Analysis of the Factors Affecting Sales Price of House in King County,USA

MBA652A – Statistical Modelling for Business Analytics Project Report

Submitted by:- Group:7

SOURADIP PATRA(19114014)

SHIVAM SHARMA(19114012)

ABHISHEK RAI(19114002)

ABHINAV PATERIA(19114001)

Guided By:

Prof. Devlina Chatterjee

Contents

Acknowledgement:

We are highly indebted to Prof. Devlina Chatterjee, for her guidance and continuous support in completing this project. It is because of the knowledge and skills acquired during the course work, along with her comprehensive style of teaching, that we are able to understand the subject in a better way and are able to complete this modelling project successfully.

Declaration:

This is to certify that the project report entitled ‘Analysis of the Factors Affecting Sales Price of House in King County’,USA is based on our original research work. Our indebtedness to other works, studies and publication’s have been duly acknowledge at the relevant places.

Souradip Patra

(IME MTech)

Abhishek Rai

(IME MTech)

Abhinav Pateria

(IME MTech)

Shivam Sharma

(IME MTech)

Introduction:

In this dataset we have to predict the sales price of houses in King County, USA. It includes homes sold between May 2014 and May 2015.
The dataset contains **19** house features including the price, along with **21613** observations.

Objective:

The objective of the project is to study the various factors affecting the **Sales Price of House** using various Regression Techniques and to formulate models depicting the effects of these factors. We will try to evaluate all the possible combination of variables that explains the variation in price of House and try to conclude the best possible combination.

Methodology:

- Summary of data and data visualization.
- Start building model using uni-variate Linear Regression model, and then using multi-variate Linear. Regression model.
- Then on the basis of threshold p-values and VIF(Variance Inflation Factor) we have done backward elimination and have eliminated insignificant variables.

Economic Theory:

One heuristic dataset commonly used for regression analysis of housing prices is the king County, USA housing dataset. Former analyses have found that the price of houses in that dataset are most strongly dependent on their size and the geographical location. Until recently, basic algorithms such as linear regression can predict prices using both intrinsic features of the real estate properties (living area, bedrooms, etc.) and additional features like view, water front.

Data Source:

https://www.kaggle.com/harlfoxem/housesalesprediction/download

The dataset we have chosen is historical dataset of houses at King County, USA from year 2014-2015.

Variables:

1) Dependent variable(Y):

- **Price :** It is a continuous variable whose change in value is to be analyzed using regressors.

2) Independent variables(X):

Following factors affect the prices of house:

1. id :- It is the unique numeric number assigned to each house being sold.
2. date :- It is the date on which the house was sold out..
3. bedrooms :- It determines number of bedrooms in a house.
4. bathrooms :- It determines number of bathrooms in a house.
5. sqft_living :- It is the measurement variable which determines the measurement of house in square foot.
6. sqft_lot : It is also the measurement variable which determines square foot of the lot.

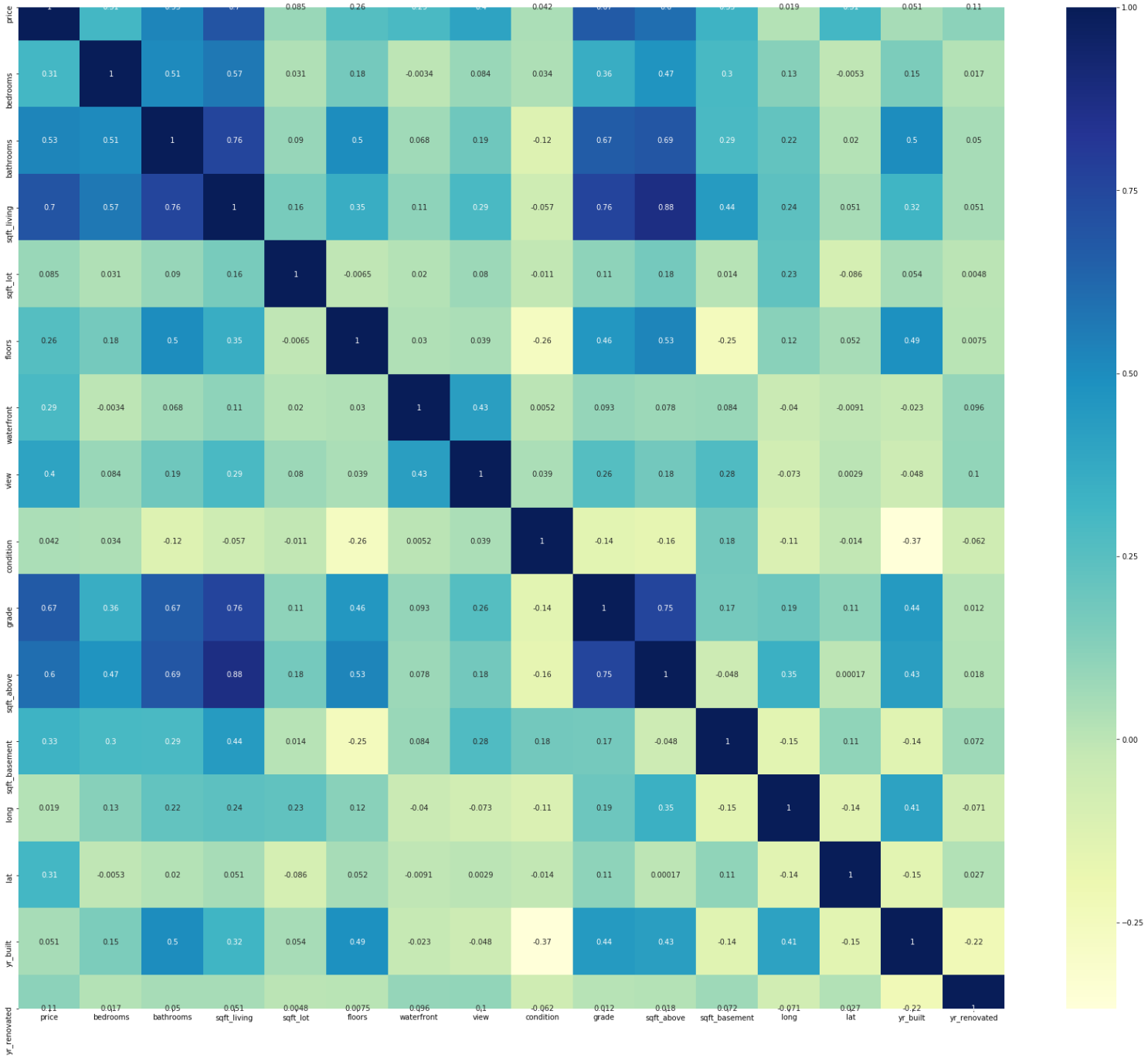
7. floors: It determines total floors means levels of house.
8. waterfront : This feature determines whether a house has a view to waterfront 0 means no 1 means yes.
9. view : This feature determines how many times a house has been viewed.
10. condition : It determines the overall condition of a house on a scale of 1 to 5.
11. grade : It determines the overall grade given to the housing unit, based on King County grading system on a scale of 1 to 13.
12. sqft_above : It determines square footage of house apart from basement.
13. sqft_basement : It determines square footage of the basement of the house.
14. yr_built : It determines the year of building of the house.
15. yr_renovated : It determines year of renovation of house.
16. zipcode : It determines the zipcode of the location of the house.
17. lat : It determines the latitude of the location of the house.
18. long : It determines the longitude of the location of the house.

3) Omitted variables:

- Number of convenient stores near houses
- Type of House(It can capture details like facilities in house, type of foundation, Build Quality of house etc.)
- Air Quality Index
- School districts
- Crime Rate
- Transportation Facility

Exploratory Data Analysis:

- **Correlation plot between variables:**



From this plot this can be concluded that there is no perfect multicollinearity but there is imperfect multi collinearity between sqft_living and sqft_above,grade and sqft_above , bathrooms and sqft_living,& grade and sqft_living .

If there is very high correlation between two features, keeping both of them is not a good idea most of the time not to cause overfitting. For instance, if there is overfitting, we may remove sqt_above or sqt_living because they are highly correlated. This relation can be estimated when we look at the definitions in the dataset but to be sure correlation matrix should be checked. For example: bathrooms and sqrt_living. They are highly correlated but we do not think that the relation among them is the same as the relation between sqt_living and sqt_above.

-All 3 attributes longitude, latitude and zipcode gives insight about the same thing i.e. the effect of location on prices. So instead using all 3 variables in regression it would be better to try only one or maximum two of them in the final equation.

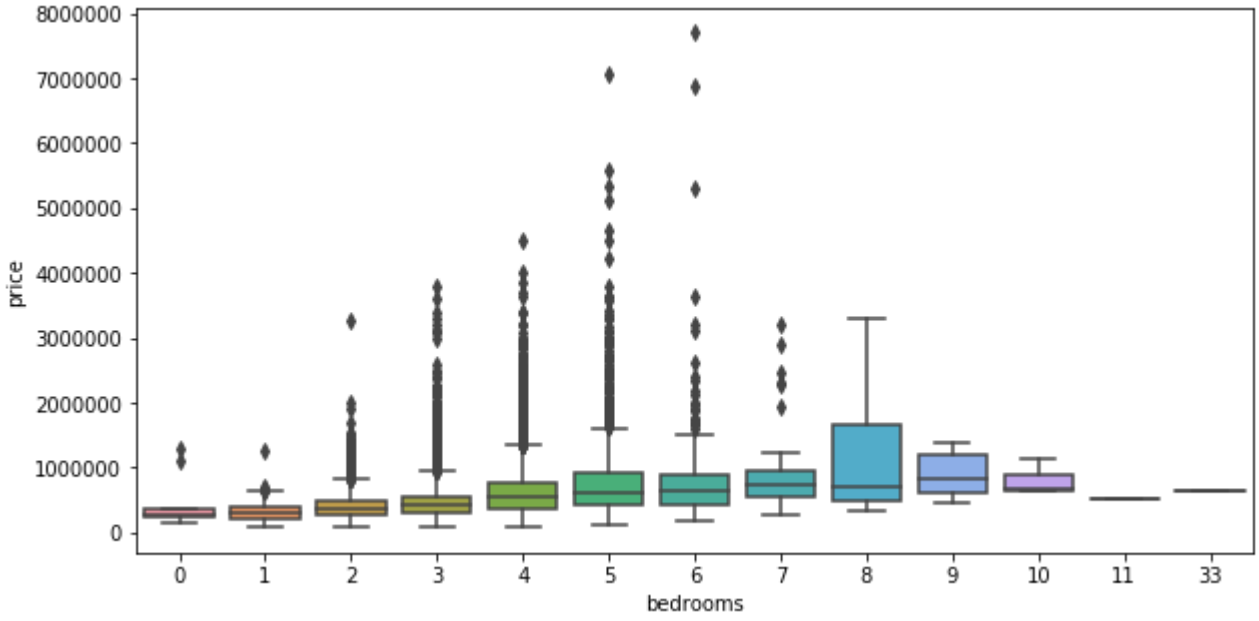
That's why we dropped "zipcode" attribute from our dataset.

