Introduction

The key of making money is through Investing, People often invest in Stocks, Liquid gold, crypto currency etc. One of the important investment is Real estate market, Real Estate Property is not only the basic need of a man but today it also represents the riches and prestige of a person. Investment in real estate generally seems to be profitable because their property values do not decline rapidly. Changes in the real estate price can affect various household investors, bankers, policy makers and many. Investment in real estate sector seems to be an attractive choice for the investments. Thus, predicting the real estate value is an important economic index. India ranks second in the world in number of households according to 2011 census with a number of 24.67 crore. According to the 2017 version of Emerging Trends in Real Estate Asia Pacific, Mumbai and Bangalore are the top-ranked cities for investment and development. These cities have supplanted Tokyo and Sydney. The house prices of 22 cities out of 26 dropped in the quarter from April to June when compared to the quarter January to March according to National Housing Bank's Residex(residential index). With the introduction of Real Estate Regulation Development Act (RERA) and Benami property Act throughout the country India, more number of investors are attracted to invest into real estate in India. The strengthening and modernizing of the Indian economy has made India as attractive Investment destination. However, past recessions show that real estate prices cannot necessarily grow. Prices of the real estate property are related to the economic conditions of the state In India, the real estate sector is the second-highest employment generator, after the agriculture sector. Real estate sector in India is expected to reach US$ 1 trillion by 2030. By 2025, it will contribute 13% to country's GDP. Emergence of nuclear families, rapid urbanisation and rising household income are likely to remain the key drivers for growth in all spheres of real estate, including residential, commercial, and retail. Rapid urbanisation in the country is pushing the growth of real estate. >70-75% of India's GDP will be contributed by urban areas by 2020. According to India Ratings and Research (Ind-Ra), the Indian real estate sector may stage a sharp K-shaped recovery in FY22. However, the overall sales in FY22 could still be ~14% below the FY20 levels.

India's Global Real Estate Transparency Index ranking improved by a notch to 34 in 2019 on the back of regulatory reforms, better market data and green initiatives according to property consultant JLL.

Indian real estate attracted U$ 5 billion institutional investments in 2020, equivalent to 93% of transactions recorded in the previous year. Investments from private equity (PE) players and VC funds reached US$ 4.06 billion in 2020.

The office market in top eight cities recorded transactions of 22.2 msf from July 2020 to December 2020, whereas new completions were recorded at 17.2 msf in the same period. In terms of share of sectoral occupiers, Information Technology (IT/ITeS) sector dominated with a 41% share in second half of 2020, followed by BSFI and Manufacturing sectors with 16% each, while Other Services and Co-working sectors recorded 17% and 10%, respectively. The office space leasing activity is expected to pick up in 2021 and is likely to be at par with the 10-year average, i.e., 30-31 million sq. ft.

Home sales volume across eight major cities in India jumped by 2x to 61,593 units from October 2020 to December 2020, compared with 33,403 units in the previous quarter, signifying healthy recovery post the strict lockdown imposed in the second quarter due to the spread of COVID-19 in the country.

According to the Economic Times Housing Finance Summit, about 3 houses are built per 1,000 people per year compared with the required construction rate of five houses per 1,000 population. The current shortage of housing in urban areas is estimated to be ~10 million units. An additional 25 million units of affordable housing are required by 2030 to meet the growth in the country's urban population.

The Government of India has been supportive towards the real estate sector. In August 2015, the Union Cabinet approved 100 Smart City Projects in India. The Government has also raised FDI (Foreign Direct Investment) limits for townships and settlements development projects to 100%. Real estate projects within Special Economic Zones (SEZ) are also permitted for 100% FDI. Construction is the third-largest sector in terms of FDI inflow. FDI in the sector (including construction development and construction activities) stood at US$ 42.97 billion between April 2000 and September 2020. Exports from SEZs reached Rs. 7.96 lakh crore (US$ 113.0 billion) in FY20 and grew ~13.6% from Rs. 7.1 lakh crore (US$ 100.3 billion) in FY19. Indian real estate is expected to attract a substantial amount of FDI in the next two years with US$ 8 billion capital infusion by FY22.

Government of India's Housing for All initiative is expected to bring US$ 1.3 trillion investments in the housing sector by 2025. As of December 2019, under Pradhan Mantri Awas

Yojana (Urban) [PMAY (U)], 1.12 crore houses were sanctioned in urban areas, with a potential to create 1.20 crore jobs. The scheme is expected to push affordable housing and construction in the country and give a boost to the real estate sector. On July 09, 2020, Union Cabinet approved the development of Affordable Rental Housing Complexes (AHRCs) for urban migrants and poor as a sub-scheme under PMAY–U.

Government has also released draft guidelines for investment by Real Estate Investment Trusts (REITs) in non-residential segment.

The Ministry of Housing and Urban Affairs has recommended all the states to consider reducing stamp duty of property transactions in a bid to push real estate activity, generate more revenue and aid economic growth.

To know more about Real estate Investment, major cities of the country are being highlighted which are as follows-

Mumbai

Mumbai experiences similar urbanisation challenges as other fast growing cities in developing countries: wide disparities in housing between the affluent, middle-income and low-income segments of the population.

Highly desirable neighborhoods such as Colaba, Malabar Hill, Marine Drive, Bandra and Juhu house professionals, industrialists, Bollywood movie stars and expatriates. Up-scale flats have 3 or more bedrooms, ocean views, tasteful interior decoration, parking for luxury cars and sleeping quarters for maids and cooks. In 2007, Mumbai condominiums were the priciest in the developing world at around US$9,000 to US$10,200 per square metre.[6] Mumbai has more than 1,500 high rise buildings, many of which are just planned, but some already constructed or under construction.

Despite the recent economic growth, there is still some poverty, unemployment and therefore, poor housing conditions for a huge section of the population. With available space at a premium, working-class Mumbai residents often reside in cramped and poor quality, yet relatively expensive housing, usually far from workplaces. Despite this, Mumbai's economic boom continues to attract migrants in search of opportunities from across the country. The number of migrants to Mumbai from outside Maharashtra during the 1991–2001 decade was 1.12 million, which amounted to 54.8% of the net addition to the population of Mumbai.

Over 7 million people, over 42% of the population of Mumbai, live in informal housing or slums, yet they cover only 6–8% of the city's land area. Slums were recognized as a problem as early as in 1919 by the Bombay Municipal Corporation. Slum growth rate in Mumbai is greater than the general urban growth rate. Financial Times writes that "Dharavi is the grand panjandrum of the Mumbai slums". Dharavi, Asia's second-largest slum is located in central Mumbai and houses over 1 million people. Slums are a growing tourist attraction in Mumbai.

Most of the remaining live in chawls and on footpaths. Chawls are a quintessentially Mumbai phenomenon of multi-storied terrible quality tenements, typically a bit higher quality than slums. 80 per cent of chawls have only one room. Pavement dwellers refers to Mumbai dwellings built on the footpaths/pavements of city streets.

With rising incomes, many residents of slums and chawls now have modern amenities such as mobile phones, access to electricity, often illegally, and television.

Rent control laws have helped to create a housing shortage. Most of the investors are looking to invest in ongoing real estate projects to get maximum returns.

The MMRDA has released the Mumbai Development Plan for 2034 which discusses means of creating affordable housing, including a criticised proposal to build on salt pan land.

Delhi

Delhi has witnessed rapid suburban growth over the past decade. South Delhi, Gurgaon and Noida have added thousands of apartment buildings, Affordable Homes, shopping centres and highways. New Delhi's famous Lutyens bungalows house the prime minister, members of his cabinet, top political and government leaders, military officials, senior judges and top bureaucrats. New Delhi is also home to thousands of diplomatic staff of foreign countries and the United Nations. With India's growth, Delhi has developed into a business center, especially for outsourcing, IT consultancy, high-tech, research, education and health care services. Employees of these institutions are the source of growing demand for high-end housing provided by major builders such as DLF.

Bangalore

In the 1990s the information technology boom hit Bangalore. Y2K projects in America's IT industry resulted in shortages for skilled computer scientists and systems programmers. Bangalore has transformed into the Silicon Valley of India as over 500,000 well-paying jobs for young college graduates were created. The demographics of the city changed, new high-rise were built, campus-style office parks sprouted, vast shopping centers started to thrive, streets became crowded with new cars and gated expatriate housing estates emerged.

Roughly 10% of Bangalore's population lives in slums.

Kolkata (Calcutta)

The most sought-after neighbourhoods of Calcutta are generally centered around Lower Circular Road, Sarat Bose Road, Salt Lake, Ballygunge, Anwar Shah Road, Chowringhee and Golf Green. A recent building boom has converted sprawling British-era bungalows into high-rise condominiums and apartment-buildings with modern amenities. Kolkata currently has the second most number of highrises and tall buildings in the country, second only to Mumbai. The highest of them is at 65 floors (The 42). New suburbs are constantly being developed in Rajarhat and along the Eastern Metropolitan Bypass. These suburbs will consist of major condominiums, complete with penthouses, many designed primarily for NRIs, expats and affluent residents. The tallest buildings in the city, The South City Towers and Urbana towers, are also condominiums.

North Calcutta contains mansions built in the early 20th century during Calcutta's heyday as capital of British India. These buildings feature a courtyard surrounded by balconies, large rooms with tall ceilings, marble floors, tall pillars and crumbling artwork. Most of them are poorly maintained.The Marble Palace and other buildings received "heritage status" which provides them municipal funds and incentives to repair and restore. These mansions serve as reminder of the era of Bengali Renaissance.

It's estimated that 5% of the population live in slums. Although the number of slums are less than Mumbai, they are scattered all over Kolkata in between affluent areas giving the city a rustic and poverty driven look.

Hyderabad

In Hyderabad, housing in modern ages in the 21st century is more modernized and developed than it has been in the past. The housing sector in Hyderabad has relatively sophisticated infrastructure. and is suitable for gated communities and villas, as well as higher-standard flats and condominiums. Hyderabad is home to several skyscrapers, including The Botanika, Lodha Belezza, etc. Roughly 15% of population is living in slum presently. Many residential infrastructure companies are well-established in Hyderabad.

Chennai

In Chennai, Housing in the 21st century is also modernized and developed that the past. The real estate sector in Chennai has modernized and luxury infrastructure. The demographics of the city changed, new high-rise building were built, campus-style office parks sprouted, vast shopping centers started thrive, streets became crowded with new cars and gated expatriate housing estates emerged. Chennai is home to several skyscrapers, including Highliving District tower-H-SPR city, Anchorage, House of Hiranandani, Bayview House of Hiranandani, etc. Many residential infrastructure companies are well-established in Chennai. Roughly 26% of population is living in slum presently.

It is observed that in India there is privation of reliable economic method for price prediction of residential properties which results in inevitable way to trust the often-manipulated prices by middle person. The increasing purchasing power of subjects have led to increasing demand of property ultimately causing rise in prices of properties. However, the rate of fluctuation in prices should have a method for its traceability. Round the world, the ways such hedonic pricing method, multiple linear regression analysis, travel cost method, fuzzy logic system, AHP technique, ARIMA, ANN (Artificial Neural Network) techniques etc.

Objectives of the study-

The objective of the study is Primarily focused on building a linear regression model using various concept of econometrics, The objective of the study are to-

(1) Using the various concept of econometrics such as checking for multi-co-linearity using variance inflating factor(VIF), Then fitting the linear regression model using ordinary least square method build a predictive model;
(2) Using random forest methodology build a predictive model;
(3) Comparing the predictive mean of each city, and give an Insight about the costliest real estate in India;
(4) Comparing the R-square value of Linear regression model and the R-square value of Random forest model, which method build a accurate predictive model.

Review of Literature

Related Work

Monk, Tang and Whitehead (2010) examined the social and economic impact of housing in Scottish country. Investment in housing finance impacts the economy directly and indirectly. Housing finance investment impacts the employment, GDP, productivity and many other important factors. The study revealed that the housing is an important indicator for increasing the wealth of nations. It was concluded that Scottish housing policy objective is to improve the quality standard of housing as well as to increase the investment in house old sector. Bhalla, Arora and Gill (2009) examined the performance of housing sector as well as the problems and challenges faced by this sector. The study showed that due to continuous changes in the global financial environment banks and financial institutions have brought sea changes in their strategies related to this sector so that slowly and gradually growth is shown by this sector. It was revealed that due to globalization process, India is witnessing competition among banks that has reduced the cost of finance for housing users.

