

PROFESSIONAL TRAINING REPORT

At

Sathyabama Institute of Science and Technology (Deemed To be University)

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in Computer Science and Engineering

BY

NAME: ALLAM RANJITH KUMAR REDDY

REGNO: 38110451

NAME: ALLAM RAVI DINESH REDDY

REGNO: 38110025



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

JEPPIAR NAGAR, RAJIV GANDHI SALAI,

CHENNAI-600119, TAMILNADU



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC
(Established Under Section 3 Of UGC Act 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI-600119

www.sathyabamauniversity.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the Bonafide work of **ALLAM RANJITH REDDY(Reg.no. 38110451)** and **ALLAM RAVI DINESH REDDY(Reg.no.38110025)** and who carried out the project entitled **"House price prediction using machine learning"** under our supervision from January 2022 to May 2022.

Internal Guide

Dr.R.JEBERSON RETNA RAJ M.E., Ph.D.

Head of the Department

Dr.S. VIGNESHWARI M.E., Ph.D.,

Submitted for Viva voce Examination held on

Internal Examiner

External Examiner

DECLARATION

I ALLAM RANJITH KUMAR REDDY (Reg.No.38110451) & ALLAM RAVI DINESH REDDY(Reg.No.38110025)hereby declare that the Project Report Entitled "House prediction using machine learning" done by me under the guidance of **Dr.R.JEBERSON RETNA RAJ M.E, PH.D**, is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering / Technology degree in Computer Science and Engineering.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey thanks to **Dr. Sasikala M.E., Ph.D., Dean**, school of computing and **Dr. S. Vigneshwari, M.E., Ph.D., Head of the Department**, Department of Computer Science Engineering for providing us the necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. R.JEBERSON RETNA RAJ M.E., Ph.D.**, for his valuable guidance, suggestions, and constant encouragement paved way for the successful completion of my project.

I wish to express our thanks to all Teaching and Non-teaching staff members of the Department of **COMPUTER SCIENCE AND ENGINEERING** who were helpful in many ways for the completion of the project.

Abstract:

House Price Index (HPI) is commonly used to estimate the changes in housing price. Since housing price is strongly correlated to other factors such as location, area, population, it requires other information apart from HPI to predict individual housing price.

There has been a considerably large number of papers adopting traditional machine learning approaches to predict housing prices accurately, but they rarely concern about the performance of individual models and neglect the less popular yet complex models.

As a result, to explore various impacts of features on prediction methods, this paper will apply both traditional and advanced machine learning approaches to investigate the difference among several advanced models. This paper will also comprehensively validate multiple techniques in model implementation on regression and provide an optimistic result for housing price prediction.

Chapter-1

Introduction:

House is one of human life's most essential needs, along with other fundamental needs such as food, water, and much more. Demand for houses grew rapidly over the years as people's living standards improved. While there are people who make their house as an investment and property, yet most people around the world are buying a house as their shelter or as their livelihood. According to housing markets have a positive impact on a country's currency, which is an important national economy scale. Homeowners will purchase goods such as furniture and household equipment for their home, and homebuilders or contractors will purchase raw material to build houses to satisfy house demand, which is an indication of the economic wave effect created by the new house supply. Besides that, consumers have capital to make a large investment, and the construction industry is in good condition can be seen through a country's high level of house supply.

In a very general way, recommender systems are algorithms aimed at suggesting relevant items to users (items being movies to watch, text to read, products to buy or anything else depending on industries). This website also provides an option for recommendations. The type of recommendation system is content based recommendation. In this project, we are using two datasets using the concept of web scraping. One dataset consists of some features such as location, BHK, floor, furnished etc. with different cities in Mumbai. This dataset is used for prediction. The other dataset consists of the House Price index of Mumbai for the last 10 years. This dataset is used for forecasting.

An increase in house demand occurs each year, indirectly causing house price increases every year. The problem arises when there are numerous variables such as location and property demand that may influence the house price, thus most stakeholders including buyers and developers, house builders and the real estate industry would like to know the exact attributes or the accurate factors influencing the house price to help investors make decisions and help house builders set the house price. House price prediction can be done by using a multiple prediction models (Machine Learning Model) such as support vector regression, artificial neural network, and more. There are many benefits that home buyers, property investors, and house builders can reap from the house-price model. This model will provide a lot of information and knowledge to home buyers, property investors and house builders, such as the valuation of house prices in the present market, which will help them determine house prices. Meanwhile, this model can help potential buyers decide the characteristics of a house they want according to their budget. Previous studies focused on analyzing the attributes that affect house price and predicting house price based on the model of machine learning separately. However, this article combines such a both predicting house price and attributes together.

Chapter 2

Literature review

2.1. HOUSE PRICE PREDICTION FORECASTING AND RECOMMENDATION SYSTEM USING MACHINE LEARNING

Abstract-

The relationship between house prices and the economy is an important motivating factor for predicting house prices. A property's value is important in real estate transactions. Housing price trends are not only the concern of buyers and sellers, but it also indicates the current economic situation. Therefore, it is important to predict housing prices without bias to help both the buyers and sellers make their decisions. In this project,

we are going to create a website where user have to add some property details for predicting the house price, enter date for forecasting the price till that

date and budget range for recommending best location. This project uses two datasets, one includes some features and large entries of housing sales in Mumbai and another contains house price index of Mumbai. We are using different feature selection methods and feature extraction method with Multiple Linear Regression to predict the current house price and using ARIMA model for forecasting the price after few years in Mumbai and also uses content based recommendation system to recommend best location according to their budget in nearby area of interest.

INTRODUCTION

Investment is a business activity on which most people are interested in this globalization era. There are several objects that are often used for investment, for

example, gold, stocks and property. In particular, property investment has increased significantly. Housing price trends are not only the concern of buyers and sellers, but it also indicates the current economic situation.

There are many factors which has

impact on house prices, such as location, BHK, floor etc. Also, a location with a great accessibility to highways, expressways, schools, shopping malls and local employment opportunities contributes to the rise in house price. Manual house prediction becomes difficult, hence there are many systems developed for

house price prediction. The aim of this system is to create a website through which the user can give his house requirements as input which is then passed on to the linear regression model for predicting the house price. The website also allows user to forecast the predicted house price to a particular date which is also specified by the user. This is done by using another model known as the ARIMA(Auto Regressive Integrated Moving Average Model). During the last few decades, with the rise of Youtube, Amazon, Netflix and many other such web services, recommender systems have taken more and more place in our lives. From e-commerce (suggest to buyers

articles that could interest them) to online advertisement (suggest to users the right contents, matching their preferences), recommender systems are today unavoidable in our daily online journeys. In a very general way, recommender systems are algorithms aimed at suggesting relevant items to users (items being movies to watch, text to read, products to buy or anything else depending on industries). This website also provides an

option for recommendations. The type of recommendation system is content based recommendation. In this project, we are using two datasets which are extracted from Makaan.com by using the concept of web scraping. One dataset consists of some features such as location, BHK, floor, furnished etc. with different cities in Mumbai. This dataset is used for prediction. The other dataset consists of the House Price index of Mumbai for the last 10 years. This dataset is used for forecasting.

REQUIREMENT ANALYSIS

Requirement analysis gives a minimum requirement that a system should have to make the software to work properly. This application can work on any website. Usually the requirement specification will be the same as that of the operating system.

A. Functional Requirements:

FR1: USER INTERFACE:

The user interface will be a website. The user has to enter all the attributes correctly and in the required format.

FR2: PROPER FORECASTING:

The system has to properly predict the price of the house according to the input given by the user.

FR3: RECOMMENDATION SYSTEM:

According to the input given by the user, the recommendation system will recommend the best property.

FR4: DATABASE:

Dataset should contain large number of entities so that it will increase the accuracy of the predicted price and suggest a better property

B. Non Functional Requirements:

QR1: Platform Independent:

The application would be platform independent if all the requirements are installed in the device.

QR2: Performance:

The application should have better accuracy and should provide the information in less time.

QR3: Capacity:

The capacity of the storage should be high so that large amount of data can be stored in order to train the model.

C. Software Requirements:

1. Coding Language: Python3, HTML, Python Flask

2. Coding software : Anaconda, Spyder, Jupyter Notebook, Sublime text
3

Safety Requirements:

For every input given by user, no incorrect format of data can be given as an input to the system which can be of various forms. All the data fields must be filled by the user to get the Output. The date provided for forecasting should be given of the future not that of the past.

DESIGN AND IMPLEMENTATION

User interface:

The user interface for our project is Website. For this software, the users are the businessmen, investors and other people searching for property. They have to enter details about the property they want and then the software will give the accurate predicted value. User can also forecast the predicted value by entering date. In this application, the user have to enter information on website about the users location such as number of floors, area in sq. feet, location, bhk, furnishing, date for forecasting and budget.

DataSets:

- Dataset is Extracted from Makaan.com by using concept of Web Scraping for house price prediction purpose and downloaded another dataset from TradingEconomics for forecasting.
- Dataset used for prediction contains names of all cities in and nearby Mumbai with their BHK, Sq.ft, Furnished or not, Floor No. and Prices.
- It contains 160000 entries which contains 1400 different cities and places in mumbai.
- Dataset used for forecasting contains House price index according to date for year 2010 to 2019.

Data Preparation:

To prepare the dataset for the prediction system, some changes were made:

1. Binary categorical variables (furniture) is represented using one binary digit (i.e. (Furnishing) 0 = Not Furnished, 1 = Furnished).
2. Also by using Label encoder names of places is to be converted into values as linear regression model is to be trained by using values.
3. As the price for properties are often quoted in Lakhs, we have rounded our dependent variable to the nearest thousand, which also helps with the numerical stability of the model.

Methodology:

Linear Regression:

- In this Project, we have used Linear Regression Algorithm for predicting the current house price.
- The Linear Regression Algorithm accepts two variables Independent variable (X) and Dependent variable (Y).
- We have used sklearn Library for importing Linear Regression model.
- The dataset containing different cities with their features and prices is used for training Linear Regression Model.
- The dataset entities will be divided into two parts 80% for training and 20% for testing.
- Linear Regression model will be trained using X_train Independent variable entries and Y_train Dependent variable entries.
- The trained model will be tested upon the 20% test dataset entities. After training and testing the model will be use for prediction purpose.
- The accuracy for trained linear regression model is 86.67%.

ARIMA(Auto Regressive Integrated Moving Average Model):

- ARIMA Model is widely used for Forecasting purpose like stock, temperature forecasting, sales predictions etc.
- In this project, the ARIMA is used to forecast house price for a particular date which is gives by the user.
- ARIMA Model is the combination for three methods for forecasting which are AutoRegressive (AR) Model, Integrated differencing and Moving Average (MA) Model.

1. AutoRegressive(AR) Model:

Y_t depends only on past values Y_{t-1} , Y_{t-2} , so on. $Y_t = F(Y_{t-1}, Y_{t-2}, Y_{t-3}, \dots)$ if no. of past values (p) increases then the accuracy of the model increases.

2. Moving Average (MA) Model :

Y_t depends only on past error terms. $Y_t = F(E_t, E_{t-1}, E_{t-2}, \dots)$ the no. of past error terms taken is mostly 0, 1 or 2. The No. of error terms is denoted by 'q'.

3. Integrated Differencing:

* In ARIMA Model Series is need to be “**Strictly Stationary**” *

It means, mean, variance and covariance must be constant over the time period. If

series is not stationary then it is converted to stationary using differencing parameter

'd' which is generally equal to 1 or 2.

□ The dataset entities will be divided into two parts 80% for training and 20% for testing. □

The ARIMA model is imported from statsmodel library which takes training dataset and order of (p, d, q) as input. After training the Model will be use for forecasting purpose. □

This project contains ARIMA model with 87% of accuracy.

Content Based Recommendation:

Recommendation system is a machine learning system that gives generalized recommendation to its users based on some data or using users preferences. It

produce a list of recommendations in any of the two ways:

- Collaborative filtering: Collaborative filtering approaches build a model from user's past behaviour (i.e. items purchased or searched by the user) as well as similar decisions made by other users.

- Content-based filtering: Content-based filtering approaches uses a series of discrete

characteristics of an item in order to recommend additional items with similar

properties.

- We are going to use content based filtering methods in our project.

- For example in our project data set contains tuples like place ,square feet , number of bkh , flat is furnished or not and floor number at which given flat is situated.

- Suppose client wants a new two bkh fully furnished flat in a area like diva of 300 sqrt feet at a floor maximum up to 5.

- After that recommendation system takes this and makes some assumptions client has

entered floor number is 5 than system will search the flate for floor number 4 to 6 from training dataset that means if product is not available then it will try to give maximum similar type of product.

- Our will match the every preference made by the user with the values present in the training data set and will try to give similar type of product.