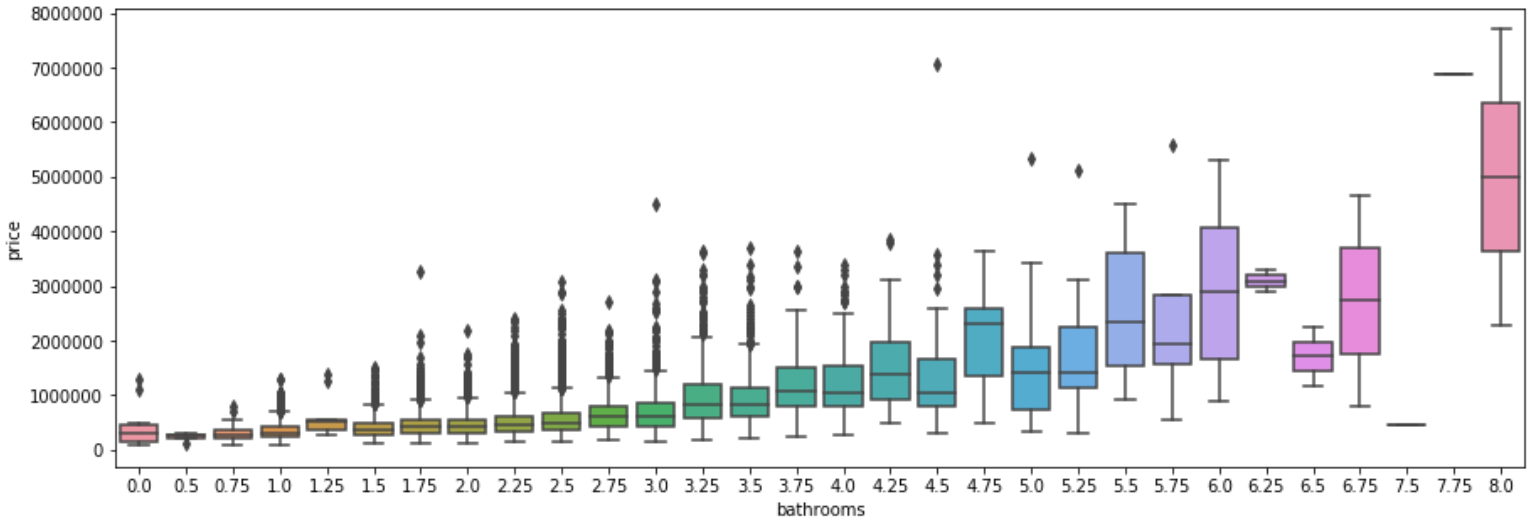
Data Visualization:

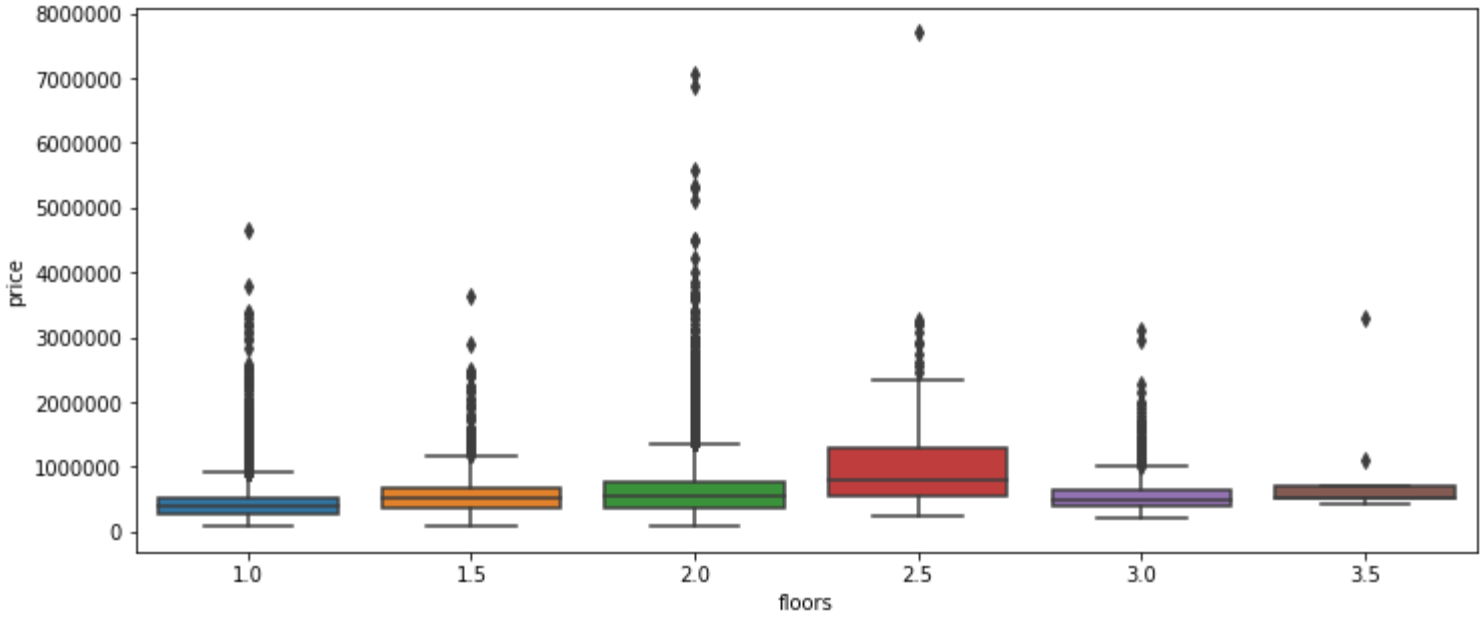
Boxplots:

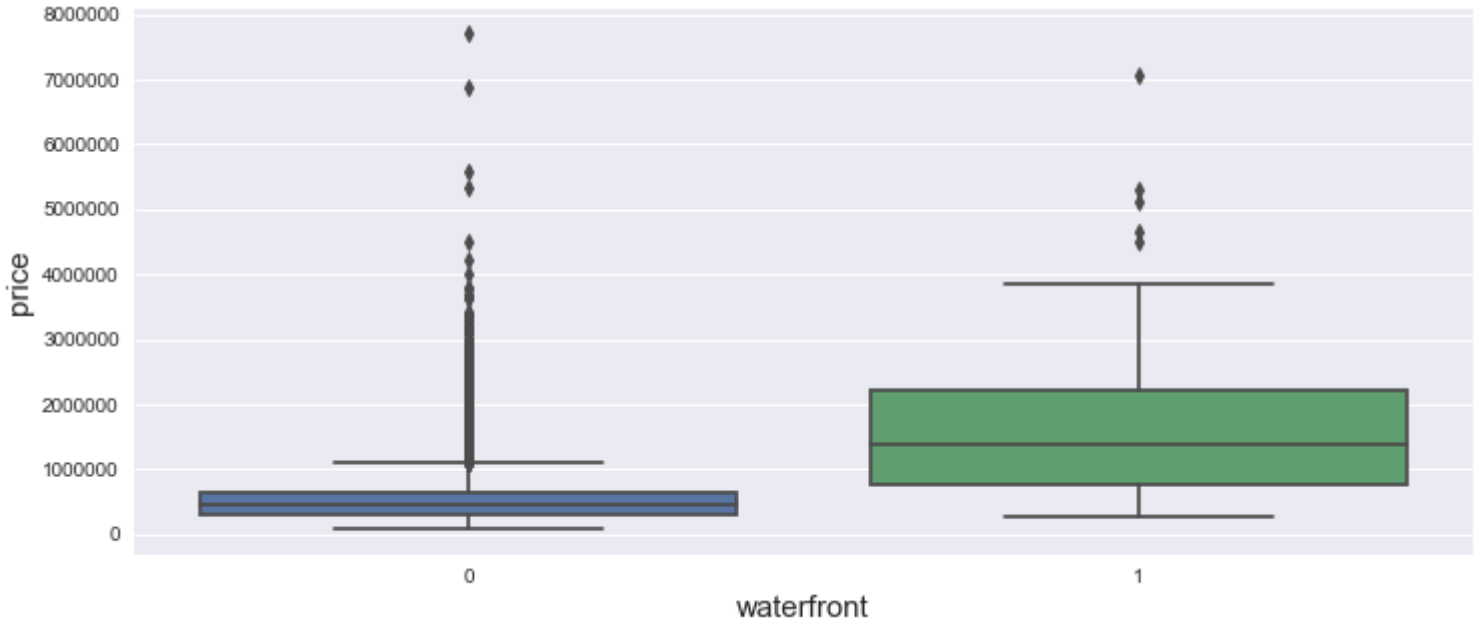
To determine bedrooms, floors or bathrooms vs price, I preferred **boxplot** because we have numerical data but they are not continuous as 1,2,... bedrooms, 2.5, 3,... floors (probably 0.5 stands for the penthouse).

