Predicting Housing Prices In Metropolitan Areas of India

Nihar Madkaiker

Contents

Introduction **3**

The Housing Market in India . . . 3
The data set . . . 3
The approach . . . 3
Methodology 1 : Linear Regression . . . 3
Ingestion . . . 4
Adding City Column . . . 8
Combining the data sets . . . 8
Identifying invalid entries . . . 9
Adding the PPA variable . . . 13
Log transforming the data . . . 14
Cleaning the data . . . 15
Identifying and Removing Outliers . . . 15
Partitioning the data . . . 18
1. Price Variation by city . . . 23
Exploratory Analysis . . . 24
Linear regression models . . . 28
LM: Model 1 : cities . . . 28
LM: Model 2: Cities + Bedrooms . . . 29
LM: Model 3: Cities + Bedrooms + Resale . . . 31
LM: Model 4: Cities + Bedrooms + Resale + Furniture . . . 32
LM: Model 5: Cities + Bedrooms + Resale + furniture + feature . . . 33
Validation . . . 35

References **35**

#Executive Summary

In this project we run a linear regression model on a data set to understand the working of the housing market in metropolitan cities in india and predict prices. I first identified the inter-correlated factors and subsequently created models by combining multiple factors. I improved the Rsquared value as much as possible, and obtained the RMSE value by running the model on a validation set.

Introduction

This report is a part of the Choose your own project in the course, Harvard X: PH125.9x Data Science: Capstone.

In this project I have tried to develop a model to predict the housing prices in metropolitan areas of India. I have looked at the cities of Bangalore, Chennai, Delhi, Hyderabad, Kolkata and Mumbai in India. In the analysis I have tried to evaluate how the prices are affected by various parameters such as, area, no. of rooms, amenities, location, state etc. The factors for consideration have been evolved through the preliminary analysis in the project.

The data set used for the analysis is obtained from Kaggle, and is authored by Ruchi Bhatia. The data set can be downloaded from here; https://www.kaggle.com/ruchi798/housing-prices-in-metropolitan-areas-of-india

The Housing Market in India

The housing market sees a clear divide in India, among rural and urban settings, with the Metropolitan cities being among the top markets. India has 6 prominent metropolitan areas Bangalore, Chennai, Delhi, Hyderabad, Kolkata and Mumbai. Each of the cities has its own dynamics. The price of housing in the cities see variation from city to city, which are driven by multiple factors.

One of the factors for consideration is the cities itself. The cities due to their economic activities, business and job opportunities have varying prices. The other factors that affect the housing prices are the area within the city, this often stems from access to schools, hospitals, office space etc.

The floor space area of the property is anyway a direct consideration for the property.

The other amenities often in consideration in the Indian housing market are, access to gymnasium, maintenance staff, swimming pools, open spaces/ garden area etc.

The level of transaction also comes into consideration, if the house has changed hands/ is a resale property the housing price would change.

Basis all of these factors the housing prize would be determined.

The data set

The data set in consideration is a combination of multiple CSV files for each of the 6 cities. The data sets have columns with the price, area, no. of rooms, amenities, location etc. The data set would be imported as a individual csv files. The CSV files would then be combined and cummulatively addressed. The individual data sets are missing the city id, which would be added in the data preparation stage.

The data sets are stored on the project repository on git hub. - https://github.com/niharonline/Housing-Prices_CYOP_Nihar

The individual file links are available in the data preparation section of the report.

The approach

Methodology 1 : Linear Regression

The methodology would be to run linear regression. We will first identify correlation among all the different factors. Based on which factors and coefficients will be identified. We will then create an equation to predict the pricing and run the equation on the data to cross validate and then compare with the validation set.

The first step would be to prepare the data, combining the multiple data set. Since the price is usually a multiple of area, the factor in consideration would the price per unit area. Hence a variable would would be

added, PPA (price per area). This would be our dependent variable for which we would conduct the analysis and this would be predicted. We will add a column to our data set, and the PPA would be calculated by dividing the Price by area.

The data set would be first divided into a training set and test set (validation set). The validation set would be a hold out set and would be tested only as the last step of the project.

We would then conduct exploratory analysis to see broad trends in the data at hand, based on which will decide what factors to add to our model.

We will first take the naive approach, and add factors onto the model one factor at a time, such that the RMSE continues to approach. We will add the further factor based on inferences drawn from looking at correlation between the factors.

We will try a regularization / ML approach by varying parameters, to arrive at the best RMSE.

The final model will be then tested on the validation data set that has been held out till the end.

#Data Ingestion and Data Preparation

Ingestion

We will import the data from the git hub repo, using the individual links for the various data sets.

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: caret
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: data.table
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
## The following object is masked from 'package:purrr':
##
##     transpose
```

```
install.packages("naniar", repos="http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/nihar/Documents/R/win-library/4.1'
## (as 'lib' is unspecified)
```

```
## package 'naniar' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\nihar\AppData\Local\Temp\RtmpKeIUqu\downloaded_packages
```

```
install.packages("corrplot", repos="http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/nihar/Documents/R/win-library/4.1'
## (as 'lib' is unspecified)
```

```
## package 'corrplot' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\nihar\AppData\Local\Temp\RtmpKeIUqu\downloaded_packages
```

```
library(tidyverse)
library(caret)
library(data.table)
library(dplyr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(ggplot2)
library(readr)
library(naniar)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

1. Bangalore

The data for Bangalore is imported and saved in dataframe named Bangalore. The link for the file is, https://raw.githubusercontent.com/niharonline/Housing-Prices_CYOP_Nihar/main/Bangalore.csv

```
##Importing Bangalore Data set

urlbangalore = "https://raw.githubusercontent.com/niharonline/Housing-Prices_CYOP_Nihar/main/Bangalore.
Bangalore<-read_csv(url(urlbangalore))
```

```
## Rows: 6207 Columns: 40
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Location
## dbl (39): Price, Area, No. of Bedrooms, Resale, MaintenanceStaff, Gymnasium,...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

2. Chennai

The data for Chennai is imported and saved in dataframe named Chennai. The link for the file is, https://raw.githubusercontent.com/niharonline/Housing-Prices_CYOP_Nihar/main/Chennai.csv

```
##Importing Chennai Data set

urlchennai = "https://raw.githubusercontent.com/niharonline/Housing-Prices_CYOP_Nihar/main/Chennai.csv"
Chennai<-read_csv(url(urlchennai))
```

```
## Rows: 5014 Columns: 40
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Location
## dbl (39): Price, Area, No. of Bedrooms, Resale, MaintenanceStaff, Gymnasium,...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

3. Importing for other cities, Similarly importing the data set for other cities

```
## similarly importing the data set for other cities
urldelhi = "https://raw.githubusercontent.com/niharonline/Housing-Prices_CYOP_Nihar/main/Delhi.csv"

urlhyderabad = "https://raw.githubusercontent.com/niharonline/Housing-Prices_CYOP_Nihar/main/Hyderabad.

urlkolkata = "https://raw.githubusercontent.com/niharonline/Housing-Prices_CYOP_Nihar/main/Kolkata.csv"

urlmumbai = "https://raw.githubusercontent.com/niharonline/Housing-Prices_CYOP_Nihar/main/Mumbai.csv"

Delhi<-read_csv(url(urldelhi))
```

```
## Rows: 4998 Columns: 40

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Location
## dbl (39): Price, Area, No. of Bedrooms, Resale, MaintenanceStaff, Gymnasium,...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Hyderabad<-read_csv(url(urlhyderabad))
```

```
## Rows: 2518 Columns: 40

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Location
## dbl (39): Price, Area, No. of Bedrooms, Resale, MaintenanceStaff, Gymnasium,...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Kolkata<-read_csv(url(urlkolkata))
```

```
## Rows: 6507 Columns: 40

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Location
## dbl (39): Price, Area, No. of Bedrooms, Resale, MaintenanceStaff, Gymnasium,...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Mumbai<-read_csv(url(urlmumbai))
```

```
## Rows: 7719 Columns: 40
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Location
## dbl (39): Price, Area, No. of Bedrooms, Resale, MaintenanceStaff, Gymnasium,...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Adding City Column

We now have 6 data frames with each for each of the factors. we will first add the city variable to each of the data sets.

```
#adding respective cities to all the data sets.
# we will create a column city which will mention the city for the respective entry.

Bangalore$city <- "Bangalore"
Chennai$city <- "Chennai"
Delhi$city <- "Delhi"
Hyderabad$city <- "Hyderabad"
Kolkata$city <- "Kolkata"
Mumbai$city <- "Mumbai"
```

Combining the data sets

We now have 6 different data sets which we have to combine in one single data set, which will be the parent data set for running any form of analysis.

```
#combining the six data sets.

Housing_data <- rbind(Bangalore, Chennai, Delhi, Hyderabad, Kolkata, Mumbai)
head(Housing_data)
```

```
## # A tibble: 6 x 41
##      Price  Area Location     `No. of Bedroom~ Resale MaintenanceStaff Gymnasium
##      <dbl> <dbl> <chr>                   <dbl>  <dbl>            <dbl>     <dbl>
## 1 30000000  3340 JP Nagar Ph~                4      0                1         1
## 2  7888000  1045 Dasarahalli~                2      0                0         1
## 3  4866000  1179 Kannur on T~                2      0                0         1
## 4  8358000  1675 Doddanekundi                3      0                0         0
## 5  6845000  1670 Kengeri                     3      0                1         1
```

```
## 6  6797000  1220 Horamavu                     2      0                0         1
## # ... with 34 more variables: SwimmingPool <dbl>, LandscapedGardens <dbl>,
## #   JoggingTrack <dbl>, RainWaterHarvesting <dbl>, IndoorGames <dbl>,
## #   ShoppingMall <dbl>, Intercom <dbl>, SportsFacility <dbl>, ATM <dbl>,
## #   ClubHouse <dbl>, School <dbl>, 24X7Security <dbl>, PowerBackup <dbl>,
## #   CarParking <dbl>, StaffQuarter <dbl>, Cafeteria <dbl>,
## #   MultipurposeRoom <dbl>, Hospital <dbl>, WashingMachine <dbl>,
## #   Gasconnection <dbl>, AC <dbl>, Wifi <dbl>, Children'splayarea <dbl>, ...
```

This gives us a data set with all the city data combined in one data frame with 32963 observations across the 41 variables. We also note that many factors have a max value of 9, the value actually goes from 0 to 1 for these, value 9 is added in case of N/A or missing entries.

##Data Preparation

As observed above, a large number of missing entries are present, We will create a parallel data set with the value 9 replaced with N/A and the N/A entries will be ommitted we will call this data set as clean. (this data set is extremely limited and hence will be used only for one set of analysis, the other analysis will proceed with the larger data set). The data set might have a large number of outliers as well. There are chances that the data observations are also not in the right type.

Identifying invalid entries

