

REAL ESTATE VALUATION USING MACHINE LEARNING

Dissertation submitted in partial fulfilment of requirements for the award of the degree of

Master of Computer Applications (M.C.A)

Submitted By

SURI SAI PRAVEEN

(Regd. No: 21131F0065)

Under the esteemed guidance of

Mr. B. BALAKRISHNA

Assistant Professor

Department of Computer Applications



**COLLEGE OF ENGINEERING
(AUTONOMOUS)**

Department of Computer Applications

GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING

(AUTONOMOUS)

(Affiliated to JNTU-K Kakinada)

VISAKHAPATNAM-530048

2021-2023

REAL ESTATE VALUATION USING MACHINE LEARNING

Dissertation submitted in partial fulfilment of requirements for the award of the degree of

Master of Computer Applications (M.C.A)

Submitted By

SURI SAI PRAVEEN

(Regd. No: 21131F0065)

Under the esteemed guidance of

Mr. B. BALAKRISHNA

Assistant Professor

Department of Computer Applications



**COLLEGE OF ENGINEERING
(AUTONOMOUS)**

Department of Computer Applications

GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING

(AUTONOMOUS)

(Affiliated to JNTU-K Kakinada)

VISAKHAPATNAM-530048

2021-2023

CERTIFICATE



COLLEGE OF ENGINEERING
(AUTONOMOUS)

GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING (AUTONOMOUS)

This is to certify that, the dissertation titled “**REAL ESTATE VALUATION USING MACHINE LEARNING**” is submitted by **Mr. SURI SAI PRAVEEN** with Regd. No **21131F0065** in partial fulfillment of the requirement for award of the Degree of **M.C.A** in **GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING (AUTONOMOUS)** affiliated to Jawaharlal Nehru Technological University, Kakinada is a bonafide record of project carried out by him under my guidance and supervision.

The contents of the project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Internal Guide

Mr. B. Balakrishna

Assistant Professor

Department of Computer Applications.

Head of the Department

Dr. Y. Anuradha, M.Tech, Ph.D

Associate Professor & Head

Department of Computer Applications.

External Examiner

CERTIFICATE OF PLAGIARISM CHECK



This is to certify the Master of Computer Applications (MCA) dissertation submitted by SURI SAI PRAVEEN (21131F0065) of Department of Computer Applications under the supervision of Mr. B.Balakrishna, Assistant professor of Department of Computer Applications, has undergone plagiarism check and found to have similarity index less than 40%. The details of the plagiarism check are as under:

File Name : **REAL_ESTATE_VALUATION_USING
_MACHINE_LEARNING.PDF(1.89M)**

Dissertation Title : **REAL_ESTATE_VALUATION_USING
_MACHINE_LEARNING**

Date and Time of Submission : **27-Jul-2023 09:20AM (UTC+0530)**

Submission ID : **2137399426**

Similarity Index : **23%**

Dean- Academic Programs (PG)

DECLARATION

I hereby declare the dissertation titled “**REAL ESTATE VALUATION USING MACHINE LEARNING**” is submitted to the Department of Computer Applications, **Gayatri Vidya Parishad College of Engineering (Autonomous)** affiliated to Jawaharlal Nehru Technological University, Kakinada in partial fulfillment of the requirements for the award of the degree of **Master of Computer Applications (MCA)**. This work is done by me and authentic to the best of my knowledge under the direction and valuable guidance of **Mr. B. Balakrishna**, Assistant Professor in the department of Computer Applications.

SURI SAI PRAVEEN

(Regd. No.21131F0065)

ACKNOWLEDGEMENT

I take the opportunity to thank everyone who has contributed to making the project possible. I am thankful to Gayatri Vidya Parishad College of Engineering (Autonomous) for giving opportunity me to work on a project as part of the curriculum.

I offer my profuse thanks to **Dr. A. B. Koteswara Rao**, the principal of **Gayatri VidyaParishad College of Engineering (Autonomous)** for providing the best faculty and labfacilities throughout the completion of Master of Computer Applications course.

I offer my profuse thanks to Prof. **Dr. K. Narasimha Rao**. Professor& Dean, Academics (PG) for his valuable suggestions and constant motivation that greatlyhelped me to complete project successfully.

I offer my profuse thanks to **Dr. Y. Anuradha** Associate Professor, Head of the Department of Computer Applications for his valuable suggestions and motivation thatgreatly helped me to complete the project successfully.

My sincere thanks to **Mr. B. Balakrishna**, Assistant Professor, Department of Computer Applications, my project guide, for his constant support, encouragement, and guidance. I am very much grateful for his valuable suggestions.

SURI SAI PRAVEEN

(Regd. No.21131F0065)

ABSTRACT

Buying a dream house is a common aspiration for people in these days, but accurately predicting its price has become more challenging, especially for those without expertise in this field. This difficulty often hinders informed decision-making when it comes to purchasing property. However, there is a proposed system that aims to simplify house price prediction by considering crucial factors. The system utilizes multiple regression techniques and bases on their performance through weighted averaging, leading to highly accurate predictions. This approach minimizes error and enhances accuracy.

INDEX

	Page No.
Certificate	i
Certificate of Plagiarism Check	ii
Declaration	iii
Acknowledgement	iv
Abstract	v
Contents	vii
List of Figures	ix
List of Tables	x
List of Acronyms	xi

CONTENTS

	Page No.
CHAPTER 1: INTRODUCTION	
1.1 Introduction	1-4
CHAPTER 2: LITERATURE SURVEY	
2.1 Literature Survey	5-8
CHAPTER 3: SYSTEM ANALYSIS	
3.1 Existing Method	10
3.2 Proposed Method	10
CHAPTER 4: SOFTWARE REQUIREMENTS AND SPECIFICATIONS	
4.1 Functional Requirements	12
4.2 Non-Functional Requirements	12
4.3 System Requirements	13
4.3.1 Hardware Requirements	13
4.3.2 Software Requirements	13
CHAPTER 5: SYSTEM DESIGN	
5.1 Architecture Diagram	15
5.2 UML Diagrams	16-19
5.2.1 Use Case Diagram	17
5.2.2 Sequence Diagram	18
5.2.3 Activity Diagram	19
CHAPTER 6: IMPLEMENTAION (CODE)	
6.1 Implementation	21
6.1.1 Software Environment	21
6.2 Libraries Used	22

6.3 Linear Regression Algorithm.....	23
6.4 Decision Tree Algorithm.....	24
6.5 K Nearest Neighbors (KNN) Algorithm... ..	25
6.6 Source Code	26-38

CHAPTER 7: RESULTS / OUTPUT SCREENS

7.1 Output Screens	39-51
--------------------------	-------

CHAPTER 8: SYSTEM TESTING (including test cases)

8.1 System Testing	53
8.2 Unit Testing.....	53
8.3 Integration Testing	53
8.4 Black-box Testing	53
8.5 Test cases.....	54

CHAPTER 9: CONCLUSION AND FUTURE SCOPE

9.1 Conclusion.....	56
---------------------	----

REFERENCES.....	57
------------------------	-----------

List of Figures

FIGURE NO	DESCRIPTION	PAGE NO.
5.1	Architecture Diagram	16
5.2.1	Use Case Diagram	18
5.2.2	Sequence Diagram	19
5.2.3	Activity Diagram	20
7.1.1	Home Page	40
7.1.2	Main Page	41
7.1.3	Upload Dataset Page	42
7.1.4	View Dataset Page	43
7.1.5	Split Page	44
7.1.6	Model Selection Page	45
7.1.7	Linear Regression Accuracy Page	46
7.1.8	Decision Tree Accuracy Page	47
7.1.9	KNN Accuracy Page	48
7.2.0	Input Values Page	49
7.2.1	Result Page	50
7.2.2	Accuracy Graph Page	51

List of Tables

TABLE NO.	TITLE	PAGE NO.
8.5	Test Cases	54

List of Symbols / Acronyms / Abbreviations

Symbol/ Acronym/ Abbreviation	DESCRIPTION
LR	LINEAR REGRESSION
KNN	K NEAREST NEIGHBOUR
GUI	GRAPHICAL USER INTERFACE
UML	UNIFIED MODELING LANGUAGE

CHAPTER 1
INTRODUCTION

INTRODUCTION

House marketing is a complex and dynamic system influenced by various factors such as floors, size, amenities, and market trends. Predicting house prices accurately is essential for both buyers and sellers to make informed decisions. Regression is a widely used statistical technique that assumes a linear relationship between the independent variables (such as square footage, number of bedrooms, etc.) and the dependent variable (house price). By analyzing historical housing data and applying some regressions, we can create a model that can estimate the price of a house based on its characteristics.