Model Procedure:

The trained linear regression model is given the user entered property details as input and model will return predicted value which is pass from flask to website. For

Arima model the input will the user entered date. The ARIMA model will give House Price Index (HPI) as output which is converted to House price by using formula

Current House Price Value * Future HPI = Future House Price Value * Current HPI

Then the Forecasted house price is displayed on web by flask.

Connectivity:

- The website is connected to backend by using framework called python flask.
- The flask provides a local IP address through which the websites is connected.
- When user enter details about property on website, the IP address provided by flask is used to pass data to flask.
- In python flask program, the trained linear regression model is imported by using library joblib and property details fetched from URL is given to the trained model. The output is given as predicted price which is displayed on screen.
- Similarly, User entered date is also fetch from URL for forecasting. This date is given to the imported ARIMA model which gives forecasted HPI(House Price Index) as output. □
- The forecasted HPI is used to calculate forecasted price which is return to the website.
- For Recommendation, the user entered budget range is used to filter out all the properties which satisfy user property requirement this is called as content based filtering.
- The filtered properties are sent to website in HTML table format.

TECHNOLOGY USED

Documentation Tools

- Microsoft Office Word.
- Snipping Tools (For Screenshots).
- StarUML (For UML Diagrams).
- LucidChart.
- Microsoft Excel

Language Used

The “**HOUSE PRICE FORECASTING AND RECOMMENDATION SYSTEM**” will be used to for predicting the house price, forecasting that price and also to get best recommendation according to users requirement. This application can be run using website. We are using Python3 for making machine learning model and Python flask for connectivity and HTML to develop our web page. We are using anaconda which contains a software Spyder and Jupyter Notebook. Spyder contains all updated and latest libraries of python which will be very useful for implementing machine learning model linear regression, ARIMA model and content based Recommendation system. Sublime Text 3 will be used for implementing HTML web page which will be user interface.

RESULT

In this project frontend contains website . On that website first we have to give input likes Location, BHK, Sq.ft , Furnish and floor for prediction.

CONCLUSION

In this project, the website allows the user to give property details according to his/her requirement. The system makes optimal use of the Data mining Algorithm i.e Linear Regression, ARIMA Model along with Content Based Recommendation System. The Linear Regression algorithm is used to predict the house price according to the property requirement given by the customer with accuracy of 86.7%. ARIMA Model is used for Forecasting the predicted house price with an accuracy of 87%. Content based Recommendation system will help the user to get the best and relevant real estates residential properties according to the budget given by the user. The connectivity between website and models is done by using python flask. The main objective of using this prediction, forecasting and recommendation system is to reduce the human physical calculation, time and carry out the whole process at ease.

2.2. House Price Prediction using a Machine Learning Model: A Survey of Literature

Abstract:

Data mining is now commonly applied in the real estate market. Data mining's ability to extract relevant knowledge from raw data makes it very useful to predict house prices, key housing attributes, and many more. Research has stated that the fluctuations in house prices are often a concern for house owners and the real estate market. A survey of literature is carried out to analyze the relevant attributes and the most efficient models to forecast the house prices. The findings of this analysis verified the use of the Artificial Neural Network, Support Vector Regression and XGBoost as the most efficient models compared to others. Moreover, our findings also suggest that locational attributes and structural attributes are prominent factors in predicting house prices. This study will be of tremendous benefit, especially to housing developers and researchers, to ascertain the most significant attributes to determine house prices and to acknowledge the best machine learning model to be used to conduct a study in this field.

Introduction

House is one of human life's most essential needs, along with other fundamental needs such as food, water, and much more. Demand for houses grew rapidly over the years as people's living standards improved. While there are people who make their house as an investment and property, yet most people around the world are buying a house as their shelter or as their livelihood. According to, housing markets have a positive impact on a country's currency, which is an important national economy scale. Homeowners will purchase goods such as furniture and household equipment for their home, and homebuilders or contractors will purchase raw material to build houses to satisfy house demand, which is an indication of the economic wave effect created by the new house supply. Besides that, consumers have capital to make a large investment, and the construction industry is in good condition can be seen through a country's high level of house supply. According to, numerous international organizations and human rights have emphasized house importance. House is profoundly rooted in the economic, financial, and political structure of each country. Nevertheless, reported that the fluctuation of house prices has always been an issue for house owners, buildings and real estate, besides stated that house has become unaffordable as there is substantial price growth in several countries in the housing sector. Residents' quality of life as well as national economy depends on the potential house price increase. Ultimately, this issue will affect investors who are making their house as an investment. An increase in house demand occurs each year, indirectly causing house price increases every year. The problem arises when there are numerous variables such as location and property demand that may influence the house price, thus most stakeholders including buyers and developers, house builders and the real estate industry would like to know the exact attributes or the accurate factors influencing the house price to help investors make decisions and help house builders set the house price. House price prediction can be done by using a multiple prediction models (Machine Learning Model) such as support vector regression, artificial neural network, and more. There are many benefits that home buyers, property investors, and house builders can reap from the house-price model. This model will provide a lot of information and knowledge to home buyers, property investors and house builders, such as the valuation of house prices in the present market, which will help them determine house prices. Meanwhile, this model can help potential buyers decide the characteristics of a house they want according to their budget. Previous studies focused on analyzing the attributes that affect house price and predicting house price based on the model of machine learning separately. However, this article combines

such a both predicting house price and attributes together. In this article, literature review focuses on predicting house price based on the model of machine learning as well as analyzing attributes primarily used in previous study that affect house price. This paper was arranged as follows: the

first section summarizing overall of this study. Second section described the common attributes used in prediction of house price around the world. It was followed by a brief discussion of machine learning model used in previous study to predict house price. For the next section, the comprehensive effects of the current house price prediction model are addressed. Ultimately, section 5 and section 6 respectively provide the description and conclusion of this comprehensive literature analysis.

Attributes

House price prediction can be divided into two categories, first by focusing on house characteristics, and secondly by focusing on the model used in house price prediction. Many researchers have produced a house price prediction model, including. A research undertaken by analyses the existing housing price in Jakarta, Indonesia using the conceptual model and questionnaires. Based on the results, the attributes or factors affecting the house price differ for each house construction in Jakarta, therefore accepting the validity of this analysis as the main purpose of this research is to classify the factor or attributes affecting the house price. Various considerations influence the price of a house. According to, the factors influencing house prices can be classified into three categories: location, structural and neighborhood condition.

A. Locational

Location is considered to be the most significant feature of house price determination in his study also observed the significant of location attributes in deciding house price. The location of the property was classified in a fixed locational attribute. All of these studies point to the close association between locational attributes such as distance from the closest shopping center, or position offering views of hills or shore, and house price variations.

B. Structural

Another significant feature influencing the house price is structural structure or some research has listed it as physical attributes. Structural characteristic is a feature that people may identify, whether number of bedrooms and bathrooms, or floor space, or garage and patio. These structural attributes, often offered by house builders or developers to attract potential buyers, therefore meet the potential buyers' wishes. In in his earlier study, structural attributes would be the key consideration for house hunters in determining what to purchase as such

attributes represent their market value. In their earlier study, stated that all these attributes have a positive relationship to rising house prices.

C. Neighbourhood

Neighborhood qualities can be included in deciding house price.

According to, efficiency of public education, community social status and proximity to shopping malls typically improve the worth of a property.

There is a substantial rise in house prices from the fifth-class suburban community to affluent neighborhood as predicted.

Nonetheless, study found that these qualities tend to be cultural based, as they are not similarly relevant in all cultures.

Machine Learning Model

According to, the paradigm of evaluating the house demand can be classified into two classes which are the traditional method and the advanced valuation method. The traditional valuation scheme, including multiple regression method and stepwise regression process, whilst hedonic pricing tool, artificial neural network (ANN) and spatial analysis framework are advances valuation method. The model selection to be used to predict house price is quite critical as varieties of models are available. One of the most commonly utilized models in this research field is Regression Analysis which is used in many studies, including.

Another common model for house price

predictions is the Support Vector Regression (SVR).

A. Regression Analysis

i. Hedonic Price Model

The housing market is slightly different from normal good consumption.

According to housing market is unique because it displays the characteristics of resilience, flexibility and spatial fixity. Therefore, hedonic approach is preferred to accurately predict market

differential. conceived hedonic model back in 1939, but this research was popularized in the early 1960s with comprehensive uses by Zvi Griliches and Rosen. In the early 1930s, Court used this model to analyse automotive value in pricing and quality characteristics. defined hedonic as "the implicit prices of attributes and are revealed to economic agents from the observed prices of differentiated products and the specific quantities of their characteristics." Following years of progress, Rosen applied the approach to the residential home price study and became commonly included in real estate sector research. Rosen's philosophy or model comprises two separate phases. The regression of a product price on its attributes is

performed in the initial stage to calculate the aggregate price of the component. A measure of a goods price will be determined in the first stage, but the inverse demand function cannot be generated at this stage.

Thus, the second stage of estimation is needed to identify the inverse demand function that can be derived from the first stage implicit price function. In an earlier study, a study compared three commonly used house price measurement methods which are simple average method, hedonic model, and matching approach. The result found that when adopted on the housing market, two methods that are simple average method and matching approach were proven biased. Thus, the hedonic model provides the highest results relative to those two most commonly encountered versions. Hedonic pricing model is a statistical model that believes the worth of the property is the sum of all its attributes based on hedonic market theory.

ii. Multiple Linear Regression

Regression analysis is a model used to determine the relationship between variables. In order to evaluate the correlation of the variables, the correlation coefficient or regression equation can be used. Multiple regression models can determine which characteristics are the most important to explain the dependent variable. Multiple regression analysis also allows certain price predictions by capturing independent and dependent variable data. In, the power of the multiple regression model can be seen when the value of the relationship between dependent and independent variables is measured. use multiple regression modelling to describe improvements to an independent variable with a dependent variable. This model can be achieved using the house price projection as separate and dependent variables like house prices, house size, property sort, number of bedrooms, and many more. Therefore, the house price is set as a target or dependency variable, while other attributes are set as independent variables to determine the main variables by identifying the correlation coefficient of each attribute.

B. Support Vector Regression

Support vector regression is a predictive model based on SVM, a neural network that usually has three layers, a powerful form of supervised learning. The model is based on a subset of training data. The advantages of support vector regression are that it is capable of processing non-linear results, provides only one possible optimal solution, and able to overcome a small sample learning issues.

The potential to produce market predictions in several markets, including real estate, shows that this model can overcome the non-linear regression problems and small sample learning problems. Moreover, as this model did not depend on probability distribution assumptions, and the ability of mapping the input attribute, either linear or non-linear, this model was commonly used at house price modelling. Support vector regression offers huge benefits in so many aspects as this model can avoid over-

fitting problems, while ensuring a single optimum solution by minimizing structural risks and empirical risks. In this field of study, support vector regression is used to collect details on neighborhood, structural and locational attributes.

C. Artificial neural Network

In 1958, created artificial neural network known as ANN. Walter Pitts and Warren McCulloch published a paper entitled "A Logical Calculus of Ideas Immanent in Nervous Activity" in the year 1943 which notes that a neural network may artificially be created, based on the role and structure of a biological neural network. In another research, as this model would often promote learning, artificial neural networks are claimed to be artificial brain diagrams.

The artificial neural network model has always been selected when a non-linear attribute is involved. The analysis of home price estimation should also use this model as a spatial consideration for the price of housing is also non-linear. Therefore, as in, their study produces a good result, thus it is promising to provide an exact predictive model utilizing the artificial neural network algorithm. This system, however, has very limited performance. ANN can model complex non-linear relationships as house price predictions involve many non-linear variables.

D. Gradient Boost

Gradient boosting was created by in 1999 and is a commonly used machine learning algorithm because of its performance, consistency and interpretability. Gradient boosting delivers state-of-the-art in various machine learning activities, such as multistage classification, click prediction and ranking. With the advent of big data in recent years, gradient boosting faces new challenges, especially with regard to the balance between accuracy and performance. There are few parameters for gradient boosting. To ensure a dynamic balance between fit and regularity, the following

steps can be taken to select parameters: (1) Setting regularization parameters (λ , α), (2) reducing learning rate and decide those optimal parameters again.

Finding and Discussion

The associations between the house price and predicting model included in this segment have been explored. In addition, the impact of various attributes on specific model have also been evaluated and debated. Based on reviewing numerous papers, there are several attributes used by researchers in their work to forecast

house prices. All of these attributes can be divided into 4 main categories which are locational, structural, neighborhood and economic attributes.

The locational attribute consists of variables which described the

accessibility to shopping mall, accessibility to school, accessibility to hospital, restaurants and availability of public transport. Meanwhile, the structural attribute consists variables of number of bedrooms, number of bathrooms, floor area, garage and patio, property age housing and lot size. The neighborhood on the other hand described the socioeconomic variables, crime rates, place of worship, pleasant landscape, and quiet atmosphere which the variables is quite subjective. And lastly the economic attributes consist of income factor and cost of material factor. The most common used attributes by previous research regarding house price prediction are locational and structural attributes, as neighborhood and economic attributes are difficult to define and measure.