```
## here we replace the entries with value 9 with N/A

Housing_data <- Housing_data %>% replace_with_na_all(condition = ~.x == 9)
summary(Housing_data)
```

```
##      Price                Area          Location         No. of Bedrooms
##  Min.   :  2000000   Min.   :  200   Length:32963       Min.   :1.000
##  1st Qu.:  4071500   1st Qu.:  853   Class :character   1st Qu.:2.000
##  Median :  6711000   Median : 1125   Mode  :character   Median :2.000
##  Mean   : 11686718   Mean   : 1293                      Mean   :2.412
##  3rd Qu.: 12000000   3rd Qu.: 1500                      3rd Qu.:3.000
##  Max.   :854599999   Max.   :16000                      Max.   :8.000
##                                                         NA's   :1
##      Resale       MaintenanceStaff   Gymnasium      SwimmingPool
##  Min.   :0.0000   Min.   :0.00     Min.   :0.000   Min.   :0.000
##  1st Qu.:0.0000   1st Qu.:0.00     1st Qu.:0.000   1st Qu.:0.000
##  Median :0.0000   Median :0.00     Median :1.000   Median :0.000
##  Mean   :0.3814   Mean   :0.17     Mean   :0.521   Mean   :0.415
##  3rd Qu.:1.0000   3rd Qu.:0.00     3rd Qu.:1.000   3rd Qu.:1.000
##  Max.   :1.0000   Max.   :1.00     Max.   :1.000   Max.   :1.000
##                   NA's   :22870    NA's   :22870   NA's   :22870
##  LandscapedGardens  JoggingTrack   RainWaterHarvesting  IndoorGames
##  Min.   :0.000     Min.   :0.000   Min.   :0.000       Min.   :0.000
##  1st Qu.:0.000     1st Qu.:0.000   1st Qu.:0.000       1st Qu.:0.000
##  Median :0.000     Median :0.000   Median :0.000       Median :0.000
##  Mean   :0.349     Mean   :0.333   Mean   :0.368       Mean   :0.274
##  3rd Qu.:1.000     3rd Qu.:1.000   3rd Qu.:1.000       3rd Qu.:1.000
##  Max.   :1.000     Max.   :1.000   Max.   :1.000       Max.   :1.000
##  NA's   :22870     NA's   :22870   NA's   :22870       NA's   :22870
##   ShoppingMall      Intercom     SportsFacility       ATM
```

```
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000
##  Median :0.000   Median :0.000   Median :0.000   Median :0.000
##  Mean   :0.075   Mean   :0.452   Mean   :0.275   Mean   :0.101
##  3rd Qu.:0.000   3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:0.000
##  Max.   :1.000   Max.   :1.000   Max.   :1.000   Max.   :1.000
##  NA's   :22870   NA's   :22870   NA's   :22870   NA's   :22870
##    ClubHouse         School       24X7Security    PowerBackup
##  Min.   :0.00    Min.   :0.000   Min.   :0.00    Min.   :0.000
##  1st Qu.:0.00    1st Qu.:0.000   1st Qu.:0.00    1st Qu.:0.000
##  Median :0.00    Median :0.000   Median :0.00    Median :1.000
##  Mean   :0.42    Mean   :0.073   Mean   :0.45    Mean   :0.653
##  3rd Qu.:1.00    3rd Qu.:0.000   3rd Qu.:1.00    3rd Qu.:1.000
##  Max.   :1.00    Max.   :1.000   Max.   :1.00    Max.   :1.000
##  NA's   :22870   NA's   :22870   NA's   :22870   NA's   :22870
##    CarParking     StaffQuarter     Cafeteria     MultipurposeRoom
##  Min.   :0.000   Min.   :0.00    Min.   :0.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.00    1st Qu.:0.000   1st Qu.:0.000
##  Median :0.000   Median :0.00    Median :0.000   Median :0.000
##  Mean   :0.465   Mean   :0.13    Mean   :0.117   Mean   :0.242
##  3rd Qu.:1.000   3rd Qu.:0.00    3rd Qu.:0.000   3rd Qu.:0.000
##  Max.   :1.000   Max.   :1.00    Max.   :1.000   Max.   :1.000
##  NA's   :22870   NA's   :22870   NA's   :22870   NA's   :22870
##     Hospital     WashingMachine  Gasconnection         AC
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000
##  Median :0.000   Median :0.000   Median :0.000   Median :0.000
##  Mean   :0.057   Mean   :0.043   Mean   :0.175   Mean   :0.062
##  3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:0.000
##  Max.   :1.000   Max.   :1.000   Max.   :1.000   Max.   :1.000
##  NA's   :22870   NA's   :22870   NA's   :22870   NA's   :22870
##       Wifi       Children'splayarea LiftAvailable        BED
##  Min.   :0.00    Min.   :0.000      Min.   :0.000   Min.   :0.000
##  1st Qu.:0.00    1st Qu.:0.000      1st Qu.:0.000   1st Qu.:0.000
##  Median :0.00    Median :1.000      Median :1.000   Median :0.000
##  Mean   :0.02    Mean   :0.513      Mean   :0.717   Mean   :0.091
##  3rd Qu.:0.00    3rd Qu.:1.000      3rd Qu.:1.000   3rd Qu.:0.000
##  Max.   :1.00    Max.   :1.000      Max.   :1.000   Max.   :1.000
##  NA's   :22870   NA's   :22870      NA's   :22870   NA's   :22870
##  VaastuCompliant   Microwave       GolfCourse          TV
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000
##  Median :0.000   Median :0.000   Median :0.000   Median :0.000
##  Mean   :0.298   Mean   :0.049   Mean   :0.022   Mean   :0.056
##  3rd Qu.:1.000   3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:0.000
##  Max.   :1.000   Max.   :1.000   Max.   :1.000   Max.   :1.000
##  NA's   :22870   NA's   :22870   NA's   :22870   NA's   :22870
##   DiningTable         Sofa          Wardrobe      Refrigerator
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000
##  Median :0.000   Median :0.000   Median :0.000   Median :0.000
##  Mean   :0.053   Mean   :0.051   Mean   :0.021   Mean   :0.051
##  3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:0.000
##  Max.   :1.000   Max.   :1.000   Max.   :1.000   Max.   :1.000
```

```
##  NA's   :22870   NA's   :22870   NA's   :22870   NA's   :22870
##      city
##  Length:32963
##  Class :character
##  Mode  :character
##
##
##
##
```

```
str(Housing_data)
```

```
## tibble [32,963 x 41] (S3: tbl_df/tbl/data.frame)
##  $ Price              : num [1:32963] 30000000 7888000 4866000 8358000 6845000 ...
##  $ Area               : num [1:32963] 3340 1045 1179 1675 1670 ...
##  $ Location           : chr [1:32963] "JP Nagar Phase 1" "Dasarahalli on Tumkur Road" "Kannur on Tha
##  $ No. of Bedrooms    : num [1:32963] 4 2 2 3 3 2 4 3 3 1 ...
##  $ Resale             : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ MaintenanceStaff   : num [1:32963] 1 0 0 0 1 0 0 0 0 1 ...
##  $ Gymnasium          : num [1:32963] 1 1 1 0 1 1 1 1 1 1 ...
##  $ SwimmingPool       : num [1:32963] 1 1 1 0 1 1 1 0 1 1 ...
##  $ LandscapedGardens  : num [1:32963] 1 1 1 0 1 1 1 0 1 1 ...
##  $ JoggingTrack       : num [1:32963] 1 1 1 0 1 1 1 1 1 1 ...
##  $ RainWaterHarvesting: num [1:32963] 1 1 1 0 1 1 0 1 1 1 ...
##  $ IndoorGames        : num [1:32963] 1 1 0 0 1 1 0 0 0 1 ...
##  $ ShoppingMall       : num [1:32963] 0 0 0 0 0 0 0 0 0 1 ...
##  $ Intercom           : num [1:32963] 1 0 1 1 1 1 1 1 1 1 ...
##  $ SportsFacility     : num [1:32963] 1 1 0 0 1 1 0 0 0 0 ...
##  $ ATM                : num [1:32963] 0 0 0 0 0 0 0 0 0 1 ...
##  $ ClubHouse          : num [1:32963] 1 1 0 0 1 1 1 0 0 1 ...
##  $ School             : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ 24X7Security       : num [1:32963] 1 1 1 0 1 1 1 1 1 1 ...
##  $ PowerBackup        : num [1:32963] 1 1 0 1 1 1 1 1 1 1 ...
##  $ CarParking         : num [1:32963] 0 1 0 0 1 1 1 1 0 1 ...
##  $ StaffQuarter       : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Cafeteria          : num [1:32963] 0 0 0 0 0 0 1 0 1 0 ...
##  $ MultipurposeRoom   : num [1:32963] 0 1 0 0 1 1 0 1 1 1 ...
##  $ Hospital           : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ WashingMachine     : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Gasconnection      : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ AC                 : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Wifi               : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Children'splayarea : num [1:32963] 1 1 1 0 1 1 1 1 0 1 ...
##  $ LiftAvailable      : num [1:32963] 1 1 1 1 1 1 1 1 0 0 ...
##  $ BED                : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ VaastuCompliant    : num [1:32963] 0 1 0 0 0 0 1 1 1 0 ...
##  $ Microwave          : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ GolfCourse         : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ TV                 : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ DiningTable        : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Sofa               : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Wardrobe           : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Refrigerator       : num [1:32963] 0 0 0 0 0 0 0 0 0 0 ...
##  $ city               : chr [1:32963] "Bangalore" "Bangalore" "Bangalore" "Bangalore" ...
```

```
colnames(Housing_data)[4] <- "Bedrooms"
```

We see that there are 22870 entries of NA in the amenities factors.

We can also see that the data has most columns in numeric format, but are infact categorical variable.

We will convert the data to categorical variables. All variables except Area, Price, and Location are categorical variables.

