Abstract

We did a regression analysis on a Kansas City Housing dataset from Kaggle. The dataset contained 21613 instances and 21 variables. We used price as our outcome of interest. We did our regression analysis using five different models: Multiple Linear, KNN, Lasso, Ridge, and Random Forest. Overall, we found Random Forest to be the most accurate with R-Squared = 0.98 on training data and R-Squared = 0.86 on test data. We performed bootstrap on our Random Forest model and found a median R-Squared of 0.98 over 1000 iterations.

Introduction

For this project, we used a Kaggle dataset on Kansas City Housing (https://www.kaggle.com/datasets/astronautelvis/kc-house-data). We decided to do an application project by doing a regression analysis using the price variable as the outcome of interest. We also utilized a GitHub repository.

GitHub Repository
https://github.com/Vishal613/House-Price-Prediction

Author Contributions

Rohil preprocessed the data for the Linear and K-NN regression models. He also modeled the linear and K-NN regression using sklearns and providing the required parameters for optimum evaluation. Additionally he also designed a webpage that can be locally hosted to check the price prediction along with the evaluation metric of the models using the Streamlit framework in python. For the presentation, he presented his work with all the results and visualizations.

Grace created the Initial Visualization jupyter notebook and added the bootstrap code to the Models notebooks for estimation stability. For the presentation, they did the visualization, correlation, and estimation stability slides. They also wrote the abstract, introduction, and data description sections of the project report, as well as added to the Estimation Stability section.

Vishal created the Github repository, and created and added jupyter notebooks for Lasso, Ridge and Random Forest Regression. This included everything from preprocessing, model building, analyzing predictor contributions, to comparing different models using Cross validation For the presentation, they showed the correlation plot between each pair of independent variables, and the Lasso and Ridge Regression slides. They also wrote the Preprocessing, correlation plot between each pair of independent variables, and about Lasso, Ridge and Random Forest Regression models under model building. They created the Bootstrapping charts for the Random Forest Regression model and added to the report. They also wrote the Conclusion section.

A. Data Description

The dataset we used had 21613 instances and 21 variables. These variables were:

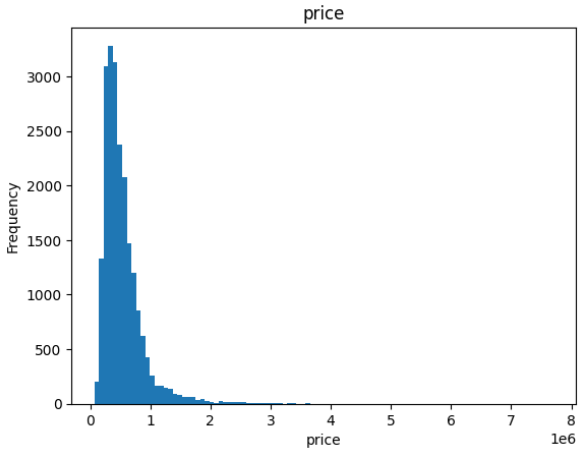
1. id - a unique ID for each home sold
2. date - date of the home sale
3. price - price of each home sold
4. bedrooms - number of bedrooms
5. bathrooms - number of bathrooms, where .5 accounts for a room with a toilet but no shower
6. sqft_living - square footage of the house's interior living space
7. sqft_lot - square footage of the land space
8. floors - number of floors
9. waterfront - a categorical variable signifying whether or not the house was overlooking a waterfront (0 = no, 1 = yes)
10. view - an index of how good the view from the house was (0, 1, 2, 3, or 4 where 0 = bad and 4 = best)
11. condition - an index on the condition of the house (0, 1, 2, 3, 4, 5 where 0 = bad and 5 = best)
12. grade - an index on quality of house construction and design (1-6 = falls short of building construction and design, 7 = an average level of construction and design, 8-13 = a high-quality level of construction and design)
13. sqft_above - square footage of the interior housing space that is above ground level
14. sqft_basement - square footage of the interior housing space that is below ground level
15. yr_built - year the house was initially built
16. yr_renovated - year of the house's last renovation (0 indicates no renovation)
17. zipcode - zipcode area of the house
18. lat - latitude
19. long - longitude
20. sqft_living15 - square footage of interior housing living space for the nearest 15 neighbors
21. sqft_lot15 - square footage of the land lots of the nearest 15 neighbors

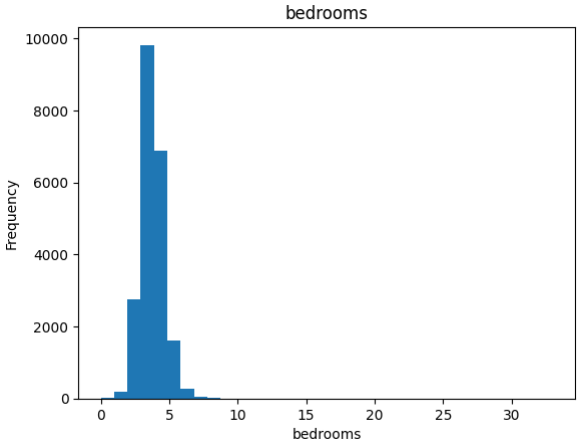
When initially looking at the data, we began by summarizing each variable, as seen in the table here:

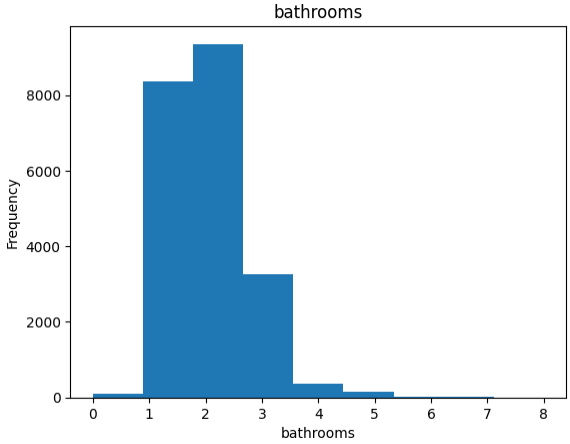
	id	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition
count	2.161300e+04	2.161300e+04	21613.000000	21613.000000	21613.000000	2.161300e+04	21613.000000	21613.000000	21613.000000	21613.000000
mean	4.580302e+09	5.400881e+05	3.370842	2.114757	2079.899736	1.510697e+04	1.494309	0.007542	0.234303	3.409430
std	2.876566e+09	3.671272e+05	0.930062	0.770163	918.440897	4.142051e+04	0.539989	0.086517	0.766318	0.650743
min	1.000102e+06	7.500000e+04	0.000000	0.000000	290.000000	5.200000e+02	1.000000	0.000000	0.000000	1.000000
25%	2.123049e+09	3.219500e+05	3.000000	1.750000	1427.000000	5.040000e+03	1.000000	0.000000	0.000000	3.000000
50%	3.904930e+09	4.500000e+05	3.000000	2.250000	1910.000000	7.618000e+03	1.500000	0.000000	0.000000	3.000000
75%	7.308900e+09	6.450000e+05	4.000000	2.500000	2550.000000	1.068800e+04	2.000000	0.000000	0.000000	4.000000
max	9.900000e+09	7.700000e+06	33.000000	8.000000	13540.000000	1.651359e+06	3.500000	1.000000	4.000000	5.000000