Bhalla (2008) in a paper discussed the current scenario, development, performance, problems, challenges and prospects of housing finance as an industry segment. According to this study housing finance grow at the rate of 36 per cent. With the changes in strategies of banks and financial institution policies there is shift from buyer market to seller market. Dr. Haripriya Gundimeda studied the applicability of HPM to value water resources such as Bays, lakes and reservoirs, building of a new harbour, river views, restoration of urban stream, noise, landfills, dumping sites etc. on nearby property values. In India, hedonic price method has been employed in evaluating the relation between land prices and surface and ground water access (both in quality and quantity), (Gundimeda and Kathuria, 2005) and benefits if air quality improvement in India (Murty and Gulati, 2006). Kanojia Anita (2016) stated that the presence of Environmental Services such as parks are connected with additional environmental qualities. For example, Playground & open parks are often associated with higher air quality and lower noise levels and the presence of water can positively influence the climate of the surrounding areas. Therefore, the estimation of the capitalization of such “Services” in house prices might be biased by price effects of additional environmental qualities. It is worth noting that a park’s shape and area also have a significant effect on neighbourhood residential property values. For this reason,

in the future, perspective of landscape ecology, including the landscape quality, diversity, and fragmentation should also be considered. This study can provide effective information for real estate developers, government (in terms of decision-making on environmental tax), urban and landscape planners or architects, and green space conservationists and managers. Nevertheless, in future planners have to consider such Environmental Services, besides their ecological benefits, as a source of utility for the inhabitants of cities. The travel cost model is often used to measure the benefits provided by access to public recreation sites, e.g., national parks and national forests, which have relatively minor, if any, entrance fees (Oh, et al., 2005). Hotelling (1947) is credited with the initial development mof the travel cost model. Using the travel cost model, observed travelers' net economic benefit, or consumer's surplus, from visiting a recreation site is calculated as the value of access to the recreation site less the travel cost and necessary entrance fees (Heberling and Templeton, 2009). The model assumes that people travel to a recreation site if the marginal value of accessing the site is at least as large as the marginal cost of traveling to the site. The estimated consumer surplus is often used as a monetary measure of consumer welfare. The aggregate net economic benefit of access to a recreation site is estimated by aggregating average individual consumer surplus per visit over all visits. Eric Slone et.al. (2014) developed the relationships between various home characteristics and the asking price of a residential property were analysed using both a simple linear regression and a multiple linear regression using the method of ordinary least squares. Home square footage was utilized as the explanatory variable in the simple linear regression, and the multiple linear regression consisted of the addition of land parcel size, number of bedrooms, year of construction, and other explanatory variables. The results of the multiple linear regression proved the bias due to the omission of crucial factors in the simple linear regression. Home square footage was found to be the most important factor in the determination of residential property price, while garage capacity proved to be the weakest factor Ezgi Candas et.al (2015) had found that if significance level is accepted as 0.05 all the 5 variables in the last regression model (Floor, Heating system, Earthquake Zone, Rental Value and Land Value) have a significant impact on the dependent variable Value. Land value and rental value have the highest impact on the housing price. Existing floor, heating system and earthquake zone are the following them. Although it is found that the other variable is not significant in this study, it can change according to the sample size. If sample size increases, regression model once again is recommended for further studies. The

application of multiple regression analysis in a house data set explains or model's variation in house price which demonstrated good examples of strategic application of mathematical tool to aid analysis hence decision making in property investment. There are quite a number of data analysis techniques that are being employed in the property pricing research domain and they range from econometrics (e.g., ARIMA, linear regression) to Artificial Intelligence (AI) (e.g., artificial neural network, fuzzy logic) (Pagourtzi et al., 2003; Brooks and Tsolacos, 2010). Researchers have investigated the applications of these techniques in different research fields and also in real estate. Limsomuchai (2004) compared the predictive power of the hedonic price model with and artificial neural network mode on house price prediction. Artificial neural network models and hedonic price models are tested for their predictive power using 200 houses information in Christchurch, New Zealand. The results from hedonic price models support the previous findings. Even if the R2 of hedonic price models are high (higher than 75%) for in sample forecast, the hedonic price models do not outperform neural network models. Moreover, the hedonic price models show poorer results on out-of-sample forecast, especially when comparing with the neural network models. Thus, the empirical evidence presented in this paper supports the potential of neural network on house price prediction, although previous literatures have commented upon its black box nature and reached different conclusions. The non-linear relationship between house attributes and house, price, the lack of some environmental attributes, and inadequate number of sample size could be the cause of poor performance of the hedonic price models. However, it should be noted that cause of the optimal artificial neural network model is created by a trail-and error strategy. Without this strategy, the results may not indicate superiority of the neural network model.

Hujia Yu, Jiafu Wu (2016) lasso regression model can provide insights about chosen features, which is helpful in helping us understanding the correlations of house features and its sale prices. According to analysis, living area square feet, material of the roof, and neighbourhood have the greatest statistical significance in predicting a house's sale price. Azme Bin Khamis et.al. (2014) compared the performance between Multiple Linear Regression (MLR) model and Neural Network model on estimate house prices in New York. A sample of 1047 houses is randomly selected and retrieved from the Math10 website. The factors in prediction house prices including living area, number of bedrooms, number of bathrooms, lot size and age of house. The methods used in this study are MLR and Artificial Neural Network. It was found that, the value of R2 in

Neural Network model is higher than MLR model by 26.475%. The value of Mean Squared Error (MSE) in Neural Network model also lower compared to MLR model. The fuzzy logic system (FLS) is one of such technique, a multi-criteria decision-making tool. The application of FLS in articles published between 1994 and 2014 was reviewed by Mardani et al. (2015). It was established that the technique is being employed in the engineering, management and business and science and technology fields of studies. However, it was found that the majority of the articles were published in 2013, the authors of the articles reviewed adopted a hybrid form of the technique and the Analytic Hierarchy Process (AHP) is the most combined technique. In addition, the results of the study show that a majority of the authors are engineers, while most of the articles originated from Taiwan (for instance, Lee et al., 2008; Lu and Wang, 2011; Chou and Cheng, 2012). The AHP technique is another multi-criterion decision-making technique that has been in the real estate research domain for a while. The study of F.Zahedi (1986) established that AHP is applicable in different fields which include economic and planning, conflict resolution, manufacturing, portfolio selection, accounting and auditing. Others are auditing, education, politics and environment marketing, amongst others. Vaidya and Kumar (2006) examined 150 articles that adopted AHP, but eventually reviewed 27 of these articles. In addition to the exposition of the applications of the technique that is similar to that of F.Zahedi (1986), it was discovered that most of the authors adopted the technique for variable selections. The United States was identified as the country where most of the articles emanated from, while a continuous trend of application of the techniques was noticed in developing countries, especially in India. The Artificial Neural Network (ANN) is an AI technique that has gained widespread popularity in different research areas as well and has proven to be highly efficient for property pricing appraisal research (Mora-Esperanza, 2004). Widrow et al. (1994) reviewed the applications of the tool in industry, business and science disciplines. The authors presented the classification of the technique in the areas of linear applications (telecommunications, sound and vibration control), multi-element nonlinear applications (credit card fraud detection, cursive hand writing recognition, loan approval, real estate analysis and marketing analysis, amongst others) and nonlinear applications on the horizon (automotive, speech recognition and mass spectra classification, amongst others). This is like the study of Paliwal and Kumar (2009) who reviewed and classified articles that applied the ANN technique in accounting and finance, health and medicine, marketing, engineering and manufacturing and general application fields.

Source of the data

To study the Real estate Prices in the metropolitan areas of India, secondary dataset has been used.

Which has Data of six metro cities namely Mumbai, Hyderabad, Chennai, Bangalore, Kolkata, Delhi.

Content

This dataset comprises data that was scraped. It includes:

- collection of prices of new and resale houses located in the metropolitan areas of India
- the amenities provided for each house

Inspiration

With 40 explanatory variables describing various aspects of new and resale houses in the metropolitan areas of India, one can predict the final price of houses in these regions.

The dataset has been collected directly from the kaggle website:

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

Methodology

To analysis the dataset, I have used the following Tools, which are written in brief are given below:

(1) Jupyter Notebook - Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating Jupyter notebook documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context. A Jupyter Notebook document is a JSON document, following a versioned schema, containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.

A Jupyter Notebook can be converted to a number of open standard output formats (HTML, presentation slides, LaTeX, PDF, ReStructuredText, Markdown, Python) through "Download As" in the web interface, via the nbconvert library or "jupyter nbconvert" command line interface in a shell. To simplify visualisation of Jupyter notebook documents on the web, the nbconvert library is provided as a service through NbViewer[11] which can take a URL to any publicly available notebook document, convert it to HTML on the fly and display it to the user.

(2) Python - Python is an interpreted high-level general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

(3) Pandas - pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals. Its name is a play on the phrase "Python data analysis" itself. Wes McKinney started building what would become pandas at AQR Capital while he was a researcher there from 2007 to 2010.[8]

Library features

- DataFrame object for data manipulation with integrated indexing.
- Tools for reading and writing data between in-memory data structures and different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and subsetting of large data sets.
- Data structure column insertion and deletion.
- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.
- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
- Time series-functionality: Date range generation and frequency conversions, moving window statistics, moving window linear regressions, date shifting and lagging.
- Provides data filtration.

The library is highly optimized for performance, with critical code paths written in Cython or C.

Data frame

Pandas is mainly used for data analysis. Pandas allows importing data from various file formats such as comma-separated values, JSON, SQL, Microsoft Excel. Pandas allows various data manipulation operations such as merging, reshaping, selecting, as well as data cleaning, and data wrangling features.

(4) Numpy - NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors. NumPy targets the CPython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays; using these requires rewriting some code, mostly inner loops, using NumPy.

Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available; SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that provides MATLAB-like plotting functionality. Internally, both MATLAB and NumPy rely on BLAS and LAPACK for efficient linear algebra computations.

Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array as universal data structure in OpenCV for images, extracted feature points, filter kernels and many more vastly simplifies the programming workflow and debugging.

(5) Statsmodels- Statsmodels is a Python package that allows users to explore data, estimate statistical models, and perform statistical tests. An extensive list of descriptive statistics, statistical tests, plotting functions, and result statistics are available for different types of data and each estimator. It complements SciPy's stats module.

Statsmodels is part of the Python scientific stack that is oriented towards data analysis, data science and statistics. Statsmodels is built on top of the numerical libraries NumPy and SciPy, integrates with Pandas for data handling, and uses Patsy for an R-like formula interface. Graphical functions are based on the Matplotlib library. Statsmodels provides the statistical backend for other Python libraries. Statmodels is free software released under the Modified BSD (3-clause) license

(6) Matplotlib- **Matplotlib** is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.

Matplotlib was originally written by John D. Hunter. Since then it has an active development community and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012 and was further joined by Thomas Caswell.

Matplotlib 2.0.x supports Python versions 2.7 through 3.10. Python 3 support started with Matplotlib 1.2. Matplotlib 1.4 is the last version to support Python 2.6. Matplotlib has pledged not to support Python 2 past 2020 by signing the Python 3 Statement.

(7) Seaborn - Seaborn is a library for making statistical graphics in Python. It is built on top of matplotlib and is tightly integrated with the PyData stack, including support for numpy and pandas data structures, and statistical routines from scipy and statsmodels. Some of the features that seaborn offers are:

- Several built-in themes that change the default matplotlib plain aesthetics
- Tools for choosing diverse color palettes to make beautiful plots
- Functions for visualizing univariate and bivariate distributions or for comparing them between subsets of data
- Tools that fit and visualize linear regression models for different kinds of independent and dependent variables
- A function to plot statistical timeseries data with flexible estimation and representation of uncertainty around the estimate
- High-level abstractions for structuring grids of plots that let you easily build complex visualizations

(8) Scikit-learn - **Scikit-learn** (formerly **scikits.learn** and also known as **sklearn**) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Scikit-learn is largely written in Python, and uses NumPy extensively for high-performance linear algebra and array operations. Furthermore, some core algorithms are written in Cython to improve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR. In such cases, extending these methods with Python may not be possible.

Scikit-learn integrates well with many other Python libraries, such as Matplotlib and plotly for plotting, NumPy for array vectorization, Pandas dataframes, SciPy, and many more.

(9) Random forest- **Random forests** or **random decision forests** are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set.[Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

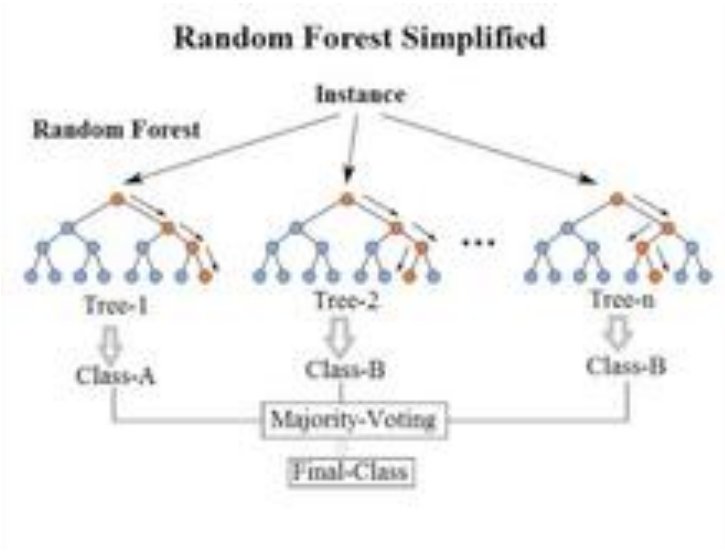
The first algorithm for random decision forests was created in 1995 by Tin Kam Ho using the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.

An extension of the algorithm was developed by Leo Breiman and Adele Cutler, who registered"Random Forests" as a trademark in 2006 (as of 2019, owned by Minitab, Inc.). The extension combines Breiman's "bagging" idea and random selection of features, introduced first by Ho and later independently by Amit and Geman in order to construct a collection of decision trees with controlled variance.

Random forests are frequently used as "blackbox" models in businesses, as they generate reasonable predictions across a wide range of data while requiring little configuration.

Decision trees are a popular method for various machine learning tasks. Tree learning "come[s] closest to meeting the requirements for serving as an off-the-shelf procedure for data mining", say Hastie *et al.*, "because it is invariant under scaling and various other transformations of feature values, is robust to inclusion of irrelevant features, and produces inspectable models. However, they are seldom accurate".In particular, trees that are grown very deep tend to learn highly irregular patterns: they overfit their training sets, i.e. have low bias, but very high variance. Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance. This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model.

