**HOUSE PRICE PREDICTION USING MACHINE LEARNING AND DEPLOYMENT IN FLASK**

**PROJECT REPORT**

***Submitted by***

**ESWARI. V – 513519104007**

**HARINI. S – 513519104010**

**KEERTHI. K – 513519104017**

**KOKILA. M – 513519104018**

***in partial fulfillment for the award of the degree***

***Of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**ANNAI MIRA COLLEGE OF ENGINEERING AND**

**TECHNOLOGY**



**ANNA UNIVERSITY :: CHENNAI 600 025**

**JUNE 2022**

**HOUSE PRICE PREDICTION USING MACHINE LEARNING AND DEPLOYMENT IN FLASK**

**PROJECT REPORT**

***Submitted by***

**ESWARI. V – 513519104007**

**HARINI. S – 513519104010**

**KEERTHI. K – 513519104017**

**KOKILA. M – 513519104018**

***in partial fulfillment for the award of the degree***

***Of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**ANNAI MIRA COLLEGE OF ENGINEERING AND**

**TECHNOLOGY**



**ANNA UNIVERSITY :: CHENNAI 600 025**

**JUNE 2022**

**ANNA UNIVERSITY : CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report “**HOUSE PRICE PREDICTION USING MACHINE LEARNING AND DEPLOYMENT IN FLASK”** is the bonafide work of “**V. ESWARI, S. HARINI, K. KEERTHI AND M. KOKILA” who** carried out the project work under my supervision.

|  |  |
| --- | --- |
| **SIGNATURE** **Mr. S. Srinivasan, M. Tech, Ph.D.,** **HEAD OF THE DEPARTMENT**  Dept. Of Computer Science &  Engineering,  Annai Mira College Of  Engineering & Technology,  Arapakkam, Ranipet. | **SIGNATURE**  **Miss. M. Varalakshmi, M.E,**  **SUPERVISOR**  Dept. Of Computer Science &  Engineering,  Annai Mira College Of  Engineering & Technology,  Arapakkam, Ranipet. |
| Submitted for the project viva voce held on …..................................... | |
| **INTERNAL EXAMINER** | **EXTERNAL EXAMINER**  **i** |

**ACKNOWLEDGEMENT**

I first offer my deepest gratitude to the Almighty GOD who has given me strength and good health during the course of the project.

I am conscious about my indebtedness our Honorable Chairman **Mr. S.Ramadoss,** Secretary **Mr. G.Thamothiran** and our Principal **Dr. T. K. Gopinathan.P.HD,** and our vice Principal **Dr. D. Saravanan.P.HD** who inspired me reach greater heights in the pursuits of knowledge.

I articulate my gratitude to my Head of the Department **Mr.S.Srinivasan.M.Tech. P.HD,** and internal guide **Miss.M.Varalakshmi.M.Tech,** had taught me the way to do a successful project and without whose constant encouragement and whole idea, the project would not have been possible.

I also express my sincere thanks to our Project Coordinator and all my staff of CSE Department who helped me in completing this project.

**ii**

**ABSTRACT**

Predictive models for determining the sale price of houses in cities like Bengaluru, Chennai, Hyderabad is still remaining as more challenging and trickier task. The rent price of houses in cities depends on a number of interdependent factors. Key factors that might affect the price include area of the house, location of the house and its amenities. In this research work, an analytical study has been carried out by considering the data set that remains open to the public by illustrating the available housing price of the respected house in the given location as per the requirements of the house seekers and with their needs and expectations. This makes them to come under the machine hackathon platform. The data set has the characteristics of the housing in respect to their location, internal and external features. In this study, an attempt has been made to construct a predictive model for evaluating the price based on the factors that affect the price.

The already existing model has the Linear Regression Algorithm for the prediction of house prices, but here we are going to propose the model with the Linear Regression Algorithm with Deployment Using Flask. The deployment of Flask is very easy to make Restful API’s using Python.

**iii**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TITLE** | **PAGE NO** |
|  | **BONAFIDE CERTIFICATE** | **i** |
|  | **ACKNOWLEDGEMENT** | **ii** |
|  | **ABSTRACT** | **iii** |
|  | **LIST OF FIGURES** | **iv** |
|  | **LIST OF ABBREVIATIONS** | **v** |
| **1.** | **INTRODUCTION**  1.1 OBJECTIVE  1.2 OVERVIEW OF PROJECT | **1** |
| **2.** | **LITERATURE SURVEY**  2.1 INTRODUCTION | **2** |
| **3.** | **SYSTEM ANALYSIS**  **3.1 EXISTING SYSTEM**  3.1.1 DISADVANTAGE  **3.2 PROPOSED SYSTEM**  3.2.1 ADVANTAGE  3.2.2 PROBLEM DEFINITION | **6**  **6**  **6**  **7**  **7**  **7** |
| **4.** | **SYSTEM ARCHITECTURE**  4.1 OVERVIEW OF SYSTEM  ARCHITECTURE | **8**  **8** |
| **5.** | **SYSTEM SPECIFICATION**  5.1 HARDWARE RQUIREMENTS  5.2 SOFTWARE REQUIREMENTS  **iv** | **9**  **9**  **9** |

|  |  |  |
| --- | --- | --- |
| **6.** | **MODULES**  6.1 USER MODULE  6.2 DATA MODULE | **10**  **10**  **11** |
| **7.** | **UML DIAGRAM**  7.1 USE CASE DIAGRAM  7.2 SEQUENCE DIAGRAM  7.3 CLASS DIAGRAM  7.4 DEPLOYMENT DIAGRAM  7.5 ACTIVITY DIAGRAM | **12**  **12**  **13**  **14**  **15**  **16** |
| **8.** | **ALGORITHM AND DESCRIPTION**  8.1 LINEAR REGRESSION | **17**  **17** |
| **9.** | **DEVELOPMENT TOOLS**  9.1 PYTHON  9.2 FEATURES OF PYTHON  9.3 PYTHON FRAMEWORKS  9.4 OBJECTIVES OF PYTHON  9.5 PACKAGES OF PYTHON  9.6 FLASK  9.7 CONCLUSION | **19**  **19**  **20**  **23**  **24**  **26**  **28**  **30** |
| **10.** | **CONCLUSION AND FUTURE WORK**  10.1 CONCLUSION  10.2 FUTURE WORK | **31**  **31**  **31** |
|  | **APPENDIX**  **SAMPLE CODING** | **32**  **32** |
|  | **SCREEN SHOTS**  **REFERENCES**  **v** | **50**  **52** |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **NAME** | **PAGE NO** |
| **1.** | Overview of System Architecture | **8** |
| **2.** | Use Case Diagram | **12** |
| **3.** | Sequence Diagram | **13** |
| **4.** | Class Diagram | **14** |
| **5.** | Deployment Diagram | **15** |
| **6.** | Activity Diagram  **vi** | **16** |

**LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| **SVM** | : Support – Vector Machine |
| **LSSVM** | : Least - Square Support – Vector Machines |
| **PLS** | : Partial Least Squares |
| **ML** | : Machine Learning |
| **VS CODE** | : Visual Studio Code |
| **IDLE** | : Interactive Development Environment |
| **REPL** | : Read-Evaluate-Print-Loop |
| **CNTK** | : Microsoft Cognitive Toolkit |
| **OOP** | : Object Oriented Programming |
| **GUI** | : Graphical User Interface    **vii** |

**INTRODUCTION**

**CHAPTER 1**

**INTRODUCTION**

**1.1 OBJECTIVES**

The performance will be measured upon predicting house prices since the prediction in many regression algorithms relies not only on a specific feature but on an unknown number of attributes that result in the value to be predicted. House prices depend on an individual house specification. Houses have a variant number of features that may not have the same cost due to its location.