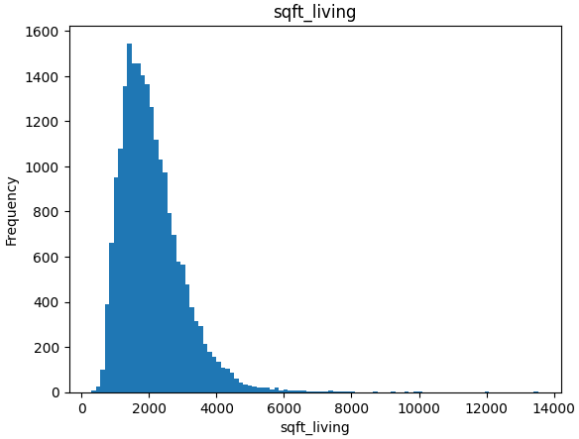
grade	sqft_above	sqft_basement	yr_built	yr_renovated	zipcode	lat	long	sqft_living15	sqft_lot15
21613.000000	21613.000000	21613.000000	21613.000000	21613.000000	21613.000000	21613.000000	21613.000000	21613.000000	21613.000000
7.656873	1788.390691	291.509045	1971.005136	84.402258	98077.939805	47.560053	-122.213896	1986.552492	12768.455652
1.175459	828.090978	442.575043	29.373411	401.679240	53.505026	0.138564	0.140828	685.391304	27304.179631
1.000000	290.000000	0.000000	1900.000000	0.000000	98001.000000	47.155900	-122.519000	399.000000	651.000000
7.000000	1190.000000	0.000000	1951.000000	0.000000	98033.000000	47.471000	-122.328000	1490.000000	5100.000000
7.000000	1560.000000	0.000000	1975.000000	0.000000	98065.000000	47.571800	-122.230000	1840.000000	7620.000000
8.000000	2210.000000	560.000000	1997.000000	0.000000	98118.000000	47.678000	-122.125000	2360.000000	10083.000000
13.000000	9410.000000	4820.000000	2015.000000	2015.000000	98199.000000	47.777600	-121.315000	6210.000000	871200.000000

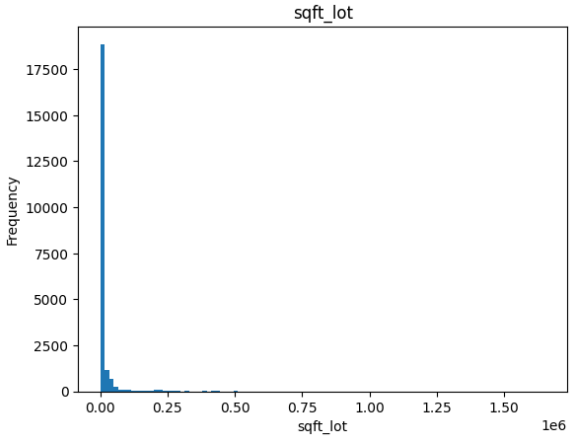
We then visualized the data for each variable in histograms:

