



Predicting Housing Prices

ST1511

Goh Kun Ming
DAAA/FT/1B/02



Data Cleaning & Analysis



No Missing Data

```
# Printing of total data that are missing in each column
df.isnull().sum()
```

```
City          0
House Area (sqm)  0
No. of Bedrooms  0
No. of Toilets  0
Stories        0
Renovation Status  0
Price ($)      0
dtype: int64
```

No Duplicated Data

```
# Count number of duplicated data rows
num_duplicates = df.duplicated().sum()
```

```
# Printing number of duplicated data rows
print(f"Number of duplicated rows: {num_duplicates}")
```

Number of duplicated rows: 0

Dropping Exclusive Data

```
# df will be redefined to the data columns that I will be using
df = df.iloc[:, 1:8]

# Viewing of the first 10 rows of the new DataFrame
df.head(10).style.background_gradient(cmap="Blues")
```

	City	House Area (sqm)	No. of Bedrooms	No. of Toilets	Stories	Renovation Status	Price (\$)
0	Chicago	742.000000	4	2	3	furnished	1330000
1	Denver	896.000000	4	4	4	furnished	1225000
2	Chicago	996.000000	3	2	2	semi-furnished	1225000
3	Seattle	750.000000	4	2	2	furnished	1221500
4	New York	742.000000	4	1	2	furnished	1141000
5	Boston	750.000000	3	3	1	semi-furnished	1085000
6	Denver	858.000000	4	3	4	semi-furnished	1015000
7	New York	1620.000000	5	3	2	unfurnished	1015000
8	Denver	810.000000	4	1	2	furnished	987000
9	Seattle	575.000000	3	2	4	unfurnished	980000

Feature Engineering (Dummy Variables)



Dummy Variable for Cities

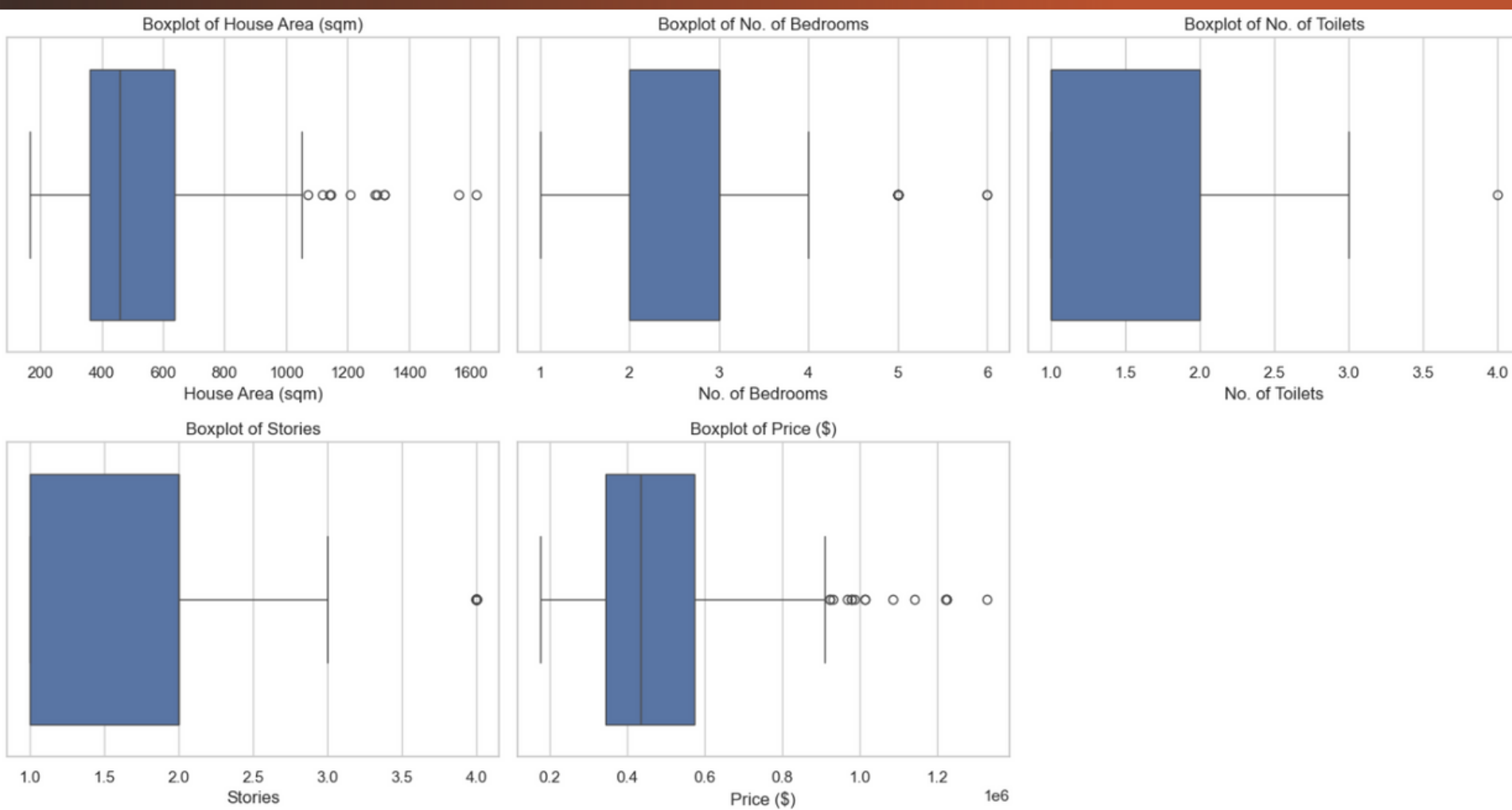
City_Chicago	City_Denver	City_New York	City_Seattle
1	0	0	0
0	1	0	0
1	0	0	0
0	0	0	1
0	0	1	0
0	0	0	0
0	1	0	0
0	0	1	0
0	1	0	0
0	0	0	1

Dummy Variable for Renovation Status

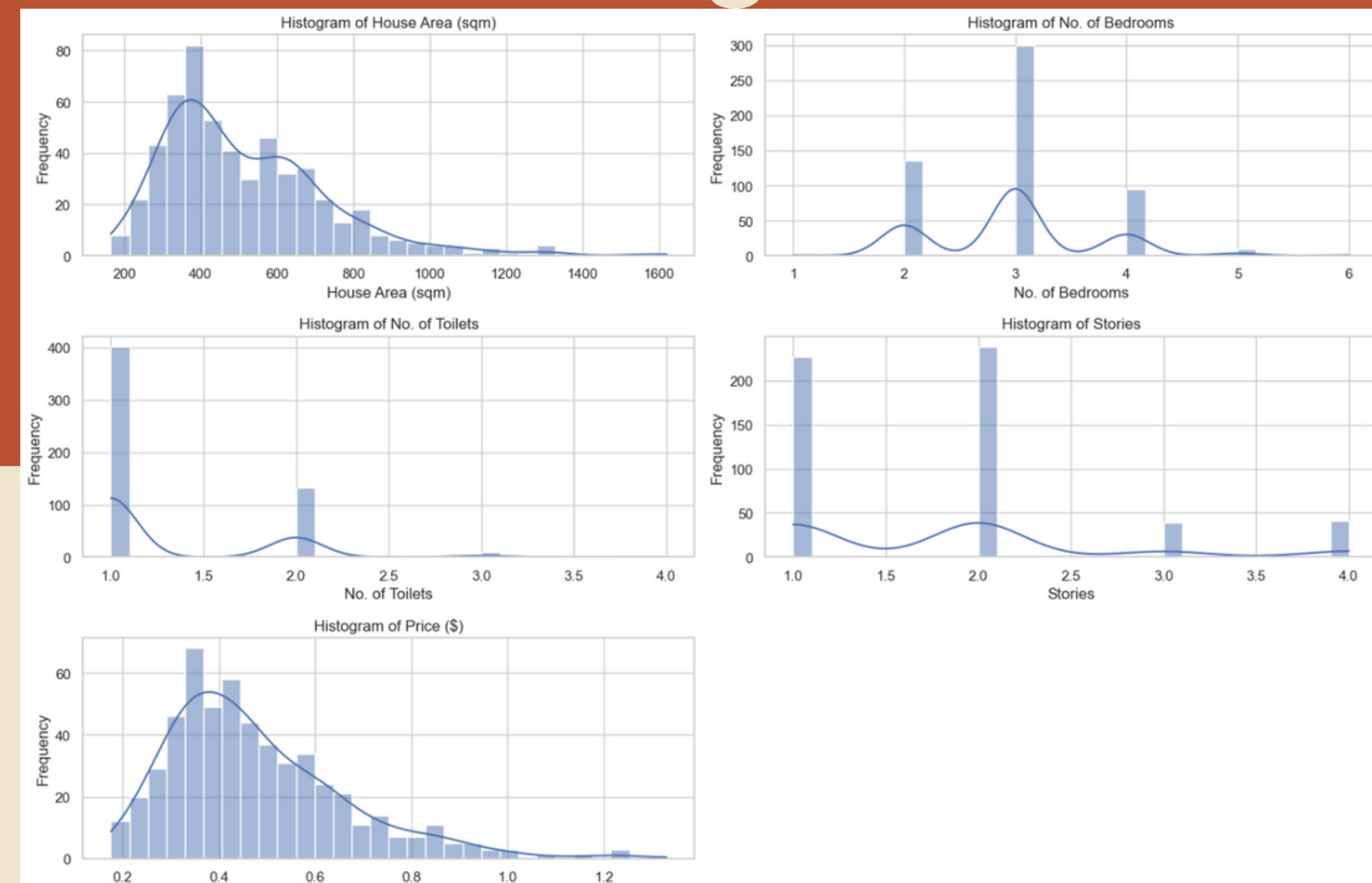
Renovation Status_semi-furnished	Renovation Status_unfurnished
0	0
0	0
1	0
0	0
0	0
1	0
1	0
0	1
0	0
0	1

Feature Engineering (Outliers Identification) 🏠

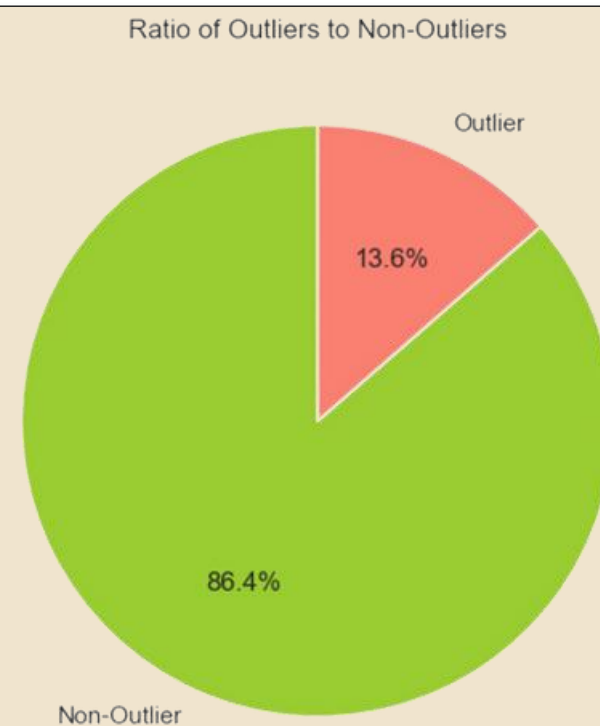
Box Plot



Histograms



Outliers Ratio:



**Not Removing Outliers
due to Limited Data.**

Feature Engineering (Feature Creation)



$$\text{House Area per Bedroom} = \frac{\text{House Area (sqm)}}{\text{No. of Bedrooms}}$$

$$\text{Toilet to Bedroom Ratio} = \frac{\text{No. of Toilets}}{\text{No. of Bedrooms}}$$

$$\text{Total Rooms} = \text{No. of Toilets} + \text{No. of Bedrooms}$$

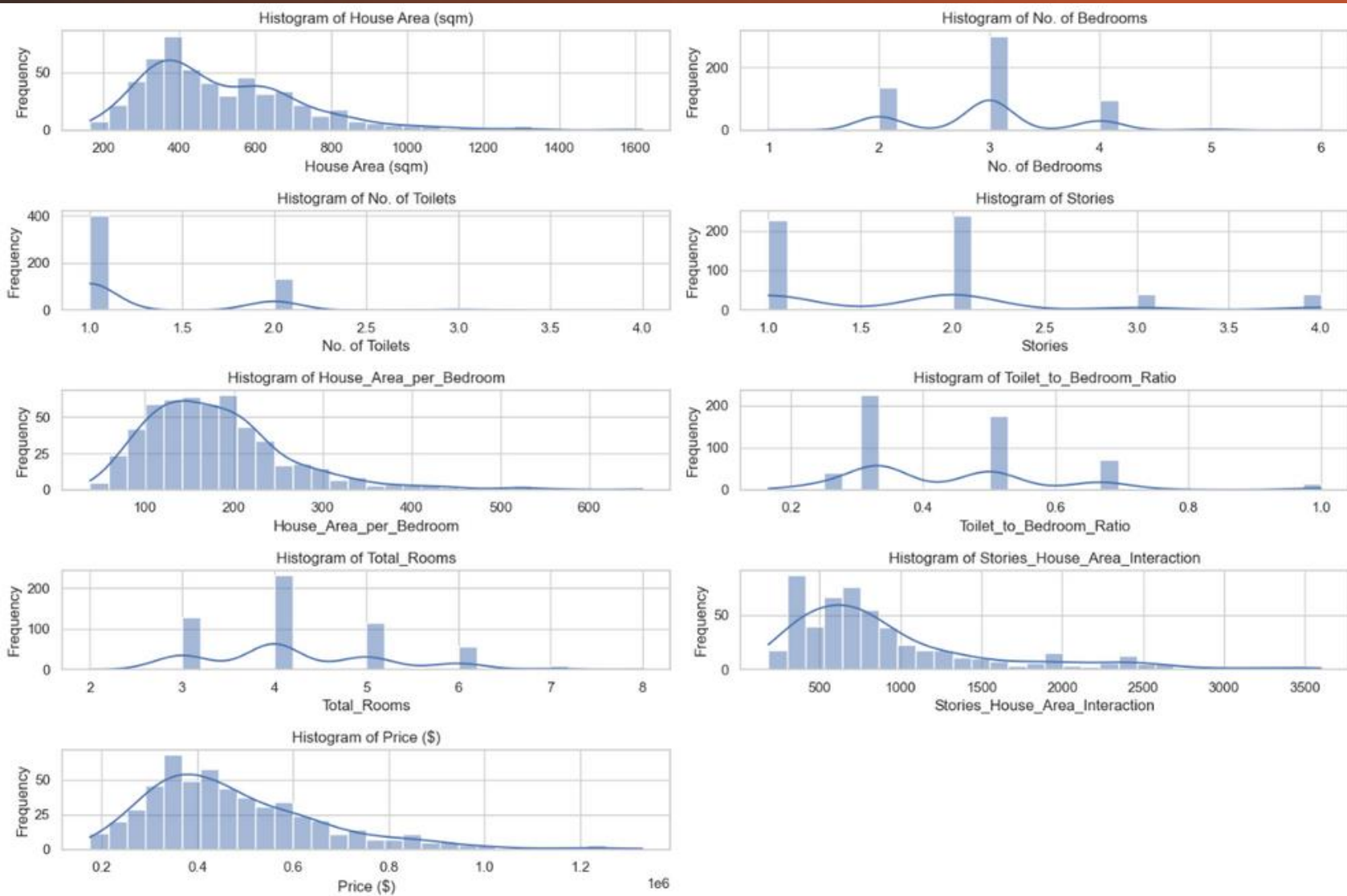
$$\text{Stories House Area Intersection} = \text{Stories} \times \text{House Area (sqm)}$$

House_Area_per_Bedroom	Toilet_to_Bedroom_Ratio	Total_Rooms	Stories_House_Area_Interaction
185.500000	0.500000	6	2226.000000
224.000000	1.000000	8	3584.000000
332.000000	0.666667	5	1992.000000
187.500000	0.500000	6	1500.000000
185.500000	0.250000	5	1484.000000
250.000000	1.000000	6	750.000000
214.500000	0.750000	7	3432.000000
324.000000	0.600000	8	3240.000000
202.500000	0.250000	5	1620.000000
191.666667	0.666667	5	2300.000000

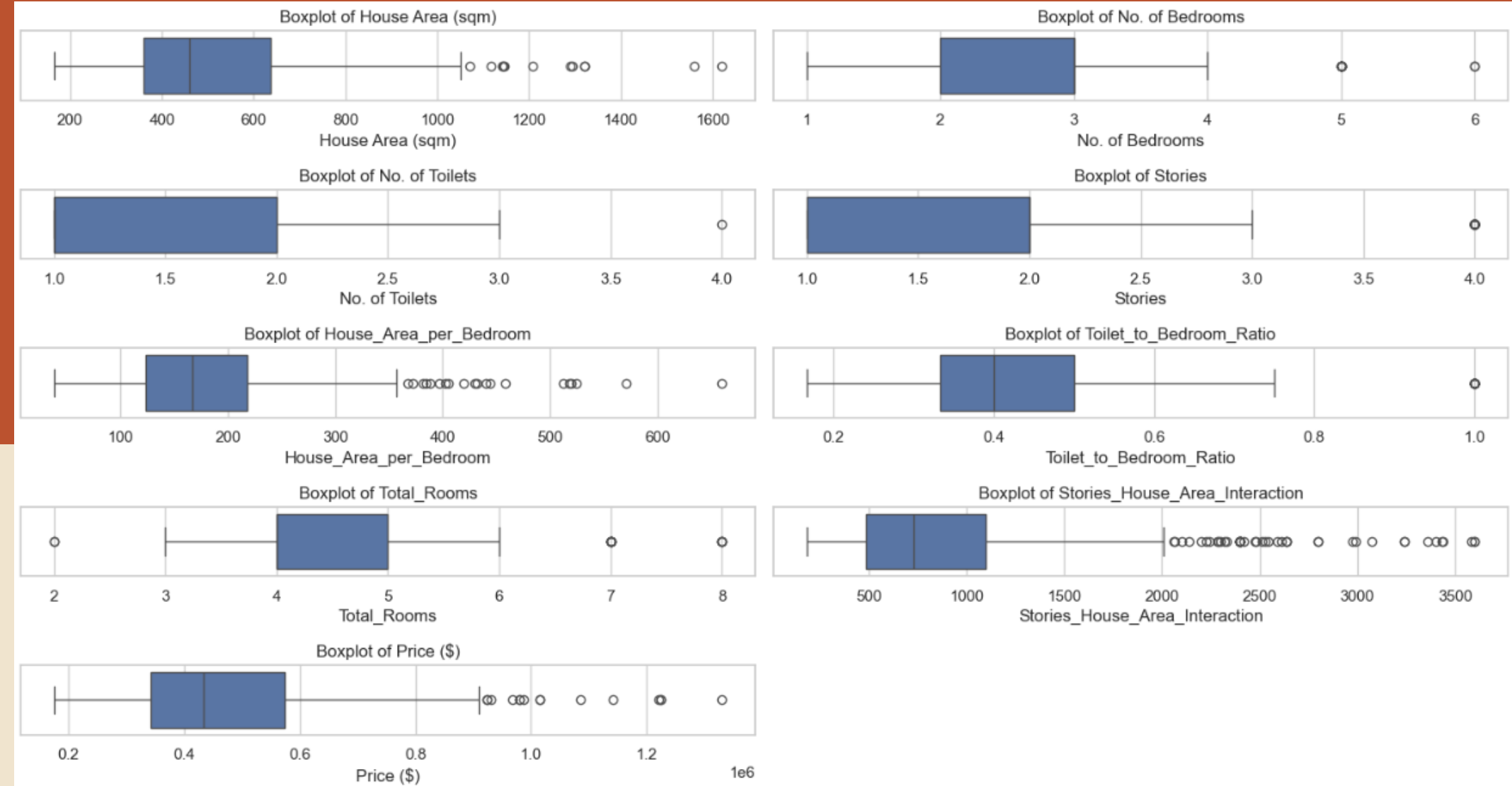
Data Visualisation (Univariate Analysis)