The classification of attributes made it easier to analyze the effects of different attributes on different models. The tabulation of finding based on evaluation of previous study can be seen. Looking at the table above, it can be seen that support vector regression has the smallest RMSE value, 0.0047. Technically, the RMSE value of a model is highly dependent on the attributes used during the prediction process. Most of the model that are using the same attributes (locational attributes) will generate a very low RMSE value indicating the best model. However, this cannot simply show the best model because several previous studies rarely provide the RMSE value to justify their model being the best model. On the basis of the analysis table, the locational attribute may be assumed to be the main attribute used by several models such as support vector regression and artificial neural network support. The RMSE value is very low with the presence of the locational attribute only, however, the RMSE value is quite high when the structural attribute is combined with the locational attribute for the input to make a prediction. Next, ANN provides the second lowest of RMSE value which is 0.0581. Findings revealed that locational attributes are indeed the relevant attributes used in the ANN model to forecast house price. Similar to SVR, the RMSE value is fairly high when locational and structural attributes are used together. This indicates that the locational attribute should be used alone to achieve low RMSE values by ANN model. In the meantime, the XGBoost model also yields the lowest RMSE while only the structural attribute is involved.

However, research on model with structural attributes alone is very limited as previous research focused mainly on locational attributes or the combination of locational and structural attributes in order to predict house prices. In general, our analysis suggests that SVR, ANN and XGBoost are the most efficient models compared to other models, whereas locational attributes are the main attribute in predicting house price.

Conclusions

This paper examined and analyzed the current research on the significant attributes of house price and analyzed the data mining techniques used to predict house price. Technically, houses with a strategic location such as the accessibility to shopping mall or other facilities tend to be more expensive than houses in rural areas with limited numbers of facilities. The accurate prediction model would allow investors or house buyers to determine the realistic price of a house as well as the house developers to decide the affordable house price. This paper addressed the attributes used by previous researchers to forecast a house price using various prediction models. Taken together, the results of the survey have shown the potential of SVR, ANN and XGBoost in predicting house prices. These models were developed based on several input attributes and they work significantly positive with house price. In conclusion, the impact of this research was intended to help and assist other researchers in developing a real model which can easily and accurately predict house prices. Further work on a real model needs to be done with the utilization of our findings to confirm them.

2.3. Housing Price Prediction via Improved Machine Learning Techniques

Abstract

House Price Index (HPI) is commonly used to estimate the changes in housing price. Since housing price is strongly correlated to other factors such as location, area, population, it requires other information apart from HPI to predict individual housing price. There has been a considerably large number of papers adopting traditional machine learning approaches to predict housing prices accurately, but they rarely concern about the performance of individual models and neglect the less popular yet complex models. As a result, to explore various impacts of features on prediction methods, this paper will apply both traditional and advanced machine learning approaches to investigate the difference among several advanced models. This paper will also comprehensively validate multiple techniques in model implementation on regression and provide an optimistic result for housing price prediction.

Introduction

Commonly, House Price Index (HPI) is used to measure price changes of residential housing in many countries, such as the US Federal Housing Finance Agency HPI, S&P/Case-Shiller price index, UK National

Statistics HPI, UK Land Registry's HPI, UK Halifax HPI, UK Rightmove HPI and Singapore URA HPI. The HPI is a weighted, repeatsales index, meaning that it measures average price changes in repeat sales or refinancings on the same properties. This information is obtained by reviewing repeat mortgage transactions on single-family properties whose mortgages have been purchased or securitized by Fannie Mae or Freddie Mac since January 1975. With some analytical tools, it allows housing economists to estimate changes in the rates of mortgage defaults, prepayments, and housing affordability in specific geographic areas. Because HPI is a rough indicator calculated from all transactions, it is inefficient to predict the price of a specific house. Many features such as district, age, and the number of floors also need to be considered instead of just the repeat sales in previous decades. In recent years, due to the growing trend towards Big Data, machine learning has become a vital prediction approach because it can predict house prices more accurately based on their attributes, regardless of the data from previous years. Several studies explored this problem and proved the capability of the machine learning approach; however, most of them compared the models' performance but did not consider the combination of different machine learning models. S. Lu et al. did conducted an experiment using a hybrid regression technique on forecasting house price data, but it requires intensive parameter tuning to find the optimal solution. Due to the importance of model combination, this paper adopted the Stacked Generalization approach a machine learning ensemble technique, to optimize the predicted values. We used the "Housing Price in Beijing" dataset, fetched and uploaded to Kaggle by Q. Qiu. By applying several methods on this dataset, we could validate the performance of each individual approach. The lowest Root Mean Squared Logarithmic Error (RMSLE) is 0.16350 on the test set, which belongs to the Stack Generalization method. The paper is structured as follows: Section 2 illustrates the details of the methodology; Section 3 compares the results; and Section 4 discusses the results, draws a conclusion, as well as proposes further potential directions to study the problem.

Methodology

Data Preprocessing

"Housing Price in Beijing" is a dataset containing more than 300, 000 data with 26 variables representing housing prices traded between 2009 and 2018. These variables, which served as features of the dataset, were then used to predict the average price per square meter of each house. The next step was to investigate missing data. Variables with more than 50% missing data would be removed from the dataset. The variable "Day on

market” was removed because of 157, 977 missing data. Any observation which had missing values were also removed from the dataset. Below are a few feature engineering processes which were done to cleanse the dataset:

- Remove attributes indicating the number of kitchens, bathrooms, and drawing rooms due to their ambiguity.
- Set the number of living rooms (bedrooms were mistranslated to living rooms) in a range from 1 to 4.
- Add attribute “distance” indicating the distance of the house from the center of Beijing.
- Replace attribute “constructionTime” with attribute “age” by deducting the year that the house constructed from the current year (2019).
- Set minimum values for attributes “price” and “area”.
- Split the attribute “flfloor” into attributes “flfloorType” and “flfloorHeight”.

Data Analysis

Exploratory data analysis is an essential step before building a regression model. In this way, researchers can discover the implicit patterns of the data, which in turn helps choose appropriate machine learning approaches. features houses in Beijing as data points on the map of Beijing. the oldest houses are concentrated densely in the center of Beijing, while the newest ones are spread sparsely in the suburban areas. the most expensive houses centralizes close to the center of Beijing, while the cheapest ones spreads in the suburban periphery. Since the patterns are similar, a strong correlation between location, age, and price can be observed. There are also noticeable differences in housing prices across 13 districts, which are summarized in.

Besides the location features, other features of the house also significantly contribute to the models’ performance. the difference in price among several building types can be clearly illustrated. Since bungalow is an old building type and is more likely to be built close to the center of Beijing, its price is costly regardless of the small area of a house.

Model Selection

Before building models, the data should be processed accordingly so that the models could learn the patterns more effifficiently. Specifically, numerical values were standardized, while categorical values were one-hot-encoded. After being processed, the dataset included 58 features. illustrates the cumulative explained variance. Since most of the features are categorical, it is evident that the variance almost converges at the 30th

component. Then, the dataset was split into training set and test set with a ratio of 4 : 1 by utilizing the scikit-learn package.

Random Forest

Random Forest is a kind of ensemble models that combines the prediction of multiple decision trees to create a more accurate final prediction.

Random Forest is a verified powerful tool based on previous studies. The random forest algorithm can be summarized in the following steps by S. Raschka and V. Mirjalili in their book “Python Machine Learning”:

1. Draw a random bootstrap sample of size n (randomly choose n samples from the training set with replacement).
2. Grow a decision tree from the bootstrap sample, at each node:
 - (a) Randomly select d features without replacement.
 - (b) Split the node using the feature that provides the best split according to the objective function, for instance, maximizing the information gain.
3. Repeat the steps 1-2 k times.
4. Aggregate the prediction by each tree to assign the class label by majority vote.

In this paper, we used the RandomForestClassifier class provided by sklearn. The

RandomForestClassifier has a `n_estimators` parameter that allows to indicate how many trees to build, which we set at 900. While adding more trees to the random forest normally improves accuracy, it also increases the overall training time of the model. The class also includes the `bootstrap` parameter which we set to `True`.

Regarding random forest subsets, however, only a constrained set of features will be used to introduce variation into the trees. By iterating the model multiple times, we also added a few parameters when we initialized the RandomForestClassifier to further enhance the performance:

- Set `max_depth = 20`, which restricts these trees can only go to 20.
- Set `min_samples_split = 10`, which only allows a node that should have at most ten rows before it can be split.

The result of this model on training data is exceptional, 0.12980. The RMSLE of the training set is the lowest among other methods.illustrates the performance of this model on the training set where the X-axis is the prediction result and the Y-axis is the actual housing price.

Extreme Gradient Boosting (XGBoost)

XGBoost is a scalable machine learning system for tree boosting. The system is available as an open-source package. The system has generated a significant impact and been widely recognized in various machine learning and data mining challenges.

The most crucial reason why XGBoost succeeds is its scalability in all scenarios. The system runs more than ten times faster than existing popular solutions on a single machine and scales to billions of examples in distributed or memory-limited settings. The scalability of XGBoost is due to several major systems and algorithmic optimizations including a novel tree learning algorithm for handling sparse data and a theoretically justified weighted quantile sketch procedure enabling instance weight handling in approximate tree learning. Parallel and distributed computing make

learning faster, which allows quicker model exploration. More importantly, the model exploits out-of-core computation and enables data scientists to process a hundred millions of examples on a desktop. Finally, after combining these techniques to make an end-to-end system, it can scale to even more extensive data with the least amount of cluster resources. In this paper, we utilized the XGBRegressor from xgboost open-source package. After tweaking the XGBoost model multiple times, we set our parameter to the following:

- Set `learning_rate` = 0.1
- Set `n_estimators` = 200
- Determined the optimal tree specific parameters `min_child_weight` = 2, `subsample` = 1, `colsample_bytree` = 0.8
- Set regularization parameter: `reg_lambda` = 0.45, `reg_alpha` = 0, `gamma` = 0.5

The model performed with a high accuracy where the RMSLE of the training set is around 0.16118. the Extreme Gradient Boosting prediction in X-axis, and the actual price in Y-axis for training data.

Light Gradient Boosting Machine (LightGBM)

LightGBM is a gradient boosting framework that uses a tree-based learning algorithm. LightGBM has faster training speed with lower memory usage compare to XGBoost. Moreover, it can also support parallel and GPU learning or handle large scale of data. Even though both of the aforementioned boosting models follow Gradient Boosting Algorithm, XGBoost grows tree level-wise and LightGBM grows tree leaf-wise. There are also detailed differences in modelling making them outstanding among different gradient boosting models. In this paper, we utilized the LGBMRegressor from lightgbm open-source package. The optimal parameters for the model are listed as follows:

- Set `learning_rate` = 0.15
- Set `n_estimators` = 64
- Set optimal tree specific parameters: `min_child_weight` = 2, `num_leaves` = 36, `colsample_bytree` = 0.8

- Set regularization parameter: $\text{reg_lambda} = 0.40$ When applying the LightGBM method, the model achieved a decent accuracy with the loss of 0.16687 on the

training set. Even though the performance was not as good as the other methods, LightGBM is the fastest among all models researched on this paper. illustrates the performance of this model on the training set where the X-axis is the prediction result and the Y-axis is the actual housing price.

Hybrid Regression

Hybrid Regression Model is a model that includes two or more different regression models. The coupling effect of multiple regression models is proven by S. Lu et al., in which a combination of 65% Lasso and 35% XGBoost achieves a RMSLE of 0.11260 on test set of Kaggle This result is significantly better than results of Lasso and XGBoost in that paper, which are 0.11449 and 0.11843 respectively. A similar approach was also tested on the “Housing Price of Beijing” dataset. The model consisted of 33.33% Random Forest, 33.33% XGBoost, and 33.33% LightGBM from the previous models. The hybrid regression model achieved 0.14969 on the training set. This method averages the individual predictions to form a final prediction.

Therefore, it is a fair approach to balance between bias and variance in the composite models. This technique also supports weight assignment to each component model, but it may lead to bias over one model, losing the benefits of generalization. illustrates the performance of this model on the training set where the X-axis is the prediction result and the Y-axis is the actual housing price.

