ADDIS ABABA CITY HOUSE PRICE PREDICTION BY USING REGRESSION ALGORITHM

BY: GIRMA TESHOME MULUGETA

OFFICE OF GRADUATION STUDY

ADAMA SCIENCE AND TECHNOLOGY UNIVERSITY

(ASTU)

JANUARY, 2021

ADAMA, ETHIOPIA

ADDIS ABABA CITY HOUSE PRICE PREDICTION BY USING REGRESSION ALGORITHM

BY: GIRMA TESHOME MULUGETA

ADVISOR: DR. TEKLU URGESSA(PHD)

A THESIS SUBMITTED TO

DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTING

OFFICE OF GRADUATION STUDY

ADAMA SCIENCE AND TECHNOLOGY UNIVERSITY(ASTU)

JANUARY, 2021

ADAMA, ETHIOPIA

DECLARATION

I declare that thesis's work entitled "Addis Ababa City House Price Prediction by Using Machine Learning" has been carried out by me in the computer science and engineering department. Any sources of materials used in this thesis have been acknowledged, and a list of references provided. This thesis work has not been presented for another degree and diploma in any other universities.

Name: Girma Teshome Mulugeta

Signature: _____

Date: _____

Advisor Name: Dr. Teklu Urgessa(PHD)

Signature: _____

Date: _____

Acknowledgement

All that we know is a sum total of what we have learned from all who have taught us, both formally and informally. I am forever indebted to the countless outstanding men and women who, by their commitment and dedication to becoming the best they could be, have inspired me to do the same.

First and above of all, I would like to thank the almighty God for giving me this wonderful opportunity and helping me in every way of my success. And, by the help of several people this thesis has reached to this position.

I would like to thank my advisor Dr. Teklu Urgessa for his continuous and constructive reviews of this work.

Finally, I would like to thank my family and friends who support me in different ways. Thank you, all my classmates, for your knowledge sharing experiences. I would like to thank everyone at the ASTU, School of Electrical Engineering and Computing, Department of Computing, who has encouraged and assisted me throughout the work of this thesis.

Table of Content

List of Figures

List of Tables

List of ACRONOYMS AND ABBREVIATIONS

AI: artificial intelligence

AIEI: Arab International for Education and Investment

ANN: Artificial Neural Networks

BPN: back propagation neural network

EMA: exponential moving average

GPS: Global positioning System

HPM: hedonic price model

KD Tree: K-Dimensional tree

LS-SVM: least square support vector machine

MACD: moving average convergence/divergence

MAE: Mean Absolute Error

MAPE Mean Absolute Percentage Error

MFI: money flow index

ML: Machine Learning

MLM: multilevel model

PSO: particle swarm optimization

RBF: radial basis function neural network

RSI: relative strength index

SO: stochastic oscillator

SVM: support vector machines algorithm

SVR: Support vector regression

USA: United States of America

CSV: Comma separated value

Abstract

House is a basic need for people and used to measure the economy of individuals, organizations, and country. However, the unavailability of the exact sales price of houses in Addis Ababa city is not well known. The aim of this study is to predict the sales price of the house by using a regression algorithm for Addis Ababa city house seller and house buyer with respect to their budgets and priorities by analyzing previous market trends and house price ranges. In order to choose a prediction method, several regression models are explored and compared to find the best performance solution for house price prediction. Methods that has been discussed in this study include ridge regression, elastic regression, k-nearest-neighbors(KNN), Neural network(NN), random forest, and decision tree. All this regression algorithm technique was compared with results of mean absolute error(MAE) and mean absolute percentage error (MAPE) to select the type of regression algorithm with the least MAE and MAPE. Result depicts that from all other algorithm techniques decision tree algorithm show minimum error of MAPE of 9.81 % while ridge algorithm technique resulted with maximum error of 17.50 %. This show that decision tree algorithm techniques best fitted house sales price for Addis Ababa city with minimum error of 15.76%.

***Keywords**: Machine Learning, Regression, Ridge, Elastic, Neural Network(NN), KNN, Random forest, Decision tree, Dataset, MAE, MAPE*

Chapter One: Introduction

1.1 Background of the Study

Addis Ababa is the capital city of Ethiopia, which is located at center of Ethiopia and one of among quickly growing cities in Ethiopia. According to the urbanization and urban population, the population of Addis Ababa in 1980 year is around 1,486,000, the population of Addis Ababa in 2000 year is around 2,639,000 and the population of Addis Ababa in 2020 year is around 5,095,000 [1]. The higher increase in population implies a higher demand for housing which in turn increases the need for residential houses in the City. In Ethiopia, urbanization is mostly accelerated by rural to urban migration which is beginning to enforce a high burden on the holding capacity of cities. It follows that the speed of urbanization is one of the indications of a country's economic development. The rate of urbanization is directly related to the demand for houses. It is expected that as a country becomes more urban, more houses were needed to accommodate the increasing population in urban centers.

Globally, an estimated 1.6 billion people do not have access to adequate housing, 25% of the world’s urban population lives in informal settlements, and homelessness is on the rise in every world region [2]. Thus, increasing the supply of homes is a key priority for the government through the later part of the 20th century. A growing population coupled with an increasing tendency of people to live alone has resulted in continuous rising demand for homes, but the supply of new housing feels dramatically over the same period. This gap between supply and demand has resulted in problems of housing affordability, with rising prices creating pressure for first-time buyers [3].

The housing market in Ethiopia has always been a topic of national attention, with news sites dedicating sections that report on key news that can potentially affect housing prices and the trends in recent years. Not only the trends in the housing market of concern to buyers and owners, but they also reflect the current economic situation and social sentiments in the country.

Addis Ababa house buyers who want to find out more about housing trends in various districts would therefore have little recourse. Many variables affect the trend in housing prices, which motivates the need for a prediction model that encompasses the

relationships between these variables and housing prices. House Property is not only the basic need of a people but today it also represents the riches and respect of a person. Generally, the property values rise with respect to time and its appraised value needs to be calculated. This appraised value is required during the sale of property or while applying for the loan and for the marketability of the property. These appraised values are determined by professional judges.

However, the drawback of this practice is that these appraisers could be biased due to best-owned interests from buyers, sellers, or loans. Thus, we require an automated prediction model that can help to predict the property price without any bias. Therefore, it is important to predict the house prices without bias to help both house buyers and house sellers make their decisions. There are many factors that affect house prices, such as the numbers of bedrooms, the total area of the house, specific location, date, bathrooms, etc. In addition, choosing different combinations of parameters in regression will also affect the predictions greatly. This automated model can help first-time buyers and less experienced customers to understand whether the property rates are overrated or underrated. Now, Property prices depend on various parameters in the economy and society. However, previous analyses show that house prices are strongly dependent on the size of the house, house quality, and its geographical location.

1.2 Motivation

As we know house prices increase day today in the city of Ethiopia and difficult to predict the correct price of house for both house sellers and house buyers. The origin in home values raises questions among house owners and house buyers regarding how accurately the value of the house can be measured and what attributes determine the desirability of certain houses compared to other houses in the market. This instability within the housing market necessitates the requirement for better methods for assessing housing prices. Most of the time in Ethiopia house prices is not increase constantly and house buyer and house seller kill their time by talking about the price of a selected house for sale. This problem existing everywhere within the city of Ethiopia, we motivate to predict the price of the house used by the regression machine-learning algorithm to solve this problem. Consequently, the predictive accuracy of housing models has gained much attention among researchers and has been extensively studied.

1.3 Statement of the Problem

Currently, in Ethiopia, the migration of individuals from rural to urban is increased to buy a house and for other reasons. Due to this, the demand for houses in all Ethiopian city's is rapidly increasing specially in Addis Ababa city. People looking to buy a new house tend to be more conservative with their budgets and current market. Most of the time house buyers and house sellers try to know the right price of houses and requirements that require to buy houses without the interference of third party. However, in Addis Ababa city there's no model to predict the price of house selected for sale so far. This is often the key problem for both house sellers and buyers. In Addis Ababa city house price is mainly predicted by the house owners or house sellers and illegal brokers. This traditional house prediction takes a long period of time to decide the correct sales price of selected houses for sale. This problem may influence the overall market of the country and therefore this problem needs to be solved. Therefore, it's important to predict housing prices neutrally to assist both the buyers and sellers make their decisions. In this study the house price prediction model will be carried out to reduce such problem. The prices of house properties are sophisticatedly related to our economy.

1.3.1 Research Questions

To conduct this study, the following question were raised:

1. Where the required dataset can be obtained?
2. How to make the obtained datasets suitable for the algorithm?
3. How to obtained dataset can be analyzed?
4. How to choose best result of algorithm?

1.4 Objectives

1.4.1 General Objectives

The general objective of this research to develop a housing price prediction model by using regression machine learning algorithm technique for Addis Ababa city.

1.4.2 Specific Objectives

The specific objective of this study were:

- To collect datasets from the website by using data miners to scrape (to change website data to CSV Excel format).
- To extract the dataset in CSV format and clean the dataset to make dataset suitable for our algorithm.
- To develop a model for house price prediction using Regression machine learning algorithm technique.
- To evaluate and compare each model to choose the one with the highest performance regression algorithm.

1.5 Methodology

In order to accomplish the objectives of the research, the following methods and procedures are used accordingly.

1.5.1 Literature review

Literature reviews are going to be done on related fields to the present research work. The research works involved a literature review to gather the most issues and concepts in the field of machine learning and house price prediction, attributes reduction, preprocessing of datasets, etc. While collecting all the above resources, the researcher has used different sources like, various articles, journals, books, and papers. As a result, reviewing and collecting data from different a kinds literature helped the researcher to get a better understanding of the research to be most vital.

1.5.2 Data Collection

To develop the needed knowledge, we use both secondary and primary (documented and undocumented) sources of information. I gather the Primary knowledge from different house owner/house seller, house buyer, and different house broker in Addis Ababa city and other Ethiopian city by observing the existing problem in day to day activity facing the house buyer and house seller to predict the house price. The secondary sources of knowledge are collected by using document analysis. Moreover, secondary sources of knowledge are gathered from the Internet, research, journal, conference, and reading textbook related to my title to develop my knowledge on the prediction of house price prediction. After we understand factor that affect the price of house we collect more than 8000 record of dataset from different housing websites by extracting using data miner from Chrome extension.

1.5.3 Programming Techniques and Tools

For the research, I use a machine learning algorithm, a regression approach for predicting the price of the house, and to extract data from a website I use data miner from Chrome extension. Regression a data mining task of predicting the value of a target (numerical variable) by building a model based on one or more predictors the predictors can either be numerical or categorical variables. To test/run collected sample data we also use python Jupiter notebook and other anaconda applications. And to make my reference standard reference style, I use Mendeley reference management software.

1.6 Scope and Limitation

1.6.1 Scope

This research deals to solve the problem of house price prediction for Addis Ababa city, to predict the price of the house to benefit both house sellers and house buyers depending on the given house attribute. And also used for municipal of the city and bank for different purposes. The scope of this research to examine the prediction model by using machine learning algorithms techniques to improve the prediction of the house prices for house sellers and house buyers only in Addis Ababa city house. More specifically, the study focused on the various market levels, roles of marketing actors in the marketing channel, price formation, negotiating characteristics of producers, buying, and selling strategies.

1.6.2 Limitation

This price prediction research only used to predict the price of residential houses for Addis Ababa city by considering different house attributes such as city, specific location number of bedroom, total area of house, house presented date for sale and price. The model does not include a large building/tower in the city, it is used to predict only residential/single-family home prices for living home. The other limitation of this research is the data used or collected and the sample used for the study is only taken from Addis Ababa city. On the other hand, finding and identification of appropriate machine learning software, getting enough resources such as relevant literature related to the specified study can be considered as some limitations of this research.

1.7 Application of Results

Apart from being a look to meet the need of the master's program research, the results of this study are believed to be used either as input for other researches or for future research. This research study is useful for Addis Ababa city to help them for knowing the price of the house. Because the development of more efficient price predictive models could improve the accuracy, time, and resource of existing traditional house price prediction for Addis Ababa city. Even though the research was done for educational purposes, its findings and results would be applicable for better decision support activities within the Addis Ababa organization to predict the price of the house. Additionally, all models and methods agreed with respect to the direction and magnitude of the relationships between the predictors and housing prices. Since we use a machine learning algorithm technique, it also helps us to decide the price of the house more accurately and avoid the problem of accuracy.

1.8 Organization of the Thesis

This study is organized into six chapters as follows:

Chapter One: this chapter includes background of the study, introduction, motivation, statement of the problem, objectives, research methodology, scope and limitation of the study, and application of the study.

Chapter Two: basically contains literature review and related works of the study. Gives review of literature on house price prediction and other related work.

Chapter three: presents the methodology of the work in detail. What design is better for house price prediction, why the specific design is selected and basically the methodologies and which machine learning technique are better for our study.

Chapter four: under this we see experiment and Implementation, source of dataset, preparation and dataset description including sample code

Chapter five: Include result and modeling. In this chapter we analyze all regression method and compare them with their mean absolute error (MAE) and select the best one from listed regression method.

Chapter Six: contains conclusion and recommendation (future work) of the study

Chapter Two: Literature Review

2.1 Overview

In the previous chapter, an introduction to the study has been presented including motivation, statement of problem, objective and methodologies of the research follows to accomplish the study. During this chapter the researcher has generated the theoretical background of the study area and what had been done before. Next, the researcher studied machine learning concepts, regression likewise as related fields to the machine learning and therefore the concept of regression method. Lastly, the researcher discovered the applying area of machine learning and related works for house price prediction of normally.

2.2 Factor affecting House price

The nature of house markets is inherently complex. Additionally, the range of housing attributes makes it challenging to construct a comprehensive model that may encompass all of the features [4]. the worth of a home is substantially suffering from its own unique set of structural attributes (e.g. living area, number of rooms, age of the building, swimming pool), locational attributes (proximity to business districts and accessibility to major highways) and neighborhood attributes (e.g. territorial division, income levels, per centum, population density) (D. Y. Li. Et al Q [5]. Chen (2009)). Given the complex nature of this problem, to realize a model that accurately predicts the worth of a house is difficult for a spread of reasons. One reason is that data sets with the entire information on all mentioned attributes don't seem to be easy to access. Secondly, the effect of a selected attribute on housing price might not have the identical amount of significance on price variation across different regions. As an example, the effect of square footage on price variation might not be consistent for all neighborhoods. Also, certain attributes are valued differently in various areas. As an example, in colder regions, a hearth may have a greater value while houses with swimming pools are more desirable in an exceedingly warmer climate. For those reasons, housing price predictions are usually conducted on a particular location, and are difficult to generalize across different geographic regions.