Histogram



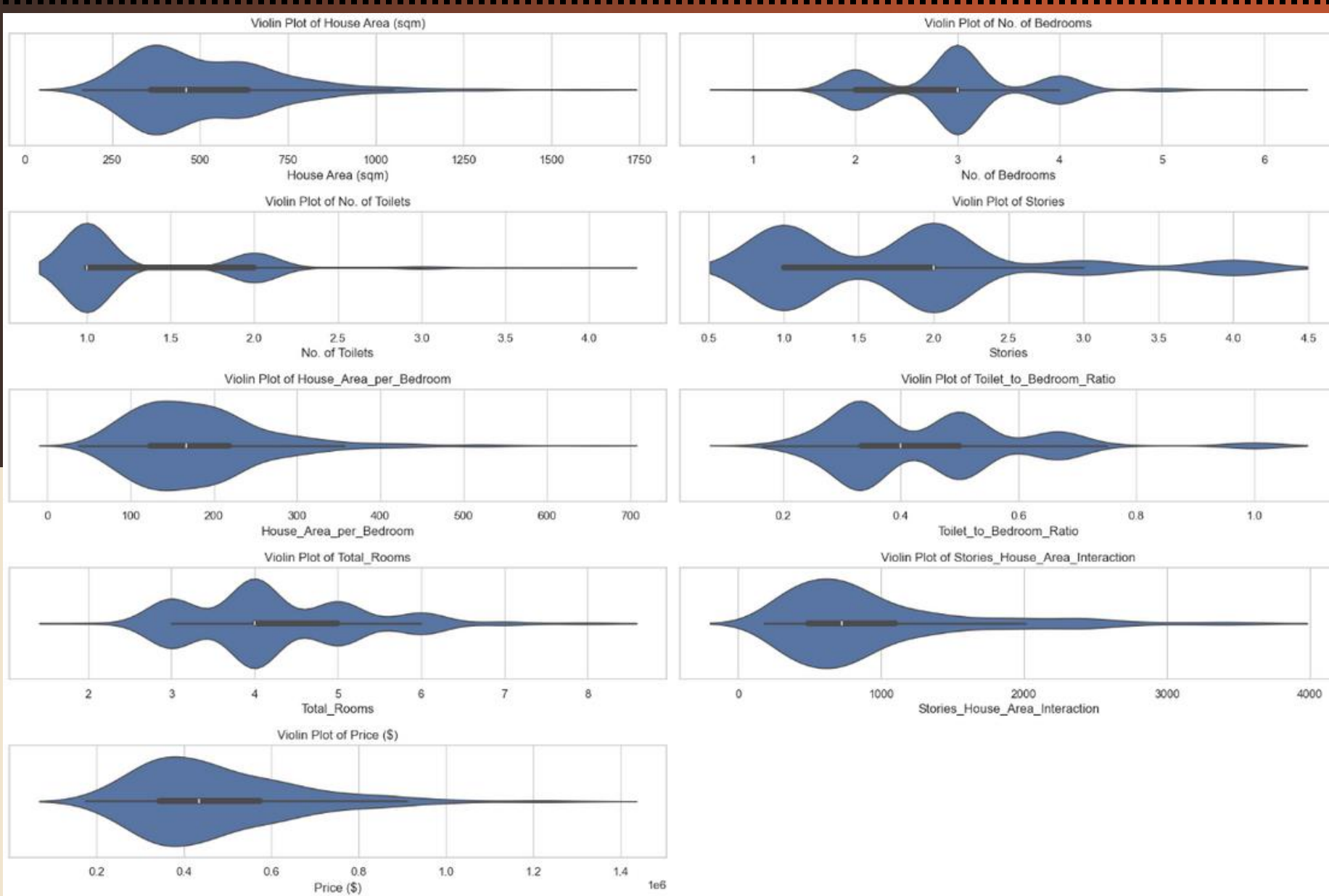
Box Plot



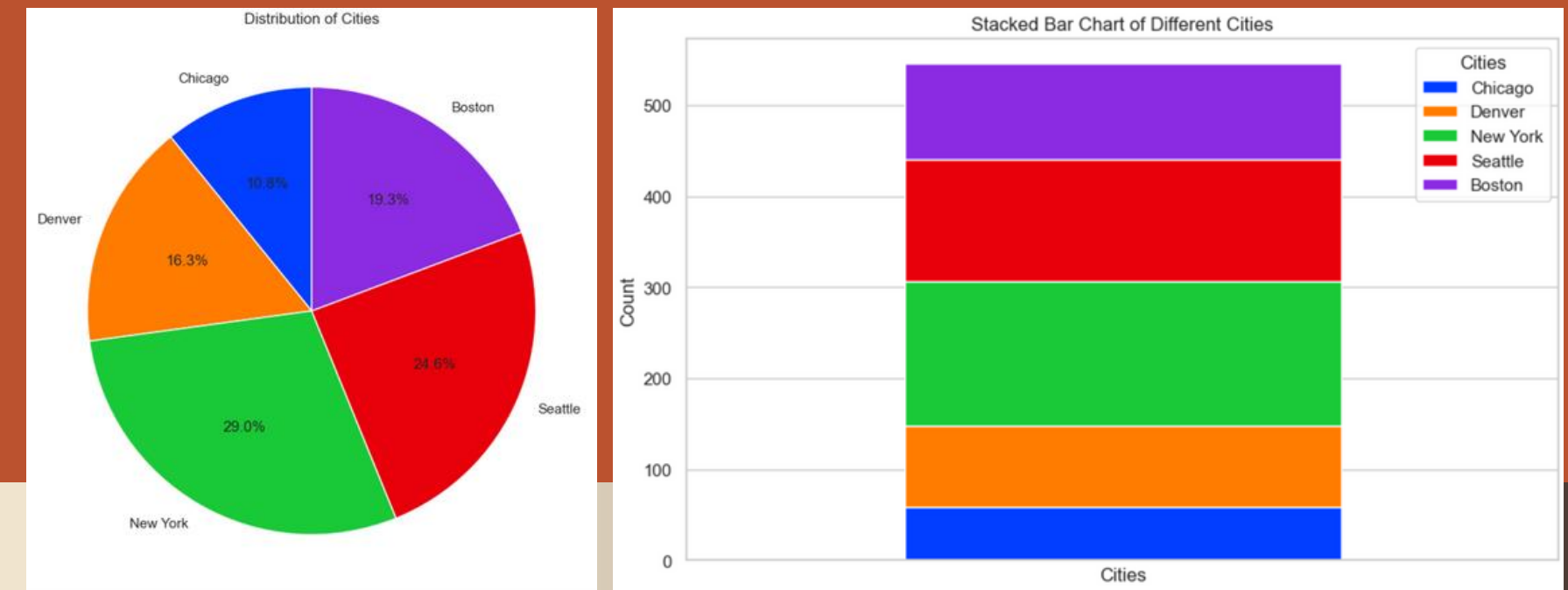
Data Visualisation (Univariate Analysis)



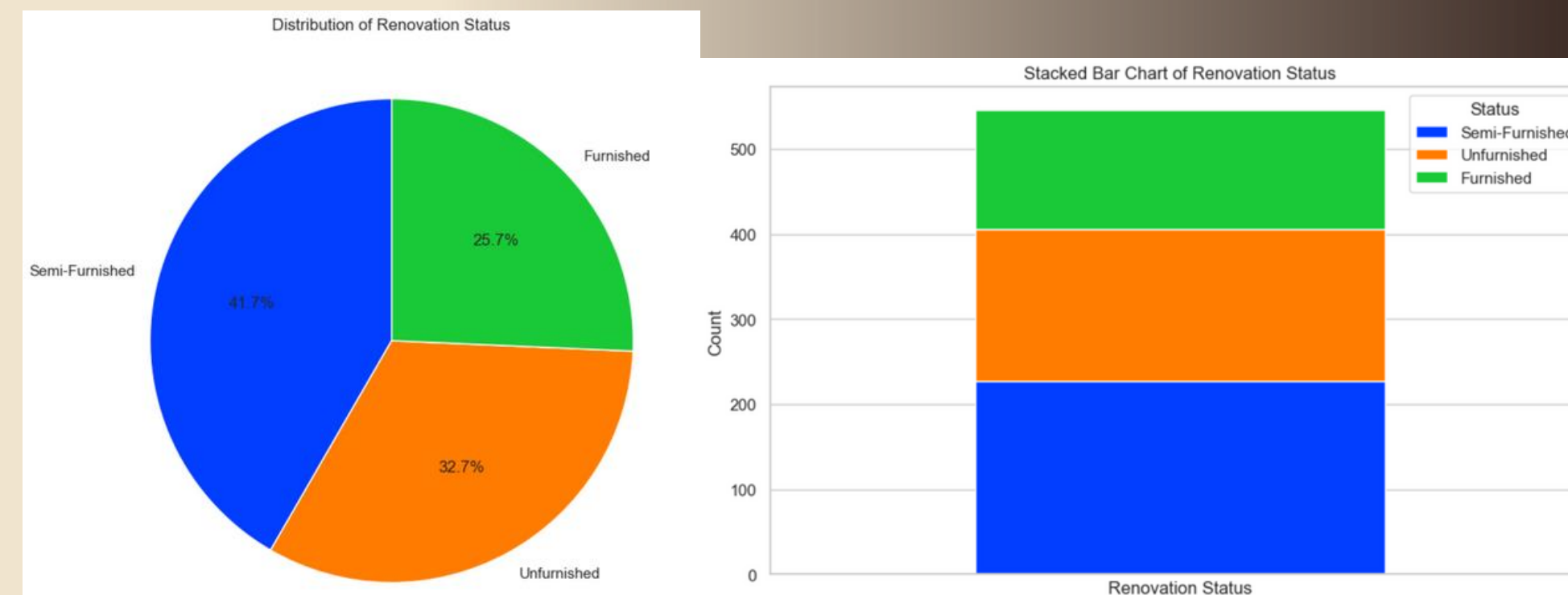
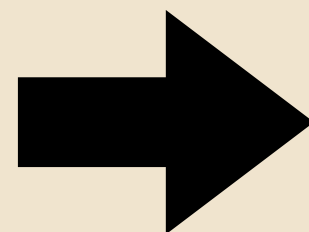
Violin Plot



Cities Pie Chart & Stacked Bar Chart



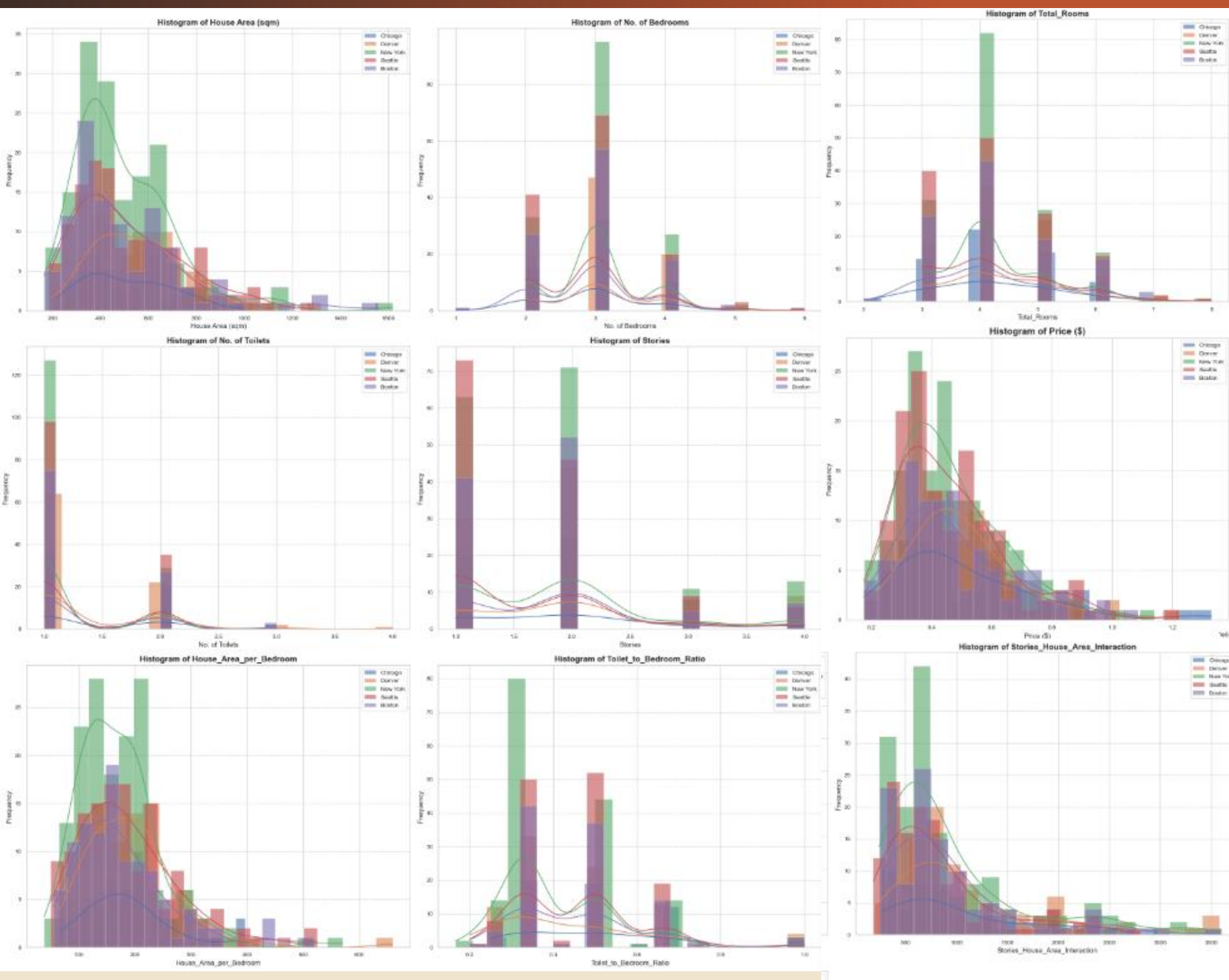
Renovation Status Pie Chart & Stacked Bar Chart