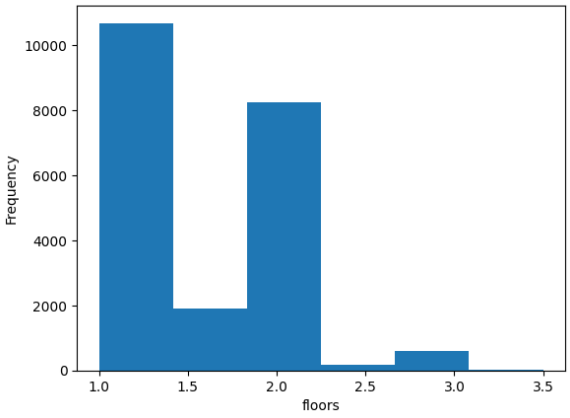

price
Frequency
price
1e6

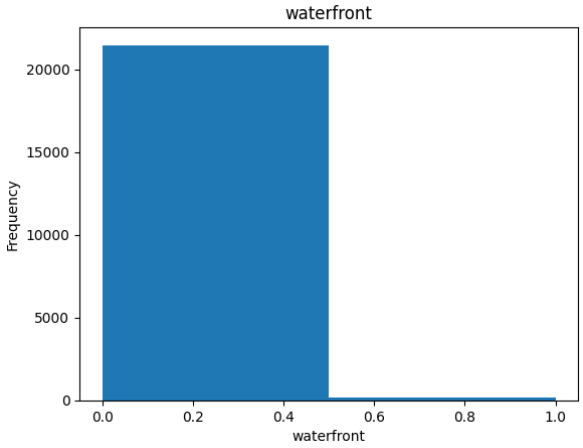

bedrooms
Frequency
bedrooms

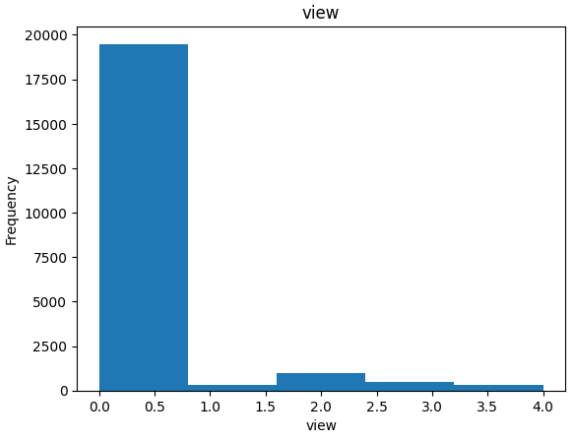

bathrooms
Frequency
bathrooms

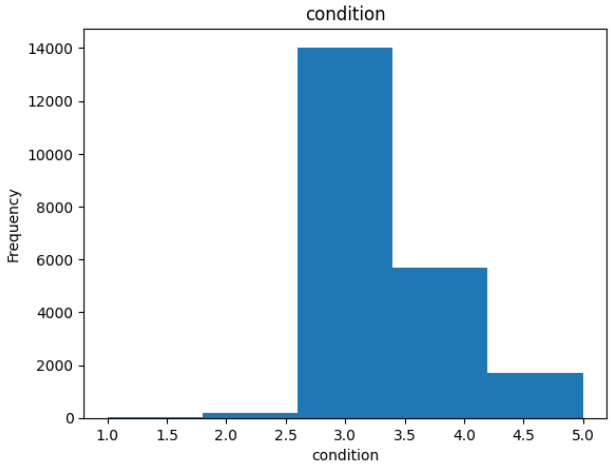

sqft_living
Frequency
sqft_living

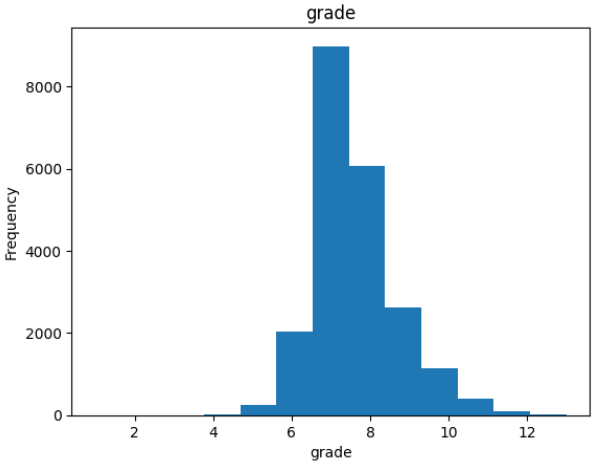

sqft_lot
Frequency
sqft_lot
1e6

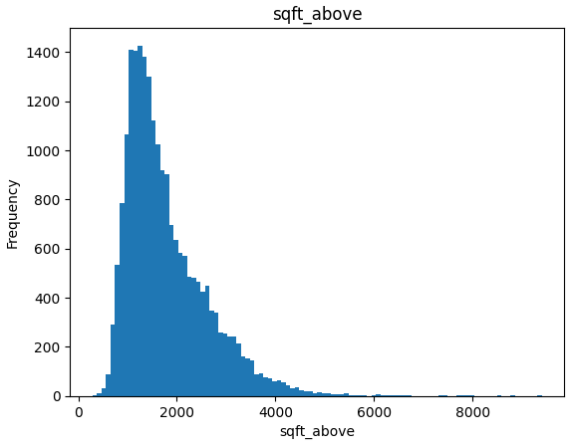

floors
Frequency
floors

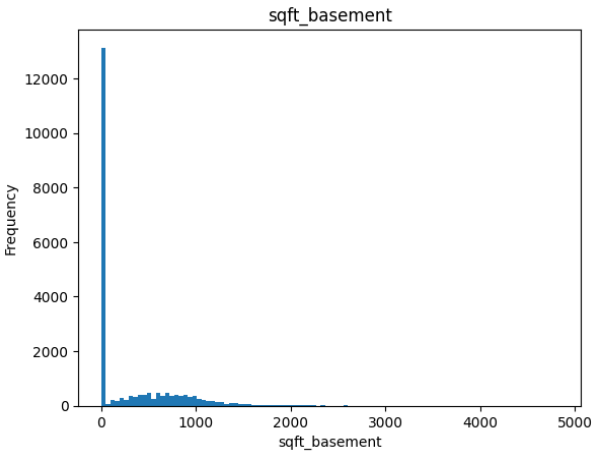

waterfront
Frequency
waterfront

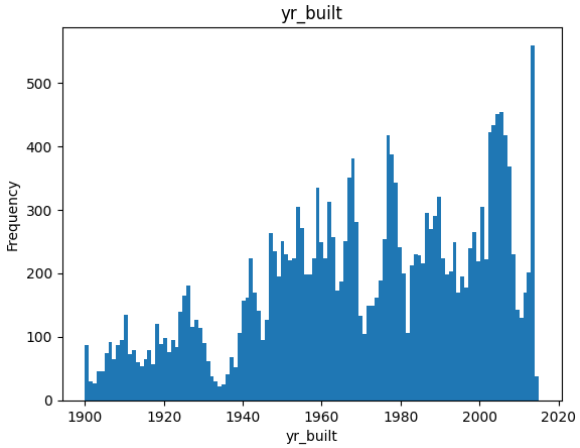

view
Frequency
view

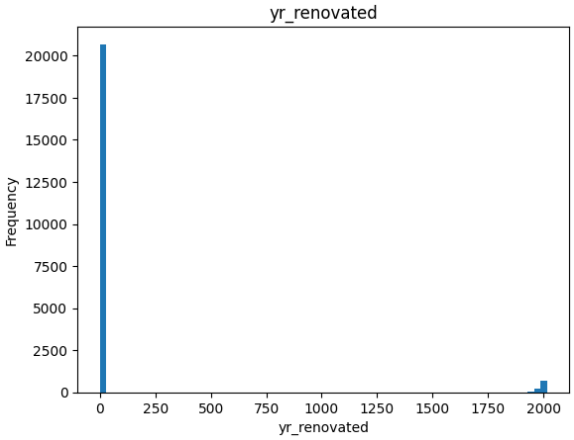

condition
Frequency
condition

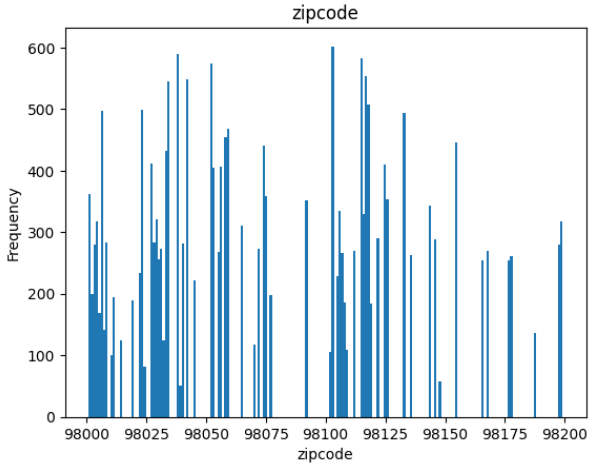

grade
Frequency
grade

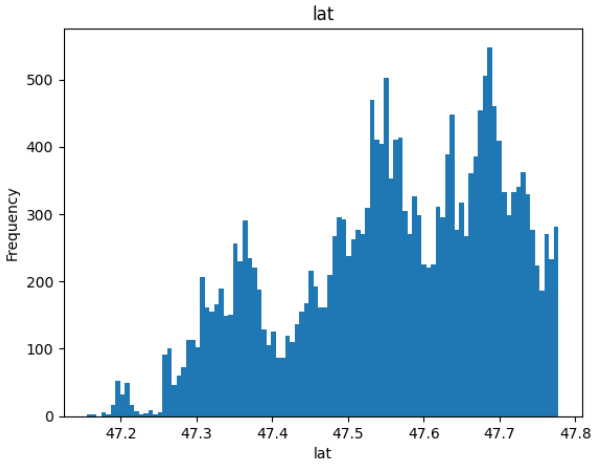

sqft_above
Frequency
sqft_above

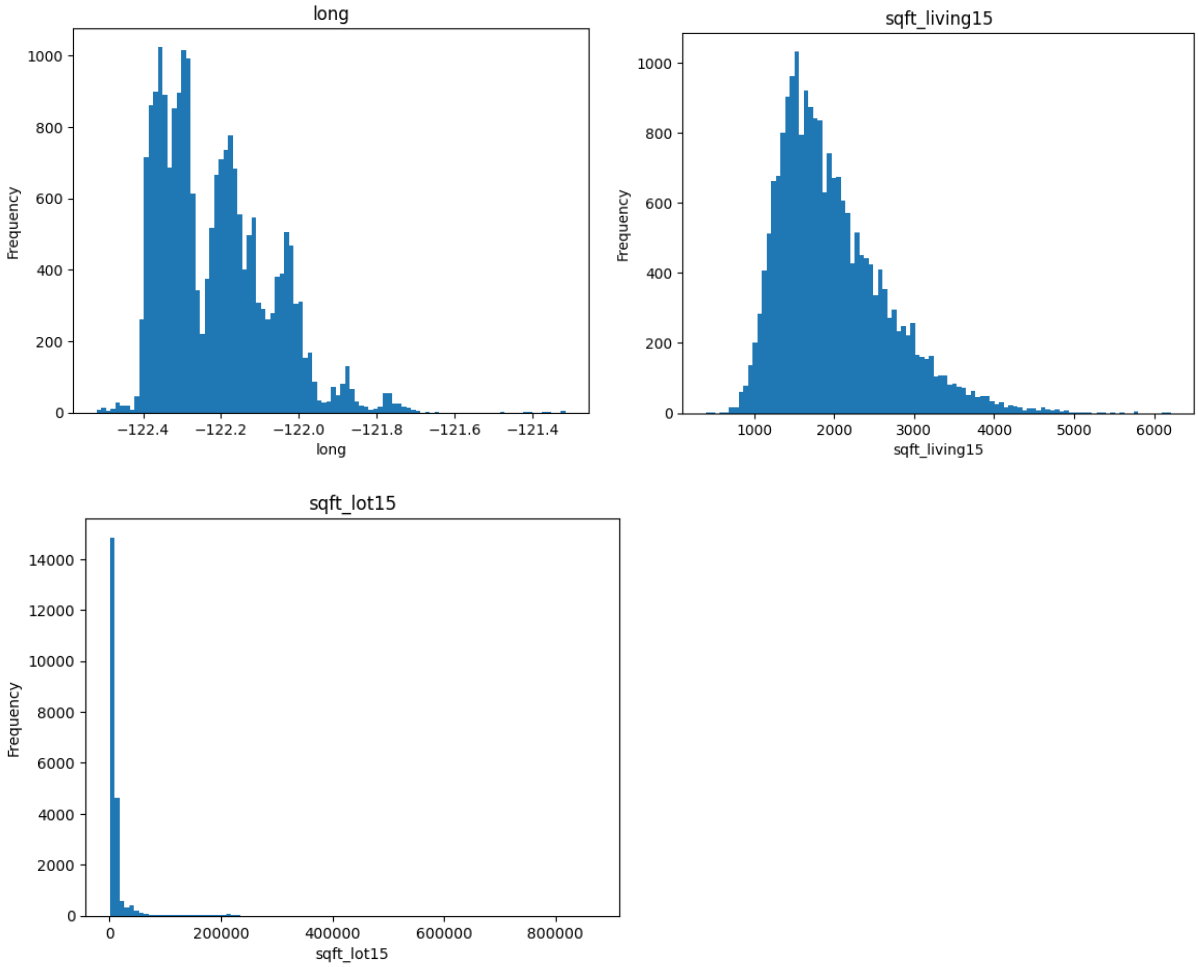

sqft_basement
Frequency
sqft_basement


yr_built
Frequency
0
100
200
300
400
500
1900
1920
1940
1960
1980
2000
2020
yr_built


yr_renovated
Frequency
0