Forests are like the pulling together of decision tree algorithm efforts. Taking the teamwork of many trees thus improving the performance of a single random tree. Though not quite similar, forests give the effects of a K-fold cross validation.



(10) SPSS- **SPSS Statistics** is a software package used for interactive, or batched, statistical analysis. Long produced by SPSS Inc., it was acquired by IBM in 2009. Current versions (post 2015) have the brand name: **IBM SPSS Statistics**.

The software name originally stood for **Statistical Package for the Social Sciences** (**SPSS**), reflecting the original market, then later changed to **Statistical Product and Service Solutions**.

SPSS is a widely used program for statistical analysis in social science.[5] It is also used by market researchers, health researchers, survey companies, government, education researchers, marketing organizations, data miners,[6] and others. The original SPSS manual (Nie, Bent & Hull, 1970)[7] has been described as one of "sociology's most influential books" for allowing ordinary researchers to do their own statistical analysis.[8] In addition to statistical analysis, data management (case selection, file reshaping, creating derived data) and data documentation (a metadata dictionary is stored in the datafile) are features of the base software.

(11) Tableau- **Tableau Software** is an American interactive data visualization software company focused on business intelligence. It was founded in 2003 in Mountain View, California, and is currently headquartered in Seattle, Washington.[6] In 2019 the company was acquired by Salesforce for $15.7 billion. CNBC reported that this acquisition is the largest acquisition by Salesforce, which is considered the strongest in the CRM field, since its foundation.

The company's founders, Christian Chabot, Pat Hanrahan and Chris Stolte, were researchers at the Department of Computer Science at Stanford University They specialized in visualization techniques for exploring and analyzing relational databases and data cubes, and started the company as a commercial outlet for research at Stanford from 1999 to 2002.

Tableau products query relational databases, online analytical processing cubes, cloud databases, and spreadsheets to generate graph-type data visualizations. The software can also extract, store, and retrieve data from an in-memory data engine.

Tableau has a mapping functionality, and is able to plot latitude and longitude coordinates and connect to spatial files like Esri Shapefiles, KML, and GeoJSON to display custom geography. The built-in geo-coding allows for administrative places (country, state/province, county/district), postal codes, US Congressional Districts, US CBSA/MSA, Area Codes, Airports, and European Union statistical areas (NUTS codes) to be mapped automatically. Geographies can be grouped to create custom territories or custom geocoding used to extend existing geographic roles in the product.

The procedure to analysis the data-set has been classified into following ways-

(1) Exploratory Data analysis(EDA)

(2) Feature Engineering

(3) Feature selection

(4) Model creation and deployment, Hyper parameter tuning, Prediction

(5) Using the SPSS software, we draw the ANOVA table for further interpretation about the significance of the model.

(6) Using the Tableau software we will carry out the necessary graph and plots for Final interpretation about the prediction mean and mean absolute error.

Exploratory data analysis(EDA)

Step-1 : Importing all the libraries such as pandas, Numpy, statsmodels, matplotlib, seaborn, sklearn etc.

For eg:

Code: #Importing libraries

```
import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

import statsmodels.api as sm
```

Step-2:Import the dataset using pandas library.

For eg:

Code: #Importing Dataset using Pandas

```
Mumbai_Dataset = pd.read_csv('Mumbai.csv')

Mumbai_Dataset
```

Output:

	Price	Area	Location	No. of Bedrooms	Resale	MaintenanceStaff	Gymnasium
0	4850000	720	Kharghar	1	1	1
1	4500000	600	Kharghar	1	1	1
2	6700000	650	Kharghar	1	1	1
3	4500000	650	Kharghar	1	1	1
4	5000000	665	Kharghar	1	1	1
...	...	...	...	...	...	...
7714	14500000	1180	Mira Road East	2	0	9
7715	14500000	530	Naigaon East	1	1	9
7716	4100000	700	Shirgaon	1	0	9
7717	2750000	995	Mira Road East	2	0	9
7718	2750000	1020	Mira Road East	2	0	9
7719 rows × 40 columns							

This Table is the short version of the original Table.

Step-3: Now finding out all the information about the data-set using the python code and calculating all the descriptive statistics of the data using python code.

For eg:

Code: Mumbai_Dataset.describe()

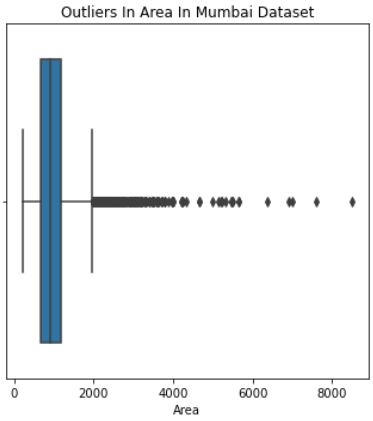
Output:

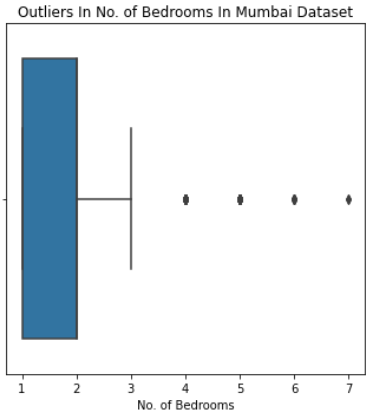
	Price	Area	No. of Bedrooms	Resale	MaintenanceStaff	Gymnasium
count	7.72E+03	7719	7719	7719	7719
mean	1.51E+07	998.40925	1.913331	0.647105	7.498899
std	2.05E+07	550.967809	0.855376	0.477901	3.197923
min	2.00E+06	200	1	0	0
25%	5.30E+06	650	1	0	9
50%	9.50E+06	900	2	1	9
75%	1.70E+07	1177	2	1	9
max	4.20E+08	8511	7	1	9

This Table is short version of the original Table.

Step-4: Using the visualization tool i.e. matplotlib and seaborn to visualize the outliers in the data-set.

For eg:





Feature Engineering

Step-1: Using the Python code handling the missing value and Duplicate value.

For eg: (1) Missing value python code

Code:

```
new_value = float("Nan")

#missing values are represented with '9' in this dataset

Mumbai.replace(to_replace =9 ,value = new_value, inplace= True)

Mumbai = Mumbai.dropna()

Mumbai.shape
```

(2) Duplicate value python code

Code:

```
Mumbai.duplicated().sum()

Mumbai.drop_duplicates(inplace=True)

#size after removing duplicates

Mumbai.shape
```

Step-2: A key metric in the housing industry, at least in India, is "Price Per Square Feet (Price/Area)", expressed as Rupees Per Square Feet. Almost everyone uses this metric to compare the prices as it eliminates the impact of house size and provides an easy comparison. Let us add a column that captures Price Per Square Feet. We will call this "PPSF" in short

For eg:

Code: Mumbai['PPSF'] = Mumbai['Price'] / Mumbai['Area']

Step-3:Handling the outliers using python code, for that we use distribution plot to check outlier.

For eg:

Code:

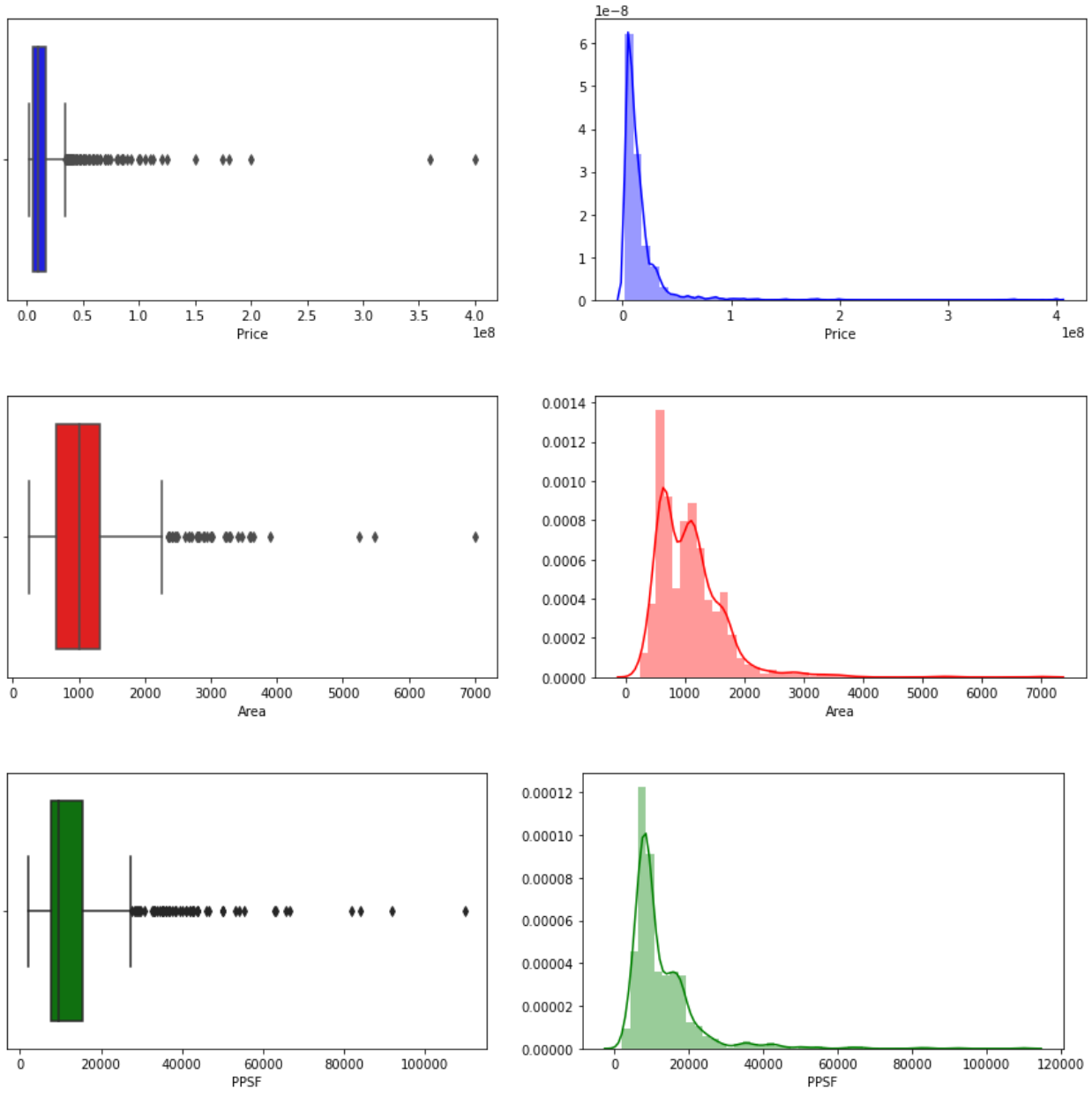
```
def  dist_plot (feature, color, position=121):
      plt.figure(figsize=(15,4))
   plt.subplot(position)
   sns.boxplot(x=feature, data=Mumbai, color=color)
   plt.subplot(position+1)
   sns.distplot(Mumbai[feature], color=color)
```

dist_plot('Price', 'blue')

dist_plot('Area', 'red')

dist_plot('PPSF', 'green')

Output:



As we have seen above, the data is **heavily right-skewed** for Price and Area both. One would expect that PPSF would be normally distributed as it is a ratio, however it is not. This indicates either or both of the following:

1. There is much more skewness in Price compared to Area
2. The houses with high Prices are not necessarily the ones with the large Area, hence PPSF is not able to normalize such cases

We now proceed to **remove outliers** so that they don't have a lopsided impact on our model. As the data is heavily right skewed, we remove a higher percentile (10%) from the top versus bottom(5%)

Now we create filters to handle the outliers for that we use python code

For eg:

Code: # define outliers

ppsf_outliers = np.percentile(Mumbai.PPSF, [5,90])

price_outliers = np.percentile(Mumbai.Price, [5,90])

area_outliers = np.percentile(Mumbai.Area, [5,90])

create filters based on outliers

ppsf_filter = (Mumbai.PPSF > ppsf_outliers[0]) & (Mumbai.PPSF < ppsf_outliers[1])

price_filter = (Mumbai.Price > price_outliers[0]) & (Mumbai.Price < price_outliers[1])

area_filter = (Mumbai.Area > area_outliers[0]) & (Mumbai.Area < area_outliers[1])

apply filters

Mumbai = Mumbai[(ppsf_filter) & (price_filter) & (area_filter)]