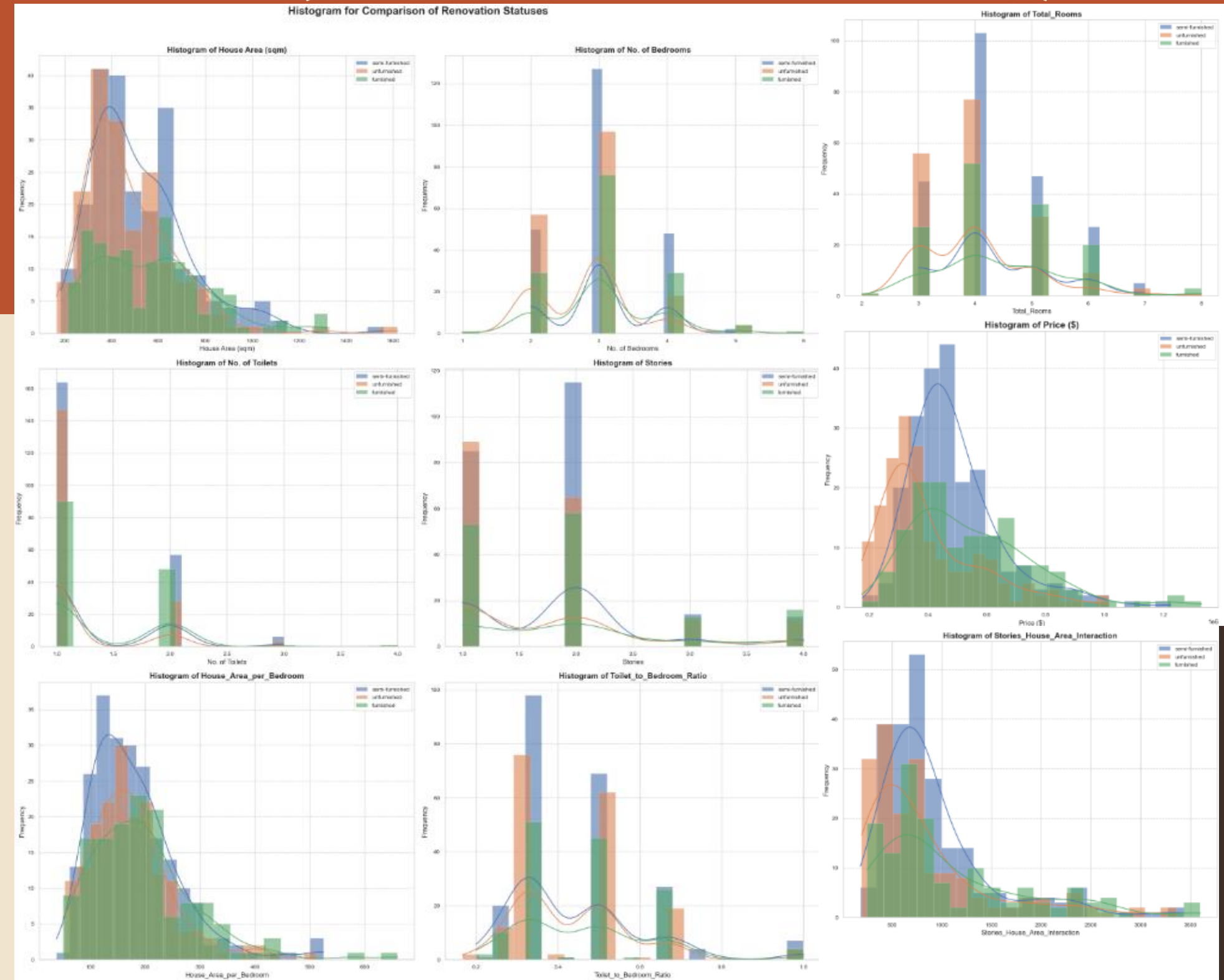
Data Visualisation (Bivariate Analysis)



Histogram (Cities)



Histogram (Renovation Status)

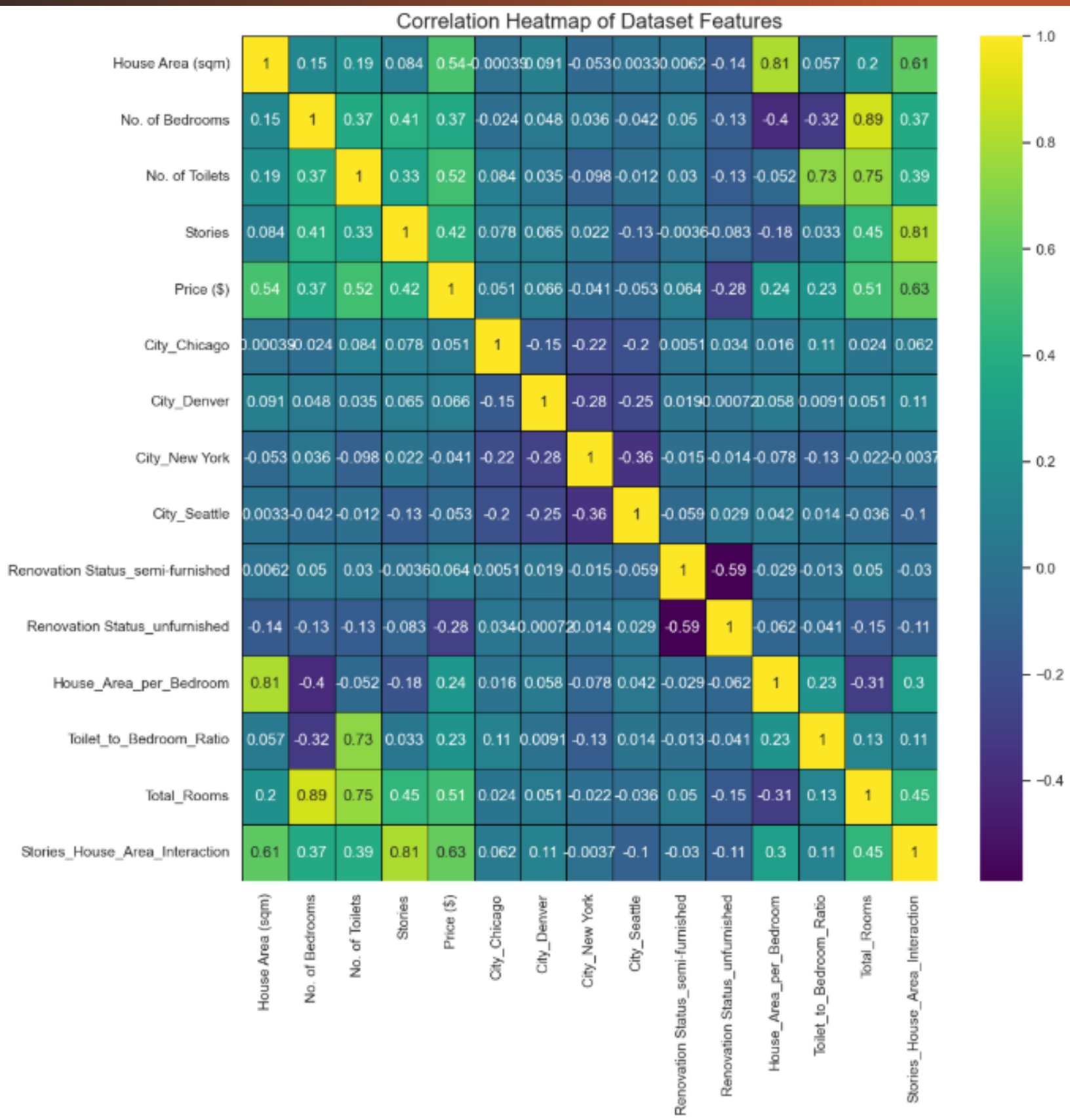




Data Visualisation (Bivariate Analysis)

Correlation Heatmap

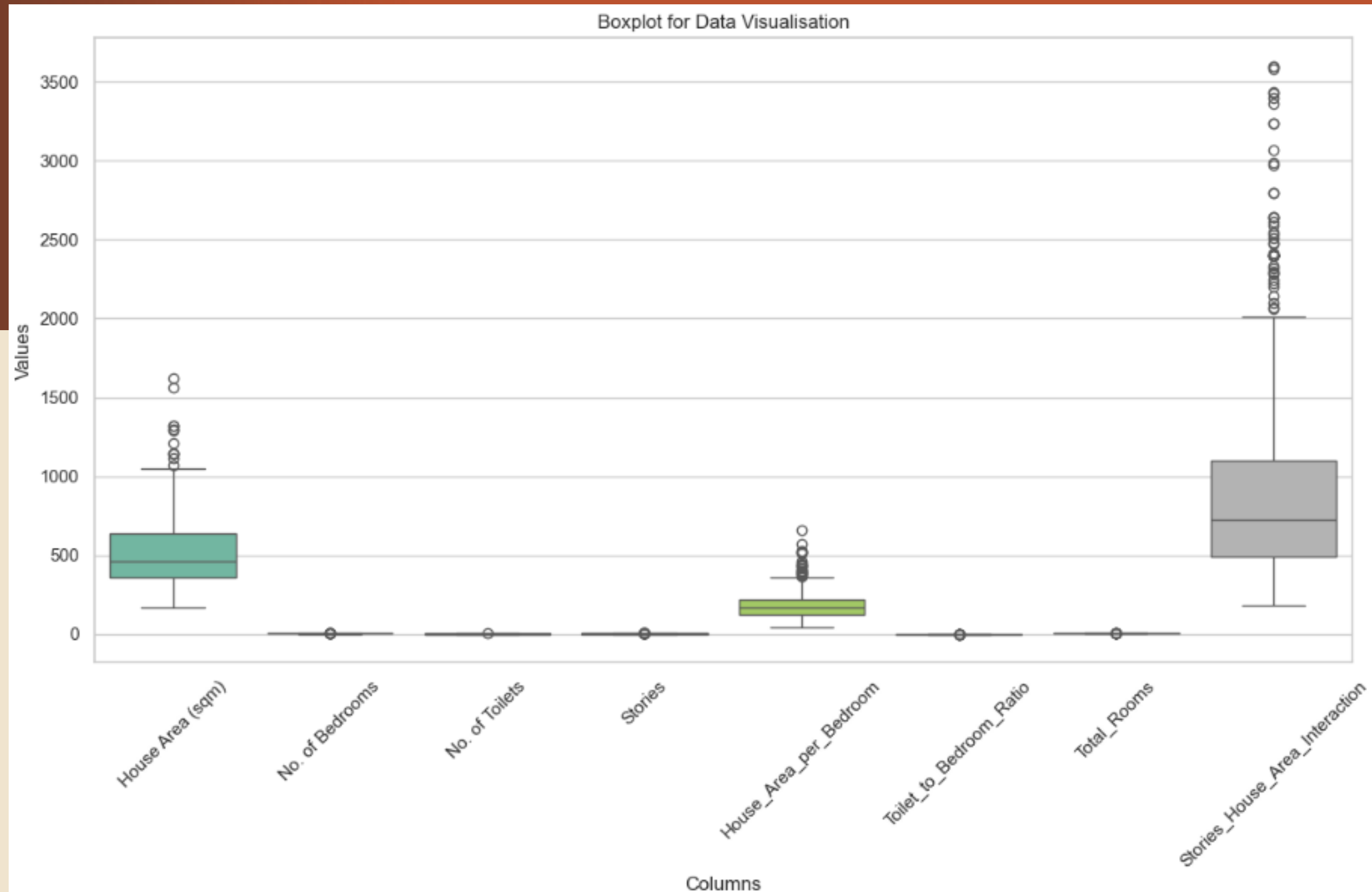
Pairplot (With Target Colour Code)