```
## seperating column names in a data frame
column_all <- colnames(Housing_data)
column_all
```

```
##  [1] "Price"               "Area"                "Location"
##  [4] "Bedrooms"            "Resale"              "MaintenanceStaff"
##  [7] "Gymnasium"           "SwimmingPool"        "LandscapedGardens"
## [10] "JoggingTrack"        "RainWaterHarvesting" "IndoorGames"
## [13] "ShoppingMall"        "Intercom"            "SportsFacility"
## [16] "ATM"                 "ClubHouse"           "School"
## [19] "24X7Security"        "PowerBackup"         "CarParking"
## [22] "StaffQuarter"        "Cafeteria"           "MultipurposeRoom"
## [25] "Hospital"            "WashingMachine"      "Gasconnection"
## [28] "AC"                  "Wifi"                "Children'splayarea"
## [31] "LiftAvailable"       "BED"                 "VaastuCompliant"
## [34] "Microwave"           "GolfCourse"          "TV"
## [37] "DiningTable"         "Sofa"                "Wardrobe"
## [40] "Refrigerator"        "city"
```

```
## only retaining columns with categorical variables.
column_factors <- column_all[-c(1,2,3)]


## transforming columns to categorical variables.
Housing_data[,column_factors] <- lapply(Housing_data[,column_factors] , factor)

str(Housing_data)
```

```
## tibble [32,963 x 41] (S3: tbl_df/tbl/data.frame)
##  $ Price              : num [1:32963] 30000000 7888000 4866000 8358000 6845000 ...
##  $ Area               : num [1:32963] 3340 1045 1179 1675 1670 ...
##  $ Location           : chr [1:32963] "JP Nagar Phase 1" "Dasarahalli on Tumkur Road" "Kannur on Thar
##  $ Bedrooms           : Factor w/ 8 levels "1","2","3","4",..: 4 2 2 3 3 2 4 3 3 1 ...
##  $ Resale             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ MaintenanceStaff   : Factor w/ 2 levels "0","1": 2 1 1 1 2 1 1 1 1 2 ...
##  $ Gymnasium          : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 2 2 2 ...
##  $ SwimmingPool       : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 1 2 2 ...
##  $ LandscapedGardens  : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 1 2 2 ...
##  $ JoggingTrack       : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 2 2 2 ...
##  $ RainWaterHarvesting: Factor w/ 2 levels "0","1": 2 2 2 1 2 2 1 2 2 2 ...
##  $ IndoorGames        : Factor w/ 2 levels "0","1": 2 2 1 1 2 2 1 1 1 2 ...
##  $ ShoppingMall       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ Intercom           : Factor w/ 2 levels "0","1": 2 1 2 2 2 2 2 2 2 2 ...
##  $ SportsFacility     : Factor w/ 2 levels "0","1": 2 2 1 1 2 2 1 1 1 1 ...
```

```
##  $ ATM                : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ ClubHouse          : Factor w/ 2 levels "0","1": 2 2 1 1 2 2 2 1 1 2 ...
##  $ School             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ 24X7Security       : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 2 2 2 ...
##  $ PowerBackup        : Factor w/ 2 levels "0","1": 2 2 1 2 2 2 2 2 2 2 ...
##  $ CarParking         : Factor w/ 2 levels "0","1": 1 2 1 1 2 2 2 2 1 2 ...
##  $ StaffQuarter       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Cafeteria          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 1 2 1 ...
##  $ MultipurposeRoom   : Factor w/ 2 levels "0","1": 1 2 1 1 2 2 1 2 2 2 ...
##  $ Hospital           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ WashingMachine     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Gasconnection      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ AC                 : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Wifi               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Children'splayarea : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 2 1 2 ...
##  $ LiftAvailable      : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 1 1 ...
##  $ BED                : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ VaastuCompliant    : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 2 2 2 1 ...
##  $ Microwave          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ GolfCourse         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ TV                 : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ DiningTable        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Sofa               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Wardrobe           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Refrigerator       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ city               : Factor w/ 6 levels "Bangalore","Chennai",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Adding the PPA variable

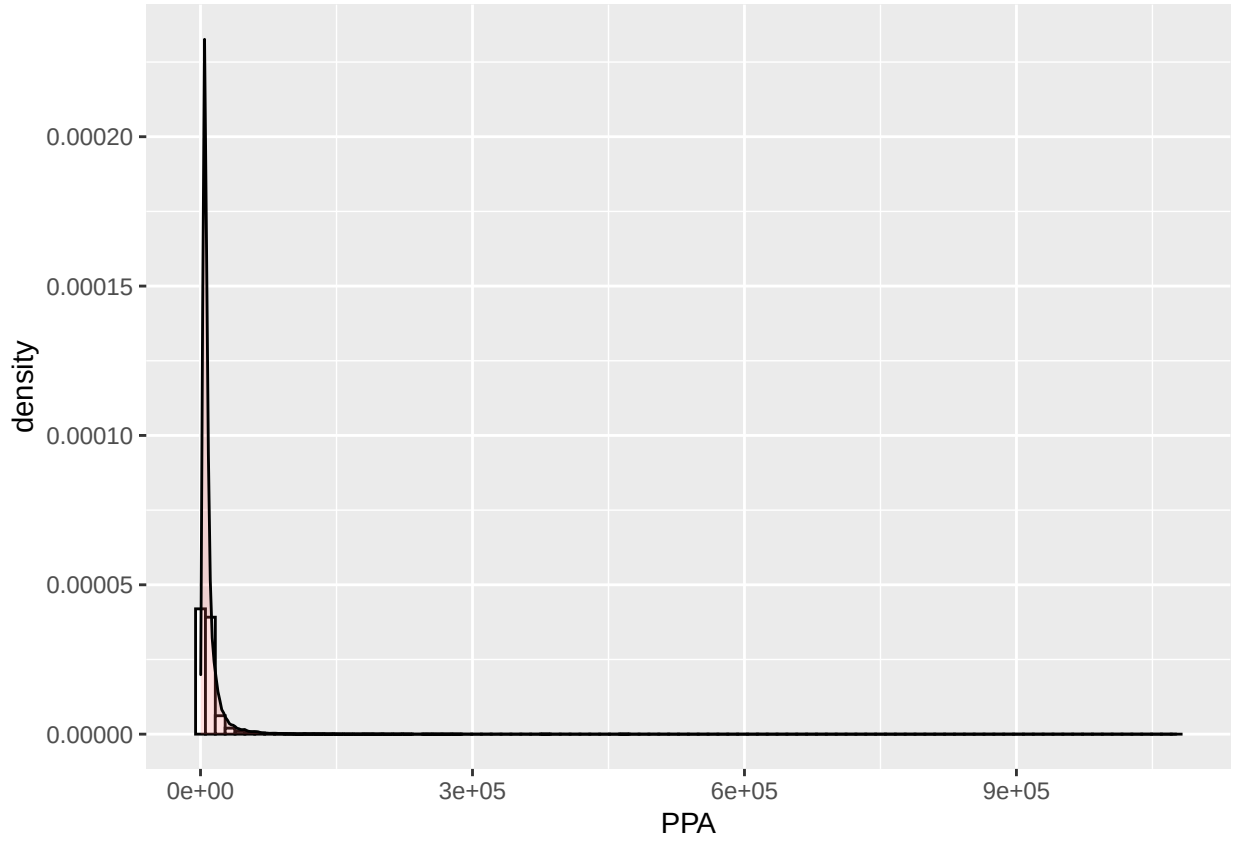
We will now add the price per unit area (PPA) variable to our data set.

```
##to add the ppa variable we use the cbind function

Housing_data <- cbind(Housing_data, PPA = Housing_data$Price/Housing_data$Area)
```

We now have a new variable, PPA in our data set. Our objective would be to predict the PPA for the property in question.

Let us visualize the variable PPA, to get a better sense of the data at hand.

```
## Plotting the PPA values

ggplot(Housing_data, aes(x=PPA)) + geom_histogram(bins=100, aes(y=..density..), colour="black", fill="wh
```



We can see that the PPA values are extremely skewed.

we will therefore have to transform the data to get a better analysis.

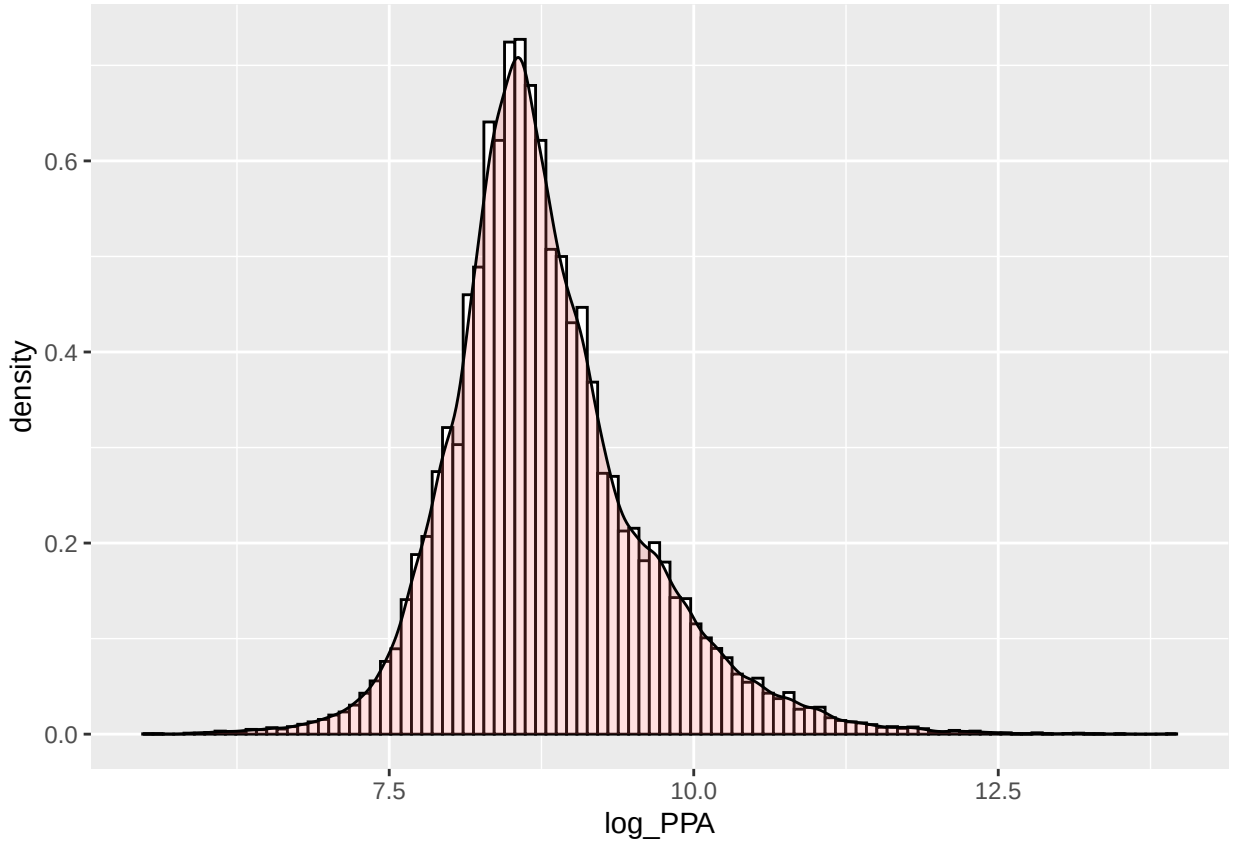
Log transforming the data

We will use a log transform on our data sets and get a better looking distribution.

```
##running a log transform on the data set.

Housing_data <- cbind(Housing_data, log_PPA = log(Housing_data$PPA))
```