The data used in the experiment will be handled by using a combination of pre-processing methods to improve the prediction accuracy. In addition, some factors will be added to the local dataset in order to study the relationship between these factors and the sale price.

**1.2 OVERVIEW OF PROJECT**

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.

**1**

**LITERATURE SURVEY**

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 INTRODUCTION**

**House Price Prediction Using Machine Learning**

**Authors:** ANAND G. RAWOOL, DATTATRAY V. ROGYE, SAINATH G. RANE, DR. VINAYK A. BHARAD

Machine Learning plays a virtual role from past years in normal speech command, product recommendation as well as in medical field also. Instead of this it provides better customer services and safer automobile system. This all of things shows that ML is trending technology in almost all fields so we are trying to coined up ML in our project. But main disadvantage of current system is to calculate a price of house without necessary prediction about future market trends and result is price increase. So, the main aim of our project is to predict accurate price of house without any loss. There are many factors that have to be taken into consideration for predicting house price and try to predict efficient house pricing for customers with respect to their budget as well as also according to their priorities. So, we are creating a housing cost prediction model. By using Machine learning algorithms like Linear Regression, Decision Tree Regression, K-Means Regression and Random Forest Regression. This model will help people to put resources into a bequest without moving towards a broker. The result of this research provide that the Random Forest Regression gives maximum accuracy.Indexed Terms- Random forest regression, machine learning.

**2**

**House price prediction using Noural Networks**

**Authors:** Reza Gharoie Ahangar, Mahmood Yahyazadehfar and Hassan Pournaghshband

A Research was done in 2010 by Reza Gharoie Ahangar, Mahmood Yahyazadehfar and Hassan Pournaghshband . The authors estimated the stock price of activated companies in Tehran 13 (Iran) stock exchange by using Linear Regression and Artificial Neural Network algorithms. The authors considered ten macroeconomic variables and 30 financial variables. Then, they obtained seven final variables, including three macroeconomic variables and four financial variables, to estimate the stock price using Independent Components Analysis (ICA). They showed that the value of estimation error square mean, the absolute mean of error percentage and R2 coefficient will be decreased significantly after training the model with ANN. A study was conducted in 2015 by Nils Landberg . Nils analysed the price development on the Swedish housing market and the influences of qualitative variables on Swedish house prices. The study shows that the increase in population and qualitative variables have a positive effect on house prices. The interest rate, the average income level, GDP, and the fokus 8 In contrast, the rise in interest rates has a significant negative influence on house prices. Besides, it showed unemployment rate effects negatively on house prices, but the sale price and unemployment rate are not directly correlated with each other.

**3**

**Literature Review on Real Estate Value Prediction Using Machine Learning**

**Authors:** Akshay Babu et al, International Journal of Computer Science and Mobile Applications, Akshay Babu, Dr. Anjana S Chandran

The real estate market is one of the most competitive in terms of pricing and same tends to vary significantly based on numerous factors; forecasting property price is an important module in decision making for both the buyers and investors in supporting budget allocation, finding property finding stratagems and determining suitable policies hence it becomes one of the prime fields to apply the concepts of machine learning to optimize and predict the prices with high accuracy. Therefore, in this paper, we present various important features to use while predicting housing prices with good accuracy. We can use regression models, using various features to have lower Residual Sum of Squares error. While using features in a regression model some feature engineering is required for better prediction. Often a set of features multiple regressions or polynomial regression (applying a various set of powers in the features) is used for making better model fit. For these models are expected to be susceptible towards over fitting ridge regression is used to reduce it. So, it directs to the best application of regression models in addition to other techniques to optimize the result. Keywords: Uses, Advantages, Literature survey.

**4**

**House Price Prediction using a Machine Learning Model**

**Authors:** Hamizah Zulkifley, Shuzlina Abdul Rahman, Nor Hasbiah Ubaidullah, Ismail Ibrahim

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

**5**

**SYSTEM ANALYSIS**

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 EXISTING SYSTEMS**

* The house price prediction project has the existing concepts with machine learning concepts.
* Algorithm – SVM, LSSVM, PLS, Random Forest.
* Deployed using Web Development Concepts.
* Classical Machine Learning Software.

**ALGORITHM:**

* **SVM -** Support Vector Machine (SVM) is **a supervised machine learning algorithm used for both classification and regression**.
* **LSSVM – Least - square support – vector Machines** for statistics and in statistics and in statistical modelling, are least- modelling are least-square version of SVM.
* **PLS -** Partial least squares regression (PLS regression) is **a statistical method that bears some relation to principal components regression.**
* **Decision Tree –** It is a Supervised learning technique that can be used for both classification and Regression problems.

**3.1.1 DISADVANTAGE**

* System requirements for the existing is very high.
* Decision tree has the accuracy of 78%

**6**

**3.2 PROPOSED SYSTEM**

* The house price prediction project is now implemented in the machine learning and the Flask is deployed for the backend process.
* Algorithm – Linear Regression
* Deployment using Flask.
* Machine learning implemented in Google Colab (Notebook).

**3.2.1 ADVANTAGE**

* System requirements for the proposed is very low.
* Linear Regression is used to improve the accuracy of the model.

**3.2.2 PROBLEM DEFINITION**

* The given algorithm defines the higher accuracy for the dataset provided.
* As the system requirements for the given existing system is higher and complex, the current system explains easier method to derive.