Data Visualisation (Bivariate Analysis)



Box Plot



Data Preprocessing and Dummy Baseline



Separation of Feature and Target

```
# X = Features
X = df.drop("Price ($)", axis = 1)

# Y = Target
Y = df["Price ($)"]

# Execution Confirmation
print("Feature and Target Separation Successfully Executed")
```

Feature and Target Separation Successfully Executed

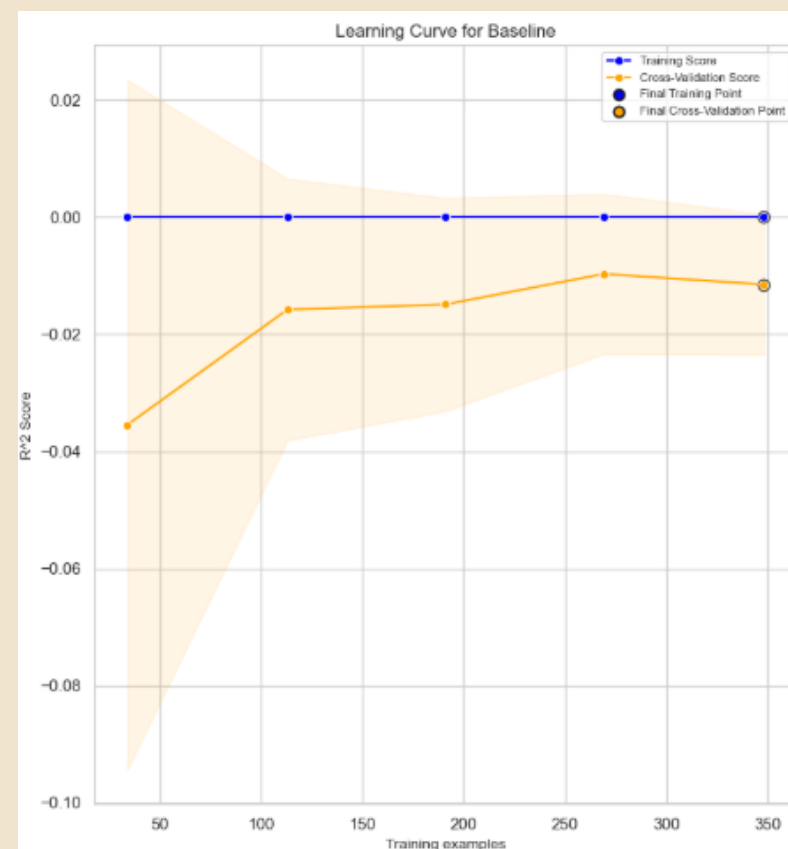
Train-Test Split

```
# Splitting of Trianing Data and Testing Data
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, shuffle = True, random_state = 28)

# Execution Confirmation
print("Train Test Split Successfully Executed")
```

