LAB REPORT

Submitted by

MODEM ANURADHA SAI SHAKTI
[RA2011003010545]

Under the Guidance of

Ms. G.K. Sandhia

Assistant Professor (Sr. G), Department of Computing Technologies

In partial satisfaction of the requirements for the degree of

BACHELOR OF TECHNOLOGY
in

COMPUTER SCIENCE ENGINEERING

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

JUNE 2022

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR-603203

BONAFIDE CERTIFICATE

Certified that this lab report titled **"HOUSE PRICE PREDICTION USING MACHINE LEARNING AND PYTHON"** is the bonafide work done by MODEM ANURADHA SAI SHAKTI(RA2011003010545) who carried out the lab exercises under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other work.

SIGNATURE

Ms. G.K. Sandhia

SEPM – Course Faculty

Assistant Professor (Sr. G)

Department of Computing Technologies

ABSTRACT

House price prediction is an important concept in the real estate industry. Thus, many researchers from different fields are interested in developing a regression model for the house price to obtain an accurate prediction and explore the factors affecting the house price. In this study, we aim to develop an accurate regression model using tree-based algorithms and explain the type of information which has an impact on the house price. For this purpose, we use the Ames and lowa House Price dataset. Index Terms—House Price Prediction, Lasso, Random Forest, XGBoost, LightGBM, Gradient Boosting, CatBoost. House price prediction has been a popular problem in research for years since the traditional house price prediction depending on cost and sale price comparison does not satisfy the accepted standards and certification process. An accurate house price prediction has importance for the stakeholders of the real estate industry, which is a constantly rising one in many countries, like homeowners, customers, and estate agents. In addition to getting accurate predictions, it is important to know the factors that have a significant impact on the house price. Real estate is the least transparent industry in our ecosystem. Housing prices keep changing day in and day out and sometimes are hyped rather than being based on valuation. Predicting housing prices with real factors is the main crux of our research project. Here we aim to make our evaluations based on every basic parameter that is considered while determining the price. We use various regression techniques in this pathway, and our results are not sole determination of one technique rather it is the weighted mean of various techniques to give most accurate results.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

API	APPLICATION PROGRAMMING INTERFACE
HTML	HYPER TEXT MARKUP LANGUAGE
CSS	CASCADING STYLES SHEET
JS	JAVASCRIPT
RAM	RANDOM ACCESS MEMORY
UI	USER INTERFACE
UX	USER EXPERIENCE
IDE	INTEGRATED DEVELOPMENT ENVIRONMENT
WBS	WORK BREAKDOWN STRUCTURE
SWOT	STRENGTH, WEAKNESS, OPPURTUNITIES AND THREATS
RMM	RISK MONITERING AND MANAGEMENT
UML	UNIFIED MODELING LANGUAGE
OOP	OBJECT ORIENTED PROGRAMMING
ER	ENTITY RELATION
DFD	DATA FLOW DIAGRAM
HTTP	HYPER TEXT TRANSFER PROTOCOL
SVM	SUPPORT VECTOR MACHINE
CRUD	CREATE, READ, UPDATE, DELETE
MLM	MULTI LEVEL MODEL
DB	DATABASE
CLI	COMMAND LINE INTERFACE
SQL	STRUCTURED QUERY LANGUAGE
ANN	ARTIFICAL NETWORK MODEL
HPM	HEDONIC PRICE MODEL
HPI	HOUSE PRICE INDEX
MLR	MULTIPLE LINEAR REGRESSION
OLS	ORDINARY LEAST SQUARES

Problem statement:
HOUSE PRICE PREDICTION USING MACHINE LEARNING AND PYTHON

Problem Statement Using Machine Learning Prices of real estate properties are sophisticatedly linked with our economy. Despite this, we do not have accurate measures of housing prices based on the vast amount of data available. Therefore, the goal of this project is to use machine learning to predict the selling prices of houses based on many economic factors. A systematic method can be built to derive a layered knowledge graph and design a structured Deep Neural Network based on it. Neurons in a structured DNN are structurally connected, which makes the network time and space efficient; and thus, it requires fewer data points for training. The structured DNN model has been designed to learn from the most recently captured data points which allows the model to adapt to the latest market trends. To demonstrate the effectiveness of the proposed approach, we can use a case study of assessing real properties in small towns. Previous work on predicting house prices has been based on regression analysis and machine learning techniques. Local linear models and random forest models, fuzzy reasoning, Backpropagation neural networks and Elman neural network can been used to forecast real estate prices. Out of these, it is found that Elman neural network can forecast more accurately and constringe faster than other approaches. Nguyen and Cripps compared the predictive performance of artificial neuralnetworks (ANNs) and multiple regression analysis for single family housing sales. They found that when enough data points were available for training, ANNs could perform better than multiple linear regressions. Additionally, a latent manifold model with two trainable components can also be used to evaluate house prices where a parametric component is used for predicting the "intrinsic" price of a house, and a nonparametric component can calculate the desirability of the neighborhood. Unlike the existing approaches, we propose to use a deep learning approach with structured DNNs, which may outperform the conventional tools for real estate assessments.

Housing prices are an important reflection of the economy, and housing price ranges are of great interest for both buyers and sellers . Ask a home buyer to describe their dream house, and they probably won't begin with the height of the basement ceiling or the proximity to an east-west railroad. But this playground competition's data-set proves that much more influences price negotiations than the number of bedrooms or a white-picket fence.

With **79** explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, this competition challenges you to predict the final price of each home.

LOGO:

Objectives:

1. Predict the sale price for each house.

2. Minimize the difference between predicted and actual rating (RMSE/MSE)

Constraints:

1. Some form of interpretability.

im

To identify the appropriate Process Model for the project and prepare Stakeholder and User Description.

Project Title:

HOUSE PRICE PREDICTION USING MACHINE LEARNING AND PYTHON

1. Executive

Summary1.1 Project

Methodology:

Agile Methodology is a people-focused, results-focused approach to software development thatrespects our rapidly changing world.
It's centred around adaptive planning, self-organization, and short delivery times.
It's flexible, fast, and aims for continuous improvements in quality, to put in simple terms, Agile helps teams in delivering value to customers quickly and effortlessly.
Thus, developing such software management project, Agile methodology brings out the most effective growth out of it.

1.2 Stakeholder Identification:

Internal stakeholders:

The internal stakeholders include the team members, managers, executives who are all internally related.

Project Role	Responsibilities	Team members assigned to
Project Manager	To guide the team and give tasks tothe team	PULIPATI SRINIVAS BHARGAV
Technical Lead	To do all technical tasks	MODEM ANURADHA SAI SHAKTI
Business Analyst	to calculate budget of the project	AKKIPALLI ROHITH
Developer	To develop the application	MODEM ANURADHA SAI SHAKTI
Tester	To test the application finding the errors and all	PULIPATI SRINIVAS BHARGAV

External stakeholders:

Project Role	Responsibilities	Team members assigned to
Suppliers	To all the data regarding house(number of bedrooms size of the house)	MODEM ANURADHA SAISHAKTI
customers(houses buyers)	the one is who is buying the house	A.ROHITH
Government	government can get to know about the house price ratio	SRINIVAS BHARGAV
share holders	house owners	M.ANURADHA

2. Stakeholder Management

Interest and Influence matrix

Interest	Influence
High	High
Low	Low
Low	High
High	Low

STAKEHOLDER INTEREST, INFLUENCE, PRIORITY IDENTIFICATION

Stakeholder	Responsibility	Interest	Influence	Estimated Priority
Owner	achieve the goals and increase the price of the house or sales	high	high	1
Sponsor	provides the new market	med	low	3
Team members	to develop the application	high	high	2
Project Manager	to lead the team in every aspect	high	high	8
Investors	provides financial reources	low	low	5
Resource Manager	resource palnning and source allocation	med	low	4
Suppliers	ensuring feasible and realistic in every	med	high	6

	aspect			
End Users	provides the feedback	low	low	7

Reference: Data set of house price prediction

Project Title: HOUSE PRICE PREDICTION USING MACHINE LEARNING AND PYTHON

User Requirements:

Consider a person wants to buy a house in Ames and Iowa then he wants to know about the price of the house and how many bedrooms and interior etc and many.. this project HOUSE PRICE PREDICTION USING MACHINE LEARNING AND PYTHON helps the users such as house buyers, sellers, real estate market investors, etc are the users of this project. this helps the users to predict the price and check whether the price is in his budget range? and for the real estate market investors can know the range and current scenario of the market

and house sellers can the house when the price is high so all users get beneficial

so the user's requirement is accurate he prediction so he can have a clear cut idea about the price and his time gets saved.

System Requirements:

an aspect of what the proposed system must do

to do this project (HOUSE PREDICTION USING MACHINE LEARNING) all requires

SR1. Sample dataset

SR2. Python(IDE)

SR3. machine learning concepts

SR4. python knowledge

SR5. jupyter notebook

SR6. Libraries - pandas, numpy etc..

Functional Requirements:

what is functional requirements?

"***Any Requirement Which Specifies What The System Should Do.***"

Some of the more typical functional requirements include:

FR1. Historical Data

FR2. Maintainability

FR3. Reliability

FR4. Scalability

FR5. Performance

FR6. Reusability

FR7.Flexibility

this project:

FR1. this roject provides accurate prediction about house price in all aspects with usedsample dataset.

FR2. machine learning advanced regression is used to predict which one of the benefit.

FR3. Applying exploratory data analysis and trying to get some insights about ourdataset.

FR4. This project will help the sellers and buyers to have an overview of the situationso that they can act accordingly.

FR5. We apply Advanced regression techniques like Random Forest and gradient boosting models for prediction of the percentage change of the housing prices.

FR6. The data set includes a large number of variables describing almost every aspectof the properties that might be of interest when evaluating a house.

Non-Functional Requirements:

What are non-functional requirements?

"*Any Requirement That Specifies How The System Performs A Certain Function.*"

Some typical non-functional requirements are:

Non-Functional Testing like Performance, Stress, Usability, Security testing, etc are done.

NFR1. Environmental
NFR2. Regulatory
NFR3.Compatibility
NFR4. Usability

this project:

NFR1: this project or application(HOUSE PREDICTION USING MACHINE LEARNING) is used to predict the house predict at particular location(here it is ames and lowa) based on the given dataset.

NFR2:this application supports 79 various variables(no of bed rooms , interior , locality, weather) which will already be mentioned in the the dataset.

NFR3: This model has learned the training data so well that it cannot generalize new data points. This should be avoided.

Aim

To Prepare Project Plan based on scope, Calculate Project effort based on resources, Find

Job roles and responsibilities

project plan template:

Project plan template

Title of the project: HOUSE PRICE PREDICTION USING PYTHON AND MACHINE LEARNING

The overall goal of project planning is to establish a pragmatic strategy forcontrolling, tracking and monitoring a complex technical project.

Abstract:

Real estate is the least transparent industry in our ecosystem.The real estate market is one of the most competitive in terms of prices and it tends to vary significantly based on a lot of factors, hence it becomes one of the prime fields to apply the concepts of machine learning. Therefore in this project, we present various algorithms while predicting house prices with good accuracy. We tested a regression models such as Simple Linear Regression, Ridge Regression, Lasso Regression, Support Vector Regression, Random Forest Regression, Decision Tree Algorithm and selected the best fit among the algorithm. This project directs us that it can be best application of machine learning models in order to optimize the result. It will help clients to put resources into a web-based application without moving toward a broker. It also provides a brief about various graphical and numerical techniques which will be required to predict the price of a house.

Keywords - House prediction: regression analysis

Project Description:

In this project we will create a machine learning model with advanced regression and random forest and gradient boosting model dataset. This model will aid us in making better real estate decisions by making house price predictions in that area. For this we will carry out data exploration for better understanding of data and will require preprocessing to improve the model accuracy.

Project Governance and Project Team Structure

Collaborator(s):

- PULIPATI SRINIVAS BHARGAV - Team lead of the project
- MODEM ANURADHA SAI SHAKTI - Team member of the project
- AKKIPALLI ROHITH - Team member of project

Project Language(s):

- Python

Difficulty:

- Beginner

Duration:

- 15 Hours

Prerequisite(s):

- Python, Basic statistics

Skills to be learned:

- Data Visualization, Basic Data Preprocessing, Model Implementation

Overview:

Objective:

Create a machine learning model using linear regression and Boston housing dataset while following the machine learning workflow.

Project Context:

In machine learning we write computer programs which automatically improve with experience which are termed as machine learning models. It saves us from explicitly writing code for complex real world data. In this project we are going to use supervised learning, which is a branch of machine learning where we teach our model by examples. Here we will first explore different attributes of Boston housing dataset then a part of dataset will be used to train the advanced regression algorithm after that we will use the trained model to give predictions on remaining part of dataset.

Project Stages:

This project(HOUSE PRICE PREDICTION USING PYTHON AND MACHINE LEARNING)consists of the following stages:

- IMPORTING LIBRARIES AND DATASET
- EXPLORING AND PREPROCESSING THE DATASET
- MODEL IMPLEMENTATION
- MODEL TESTING

High-Level Approach:

• Exploring and analyzing the data used for making prediction

• Creating a simple model using linear regression

• Using the model to carryout prediction and evaluating it's efficiency

Requirements Management:

- Python ide or google collab
- Simple dataset
- knowledge of machine learning and python
- Jupyter books
    - Jupyter notebooks are popular among data scientist because they are easy to follow and show your working steps.

Integration Management:

Project structure& roles and responsibilities:

Srinivas bhargav – team lead

Anuradha sai shakti - team member

Rohith - team member

Scope management:

Project scope management is a process that helps in determining and documenting the list of all the project goals, tasks, deliverables, deadlines, and budgets as a part of the planning

process. In project management, it is common for a big project to have modifications along the way.

In this house prediction using machine learning and python:
The House Prediction dataset is imported from Kaggle in Comma Separated Values (csv) format. The dataset is analyzed with the help of pandas, numpy and scikit-learn. Tableau is used as a data visualization tool. After drawing insights from the dataset with the help of Tableau, we identify the important factors i.e. factors majorly affecting the change in prices. The factors adding insignificant values to the overall result are omitted. The dataset is divided into two parts - training set and testing set.

deliverable :

A deliverable is any product, service, or result that must be completed to finish a project.

Cost management:

As no hardware elements are required in this system, the constraints are very few. So, the cost of the requirements for the project are very less. Machine learning algorithms require systems with high processing power i.e. systems with high Random Access Memory (RAM) are required For the smooth functioning of all the machine learning models, Anaconda - Python data science platform is installed. Python libraries - Numpy, Pandas and seaborn should be installed in the system. Tableau - Data visualization tool is also to be installed on the system.

Estimated effort: requires little more effect as no hardware tools are used all are software related.

Stakeholders:

Stakeholders for this kind of project will be:

- Customers and Real Estate Agents :

  The real estate industry has long operated according to its own traditions, but the availability of huge volumes of data is revolutionizing the way the industry works,

- Companies :

such as Zillow and Trulia can use this analysis to calculate an estimated value of the price that the home might attract based on factors like local schools, crime rates, income level, hospitals etc. and decide marketing strategy.

- Banks :

It's not just consumers who are using big data to inform their house buying and selling decisions. Banks are also drawing on vast pools of data to predict the risk that a particular mortgage application could pose, using information about both the value of the home and the applicant's financial situation and history. In addition, banks are also avoiding losing out on foreclosure and short sales, as big data is helping them to predict the maximum sale value that the market can bear.

Based on house prices predicted one can invest in real estate, find a county house better suited for their needs where they can buy a house. House buying and selling decision would become easier with the prediction done by this data science project.

Risk management:

In most developed countries, property is the single most important asset of private households. For example, in Germany, property makes up 44% of the total asset volume held by private households and in the US and Australia, real estate represents around half of total private wealth. The risk management infrastructure to deal with house price risk is often underdeveloped. Only limited financial instruments are available to diversify house price risk. The literature quotes a number of reasons for the lack of risk management

opportunities (for example, Neukirchen, 2005) ranging from the high complexity and volume of the risk transfer to missing risk awareness among the risk bearers.

- STRATEGIES TO MANAGE HOUSE PRICE RISK:
- Risk avoidance:

Risk avoidance is the first house price risk management strategy (Rejda, 1998). Risk avoidance implies that all the risk exposure is eliminated in the first place. Realization of this strategy means that the property is sold. Risk avoidance is the most basic risk management approach. However, this strategy is not discussed extensively in this paper, as it does not require any risk management.

- Risk retention
- Home value insurance
- Home equity conversions

- ❖ Satisfaction of customers by expanding the exactness of their decision and diminishing the danger of putting resources into a home. The sales prices will be calculated with better accuracy and precision. The system will satisfy customers by providing accurate output and preventing the risk of investing in the wrong house. That would make it even easier for the people to select the houses that best suits their budgets.
- ❖ So we conclude that the system that we proposed solves most of the problemsthat we have with the existing system. After training and testing of datasets

with all models, the random forest classifier and ridge classifier models

performs better than the simple linear regression model. The highest accuracy score is achieved by the Random Forest Classifier. So, we suggest that this regression model be used for future house price predictions. Therefore, the outcome of our project will be predicting house prices with good accuracy which can help the customer as well as developer.

1. Project Management Plan

Describe the key issues driving the project. **[Min 3 Focus Areas]**

Focus Area	Details
Integration Management	Governance FrameworkProject Team Structure Roles & Responsibilities of Team Change Management (Change Control, Issue Management)Project Closure
Scope Management	Scope Statement Requirement Management (Gathering, Control, Assumption,Constraint Stakeholder) Define Deliverable Requirement Change ControlActivities and Sub- Tasks
Schedule Management	Define Milestones Schedule Control
Cost Management	Estimate EffortAssign Team Budget Control
Quality Management	Quality Assurance: Quality assurance will be managed including governance, roles and responsibilities, tools and techniques and reporting Quality Control: Specify the mechanisms to be used to measure and control the quality of the work products
Resource Management	Estimate and Manage the need People: People & Skills Required Finance: Budget Required Physical: Facilities, IT Infrastructure
Stakeholder	Identifying, Analyzing, Engaging Stakeholders
Communication Management	Determine communication requirements, roles and responsibilities, tools and techniques. [Type of Communication, Schedule, Mechanism Recipient]
Risk Management	Identifying, analysing, and prioritizing project risks

Procurement Management	Adhering to organization procurement process

2. Estimation

Effort and Cost Estimation

Activity Description	Sub-Task	Sub-Task Description	Effort (in hours)	Cost in INR
Design the user screen	E1R1A1T1 (Effort-Requirement-Activity-Task)	Confirm the user requirements (acceptance criteria)	3	1500/-
	E1R1A1T2	user requirements	4	2500/-
	E1R1A1T3	customers expectations	5	3000/-
Identify Data Source for displaying units of Energy Consumption		Go through Interface contract (ApplicationData Exchange) documents	5	2500/-
		Document	2	1000/-

Effort (hr)	Cost (INR)
1	500

Infrastructure/Resource Cost [CapEx]

< OneTime Infra requirements >

Infrastructure Requirement	Qty	Cost per qty	Cost per item
IR1	3	3000/-	1000/-
IR2	4	1000/-	250/-
IR3	2	5000/-	2500/-

Maintenance and Support Cost [OpEx]

Category	Details	Qty	Cost per qty per annum	Cost per item
People	Network, System, Middleware and DB admin Developer , Support Consultant	3	2,000,000	6,000,000

License	Operating System Database Middleware IDE	10	10000	100,000
Infrastructures	Server, Storage and Network	20	20000	400,000

3. Project Team Formation

Identification Team members

Name	Role	Responsibilities
ANURADHA	Key Business User (Product Owner)	Provide clear business and user requirements
SRINIVAS BHARGAV	Project Manager	Manage the project
AKKIPALLI ROHITH	Business Analyst	Discuss and Document Requirements
ANURADHA	Technical Lead	Design the end-to-end architecture
ANURADHA	UX Designer	Design the user experience
ANURADHA	Frontend Developer	Develop user interface
SRINIVAS BHARGAV	Backend Developer	Design, Develop and Unit Test Services/API/DB
ROHITH	Cloud Architect	Design the cost effective, highly available and scalable architecture
BHARGAV	Cloud Operations	Provision required Services
ANURADHA	Tester	Define Test Cases and Perform Testing

Responsibility Assignment Matrix

RACI Matrix	Team Members			
Activity	**Name (BA)**	**Name (Developer)**	**Name (Project Manager)**	**Key Business User**
User Requirement Documentation	A	C/I	I	R
	ROHITH	ANURADHA	BHARGAV	ANURADHA

A	Accountable
R	Responsible
C	Consult
I	Inform

Reference

1. https://www.pmi.org/

2. https://www.projectmanagement.com/
3. https://www.tpsgc-pwgsc.gc.ca/biens-property/sngp-npms/ti-it/ervcpgpm-dsfvpmpt-eng.html

Aim

To Prepare Work breakdown structure, Timeline Chart, and Risk identification table

WBS:

Work breakdown structure (WBS) in project management is a method for completing a complex, multi-step project. It's a way to divide and conquer large projects to get things done faster and more efficiently.

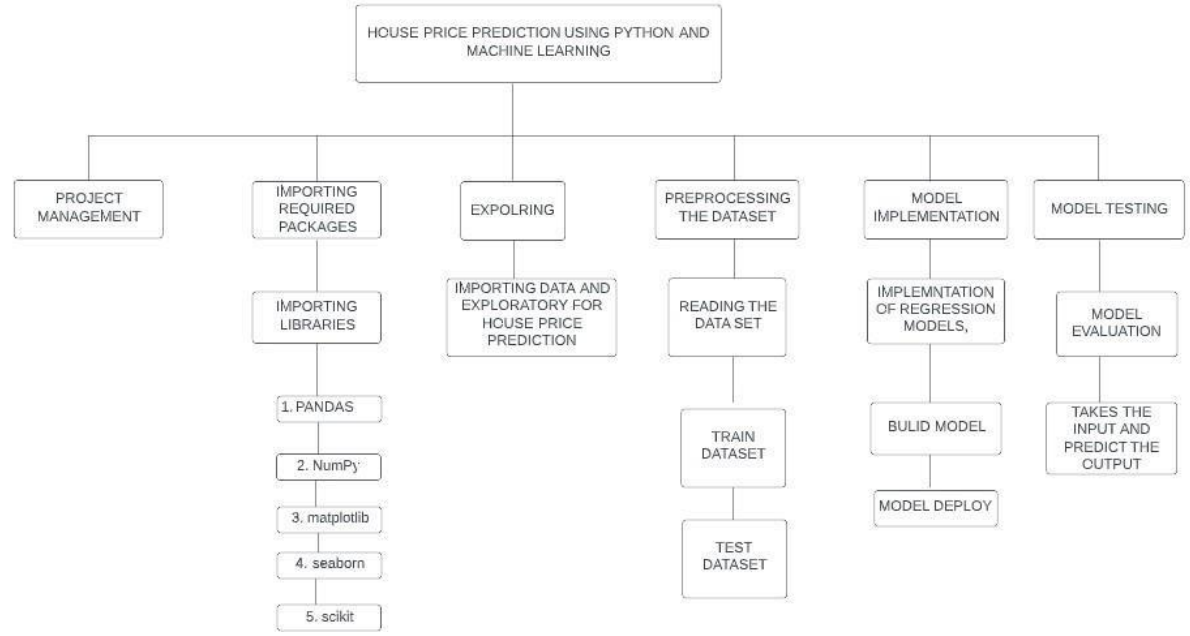
The goal of a WBS is to make a large project more manageable. Breaking it down into smaller chunks means work can be done simultaneously by different team members, leading to better team productivity and easier project management.

