Machine Learning Methods for Predicting the Price of Housing in Various Pakistani Cities

Raafi Rahman

1. Introduction

We will be describing machine learning methods used on a large dataset to analyze the housing market in multiple cities in Pakistan. Pakistan is a very populous country, currently having the fifth-highest population out of all countries. The country has more than 220 million people with a yearly growth rate of 2%. This yearly change is on the higher end. For comparison, the rates of the three most populous countries, China, India, and the United States are 0.39%, 0.99%, and 0.59% respectively. It is clear to see that the housing market in Pakistan is expanding, and will continue to expand. Understanding the trends in this market, and which factors cause those trends, is essential for the Pakistani Government, land developers, real estate agencies, and real estate investors.

The goal of my analysis is to compare the performance of different machine learning models, all predicting the prices of houses based on known factors. We will build different types of models and see how the results compare. We will also look at how different factors impact the price of a house, and which factors, in particular, are the most impactful, and the least impactful. The following machine learning methods will be tested throughout this paper: Multiple Linear Regression, Quadratic Regression, K-Nearest Neighbors, Decision Trees, and Random Forest. We will split our data into disjoint training and testing sets in order to have accurate assessments and reliable results. This will allow us to fairly judge the performance of a specific model. Whenever possible, we will try to employ techniques to improve the quality of our model.

2. Dataset

The dataset we are using was scraped by Huzefa Khan from a popular Pakistani real estate website, Zameen.com, and uploaded to Open Data Pakistan and Kaggle. “Zameen” is an Urdu and Hindi word that translates to “ground” or “land”. The dataset we are using contains almost 170,000 instances of data. Originally the dataset contained 18 columns, some were unnecessary for our purposes. The columns in the original dataset are as follows.

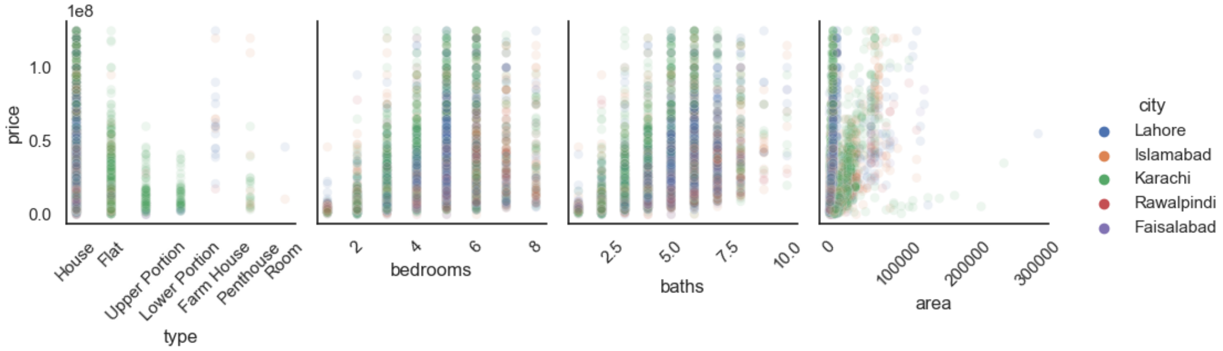
- **INT property_id**: A unique ID given to each house by Zameen.com
- **INT location_id**: An ID given to each plot by Zameen.com
- **STRING page_url**: Link where listing was posted
- **STRING property_type**: Type of property (‘House’, ‘Farm House’, ‘Flat’, ‘Upper Portion’, ‘Lower Portion’, ‘Room’, ‘Penthouse’)
- **INT price**: For Sale or For Rent price in Pakistani Rupees
- **STRING location**: Block/street name
- **STRING city**: City name (‘Karachi’, ‘Lahore’, ‘Islamabad’, ‘Rawalpindi’, ‘Faisalabad’)
- **STRING province_name**: Province/state name (‘Sindh’, ‘Punjab’, ‘Islamabad Capital’)
- **FLOAT latitude**: Latitude coordinate
- **FLOAT longitude**: Longitude coordinate
- **INT baths**: Number of bathrooms
- **STRING purpose**: Purpose of listing (‘For Sale’, ‘For Rent’)
- **INT bedrooms**: Number of bedrooms
- **STRING date_added**: The listing date on Zameen.com
- **STRING agency**: Company selling property