2.3 A Review of house Price predictions

At present each framework is also moved towards innovation for the simplicity from claiming operations. Individuals tend to maneuver from the manual to Robotized methodology to predict the worth of house. Increase within the demand for house property and unpredictable behavior of economy require researchers to search out how that predict the house prices with none biases. Thus, it's a challenge for researchers to seek out all the minute factors which will affect the value of property and make a predictive model by taking into consideration all the factors. Many researchers have worked on this problem and communicated their research work. The land operators give acceptable a suggestive on the lodging costs prediction. This strategy includes high hazard a right away result the land operator might supply the bad data of the clients. They employments those straight relapse calculations should figure the value. This analyses likewise utilized to foresee the most effective area for the clients to buying the homes. Eventually, Tom's perusing utilizing this straight relapse he predicted the speed for each area unit. This prediction indicates the square feet of the house are raised eventually towards 2018. There needs aid where both the purchasers and sellers should get them an equal amount of profit. Lots of researches are done on Artificial Neural Networks. This has helped many researchers that specialize in house problem to resolve using neural networks. In [6] the author has compared hedonic price model and ANN model that predict the house prices. Hedonic price models are basically wont to calculate the value of any commodity that are addicted to internal characteristics furthermore as external characteristics. The hedonic model basically involves regression technique that considers various parameters like area of the property, age, number of bedrooms then on. The Neural Network is trained initially, and therefore the weights and biases of the perimeters and nodes respectively are considered using trial and error method. Is concluded that Artificial Neural Network performs superior than Hedonic model. Performance comparison of other machine learning algorithms should even be considered. In [10], the author has considered the foremost macroeconomic parameters that affect the house prices variation. In this, the author has used back propagation neural network (BPN) and radial basis function neural network (RBF) to ascertain the nonlinear model for real estate's price variation prediction. Some research articles describe the in-depth methods

and procedures to gather the important estate data and their pre-processing techniques. Some researchers have focused on feature selection and have extraction procedure. The author in article Limsombunchai, V. (2004, June) [11] uses an open source data set of the housing sales in King County, USA. Has predicted house prices given explanatory variables that cover many aspects of residential houses. As continuous house prices, they're predicted with various regression techniques including Lasso, Ridge, SVM regression, and Random Forest regression; as individual price ranges, they're predicted with classification methods including Naive Bayes, logistic regression, SVM classification, and Random Forest classification. Da-Ying Li, et al (2009) An SVR Based Forecasting Approach for realty Price Prediction, Da-Ying Li proposes support vector regression (SVR) to forecast land prices in China [10].

2.4 House Price Prediction Using Multilevel Model and Neural Networks

A different study was done by Feng, Y., & Jones, K. [12]. To predict house prices. Two models were built: a multilevel model (MLM) and a man-made neural network model (ANN). These two models were compared to every other and to a hedonic price model (HPM). The multilevel model integrates the micro-level that specifies the relationships between houses within a given neighborhood, and therefore the macro-level equation which specifies the relationships between neighborhoods. The hedonic price model may be a model that estimates house prices using some attributes like the amount of bedrooms within the house, the dimensions of the house, etc. Secondary data was obtained from the Land Registry, the Population Census and Neighborhood Statistics to be utilized in order to create the models suitable for national usage. The authors listed many reasons on why they chose the Greater Bristol area like its diverse urban and rural blend and its different property types. Each record within the dataset contains data a couple of house within the area: it contains the address, the unit postcode, property type, the duration (freehold or leasehold), the sale price, the date of the sale, and whether the house was newly-built when it had been sold. In total, the dataset contains around 65,000 entries. To enable model training and testing, the dataset was divided into a training set that contains data about house sales from 2001 to 2012, and a test set that contains data about house sales in 2013. The three models (MLM, ANN, and HPM) were tested using three scenarios. Within the first scenario,

locational and measured neighborhood attributes weren't included within the data. Within the second scenario, grid references of house location were included within the data. Within the third scenario, measured neighborhood attributes were included within the data. The models were compared in goodness of fit where R2 was the metric, predictive accuracy where mean absolute error (MAE) and mean absolute percentage error (MAPE) were the metrics, and explanatory power. HPM and MLM models were fitted using MLwiN software, and ANN were fitted using IBM SPSS software.

2.5 L.Li, K.-H. Chu, 2018, Prediction of Real Estate Price Variation Based on Economic Parameters

In Prediction of assets Price Variation supported Economic Parameters [7] has used macroeconomic parameters on land price variation are investigated before establishing the value fluctuation prediction model [8]. Here, back propagation neural network (BPNN) and radial basis function neural network (RBF) two schemes are employed to determine the nonlinear model for real estate's price variation prediction of Taipei, Taiwan supported leading and simultaneous economic indices. The general public related data of Taipei, Taiwan land variation during 2005-2015 are adopted for analysis and prediction comparison.

2.6 Real Estate Price Prediction Using Machine Learning

We need a correct prediction on the real estate and therefore the houses in housing market we are able to see a mechanism that runs throughout the properties buying and selling and buying a house are a life time goal for many of the individual. But There are lot of individuals making huge mistakes in us of America straight away when buying the properties most of the people are buying properties unseen from the people they don't know by seeing the advertisements and every one over the grooves coming round the America one in all the common mistakes is buying the properties that are too expensive but it's not worthwhile [9].

2.7 Summary of Literature Review

For every Research the literature review will give clear idea and it'll function the bottom line. Here most of the authors have concluded that Regression machine learning have the more influence in predicting the price of house. But within the world the opposite algorithms should be also taken into consideration. By conducting this study, it helps to hold about both the pros and cons and it had helped me too successfully Implement their search. However, in Ethiopia there is no any model to predict the sales price of house. In this study we predict the sales price of house price for Addis Ababa city by reviewing different study worked before in different country. There is different accuracy rate in different country by using different machine learning algorithm technique. The following table show list of related work and their accuracy.

Authors	Title	Methods	MAPE	Gap
Erik van der Burgt	Data Engineering for house price prediction	Regression	14.1%	Not work for Ethiopia
Anh Komagome-Citye	models and visualizations for housing price prediction	Regression	10%	Not work for Ethiopia
neelam shinde, kiran gawande	valuation of house prices using predictive techniques	Regression	15.36%	Not work for Ethiopia
Ammar Alyousfi	House Price Prediction Using Machine Learning Techniques	Regression	7%	Not work for Ethiopia

Table 2. 1 Sample literature review

Chapter Three: Methodology

3.1 Introduction

Research methodology is the technique to scientifically solve the research problem. Detailed description has made about the procedures followed to conduct the study. It includes we review machine learning method, discuss why regression methods are what we need to predict future housing prices, discussions about research design, sources of data, data collection techniques, further as tools utilized to design and develop the prototype system. The aim is to clarify the procedures and steps adopted that address the research problem. Each method contains basics that build up to the following, making it easier to understand how we reach the method of our choice.

3.2 Machine Learning

Machine learning is a discipline that evolved from artificial intelligence, but it focuses more on cognitive learning capabilities. AI has many other aspects that attempt to model human function and intelligence (such as problem-solving). However, ML is a subset technology specific to the use of data to simulate human learning Sharp, et al [13]. A computer program is said to learn from experience E with relevancy some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E. In keeping with Tom Mitchell definition, the machine learns with regard to particular tasks T, performance P and experience E Tom M. Mitchell, 2006 [14]. The machine learning algorithm will be classified into four, namely supervised, unsupervised, semi-supervised and Reinforcement learning, however the foremost known algorithms are supervised and unsupervised learning. From the categories, supervised learning requires a training data set with determined labeled class while unsupervised don't required to possess labeled class. An issue solved through classification and regressions methods are classified under supervised learning, due to data sets have known output values. Clustering problems may be solved with an unsupervised approach. Machine learning draws on concepts and results from many fields, including statistics, AI, philosophy, scientific theory, biology, scientific discipline, computational complexity, and control theory.

As our understanding of computers continues to mature, it seems inevitable that machine learning will play an increasingly central role in engineering science and technology. Machine learning usually refers to the changes in systems that perform tasks related to AI. Such tasks involve recognition, diagnosis, planning, robot control, prediction, etc. Basically, machine learning affords computer programs with capabilities to imitate the human learning process. In machine learning and statistics, feature selection, also called variable selection, attribute selection or variable subset selection, is that process of choosing a subset of relevant features (variables, predictors) to be used in model construction. From the categories machine learning, supervised learning requires a training data set with determined labeled class while unsupervised don't required to own labeled class. An issue solved through classification and regressions methods are classified under supervised learning, thanks to data sets have known output values. Clustering problems may be solved with an unsupervised approach.

Figure 3. 1: Traditional programming paradigm

Machine learning is field of study that gives computers the ability to learn without being explicitly programmed [19].

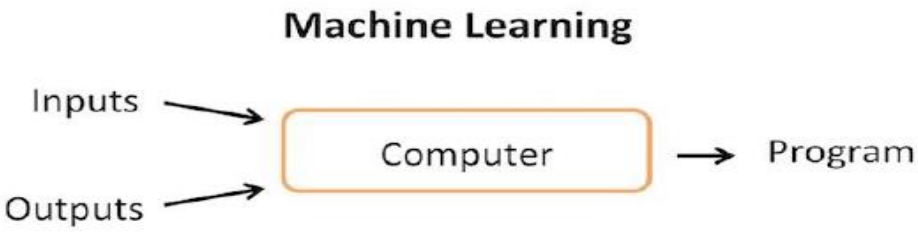

Figure 3. 2: Machine Learning programming paradigm

3.2.1 Supervised Learning

The main idea of supervised learning is to learn a mapping between the input and the output whose correct values provided by a supervisor (Antonelo, E. A., & Schrauwen, B. (2010, May)).

Supervised learning is that the machine learning task of inferring a function from labeled training data (Antonelo, E. A., & Schrauwen, B. (2010, May)). There are two main types of supervised learning, classification and regression, where there is an input and output, and the main role is to find a mapping between the input and the output. In classification, the task is to assign the training input to one of the predefined classes. The training data accommodates a group of coaching examples. The outputs may be real numbers in regression or class labels in classification. As classifying an input into two or more classes, supervised learning is usually referred to as classification. A supervised learning algorithm analyzes the training data and produces an inferred function, which may be used for mapping new examples. An optimal scenario will afford the algorithm to properly determine the category labels for unseen instances. This needs the training algorithm to generalize from the training data to unseen situations in a very "reasonable" way. There are a range of classification techniques that are widely exploited within the literature for labeling software instances as defective or defect-free. It's the machine learning task of inferring a function from labeled training data. The training data consists of a group of coaching examples. In supervised learning, each example could be a pair consisting of an input object and a desired output value. It's two known supervised learning tasks, classification and regression. Classification concerns on building predictive model for function with discrete range, while regression concern on continuous range model building.

3.2.2 Classification

Classification is that problem of identifying to which of a collection of categories (sub-populations) a make new observation belongs, supported a training set of knowledge containing observations (or instances) whose category membership is thought Khan, A., Baharudin, et al 2010 [15]. Many real-world problems may be modeled as classification problems like assigning a given email into spam or non-spam" classes, automatically assigning the categories (e.g., "Sports" and "Entertainment") of coming news, and assigning a diagnosis to a given patient as described by observed characteristics of the patient (gender, pressure level, presence or absence of certain symptoms, etc.).Classification may be a widely held method for software defect prediction and categorizes the software code attributes into defective or not defective, which is completed by means of a classification model derived from software metrics data of previous development projects. Classification is kinds of supervised learning whereby mapping separate class for instances. The various classes are determined by the output, in ML which called a label (Kennedy, Credit Scoring Using Machine Learning, 2013) [16]. In classification decision tree is that the popular learning and used for approximation of discrete target functions. The choice tree is employed to construct a tree structure that represents graphically the training software data. Classification won't to classify software faults in step with the feature of the item with relevance the predefined set of classes. classification refers to a predictive modeling problem where a class label is predicted for a given example of input data. Examples of classification problems include: Given an example, classify if it is spam or not. Given a handwritten character, classify it as one of the known characters.

3.3 Regression Methods

Machine learning is that study of algorithms that may learn from data and make predictions, by building a model from example inputs instead of following static instructions. These algorithms are typically classified into three categories: supervised learning, unsupervised learning and reinforcement learning. In supervised learning, the system is presented with example inputs and outputs, with the aim of manufacturing a function that maps the inputs to outputs. Regression and classification problems are the two main classes of supervised learning (Kennedy, K. (2013)). Unsupervised learning is worried with leaving the system to seek out a structure supported the inputs, hopefully finding hidden patterns.

Given a training data set comprising observations $\{\mathbf{x}n\}$, where $n = 1, \ldots, N$, together with corresponding target values $\{tn\}$, the goal is to predict the value of t for a new value of $\mathbf{x}$. In the simplest approach, this can be done by directly constructing an appropriate function $y(\mathbf{x})$ whose values for new inputs $\mathbf{x}$ constitute the predictions for the corresponding values of t. More generally, from a probabilistic perspective, we aim to model the predictive distribution $p(t|\mathbf{x})$ because this expresses our uncertainty about the value of t for each value of $\mathbf{x}$. From this conditional distribution
we can make predictions of t, for any new value of $\mathbf{x}$, in such a way as to minimize the expected value of a suitably chosen loss function. loss, for which the optimal solution is given by the conditional expectation of t. Although linear models have significant limitations as practical techniques for pattern recognition, particularly for problems involving input spaces of high dimensionality [42].

3.4 Data Collections

For this study, we collect more than 8000 records of datasets house presented for sale from two well-known housing websites in Ethiopia (et.loozap and Qefira). To extract website data, we use data miner (automatic browser data scraper from Chrome extension) to change website data to excel CSV format. After we clean all unreliable data, missed value, duplicate records, and other unnecessary values only remain 5402 records of datasets. These houses are selected from Addis Ababa city house presented for sale from 2017 to the 2020 year. Figure 3.3 shows the geographical locations of this city from the Ethiopia google map along with their corresponding city.

This raw data is extracted from different Addis Ababa sub-city.