Regression models, such as Decision tree, play a crucial role in the house marketing domain by providing insights into the factors that drive house prices. These models enable us to understand the impact of variables like floors, size, amenities, and market trends on house prices and make predictions based on this understanding.

The real estate market plays a pivotal role in the economy, and understanding the factors influencing property values is of paramount importance for buyers, sellers, and investors alike. Through this system, we embark on a data-driven journey, harnessing the power of machine learning to make informed and reliable predictions about property prices.

Our comprehensive approach involves collecting historical real estate data, preprocessing it to ensure its quality, and selecting the most influential features for our models. We then train the linear regression, decision tree regression, and k-nearest neighbors regression algorithms, meticulously evaluating their performance to identify the optimal predictor.

Among the three models, the decision tree regression algorithm stands out as a potent candidate due to its ability to handle complex relationships between various features and house prices. By leveraging the strengths of this algorithm, we aspire to deliver an accurate and interpretable valuation model that can guide buyers and sellers towards well-informed decisions.

This system significance lies in its potential to empower individuals and real estate professionals alike, enabling them to navigate the ever-changing property market with confidence. Through the deployment of our predictive model, users can obtain estimated house prices based on input features, thus facilitating sound financial choices.

The depths of real estate valuation, we anticipate unearthing valuable insights and making a positive impact in the domain of property transactions. Let us embark on this exciting journey of prediction, discovery, and innovation in the realm of real estate valuation using machine learning algorithms.

Real estate valuation is an essential aspect of the property market, as it directly influences buying, selling, and investment decisions. For sellers, accurate property valuation ensures that they price their homes competitively, attracting potential buyers while maximizing their returns. On the other hand, buyers rely on valuation to gauge the fair market value of a property, helping them make informed offers and avoid overpaying.

Additionally, real estate investors depend on precise valuation models to identify lucrative investment opportunities and optimize their portfolio strategies. In this context, our system focusses on developing a reliable and accurate valuation model and holds immense value for all stakeholders involved in real estate transactions.

Real estate valuation poses unique challenges due to the multifaceted nature of property markets. Housing prices are influenced by a myriad of factors, including floors, property size, condition, amenities, and local market conditions.

Moreover, non-linear relationships between these features can complicate the valuation process, necessitating sophisticated algorithms capable of capturing complex interactions. By exploring multiple regression techniques, we aim to address these challenges and construct a comprehensive model that embraces the intricacies of real estate valuation.

Machine learning techniques offer a promising avenue to tackle the complexities of real estate valuation. Unlike traditional approaches, which may rely on simplistic assumptions, machine learning algorithms can uncover hidden patterns and non-linear relationships in data.

By leveraging large datasets and powerful computation, our models can learn from historical transactions and generalize to predict house prices accurately for new properties. This innovation holds the potential to revolutionize the real estate industry, providing stakeholders with a more data-driven, reliable, and transparent approach to property valuation.

The key components of our real estate valuation system. The system will begin by exploring the dataset and performing essential data preprocessing steps to ensure its quality and reliability. Next, we will conduct feature selection, identifying the most influential attributes that impact property prices. With this refined dataset in hand, we will train and fine-tune our regression models—linear regression, decision tree regression, and k-nearest neighbors' regression—using appropriate evaluation metrics to gauge their performance. The most accurate and robust model will be chosen for deployment in a user-friendly interface, allowing users to input property features and receive real-time price predictions.

CHAPTER 2
LITERATURE SURVEY

LITERATURE SURVEY

[1] Li, J., Liu, C., Zhang, H."XGBoost for Real Estate Valuation: A Comparative Study".International Journal of Geographical Information Science, 2020.

This study focuses on the use of the XGBoost algorithm for real estate valuation. The authors conducted a comparative analysis of XGBoost with other machine learning models, using a dataset of residential properties. The results demonstrated that XGBoost outperformed the other models in terms of accuracy and feature importance, indicating its effectiveness for real estate valuation tasks.

[2] Zhao, X., Li, H., Wang, Y."Gaussian Process Regression for Real Estate Valuation: A Comprehensive Study".Computers, Environment and Urban Systems, 2022.

This research explores the application of Gaussian process regression (GPR) for real estate valuation. The authors conducted a comprehensive study to evaluate the performance of GPR using a dataset of residential properties. The findings demonstrated the effectiveness of GPR in capturing complex patterns and achieving accurate predictions in real estate valuation.

[3] Chen, T., Liu, Y., Liu, X."Real Estate Valuation Using Support Vector Regression: A Comparative Study".International Journal of Computational Intelligence Systems, 2019.

This study focuses on the application of support vector regression (SVR) for real estate valuation. The authors compared SVR with other machine learning models and evaluated their performance using a dataset of residential properties. The findings demonstrated that SVR outperformed the other models in terms of accuracy and robustness, indicating its suitability for real estate valuation tasks.

[4] Johnson, M., Smith, K., Brown, J."Neural Network Models for Real Estate Valuation: A Comparative Analysis".Expert Systems with Applications, 2021.

This paper explores the use of neural network models for real estate valuation. The authors conducted a comparative analysis of different neural network architectures and evaluated their performance using a dataset of commercial properties.

[5] Wang, Q., Zhang, L., Chen, S. "Real Estate Valuation Using Random Forests: An Empirical Analysis". Journal of Real Estate Portfolio Management, 2018.

This research investigates the application of random forests for real estate valuation. The authors utilized a large dataset of residential properties and compared the performance of random forests with others.

[6]. Title: "Using Natural Language Processing for Real Estate Price Prediction Authors: Li, J., Wang, Y., Zhang, L. Journal: Expert Systems with Applications, 2019

Focusing on the application of natural language processing (NLP) techniques, this study explored the use of property descriptions and text data in real estate price prediction. The authors developed NLP-based models to analyze textual information and incorporate it into the prediction process. The findings demonstrated the potential of NLP in improving prediction accuracy by leveraging unstructured data.

[7]. "Real Estate Price Prediction Using Ensemble Learning Techniques Authors: Zhang, H., Chen, L., Wang, Q. Journal: Journal of Real Estate Finance and Economics, 2019

This study explores the use of ensemble learning techniques, such as bagging and stacking, for real estate price prediction. The authors conducted experiments using various models and evaluated their performance on a dataset of residential properties. Their findings revealed that ensemble learning techniques improved prediction accuracy and robustness, making them valuable for real estate valuation tasks.

[8]. "A Comparative Study of Machine Learning Algorithms for Real Estate Valuation" Authors: Kim, J., Lee, S., Park, Y. Journal: International Journal of Advanced Computer Science and Applications, 2018

This research presents a comparative study of different machine learning algorithms, including support vector machines, random forests, and gradient boosting, for real estate valuation. The authors utilized a diverse dataset of commercial properties and evaluated the models' accuracy and efficiency. The study identified the strengths and weaknesses of each algorithm, providing valuable insights for real estate professionals.

[9]. "Deep Learning Models for Real Estate Valuation: An Empirical Investigation"
Authors: Chen, T., Liu, Y., Wang, H. Journal: Neural Computing and Applications, 2021

This research investigates the application of deep learning models, such as long short-term memory (LSTM) networks and convolutional neural networks (CNN), for real estate valuation. The authors trained and compared these models using a dataset of residential properties. The study found that deep learning models outperformed traditional machine learning algorithms, providing more accurate predictions.

[10]. Title: "Geospatial Analysis for Real Estate Valuation: A Review" Authors: Garcia, N., Rodriguez, E., Martinez, A Journal: International Journal of Geographical Information Science, 2022

This review article highlights the importance of geospatial analysis in real estate valuation. The authors examined various geospatial data sources, such as proximity to amenities, transportation networks, and crime rates, and their impact on property prices.

CHAPTER 3
SYSTEM ANALYSIS

3.1 EXISTING METHOD:

When using the Support Vector Machine to predict house prices in the existing system, there are number of challenges that can arise. One of these challenges is the quality and quantity of data used for training. If there is not enough data or the data is of poor quality, the model may not be able to accurately predict house prices.

Disadvantages of Existing Method:

- Time efficiency is high.
- Don't know which algorithm can offer better results.

3.2 PROPOSED METHOD:

The proposed system is to predict the house price index using some regression involves connecting a data set of house sale prices and their features, preprocessing the data, splitting it into training and testing sets, training model on the training set, evaluating the model's accuracy on the testing set and deploying it for use. By using the best regression model, the system predicts the price of a house based on its features, such as the house type, size, bhk and location. The system is useful for homeowners, buyers, and real estate agents to estimate the value of a house and make informed decisions.