2500
5000
7500
10000
12500
15000
17500
20000
0
250
500
750
1000
1250
1500
1750
2000
yr_renovated

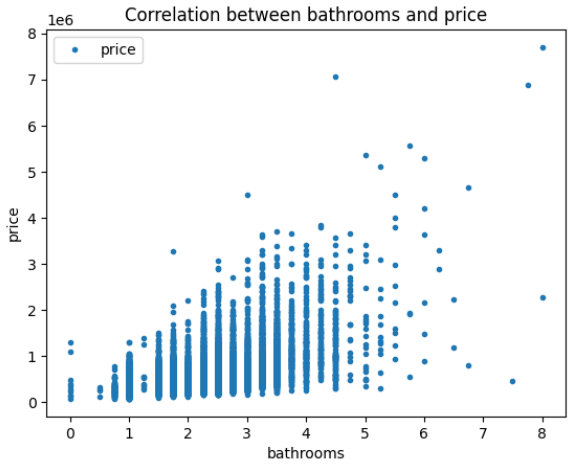

zipcode
Frequency
0
100
200
300
400
500
600
98000
98025
98050
98075
98100
98125
98150
98175
98200
zipcode

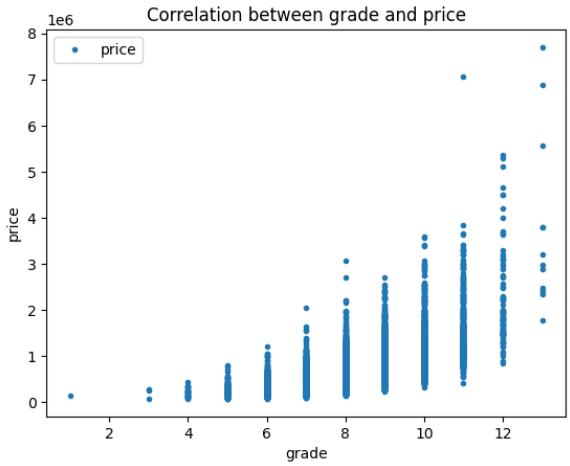

lat
Frequency
0
100
200
300
400
500
47.2
47.3
47.4
47.5
47.6
47.7
47.8
lat

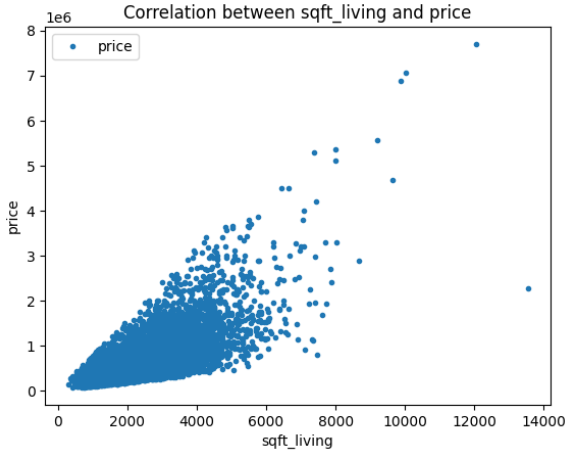


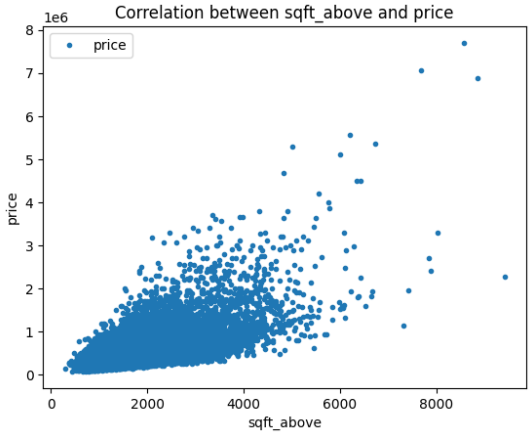
These visualizations are a sometimes clearer way of understanding the data from the table. We can see the spread for each variable.

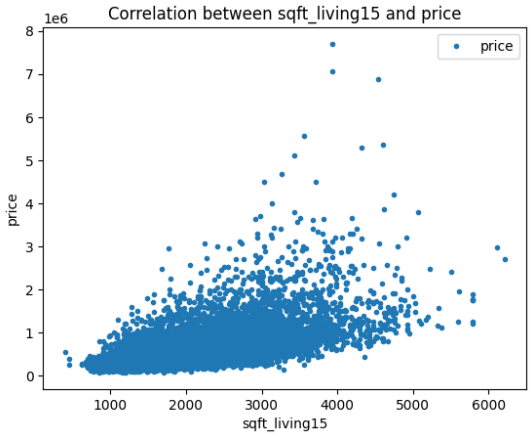
After this visualization, we decided to look at the correlation and Pearson coefficient of each variable and price. The variables of note were bathrooms (Pearson coefficient = 0.525), grade (Pearson coefficient = 0.667), sqft_living (Pearson coefficient = 0.702), sqft_above (Pearson coefficient = 0.606), and sqft_living15 (Pearson coefficient = 0.585).