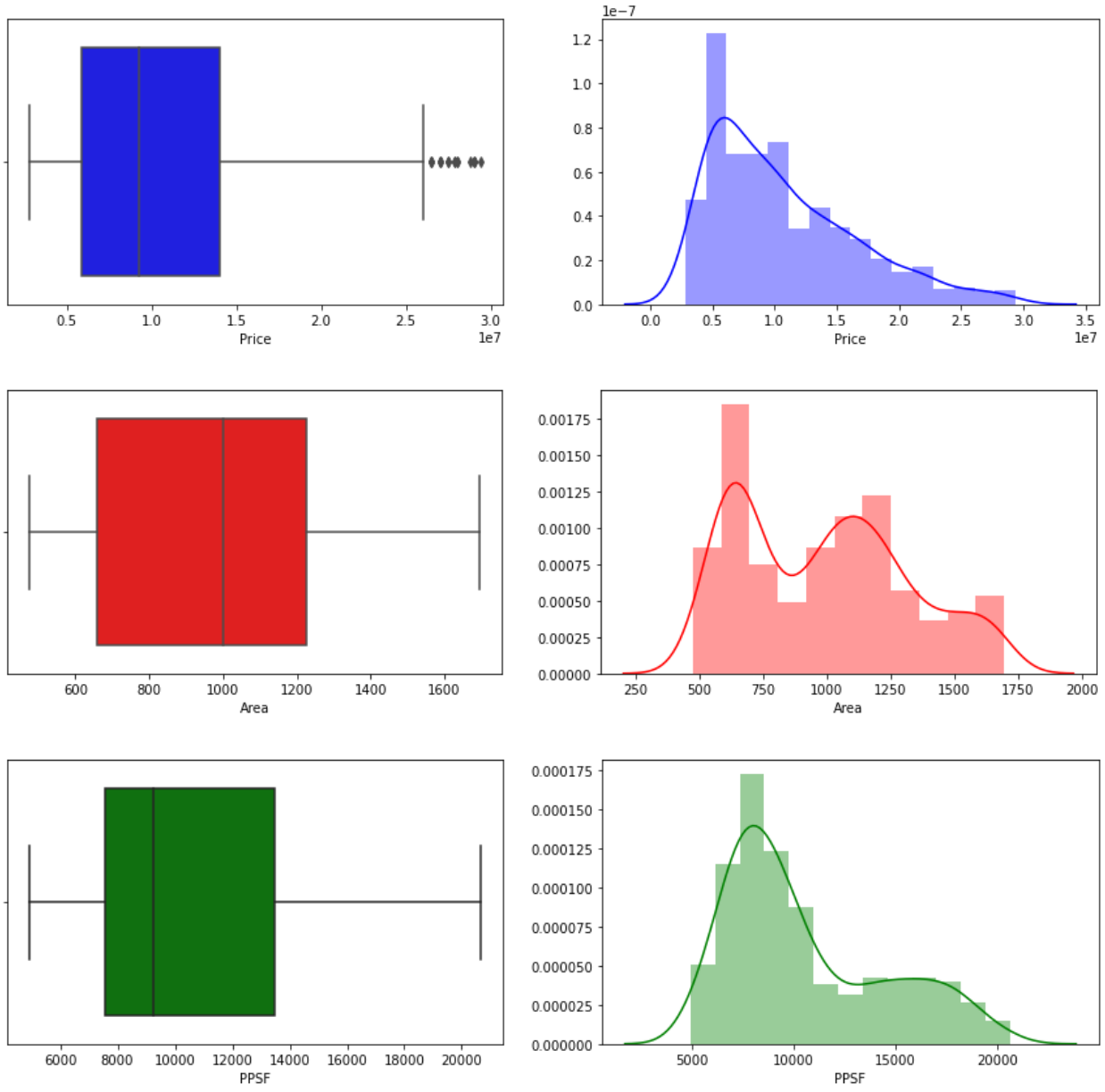
Now we use distribution plot to check the outliers

For eg: dist_plot('Price', 'blue')

dist_plot('Area', 'red')

dist_plot('PPSF', 'green')

Output:



Now the outliers has been removed which can be seen in the above plots.

Step-4: Let us now check the features available to us and also clean up the names so that they do not cause any problems later

For eg:

Code: Mumbai.shape, Mumbai.columns

```
((842, 41),
 Index(['Price', 'Area', 'Location', 'No. of Bedrooms', 'Resale',
        'MaintenanceStaff', 'Gymnasium', 'SwimmingPool', 'LandscapedGardens',
        'JoggingTrack', 'RainWaterHarvesting', 'IndoorGames', 'ShoppingMall',
        'Intercom', 'SportsFacility', 'ATM', 'ClubHouse', 'School',
        '24X7Security', 'PowerBackup', 'CarParking', 'StaffQuarter',
        'Cafeteria', 'MultipurposeRoom', 'Hospital', 'WashingMachine',
        'Gasconnection', 'AC', 'Wifi', 'Children'splayarea', 'LiftAvailable',
        'BED', 'VaastuCompliant', 'Microwave', 'GolfCourse', 'TV',
        'DiningTable', 'Sofa', 'Wardrobe', 'Refrigerator', 'PPSF'],
       dtype='object'))
```

Two columns seem to have problematic names. Let's rename them

For eg:

Code: Mumbai.rename(columns={'No. of Bedrooms':'Bedrooms',
"Children'splayarea":'PlayArea'}, inplace=True)

As we can see from the columns list, there are a lot of features, most of them binary. Most of the binary features will fit under the broad category of **Amenities**. In the following section I have tried to calculate feature scores for each house based on the Amenities being absent or present.

Note: Important - I have assigned weights to binary features based on my discretion. I have also used a scale of 0-4 (0 being least important and 4 being most important).

For eg:

Code: # assign weights to features

```
feature_dict = {'MaintenanceStaff':2, 'Gymnasium':4, 'SwimmingPool':4,'LandscapedGardens':3,
'JoggingTrack':3, 'RainWaterHarvesting':2,'IndoorGames':3, 'ShoppingMall':2, 'Intercom':2,
'SportsFacility':3, 'ATM':2, 'ClubHouse':2, 'School':2, '24X7Security':1, 'PowerBackup':4,
'CarParking':3, 'StaffQuarter':0, 'Cafeteria':0, 'MultipurposeRoom':2, 'Hospital':3,
'WashingMachine':0, 'Gasconnection':2, 'AC':0, 'Wifi':0, 'PlayArea':3, 'LiftAvailable':0, 'BED':0,
'VaastuCompliant':0, 'Microwave':0, 'GolfCourse':0, 'TV':0, 'DiningTable':0, 'Sofa':0,
'Wardrobe':0, 'Refrigerator':0}
```

For eg:

Code: # convert to Dataframe

```
features = pd.DataFrame(feature_dict.items(), columns=['Features', 'Weight'])


features.head()
```

Output:

	Features	weight
0	MaintenanceStaff	2
1	Gymnasium	4
2	SwimmingPool	4
3	LandscapedGardens	3
4	JoggingTrack	3

For eg:

Code: # The features matrix has 35 features

```
features.shape
```

Output:

(35, 2)

To arrive at Feature Score for each house, we need to calculate dot product between the features matrix and the features binaries. Before this, we need to **subset a features matrix** from the Mumbai dataset

For eg:

Code:

```
features_matrix = Mumbai[['MaintenanceStaff', 'Gymnasium', 'SwimmingPool',
    'LandscapedGardens', 'JoggingTrack', 'RainWaterHarvesting',
    'IndoorGames', 'ShoppingMall', 'Intercom', 'SportsFacility', 'ATM',
    'ClubHouse', 'School', '24X7Security', 'PowerBackup', 'CarParking',
    'StaffQuarter', 'Cafeteria', 'MultipurposeRoom', 'Hospital',
    'WashingMachine', 'Gasconnection', 'AC', 'Wifi', 'PlayArea',
```

'LiftAvailable', 'BED', 'VaastuCompliant', 'Microwave',

'GolfCourse', 'TV', 'DiningTable', 'Sofa', 'Wardrobe',

'Refrigerator']]

features_matrix.tail()

Output:

	MaintenanceStaff	Gymnasium	SwimmingPool	LandscapedGardens	JoggingTrack
1390	1	1	1	1	1
1392	0	1	1	1	1
1395	0	1	1	1	1
1396	0	0	0	0	0
1397	0	1	1	1	1
5 rows × 35 columns					

This table is the short version of the original Table.

For eg:

Code: features_matrix.shape

Output:(842, 35)

We will now compute **dot product** between the "features_matrix" **(shape = 842 x 35)** and 'Weight' column of the "features" dataframe **(shape = 35 x 1)**. This will result in a column vector of shape **842 x 1** and is nothing but the FeatureScore for each house. We will then join this resulting vector with the Mumbai dataframe

For eg:

Code: # apply dot product to compute feature score for every row

feature_scores_df = np.dot(features_matrix, features['Weight'])

join feature scores column with Mumbai dataframe

Mumbai['FeatureScore'] = feature_scores_df

For eg:

Code: Mumbai.head(3)

Output:

	Price	Area	Location	Bedrooms	Resale	PPSF	Feature score
0	4850000	720	Kharghar	1	6736.11111	10
1	4500000	600	Kharghar	1	7500	25
2	6700000	650	Kharghar	1	10307.6923	30
3 rows × 42 columns							

This table is the short version of the original Table.

We have now reduced the number of features on which to run our regression model. These are:

1. Area (numeric)
2. Location (string).
3. Bedrooms (numeric)
4. Feature Score (numeric)
5. Resale (boolean)

Step-5: Now using the python code we can model Location as a numeric feature

For eg:

Code: # Create a pivot table of Locations with PPSF as the value

location_pivot = pd.pivot_table(data=Mumbai, index='Location', aggfunc='mean', values='PPSF')

location_pivot

Output:

Location	PPSF
Airoli	11100.73034
Ambernath East	5306.849315
Andheri	12434.13729
Andheri East	14877.11884

Badlapur West	5149.65035
...	...
no 9	19259.25926
raheja vihar	18351.7094
taloja panchanand	6948.397998
thakur village kandivali east	17458.79121
vasant vihar thane west	12648.22134
171 rows × 1 columns	

Location is one of the most important indicator of house price. As a bit of feature engineering, we shall try to calculate the 'Location Premium' for every Location. This is nothing but the PPSF for every location divided by the minimum PPSF. This sets the cheapest location as the base location (with a score of 1) and every other location has a premium as a multiple of that base location

For eg:

Code:

```
location_pivot['LocationPremium'] = location_pivot['PPSF'] / location_pivot['PPSF'].min()

 location_pivot.sort_values('LocationPremium', ascending=False)
```

Output:

Location	PPSF	Location premium
Vivek Vidyalaya Marg	20000	4.060606
no 9	19259.2593	3.910213
Samata Nagar Thakur Village	19259.2593	3.910213
Mahatma Gandhi Road	19200	3.898182
raheja vihar	18351.7094	3.725953
...	...	...

Ambernath East	5306.84932	1.077451
Vasai east	5197.13262	1.055175
Taloja	5160.79494	1.047798
Badlapur West	5149.65035	1.045535
Koproli	4925.37313	1
171 rows × 2 columns		

We now merge the location pivot with the Mumbai dataframe on 'Location' column

For eg:

Code:

Mumbai = pd.merge(Mumbai, location_pivot['LocationPremium'], on='Location')

As a final step we calculate the **log** of LocationPremium. I observed that it yields better results than simply using LocationPremium

For eg:

Code:

Mumbai['LogPremium'] = np.log(Mumbai['LocationPremium'])

check if all required columns are present

Mumbai.head(3)

Output:

	Price	Area	Location	Bedrooms	Resale	PPSF	FeatureScore	LocationPremium	LogPremium
0	4850000	720	Kharghar	1	6736.111	10	1.73745	0.552419	
1	4500000	600	Kharghar	1	7500	25	1.73745	0.552419	
2	6700000	650	Kharghar	1	10307.69	30	1.73745	0.552419	
3 rows × 44 columns									

Step-6: Now Convert categorical variable into dummy/indicator variables

For eg:

Code: print(list(Mumbai.columns))

```
Mumbai = pd.get_dummies(Mumbai)

Mumbai.shape, Mumbai.columns
```

['Price', 'Area', 'Location', 'Bedrooms', 'Resale', 'MaintenanceStaff', 'Gymnasium',
'SwimmingPool', 'LandscapedGardens', 'JoggingTrack', 'RainWaterHarvesting', 'IndoorGames',
'ShoppingMall', 'Intercom', 'SportsFacility', 'ATM', 'ClubHouse', 'School', '24X7Security',
'PowerBackup', 'CarParking', 'StaffQuarter', 'Cafeteria', 'MultipurposeRoom', 'Hospital',
'WashingMachine', 'Gasconnection', 'AC', 'Wifi', 'PlayArea', 'LiftAvailable', 'BED',
'VaastuCompliant', 'Microwave', 'GolfCourse', 'TV', 'DiningTable', 'Sofa', 'Wardrobe',
'Refrigerator', 'PPSF', 'FeatureScore', 'LocationPremium', 'LogPremium']

Output: ((842, 214),
Index(['Price', 'Area', 'Bedrooms', 'Resale', 'MaintenanceStaff', 'Gymnasium',
'SwimmingPool', 'LandscapedGardens', 'JoggingTrack',
'RainWaterHarvesting',
...
'Location_West Amardeep Colony',
'Location_Western Express Highway Kandivali East', 'Location_kandivali',
'Location_kolshet', 'Location_mumbai', 'Location_no 9',
'Location_raheja vihar', 'Location_taloja panchanand',
'Location_thakur village kandivali east',
'Location_vasant vihar thane west'],
dtype='object', length=214))