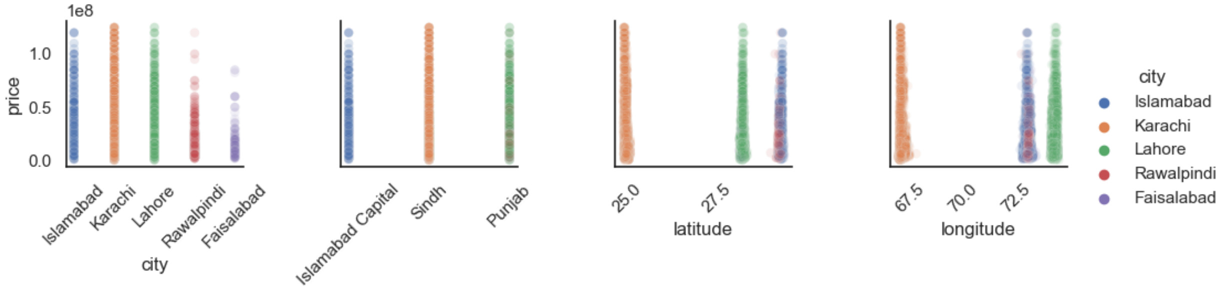
- **STRING agent**: Property dealer
- **FLOAT Total_Area**: Property area in square feet

I dropped some of the columns I found to be unnecessary e,g ‘page_url’, ‘agency’, ‘agent’, ‘date_added’, ‘location’ (since there are too many street names to keep track of. Not enough properties share the same street name, therefore will not help us in our analysis), ‘purpose’ (because we will only be looking at properties that are ‘For Sale’), ‘property_id’, and ‘location_id’. The target column is the ‘price’ of the house given in Pakistani Rupees (PKR). The cleaned dataset contains the following columns.

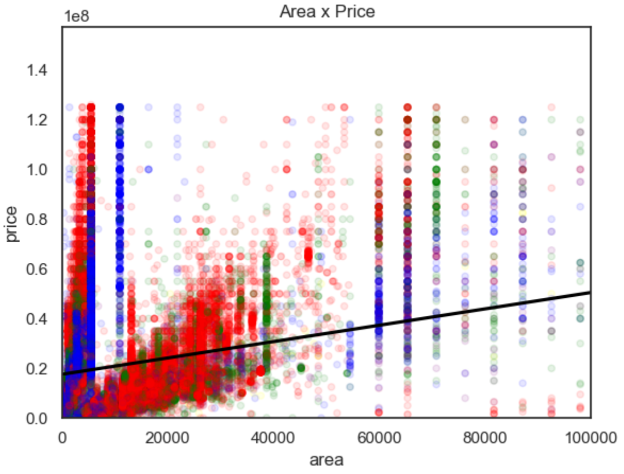
- **STRING type**: Type of property (‘House’, ‘Farm House’, ‘Flat’, ‘Upper Portion’, ‘Lower Portion’, ‘Room’, ‘Penthouse’)
- **INT bedrooms**: Number of bedrooms
- **INT baths**: Number of bathrooms
- **FLOAT area**: Property area in square feet
- **STRING city**: City name (‘Karachi’, ‘Lahore’, ‘Islamabad’, ‘Rawalpindi’, ‘Faisalabad’)
- **STRING province**: Province/state name (‘Sindh’, ‘Punjab’, ‘Islamabad Capital’)
- **FLOAT latitude**: Latitude coordinate
- **FLOAT longitude**: Longitude coordinate
- **INT price**: For sale or for rent price in Pakistani Rupees.

Many datasets are prone to human error, including this one. Human error can be introduced by Zameen.com when entering values or during the process of web scraping. The dataset contains entries that have 0 bedrooms, 0 baths, and 0 square feet. These entries were dropped since they don’t make sense in our context. We also removed a few entries that were outliers. There were many outliers in terms of ‘price’, ‘bedrooms’, ‘bathrooms’, and ‘area’ that would skew our models if not removed. We begin by exploring the overall data in the hopes of finding patterns. Below we have multiple plots of various predictors compared against price. We want to see if there exists any relationship between the price and any of the predictors.