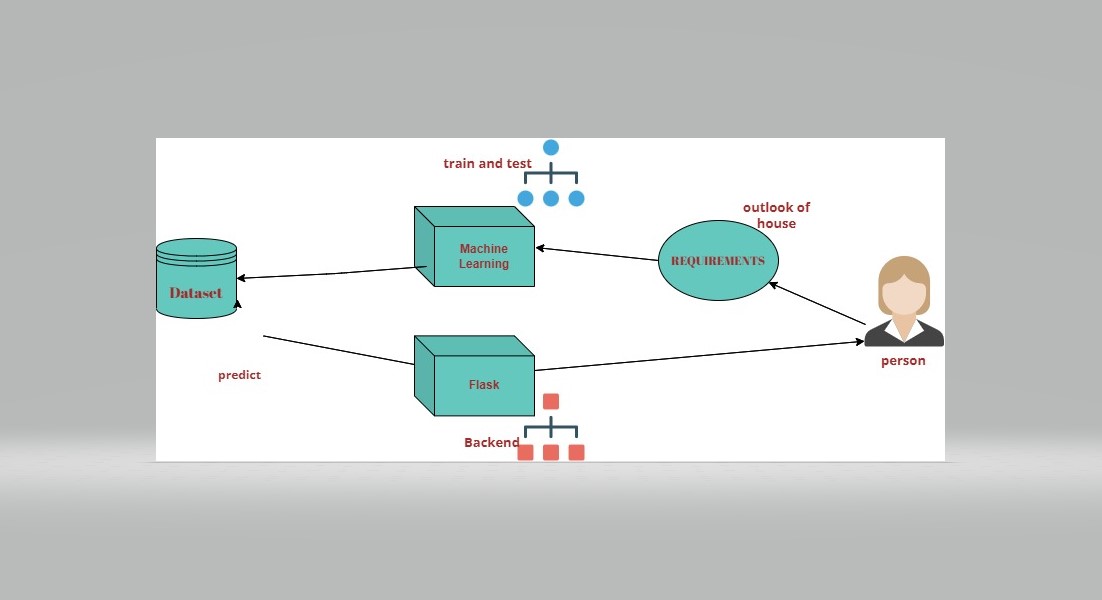
**7**

**SYSTEM ARCHITECTURE**

**CHAPTER 4**

**SYSTEM ARCHITECTURE**

**4.1 OVERVIEW OF SYSTEM ARCHITECTURE**

**4.1 Overview of the system architecture**

**8**

**SYSTEM SPECIFICATIONS**

**CHAPTER 5**

**SYSTEM SPECIFICATIONS**

**5.1 HARDWARE AND SOFTWARE REQUIREMENTS**

**HARDWARE REQUIREMENTS**

|  |  |
| --- | --- |
| PROCESSOR | **:** AMD PRO A4-4350B R4, COMPUTE CORES  2C+3G 2.50GHz |
| RAM | **:** 4GB |
| OS | : 64-bit Operating System Windows 10 |

**SOFTWARE REQUIREMENTS**

|  |  |
| --- | --- |
| Machine Learning | : Google Colab |
| Front End | : Visual Studio Code |
| Back End | : Jypter  **9** |

**MODULES**

**CHAPTER 6**

**MODULES**

**6.1 USER MODULE**

**Web page for the submission of the requirements of house:**

* As mentioned, there are numerous amounts of field that is responsible for the price of the house.
* Those fields are identified and the requirements for the fields are given by the user.
* The field entered by the user is responsible for the prediction of the price of the house.

**Displaying the price of the house as per the requirements:**

* The price of the house is said that is entered through user requirements model is said to predict the price of the house.
* The price of the house is predicted and the rate is displayed over the screen as per the requirements.

**10**

**6.2 DATA MODULE**

Machine Learning backend as the phase of prediction of the house price

* As the user gives the requirements for the house price, the backend that is attached with the Machine Learning code is being in the form of executing the price.
* The user after giving the certain requirements hits the SUBMIT button.
* After the user hits the submit button the data entries are taken into the consideration.
* After the data is included in the account the machine predicts the price of the house as it is already trained and tested with the given dataset.

**11**

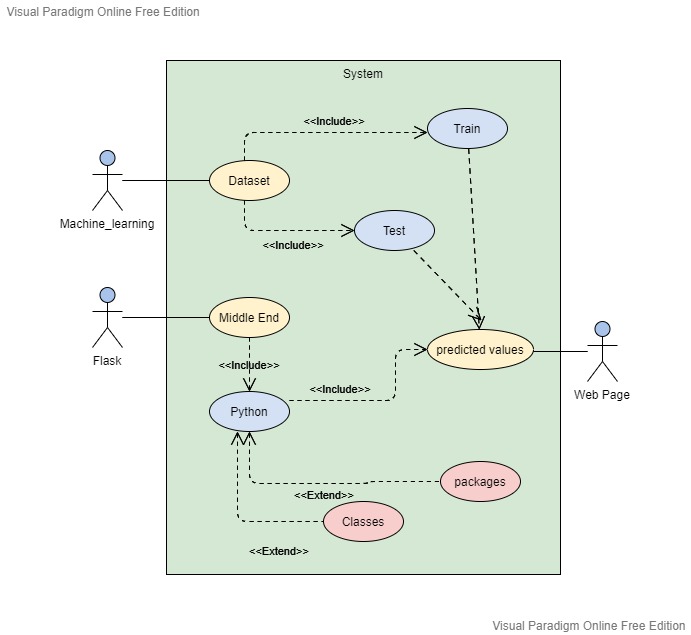
**UML DIAGRAM**

**CHAPTER 7**

**UML DIAGRAM**

**7.1 USE CASE DIAGRAM**

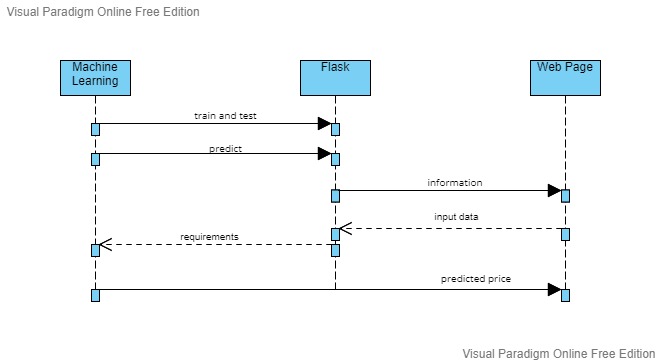
Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.



**4.2 Use case Diagram**

**127.2 SEQUENCE DIAGRAM**

A sequence diagram or system sequence diagram **shows process interactions arranged in time sequence in the field of software engineering**. It depicts the processes involved and the sequence of messages exchanged between the processes needed to carry out the functionality.

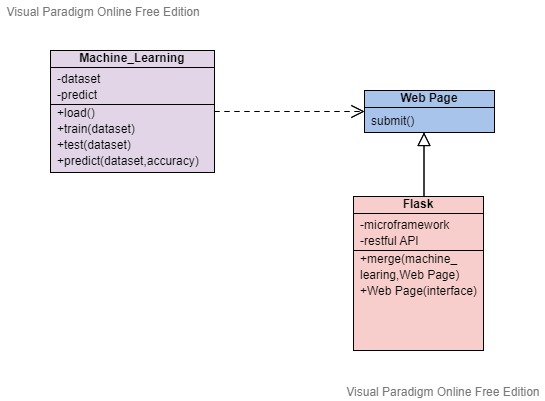


**4.3 Sequence Diagram**

**13**

**7.3 CLASS DIAGRAM**

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

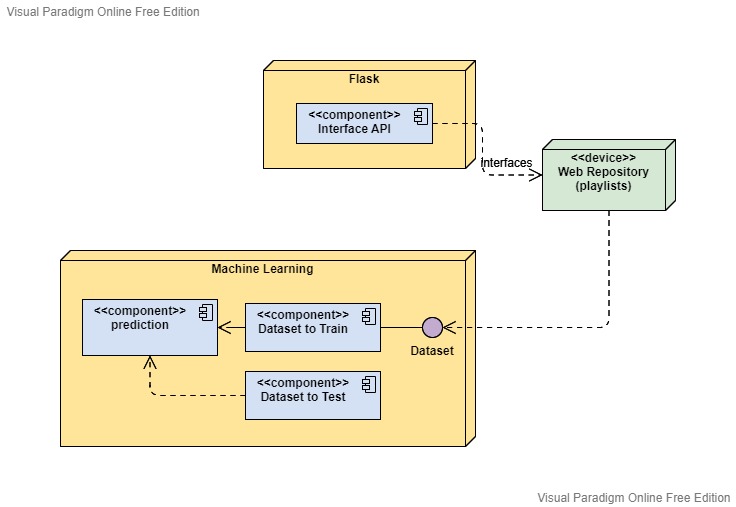


**4.4 Class Diagram**

**14**

**7.4 DEPLOYMENT DIAGRAM**

The term Deployment itself describes the purpose of the diagram. Deployment diagrams are used for describing the hardware components, where software components are deployed. Component diagrams and deployment diagrams are closely related.