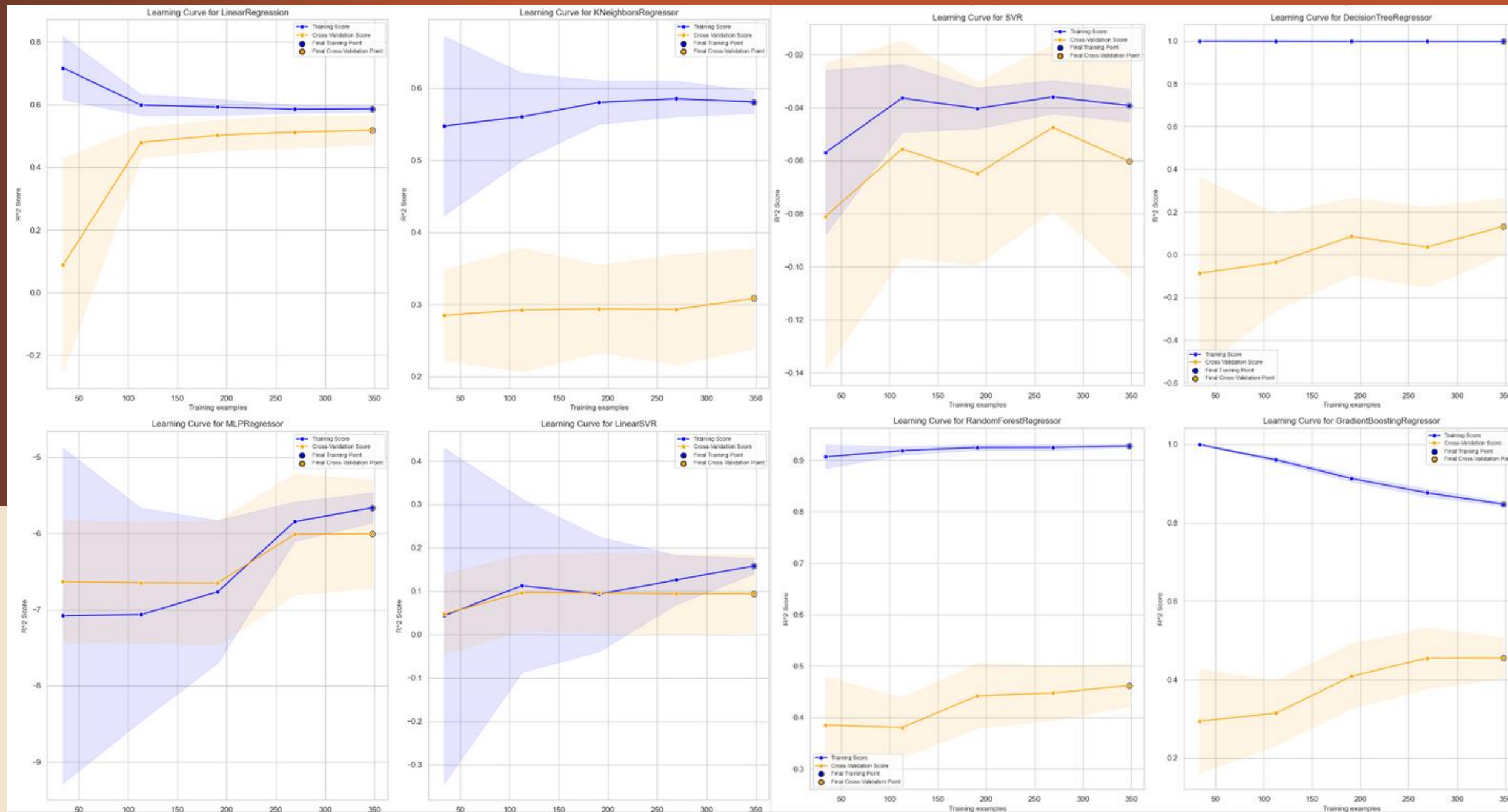
Train Test Split Successfully Executed

Dummy Baseline



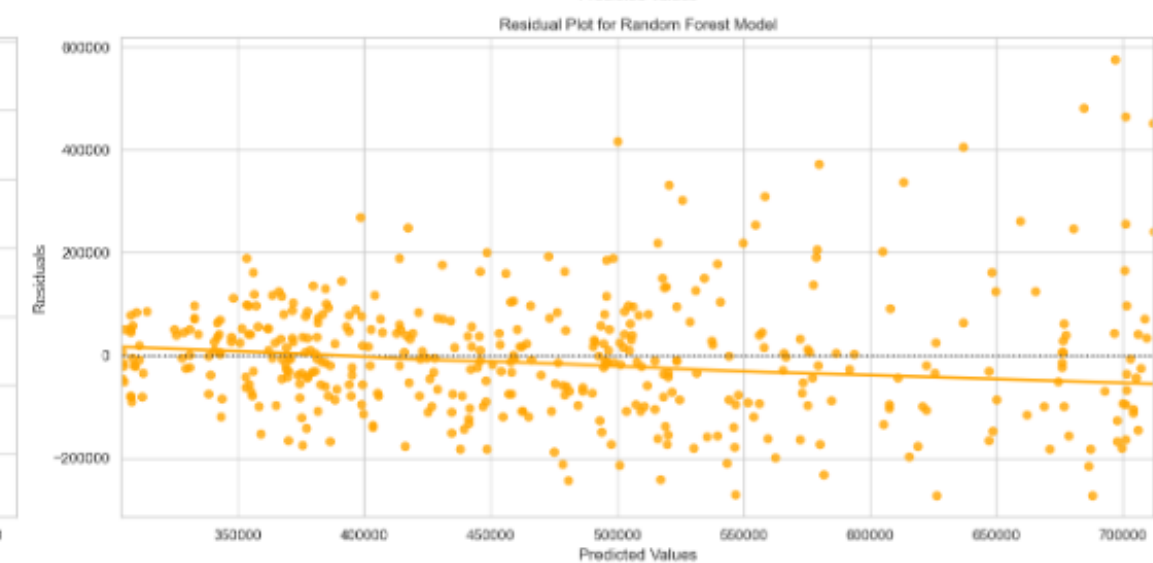
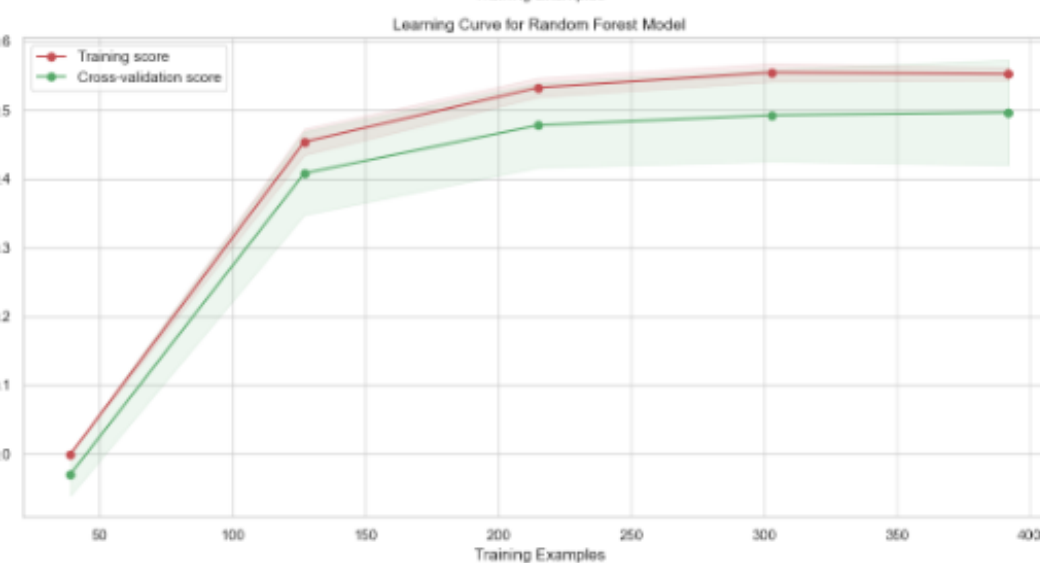
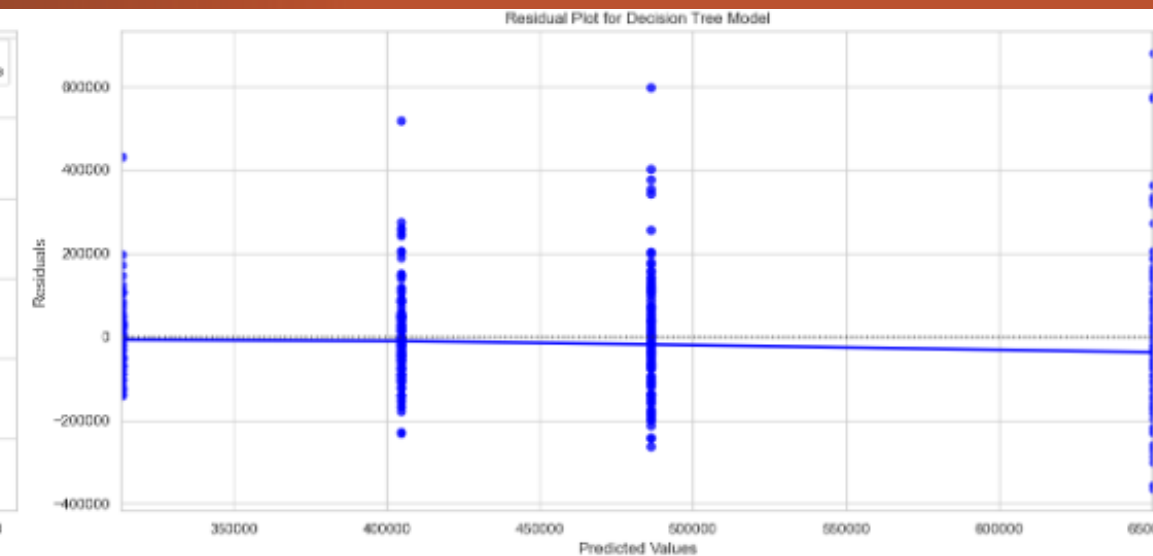
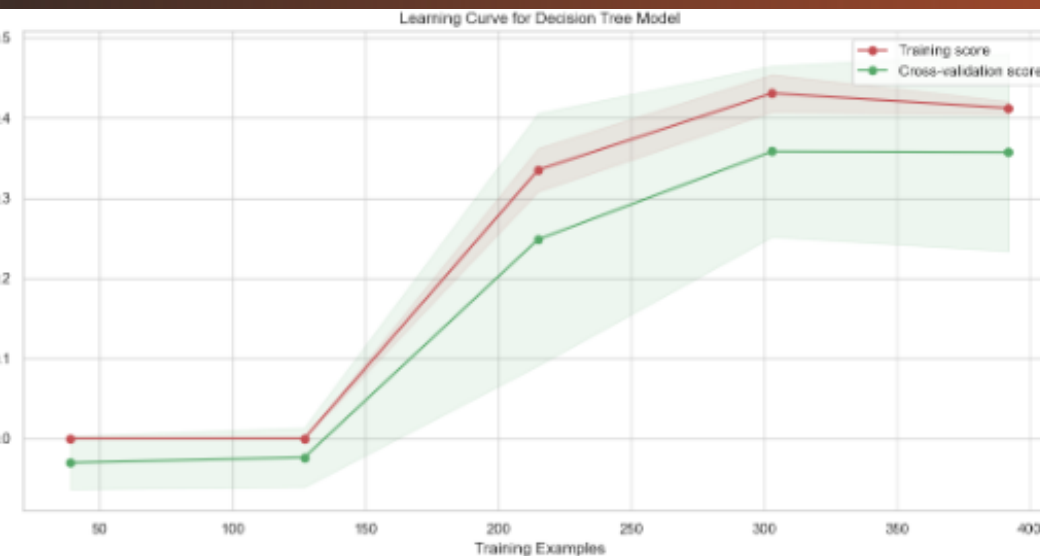
	Mean Squared Error	Root Mean Squared Error	Mean Absolute Error	Mean Absolute Percentage Error	Explained Variance Score	R-Squared Score
Model						
Dummy Regressor	3.813094e+10	195271.442819	154755.718668	0.327846	-2.220446e-16	-0.01133

Model Training

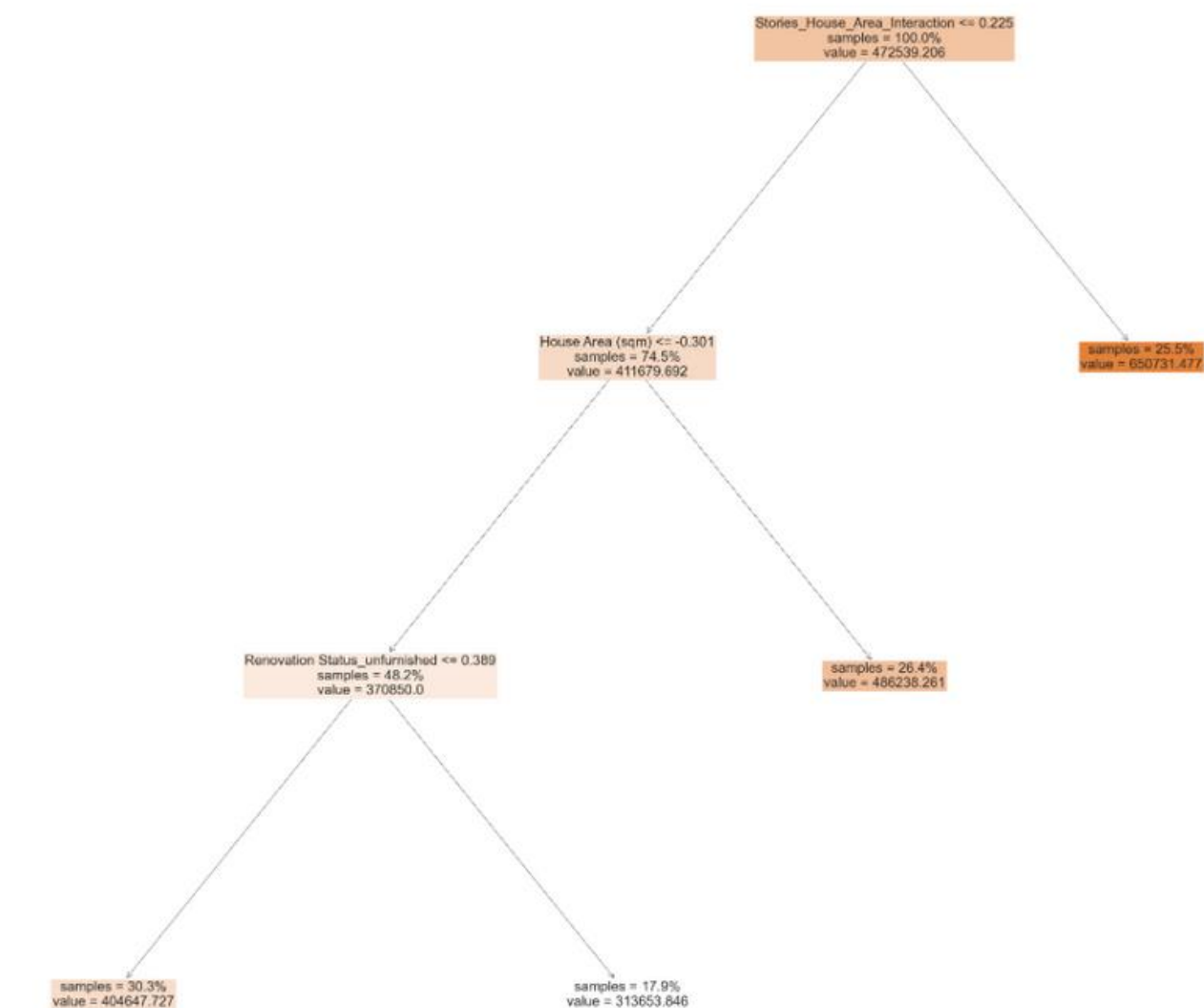


	fit_time	score_time	test_MAE	train_MAE	test_MSE	train_MSE	test_RMSE	train_RMSE	test_R2	train_R2	test_MAPE	train_MAPE	test_EVS	train_EVS
model														
LinearRegression	0.008046	0.003690	93553.743730	88160.958105	16179451609.653097	14121868215.156702	126813.718898	118809.045645	0.518543	0.586864	0.211662	0.200163	0.529286	0.586864
KNeighborsRegressor	0.002801	0.003618	109564.361136	86660.570057	23188654358.532410	14322733310.682678	151849.161137	119657.504558	0.308369	0.580756	0.246807	0.197176	0.318706	0.580871
MLPRegressor	0.114759	0.003000	448242.373043	442244.896203	231595750286.174255	226262414552.312317	480953.624679	475636.758660	-5.975357	-5.622224	0.938302	0.937233	0.095058	0.103287
LinearSVR	0.004371	0.003266	127657.683808	125423.860593	30302587731.855919	28788555850.156055	173693.327773	169648.652285	0.094275	0.157599	0.272306	0.270457	0.126324	0.167555
SVR	0.004521	0.003986	137462.576613	137384.015907	35776117457.523712	35525294315.161705	188533.292664	188448.933911	-0.060316	-0.039173	0.301398	0.303887	0.000214	0.000240
DecisionTreeRegressor	0.002597	0.002554	120269.031061	1067.567050	27742998867.694447	57552424.329502	166368.335695	7314.774932	0.164684	0.998309	0.262900	0.002435	0.177104	0.998309
RandomForestRegressor	0.182302	0.005285	97031.296362	35631.239228	18370291159.320869	2435409548.489998	135115.254217	49337.273401	0.454228	0.928756	0.213148	0.079927	0.466167	0.928776
GradientBoostingRegressor	0.070531	0.002741	99857.642861	55625.344243	18331597224.331963	5207383315.717894	135040.663123	72135.959639	0.453296	0.847694	0.221468	0.131679	0.466632	0.847694
Baseline	0.000542	0.001450	139287.828677	140595.446875	34046555803.361290	34183287529.111183	184059.699011	184859.877007	-0.011487	0.000000	0.330643	0.336658	0.000000	-0.000000

Model Comparison (RFR and DT)