WBS for house price prediction using python and machine learning:

project stages:

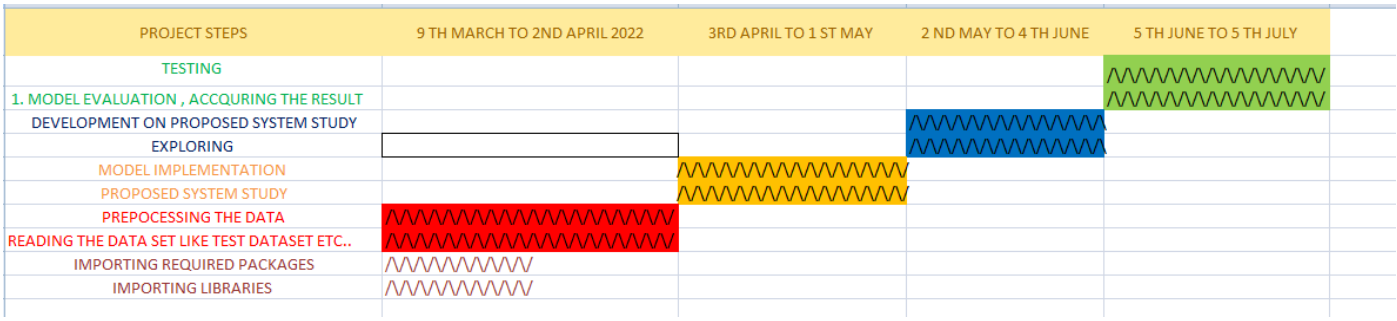
- importing libraries and dataset
- exploring and pre-processing the dataset
- model implementation
- model testing

WBS(WORK BREAKDOWN STRUCTURE):



- Project management
- importing required packages
    - imporing libraries
    - pandas
    - numpy
    - matplotlib
    - seaborn
- Importing the house price data and do some EDA on it
- Data Visualization on the house price data
- Reading the dataset
- train data and test data
- model implementation(bulding the model and carrying out the model)
- model evaluation

TIME LINE GNATT CHART:



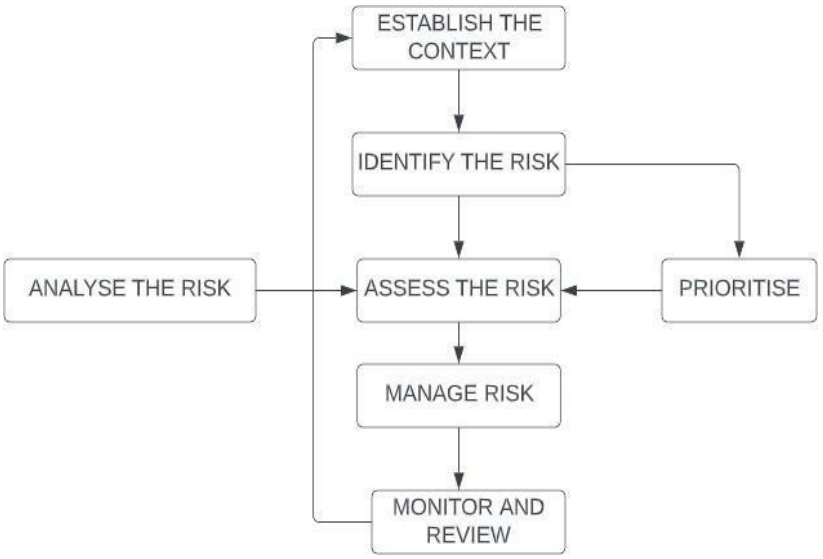
Timeline Gantt chart description:

- identifying and selecting the suitable libraries and importing required packages- 15 days
- preprocessing (understanding the data) and reading the data sets like test data set, train data set - 25-28 days
- model implementation and proposed system study analysis- 28 days
- development on proposed system study and exploring the project- 29 days
- testing validation model evaluation and acquiring the result- one-month

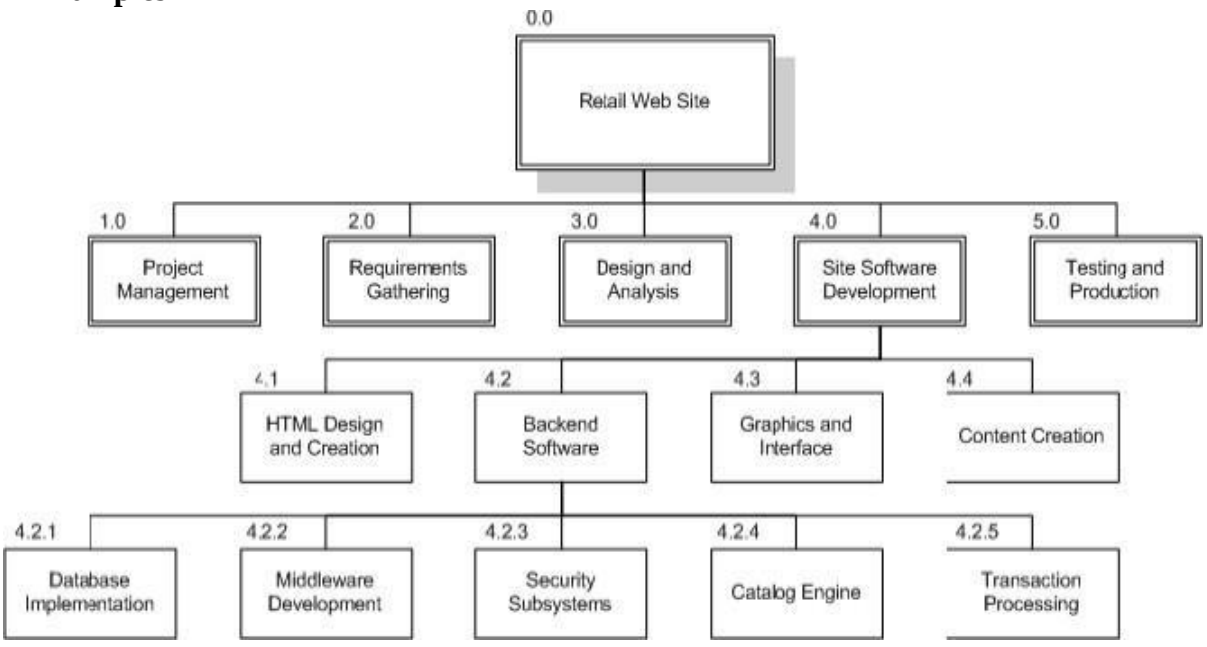
RISK ANALYSIS:

Risk Management is the system of identifying addressing and eliminating these problems before they can damage the project. the house has both consumption and investment properties (demands), and it is very hard to illustrate the consistency between models when financial theories are embedded into economic theory. The risk elements cannot be directly introduced into the standard housing model, especially the housing life-cycle model because of the underlying assumption of certainty.In recent years, the simultaneous increase in house prices in many countries has raised concerns about the potential consequences of large and contagious declines in the near future. In many countries and cities, house prices have increased substantially, reflecting the increased synchronization of house prices. To be more realistic, expected housing capital return is redefined to be uncertain and then households maximize the expectation of their uncertain future lifetime utility.

RISK ANALYSIS IN HOUSE PRICES PREDICTION PYTHON AND MACHINELEARNING:



WBS – Examples



0.0 Retail Web Site

1.0 Project Management

2.0 Requirements Gathering

Analysis &

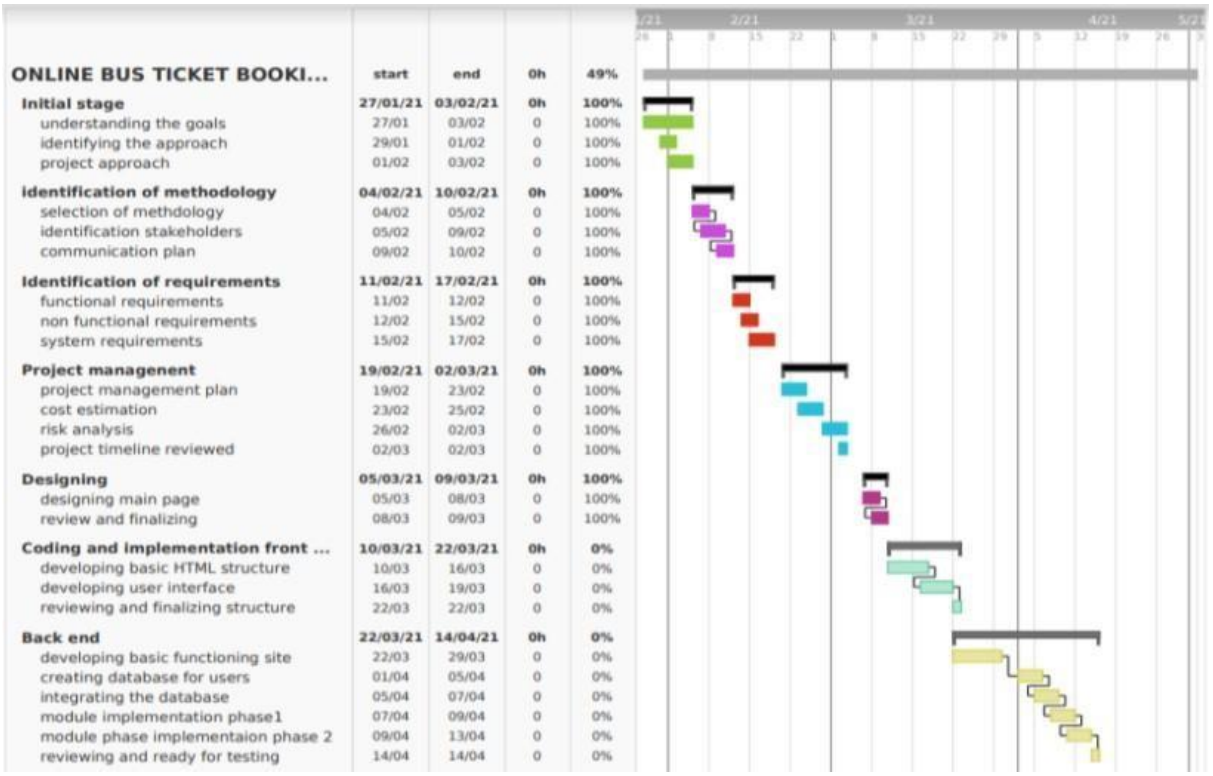
DesignSite Software

Development

- 4.1 HTML Design and Creation
- 4.2 Backend Software
  - 4.2.1 Database Implementation
  - 4.2.2 Middleware Development
  - 4.2.3 Security Subsystems
  - 4.2.4 Catalog Engine
  - 4.2.5 Transaction Processing
- 4.3 Graphics and Interface
- 4.4 Content Creation

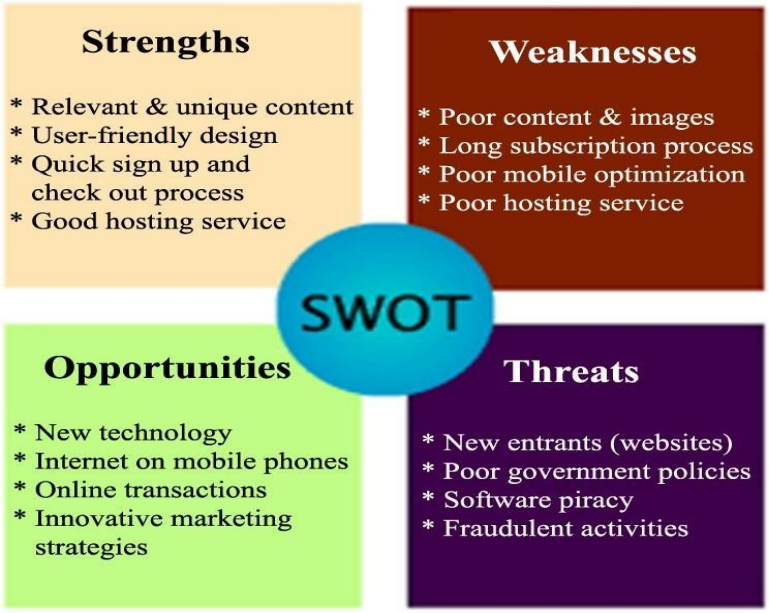
5.0 Testing and Production

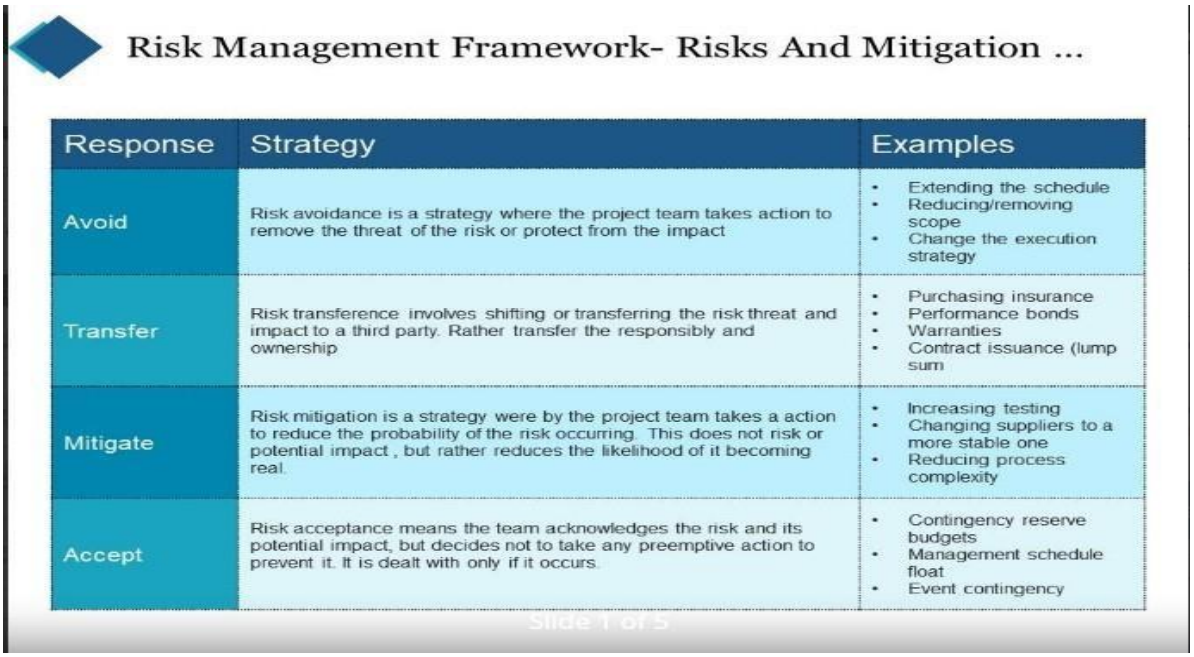
TIMELINE – GANTT CHART



ONLINE BUS TICKET BOOKI...	start	end	0h	49%
Initial stage	27/01/21	03/02/21	0h	100%
understanding the goals	27/01	03/02	0	100%
identifying the approach	29/01	01/02	0	100%
project approach	01/02	03/02	0	100%
identification of methodology	04/02/21	10/02/21	0h	100%
selection of methdology	04/02	05/02	0	100%
identification stakeholders	05/02	09/02	0	100%
communication plan	09/02	10/02	0	100%
Identification of requirements	11/02/21	17/02/21	0h	100%
functional requirements	11/02	12/02	0	100%
non functional requirements	12/02	15/02	0	100%
system requirements	15/02	17/02	0	100%
Project managenent	19/02/21	02/03/21	0h	100%
project management plan	19/02	23/02	0	100%
cost estimation	23/02	25/02	0	100%
risk analysis	26/02	02/03	0	100%
project timeline reviewed	02/03	02/03	0	100%
Designing	05/03/21	09/03/21	0h	100%
designing main page	05/03	08/03	0	100%
review and finalizing	08/03	09/03	0	100%
Coding and implementation front ...	10/03/21	22/03/21	0h	0%
developing basic HTML structure	10/03	16/03	0	0%
developing user interface	16/03	19/03	0	0%
reviewing and finalizing structure	22/03	22/03	0	0%
Back end	22/03/21	14/04/21	0h	0%
developing basic functioning site	22/03	29/03	0	0%
creating database for users	01/04	05/04	0	0%
integrating the database	05/04	07/04	0	0%
module implementation phase1	07/04	09/04	0	0%
module phase implementaion phase 2	09/04	13/04	0	0%
reviewing and ready for testing	14/04	14/04	0	0%

RISK ANALYSIS – SWOT & RMMM





Aim

To prepare architecture and design of the system

Software Used

Star UML, Rational Rose, Etc…

Architecture Diagram with description

Use Case Diagram With Description

ER Diagram With Description (optional)

DFD Diagram (process) With

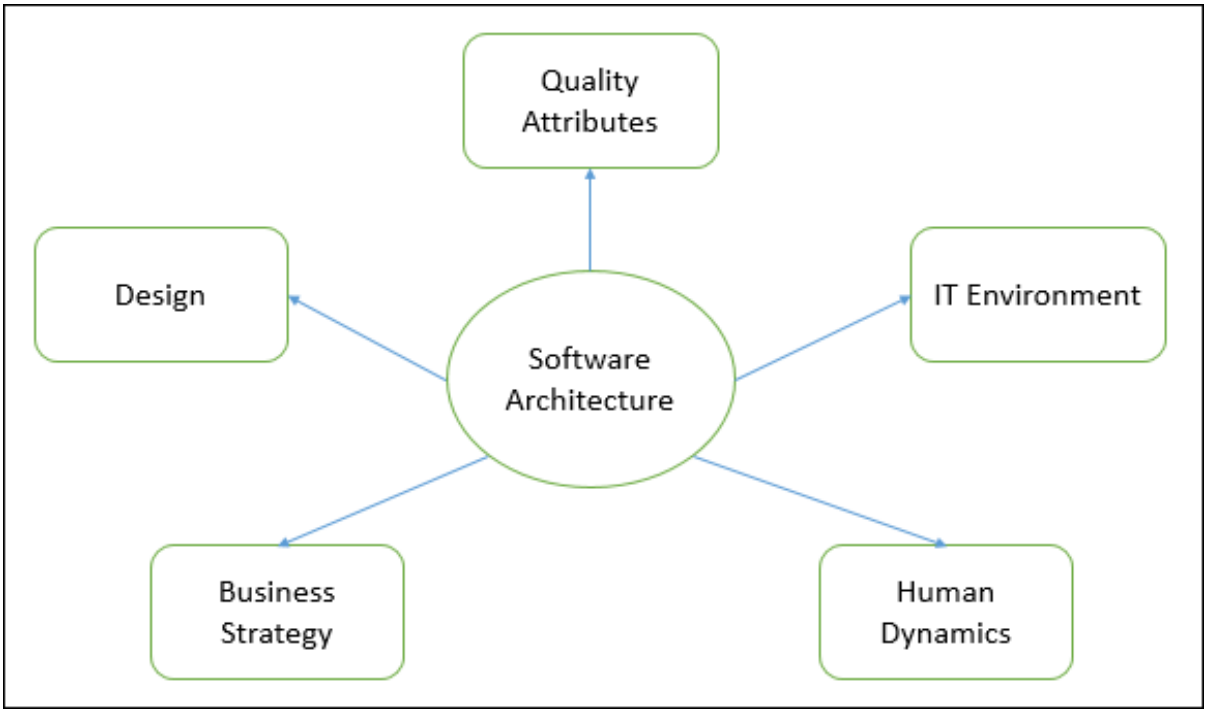
Description

Class Diagram (Applied For OOPS based Project),

Collaboration Diagram (Applied For OOPS based Project)

Architecture Daigram:

The software needs the architectural design to represents the design of software. IEEE defines architectural design as “the process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system.” The software that is built for computer-based systems can exhibit one of these many architectural styles.



Architecture serves as a blueprint for a system. It provides an abstraction to manage the system complexity and establish a communication and coordination mechanism among components.

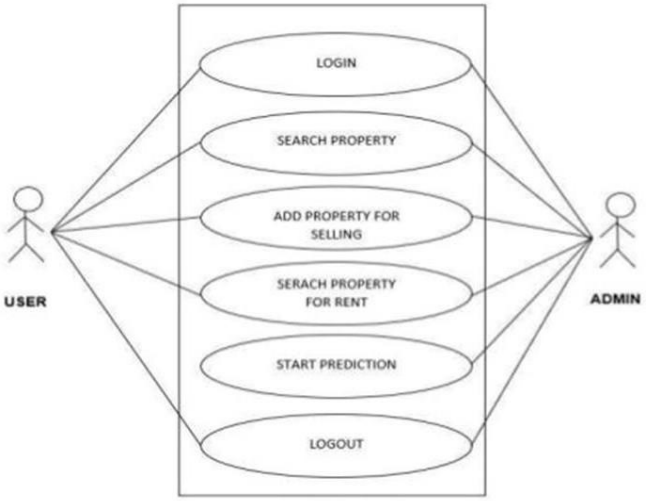
Software design provides a design plan that describes the elements of a system, how they fit, and work together to fulfill the requirement of the system. The objectives of having a design plan are as follows −

- To negotiate system requirements, and to set expectations with customers, marketing, and management personnel.
- Act as a blueprint during the development process.
- Guide the implementation tasks, including detailed design, coding, integration, and testing.

Use Case Diagram With Description:

UML DIAGRAMS

- Use case diagrams are used to gather the requirements of a system including internal and external influences.



USE CASE DIAGRAM

In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent:

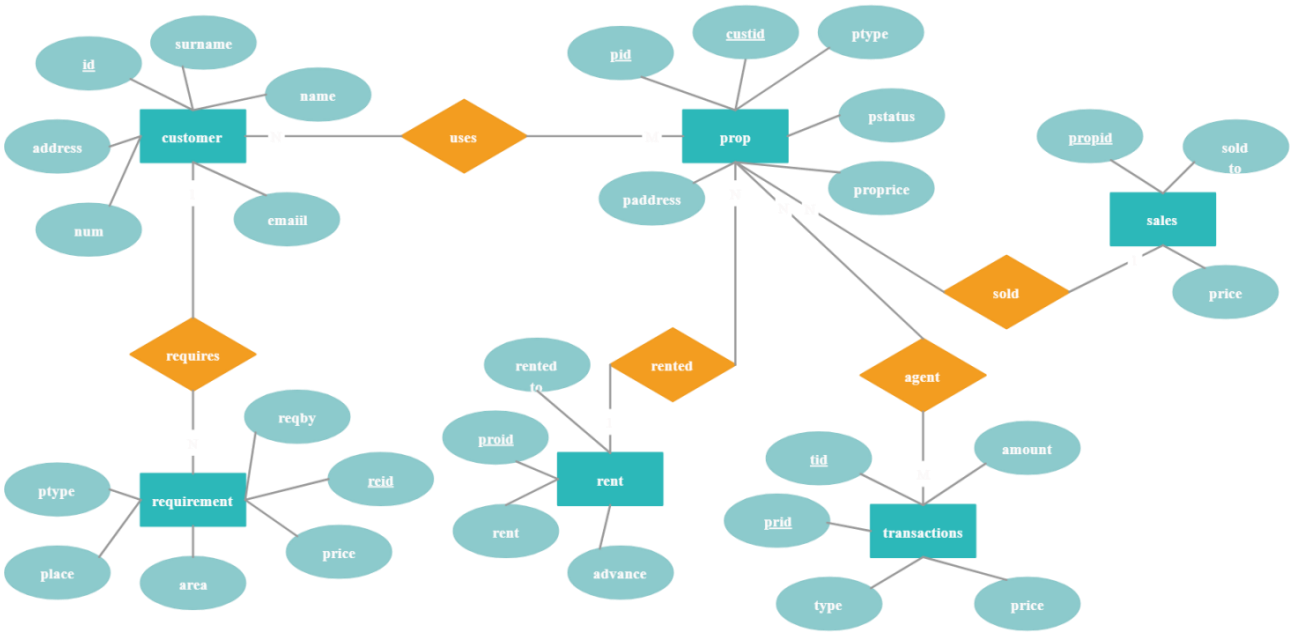
- Scenarios in which your system or application interacts with people, organizations, or external systems
- Goals that your system or application helps those entities (known as actors) achieve
- The scope of your system

UML use case diagrams are ideal for:

- Representing the goals of system-user interactions
- Defining and organizing functional requirements in a system
- Specifying the context and requirements of a system
- Modeling the basic flow of events in a use case

ER Diagram With Description (optional):

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation that depicts relationships among people, objects, places, concepts or events within an information technology (IT) system.

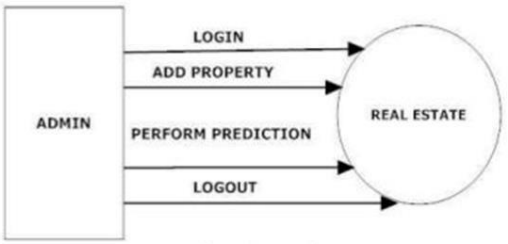


An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties.
By defining the entities, their attributes, and showing the relationships between them, an ER diagram illustrates the logical structure of databases.

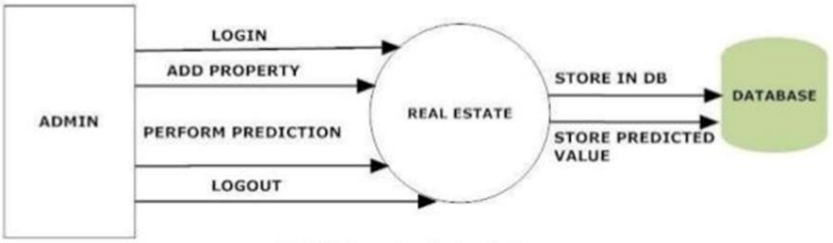
DFD Diagram (process) With Description:

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled.

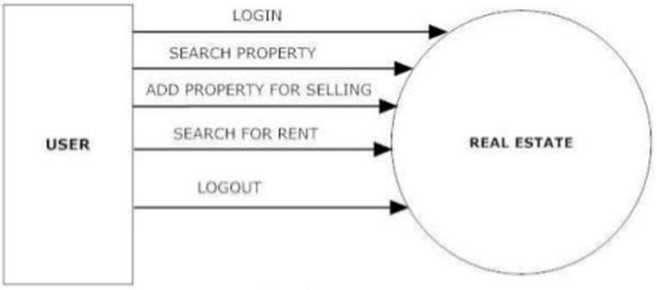
- A data-flow diagram (DFD) is a way of representing a flow of a data of a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself.