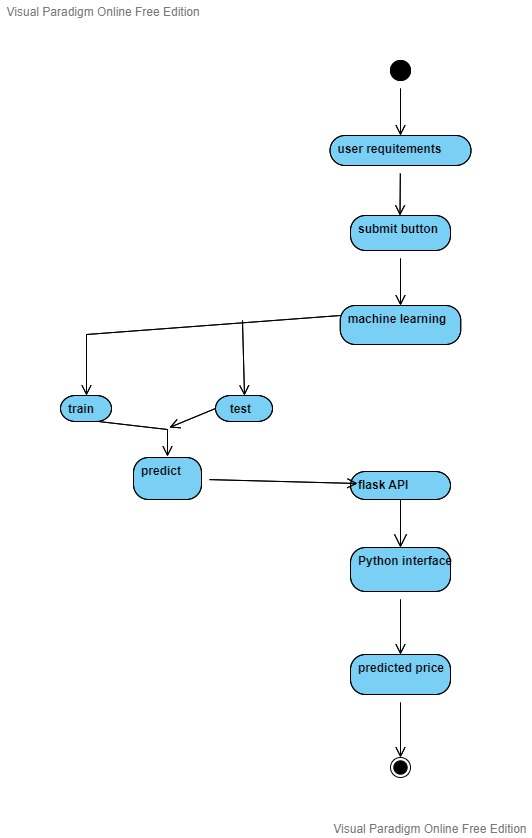
**4.5 Deployment Diagram**

**15**

**7.5 ACTIVITY DIAGRAM**

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.



**4.6 Activity Diagram**

**16**

**ALGORITHM AND DESCRIPTION**

**CHAPTER 8**

**ALGORITHM AND DESCRIPTION**

**8.1 LINEAR REGRESSION**

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as **sales, salary, age, product price,** etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

**While training the model we are given :**

x: input training data (univariate – one input variable(parameter))

y: labels to data (supervised learning)

When training the model – it fits the best line to predict the value of y for a given value of x. The model gets the best regression fit line by finding the best θ1 and θ2 values.

θ1: intercept

θ2: coefficient of x

We find the best θ1 and θ2 values, we get the best fit line. We are finally using our model for prediction, it will predict the value of y for the input value of x.

**17**

**How to update θ1 and θ2 values to get the best fit line ?**

Cost Function (J):

By achieving the best-fit regression line, the model aims to predict y value such that the error difference between predicted value and true value is minimum. So, it is very important to update the θ1 and θ2 values, to reach the best value that minimize the error between predicted y value (pred) and true y value (y).

The linear regression model provides a sloped straight line representing the relationship between the variables.

**Algorithm:**

y= Dependent Variable.  
  
x= Independent Variable.  
  
a0= intercept of the line.  
  
a1 = Linear regression coefficient.

**18**

**DEVELOPMENT TOOLS**

**CHAPTER 9**

**DEVELOPMENT TOOLS**

**9.1 PYTHON**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

**19**

**9.2 FEATURES OF PYTHON**

1. **EASY LANGUAGE**

* Python is an easy language. It is easy to read, write, learn and understand.
* Python has a smooth learning curve. It is easy to learn.
* Python has a simple syntax and Python code is easy to understand.
* Since it’s easy to understand, you can easily read and understand someone else’s code.
* Python is also easy to write because of its simple syntax.

**2. READABLE**

* The Python language is designed to make developers life easy. Reading a Python code is like reading an English sentence. This is one of the key reason that makes Python best for beginners.
* Python uses indentation instead of curly braces, unlike other programming languages. This makes the code look clean and easier to understand.

**3. INTERPRETED LANGUAGE:**

* Python is an interpreted language. It comes with the IDLE (Interactive Development Environment). This is an interpreter and follows the REPL structure (Read-Evaluate-Print-Loop). It executes and displays the output of one line at a time.

**20**

**4. DYNAMICALLY – TYPED LANGUAGE:**

* Python is not statically-typed like Java. You don’t need to declare data type while defining a variable. The interpreter determines this at runtime based on the types of the parts of the expression. This is easy for programmers but can create runtime errors.

**5. OBJECT – ORIENTED:**

* Python is object-oriented but supports both functional and object-oriented programming. Everything in Python is an object.
* It has the OOP (Object-oriented programming) concepts like inheritance and polymorphism.

**6. POPULAR AND LARGE COMMUNITY SUPPORT:**

* Python has one of the largest communities on StackOverflow and Meetup. If you need help, the community will answer your questions.

**7. OPEN – SOURCE:**

* Python is open-source and the community is always contributing to it to improve it. It is free and its source code is freely available to the public. You can download Python from the official Python Website.

**21**

**8. LARGE STANDARD LIBRARY:**

* The standard library is large and has many packages and modules with common and important functionality. If you need something that is available in this standard library, you don’t need to write it from scratch.
* Because of this, you can focus on more important thing.

**9. PLATFORM INDEPENDENT:**

* Python is platform-independent. If you write a program, it will run on different platforms like Windows, Mac and Linux. You don’t need to write them separately for each platform.

**10. EXTENSABLE AND EMBEDABLE:**

* Python is extensible. You can use code from other languages like C++ in your Python code.

**11. GUI SUPPORT**

* Python features a huge number of GUI frameworks available for it to variety of other cross-platform solutions. It binds to platform-specific technologies.

**12. HIGH – LEVEL LANGUAGE:**

* Python is a high-level language and C++ is mid-level. It is easy to understand and closer to the user. You don’t need to remember system architecture or manage the memory.

**22**

**9.3 PYTHON FRAMEWORKS**

A framework is a collection of modules or packages which helps in writing web applications. While working on frameworks in python we don’t have to worry about the low level details such as protocols, sockets or thread management.

Frameworks automate the common implementation of common solutions which gives the flexibility to the users to focus on the application logic instead of the basic routine processes.

Frameworks make the life of web developers easier by giving them a structure for app development. They provide common patterns in a web application that are fast, reliable and easily maintainable.

Depending upon the sort of functionalities and key features they provide to the user, these are top 5 frameworks in python, both micro-frameworks and full-stack frameworks.

* Django
* Web2Py
* Flask
* Bottle
* CherryPy

**9.4 OBJECTIVES OF PYTHON**

* Master the fundamentals of writing Python scripts
* Discover how to work with lists and sequence data
* Write Python functions to facilitate code reuse
* Use Python to read and write files

**23**

* Make their code robust by handling errors and exceptions properly
* Work with the Python standard library
* Explore Python's object-oriented features
* Search text using regular expressions

**9.5 PACKAGES OF PYTHON**

Machine Learning tutorial provides basic and advanced concepts of machine learning. Our machine learning tutorial is designed for students and working professionals.

Machine learning is a growing technology which enables computers to learn automatically from past data. Machine learning uses various algorithms for **building mathematical models and making predictions using historical data or information**. Currently, it is being used for various tasks such as **image recognition**, **speech recognition**, **email filtering**, **Facebook auto-tagging**, **recommender system**, and many more.

This machine learning tutorial gives you an introduction to machine learning along with the wide range of machine learning techniques such as **Supervised**, **Unsupervised**, and **Reinforcement** learning.