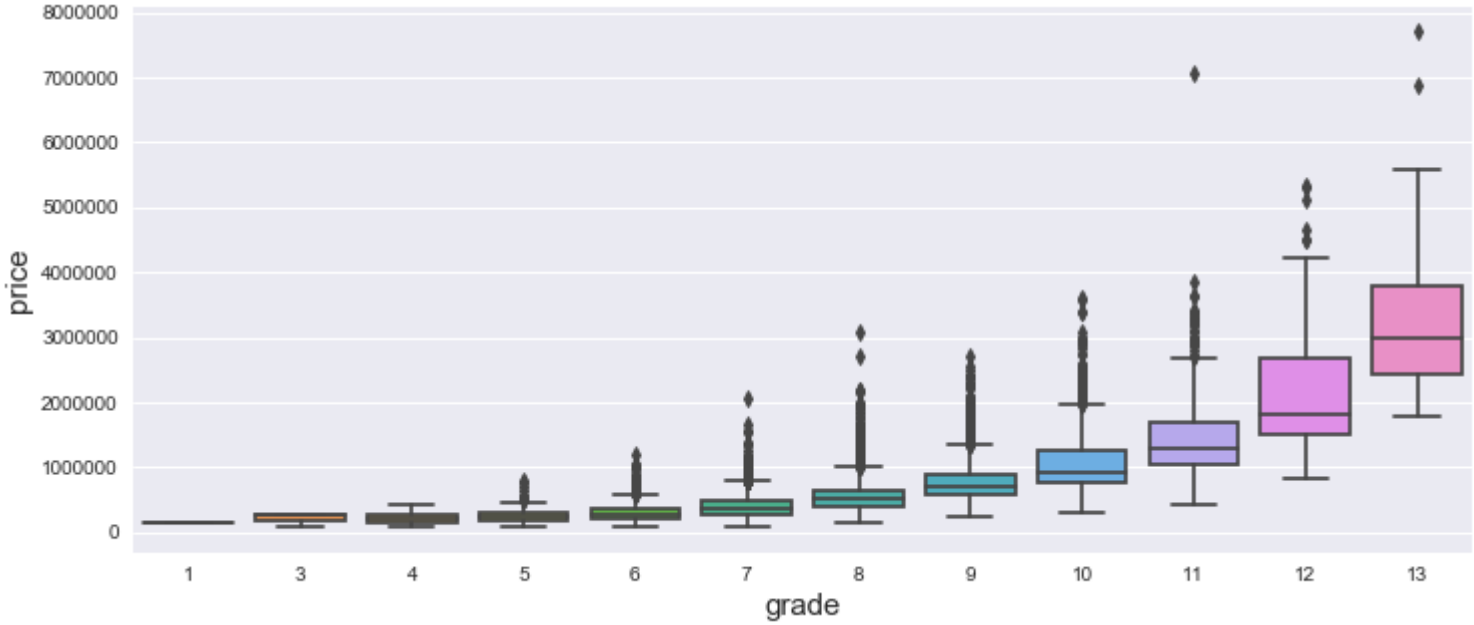
From the below charts, it can be seen that there are very few houses which have some features or price appears far from others like 33 bedrooms or price around 7000000. However, determining their possible negative effect will be time consuming and in the real data sets there will always be some outliers like some luxury house prices in this dataset. That's why we are not planning to remove outliers.




price
8000000
7000000
6000000
5000000
4000000
3000000
2000000
1000000
0
0.0 0.5 0.75 1.0 1.25 1.5 1.75 2.0 2.25 2.5 2.75 3.0 3.25 3.5 3.75 4.0 4.25 4.5 4.75 5.0 5.25 5.5 5.75 6.0 6.25 6.5 6.75 7.5 7.75 8.0
bathrooms

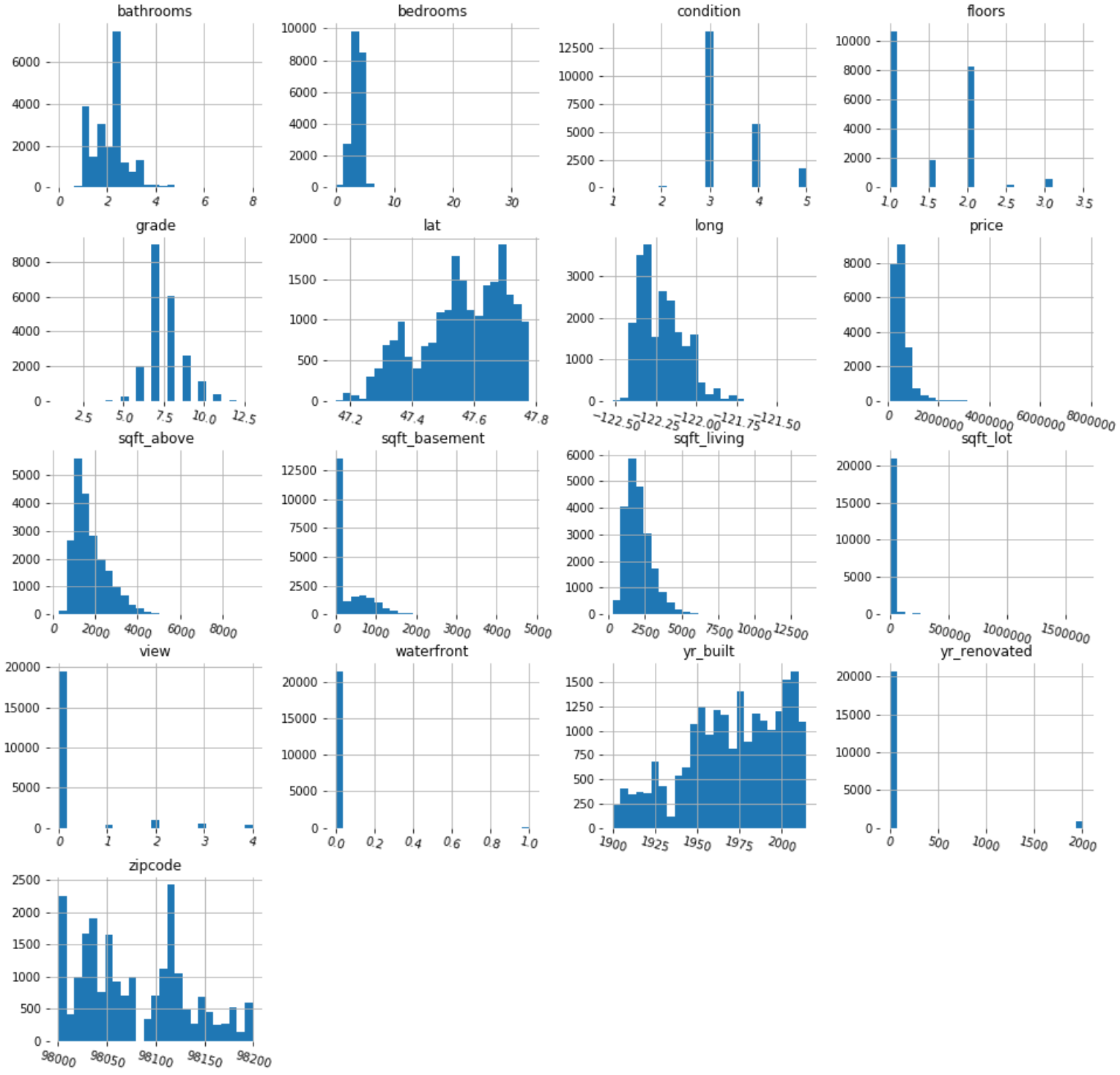

price
8000000
7000000
6000000
5000000
4000000
3000000
2000000
1000000
0
1.0 1.5 2.0 2.5 3.0 3.5
floors


price
8000000
7000000
6000000
5000000
4000000
3000000
2000000
1000000
0
0 1
waterfront

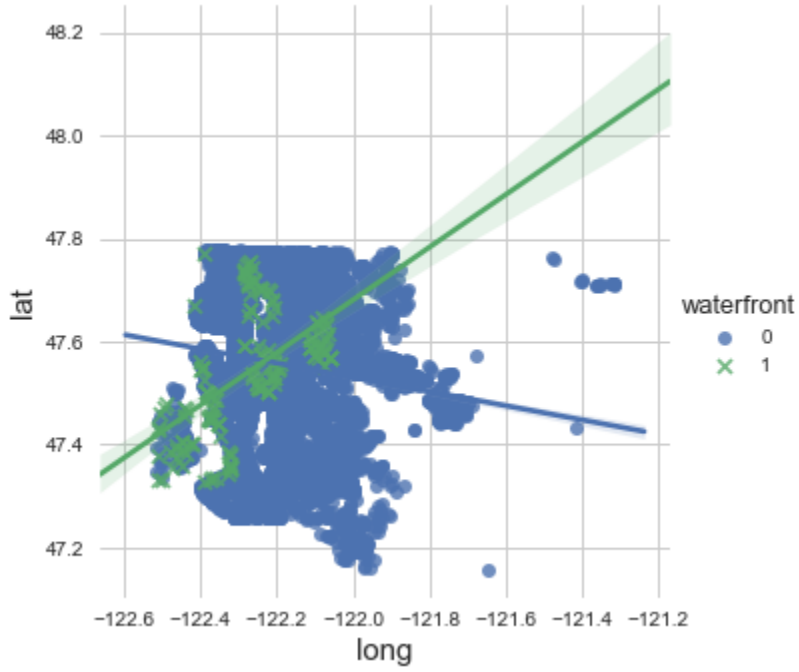


Histograms:

This is not a very big data and we do not have too many features. Thus, we have chance to plot most of them and reach some useful analytical results. Drawing charts and examining the data before applying a model is a very good practice because we may detect some possible outliers or decide to do normalization. This is not a must but get know the data is always good. Then, we started with the histograms of dataframe.