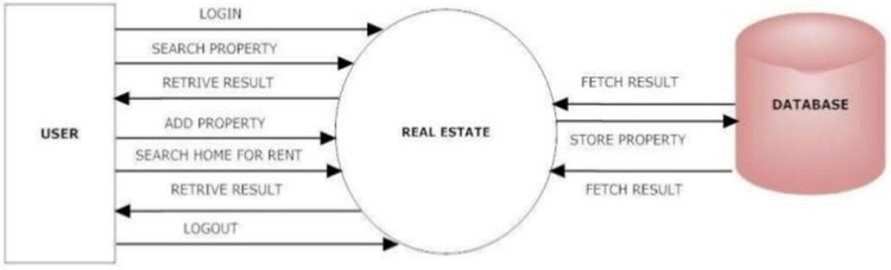
DFD level 0(admin)



DFD level 1(admin)

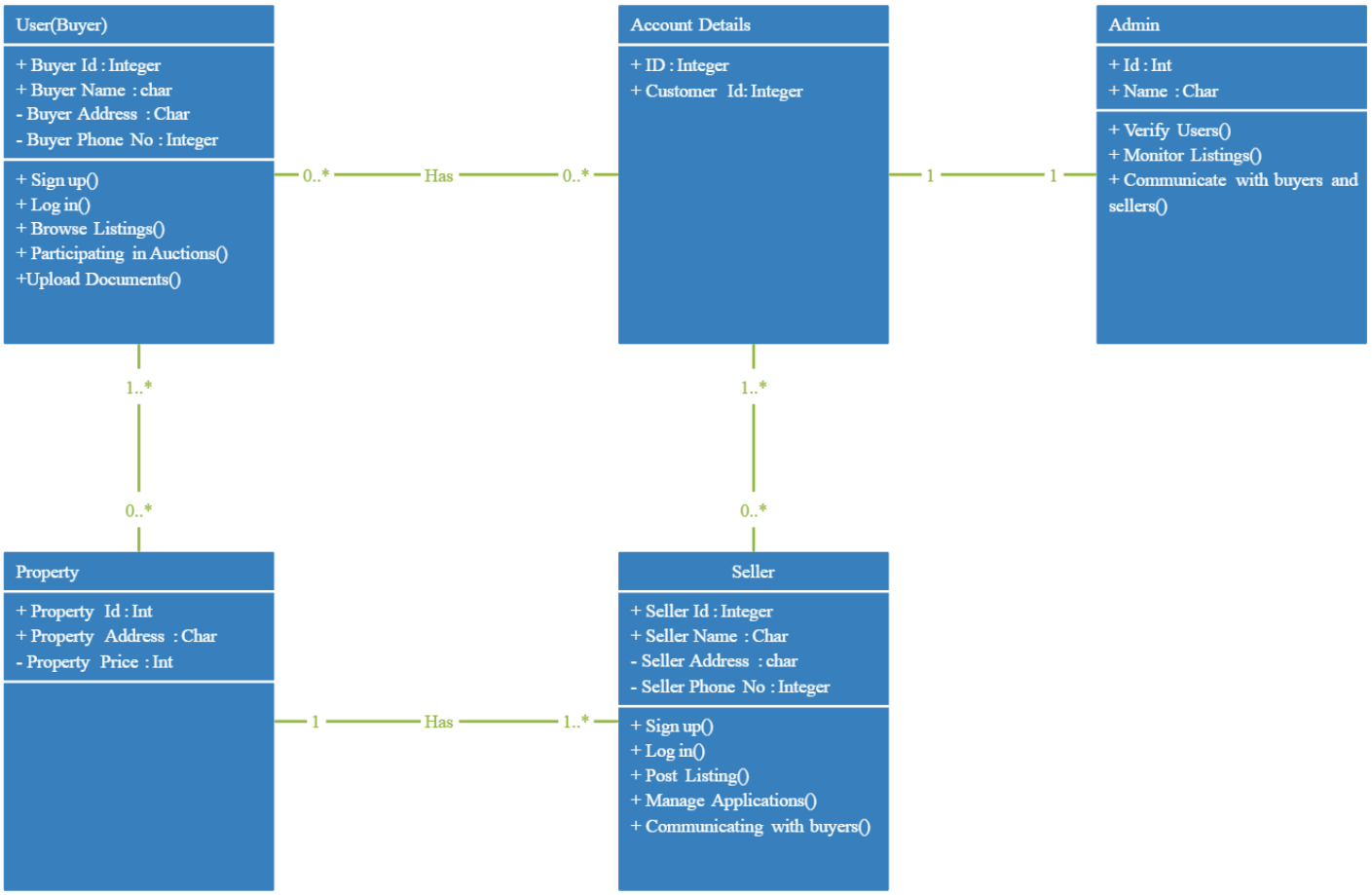


DFD level 0(user)



DFD level 1(user)

CLASS DAIGRAM:

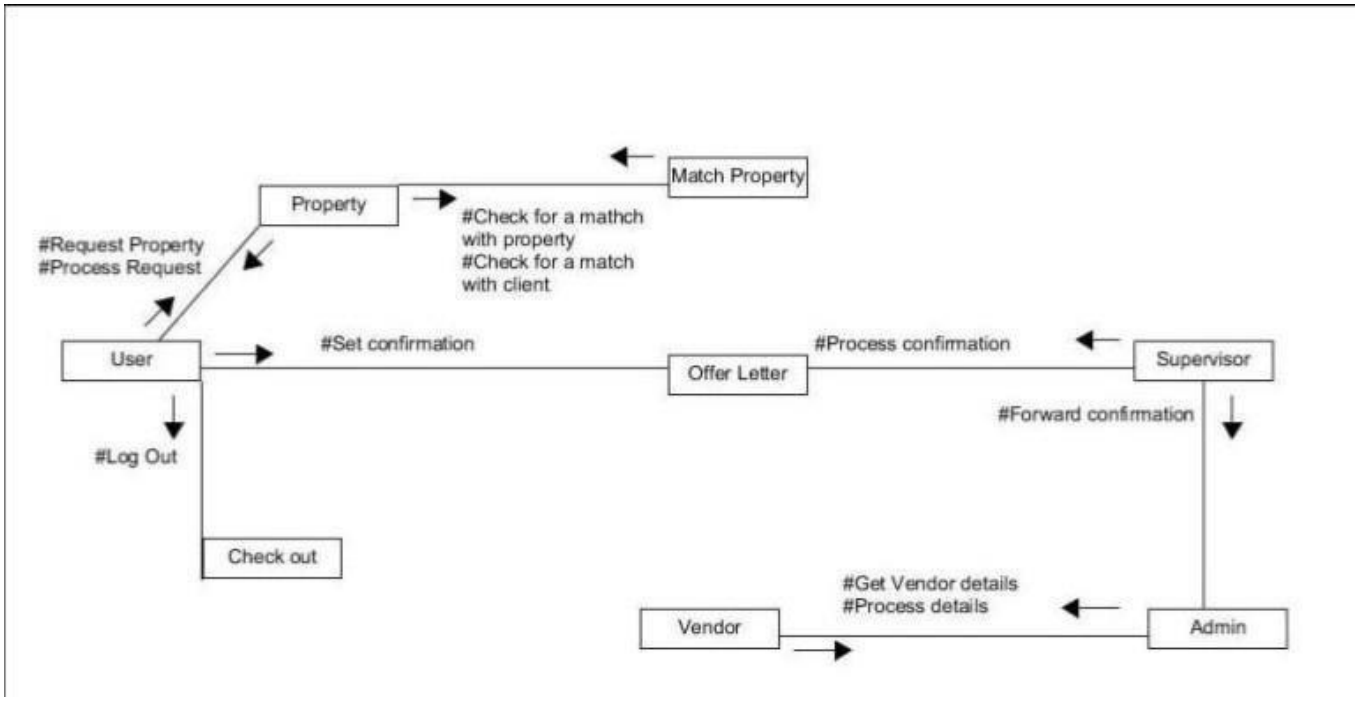


In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static
structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.A class may be involved in one or more relationships with other classes. A relationship can be one of the following types: (Refer to the figure on theright for the graphical representation of relationships).

A class diagram is an illustration of the relationships and source code dependencies among classes in the
Unified Modeling Language (UML). In this context, a class defines the methods and variables in an object, which is a specific entity in a program or the unit of code representing that entity. Class diagrams are usefulin all forms of object-oriented programming (OOP). The concept is several years old but has been refined asOOP modeling paradigms have evolved.

In a class diagram, the classes are arranged in groups that share common characteristics. A class diagram resembles a flowchart in which classes are portrayed as boxes, each box having three rectangles inside. Thetop rectangle contains the name of the class; the middle rectangle contains the attributes of the class; the lower rectangle contains the methods, also called operations, of the class. Lines, which may have arrows at one or both ends, connect the boxes. These lines define the relationships, also called associations, between the classes.

COLLABORATION DIAGRAM:



- A Collaboration is a collection of named objects and actors with links connecting them. Theycollaborate in performing some task.

Unlike a sequence diagram, a collaboration diagram shows the relationships among the objects. Sequencediagrams and collaboration diagrams express similar information, but show it in different ways.

Because of the format of the collaboration diagram, they tend to better suited for analysis activities (see
Activity: Use-Case Analysis). Specifically, they tend to be better suited to depicting simpler interactions ofsmaller numbers of objects. However, if the number of objects and messages grows, the diagram becomes increasingly hard to read. In addition, it is difficult to show additional descriptive information such as timing, decision points, or other unstructured information that can be easily added to the notes in a sequencediagram. So, here are some use cases that we want to create a collaboration diagram for:

- Model collaborations between objects or roles that deliver the functionalities of use cases andoperations
- Model mechanisms within the architectural design of the system
- Capture interactions that show the messages passing between objects and roles within thecollaboration
- Model alternative scenarios within use cases or operations that involve the collaboration of differentobjects and interactions
- Support the identification of objects (hence classes) that participate in use cases
- Each message in a collaboration diagram has a sequence number.

- The top-level message is numbered 1. Messages sent during the same call have the same decimal prefix but suffixes of 1, 2, etc. according to when they occur.

Aim

To Design State, Collaboration, Deployment Diagram, Sample Frontend Design (UI/UX) for the project

Software Used

Star UML, Rational Rose, Etc…

Architecture Diagram with description

State Diagram with Description

Collaboration Diagram with Description

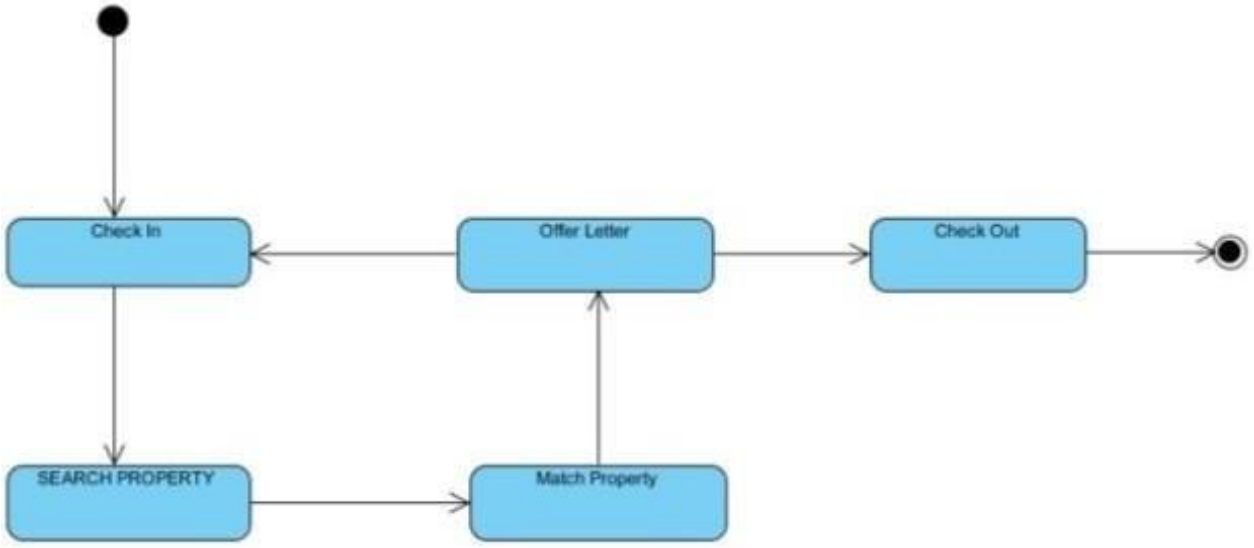
Deployment Diagram with Description

Sample Frontend design

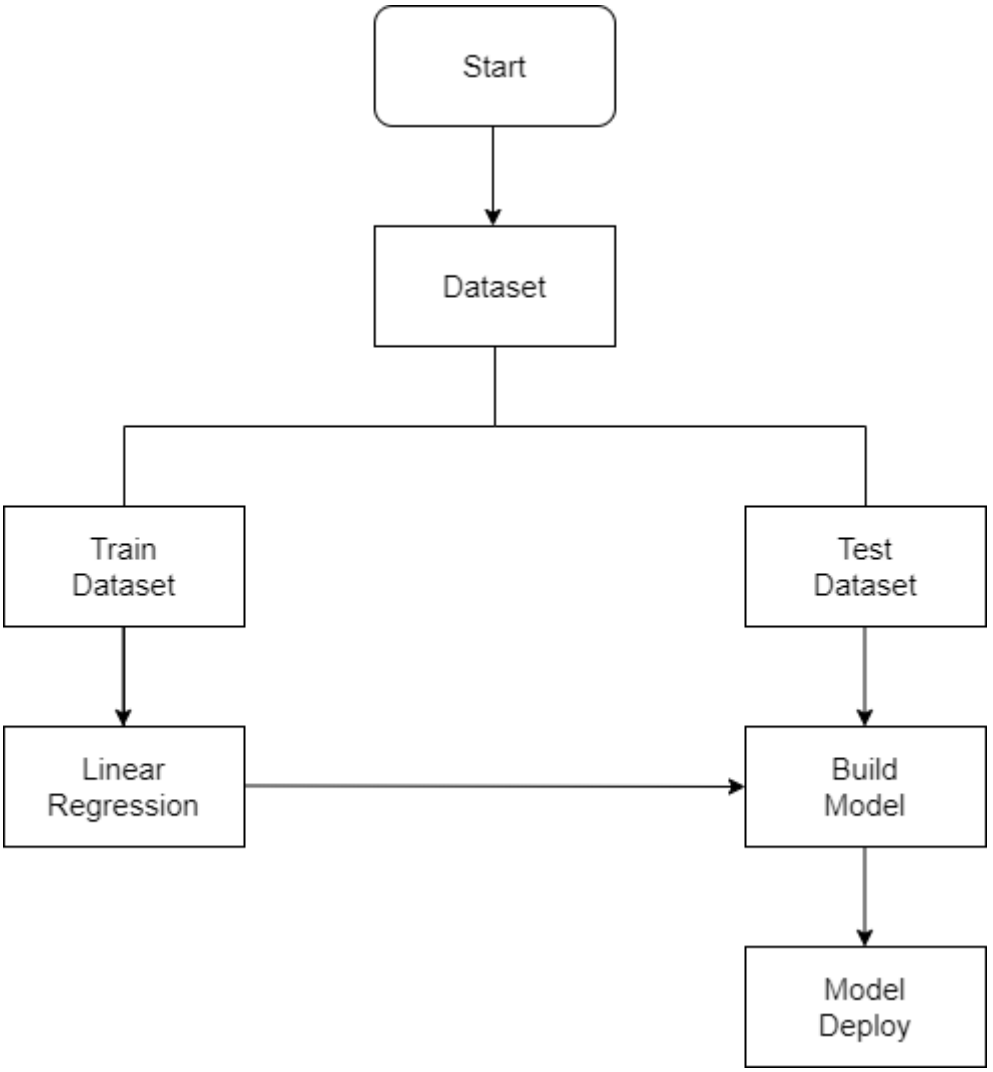
- State Diagram with Description:

A state diagram is the graphical representation of a state machine and one of the 14 UML diagram types for software and systems. State diagrams show a behavioral model consisting of states, state transitions, and actions. UML state diagrams are based on the concept of state diagrams by David Harel. State diagrams depict the permitted states and transitions as well as the events that affect these transitions.

State diagrams are commonly used in the area of embedded systems. State diagrams help to visualize the entire life cycle of objects and thus help to provide a better understanding of state-based systems. An example of such a state-based system is a cash machine: Upon activation either the state ready or the state malfunction could be reached. As soon as the debit card is inserted it is verified. Depending on the result of the verification the pin number is requested or the process is aborted. Other possible states are account query or availability check etc.

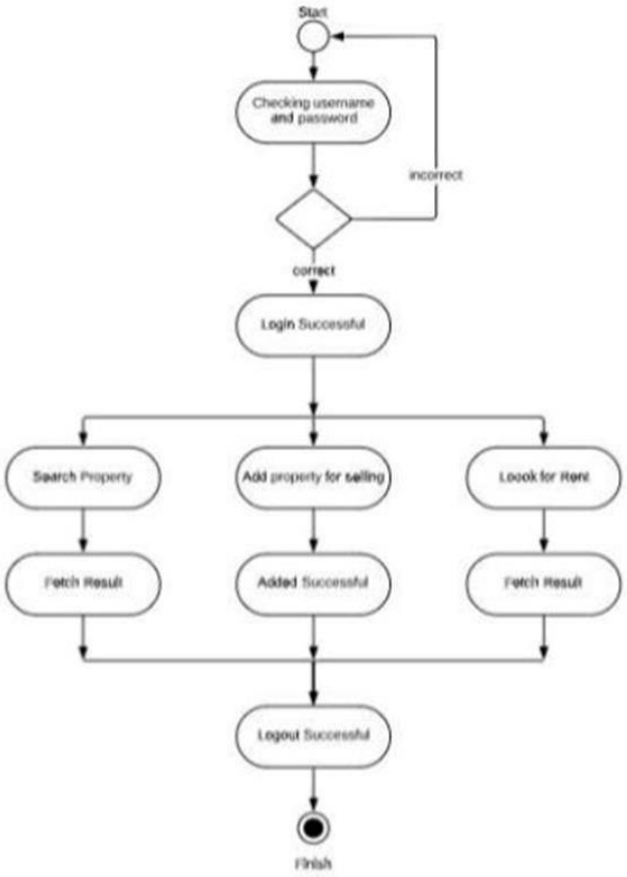


A flowchart illustrates processes that are executed in the system that change the state of objects. A state diagram shows the actual changes in state, not the processes or commands that created those changes. State diagrams mainly depict states and transitions. States are represented with rectangles with rounded corners that are labeled with the name of the state. Transitions are marked with arrows that flow from one state to another, showing how the states change. Each state diagram typically begins with a dark circle that indicates the initial state and ends with a bordered circle that denotes the final state. However, despite having clear start and end points, state diagrams are not necessarily the best tool for capturing an overall progression of events. Rather, they illustrate specific kinds of behavior—in particular, shifts from one state to another.



not only linear regression we can use advanced regression, and random forest also for house prediction using machine learning and python.

- Activity diagram is basically a flowchart to represent the flow from one activity to another activity.



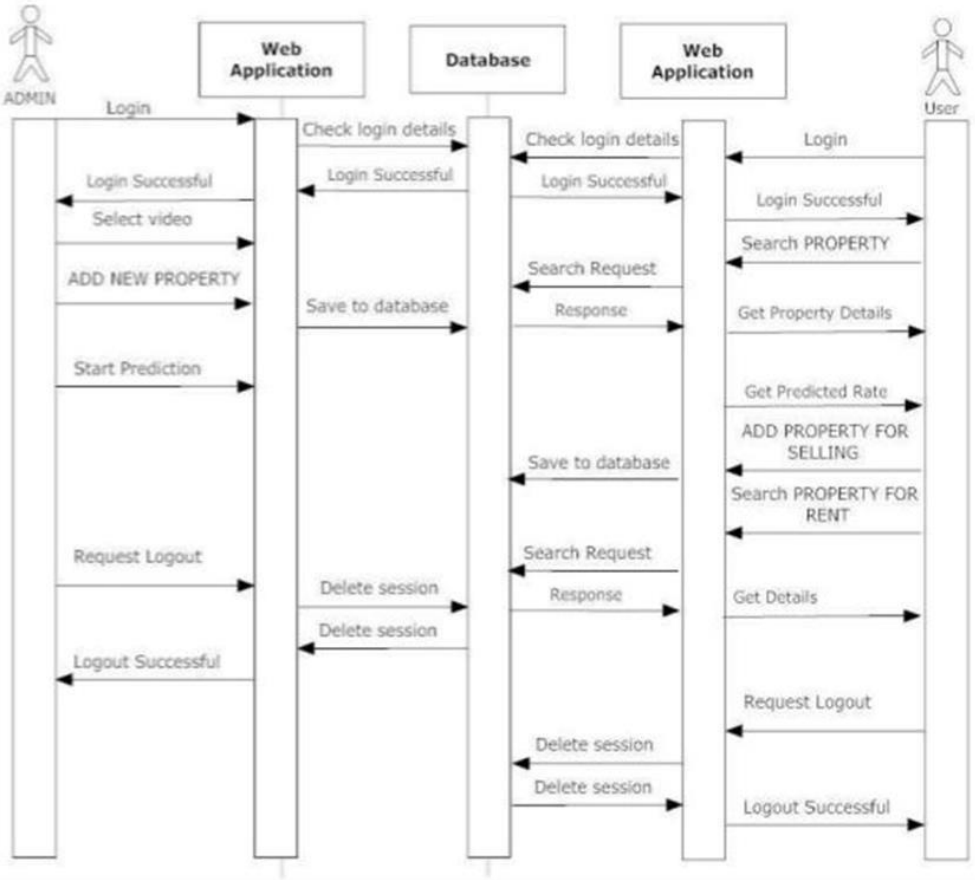
Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

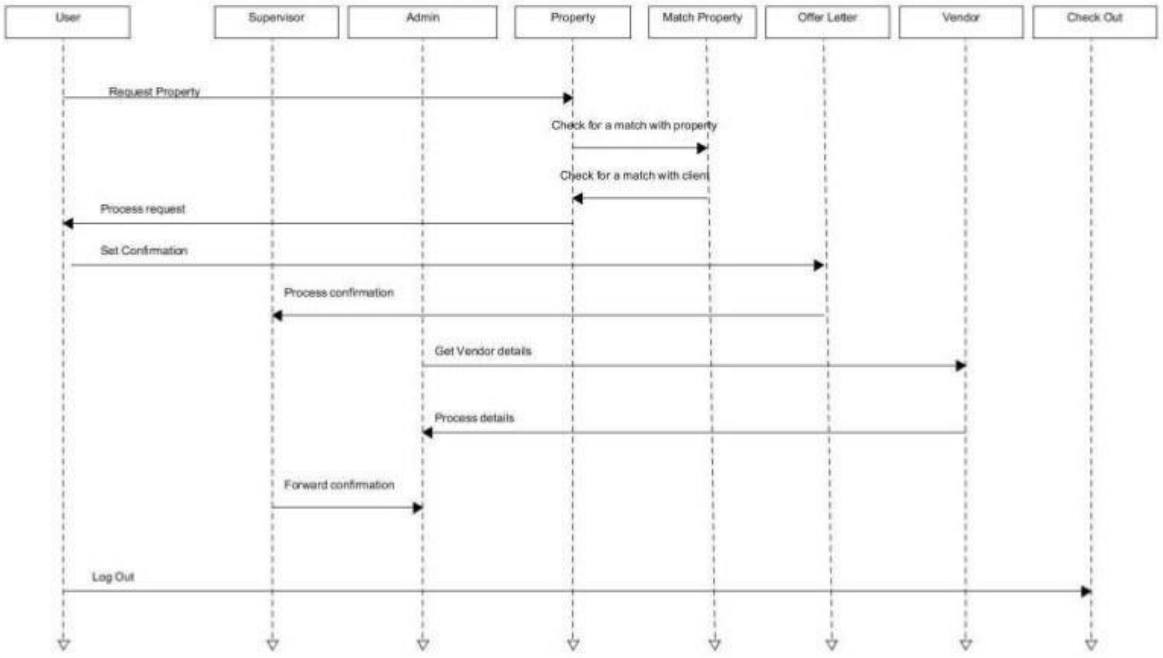
The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

- Sequence diagram emphasizes on time sequence of messages.