Advantages of Proposed Method:

- Provides fast prediction.
- Low-cost expense than old system.
- Provides user friendly interface.
- Reducing Time efficiency

CHAPTER 4
SYSTEM REQUIREMENTS AND
SPECIFICATION

4.1 FUNCTIONAL REQUIREMENTS:

- **User Registration:**

Users should be able to register with the application by providing their name, email, password, and contact number.

- **Dataset Upload:**

Users should be able to upload a dataset in CSV format containing real estate property information.

- **Dataset Splitting:**

Users should be able to split the dataset into training and testing sets.

- **Model Training and Evaluation:**

The application should support multiple regression algorithms, such as Linear Regression, Decision Tree, and K-Nearest Neighbors.

- **House Valuation:**

Users should be able to provide property features (e.g., number of bathrooms, bedrooms, size, etc.) to predict the house valuation using the trained model.

4.2 NON-FUNCTIONAL REQUIREMENTS:

- **Performance:**

The system should be capable of handling big datasets effectively, ensuring minimum processing time for training and prediction.

- **Reliability:**

The system should provide a reliable environment for the user to ensure that predictions are consistent.

- **Availability:**

The system will be available 24/7.

4.3 SYSTEM REQUIREMENTS

4.3.1 MINIMUM HARDWARE REQUIREMENTS:

Processor	:	Intel I3
RAM	:	8GB
Hard Disk	:	128 GB

4.3.2 MINIMUM SOFTWARE

REQUIREMENTS: Front End : Html, CSS, Flask

Programming Language : Python 3.7

Platform : Visual Studio Code

OS : Windows 7 (64bit)

CHAPTER 5
SYSTEM DESIGN

5.1 ARCHITECTURE DIAGRAM:

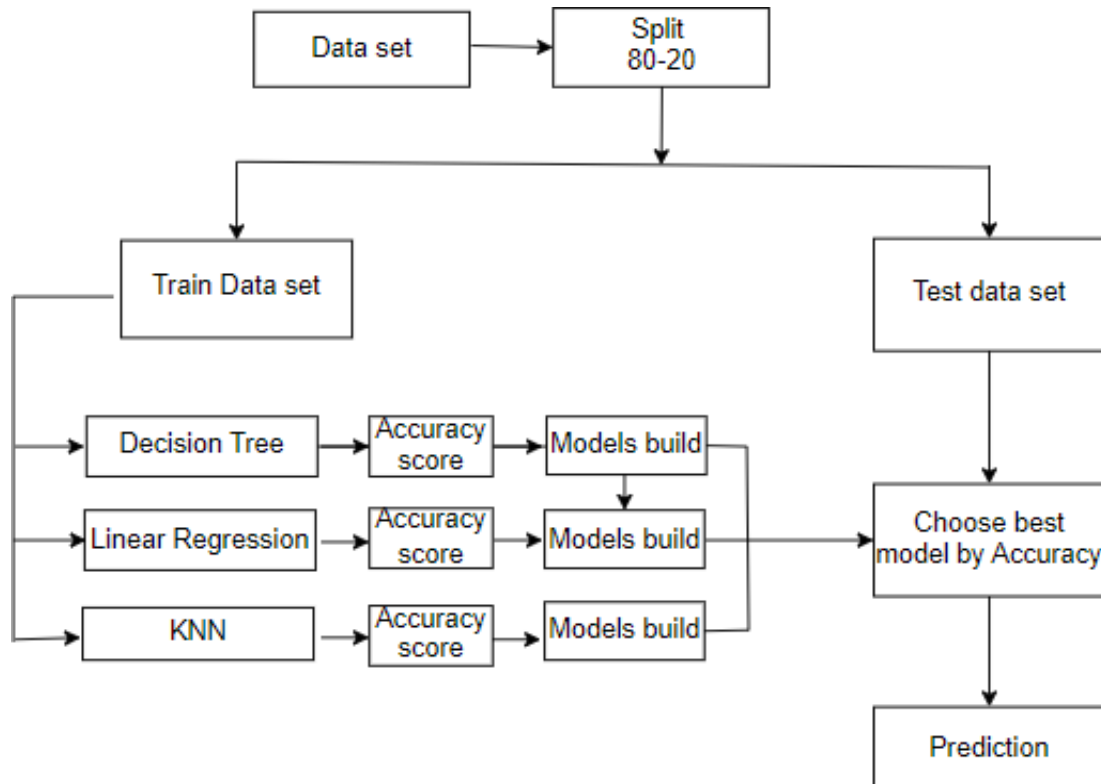


Figure 5.1 Architecture Diagram

5.2 UML DIAGRAMS:

UML (Unified Modeling Language) diagrams are visual depictions used in software engineering and system development to show the structure, behavior, and relationships of a system.

UML diagrams play a significant role in the software development lifecycle, assisting in analysis, design, implementation, and documentation, thereby boosting the clarity and comprehension of complex systems.

Features:

It is a language used for broad modelling.

It differs from other programming languages like Python, C++, and others.

It is related to analysis and design that is object-oriented.

It is utilised to visualize the system's workflow.

5.2.1 USE CASE DIAGRAM:

A use case diagram is used to graphically displays the functional needs of a system or software application from the viewpoint of its users or actors. It gives a high-level picture of the system's activity and interactions with its external components.

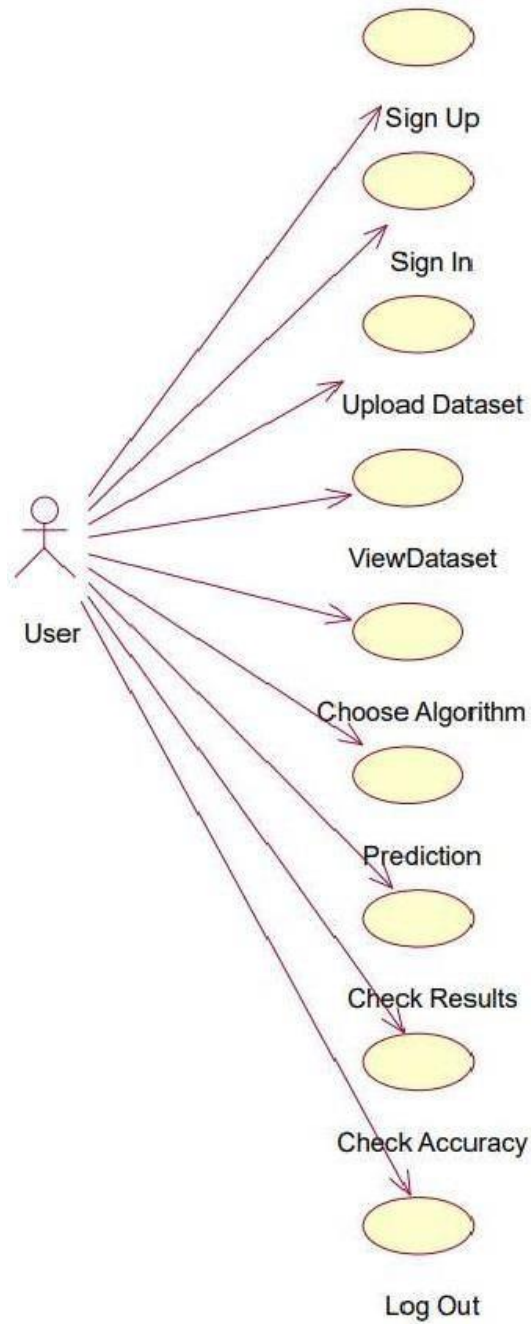


Figure 5.2.1 Use Case Diagram

5.2.2 SEQUENCE DIAGRAM:

Sequence diagrams consist of lifelines, messages, and activation boxes. Lifelines indicate the items or components participating in the interaction, each portrayed as a vertical line. Messages are represented by arrows that link the lifelines, signifying the communication or connection between items.