**24**

You will learn about regression and classification models, clustering methods, hidden Markov models, and various sequential models.

## **What is Machine Learning?**

In the real world, we are surrounded by humans who can learn everything from their experiences with their learning capability, and we have computers or machines which work on our instructions.

But can a machine also learn from experiences or past data like a human does? So here comes the role of **Machine Learning**.

Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed.

## **How does Machine Learning work?**

A Machine Learning system **learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it**. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output.

**25**

## **Features of Machine Learning:**

* Machine learning uses data to detect various patterns in a given dataset.
* It can learn from past data and improve automatically.
* It is a data-driven technology.
* Machine learning is much similar to data mining as it also deals with the huge amount of the data.

## **Need for Machine Learning**

The need for machine learning is increasing day by day. The reason behind the need for machine learning is that it is capable of doing tasks that are too complex for a person to implement directly. As a human, we have some limitations as we cannot access the huge amount of data manually, so for this, we need some computer systems and here comes the machine learning to make things easy for us.

We can train machine learning algorithms by providing them the huge amount of data and let them explore the data, construct the models, and predict the required output automatically. The performance of the machine learning algorithm depends on the amount of data, and it can be determined by the cost function. With the help of machine learning, we can save both time and money.

**Regression:**

Regression is the process of finding a model or function for distinguishing the data into continuous real values instead of using classes or discrete values. It can also identify the distribution movement depending on the historical data.

**26**

**Use of Python Packages:**

An ML Package is a folder with all the code and metadata needed to train and serve a machine learning model. An ML Package can have multiple versions and is in some way analogous to a package in Orchestrator. Each version can have an associated change log. It is recommended that users acting as Data Scientists handle packages. More details about recommended user personas here.

In order to be used within your workflows in Studio, you first have to deploy them as skills in your tenant.

The ML Packages page, accessible from the ML Packages menu after selecting a project, enables you to view all the available versions of a package, along with their statuses, change logs, and pipelines. Here you can upload new packages or new versions for existing ones, delete undeployed packages, view available information about them, or manage their pipelines.

.A python package is a collection of modules. Modules that are related to each other are mainly put in the same package.

The major packages used here in machine learning are:

* **Matplotlib** is an interactive, cross-platform library for two-dimensional plotting. It can produce high-quality graphs, charts and plots in several hardcopy formats.
* **Pandas** is a Python library for providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

**27**

* The Python library, **Scikit-Learn**, is built on top of the matplotlib, NumPy, and SciPy libraries. This Python ML library has several tools for data analysis and data mining tasks.
* **Seaborn** is a library for making statistical graphs in Python. It is built on top of matplotlib and also integrated with pandas data structures.
* **NumPy** adds multi-dimensional array and matrix processing to Python, as well as a large collection of high-level mathematical functions. It is commonly used for scientific computing and hence, one of the most used Python Packages for machine learning.
* **Keras** is a very popular ML for Python, providing a high-level neural network API capable of running on top of TensorFlow, CNTK, or Theano.
* **TensorFlow** is an open-source library for high-performance numerical computation using data flow graphs.

**9.6 FLASK**

## **What is Flask?**

Flask is a web application framework written in Python. It is developed by **Armin Ronacher**, who leads an international group of Python enthusiasts named Pocco. Flask is based on the Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

It has a number of out of the box features listed below:

* Integrated support for unit testing
* RESTful request dispatching
* Jinja2 templating

**28**

* Secure cookies support
* Unicode-based
* WSGI compliance
* Ability to plug any ORM
* HTTP request handling

## **What is Web Framework?**

Web Application Framework or simply Web Framework represents a collection of libraries and modules that enables a web application developer to write applications without having to bother about low-level details such as protocols, thread management etc.

## **WSGI**

Web Server Gateway Interface (WSGI) has been adopted as a standard for Python web application development. WSGI is a specification for a universal interface between the web server and the web applications.

## **Werkzeug**

It is a WSGI toolkit, which implements requests, response objects, and other utility functions.

This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases.

**29**

## **Jinja2**

Jinja2 is a popular templating engine for Python. A web templating system combines a template with a certain data source to render dynamic web pages.

Flask is often referred to as a micro framework. It aims to keep the core of an application simple yet extensible. Flask does not have built-in abstraction layer for database handling, nor does it have form a validation support. Instead, Flask supports the extensions to add such functionality to the application. Some of the popular Flask extensions are discussed later in the tutorial.

**9.7 CONCLUSION**

Hence Python is used as it is easy to use and it flexible to debug the errors and it is easier to understand. As python is platform independent and Object –Oriented Programming, so it is far easier to work in Python. Flask is implemented that it is very easier to manage the understand the need of the developers and it is a RESTful API that is used as the interface to implement as it is faster to debug.

**30**

**CONCLUSION &**

**FUTURE WORK**

**CHAPTER 10**

**CONCLUSION AND FUTURE WORK**

**10.1 CONCLUSION**

The paper entitled “House Price Prediction Using Machine Learning” has presented to predict house price based on various features on given data. From our analysis we set value of Linear regression with the prediction value of percentage 84%. In this model we have to add additional features like tax, air quality so it become different from other prediction system. It helps people to buy house in budget and reduce loss of money. This improves the customer to get the higher standard for finding the house at their range of budget in their desired location and requirements. So this makes the customers to get into the various forms and find the uses in the finding the house prices.

**10.2 FUTURE WORK**

The house price can be predicted using the various other algorithm to get the higher value prediction of percentage. As we are working to get in the Random Forest algorithm to get the higher prediction of the value. If the dataset id of higher value we can implement in Django either implementing in Flask.

**31**

**APPENDIX**

**SAMPLE CODE**

**house\_price.ipynb**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import warnings

warnings.filterwarnings('ignore')

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelEncoder

from sklearn import metrics

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

from sklearn.svm import SVR

from sklearn.tree import DecisionTreeRegressor

from sklearn.ensemble import RandomForestRegressor

**32**

from sklearn import neighbors

from math import sqrt

%matplotlib inline

df1 = pd.read\_csv('data.csv')

df1

df1.head()

df1.describe()

df1.shape

df1.ndim

df1 = df1.dropna(thresh=0.70\*len(df1),axis=1)

X = X.fillna(df1.mean())

y = y.fillna(df1.mean())

corr = df1.corr()

print (corr['price'].sort\_values(ascending=False)[:5], '\n')

print (corr['price'].sort\_values(ascending=False)[-5:])

df1.isnull().sum()

df1.isnull().mean()

df1

**33**

def missing (df):

missing\_number = df1.isnull().sum().sort\_values(ascending=False)

missing\_percent = ((df1.isnull().sum()/df1.isnull().count())\*100).sort\_values(ascending=False)

missing\_values = pd.concat([missing\_number, missing\_percent], axis=1, keys=['Missing\_Number', 'Missing\_Percent'])

return missing\_values

total = df1.isnull().sum().sort\_values(ascending = False)

percent = (df1.isnull().sum()/df1.isnull().count()).sort\_values(ascending = False)

missing\_data = pd.concat([total,percent], axis = 1, keys = ['Total', 'Percent'])

missing\_data.head(18)

missing(df1)

for col in df1.columns:

if df1[col].isnull().mean()\*100>40:

df1.drop(col,axis=1,inplace=True)

df1

df1.columns

**34**

df1.dtypes.value\_counts()

f = lambda x: x.median() if np.issubdtype(x.dtype, np.number) else x.mode().iloc[0]

df1 = df1.fillna(df1.groupby('date').transform(f))

df1

for col in df1.columns:

if df1[col].dtypes != object:

q1 = df1[col].quantile(0.25)

q2 = df1[col].quantile(0.50)

q3 = df1[col].quantile(0.75)