ANOTHER WAY OF SEQUENCE DIAGRAM:



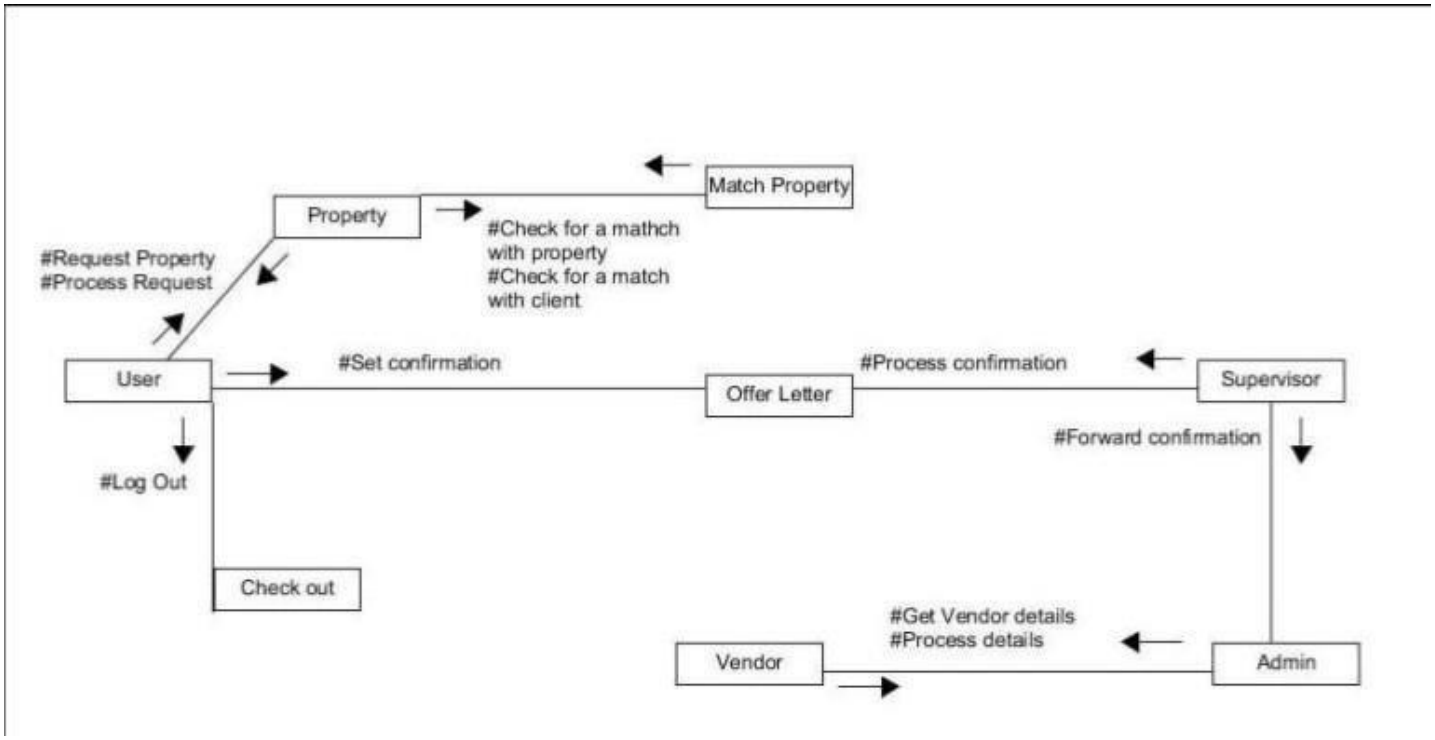
Sequence Diagrams captures:

- the interaction that takes place in a collaboration that either realizes a use case or an operation (instance diagrams or generic diagrams)
- high-level interactions between user of the system and the system, between the system and other systems, or between subsystems (sometimes known as system sequence diagrams)Purpose of Sequence Diagram
- Model high-level interaction between active objects in a system
- Model the interaction between object instances within a collaboration that realizes a use case
- Model the interaction between objects within a collaboration that realizes an operation
- Either model generic interactions (showing all possible paths through the interaction) or specific instances of a interaction (showing just one path through the interaction)

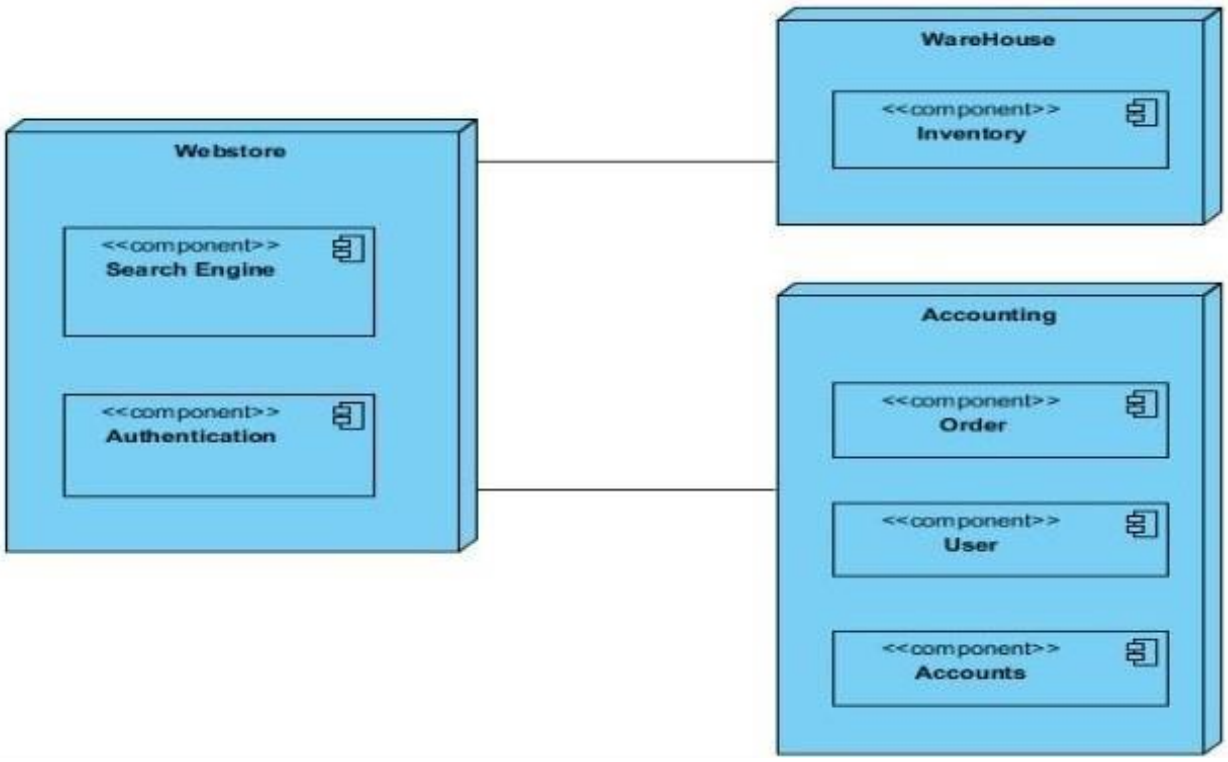
Purpose of Sequence Diagram

- Model high-level interaction between active objects in a system
- Model the interaction between object instances within a collaboration that realizes a use case
- Model the interaction between objects within a collaboration that realizes an operation
- Either model generic interactions (showing all possible paths through the interaction) or specific instances of a interaction (showing just one path through the interaction)

COLLABORATION DAIGRAM:



DEPLOYMENT DIAGRAM:



Aim:

To describe modules and implement Module1

Software Used

C, C++, Python, HTML, Mysql, Etc...

Code of Module 1

Result of Module 1

In this project, I have applied some regression methods of supervised learning using Python in Machine Learning to predict the house price.we will be using Pycharm IDE to solve this problem if you have not that IDE in your system you can download it from their official website(make sure you download the community version) and we will be using Python language to solve this problem.

Regression is used to predict future values based on the independent variable. model's evaluation is done by calculating the error value. The smaller the error the greater the accuracy of our regression model.

Example of Supervised Learning Algorithms:

1)Linear Regression

2)Nearest Neighbor

3)Gaussian Naive Bayes

4)Decision Trees

5)Support Vector Machine (SVM)

6)Random Forest

7) XGBoost

8)ADA Boost

MODULE1:

Here, in this project(house price prediction using machine learning and python) module 1 consists of basic code of the project . module 1 all shows basic and starting steps towards prediction houses in ames and lowa using machine learning and python.

I have taken a very basic approach and I hope you find it useful.

My main objectives on this project are:

- Applying exploratory data analysis and trying to get some insights about our dataset
- Getting data in better shape by transforming and feature engineering to help us in building better models
- Building and tuning couple models to get some stable results on predicting housing prices

importing libraries

Concatenating both the datasets or Loading data sets

IMPORTING LIBRARIES:

First thing first , we import our libraries and dataset and then we see the head of the data to know how the data looks like and use describe function to see the percentile's and other key statistics.

We are going to need some packages and libraries:

1)Numpy-for linear algebraic operations.

2)Scikit-learn-includes many statistical models.

3)Pandas-To load the dataset.

4)matplotlib and seaborn-to to different plotting.

code:

```
%matplotlib inline
import seaborn as sns

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats



data_train = pd.read_csv("train (1).csv")
data_test = pd.read_csv("test (1).csv")



data_train.head(10)
```

output: 10 rows × 81 columns

```
data_test.head(10)
```

output: 10 rows × 80 columns

Importing Libraries

In [11]:
```
%matplotlib inline
import seaborn as sns
```

In [12]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [13]:
```
data_train = pd.read_csv("train (1).csv")
data_test = pd.read_csv("test (1).csv")
```

In [14]:
```
data_train.head(10)
```

Out[14]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold	YrSold	SaleTy
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	2	2008	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	5	2007	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	9	2008	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	2	2006	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	12	2008	
5	6	50	RL	85.0	14115	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	MnPrv	Shed	700	10	2009	
6	7	20	RL	75.0	10084	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	8	2007	
7	8	60	RL	NaN	10382	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	Shed	350	11	2009	
8	9	50	RM	51.0	6120	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	4	2008	
9	10	190	RL	50.0	7420	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	1	2008	

In [13]:
```
data_train = pd.read_csv("train (1).csv")
data_test = pd.read_csv("test (1).csv")
```

In [14]:
```
data_train.head(10)
```

Out[14]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold	YrSold	SaleTy
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	2	2008	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	5	2007	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	9	2008	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	2	2006	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	12	2008	
5	6	50	RL	85.0	14115	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	MnPrv	Shed	700	10	2009	
6	7	20	RL	75.0	10084	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	8	2007	
7	8	60	RL	NaN	10382	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	Shed	350	11	2009	
8	9	50	RM	51.0	6120	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	4	2008	
9	10	190	RL	50.0	7420	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0	1	2008	

10 rows × 81 columns

In [15]:
```
data_test.head(10)
```

Out[15]:

LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	ScreenPorch	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold	YrSold	SaleType	SaleCondition
80.0	11622	Pave	NaN	Reg	Lvl	AllPub	...	120	0	NaN	MnPrv	NaN	0	6	2010	WD	Normal
81.0	14267	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	Gar2	12500	6	2010	WD	Normal
74.0	13830	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	MnPrv	NaN	0	3	2010	WD	Normal

In [15]:
```
data_test.head(10)
```

Out[15]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	ScreenPorch	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold
0	1461	20	RH	80.0	11622	Pave	NaN	Reg	Lvl	AllPub	...	120	0	NaN	MnPrv	NaN	0	6
1	1462	20	RL	81.0	14267	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	Gar2	12500	6
2	1463	60	RL	74.0	13830	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	MnPrv	NaN	0	3
3	1464	60	RL	78.0	9978	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	NaN	0	6
4	1465	120	RL	43.0	5005	Pave	NaN	IR1	HLS	AllPub	...	144	0	NaN	NaN	NaN	0	1
5	1466	60	RL	75.0	10000	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	NaN	0	4
6	1467	20	RL	NaN	7980	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	GdPrv	Shed	500	3
7	1468	60	RL	63.0	8402	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	NaN	0	5
8	1469	20	RL	85.0	10176	Pave	NaN	Reg	Lvl	AllPub	...	0	0	NaN	NaN	NaN	0	2
9	1470	20	RL	70.0	8400	Pave	NaN	Reg	Lvl	AllPub	...	0	0	NaN	MnPrv	NaN	0	4

10 rows × 80 columns

In [16]:
```
print(data_train.shape)
print(data_test.shape)
```

```
(1460, 81)
(1459, 80)
```

loading the data or concatenating the data:

Now I will explain the whole dataset that how many columns are there and which columns stand for which data.

Here's a brief version of the data.

- SalePrice - the property's sale price in dollars. This is the target variable that you're trying to predict. (Dependent Variable)

- MSSubClass: The building class

- MSZoning: The general zoning classification
- LotFrontage: Linear feet of street connected to property
- LotArea: Lot size in square feet

- Street: Type of road access
- Alley: Type of alley access

- LotShape: General shape of property
- LandContour: Flatness of the property

- Utilities: Type of utilities available

- LotConfig: Lot configuration
- LandSlope: Slope of property
- Neighborhood: Physical locations within Ames city limits

- Condition1: Proximity to main road or railroad
- Condition2: Proximity to main road or railroad (if a second is present)

- BldgType: Type of dwelling
- HouseStyle: Style of dwelling

- OverallQual: Overall material and finish quality
- OverallCond: Overall condition rating

- YearBuilt: Original construction date
- YearRemodAdd: Remodel date

- RoofStyle: Type of roof
- RoofMatl: Roof material

- Exterior1st: Exterior covering on house
- Exterior2nd: Exterior covering on house (if more than one material)
- MasVnrType: Masonry veneer type

- MasVnrArea: Masonry veneer area in square feet
- ExterQual: Exterior material quality

- ExterCond: Present condition of the material on the exterior
- Foundation: Type of foundation

- BsmtQual: Height of the basement
- BsmtCond: General condition of the basement

- BsmtExposure: Walkout or garden level basement walls
- BsmtFinType1: Quality of basement finished area

- BsmtFinSF1: Type 1 finished square feet

- BsmtFinType2: Quality of second finished area (if present)

- BsmtFinSF2: Type 2 finished square feet
- BsmtUnfSF: Unfinished square feet of basement area
- TotalBsmtSF: Total square feet of basement area

- Heating: Type of heating
- HeatingQC: Heating quality and condition

- CentralAir: Central air conditioning
- Electrical: Electrical system

- 1stFlrSF: First Floor square feet
- 2ndFlrSF: Second floor square feet

- LowQualFinSF: Low quality finished square feet (all floors)
- GrLivArea: Above grade (ground) living area square feet

- BsmtFullBath: Basement full bathrooms
- BsmtHalfBath: Basement half bathrooms
- FullBath: Full bathrooms above grade

- HalfBath: Half baths above grade
- Bedroom: Number of bedrooms above basement level

- Kitchen: Number of kitchens
- KitchenQual: Kitchen quality

- TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)
- Functional: Home functionality rating

- Fireplaces: Number of fireplaces
- FireplaceQu: Fireplace quality

- GarageType: Garage location
- GarageYrBlt: Year garage was built

- GarageFinish: Interior finish of the garage

- GarageCars: Size of garage in car capacity

- GarageArea: Size of garage in square feet
- GarageQual: Garage quality
- GarageCond: Garage condition

- PavedDrive: Paved driveway
- WoodDeckSF: Wood deck area in square feet

- OpenPorchSF: Open porch area in square feet
- EnclosedPorch: Enclosed porch area in square feet

- 3SsnPorch: Three season porch area in square feet
- ScreenPorch: Screen porch area in square feet

- PoolArea: Pool area in square feet
- PoolQC: Pool quality

- Fence: Fence quality
- MiscFeature: Miscellaneous feature not covered in other categories
- MiscVal: $Value of miscellaneous feature

- MoSold: Month Sold
- YrSold: Year Sold

- SaleType: Type of sale
- SaleCondition: Condition of sale

We have 1460 observations of 80 variables in the training data frame and 1459 rows and 79 columns test dataset.

In [498…
```
df_train.shape
```
Out[498…
```
(1460, 81)
```
In [499…
```
df_test.shape
```
Out[499…
```
(1459, 80)
```
In [500…
```
df_train.columns
```
Out[500…
```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
       'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
       'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
       'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',
       'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',
       'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',
       'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition', 'SalePrice'],
      dtype='object')
```
In [501…
```
df_test.columns
```
Out[501…
```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
       'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
       'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
       'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',
       'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',
       'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',
       'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition'],
      dtype='object')
```

Here we have written code for importing and loading the data sets(train sets, test sets) for the project.

In upcoming modules 2, 3 we will be discussing about Handling missing values Handling Categorical Features, Handling training dataHandling Test DataFinding predictions for Test data and make .csv file

Aim

To implement Module 2 of the project and display the output of the module with new requirements may assimilated

Software Used

C, C++, Python, HTML, Mysql, Etc...

Code of Module 2

Result of Module 2

PROJECT: HOUSE PRICE PREDICTION USING MACHINE LEARNING AND PYTHON

In the previous module i.e module 1 we have discussed importing the libraries required for the data set and loading the datasets(train dataset and test data set)

brief revision of module 1:

importing libraries

loading or concatenating both data sets

My main objectives on this project are:

- Applying exploratory data analysis and trying to get some insights about our dataset
- Getting data in better shape by transforming and feature engineering to help us in building better models
- Building and tuning couple models to get some stable results on predicting housing prices

module1:

Acquire the data and create our environment which also means importing libraries

We need to acquire the data for the competition. The descriptions of the features and some other helpful information are contained in a file with an obvious name, data_description.txt.

Download the data and save it into a folder where you'll keep everything you need for the competition.

We will first look at the train.csv data. After we've trained a model, we'll make predictions using the test.csv data.

First, import Pandas, a fantastic library for working with data in Python. Next we'll import Numpy.

We can use Pandas to read in csv files. The pd.read_csv() method creates a DataFrame from a csv file.

We see that test has only 80 columns, while train has 81. This is due to, of course, the fact that the test data do not include the final sale price information!

1.2 loading the dataset or combining or it could be exploring the data required for the house price prediction in ames and lowa

The training dataset includes 1460 examples, 80 features, and 1 label, while the test data contains 1459 examples and 80 features.

here in the module we will be discussing about the handling the data set of both sets i.e train and test data sets… handling missing values, handling categorical features,handling training data set and handling testing dataset..

code for module 2 with output:

handling the training data:

df_train.info()

output: (for the df_train.info()):

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 1460 entries, 0 to 1459

Data columns (total 81 columns):

#	Column	Non-Null Count	Dtype
0	Id	1460 non-null	int64
1	MSSubClass	1460 non-null	int64
2	MSZoning	1460 non-null	object
3	LotFrontage	1201 non-null	float64
4	LotArea	1460 non-null	int64
5	Street	1460 non-null	object
6	Alley	91 non-null	object
7	LotShape	1460 non-null	object
8	LandContour	1460 non-null	object
9	Utilities	1460 non-null	object
10	LotConfig	1460 non-null	object
11	LandSlope	1460 non-null	object
12	Neighborhood	1460 non-null	object
13	Condition1	1460 non-null	object
14	Condition2	1460 non-null	object
15	BldgType	1460 non-null	object
16	HouseStyle	1460 non-null	object
17	OverallQual	1460 non-null	int64
18	OverallCond	1460 non-null	int64
19	YearBuilt	1460 non-null	int64

20 YearRemodAdd 1460 non-null int64
21 RoofStyle 1460 non-null object
22 RoofMatl 1460 non-null object
23 Exterior1st 1460 non-null object
24 Exterior2nd 1460 non-null object
25 MasVnrType 1452 non-null object
26 MasVnrArea 1452 non-null float64
27 ExterQual 1460 non-null object
28 ExterCond 1460 non-null object
29 Foundation 1460 non-null object
30 BsmtQual 1423 non-null object
31 BsmtCond 1423 non-null object
32 BsmtExposure 1422 non-null object
33 BsmtFinType1 1423 non-null object
34 BsmtFinSF1 1460 non-null int64
35 BsmtFinType2 1422 non-null object
36 BsmtFinSF2 1460 non-null int64
37 BsmtUnfSF 1460 non-null int64
38 TotalBsmtSF 1460 non-null int64
39 Heating 1460 non-null object
40 HeatingQC 1460 non-null object
41 CentralAir 1460 non-null object
42 Electrical 1459 non-null object
43 1stFlrSF 1460 non-null int64
44 2ndFlrSF 1460 non-null int64
45 LowQualFinSF 1460 non-null int64
46 GrLivArea 1460 non-null int64
47 BsmtFullBath 1460 non-null int64
48 BsmtHalfBath 1460 non-null int64
49 FullBath 1460 non-null int64
50 HalfBath 1460 non-null int64
51 BedroomAbvGr 1460 non-null int64
52 KitchenAbvGr 1460 non-null int64

```
53  KitchenQual     1460 non-null object
54  TotRmsAbvGrd 1460 non-null int64
55  Functional      1460 non-null   object
56  Fireplaces      1460 non-null   int64
57  FireplaceQu     770 non-null      object
58  GarageType      1379 non-null   object
59  GarageYrBlt     1379 non-null   float64
60  GarageFinish 1379 non-null object
61  GarageCars      1460 non-null   int64
62  GarageArea      1460 non-null   int64
63  GarageQual      1379 non-null   object
64  GarageCond      1379 non-null object
65  PavedDrive      1460 non-null object
66  WoodDeckSF      1460 non-null   int64
67  OpenPorchSF     1460 non-null   int64
68  EnclosedPorch 1460 non-null int64
69  3SsnPorch       1460 non-null int64
70  ScreenPorch     1460 non-null int64
71  PoolArea        1460 non-null int64
72  PoolQC          7 non-null       object
73  Fence           281 non-null      object
74  MiscFeature     54 non-null       object
75  MiscVal         1460 non-null   int64
76  MoSold          1460 non-null   int64
77  YrSold          1460 non-null int64
78  SaleType        1460 non-null object
79  SaleCondition 1460 non-null object
80  SalePrice   1460    non-null    int64
dtypes: float64(3), int64(35), object(43)
memory usage: 924.0+ KB
```

as we discussed earlier there will be 80 variables and in the output we got all 80 objects.

Handling training data

In [502…

```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 81 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             1460 non-null   int64
 1   MSSubClass     1460 non-null   int64
 2   MSZoning       1460 non-null   object
 3   LotFrontage    1201 non-null   float64
 4   LotArea        1460 non-null   int64
 5   Street         1460 non-null   object
 6   Alley          91 non-null     object
 7   LotShape       1460 non-null   object
 8   LandContour    1460 non-null   object
 9   Utilities      1460 non-null   object
 10  LotConfig      1460 non-null   object
 11  LandSlope      1460 non-null   object
 12  Neighborhood   1460 non-null   object
 13  Condition1     1460 non-null   object
 14  Condition2     1460 non-null   object
 15  BldgType       1460 non-null   object
 16  HouseStyle     1460 non-null   object
 17  OverallQual    1460 non-null   int64
 18  OverallCond    1460 non-null   int64
 19  YearBuilt      1460 non-null   int64
 20  YearRemodAdd   1460 non-null   int64
 21  RoofStyle      1460 non-null   object
 22  RoofMatl       1460 non-null   object
 23  Exterior1st    1460 non-null   object
 24  Exterior2nd    1460 non-null   object
 25  MasVnrType     1452 non-null   object
 26  MasVnrArea     1452 non-null   float64
 27  ExterQual      1460 non-null   object
 28  ExterCond      1460 non-null   object
 29  Foundation     1460 non-null   object
 28  ExterCond      1460 non-null   object
 29  Foundation     1460 non-null   object
 30  BsmtQual       1423 non-null   object
 31  BsmtCond       1423 non-null   object
 32  BsmtExposure   1422 non-null   object
 33  BsmtFinType1   1423 non-null   object
 34  BsmtFinSF1     1460 non-null   int64
 35  BsmtFinType2   1422 non-null   object
 36  BsmtFinSF2     1460 non-null   int64
 37  BsmtUnfSF      1460 non-null   int64
 38  TotalBsmtSF    1460 non-null   int64
 39  Heating        1460 non-null   object
 40  HeatingQC      1460 non-null   object
 41  CentralAir     1460 non-null   object
 42  Electrical     1459 non-null   object
 43  1stFlrSF       1460 non-null   int64
 44  2ndFlrSF       1460 non-null   int64
 45  LowQualFinSF   1460 non-null   int64
 46  GrLivArea      1460 non-null   int64
 47  BsmtFullBath   1460 non-null   int64
 48  BsmtHalfBath   1460 non-null   int64
 49  FullBath       1460 non-null   int64
 50  HalfBath       1460 non-null   int64
 51  BedroomAbvGr   1460 non-null   int64
 52  KitchenAbvGr   1460 non-null   int64
 53  KitchenQual    1460 non-null   object
 54  TotRmsAbvGrd   1460 non-null   int64
 55  Functional     1460 non-null   object
 56  Fireplaces     1460 non-null   int64
 57  FireplaceQu    770 non-null    object
 58  GarageType     1379 non-null   object
 59  GarageYrBlt    1379 non-null   float64
 60  GarageFinish   1379 non-null   object
 61  GarageCars     1460 non-null   int64
 62  GarageArea     1460 non-null   int64
 63  GarageQual     1379 non-null   object
 64  GarageCond     1379 non-null   object
 65  PavedDrive     1460 non-null   object
 66  WoodDeckSF     1460 non-null   int64
 67  OpenPorchSF    1460 non-null   int64
 68  EnclosedPorch  1460 non-null   int64
```

```
 43  1stFlrSF       1460 non-null   int64
 44  2ndFlrSF       1460 non-null   int64
 45  LowQualFinSF   1460 non-null   int64
 46  GrLivArea      1460 non-null   int64
 47  BsmtFullBath   1460 non-null   int64
 48  BsmtHalfBath   1460 non-null   int64
 49  FullBath       1460 non-null   int64
 50  HalfBath       1460 non-null   int64
 51  BedroomAbvGr   1460 non-null   int64
 52  KitchenAbvGr   1460 non-null   int64
 53  KitchenQual    1460 non-null   object
 54  TotRmsAbvGrd   1460 non-null   int64
 55  Functional     1460 non-null   object
 56  Fireplaces     1460 non-null   int64
 57  FireplaceQu    770 non-null    object
 58  GarageType     1379 non-null   object
 59  GarageYrBlt    1379 non-null   float64
 60  GarageFinish   1379 non-null   object
 61  GarageCars     1460 non-null   int64
 62  GarageArea     1460 non-null   int64
 63  GarageQual     1379 non-null   object
 64  GarageCond     1379 non-null   object
 65  PavedDrive     1460 non-null   object
 66  WoodDeckSF     1460 non-null   int64
 67  OpenPorchSF    1460 non-null   int64
 68  EnclosedPorch  1460 non-null   int64
 69  3SsnPorch      1460 non-null   int64
 70  ScreenPorch    1460 non-null   int64
 71  PoolArea       1460 non-null   int64
 72  PoolQC         7 non-null      object
 73  Fence          281 non-null    object
 74  MiscFeature    54 non-null     object
 75  MiscVal        1460 non-null   int64
 76  MoSold         1460 non-null   int64
 77  YrSold         1460 non-null   int64
 78  SaleType       1460 non-null   object
 79  SaleCondition  1460 non-null   object
 80  SalePrice      1460 non-null   int64
dtypes: float64(3), int64(35), object(43)
memory usage: 924.0+ KB
```

output from ide python has been incorporated .

- df_train.isnull().sum()

output:

Id	0
MSSubClass	0
MSZoning	0
LotFrontage	259
LotArea	0
Street	0
Alley	1369
LotShape	0
LandContour	0
Utilities	0

LotConfig 0
LandSlope 0
Neighborhood 0
Condition1 0
Condition2 0
BldgType 0
HouseStyle 0
OverallQual 0
OverallCond 0
YearBuilt 0
YearRemodAdd 0
RoofStyle 0
RoofMatl 0
Exterior1st 0
Exterior2nd 0
MasVnrType 8
MasVnrArea 8
ExterQual 0
ExterCond 0
Foundation 0
BsmtQual 37
BsmtCond 37
BsmtExposure 38
BsmtFinType1 37
BsmtFinSF1 0
BsmtFinType2 38
BsmtFinSF2 0
BsmtUnfSF 0

TotalBsmtSF 0
Heating 0
HeatingQC 0
CentralAir 0
Electrical 1
1stFlrSF 0
2ndFlrSF 0
LowQualFinSF 0
GrLivArea 0
BsmtFullBath 0
BsmtHalfBath 0
FullBath 0
HalfBath 0
BedroomAbvGr 0
KitchenAbvGr 0
KitchenQual 0
TotRmsAbvGrd 0
Functional 0
Fireplaces 0
FireplaceQu 690
GarageType 81
GarageYrBlt 81
GarageFinish 81
GarageCars 0
GarageArea 0
GarageQual 81
GarageCond 81
PavedDrive 0

```
WoodDeckSF        0
OpenPorchSF       0
EnclosedPorch     0
3SsnPorch         0
ScreenPorch       0
PoolArea          0
PoolQC         1453
Fence          1179
MiscFeature    1406
MiscVal           0
MoSold            0
YrSold            0
SaleType          0
SaleCondition     0
SalePrice         0
dtype: int64
```

```
In [503…  df_train.isnull().sum()

Out[503…
Id                0
MSSubClass        0
MSZoning          0
LotFrontage     259
LotArea           0
Street            0
Alley          1369
LotShape          0
LandContour       0
Utilities         0
LotConfig         0
LandSlope         0
Neighborhood      0
Condition1        0
Condition2        0
BldgType          0
HouseStyle        0
OverallQual       0
OverallCond       0
YearBuilt         0
YearRemodAdd      0
RoofStyle         0
RoofMatl          0
Exterior1st       0
Exterior2nd       0
MasVnrType        8
MasVnrArea        8
ExterQual         0
ExterCond         0
Foundation        0
BsmtQual         37
BsmtCond         37
BsmtExposure     38
```

in the same way we get the output as mentioned above

The DataFrame.corr() method displays the correlation (or relationship) between the columns. We'll examine the correlations between the features and the target.

cod efor correction:

df_train.corr()

correlation matrix:-

A correlation matrix is a table showing correlation coefficients between variables. Each cell in the table shows the correlation between the two variables. A correlation matrix is used to summarize data, as an input into a more advanced analysis.