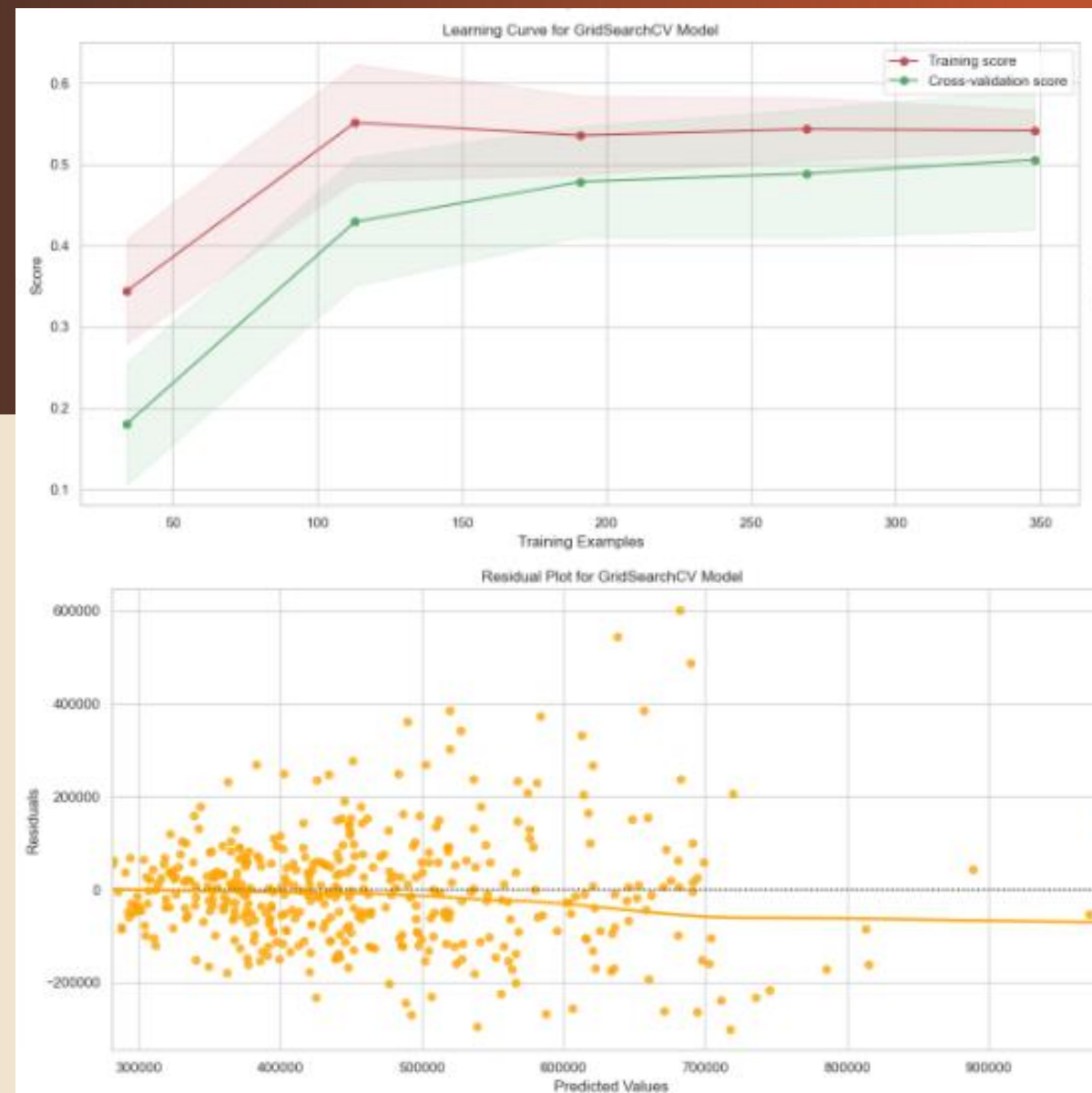
Model	R-squared	Mean Absolute Error (MAE)	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Percentage Error (MAPE)	Explained Variance Score (EVS)
Decision Tree	0.411391	101250.280967	2.009522e+10	141757.625562	0.230072	0.411391
Random Forest	0.556153	88221.799439	1.515301e+10	123097.548725	0.203849	0.556180



Model Selection and Improvement

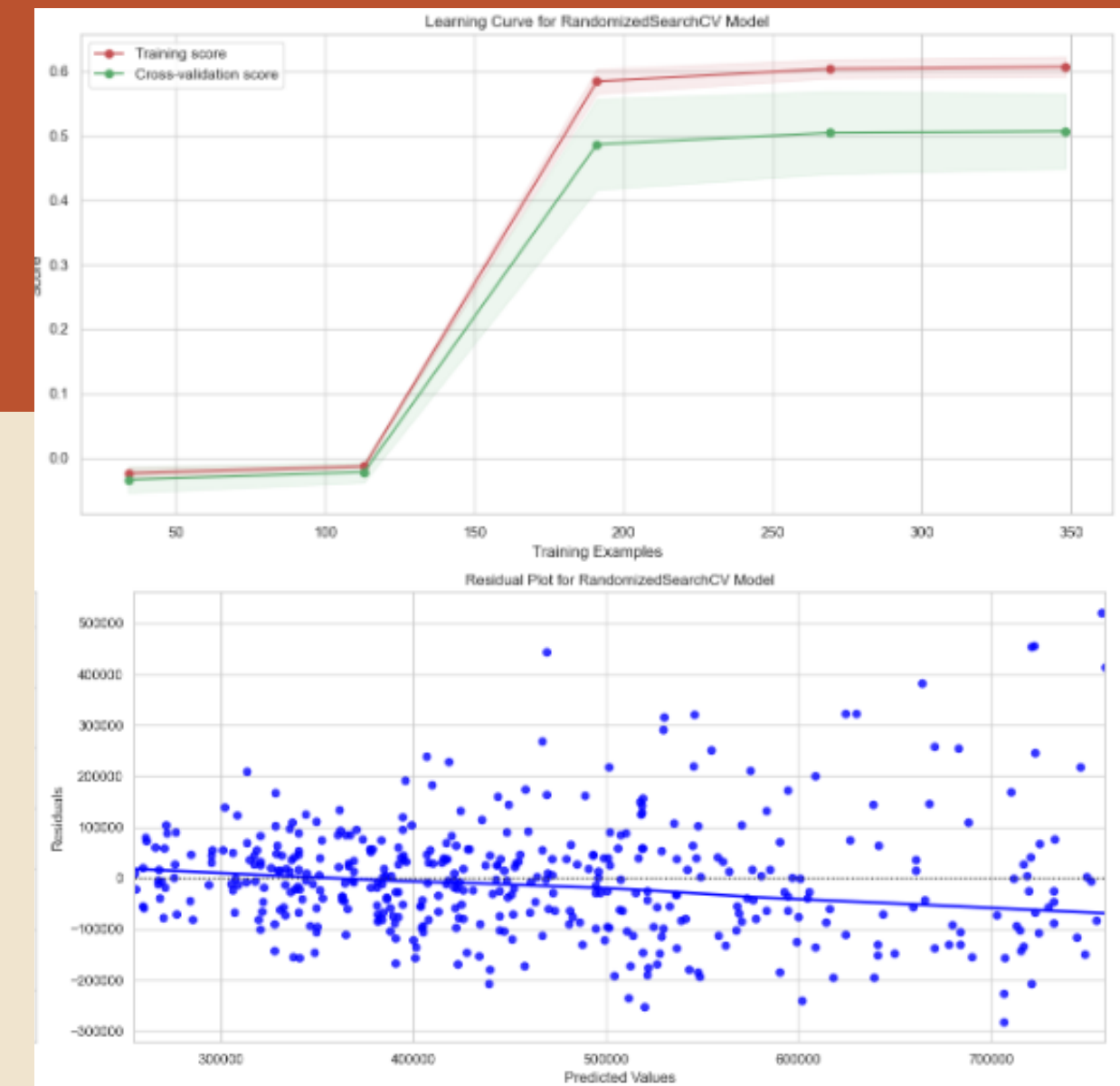


Linear Support Vector Regression



	R-squared	Mean Absolute Error (MAE)	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Percentage Error (MAPE)	Explained Variance Score (EVS)
RandomizedSearchCV	0.082846	128705.471448	3.131182e+10	176951.451201	0.287277	0.110374
GridSearchCV	0.546845	89889.276725	1.547080e+10	124381.683125	0.202647	0.549165

Gradient Booster Regressor



	R-squared	Mean Absolute Error (MAE)	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Percentage Error (MAPE)	Explained Variance Score (EVS)
RandomizedSearchCV	0.611144	81893.121264	1.327561e+10	115219.840686	0.180272	0.613753
GridSearchCV	0.624294	80566.160834	1.282666e+10	113254.841310	0.178274	0.626502

Model Selection and Improvement

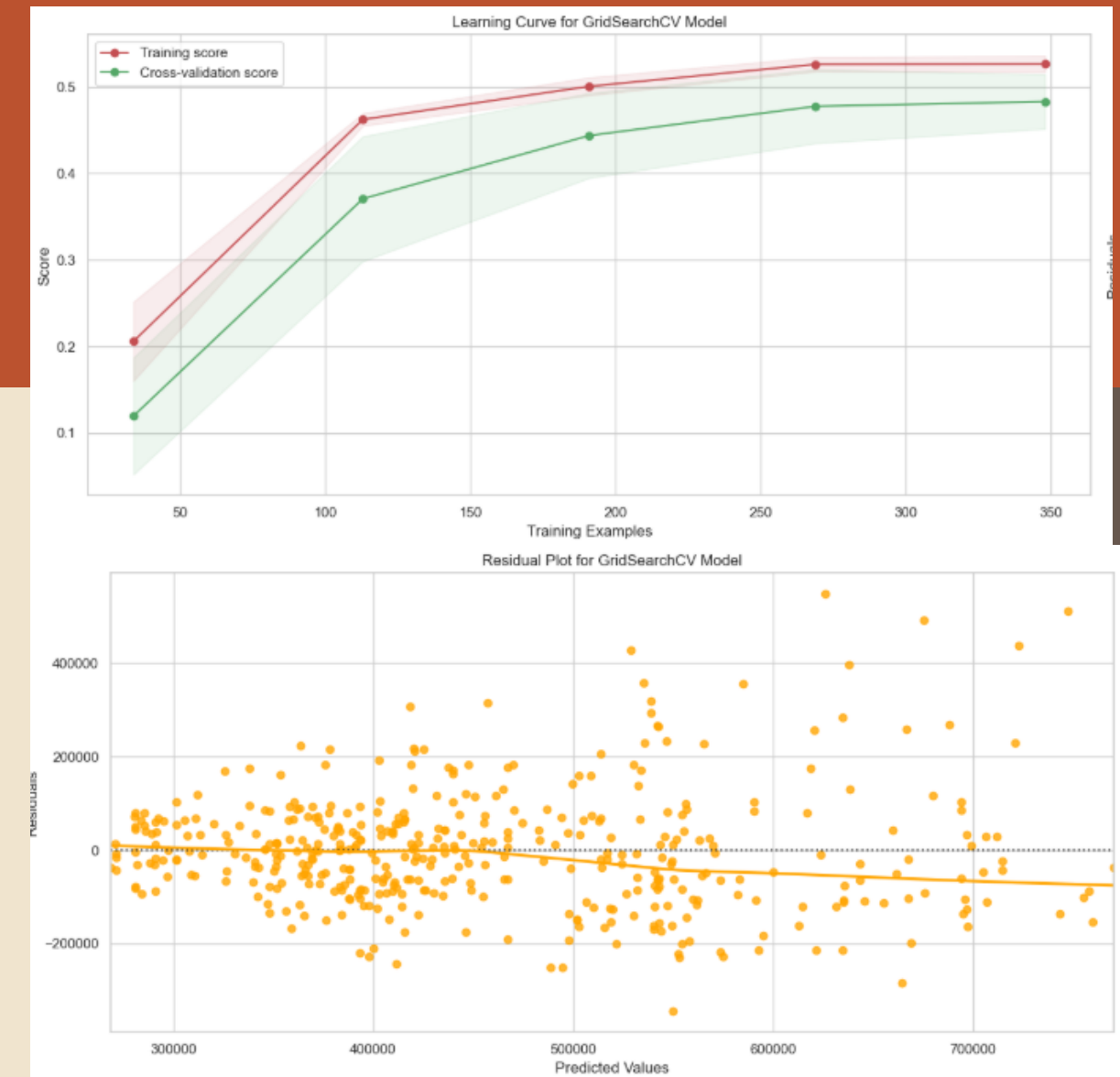


MLP Regressor



	R-squared	Mean Absolute Error (MAE)	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Percentage Error (MAPE)	Explained Variance Score (EVS)
RandomizedSearchCV	0.092542	133629.164866	3.098078e+10	176013.585159	0.318471	0.092542
GridSearchCV	0.572714	90131.349165	1.458762e+10	120779.236669	0.205220	0.572991