IQR = q3 - q1

llp = q1-1.5\*IQR

ulp = q3+1.5\*IQR

print('column name',col)

print('q1',q1)

print('q2',q2)

print('q3',q3)

**35**

print('IQR',IQR)

print('llp',llp)

print('ulp',ulp)

print('mean:',df1[col].mean())

print('median:',df1[col].median())

print('mode',df1[col].mode()[0])

print('skewness:',df1[col].skew())

print('kurtosis:',df1[col].kurtosis())

print('std',df1[col].std())

print('max',df1[col].max())

print('min',df1[col].min())

print('null\_value count:',df1[col].isnull().sum())

print('\n')

df1['floors'].unique()

df1['bedrooms'].unique()

Q1 = df1.quantile(0.25)

Q3 = df1.quantile(0.75)

IQR = Q3 – Q1

print('outliers count of each columns')

**36**

((df1 < (Q1 - 1.5 \* IQR)) | (df1 > (Q3 + 1.5 \* IQR))).sum()

df1.dtypes

le=LabelEncoder()

for col in df1.columns:

if df1[col].dtypes == object:

df1[col]= le.fit\_transform(df1[col])

df1.ndim

y = np.log(df1.price)

X = df1.drop(['price'], axis=1)

print(np.any(np.isnan(X)))

data = np.nan\_to\_num(X)

print(data)

df1.ndim

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.25,random\_state=42)

X=df1.drop('price',axis=1)

y=df1['price']

**37**

df1

a1=df1['date']

a2=df1['price']

a3=df1['bedrooms']

a4=df1['bathrooms']

a5=df1['sqft\_living']

a6=df1['sqft\_lot']

a7=df1['floors']

a8=df1['waterfront']

a9=df1['view']

a10=df1['condition']

a11=df1['sqft\_above']

a12=df1['sqft\_basement']

a13=df1['yr\_built']

a14=df1['yr\_renovated']

a15=df1['street']

a16=df1['city']

**38**

a17=df1['statezip']

a18=df1['country']

df1 = np.stack((a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,a11,a12,a13,a14,a15,a16,a17,a18),axis=1

df1 = np.hstack((a1, a2))

print(df1.shape)

from sklearn.impute import SimpleImputer

imp\_mean = SimpleImputer(missing\_values=np.nan, strategy='constant', fill\_value=0)

imputer = imp\_mean.fit([df1])

df1 = imputer.transform([df1])

print(df1)

def train\_models(X\_train, y\_train)

#use Decision Tree

tree = DecisionTreeRegressor(max\_features=75,max\_depth=4, random\_state = 0)

tree.fit(X\_train, y\_train)

**39**

y\_pred\_tree = tree.predict(X\_test)

#use the RandomForestRegressor

rf = RandomForestRegressor(n\_estimators = 100,max\_features =75, random\_state = 0)

rf.fit(X\_train, y\_train)

y\_pred\_rf= rf.predict(X\_test)

# use the support vector regressor

#from sklearn.svm import SVR

svr= SVR(kernel = 'rbf')

svr.fit(X\_train, y\_train)

y\_pred\_svr = svr.predict(X\_test)

#from sklearn.svm import SVR

svr\_l= SVR(kernel = 'linear')

svr\_l.fit(X\_train, y\_train)

y\_pred\_svr\_linear = svr\_l.predict(X\_test)

# use the knn regressor

knn = neighbors.KNeighborsRegressor()

knn.fit(X\_train, y\_train)

**40**

y\_pred\_knn = knn.predict(X\_test)

# metrics of decision tree regressor

meanAbErr\_tree= metrics.mean\_absolute\_error(y\_test, y\_pred\_tree)

meanSqErr\_tree= metrics.mean\_squared\_error(y\_test, y\_pred\_tree)

rootMeanSqErr\_tree= np.sqrt(metrics.mean\_squared\_error(y\_test, y\_pred\_tree))

# metrics of random forest regressor

meanAbErr\_rf= metrics.mean\_absolute\_error(y\_test, y\_pred\_rf)

meanSqErr\_rf= metrics.mean\_squared\_error(y\_test, y\_pred\_rf)

rootMeanSqErr\_rf= np.sqrt(metrics.mean\_squared\_error(y\_test, y\_pred\_rf))

# metrics of knn regressor

meanAbErr\_knn = metrics.mean\_absolute\_error(y\_test, y\_pred\_knn)

meanSqErr\_knn = metrics.mean\_squared\_error(y\_test, y\_pred\_knn)

rootMeanSqErr\_knn= np.sqrt(metrics.mean\_squared\_error(y\_test, y\_pred\_knn))

# metrics of svr regressor

meanAbErr\_svr = metrics.mean\_absolute\_error(y\_test, y\_pred\_svr\_linear)

**41**

meanSqErr\_svr = metrics.mean\_squared\_error(y\_test, y\_pred\_svr\_linear)

rootMeanSqErr\_svr= np.sqrt(metrics.mean\_squared\_error(y\_test,

y\_pred\_svr\_linear))

#print the tranning accurancy of each model

print('[1]Decision Tree Training Accurancy: ', r2\_score(y\_test,y\_pred\_tree))

print('Mean Absolute Error:', meanAbErr\_tree)

print('Mean Square Error:', meanSqErr\_tree)

print('Root Mean Square Error:', rootMeanSqErr\_tree)

print('\t')

print('[2]RandomForestRegressor Training Accurancy: ',r2\_score(y\_test,y\_pred\_rf))

print('Mean Absolute Error:', meanAbErr\_rf)

print('Mean Square Error:', meanSqErr\_rf)

print('Root Mean Square Error:', rootMeanSqErr\_rf)

print('\t')

print('[3]SupportvectorRegression Accuracy(rbf): ', r2\_score(y\_test,y\_pred\_svr))

print('\t')

**42**

print('[4]SupportvectorRegression Accuracy(linear): ',

r2\_score(y\_test,y\_pred\_svr\_linear))

print('Mean Absolute Error:', meanAbErr\_svr)

print('Mean Square Error:', meanSqErr\_svr)

print('Root Mean Square Error:', rootMeanSqErr\_svr)

print('\t')

print('[5]knn Training Accurancy: ', r2\_score(y\_test,y\_pred\_knn))

print('Mean Absolute Error:', meanAbErr\_knn)

print('Mean Square Error:', meanSqErr\_knn)

print('Root Mean Square Error:', rootMeanSqErr\_knn)

print('\t')

train\_models(X\_train, y\_train)

from sklearn.impute import SimpleImputer

imp = SimpleImputer(missing\_values=np.nan, strategy='mean')

imp.fit([[1, 2], [np.nan, 3], [7, 6]])

X = [[np.nan, 2], [6, np.nan], [7, 6]]

print(imp.transform(X))

from sklearn.linear\_model import LinearRegression

**43**

mlr = LinearRegression()

mlr.fit(X\_train, y\_train)

y\_pred\_mlr= mlr.predict(X\_test)

y\_pred\_mlr

r2\_mlr =r2\_score(y\_test,y\_pred\_mlr)

print('r2\_score:',r2\_mlr\*100)

import pickle

with open('model.pkl','wb') as files:

pickle.dump(mlr,files)