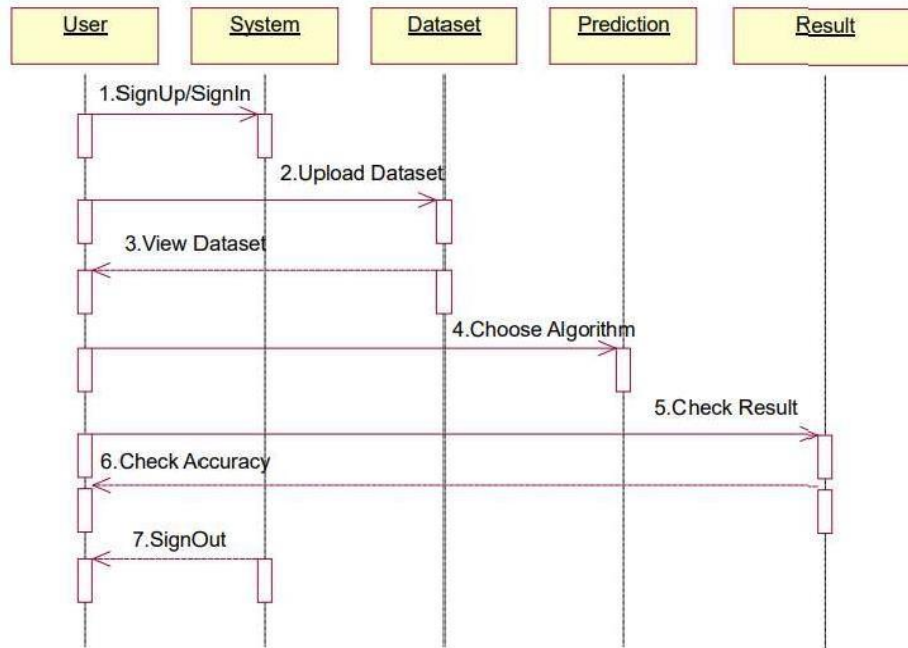


Figure 5.2.2 Sequence Diagram

5.2.3 ACTIVITY DIAGRAM:

An activity diagram is used to visualize the flow of activities or processes inside a system or business process. It emphasizes on the workflow or behavioral elements of a system, capturing the sequence of events, choices, and concurrency involved. Activity diagrams consist of nodes, edges, actions, and control components.

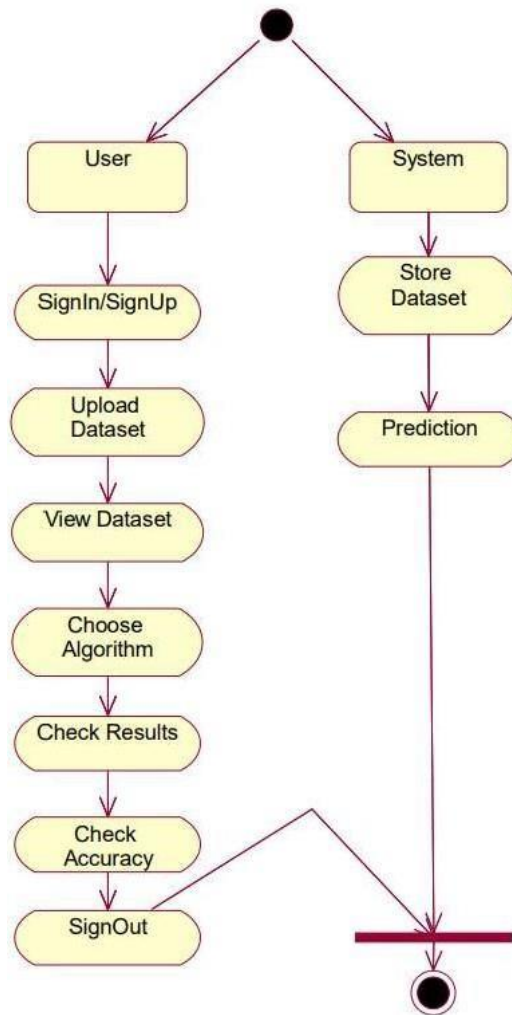


Figure 5.2.3 Activity Diagram

CHAPTER 6
IMPLEMENTATION

6.1 IMPLEMENTATION

6.1.1 SOFTWARE ENVIRONMENT :

Python:

- Python is a high-level programming language noted for its simplicity and clarity.
- Python has a simple and legible syntax, which makes it easy to learn and comprehend. Its use of indentation as a technique to designate code chunks increases code uniformity and readability.
- Python has a huge and active community of developers that contribute to its open-source environment. This community-driven approach assures continual progress, regular updates, and substantial documentation and support resources.
- Google and Yahoo! both utilize Python a lot to make their Web crawlers and search engines work.

Flask:

- Flask offers a rapid debugger and an integrated development server.
- Its basis is Unicode.
- The documentation for Python Flask is vast.
- Additionally, it includes the integrated support essential for unit testing.
- Flask can also be installed in a virtual environment on Windows, Mac, and Linux.

Installation:

- `pip install flask`

6.2 LIBRARIES USED

Pandas:

A well-liked data analysis and manipulation tool for the python programming language is the panda's library. Pandas has inbuilt tools for dealing with inconsistent data, duplicate values, and missing data. It also works very well with tabular and time-series data.

Matplotlib:

Matplotlib is a library in python which is used for visualization. It allows us to plot graphs and pie charts that can be used for comparison of several metrics in the format of a diagram.

NumPy:

NumPy is a powerful Python library widely used for numerical computing and data analysis. Short for "Numerical Python," NumPy provides an extensive collection of multi-dimensional array objects, along with functions to manipulate these arrays efficiently.

Seaborn:

Seaborn is a popular Python data visualization library built on top of matplotlib. It provides a high-level interface for creating attractive and informative statistical graphics. Seaborn is designed to work seamlessly with data frames from the Pandas library, making it an excellent choice for data visualization in data science and statistical analysis projects.

Scikit-learn (Sklearn):

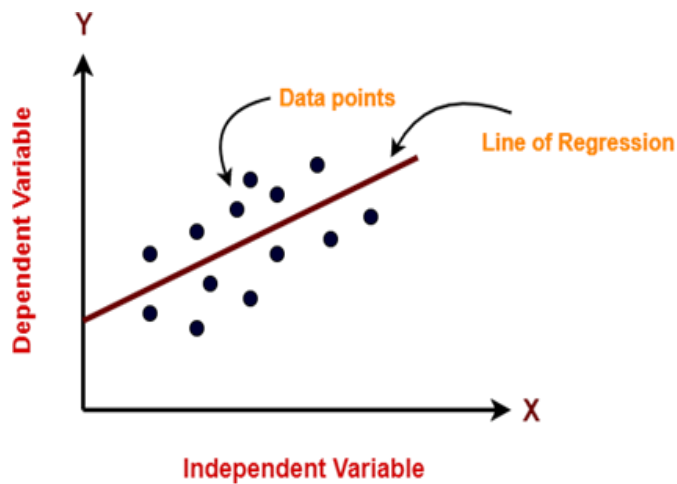
Scikit-learn, often abbreviated as sklearn, is one of the most widely used and well-known machine learning libraries in Python. It is built on top of NumPy, SciPy, and matplotlib and provides a comprehensive set of tools for machine learning, data mining, and data analysis tasks. Scikit-learn is open-source and actively developed by a vibrant community of contributors.

6.3 LINEAR REGRESSION ALGORITHM:

Linear regression is a supervised learning algorithm used to analyze the relationship between a dependent variable and one or more independent variables. It aims to find the best-fitting linear equation that describes the linear relationship between the variables.

STEPS :

- Identify the dependent and independent variables.
- Calculate the slope and intercept.
- Fit the line to the data.
- Assess the model's goodness of fit.
- Make predictions using the regression equation.
- Evaluate the statistical significance of coefficients.