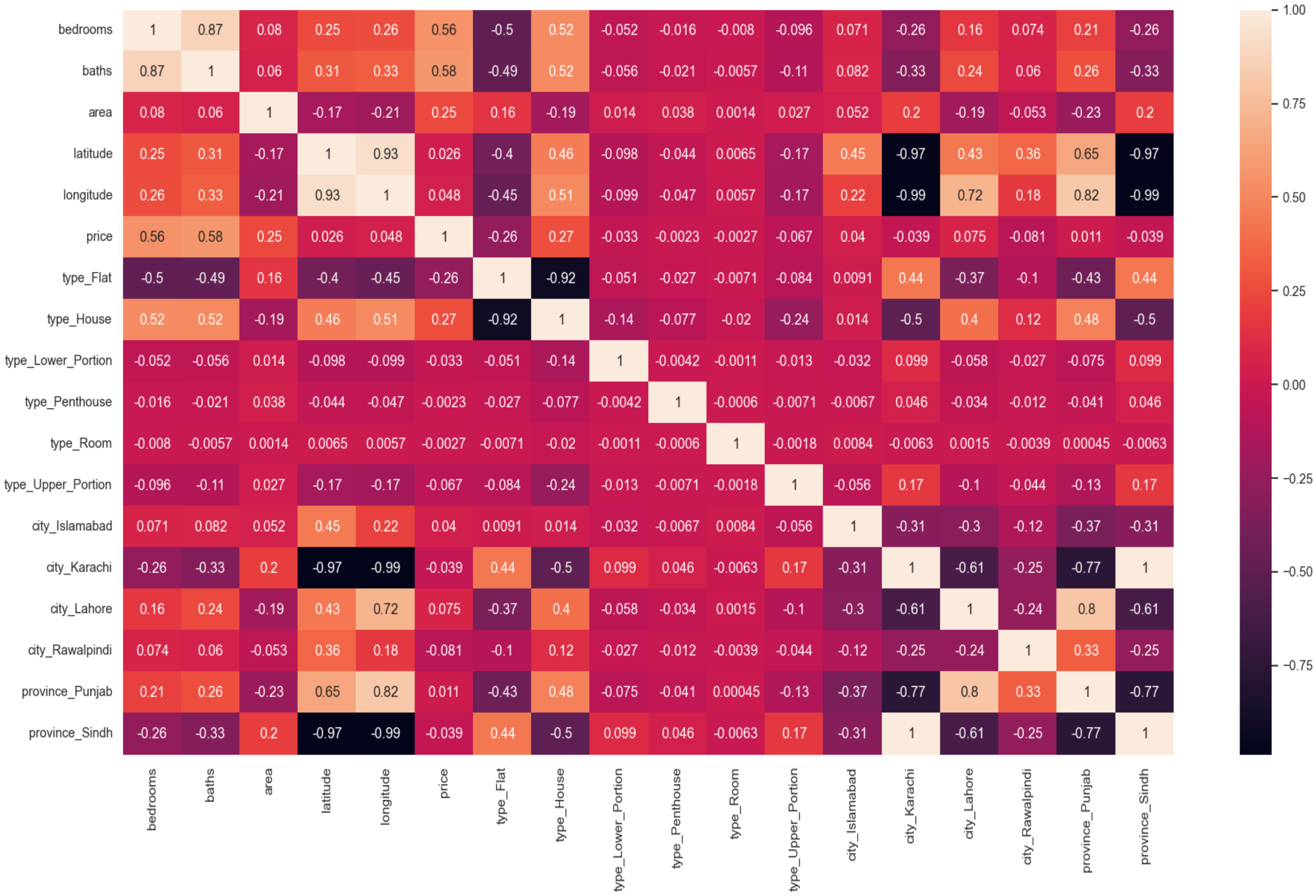




From the above plots, I can see there is a correlation between 'bathrooms' and 'price', 'bedrooms' and 'price', and between 'area' and 'price'. Below is a plot of 'area' compared to 'price', along with a regression line to give us an idea of the trend.



We can see, in general, the price of a property increases as the number of bedrooms and bathrooms increases. After the 6th or 7th bedroom or bathroom, the price begins to plateau. Also a note on the price axis: The Pakistani Rupee has undergone a lot of inflation. The value of the Pakistani Rupee is very low compared to the United States Dollar. The relationship between the two is 1 USD = 179.46 PKR at the of writing. This explains the large scale for the price axis. We can see, especially towards the lower end of the area plot, that the graph "splits" into two directions. This shows us that area, overall, has a relationship with price, but other factors also have an impact. Another thing to notice is the prices of houses in Rawalpindi and Faisalabad are lower than those in Lahore, Karachi, or Islamabad. This is because Lahore, Karachi, and Islamabad are much more industrial cities. After undergoing One Hot Encoding, we come up with the following correlation heat map.