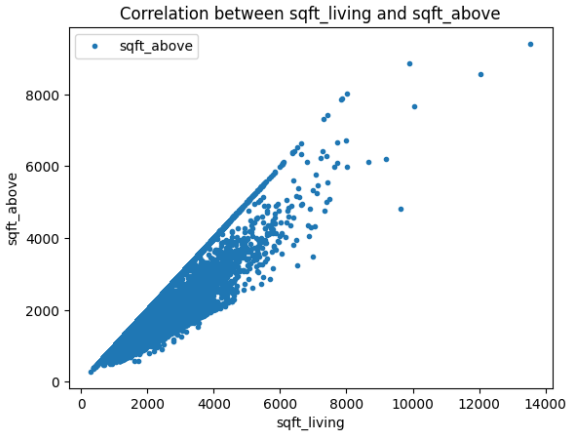






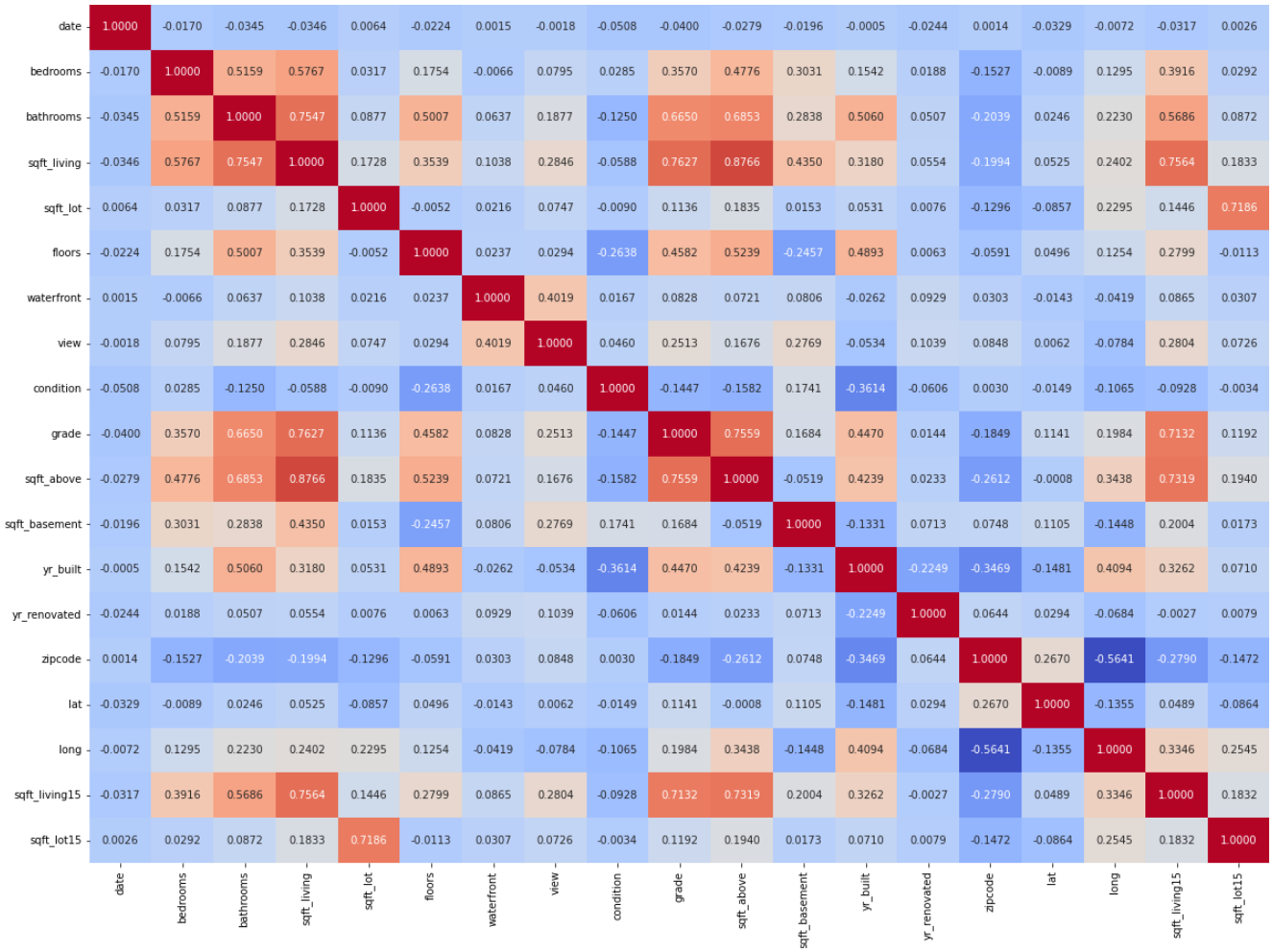


Additionally, there was a strong correlation between sqft_living and sqft_above (Pearson coefficient = 0.877):



This makes sense because if a house doesn't have a basement then sqft_above and sqft_living should be the same value.

The plot below shows the correlation between each pair of independent variables.



Looking at these graphs we also noticed that the correlation seems to change as x increases. This suggested to us that a linear-based regression wouldn't give the best result. We then decided to add random forest regression to the list of regressions we were going to complete.

B. Materials and Methods

Preprocessing:

- Id column contained more than 99% unique values and hence it was removed since it won't be useful to the model.
- The data didn't have any Null values.
- Date column was Label encoded since there is a certain order to date, and the Label encoder preserves this order.
- Zipcode variable was One hot encoded since there is no order to zipcodes, and keeping them as such might make the model give them a certain false order.
- The input variables were scaled to obtain a mean value of 0 and standard deviation value of 1.
- The variable 'sqft_above' is highly correlated with 'sqft_living' (The threshold was set to 0.8). But the variable wasn't removed for modeling to check how Ridge/Lasso regression is able to handle this. The 'sqft_above' variable was removed for other models.

Model Building:

- Five Regression models were built, and performance between these models were evaluated using Cross validation.
- The data was split into 70% training and 30% test data.

1. **Multiple Linear regression**
    - Multiple linear regression is a type of regression used to predict the value of a variable based on the value of another variable. Linear regression comes under the Regression techniques of supervised learning methods. It is used to predict the character and strength of a dependent variable with other independent variables.
    - The data accuracy is measured by taking the distance between the points plotted on the graph to the line y=mx+b or y = b0 + b1x graph.
    - For regression, the best values of variance (R2) are taken. If variance is close to 1, the model has performed well. Other evaluation metrics include Mean Square Error (MSE), Mean Absolute Error (MAE).

2. **K Nearest Neighbors**
    - K Nearest Neighbour is used as a classification model. The K-NN algorithm is used to classify or label data points.

- If the data is numerical in nature, there are many different numbers of known ways to define the distance between two things. E.g. Euclidean Distance, Cosine, Manhattan etc.
- But, the model used for training is the K-NN regression model which slightly varies from K-NN classification model. The K-NN regression model takes the average of the K-NN clusters formed.
- K-NN regression is best used for numerical data types.

3. **Lasso Regression**
   - Lasso (least absolute shrinkage and selection operator) is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model.
   - Lasso regression performs L1 regularization, which adds a penalty equal to the absolute value of the magnitude of coefficients. This type of regularization can result in sparse models with few coefficients. Some coefficients can even become zero and eliminated from the model. Larger penalties result in coefficient values closer to zero, which is the ideal for producing simpler models.
   - The best value of alpha is chosen using the LassoCV method in Python.
   - Using the best alpha obtained, the model is fitted to the training data.

4. **Ridge Regression**
   - Ridge regression is a method of estimating the coefficients of multiple-regression models in scenarios where the independent variables are highly correlated.
   - Ridge regression belongs to a class of regression tools that use L2 regularization. L2 regularization adds an L2 penalty, which equals the square of the magnitude of coefficients. All coefficients are shrunk by the same factor (so none are eliminated). Unlike L1 regularization, L2 will not result in sparse models.
   - The best value of alpha is chosen using the RidgeCV method in Python.
   - Using the best alpha obtained, the model is fitted to the training data.

5. **Random Forest Regression**
   - Random forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. Here we are utilizing it for Regression analysis, where the mean or average prediction of the individual trees is returned.
   - The best parameters for the Random Forest model were found using RandomizedSearchCV method in Python.
   - Using the best parameters obtained, the model is fitted to the training data.