Now we can plot the new data set,

```
## Plotting the log PPA values

ggplot(Housing_data, aes(x=log_PPA)) + geom_histogram(bins=100, aes(y=..density..), colour="black", fil
```



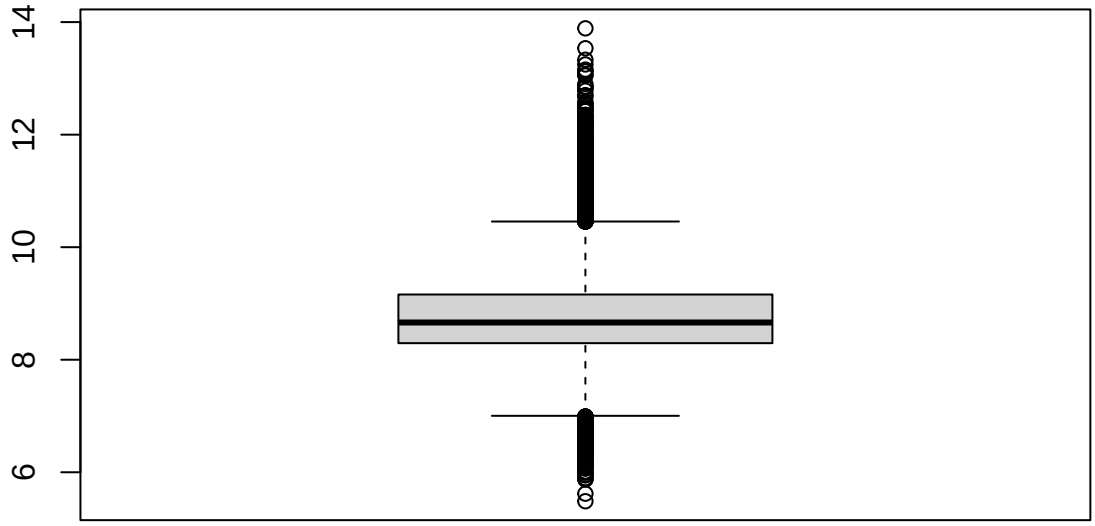
We observe that the log transformed data is normally distributed and hence a better variable to work with.

Cleaning the data

Identifying and Removing Outliers

We will first plot the target variable i.e. Log PPA in a box plot and look at the outliers.

```
## boxplot of the PPA values in the housing_data set to identify outliers
boxplot(Housing_data$log_PPA)
```



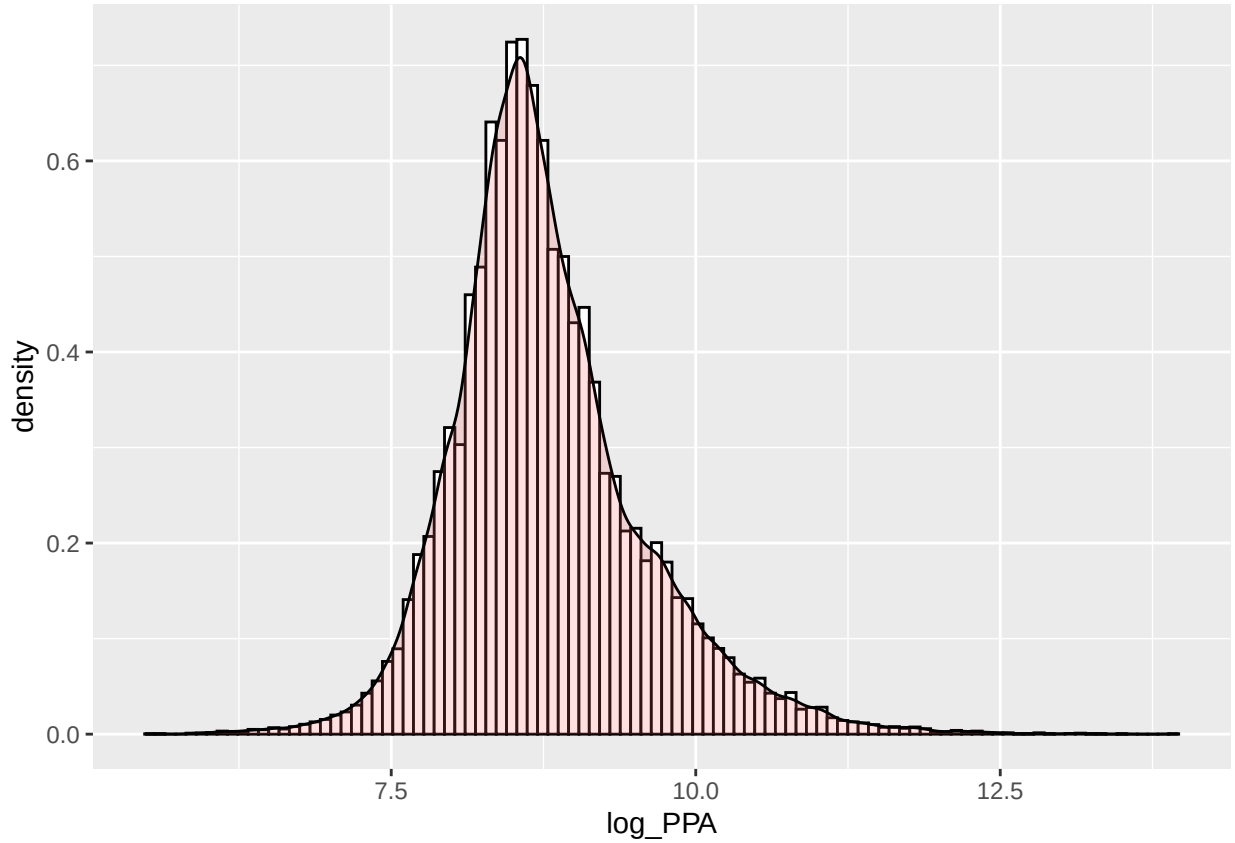
We can see from the plot the long trail of outliers.

We will remove these outliers by comparing the zscores.

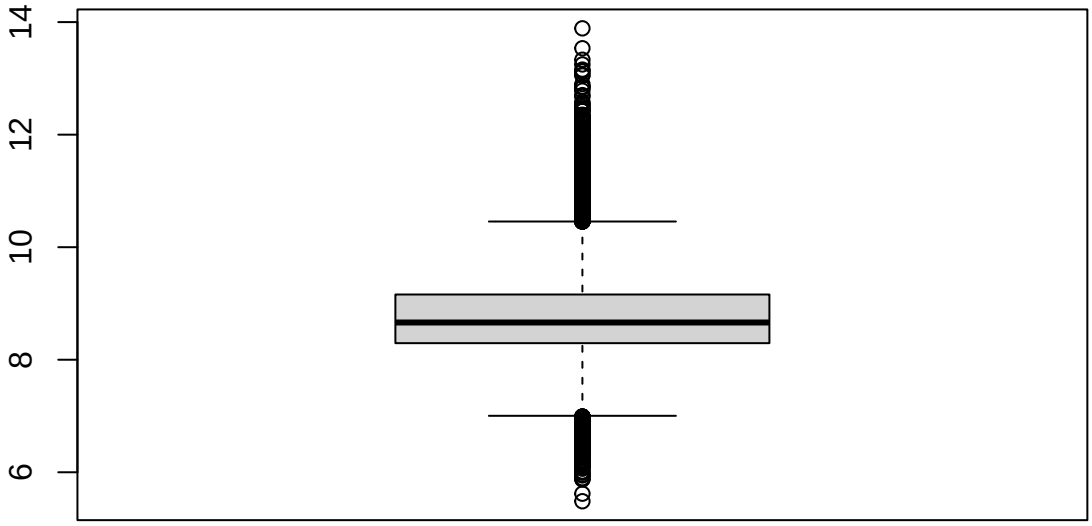
```
##we will add a column with the z_scores of the log_PPA values

Housing_data <- cbind(Housing_data, z_scores = (Housing_data$log_PPA - mean(Housing_data$log_PPA))/sd(H

## we now remove the the z_scores of more than 3 standard deviations
Housing_data <-  Housing_data[!(abs(Housing_data$z_scores) > 3), ]
```

We now have a data set with the outliers removed, we can plot the data in a box plot and on a normal distribution and see how the data looks.

```
## plot of the data without outliers

## normal distribution plot
ggplot(Housing_data, aes(x=log_PPA)) + geom_histogram(bins=100, aes(y=..density..), colour="black", fil
```




```
##box plot

boxplot(Housing_data$log_PPA)
```



We can see that the data distribution looks much better now. The boxplot shows reduced outliers and the trails are reduced. We have rid the data set of the outliers we still have the N/A entries to deal with. ### Removing the NA entries

```
## we use the below code to eliminate the NA.

Housing_data_clean <- na.omit(Housing_data)
```

We can see that after eliminating all rows with NA, we have only 10093 observations left in the clean data set. Removing these entries will make our data set extremely limited. Hence this data set will be used only for limited analysis. (for the regression analysis mainly).

We now have two data sets, Housing_data and Housing_data_clean as two primary data sets. Which are ready to be partitioned and analysed.

Partitioning the data

We now partition the data into the training set and the hold out set. They are named, edx and validation for clarity (and continuity of naming convention used previously).

```
##data partition

set.seed(1, sample.kind="Rounding")
```

Partitioning the whole data

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
test_index <- createDataPartition(y = Housing_data$log_PPA, times = 1, p = 0.1, list = FALSE)
edx <- Housing_data[-test_index,]
temp <- Housing_data[test_index,]

# ensuring that the city and locations are covered in the validation set;
validation <- temp %>% semi_join(edx, by = "city") %>% semi_join(edx, by = "Location")

# Add rows removed from validation set back into edx set
removed <- anti_join(temp, validation)
```

```
## Joining, by = c("Price", "Area", "Location", "Bedrooms", "Resale", "MaintenanceStaff", "Gymnasium",
```

```
edx <- rbind(edx, removed)
```

We now have two data sets, edx, and validation. The edx data set would be the here on which will be used for analysis and cross validation. The validation set will be used only at the end.

#Exploratory Analysis and Summary statistics

Looking at some of the summary statistics and the data set composition to get an understanding of the data set at hand.