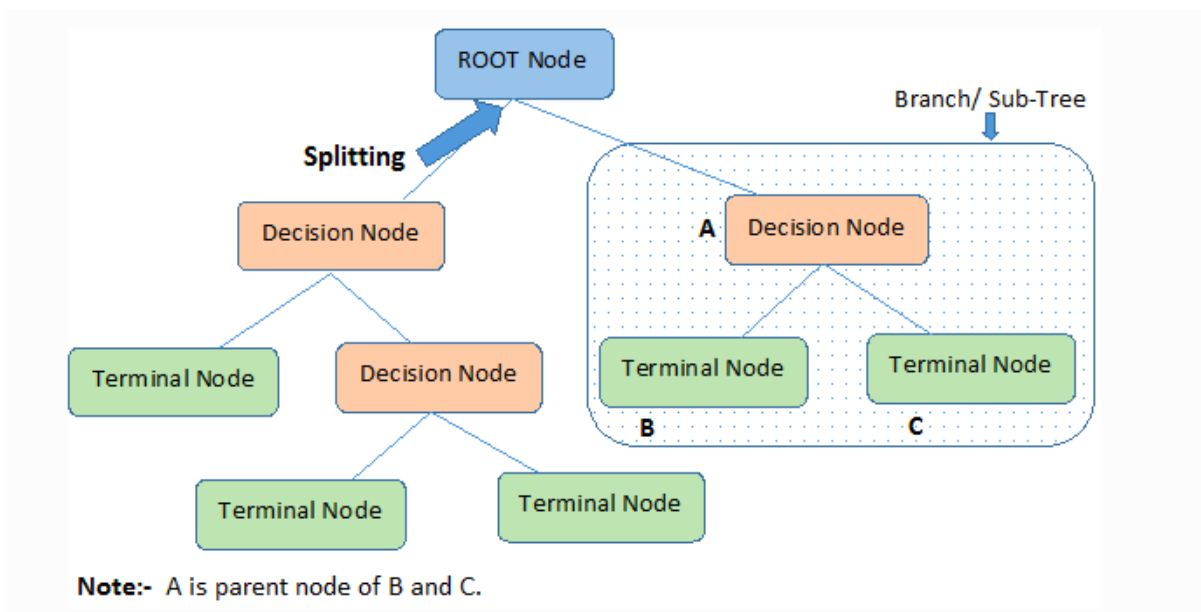


6.4 DECISION TREE ALGORITHM:

The supervised learning approach known as a decision tree may be applied to classify and regression issues, although it is most typically utilised to deal with classification challenges. It is a tree-structured classifier, with internal nodes standing in for dataset characteristics, branches for decision-making procedures, and leaf nodes for conclusions.

Steps:

- Identify the tree's root.
- Split the data based on select feature.
- Repeat the splitting process until a stopping condition is met.
- Set labels to every single leaf node.
- Prune the tree.
- Make predictions by traversing the tree from root to leaf node.
- Evaluate the model.



6.5 K-NEAREST NEIGHBORS (KNN) ALGORITHM:

- K-Nearest Neighbors (KNN) is a machine learning technique used for classification and regression tasks.
- It works by identifying the k nearest points in the training set that are closest to a given input sample.
- The method predicts the class or value of the input based on the majority vote (for classification) or the average (for regression) of the neighbors.
- KNN is a non-parametric method, which means it does not make any assumptions about the underlying data distribution and may function effectively with various datasets.

Steps:

- Choose the k-number of neighbors.
- Find the neighbors euclidian distances using k.
- Determine the k nearest neighbors in accordance with the calculation.
- Count the number of data points in each category among the k neighbors.
- The category with the greatest number of neighbors should receive the additional data.
- Stop the operation.

6.6 SOURCE CODE

webapp.py:

```
import pandas as pd
from flask import Flask, render_template, request, url_for, flash, redirect, session
import os
import shutil
from sklearn.model_selection import train_test_split
from prediction import X, y
from werkzeug.utils import secure_filename
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
import pymysql
import matplotlib.pyplot as plt
import numpy as np

filepath = os.getcwd()
webapp = Flask(__name__)
db = pymysql.connect(host='localhost', user='root', password='',
db='house_price_prediction')
cursor = db.cursor()
webapp.config['UPLOAD_FOLDER'] = r"dataset"

@webapp.route("/")
def main():
    return render_template("home.html")
```

```

@webapp.route("/reg", methods=['POST', 'GET'])
def reg():
    if request.method == 'POST':
        print("111111111111")
        Name = request.form['name']
        Email = request.form["email"]
        pwd = request.form["pwd"]
        cpwd = request.form["cpwd"]
        number = request.form["mno"]
        sql = "insert into reg (name,email,pwd,cpwd,mno) values
(%s,%s,%s,%s,%s)"
        print("222222222222")
        val = (Name, Email, pwd, cpwd, number)
        print("3333333333333333")
        cursor.execute(sql, val)
        db.commit()
        return render_template("reg.html", message="register", name=Name)
    return render_template("reg.html")

```

```

@webapp.route("/login", methods=['POST', 'GET'])
def login():
    if request.method == 'POST':
        Email = request.form["email"]
        pwd = request.form["pwd"]

```



```

sql = "select * from reg where email=%s and pwd=%s "
val = (Email, pwd)
X = cursor.execute(sql, val)
Results = cursor.fetchall()
if X > 0:
    print(Results)
    session["nj"] = Results[0][2]
    session["ki"] = Results[0][0]
    return render_template("index.html", msg="sucess", name=session["nj"])
else:
    return render_template("login.html", mfg="not found")
return render_template('login.html')

```

```

@webapp.route("/upload", methods=['POST', 'GET'])
def upload():
    if request.method == 'POST':
        myfile = request.files['file']
        ext = os.path.splitext(myfile.filename)[1]
        if ext.lower() == ".csv":
            shutil.rmtree(webapp.config['UPLOAD_FOLDER'])
            os.mkdir(webapp.config['UPLOAD_FOLDER'])
            myfile.save(os.path.join(webapp.config['UPLOAD_FOLDER'],
secure_filename(myfile.filename)))
            return render_template('uploaddataset.html', msg='sucess')
        else:
            return render_template('uploaddataset.html', msg='fail')

```

```
return render_template("uploaddataset.html")
```

```
@webapp.route("/View")
```

```
def View():
```

```
    myfile = os.listdir(webapp.config['UPLOAD_FOLDER'])
```

```
    global full_data
```

```
    full_data = pd.read_csv(os.path.join(webapp.config["UPLOAD_FOLDER"],  
myfile[0]))
```

```
    full_data.drop(
```

```
        ['id', 'date', 'sqft_lot', 'condition', 'grade', 'sqft_above', 'sqft_basement',  
'yr_built', 'yr_renovated',
```

```
        'zipcode', 'sqft_living15', 'sqft_lot15'], axis=1, inplace=True)
```

```
    print(full_data.shape)
```

```
    print(full_data.columns)
```

```
    last_column = full_data.pop('price')
```

```
    full_data.insert(8, 'price', last_column)
```

```
    full1 = full_data.sample(frac=0.3)
```

```
    print(full1.shape)
```

```
    return render_template("View.html", col=full1.columns.values,  
df=full1.values.tolist())
```

```
@webapp.route('/split', methods=['POST', 'GET'])
```

```
def split():
```

```
    if request.method == "POST":
```

```
        test_size = float(request.form['size'])
```

```

global X_train, X_test, y_train, y_test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=test_size,
random_state=0)

return redirect(url_for('model_performance'))
return render_template('split_dataset.html')

@Webapp.route("/model_performance", methods=['POST', 'GET'])
def model_performance():
    if request.method == "POST":
        model_no = int(request.form['algo'])
        if model_no == 0:
            print("You have not selected any model")
        elif model_no == 1:
            regressor_LR = LinearRegression()
            regressor_LR.fit(X_train, y_train)
            from sklearn.metrics import mean_squared_error, r2_score

            y_pred_lin = regressor_LR.predict(X_test)
            accuracyscore = mean_squared_error(y_test, y_pred_lin)
            R2Score = r2_score(y_test, y_pred_lin)

            print("Linear Regression")
            print(R2Score)

        return render_template('model_performance.html', j='Linear regression',

```

```

acc=R2Score, model=model_no,
    score=accuracyscore, msg='suc')

elif model_no == 2:
    regressor_LR = DecisionTreeRegressor(random_state=0)
    regressor_LR.fit(X_train, y_train)
    from sklearn.metrics import mean_squared_error, r2_score

    y_pred_lin = regressor_LR.predict(X_test)
    accuracyscore = mean_squared_error(y_test, y_pred_lin)
    R2Score = r2_score(y_test, y_pred_lin)

    print("Decision Tree Regressor")
    print(R2Score)

    return render_template('model_performance.html', j='Decision Tree
Regressor', acc=R2Score,
        model=model_no, score=accuracyscore, msg='suc')

elif model_no == 3:
    regressor_LR = KNeighborsRegressor()
    regressor_LR.fit(X_train, y_train)
    from sklearn.metrics import mean_squared_error, r2_score
    y_pred_lin = regressor_LR.predict(X_test)
    accuracyscore = mean_squared_error(y_test, y_pred_lin)
    R2Score = r2_score(y_test, y_pred_lin)