#correlation matrix

import matplotlib.pyplot as plt

import seaborn as sns

corrmat = df_train.corr()

f, ax = plt.subplots(figsize=(15, 12))

sns.heatmap(corrmat, vmax=.8, square=True)

In [504...
```
df_train.corr()
```

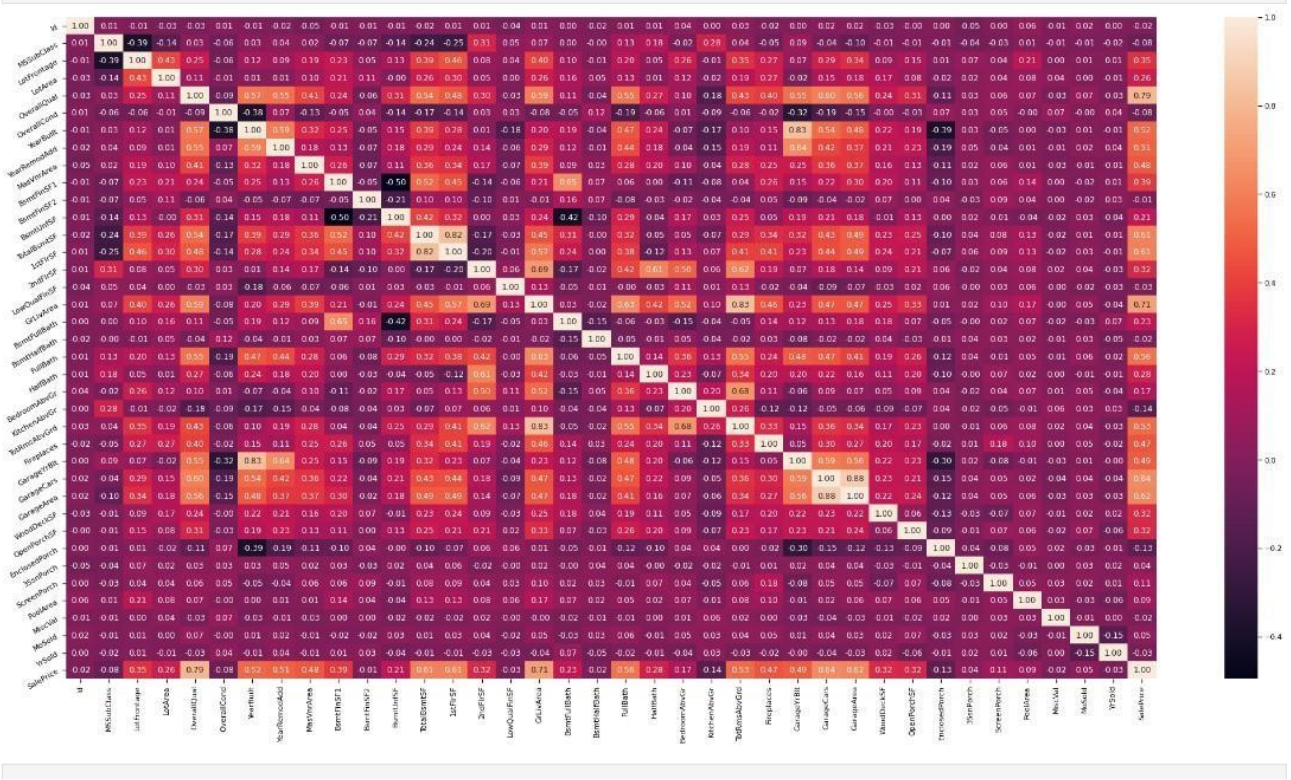
Out[504...

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2	BsmtUnfSF	TotalBsmt
Id	1.000000	0.011156	-0.010601	-0.033226	-0.028365	0.012609	-0.012713	-0.021998	-0.050298	-0.005024	-0.005968	-0.007940	-0.0154
MSSubClass	0.011156	1.000000	-0.386347	-0.139781	0.032628	-0.059316	0.027850	0.040581	0.022936	-0.069836	-0.065649	-0.140759	-0.2385
LotFrontage	-0.010601	-0.386347	1.000000	0.426095	0.251646	-0.059213	0.123349	0.088866	0.193458	0.233633	0.049900	0.132644	0.3920
LotArea	-0.033226	-0.139781	0.426095	1.000000	0.105806	-0.005636	0.014228	0.013788	0.104160	0.214103	0.111170	-0.002618	0.2608
OverallQual	-0.028365	0.032628	0.251646	0.105806	1.000000	-0.091932	0.572323	0.550684	0.411876	0.239666	-0.059119	0.308159	0.5378
OverallCond	0.012609	-0.059316	-0.059213	-0.005636	-0.091932	1.000000	-0.375983	0.073741	-0.128101	-0.046231	0.040229	-0.136841	-0.1710
YearBuilt	-0.012713	0.027850	0.123349	0.014228	0.572323	-0.375983	1.000000	0.592855	0.315707	0.249503	-0.049107	0.149040	0.3914
YearRemodAdd	-0.021998	0.040581	0.088866	0.013788	0.550684	0.073741	0.592855	1.000000	0.179618	0.128451	-0.067759	0.181133	0.2910
MasVnrArea	-0.050298	0.022936	0.193458	0.104160	0.411876	-0.128101	0.315707	0.179618	1.000000	0.264736	-0.072319	0.114442	0.3639
BsmtFinSF1	-0.005024	-0.069836	0.233633	0.214103	0.239666	-0.046231	0.249503	0.128451	0.264736	1.000000	-0.050117	-0.495251	0.5223
BsmtFinSF2	-0.005968	-0.065649	0.049900	0.111170	-0.059119	0.040229	-0.049107	-0.067759	-0.072319	-0.050117	1.000000	-0.209294	0.1048
BsmtUnfSF	-0.007940	-0.140759	0.132644	-0.002618	0.308159	-0.136841	0.149040	0.181133	0.114442	-0.495251	-0.209294	1.000000	0.4153
TotalBsmtSF	-0.015415	-0.238518	0.392075	0.260833	0.537808	-0.171098	0.391452	0.291066	0.363936	0.522396	0.104810	0.415360	1.0000
1stFlrSF	0.010496	-0.251758	0.457181	0.299475	0.476224	-0.144203	0.281986	0.240379	0.344501	0.445863	0.097117	0.317987	0.8195
2ndFlrSF	0.005590	0.307886	0.080177	0.050986	0.295493	0.028942	0.010308	0.140024	0.174561	-0.137079	-0.099260	0.004469	-0.1745
LowQualFinSF	-0.044230	0.046474	0.038469	0.004779	-0.030429	0.025494	-0.183784	-0.062419	-0.069071	-0.064503	0.014807	0.028167	-0.0332
GrLivArea	0.008273	0.074853	0.402797	0.263116	0.593007	-0.079686	0.199010	0.287389	0.390857	0.208171	-0.009640	0.240257	0.4548
BsmtFullBath	0.002289	0.003491	0.100949	0.158155	0.111098	-0.054942	0.187599	0.119470	0.085310	0.649212	0.158678	-0.422900	0.3073
BsmtHalfBath	-0.020155	-0.002333	-0.007234	0.048046	-0.040150	0.117821	-0.038162	-0.012337	0.026673	0.067418	0.070948	-0.095804	-0.0003
FullBath	0.005587	0.131608	0.198769	0.126031	0.550600	-0.194149	0.468271	0.439046	0.276833	0.058543	-0.076444	0.288886	0.3237

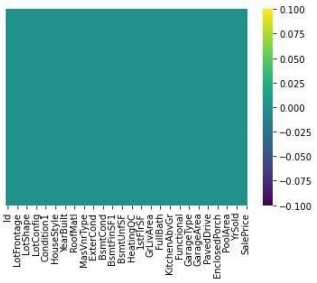
LowQualFinSF	-0.044230	0.046474	0.038469	0.004779	-0.030429	0.025494	-0.183784	-0.062419	-0.069071	-0.064503	0.014807	0.028167	-0.0332
GrLivArea	0.008273	0.074853	0.402797	0.263116	0.593007	-0.079686	0.199010	0.287389	0.390857	0.208171	-0.009640	0.240257	0.4548
BsmtFullBath	0.002289	0.003491	0.100949	0.158155	0.111098	-0.054942	0.187599	0.119470	0.085310	0.649212	0.158678	-0.422900	0.3073
BsmtHalfBath	-0.020155	-0.002333	-0.007234	0.048046	-0.040150	0.117821	-0.038162	-0.012337	0.026673	0.067418	0.070948	-0.095804	-0.0003
FullBath	0.005587	0.131608	0.198769	0.126031	0.550600	-0.194149	0.468271	0.439046	0.276833	0.058543	-0.076444	0.288886	0.3237
HalfBath	0.006784	0.177354	0.053532	0.014259	0.273458	-0.060769	0.242656	0.183331	0.201444	0.004262	-0.032148	-0.041118	-0.0488
BedroomAbvGr	0.037719	-0.023438	0.263170	0.119690	0.101676	0.012980	-0.070651	-0.040581	0.102821	-0.107355	-0.015728	0.166643	0.0504
KitchenAbvGr	0.002951	0.281721	-0.006069	-0.017784	-0.183882	-0.087001	-0.174800	-0.149598	-0.037610	-0.081007	-0.040751	0.030086	-0.0689
TotRmsAbvGrd	0.027239	0.040380	0.352096	0.190015	0.427452	-0.057583	0.095589	0.191740	0.280682	0.044316	-0.035227	0.250647	0.2855
Fireplaces	-0.019772	-0.045569	0.266639	0.271364	0.396765	-0.023820	0.147716	0.112581	0.249070	0.260011	0.046921	0.051575	0.3395
GarageYrBlt	0.000072	0.085072	0.070250	-0.024947	0.547766	-0.324297	0.825667	0.642277	0.252691	0.153484	-0.088011	0.190708	0.3224
GarageCars	0.016570	-0.040110	0.285691	0.154871	0.600671	-0.185758	0.537850	0.420622	0.364204	0.224054	-0.038264	0.214175	0.4345
GarageArea	0.017634	-0.098672	0.344997	0.180403	0.562022	-0.151521	0.478954	0.371600	0.373066	0.296970	-0.018227	0.183303	0.4866
WoodDeckSF	-0.029643	-0.012579	0.088521	0.171698	0.238923	-0.003334	0.224880	0.205726	0.159718	0.204306	0.067898	-0.005316	0.2320
OpenPorchSF	-0.000477	-0.006100	0.151972	0.084774	0.308819	-0.032589	0.188686	0.226298	0.125703	0.111761	0.003093	0.129005	0.2472
EnclosedPorch	0.002889	-0.012037	0.010700	-0.018340	-0.113937	0.070356	-0.387268	-0.193919	-0.110204	-0.102303	0.036543	-0.002538	-0.0954
3SsnPorch	-0.046635	-0.043825	0.070029	0.020423	0.030371	0.025504	0.031355	0.045286	0.018796	0.026451	-0.029993	0.020764	0.0373
ScreenPorch	0.001330	-0.026030	0.041383	0.043160	0.064886	0.054811	-0.050364	-0.038740	0.061466	0.062021	0.088871	-0.012579	0.0844
PoolArea	0.057044	0.008283	0.206167	0.077672	0.065166	-0.001985	0.004950	0.005829	0.011723	0.140491	0.041709	-0.035092	0.1260
MiscVal	-0.006242	-0.007683	0.003368	0.038068	-0.031406	0.068777	-0.034383	-0.010286	-0.029815	0.003571	0.004940	-0.023837	-0.0184
MoSold	0.021172	-0.013585	0.011200	0.001205	0.070815	-0.003511	0.012398	0.021490	-0.005965	-0.015727	-0.015211	0.034888	0.0131
YrSold	0.000712	-0.021407	0.007450	-0.014261	-0.027347	0.043950	-0.013618	0.035743	-0.008201	0.014359	0.031706	-0.041258	-0.0149
SalePrice	-0.021917	-0.084284	0.351799	0.263843	0.790982	-0.077856	0.522897	0.507101	0.477493	0.386420	-0.011378	0.214479	0.6135

```
corr = df_train.corr()

sns.set_context("notebook", font_scale=1.0, rc={"lines.linewidth": 2.5})

plt.figure(figsize=(36,18))

a = sns.heatmap(corr, annot=True, fmt='.2f')

rotx = a.set_xticklabels(a.get_xticklabels(), rotation=90)

roty = a.set_yticklabels(a.get_yticklabels(), rotation=30)
```

output:



```
In [32]: data.isnull().sum()

Out[32]: Id              0
         MSSubClass      0
         MSZoning        0
         LotFrontage     0
         LotArea         0
                        ..
         MoSold          0
         YrSold          0
         SaleType        0
         SaleCondition   0
         SalePrice       0
         Length: 76, dtype: int64

In [33]: sns.heatmap(data.isnull() , yticklabels = False , cmap = 'viridis')

Out[33]: <AxesSubplot:>
```



handling the missing values:

Handling missing values

data['LotFrontage'] = data['LotFrontage'].fillna(data['LotFrontage'].mean())

data['BsmtFinSF1'] = data['BsmtFinSF1'].fillna(data['BsmtFinSF1'].mean())

```
data['BsmtFinSF2'] = data['BsmtFinSF2'].fillna(data['BsmtFinSF2'].mean())
data['BsmtUnfSF'] = data['BsmtUnfSF'].fillna(data['BsmtUnfSF'].mean())
data['TotalBsmtSF'] = data['TotalBsmtSF'].fillna(data['TotalBsmtSF'].mean())
data['GarageCars'] = data['GarageCars'].fillna(data['GarageCars'].mean())
data['GarageArea'] = data['GarageArea'].fillna(data['GarageArea'].mean())


data.drop(['Alley' , 'PoolQC' , 'Fence' , 'MiscFeature'], axis = 1 , inplace = True)
data.drop(['GarageYrBlt'], axis = 1 , inplace = True)


data['MSZoning'] = data['MSZoning'].fillna(data['MSZoning'].mode()[0])
data['MasVnrType'] = data['MasVnrType'].fillna(data['MasVnrType'].mode()[0])
data['BsmtQual'] = data['BsmtQual'].fillna(data['BsmtQual'].mode()[0])
data['BsmtCond'] = data['BsmtCond'].fillna(data['BsmtCond'].mode()[0])
data['BsmtExposure'] = data['BsmtExposure'].fillna(data['BsmtExposure'].mode()[0])
data['BsmtFinType1'] = data['BsmtFinType1'].fillna(data['BsmtFinType1'].mode()[0])
data['BsmtFinType2'] = data['BsmtFinType2'].fillna(data['BsmtFinType2'].mode()[0])
data['FireplaceQu'] = data['FireplaceQu'].fillna(data['FireplaceQu'].mode()[0])
data['GarageType'] = data['GarageType'].fillna(data['GarageType'].mode()[0])
data['GarageFinish'] = data['GarageFinish'].fillna(data['GarageFinish'].mode()[0])
data['GarageQual'] = data['GarageQual'].fillna(data['GarageQual'].mode()[0])
data['GarageCond'] = data['GarageCond'].fillna(data['GarageCond'].mode()[0])
data['SaleType'] = data['SaleType'].fillna(data['SaleType'].mode()[0])
data['Utilities'] = data['Utilities'].fillna(data['Utilities'].mode()[0])
data['Exterior1st'] = data['Exterior1st'].fillna(data['Exterior1st'].mode()[0])
data['Exterior2nd'] = data['Exterior2nd'].fillna(data['Exterior2nd'].mode()[0])
data['Electrical'] = data['Electrical'].fillna(data['Electrical'].mode()[0])
data['Functional'] = data['Functional'].fillna(data['Functional'].mode()[0])
data['MasVnrArea'] = data['MasVnrArea'].fillna(data['MasVnrArea'].mode()[0])
data['BsmtFullBath'] = data['BsmtFullBath'].fillna(data['BsmtFullBath'].mode()[0])
data['BsmtHalfBath'] = data['BsmtHalfBath'].fillna(data['BsmtHalfBath'].mode()[0])
data['KitchenQual'] = data['KitchenQual'].fillna(data['KitchenQual'].mode()[0])
```

code in ide:

Handling missing values

In [26]:
```
data['LotFrontage'] = data['LotFrontage'].fillna(data['LotFrontage'].mean())
data['BsmtFinSF1'] = data['BsmtFinSF1'].fillna(data['BsmtFinSF1'].mean())
data['BsmtFinSF2'] = data['BsmtFinSF2'].fillna(data['BsmtFinSF2'].mean())
data['BsmtUnfSF'] = data['BsmtUnfSF'].fillna(data['BsmtUnfSF'].mean())
data['TotalBsmtSF'] = data['TotalBsmtSF'].fillna(data['TotalBsmtSF'].mean())
data['GarageCars'] = data['GarageCars'].fillna(data['GarageCars'].mean())
data['GarageArea'] = data['GarageArea'].fillna(data['GarageArea'].mean())
```

In [27]:
```
data.drop(['Alley' , 'PoolQC' , 'Fence' , 'MiscFeature'], axis = 1 , inplace = True)
data.drop(['GarageYrBlt'], axis = 1 , inplace = True)
```

In [28]:
```
data['MSZoning'] = data['MSZoning'].fillna(data['MSZoning'].mode()[0])
data['MasVnrType'] = data['MasVnrType'].fillna(data['MasVnrType'].mode()[0])
data['BsmtQual'] = data['BsmtQual'].fillna(data['BsmtQual'].mode()[0])
data['BsmtCond'] = data['BsmtCond'].fillna(data['BsmtCond'].mode()[0])
data['BsmtExposure'] = data['BsmtExposure'].fillna(data['BsmtExposure'].mode()[0])
data['BsmtFinType1'] = data['BsmtFinType1'].fillna(data['BsmtFinType1'].mode()[0])
data['BsmtFinType2'] = data['BsmtFinType2'].fillna(data['BsmtFinType2'].mode()[0])
data['FireplaceQu'] = data['FireplaceQu'].fillna(data['FireplaceQu'].mode()[0])
data['GarageType'] = data['GarageType'].fillna(data['GarageType'].mode()[0])
data['GarageFinish'] = data['GarageFinish'].fillna(data['GarageFinish'].mode()[0])
data['GarageQual'] = data['GarageQual'].fillna(data['GarageQual'].mode()[0])
data['GarageCond'] = data['GarageCond'].fillna(data['GarageCond'].mode()[0])
data['SaleType'] = data['SaleType'].fillna(data['SaleType'].mode()[0])
data['Utilities'] = data['Utilities'].fillna(data['Utilities'].mode()[0])
data['Exterior1st'] = data['Exterior1st'].fillna(data['Exterior1st'].mode()[0])
data['Exterior2nd'] = data['Exterior2nd'].fillna(data['Exterior2nd'].mode()[0])
data['Electrical'] = data['Electrical'].fillna(data['Electrical'].mode()[0])
data['Functional'] = data['Functional'].fillna(data['Functional'].mode()[0])
data['MasVnrArea'] = data['MasVnrArea'].fillna(data['MasVnrArea'].mode()[0])
data['BsmtFullBath'] = data['BsmtFullBath'].fillna(data['BsmtFullBath'].mode()[0])
data['BsmtHalfBath'] = data['BsmtHalfBath'].fillna(data['BsmtHalfBath'].mode()[0])
data['KitchenQual'] = data['KitchenQual'].fillna(data['KitchenQual'].mode()[0])
```

Thus, module 2 codes and result has been incorporated .In upcoming modules we will be implementing the regression which suits the project more accurate and gives the appropriate values that is the house price prediction in that particulararеas called ames and lowa. handling the train dataset test data set willbe implemented in the module 3

Aim:

To implement Module 3 of the project and display the output of the module with solving New Issues.

Software Used

C, C++, Python, HTML, Mysql, Etc...

Code of Module 3

Result of Module 3

In previous as we discussed about the stages of the project here comes the final stage of the project thats the module 3 implementation of suitable regression for predicting the price of the house

here is the breif recap of module 1 and 2:

- requirement analysis:

importing the libraries

concatenating both the datasets or loading data sets

- data processing:

handling the missing values

handling the datasets

- 2.2.1 handling the train dataset
- 2.2.2 handling the test dataset

here in this project we have come to last module of the project that is implementation evaluation of the project

project: HOUSE PRICE PREDICTION USING MACHINE LEARNING AND PYTHON

As many weeks passed lets have a quick recap about the project so that it will be very useful even for the layman to understand whats happeningin this module

Predicting house prices can help to determine the selling price of a house of a particular region and can help people to find the right time to buy a home. In this article, I will introduce you to a machine learning project on house price prediction with Python.

Machine learning plays a major role from past years in image detection, spam reorganization, normal speech command, product recommendation and medical diagnosis. Present machine learning algorithm helps us in enhancing security alerts, ensuring public safety and improve medical enhancements. Machine learning system also provides better customer service and safer automobile systems. In the present paper we discuss about the prediction of future housing prices that is generated by machine learning algorithm. For the selection of prediction methods we compare and explore various prediction methods. We utilize lasso regression as our model because of its adaptable and probabilistic methodology on model selection. Our result exhibit that our approach of the issue need to be successful, and has the ability to process predictions that would be comparative with other house cost prediction models. More over on other hand housing value indices, the advancement of a housing cost prediction that tend to the advancement of real estate policies schemes. This study utilizes machine learning algorithms as a research method that develops housing price prediction models. We create a housing cost prediction model In view of machine learning algorithm models for example, XGBoost, lasso regression and neural system on look at their order precision execution. We in that point recommend a housing cost prediction model to support a house vender or a real estate agent for better information based on the valuation of house. Those examinations exhibit that lasso regression algorithm, in view of accuracy, reliably outperforms alternate models in the execution of housing cost prediction.What Is Learning? Rats Learning to Avoid Poisonous Baits: Rats normally stumble upon food items by its look and smell and start eating in small amounts and later depending on food and physiological effect the feeding of food goes on. If the rat notices the illness of the food, the rat will not touch that food. Similarly the machine learning mechanism plays a vital role same as animal usage of past experience for acquiring and expertise in detecting the food safety. By mistake if the knowledge with the food is negatively labeled, the prediction of the animal will also will be negatively affected and encountered in the future. With the inspiration of the previous example of successful learning we demonstrate a typical machine learning algorithm .Likewise we would want to program a machine that learns how to filter spam e-mails. A credulous result might be apparently comparable of the lifestyle of rats that, how to keep away from poisonous baits. The machine will basically remember the past e-mails that needed been named similarly as spam e-mails by the human user. When another email arrives, the machine will look for it in the past set about spam e-mails. Though it matches among them, it will be trashed. Otherwise, it will make moved of the user’s inbox organizer. At the same time the first “learning by memorization” methodology may be useful, it fails to

offer an important aspect known as learning systems – the capacity to mark unseen email messages. A fruitful learner ought to have the ability which will be the advancement from distinctive samples to more extensive generalization. This may be Likewise as inductive thinking or inductive induction. In the attraction nervousness exhibited previously, after the rats experience a sample of a certain sort about food; they apply their disposition at it once new, unseen illustrations from claiming nourishment of comparable emanation Also taste. Should attain generalization in the spam sifting task, those learner might examine the Awhile ago seen e-mails, and extricate An situated about expressions whose presence for a email message is characteristic of spam. Then, At another email arrives, those machine could weigh if a standout among the suspicious expressions gives the idea On it, and foresee its mark Appropriately. Such an arrangement might possibly have the ability effectively to foresee the name about unseen e-mails. Responsibilities further than Human Capabilities: an additional totally crew about errands that profit starting with machine Taking in systems are identified with the Investigation for extremely substantial and intricate information sets: galactic data, turning restorative chronicles under restorative knowledge, climate prediction, and dissection of genomic data, Web serch engines, Also electronic trade. With an ever increasing amount accessible digitally recorded data, it gets evident that there would treasures about serious majority of the data covered clinched alongside information chronicles that would best approach excessively little and also as well perplexing to people with bode well about. Taking in with recognize serious examples over substantial Also complex information sets may be a guaranteeing space for which the blending of projects that take for the Just about boundless memory limit and ever expanding transforming velocity about PCs opens up new horizons.

little introduction about the similar projects which gives clear picture about the present project (house price prediction):

A.1:

Stock Market Prediction Using Bayesian-Regularized Neural Networks:

In a study done by Ticknor (2013), he used Bayesian regularized articial neural network to predict the future operation of financial market. Specifically, he built a model to predict future stock prices. The input of the model is previous stock statistics in addition to some financial technical data. The output of the model is the next-day closing price of the corresponding stocks. The model proposed in the study is built using Bayesian regularized neural network. The weights of this type of networks are given a probabilistic nature. This allows the network to penalize very complex models (with many hidden layers) in an automatic manner. This in turn will reduce the overfitting of the model. The model consists of a feedforward neural network which has three layers: an input layer, one hidden layer, and an output layer. The author chose the number of neurons in the hidden layer based on experimental methods.The input data of the model is normalized to be between -1 and 1, and this opertion is reversed for the output so the predicted price appears in the appropriate scale.