```
##exploratory analysis of the data at hand.

head(edx)
```

```
##      Price Area                         Location Bedrooms Resale
## 1 30000000 3340                JP Nagar Phase 1        4      0
## 2  7888000 1045      Dasarahalli on Tumkur Road        2      0
## 3  4866000 1179 Kannur on Thanisandra Main Road        2      0
## 4  8358000 1675                    Doddanekundi        3      0
## 5  6845000 1670                         Kengeri        3      0
## 6  6797000 1220                        Horamavu        2      0
##   MaintenanceStaff Gymnasium SwimmingPool LandscapedGardens JoggingTrack
## 1                1         1            1                 1            1
## 2                0         1            1                 1            1
## 3                0         1            1                 1            1
## 4                0         0            0                 0            0
## 5                1         1            1                 1            1
## 6                0         1            1                 1            1
##   RainWaterHarvesting IndoorGames ShoppingMall Intercom SportsFacility ATM
## 1                   1           1            0        1              1   0
## 2                   1           1            0        0              1   0
## 3                   1           0            0        1              0   0
## 4                   0           0            0        1              0   0
## 5                   1           1            0        1              1   0
## 6                   1           1            0        1              1   0
##   ClubHouse School 24X7Security PowerBackup CarParking StaffQuarter Cafeteria
```

```
## 1         1      0           1           1          0           0         0
## 2         1      0           1           1          1           0         0
## 3         0      0           1           0          0           0         0
## 4         0      0           0           1          0           0         0
## 5         1      0           1           1          1           0         0
## 6         1      0           1           1          1           0         0
##   MultipurposeRoom Hospital WashingMachine Gasconnection AC Wifi
## 1                0        0             0             0  0    0
## 2                1        0             0             0  0    0
## 3                0        0             0             0  0    0
## 4                0        0             0             0  0    0
## 5                1        0             0             0  0    0
## 6                1        0             0             0  0    0
##   Children'splayarea LiftAvailable BED VaastuCompliant Microwave GolfCourse TV
## 1                  1             1   0              0         0          0  0
## 2                  1             1   0              1         0          0  0
## 3                  1             1   0              0         0          0  0
## 4                  0             1   0              0         0          0  0
## 5                  1             1   0              0         0          0  0
## 6                  1             1   0              0         0          0  0
##   DiningTable Sofa Wardrobe Refrigerator      city      PPA  log_PPA
## 1           0    0        0            0 Bangalore 8982.036 9.102982
## 2           0    0        0            0 Bangalore 7548.325 8.929081
## 3           0    0        0            0 Bangalore 4127.226 8.325361
## 4           0    0        0            0 Bangalore 4989.851 8.515161
## 5           0    0        0            0 Bangalore 4098.802 8.318450
## 6           0    0        0            0 Bangalore 5571.311 8.625386
##        z_scores
## 1  1.782259e-05
## 2  8.285009e-06
## 3 -2.482600e-05
## 4 -1.441641e-05
## 5 -2.520502e-05
## 6 -8.371147e-06
```

```
str(edx)
```

```
## 'data.frame':    29732 obs. of  44 variables:
##  $ Price              : num  30000000 7888000 4866000 8358000 6845000 ...
##  $ Area               : num  3340 1045 1179 1675 1670 ...
##  $ Location           : chr  "JP Nagar Phase 1" "Dasarahalli on Tumkur Road" "Kannur on Thanisandra M
##  $ Bedrooms           : Factor w/ 8 levels "1","2","3","4",..: 4 2 2 3 3 2 4 3 3 1 ...
##  $ Resale             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ MaintenanceStaff   : Factor w/ 2 levels "0","1": 2 1 1 1 2 1 1 1 1 2 ...
##  $ Gymnasium          : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 2 2 2 ...
##  $ SwimmingPool       : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 1 2 2 ...
##  $ LandscapedGardens  : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 1 2 2 ...
##  $ JoggingTrack       : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 2 2 2 ...
##  $ RainWaterHarvesting: Factor w/ 2 levels "0","1": 2 2 2 1 2 2 1 2 2 2 ...
##  $ IndoorGames        : Factor w/ 2 levels "0","1": 2 2 1 1 2 2 1 1 1 2 ...
##  $ ShoppingMall       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ Intercom           : Factor w/ 2 levels "0","1": 2 1 2 2 2 2 2 2 2 2 ...
##  $ SportsFacility     : Factor w/ 2 levels "0","1": 2 2 1 1 2 2 1 1 1 1 ...
##  $ ATM                : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
```

```
##  $ ClubHouse          : Factor w/ 2 levels "0","1": 2 2 1 1 2 2 2 1 1 2 ...
##  $ School             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ 24X7Security       : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 2 2 2 ...
##  $ PowerBackup        : Factor w/ 2 levels "0","1": 2 2 1 2 2 2 2 2 2 2 ...
##  $ CarParking         : Factor w/ 2 levels "0","1": 1 2 1 1 2 2 2 2 1 2 ...
##  $ StaffQuarter       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Cafeteria          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 1 2 1 ...
##  $ MultipurposeRoom   : Factor w/ 2 levels "0","1": 1 2 1 1 2 2 1 2 2 2 ...
##  $ Hospital           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ WashingMachine     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Gasconnection      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ AC                 : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Wifi               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Children'splayarea : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 2 1 2 ...
##  $ LiftAvailable      : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 1 1 ...
##  $ BED                : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ VaastuCompliant    : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 2 2 2 1 ...
##  $ Microwave          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ GolfCourse         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ TV                 : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ DiningTable        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Sofa               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Wardrobe           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Refrigerator       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ city               : Factor w/ 6 levels "Bangalore","Chennai",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ PPA                : num  8982 7548 4127 4990 4099 ...
##  $ log_PPA            : num  9.1 8.93 8.33 8.52 8.32 ...
##  $ z_scores           : num  1.78e-05 8.29e-06 -2.48e-05 -1.44e-05 -2.52e-05 ...
```

```
summary(edx)
```

```
##      Price                Area         Location            Bedrooms
##  Min.   :  2000000   Min.   :  200   Length:29732       2      :12899
##  1st Qu.:  4084750   1st Qu.:  855   Class :character   3      :11375
##  Median :  6704500   Median : 1126   Mode  :character   1      : 3451
##  Mean   : 11681001   Mean   : 1292                      4      : 1767
##  3rd Qu.: 12000000   3rd Qu.: 1500                      5      :  197
##  Max.   :854599999   Max.   :16000                      6      :   28
##                                                         (Other):   15
##  Resale    MaintenanceStaff Gymnasium    SwimmingPool LandscapedGardens
##  0:18363   0   : 7559       0   : 4380   0   : 5338   0   : 5918
##  1:11369   1   : 1542       1   : 4721   1   : 3763   1   : 3183
##            NA's:20631       NA's:20631   NA's:20631   NA's:20631
##
##
##
##
##  JoggingTrack RainWaterHarvesting IndoorGames  ShoppingMall Intercom
##  0   : 6083   0   : 5763          0   : 6625   0   : 8410   0   : 4997
##  1   : 3018   1   : 3338          1   : 2476   1   :  691   1   : 4104
##  NA's:20631   NA's:20631          NA's:20631   NA's:20631   NA's:20631
##
##
##
```

```
##
##  SportsFacility   ATM        ClubHouse     School      24X7Security
##  0   : 6613     0   : 8169   0   : 5311   0   : 8430   0   : 5012
##  1   : 2488     1   :  932   1   : 3790   1   :  671   1   : 4089
##  NA's:20631     NA's:20631   NA's:20631   NA's:20631   NA's:20631
##
##
##
##
##  PowerBackup  CarParking   StaffQuarter Cafeteria    MultipurposeRoom
##  0   : 3169   0   : 4860   0   : 7918   0   : 8037   0   : 6920
##  1   : 5932   1   : 4241   1   : 1183   1   : 1064   1   : 2181
##  NA's:20631   NA's:20631   NA's:20631   NA's:20631   NA's:20631
##
##
##
##
##  Hospital     WashingMachine Gasconnection    AC          Wifi
##  0   : 8574   0   : 8704     0   : 7492    0   : 8520   0   : 8916
##  1   :  527   1   :  397     1   : 1609    1   :  581   1   :  185
##  NA's:20631   NA's:20631     NA's:20631    NA's:20631   NA's:20631
##
##
##
##
##  Children'splayarea LiftAvailable   BED        VaastuCompliant Microwave
##  0   : 4446         0   : 2586    0   : 8274   0   : 6385      0   : 8657
##  1   : 4655         1   : 6515    1   :  827   1   : 2716      1   :  444
##  NA's:20631         NA's:20631    NA's:20631   NA's:20631      NA's:20631
##
##
##
##
##  GolfCourse      TV         DiningTable    Sofa       Wardrobe     Refrigerator
##  0   : 8901   0   : 8587   0   : 8612   0   : 8628   0   : 8907   0   : 8627
##  1   :  200   1   :  514   1   :  489   1   :  473   1   :  194   1   :  474
##  NA's:20631   NA's:20631   NA's:20631   NA's:20631   NA's:20631   NA's:20631
##
##
##
##
##        city           PPA             log_PPA          z_scores
##  Bangalore:5584   Min.   :    241   Min.   : 5.485   Min.   :-1.806e-04
##  Chennai  :4520   1st Qu.:   4000   1st Qu.: 8.294   1st Qu.:-2.654e-05
##  Delhi    :4525   Median :   5776   Median : 8.661   Median :-6.395e-06
##  Hyderabad:2269   Mean   :   9750   Mean   : 8.777   Mean   :-3.213e-08
##  Kolkata  :5863   3rd Qu.:   9500   3rd Qu.: 9.159   3rd Qu.: 2.090e-05
##  Mumbai   :6971   Max.   :1076444   Max.   :13.889   Max.   : 2.803e-04
##
```

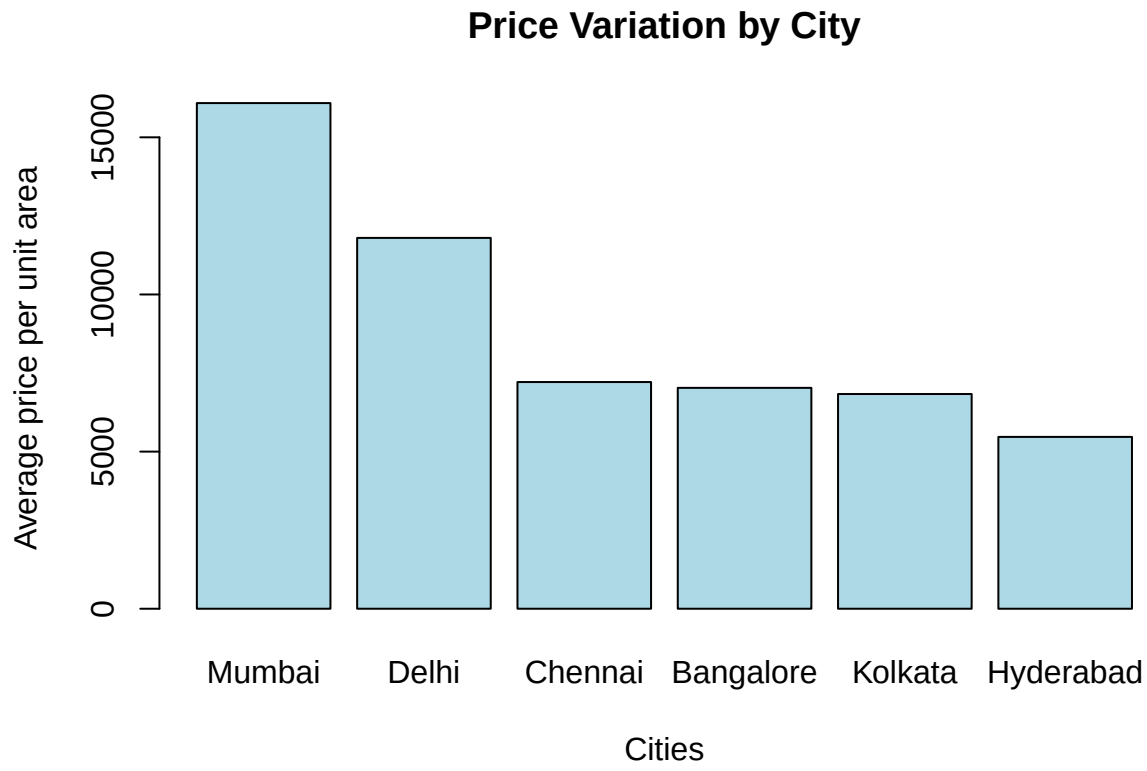
We have a data frame with 29394 observations across 43 variables, the variable of interest here is log_PPA for us.

we will first see how price varies by city.

1. Price Variation by city

As shown in various industry reports and new reports there is a strong price variation, among the cities, this can be easily identified by looking at a plot of the mean price per unit area across the cities.

