLivArea", y = "SalePrice", hue = "Category_OverallQual", data = da
```

Out[155]:

```
<seaborn.axisgrid.FacetGrid at 0x2b9d532d220>
```



Decription of the plot above:

We can see that we will get better estimates of the sale price of the house by using two different subsets of the data. In both cases, the predictor is Above grade, ground living area in sq feet. In one case values corresponding to Overall quality above rating 5 are taken, and in another case, values corresponding to Overall Quality below rating 5 is taken.

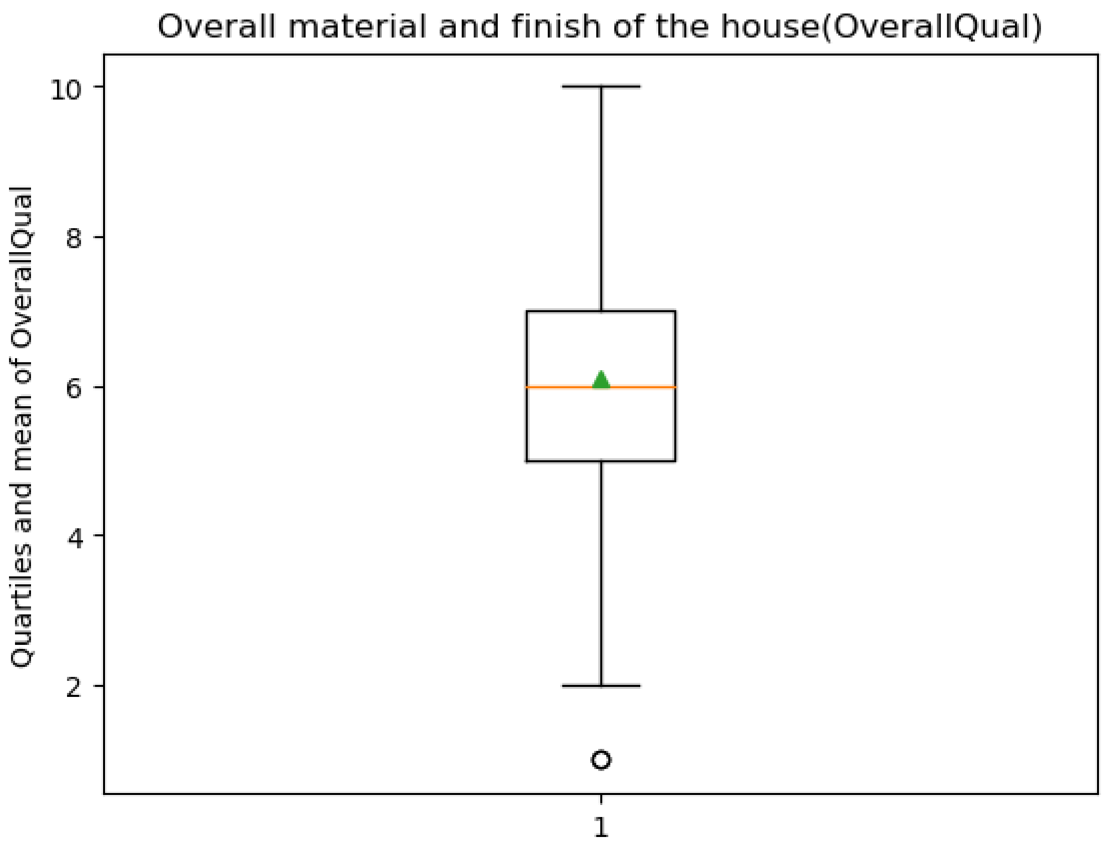
Let us also look at muliple box-plots to describe the above data.

In [156]:

```
ax = plt.boxplot(x1,showmeans = True)
plt.title("Overall material and finish of the house(OverallQual)")
plt.ylabel("Quartiles and mean of OverallQual")
whiskers = [item.get_ydata() for item in ax['whiskers']]
medians = [item.get_ydata() for item in ax['medians']]
boxes = [item.get_ydata() for item in ax['boxes']]
whiskers, medians, boxes
```

Out[156]:

```
([array([5., 2.]), array([ 7., 10.])],
 [array([6., 6.])],
 [array([5., 5., 7., 7., 5.])])
```



In [178]:

```
import matplotlib.pyplot as axs
labels = ['x1','x2','x3','x4','x5','x6','x7','x8']
data4= pd.concat([df["OverallQual"],df["YearRemodAdd"],df["GrLivArea"],df["MasVnrAre
                  df["TotalBsmtSF"],df["1stFlrSF"],df["FullBath"],df["KitchenQual"]],
#data4 = list(data4)
data4, type(data4)
```

Out[178]:

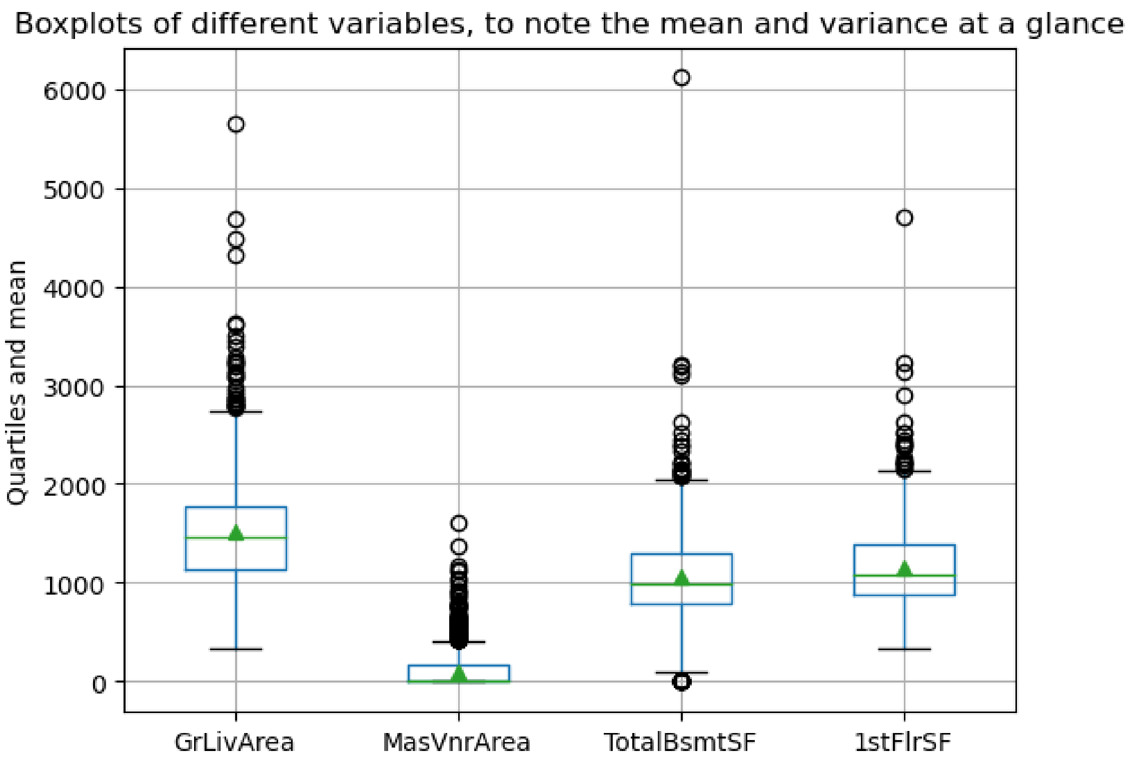
```
(      OverallQual  YearRemodAdd  GrLivArea  MasVnrArea  TotalBsmtSF  1stF
lrSF  \
 0               7          2003       1710       196.0          856
856
 1               6          1976       1262         0.0         1262
1262
 2               7          2002       1786       162.0          920
920
 3               7          1970       1717         0.0          756
961
 4               8          2000       2198       350.0         1145
1145
 ...           ...           ...        ...         ...          ...
...
 1455            6          2000       1647         0.0          953
953
 1456            6          1988       2073       119.0         1542
2073
 1457            7          2006       2340         0.0         1152
1188
 1458            5          1996       1078         0.0         1078
1078
 1459            5          1965       1256         0.0         1256
1256