(c) Results

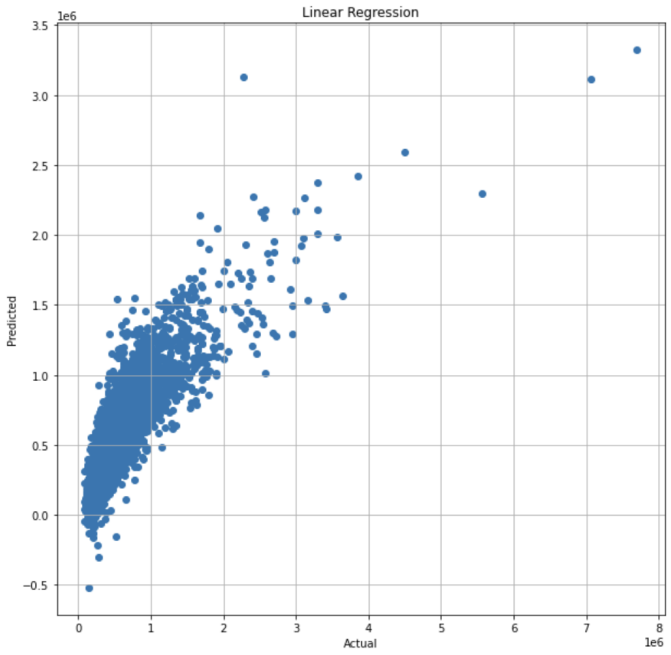
1. **Multiple Linear regression**
   - Four different evaluation metrics have been taken to check the performance of the model trained. Mean Square Error (MSE), Root Mean Square Error (RMSE), R-Squared (R2) and Mean Absolute Error (MAE).
   - Evaluation Metric -

     ```
     MSE:  42963995305.10944
     MAE:  126043.12774403258
     R2 Score :  0.6934591631341724
     RMSE :  207277.5803243309
     ```

     We can observe that R2 is 0.69 which is not close to 1, and isn't as good as Random Forest regression.
   - Predictor Contribution -

	columns	Linear_coef
0	date	131.412008
1	bedrooms	-34087.812231
2	bathrooms	47948.342682
3	sqft_living	185.728494
4	sqft_lot	0.136355
5	floors	-201.769666
6	waterfront	665194.928600
7	view	55027.848050
8	condition	26306.432189
9	grade	94715.213810
10	sqft_basement	-44.028817
11	yr_built	-2658.288905
12	yr_renovated	15.342792
13	zipcode	-1691.417790
14	lat	618145.916699
15	long	-212883.895201
16	sqft_living15	20.721414
17	sqft_lot15	-0.405376

     When taking the coefficients between bedrooms and bathrooms, the price of the house is decreasing when more bedrooms are added when compared to increasing number of bathrooms . This is a weird output obtained by the model.
   - Visualization Graph -



The predicted vs actual data can be observed from the above visualization. When an imaginary y = mx+b line is taken, most of the data is a little far away from the line which shows the data is not modeled properly.

2. K Nearest Neighbors

- The same four different evaluation metrics have been taken to check the performance of the model trained. Mean Square Error (MSE), Root Mean Square Error (RMSE), R-Squared (R2) and Mean Absolute Error (MAE).
- Evaluation Metric -

```
MSE:  82708611258.61243
MAE:  161322.45971314004
R2 Score :  0.4098880532134703
RMSE :  287591.0486413171
```

The R2 score is lowest when compared to all the other models which shows that this type of regression is not suitable for the dataset. Since KNN is a different type of regression compared to the other linear regressions performed in the project.

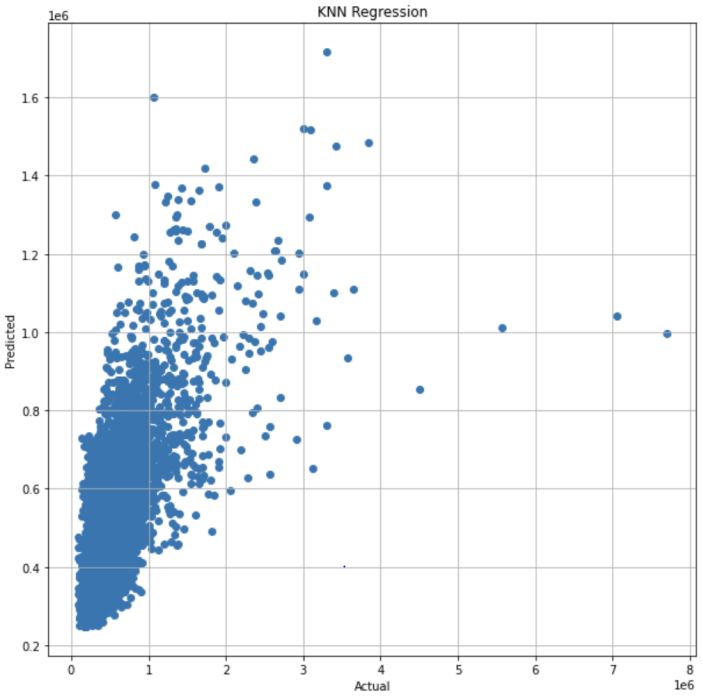
- Predictor Contribution -
  Unlike the linear regression models, the prediction contribution for this regression is stored in a sparse matrix.

```
<6484x15129 sparse matrix of type '<class 'numpy.float64'>'
        with 648400 stored elements in Compressed Sparse Row format>
```

The matrix can be displayed as an array with values between 0 and 1.

```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

- Visualization -



From the above predicted vs actual data, we can observe that the points scattered are more away from the y = mx+b line. This observation along with R2 score shows that K-NN regression is not a suitable model.

3. Lasso Regression

- Three metrics were used to judge the performance of the models : Root mean square error (RMSE), Mean absolute error (MAE) & R-Squared (R2).
- Below are the results for the Train & Test data, as well as Cross validation results.

Lasso Train metrics

	Metric	Value
0	RMSE	151235.932206
1	MAE	92490.326196
2	R2	0.817940

Lasso Test metrics

	Metric	Value
0	RMSE	180418.886333
1	MAE	98377.695619
2	R2	0.791434

```
Cross Validation Scores:  [0.79024002 0.80239119 0.80529852 0.77696371 0.79297241 0.81864429
 0.81811265 0.83593176 0.80650679 0.81784386]
Average CV Score:  0.8064905198486928
```

- The R-Squared scores are similar to the test R-Squared score we got before, meaning the model is generalizing well. The Lasso model is able to handle collinearity in variables better than the Ridge model.
- Below are the predictor contributions of the tuned Lasso model.

	columns	Lasso_coef
1	bedrooms	-21924.140260
5	floors	-21588.424271
11	sqft_basement	-32727.702494
12	yr_built	-20834.679681
15	long	-18511.959308
17	sqft_lot15	-1717.800404
18	zipcode_98001	-14703.241863

	columns	Lasso_coef
3	sqft_living	180848.975467
10	sqft_above	353.074953

- It is interesting to note that variables like bedrooms, floors, yr_built, long etc have a negative correlation to the price variable. It means when the number of bedrooms increase, or the number of floors increase, or the yr_built increases, the price reduces by the coefficient of each associated variable for every one unit increase of the associated variable. It is interesting to note that houses built recently sell less in comparison to old houses.
- Also, the variable ‘sqft_living’ has a high coefficient value in comparison to its correlated variable ‘sqft_above’, which has a small coefficient value, like we expected for Lasso Regression.

4. Ridge Regression

- Three metrics were used to judge the performance of the models : Root mean square error (RMSE), Mean absolute error (MAE) & R-Squared (R2).

- Below are the results for the Train & Test data, as well as Cross validation results.

Ridge Train metrics

	Metric	Value
0	RMSE	151256.521205
1	MAE	92202.886154
2	R2	0.817891

Ridge Test metrics

	Metric	Value
0	RMSE	180618.561321
1	MAE	98070.429572
2	R2	0.790972