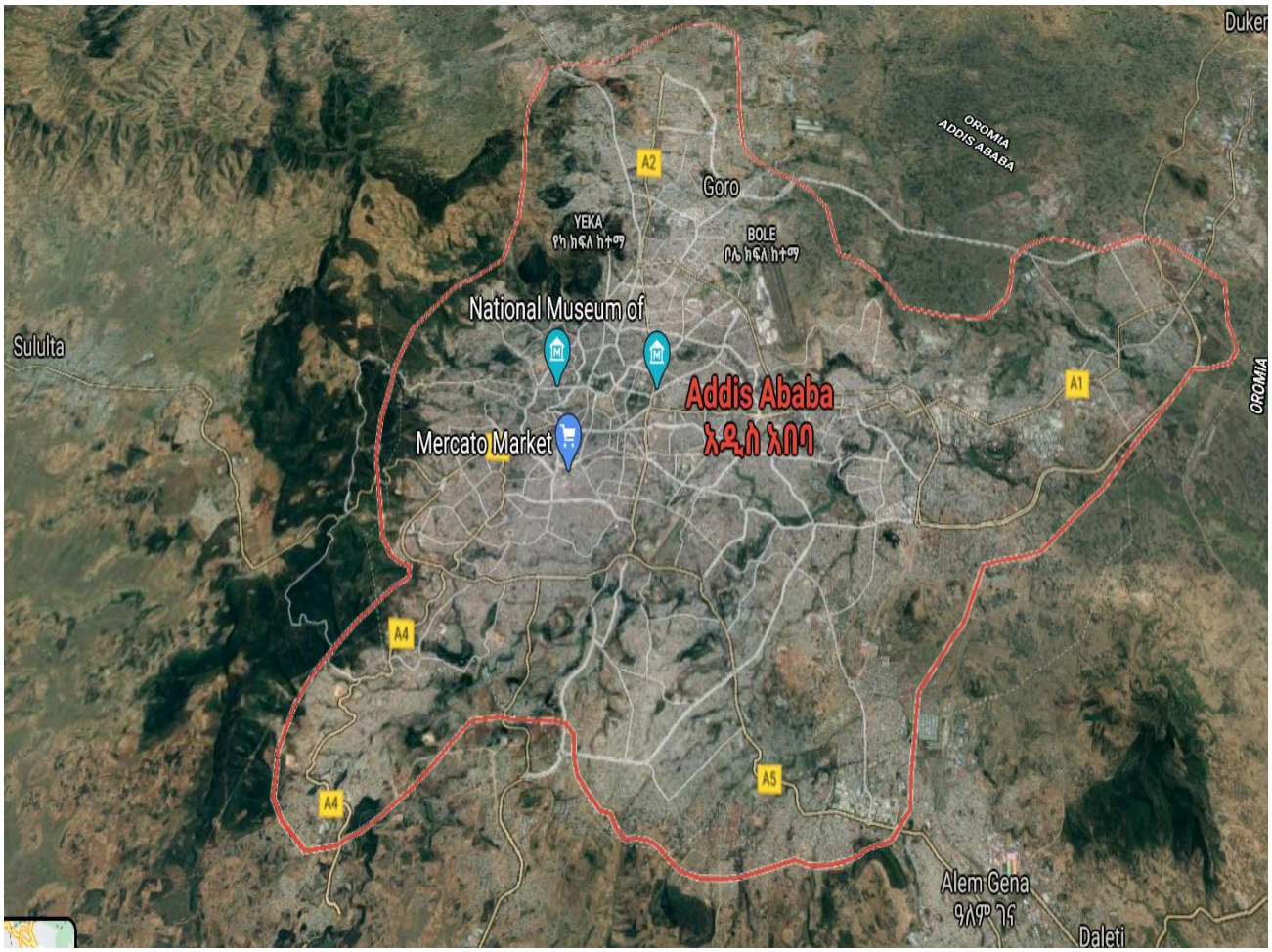

Figure 3. 3 Addis Ababa city map from google map

3.5 Data Processing

In Ethiopia, there are several housing websites (ethiobethoch, qefira, realethio, zegebeya, elfegn, ethiopianhome, etc.) that used to advert items including houses for their customer. But for this study, I select only two websites (qefira and et. Loozap) because all other websites miss either of the important houses attributes such as the total size of house area, house price, posted date (house presented date for sale). I extract all list of houses posted on two websites (qefira and et. loozap). Figure 3.4, which shows how house details listed from qefira website by creating different house features. The figure shows the total area of house, number of bedroom, city, specific location, Utility, Fence, Number bath room, Street, date presented for sale and price of the house. I extract all this house attribute information by using data miner to convert the website list of data to excel. All data processing steps are done in Excel's CSV format.

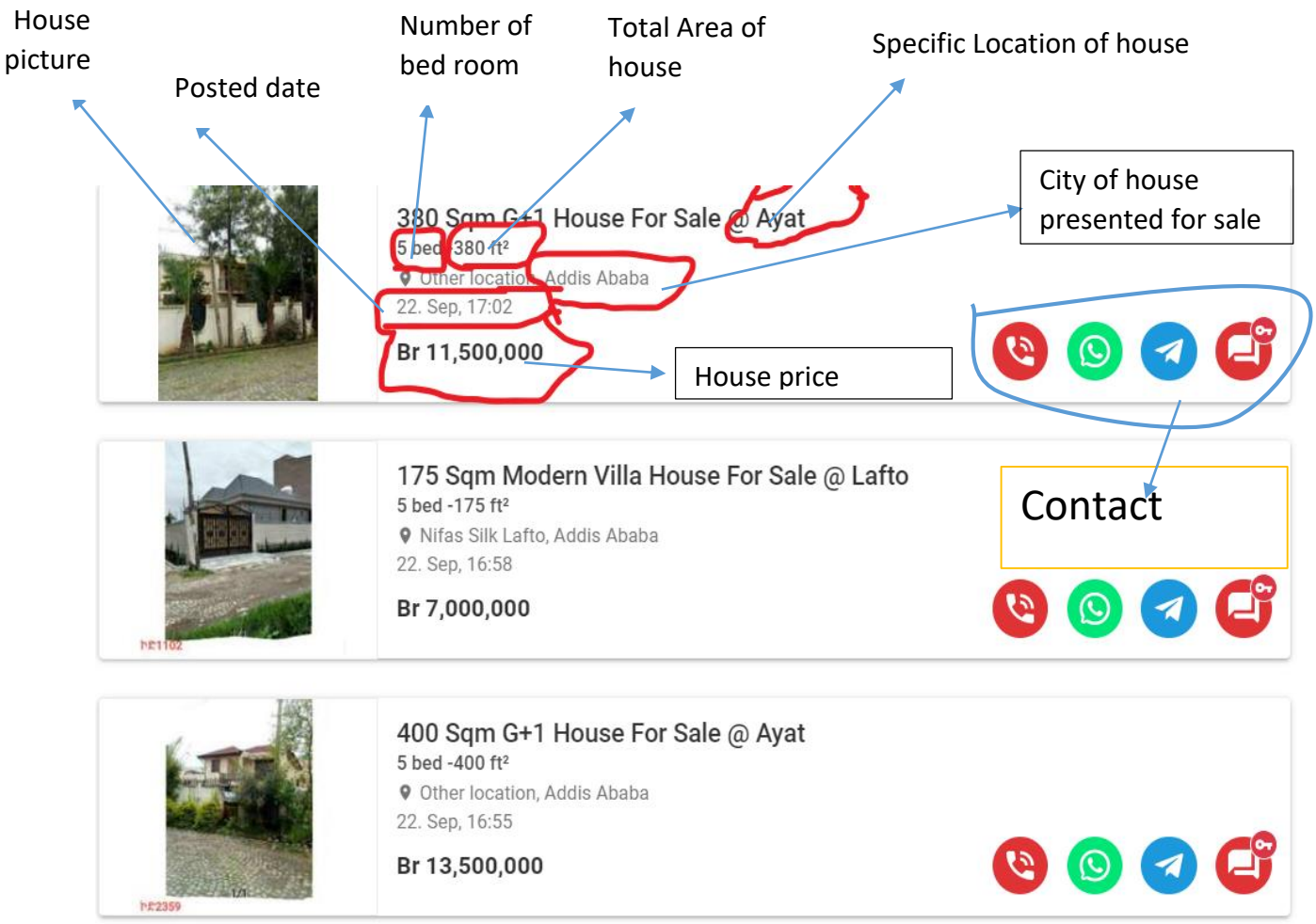

Figure 3. 4 house attributes from qefira website

3.6 Data cleaning

Data in raw form are not good for analysis. To be useful for predictive modeling the data must first be cleaned. The purpose of data cleaning is to improve the quality of data by identifying and removing errors and inconsistencies from data. Data quality problems are often present due to mistakes during data entry, missing information, duplicates, redundant data, or other general invalid data. The data cleaning process includes removing missing or duplicate information, filtering unnecessary data, and combining different data representations to provide access to consistent and accurate data. We removed all data not meeting the following criteria:

- The house has a non-missing house sale price.
- The house has a non-missing house total area.
- The house has a non-missing house presented sale year.
- The house must be in Addis Ababa.
- The house was presented for sale between 2017 and November 2020 year.
- The house is a residential house, city home, or condominium.
- The house must be only presented for sale not for rent

The data in the CSV need to be checked whether it has any null values, as the

data source has missing values every attribute has checked using the filters and missed/blank values are removed which helps to increase the accuracy level of the dataset. To remove invalid input from dataset we use different excel formula such as Conditional Formatting, **Remove** Duplicates, extract number only and etc.

3.7 Data description

After data cleaning, our data set consists of 5400 data records and Eleven (11) attributes (City, Specific location, total area, number of bedroom, Utility, Fence, Number of Bath room, Street, presented month for sale, presented year for sale, and price of the house) with complete information. The characteristic features or attributes of the house (Specific location, total area, number of bedroom, presented month for sale, presented year for sale) that are used to predict the house price are called independent or predictor variables, and house price is called target variable or dependent variable. The price of house is depending on other attributes listed above.

Here, we will provide a brief description of dataset features or attribute in the following table.

Feature/attribute	**Descriptions**
City	the city of house located presented for sales
SpecLoc	The specific location of house presented for sales
TotArea	The total area of house including free area
NuBedRoom	The Number of bed room house have
Utility	Type of utilities available(Electricity, water, network, etc.)
Fence	Fence type
NoBath	Number of Bed room house have
Street	Type of road access to the house
PrMoFrSal	House presented month for sales
PryrFrSal	Year of house presented for sale
price	The price of house in Ethiopian birr

Table 3. 1 Dataset Description

3.8 Data type

The housing data we used in this study have 5400 rows and six (11) attributes. The description of the data set data type is given below: -

Feature/attribute	**Data type**
City	String
SpecLoc	String
TotArea	Real number
NuBedRoom	Integer number
PrMoFrSal	Date
PryrFrSal	Date
price	Real number
Utility	String
Fence	String
NoBath	Natural number
Street	String

Table 3. 2 Dataset data type

Description of some Attribute values:

Utility: - AllPub All public Utilities (E, G, W, & S)

NoSewr Electricity, Gas, and Water (Septic Tank)

NoSeWa Electricity and Gas Only

ELO Electricity only

Fence: - GdPrv Good Privacy

MnPrv Minimum Privacy

GdWo Good Wood

MnWw Minimum Wood/Wire

NA No Fence

3.9 Sampling Techniques

For this study, I use probability sampling technique. To evaluate the performance of the model we observe the test error. The test error is estimated by holding out a subset of the training data during the fitting and then applying the fitted model to the held-_out subset. And we use the k-Fold Cross-Validation sampling technique to test and train our dataset.

I use a sampling technique in order to divide the dataset and each own utility. I divide the dataset as follows in figure 3.5:

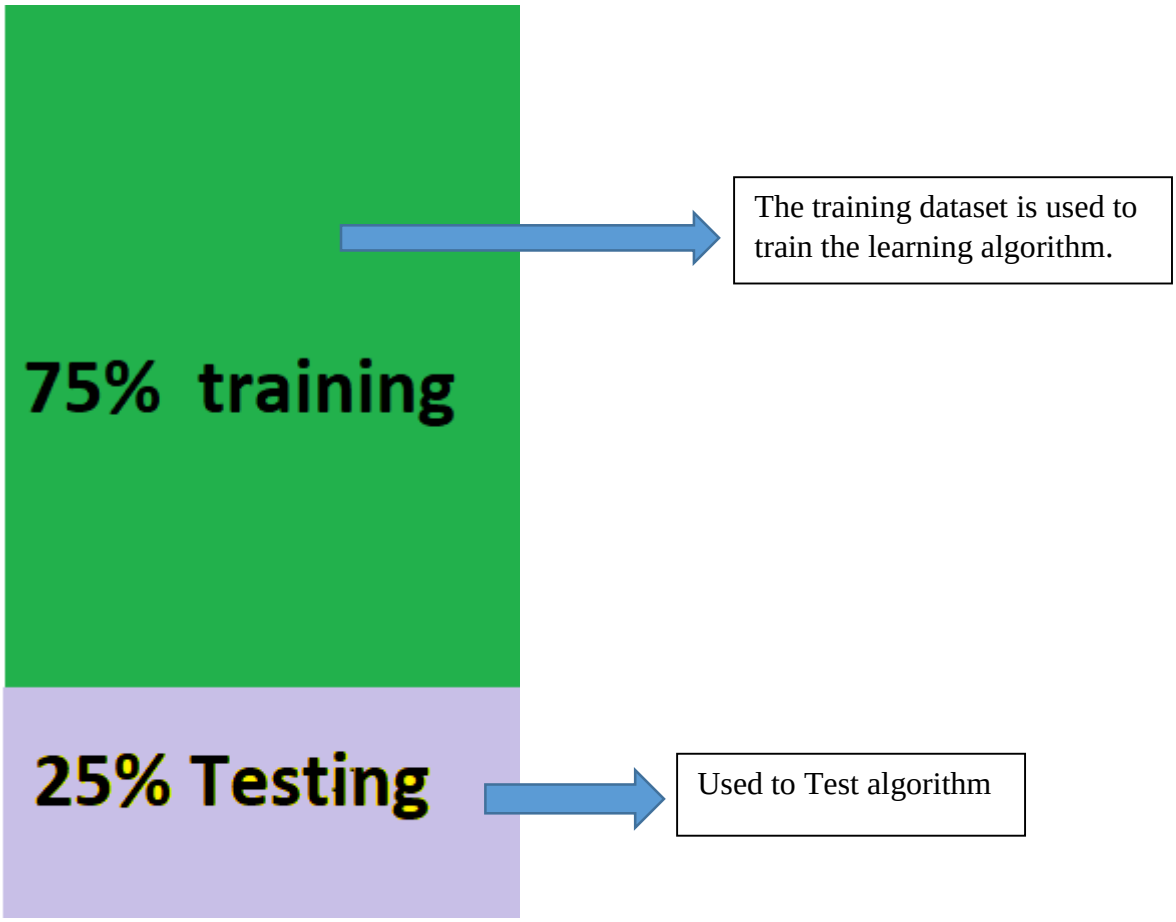

Figure 3. 5 sampling technique to dived the data in to training and testing dataset

Chapter Four: Experiment and Implementation:

4.1 Overview

In this chapter, we discuss the design and implementation of our prediction system for Addis Ababa housing prices, based upon a Regression machine learning algorithm process model. This includes how the dataset is being prepared, the implementation language and library used for training the dataset, the methodologies followed to analysis data, the sources of knowledge sets and the way predictions are made to suit the needs of the research. We also compare the accuracies of the models discussed within the previous chapter for predicting the sale price of residential family homes.

4.2 Prediction Techniques

In this section, we decide the type of machine learning prediction that's suitable to our problem. We would like to see if this can be a regression problem or a classification problem. We also predict the price of a house by using given information about it. The price we predict is a continuous value; it is often any real number. So we use regression machine learning to predict the price of house. Because the target variable (house price) is continues value. We see this during implementation section in details.

4.3 Modeling Approach

- Choose an algorithm that implements the corresponding technique
- explore for a good parameter combination for the chosen algorithm
- Create a model using the founded parameters
- Train the model on the training dataset
- Test the model on the test dataset and acquire results

4.4 Correlation between Variables

We want to work out how the dataset variables are correlated with one another and the way predictor variables are correlated with the target variable (price of house). For instance, TotArea and price are highly positively correlated. Which means when total area of house increases the price of house also increase. And also as house presented date increase the price of house also increase. We are able to see that these variables include categorical variables, numerical variables, and statistic variables. We see the details correlation between variable during implementation section by using hit map graph.