```
## plot of the price distribution by city

city_sum <- edx %>% group_by(city) %>% summarize(PPA = mean(PPA))
city_sum <- city_sum[order(-city_sum$PPA),]

barplot(city_sum$PPA, names.arg=city_sum$city,  xlab="Cities",ylab="Average price per unit area", col="
```



We can clearly see that city wise there is a strong price variation. Hence city would be an important factor in consideration for our analysis.

#Linear Regression Analysis

Note: for this part we are only going to use the Housing_data_clean data set.

We can see that there are categorical variables, such as city, number of bedrooms which have multiple values (names of cities, and number of bedrooms) within one column. We would have to seperate this out into multiple columns.

We will have to create a seperate column for the cities, with each column just mentioning, 0 or 1, 0 if it doesnt belong to the city of the respective column, and 1 if it belongs to the city for that column.

This form of representation it called a contrast matrix.

##Creating Contrast matrix

We will create a contrast matrix for the cities and the number of bedrooms.

Normally, a categorical variable that has n levels will be converted into n-1 variables,each with two levels 0 and 1. These n-1 new variables contain the same information rather than the single variable. This reformatting creates a table called contrast matrix.

```
##creating city contrast matrix

Contrast_matrix_city <- model.matrix(~city, data = Housing_data_clean)

##adding the contrast matrix to the data set

Housing_data_clean <- cbind(Housing_data_clean, Contrast_matrix_city[,-1])
```

Similarly creating a contrast matrix for the number of bedrooms.

```
##creating bedroom contrast matrix

Contrast_matrix_bedroom <- model.matrix(~Bedrooms, data = Housing_data_clean)

##adding the contrast matrix to the data set

Housing_data_clean <- cbind(Housing_data_clean, Contrast_matrix_bedroom[,-1])
```

Exploratory Analysis

We will first look at correlation amongst all the different variables present, after which we would modify the data set further to arrive at a model.

We first create a data frame with just the variables of relevance.

```
##creating the variable set data frame

column_all <- colnames(Housing_data_clean)
column_all
```

```
##  [1] "Price"              "Area"               "Location"
##  [4] "Bedrooms"           "Resale"             "MaintenanceStaff"
##  [7] "Gymnasium"          "SwimmingPool"       "LandscapedGardens"
## [10] "JoggingTrack"       "RainWaterHarvesting" "IndoorGames"
## [13] "ShoppingMall"       "Intercom"           "SportsFacility"
## [16] "ATM"                "ClubHouse"          "School"
## [19] "24X7Security"       "PowerBackup"        "CarParking"
## [22] "StaffQuarter"       "Cafeteria"          "MultipurposeRoom"
## [25] "Hospital"           "WashingMachine"     "Gasconnection"
## [28] "AC"                 "Wifi"               "Children'splayarea"
## [31] "LiftAvailable"      "BED"                "VaastuCompliant"
## [34] "Microwave"          "GolfCourse"         "TV"
## [37] "DiningTable"        "Sofa"               "Wardrobe"
## [40] "Refrigerator"       "city"               "PPA"
## [43] "log_PPA"            "z_scores"           "cityChennai"
## [46] "cityDelhi"          "cityHyderabad"      "cityKolkata"
## [49] "cityMumbai"         "Bedrooms2"          "Bedrooms3"
```

```
## [52] "Bedrooms4"           "Bedrooms5"           "Bedrooms6"
## [55] "Bedrooms7"           "Bedrooms8"
```

Removing the unncessary columns:

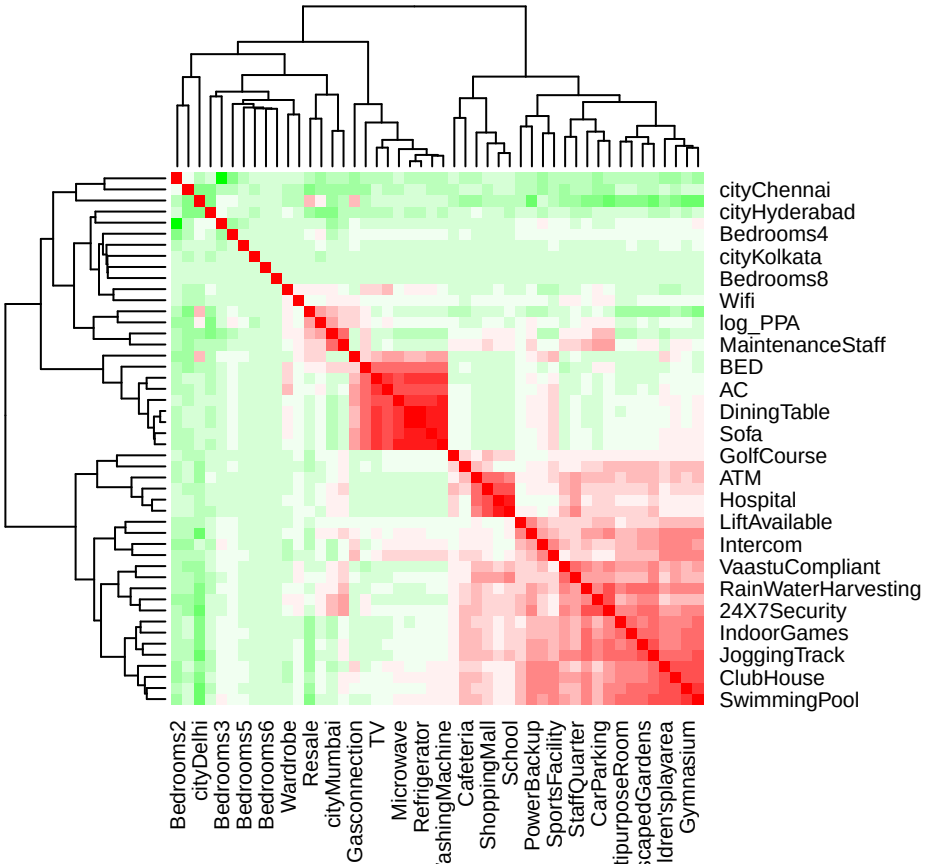
```
##We dont need the columns with price, area, location, city, ppa and z scores, there are no 7 bedroom h

column_variables <- column_all[-c(1,2,3,4,41,42,44,55)]
column_variables
```

```
##  [1] "Resale"              "MaintenanceStaff"    "Gymnasium"
##  [4] "SwimmingPool"        "LandscapedGardens"   "JoggingTrack"
##  [7] "RainWaterHarvesting" "IndoorGames"         "ShoppingMall"
## [10] "Intercom"            "SportsFacility"      "ATM"
## [13] "ClubHouse"           "School"              "24X7Security"
## [16] "PowerBackup"         "CarParking"          "StaffQuarter"
## [19] "Cafeteria"           "MultipurposeRoom"    "Hospital"
## [22] "WashingMachine"      "Gasconnection"       "AC"
## [25] "Wifi"                "Children'splayarea"  "LiftAvailable"
## [28] "BED"                 "VaastuCompliant"     "Microwave"
## [31] "GolfCourse"          "TV"                  "DiningTable"
## [34] "Sofa"                "Wardrobe"            "Refrigerator"
## [37] "log_PPA"             "cityChennai"         "cityDelhi"
## [40] "cityHyderabad"       "cityKolkata"         "cityMumbai"
## [43] "Bedrooms2"           "Bedrooms3"           "Bedrooms4"
## [46] "Bedrooms5"           "Bedrooms6"           "Bedrooms8"
```

```
##we now create the variable set
Variables_set <- Housing_data_clean[,column_variables]

## transforming the variable set to integer.
Variables_set[] <- lapply(Variables_set,as.integer)
```

We now arrive at the correlation amongst all the variables,

```
##calculating the correlation.