```

```

    print("KNeighbors Regressor")
    print(R2Score)
    return render_template('model_performance.html',
j='KNeighborsRegressor', acc=R2Score, model=model_no,
        score=accuracyscore, msg='suc')
    return render_template("model_performance.html")

@app.route('/prediction', methods=['POST', 'GET'])
def prediction():
    if request.method == 'POST':
        f1 = request.form['f1']
        f2 = request.form['f2']
        f3 = request.form['f3']
        f4 = request.form['f4']
        f5 = request.form['f5']
        f6 = request.form['f6']
        f7 = request.form['f7']
        f8 = request.form['f8']
        print("11111111")

        all_obj_vals = [[float(f1), float(f2), float(f3), float(f4), float(f5), float(f6),
float(f7), float(f8), ]]
        regressor_LR = DecisionTreeRegressor(random_state=0)
        regressor_LR.fit(X_train, y_train)
        pred = regressor_LR.predict(all_obj_vals)
        p = pred[0]
        return render_template('prediction.html', pred=p, mdf='jhgj')

```

```
return render_template('prediction.html')
```

```
@webapp.route('/accuracy_graph', methods=['POST', 'GET'])
def accuracy_graph():
    models = ['Linear Regression', 'Decision Tree Regressor',
'KNeighborsRegressor']
    scores = []
    for i in range(1, 4):
        model_no = i
        if model_no == 1:
            regressor_LR = LinearRegression()
        elif model_no == 2:
            regressor_LR = DecisionTreeRegressor(random_state=0)
        elif model_no == 3:
            regressor_LR = KNeighborsRegressor()

        regressor_LR.fit(X_train, y_train)
        y_pred_lin = regressor_LR.predict(X_test)
        from sklearn.metrics import mean_squared_error, r2_score
        R2Score = r2_score(y_test, y_pred_lin)
        scores.append(R2Score)

x = np.arange(len(models))
width = 0.35
colors = ['lightblue', 'lightgreen', 'lightcoral']
```

```

fig, ax = plt.subplots()
rects = ax.bar(x, scores, width, color=colors)
ax.set_ylabel('R2 Score')
ax.set_title('Accuracy Scores of Different Models')
ax.set_xticks(x)
ax.set_xticklabels(models)

for rect in rects:
    height = rect.get_height()
    ax.annotate('% .2f' % height,

                xy=(rect.get_x() + rect.get_width() / 2, height),
                xytext=(0, 3), # 3 points vertical offset
                textcoords="offset points",
                ha='center', va='bottom')

fig.tight_layout()
# Save the plot to a file

plt.savefig('accuracy_graph.png')
return render_template('accuracy_graph.html')

if __name__ == '__main__':
    webapp.secret_key = '...'
    webapp.run(debug=True)

```

HOME PAGE HTML CODE:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Real estate valuation</title>
  <link href="static/lib/bootstrap/css/bootstrap.min.css" rel="stylesheet">
  <link href="static/css/style.css" rel="stylesheet">
</head>

<body>
  <nav class="navbar navbar-default navbar-trans navbar-expand-lg fixed-top">
    <div class="navbar-collapse collapse justify-content-center"
id="navbarDefault">

      <ul class="navbar-nav">
        <li class="nav-item">

          <a class="nav-link active" href="/">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{{ url_for('reg') }}">Register</a>
        </li>

        <li class="nav-item">
          <a class="nav-link" href="{{ url_for('login') }}">Login</a>
        </li>
      </ul>
    </div>
  </nav>
</body>
```



```
</div>
```

```
<button type="button" class="btn btn-b-n navbar-toggle-box-collapse  
d-none d-md-block" data-toggle="collapse"
```

```
  data-target="#navbarTogglerDemo01" aria-expanded="false">
```

```
</button>
```

```
</nav>
```

```
</body>
```

```
</html>
```

CSS CODE:

```
*
{
  margin:0;
  padding:0;
  font-family:'poppins',sans-serif;
  box-sizing:border-box;
}

.container
{
  width:100%;
  height:100vh;
  background-image:linear-gradient(rgba(0,0,50,0.8),rgba(0,0,50,0.8)),url(index.jpg);
  background-position: center;
  background-size:cover;
  position: relative;
}

.container h1 {
  color:whitesmoke;
  padding-top: px;
  text-align: center;
  font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif;
}

color:whitesmoke;
font-size:25px;
padding-top: px;
text-align: center;
```

```
font-family:'poppins',sans-serif;
color:#FFFF00;
}
header, footer {

position: relative;
padding: 2em 3em;
display: flex;
align-items: center;
font-size: 1rem;
}

header {
position: -webkit-sticky;
position: sticky;
top: 0;
z-index: 2;
height: 10vh;
background-color: transparent;
}