A.2:Stock Market Prediction Using A Machine Learning Model:

In another study done by Hegazy, Soliman, and Salam (2014), a system was proposed to predict daily stock market prices. The system combines particle swarm optimization (PSO) and least square support vector machine (LS-SVM), where PSO was used to optimize LV-SVM. The authors claim that in most cases, artificial neural networks (ANNs) are subject to the overfitting problem. They state that support vector machines algorithm (SVM) was developed as an alternative that doesn't suffer from overfitting. They attribute this advantage to SVMs being based on the solid foundations of VC-theory. They further elaborate that LS-SVM method was reformulation of traditional SVM method that uses a regularized least squares function with equality.

A.3:House Price Prediction Using Multilevel Model and Neural Networks:

A different study was done by Feng and Jones (2015) to preduct house prices. Two models were built: a multilevel model (MLM) and an artificial neural network model (ANN). These two models were compared to each other and to a hedonic price model (HPM). The multilevel model integrates the micro-level that specifies the relationships between houses within a given neighbourhood, and the macro-level equation which specifies the relationships between neighbouhoods. The hedonic price model is a model that estimates house prices using 6 Figure 3: MSE comparison some attributes such as the number of bedrooms in the house, the size of the house, etc. The data used in the study contains house prices in Greater Bristol area between 2001 and 2013. Secondary data was obtained from the Land Registry, the Population Census and Neighbourhood Statistics to be used in order to make the models suitable for national usage. The authors listed many reasons on why they chose the Greater Bristol area such as its diverse urban and rural blend and its different property types. Each record in the dataset contains data about a house in the area: it contains the address, the unit postcode, property type, the duration (freehold or leasehold), the sale price, the date of the sale, and whether the house was newly-built when it was sold. In total, the dataset contains around 65,000 entries. To enable model training and testing, the dataset was divided into a training set that contains data about house sales from 2001 to 2012, and a test set that contains data about house sales in 2013.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..

This the breif introduction about the machine learning projects which will be useful for this projects.

for importing libraries we used requirement analysis step

for data pre processing and data processing we used simple techinque called data loading which hleps us to load the data sets i.e is train data set and test data set

for handling the missing values and trai data sets and test dats sets all the codes and were shared in module 2 which also be useful for understanding the module 3 thats the last module for house price prediction using machine learning and pyhton project

for data evaluavtion we will setting the algorithm

lets discuss som regressions we have in machine learning:

Multiple Linear Regression:

Multiple Linear Regression (MLR) is a supervised technique used to estimate the relationship between one dependent variable and more than one independent variables. Identifying the correlation and its cause-effect helps to make predictions by using these relations . To estimate these relationships, the prediction accuracy of the model is essential; the complexity of the model is of more interest. However, Multiple Linear Regression is prone to many problems such as multicollinearity, noises, and overfitting, which effect on the prediction accuracy. Regularised regression plays a significant part in Multiple Linear Regression because it helps to reduce variance at the cost of introducing some bias, avoid the overfitting problem and solve ordinary least squares (OLS) problems. There are two types of regularisation techniques L1 norm (least absolute deviations) and L2 norm (least squares). L1 and L2 have different cost functions regarding model complexity

Lasso Regression:

Least Absolute Shrinkage and Selection Operator (Lasso) is an L1-norm regularised regression technique that was formulated by Robert Tibshirani in 1996. Lasso is a powerful technique that performs regularisation and feature selection. Lasso introduces a bias term, but instead of squaring the slope like Ridge regression, the absolute value of the slope is added as a penalty term. Lasso is defined as: □ = □□(□□□ □□ □□□□□□□□ □□□□□□□□□ + □ ∗ |□□□□□|) (1) Where □□□(□□□ □□ □□□□□□□□ □□□□□□□□□) is the Least Squared Error, and □ ∗ |□□□□□| is the penalty term. However, alpha □ is the tuning parameter which controls the strength of the penalty term. In other words, the tuning parameter is the value of shrinkage. |□□□□□| is the sum of the absolute value of the coefficients

Ridge Regression:

The Ridge Regression is an L2-norm regularised regression technique that was introduced by Hoerl in 1962 . It is an estimation procedure to manage collinearity without removing variables from the regression model. In multiple linear regression, the multicollinearity is a common problem that leads least square estimation to be unbiased, and its variances are far from the correct value. Therefore, by adding a degree of bias to the regression model, Ridge Regression reduces the standard errors, and it shrinks the least square coefficients towards the origin of the parameter space . Ridge formula is: □ = □□(□□□ □□ □□□□□□□□ □□□□□□□□□ + □ ∗ □□□□□ 2)

Random Forest Regression:

A Random Forest is an ensemble technique qualified for performing classification and regression tasks with the help of multiple decision trees and a method called Bootstrap Aggregation known as Bagging.Decision Trees are used in classification and regression tasks, where the model (tree) is formed of nodes and branches. The tree starts with a root node, while the internal nodes correspond to an input attribute. The nodes that do not have children are called leaves, where each leaf performs the prediction of the output variable

Artificial Neural Network:

Artificial neural network (ANN) is an attempt to simulate the work of a biological brain. The brain learns and evolves through the experiments that it faces through time to make decisions and predict the result of particular actions. Thus, ANN tries to simulate the brain to learn the pattern in a given data to predict the output of that data whether the expected data was provided in the learning process or not [17]. ANN is based on an assemblage of connected elements or nodes called neurons. Neurons act as channels that take an input, process it, and then pass it to other neurons for further processing. This transaction or the process of transferring data between neurons is handled in layers. Layers consist of at least three layers, input layer, one or more of hidden layers and output layer. Each layer holds a set of neurons that takes input and process data and finally pass the output to other neurons in the next layer. This process is repetitive until the output layer has been reached, so 7 eventually, the result can be presented. ANN architecture is shown in the following figure as is also known as feed-forward, which values pass in one direction.

here comes the coding part of module 3 which includes output and result as well:

Handling Categorical Features

columns=['MSZoning','Street','LotShape','LandContour','Utilities','LotConfig','LandSlope','Neighborhood',

'Condition2','BldgType','Condition1','HouseStyle','SaleType','SaleCondition','ExterCond',

'ExterQual','Foundation','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2',

'RoofStyle','RoofMatl','Exterior1st','Exterior2nd','MasVnrType','Heating','HeatingQC','CentralAir',

```
'Electrical','KitchenQual','Functional','FireplaceQu','GarageType','GarageFinish','GarageQual',
'GarageCond','PavedDrive']

len(columns)

def category(multcolumns):
    data_final=data
    i=0
    for field in multcolumns:

        df1=pd.get_dummies(data[field],drop_first=True)


        data.drop([field],axis=1,inplace=True)
        if i==0:
            data_final=df1.copy()
        else:

            data_final=pd.concat([data_final,df1],axis=1)
        i=i+1



    data_final=pd.concat([data,data_final],axis=1)


    return data_final
data = category(columns)

data.shape

data = data.loc[:,~data.columns.duplicated()]



data.shape
```

(2919, 177)

data.head()

Handling Categorical Features

In [35]:
```
columns=['MSZoning','Street','LotShape','LandContour','Utilities','LotConfig','LandSlope','Neighborhood',
         'Condition2','BldgType','Condition1','HouseStyle','SaleType','SaleCondition','ExterCond',
         'ExterQual','Foundation','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2',
        'RoofStyle','RoofMatl','Exterior1st','Exterior2nd','MasVnrType','Heating','HeatingQC','CentralAir',
         'Electrical','KitchenQual','Functional','FireplaceQu','GarageType','GarageFinish','GarageQual','GarageCond','PavedDrive']
```

In [36]:
```
len(columns)
```

Out[36]: 39

In [37]:
```
def category(multcolumns):
    data_final=data
    i=0
    for field in multcolumns:

        df1=pd.get_dummies(data[field],drop_first=True)

        data.drop([field],axis=1,inplace=True)
        if i==0:
            data_final=df1.copy()
        else:

            data_final=pd.concat([data_final,df1],axis=1)
        i=i+1


    data_final=pd.concat([data,data_final],axis=1)

    return data_final
```

In [38]:
```
data = category(columns)
```

In [39]:
```
data.shape
```

In [39]:
```
data.shape
```

Out[39]:
```
(2919, 237)
```

In [40]:
```
data = data.loc[:,~data.columns.duplicated()]
```

In [41]:
```
data.shape
```

Out[41]:
```
(2919, 177)
```

In [42]:
```
data.head()
```

Out[42]:

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	...	Min1	Min2	Typ	Attchd	Basment	BuiltIn	CarF
0	1	60	65.0	8450	7	5	2003	2003	196.0	706.0	...	0	0	1	1	0	0	
1	2	20	80.0	9600	6	8	1976	1976	0.0	978.0	...	0	0	1	1	0	0	
2	3	60	68.0	11250	7	5	2001	2002	162.0	486.0	...	0	0	1	1	0	0	
3	4	70	60.0	9550	7	5	1915	1970	0.0	216.0	...	0	0	1	0	0	0	
4	5	60	84.0	14260	8	5	2000	2000	350.0	655.0	...	0	0	1	1	0	0	

5 rows × 177 columns

In [43]:
```
data_train = data.iloc[:1460,:]
data_test = data.iloc[1460:,:]
```

In [44]:
```
data_test.drop(['SalePrice'] , axis = 1 , inplace = True)
```

```
C:\Users\User\anaconda3\lib\site-packages\pandas\core\frame.py:4308: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  return super().drop(
```

```
data_train = data.iloc[:1460,:]

data_test = data.iloc[1460:,:]

data_test.drop(['SalePrice'] , axis = 1 , inplace = True)

y_train = data_train['SalePrice']

x_train = data_train.drop(['SalePrice'] , axis = 1)
```

```
x_train.shape
```

```
(1460, 176)
```

```
from sklearn.preprocessing import StandardScaler

scalar = StandardScaler()
```

```
x_train = scalar.fit_transform(x_train)
data_test = scalar.transform(data_test)

from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)

LinearRegression()

y_tr = lr.predict(x_train)
lr.coef_

array([ 4.41411789e+02, -2.68262474e+03,  2.23943826e+03,  5.54471919e+03,
        1.27647032e+04, 6.64048439e+03, 9.76742938e+03, 2.05814564e+03,
        5.45121738e+03, -2.56247951e+16, -9.06334272e+15, -2.48252523e+16,
        2.46476236e+16,  2.70292482e+16,  3.05209770e+16,  3.39960431e+15,
       -3.67402748e+16, 2.70978715e+02, 1.33046507e+01, 2.09621577e+03,
        1.14483144e+03, -6.23384883e+03, -3.81353001e+03,  4.19953273e+03,
        1.70711261e+03,  2.07868639e+03,  2.99520823e+03,  1.43060556e+03,
        1.11601863e+03,  2.45432582e+02,  1.22188497e+03,  2.09857083e+03,
        2.97539978e+03,  5.35816627e+01, -1.71384258e+03,  2.99235140e+02,
        5.11713278e+03,  2.38693853e+03,  9.67616849e+03,  6.72159920e+03,
        1.59005088e+03,  1.31123496e+03, -3.48316707e+01,  3.22994639e+02,
        2.24376275e+03, -1.00494647e+03,  1.50748337e+03, -9.35483257e+02,
        2.16530614e+03, -1.02330871e+03, -8.90848262e+02,  1.23318189e+02,
        1.33203159e+03, -1.99603909e+03,  3.90240771e+02,  1.61158578e+03,
        1.55877389e+03, -1.34407638e+03, -1.97044981e+03,  2.38937845e+03,
```

-2.19928618e+03, -2.52219918e+03, -3.23991404e+01, -2.39843836e+01,
-2.11089907e+03, -3.30220265e+03, 1.41552758e+03, -3.69381470e+03,
4.16283306e+03, 8.75788625e+03, -1.56371669e+03, -3.21452540e+02,
-1.30310259e+03, -6.79703611e+02, 1.42886175e+03, 5.55043540e+03,
-6.03907356e+02, 4.60515339e+02, -6.96850384e+02, -3.66365826e+02,
6.13802090e+02, -8.85430798e+03, -2.96487018e+03, -1.45470957e+02,
-3.12614071e+02, 4.96395686e+02, -5.69663338e+02, -3.40698258e+03,
-4.42401895e+03, -4.04247156e+02, 1.08247083e+03, 6.68162042e+03,
-1.25698441e+03, -6.60778862e+02, -1.29058234e+03, 8.00459080e+02,
1.85542490e+03, 9.81681729e+02, 9.72213846e+02, 1.11889178e+03,
6.39516405e+02, -3.54090993e+01, 7.56174084e+03, 7.39633478e+02,
5.08832522e+02, 9.31205318e+02, 3.34532495e+00, -1.96101915e+02,
2.01103910e+03, -1.19577978e+02, -7.43556606e+01, -2.93468895e+03,
2.10593444e+02, -2.17049051e+03, 3.10674245e+02, 1.45688895e+03,
1.85116481e+03, 2.44480514e+02, -1.75767523e+03, -3.40392265e+03,
-2.45518287e+02, 2.41922361e+03, -5.77816011e+03, 1.37511478e+02,
-2.76743445e+03, 1.81407958e+03, 3.53029973e+02, 3.75423749e+02,
3.34392410e+03, 1.30306179e+03, 2.94050329e+03, 8.94190327e+04,
1.96857878e+04, 1.89280717e+04, 1.71861688e+04, 5.79590349e+04,
3.76726387e+04, 4.80043645e+04, -1.40803133e+02, -3.72845469e+02,
1.22757562e+03, -1.16687641e+03, -3.85024463e+03, -7.19151209e+02,
-1.16415390e+03, -3.34996242e+03, -5.04564577e+02, -2.96993869e+03,
-2.25884738e+03, -7.31361301e+01, 3.36078201e+02, 2.48389869e+03,
-1.01798112e+03, -1.35155201e+03, 2.67944088e+03, 9.26842211e+02,
-1.03136540e+01, 1.00754003e+02, -1.02199651e+03, 8.37661528e+02,
-5.61698413e+02, -1.30226861e+02, -5.47283852e+02, 2.72644254e+01,

-5.28991030e+02, -2.19433696e+02, 8.71706822e+02, 1.50911114e+03,

4.60582268e+03, 4.90910632e+03, 1.51514931e+03, 2.73940671e+03,

3.40676289e+02, 4.99030882e+03, -2.92426208e+03, -4.62472539e+02])

lr.intercept_

output:

180906.6299657845

```
y_pred = lr.predict(data_test)
```

```
l1 = sol
l2 = list(y_pred)
df = pd.DataFrame(list(zip(l1,l2)), columns = ['id' , 'SalePrice'])
```

```
df.to_csv('FinalResult.csv' , index = False)
```

code in jupyter notebook:

```
C:\Users\User\anaconda3\lib\site-packages\pandas\core\frame.py:4308: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  return super().drop(
```

In [45]:
```
y_train = data_train['SalePrice']
x_train = data_train.drop(['SalePrice'] , axis = 1)
```

In [47]:
```
x_train.shape
```

Out[47]:
```
(1460, 176)
```

In [48]:
```
from sklearn.preprocessing import StandardScaler
scalar = StandardScaler()
x_train = scalar.fit_transform(x_train)
data_test = scalar.transform(data_test)
```

In [50]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
```

Out[50]:
```
LinearRegression()
```

In [51]:
```
y_tr = lr.predict(x_train)
lr.coef_
```

Out[51]:
```
array([ 4.41411789e+02, -2.68262474e+03,  2.23943826e+03,  5.54471919e+03,
        1.27647032e+04,  6.64048439e+03,  9.76742938e+03,  2.05814564e+03,
        5.45121738e+03, -2.56247951e+16, -9.06334272e+15, -2.48252523e+16,
        2.46476236e+16,  2.70292482e+16,  3.05209770e+16,  3.39960431e+15,
       -3.67402748e+16,  2.70978715e+02,  1.33046507e+01,  2.09621577e+03,
        1.14483144e+03, -6.23384883e+03, -3.81353001e+03,  4.19953273e+03,
        1.70711261e+03,  2.07868639e+03,  2.99520823e+03,  1.43060556e+03,
        1.11601863e+03,  2.45432582e+02,  1.22188497e+03,  2.09857083e+03,
        2.97539978e+03,  5.35816627e+01, -1.71384258e+03,  2.99235140e+02,
        5.11713278e+03,  2.38693853e+03,  9.67616849e+03,  6.72159920e+03,
        1.59005088e+03,  1.31123496e+03, -3.48316707e+01,  3.22994639e+02,
```

In [51]:
```
y_tr = lr.predict(x_train)
lr.coef_
```

Out[51]:
```
array([ 4.41411789e+02, -2.68262474e+03,  2.23943826e+03,  5.54471919e+03,
        1.27647032e+04,  6.64048439e+03,  9.76742938e+03,  2.05814564e+03,
        5.45121738e+03, -2.56247951e+16, -9.06334272e+15, -2.48252523e+16,
        2.46476236e+16,  2.70292482e+16,  3.05209770e+16,  3.39960431e+15,
       -3.67402748e+16,  2.70978715e+02,  1.33046507e+01,  2.09621577e+03,
        1.14483144e+03, -6.23384883e+03, -3.81353001e+03,  4.19953273e+03,
        1.70711261e+03,  2.07868639e+03,  2.99520823e+03,  1.43060556e+03,
        1.11601863e+03,  2.45432582e+02,  1.22188497e+03,  2.09857083e+03,
        2.97539978e+03,  5.35816627e+01, -1.71384258e+03,  2.99235140e+02,
        5.11713278e+03,  2.38693853e+03,  9.67616849e+03,  6.72159920e+03,
        1.59005088e+03,  1.31123496e+03, -3.48316707e+01,  3.22994639e+02,
        2.24376275e+03, -1.00494647e+03,  1.50748337e+03, -9.35483257e+02,
        2.16530614e+03, -1.02330871e+03, -8.90848262e+02,  1.23318189e+02,
        1.33203159e+03, -1.99603909e+03,  3.90240771e+02,  1.61158578e+03,
        1.55877389e+03, -1.34407638e+03, -1.97044981e+03,  2.38937845e+03,
       -2.19928618e+03, -2.52219918e+03, -3.23991404e+01, -2.39843836e+01,
       -2.11089907e+03, -3.30220265e+03,  1.41552758e+03, -3.69381470e+03,
        4.16283306e+03,  8.75788625e+03, -1.56371669e+03, -3.21452540e+02,
       -1.30310259e+03, -6.79703611e+02,  1.42886175e+03,  5.55043540e+03,
       -6.03907356e+02,  4.60515339e+02, -6.96850384e+02, -3.66365826e+02,
        6.13802090e+02, -8.85430798e+03, -2.96487018e+03, -1.45470957e+02,
       -3.12614071e+02,  4.96395686e+02, -5.69663338e+02, -3.40698258e+03,
       -4.42401895e+03, -4.04247156e+02,  1.08247083e+03,  6.68162042e+03,
       -1.25698441e+03, -6.60778862e+02, -1.29058234e+03,  8.00459080e+02,
        1.85542490e+03,  9.81681729e+02,  9.72213846e+02,  1.11889178e+03,
        6.39516405e+02, -3.54090993e+01,  7.56174084e+03,  7.39633478e+02,
        5.08832522e+02,  9.31205318e+02,  3.34532495e+00, -1.96101915e+02,
        2.01103910e+03, -1.19577978e+02, -7.43556606e+01, -2.93468895e+03,
        2.10593444e+02, -2.17049051e+03,  3.10674245e+02,  1.45688895e+03,
        1.85116481e+03,  2.44480514e+02, -1.75767523e+03, -3.40392265e+03,
       -5.77816011e+03,  2.45518287e+02,  2.41922361e+03, -1.37511478e+02,
       -3.75423749e+02,  2.76743445e+03,  1.81407958e+03,  3.53029973e+02,
        3.34392410e+03,  1.30306179e+03,  2.94050329e+03,  8.94190327e+04,
        1.96857878e+04,  1.89280717e+04,  1.71861688e+04,  5.79590349e+04,
        3.76726387e+04,  4.80043645e+04, -1.40803133e+02, -3.72845469e+02,
        1.22757562e+03, -1.16687641e+03, -3.85024463e+03, -7.19151209e+02,
       -1.16415390e+03, -3.34996242e+03, -5.04564577e+02, -2.96993869e+03,
       -2.25884738e+03, -7.31361301e+01,  3.36078201e+02,  2.48389869e+03,
       -1.01798112e+03, -1.35155201e+03,  2.67944088e+03,  9.26842211e+02,
       -1.03136540e+01,  1.00754003e+02, -1.02199651e+03,  8.37661528e+02,
       -5.61698413e+02, -1.30226861e+02, -5.47283852e+02,  2.72644254e+01,
```


```
       -4.42401895e+03, -4.04247156e+02,  1.08247083e+03,  6.68162042e+03,
       -1.25698441e+03, -6.60778862e+02, -1.29058234e+03,  8.00459080e+02,
        1.85542490e+03,  9.81681729e+02,  9.72213846e+02,  1.11889178e+03,
        6.39516405e+02, -3.54090993e+01,  7.56174084e+03,  7.39633478e+02,
        5.08832522e+02,  9.31205318e+02,  3.34532495e+00, -1.96101915e+02,
        2.01103910e+03, -1.19577978e+02, -7.43556606e+01, -2.93468895e+03,
        2.10593444e+02, -2.17049051e+03,  3.10674245e+02,  1.45688895e+03,
        1.85116481e+03,  2.44480514e+02, -1.75767523e+03, -3.40392265e+03,
       -5.77816011e+03,  2.45518287e+02,  2.41922361e+03, -1.37511478e+02,
       -3.75423749e+02,  2.76743445e+03,  1.81407958e+03,  3.53029973e+02,
        3.34392410e+03,  1.30306179e+03,  2.94050329e+03,  8.94190327e+04,
        1.96857878e+04,  1.89280717e+04,  1.71861688e+04,  5.79590349e+04,
        3.76726387e+04,  4.80043645e+04, -1.40803133e+02, -3.72845469e+02,
        1.22757562e+03, -1.16687641e+03, -3.85024463e+03, -7.19151209e+02,
       -1.16415390e+03, -3.34996242e+03, -5.04564577e+02, -2.96993869e+03,
       -2.25884738e+03, -7.31361301e+01,  3.36078201e+02,  2.48389869e+03,
       -1.01798112e+03, -1.35155201e+03,  2.67944088e+03,  9.26842211e+02,
       -1.03136540e+01,  1.00754003e+02, -1.02199651e+03,  8.37661528e+02,
       -5.61698413e+02, -1.30226861e+02, -5.47283852e+02,  2.72644254e+01,
```