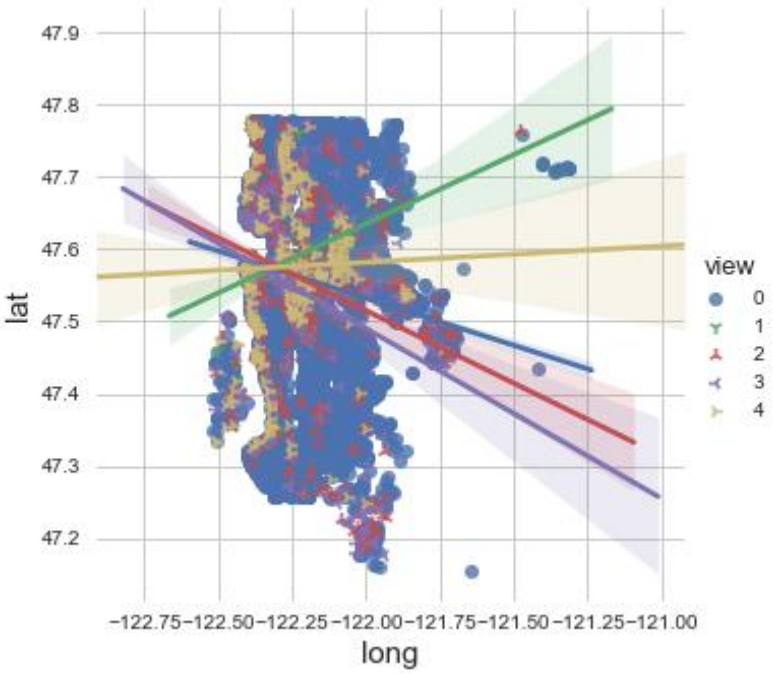


- **Scatterplot of latitude vs longtitude decision variable being waterfront.**

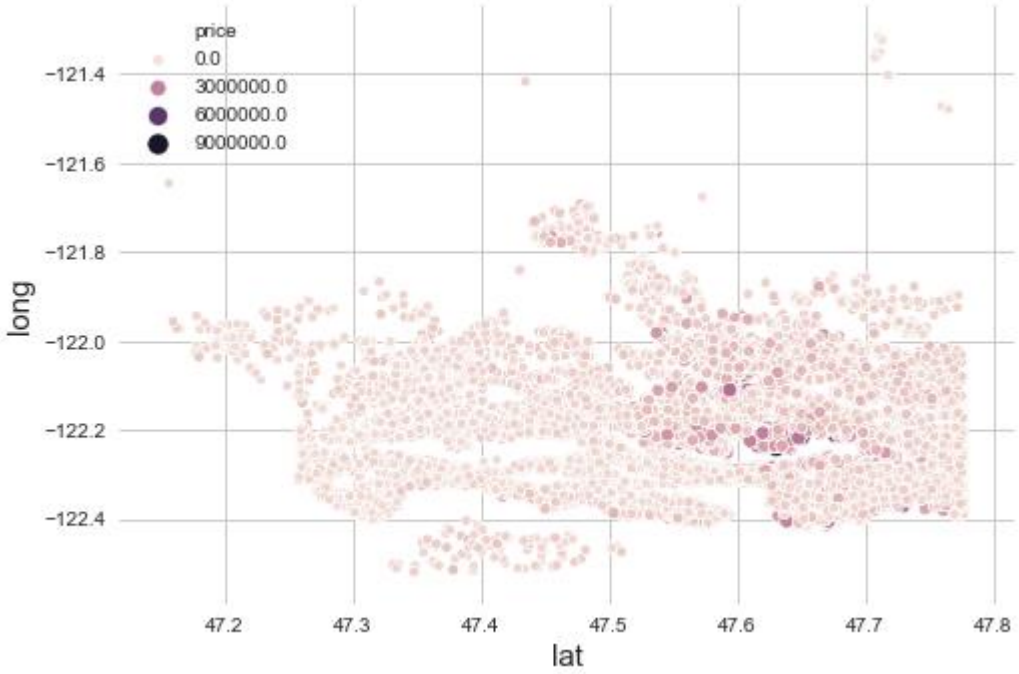


From this plot, it can be seen that blue regions indicate houses doesn't have any waterfront and rest green part has waterfront .

- **Scatterplot of latitude vs longtitude decision variable being view.**



- **Scatter plot of Latitude with Longitude with decision variable being House Prices.**

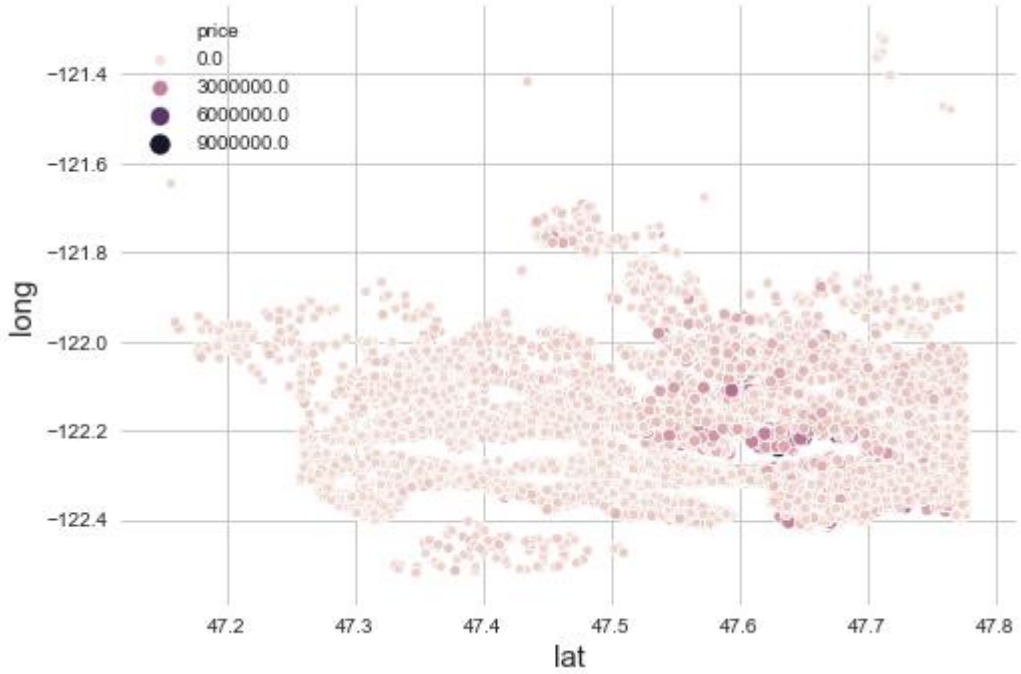


X-axis=Latitude

Y-axis=Longitude

Decision Variable=House Price

- **Scatter plot of longitude vs latitude decision variable being grade:**



x- axis=latitude

Y-axis=Longitude

Decision Variable= Grade

The darker region is the region where the grades of the houses are quite higher as compared to other zones.

Splitting the Dataset into train and test:

- The training dataset and test dataset must be similar, usually have the same predictors or variables.
- They differ on the observations and specific values in the variables. If we fit the model on the training dataset, then we implicitly minimize error or find correct responses. The fitted model provides a good prediction on the training dataset. Then we test the model on the test dataset.
- So, by splitting dataset into training and testing subset, we can efficiently measure our trained model.
- Since it never sees testing data before. Thus it's possible to prevent overfitting.
- We are just splitting dataset into 30% of test data and remaining 70% will used for training the model.

Linear Models:

Linear regression is a method in which we fit regression line between a dependent variable and one or more independent variables. We formulated 12 different models to understand their relationship with house sales price.

When we examine the **correlation matrix**, we may observe that **price** has the highest correlation coefficient with **living area (sqft_living)** (0.7) We will use living area (sqft_living) as feature while creating regression When we model a linear relationship between a response and just one explanatory variable, this is called simple linear regression.

According to Correlation Matrix ,we find out the ranking of features which have the most important relationship on House Price:

1. sqrt_living
2. grade
3. sqft_above
4. bathrooms
5. view
6. sqft_basement
7. lat/bedrooms
8. condition

Following is the list of models formulated in this study:

(Simple Linear Regression Models)

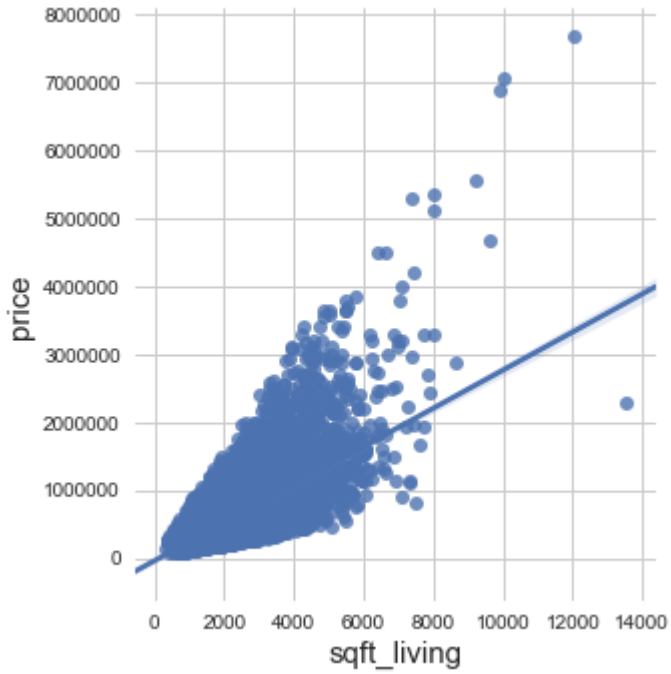
Model 1:

Price = -4.387*10^4+280.8067*sqft_living

R-squared: 0.493, p-value=0.000

Here beta value is 280.8067. The numeric value of beta is large and we get a p-value close to zero and beta is statistically significant. Thus, we rejected the null hypothesis that sales price has no effect on sqft_living as the p-value is very low.