K Neighbors Regressor



	R-squared	Mean Absolute Error (MAE)	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Percentage Error (MAPE)	Explained Variance Score (EVS)
RandomizedSearchCV	0.569603	86774.793578	1.469382e+10	121218.076894	0.194526	0.571325
GridSearchCV	0.529603	89991.506116	1.605945e+10	126725.878999	0.198996	0.536777



Voting Regressor

Testing on Training Data (Before Hypertuning)

	R-squared	Mean Absolute Error (MAE)	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Percentage Error (MAPE)	Explained Variance Score (EVS)
VotingRegressor_Train	0.582522	85344.706183	14252765612.012691	119384.947175	0.192798	0.583253

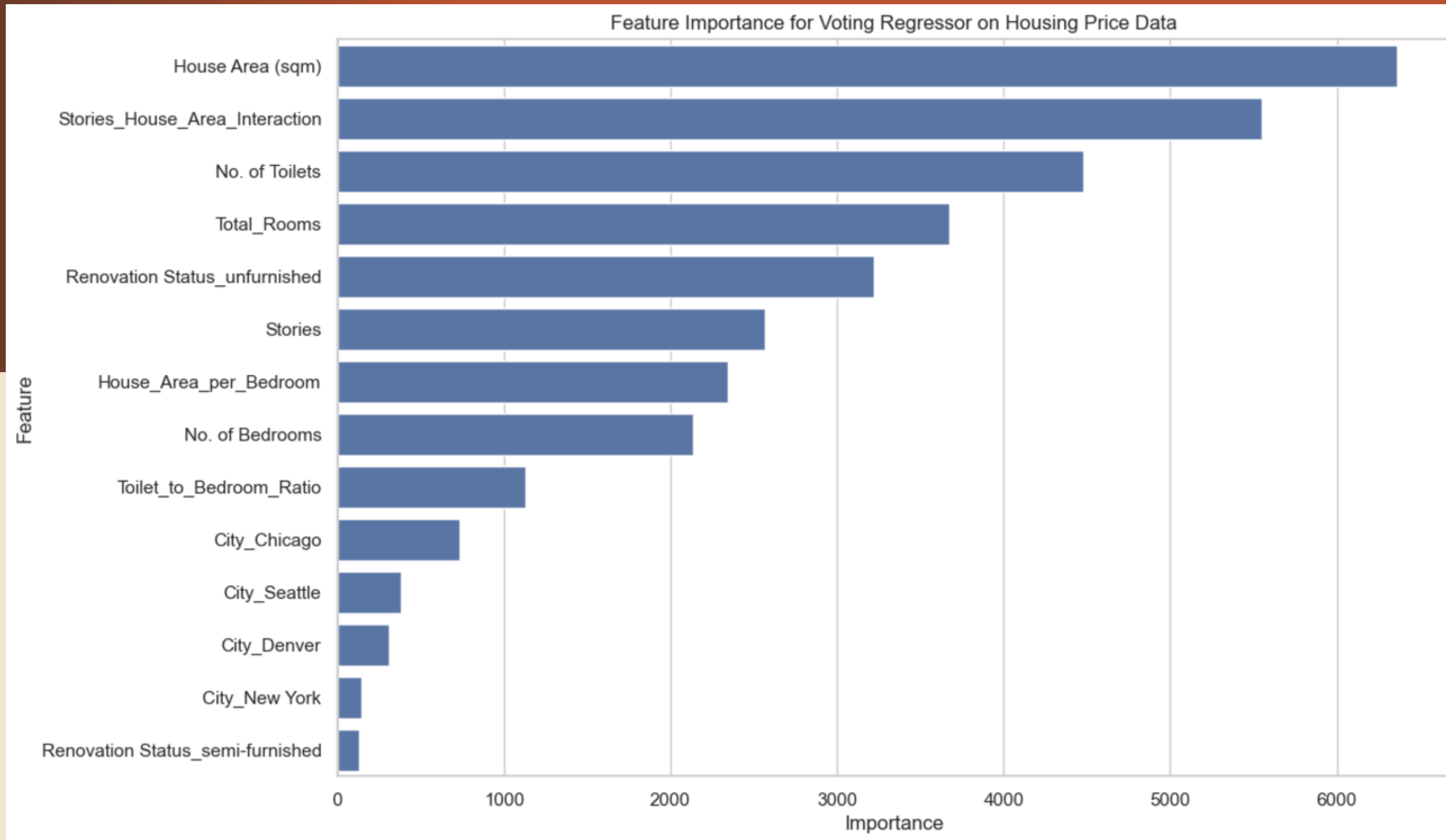
Testing on Training Data (After Hypertuning)

Fitting 15 folds for each of 24 candidates, totalling 360 fits Best Parameters for Voting Regressor: {'votingregressor__weights': [2, 5, 5, 15, 11, 5]}						
	R-squared	Mean Absolute Error (MAE)	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Percentage Error (MAPE)	Explained Variance Score (EVS)
VotingRegressor_Train	0.600378	83610.340856	13643181241.630388	116804.029218	0.187777	0.601163

Testing on Training Data (After Hypertuning)

	R-squared	Mean Absolute Error (MAE)	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Percentage Error (MAPE)	Explained Variance Score (EVS)
VotingRegressor_Test	0.532031	93430.402598	17644182944.199333	132831.407973	0.183723	0.557734

Voting Regressor Feature Importance



Voting Regressor Comparison



Voting Regressor vs Dummy Baseline

	R-squared	Mean Absolute Error (MAE)	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Percentage Error (MAPE)	Explained Variance Score (EVS)
VotingRegressor	0.532031	93430.402598	1.764418e+10	132831.407973	0.183040	5.567822e-01
Baseline	-0.011330	154755.718668	3.813094e+10	195271.442819	0.327846	-2.220446e-16

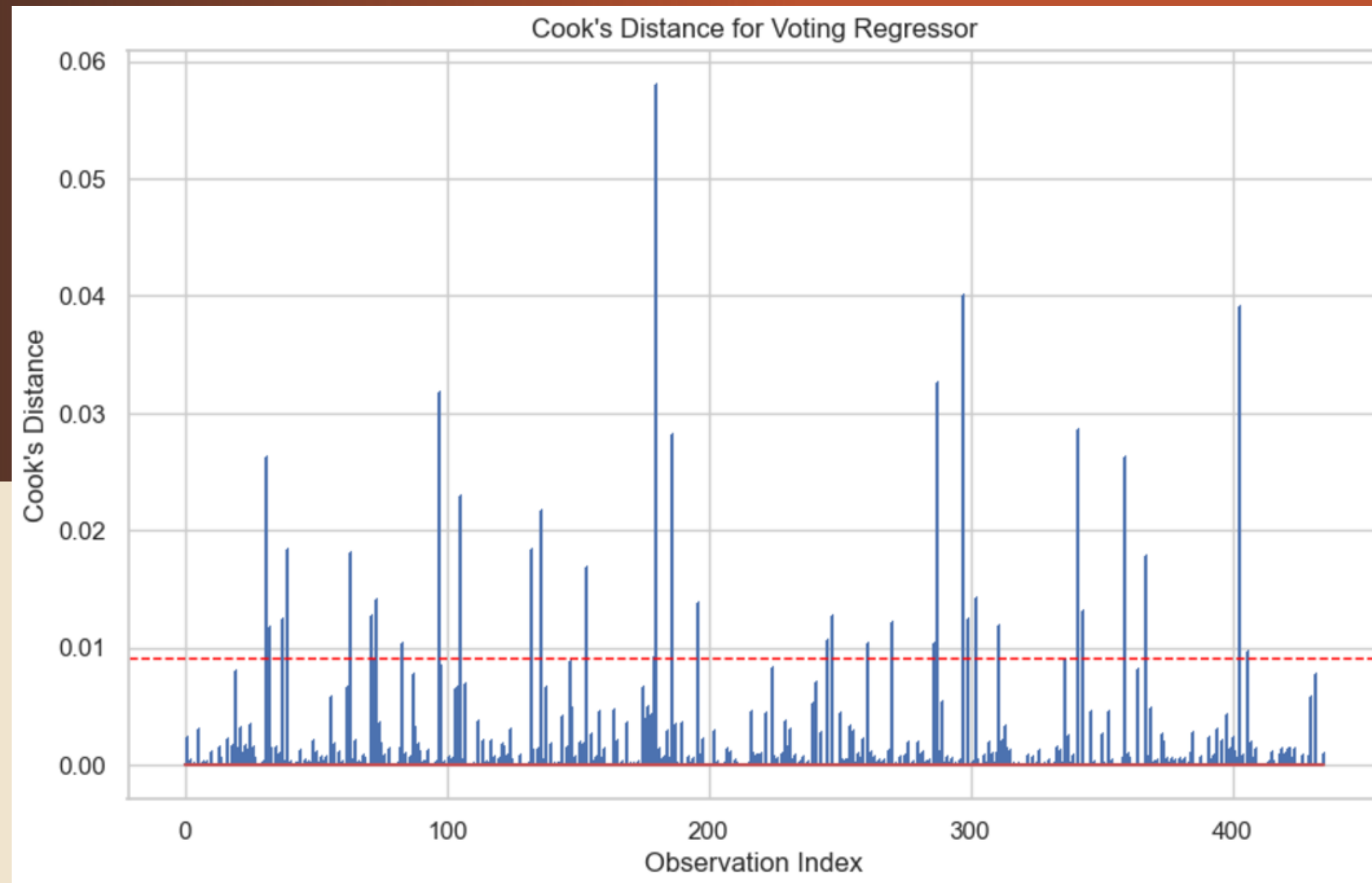
Voting Regressor vs Dummy Baseline

	R-squared	Mean Absolute Error (MAE)	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Percentage Error (MAPE)	Explained Variance Score (EVS)
VotingRegressor	0.532031	93430.402598	1.764418e+10	132831.407973	0.183040	0.556782
LinearRegression	0.514622	90503.663276	1.830056e+10	135279.576626	0.171946	0.538062
LassoRegression	0.514642	90508.993889	1.829981e+10	135276.793828	0.171960	0.538076
RidgeRegression	0.516281	91232.044642	1.823803e+10	135048.250975	0.173803	0.539100

Voting Regressor Influential Points



Cook's Distance Plot



Saving of Final Model

```
# Saving of Voting Regressor
joblib.dump(best_voting_regressor, 'CA1-Saved-Models/final-model-B.pkl')

# Code Execution Confirmation
print("Voting Regressor Model Saved Successfully")

Voting Regressor Model Saved Successfully
```

Scoring Table With Outliers and Without

	R-squared	Mean Absolute Error (MAE)	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Percentage Error (MAPE)	Explained Variance Score (EVS)
With Outliers	0.532031	93430.402598	1.764418e+10	132831.407973	0.18304	0.556782
Without Outliers	0.528145	91975.123340	1.779071e+10	133381.803603	0.17709	0.557462



THANK YOU!