Feature selection

Step-6: Handling the Multicollinearity

Heatmap for understanding correlation

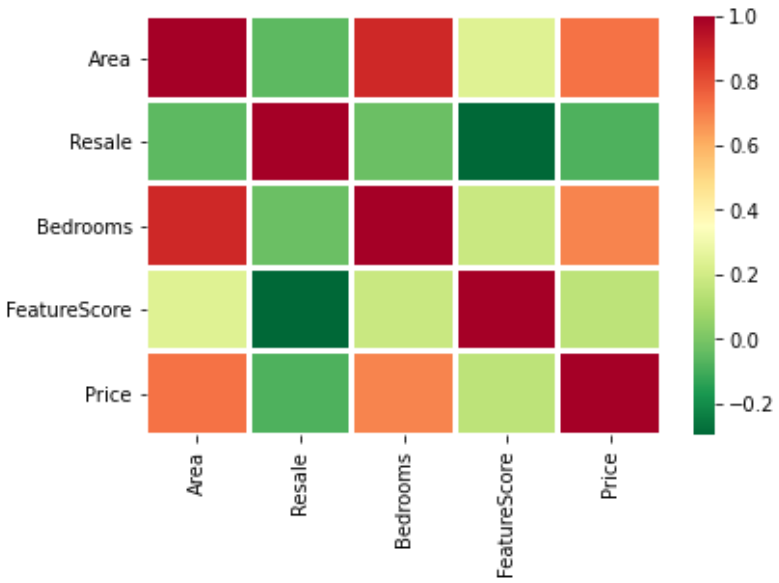
For eg:

Code: corr_df = Mumbai[['Area', 'Resale', 'Bedrooms', 'FeatureScore', 'Price']]

sns.heatmap(corr_df.corr(method='pearson'), cmap='RdYlGn_r', linewidths=2)

Output:

```
<AxesSubplot:>
```



Step-7: Calculating VIF using python code

Before calculating VIF, let us first talk about VIF(variance inflating factor)

A variance inflation factor(VIF) detects multicollinearity in regression analysis. Multicollinearity is when there's correlation between predictors (i.e. independent variables) in a model; it's presence can adversely affect your regression results. The VIF estimates how much the variance of a regression coefficient is inflated due to multicollinearity in the model.
VIFs are usually calculated by software, as part of regression analysis. You'll see a VIF column as part of the output. VIFs are calculated by taking a predictor, and regressing it against every other predictor in the model. This gives you the R-squared values, which can then be plugged into the VIF formula. "i" is the predictor you're looking at (e.g. x_1 or x_2):

$$\mathrm{VIF} = \frac{1}{1 - R_i^2}$$

Interpreting the Variance Inflation Factor

Variance inflation factors range from 1 upwards. The numerical value for VIF tells you (in decimal form) what percentage the variance (i.e. the standard error squared) is inflated for each coefficient. For example, a VIF of 1.9 tells you that the variance of a particular coefficient is 90% bigger than what you would expect if there was no multicollinearity — if there was no correlation with other predictors.

A **rule of thumb** for interpreting the variance inflation factor:

- 1 = not correlated.
- Between 1 and 5 = moderately correlated.
- Greater than 5 = highly correlated.

Exactly how large a VIF has to be before it causes issues is a subject of debate. What is known is that the more your VIF increases, the less reliable your regression results are going to be. In general, a VIF above 10 indicates high correlation and is cause for concern. Some authors suggest a more conservative level of 2.5 or above.

Sometimes a high VIF is no cause for concern at all. For example, you can get a high VIF by including products or powers from other variables in your regression, like x and x^2. If you have high VIFs for dummy variables representing nominal variables with three or more categories, those are usually not a problem.

For eg:

Calculating VIF using python code

Code:from sklearn.preprocessing import StandardScaler as ss

#Scaling the data set

scalar=ss()

Y=Mumbai['Price']

Scaling

X=scalar.fit_transform(Mumbai.drop(columns=['Price']))

Converting to pandas dataframe for easy manipulation

X=pd.DataFrame(data=X,columns=Mumbai.drop(columns=['Price']).columns)

X.head()

Output:

Area	Bedrooms	Resale	MaintenanceStaff			Location_thakur village kandivali east
0	-0.813497	-1.194214	0.502967		...	-0.048795
1	-1.176893	-1.194214	0.502967		...	-0.048795
2	-1.025478	-1.194214	0.502967		...	-0.048795
3	-1.025478	-1.194214	0.502967		...	-0.048795
4	-0.980053	-1.194214	0.502967		...	-0.048795
5 rows × 213 columns						

This table is the short version of the original Table.

For eg:

Code: # Checking multicollinearity

```
k=X.corr()

z=[[str(i),str(j)] for i in k.columns for j in k.columns if (k.loc[i,j]>abs(0.5)) & (i!=j)]

z,len(z)
```

Output:

```
([['Area', 'Bedrooms'],
  ['Bedrooms', 'Area'],
  ['Gymnasium', 'SwimmingPool'],
  ['Gymnasium', 'ClubHouse'],
  ['Gymnasium', 'FeatureScore'],
  ['LocationPremium', 'PPSF'],
  ['LocationPremium', 'LogPremium'],
  ['LogPremium', 'PPSF'],
  ['LogPremium', 'LocationPremium'],
  ['Location_Kharghar', 'VaastuCompliant'],
  ['Location_Link Road', 'Microwave']],
 144)
```

Again this the short version of the original result

For eg:

Code:

```
#Caluclating VIF

from statsmodels.stats.outliers_influence import variance_inflation_factor as vif

VIF=pd.Series([vif(X.values,i) for i in range (X.shape[1])],index=X.columns)

VIF
```

Output:

```
Area                                  10.251693
Bedrooms                               9.339962
Resale                                 3.404899
MaintenanceStaff                            inf
Gymnasium                                   inf
                                        ...
Location_no 9                               inf
Location_raheja vihar                       inf
Location_taloja panchanand                  inf
Location_thakur village kandivali east      inf
Location_vasant vihar thane west            inf
Length: 213, dtype: float64
```

For eg:

Code:

```
# We need to remove vif > 5 and each time we remove one column the vif data changes so we remove,check vif,remove

def MC_rem(data):
    VIF=pd.Series([vif(data.values,i) for i in range (data.shape[1])],index=data.columns)
    if(VIF.max()>5):
        data.drop(columns=[VIF[VIF==VIF.max()].index[0]],inplace=True)
        print(VIF[VIF==VIF.max()].index[0],'has been removed from "X_copy"')
        return data
```

```
    else:
        print('no multicollinearity')
        return data
```

continue with code:

```
X_copy=X.copy()
for i in range(5):
    X_copy=MC_rem(X_copy)
X=X_copy
```

###After Removing collinearity

```
VIF=pd.Series([vif(X_copy.values,i) for i in range (X_copy.shape[1])],index=X_copy.columns)
VIF
```

Output:

```
Area                                  10.180599
Bedrooms                               9.310975
Resale                                 3.374238
Gymnasium                              7.029648
SwimmingPool                           6.592871
                                         ...
Location_no 9                          1.407113
Location_raheja vihar                  1.920898
Location_taloja panchanand             1.751822
Location_thakur village kandivali east 1.833091
Location_vasant vihar thane west       1.399950
Length: 208, dtype: float64
```

Using the above code we successfully remove the multi-co-linearity from the model

Step-8: Now we Remove Features that aren't important or redundant

For eg:

Code:

```
features = ['MaintenanceStaff', 'CarParking', 'Intercom',
        'Gymnasium', 'JoggingTrack', 'RainWaterHarvesting',
        'ShoppingMall', 'SportsFacility', 'ATM', 'ClubHouse', 'School',
        '24X7Security', 'StaffQuarter', 'Cafeteria', 'MultipurposeRoom',
        'Hospital', 'WashingMachine', 'Gasconnection', 'AC', 'Wifi', ''PlayArea'',
        'BED', 'VaastuCompliant', 'Microwave', 'GolfCourse', 'TV', 'DiningTable',
        'Sofa', 'Wardrobe', 'Refrigerator']


X = Mumbai.drop(Mumbai[features], axis=1)
```

Model creation and deployment, Hyper parameter tuning and Prediction

Step-1: We Split the Dataset into Test and Train dataset

Code:

```
from sklearn.model_selection import train_test_split

X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.3,random_state=101)

X_train.shape,X_test.shape,Y_train.shape,Y_test.shape
```

Output: `((589, 184), (253, 184), (589,), (253,))`

Step-2: We Defining parameters for hyper parameter tuning, which is an important part before building a model, which sets constraints on the parameter to obtain accuracy of the model.

Code: Hyper parameter tuning for Random forest methodology using Random search cv

```
#Number of trees in random forest

n_estimators = [int(x) for x in np.linspace(start=100, stop=1200, num=12)]
```

```
#Number of features to consider in every split

max_features = ['auto', 'sqrt']


#Maximum number of levels in a tree

max_depth = [int(x) for x in np.linspace(start=5, stop=30, num=6)]


#Minimum number of samples required to split a node

min_samples_split = [2, 5, 10, 15, 100]


#Minimum number of samples required at each leaf node

min_samples_leaf = [1, 2, 5, 10]

#Random Grid

random_grid = {'n_estimators' : n_estimators,

        'max_features' : max_features,

        'max_depth' : max_depth,

        'min_samples_split' : min_samples_split,

        'min_samples_leaf' : min_samples_leaf}
```

Step-3: We now build the linear regression model using ordinary least square method(OLS), which has been imported using Statsmodels library, and then we will formulate the summary from the model, which will give us all the necessary values such R-square, adjusted R-square, F-statistic etc.

Code:

```
x = Mumbai[['Area', 'FeatureScore', 'Resale','Bedrooms','LogPremium']]

y = Mumbai['Price']

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=19)

model = sm.OLS(y_train, x_train).fit()
```

model.summary()

Output:

OLS Regression Results

Dep. Variable:	Price	**R-squared (uncentered):**	0.953
Model:	OLS	**Adj. R-squared (uncentered):**	0.953
Method:	Least Squares	**F-statistic:**	2720.
Date:	Thu, 03 Jun 2021	**Prob (F-statistic):**	0.00
Time:	21:22:07	**Log-Likelihood:**	-10906.
No. Observations:	673	**AIC:**	2.182e+04
Df Residuals:	668	**BIC:**	2.184e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Area	7043.8083	666.201	10.573	0.000	5735.707	8351.909
FeatureScore	-5.342e+04	7900.886	-6.761	0.000	-6.89e+04	-3.79e+04
Resale	-3.027e+06	2.19e+05	-13.822	0.000	-3.46e+06	-2.6e+06
Bedrooms	8.726e+05	3.23e+05	2.701	0.007	2.38e+05	1.51e+06
LogPremium	8.599e+06	3.02e+05	28.502	0.000	8.01e+06	9.19e+06

Omnibus:	87.334	**Durbin-Watson:**	2.042
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	148.759
Skew:	0.821	**Prob(JB):**	4.98e-33
Kurtosis:	4.616	**Cond. No.**	3.44e+03

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The condition number is large, 3.44e+03. This might indicate that there are
strong multi-co-linearity or other numerical problems.

Note:

(1) Prediction mean - It is the mean or average of all the prediction values of the model.

(2) Mean absolute error- It is the difference between original value and prediction value of the model.

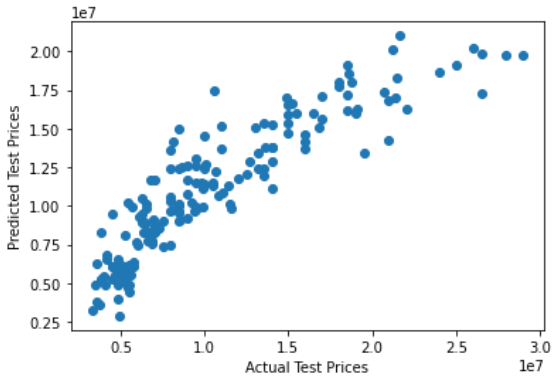
Step-4: Let us now see how the model performs on test data

Code:

```
predictions = model.predict(x_test)

plt.scatter(x=y_test, y=predictions)

plt.xlabel('Actual Test Prices')

plt.ylabel('Predicted Test Prices')
```

Output:

```
Text(0, 0.5, 'Predicted Test Prices')
```



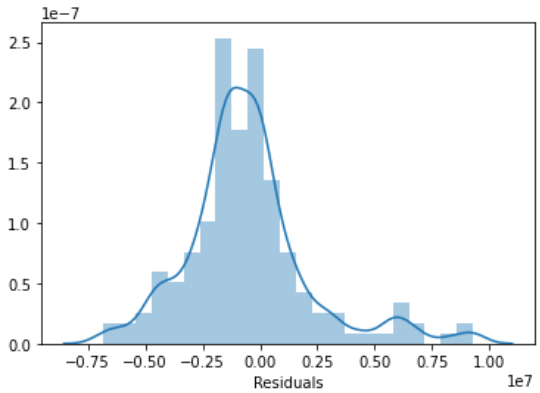
Although there seems to be a linear correlation between test set Prices and predicted Prices, there is also a fair bit of scatter. Let us check the residuals plot.

Code:

```
residuals = y_test - predictions

sns.distplot(residuals)

plt.xlabel('Residuals')
```

Output:

```
Text(0.5, 0, 'Residuals')
```



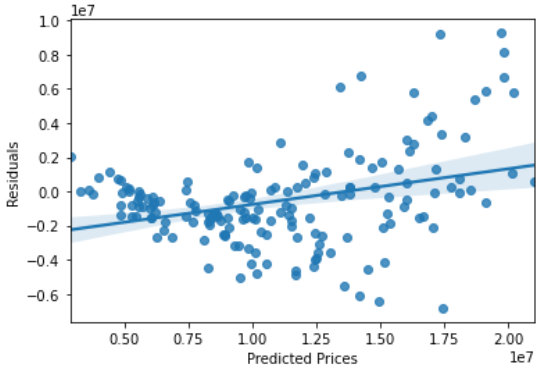
The residuals seem to be mostly normally distributed, except for a slight skew to the right. Residuals bordering on close to Rs 10 million are problematic. Let us now check the plot of residuals against predicted prices

Code:

```
sns.regplot(x=predictions, y=residuals)

plt.xlabel('Predicted Prices')

plt.ylabel('Residuals')
```

Output:

```
Text(0, 0.5, 'Residuals')
```



Step-5: Now we calculate the mean absolute error and the prediction mean using python code, which has been imported from the sklearn library, The prediction mean will provide the prediction mean value of the real estate property in an metropolitan city of India.