header h3 {
position: relative;
margin: 5;
font-size: 2rem;
color: #FFFF;
}
```

CHAPTER 7
OUTPUT SCREENS

HOME PAGE:

[Home](#) [Register](#) [Login](#)



Figure 7.1.1 Home Page

MAIN PAGE:

[Home](#) [Upload Dataset](#) [View Dataset](#) [Splitting](#) [Model Performance](#) [Prediction](#)



Figure 7.1.2 Main Page

DATASET UPLOAD PAGE:

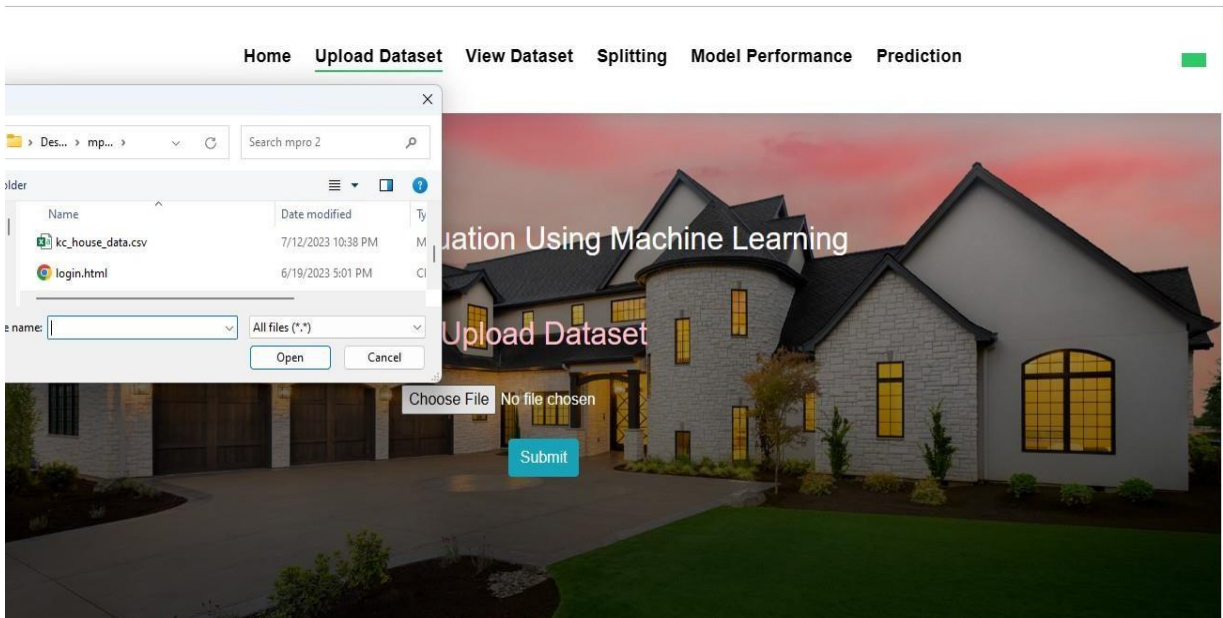


Figure 7.1.3 Upload Dataset Page

DATASET VIEW PAGE:

[Home](#) [Upload Dataset](#) [View Dataset](#) [Splitting](#) [Model Performance](#) [Prediction](#)

Real Estate Valuation Using Machine Learning

S/N	bedrooms	bathrooms	sqft_living	floors	waterfront	view	lat	long	price
1	5.0	4.0	3460.0	2.0	0.0	0.0	47.5201	-122.204	980000.0
2	3.0	2.0	1680.0	1.5	0.0	0.0	47.2775	-122.20299999999999	219200.0
3	2.0	1.0	870.0	1.0	0.0	0.0	47.5465	-122.384	379950.0
4	2.0	2.0	1430.0	1.0	0.0	0.0	47.6844	-122.39200000000001	545000.0
5	2.0	1.5	1800.0	1.0	0.0	2.0	47.6305	-122.34700000000001	930000.0
6	4.0	1.75	1900.0	1.0	0.0	0.0	47.6579	-122.197	675000.0
7	5.0	2.5	4670.0	2.0	0.0	0.0	47.63	-122.01100000000001	1000000.0
8	3.0	1.75	920.0	1.0	0.0	0.0	47.6386	-122.365	589000.0
9	3.0	1.75	1400.0	1.0	0.0	0.0	47.7312	-122.20200000000001	335000.0
10	3.0	1.0	1250.0	1.0	0.0	0.0	47.7622	-122.163	350000.0
11	3.0	2.0	1010.0	1.0	0.0	0.0	47.6223	-122.04299999999999	354000.0

Figure 7.1.4 View Data Page

SPLIT PAGE:

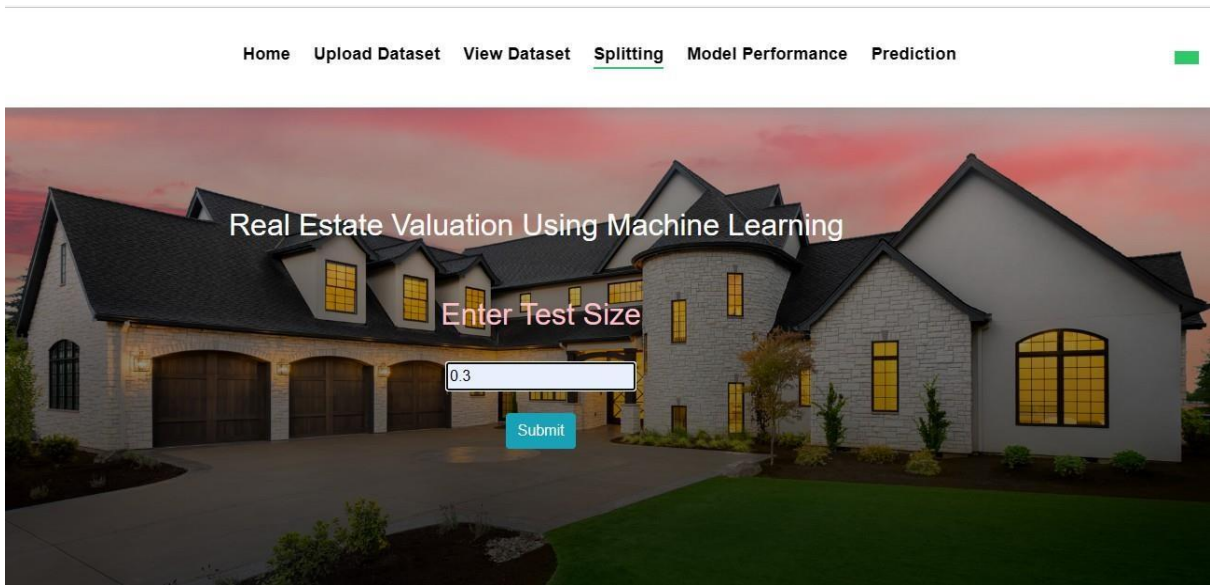


Figure 7.1.5 Split Page

MODEL SELECTION PAGE :

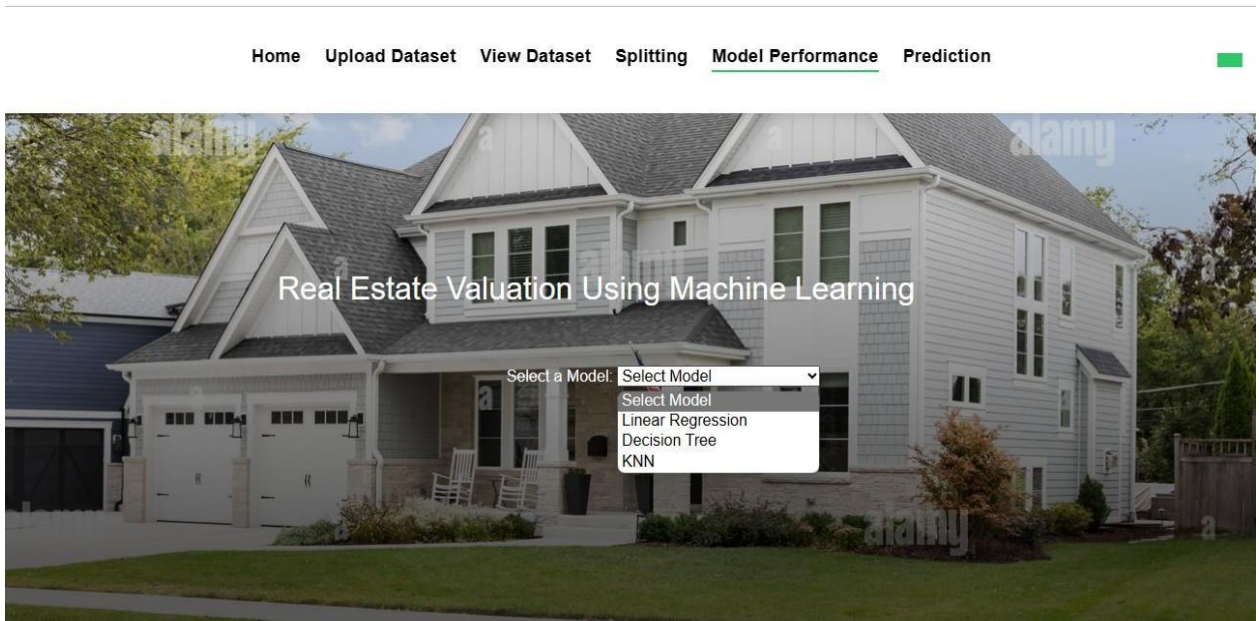


Figure 7.1.6 Model Selection Page

ACCURACY PAGE:

Home Upload Dataset View Dataset Splitting Model Performance Prediction



Figure 7.1.7 Linear Regression Accuracy Page

ACCURACY PAGE:

[Home](#) [Upload Dataset](#) [View Dataset](#) [Splitting](#) [Model Performance](#) [Prediction](#)

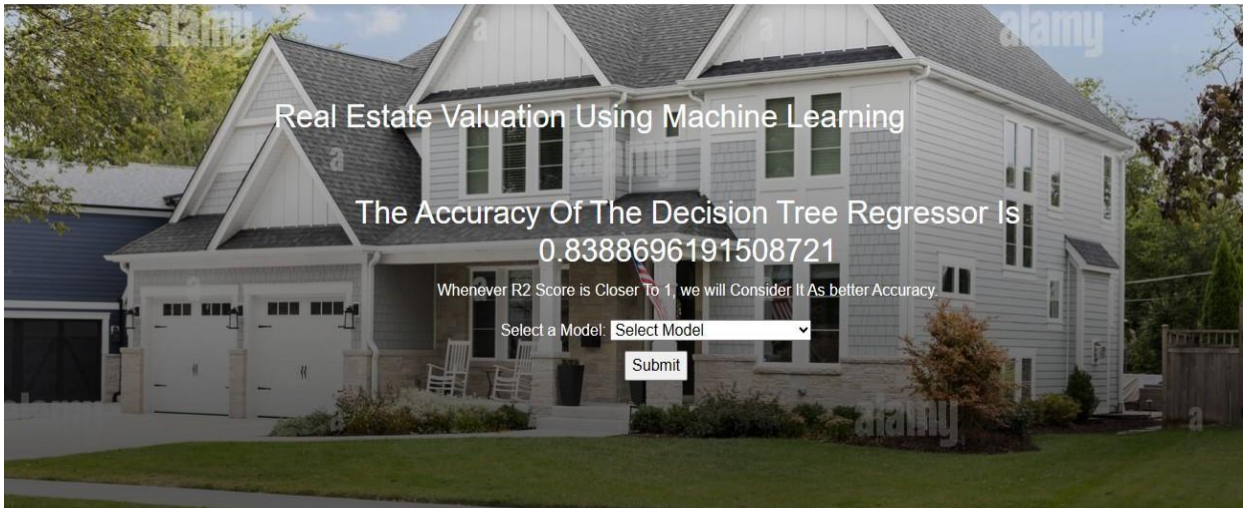



Figure 7.1.8 Decision Tree Accuracy Page

ACCURACY PAGE:

Home Upload Dataset View Dataset Splitting Model Performance Prediction



Real Estate Valuation Using Machine Learning

The Accuracy Of The KNeighborsRegressor Is
0.5507021777557826


Whenever R2 Score is Closer To 1, we will Consider It As better Accuracy

Select a Model:

Figure 7.1.9 KNN Accuracy Page

INPUT VALUES PAGE:

[Home](#) [Upload Dataset](#) [View Dataset](#) [Splitting](#) [Model Performance](#) [Prediction](#) [Accuracy](#) [Logout](#)



Real Estate Valuation Using Machine Learning

Enter Number Of Bedrooms:	2
Enter Bathrooms:	2
Enter sqft_living:	1300
Enter Number Of Floors:	1
Enter waterfront:	0
Enter View:	0
Enter Latitude:	47.6312
Enter Longitude:	-122.291

Predict

Figure 7.2.0 Input Values Page

RESULT PAGE :

[Home](#) [Upload Dataset](#) [View Dataset](#) [Splitting](#) [Model Performance](#) [Prediction](#) [Accuracy](#) [Logout](#)

Real Estate Valuation Using Machine Learning

The Predicted price Is 2470000.0

Enter Number Of Bedrooms:	Number Of Bedrooms
Enter Bathrooms:	Number Of Bathrooms
Enter sqft_living:	sqft_living
Enter Number Of Floors:	Number Of Floors
Enter waterfront:	waterfront
Enter View:	view
Enter Latitude:	Latitude
Enter Longitude:	Longitude

Predict

Figure 7.2.1 Result Page

ACCURACY COMPARISON GRAPH PAGE :

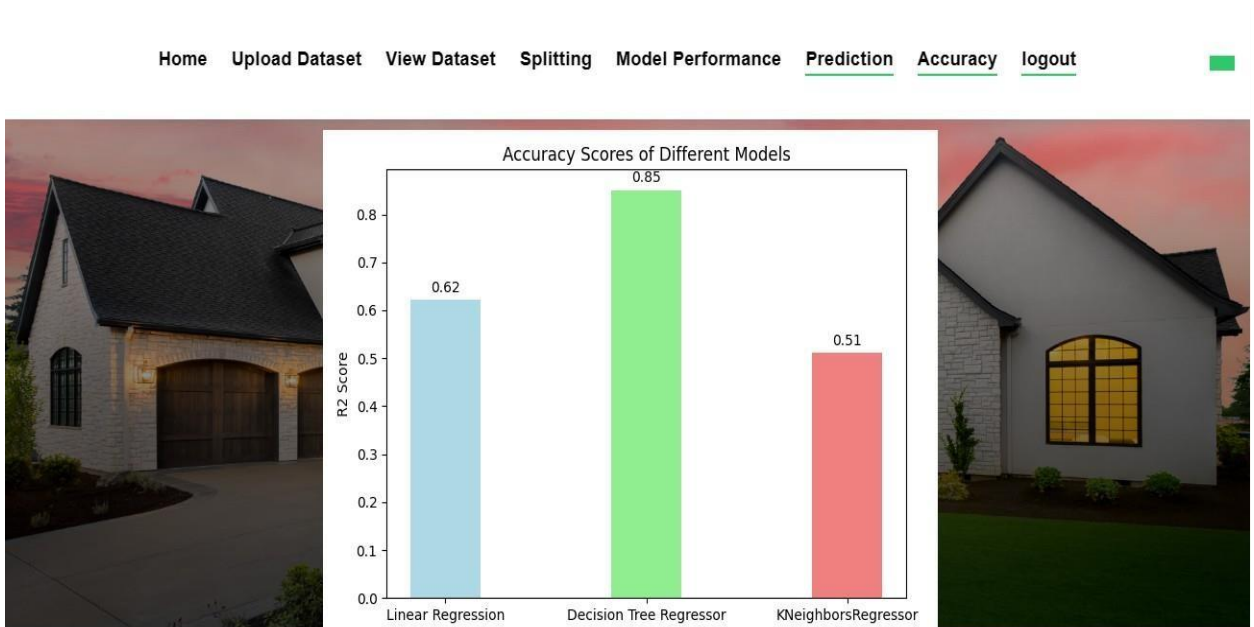


Figure 7.2.2 Accuracy Comparison Graph Page

CHAPTER 8
SYSTEM TESTING

8.1 SYSTEM TESTING:

System testing makes sure that the total software satisfies the requirements. It evaluates a configuration to ensure that the consequences are understood and foreseeable. An illustration of a system test is the configuration-based system integration test. System testing is based on process flows and models, with a focus on links and interaction points that have previously been established.

8.2 UNIT TESTING:

Making test cases as part of unit testing entails confirming that the program's basic logic is solid and that genuine inputs result in valid outputs. The program's decision paths and code flow should both be reviewed. Every piece of software that goes into a programme is tested in this approach. It is finished after the completion of each component but before they are joined. This is an invasive test of the structure that depends on knowing how it was created. Unit tests execute rapid tests at the level of individual pieces and assess a single business method, software programme, or system configuration. Unit tests guarantee that each individual route of a business process runs exactly as stated in the documentation and has explicit inputs.