	bedrooms	baths	area	latitude	longitude	price	type_Flat	type_House	type_Lower_Portion	type_Penthouse	type_Room	type_Upper_Portion	city_Islamabad	city_Karachi	city_Lahore	city_Rawalpindi	province_Punjab	province_Sindh
bedrooms	1	0.87	0.08	0.25	0.26	0.56	-0.5	0.52	-0.052	-0.016	-0.008	-0.096	0.071	-0.26	0.16	0.074	0.21	-0.26
baths	0.87	1	0.06	0.31	0.33	0.58	-0.49	0.52	-0.056	-0.021	-0.0057	-0.11	0.082	-0.33	0.24	0.06	0.26	-0.33
area	0.08	0.06	1	-0.17	-0.21	0.25	0.16	-0.19	0.014	0.038	0.0014	0.027	0.052	0.2	-0.19	-0.053	-0.23	0.2
latitude	0.25	0.31	-0.17	1	0.93	0.026	-0.4	0.46	-0.098	-0.044	0.0065	-0.17	0.45	-0.97	0.43	0.36	0.65	-0.97
longitude	0.26	0.33	-0.21	0.93	1	0.048	-0.45	0.51	-0.099	-0.047	0.0057	-0.17	0.22	-0.99	0.72	0.18	0.82	-0.99
price	0.56	0.58	0.25	0.026	0.048	1	-0.26	0.27	-0.033	-0.0023	-0.0027	-0.067	0.04	-0.039	0.075	-0.081	0.011	-0.039
type_Flat	-0.5	-0.49	0.16	-0.4	-0.45	-0.26	1	-0.92	-0.051	-0.027	-0.0071	-0.084	0.0091	0.44	-0.37	-0.1	-0.43	0.44
type_House	0.52	0.52	-0.19	0.46	0.51	0.27	-0.92	1	-0.14	-0.077	-0.02	-0.24	0.014	-0.5	0.4	0.12	0.48	-0.5
type_Lower_Portion	-0.052	-0.056	0.014	-0.098	-0.099	-0.033	-0.051	-0.14	1	-0.0042	-0.0011	-0.013	-0.032	0.099	-0.058	-0.027	-0.075	0.099
type_Penthouse	-0.016	-0.021	0.038	-0.044	-0.047	-0.0023	-0.027	-0.077	-0.0042	1	-0.0006	-0.0071	-0.0067	0.046	-0.034	-0.012	-0.041	0.046
type_Room	-0.008	-0.0057	0.0014	0.0065	0.0057	-0.0027	-0.0071	-0.02	-0.0011	-0.0006	1	-0.0018	0.0084	-0.0063	0.0015	-0.0039	0.00045	-0.0063
type_Upper_Portion	-0.096	-0.11	0.027	-0.17	-0.17	-0.067	-0.084	-0.24	-0.013	-0.0071	-0.0018	1	-0.056	0.17	-0.1	-0.044	-0.13	0.17
city_Islamabad	0.071	0.082	0.052	0.45	0.22	0.04	0.0091	0.014	-0.032	-0.0067	0.0084	-0.056	1	-0.31	-0.3	-0.12	-0.37	-0.31
city_Karachi	-0.26	-0.33	0.2	-0.97	-0.99	-0.039	0.44	-0.5	0.099	0.046	-0.0063	0.17	-0.31	1	-0.61	-0.25	-0.77	1
city_Lahore	0.16	0.24	-0.19	0.43	0.72	0.075	-0.37	0.4	-0.058	-0.034	0.0015	-0.1	-0.3	-0.61	1	-0.24	0.8	-0.61
city_Rawalpindi	0.074	0.06	-0.053	0.36	0.18	-0.081	-0.1	0.12	-0.027	-0.012	-0.0039	-0.044	-0.12	-0.25	-0.24	1	0.33	-0.25
province_Punjab	0.21	0.26	-0.23	0.65	0.82	0.011	-0.43	0.48	-0.075	-0.041	0.00045	-0.13	-0.37	-0.77	0.8	0.33	1	-0.77
province_Sindh	-0.26	-0.33	0.2	-0.97	-0.99	-0.039	0.44	-0.5	0.099	0.046	-0.0063	0.17	-0.31	1	-0.61	-0.25	-0.77	1

3. Analysis

Many different models can be used in regression situations. We will be testing We start by splitting our data into disjoint training and testing sets. This is to ensure we are getting good results. I reserve 20% of the data for testing.