4.5 Reading the Dataset

The first step is reading the dataset from the csv file we save. We are going to use the read_csv () function from Pandas Python package:

DS = pd.read_csv('DatasetAA.csv')

4.6 Getting Dataset

Let's display the first a small number of rows from dataset:

DS. head ()

[3]:

	City	SpecLoc	TotArea	NuBedRoom	Utility	Fence	NoBath	Street	PrMoFrSal	PrYrFrsal	Price
0	Addis Ababa	abado	150.0	NaN	AllPub	No Fence	2	Pave	7	2020	2500000
1	Addis Ababa	abado	150.0	NaN	AllPub	NaN	2	Pave	5	2020	4356000
2	Addis Ababa	abuare	450.0	NaN	AllPub	NaN	2	Pave	9	2020	17000000
3	Addis Ababa	abuare	450.0	NaN	AllPub	NaN	1	Pave	8	2020	20000000
4	Addis Ababa	addissefer	600.0	8.0	AllPub	NaN	2	Pave	10	2020	30000000

Table 4. 1 Some list of dataset

Now, let's get statistical information about the numeric columns in our dataset. We would like to understand the mean, the standard deviation, the minimum, the maximum, and also the 50th percentile (the median) for every numeric column within the dataset:

DS.describe (include=[np.number],percentiles=[.5])\.transpose().drop("count",axis=1)

[4]:

	mean	std	min	50%	max
TotArea	2.347272e+02	1.527908e+02	43.0	200.0	1500.0
NuBedRoom	4.728425e+00	2.086851e+00	1.0	4.0	18.0
NoBath	1.564340e+00	5.514110e-01	0.0	2.0	4.0
PrMoFrSal	8.082577e+00	3.252155e+00	1.0	9.0	12.0
PrYrFrsal	2.019933e+03	3.208025e-01	2017.0	2020.0	2020.0
Price	8.412888e+06	7.258053e+06	780000.0	6000000.0	36000000.0

Table 4. 2 statistical information about the numeric columns in our dataset

From the table above, we can see, as an example, that the typical TotArea of the house in our dataset have mean of 234.72m^2 with a standard deviation of 152.7m^2. We are able to see that the minimum total area is 43m^2 and the maximum lot area is 1500^2 with a median of 200m^2. Similarly, we will get lots of knowledge about our dataset variables from the table. Then, we move to determine statistical information about the non-numerical columns in our dataset:

```
DS.describe(include=[np.object]).transpose() \
drop ("count", axis=1)
```

[5]:

	unique	top	freq
City	1	Addis Ababa	5401
SpecLoc	95	summit	451
Utility	2	AllPub	5398
Fence	5	No Fence	1078
Street	2	Pave	5378

Table 4. 3 statistical information about the non-numerical columns in our dataset

In the above table we got, count represents the amount of non-null values in each column, unique represents the quantity of unique values, top represents the foremost frequent element, and freq represents the frequency of the foremost frequent element.

4.7 Addressing with Missing Values

We should deal to the problem of missing values because machine learning models don't accept data with missing values. Firstly, let's see the amount of missing values in our dataset. We would like to determine the quantity and therefore the percentage of missing values for every column that really contains missing values.

```
missing = DS.isna().sum()
missing = missing [missing > 0]
perce_missing = missing * 100 / DS.shape[0]
pd.concat([missing, perce_missing], axis=1,
keys=['Missing Values', 'Percentage']).\
sort_values(by="Missing Values", ascending=False)
```

[6]:

	Missing Values	Percentage
NuBedRoom	3744	69.320496
SpecLoc	2202	40.770228

Table 4. 4 Missed values from column of dataset

Now we start handling these missing values.

SpecLoc: NA in SpecLoc columns represents the specific location does not exist or not described in the dataset and in house website. So we fill within the missing values in these columns with specific location not indicated (“not indicated”).

NuBedRoom: - most of the house we scrap from housing website are not describe the number of bed (missing value). So We’ll fill this value with Zero (0).

```
DS['NuBedRoom'].fillna(0, inplace=True)
DS['SpecLoc'].fillna("Not Indicated", inplace=True)
DS['Fence'].fillna("No Fence", inplace=True)
DS['Utility'].fillna("No Utility", inplace=True)
```

Now let’s check if there’s any remaining missing value in our dataset:

```
DS.isna().values.sum()
```

```
0
```

This means that our dataset is now complete; it doesn’t contain any missing value anymore.

4.8 Investigating Data Analysis

In this section, we'll explore the information using visualizations. This can allow us to know the information and also the relationships between variables better, which is able to help us build an improved model. Our dataset contains seven variables, but the foremost important one for us to explore is that the target variable. We'd like to know its distribution. First, we start by plotting the violin plot for the target variable. The width of the violin represents the frequency.

```
sns.violinplot(x=DS['Price'], inner="quartile", color="#96B37E");
```

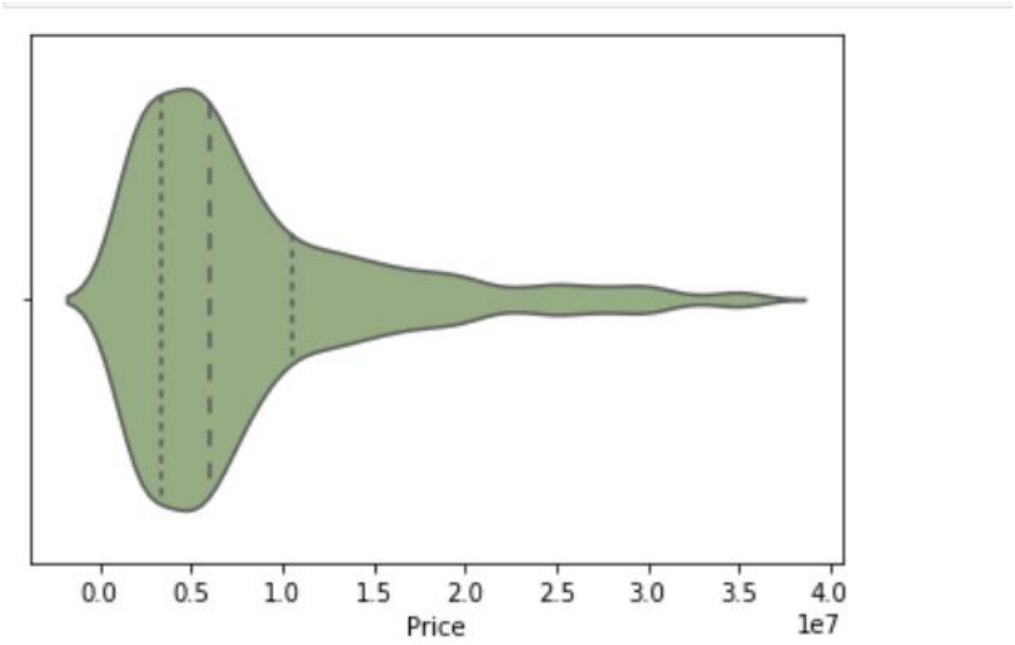

Figure 4. 1 Target variable distribution using violin

We can see from the plot that most house prices fall between 1,000,000 and 15,000,000. The dashed lines represent the locations of the three quartiles Q1, Q2 (the median), and Q3.

Now let's see the box plot of SalePrice:

```
sns.boxplot(DS['Price'], whis=10, color="#D008A9");
```

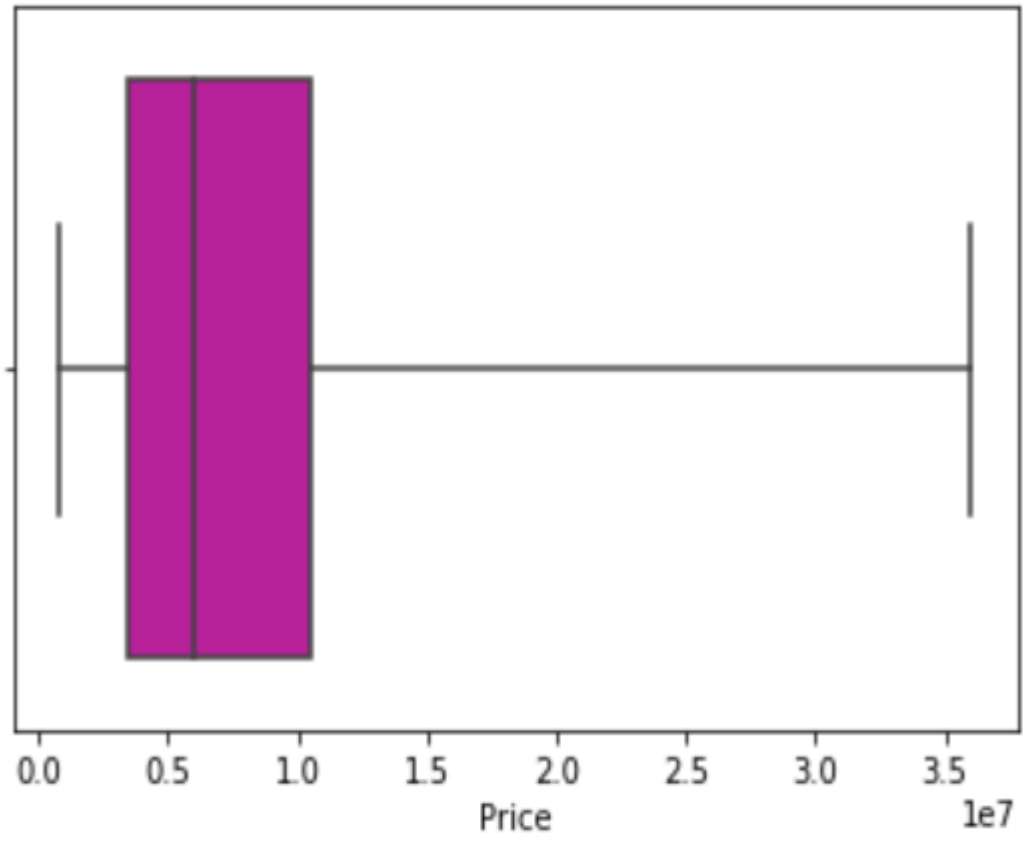

Figure 4. 2 minimum and maximum values of house Price (Target variable).

This shows us the minimum and maximum values of house price. It shows us also the three quartiles represented by the box and also the vertical line within it. Lastly, we plot the histogram of the variable to determine a more detailed view of the distribution:

```
sns.distplot(DS['Price'], kde=False,
color="#172B4D", hist_kws={"alpha": 0.8});
plt.ylabel("Count");
```

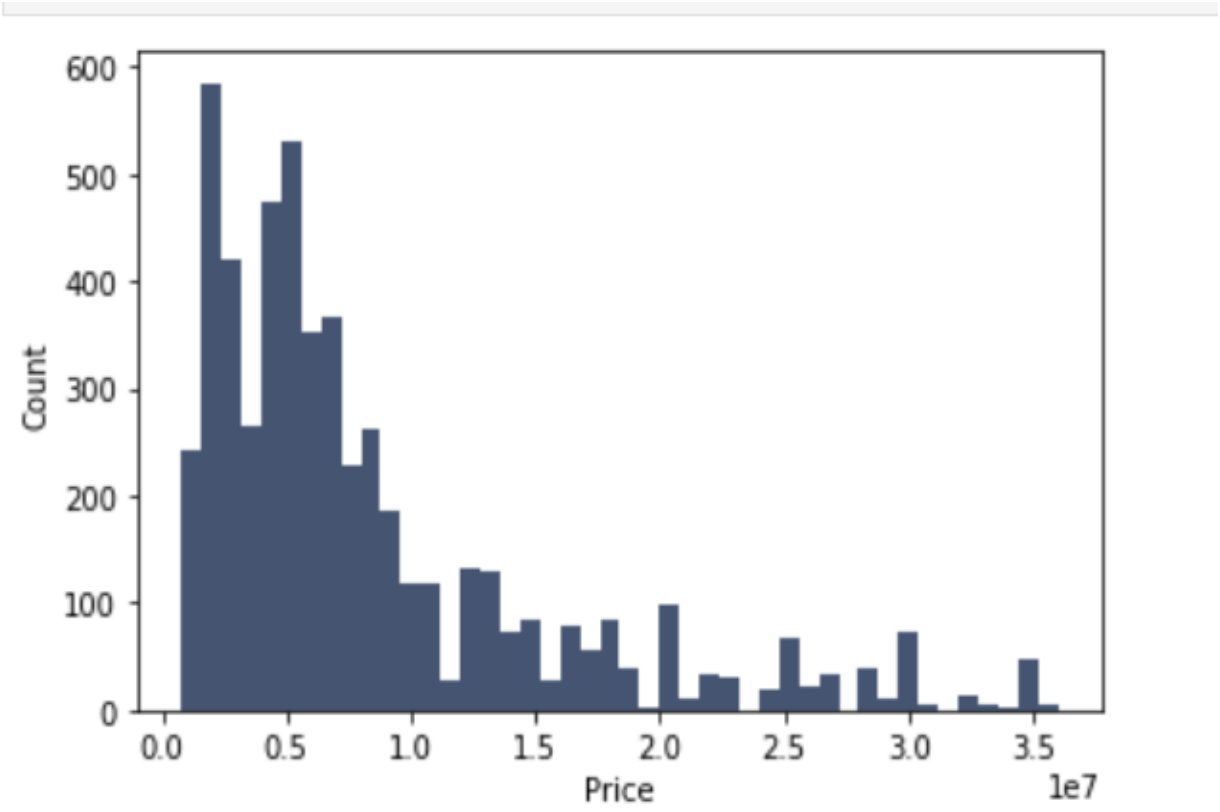

Figure 4. 3 detailed distribution of target variable (price) using histogram

4.9 Correlation between Variables

We want to determine how the dataset variables are correlated with one another and the way predictor variables are correlated with the target variable (house price). As an example, we'd prefer to see how TotArea and Price are related: Do they increase and decrease together (positive correlation)? Does one of them increase when the other decrease or vice versa (negative correlation)? Or are they not correlated?

We will show relation between our dataset variables (numerical and Boolean variables only) by using a heat map graph as follow:

sns.heatmap(DS.corr(),cmap="YlGnBu");

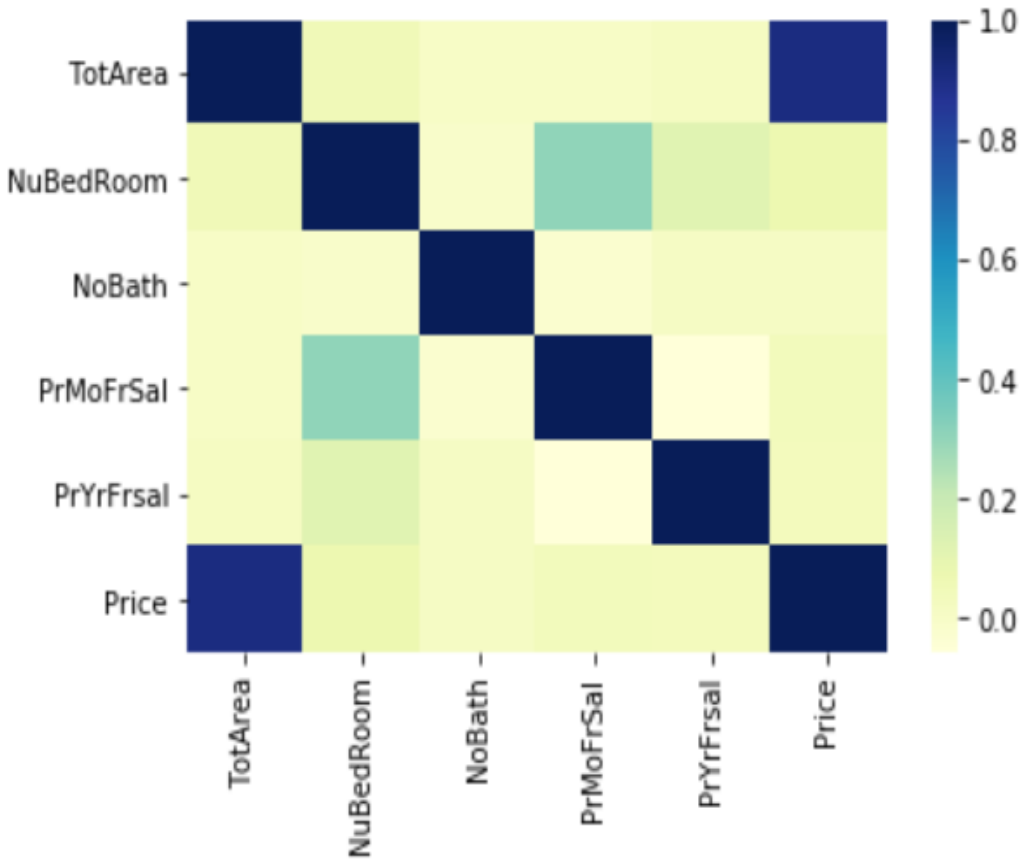

Figure 4. 4 Correlation between Variables using heat map graph

We can see that there are many correlated variables in our dataset. We notice that TotArea and price are highly positively correlated which means when number of total area of house increase, it's expected for the price of the house also increase too. Most significantly, we would like to appear at the predictor variables that are correlated with the target variable (Price). By watching the last row of the heatmap, we see that the target variable is extremely positively correlated with total area of house.