```
Cross Validation Scores:  [0.75366342 0.76178453 0.76794843 0.75197049 0.76019774 0.78911627
 0.79068532 0.79661746 0.77320446 0.76296432]
Average CV Score:  0.7708152441775755
```

- The R-Squared scores are less in comparison to the test R-Squared score we got before, meaning the model isn't generalizing well.
- Below are the predictor contributions of the tuned Ridge model.

	columns	Ridge_coef
1	bedrooms	-21156.071721
5	floors	-20222.767761
12	yr_built	-21431.588613
15	long	-17938.682435
17	sqft_lot15	-1663.415153
18	zipcode_98001	-15969.103784

	columns	Ridge_coef
3	sqft_living	83735.909692
10	sqft_above	84944.243422

- It is interesting to note that variables like bedrooms, floors, yr_built, long etc have a negative correlation to the price variable. It means when the number of bedrooms increase, or the number of floors increase, or the yr_built increases, the price reduces by the coefficient of each associated variable for every one unit increase of the associated variable. It is interesting to note that houses built recently sell less in comparison to old houses.
- Also, the two correlated variables 'sqft_above', 'sqft_living' have similar coefficients, like we expected for Ridge Regression.

5. Random Forest Regression

- Three metrics were used to judge the performance of the models : Root mean square error (RMSE), Mean absolute error (MAE) & R-Squared (R2).
- Below are the results for the Train & Test data, as well as Cross validation results.

RandomForest Train metrics

	Metric	Value
0	RMSE	45555.756671
1	MAE	158.508305
2	R2	0.983481

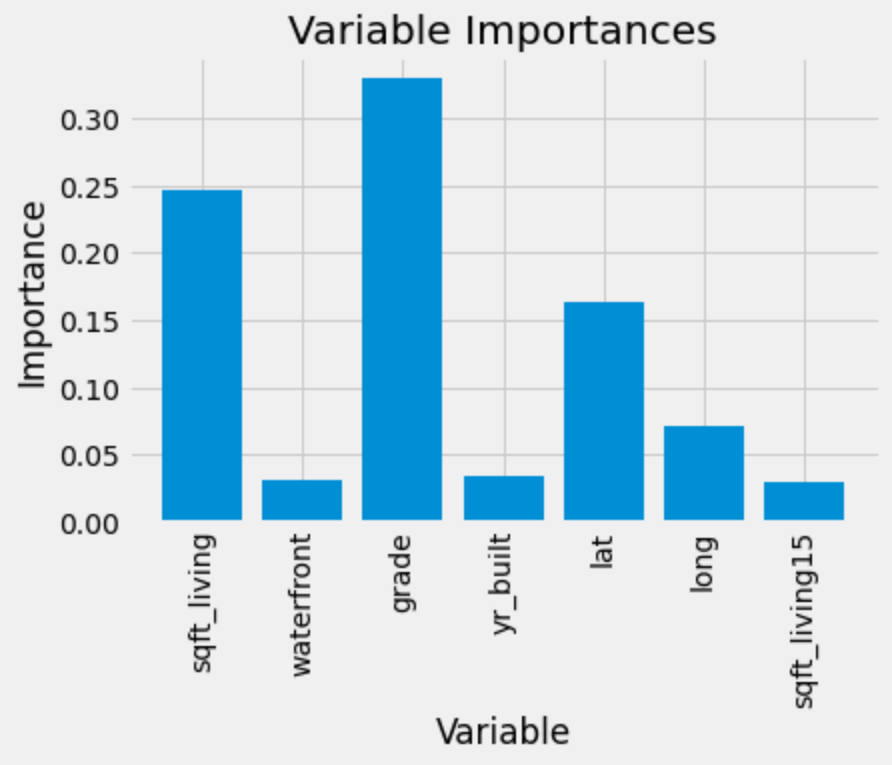
RandomForest Test metrics

	Metric	Value
0	RMSE	146123.935225
1	MAE	265.770722
2	R2	0.863189