       FullBath KitchenQual
 0            2          Gd
 1            2          TA
 2            2          Gd
 3            1          Gd
 4            2          Gd
 ...        ...         ...
 1455         2          TA
 1456         2          TA
 1457         2          Gd
 1458         1          Gd
 1459         1          TA

 [1460 rows x 8 columns],
 pandas.core.frame.DataFrame)
```

In [202]:

```
axnew = data4.boxplot(column = ['GrLivArea','MasVnrArea','TotalBsmtSF',"1stFlrSF"], 
plt.title("Boxplots of different variables, to note the mean and variance at a glanc
plt.ylabel("Quartiles and mean")
```

Out[202]:

```
Text(0, 0.5, 'Quartiles and mean')
```

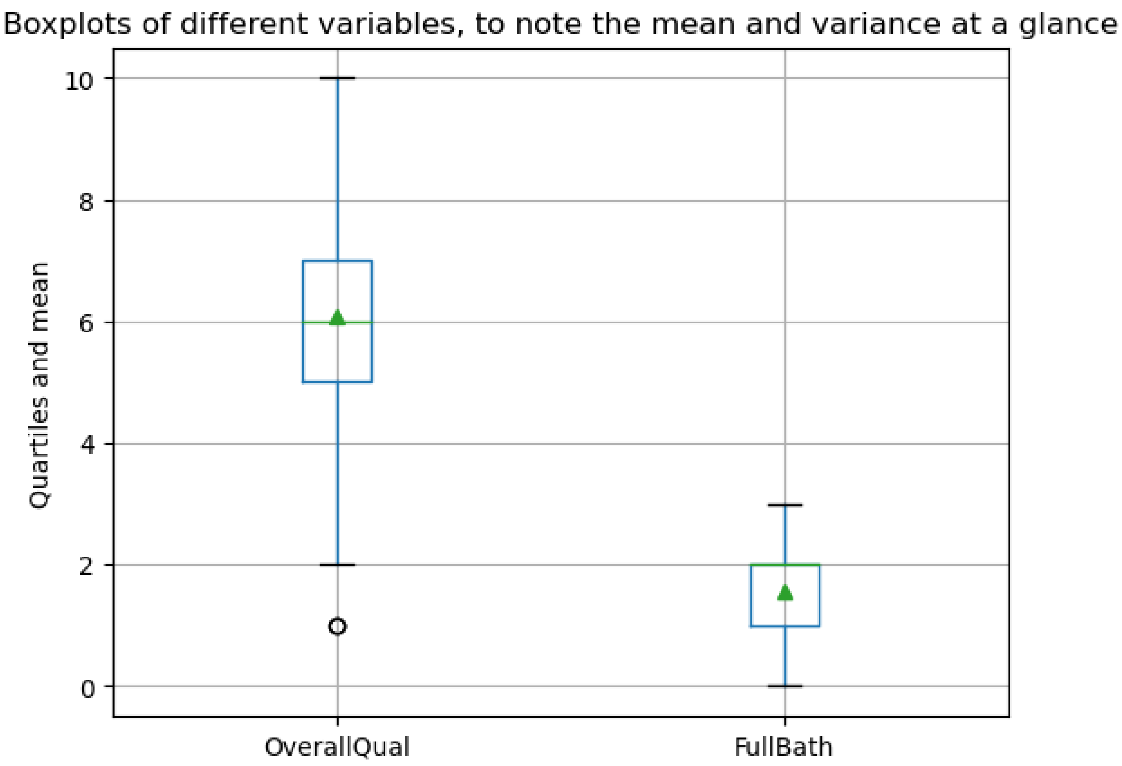


In [199]:

```
axd = data4.boxplot(column = ['OverallQual','FullBath'], showmeans = True)
plt.title("Boxplots of different variables, to note the mean and variance at a glanc
plt.ylabel("Quartiles and mean")
```

Out[199]:

```
Text(0, 0.5, 'Quartiles and mean')
```



In [ ]:

```
 
```