High correlation Firstly, we would like to visualize the relationships between the target variable and other variables that are highly and positively correlated with it, consistent with what we saw within the heat map. We start with the relationship between the target variable and TotArea, but before that, let’s see the distribution of every of them. We can see that the majority house prices fall between 1,000,000 and 15,000,000.

We discuss that there's variety of high-priced houses to the proper of the plot. Now, we move to work out the distribution of TotArea variable:

```
sns.distplot(DS['TotArea'], kde=False,
color="#000B9B", hist_kws={"alpha": 0.8});
plt.ylabel("Count");
```

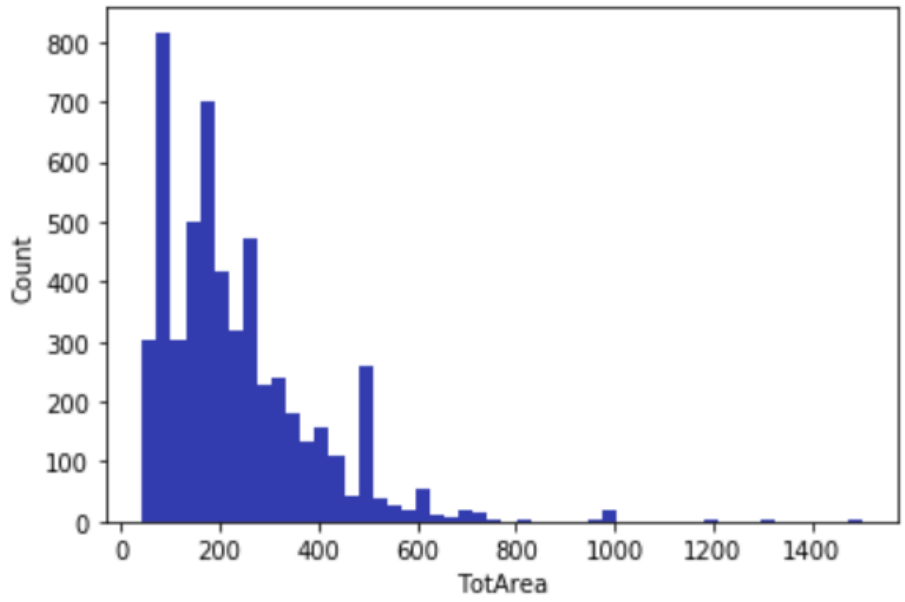

Figure 4. 5 distribution of TotArea variable by histogram graph

We see that TotArea takes an integer value between 43 and 1500, which most houses have total area between 43 and 1500. Now we plot the scatter plot of Price and TotArea to determine the relationship between them:

```
plt.scatter(x=DS['TotArea'], y=DS['Price'],
color="orange", edgecolors="#000000", linewidths=0.5);
plt.xlabel("TotArea"); plt.ylabel("Price");
```

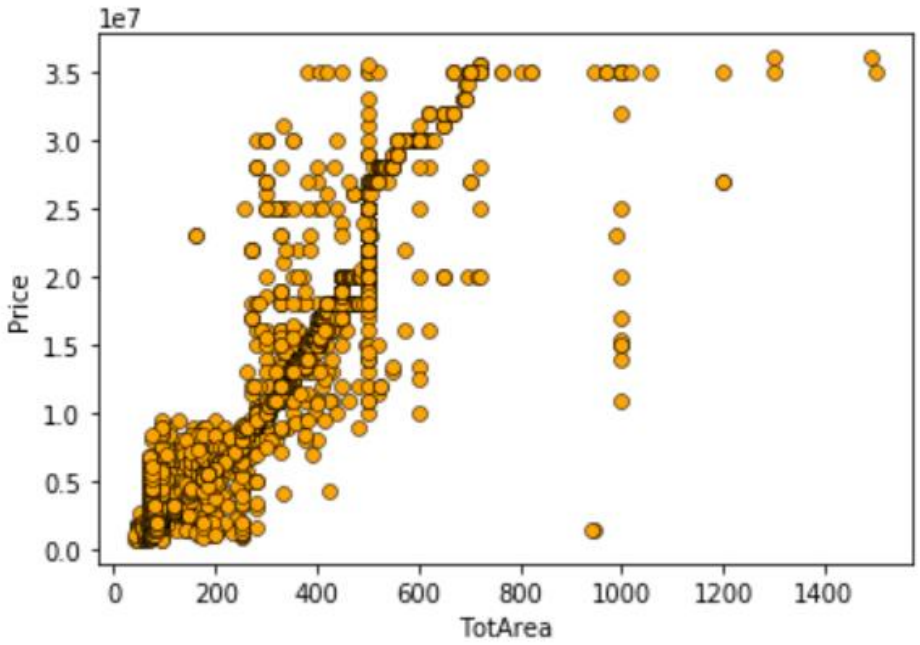

Figure 4. 6 relationship between TotArea and target variable (Price)

We can see that they're truly positively correlated; generally, total area increases, the house price increases too. This verifies what we got from the heatmap above. Now, we will to determine the relation between the target variable and total area variable which represents the total area. Let us to first see the distribution of total area:

```
sns.distplot(DS['TotArea'], kde=False,
color="#000B9B", hist_kws={"alpha": 0.8});
plt.ylabel("Count")
```

```
[15]: Text(0, 0.5, 'Count')
```

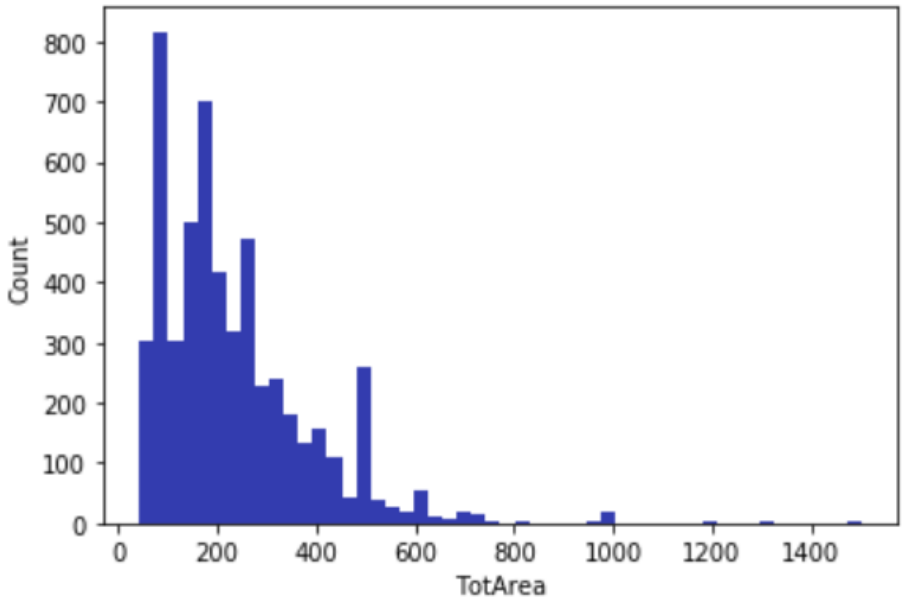

Figure 4. 7Distribution of TotArea

We can see that the above graph total area falls approximately between 43m^2 and 1500m^2.

Moderate correlation Next: we would like to visualize the correlation between the target variable and other variables that are positively correlated with it, but the correlation isn't very strong. Namely, these variables are house presented month for sale and presented year for sale. Allow us to see the distribution of every of them:

```
fig, axes = plt.subplots(1, 2, figsize=(18,5))
fig.subplots_adjust(hspace=0.5, wspace=0.6)
for ax, v in zip(axes.flat, ["PrMoFrSal", "PrYrFrsal"]):sns.distplot(DS[v], kde=False,
color="#DA0B4B",hist_kws={"alpha": 0.8},
```

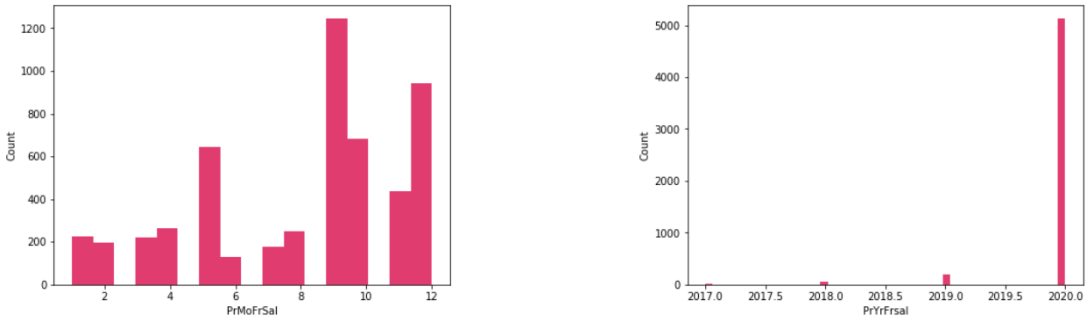

Figure 4. 8 Distribution of house PrMoFrSal and PrYrFrSal

Now allow us to see their relationships with the target variable using scatter plots:

```
x_vars = ["PrMoFrSal", "PrYrFrsal"]
g = sns.PairGrid(DS, y_vars=["Price"], x_vars=x_vars);
g.map(plt.scatter, color="green", edgecolors="#000000", linewidths=0.5);
```

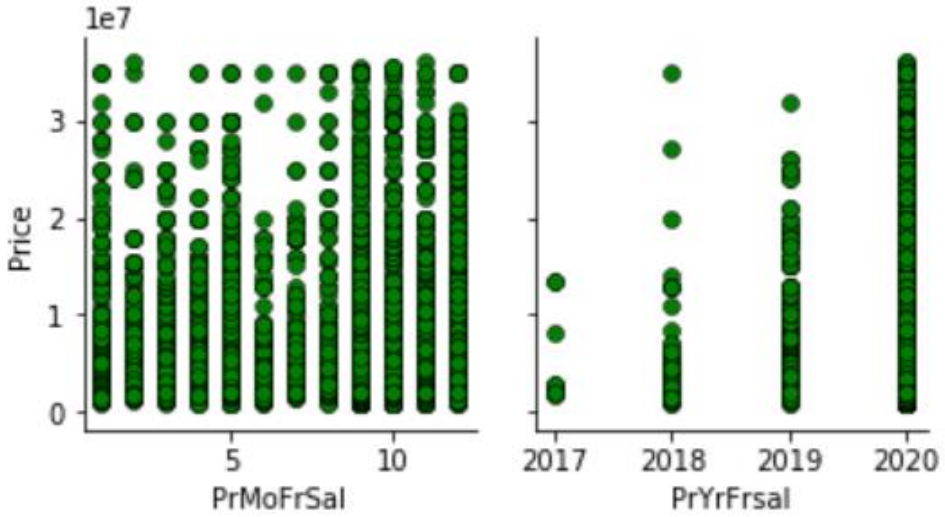

Figure 4. 9 Relationship between Distribution of PreMoFrSal, PreYFrSal and Target variable

4.11 One-Hot Encoding for Categorical Features

Machine learning models accept only numbers as input, and since our dataset contains categorical features, we like to encode them so as for our dataset to be suitable for modeling. We encode our categorical features using one-hot encoding technique which transforms the explicit variable into variety of binary variables supported the amount of unique categories within the categorical variable; each of the resulting binary variables has only 0 and 1 as its possible values. Pandas package provides a convenient function get_dummies () that may be used for performing one-hot encoding on our dataset.

4.12 Prediction Type and Modeling Techniques

In this section, we decide the kind of machine learning prediction that's suitable to our problem. We would like to work out if this can be a regression problem or a classification problem. During this study, we would like to predict the price of a house given information about it. The price of the house predict could be a continuous value; it may be any real number. This may be seen by looking the target variable in our dataset SalePrice:

```
DS['Price'].head()
```

```
[20]: 0     2500000
      1     4356000
      2    17000000
      3    20000000
      4    30000000
      Name: Price, dtype: int64
```

Table 4. 5 first five list of house Price from dataset

Us we see the above table 4.5 the price of the house is real number. That means that the prediction type that's appropriate to our problem is regression. Now we move to match the modeling techniques we chose with other regression algorithm. There are a lot of techniques available for regression problems like linear regression (elastic Regression, Ridge Regression), KNN, decision tree, Neural Networks, random forest, etc.

In this study, we test all the above modeling techniques, and that we choose the regression technique that produce the most effective results.

4.12.1 Linear Regression

This technique models the relation between the target variable and independent variables (predictors). It fits a linear model with coefficients to the data in order to minimize the residual sum of squares between the target variable in the dataset, and the predicted values by the linear approximation [17].

4.12.2 Decision Trees

For this system, the goal is to form a model that predicts the value of a target variable by learning simple decision rules inferred from the information features [17].

4.12.3 K-nearest Neighbors

The nearest Neighbors could be a variety of instance-based learning. For this method, the model tries to seek out variety (k) of coaching examples closest in distance to a brand new point, and predict the output for this new point from these closest neighbors. K may be a user-defined number (k-nearest neighbors), or vary supported the local density of points (radius-based neighbors). The space metric wont to measure the closeness is generally the Euclidean distance [17].

4.12.4 Artificial Neural Networks

Neural network is a machine learning model that tries to mimic the way of working of the biological brain. A neural network consists of multiple layers. Each layer consists of a number of nodes. The nodes of each layer are connected to the nodes of adjacent layers. Each node can be activated [17].

4.12.5 Random Forest

Bagging is an ensemble method where many base models are used with a randomized subset of data to reduce the variance of the base model [17].

4.13 Feature Scaling

In order to form all algorithms, work properly with our data, we want to scale the features in our dataset. For that, we use a helpful function named Standard Scaler () from the popular Scikit-Learn Python package. This function standardizes features by subtracting the mean and scaling to unit variance. It works on each feature independently.

Scikit-learn: was initially developed by David Cournapeau as a Google summer of code project in 2007. It deals with learning information from one or more datasets that are represented as 2D arrays. They can be understood as a list of multi-dimensional observations. We say that the first axis of these arrays is the samples axis, while the second is the features axis.

scikit-learn comes with a few standard datasets, for instance the iris and digits datasets for classification and the diabetes dataset for regression.