3.1 Multiple Linear Regression

We begin by using Multiple Linear Regression to predict the prices on various properties. Linear Regression is simple to implement and is not as complex as other models. Linear Regression can be impacted by outliers, which is why we removed outliers from our dataset. Using Multiple Linear Regression, “sklearn.linear_model.LinearRegression()”, our model achieves an R-squared score of 0.409 with an intercept of 133230926.52 and coefficients of

[2.65e+06, 4.87e+06 2.51e+02, -3.75e+06, -2.93e+05, -1.16e+07, -8.42e+06, -1.34e+07, -1.19e+07, -7.09e+06, -1.44e+07, 1.20e+07, -8.83e+06, 7.17e+06, 8.26e+06, -3.14e+06, -8.83e+06].

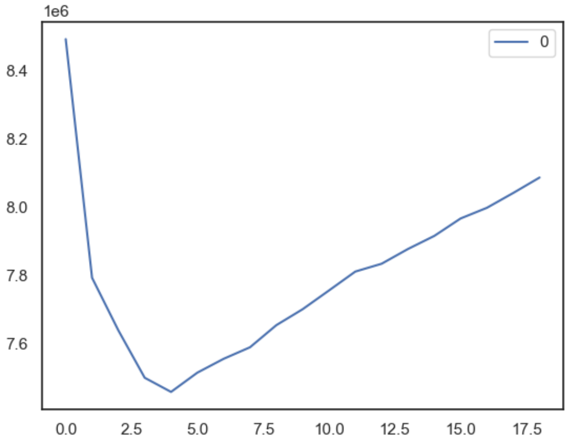
Next, we try to make a more complex model with higher degrees of freedom in the hopes that our model will fit the data better. First, I take the first three principal components of the data set. This is done using “sklearn.decomposition.PCA(n_components=3)”. Then we fit a Quadratic Regression model to it. A Quadratic Regression model is a Polynomial Regression model with degree = 2. This is done by calling “sklearn.preprocessing.PolynomialFeatures(degree = 2)”. The results are slightly better. We get an R-squared score of 0.420. Our intercept is 20266031.50 and our coefficients are

[0, 4.05e+02, 5.75e+05, -5.77e+06, -1.12e-03, 1.43e+01, -1.77e+01, 5.33e+03, 2.08e+05, 6.65e+04].

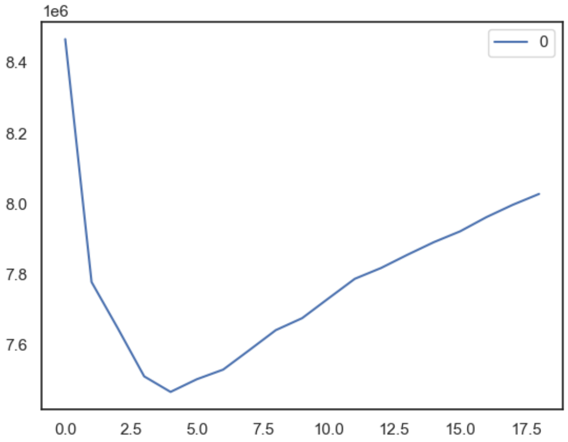
3.2 K-Nearest Neighbors

Next, we use K-Nearest Neighbors. K-Nearest Neighbors is another simple model that is easy to implement. One benefit this model has over Linear Regression is that K-Nearest Neighbors is non-parametric, meaning it has no assumption about the data. Linear Regression assumes the predictors have a linear relationship to the target. This model is also sensitive to outliers, but we don't need to worry about that. We start with default settings by calling "KNeighborsRegressor()". Our k = 5 (by default) and our distance metric is the "Minkowski distance" (by default). Using K-nearest neighbors, we already see better results compared to Multiple Linear Regression. With no alterations made, we achieve an R-squared score of 0.859.

We want to see if we can improve this model. Next, we will iterate over a range of k and use both the "Minkowski distance" and the "Manhattan distance". I created a KNN loop that tests all integers from 1 to 20. We did not test any higher k values, revealing one of the flaws of K-Nearest Neighbors, it is very computationally expensive. For the "Minkowski distance", we find that a k = 4 or 5 is ideal, giving us a similar R-squared score as above. For the "Manhattan distance" we find the same result. Changing the distance metric did not impact our model. Below we have elbow graphs charting k to RMSE.