Stacked Generalization

Stacked Generalization is a machine learning ensembling technique introduced by D. Wolpert. A Python package for stacking named “Vecstack” was built and uploaded to GitHub by I. Ivanov. The main idea of this method is to use the predictions of previous models as features for another model. This approach also utilizes the k-fold cross-validation technique to avoid overfitting. This paper used the most common architecture, 2-level stacking architecture, to predict the housing price, where

the first stacking level consisted of Random Forest and LightGBM and the second stacking level was XGBoost. The stacking model also used 5-fold cross-validation because the dataset is relatively large. The result of the model on the training set, which is 0.16404, is not as impressive as the Hybrid Regression approach. shows the prediction of the Stacked Generalization Model on the X-axis and the actual price on the Y-axis.

Results

Many iterations of performance tuning were done to find the optimal solution of each model. Random Forest Regression, XGBoost, and LightGBM were intensively tuned by function GridSearchCV provided by scikit-learn to achieve the results listed. For Hybrid Regression and Stacking methods, performance tuning was not required since both methods were combinations of the best regressions. Instead, architecture implementation could be considered to further enhance the prediction.

As listed, the best results belong to Random Forest for the training set and Stacked Generalization Regression for the test set. Since 49 of 58 features of the one-hot-encoded dataset were boolean values, it is reasonable that the Random Forest worked well on this dataset. However, Random Forest was prone to overfitting, which led to a decent performance on unseen data. Both XGBoost and LightGBM were not subject to overfitting, but the accuracy of their predictions was not as good as Random Forest on both training and test data. Unlike the three traditional machine learning methods, Hybrid Regression and Stacked Generalization Regression were neither tuned nor implemented sophisticatedly but delivered promising results on the training set and test set. Since Random Forest was proven to be overfitting, Hybrid Regression could be considered as the best model on training set where the RSMLE is 0.14969. Surprisingly, Stacked Generalization Regression did not work well on the training set as Hybrid Regression, but this model did exceptionally on the test set. This is probably because of the two reasons:

- K-fold cross-validation: k-fold cross-validation is a suitable method to find an acceptable bias-variance tradeoff. Stacking Regression utilizes this technique to obtain the generalization performance for each component model.
- Coupling effect of multiple regressions: different regression methods may support each other. The second stacking level can learn again and predict the housing prices accurately based on the pre-estimated prices from the first stacking level.

Even though Hybrid Regression shares the coupling effect of multiple regressions with Stacked Generalization Regression, Hybrid Regression is less competitive because its learning mechanism is only averaging every prediction. Based on which illustrate the performance of both models on the first 200 observations of the test set, the difference between the two models is not considerably large. On one hand, Hybrid Regression predicts more accurately on outliers (extremely high or low housing prices). On the other hand, Stacked Generalization Regression

Re-442 Quang Truong et al. / Procedia Computer Science
174 (2020) 433–442 10 Quang Truong et al. / Procedia Computer Science

00 (2019) 000–000 gression performs better on common observations (standard housing prices). This proves that Stacked Generalization Regression is slightly better than Hybrid Regression in generalization.

Discussion and Conclusion

This paper investigates different models for housing price prediction. Three different types of Machine Learning methods including Random Forest, XGBoost, and LightGBM and two techniques in machine learning including Hybrid Regression and Stacked Generalization Regression are compared and analyzed for optimal solutions. Even though all of those methods achieved desirable results, different models have their own pros and cons. The Random Forest method has the lowest error on the training set but is prone to be overfitting. Its time complexity is high since the dataset has to be fit multiple times. The XGBoost and LightGBM are decent methods when comparing accuracy, but their time complexities are the best, especially LightGBM. The Hybrid Regression method is simple but performs

a lot better than the three previous methods due to the generalization. Finally, the Stacked Generalization Regression method has a complicated architecture, but it is the best choice when accuracy is the top priority. Even though Hybrid Regression and Stacked Generalization Regression deliver satisfactory results, time complexity must be taken into consideration since both of them contain Random Forest, a high time complexity model. Stacked Generalization Regression also has K-fold cross-validation in its mechanism so it has the worst time complexity. Further research

about the following topics should be conducted to further investigate these models, especially the combinations of different models:

- The coupling effect of multiple regression models.
- The “re-learn” ability of machine learning models.
- The combination of Machine Learning and Deep Learning methods.
- The driven factors for the good performance of tree-based models.
- The faster ways to fit complex models.

Chapter 3

Existing system:

- House price prediction can be divided into two categories, first by focusing on house characteristics, and secondly by focusing on the model used in house price prediction.
- the existing housing price in Jakarta, Indonesia using the conceptual model and questionnaires.
- Based on the results, the attributes or factors affecting the house price differ for each house construction in Jakarta, therefore accepting the validity of this analysis as the main purpose of this research is to classify the factor or attributes affecting the house price prediction value is less accuracy .

Disadvantage:

- Error in price prediction.
- Low accuracy by using the algorithm.
- It give the output random only not a perfect result.
- Slow process to predict the house value.
- The house price prediction is nit depend any parameter so it is less accuracy process.

Proposed system:

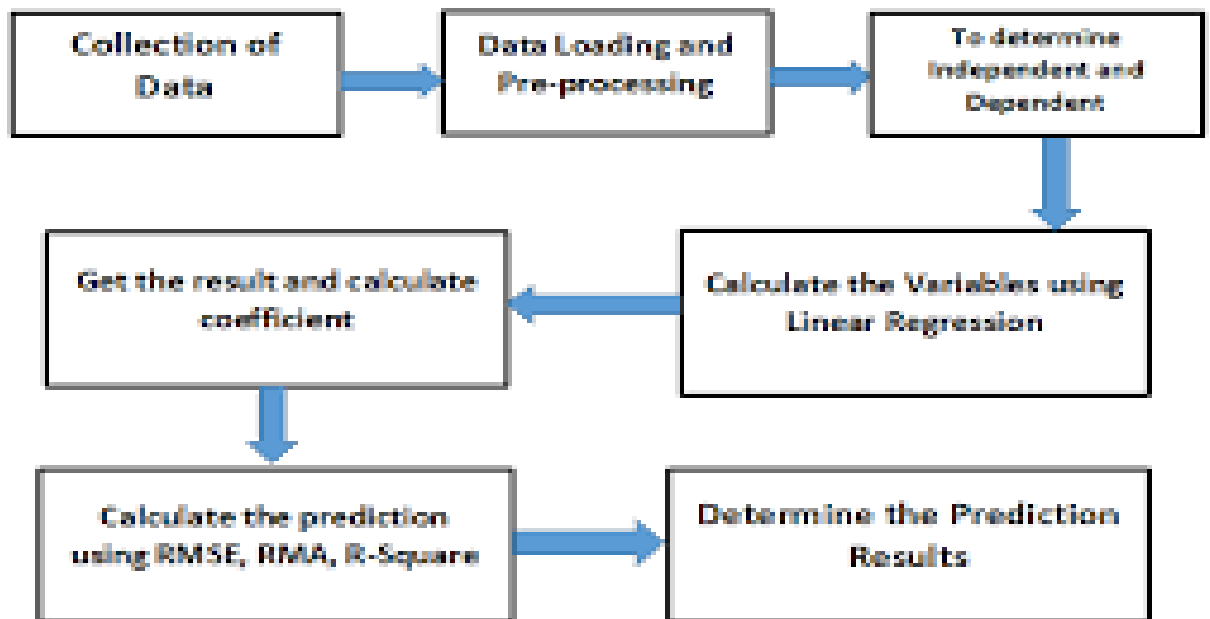
- The dataset containing more than 300, 000 data with 26 variables representing housing prices traded the house price.
- These variables, which served as features of the dataset, were then used to predict the average price per square meter of each house.
- Exploratory data analysis is an essential step before building a regression model. In this way, researchers can discover the implicit patterns of the data, which in turn helps choose appropriate machine learning approaches.
- We have using the random forest algorithm, Extreme Gradient Boosting
- Before building models, the data should be processed accordingly so that the models could learn the patterns more efficiently

Advantage:

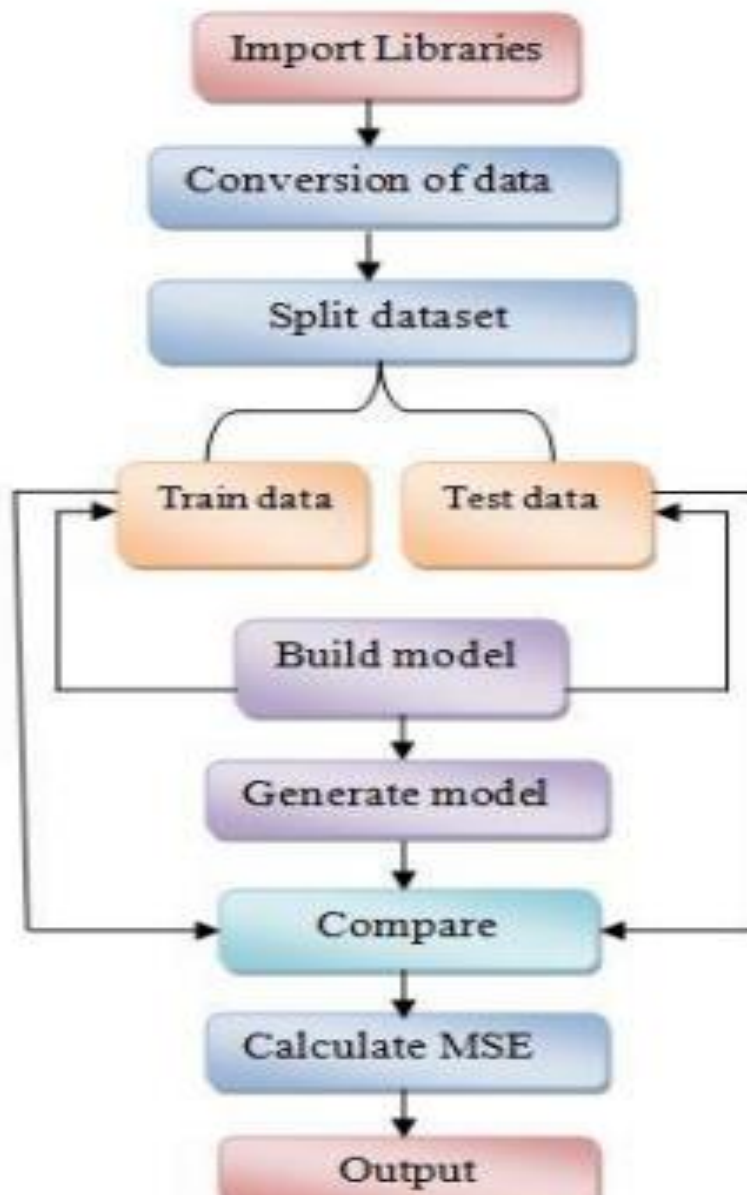
- Random Forest is a kind of ensemble models that combines the prediction of multiple decision trees to create a more accurate final prediction.
- Random Forest is a verified powerful tool.
- It give a perfect result by using of random forest algorithm.

- XGBoost is a scalable machine learning system for tree boosting. The system is available as an open-source package. The system has generated a significant impact and been widely recognized in various machine learning and data mining challenges .

Block diagram:



Flow diagram:



Linear Regression:

Introduction

Regression is a supervised learning technique that supports finding the correlation among variables. A regression problem is when the output variable is a real or continuous value.

In this article, we will understand the following concepts:

1. What is a Regression?
2. Types of a Regression.

3. What is the mean of Linear regression and the importance of Linear regression?
4. Importance of cost function and gradient descent in a Linear regression.
5. Impact of different values for learning rate.
6. Implement use case of Linear regression with python code.

What is a Regression

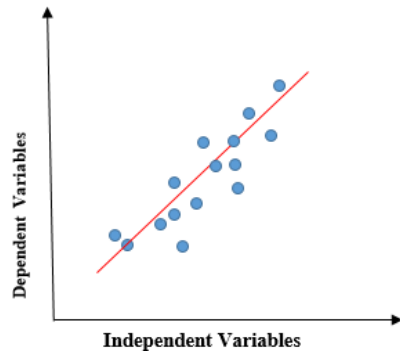
In Regression, we plot a graph between the variables which best fit the given data points. The machine learning model can deliver predictions regarding the data. In naïve words, **“Regression shows a line or curve that passes through all the data points on a target-predictor graph in such a way that the vertical distance between the data points and the regression line is minimum.”** It is used principally for prediction, forecasting, time series modeling, and determining the causal-effect relationship between variables.

Types of Regression models

1. Linear Regression
2. Polynomial Regression
3. Logistics Regression

Linear Regression

Linear regression is a quiet and simple statistical regression method used for predictive analysis and shows the relationship between the continuous variables. Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis), consequently called linear regression. If there is a single input variable (x), such linear regression is called **simple linear regression**. And if there is more than one input variable, such linear regression is called **multiple linear regression**. The linear regression model gives a sloped straight line describing the relationship within the variables.