4.14 Splitting the Dataset

As usual for supervised machine learning problems, we want a training dataset to train our model and a test dataset to scale the model. So we'll split our dataset randomly into two parts, one for training and then other for testing. For that, we use another function from Scikit-Learn called train_test_split():

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(DS.drop('Price', axis=1),
DS[['Price']],test_size=0.25, random_state=2)
```

We specified the dimensions of the test set to be 25% of the entire dataset. This leaves 75% for the training dataset. Now we have four subsets:
X_train, X_test, y_train, and y_test. Later we will use X_train and y_train to train our model, and X_test and y_test to test and evaluate the model. X_train and X_test represent features (predictors); y_train and y_test represent the target.
From now on, we'll check with X_train and y_train because the training dataset, and to X_test and y_test because the test dataset.

	Feature1	Feature2	Target		
	1	2	6		
X_train	4	1	8		Y_train
	2	3	7		
	6	2	7		
Y_test	2	6	9		Y_test

Table 4. 6 train_test_split operation

Chapter Five: Modeling and result

After we split our dataset, we build model for each one of the techniques we mention in the previous section (Linear Regression, Nearest Neighbor, Artificial Neural Network, random forest, decision tree etc.), we follow these steps to build a model:

- Select an algorithm that implements the corresponding technique
- Search for an effective parameter combination for the chosen algorithm
- Create a model using the founded parameters
- Train (fit) the model on the training dataset
- Test the model on the test dataset and get the results

As we mentioned before we already select regression algorithm. Because our target value (house sales price) is continuous number.

5.1 Searching for Effective Parameters

Using Scikit-Learn, we build a decision-tree model for example as follows: model = DecisionTreeRegressor(max_depth=5, min_samples_split=4 max_features=5) We can do this but to probably achieve a better performance if we choose better values for the parameters max_depth, min_samples_split, and max_features. To do so, we are going to examine many parameter combinations and choose the combination that gives ithe best score. Scikit-Learn provides a useful function for that purpose: GridSearchCV (). So for the example above, we will do the following:

```
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RandomizedSearchCV
from sklearn.tree import DecisionTreeRegressor
parameter_space = {
"max_depth": [4, 5],
"min_samples_split": [3, 4],
"max_features": [4, 5]
}
```

The code above will test the decision-tree model using all the parameter combinations. It will use cross validation with 4 folds and it will use the mean absolute error for scoring and comparing different parameter combinations. At the end, it will provide us with the best parameter combination that achieved the best score so we are able to use it to build our model. Sometimes, when the number of parameter combinations is large, GridSearchCV () can take very long time to run. So in addition to GridSearchCV (), we will sometimes use RandomizedSearchCV () which is similar to GridSearchCV () but instead of using all parameter combinations, it picks a number of random combinations specified by n_iter.

This will make RandomizedSearchCV () pick 100 parameter combinations randomly

For this study we test many modeling techniques, and then we select the technique that produce the best results. The techniques that we will try are defined in the following section:

5.2 Linear Regression

For Linear Regression, we will choose two algorithmic implementations: Ridge Regression and Elastic Net. We will use the implementations provided in the Scikit-Learn package of these algorithms.

5.2.1 Ridge Regression

Firstly, we will use GridSearchCV () to search for the simplest model parameters in a parameter space provided by us. The parameter alpha represents the regularization strength, fit_intercept determines whether to calculate the intercept for this model, and solver controls which solver to use in the computational routines.

```
From sklearn.linear_model import Ridge
parameter_space = {
"alpha": [1, 10, 100, 290, 500],
"fit_intercept": [True, False],
"solver": ['svd', 'cholesky', 'lsqr', 'sparse_cg', 'sag', 'saga'],
}
```

```
clf = GridSearchCV(Ridge(random_state=3), parameter_space, n_jobs=4,
cv=3, scoring="neg_mean_absolute_error")
clf.fit(X_train, y_train)
print("Best parameters:")
print(clf.best_params_)
```

We defined the parameter space above using reasonable values for chosen parameters. Then we used GridSearchCV() with 3 folds (cv=3). Now we build our Ridge model with the best parameters found: Then we train our model using our training set (X_train and y_train):

```
ridge_model = Ridge(random_state=3, **clf.best_params_)
ridge_model.fit(X_train, y_train);
```

Finally, we test our model on X_test. Then we evaluate the model performance by comparing its predictions with the actual true values in y_test using the MAE metric as we described above:

```
from sklearn.metrics import mean_absolute_error
y_pred = ridge_model.predict(X_test)
ridge_mae = mean_absolute_error(y_test, y_pred)
print("Ridge MAE =", ridge_mae)
print("Ridge MAPE=",(ridge_mae/mn)*100,"%")
```

Ridge MAE = 1472549.94

Ridge MAPE= 17.50 %

5.2.2 Elastic Net

Firstly, we will use GridSearchCV () to search for the best model parameters in a parameter space provided by us. The parameter alpha is a constant that multiplies the penalty terms, l1_ratio determines the amount of L1 and L2 regularizations, fit_intercept is the same as Ridge's.

```
from sklearn. linear_model import ElasticNet
parameter_space = {
"alpha": [1, 10, 100, 280, 500],
"l1_ratio": [0.5, 1],
"fit_intercept": [True, False],
}
clf = GridSearchCV(ElasticNet(random_state=3), parameter_space,
n_jobs=4, cv=3, scoring="neg_mean_absolute_error")
clf.fit(X_train, y_train)
print("Best parameters:")
print(clf.best_params_)
```

We defined the parameter space above using reasonable values for chosen parameters. Then we used GridSearchCV() with 3 folds (cv=3). Now we build our Ridge model with the best parameters found:

```
elasticNet_model = ElasticNet(random_state=3, **clf.best_params_)
```

Then we train our model using our training set (X_train and y_train):

```
elasticNet_model.fit(X_train, y_train);
```

Finally, we test our model on X_test. Then we evaluate the model performance by comparing its predictions with the actual true values in y_test using the MAE metric as we described above:

```
y_pred = elasticNet_model.predict(X_test)
elasticNet_mae = mean_absolute_error(y_test, y_pred)
print("Elastic Net MAE =", elasticNet_mae)
print("Elastic Net MAPE=",(elasticNet_mae/mn)*100,"%")
```

Elastic Net MAE = 1470060.97

Elastic Net MAPE= 17.47 %

5.3 K-Nearest Neighbors

For Nearest Neighbors, we will use an implementation of the k-nearest neighbors (KNN) algorithm provided by Scikit-Learn package.

Firstly, we will use GridSearchCV () to search for the best model parameters in a parameter space provided by us. The parameter n_neighbors represents k which is the number of neighbors to use, weights determines the weight function used in prediction: uniform or distance, algorithm specifies the algorithm used to compute the nearest neighbors, leaf_size is passed to BallTree or KD Tree algorithm. It can affect the speed of the construction and query, as well as the memory required to store the tree.

```
from sklearn. neighbors import KNeighborsRegressor
parameter_space = {
"n_neighbors": [9, 10, 11,50],
"weights": ["uniform", "distance"],
"algorithm": ["ball_tree", "kd_tree", "brute"],
"leaf_size": [1,2,20,50,200]
}
clf = GridSearchCV(KNeighborsRegressor(), parameter_space, cv=3,
scoring="neg_mean_absolute_error", n_jobs=4)
clf.fit(X_train, y_train)
print("Best parameters:")
print(clf.best_params_)
```

We defined the parameter space above using reasonable values for chosen parameters. Then we used GridSearchCV() with 3 folds (cv=3). Now we build our Ridge model with the best parameters found: Then we train our model using our training set (X_train and y_train):

```
knn_model = KNeighborsRegressor(**clf.best_params_)
knn_model.fit(X_train, y_train);
```

Finally, we test our model on X_test. Then we evaluate the model performance by comparing its predictions with the actual true values in y_test using the MAE metric as we described above:

```
y_pred = knn_model.predict(X_test)
knn_mae = mean_absolute_error(y_test, y_pred)
print("K-Nearest Neighbors MAE =", knn_mae)
print("K-Nearest Neighbors MAPE=",(knn_mae/mn)*100,"%")
```

K-Nearest Neighbors MAE = 864142.58

K-Nearest Neighbors MAPE= 10.27 %

5.4 Decision Tree

For Decision Tree (DT), we will use an implementations provided by the Scikit-Learn package.

The Decision Tree model has the following syntax:

Firstly, we will use GridSearchCV () to search for the best model parameters in a parameter space provided by us. The parameter criterion specifies the function used to measure the quality of a split, min_samples_split determines the minimum number of samples required to split an internal node, min_samples_leaf determines the minimum number of samples required to be at a leaf node, and max_features controls the number of features to consider when looking for the best split.

```
from sklearn.tree import DecisionTreeRegressor
parameter_space = \
{
"criterion": ["mse", "friedman_mse", "mae"],
"min_samples_split": [3, 9, 15, 25],
"min_samples_leaf": [2, 4, 8, 13],
"max_features": [10, 25, 75, 100, X_train.shape[1]],
}
clf = GridSearchCV(DecisionTreeRegressor(random_state=3), parameter_space,
cv=3, scoring="neg_mean_absolute_error", n_jobs=4)
clf.fit(X_train, y_train)
print("Best parameters:")
print(clf.best_params_)
```

We defined the parameter space above using reasonable values for chosen parameters. Then we used GridSearchCV() with 3 folds (cv=3). Now we build our Decision Tree model with the best parameters found:

dt_model = DecisionTreeRegressor(**clf.best_params_)

Then we train our model using our training set (X_train and y_train):

dt_model.fit(X_train, y_train);

Finally, we test our model on X_test. Then we evaluate the model performance by comparing its predictions with the actual true values in y_test using the MAE metric as we described above:

y_pred = dt_model.predict(X_test)

dt_mae = mean_absolute_error(y_test, y_pred)

Print ("Decision Tree MAE =", dt_mae)

print("Decision Tree MAPE=",(dt_mae/mn)*100,"%")

Decision Tree MAE = 825248.92

Decision Tree MAPE= 9.81 %

5.5 Neural Network (NN)

For Neural Network (NN), we will use an implementations provided by the Scikit-Learn package. The Neural Network model has the following syntax:

MLPRegressor(hidden_layer_sizes=(100,), activation=relu, solver=adam, alpha=0.0001, batch_size=auto, learning_rate=constant, learning_rate_init=0.001, power_t=0.5, max_iter=200, shuffle=True, random_state=None, tol=0.0001, verbose=False, warm_start=False, momentum=0.9, nesterovs_momentum=True, early_stopping=False, validation_fraction=0.1, beta_1=0.9, beta_2=0.999, epsilon=1e-08, n_iter_no_change=10)

Firstly, we will use GridSearchCV() to search for the best model parameters in a parameter space provided by us. The parameter hidden_layer_sizes is a list where its ith element represents the number of neurons in the ith hidden layer, activation specifies the activation functionfor the hidden layer, solver determines the solver for weight optimization, and alpha represents L2 regularization penalty.

```
from sklearn.neural_network import MLPRegressor
parameter_space = \
{
"hidden_layer_sizes": [(7,)*3, (19,), (100,), (154,)],
"activation": ["identity", "logistic", "tanh", "relu"],
"solver": ["lbfgs"],
"alpha": [1, 10, 100],
}
clf = GridSearchCV(MLPRegressor(random_state=3), parameter_space,
cv=3, scoring="neg_mean_absolute_error", n_jobs=4)
clf.fit(X_train, y_train)
print("Best parameters:")
print(clf.best_params_)
```

We defined the parameter space above using reasonable values for chosen parameters. Then
we used GridSearchCV() with 3 folds (cv=3). Now we build our Neural Network model with the best parameters found:

```
nn_model = MLPRegressor(**clf.best_params_)
```

Then we train our model using our training set (X_train and y_train):

```
nn_model.fit(X_train, y_train);
```

Finally, we test our model on X_test. Then we evaluate the model performance by comparing
its predictions with the actual true values in y_test using the MAE metric as we described above:

```
y_pred = nn_model.predict(X_test)
nn_mae = mean_absolute_error(y_test, y_pred)
print("Neural Network MAE =", nn_mae)
print("Neural Network MAPE=",(nn_mae/mn)*100,"%")
```

Neural Network MAE = 1488718.18

Neural Network MAPE= 17.69 %

5.6 Random Forest

For Random Forest (RF), we will use an implementations provided by the Scikit-Learn package. The Random Forest model has the following syntax:

```
RandomForestRegressor(n_estimators=warn, criterion=mse, max_depth=None,
min_samples_split=2, min_samples_leaf=1,
min_weight_fraction_leaf=0.0, max_features=auto,
max_leaf_nodes=None, min_impurity_decrease=0.0,
min_impurity_split=None, bootstrap=True, oob_score=False,
n_jobs=None, random_state=None, verbose=0, warm_start=False)
```

Firstly, we will use GridSearchCV () to search for the best model parameters in a parameter space provided by us. The parameter n_estimators specifies the number of trees in the forest, bootstrap determines whether bootstrap samples are used when building trees. criterion, max_depth, min_samples_split, min_samples_leaf, max_features are the same as those of the decision tree model.

```
from sklearn.ensemble import RandomForestRegressor
parameter_space = \
{
"n_estimators": [10, 100, 300, 600],
"criterion": ["mse", "mae"],
"max_depth": [7, 50, 254],
"min_samples_split": [2, 5],
"min_samples_leaf": [1, 5],
"max_features": [19, 100, X_train.shape[1]],
"bootstrap": [True, False],
}
clf = RandomizedSearchCV(RandomForestRegressor(random_state=3),
parameter_space, cv=3, n_jobs=4,
scoring="neg_mean_absolute_error",
n_iter=10, random_state=3)
clf.fit(X_train, y_train)
```

print("Best parameters:")

We defined the parameter space above using reasonable values for chosen parameters. Then

we used GridSearchCV () with 3 folds (cv=3). Now we build our Random Forest model with the best parameters found:

```
rf_model = RandomForestRegressor(**clf.best_params_)
```

Then we train our model using our training set (X_train and y_train):

```
rf_model.fit(X_train, y_train);
```

Finally, we test our model on X_test. Then we evaluate the model performance by comparing

its predictions with the actual true values in y_test using the MAE metric as we described above:

```
y_pred = rf_model.predict(X_test)
rf_mae = mean_absolute_error(y_test, y_pred)
print("Random Forest MAE =", rf_mae)
print("Random Forest MAPE=",(rf_mae/mn)*100,"%")
```

Random Forest MAE = 828489.23
Random Forest MAPE= 9.85 %

5.7 Result

In the previous section, we created many models: for each model, we searched for best parameters then we constructed the model using those parameters, then trained (fitted) the model to our training data (X_train and y_train), then tested the model on our test data (X_test) and finally, we evaluated the model performance by comparing the model predictions with the true values in y_test. We used the mean absolute error (MAE) or MAPE to evaluate model performance. Using the results, we got in the previous section, we present a table that shows the mean absolute error (MAE) for each model when applied to the test set X_test. The table is sorted ascending order according to MAE and MAPE score.

Model	**MAE**	**MAPE**
Decision Tree	825248.92	9.81 %
Random Forest	828489.23	9.85 %
K-Nearest Neighbors(KNN)	864142.58	10.27 %
Neural Network(NN)	1488718.18	17.69 %
Elastic net	1470060.97	17.47 %
Ridge	1472549.94	17.50 %

Table 5. 1 the mean absolute error (MAE) for each Regression model

We also present a graph that visualizes the table contents:

```
x = ['Ridge', ' Elastic Net ','KNN', 'Decision Tree', 'NN','Random forest']
y = [1472549.94,1470060.97,864142.58,864142.58,1488718.18,828489.23]
colors = ["#392834","#5a3244", "#a1484f","#c05949", "#e88b2b"]
fig, ax = plt.subplots()
plt.barh(y=range(len(x)), tick_label=x, width=y, height=0.4, color=colors);
ax.set(xlabel="MAE (smaller is better)", ylabel="Model");
```

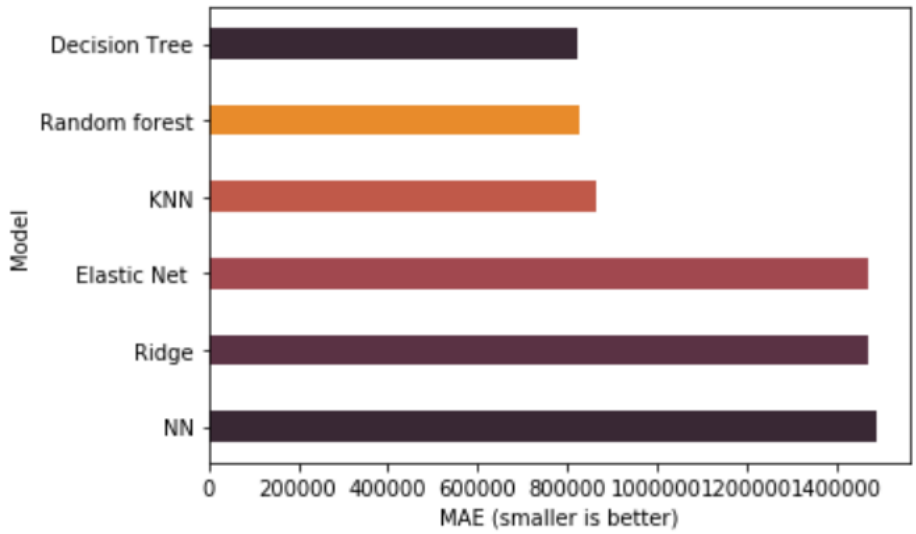

Figure 5. 1 graph that visualizes mean absolute error (MAE) for each model

5.8 Discussion

By looking at the table and the graph, we can see that decision tree model has the least MAE, 825248.92 and MAPE 9.81 %, followed by Random forest model with a little larger error of MAE 828489.23 and MAPE 9.85 %. After that, KNN MAE of 864142.58 and MAPE of 10.27 %, Elastic net MAE of 1470060.97 and MAPE 17.47 % and Ridge MAE models come with MAE of 1472549.94 and MAPE of 17.50 %. So, in our experiment, the most effective model is Decision model for this study and the worst model is Ridge regression. We can see that the difference in MAE between the best model and the worst model is significant; the best model has almost less than half of the error of the worst model. We chose the mean absolute error (MAE) as our performance metric to evaluate and compare models. MAE presents a value that's easy to understand; it shows the average price of model error. As an example, for our decision tree model, its MAE is 825248.92 which implies that on average, the decision tree will predict a value that is bigger or smaller than the true value by 825248.92. Now to understand how good this MAE is, we want to determine the range and distribution of the data. In our case, we need to see the values of the target variable price which contains the actual house prices.

Chapter Six: Conclusions and Recommendations

6.1 Conclusions

In this house features, we evaluated and compared each model to see the one with the highest performance. We also checked out how some models rank the features in step with their importance. During this study, we followed the data science process starting with getting the information, then cleaning and preprocessing the information, followed by exploring the information and building models, then evaluating the results and communicating them with visualizations and comparing the result with MAE or MAPE.

To choose out the simplest regression model, we compare results of MAE and MAPE to choose regression model that has least MAE or MAPE result. As we see in chapter five at implementation section decision tree model has the least MAE result. So the decision tree model is high performance to predict the price of a house for Addis Ababa city.

6.2 Recommendations

We advise using this model with more and modern data by those who want to buy a house within the area covered by the dataset to possess a concept about the particular price to get the best result. The model will be used also with datasets that cover different cities and areas as long as they contain the identical features. We also suggest that individuals take into consideration the features that were believed as most vital as seen within the previous section this might help them estimate the house price better. To increase the performance of this model, increase the sizes of the dataset by adding more correct parameters attribute and columns from real stakeholders. Since location or zone is one factor that affects the sales price of a house for future researcher they may use GPS coordination to point the correct specific location of house.

References

[1]. Golini, A. (1994). Urbanization and Urban Population. *Ethiopia: CSA& the Institute of Population Research*.

[2]. King, R., Orloff, M., Virsilas, T., & Pande, T. (2017). Confronting the Urban Housing Crisis in the Global South: Adequate, Secure, and Affordable Housing. *World Resources Institute Working study*.

[3]. Habitat, U. N. (2013). *State of the world's cities 2012/2013: Prosperity of cities*. Routledge.

[4]. Yu, H., & Wu, J. (2016). Real estate price prediction with regression and classification. *CS229 (Machine Learning) Final Project Reports*.

[5]. Li, D. Y., Xu, W., Zhao, H., & Chen, R. Q. (2009, July). A SVR based forecasting approach for real estate price prediction. In *2009 International Conference on Machine Learning and Cybernetics* (Vol. 2, pp. 970-974). IEEE.

[6]. Li, L., & Chu, K. H. (2017, May). Prediction of real estate price variation based on economic parameters. In *2017 International Conference on Applied System Innovation (ICASI)* (pp. 87-90). IEEE.

[7]. Li, L., & Chu, K. H. (2018). Influence Analysis of Macroeconomic Parameters on Real Estate Price Variation in Taipei, Taiwan. *International Journal of Economics and Management Engineering*, *12*(11), 1564-1570.

[8] Khamis, A. B., Hameed, R., Nor, M. E., Him, N. C., Salleh, R. M., & Razali, S. N. A. M. (2018).

[9]. AR, P. D., & Zdenek, S. (2014). Comparative study of machine learning techniques for supervised classification of biomedical data. *Act Electro technician et Informatica*, *14*(3), 5-10.

[10]. Hegazy, O., Soliman, O. S., & Salam, M. A. (2014). LSSVM-ABC algorithm for stock price prediction. *arXiv preprint arXiv:1402.6366*.

[11]. Limsombunchai, V. (2004, June). House price prediction: hedonic price model vs. artificial neural network. In *New Zealand agricultural and resource economics society conference* (pp. 25-26).

[12]. Feng, Y., & Jones, K. (2015, July). Comparing multilevel modelling and artificial neural networks in house price prediction. In 2015 2nd IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services (ICSDM) (pp. 108-114). IEEE.

[13]. Sharp, M., Ak, R., & Hedberg Jr, T. (2018). A survey of the advancing use and development of machine learning in smart manufacturing. *Journal of manufacturing systems*, *48*, 170-179.

[14]. Mitchell, T. M. (2006). *The discipline of machine learning* (Vol. 9). Pittsburgh: Carnegie Mellon University, School of Computer Science, Machine Learning Department. Antonelo, E. A., & Schrauwen, B. (2010, May). In *2010 IEEE International Conference on Robotics and Automation* (pp. 2959-2964). IEEE.

[15]. Khan, A., Baharudin, B., Lee, L. H., & Khan, K. (2010). A review of machine learning algorithms for text-documents classification. *Journal of advances in information technology*, *1*(1), 4-20.

[16]. Kennedy, K. (2013). Credit scoring using machine learning.

[17]. Phan, T. D. (2018, December). Housing price prediction using machine learning algorithms: The case of Melbourne city, Australia. In *2018 International Conference on Machine Learning and Data Engineering (iCMLDE)* (pp. 35-42). IEEE.

[18]. Escalante-B, A. N., & Wiskott, L. (2013). How to solve classification and regression problems on high-dimensional data with a supervised extension of slow feature analysis. *The Journal of Machine Learning Research*, *14*(1), 3683-3719.

[19]. Lee, A., Taylor, P., Kalpathy-Cramer, J., & Tufail, A. (2017). Machine learning has arrived! *Ophthalmology*, *124*(12), 1726-1728.

[20]. Komagome-Citye, A. (2017). *Models and Visualizations for Housing Price Prediction* (Doctoral dissertation, California State Polytechnic University, Pomona).

[21]. Nguyen, An. "Housing price prediction." (2018).

[22]. Li, D. Y., Xu, W., Zhao, H., & Chen, R. Q. (2009, July). A SVR based forecasting approach for real estate price prediction. In *2009 International Conference on Machine Learning and Cybernetics* (Vol. 2, pp. 970-974). IEEE.

[23]. Wu, J. Y. (2017). Housing Price Prediction Using Support Vector Regression.

[24]. Sawant, R., Jangid, Y., Tiwari, T., Jain, S., & Gupta, A. (2018, August). Comprehensive Analysis of Housing Price Prediction in Pune Using Multi-Featured Random Forest Approach. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)* (pp. 1-5). IEEE.

[25]. Ravikumar, A. S. (2017). *Real Estate Price Prediction Using Machine Learning* (Doctoral dissertation, Dublin, National College of Ireland).

[26]. Chen, Y. C. (2018). Modal regression using kernel density estimation: A review. *Wiley Interdisciplinary Reviews: Computational Statistics*, *10*(4), e1431.

[27]. Vineeth, N., Ayyappa, M., & Bharathi, B. (2018, April). House Price Prediction Using Machine Learning Algorithms. In *International Conference on Soft Computing Systems* (pp. 425-433). Springer, Singapore.

[29]. Wu, J. Y. (2017). Housing Price Prediction Using Support Vector Regression. J. Mu, F. Wu, and A. Zhang, "Housing Value Forecasting Based on Machine Learning Methods," *Abstr. Appl. Anal.*, vol. 2014, 2014.

[30]. Aaron, N. (2015). Machine Learning for a London Housing Price Prediction Mobile Application.

[31]. S. Ravikumar, "Real Estate Price Prediction Using Machine Learning," *Sch. Comput. Natl. Coll. Irel.*, pp. 2–19, 2018.

[32]. Yu, H., & Wu, J. (2016). Real estate price prediction with regression and classification. *CS229 (Machine Learning) Final Project Reports*.

[33]. Henrique, B. M., Sobreiro, V. A., & Kimura, H. (2018). Stock price prediction using support vector regression on daily and up to the minute prices. *The Journal of finance and data science*, *4*(3), 183-201.

[34]. Limsombunchai, V. (2004, June). House price prediction: hedonic price model vs. artificial neural network. In *New Zealand agricultural and resource economics society conference* (pp. 25-26).

[35]. Afonso, B., Melo, L., Oliveira, W., Sousa, S., & Berton, L. (2019, October). Housing Prices Prediction with a Deep Learning and Random Forest Ensemble. In *Anais do XVI Encontro Nacional de Inteligência Artificial e Computacional* (pp. 389-400). SBC.

[36]. Yohannes, S., & Dinku, A. (2018). Housing provisions and affordability In private residential real estates in Addis Ababa. *Zede Journal*, *36*, 13-27.

[37]. De Stefani, A. (2018). House price history, biased expectations and credit cycles. In *Danmarks Nationalbank Working Paper*.

[38]. Pandiri, V. S. (2017). Machine Learning Models to Predict House Prices based on Home Features.

[39]. Li, W., Zhao, Y., Meng, W., & Xu, S. (2009, January). Study on the risk prediction of real estate investment whole process based on support vector machine. In *2009 Second International Workshop on Knowledge Discovery and Data Mining* (pp. 167-170). IEEE.

[40]. Abbasi, S. Advanced Regression Techniques Based Housing Price Prediction Model. In *13th Int. Conf. Iran. Oper. Res. Soc* (pp. 1-10).

[41]. Mali, P., Karchalkar, H., Jain, A., Singh, A., & Kumar, V. (2017). Open Price Prediction of Stock Market Using Regression Analysis. *International Journal of Advanced Research in Computer and Communication Engineering*, *6*(5).

[42] Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.

Appendixes

Appendix I: – Source Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:

```
DS = pd.read_csv('DatasetAA.csv')
```

In [3]:

```
DS.head()
```

Out[3]:

	City	SpecLoc	TotArea	NuBedRoom	PrMoFrSal	PrYrFrsal	Price
0	Addis Ababa	abado	150.0	NaN	7	2020	2500000
1	Addis Ababa	abado	150.0	NaN	5	2020	4356000
2	Addis Ababa	abuare	450.0	NaN	9	2020	17000000
3	Addis Ababa	abuare	450.0	NaN	8	2020	20000000
4	Addis Ababa	addissefer	600.0	8.0	10	2020	30000000

In [4]:

```
DS.describe(include=[np.number], percentiles=[.5])\
.transpose().drop("count", axis=1)
```

Out[4]:

	mean	std	min	50%	max
TotArea	2.347272e+02	1.527908e+02	43.0	200.0	1500.0
NuBedRoom	4.728425e+00	2.086851e+00	1.0	4.0	18.0
PrMoFrSal	8.082577e+00	3.252155e+00	1.0	9.0	12.0

	mean	std	min	50%	max
PrYrFrsal	2.019933e+03	3.208025e-01	2017.0	2020.0	2020.0
Price	8.412888e+06	7.258053e+06	780000.0	6000000.0	36000000.0

In [5]:

```
DS.describe(include=[np.object]).transpose() \
.drop("count", axis=1)
```

Out[5]:

	unique	top	freq
City	1	Addis Ababa	5401
SpecLoc	95	summit	451

In [6]:

```
missing = DS.isna().sum()
missing = missing[missing > 0]
perce_missing = missing * 100 / DS.shape[0]
pd.concat([missing, perce_missing], axis=1,
keys=['Missing Values', 'Percentage']).\
sort_values(by="Missing Values", ascending=False)
```

Out[6]:

	Missing Values	Percentage
NuBedRoom	3744	69.320496
SpecLoc	2202	40.770228

In [7]:

```
DS['NuBedRoom'].fillna(0, inplace=True)
DS['SpecLoc'].fillna("Not Indicated", inplace=True)
```

In [8]:

```
DS.isna().values.sum()
```

Out[8]:

```
0
```

In [9]:

```
sns.violinplot(x=DS['Price'], inner="quartile", color="#96B37E");
```

In [10]:

```
sns.boxplot(DS['Price'], whis=10, color="#D008A9");
```

In [11]:

```
sns.distplot(DS['Price'], kde=False,
color="#172B4D", hist_kws={"alpha": 0.8});
plt.ylabel("Count");
```

In [12]:

```
sns.heatmap(DS.corr(),cmap="YlGnBu");
```

In [13]:

```
sns.distplot(DS['TotArea'], kde=False,
color="#000B9B", hist_kws={"alpha": 0.8});
plt.ylabel("Count");
```

In [14]:

```
plt.scatter(x=DS['TotArea'], y=DS['Price'],
color="orange", edgecolors="#000000", linewidths=0.5);
plt.xlabel("TotArea"); plt.ylabel("Price");
```

In [15]:

```
sns.distplot(DS['TotArea'], kde=False,
color="#000B9B", hist_kws={"alpha": 0.8});
plt.ylabel("Count")
```

Out[15]:

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

In [16]:

```
fig, axes = plt.subplots(1, 2, figsize=(18,5))
fig.subplots_adjust(hspace=0.5, wspace=0.6)
for ax, v in zip(axes.flat, ["PrMoFrSal", "PrYrFrsal"]):sns.distplot(
DS[v], kde=False, color="#DA0B4B",hist_kws={"alpha": 0.8}, ax=ax);ax.
set(ylabel="Count");
```

In [17]:

```
x_vars = ["PrMoFrSal", "PrYrFrsal"]
g = sns.PairGrid(DS, y_vars=["Price"], x_vars=x_vars);
g.map(plt.scatter, color="green", edgecolors="#000000", linewidths=0.
5);
```

In [18]:

```
DS = pd.get_dummies(DS)
```

In [19]:

```
DS.fillna(DS.mean(), inplace=True)
```

In [20]:

```
DS['Price'].head()
```

Out[20]:

```
0     2500000
1     4356000
2    17000000
3    20000000
4    30000000
Name: Price, dtype: int64
```

In [21]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(DS.drop('Price',
axis=1), DS[['Price']],test_size=0.25, random_state=3)
```

In [22]:

```
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RandomizedSearchCV
from sklearn.tree import DecisionTreeRegressor
parameter_space = {
"max_depth": [4, 5],
"min_samples_split": [3, 4],
"max_features": [4, 5]
}

clf = GridSearchCV(DecisionTreeRegressor(), parameter_space, cv=4,sco
ring="neg_mean_absolute_error")
clf.fit(X_train,y_train)
clf = RandomizedSearchCV(DecisionTreeRegressor(), parameter_space, cv
=4,scoring="neg_mean_absolute_error", n_iter=100)
```

In [23]:

```
from sklearn.linear_model import Ridge
parameter_space = {
"alpha": [1, 10, 100, 290, 500],
"fit_intercept": [True, False],
"solver": ['svd', 'cholesky', 'lsqr', 'sparse_cg', 'sag', 'saga'],
}
clf = GridSearchCV(Ridge(random_state=3), parameter_space, n_jobs=4,
cv=3, scoring="neg_mean_absolute_error")
clf.fit(X_train, y_train)
print("Best parameters:")
print(clf.best_params_)
```

```
Best parameters:
{'alpha': 1, 'fit_intercept': False, 'solver': 'lsqr'}
```

In [24]:

```
ridge_model = Ridge(random_state=3, **clf.best_params_)
ridge_model.fit(X_train, y_train);
```

In [25]:

```
from sklearn.metrics import mean_absolute_error
y_pred = ridge_model.predict(X_test)
```

```
ridge_mae = mean_absolute_error(y_test, y_pred)
mn=DS["Price"].mean()
print("Ridge MAE =", ridge_mae)
print("Ridge MAPE=",(ridge_mae/mn)*100,"%")
```

```
Ridge MAE = 1472549.9411575748
Ridge MAPE= 17.503500292955867 %
```

In [26]:

```
from sklearn.linear_model import ElasticNet
parameter_space = {
"alpha": [1, 10, 100, 290, 500],
"l1_ratio": [0.5, 1],
"fit_intercept": [True, False],
}
clf = GridSearchCV(ElasticNet(random_state=3), parameter_space,
n_jobs=4, cv=3, scoring="neg_mean_absolute_error")
clf.fit(X_train, y_train)
print("Best parameters:")
print(clf.best_params_)
```

```
Best parameters:
{'alpha': 100, 'fit_intercept': True, 'l1_ratio': 0.5}
```

In [27]:

```
elasticNet_model = ElasticNet(random_state=3, **clf.best_params_)
```

In [28]:

```
elasticNet_model.fit(X_train, y_train);
```

In [29]:

```
y_pred = elasticNet_model.predict(X_test)
elasticNet_mae = mean_absolute_error(y_test, y_pred)
print("Elastic Net MAE =", elasticNet_mae)
print("Elastic Net MAPE=",(elasticNet_mae/mn)*100,"%")
```

```
Elastic Net MAE = 1470060.966699363
Elastic Net MAPE= 17.47391503819417 %
```

In [30]:

```
from sklearn.neighbors import KNeighborsRegressor
parameter_space = {
"n_neighbors": [9, 10, 11,50],
"weights": ["uniform", "distance"],
"algorithm": ["ball_tree", "kd_tree", "brute"],
"leaf_size": [1,2,20,50,200]
}
clf = GridSearchCV(KNeighborsRegressor(), parameter_space, cv=3,
scoring="neg_mean_absolute_error", n_jobs=4)
clf.fit(X_train, y_train)
print("Best parameters:")
print(clf.best_params_)
```

```
Best parameters:
```

```
{'algorithm': 'ball_tree', 'leaf_size': 2, 'n_neighbors': 9, 'weights
': 'distance'}
```

In [31]:

```
knn_model = KNeighborsRegressor(**clf.best_params_)
knn_model.fit(X_train, y_train);
```

In [32]:

```
y_pred = knn_model.predict(X_test)
knn_mae = mean_absolute_error(y_test, y_pred)
print("K-Nearest Neighbors MAE =", knn_mae)
print("K-Nearest Neighbors MAPE=",(knn_mae/mn)*100,"%")
```

```
K-Nearest Neighbors MAE = 864142.5803274602
K-Nearest Neighbors MAPE= 10.271651565193867 %
```

In [33]:

```
from sklearn.tree import DecisionTreeRegressor
parameter_space = \
{
"criterion": ["mse", "friedman_mse", "mae"],
"min_samples_split": [5, 18, 29, 50],
"min_samples_leaf": [3, 7, 15, 25],
"max_features": [10, 25, 75, 100, X_train.shape[1]],
}
clf = GridSearchCV(DecisionTreeRegressor(random_state=3), parameter_s
pace,
cv=3, scoring="neg_mean_absolute_error", n_jobs=4)
clf.fit(X_train, y_train)
print("Best parameters:")
print(clf.best_params_)
```

```
Best parameters:
{'criterion': 'mae', 'max_features': 100, 'min_samples_leaf': 7, 'min
_samples_split': 5}
```

In [34]:

```
dt_model = DecisionTreeRegressor(**clf.best_params_)
dt_model.fit(X_train, y_train);
```

In [35]:

```
y_pred = dt_model.predict(X_test)
dt_mae = mean_absolute_error(y_test, y_pred)
print("Decision Tree MAE =", dt_mae)
print("Decision Tree MAPE=",(dt_mae/mn)*100,"%")
```

```
Decision Tree MAE = 803513.1665433013
Decision Tree MAPE= 9.550978580005646 %
```

In [36]:

```
from sklearn.neural_network import MLPRegressor
parameter_space = \
{
"hidden_layer_sizes": [(7,)*3, (19,), (100,), (154,)],
```

```
"activation": ["identity", "logistic", "tanh", "relu"],
"solver": ["lbfgs"],
"alpha": [1, 10, 100],
}
clf = GridSearchCV(MLPRegressor(random_state=3), parameter_space,
cv=3, scoring="neg_mean_absolute_error", n_jobs=4)
clf.fit(X_train, y_train)
print("Best parameters:")
print(clf.best_params_)
```

```
C:\Anaconda\lib\site-packages\sklearn\neural_network\_multilayer_perc
eptron.py:1342: DataConversionWarning: A column-vector y was passed w
hen a 1d array was expected. Please change the shape of y to (n_sampl
es, ), for example using ravel().
  y = column_or_1d(y, warn=True)
Best parameters:
{'activation': 'relu', 'alpha': 100, 'hidden_layer_sizes': (19,), 'so
lver': 'lbfgs'}
C:\Anaconda\lib\site-packages\sklearn\neural_network\_multilayer_perc
eptron.py:470: ConvergenceWarning: lbfgs failed to converge (status=1
):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as sho
wn in:
    https://scikit-learn.org/stable/modules/preprocessing.html
  self.n_iter_ = _check_optimize_result("lbfgs", opt_res, self.max_it
er)
```

In [37]:

```
nn_model = MLPRegressor(**clf.best_params_)
```

In [38]:

```
nn_model.fit(X_train, y_train);
```

```
C:\Anaconda\lib\site-packages\sklearn\neural_network\_multilayer_perc
eptron.py:1342: DataConversionWarning: A column-vector y was passed w
hen a 1d array was expected. Please change the shape of y to (n_sampl
es, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Anaconda\lib\site-packages\sklearn\neural_network\_multilayer_perc
eptron.py:470: ConvergenceWarning: lbfgs failed to converge (status=1
):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as sho
wn in:
    https://scikit-learn.org/stable/modules/preprocessing.html
  self.n_iter_ = _check_optimize_result("lbfgs", opt_res, self.max_it
er)
```

In [39]:

```
y_pred = nn_model.predict(X_test)
nn_mae = mean_absolute_error(y_test, y_pred)
print("Neural Network MAE =", nn_mae)
print("Neural Network MAPE=",(nn_mae/mn)*100,"%")
```

```
Neural Network MAE = 1498750.413108708
Neural Network MAPE= 17.81493283296992 %
```

In [40]:

```
from sklearn.ensemble import RandomForestRegressor
parameter_space = \
{
"n_estimators": [10, 100, 300, 600],
"criterion": ["mse", "mae"],
"max_depth": [7, 50, 254],
"min_samples_split": [2, 5],
"min_samples_leaf": [1, 5],
"max_features": [19, 100, X_train.shape[1]],
"bootstrap": [True, False],
}
clf = RandomizedSearchCV(RandomForestRegressor(random_state=3),
parameter_space, cv=3, n_jobs=4,
scoring="neg_mean_absolute_error",
n_iter=10, random_state=3)
clf.fit(X_train, y_train)
print("Best parameters:")
print(clf.best_params_)
```

```
C:\Anaconda\lib\site-packages\sklearn\model_selection\_search.py:739:
DataConversionWarning: A column-vector y was passed when a 1d array w
as expected. Please change the shape of y to (n_samples,), for exampl
e using ravel().
  self.best_estimator_.fit(X, y, **fit_params)
Best parameters:
{'n_estimators': 10, 'min_samples_split': 2, 'min_samples_leaf': 5, '
max_features': 101, 'max_depth': 254, 'criterion': 'mae', 'bootstrap'
: True}
```

In [41]:

```
rf_model = RandomForestRegressor(**clf.best_params_)
```

In [42]:

```
rf_model.fit(X_train, y_train);
```

```
C:\Anaconda\lib\site-packages\ipykernel_launcher.py:1: DataConversion
Warning: A column-vector y was passed when a 1d array was expected. P
lease change the shape of y to (n_samples,), for example using ravel(
).
  """Entry point for launching an IPython kernel.
```

In [43]:

```
y_pred = rf_model.predict(X_test)
```

```
rf_mae = mean_absolute_error(y_test, y_pred)
print("Random Forest MAE =", rf_mae)
print("Random Forest MAPE=",(rf_mae/mn)*100,"%")
```

```
Random Forest MAE = 831773.4633604737
Random Forest MAPE= 9.88689527783224 %
```

In [44]:

```
x = ['Ridge', ' Elastic Net ','KNN', 'Decision Tree', 'NN','Random fo
rest']
y = [1472549.94,1470060.97,864142.58,864142.58,1488718.18,828489.23]
colors = ["#392834","#5a3244", "#a1484f","#c05949", "#e88b2b"]
fig, ax = plt.subplots()
plt.barh(y=range(len(x)), tick_label=x, width=y, height=0.4, color=co
lors);
ax.set(xlabel="MAE (smaller is better)", ylabel="Model");
```

Appendix II: – Sample dataset list

City	SpecLoc	TotArea	NuBedRoom	PrMoFrSal	PrYrFrsal	Price
Addis Ababa	summit	47		3	2020	780000
Addis Ababa	lebu	94	6	9	2020	800000
Addis Ababa	haile garment	52		4	2020	800000
Addis Ababa		58	5	12	2020	820000
Addis Ababa	lideta	60	3	8	2020	830000
Addis Ababa	summit	94	5	9	2020	840000
Addis Ababa		60	3	11	2020	840000
Addis Ababa		94	5	5	2020	850000
Addis Ababa		60	5	10	2020	850000
Addis Ababa		61	5	11	2020	850000
Addis Ababa		62	3	10	2020	850000
Addis Ababa		64	6	9	2020	850000
Addis Ababa		66	3	10	2020	850000
Addis Ababa		66	5	10	2020	850000
Addis Ababa		68	3	4	2020	850000
Addis Ababa		48		8	2020	900000
Addis Ababa		47		9	2020	900000
Addis Ababa		94	5	10	2020	900000
Addis Ababa		68		9	2020	900000
Addis Ababa	hayat	68		8	2020	900000
Addis Ababa	summit	69		5	2020	900000
Addis Ababa	summit	69		9	2020	900000

Addis Ababa	summit	69		1	2020	900000
Addis Ababa		69	5	10	2020	900000
Addis Ababa		70	6	10	2020	900000
Addis Ababa		70	6	11	2020	900000
Addis Ababa		71	7	5	2020	900000
Addis Ababa		71	6	9	2020	900000
Addis Ababa		72	3	10	2020	900000
Addis Ababa	hailegarment	72		5	2020	900000
Addis Ababa	tuludimtu	72		6	2018	930000
Addis Ababa	jemo	94	5	11	2020	950000
Addis Ababa	summit	72	5	11	2020	950000
Addis Ababa	summit	72	7	10	2020	950000
Addis Ababa	summit	72		9	2020	950000
Addis Ababa		72		5	2020	950000
Addis Ababa	hailegarment	72		12	2019	950000
Addis Ababa		72	3	10	2020	950000
Addis Ababa		72	4	10	2020	950000
Addis Ababa		47		9	2020	980000
Addis Ababa	hayat	72		9	2020	980000
Addis Ababa	hayat	72	5	11	2020	980000
Addis Ababa		50	1	9	2020	990000
Addis Ababa		72	1	9	2020	990000
Addis Ababa	summit	72		9	2020	990000
Addis Ababa		59		4	2020	1000000
Addis Ababa		140		5	2020	1000000
Addis Ababa		200		4	2020	1000000
Addis Ababa		86		12	2019	1000000
Addis Ababa	summit	175	6	9	2020	1000000
Addis Ababa		210	3	10	2020	1000000
Addis Ababa		212	5	1	2020	1000000
Addis Ababa		220	3	9	2020	1000000
Addis Ababa		225	3	9	2020	1000000
Addis Ababa	hayat	250	5	9	2020	1000000
Addis Ababa		280	2	10	2020	1000000
Addis Ababa		72		12	2019	1000000
Addis Ababa		72		6	2018	1000000
Addis Ababa		72		4	2020	1000000

Addis Ababa		72		5	2020	1000000
Addis Ababa	hayat	72	6	9	2020	1000000
Addis Ababa		72	6	9	2020	1000000
Addis Ababa	akaki	72	6	10	2020	1000000
Addis Ababa	akaki	72	3	10	2020	1000000
Addis Ababa	akaki	72	6	11	2020	1000000
Addis Ababa	summit	72	3	10	2020	1000000
Addis Ababa		72	3	10	2020	1000000
Addis Ababa	hailegarment	72	3	11	2020	1000000
Addis Ababa	summit	72	5	5	2020	1000000
Addis Ababa	tuludimtu	72	3	9	2020	1000000
Addis Ababa		72	3	9	2020	1000000
Addis Ababa	hayat	72	3	9	2020	1000000
Addis Ababa	summit	72	5	9	2020	1000000
Addis Ababa	hayat	72	5	9	2020	1000000
Addis Ababa		72	3	11	2020	1000000
Addis Ababa		72	5	11	2020	1000000
Addis Ababa	summit	72		1	2020	1000000
Addis Ababa	summit	72	2	5	2020	1000000
Addis Ababa	summit	72	2	10	2020	1000000
Addis Ababa	summit	72	2	11	2020	1000000
Addis Ababa	summit	72	4	5	2020	1000000
Addis Ababa	summit	72	4	9	2020	1000000
Addis Ababa	summit	72	3	9	2020	1000000
Addis Ababa		72	4	11	2020	1000000
Addis Ababa		72	4	5	2020	1000000
Addis Ababa	summit	72	4	10	2020	1000000