For eg:

Code:

from sklearn.linear_model import LinearRegression

from sklearn import metrics

from sklearn.metrics import mean_absolute_error

Code: mean_absolute_error(y_test,predictions)

output: 2007730.9314061664

Code: predictions.mean()

output: 11077270.777435927

Observations

1. The residuals-predictions plot has a somewhat funnel/cone shape towards the right, indicating Heteroscadasticity. This is to be expected in housing prices as they do not vary uniformly. Houses which are very large or very small tend to be in short supply and hence command a higher PPSF compared to commonly available sizes.
2. For higher prices our model makes larger errors. I had expected this
3. The residuals seem to be distributed almost equally on both sides of the straight line, indicating that our model over predicts and under predicts almost equally

4. Mean absolute error of Rupees 2.0 million looks high for a range of values which has a mean of Rs 11.0 million

Step-6: Now we build a model using Random forest methodology i.e. an machine learning method

For eg:

Code:

```
from sklearn.model_selection import RandomizedSearchCV

from sklearn.ensemble import RandomForestRegressor

random_forest = RandomForestRegressor()

random_forest_model = RandomizedSearchCV(estimator=random_forest,
param_distributions=random_grid, n_jobs=1, random_state=42,

                        cv=5, n_iter=10, verbose=2, scoring='neg_mean_squared_error')

random_forest_model.fit(X_train,Y_train)
```

Output:
```
RandomizedSearchCV(cv=5, estimator=RandomForestRegressor(), n_jobs=1,
        param_distributions={'max_depth': [5, 10, 15, 20, 25, 30],
                    'max_features': ['auto', 'sqrt'],
                    'min_samples_leaf': [1, 2, 5, 10],
                    'min_samples_split': [2, 5, 10, 15,
                                100],
                    'n_estimators': [100, 200, 300, 400,
                                500, 600, 700, 800,
                                900, 1000, 1100,
                                1200]},
        random_state=42, scoring='neg_mean_squared_error',
        verbose=2)
```

Step-7: Now we calculate the prediction mean, mean absolute error and we evaluate the model performance

For eg:

Code:

```
Y_pred = random_forest_model.predict(X_test)

Y_pred
```

Output:

```
array([ 8492593.42903682,  8039507.20554062,  9104596.06844743,
       11157357.66083356,  9426339.52016068,  7140142.88633867,
       13505996.98754307, 11008648.35160509,  3683770.74047355,
        5495447.97209172, 11611938.70428324, 19033307.00568063,
       16522965.5424884 ,  9028912.39139795,  5989966.11618862,
        4489513.27266645, 11113024.24433633, 12484968.99326487,
        4715465.77146974,  7508371.72578362,  5893331.46702895,
        5991009.8953643 ,  8492716.00116445,  9986735.75607399,
       13972839.02712946,  4917858.99322695,  6502032.79487798,
        4810540.35937733, 11978006.75357558, 13011739.3649576 ,


        6197160.45407657,  4297210.01461291,  9986116.70845494,
       20382377.15208115, 13531372.22524481,  4201755.50497324,
        6864173.52742656,  4299806.4121696 , 11003688.06810892,
       22533791.2646897 ,  6676803.76385585, 11539605.74699865,
       15013205.60716695,  3988920.50791917,  9835417.26804649,
       16562561.7769492 , 20386441.13832619,  8013728.27442541,
        7553636.30330291, 23175491.64640615,  9992177.67908153,
        7500507.64207565,  6000066.43064537, 10493736.43786405,
        3206070.50230652,  8489988.16970772,  4516535.9747848 ,
        8037594.35261805,  7006320.1325168 ,  8506553.25602668,
        4989588.7258773 ,  8023304.51016884,  9011123.46799496,
       11028845.3225226 ])
```

This is the short version of the original data.

Code:

```
from sklearn.metrics import r2_score

r2_score(Y_train, random_forest_model.predict(X_train))
```

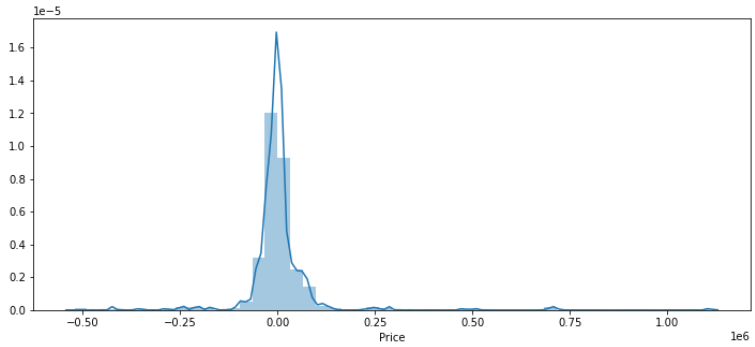
Output:

```
0.9997312058994824
```

Code:

```
plt.figure(figsize=(12,5))
sns.distplot(Y_train-random_forest_model.predict(X_train))
plt.show()
```

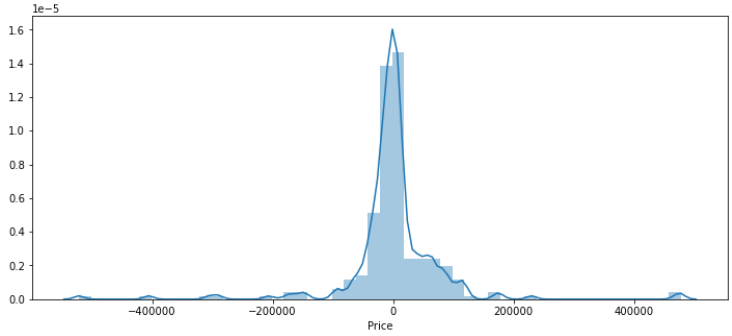
Output:



Code:

```
plt.figure(figsize=(12,5))
sns.distplot(Y_test-random_forest_model.predict(X_test))
plt.show()
```

Output:



Code:

```
mean_absolute_error(Y_test,Y_pred)
```

Output:

```
40626.01226960377
```

Code:

```
Y_pred.mean()
```

Output:

```
10465436.349656964
```

The methodology that has been mentioned above is use to analysis the dataset of the six cities namely- Mumbai, Bangalore, Chennai, Delhi, Kolkata and Hyderabad. Which is the detail procedure of my analysis, which has been classified into five main parts namely:

(1) Exploratory Data analysis(EDA)
(2) Feature Engineering
(3) Feature selection
(4) Model creation and deployment, Hyper parameter tuning, Prediction
(5) Using the SPSS software, we draw the ANOVA table for further interpretation about the significance of the model.
(6) Using the Tableau software we will carry out the necessary graph and plots for Final interpretation about the prediction mean and mean absolute error.

Which I have shown in detail, Now I step forward to my results and conclusion which I have obtained from the analysis.

Results and conclusions

After completing the analysis of the Datasets for six cities, The following results have been obtained from the analysis which are:

(1) To check whether multi-co-linearity exist in the dataset, I have calculated the variance inflating factor(VIF) for each cities which are as follows:

VIF-table for Mumbai dataset:

```
Area                                    10.180599
Bedrooms                                 9.310975
Resale                                   3.374238
Gymnasium                                7.029648
SwimmingPool                             6.592871
                                           ...
Location_no 9                            1.407113
Location_raheja vihar                    1.920898
Location_taloja panchanand               1.751822
Location_thakur village kandivali east   1.833091
Location_vasant vihar thane west         1.399950
Length: 208, dtype: float64
```

VIF-table for Bangalore dataset:

```
Area                                     4.218759
Bedrooms                                 3.180027
Resale                                   3.209462
Gymnasium                               19.860166
SwimmingPool                            26.932949
                                           ...
Location_Varthur                             inf
Location_Vidyaranyapura                      inf
Location_Whitefield Hope Farm Junction       inf
Location_Yelahanka                           inf
Location_Yerthiganahalli                     inf
Length: 124, dtype: float64
```

VIF-table for Delhi dataset:

```
Area                                     8.768304
Bedrooms                                 2.766580
Resale                                   4.477686
Gymnasium                               17.260563
SwimmingPool                             6.877760
                                           ...
```

```
Location_Vasant Kunj                    5.575837
Location_West Sagarpur                  1.125693
Location_greater kailash Enclave 1      1.084679
Location_mayur vihar phase 1            1.135669
Location_nawada                         3.946635
Length: 122, dtype: float64
```

VIF-table for Chennai dataset:

```
Area                        5.766694
Bedrooms                    3.579422
Resale                      1.751947
Gymnasium                  11.878468
SwimmingPool                6.694276
                             ...
Location_Vellakkal          2.429954
Location_Vengaivasal        1.331238
Location_Villivakkam        1.572014
Location_West Tambaram      2.174783
Location_tambaram west      8.072015
Length: 147, dtype: float64
```

VIF-table for Kolkata dataset:

```
Area                          inf
Bedrooms                      inf
Resale                        inf
MaintenanceStaff              inf
Gymnasium                     inf
                              ...
Location_Sonarpur             inf
Location_Tollygunge           inf
Location_Ultadanga            inf
Location_Uttarpara Kotrung    inf
Location_south dum dum        inf
Length: 64, dtype: float64
```

VIF-table for Hyderabad dataset:

```
Area                          3.409339
Bedrooms                      3.028996
Resale                        2.186807
Gymnasium                     8.995529
SwimmingPool                  7.915541
                                ...
Location_muthangi             2.044062
Location_new nallakunta       4.075025
Location_nizampet road        3.069351
Location_raidurgam            2.315374
Location_west venkatapuram    2.018404
Length: 232, dtype: float64
```

A **rule of thumb** for interpreting the variance inflation factor:

- 1 = not correlated.
- Between 1 and 5 = moderately correlated.
- Greater than 5 = highly correlated.

From the above tables, we see that each of the six dataset contains some sort of multi-co-linearity, which varies with different dataset. To reduce multi-co-linearity we might use certain remedies such as dropping the variables, but dropping the significant variable may reduce the value of R-square which may not help as an remedy, then also we will try to reduce the multi-co-linearity by dropping some of the insignificant variable which are not significant for the model to be fitted.

(2) After Fitting the linear regression model using ordinary least square method(OLS), the summary from the models and to test the significance of the model using ANOVA technique and Building a predictive model using the Random forest methodology and evaluating the model performance for each six cities is given below:

Summary table for Mumbai dataset:

OLS Regression Results

Dep. Variable:	Price	**R-squared (uncentered):**	0.953
Model:	OLS	**Adj. R-squared (uncentered):**	0.953
Method:	Least Squares	**F-statistic:**	2720.
Date:	Thu, 03 Jun 2021	**Prob (F-statistic):**	0.00
Time:	21:22:07	**Log-Likelihood:**	-10906.
No. Observations:	673	**AIC:**	2.182e+04
Df Residuals:	668	**BIC:**	2.184e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Area	7043.8083	666.201	10.573	0.000	5735.707	8351.909

FeatureScore	-5.342e+04	7900.886	-6.761	0.000	-6.89e+04	-3.79e+04
Resale	-3.027e+06	2.19e+05	-13.822	0.000	-3.46e+06	-2.6e+06
Bedrooms	8.726e+05	3.23e+05	2.701	0.007	2.38e+05	1.51e+06
LogPremium	8.599e+06	3.02e+05	28.502	0.000	8.01e+06	9.19e+06

Omnibus:	87.334	**Durbin-Watson:**	2.042
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	148.759
Skew:	0.821	**Prob(JB):**	4.98e-33
Kurtosis:	4.616	**Cond. No.**	3.44e+03

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The condition number is large, 3.44e+03. This might indicate that there are strong multi-co-linearity or other numerical problems.

The required linear regression model of the Mumbai city dataset is:

Y[Price] = 7043.8083[$Area_i$]+(-5.342e+04) [$FeatureScore_i$]+(-3.027e+06) [$Resale_i$]+ (8.726e+05) [$Bedrooms_i$]+(8.599e+06) [$LogPremium_i$]

Which is the predictive model, using this model we can predict the price of the city.

Now we would like to test if these linear regression model is significant i.e. the model fits the data. One approach is compute the coefficient of determination R-square and other approach is to use the technique of ANOVA, so we draw the ANOVA table using the SPSS software and interpret the results

Now we first state the null hypothesis that:

H_o: Area = FeatureScore = Resale = Bedrooms = LogPremium = 0
i.e. None of the explanatory variable has significant contribution to the Dependent variable.

Against the alternative hypothesis that:

H_1: Area ≠ FeatureScore ≠ Resale ≠ Bedrooms ≠ LogPremium ≠ 0
i.e. There is at-least one explanatory variable, which has significant contribution to the dependent variable.

ANOVA table for Mumbai city

ANOVA[a]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	2516596689005 5340.000	5	5033193378011 068.000	1129.149	.000[b]
	Residual	3726477832295 340.000	836	4457509368774 .330		
	Total	2889244472235 0680.000	841			

a. Dependent Variable: Price

b. Predictors: (Constant), LogPremium, FeatureScore, Bedrooms, Resale, Area

Interpretation: The calculated value of F is significantly large, therefore we reject the null hypothesis and conclude that at-least one of the explanatory variable, which has significant contribution to the dependent variable. Therefore the model significantly fits the data.
From the summary table we see that the coefficient of determination of the model is 0.953 i.e. around 95% which imply that 95% of the variable in Y is explained by the regression model.