**app.py:**

from array import array

from distutils.log import debug

import pickle

import numpy as np

from flask import Flask, render\_template, request

app = Flask(\_name\_, template\_folder='template')

model = pickle.load(open('model.pkl','rb'))

def city():

try:

float(request.form.get('city'))

**44**

except:

print('Not a number')

def dateobj():

try:

request.form.get('date')

except:

print('Invalid')

def street():

try:

request.form.get('street')

except:

print('Invalid')

def statezip():

try:

request.form.get('statezip')

except:

print('Invalid')

def country():

try:

request.form.get('country')

except:

print('Invalid')

@app.route('/')

def index():

return render\_template('index.html')

**45**

@app.route('/predict',methods=['GET','POST'])

def predict():

arr=[]

l=[]

l.append(float(request.form.get('bedrooms')))

l.append(float(request.form.get('bathrooms')))

l.append(float(request.form.get('sqft\_living')))

l.append(float(request.form.get('sqft\_lot')))

l.append(request.form.get('floors'))

l.append(dateobj())

l.append(request.form.get('waterfront'))

l.append(request.form.get('view'))

l.append(request.form.get('condition'))

l.append(request.form.get('sqft\_above'))

l.append(request.form.get('sqft\_basement'))

l.append(street())

l.append(city())

l.append(statezip())

l.append(country())

l.append(float(request.form.get('yr\_built')))

l.append(float(request.form.get('yr\_renovated')))

arr = np.array(l)

arr=arr.reshape(1,-1)

prediction = round(model.predict(arr)[0],2)

**46**

print(prediction)

return render\_template('index.html')

if \_name=='main\_':

app.run(port=5000,debug=True)