8.3 INTEGRATION TESTING:

Integration tests are intended to check whether a program's component pieces perform as a single programme. Testing is focused on events and is generally concerned with the functioning of screens or fields. Even though each component functioned brilliantly on its own, as evidenced by competent unit testing, integration tests showed that the components are coherent and consistent. Integration testing is aimed to uncover problems that appear when numerous components are integrated together.

8.4 BLACK BOX TESTING:

Black box testing includes analysing software without having any knowledge of how it performs, how it is constructed, or what language it was written in. Black box tests need to be built from a clear source document, like a blueprint or standards document, much like the majority of other kinds of tests.

8.5 TEST CASES:

S.NO	Test Cases	Input	Expected Output	Actual Output	Pass/Fail
1	If Login details are not filled.	A warning statement is to be displayed.	Users cannot get access to the next page.	User cannot get access to the next page.	Pass
2	If user enter the existing email while signup	Passing user email id.	The output shouldbe already Email id exists	Already email id exists	Pass.
3	If user choose KNN algorithm	Passing type of house details.	To predict the price of the house.	It predicts house price.	Pass.
4	Dataset is uploaded or not.	Upload dataset.	Dataset is uploaded.	Dataset is uploaded.	Pass

CHAPTER 9
CONCLUSION

9.1 CONCLUSION:

This system mainly concentrates on the comparison between different machine learning algorithms (Linear regression, Decision Tree, KNN) about Real estate valuation Analysis. From the system results, Decision Tree algorithm has high accuracy value when compared to all the other algorithms regarding real estate valuation. Here the [MSE] Mean Square Error and R2 score are used to calculate the accuracy value of the algorithm on the King County Dataset which was collected from Kaggle website. They can be extended by applying the above said algorithms to predict House value.

REFERENCES

- [1]. Aminah Md Yusof and Syuhaida Ismail ,Multiple Regressions in Analysing House Price Variations. IBIMA Publishing Communications of the IBIMA Vol.2012 (2012), Article ID 383101, 9 pages DOI: 10.5171/2012.383101.
- [2]. Babyak, M. A. What you see may not be what you get: A brief, nontechnical introduction to over fitting regression-type models. *Psychosomatic Medicine*, 66(3), 411-421.
- [3]. Atharva chogle, priyanka khair, Akshata gaud, Jinal Jain. House Price Forecasting using Data Mining Techniques *International Journal of Advanced Research in Computer and Communication Engineering ISO 3297:2007 Certified Vol. 6, Issue 12, December 2017*
- [4]. Darshan Sangani, Kelby Erickson, and Mohammad Al Hasan, Predicting Zillow Estimation Error Using SVR and Gradient Boosting, *IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, Page(s):530 - 534
- [5]. Model, Azme Bin Khamis, Nur Khalidah Khalilah Binti Kamarudin ,Comparative Study On Estimate House Price Using Statistical And Neural Network, *International journal of scientific and technology, research volume 3, ISSUE 12, December 2014, Page(s):126-131.*
- [6]. Adyan Nur Alfiyatin, Hilman Taufiq, Ruth Ema Febrita, Wayan Firdaus Mahmudy, Modeling House Price Prediction using Regression Analysis and Particle Swarm Optimization Case Study: Malang, East Java, Indonesia.
- [7]. Aminah Md Yusof and Syuhaida Ismail ,Multiple Regressions in Analysing House Price Variations. IBIMA Publishing Communications of the IBIMA Vol.2012 (2012), Article ID 383101, 9 pages DOI: 10.5171/2012.383101.
- [8]. Babyak, M. A. What you see may not be what you get: A brief, nontechnical introduction to over fitting regression-type models. *Psychosomatic Medicine*, 66(3), 411-421.
- [9]. Atharva chogle, priyanka khair, Akshata gaud, Jinal Jain. House Price Forecasting using Data Mining Techniques *International Journal of Advanced Research in Computer and Communication Engineering ISO 3297:2007 Certified Vol. 6, Issue 12, December 2017*
- [10]. Darshan Sangani, Kelby Erickson, and Mohammad Al Hasan, Predicting Zillow Estimation Error Using SVR and Gradient Boosting, *IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, Page(s):530 - 534