```
Cross Validation Scores:  [0.87772059 0.87395117 0.89132379 0.89075401 0.88726113 0.85669253
 0.84819479 0.89762135 0.89130294 0.87777127]
Average CV Score:  0.8792593556465482
```

- The R-Squared scores are similar to the test R-Squared score we got before, meaning the model is generalizing well. The Lasso model is able to handle collinearity in variables better than the Ridge model.
- Below are the predictor contributions of the tuned Random Forest model.

	columns	RandomForest_coef
3	sqft_living	0.246943
6	waterfront	0.031554
9	grade	0.330070
11	yr_built	0.034564
13	lat	0.162877
14	long	0.071491
15	sqft_living15	0.030400

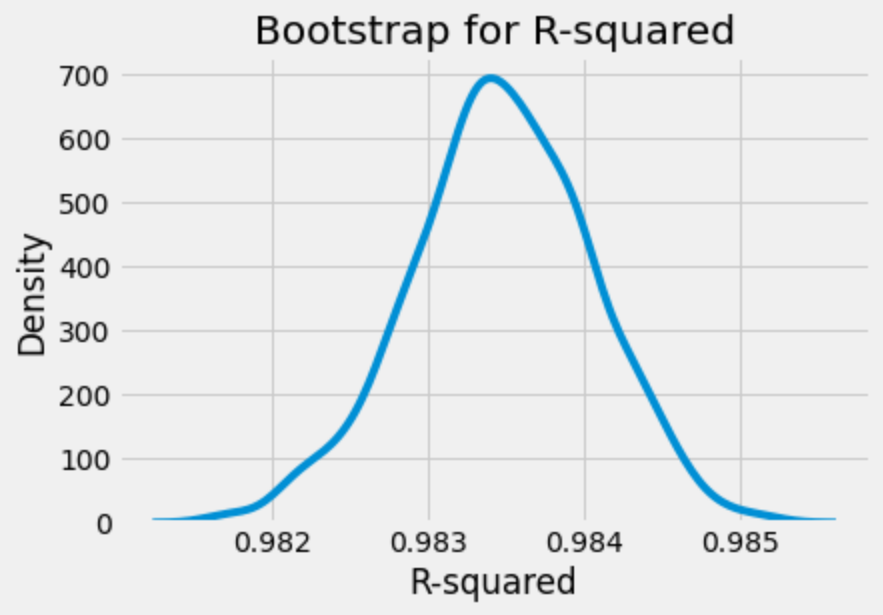


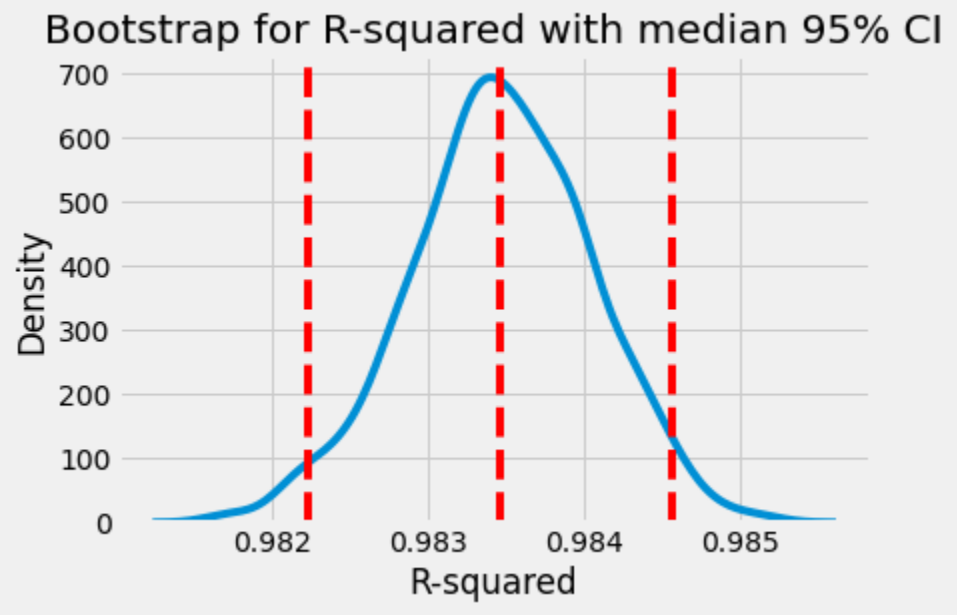
- The most important variable for the model is grade (0.33), and the subsequent important variables were sqft_living (0.24) and lat (0.16). Some of the least important variables are yr_built, waterfront, long, sqft_living15. The other predictors had almost 0 variable importances.

From the R-Squared values, we can conclude that the Random Forest model performed the best for this dataset as it had the highest test and Cross validated R-Squared values among all the models.

Estimation stability

We performed estimation stability of the random forest regression model because that model performed the best. Using 1000 iterations, we used bootstrap to determine the 95% confidence interval for R-Squared on our random forest model using our dataset. We found it performed very well, with a median R-Squared of 0.98 and a 95% confidence interval of (0.982, 0.985)



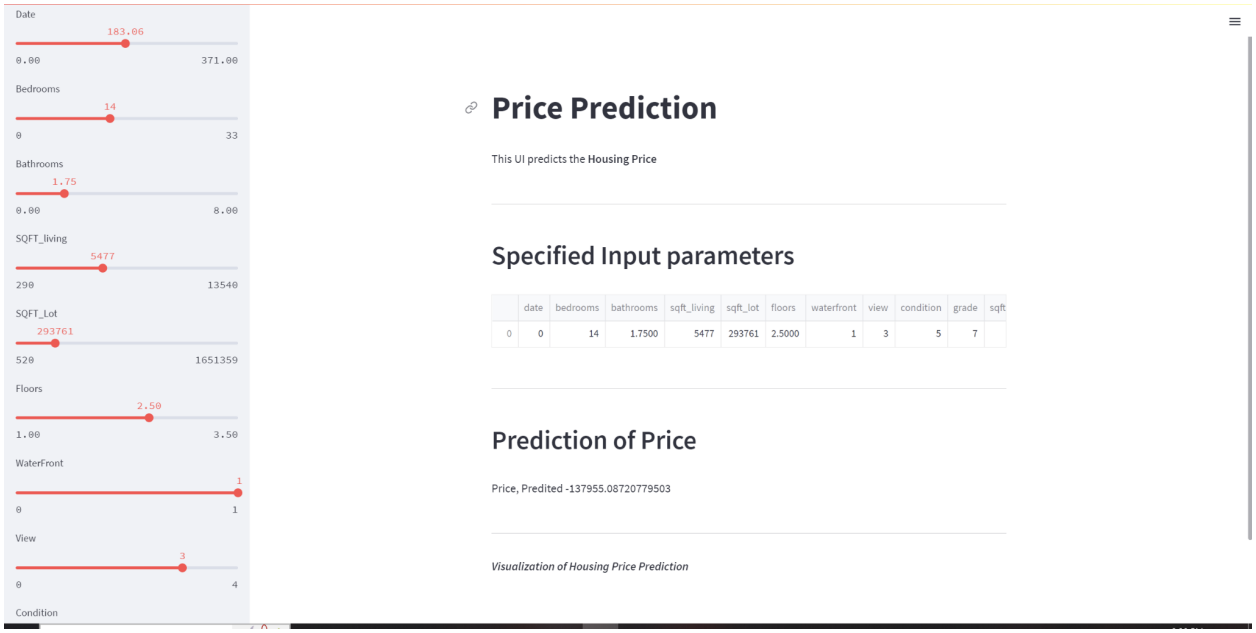


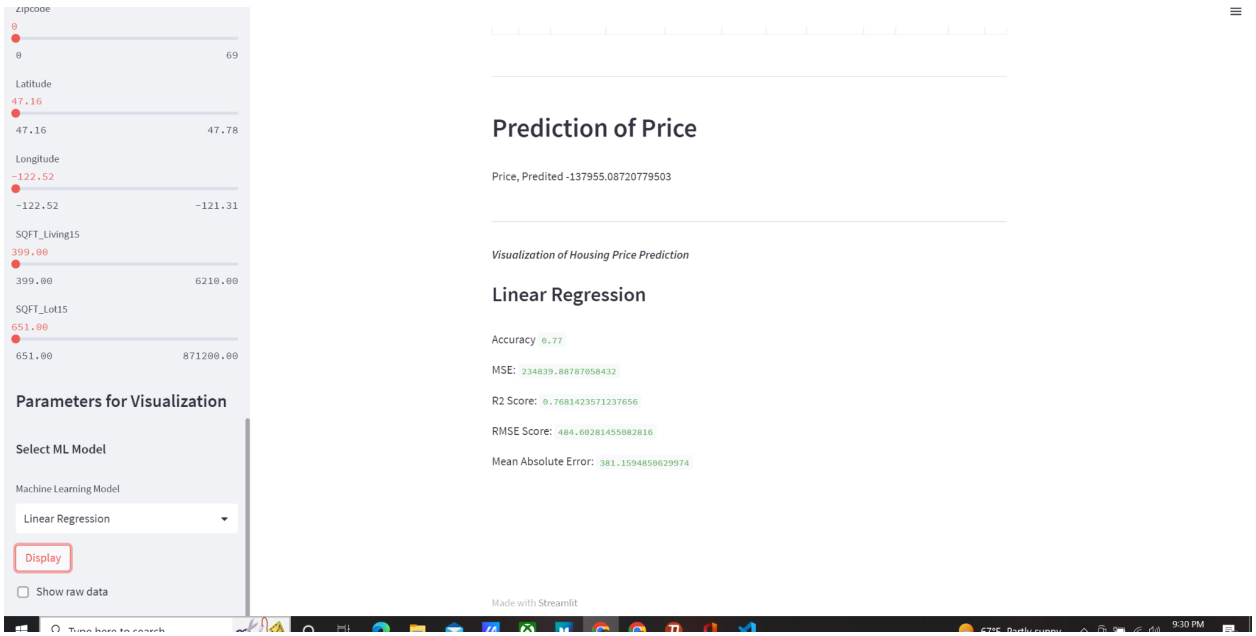
Additional:

UI Webpage:

A web page has been designed to display the output price of the house based on the given parameters that can be changed in real time. Also, the evaluation metric of the different models used in the project have been displayed to verify the differences. The webpage has been designed in python using the Streamlit framework.

Screenshots -





Conclusion:

The Random Forest Regression model gave the best performance on the test data with R-Squared = 0.86. We performed bootstrap on our Random Forest model and found a median R-Squared of 0.98 over 1000 iterations. The model seems to generalize well, since the mean R-Squared for 10 Cross validation folds was 0.879. This means that the model isn't overfitting and generalizing well.