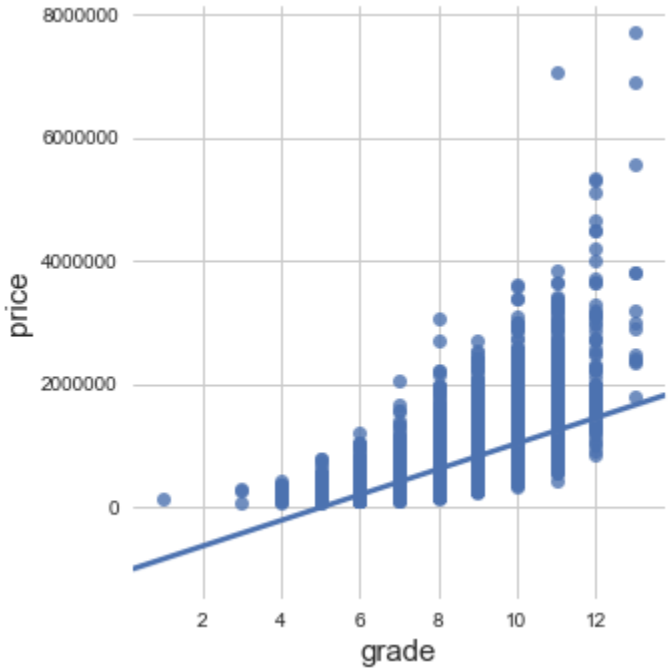
The R-Square value of this model is 0.493. It means that we can explain 49.3% of the variations in Sales price by the variable sqft_living. Our R-square value is high as it is the most correlated parameter with price a/c to correlation matrix.



Model 2:

Sales=-1.507*10^6+2.086*10^5*grade

R-squared: 0.446, p-value=0.000



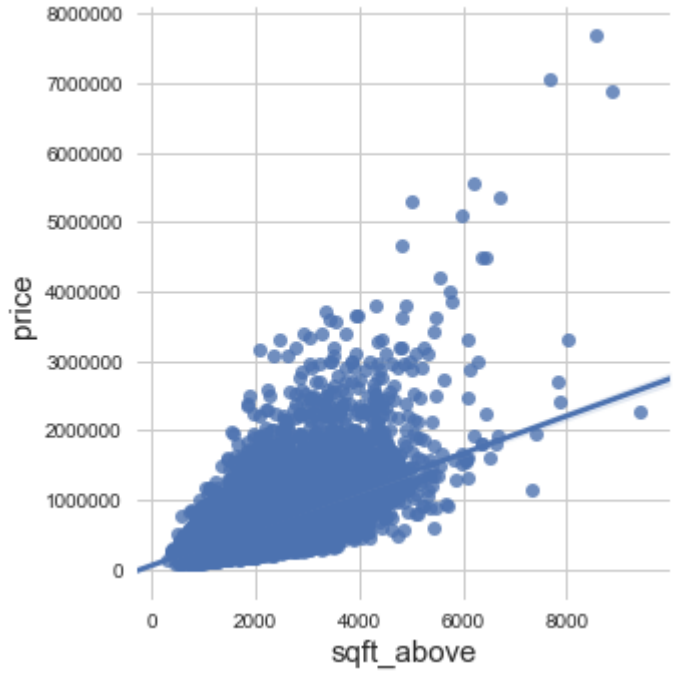
The R- Square value of this model is 0.446. It means that we can explain 44.6% of the variations in Sales price by the variable grade. Our R-square value is reduced than previous .

Here beta value is 2.086*10^5. The numeric value of beta is large and we get a p-value close to zero and beta is statistically significant. Thus, we rejected the null hypothesis that sales price has no effect on grade at 1% level of significance as the p-value is very low.

Model 3:

Sales= 5.974*10^4+ 268.443*sqft-above

R-squared: 0.367, p-value=0.000

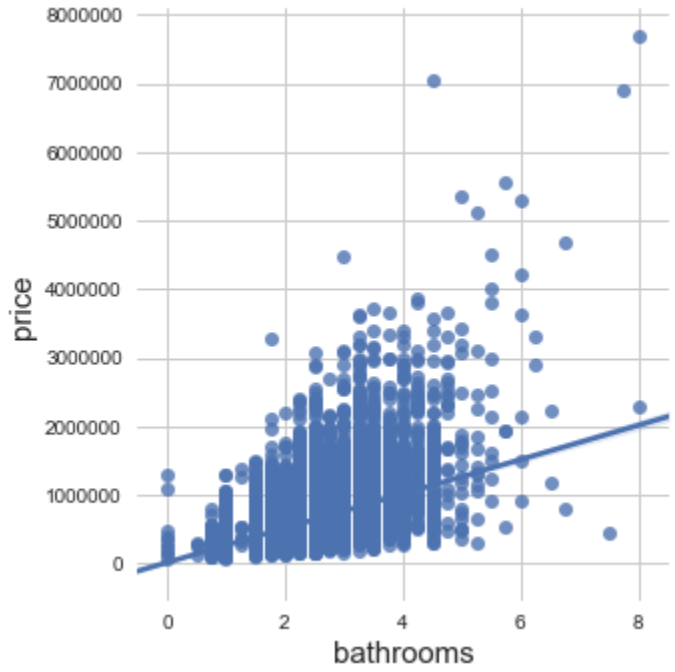


The R-Square value of this model is 0. 367. It means that we can explain 36.7% of the variations in Sales price by the variable sqft_above. Our R-square value is low such that there seems possibility of omitted variable bias.

Model 4:

Sales=1.047*10^4+2.505*10^5*bathrooms

R-squared: 0.276, p-value=0.000.

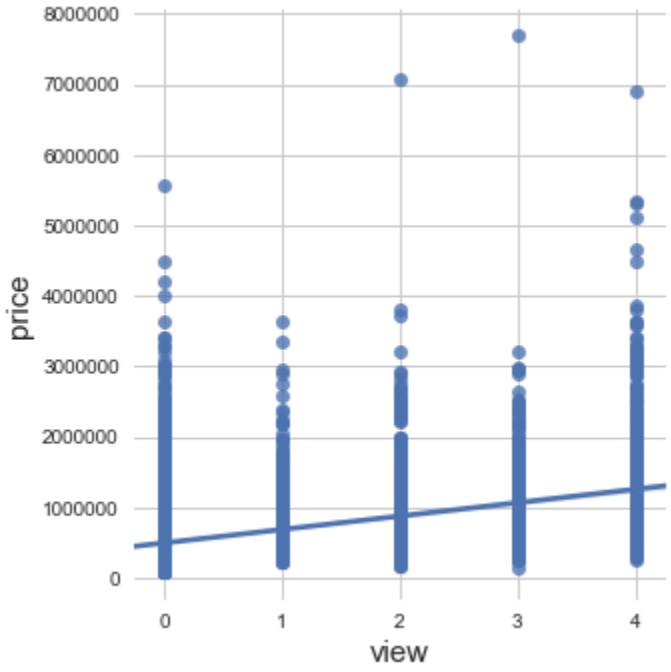


Here the numerical value of beta is 2.505*10^5. The numeric value of beta is quite large, so the standard error is significantly small. Hence we get a p-value close to zero and beta is statistically significant. Thus, we rejected the null hypothesis that sales price has no effect on bathrooms as the p-value is very low.

Model 5:

Sales=4.956*10^5+1.905*10^5*view

R-squared: 0.158, p-value=0.000.

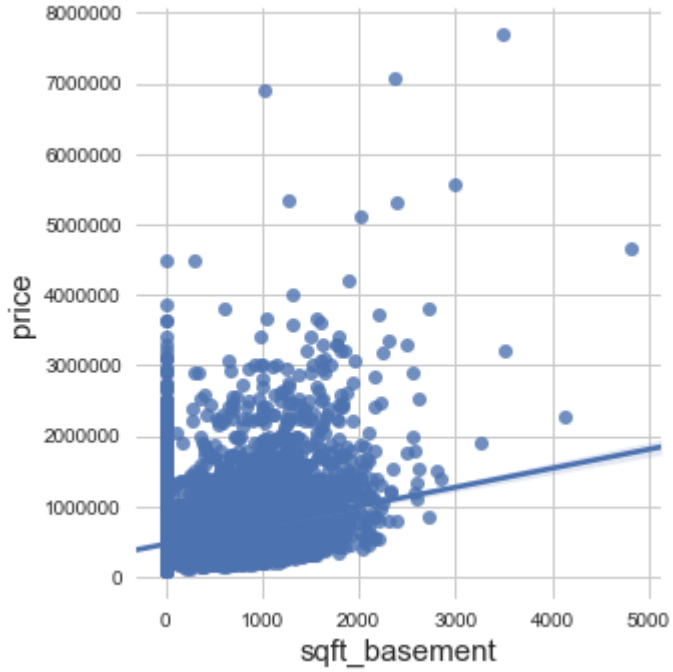


Here the numerical value of beta is 1.905*10^5. The numeric value of beta is quite large, so the standard error is significantly small. Hence we get a p-value close to zero and beta is statistically significant. Thus, we rejected the null hypothesis that sales price has no effect on bathrooms as the p-value is very low.

Model 6:

Sales=4.616*10^5+266.8033*sqft_basement

R-squared: 0.105 , p-value=0.000.