```
       -5.28991030e+02, -2.19433696e+02,  8.71706822e+02,  1.50911114e+03,
        4.60582268e+03,  4.90910632e+03,  1.51514931e+03,  2.73940671e+03,
        3.40676289e+02,  4.99030882e+03, -2.92426208e+03, -4.62472539e+02])
```

In [52]:
```
lr.intercept_
```

Out[52]:
```
180906.6299657845
```

In [53]:
```
y_pred = lr.predict(data_test)
```

In [54]:
```
l1 = sol
l2 = list(y_pred)
df = pd.DataFrame(list(zip(l1,l2)), columns = ['id' , 'SalePrice'])
```

In [55]:
```
df.to_csv('FinalResult.csv' , index = False)
```

In [ ]:
```

```

the code for module 3 has been documented

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 Machine Learning

Extreme Gradient Boosting (XGBoost)

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

Test Plan, Test Case

Lab Session #11

Table of Contents

1. Executive Summary

Scope:

Machine learning is a subfield of Artificial Intelligence (AI) that works with algorithms and technologies to extract useful information from data. Machine learning methods are appropriate in big data since attempting to manually process vast volumes of data would be impossible without the support of machines. Machine learning in computer science attempts to solve problems algorithmically rather than purely mathematically. Therefore, it is based on creating algorithms that permit the machine to learn. However, there are two general groups in machine learning which are supervised and unsupervised. Supervised is where the program gets trained on pre-determined set to be able to predict when a new data is given. Unsupervised is where the program tries to find the relationship and the hidden pattern between the data.

Abstract:

Machine learning (ML) aims at developing self-learning algorithms using datasets, such that the machine can be enabled to project future activity based on the past data. It has exhibited impressive developments over the years with the rapid increase in storage capacity and processing power of computers, and achieved high significance with its ability to produce systems used regularly in industry, education, and elsewhere. Taking the case of one such application in real estate, in this paper, machine learning has been employed to help predict the prices of houses by learning from a sample dataset consisting of attributes which influence this cost. Firstly, the fundamental concepts of machine learning, along with its applications are explored. Next, in reference to house price prediction, feature assessment, learning techniques and the libraries used for implementation of the system using Python are discussed. Based on the obtained prediction results, performance of the ML approach is evaluated and conclusions are drawn.

Objective:

goal is to predict sale prices for homes in Ames, Iowa. You're given a training and testing data set in csv format as well as a data dictionary.In addition, the given datasets should be processed to enhance performance, which is accomplished by identifying the necessary features by applying one of the selection methods 2 to eliminate the unwanted variables since each house has its unique features that help to estimate its price. These features may or may not be shared with all houses, which means they do not have the same influence on the house pricing resulting in inaccurate output.

With 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, this competition challenges you to predict the final price of each home.

Practice Skills

- Creative feature engineering
- Advanced regression techniques like random forest and gradient boosting

approach:

- Acquire the data
- Explore the data
- Engineer and transform the features and the target variable
- Build a model
- Make and submit predictions

2. Test Plan

Scope of Testing

Summarizing the scope of testing :
considering the ames and lowa datasets we have performed advanced regression from machine learning and found out the or predicted the housse prices

The real estate market is a standout amongst the most focused regarding pricing and keeps fluctuating. It is one of the prime fields to apply the ideas of machine learning on how to enhance and foresee the costs with high accuracy. The objective of the paper is the prediction of the market value of a real estate property. This system helps find a starting price for a property based on the geographical variables. By breaking down past market patterns and value ranges, and coming advancements future costs will be anticipated. This examination means to predict house prices in Mumbai city with Decision tree regressor. It will help clients to put resources into a bequest without moving towards a broker. The result of this research proved that the Decision tree regressor gives an accuracy of 89%.

Functional: Are all modules covered? Any exception for any modules ? Does automation cover all functional test cases or Regression – Critical Path Test Cases ?

Non-Functional: Are all NFR (Non-Functional Requirements) covered?

Types of Testing , Methodology , Tools

Category	Methodology	Tools Required
Functional Requirements	Manual	Excel Template

Test Deliverables

documentation od test case and all are below:

What is a Test case?
A test case has components that describe input, action, and an expected response, in order to determine if a feature of an application works correctly.
A test case is a set of instructions on "HOW" to validate a particular test objective/target, which, when followed will tell us if the expected behavior of the system is satisfied or not.
Most of them prefer excel spreadsheets because they can easily group test cases by test types and most importantly they can easily get test metrics with Excel formulas. But I'm sure that as the volume of your tests goes on increasing, you will find it extremely difficult to manage.
If you are not using any Test case management tool, then I would strongly recommend you to use an open-source tool to manage and execute your test cases.
Features:

- TestRail makes tracking test results easier.
- It seamlessly gets integrated with bug trackers, automated tests, etc.
- Personalized to-do lists, filters, and email notifications will help with boosting productivity.
- Dashboards and activity reports are for easy tracking and following the status of individual tests, milestones, and projects.

Pre-conditions: Any prerequisite that must be fulfilled before the execution of this test case. List all the pre-conditions in order to execute this test case successfully.

Dependencies: Mention any dependencies on other test cases or test requirements.

Test Steps: List all the test execution steps in detail. Write test steps in the order in which they should be executed. Make sure to provide as many details as you can.

Test Data: Use of test data as an input for this test case. You can provide different data sets with exact values to be used as an input.

Expected Result: What should be the system output after test execution? Describe the expected result in detail including the message/error that should be displayed on the screen.

Post-condition: What should be the state of the system after executing this test case?

Actual result: The actual test result should be filled after test execution. Describe the system behavior after test execution.

Status (Pass/Fail): If the actual result is not as per the expected result, then mark this test as failed. Otherwise, update it as passed.

Defect ID/Link: If the test status fails, then include the link to the defect log or mention the defect number.

Test Type/Keywords: This field can be used to classify tests based on test types. For Example, functional, usability, business rules, etc.

Requirements: Requirements for which this test case is being written for. Preferably the exact section number in the requirement doc.
Attachments/References: This field is useful for complex test scenarios in order to explain the test steps or expected results using a Visio diagram as a reference. Provide a link or location to the actual path of the diagram or document.

***Automation? (Yes/No)*:** Whether this test case is automated or not. It is useful to track automation status when test cases are automated

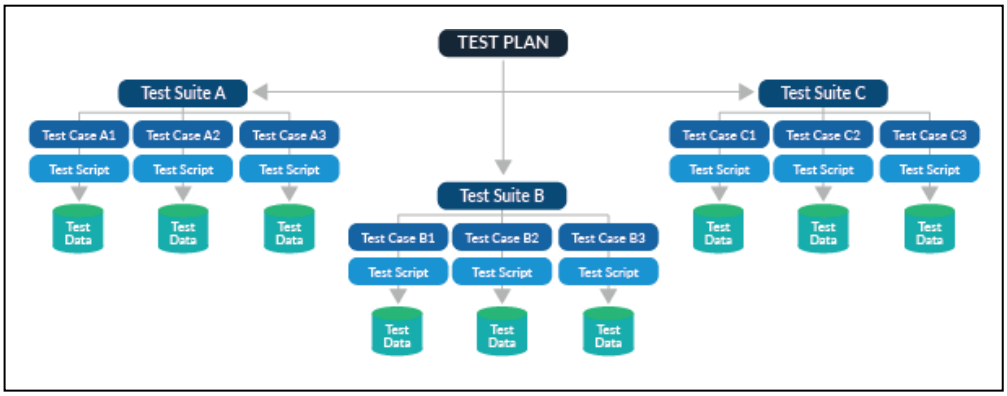
Levels of Test writing process:

- Level 1: In this level, you will write the basic cases from the available specification and user documentation.
- Level 2: This is the practical stage in which writing cases depend on the actual functional and system flow of the application.
- Level 3: This is the stage in which you will group some cases and write a test procedure. The test procedure is nothing but a group of small cases, maybe a maximum of 10.
- Level 4: Automation of the project. This will minimize human interaction with the system and thus the QA can focus on the currently updated functionalities to test rather than remaining busy with Regression testing.

Why do we Write Tests?

The basic objective of writing cases is to validate the test coverage of an application.

If you are working in any CMMi organization, then the test standards are followed more closely. Writing cases brings some sort of standardization and minimizes the ad hoc approach in testing.



here are the test cases are saleprice:

Id	prediction	SalePrice
1	208352.625	208500
2	173541.546875	181500
3	214273.046875	223500
4	161319.09375	140000
5	302222.90625	250000
6	150772.65625	143000
7	293506.1875	307000
8	227421.703125	200000
9	143611.328125	129900
10	120450.984375	118000
11	129962.398438	129500
12	367707.1875	345000
13	131885.984375	144000
14	218455.046875	279500
15	148784.421875	157000
16	134644.671875	132000
17	148606.125	149000
18	109368.335938	90000
19	150704.96875	159000
20	131534.65625	139000
21	331124.90625	325300
22	131490.796875	139400
23	227205.71875	230000
24	139552.765625	129900
25	144103.296875	154000
26	252671.921875	256300
27	131141.1875	134800
28	304084.90625	306000
29	182213.65625	207500
30	73132.65625	68500
31	95767.7734375	40000
32	137177	149350
33	202056.984375	179000

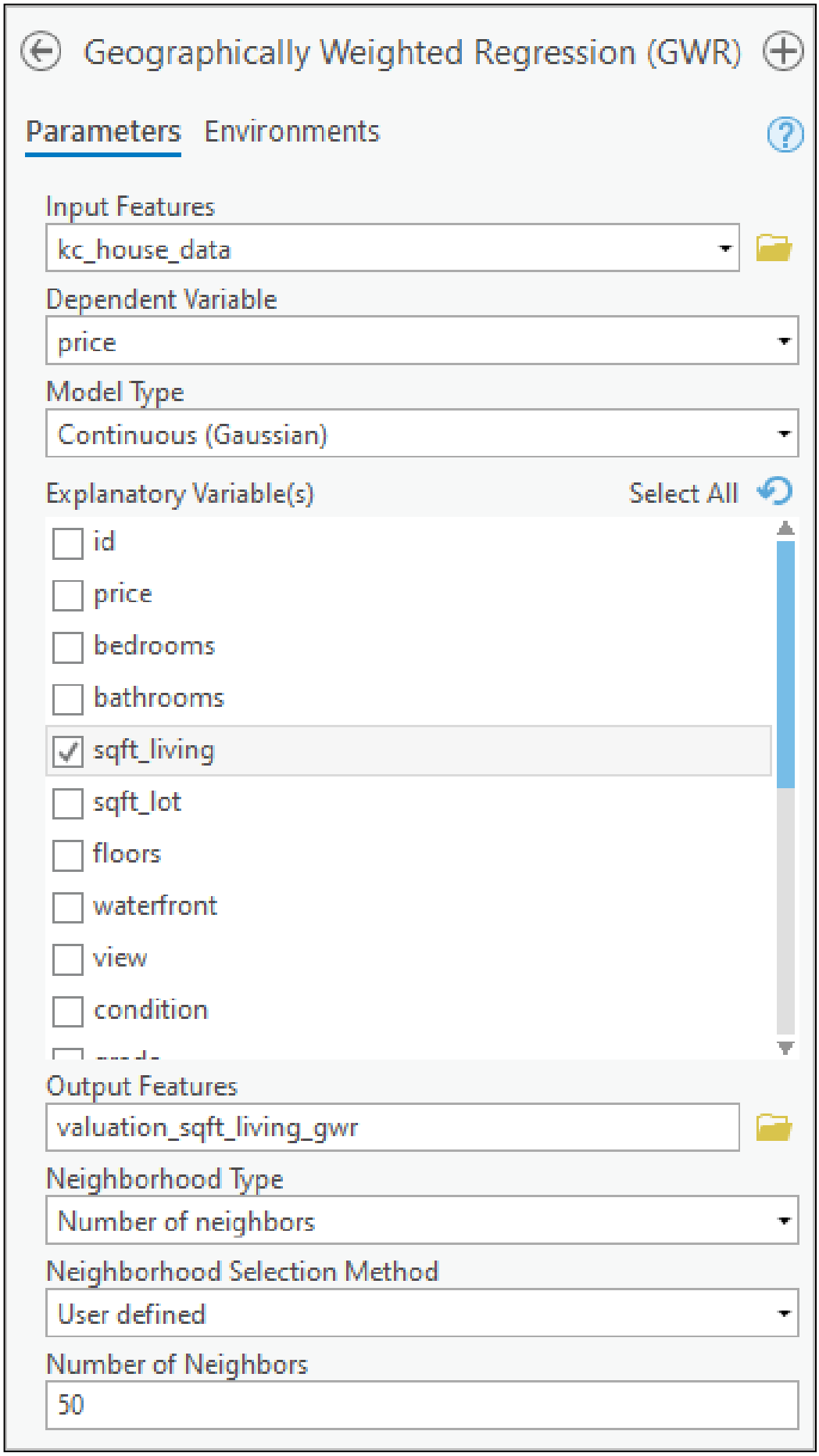
longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
-122.23	37.88	41	880	129	322	126	8.3252	452600	NEAR BAY
-122.22	37.86	21	7099	1106	2401	1138	8.3014	358500	NEAR BAY
-122.24	37.85	52	1467	190	496	177	7.2574	352100	NEAR BAY
-122.25	37.85	52	1274	235	558	219	5.6431	341300	NEAR BAY
-122.25	37.85	52	1627	280	565	259	3.8462	342200	NEAR BAY
-122.25	37.85	52	919	213	413	193	4.0368	269700	NEAR BAY
-122.25	37.84	52	2535	489	1094	514	3.6591	299200	NEAR BAY
-122.25	37.84	52	3104	687	1157	647	3.12	241400	NEAR BAY

```
median_house_value          1.000000
median_income               0.687160
rooms_per_household         0.146285
total_rooms                 0.135097
housing_median_age          0.114110
households                  0.064506
total_bedrooms              0.047689
population_per_household   -0.021985
population                 -0.026920
longitude                  -0.047432
latitude                   -0.142724
bedrooms_per_room          -0.259984
Name: median_house_value, dtype: float64
```

Data preprocessing:

Age and floor parameters were handled for their missing values. the target attribute is also dropped off from the training dataset. Pandas library is used for this purpose. For statistical visualization of the dataset, the min, max, standard deviation, mean of the target attribute were found out. We split the dataset into a training set (80%) and a test set (20%).

The value of a particular property depends on the infrastructure amenities surrounding the property. Recently, a few writers' scopes for finding the best properties for the customers came along with various technologies. mentioned the basic data mining concepts of how it works and supporting algorithms for the purpose of prediction. The most important part is which machine learning algorithm is best suited for predicting the house price. Often the location's environmental conditions decide what kind of price we can expect for different types of houses, presents various important features to use when forecasting property prices with good precision using a regression model. A. designed a system that used real-time neighborhood data to get precise real-world valuations using Google maps.


Geographically Weighted Regression (GWR)
Parameters
Environments
Input Features
kc_house_data
Dependent Variable
price
Model Type
Continuous (Gaussian)
Explanatory Variable(s)
Select All
id
price
bedrooms
bathrooms
sqft_living
sqft_lot
floors
waterfront
view
condition
Output Features
valuation_sqft_living_gwr
Neighborhood Type
Number of neighbors
Neighborhood Selection Method
User defined
Number of Neighbors
50

o

3. Test Case

3.1. Functional Test Cases

Test ID (#)	Test Scenario	Test Case	Execution Steps	Expected Outcome	Actual Outcome	Status	Remarks
1.	Verify User Registration from India	Accept Valid India Mobile Number on the Page#1	1. User clicks on User Registration link 2. Enter the mobile Number on the text box 3. Click Register button	User should be taken to the next page for entering more user details	user was taken to next page and aksed to enter more details	Pass / Failure pass	success
2.	Verify User Registration from India	Don't Accept Non Indian Mobile Number on the Page#1	if the user is not from india then pop up the invalid message on to the screen	ask to verify or the number or try with indian number	as its not india number was asked to verify number	pass/ failure pass	success
3.	prediction of house price	loading the datasets i.e train set and test case	the compiler get the datasets from the predictor and load those for prediction purpose	datasets analysis for prediction are required	datasets loaded and set to train and testing purpose	pass/ failure pass	success
4.	applying the regression methods from machine learning concepts	after training and testing sets got loaded	apply and try out the different regression methods	should get the prediction s	got the prediction s of houses situated in ames and lowa	pass/ failure pass	success

3.1. Non-Functional Test Cases

Test ID (#)	Test Scenario	Test Case	Execution Steps	Expected Outcome	Actual Outcome	Status	Remarks
1.	importing	tc1	should import	all libraries are imported	libraries are impotred	pass/ failure pass	success
2.	processing	tc2	data processing	processed	processed	pass/ failure pass	sucess
3.	evaludation	tc3	checking for results	get prices	got the price	pass/ failure pass	sucess

4. Test ReportReference

1. https://www.pmi.org/

Manual Testing with report

Lab Session #12 ss

Table of Contents

1. Executive Summary

Scope:

Machine learning is a subfield of Artificial Intelligence (AI) that works with algorithms and technologies to extract useful information from data. Machine learning methods are appropriate in big data since attempting to manually process vast volumes of data would be impossible without the support of machines. Machine learning in computer science attempts to solve problems algorithmically rather than purely mathematically. Therefore, it is based on creating algorithms that permit the machine to learn. However, there are two general groups in machine learning which are supervised and unsupervised. Supervised is where the program gets trained on pre-determined set to be able to predict when a new data is given. Unsupervised is where the program tries to find the relationship and the hidden pattern between the data.

Abstract:

Machine learning (ML) aims at developing self-learning algorithms using datasets, such that the machine can be enabled to project future activity based on the past data. It has exhibited impressive developments over the years with the rapid increase in storage capacity and processing power of computers, and achieved high significance with its ability to produce systems used regularly in industry, education, and elsewhere. Taking the case of one such application in real estate, in this paper, machine learning has been employed to help predict the prices of houses by learning from a sample dataset consisting of attributes which influence this cost. Firstly, the fundamental concepts of machine learning, along with its applications are explored. Next, in reference to house price prediction, feature assessment, learning techniques and the libraries used for implementation of the system using Python are discussed. Based on the obtained prediction results, performance of the ML approach is evaluated and conclusions are drawn.

Objective:

goal is to predict sale prices for homes in Ames, Iowa. You're given a training and testing data set in csv format as well as a data dictionary.In addition, the given datasets should be processed to enhance performance, which is accomplished by identifying the necessary features by applying one of the selection methods 2 to eliminate the unwanted variables since each house has its unique features that help to estimate its price. These features may or may not be shared with all houses, which means they do not have the same influence on the house pricing resulting in inaccurate output.

With 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, this competition challenges you to predict the final price of each home.

Practice Skills

- Creative feature engineering
- Advanced regression techniques like random forest and gradient boosting

approach:

Acquire the data

Explore the data

Engineer and transform the features and the target variable

Build a model

Make and submit predictions

2. Test Plan

Scope of Testing

summarizes the scope of testing:

considering the ames and lowa datasets, we have performed advanced regression from machine learning and found out the or predicted the housse prices

The real estate market is a standout amongst the most focused regarding pricing and keeps fluctuating. It is one of the prime fields to apply the ideas of machine learning on how to enhance and foresee the costs with high accuracy. The objective of the paper is the prediction of the market value of a real estate property. This system helps find a starting price for a property based on geographical variables. By breaking down past market patterns and value ranges, and coming advancements future costs will be anticipated. This examination means to predict house prices in Mumbai city with a Decision tree regressor. It will help clients to put resources into a bequest without moving towards a broker. The result of this research proved that the Decision tree regressor gives an accuracy of 89%.

Functional: Are all modules covered? Any exception for any modules ? Does automation cover all functional test cases or Regression – Critical Path Test Cases ?

Non-Functional: Are all NFR (Non-Functional Requirements) covered?
We'll use the Random Forest regression algorithm to predict the price of the houses. In this article, we'll consider machine learning algorithms as a black box that fits the data.
This article focuses more on the machine learning pipeline. We'll begin by loading the data. Since we're using an inbuilt dataset, we'll be calling the load_boston function from the sklearn.datasets module. We load the data into the data variable.
Once the data is loaded, we separate the data and target attributes of the data variable. We store them in variables data and target, respectively.
Once we have the data and target values in 2 different variables, we can divide the data into two parts: the testing data and training data.

The theory behind dividing the dataset into two parts is to ensure the model doesn't overfit the training data. Otherwise, the model will perform well on the training data and perform poorly on the test data.
This means that the model has learned the training data so well that it cannot generalize new data points. This should be avoided.
Once we have the dataset split into training and testing sets, we can pre-process the data. Pre-processing involves scaling the values and converting the categorical values into numerical values.
For example, there is a variable in the given dataset that indicates whether the Charles river is close to the house or not. This variable takes the values Near and Far.
We need to convert this into a numerical value. To do this, we can use the LabelEncoder function available in the pre-processing module of sklearn. This will replace the column with numerical values of 0 and 1, respectively. 0 indicates Near, and 1 indicates Far.
Once we perform the pre-processing of the dataset, we can fit the data to the model. We begin with instantiating an object of the RandomForestRegressor class. This is available in the sklearn.ensemble module. We use the fit method to fit the data to the model.
Once the model is fit, we evaluate the model's performance on the test set we got earlier. We use the predict method present in the RandomForestRegressor class.
The predict method takes in the test input data and predicts an output. Using the predicted output and the actual output known from the dataset, we compute the test accuracy.
Another useful evaluation metric is the Mean Squared Error. The Means Squared Error (MSE) loss estimates how far the prediction is from the mean of the output. Computing the MSE gives us an idea about the performance of the algorithm.
Training: Our training data consists of 1,460 examples of houses with 79 features describing every aspect of the house. We are given sale prices (labels) for each house. The training data is what we will use to "teach" our models.
Testing: The test data set consists of 1,459 examples with the same number of features as the training data. Our test data set excludes the sale price because this is what we are trying to predict. Once our models have been built we will run the best one the test data

Types of Testing, Methodology, Tools

Category	Methodology	Tools Required
Functional Requirements	Manual	Word Template

Test Deliverables

documentation od test case and all are below:

What is a Test case?
A test case has components that describe input, action, and an expected response, in order to determine if a feature of an application works correctly.
A test case is a set of instructions on "HOW" to validate a particular test objective/target, which, when followed will tell us if the expected behavior of the system is satisfied or not.

Most of them prefer excel spreadsheets because they can easily group test cases by test types and most importantly they can easily get test metrics with Excel formulas. But I'm sure that as the volume of your tests goes on increasing, you will find it extremely difficult to manage. If you are not using any Test case management tool, then I would strongly recommend you to use an open-source tool to manage and execute your test cases.
Features:

- TestRail makes tracking test results easier.
- It seamlessly gets integrated with bug trackers, automated tests, etc.
- Personalized to-do lists, filters, and email notifications will help with boosting productivity.
- Dashboards and activity reports are for easy tracking and following the status of individual tests, milestones, and projects.

Pre-conditions: Any prerequisite that must be fulfilled before the execution of this test case. List all the pre-conditions in order to execute this test case successfully.

Dependencies: Mention any dependencies on other test cases or test requirements.

Test Steps: List all the test execution steps in detail. Write test steps in the order in which they should be executed. Make sure to provide as many details as you can.

Test Data: Use of test data as an input for this test case. You can provide different data sets with exact values to be used as an input.

Expected Result: What should be the system output after test execution? Describe the expected result in detail including the message/error that should be displayed on the screen.

Post-condition: What should be the state of the system after executing this test case?

Actual result: The actual test result should be filled after test execution. Describe the system behavior after test execution.

Status (Pass/Fail): If the actual result is not as per the expected result, then mark this test as failed. Otherwise, update it as passed.

Defect ID/Link: If the test status fails, then include the link to the defect log or mention the defect number.

Test Type/Keywords: This field can be used to classify tests based on test types. For Example, functional, usability, business rules, etc.

Requirements: Requirements for which this test case is being written for. Preferably the exact section number in the requirement doc.
Attachments/References: This field is useful for complex test scenarios in order to explain the test steps or expected results using a Visio diagram as a reference. Provide a link or location to the actual path of the diagram or document.

Automation? (Yes/No): Whether this test case is automated or not. It is useful to track automation status when test cases are automated

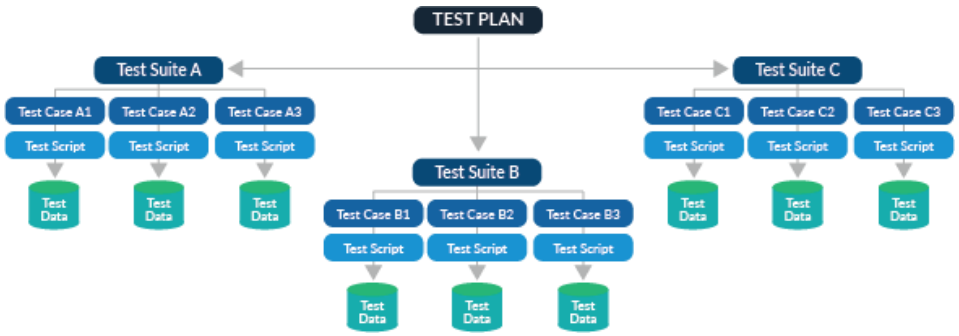
Levels of Test writing process:

- Level 1: In this level, you will write the basic cases from the available specification and user documentation.
- Level 2: This is the practical stage in which writing cases depend on the actual functional and system flow of the application.
- Level 3: This is the stage in which you will group some cases and write a test procedure. The test procedure is nothing but a group of small cases, maybe a maximum of 10.
- Level 4: Automation of the project. This will minimize human interaction with the system and thus the QA can focus on the currently updated functionalities to test rather than remaining busy with Regression testing.

Why do we Write Tests?

The basic objective of writing cases is to validate the test coverage of an application.

If you are working in any CMMi organization, then the test standards are followed more closely. Writing cases brings some sort of standardization and minimizes the ad hoc approach in testing.



here are the test cases are saleprice:

Id	prediction	SalePrice
1	208352.625	208500
2	173541.546875	181500
3	214273.046875	223500
4	161319.09375	140000
5	302222.90625	250000
6	150772.65625	143000
7	293506.1875	307000
8	227421.703125	200000
9	143611.328125	129900
10	120450.984375	118000
11	129962.398438	129500
12	367707.1875	345000
13	131885.984375	144000
14	218455.046875	279500
15	148784.421875	157000
16	134644.671875	132000
17	148606.125	149000
18	109368.335938	90000
19	150704.96875	159000
20	131534.65625	139000
21	331124.90625	325300
22	131490.796875	139400
23	227205.71875	230000
24	139552.765625	129900
25	144103.296875	154000
26	252671.921875	256300
27	131141.1875	134800
28	304084.90625	306000
29	182213.65625	207500
30	73132.65625	68500
31	95767.7734375	40000
32	137177	149350
33	202056.984375	179000

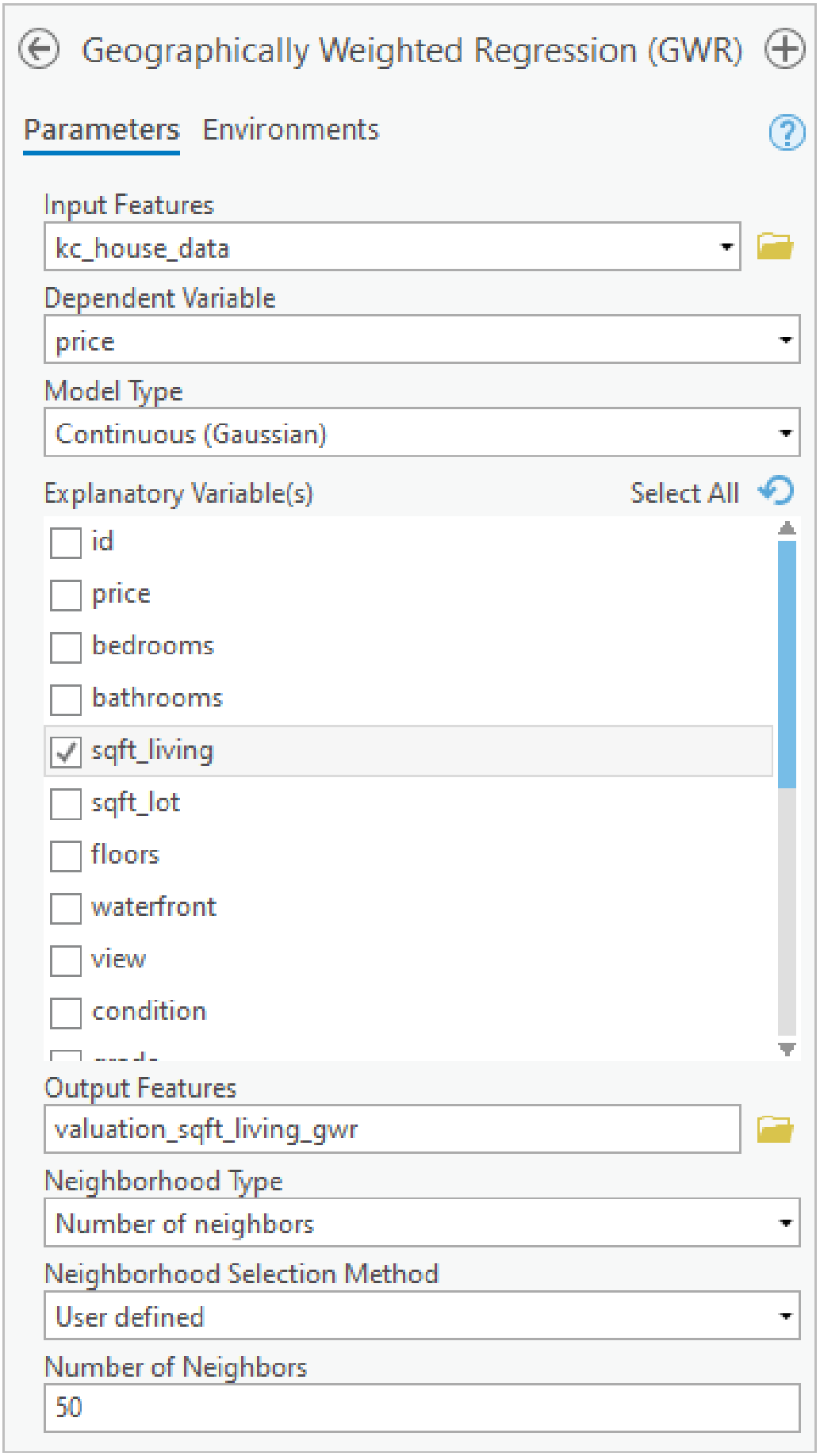
longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
-122.23	37.88	41	880	129	322	126	8.3252	452600	NEAR BAY
-122.22	37.86	21	7099	1106	2401	1138	8.3014	358500	NEAR BAY
-122.24	37.85	52	1467	190	496	177	7.2574	352100	NEAR BAY
-122.25	37.85	52	1274	235	558	219	5.6431	341300	NEAR BAY
-122.25	37.85	52	1627	280	565	259	3.8462	342200	NEAR BAY
-122.25	37.85	52	919	213	413	193	4.0368	269700	NEAR BAY
-122.25	37.84	52	2535	489	1094	514	3.6591	299200	NEAR BAY
-122.25	37.84	52	3104	687	1157	647	3.12	241400	NEAR BAY