**index.html:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>House Price Prediction</title>

</head>

<body>

<center>

<table>

<td>Enter the no. of bedrooms:</td>

<td>

<select id="bedroom">

<option value="One">One</option>

<option value="Two">Two</option>

</select>

</td>

</tr>

**47**

<tr>

<td>Enter the floor:</td>

<td>

<select id="floor">

<option value="Zero">Zero</option>

<option value="One">One</option>

<option value="Two">Two</option>

</select>

</td>

</tr>

<tr>

<td>Enter the area:</td>

<td>

<select id="area">

<option value="rural">Rural</option>

<option value="Urban">Urban</option>

</select>

</td>

</tr>

<tr>

<td>Enter the model of house:</td>

<td>

<select id="model">

<option value="New">New</option>

<option value="Old">Old</option>

**48**

</select>

</td>

</tr>

<tr>

<td>Enter the way furnished:</td>

<td>

<select id="furnished">

<option value="Fully Furnished">Fully Furnished</option>

<option value="Partially Furnished">Fully Furnished</option>

</select>

</td>

</tr>

</table>

<input type="submit" value="submit">

</form>

</center>

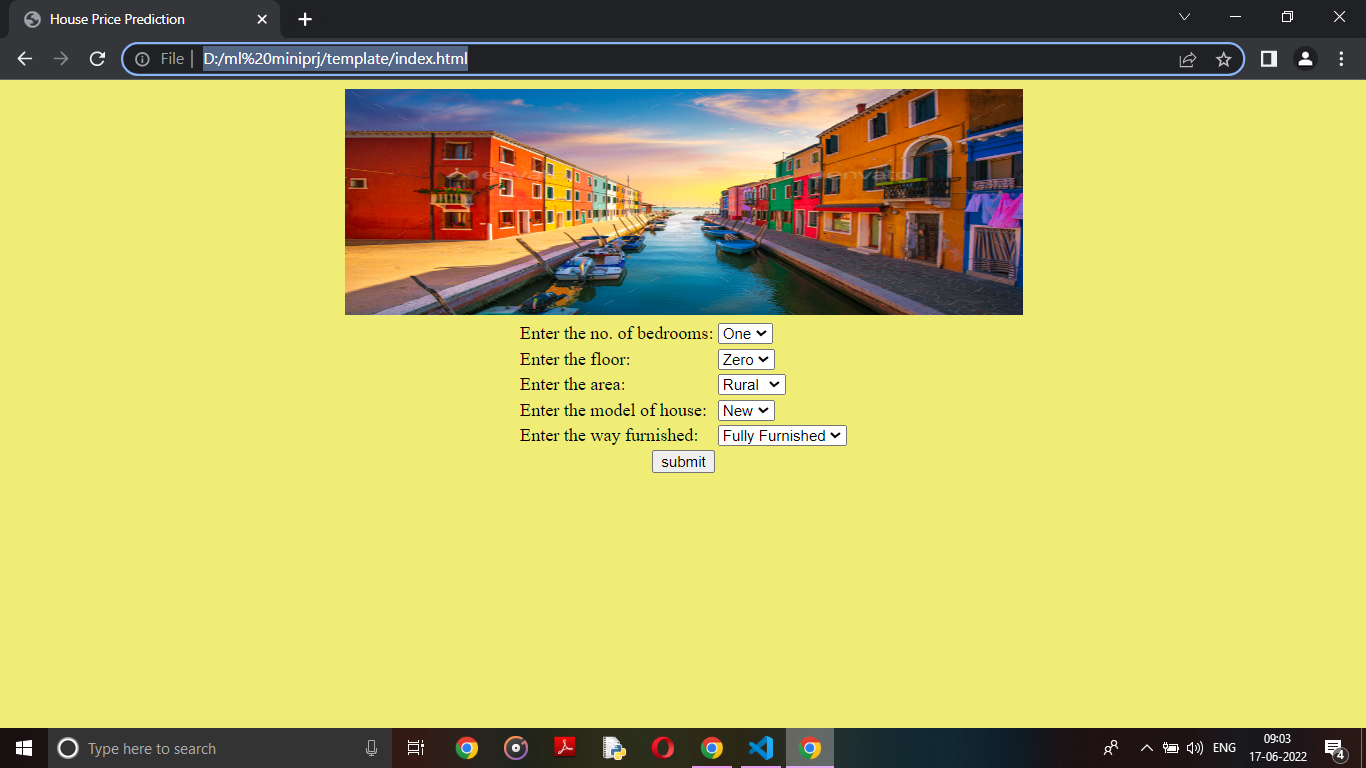
</body>

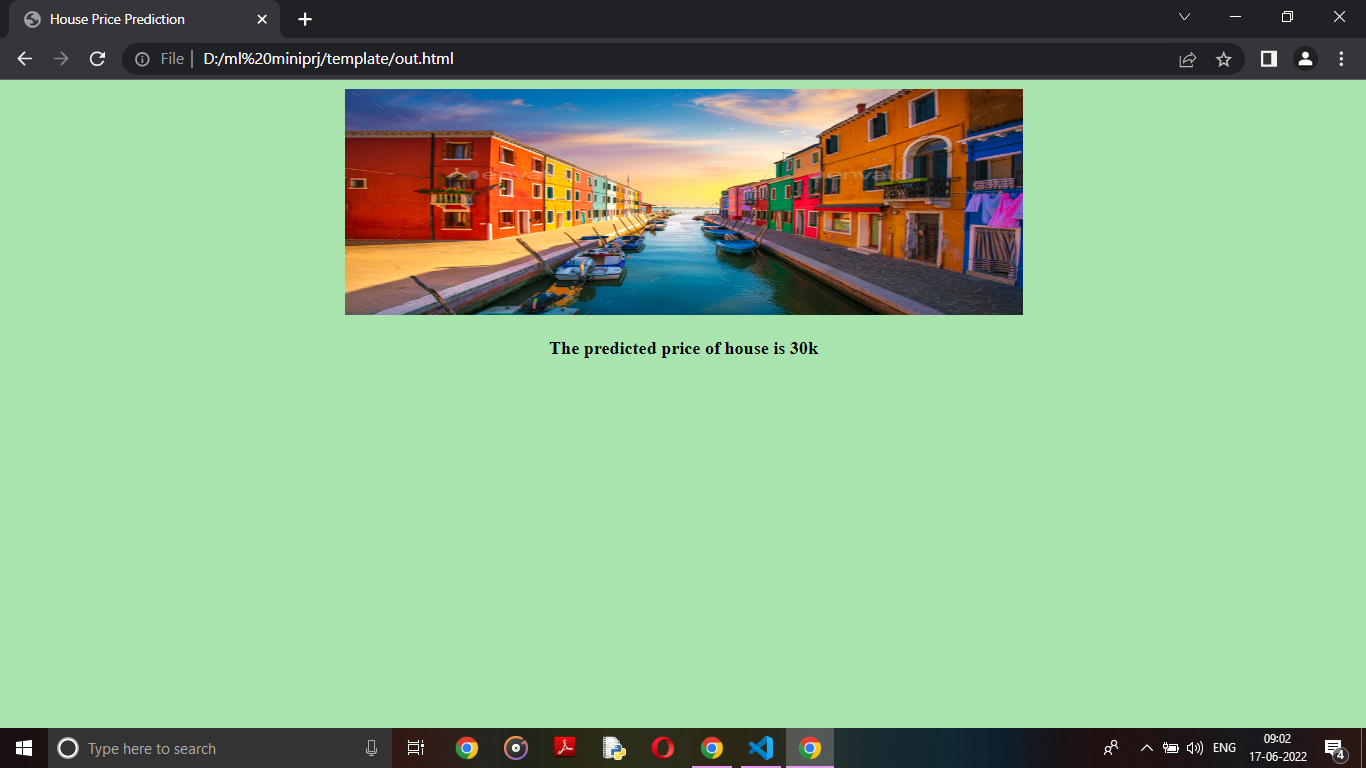
</html>

**49**

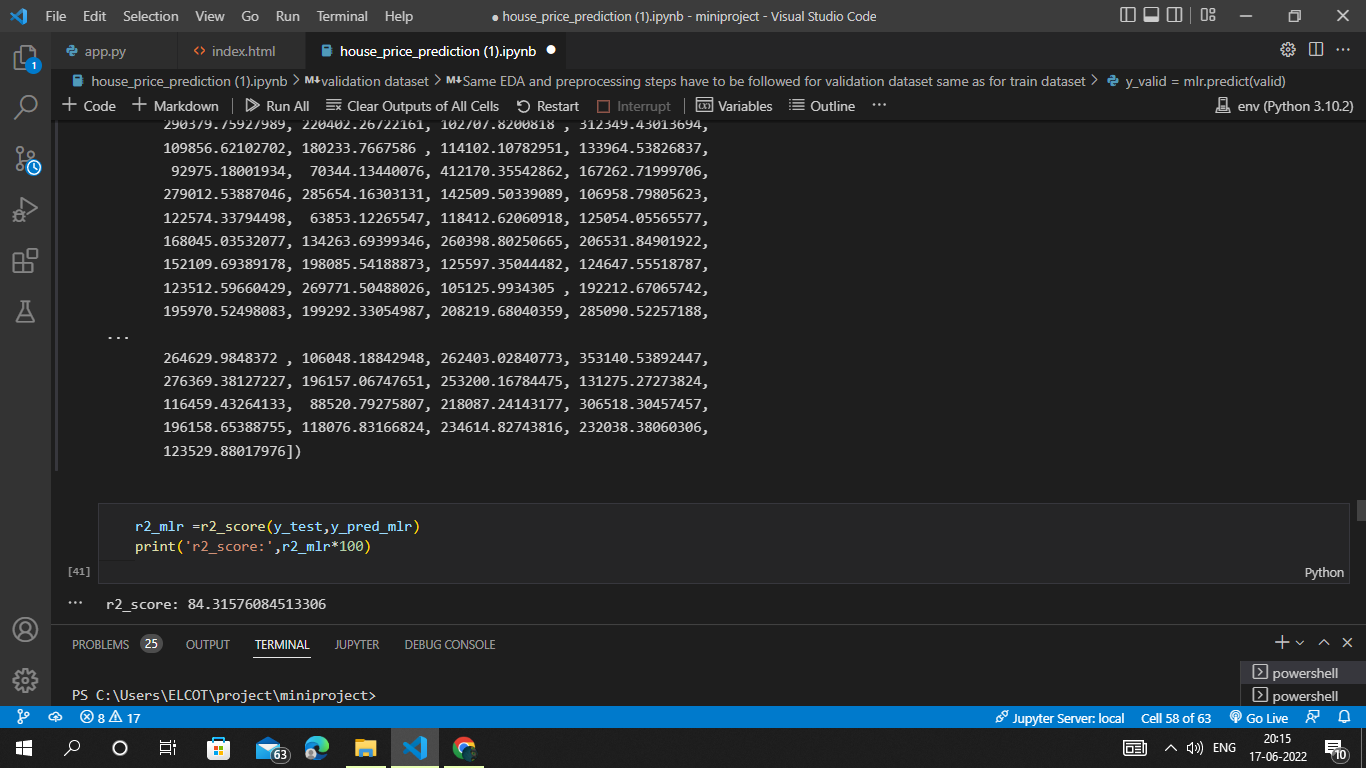
**SCREENSHOTS**

**Screenshots**





**50**



**51**

**REFERENCES**

[1] Ayush Varma, Abhijit Sarma, Sagar Doshi, Rohini Nair - “Housing Price Prediction Using Machine Learning and Neural Networks” 2018, IEEE.

[2] G.Naga Satish, Ch.V.Raghavendran, M.D.Sugnana Rao, Ch.Srinivasulu “House Price Prediction Using Machine Learning”. IJITEE, 2019.

[3] CH. Raga Madhuri, G. Anuradha, M. Vani Pujitha -” House Price Prediction Using Regression Techniques: A Comparative Study” 2019 in (ICSSS),IEEE.

[4] Sifei Lu, Zengxiang Li, Zheng Qin , Xulei Yang , Rick Siow Mong Goh - “A hybrid regression technique for house prices prediction” 2017,IEEE

[5] Bharatiya, Dinesh, et al. “Stock market prediction using linear regression.” Electronics, Communication, and Aerospace Technology (ICECA), 2017 International conference of. Vol. 2. IEEE, 2017.

[6] Vincy Joseph, Anuradha Srinivasaraghavan- “Machine Learning”.

[7] Trevor Hastie, Robert Tibshirani, Jerome Friedman- “The Elements of Statistical Learning”.

[8] Tom M Mitchell- “Machine Learning”

[9] Saleh Hyatt- “Machine Learning Fundamentals”.

**52**