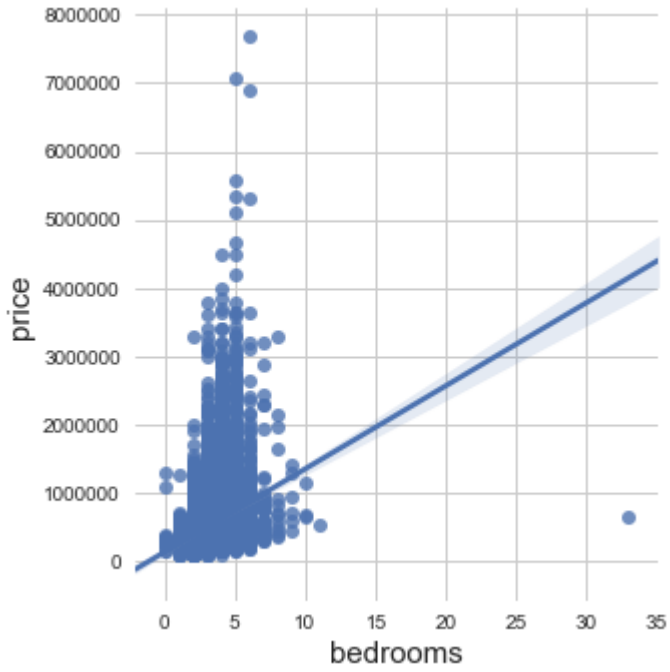
-Our R-square value is much low such that there seems possibility of omitted variable bias.

Model 7:

Sales=1.296*10^5+1.216810^5*bedrooms

R-squared: 0.095 , p-value=0.000.

- Our R- square value is still too much low such that there seems possibility of omitted variable bias.



Model 8:

Sales=4.701*10^5+2.054*10^4*condition

R-squared: 0.001, p-value=0.000.

Even if the numeric value of beta is large, We reject the null hypothesis that sales price has no effect on latitude as the p-value is very less. The R-Square value of this model is 0.01. It means that we can explain 0.01% of the variations in Sales price by the variable latitude. Our R-square value is still too much low such that there seems possibility of omitted variable
bias.

Observations:

- Model 1 is the best fit as it has the highest adjusted R square value of 0.493. Both the intercept and estimates for this model is statistically significant. The standard error of regression is also low for this model. This model explains nearly 49.3 % of the variance in house price which is still quite low.

- Model 7 and model 8 has very low R squared value. So they cannot explains much variance in house price.

Table for Linear Regression:

		Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8
intercept	Estimate	-43870***	-1507000***	59740***	10470***	495600***	461600***	129600***	470100***
Sqft_living	Estimate	280.80***							
grade	Estimate		208600***					-	
Sqft_above	Estimate			268.443***					
bathrooms	Estimate				250500***				
view	Estimate					190500***			
Sqft_basement	Estimate						266.80**		
bedrooms	Estimate							121681***	
condition	Estimate								20540***

Multiple Regressors:

Model 9:

	coef	std err	t	P>|t|	[0.025	0.975]
const	-0.0848	0.002	-39.549	0.000	-0.089	-0.081
bedrooms	0.1409	0.010	-14.643	0.000	-0.160	-0.122
bathrooms	0.0501	0.004	12.232	0.000	0.042	0.058
sqft_living	0.1101	0.002	46.438	0.000	0.105	0.115
sqft_lot	-0.0083	0.008	-0.993	0.321	-0.025	0.008
floors	-0.0002	0.001	-0.122	0.903	-0.003	0.003
waterfront	0.0840	0.003	32.137	0.000	0.079	0.089
view	0.0242	0.001	18.153	0.000	0.022	0.027
condition	0.0167	0.001	11.348	0.000	0.014	0.020
grade	0.1667	0.004	43.491	0.000	0.159	0.174
sqft_above	0.1394	0.003	43.192	0.000	0.133	0.146
sqft_basement	0.0454	0.002	18.538	0.000	0.041	0.050
long	-0.0143	0.002	-6.560	0.000	-0.019	-0.010
lat	0.0454	0.001	45.129	0.000	0.043	0.047
yr_built	-0.0393	0.001	-30.481	0.000	-0.042	-0.037
yr_renovated	0.0024	0.001	2.081	0.037	0.000	0.005

Multiple R-squared: 0.699, Adjusted R-squared:0.698, F-statistic:2504

All the variables have p-value of less than 0.01 except for “floors” which shows “floors” is not statistically significant variable. The p-value of floors is 0.903 so we will not **reject** the null hypothesis that all the beta values are zeros. So,we are removing “floors” attribute in our next model.

Model 10:

	coef	std err	t	P>|t|	[0.025	0.975]
const	-0.0848	0.002	-39.746	0.000	-0.089	-0.081
bedrooms	0.1409	0.010	-14.646	0.000	-0.160	-0.122
bathrooms	0.0500	0.004	12.675	0.000	0.042	0.058
sqft_living	0.1100	0.002	46.454	0.000	0.105	0.115
sqft_lot	-0.0082	0.008	-0.987	0.323	-0.025	0.008
waterfront	0.0840	0.003	32.138	0.000	0.079	0.089
view	0.0242	0.001	18.155	0.000	0.022	0.027
condition	0.0167	0.001	11.378	0.000	0.014	0.020
grade	0.1667	0.004	43.555	0.000	0.159	0.174
sqft_above	0.1393	0.003	43.788	0.000	0.133	0.146
sqft_basement	0.0455	0.002	20.293	0.000	0.041	0.050
long	-0.0142	0.002	-6.639	0.000	-0.018	-0.010
lat	0.0453	0.001	45.371	0.000	0.043	0.047
yr_built	-0.0393	0.001	-31.167	0.000	-0.042	-0.037
yr_renovated	0.0024	0.001	2.077	0.038	0.000	0.005

Multiple R-squared:0.699 , Adjusted R-squared:0.699 , F-statistic:2697

Model 10 is constructed by removing “floors” variable which was a statistically non significant variable. Here F-statistic is 2697 so we so we will not **reject** the null hypothesis that all the beta values are zeros. All remaining variables are statistically significant with very low p-values.Here “sqft_lot” is not statistically significant variable. The p-value of sqft_lot is 0.323 . So we are removing “sqft_lot” attribute in our next model.

Model 11:

	coef	std err	t	P>|t|	[0.025	0.975]
const	-0.0849	0.002	-39.827	0.000	-0.089	-0.081
bedrooms	0.1402	0.010	-14.613	0.000	-0.159	-0.121
bathrooms	0.0500	0.004	12.693	0.000	0.042	0.058
sqft_living	0.1098	0.002	46.633	0.000	0.105	0.114
waterfront	0.0841	0.003	32.160	0.000	0.079	0.089
view	0.0241	0.001	18.128	0.000	0.022	0.027
condition	0.0167	0.001	11.386	0.000	0.014	0.020
grade	0.1667	0.004	43.573	0.000	0.159	0.174
sqft_above	0.1389	0.003	43.943	0.000	0.133	0.145
sqft_basement	0.0454	0.002	20.271	0.000	0.041	0.050
long	-0.0146	0.002	-6.957	0.000	-0.019	-0.010
lat	0.0454	0.001	45.570	0.000	0.043	0.047
yr_built	-0.0392	0.001	-31.166	0.000	-0.042	-0.037
yr_renovated	0.0024	0.001	2.086	0.037	0.000	0.005

Multiple R-squared: 0.699, Adjusted R-squared: 0.699,F-statistic:29

Model 11 is constructed by removing “sqft_lot” variable which was a statistically non significant variable.

Here “yr_renovated” is not statistically significant variable. The p-value of yr_renovated is 0.037 so we will not **reject** the null hypothesis that all the beta values are zeros. So,we are removing “yr_renovated” attribute in our next model.

Model 12:

	coef	std err	t	P>|t|	[0.025	0.975]
const	-0.0843	0.002	-39.901	0.000	-0.088	-0.080
bedrooms	0.1408	0.010	-14.679	0.000	-0.160	-0.122
bathrooms	0.0513	0.004	13.172	0.000	0.044	0.059
sqft_living	0.1097	0.002	46.593	0.000	0.105	0.114
waterfront	0.0844	0.003	32.340	0.000	0.079	0.090
view	0.0242	0.001	18.170	0.000	0.022	0.027
condition	0.0162	0.001	11.192	0.000	0.013	0.019
grade	0.1669	0.004	43.634	0.000	0.159	0.174
sqft_above	0.1388	0.003	43.908	0.000	0.133	0.145
sqft_basement	0.0453	0.002	20.231	0.000	0.041	0.050
long	-0.0145	0.002	-6.893	0.000	-0.019	-0.010
lat	0.0453	0.001	45.517	0.000	0.043	0.047
yr_built	-0.0400	0.001	-33.441	0.000	-0.042	-0.038

Multiple R-squared: 0.699, Adjusted R-squared: 0 .698,F-statistic:3186

Model 12 is constructed by removing yr_renovated variable. Here the p-value of all the variables are less than 0.01 at 99% confidence interval. But three variables have very high VIF .So, to remove multicollinearity we remove sqft_living attribute.

Model 13:

	coef	std err	t	P>|t|	[0.025	0.975]
const	-0.0843	0.002	-39.901	0.000	-0.088	-0.080
bedrooms	0.1408	0.010	-14.679	0.000	-0.160	-0.122
bathrooms	0.0513	0.004	13.172	0.000	0.044	0.059
waterfront	0.0844	0.003	32.340	0.000	0.079	0.090
view	0.0242	0.001	18.170	0.000	0.022	0.027
condition	0.0162	0.001	11.192	0.000	0.013	0.019
grade	0.1669	0.004	43.634	0.000	0.159	0.174
sqft_above	0.2143	0.005	45.271	0.000	0.205	0.224
sqft_basement	0.0795	0.003	30.382	0.000	0.074	0.085
long	-0.0145	0.002	-6.893	0.000	-0.019	-0.010
lat	0.0453	0.001	45.517	0.000	0.043	0.047
yr_built	-0.0400	0.001	-33.441	0.000	-0.042	-0.038

Multiple R-squared: 0.699, Adjusted R-squared: 0.698,F-statistic:3186

Model 13 is obtained by dropping sqft_living attribute. Here the p-value of all the variables are in acceptable range .In our next model,we are going to remove sqft_basement as it has less significance obtained from single regression model and also Adjusted Rsquared is decreasing only by 0.01.