Now we use the Random forest methodology which is an part of Machine learning tool, to build an predictive model. After build the predictive model, we check the performance of the model by evaluating the R-square-score which is 0.9997294868500382 for the Mumbai dataset, Which is better than Linear regression model.

Code:

```
from sklearn.metrics import r2_score
r2_score(Y_train, random_forest_model.predict(X_train))
```

Output: `0.9997294868500382`

Summary table for Bangalore dataset:

OLS Regression Results

Dep. Variable:	Price	**R-squared (uncentered):**	0.968
Model:	OLS	**Adj. R-squared (uncentered):**	0.968
Method:	Least Squares	**F-statistic:**	5988.
Date:	Fri, 04 Jun 2021	**Prob (F-statistic):**	0.00
Time:	11:51:49	**Log-Likelihood:**	-15398.
No. Observations:	989	**AIC:**	3.081e+04
Df Residuals:	984	**BIC:**	3.083e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Area	4572.9604	234.934	19.465	0.000	4111.932	5033.989
FeatureScore	346.1783	4235.076	0.082	0.935	-7964.640	8656.996
Resale	-7.743e+05	1.68e+05	-4.602	0.000	-1.1e+06	-4.44e+05
Bedrooms	-5.755e+05	1.12e+05	-5.157	0.000	-7.94e+05	-3.57e+05
LogPremium	5.77e+06	2.82e+05	20.433	0.000	5.22e+06	6.32e+06

Omnibus:	123.691	**Durbin-Watson:**	1.981
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	184.989
Skew:	0.877	**Prob(JB):**	6.76e-41
Kurtosis:	4.190	**Cond. No.**	8.89e+03

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The condition number is large, 8.89e+03. This might indicate that there are strong multi-co-linearity or other numerical problems.

The required linear regression model of the Bangalore city dataset is:

Y[Price] = 4572.9604 $[Area_i]$+(346.1783) $[FeatureScore_i]$+(-7.743e+05) $[Resale_i]$+
(-5.755e+05) $[Bedrooms_i]$+(5.77e+06) $[LogPremium_i]$

Which is the predictive model, using this model we can predict the price of the city.

Now we would like to test if these linear regression model is significant i.e. the model fits the data. One approach is compute the coefficient of determination R-square and other approach is to use the technique of ANOVA, so we draw the ANOVA table using the SPSS software and interpret the results

Now we first state the null hypothesis that:

H_o: Area = FeatureScore = Resale = Bedrooms = LogPremium = 0
i.e. None of the explanatory variable has significant contribution to the Dependent variable.

Against the alternative hypothesis that:

H_1: Area ≠ FeatureScore ≠ Resale ≠ Bedrooms ≠ LogPremium ≠ 0
i.e. There is at-least one explanatory variable, which has significant contribution to the dependent variable.

ANOVA table for Bangalore city

ANOVA[a]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	6956212971484335.000	5	1391242594296867.000	1114.133	.000[b]
	Residual	1537176617036598.000	1231	1248721865992.362		
	Total	8493389588520933.000	1236			

a. Dependent Variable: Price

b. Predictors: (Constant), LogPremium, Resale, Bedrooms, FeatureScore, Area

Interpretation: The calculated value of F is significantly large, therefore we reject the null hypothesis and conclude that at-least one of the explanatory variable, which has significant contribution to the dependent variable. Therefore the model significantly fits the data.

From the summary table we see that the coefficient of determination of the model is 0.968 i.e. around 96% which imply that 96% of the variable in Y is explained by the regression model.

Now we use the Random forest methodology which is an part of Machine learning tool, to build an predictive model. After build the predictive model, we check the performance of the model by evaluating the R-square-score which is 0.9996961469785727 for the Bangalore dataset, Which is better than Linear regression model.

Code:

```
from sklearn.metrics import r2_score

r2_score(Y_train, random_forest_model.predict(X_train))
```

Output: `0.9996961469785727`

Summary table for Delhi dataset:

OLS Regression Results

Dep. Variable:	Price	**R-squared (uncentered):**	0.978
Model:	OLS	**Adj. R-squared (uncentered):**	0.977
Method:	Least Squares	**F-statistic:**	7987.
Date:	Fri, 04 Jun 2021	**Prob (F-statistic):**	0.00
Time:	14:52:43	**Log-Likelihood:**	-14404.
No. Observations:	920	**AIC:**	2.882e+04
Df Residuals:	915	**BIC:**	2.884e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Area	8695.0419	218.830	39.734	0.000	8265.575	9124.509
FeatureScore	1.873e+04	6867.942	2.727	0.007	5250.348	3.22e+04
Resale	1.518e+05	1.27e+05	1.198	0.231	-9.69e+04	4e+05
Bedrooms	-1.31e+06	6.37e+04	-20.578	0.000	-1.44e+06	-1.19e+06
LogPremium	4.745e+06	2.72e+05	17.419	0.000	4.21e+06	5.28e+06

Omnibus:	52.986	**Durbin-Watson:**	1.925
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	169.199
Skew:	0.185	**Prob(JB):**	1.82e-37
Kurtosis:	5.068	**Cond. No.**	6.74e+03

Notes:

[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The condition number is large, 6.74e+03. This might indicate that there are strong multi-co-linearity or other numerical problems.

The required linear regression model of the Delhi city dataset is:

Y[Price] = 8695.0419[$Area_i$]+(1.873e+04) [$FeatureScore_i$]+(1.518e+05)[$Resale_i$]+
(-1.31e+06) [$Bedrooms_i$]+(4.745e+06) [$LogPremium_i$]

Which is the predictive model, using this model we can predict the price of the city.

Now we would like to test if these linear regression model is significant i.e. the model fits the data. One approach is compute the coefficient of determination R-square and other approach is to use the technique of ANOVA, so we draw the ANOVA table using the SPSS software and interpret the results

Now we first state the null hypothesis that:

H_o: Area = FeatureScore = Resale = Bedrooms = LogPremium = 0
i.e. None of the explanatory variable has significant contribution to the Dependent variable.

Against the alternative hypothesis that:

H_1: Area ≠ FeatureScore ≠ Resale ≠ Bedrooms ≠ LogPremium ≠ 0
i.e. There is at-least one explanatory variable, which has significant contribution to the dependent variable.

ANOVA table for Delhi city

ANOVA[a]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	2999632395750 6912.000	5	5999264791501 382.000	3189.928	.000[b]
	Residual	2151509219217 002.000	1144	1880689876937 .939		
	Total	3214783317672 3912.000	1149			

a. Dependent Variable: Price

b. Predictors: (Constant), LogPremium, Bedrooms, Resale, FeatureScore, Area

Interpretation: The calculated value of F is significantly large, therefore we reject the null hypothesis and conclude that at-least one of the explanatory variable, which has significant contribution to the dependent variable. Therefore the model significantly fits the data.

From the summary table we see that the coefficient of determination of the model is 0.968 i.e. around 96% which imply that 96% of the variable in Y is explained by the regression model.

Now we use the Random forest methodology which is an part of Machine learning tool, to build an predictive model. After build the predictive model, we check the performance of the model by evaluating the R-square-score which is 0.999937818688379 for the Delhi dataset, Which is better than Linear regression model.

Code:

```
from sklearn.metrics import r2_score

r2_score(Y_train, random_forest_model.predict(X_train))
```

Output: `0.999937818688379`

Summary table for Chennai dataset:

OLS Regression Results

Dep. Variable:	Price	**R-squared (uncentered):**	0.984
Model:	OLS	**Adj. R-squared (uncentered):**	0.984
Method:	Least Squares	**F-statistic:**	1.304e+04
Date:	Fri, 04 Jun 2021	**Prob (F-statistic):**	0.00
Time:	21:10:32	**Log-Likelihood:**	-15544.
No. Observations:	1035	**AIC:**	3.110e+04
Df Residuals:	1030	**BIC:**	3.112e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Area	4691.6198	164.685	28.488	0.000	4368.463	5014.776
FeatureScore	1.77e+04	2231.535	7.933	0.000	1.33e+04	2.21e+04
Resale	-1.377e+05	8.66e+04	-1.590	0.112	-3.08e+05	3.22e+04
Bedrooms	-3.35e+05	6.88e+04	-4.869	0.000	-4.7e+05	-2e+05
LogPremium	5.222e+06	1.83e+05	28.499	0.000	4.86e+06	5.58e+06

Omnibus:	191.518	**Durbin-Watson:**	1.874
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	443.865
Skew:	1.009	**Prob(JB):**	4.13e-97
Kurtosis:	5.494	**Cond. No.**	8.30e+03

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The condition number is large, 8.3e+03. This might indicate that there are strong multi-co-linearity or other numerical problems.

The required linear regression model of the Chennai city dataset is:

Y[Price] = 4691.6198 [$Area_i$]+(1.77e+04) [$FeatureScore_i$]+(-1.377e+05)[$Resale_i$]+
(-3.35e+05) [$Bedrooms_i$]+(5.222e+06) [$LogPremium_i$]

Which is the predictive model, using this model we can predict the price of the city.

Now we would like to test if these linear regression model is significant i.e. the model fits the data. One approach is compute the coefficient of determination R-square and other approach is to use the technique of ANOVA, so we draw the ANOVA table using the SPSS software and interpret the results

Now we first state the null hypothesis that:

H_o: Area = FeatureScore = Resale = Bedrooms = LogPremium = 0
i.e. None of the explanatory variable has significant contribution to the Dependent variable.

Against the alternative hypothesis that:

H_1: Area ≠ FeatureScore ≠ Resale ≠ Bedrooms ≠ LogPremium ≠ 0
i.e. There is at-least one explanatory variable, which has significant contribution to the dependent variable.

ANOVA table for Chennai city

ANOVA[a]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	5409195416422680.000	5	1081839083284536.000	2218.114	.000[b]
	Residual	628195216213932.400	1288	487729205135.041		
	Total	6037390632636612.000	1293			

a. Dependent Variable: Price

b. Predictors: (Constant), LogPremium, Resale, Bedrooms, FeatureScore, Area

Interpretation: The calculated value of F is significantly large, therefore we reject the null hypothesis and conclude that at-least one of the explanatory variable, which has significant contribution to the dependent variable. Therefore the model significantly fits the data.

From the summary table we see that the coefficient of determination of the model is 0.984 i.e. around 98% which imply that 98% of the variable in Y is explained by the regression model.

Now we use the Random forest methodology which is an part of Machine learning tool, to build an predictive model. After build the predictive model, we check the performance of the model by evaluating the R-square-score which is 0.9992769442781928 for the Chennai dataset, Which is better than Linear regression model.

Code:

```
from sklearn.metrics import r2_score

r2_score(Y_train, random_forest_model.predict(X_train))
```

Output: `0.9992769442781928`

Summary table for Kolkata dataset:

OLS Regression Results

Dep. Variable:	Price	**R-squared (uncentered):**	0.984
Model:	OLS	**Adj. R-squared (uncentered):**	0.981
Method:	Least Squares	**F-statistic:**	301.6
Date:	Thu, 03 Jun 2021	**Prob (F-statistic):**	7.75e-21
Time:	21:08:39	**Log-Likelihood:**	-430.00
No. Observations:	29	**AIC:**	870.0
Df Residuals:	24	**BIC:**	876.8
Df Model:	5		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Area	3730.7291	876.385	4.257	0.000	1921.959	5539.499
FeatureScore	1.149e+04	1.13e+04	1.017	0.319	-1.18e+04	3.48e+04
Resale	1.145e+06	3.64e+05	3.144	0.004	3.93e+05	1.9e+06
Bedrooms	-5.748e+05	3.23e+05	-1.777	0.088	-1.24e+06	9.27e+04
LogPremium	3.469e+06	5.42e+05	6.396	0.000	2.35e+06	4.59e+06

Omnibus:	0.882	**Durbin-Watson:**	1.674
Prob(Omnibus):	0.643	**Jarque-Bera (JB):**	0.194
Skew:	-0.146	**Prob(JB):**	0.908
Kurtosis:	3.274	**Cond. No.**	4.69e+03

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The condition number is large, 4.69e+03. This might indicate that there are strong multi-co-linearity or other numerical problems.

The required linear regression model of the Kolkata city dataset is:

Y[Price] = 3730.7291 [$Area_i$]+(1.149e+04) [$FeatureScore_i$]+(1.145e+06)[$Resale_i$]+
(-5.748e+05) [$Bedrooms_i$]+(3.469e+06) [$LogPremium_i$]

Which is the predictive model, using this model we can predict the price of the city.

Now we would like to test if these linear regression model is significant i.e. the model fits the data. One approach is compute the coefficient of determination R-square and other approach is to use the technique of ANOVA, so we draw the ANOVA table using the SPSS software and interpret the results

Now we first state the null hypothesis that:

H_o: Area = FeatureScore = Resale = Bedrooms = LogPremium = 0
i.e. None of the explanatory variable has significant contribution to the Dependent variable.

Against the alternative hypothesis that:

H_1: Area ≠ FeatureScore ≠ Resale ≠ Bedrooms ≠ LogPremium ≠ 0
i.e. There is at-least one explanatory variable, which has significant contribution to the dependent variable.