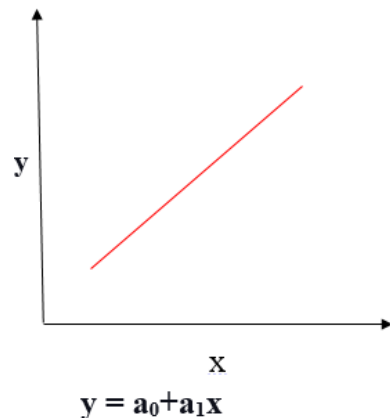
The above graph presents the linear relationship between the dependent variable and independent variables. When the value of x (**independent variable**) increases, the value of y (**dependent variable**) is likewise increasing. The red line is referred to as the best fit straight line. Based on the given data points, we try to plot a line that models the points the best.

Let's say we want to estimate the salary of an employee based on year of experience. You have the recent company data, which indicates that the relationship between experience and salary. Here year of experience is an independent variable, and the salary of an employee is a dependent variable, as the salary of an employee is dependent on the experience of an employee. Using this insight, we can predict the future salary of the employee based on current & past information.

A regression line can be a Positive Linear Relationship or a Negative Linear Relationship.

Positive Linear Relationship

If the dependent variable expands on the Y-axis and the independent variable progress on X-axis, then such a relationship is termed a Positive linear relationship.



Negative Linear Relationship

If the dependent variable decreases on the Y-axis and the independent variable increases on the X-axis, such a relationship is called a negative linear relationship. The goal of the linear regression algorithm is to get the best values for a_0 and a_1 to find the best fit line. The best fit line should have the least error means the error between predicted values and actual values should be minimized.

Cost function

The cost function helps to figure out the best possible values for a_0 and a_1 , which provides the best fit line for the data points.

Cost function optimizes the regression coefficients or weights and measures how a linear regression model is performing. The cost function is used to find the accuracy of the **mapping function** that maps the input variable to the output variable. This mapping function is also known as **the Hypothesis function**.

In Linear Regression, **Mean Squared Error (MSE)** cost function is used, which is the average of squared error that occurred between the predicted values and actual values.

By simple linear equation $y = mx + b$ we can calculate MSE as:

Let's y = actual values, y_i = predicted values Using the MSE function, we will change the values of a_0 and a_1 such that the MSE value settles at the minima. Model parameters **x_i , b (a_0 , a_1)** can be manipulated to minimize the cost function. These parameters

can be determined using the gradient descent method so that the cost function value is minimum.

Gradient descent

Gradient descent is a method of updating a_0 and a_1 to minimize the cost function (MSE). A regression model uses gradient descent to update the coefficients of the line ($a_0, a_1 \Rightarrow x_i, b$) by reducing the cost function by a random selection of coefficient values and then iteratively update the values to reach the minimum cost function.

Imagine a pit in the shape of U. You are standing at the topmost point in the pit, and your objective is to reach the bottom of the pit. There is a treasure, and you can only take a discrete number of steps to reach the bottom. If you decide to take one footstep at a time, you would eventually get to the bottom of the pit but, this would take a longer time. If you choose to take longer steps each time, you may get to sooner but, there is a chance that you could overshoot the bottom of the pit and not near the bottom. In the gradient descent algorithm, the number of steps you take is the learning rate, and this decides how fast the algorithm converges to the minima.

To update a_0 and a_1 , we take gradients from the cost function. To find these gradients, we take partial derivatives for a_0 and a_1 . The partial derivatives are the gradients, and they are used to update the values of a_0 and a_1 . Alpha is the learning rate.

Impact of different values for learning rate

The blue line represents the optimal value of the learning rate, and the cost function value is minimized in a few iterations. The green line represents if the learning rate is lower than the optimal value, then the number of iterations required high to minimize the cost function. If the learning rate selected is very high, the cost function could continue to increase with iterations and saturate at a value higher than the minimum value, that represented by a red and black line.

Use case

In this, I will take random numbers for the dependent variable (salary) and an independent variable (experience) and will predict the impact of a year of experience on salary.

Steps to implement Linear regression model

import some required libraries

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

Define the dataset

```
x=np.array([2.4,5.0,1.5,3.8,8.7,3.6,1.2,8.1,2.5,5,1.6,1.6,2.4,3.9,5.4])
```

```
y=np.array([2.1,4.7,1.7,3.6,8.7,3.2,1.0,8.0,2.4,6,1.1,1.3,2.4,3.9,4.8])
```

```
n = np.size(x)
```

Plot the data points

```
plt.scatter(experience,salary, color = 'red')
plt.xlabel("Experience") plt.ylabel("Salary") plt.show()
```

The main function to calculate values of coefficients

1. Initialize the parameters.
2. Predict the value of a dependent variable by given an independent variable.
3. Calculate the error in prediction for all data points.
4. Calculate partial derivative w.r.t a_0 and a_1 .
5. Calculate the cost for each number and add them.
6. Update the values of a_0 and a_1 .

```
#initialize the parameters a0 = 0 #intercept a1 = 0 #Slop lr
= 0.0001 #Learning rate iterations = 1000 # Number of
iterations error = [] # Error array to calculate cost for each
```

```

iterations. for itr in range(iterations): error_cost = 0 cost_a0 =
0 cost_a1 = 0 for i in range(len(experience)): y_pred =
a0+a1*experience[i] # predict value for given x error_cost =
error_cost +(salary[i]-y_pred)**2 for j in
range(len(experience)): partial_wrt_a0 = -2 *(salary[j] - (a0
+ a1*experience[j])) #partial derivative w.r.t a0 partial_wrt_a1
= (-2*experience[j])*(salary[j]-(a0 + a1*experience[j]))
#partial derivative w.r.t a1 cost_a0 = cost_a0 + partial_wrt_a0
#calculate cost for each number and add cost_a1 = cost_a1 +
partial_wrt_a1 #calculate cost for each number and add a0 = a0 -
lr * cost_a0 #update a0 a1 = a1 - lr * cost_a1 #update a1
print(itr,a0,a1) #Check iteration and updated a0 and a1
error.append(error_cost) #Append the data in array

At approximate iteration 50- 60, we got the value of a0 and
a1.print(a0) print(a1)

```

Plotting the error for each iteration.

```

plt.figure(figsize=(10,5)) plt.plot(np.arange(1,len(error)+1),error,color='red',li
newwidth
plt.title("Iteration vs error") plt.xlabel("iterations") plt.ylabel("Error")

```

Predicting the values.

```
pred = a0+a1*experience print(pred)
```

Plot the regression line:

```

plt.scatter(experience,salary,color = 'red')
plt.plot(experience,pred, color = 'green')
plt.xlabel("experience") plt.ylabel("salary")

```

Analyze the performance of the model by calculating the mean squared error.

```
error1 = salary - pred
se = np.sum(error1 ** 2)
mse = se/n
print("mean squared error is", mse)
```

Use the scikit library to confirm the above steps.