Model 14:

	coef	std err	t	P>|t|	[0.025	0.975]
const	-0.1025	0.002	-49.111	0.000	-0.107	-0.098
bedrooms	0.0598	0.009	-6.302	0.000	-0.078	-0.041
bathrooms	0.0952	0.004	25.572	0.000	0.088	0.103
waterfront	0.0836	0.003	31.090	0.000	0.078	0.089
view	0.0322	0.001	23.975	0.000	0.030	0.035
condition	0.0201	0.001	13.581	0.000	0.017	0.023
grade	0.1957	0.004	51.254	0.000	0.188	0.203
sqft_above	0.1496	0.004	34.353	0.000	0.141	0.158
long	-0.0140	0.002	-6.452	0.000	-0.018	-0.010
lat	0.0463	0.001	45.148	0.000	0.044	0.048
yr_built	-0.0481	0.001	-40.048	0.000	-0.050	-0.046

Multiple R-squared: 0.68, Adjusted R-squared: 0.68, F-statistic:3217

From residual plot it can be concluded that the data is not heteroskedastic as there is not too much variation of residue with changes in fitted values. But zero line is still not perfectly flat.

This model is obtained by dropping sqft_basement variable.
In our next model,we are going to remove condition variable as it has less significance obtained from single regression model and also Adjusted Rsquared is decreasing only by 0.01.

Model 15:

	coef	std err	t	P>|t|	[0.025	0.975]
const	-0.0890	0.002	-48.229	0.000	-0.093	-0.085
bedrooms	0.0488	0.010	-5.134	0.000	-0.067	-0.030
bathrooms	0.0985	0.004	26.337	0.000	0.091	0.106
waterfront	0.0833	0.003	30.807	0.000	0.078	0.089
view	0.0324	0.001	23.981	0.000	0.030	0.035
grade	0.1978	0.004	51.542	0.000	0.190	0.205
sqft_above	0.1438	0.004	32.977	0.000	0.135	0.152
long	-0.0121	0.002	-5.576	0.000	-0.016	-0.008
lat	0.0452	0.001	43.936	0.000	0.043	0.047
yr_built	-0.0536	0.001	-47.044	0.000	-0.056	-0.051

Multiple R-squared: 0.676, Adjusted R-squared: 0.676, F-statistic:3511

In this model we applied those function to variables which best describes each individual model from model1 to model14 .
All the beta's value is significant with very low p-value .If we remove any other attribute from our model the adjusted R-squared is reduced significantly. We get the best R squared value.

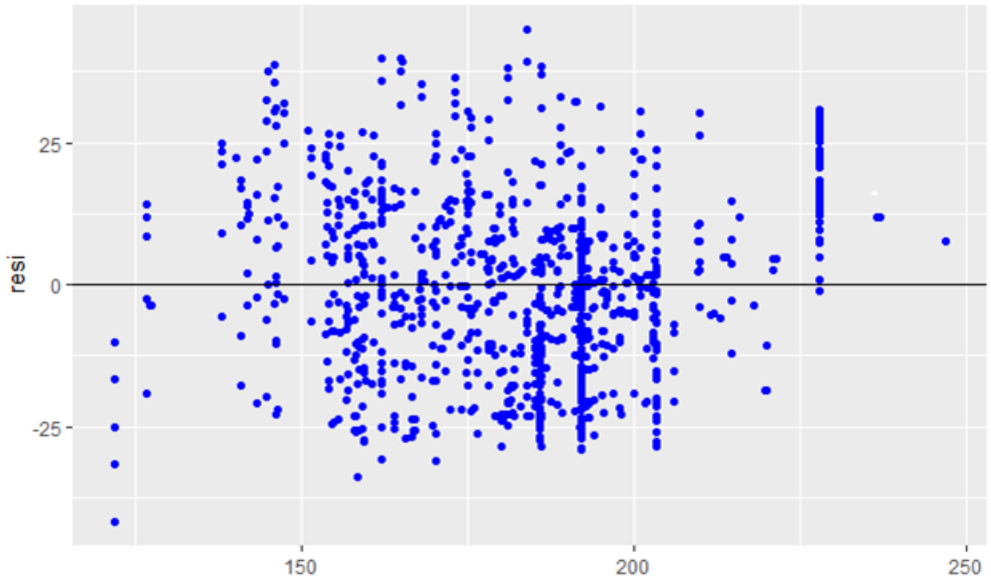
So, this is our best model we have obtained so far.

Best Model :

Sales=-0.0890+0.0488*bedrooms+0.0985*bathrooms+0.0833*waterfront+0.0324*view+0.1978*grade+
0.1438*sqft_above-0.0121*longitude+0.0452*latitude-0.0536*yr_built

Heteroskedasticity Test:

One of the important assumptions of linear regression is that, there should be no heteroskedasticity of residuals. In simpler terms, this means that the variance of residuals should not increase with fitted values of response variable.



Residual Plot

From residual plot it can be concluded that the data is homoskedastic as there is not much variation of residue with changes in fitted values.

Table for Multiple Regression:

	MODEL 9			MODEL 10			MODEL 11		
Variable	Beta	P-Value	VIF	Beta	P-Value	VIF	Beta	P-Value	VIF
Constant	-.0848	.000	99.73	-0.0848	0.00	98.69	-.0849	0.000	98.50
bedrooms	.1409	.000		0.1409	0.000	1.62	.1402	.000	1.61
Bathrooms	0.0501	.000	3.36	0.0500	0.000	3.11	.05	.000	3.11
Sqft_livings	0.1101	.000	inf	0.1100	0.000	inf	.1098	.000	Inf
Sqft_lot	-0.0083	.321		-0.0082	0.323	1.09		0.000	
Floors	-.0002	.903						0.000	
Waterfront	0.0840	.000		0.0840	0.000		.0849	0.000	1.23
View	0.0242	.000		0.0242	0.000	1.43	.0241	0.000	1.42
Condition	0.0167	.000		0.0167	0.000	1.24	.0167	0.000	1.24
Grade	0.1667	.000	3.10	0.1667	0.000	3.09	.1667	0.000	3.09
Sqft_above	0.1394	.000	inf	0.1393	0.000	inf	.1389	0.000	Inf
Sqft_basement	0.0454	.000		0.0455	0.000	inf	.0454	0.000	Inf
long	-0.0143	.000		-0.0142	0.000	1.38	-.0146	0.000	1.33
lat	0.0454	.000		0.0453	0.000	1.11	.0454	0.000	
Yr_built	-0.0393	.000		-0.0393	0.000	2.27	-.0392	0.000	2.26
Yr_renovated	-0.0393	.037		0.0024	0.038	1.14	.0024	.037	1.14
R-Square	0.699			.699			.699		
Adj R-Square	0.698			.699			.699		

	MODEL12			MODEL13			MODEL14			MODEL 15		
Variable	Beta	P-Value	VIF	Beta	P-Value	VIF	Beta	P-Value	VIF	Beta	P-Value	VIF
Constant	-.0843	.000	96.75	-.0843	0.000	96.75	-.1025	0.000	88.98	-.08	0.000	68.65
bedrooms	.1408	0.000	1.61	.1408	0.000	1.61	.0598	0.000	1.48	.0488	0.000	1.47
Bathrooms	.0513	0.000	3.03	.0513	0.000	3.03	.0952	0.000	2.62	.0985	0.000	2.61
Sqft_livings	.1097	0.000	Inf		.							
Sqft_lot												
Floors												
Waterfront	.0844	0.000	1.23	.0844	0.000	1.23	.0836	0.000	1.23	.0833	0.000	1.23
View	.0242	0.000	1.42	.0242	0.000	1.42	.0322	0.000	1.37	.0324	0.000	1.37
Condition	.0162	**0.000**	1.20	.0162	0.000	1.2	.0201	0.000	1.19			
Grade	.1669	0.000	3.09	.1669	0.000	3.09	.1957	0.000	2.9	.1978	0.000	2.90
Sqft_above	.1388	0.000	Inf	.2143	0.000		0.1438	0.000	3.15	.1438	0.000	3.12
Sqft_basement	.0453	0.000	Inf	.0795	0.000	1.70						
long	-.0145	0.000	1.33	-.0145	0.000	1.33	-.0140	0.000	1.33	-.0121	0.000	1.32
lat	.0453	0.000		.0453	0.000	1.10	.0463	0.000	1.10	.0452	0.000	1.09
Yr_built	-.0400	0.000	2.05	-.040	0.000	2.05	-.0481	0.000	1.95	-.0536	0.000	1.73
Yr_renovated												
R-Square	.699			.699			.68			.676		
Adj R-Square	.698			.698			.68			.676		

Test for Multi Collinearity using Variable Inflation factor:

- Detecting multi-collinearity can be done in different ways. Below are three common ways to examine the phenomenon.

1. Correlation Matrix
2. Variance Inflation Factor (V IF)

   The Variance Inflation Factor (VIF) is a measure of collinearity among predictor variables within a multiple regression. It is calculated by taking the the ratio of the variance of all a given model's betas divide by the variance of a single beta if it were fit alone.

variable	vif
bedrooms	1.47
Bathrooms	2.61
Waterfront	1.23
View	1.37
Grade	2.90
Sqft_above	3.12
long	1.32
lat	1.09
Yr_built	1.73

Here all the attributes have VIF value less than 5,So there is no multicollinearity among independent variables in our final model.

CONCLUSION:

Follwing is our best fitting multiple regressor model:

Price= -0.089+ 0.1978(Grade)+ 0.1438(Sqft_above)+ 0.0958(Bathrooms)

+0.0833(Waterfront) -0.0536(Yr_built)+ 0.0488(Bedrooms)

+0.0452(latitude)- 0.0121(longitude) +0.0324(View)

- Attributes grade and sqft_above have the most effect on our model as they explains our data very well.
- Our model is able to explain 67.6% of the variance, rest explains the role of omitted variables discussed earlier.