Variables_set.cor <- cor(Variables_set)

##creating a heatmap of the data

palette = colorRampPalette(c("green", "white", "red")) (20)

heatmap(x = Variables_set.cor, col = palette, symm = TRUE)
```



We can see that there are some factors that are strongly intercorrelated and hence can be grouped together.

There are largely 2 types,

A. Furniture , which includes, Bed, microwave, Sofa, Tv, dinning table, gas connection, refrigerator, wardrobe, wifi and washing machine.

B. Features, which includes, golfcourse, cafeteria, atm, shopping mall, hospital, school, lift, powerbackup, intercom, sports facility, vastu, staffroom, rain water harvesting, car park, security, multipurpose room, indoor games, landscaped gardens, jogging tack, childrens area, clubhosue, gym, and swimming pool.

We will group this area into combined columns. and replace all these variables by just 2 columns,

```
## if the sum of individual furniture is 9 (since absence is marked by integer 1) or more, we will plac
Variables_set <- cbind(Variables_set, Furniture_sum = Variables_set$BED + Variables_set$Microwave +Varia


##replacing furniture with a categorical value

Variables_set$furniture <- ifelse(Variables_set$Furniture_sum > 10, 1, 0)
```

adding the feature column

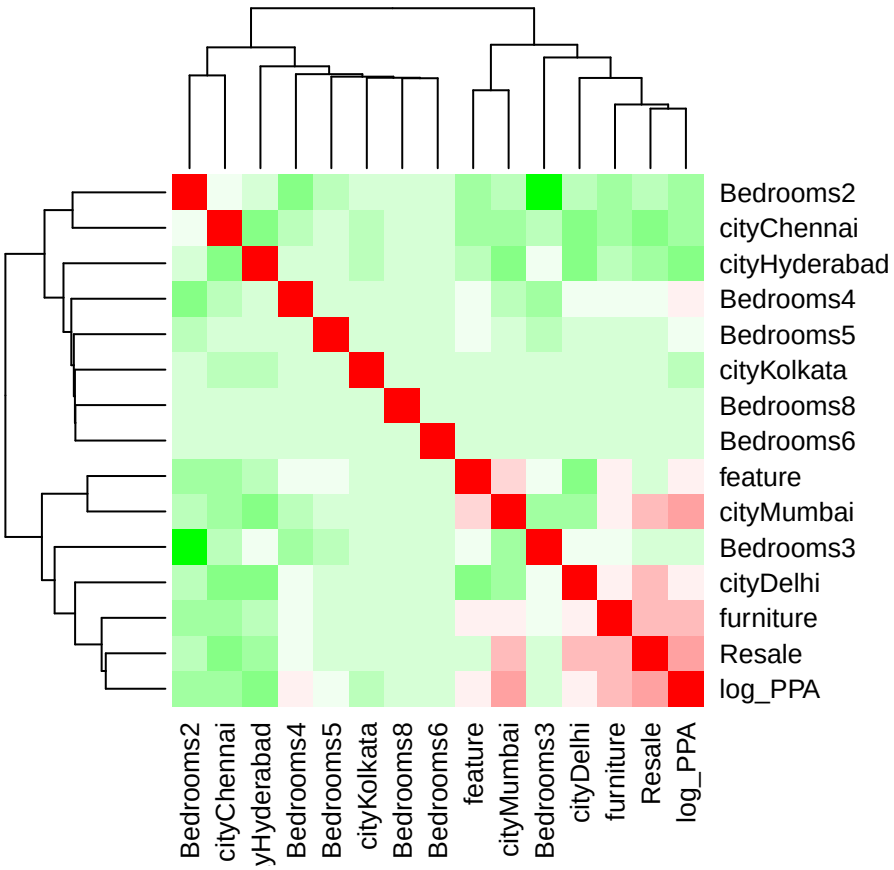
```
## if the sum of individual furniture is 9 (since absence is marked by integer 1) or more, we will plac
Variables_set <- cbind(Variables_set, Feature_sum = Variables_set$GolfCourse+ Variables_set$Cafeteria +


##replacing furniture with a categorical value
```

```
Variables_set$feature <- ifelse(Variables_set$Feature_sum > 25 , 1, 0)
```

we will now run a correlation check again with the limited number of factors.

```
## creating a variable set with limited variables

lim_column <- c("Resale", "feature", "furniture", "log_PPA", "cityChennai", "cityDelhi", "cityHyderabad

new_set <- as.data.frame(Variables_set[,lim_column])
```

```
##checking the correlation in the limited set

new_set.cor <- cor(new_set)

##creating a heatmap of the data

palette = colorRampPalette(c("green", "white", "red")) (20)

heatmap(x = new_set.cor, col = palette, symm = TRUE)
```



We can now see that there is reduced inter - correlation among factors. We can now add our two new variables, i.e. Feature and Furniture to the main set i.e. the housing_data_clean.

```
##adding the furniture and feature column to the housing data.

Housing_data_clean <- cbind(Housing_data_clean, furniture = Variables_set$furniture)

Housing_data_clean <- cbind(Housing_data_clean, feature = Variables_set$feature)

Housing_data_clean$Resale <- as.numeric(Housing_data_clean$Resale)
```

Partitioning the clean data We can now partition the clean data set, similar to the partition done in the main set previously.

```
##data partition

set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
test_index_clean <- createDataPartition(y = Housing_data_clean$log_PPA, times = 1, p = 0.1, list = FALSE
edx_clean <- Housing_data_clean[-test_index_clean,]
temp_clean <- Housing_data_clean[test_index_clean,]

# ensuring that the city and locations are covered in the validation set;
validation_clean <- temp_clean %>% semi_join(edx_clean, by = "city") %>% semi_join(edx_clean, by = "Loca

# Add rows removed from validation set back into edx set
removed_clean <- anti_join(temp_clean, validation_clean)
```

```
## Joining, by = c("Price", "Area", "Location", "Bedrooms", "Resale", "MaintenanceStaff", "Gymnasium",
```

```
edx_clean <- rbind(edx_clean, removed_clean)
```

We now have further more two data sets, edx_clean, and validation_clean. The edx_clean data set would be the here on which will be used for analysis and cross validation in the Linear regression appraoch. The validation_clean set will be used only at the end.

Linear regression models

LM: Model 1 : cities

We will first create a linear regression model, only taking the cities into consideration.

```
#writing a linear regression model with only cities

model_1 <- lm(log_PPA~cityChennai+ cityDelhi+ cityHyderabad+ cityKolkata + cityMumbai, data = edx_clean)
summary(model_1)
```

```
##
## Call:
```

```
## lm(formula = log_PPA ~ cityChennai + cityDelhi + cityHyderabad +
##     cityKolkata + cityMumbai, data = edx_clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.65081 -0.27579 -0.04253  0.22973  2.33138
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    8.590523   0.009377 916.168  < 2e-16 ***
## cityChennai    0.064394   0.012855   5.009 5.57e-07 ***
## cityDelhi      0.271670   0.013146  20.666  < 2e-16 ***
## cityHyderabad -0.046715   0.012593  -3.709 0.000209 ***
## cityKolkata   -0.227674   0.047969  -4.746 2.10e-06 ***
## cityMumbai     0.685883   0.014518  47.244  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3936 on 9107 degrees of freedom
## Multiple R-squared:  0.2762, Adjusted R-squared:  0.2758
## F-statistic: 694.9 on 5 and 9107 DF,  p-value: < 2.2e-16
```

We can see that the rsquared for this model is only 27.48% which can be further improved.

saving the rsquare value in a table.

```
##saving rsquare in a table

Rsqr_model1 <- summary(model_1)$r.squared
Rsqr_table <- tibble(method="Model 1: cities", R_sqr = Rsqr_model1)
Rsqr_table
```

```
## # A tibble: 1 x 2
##   method          R_sqr
##   <chr>           <dbl>
## 1 Model 1: cities 0.276
```

we will bring the next factor into consideration, which is number of Bedrooms.

LM: Model 2: Cities + Bedrooms

We will create a linear regression model, taking the cities and the number of bedrooms into consideration.

```
#writing a linear regression model with only cities

model_2 <- lm(log_PPA~cityChennai+ cityDelhi+ cityHyderabad+ cityKolkata + cityMumbai + Bedrooms2 + Bedr
summary(model_2)
```

```
##
## Call:
## lm(formula = log_PPA ~ cityChennai + cityDelhi + cityHyderabad +
##     cityKolkata + cityMumbai + Bedrooms2 + Bedrooms3 + Bedrooms4 +
```

```
##     Bedrooms5 + Bedrooms6 + Bedrooms8, data = edx_clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.61438 -0.24220 -0.04701  0.19569  2.10695
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    8.39132    0.01676 500.667  < 2e-16 ***
## cityChennai    0.10234    0.01198   8.541  < 2e-16 ***
## cityDelhi      0.23638    0.01224  19.316  < 2e-16 ***
## cityHyderabad -0.06468    0.01169  -5.532 3.25e-08 ***
## cityKolkata   -0.23863    0.04449  -5.364 8.34e-08 ***
## cityMumbai     0.76613    0.01426  53.712  < 2e-16 ***
## Bedrooms2      0.08253    0.01531   5.392 7.15e-08 ***
## Bedrooms3      0.28034    0.01564  17.928  < 2e-16 ***
## Bedrooms4      0.59678    0.02144  27.838  < 2e-16 ***
## Bedrooms5      0.82474    0.05633  14.642  < 2e-16 ***
## Bedrooms6      0.27681    0.21117   1.311    0.190
## Bedrooms8      0.41537    0.25863   1.606    0.108
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.365 on 9101 degrees of freedom
## Multiple R-squared:  0.3779, Adjusted R-squared:  0.3772
## F-statistic: 502.6 on 11 and 9101 DF,  p-value: < 2.2e-16
```

we can see that the R square value has improved hence we can say we are moving in the right direction.

saving the rsquare value in a table.

```
##saving rsquare in a table

Rsqr_model2 <- summary(model_2)$r.squared
Rsqr_table <- bind_rows(Rsqr_table, data_frame(method="Model 2: cities + Bedroom", R_sqr = Rsqr_model2))
```

```
## Warning: `data_frame()` was deprecated in tibble 1.1.0.
## Please use `tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```

```
Rsqr_table
```

```
## # A tibble: 2 x 2
##   method                   R_sqr
##   <chr>                    <dbl>
## 1 Model 1: cities          0.276
## 2 Model 2: cities + Bedroom 0.378
```

Further adding the next varibale of resale.