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
experience =
experience.reshape(-1,1)
model = LinearRegression()
model.fit(experience,salary)
salary_pred =
model.predict(experience)
Mse = mean_squared_error(salary,
salary_pred)
print('slop', model.coef_)
print("Intercept",
model.intercept_)
print("MSE", Mse)
```

Summary:

In Regression, we plot a graph between the variables which best fit the given data points. Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis). To calculate best-fit line linear regression uses a traditional slope-intercept form. A regression line can be a Positive Linear Relationship or a Negative Linear Relationship.

The goal of the linear regression algorithm is to get the best values for θ_0 and θ_1 to find the best fit line and the best fit line should have the least error. In Linear Regression, **Mean Squared Error (MSE)** cost function is used, which helps to figure out the best possible values for θ_0 and θ_1 , which provides the best fit line for the data points. Using the

MSE function, we will change the values of a_0 and a_1 such that the MSE value settles at the minima. Gradient descent is a method of updating a_0 and a_1 to minimize the cost function (MSE)

Decision Tree Algorithm:

Introduction

Till now we have learned about linear regression, logistic regression, and they were pretty hard to understand. Let's now start with Decision tree's and I assure you this is probably the easiest algorithm in Machine Learning. There's not much mathematics involved here. Since it is very easy to use and interpret it is one of the most widely used and practical methods used in Machine Learning.

Contents

1. What is a Decision Tree?
2. Example of a Decision Tree
3. Entropy
4. Information Gain
5. When to stop Splitting?
6. How to stop overfitting?

- `max_depth`
- `m`
- `i`
- `n`
- `—`
- `s`

a
m
p
l
e
s
—
s
p
l
i
t

m
i
n
—
s
a
m
p
l
e
s
—
l
e

a

f

m

a

x

—

f

e

a

t

u

r

e

s

7. Pruning

-

P

-

o

s

t

-

p

r

u

n

i

n

g

P

r

e

-

p

r

u

n

i

n

g

8. Endnotes

What is a Decision Tree?

It is a tool that has applications spanning several different areas. Decision trees can be used for classification as well as regression problems. The name itself suggests that it uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits. It starts with a root node and ends with a decision made by leaves.

Before learning more about decision trees let's get familiar with some of the terminologies.

Root Nodes – It is the node present at the beginning of a decision tree from this node the population starts dividing according to various features.

Decision Nodes – the nodes we get after splitting the root nodes are called Decision Node

Leaf Nodes – the nodes where further splitting is not possible are called leaf nodes or terminal nodes

Sub-tree – just like a small portion of a graph is called sub-graph similarly a sub-section of this decision tree is called sub-tree.

Pruning – is nothing but cutting down some nodes to stop overfitting.

Example of a decision tree.

Let's understand decision trees with the help of an example.

Decision trees are upside down which means the root is at the top and then this root is split into various several nodes. Decision trees are nothing but a bunch of if-else statements in layman terms. It checks if the condition is true and if it is then it goes to the next node attached to that decision.

In the below diagram the tree will first ask what is the weather?

Is it sunny, cloudy, or rainy? If yes then it will go to the next feature which is humidity and wind. It will again check if there is a strong wind or weak, if it's a weak wind and it's rainy then the person may go and play.

Did you notice anything in the above flowchart? We see that if the weather is cloudy then we must go to play. Why didn't it split more? Why did it stop there?

To answer this question, we need to know about few more concepts like entropy, information gain, and Gini index. But in simple terms, I can say here that the output for the training dataset is always yes for cloudy weather, since there is no disorderliness here we don't need to split the node further.

The goal of machine learning is to decrease uncertainty or disorders from the dataset and for this, we use decision trees.

Now you must be thinking how do I know what should be the root node? what should be the decision node? when should I stop splitting? To decide this, there is a metric called "Entropy" which is the amount of uncertainty in the dataset.

Entropy

Entropy is nothing but the uncertainty in our dataset or measure of disorder. Let me try to explain this with the help of an example.

Suppose you have a group of friends who decides which movie they can watch together on Sunday. There are 2 choices for movies, one is “**Lucy**” and the second is “**Titanic**” and now everyone has to tell their choice. After everyone gives their answer we see that “Lucy” gets 4 votes and “Titanic” gets 5 votes. Which movie do we watch now? Isn’t it hard to choose 1 movie now because the votes for both the movies are somewhat equal.

This is exactly what we call disorderness, there is an equal number of votes for both the movies, and we can’t really decide which movie we should watch. It would have been much easier if the votes for “Lucy” were 8 and for “Titanic” it was 2. Here we could easily say that the majority of votes are for “Lucy” hence everyone will be watching this movie.

In a decision

tree, the output

is mostly “yes”

or “no” The

formula for

Entropy is

shown below:

Here p_+

is the

probabili

ty of

positive

classp—

is the

probabili

ty of

negative

class

S is the subset of the training example

How do Decision Trees use Entropy?

Now we know what entropy is and what is its formula, Next, we need to know that how exactly does it work in this algorithm.

Entropy basically measures the impurity of a node. Impurity is the degree of randomness; it tells how random our data is. A **pure sub-split** means that either you should be getting “yes”, or you should be getting “no”.

Suppose feature 1 had 8 yes and 4 no, after the split feature 2 get 5 yes and 2 no whereas feature 3 gets 3 yes and 2 no.

We see here the split is not pure, why? Because we can still see some negative classes in both the feature. In order to make a decision tree, we need to calculate the impurity of each split, and when the purity is 100% we make it as a leaf node.

To check the impurity of feature 2 and feature 3 we will take the help for Entropy formula. For feature 2 the entropy is as follows: feature 3,

We can clearly see from the tree itself that feature 2 has low entropy or more purity than feature 3 since feature 2 has more “yes” and it is easy to make a decision here.

Always remember that the higher the Entropy, the lower will be the purity and the higher will be the impurity.

As mentioned earlier the goal of machine learning is to decrease the uncertainty or impurity in the dataset, here by using the entropy we are getting the impurity of a feature or a particular node, we don't know if the parent entropy or the entropy of a particular node has decreased or not.

For this, we bring a new metric called “Information gain” which tells us how much the parent entropy has decreased after splitting it with some feature.

Information Gain

Information gain measures the reduction of uncertainty given some feature and it is also a deciding factor for which attribute should be selected as a decision node or root node.

It is just entropy of the full dataset – entropy of the dataset given some feature. To understand this better let's consider an example: Suppose our entire population has a total of 30 instances. The dataset is to predict whether the person will go to the gym or not. Let's say 16 people go to the gym and 14 people don't

Now we have two features to predict whether he/she will go to the gym or not. Feature 1 is “**Energy**” which takes two values “high” and “low”

Feature 2 is “**Motivation**” which takes 3 values “No motivation”, “Neutral” and “Highly motivated”.

Let’s see how our decision tree will be made using these 2 features. We’ll use information gain to decide which feature should be the root node and which feature should be placed after the split.

Let’s calculate the entropy:

To see the weighted average of entropy of each node we will do as follows:

Now we have the value of $E(\text{Parent})$ and $E(\text{Parent}|\text{Energy})$, information gain will be:

Our parent entropy was near 0.99 and after looking at this value of information gain, we can say that the entropy of the dataset will decrease by 0.37 if we make “Energy” as our root node.

Similarly, we will do this with the other feature “Motivation” and calculate its information gain.

Let’s calculate the entropy here:

To see the weighted average of entropy of each node we will do as follows:

Now we have the value of $E(\text{Parent})$ and $E(\text{Parent}|\text{Motivation})$, information gain will be:

We now see that the “Energy” feature gives more reduction which is 0.37 than the “Motivation” feature. Hence we will select the feature which has the highest information gain and then split the node based on that feature.

In this example “Energy” will be our root node and we’ll do the same for sub-nodes. Here we can see that when the energy is “high” the entropy is low and hence we can say a person will definitely go to the gym if he has high energy, but what if the energy is low? We will again split the node based on the new feature which is “Motivation”.

When to stop splitting?

You must be asking this question to yourself that when do we stop growing our tree? Usually, real-world datasets have a large number of features, which will result in a large number of splits, which in turn gives a huge tree. Such trees take time to build and can lead to overfitting. That means the tree will give very good accuracy on the training dataset but will give bad accuracy in test data.

There are many ways to tackle this problem through hyperparameter tuning. We can set the maximum depth of our decision tree using the **max_depth** parameter. The more the value of **max_depth**, the more complex your tree will be. The training error will off-course decrease if we increase the **max_depth** value but when our test data comes into the picture, we will get a very bad accuracy. Hence you need a value that will not overfit as well as underfit our data and for this, you can use GridSearchCV.

Another way is to set the minimum number of samples for each spilt. It is denoted by **min_samples_split**. Here we specify the minimum number of samples required to do a spilt. For example, we can use a minimum of 10 samples to reach a decision. That means if a node has less than 10 samples then using this parameter, we can stop the further splitting of this node and make it a leaf node.

There are more hyperparameters such as :

min_samples_leaf – represents the minimum number of samples required to be in the leaf node. The more you increase the number, the more is the possibility of overfitting.

max_features – it helps us decide what number of features to consider when looking for the best split.

Pruning

It is another method that can help us avoid overfitting. It helps in improving the performance of the tree by cutting the nodes or sub-nodes which are not significant. It removes the branches which have very low importance.

There are mainly 2 ways for pruning:

(i) **Pre-pruning** – we can stop growing the tree earlier, which means we can prune/remove/cut a node if it has low importance **while growing** the tree.

(ii) **Post-pruning** – once our **tree is built to its depth**, we can start pruning the nodes based on their significance.

Endnotes

To summarize, in this article we learned about decision trees. On what basis the tree splits the nodes and how to stop overfitting. why linear regression doesn't work in the case of classification problems.

Lasso Regression:

Introduction

When we talk about Regression, we often end up discussing Linear and Logistic Regression. But, that's not the end. Do you know there are 7 types of Regressions?

Linear and logistic regression is just the most loved members from the family of regressions. Last week, I saw a recorded talk at NYC Data Science Academy from **Owen Zhang**, Chief Product Officer at DataRobot.

He said, **'if you are using regression without regularization, you have to be very special!'**. I hope you get what a person of his stature referred to.

I understood it very well and decided to explore regularization techniques in detail.

In this article, I have explained the complex science behind 'Ridge Regression' and 'Lasso Regression' which are the most fundamental regularization techniques used in data science, sadly still not used by many.

The overall idea of regression remains the same. It's the way in which the model coefficients are determined which makes all the difference. I strongly encourage you to go through multiple regression before reading this. You can take help from this article or any other preferred material.

Brief Overview

Ridge and Lasso regression are powerful techniques generally used for creating parsimonious models in presence of a 'large' number of features. Here 'large' can typically mean either of two things:

1. Large enough to enhance the tendency of a model to overfit (as low as 10 variables might cause overfitting)
2. Large enough to cause computational challenges. With modern systems, this situation might arise in case of millions or billions of

features Though Ridge and Lasso might appear to work towards a common goal, the inherent properties and practical use cases differ substantially. If you've heard of them before, you must know that they work by penalizing the magnitude of coefficients of features along with minimizing the error between predicted and actual observations. These are called 'regularization' techniques. The key difference is in how they assign penalty to the coefficients:

1. Ridge Regression:

Performs L2 regularization, i.e. adds penalty equivalent to **square of the magnitude** of coefficients Minimization objective = $LS\ Obj + \alpha * (\text{sum of square of coefficients})$