Python Codes:

#importing Libraries

```
import numpy as np
import pandas as pd
import math
import seaborn as sns
import matplotlib
import matplotlib.pyplot as plt
from scipy.stats import pearsonr
from sklearn import tree
from sklearn import linear_model
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import explained_variance_score
from time import time
from sklearn.metrics import r2_score
import os
```

#importing Dataset

```
data = pd.read_csv('C:/Users/user/Desktop/kc_house_data.csv')
data.head()
data.describe()
#Check any number of columns with NaN or missing values
print(data.isnull().any().sum(), ' / ', len(data.columns))
# Check any number of data points with NaN
print(data.isnull().any(axis=1).sum(), ' / ', len(data))
sns.heatmap(data.isnull(), yticklabels= False, cbar=False, cmap= 'vlag')
data['view'].value_counts()
from sklearn.preprocessing import MinMaxScaler
from sklearn.feature_selection import RFE
from sklearn.linear_model import LinearRegression
import statsmodels.api as sm
from statsmodels.stats.outliers_influence import variance_inflation_factor


dataset = data[['price','bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot', 'floors', 'waterfront', 'view', 'condition', 'grade', 'sqft_above',
'sqft_basement','long','lat']]
```

dataset.head()

#Data Visualization

sns.pairplot(dataset)

plt.show()

data.plot(kind='scatter', x = 'sqft_living', y = 'price');

plt.show()

data.plot(kind='scatter', y = 'price', x = 'bedrooms');

plt.show()

data1=data[['price','bedrooms','bathrooms','sqft_living','sqft_lot','floors','waterfront','view','condition','grade','sqft_above','sqft_basement','yr_built','yr_renovated','zipcode','lat','long']]

h = data1.hist(bins=25,figsize=(16,16),xlabelsize='10',ylabelsize='10',xrot=-15)

sns.despine(left=True, bottom=True)

[x.title.set_size(12) for x in h.ravel()];

[x.yaxis.tick_left() for x in h.ravel()];h = data1.hist(bins=25,figsize=(16,16),xlabelsize='10',ylabelsize='10',xrot=-15)

sns.despine(left=True, bottom=True)

[x.title.set_size(12) for x in h.ravel()];

[x.yaxis.tick_left() for x in h.ravel()];

sns.countplot(data.bedrooms, order = data['bedrooms'].value_counts().index)

sns.countplot(data.view, order = data['view'].value_counts().index)

sns.countplot(data.waterfront, order = data['waterfront'].value_counts().index)

sns.countplot(data.grade, order = data['grade'].value_counts().index)

f, axe = plt.subplots(1, 1,figsize=(10,5))

sns.boxplot(x=data['bedrooms'],y=data['price'],ax=axe)

f, axe = plt.subplots(1, 1,figsize=(15,5))

sns.boxplot(x=data['bathrooms'],y=data['price'],ax=axe)

f, axe = plt.subplots(1, 1,figsize=(12,5))

sns.boxplot(x=data['floors'],y=data['price'],ax=axe)

from sklearn.model_selection import train_test_split

np.random.seed(0)

df_train, df_test = train_test_split(dataset, train_size = 0.7, test_size = 0.3, random_state = 100)

#Min-Max Normalization

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

num_vars = ['price','bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot', 'floors', 'waterfront', 'view', 'condition', 'grade', 'sqft_above',

'sqft_basement','long','lat']

df_train[num_vars] = scaler.fit_transform(df_train[num_vars])

plt.figure(figsize = (30, 25))

sns.heatmap(df_train.corr(), annot = True, cmap="YlGnBu")

plt.show()

#Simple Linear Regression

x = dataset.iloc[:,[3]].values

y = dataset.iloc[:,[0]].values

x= np.append(arr=np.ones((21613,1)).astype(int),values = x,axis = 1)

regressor_OLS=sm.OLS(endog=y,exog=x).fit()

regressor_OLS.summary()

#Grade

x = dataset.iloc[:,[9]].values

y = dataset.iloc[:,[0]].values

x= np.append(arr=np.ones((21613,1)).astype(int),values = x,axis = 1)

regressor_OLS=sm.OLS(endog=y,exog=x).fit()

regressor_OLS.summary()

#sqft_above

x = dataset.iloc[:,[10]].values

y = dataset.iloc[:,[0]].values

x= np.append(arr=np.ones((21613,1)).astype(int),values = x,axis = 1)

regressor_OLS=sm.OLS(endog=y,exog=x).fit()

regressor_OLS.summary()

#bathrooms

x = dataset.iloc[:,[2]].values

y = dataset.iloc[:,[0]].values

x= np.append(arr=np.ones((21613,1)).astype(int),values = x,axis = 1)

regressor_OLS=sm.OLS(endog=y,exog=x).fit()

regressor_OLS.summary()

#view

x = dataset.iloc[:,[7]].values

y = dataset.iloc[:,[0]].values

x= np.append(arr=np.ones((21613,1)).astype(int),values = x,axis = 1)

regressor_OLS=sm.OLS(endog=y,exog=x).fit()

regressor_OLS.summary()

#sqft_basemwnt

```
x = dataset.iloc[:,[11]].values
y = dataset.iloc[:,[0]].values
x= np.append(arr=np.ones((21613,1)).astype(int),values = x,axis = 1)
regressor_OLS=sm.OLS(endog=y,exog=x).fit()
regressor_OLS.summary()
#bedrooms
x = dataset.iloc[:,[1]].values
y = dataset.iloc[:,[0]].values
x= np.append(arr=np.ones((21613,1)).astype(int),values = x,axis = 1)
regressor_OLS=sm.OLS(endog=y,exog=x).fit()
regressor_OLS.summary()
#Dividing data into X and y variables
y_train = df_train.pop('price')
X_train = df_train
from sklearn.feature_selection import RFE
from sklearn.linear_model import LinearRegression
import statsmodels.api as sm
from statsmodels.stats.outliers_influence import variance_inflation_factor

lm = LinearRegression()
lm.fit(X_train,y_train)
```

#Multiple Linear Regression

```
def build_model(X,y):
    X = sm.add_constant(X) #Adding the constant
    lm = sm.OLS(y,X).fit() # fitting the model
    print(lm.summary()) # model summary
    return X

def checkVIF(X):
    vif = pd.DataFrame()
    vif['Features'] = X.columns
    vif['VIF'] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
    vif['VIF'] = round(vif['VIF'], 2)
    vif = vif.sort_values(by = "VIF", ascending = False)
    return(vif)
X_train_new = build_model(X_train,y_train)
```

```
checkVIF(X_train_new)
X = pd.DataFrame(X_train)
y = pd.DataFrame(y_train)
model = sm.OLS(y, sm.add_constant(X))
model_fit = model.fit()

# create dataframe from X, y for easier plot handling
dataframe = pd.concat([X, y], axis=1)
# model values
model_fitted_y = model_fit.fittedvalues
# model residuals
model_residuals = model_fit.resid
# normalized residuals
model_norm_residuals = model_fit.get_influence().resid_studentized_internal
# absolute squared normalized residuals
model_norm_residuals_abs_sqrt = np.sqrt(np.abs(model_norm_residuals))
# absolute residuals
model_abs_resid = np.abs(model_residuals)
# leverage, from statsmodels internals
model_leverage = model_fit.get_influence().hat_matrix_diag
# cook's distance, from statsmodels internals
model_cooks = model_fit.get_influence().cooks_distance[0]

plot_lm_1 = plt.figure()
plot_lm_1.axes[0] = sns.residplot(model_fitted_y, dataframe.columns[-1], data=dataframe,
                          lowess=True,
                          scatter_kws={'alpha': 0.5},
                          line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})

plot_lm_1.axes[0].set_title('Residuals vs Fitted')
plot_lm_1.axes[0].set_xlabel('Fitted values')
plot_lm_1.axes[0].set_ylabel('Residuals');
X_train_new = X_train.drop(["floors"], axis = 1)
X_train_new = build_model(X_train_new,y_train)
checkVIF(X_train_new)
X_train_new = X_train_new.drop(["sqft_lot"], axis = 1)
```

```
X_train_new = build_model(X_train_new,y_train)
checkVIF(X_train_new)
X_train_new = X_train_new.drop(["yr_renovated"], axis = 1)
X_train_new = build_model(X_train_new,y_train)
checkVIF(X_train_new)
X_train_new = X_train_new.drop(["sqft_living"], axis = 1)
X_train_new = build_model(X_train_new,y_train)
checkVIF(X_train_new)
X_train_new = X_train_new.drop(["sqft_basement"], axis = 1)
X_train_new = build_model(X_train_new,y_train)
checkVIF(X_train_new)
X_train_new = X_train_new.drop(["condition"], axis = 1)
X_train_new = build_model(X_train_new,y_train)
checkVIF(X_train_new)
# model values
model_fitted_y = model_fit.fittedvalues
# model residuals
model_residuals = model_fit.resid
# normalized residuals
model_norm_residuals = model_fit.get_influence().resid_studentized_internal
# absolute squared normalized residuals
model_norm_residuals_abs_sqrt = np.sqrt(np.abs(model_norm_residuals))
# absolute residuals
model_abs_resid = np.abs(model_residuals)
# leverage, from statsmodels internals
model_leverage = model_fit.get_influence().hat_matrix_diag
# cook's distance, from statsmodels internals
model_cooks = model_fit.get_influence().cooks_distance[0]

plot_lm_1 = plt.figure()
plot_lm_1.axes[0] = sns.residplot(model_fitted_y, dataframe.columns[-1], data=dataframe,
                          lowess=True,
                          scatter_kws={'alpha': 0.5},
                          line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})

plot_lm_1.axes[0].set_title('Residuals vs Fitted')
```

```
plot_lm_1.axes[0].set_xlabel('Fitted values')
plot_lm_1.axes[0].set_ylabel('Residuals');
```