ANOVA table for Kolkata city

ANOVA[a]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	150501490761901.660	5	301002981523810.332	68.321	.000[b]
	Residual	136576408396115.338	31	4405690593420.430		
	Total	164159131601517.000	36			

a. Dependent Variable: Price

b. Predictors: (Constant), LogPremium, Area, FeatureScore, Resale, Bedrooms

Interpretation: The calculated value of F is significantly large, therefore we reject the null hypothesis and conclude that at-least one of the explanatory variable, which has significant contribution to the dependent variable. Therefore the model significantly fits the data.

From the summary table we see that the coefficient of determination of the model is 0.984 i.e. around 98% which imply that 98% of the variable in Y is explained by the regression model.

Now we use the Random forest methodology which is an part of Machine learning tool, to build an predictive model. After build the predictive model, we check the performance of the model by evaluating the R-square-score which is 0.9735968057224695 for the Kolkata dataset, Which is better than Linear regression model.

Code:

```
from sklearn.metrics import r2_score

r2_score(Y_train, random_forest_model.predict(X_train))
```

Output: `0.9735968057224695`

Summary table for Hyderabad dataset:

OLS Regression Results

Dep. Variable:	Price	**R-squared (uncentered):**	0.971
Model:	OLS	**Adj. R-squared (uncentered):**	0.971
Method:	Least Squares	**F-statistic:**	7536.
Date:	Fri, 04 Jun 2021	**Prob (F-statistic):**	0.00
Time:	22:29:36	**Log-Likelihood:**	-17601.
No. Observations:	1128	**AIC:**	3.521e+04
Df Residuals:	1123	**BIC:**	3.524e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Area	5213.1300	188.054	27.722	0.000	4844.154	5582.106
FeatureScore	1.323e+04	3039.195	4.352	0.000	7262.735	1.92e+04
Resale	3.417e+05	1.11e+05	3.079	0.002	1.24e+05	5.59e+05
Bedrooms	-1.146e+06	1.05e+05	-10.942	0.000	-1.35e+06	-9.4e+05
LogPremium	5.728e+06	2.51e+05	22.864	0.000	5.24e+06	6.22e+06

Omnibus:	70.717	**Durbin-Watson:**	1.991
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	109.480
Skew:	0.495	**Prob(JB):**	1.69e-24
Kurtosis:	4.161	**Cond. No.**	8.99e+03

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[3] The condition number is large, 8.99e+03. This might indicate that there are strong multi-co-linearity or other numerical problems.

The required linear regression model of the Hyderabad city dataset is:

Y[Price] = 5213.1300 [$Area_i$]+(1.323e+04) [$FeatureScore_i$]+(3.417e+05)[$Resale_i$]+
(-1.146e+06) [$Bedrooms_i$]+(5.728e+06) [$LogPremium_i$]

Which is the predictive model, using this model we can predict the price of the city.

Now we would like to test if these linear regression model is significant i.e. the model fits the data. One approach is compute the coefficient of determination R-square and other approach is to use the technique of ANOVA, so we draw the ANOVA table using the SPSS software and interpret the results

Now we first state the null hypothesis that:

H_o: Area = FeatureScore = Resale = Bedrooms = LogPremium = 0
i.e. None of the explanatory variable has significant contribution to the Dependent variable.

Against the alternative hypothesis that:

H_1: Area ≠ FeatureScore ≠ Resale ≠ Bedrooms ≠ LogPremium ≠ 0
i.e. There is at-least one explanatory variable, which has significant contribution to the dependent variable.

ANOVA table for Hyderabad city

ANOVA[a]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	8489787061885649.000	5	1697957412377129.800	1003.857	.000[b]
	Residual	1821673590462321.000	1077	1691433231627.039		
	Total	1031146065234 7970.000	1082			

a. Dependent Variable: Price

b. Predictors: (Constant), LogPremium, Resale, Bedrooms, FeatureScore, Area

Interpretation: The calculated value of F is significantly large, therefore we reject the null hypothesis and conclude that at-least one of the explanatory variable, which has significant contribution to the dependent variable. Therefore the model significantly fits the data.

From the summary table we see that the coefficient of determination of the model is 0.971 i.e. around 97% which imply that 97% of the variable in Y is explained by the regression model.

Now we use the Random forest methodology which is an part of Machine learning tool, to build an predictive model. After build the predictive model, we check the performance of the model by evaluating the R-square-score which is 0.9997239136064422 for the Hyderabad dataset, Which is better than Linear regression model.

Code:

```
from sklearn.metrics import r2_score

r2_score(Y_train, random_forest_model.predict(X_train))
```

Output: `0.9997239136064422`

Table for comparison of R-square value of linear regression model and Random forest model

City	R-square-value of Linear regression model	R-square-value of Random forest model
Mumbai	0.953	0.999729487
Bangalore	0.968	0.999696147
Delhi	0.978	0.999937819
Chennai	0.984	0.999276944
Kolkata	0.984	0.973596806
Hyderabad	0.971	0.999723914

Interpretation:

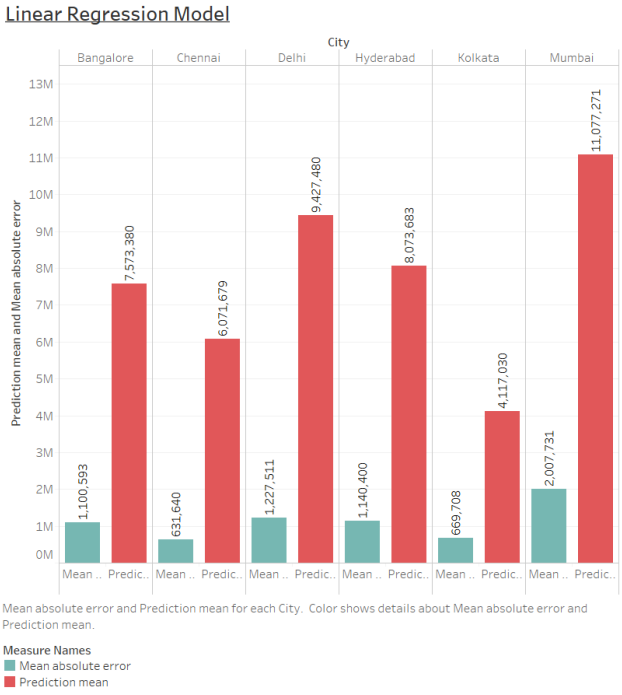
From the above table, we see that the R-square value of linear regression model of each city is less than the R-square value of Random forest model of each city except in case of Kolkata city dataset. Hence we can interpret that the Random forest methodology provides more accurate model than the Linear regression model, if we compare by using the values of R-square of both the models.

(3) Comparing the prediction mean and mean absolute error of each cities of Linear regression model and random forest model:

Table 1: prediction mean and mean absolute error of linear regression model:

	Linear regression model	
City	**Prediction mean**	**Mean absolute error**
Mumbai	11077270.78	2007730.931
Bangalore	7573379.595	1100592.783
Delhi	9427479.741	1227510.507
Chennai	6071679.38	631639.5474
Kolkata	4117029.804	669708.2045
Hyderabad	8073683.209	1140399.936

Graph 1: prediction mean and mean absolute error of linear regression model:



Graph 2: prediction mean of linear regression model:

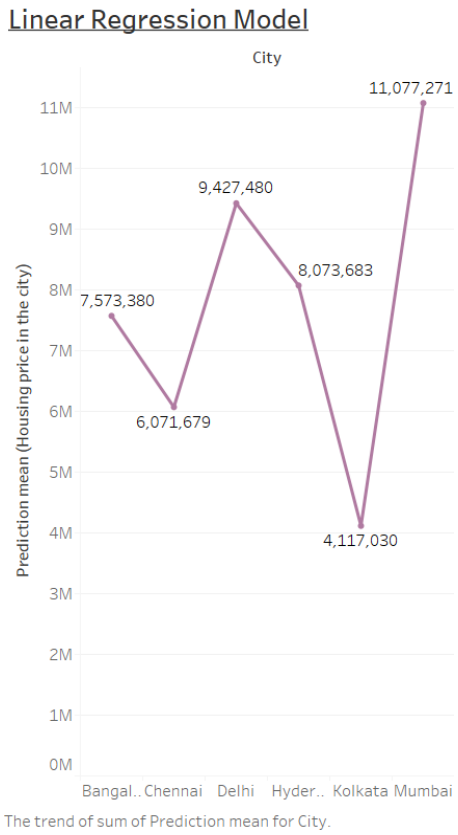
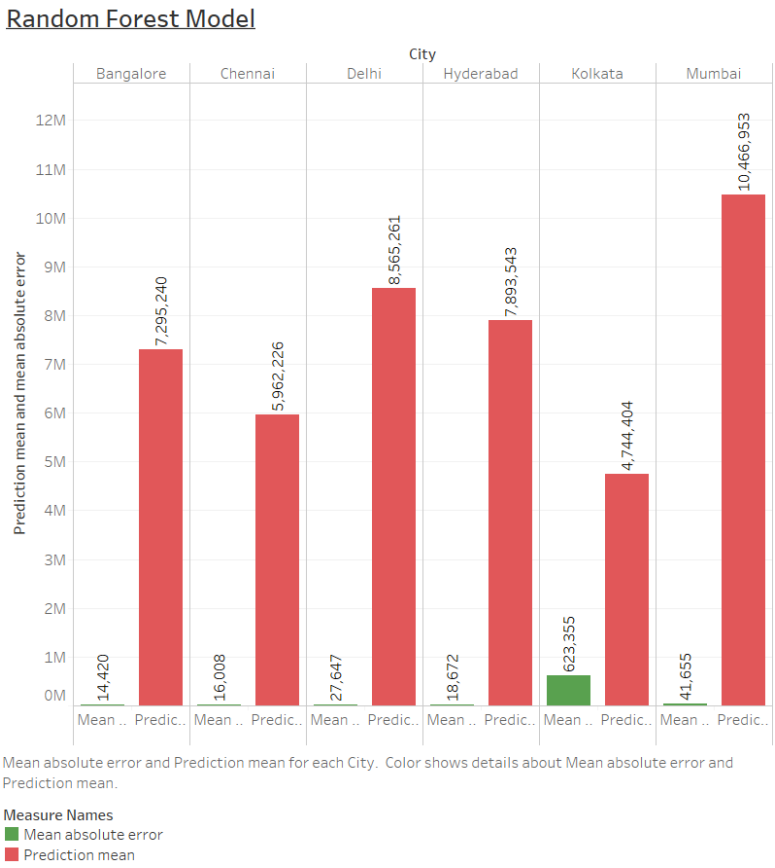


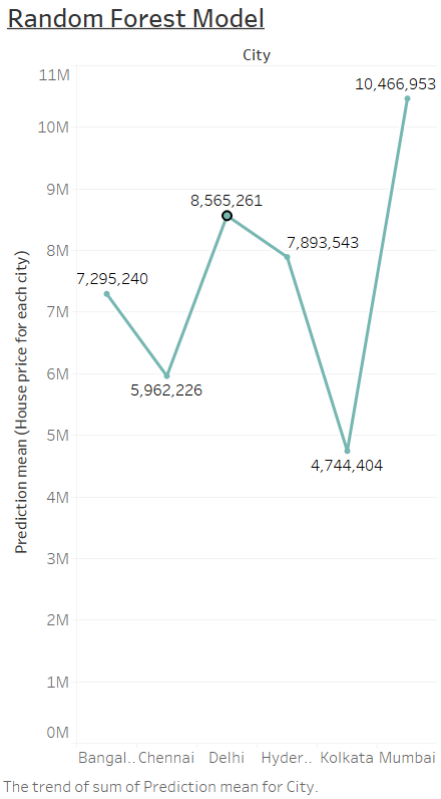
Table 2: prediction mean and mean absolute error of Random forest model:

	Random forest model	
City	Prediction mean	Mean absolute error
Mumbai	10466953.09	41654.59069
Bangalore	7295239.812	14419.82601
Delhi	8565260.906	27646.91349
Chennai	5962225.508	16008.24813
Kolkata	4744404.163	623354.7196
Hyderabad	7893542.604	18671.62035

Graph 3: prediction mean and mean absolute error of Random forest model:



Graph 4: prediction mean of Random forest model:



Interpretation:

From the above Tables and Graphs of Prediction mean and Mean absolute error of Linear regression model and Random forest model, we can say that Mumbai is the costliest city among the six cities to buy a house with prediction mean of 11077270.78 from linear regression model and 10466953.09 from Random forest model which is approximately around Rs 11.0 million or Rs 1.1 crore.

Now from the mean absolute error of the two models, we can say that Random forest model has least error than Linear regression model, which provides more accuracy than the Linear regression model.

Final Interpretation:

After analysis the six cities dataset what I have found that are:

(1) For checking the multi-co-linearity in the dataset by using variance inflating factor (VIF), I have found that multi-co-linearity exist in the dataset, which can be reduce by dropping some of the explanatory variable, but doing so we have to be very cautious about dropping the variable, which can significantly reduce the R-square value.
(2) After fitting the linear regression model using OLS method and Building the predictive using Random forest methodology, we get the R-square values for both the models, If we compare both the R-square values we come to the conclusion that Random forest methodology builds more accurate predictive model.
(3) After calculating the predictive mean and mean square error of both the models for six cities dataset, we come to the conclusion that Mumbai is the costliest city of all six cities dataset to buy a house and Random forest methodology provides least error and accurate predictive model than the linear regression model.