2. Lasso Regression:

Performs L1 regularization, i.e. adds penalty equivalent to **absolute value of the magnitude** of coefficients Minimization objective = $LS\ Obj + \alpha * (\text{sum of absolute value of coefficients})$ Note that here 'LS Obj' refers to 'least squares objective', i.e. the linear regression objective without regularization.

If terms like 'penalty' and 'regularization' seem very unfamiliar to you, don't worry we'll talk about these in more detail through the course of this article. Before digging further into how they work, let's try to get some intuition into why penalizing the magnitude of coefficients should work in the first place.

Ridge Regression

As mentioned before, ridge regression performs '**L2 regularization**', i.e. it adds a factor of sum of squares of coefficients in the optimization objective. Thus, ridge regression optimizes the following:

Objective = $RSS + \alpha * (\text{sum of square of coefficients})$

Here, α (alpha) is the parameter which balances the amount of emphasis given to minimizing RSS vs minimizing sum of square of coefficients. α can take various values:

1. $\alpha = 0$:

The objective becomes same as simple linear regression. We'll get the same coefficients as simple linear regression.

2. $\alpha = \infty$:

The coefficients will be zero. Why? Because of infinite weightage on square of coefficients, anything less than zero will make the objective infinite.

3. $0 < \alpha < \infty$:

The magnitude of α will decide the weightage given to different parts of objective. The coefficients will be somewhere between 0 and ones for simple linear regression.

I hope this gives some sense on how α would impact the magnitude of coefficients. One thing is for sure that any non-zero value would give values less than that of simple linear regression. By how much? We'll find out soon. Leaving the mathematical details for later, let's see ridge regression in action on the same problem as above. First, let's define a generic function for ridge regression similar to the one defined for simple linear regression. Note the 'Ridge' function used here. It takes 'alpha' as a parameter on initialization. Also, keep in mind that normalizing the inputs is generally a good idea in every type of regression and should be used in case of ridge regression as well. Now, let's analyze the result of Ridge regression for 10 different values of α ranging from $1e-15$ to 20. These values have been chosen so that we can easily analyze the trend with change in values of α . These would however differ from case to case. Note that each of these 10 models will contain all the 15 variables and only the value of alpha would differ. This is different from the simple linear regression case where each model had a subset of features.

Lasso Regression:

LASSO stands for Least Absolute Shrinkage and Selection Operator. I know it doesn't give much of an idea but there are 2 key words here – 'absolute' and 'selection'. Let's consider the former first and worry about the latter later.

Lasso regression performs **L1 regularization**, i.e. it adds a factor of sum of absolute value of coefficients in the optimization objective. Thus, lasso regression optimizes the following:

Objective = RSS + α * (sum of absolute value of coefficients)

Here, α (alpha) works similar to that of ridge and provides a trade-off between balancing RSS and magnitude of coefficients. Like that of ridge, α can take various values. Let's iterate it here briefly:

1. $\alpha = 0$: Same coefficients as simple linear regression
2. $\alpha = \infty$: All coefficients zero (same logic as before)
3. $0 < \alpha < \infty$: coefficients between 0 and that of simple linear regression

Yes it's appearing to be very similar to Ridge till now. But just hang on with me and you'll know the difference by the time we finish. Like before, let's run lasso regression on the same problem as above.

Notice the additional parameters defined in Lasso function – 'max_iter'. This is the maximum number of iterations for which we want the model to run if it doesn't converge before. This exists for Ridge as well but setting this to a higher than default value was required in this case. Why? I'll come to this in next

section, just keep it in the back of the envelope.

We can observe that **even for a small value of alpha, a significant number of coefficients are zero**. This also explains the horizontal line fit for $\alpha=1$ in the lasso plots, its just a baseline model! This phenomenon of most of the coefficients being zero is called ‘**sparsity**’. Although lasso performs feature selection, this level of sparsity is achieved in special cases only which we’ll discuss towards the end.

This has some really interesting implications on the use cases of lasso regression as compared to that of ridge regression. But before coming to the final comparison, lets take a **bird’s eye view** of the mathematics behind why coefficients are zero in case of lasso but not ridge.

Chapter 4

Machine learning

What are the 7 steps of machine learning?

7 Steps of Machine Learning

- Step 1: Gathering Data. ...
- Step 2: Preparing that Data. ...
- Step 3: Choosing a Model. ...
- Step 4: Training. ...
- Step 5: Evaluation. ...
- Step 6: Hyper parameter Tuning. ...
- Step 7: Prediction.

Introduction:

In this blog, we will discuss the workflow of a Machine learning project this includes all the steps required to build the proper machine learning project from scratch.

We will also go over data pre-processing, data cleaning, feature exploration and feature engineering and show the impact that it has on Machine Learning Model Performance. We will also cover a couple of the pre-modelling steps that can help to improve the model performance.

Python Libraries that would be need to achieve the task:

1. Numpy
2. Pandas
3. Sci-kit Learn
4. Matplotlib

Understanding the machine learning workflow

We can define the machine learning workflow in 3 stages.

1. Gathering data
2. Data pre-processing
3. Researching the model that will be best for the type of data
4. Training and testing the model
5. Evaluation

Okay but first let's start from the basics

What is the machine learning Model?

The machine learning model is nothing but a piece of code; an engineer or data scientist makes it smart through training with data. So, if you give garbage to the model, you will get garbage in return, i.e. the trained model will provide false or wrong prediction

1. Gathering Data

The process of gathering data depends on the type of project we desire to make, if we want to make an ML project that uses real-time data, then we

can build an IoT system that using different sensors data. The data set can be collected from various sources such as a file, database, sensor and many other such sources but the collected data cannot be used directly for performing the analysis process as there might be a lot of missing data, extremely large values, unorganized text data or noisy data. Therefore, to solve this problem Data Preparation is done.

We can also use some free data sets which are present on the internet. Kaggle and UCI Machine learning Repository are the repositories that are used the most for making Machine learning models. Kaggle is one of the most visited websites that is used for practicing machine learning algorithms, they also host competitions in which people can participate and get to test their knowledge of machine learning.

2. Data pre-processing

Data pre-processing is one of the most important steps in machine learning. It is the most important step that helps in building machine learning models more accurately. In machine learning, there is an 80/20 rule. Every data scientist should spend 80% time for data pre-processing and 20% time to actually perform the analysis.

What is data pre-processing?

Data pre-processing is a process of cleaning the raw data i.e. the data is collected in the real world and is converted to a clean data set. In other words, whenever the data is gathered from different sources it is collected in a raw format and this data isn't feasible for the analysis.

Therefore, certain steps are executed to convert the data into a small clean data set, this part of the process is called as data pre-processing.

Why do we need it?

As we know that data pre-processing is a process of cleaning the raw data into clean data, so that can be used to train the model. So, we definitely need data pre-processing to achieve good results from the applied model in machine learning and deep learning projects. Most of the real-world data is messy, some of these types of data are:

1. **Missing data:** Missing data can be found when it is not continuously created or due to technical issues in the application (IOT system).
2. **Noisy data:** This type of data is also called outliers, this can occur due to human errors (human manually gathering the data) or some technical problem of the device at the time of collection of data.
3. **Inconsistent data:** This type of data might be collected due to human errors (mistakes with the name or values) or duplication of data.

Three Types of Data

1. Numeric e.g. income, age
2. Categorical e.g. gender, nationality
3. Ordinal e.g. low/medium/high

How can data pre-processing be performed?

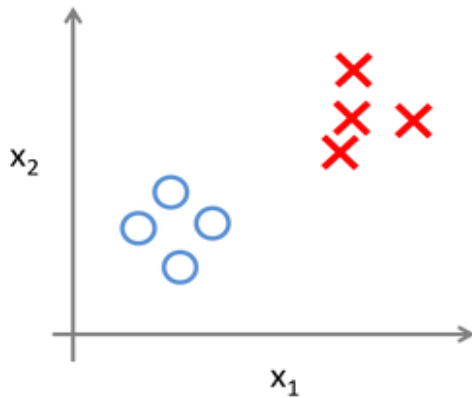
These are some of the basic pre — processing techniques that can be used to convert raw data.

1. **Conversion of data:** As we know that Machine Learning models can only handle numeric features, hence categorical and ordinal data must be somehow converted into numeric features.
2. **Ignoring the missing values:** Whenever we encounter missing data in the data set then we can remove the row or column of data depending on our need. This method is known to be efficient but it shouldn't be performed if there are a lot of missing values in the dataset.
3. **Filling the missing values:** Whenever we encounter missing data in the data set then we can fill the missing data manually, most commonly the mean, median or highest frequency value is used.
4. **Machine learning:** If we have some missing data then we can predict what data shall be present at the empty position by using the existing data.
5. **Outliers detection:** There are some error data that might be present in our data set that deviates drastically from other observations in a data set.
[Example: human weight = 800 Kg; due to mistyping of extra 0]

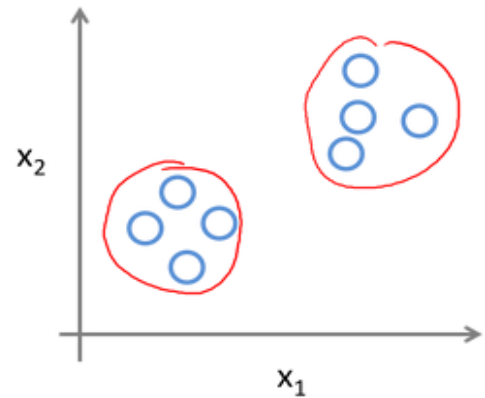
Researching the model that will be best for the type of data

Our main goal is to train the best performing model possible, using the pre-processed data.

Supervised Learning



Unsupervised Learning



Supervised Learning:

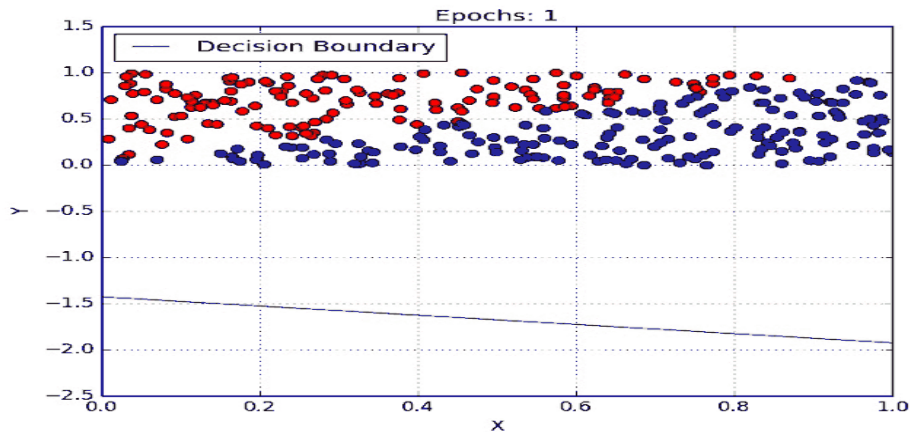
In Supervised learning, an AI system is presented with data which is labelled, which means that each data is tagged with the correct label.

The supervised learning is categorized into 2 other categories which are “**Classification**” and “**Regression**”.

Classification:

Classification problem is when the target variable is **categorical** (i.e. the output could be classified into classes — it belongs to either Class A or B or something else).

A classification problem is when the output variable is a category, such as “red” or “blue”, “disease” or “no disease” or “spam” or “not spam”.



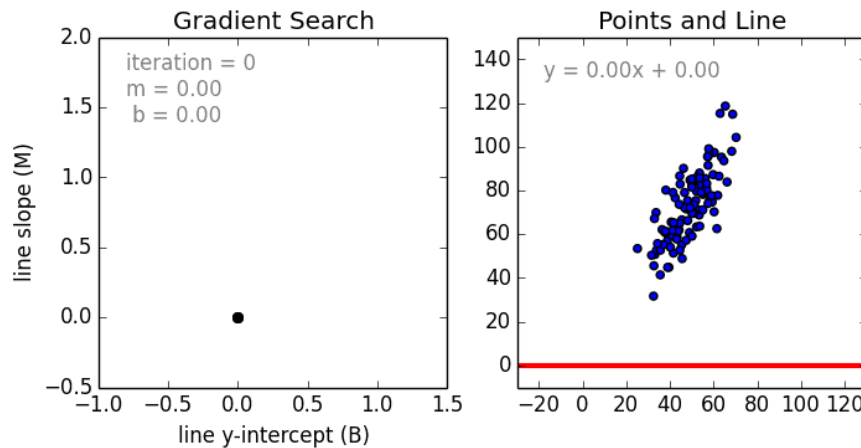
As shown in the above representation, we have 2 classes which are plotted on the graph i.e. red and blue which can be represented as ‘setosa flower’ and ‘versicolor flower’, we can image the X-axis as there ‘Sepal Width’ and the Y-axis as the ‘Sepal Length’, so we try to create the best fit line that separates both classes of flowers.

These some most used classification algorithms.

- **K-Nearest Neighbor**
- **Naive Bayes**
- **Decision Trees/Random Forest**
- **Support Vector Machine**
- **Logistic Regression**

Regression:

While a **Regression** problem is when the target variable is **continuous** (i.e. the output is numeric).



As shown in the above representation, we can imagine that the graph's X-axis is the 'Test scores' and the Y-axis represents 'IQ'. So we try to create the best fit line in the given graph so that we can use that line to predict any approximate IQ that isn't present in the given data.

These some most used regression algorithms.

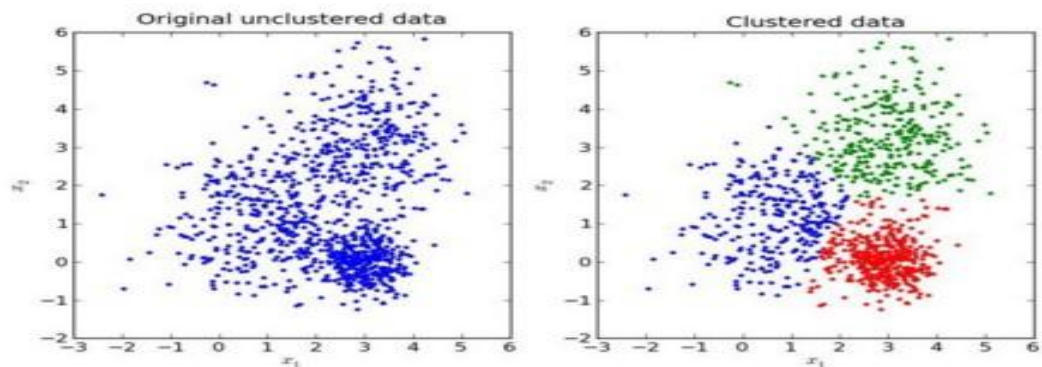
- **Linear Regression**
- **Support Vector Regression**
- **Decision Tress/Random Forest**
- **Gaussian Progresses Regression**
- **Ensemble Methods**

Unsupervised Learning:

The unsupervised learning is categorized into 2 other categories which are "Clustering" and "Association".

Clustering:

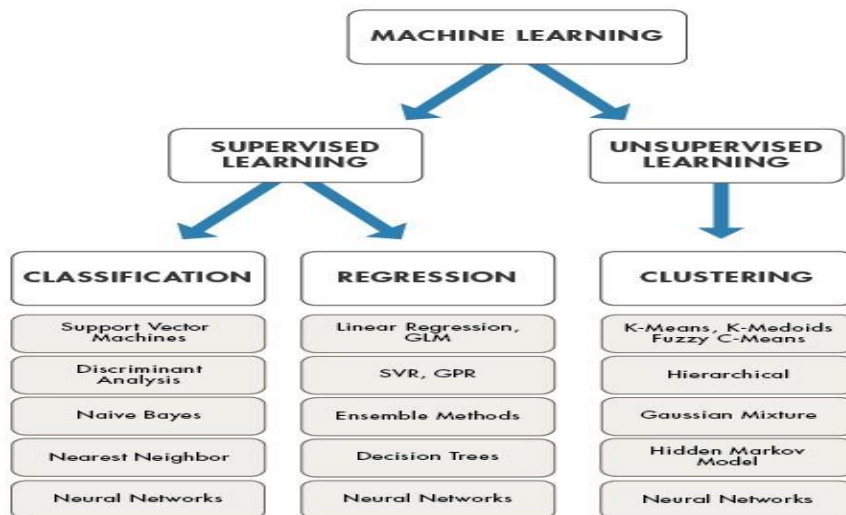
A set of inputs is to be divided into groups. Unlike in classification, the groups are not known beforehand, making this typically an unsupervised task.



Methods used for clustering are:

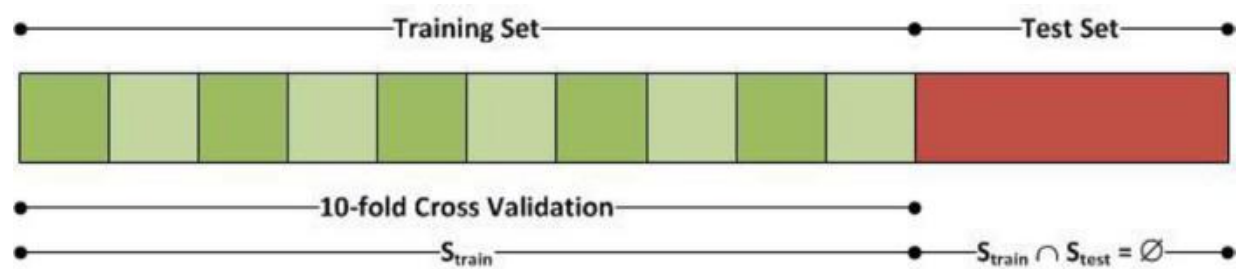
- **Gaussian mixtures**
- **K-Means Clustering**
- **Boosting**
- **Hierarchical Clustering**
- **K-Means Clustering**
- **Spectral Clustering**

Overview of models under categories:



4. Training and testing the model on data

For training a model we initially split the model into 3 three sections which are ‘**Training data**’ , ‘**Validation data**’ and ‘**Testing data**’. You train the classifier using ‘**training data set**’, tune the parameters using ‘**validation set**’ and then test the performance of your classifier on unseen ‘**test data set**’. An important point to note is that during training the classifier only the training and/or validation set is available. The test data set must not be used during training the classifier. The test set will only be available during testing the classifier.



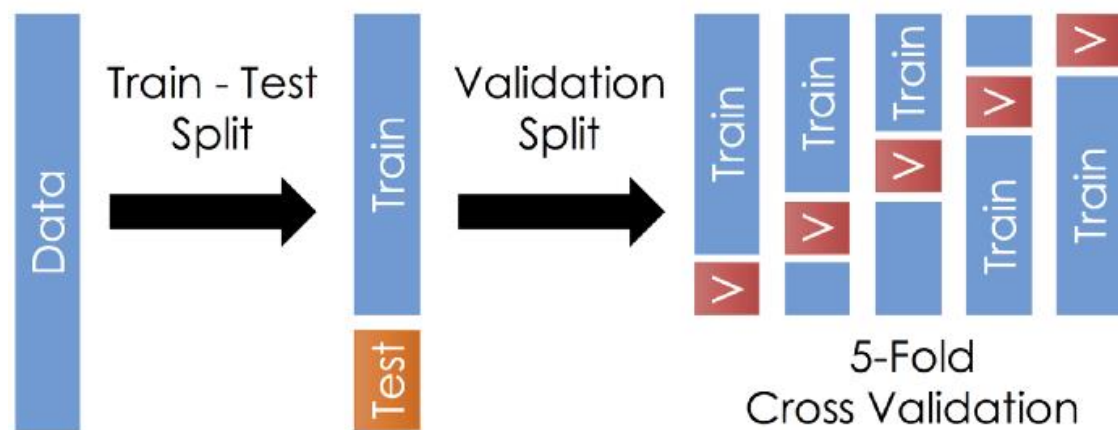
Training set:

The training set is the material through which the computer learns how to process information. Machine learning uses algorithms to perform the

training part. A set of data used for learning, that is to fit the parameters of the classifier.

Validation set:

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. A set of unseen data is used from the training data to tune the parameters of a classifier.