LM: Model 3: Cities + Bedrooms + Resale

We will create a linear regression model, taking the cities and the number of bedrooms and the resale position into consideration.

```
#writing a linear regression model with only cities

model_3 <- lm(log_PPA~cityChennai+ cityDelhi+ cityHyderabad+ cityKolkata + cityMumbai + Bedrooms2 + Bedr
summary(model_3)
```

```
##
## Call:
## lm(formula = log_PPA ~ cityChennai + cityDelhi + cityHyderabad +
##     cityKolkata + cityMumbai + Bedrooms2 + Bedrooms3 + Bedrooms4 +
##     Bedrooms5 + Bedrooms6 + Bedrooms8 + Resale, data = edx_clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.6483 -0.2375 -0.0405  0.1935  2.1211
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    8.240547   0.019733 417.596  < 2e-16 ***
## cityChennai    0.099372   0.011856   8.382  < 2e-16 ***
## cityDelhi      0.152253   0.013498  11.280  < 2e-16 ***
## cityHyderabad -0.082563   0.011637  -7.095 1.39e-12 ***
## cityKolkata   -0.271739   0.044075  -6.165 7.33e-10 ***
## cityMumbai     0.670176   0.015668  42.774  < 2e-16 ***
## Bedrooms2      0.083838   0.015144   5.536 3.18e-08 ***
## Bedrooms3      0.277275   0.015472  17.922  < 2e-16 ***
## Bedrooms4      0.585392   0.021224  27.581  < 2e-16 ***
## Bedrooms5      0.812964   0.055733  14.587  < 2e-16 ***
## Bedrooms6      0.264700   0.208914   1.267    0.205
## Bedrooms8      0.304659   0.255989   1.190    0.234
## Resale         0.139685   0.009912  14.093  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3611 on 9100 degrees of freedom
## Multiple R-squared:  0.3912, Adjusted R-squared:  0.3904
## F-statistic: 487.3 on 12 and 9100 DF,  p-value: < 2.2e-16
```

The rsquare has improved from 37.7% to 39.1%.

```
##saving rsquare in a table

Rsqr_model3 <- summary(model_3)$r.squared
Rsqr_table <- bind_rows(Rsqr_table, data_frame(method="Model 3: cities + Bedroom + Resale", R_sqr = Rsq
Rsqr_table
```

```
## # A tibble: 3 x 2
##   method                              R_sqr
##   <chr>                               <dbl>
```

```
## 1 Model 1: cities                    0.276
## 2 Model 2: cities + Bedroom          0.378
## 3 Model 3: cities + Bedroom + Resale 0.391
```

LM: Model 4: Cities + Bedrooms + Resale + Furniture

We will create a linear regression model, taking the cities, the number of bedrooms, the resale position and the presence of furniture into consideration.

```
#writing a linear regression model with only cities

model_4 <- lm(log_PPA~cityChennai+ cityDelhi+ cityHyderabad+ cityKolkata + cityMumbai + Bedrooms2 + Bedi
summary(model_4)
```

```
##
## Call:
## lm(formula = log_PPA ~ cityChennai + cityDelhi + cityHyderabad +
##     cityKolkata + cityMumbai + Bedrooms2 + Bedrooms3 + Bedrooms4 +
##     Bedrooms5 + Bedrooms6 + Bedrooms8 + Resale + furniture, data = edx_clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.5692 -0.2245 -0.0388  0.1810  2.1571
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    8.251082   0.019362 426.141  < 2e-16 ***
## cityChennai    0.115646   0.011660   9.918  < 2e-16 ***
## cityDelhi      0.138281   0.013259  10.429  < 2e-16 ***
## cityHyderabad -0.071544   0.011428  -6.260 4.01e-10 ***
## cityKolkata   -0.242291   0.043256  -5.601 2.19e-08 ***
## cityMumbai     0.654686   0.015389  42.543  < 2e-16 ***
## Bedrooms2      0.089421   0.014856   6.019 1.82e-09 ***
## Bedrooms3      0.270623   0.015179  17.829  < 2e-16 ***
## Bedrooms4      0.563988   0.020847  27.053  < 2e-16 ***
## Bedrooms5      0.771208   0.054707  14.097  < 2e-16 ***
## Bedrooms6      0.208482   0.204923   1.017    0.309
## Bedrooms8      0.362869   0.251092   1.445    0.148
## Resale         0.099803   0.009946  10.035  < 2e-16 ***
## furniture      0.178209   0.009382  18.996  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3542 on 9099 degrees of freedom
## Multiple R-squared:  0.4144, Adjusted R-squared:  0.4136
## F-statistic: 495.3 on 13 and 9099 DF,  p-value: < 2.2e-16
```

The rsquare has improved from 39.1% to 41.4%

```
##saving rsquare in a table

Rsqr_model4 <- summary(model_4)$r.squared
```

```
Rsqr_table <- bind_rows(Rsqr_table, data_frame(method="Model 4: cities, bedrooms, resale + furniture", R
Rsqr_table
```

```
## # A tibble: 4 x 2
##   method                                        R_sqr
##   <chr>                                         <dbl>
## 1 Model 1: cities                               0.276
## 2 Model 2: cities + Bedroom                     0.378
## 3 Model 3: cities + Bedroom + Resale            0.391
## 4 Model 4: cities, bedrooms, resale + furniture 0.414
```

and finally we will add the feature factors.

LM: Model 5: Cities + Bedrooms + Resale + furniture + feature

In addition to the previously considered models we will now add the features.

```
#writing a linear regression model with only cities

model_5 <- lm(log_PPA~cityChennai+ cityDelhi+ cityHyderabad+ cityKolkata + cityMumbai + Bedrooms2 + Bedr
summary(model_5)
```

```
##
## Call:
## lm(formula = log_PPA ~ cityChennai + cityDelhi + cityHyderabad +
##     cityKolkata + cityMumbai + Bedrooms2 + Bedrooms3 + Bedrooms4 +
##     Bedrooms5 + Bedrooms6 + Bedrooms8 + Resale + furniture +
##     feature, data = edx_clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.51259 -0.21429 -0.04131  0.16880  2.15641
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    8.129386   0.021063 385.962  < 2e-16 ***
## cityChennai    0.166803   0.012110  13.774  < 2e-16 ***
## cityDelhi      0.217602   0.014306  15.210  < 2e-16 ***
## cityHyderabad -0.028504   0.011724  -2.431   0.0151 *
## cityKolkata   -0.216104   0.042847  -5.044 4.66e-07 ***
## cityMumbai     0.656341   0.015229  43.099  < 2e-16 ***
## Bedrooms2      0.093996   0.014705   6.392 1.71e-10 ***
## Bedrooms3      0.257430   0.015050  17.105  < 2e-16 ***
## Bedrooms4      0.535100   0.020734  25.807  < 2e-16 ***
## Bedrooms5      0.731103   0.054214  13.486  < 2e-16 ***
## Bedrooms6      0.237111   0.202799   1.169   0.2424
## Bedrooms8      0.364685   0.248476   1.468   0.1422
## Resale         0.105838   0.009852  10.743  < 2e-16 ***
## furniture      0.150008   0.009502  15.786  < 2e-16 ***
## feature        0.129542   0.009311  13.913  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.3505 on 9098 degrees of freedom
## Multiple R-squared:  0.4266, Adjusted R-squared:  0.4257
## F-statistic: 483.5 on 14 and 9098 DF,  p-value: < 2.2e-16
```

The rsquare has improved from 41.4% to 42.5%.

```
##saving rsquare in a table

Rsqr_model5 <- summary(model_5)$r.squared
Rsqr_table <- bind_rows(Rsqr_table, data_frame(method="Model 5: cities, bedroom, resale, furniture, fea
Rsqr_table
```

```
## # A tibble: 5 x 2
##   method                                              R_sqr
##   <chr>                                               <dbl>
## 1 Model 1: cities                                     0.276
## 2 Model 2: cities + Bedroom                           0.378
## 3 Model 3: cities + Bedroom + Resale                  0.391
## 4 Model 4: cities, bedrooms, resale + furniture       0.414
## 5 Model 5: cities, bedroom, resale, furniture, feature 0.427
```

We have a model which gives us 42.5% R squared. This is not the best value for Rsquared but is sufficient given the limitations in the data sets.

The equation for the regression model will finally look as below:

$log PPA = 0.159*cityChennai + 0.206*cityDelhi - 0.0332*cityHyderabad - 0.2221*cityKolkata + 0.657*cityMumbai + 0.101*Bedrooms2 + 0.269*Bedrooms3 + 0.542*Bedrooms4 + 0.78*Bedrooms5 + 0.2147*Bedrooms6 + 0.3729*Bedrooms8 + 0.104*Resale1 + 0.1536*furniture + 0.12069*feature $

Running this equation through the edx_clean set for cross validation we can arrive at an RMSE.

```
## we first predict the the value

edx_clean <- cbind(edx_clean, prediction = 0.159*edx_clean$cityChennai + 0.206*edx_clean$cityDelhi - 0.
```

we can now find the RMSE

```
RMSE_cv <- RMSE(edx_clean$log_PPA, edx_clean$prediction)

RMSE_cv
```

```
## [1] 8.141161
```

since this is the RMSE for the log_PPA value we take an anti log to arrive at the RMSE_PPA

```
exp(RMSE_cv)
```

```
## [1] 3432.901
```

The root mean squared error at the PPA level is 3431. which is Rs. 3431 per unit area. The mean Price per unit area was Rs. 9750.

Validation

Running the RMSE check on the validation data set.

```
## we first predict the the value

validation_clean <- cbind(validation_clean, prediction = 0.159*validation_clean$cityChennai + 0.206*vali
```

we can now find the RMSE

```
#################### RMSE VALIDATION ###################################

RMSE_v <- RMSE(validation_clean$log_PPA, validation_clean$prediction)

RMSE_v
```

```
## [1] 8.148511
```

since this is the RMSE for the log_PPA value we take an anti log to arrive at the RMSE_PPA

```
exp(RMSE_v)
```

```
## [1] 3458.225
```

The root mean squared error at the PPA level is 3458. which is Rs. 3458 per unit area. The mean Price per unit area was Rs. 9750.

#Conclusion

The RMSE for the data set is calculated and shows a sizeable large error. Given the limitations in the data set, the RMSE can still be considered to be sufficiently low. The data set can be improved by obtaining more data points, better weeding out of outliers and better documentation of some of the factors. Some of the factors could have been more comprehensive.

References

1] https://www.kaggle.com/ruchi798/housing-prices-in-metropolitan-areas-of-india

2] Git hub repo: <https://github.com/niharonline/Housing-Prices_CYOP_Nihar>

3] https://economictimes.indiatimes.com/industry/services/property-/-cstruction/residential-real-estate-market-beats-pandemic-blues-sales-in-top-7-housing-markets-grow-71-yoy/articleshow/88641967.cms

4] https://www.ibef.org/industry/real-estate-india.aspx

5] https://www.newindianexpress.com/cities/hyderabad/2022/feb/18/hyderabad-second-most-expensive-housing-market-in-india-2420903.html