```
median_house_value          1.000000
median_income               0.687160
rooms_per_household         0.146285
total_rooms                 0.135097
housing_median_age          0.114110
households                  0.064506
total_bedrooms              0.047689
population_per_household   -0.021985
population                 -0.026920
longitude                  -0.047432
latitude                   -0.142724
bedrooms_per_room          -0.259984
Name: median_house_value, dtype: float64
```

Data preprocessing:

Age and floor parameters were handled for their missing values. the target attribute is also dropped off from the training dataset. Pandas library is used for this purpose. For statistical visualization of the dataset, the min, max, standard deviation, mean of the target attribute were found out. We split the dataset into a training set (80%) and a test set (20%).

The value of a particular property depends on the infrastructure amenities surrounding the property. Recently, a few writers' scopes for finding the best properties for the customers came along with various technologies. mentioned the basic data mining concepts of how it works and supporting algorithms for the purpose of prediction. The most important part is which machine learning algorithm is best suited for predicting the house price. Often the location's environmental conditions decide what kind of price we can expect for different types of houses, presents various important features to use when forecasting property prices with good precision using a regression model. A. designed a system that used real-time neighborhood data to get precise real-world valuations using Google maps.


Geographically Weighted Regression (GWR)
Parameters
Environments
Input Features
kc_house_data
Dependent Variable
price
Model Type
Continuous (Gaussian)
Explanatory Variable(s)
Select All
id
price
bedrooms
bathrooms
sqft_living
sqft_lot
floors
waterfront
view
condition
Output Features
valuation_sqft_living_gwr
Neighborhood Type
Number of neighbors
Neighborhood Selection Method
User defined
Number of Neighbors
50

3. Test Case

Functional Test Cases

Test ID (#)	Test Scenario	Test Case	Execution Steps	Expected Outcome	Actual Outcome	Status	Remarks
	Verify User Registrationfrom India	Accept Valid India Mobile Numberon the Page#1	1. User clicks on User Registrationlink 2. Enter the mobile Number on the text box 3. Click Registerbutton	User should be takento the next page for enteringmore user details	user was taken to next page and aksed to enter more details	Pass / Failure Pass	success
2.	Verify User Registrationfrom India	Don't Accept Non Indian Mobile Numberon the Page#1	if the user is not from india then pop up theinvalid messageon to the screen	ask to verify orthe number or try with indian number	as its not india number was asked to verify number	pass/ failur epass	success
3.	prediction of house price	loading the datasets i.e train set and test case	the compiler get the datasets from the predictor and load those for prediction purpose	datasets analysis for prediction are required	datasets loaded and set to train and testing purpose	pass/ failure pass	success
4.	applying the regression methods from machine learning concepts	after training and testing sets got loaded	apply and try out the different regression methods	should get the predictions	got the predictions of houses situated in ames and lowa	pass/ failure pass	success

Non-Functional Test Cases

Test ID (#)	Test Scenario	Test Case	Execution Steps	Expected Outcome	Actual Outcome	Status	Remarks
1.	importing	tc1	should import	all libraries are imported	libraries are imported	pass/ failure pass	success
2.	processing	tc2	data processing	processed	processed	pass/ failure pass	sucess
3.	evaludation	tc3	checking for results	get prices	got the price	pass/ failure pass	sucess

The dataset is the prices and features of residential houses sold from 2006 to 2010 in Ames, Iowa, obtained from the Ames Assessor's Office. This dataset consists of 79 house features and 1460 houses with sold prices. Although the dataset is relatively small with only 1460 examples, it contains 79 features such as areas of the houses, types of the floors, and numbers of bathrooms. Such large amounts of features enable us to explore various techniques to predict the house prices. The dataset consists of features in various formats. It has numerical data such as prices and numbers of bathrooms/bedrooms/living rooms, as well as categorical features such as zone classifications for sale, which can be 'Agricultural', 'Residential High Density', 'Residential Low Density', 'Residential Low Density Park', etc. In order to make this data with different format usable for our algorithms, categorical data was converted into separated indicator data, which expands the number of features in this dataset. The final dataset has 288 features. We splitted our dataset into training and testing set with a roughly 70/30 split, with 1000 training examples and 460 testing examples. Besides, there were some features that had values of N/A; we replaced them with the mean of their columns so that they don't influence the distribution.

Models:

We would perform two types of supervised learning algorithms: classification and regression. While it seems more reasonable to perform regression since house prices are continuous, classifying house prices into individual ranges of prices would also provide helpful insight for the users; also, this helps us explore different techniques which might be regression- or classification-specific. Since there are 288 features in the dataset, regularization is needed to prevent overfit. In order to determine the regularization parameter, throughout the project in both classification and regression parts, we would first perform

K-fold cross validation with k = 5 on a wide range of selection of regularization parameters; this helped us to select the best regularization parameters in the training phase. In order to further improve our models, we also performed principal component analysis pipeline on all

models, and cross validated number of components to fit in each of the model to give the optimized results

4. Defect Log

Requirement #	Defect ID #	Defect Description	Assignee	Status
M1R1	1124	could not predictas it was out of box	Anuradha	to fail and thenrectified and got passed success
M1R2	1125	Could not get thegraph	Anuradha	fail and then rectified and got passed success

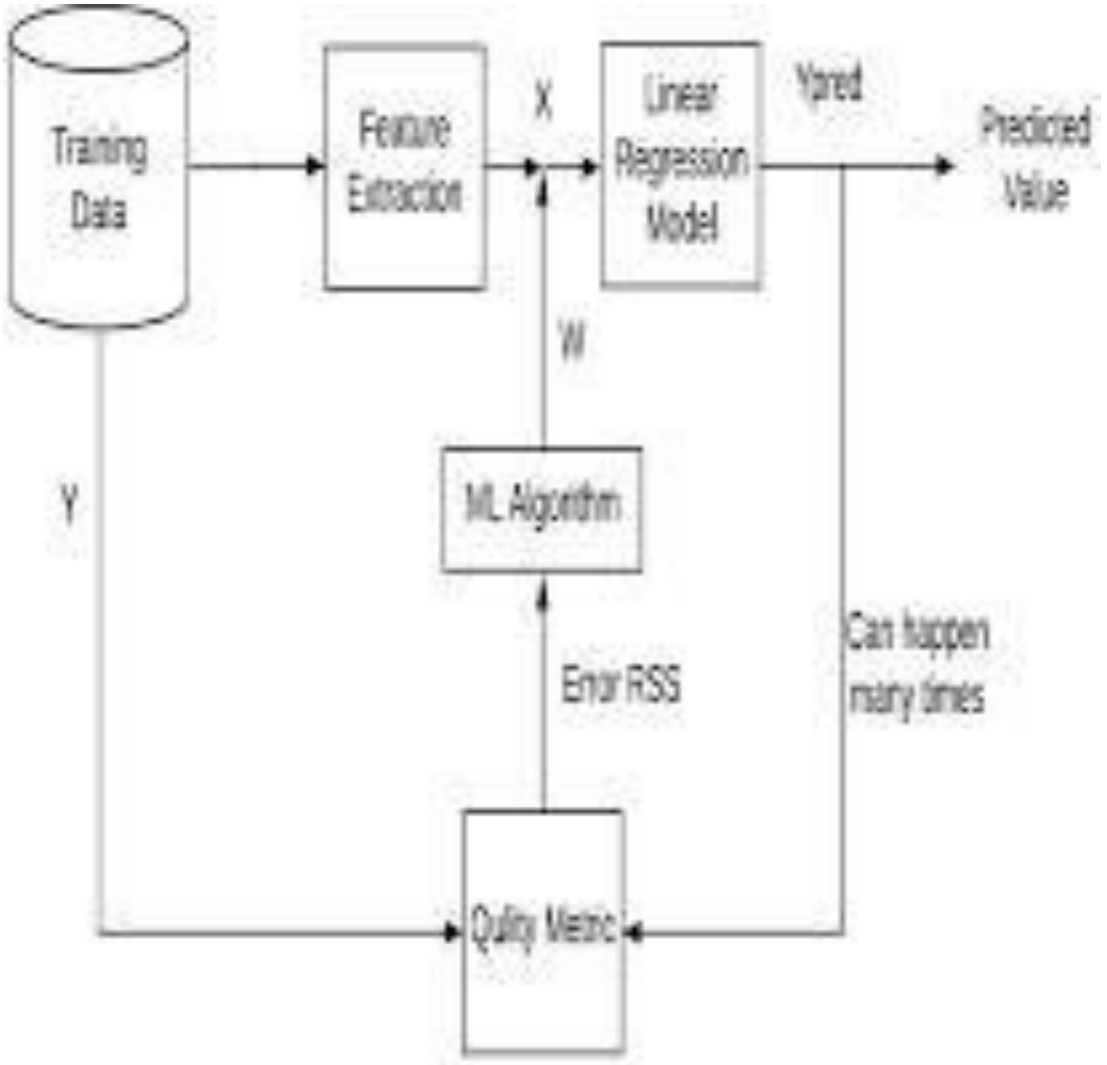
For regression models, we try to solve the following problem: given a processed list of features for a house, we would like to predict its potential sale price. Linear regression is a natural choice of baseline model for regression problems. So we first ran linear regression including all features, using our 288 features and 1000 training samples. The model is then used to predict sale prices of houses given features in our test data and is compared to the actual sale prices of houses given in test data set. The performance was measured by Root Mean Square Error (rmse) of the predicted results and the actual results. Our baseline model generated a rmse of 0.5501. Note that since the target variable SalePrice is log-transformed before model fitting, the resulting rmse is based on differences in the log-transformed sale prices, which accounts for the small values of rmse for regression models. After using linear regression model as the baseline model, we included the regularization parameters in linear regression models to reduce overfitting. Linear regression with Lasso after 5-fold cross validation generated a rmse of 0.5418, which is better than our baseline model. Also, linear regression with lasso automatically picked 110 variables and eliminated the other 178 variables to fit in the mode. The plot on selected features and their weights in lasso regularized model.

The pipeline's modules are the following :

● Cleaning, is the first module called to clean the item and verify that all the information in it correspond to the pattern used to extract it. For instance, suppose that we have several estates on a given web page. Each estate is presented using images and many basic information such as, the gross area, the saleable area, the price, etc. When the spider extract them, it is not unlikely that some noise (like extra characters) was present with the value, or just the value does not exist. The cleaning module removes the noise, and check that all the values are not empty, otherwise the item is dropped. This is done for simplicity, indeed, it could be better to try to inference them later. After the cleaning part done, the item is sent to the formatting module.

● Formatting. the second module is used to format the item's values as we want. A basic example could be for the price, initially got being string type, is converted as float. This is done for every numeric values. The item formatted is then sent to the last module called Integrating.

● Integrating, this module, the last one, is basically the one in charges of saving the items in the format that we want. It also checks that there is no redundancies between the tuples. In my case, I decided to save them in an excel sheet for each website.



5. Test Report

Summarize the current status of the Testing
present obstacles to proceed further
Seek help from stakeholders to remove obstacles/constraints

The experiment is done to pre-process the data and evaluate the prediction accuracy of the models. The experiment has multiple stages that are required to get the prediction results. These stages can be defined as: -

Pre-processing:
both datasets will be checked and pre-processed using the methods from section 4.2. These methods have various ways of handling data. Thus, the preprocessing is done on multiple iterations where each time the accuracy will be evaluated with the used combination

- Data splitting:
dividing the dataset into two parts is essential to train the model with one and use the other in the evaluation. The dataset will be split 75% for training and 25% for testing.

Evaluation:
the accuracy of both datasets will be evaluated by measuring the R2 and RMSE rate when training the model alongside an evaluation of the actual prices on the test dataset with the prices that are being predicted by the model. 10 - Performance: alongside the evaluation metrics, the required time to train the model will be measured to show the algorithm vary in terms of time.

Correlation:
correlation between the available features and house price will be evaluated using the Pearson Coefficient Correlation to identify whether the features have a negative, positive or zero correlation with the house price.

Evaluation Metrics
The prediction accuracy will be evaluated by measuring the R-Squared (R2), and Root Mean Square Error (RSME) of the model used in training. R2 will show if the model is overfitted, whereas RSME shows the error percentage between the actual and predicted data, which in this case, the house prices.

Experiment Results

Many machine learning algorithms are used to predict. However, previous researches have shown a comparison between them alongside Artificial neural network in different datasets. Therefore, using these algorithms is beneficial so that the result can be as near to the claimed results. However, the prediction accuracy of these algorithms depends heavily on the given data when training the model. If the data is in bad shape, the model will be overfitted and inefficient, which means that data pre-processing is an important part of this experiment and will affect the final results. Thus, multiple combinations of pre-processing methods need to be tested before getting the data ready to be used in training.

This study was conducted on two different datasets, public and local. The public data set has 80 features and 1460 rows, and the local dataset has 9136 rows and a total of 13 features and 19 after adding the features presented in this study. However, the final trained model gave promising results regards the house prices.

Conclusion:

The study shows a comparison between the regression algorithms and artificial neural network when predicting house prices in Ames, Iowa, United States and Malmö, Sweden. The results were promising for the public data due to it being rich with features and having strong correlation, whereas the local data gave a worse outcome when the same pre-processing strategy was implemented due to it being in a different shape compared with the public data in terms of the number of features and the correlation strength.

Question 1 – Which machine learning algorithm performs better and has the most accurate result in house price prediction? And why?

Lasso made the best performance overall when both R2 and RMSE scores are taking into consideration. It has achieved the best performance due to its L1 norm regularisation for assigning zero weights to the insignificant features.

Question 2 – What are the factors that have affected house prices in Malmö over the years?

The number of crimes, repo, lending, and deposit rates has a weak correlation with the house prices. Which means there are lower likelihood relationships between these factors and sale price. However, when these factors increase the house price decrease. Besides, inflation and year have changed the house prices positively, which means when these factors increase, the house price increase.

Category	**Progress Against Plan**	**Status**
Functional Testing	Green	Completed
Non-Functional Testing	Green	Completed
system testing	Green	Completed

Functional	**Test Case Coverage(%)**	**Status**
Training set	60%	Completed
Cross set	20%	Completed

Test set	20%	Completed

Reference

1. https://www.pmi.org/

LIST OF ABBREVIATIONS

API	APPLICATION PROGRAMMING INTERFACE
HTML	HYPER TEXT MARKUP LANGUAGE
CSS	CASCADING STYLES SHEET
JS	JAVASCRIPT
RAM	RANDOM ACCESS MEMORY
UI	USER INTERFACE
UX	USER EXPERIENCE
IDE	INTEGRATED DEVELOPMENT ENVIRONMENT
WBS	WORK BREAKDOWN STRUCTURE
SWOT	STRENGTH, WEAKNESS, OPPURTUNITIES AND THREATS
RMM	RISK MONITERING AND MANAGEMENT
UML	UNIFIED MODELING LANGUAGE
OOP	OBJECT ORIENTED PROGRAMMING
ER	ENTITY RELATION
DFD	DATA FLOW DIAGRAM
HTTP	HYPER TEXT TRANSFER PROTOCOL
SVM	SUPPORT VECTOR MACHINE
CRUD	CREATE, READ, UPDATE, DELETE
MLM	MULTI LEVEL MODEL
DB	DATABASE
CLI	COMMAND LINE INTERFACE
SQL	STRUCTURED QUERY LANGUAGE
ANN	ARTIFICAL NETWORK MODEL
HPM	HEDONIC PRICE MODEL
HPI	HOUSE PRICE INDEX
MLR	MULTIPLE LINEAR REGRESSION
OLS	ORDINARY LEAST SQUARES

CONCLUSION

Improvement in computing technology has made it possible to examine social information that cannot previously be captured, processed and analysed. New analytical techniques of machine learning can be used in property research. This study is an exploratory attempt to use machine learning algorithms in estimating housing prices, and then compare their results. To conclude, the application of machine learning in property research is still at an early stage. We hope this study has moved a small step ahead in providing some methodological and empirical contributions to property appraisal, and presenting an alternative approach to the valuation of housing prices. Future direction of research may consider incorporating additional property transaction data from a larger geographical location with more features, or analysing other property types beyond housing development. Buying your own house is what every human wish for. Using this proposed model, we want people to buy houses and real estate at their rightful prices and want to ensure that they don't get tricked by sketchy agents who just are after their money. Additionally, this model will also help Big companies by giving accurate predictions for them to set the pricing and save them from a lot of hassle and save a lot of precious time and money. Correct real estate prices are the essence of the market and we want to ensure that by using this model.

The system is apt enough in training itself and in predicting the prices from the raw data provided to it. After going through several research papers and numerous blogs and articles, a set of algorithms were selected which were suitable in applying on both the datasets of the model. After multiple testing and training sessions, it was determined that the XGBoost Algorithm showed the best result amongst the rest of the algorithms. The system was potent enough for Predicting the prices of different houses with various features and was able to handle large sums of data. The system is quite user-friendly and time-saving.

REFERENCES

Dataset link:

https://www.kaggle.com/c/house-prices-advanced-regression-techniques

House price prediction zip file:

6.house-prices-advanced-regression-techniques.zip

Train dataset link:

..\Desktop\test.csv

Test dataset link:

..\Desktop\test.csv

Sample submission:

..\Desktop\sample_submission.csv

Final Result link:

..\Desktop\FinalResult.csv

Machine learning with python :

https://towardsdatascience.com/predicting-house-prices-with-linear-regression-machine-learning-from-scratch-part-ii-47a0238aeac1

House price prediction with different datasets:

https://thecleverprogrammer.com/2020/12/29/house-price-prediction-with-python/

Machine learning algorithms:

https://www.diva-portal.org/smash/get/diva2:1456610/FULLTEXT01.pdf

APPENDIX(CODE):

GOOGLE COLLAB LINK:

https://colab.research.google.com/drive/1M4Rz03CadoBosvu4G9lg8z9qUjoZ4OU1#scrollTo=zSLF0E5KUtCJ

Code:

```
%matplotlib inline
import seaborn as sns
```

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

```
data_train = pd.read_csv("train.csv")
data_test = pd.read_csv("test.csv")
```

```
data_test.head(10)
```

```
print(data_train.shape)
print(data_test.shape)
```

```
data_test['SalePrice'] = 0
```

```
data = pd.concat([data_train , data_test] , axis = 0)
```

```
data.shape
```

```
data.columns
```

```
data.describe
```

```
data['SalePrice']
```

```
data.isnull().sum()
```

```
plt.figure(figsize = (10,6))
sns.heatmap(data.isnull() , yticklabels = False , cbar = False , cmap = "viridis")
```

```
data.info()
```

```
data['LotFrontage'] = data['LotFrontage'].fillna(data['LotFrontage'].mean())
data['BsmtFinSF1'] = data['BsmtFinSF1'].fillna(data['BsmtFinSF1'].mean())
data['BsmtFinSF2'] = data['BsmtFinSF2'].fillna(data['BsmtFinSF2'].mean())
data['BsmtUnfSF'] = data['BsmtUnfSF'].fillna(data['BsmtUnfSF'].mean())
data['TotalBsmtSF'] = data['TotalBsmtSF'].fillna(data['TotalBsmtSF'].mean())
data['GarageCars'] = data['GarageCars'].fillna(data['GarageCars'].mean())
data['GarageArea'] = data['GarageArea'].fillna(data['GarageArea'].mean())
```

```
data.drop(['Alley' , 'PoolQC' , 'Fence' , 'MiscFeature'], axis = 1 , inplace = True)
data.drop(['GarageYrBlt'], axis = 1 , inplace = True)
```

```
data['MSZoning'] = data['MSZoning'].fillna(data['MSZoning'].mode()[0])
data['MasVnrType'] = data['MasVnrType'].fillna(data['MasVnrType'].mode()[0])
data['BsmtQual'] = data['BsmtQual'].fillna(data['BsmtQual'].mode()[0])
data['BsmtCond'] = data['BsmtCond'].fillna(data['BsmtCond'].mode()[0])
data['BsmtExposure'] = data['BsmtExposure'].fillna(data['BsmtExposure'].mode()[0])
data['BsmtFinType1'] = data['BsmtFinType1'].fillna(data['BsmtFinType1'].mode()[0])
data['BsmtFinType2'] = data['BsmtFinType2'].fillna(data['BsmtFinType2'].mode()[0])
data['FireplaceQu'] = data['FireplaceQu'].fillna(data['FireplaceQu'].mode()[0])
data['GarageType'] = data['GarageType'].fillna(data['GarageType'].mode()[0])
data['GarageFinish'] = data['GarageFinish'].fillna(data['GarageFinish'].mode()[0])
data['GarageQual'] = data['GarageQual'].fillna(data['GarageQual'].mode()[0])
data['GarageCond'] = data['GarageCond'].fillna(data['GarageCond'].mode()[0])
data['SaleType'] = data['SaleType'].fillna(data['SaleType'].mode()[0])
data['Utilities'] = data['Utilities'].fillna(data['Utilities'].mode()[0])
data['Exterior1st'] = data['Exterior1st'].fillna(data['Exterior1st'].mode()[0])
data['Exterior2nd'] = data['Exterior2nd'].fillna(data['Exterior2nd'].mode()[0])
data['Electrical'] = data['Electrical'].fillna(data['Electrical'].mode()[0])
data['Functional'] = data['Functional'].fillna(data['Functional'].mode()[0])
data['MasVnrArea'] = data['MasVnrArea'].fillna(data['MasVnrArea'].mode()[0])
data['BsmtFullBath'] = data['BsmtFullBath'].fillna(data['BsmtFullBath'].mode()[0])
data['BsmtHalfBath'] = data['BsmtHalfBath'].fillna(data['BsmtHalfBath'].mode()[0])
data['KitchenQual'] = data['KitchenQual'].fillna(data['KitchenQual'].mode()[0])
```

```
sol = data_test['Id']
data_test.shape
```

```
data.info()
data.shape
```

```
data.isnull().sum()
```

```
sns.heatmap(data.isnull() , yticklabels = False , cmap = 'viridis')
```

```
data.corr()
```

```
columns=['MSZoning','Street','LotShape','LandContour','Utilities','LotConfig','LandSlope','Neighborhood',
       'Condition2','BldgType','Condition1','HouseStyle','SaleType','SaleCondition','ExterCond',
'ExterQual','Foundation','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2',
'RoofStyle','RoofMatl','Exterior1st','Exterior2nd','MasVnrType','Heating','HeatingQC','CentralAir',
'Electrical','KitchenQual','Functional','FireplaceQu','GarageType','GarageFinish','GarageQual','GarageCond','PavedDrive']
```

```
len(columns)
```

```
def category(multcolumns):
    data_final=data
    i=0
    for field in multcolumns:

        df1=pd.get_dummies(data[field],drop_first=True)

        data.drop([field],axis=1,inplace=True)
        if i==0:
            data_final=df1.copy()
        else:

            data_final=pd.concat([data_final,df1],axis=1)
        i=i+1
```

```
    data_final=pd.concat([data,data_final],axis=1)

    return data_final
```

```
data = category(columns)
data.shape
```

```
data = data.loc[:,~data.columns.duplicated()]
```

```
data.shape
```

```
data.head()
```

```
data_train = data.iloc[:1460,:]
data_test = data.iloc[1460:,:]
```

```
data_test.drop(['SalePrice'] , axis = 1 , inplace = True)
```

```
y_train = data_train['SalePrice']
x_train = data_train.drop(['SalePrice'] , axis = 1)
```

```
x_train.shape
```

```
from sklearn.preprocessing import StandardScaler
scalar = StandardScaler()
x_train = scalar.fit_transform(x_train)
data_test = scalar.transform(data_test)
```

```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
```

```
y_tr = lr.predict(x_train)
lr.coef_
```

```
lr.intercept_
```

```
y_pred = lr.predict(data_test)
```

```
l1 = sol
```

```
l2 = list(y_pred)
df = pd.DataFrame(list(zip(l1,l2)), columns = ['id' , 'SalePrice'])
```

```
df.to_csv('FinalResult.csv' , index = False)
```

add a new cell to get effective and efficient output.

The FinalResult file will be uploaded in the google colab file or drive you can download that FianlResult from uploaded files section

datasets and final results csv files link:

train dataset:
..\Desktop\train.csv

Test dataset:
..\Desktop\test.csv

Final Result csv file:
..\Desktop\FinalResult.csv