Once the data is divided into the 3 given segments we can start the training process.

In a data set, a training set is implemented to build up a model, while a test (or validation) set is to validate the model built. Data points in the training set are excluded from the test (validation) set. Usually, a data set is divided into a training set, a validation set (some people use 'test set' instead) in each iteration, or divided into a training set, a validation set and a test set in each iteration.

The model uses any one of the models that we had chosen in step 3/ point 3. Once the model is trained we can use the same trained model to predict

using the testing data i.e. the unseen data. Once this is done we can develop a confusion matrix, this tells us how well our model is trained. A confusion matrix has 4 parameters, which are '**True positives**', '**True Negatives**', '**False Positives**' and '**False Negative**'. We prefer that we get more values in the True negatives and true positives to get a more accurate model. The size of the Confusion matrix completely depends upon the number of classes.

n=165	Predicted: NO	Predicted: YES
	Actual: NO	Actual: YES
	50	10
	5	100

True positives : These are cases in which we predicted TRUE and our predicted output is correct.

True negatives : We predicted FALSE and our predicted output is correct.

False positives : We predicted TRUE, but the actual predicted output is FALSE.

False negatives : We predicted FALSE, but the actual predicted output is TRUE.

We can also find out the accuracy of the model using the confusion matrix.

Accuracy = (True Positives + True Negatives) / (Total number of classes)

i.e. for the above example:

$$\text{Accuracy} = (100 + 50) / 165 = 0.9090 \text{ (90.9\% accuracy)}$$

5. Evaluation

Model Evaluation is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future.

To improve the model we might tune the hyper-parameters of the model and try to improve the accuracy and also looking at the confusion matrix to try to increase the number of true positives and true negatives.

Conclusion:

In this blog, we have discussed the workflow a Machine learning project and gives us a basic idea of how a should the problem be tackled.

Chapter 5 SYSTEM DESIGN

Modules:

- Dataset collection
- Data Cleaning
- Feature Extraction
- Data training
- Data testing
- Performance Evaluation
- Prediction

Dataset collection:

- Collecting data allows you to capture a record of past events so that we can use data analysis to find recurring patterns. From those patterns, you build predictive models using machine learning algorithms that look for trends and predict future changes.
- Predictive models are only as good as the data from which they are built, so good data collection practices are crucial to developing high-performing models.
- The data need to be error-free (garbage in, garbage out) and contain relevant information for the task at hand. For example, a loan default model would not benefit from tiger population sizes but could benefit from gas prices over time.
- In this module, we collect the data from kaggle dataset archives. This dataset contains the information of divorce in previous years.

Data Cleaning:

- Data cleaning is a critically important step in any machine learning project.
- In this module data cleaning is done to prepare the data for analysis by removing or modifying the data that may be incorrect, incomplete, duplicated or improperly formatted.
- In tabular data, there are many different statistical analysis and data visualization techniques you can use to explore your data in order to identify data cleaning operations you may want to perform

Feature Extraction:

- This is done to reduce the number of attributes in the dataset hence providing advantages like speeding up the training and accuracy improvements.
- In machine learning, pattern recognition, and image processing, feature extraction starts from an initial set of measured data and builds derived values (features) intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps, and in some cases leading to better human interpretations. Feature extraction is related to dimensionality reduction

- When the input data to an algorithm is too large to be processed and it is suspected to be redundant (e.g. the same measurement in both feet and meters, or the repetitiveness of images presented as pixels), then it can be transformed into a reduced set of features (also named a feature vector).
- Determining a subset of the initial features is called feature selection. The selected features are expected to contain the relevant information from the input data, so that the desired task can be performed by using this reduced representation instead of the complete initial data.
- A training model is a dataset that is used to train an ML algorithm. It consists of the sample output data and the corresponding sets of input data that have an influence on the output.
- The training model is used to run the input data through the algorithm to correlate the processed output against the sample output. The result from this correlation is used to modify the model.
- This iterative process is called “model fitting”. The accuracy of the training dataset or the validation dataset is critical for the precision of the model.
- Model training in machine language is the process of feeding an ML algorithm with data to help identify and learn good values for all attributes involved.
- There are several types of machine learning models, of which the most common ones are supervised and unsupervised learning.
- In this module we use supervised classification algorithms like linear regression to train the model on the cleaned dataset after dimensionality reduction.

Testing model:

- In this module we test the trained machine learning model using the test dataset
- Quality assurance is required to make sure that the software system works according to the requirements. Were all the features implemented as agreed? Does the program behave as expected? All the parameters that you test the program against should be stated in the technical specification document.
- Moreover, software testing has the power to point out all the defects and flaws during development. You don’t want your clients to encounter bugs after the software is released and come to you waving their fists. Different kinds of testing allow us to catch bugs that are visible only during runtime.

Performance Evaluation:

- In this module, we evaluate the performance of trained machine learning model using performance evaluation criteria such as F1 score, accuracy and classification error.
- In case the model performs poorly, we optimize the machine learning algorithms to improve the performance.
- Performance Evaluation is defined as a formal and productive procedure to measure an employee's work and results based on their job responsibilities. It is used to gauge the amount of value added by an employee in terms of increased business revenue, in comparison to industry standards and overall employee return on investment (ROI).
- All organizations that have learned the art of "winning from within" by focusing inward towards their employees, rely on a systematic performance evaluation process to measure and evaluate employee performance regularly.
- Ideally, employees are graded annually on their work anniversaries based on which they are either promoted or are given suitable distribution of salary raises
- Performance evaluation also plays a direct role in providing periodic feedback to employees, such that they are more self-aware in terms of their performance metrics.

Prediction:

- "Prediction" refers to the output of an algorithm after it has been trained on a historical dataset and applied to new data when forecasting the likelihood of a particular outcome, such as whether or not a customer will churn in 30 days.
- The algorithm will generate probable values for an unknown variable for each record in the new data, allowing the model builder to identify what that value will most likely be.
- The word "prediction" can be misleading. In some cases, it really does mean that you are predicting a future outcome, such as when you're using machine learning to determine the next best action in a marketing campaign.
- Other times, though, the "prediction" has to do with, for example, whether or not a transaction that already occurred was fraudulent.
- In that case, the transaction already happened, but you're making an educated guess about whether or not it was legitimate, allowing you to take the appropriate action.
- In this module we use trained and optimized machine learning model to predict whether the house price.

Chapter 6

Conclusions:

This paper examined and analyzed the current research on the significant attributes of house price and analyzed the data mining techniques used to predict house price. Technically, houses with a strategic location such as the accessibility to shopping mall or other facilities tend to be more expensive than houses in rural areas with limited numbers of facilities. The accurate prediction model would allow investors or house buyers to determine the realistic price of a house as well as the house developers to decide the affordable house price. This paper addressed the attributes used by previous researchers to forecast a house price using various prediction models. Taken together, the results of the survey have shown the potential of SVR, ANN and XGBoost in predicting house prices. These models were developed based on several input attributes and they work significantly positive with house price. In conclusion, the impact of this research was intended to help and assist other researchers in developing a real model which can easily and accurately predict house prices. Further work on a real model needs to be done with the utilization of our findings to confirm them.

References:

- [1] A. S. Temür, M. Akgün, and G. Temür, "Predicting Housing Sales in Turkey Using Arima, Lstm and Hybrid Models," *J. Bus. Econ. Manag.*, vol. 20, no. 5, pp. 920–938, 2019, doi: 10.3846/jbem.2019.10190.
- [2] A. Ebekozien, A. R. Abdul-Aziz, and M. Jaafar, "Housing finance inaccessibility for low-income earners in Malaysia: Factors and solutions," *Habitat Int.*, vol. 87, no. April, pp. 27–35, 2019, doi: 10.1016/j.habitatint.2019.03.009.
- [3] A. Jafari and R. Akhavian, "Driving forces for the US residential housing price: a predictive analysis," *Built Environ. Proj. Asset Manag.*, vol. 9, no. 4, pp. 515–529, 2019, doi: 10.1108/BEPAM-07-2018-0100.

- [4] Choong Wei Cheng, "Statistical Analysis of Housing Prices in Petaling," Universiti Tunku Abdul Rahman, 2018. House Price Prediction using a Machine Learning Model: A Survey of Literature
- [5] R. E. Febrita, A. N. Alfiyatin, H. Taufiq, and W. F. Mahmudy, "Data-driven fuzzy rule extraction for housing price prediction in Malang, East Java," *2017 Int. Conf. Adv. Comput. Sci. Inf. Syst. ICACSYS 2017*, vol. 2018-Janua, pp. 351–358, 2018, doi: 10.1109/ICACSYS.2017.8355058.
- [6] G. Gao *et al.*, "Location-Centered House Price Prediction: A Multi-Task Learning Approach," pp. 1–14, 2019, [Online]. Available: <http://arxiv.org/abs/1901.01774>.
- [7] T. D. Phan, "Housing price prediction using machine learning algorithms: The case of Melbourne city, Australia," *Proc. - Int. Conf. Mach. Learn. Data Eng. iCMLDE 2018*, pp. 8–13, 2019, doi: 10.1109/iCMLDE.2018.00017.
- [8] Y. Y. S. Song, T. Zhou, H. Yachi, and S. Gao, "Forecasting house price index of China using dendritic neuron model," *PIC 2016 - Proc. 2016 IEEE Int. Conf. Prog. Informatics Comput.*, pp. 37–41, 2017, doi: 10.1109/PIC.2016.7949463.
- [9] R. Aswin Rahadi, S. K. Wiryono, D. P. Koesrindartoto, and I. B. Syamwil, "Factors Affecting Housing Products Price in Jakarta Metropolitan Region," *Int. J. Prop. Sci.*, vol. 6, no. 1, pp. 1–21, 2016, doi: 10.22452/ijps.vol6no1.2.
- [10] A. Nur, R. Ema, H. Taufiq, and W. Firdaus, "Modeling House Price Prediction using Regression Analysis and Particle Swarm Optimization Case Study : Malang, East Java, Indonesia," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 10, pp. 323–326, 2017, doi: 10.14569/ijacsa.2017.081042.
- [11] A. Yusof and S. Ismail, "Multiple Regressions in Analysing House Price Variations," *Commun. IBIMA*, vol. 2012, pp. 1–9, 2012, doi: 10.5171/2012.383101.
- [12] A. Osmadi, E. M. Kamal, H. Hassan, and H. A. Fattah, "Exploring the elements of housing price in Malaysia," *Asian Soc. Sci.*, vol. 11, no. 24, pp. 26–38, 2015, doi: 10.5539/ass.v11n24p26.
- [13] T. L. Chin and K. W. Chau, "A critical review of literature on the hedonic price model," *Int. J. Hous. Sci. Its Appl.*, vol. 27, no. 2, pp. 145–165, 2003.
- [14] M. J. Ball, "Recent Empirical Work on the Determinants of Relative House Prices," *Urban Stud.*, vol. 10, no. 2, pp. 213–233, 1973, doi: 10.1080/00420987320080311.
- [15] M. Rodriguez, "Managing Corporate Real Estate: Evidence from the Capital Markets." *Journal of Real Estate Literature*, 1996.

- [16] D. G. Owusu-Manu, D. J. Edwards, K. A. Donkor-Hyiaman, R. O. Asiedu, M. R. Hosseini, and E. Obiri-Yeboah, "Housing attributes and relative house prices in Ghana," *Int. J. Build. Pathol. Adapt.*, vol. 37, no. 5, pp. 733–746, 2019, doi: 10.1108/IJBPA-01-2019-0003.
- [17] D.-G. Owusu-Manu, D. J. Edwards, K. A. Donkor-Hyiaman, R. O. Asiedu, M. R. Hosseini, and E. Obiri-Yeboah, "Housing attributes and relative house prices in Ghana," *Int. J. Hous. Mark. Anal.*, vol. 8, no. 2, p. 1998, 2018, doi: 10.1017/CBO9781107415324.004.
- [18] J. M. Montero, R. Mínguez, and G. Fernández-Avilés, "Housing price prediction: parametric versus semi-parametric spatial hedonic models," *J. Geogr. Syst.*, vol. 20, no. 1, pp. 27–55, 2018, doi: 10.1007/s10109-017-0257-y.
- [19] S. Lu, Z. Li, Z. Qin, X. Yang, and R. S. M. Goh, "A hybrid regression technique for house prices prediction," *IEEE Int. Conf. Ind. Eng. Eng. Manag.*, vol. 2017-Decem, pp. 319–323, 2018, doi: 10.1109/IEEM.2017.8289904.
- [20] E. Pagourtzi, V. Assimakopoulos, T. Hatzichristos, and N. French, "Real estate appraisal: A review of valuation methods," *J. Prop. Invest. Financ.*, vol. 21, no. 4, pp. 383–401, 2003, doi: 10.1108/14635780310483656.