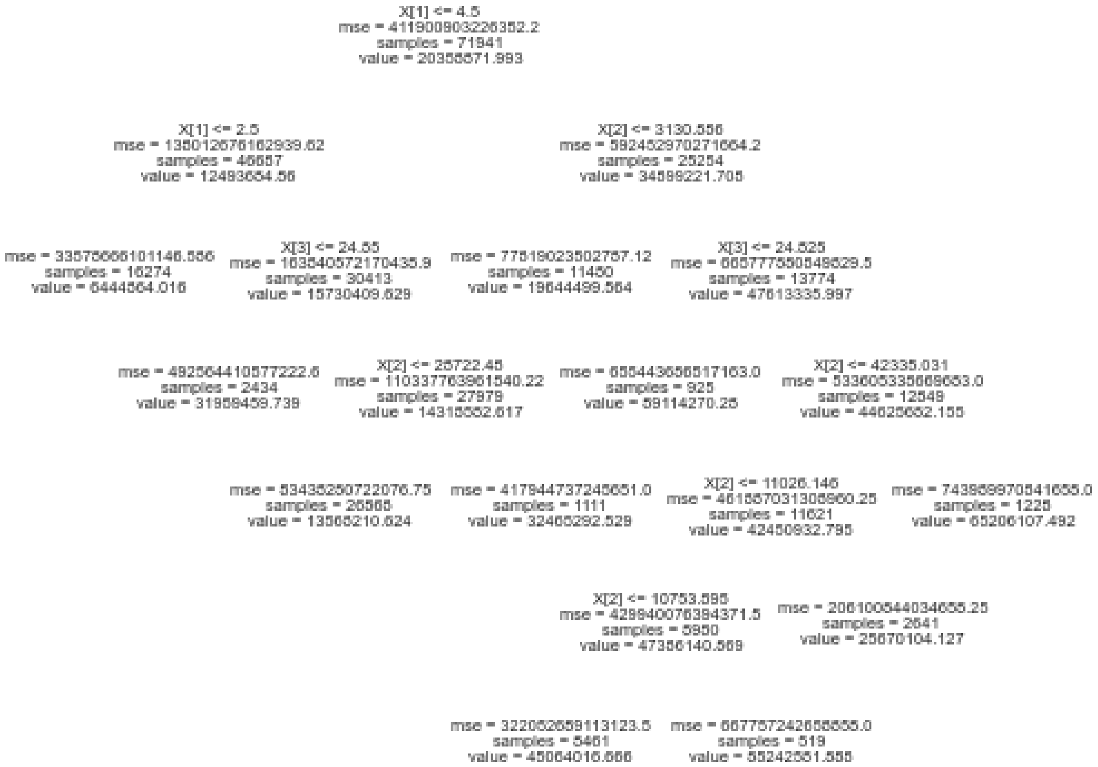
Minkowski Distance



Manhattan Distance

3.3 Decision Trees

Next, we will discuss Decision Trees. Decision Trees are not as easily influenced by outliers as Linear Regression of K-Nearest Neighbors. Like K-Nearest Neighbors, Decision Trees are non-parametric and will not make assumptions about the data. First, we start with a standard Decision Tree using “DecisionTreeRegressor()”. Without setting a ‘max_leaf_node’, we allow the method to run indefinitely. This can cause overfitting. With our current model, we achieve an R-squared score of 0.87. Even with overfitting, this model outperforms K-Nearest Neighbors. Creating a tree with the depth we used above would take too long to compile, so below is a sample tree in the case ‘max_leaf_node=10’.



Our model can still be improved. Using Random Forest, we can create an even more accurate model. Random Forest, in general, performs better since this method creates multiple Decision Trees and then takes the majority vote from all of them. By calling "sklearn.ensemble.RandomForestRegressor()", we can create our model. Running this model gives us an R-squared score of 0.91. This is the highest score we have achieved out of all the methods we have tried.

4. Conclusion

Throughout this paper, we used various types of models to assess the Pakistani house price dataset. From our pre-analysis steps, we found that the number of bedrooms and the number of bathrooms had a positive correlation to price. We also found that the property area was positively correlated to price. After experimenting with many different models, we found that Linear Regression was not well suited for this dataset. Linear models assumed the predictors had a linear correlation with the target. It turns out that this was not the case. Simple Linear Regression achieved the lowest performance out of all the methods we tried. Quadratic Regression slightly increased performance, but not by much. Using K-Nearest Neighbors Regression, we had a significant increase in accuracy. Trying different values for k, and trying different metrics, we conclude that k = 4 or 5 yields the best result. Lastly, we used decision trees to make predictions. Using a simple decision tree we again saw an increase in accuracy. Random Forest achieved the highest accuracy out of all of our models with an R-squared score of 0.91. All this being said, we now understand the main factors that determine house prices in Pakistan, and how we can accurately predict them.