Tribhuvan University

Faculty of Humanities and Social Sciences

Laptop Price Prediction System
Using Random Forest Algorithm

PROJECT REPORT

Submitted to
Department of Computer Application
Patan Multiple Campus
Patan Dhoka, Lalitpur

In partial fulfillment of the requirements for the Bachelors in Computer Application

Submitted by
Ayush Silwal
BCA 6th Semester
59/77
University SN: 6-2-22-11-2020

Under the Supervision of
Jagdish Bhatta

TRIBHUVAN UNIVERSITY
Patan Multiple Campus
Patan Dhoka, Lalitpur, Nepal

Supervisor's Recommendation

I hereby recommend that this project prepared under my supervision by Ayush Silwal entitled "**Laptop Price Prediction System"** in partial fulfillment of the requirements for the degree of Bachelor of Computer Application is recommended for the final evaluation.

____

Mr. Jagdish Bhatta
Project Supervisor
Department of Computer Application
Patan Multiple Campus
Patan Dhoka, Lalitpur

TRIBHUVAN UNIVERSITY

Patan Multiple Campus

Patan Dhoka, Lalitpur, Nepal

Approval of Project Proposal

The Project entitled "**Laptop Price Prediction System"** Proposed by **Ayush Silwal** for the Partial Fulfillment of the requirement for Bachelor in Computer Application, sixth semester

has been approved for further development

<u>Proposal Evaluation Committee</u>

SIGNATURE Of Supervisor **Jagdish Bhatta** Lecturer/Supervisor Department of Computer application Patan Multiple Campus Patan Dhoka, Lalitpur	**SIGNATURE Of HOD/Coordinator** **Roshan Tandukar** Program Coordinator Department of Computer application Patan Multiple Campus Patan Dhoka, Lalitpur
SIGNATURE Of Internal Examiner **Internal Examiner**	**SIGNATURE Of External Examiner** **External Examiner**

ABSTRACT

The machine learning-based laptop price prediction system, utilizing the Random Forest Algorithm, offers a method to estimating laptop prices based on a diverse set of specifications. This paper proposes a system where price is dependent variable which is predicted and this price is derived from factors like Laptop's model, RAM, HDD, SSD, GPU, CPU, IPS Display, and Touchscreen. By employing advanced machine learning techniques, the system analyzes these features to predict prices with high accuracy, thereby assisting consumers and retailers in making informed decisions. The integration of the Random Forest Algorithm ensures a balanced approach to handling the complexity and variability inherent in laptop specifications, making it a valuable tool in the dynamic tech market. This system is implemented using HTML, CSS, JavaScript for the frontend development while Python and Flask were used for implementation of the backend side.

Key Words:

Laptop Price Prediction System, Machine learning, laptop price prediction, Random Forest Algorithm, Laptop Specifications, HTML, CSS, JS, Python, Flask

ACKNOWLEDGEMENT

We want to thank our supervisor, Jagdish Bhatta, for giving us the chance to work on the interesting project about laptop Price Prediction System. It provided us with the opportunity to conduct extensive research and discover new tools and technologies.

We would like to express our special thanks of gratitude to our Campus Chief **Dr. Laxman Singh Kunwar** who gave us permission for doing this Project.

We would like to acknowledge the guidance and continuous supervision of Patan Multiple Campus, as well as their provision of necessary project information and support.

Lastly, we extend our gratitude to Tribhuvan University for offering us the chance to explore project ethics through the Computer Application course, helping us assess our knowledge and expand it further.

Yours sincerely

Ayush Silwal

Table of Content

LIST OF FIGURE

LIST OF TABLES

LIST OF ABBREVIATIONS

CSS	Cascading Style Sheet
DFD	Data Flow Diagram
ERD	Entity Relationship Diagram
HTML	Hyper Text Markup Language
JS	Java Script
LPPS	Laptop Price Prediction System
MySQL	My Structured Query Language

Chapter 1: Introduction

1.1 Introduction

The Laptop Price Prediction System is an application developed to predict laptop prices based on Random Forest Algorithm using their specifications, making the purchasing and pricing process more informed and efficient. This system works by analyzing features such as brand, processor type, RAM size, storage capacity, screen size, and other relevant attributes to estimate the price of the laptop accurately. This saves time and money for users utilizing the system.

By using the Laptop Price Prediction System, users can make smarter decisions when buying a laptop. They can quickly find out if a laptop is worth its price by comparing its specifications to similar models. This tool simplifies the buying process, ensuring users don't overpay for features they don't need or miss out on important ones. Ultimately, this system helps users feel confident and satisfied with their laptop purchases, knowing they have made a well-informed choice.

1.2 Problem Statement

In today's technology-driven world, laptops have become essential tools for both personal and professional use. With a wide range of brands, models, and specifications available in the market, predicting the price of a laptop based on its features can be a challenging yet valuable task. An accurate price prediction model can assist users in making informed purchasing decisions. Traditional methods of price estimation often fall short due to their inability to capture the non-linear relationships between different specifications and the final price.

The objective of this project is to develop a machine learning-based system that can predict the price of a laptop with high accuracy using the Random Forest algorithm. This algorithm is chosen for its ability to handle large datasets, manage the complexity of numerous input variables, and reduce overfitting through the aggregation of multiple decision trees. By leveraging this ensemble method, the system aims to provide more reliable and precise price predictions, helping consumers make informed purchasing decisions and assisting manufacturers in pricing strategy and market analysis.

1.3 Objectives

Some of the objectives of this system are as follows:

- To predict price of laptop for users using Random Forest Algorithm.
- To give predicted history to users

1.4 Scope and limitation

1.4.1 Scope

This system will be used in Laptop Price Prediction. This system can be accessed by every user having a proper username and password. Admin can manage the data of users and prediction history either to modify or delete them.

1.4.2 Limitations

i. Reliable Internet: The Laptop Price Prediction System needs good internet to work well. If the internet is not good or keeps going in and out, it might be hard for users to use the system or make bookings.

ii. Technical Issues: Laptop Price Prediction System can encounter issues such as offline, browser compatibility issues, and software bugs. These can affect the entire process of booking.

iii. Security: Laptop Price Prediction System stores and collects various data about its users, such as personal information. To prevent unauthorized access, the system should implement robust measures such as encryption.

1.5 Methodology

The methodology behind the Laptop Price Prediction System involves collecting a large dataset of laptop prices and their specifications. The data is then preprocessed to clean and organize it, removing any inconsistencies or errors. Next, the system uses machine learning algorithms to analyze the data. Key features like brand, processor type, RAM size, storage capacity, and screen size are used as input variables. The machine learning model is trained on this data to learn patterns and relationships between these features and the laptop prices. Once trained, the model can predict the price of a laptop based on its specifications. This prediction is continually refined as the system is updated with new data, ensuring accurate and up-to-date price estimates.

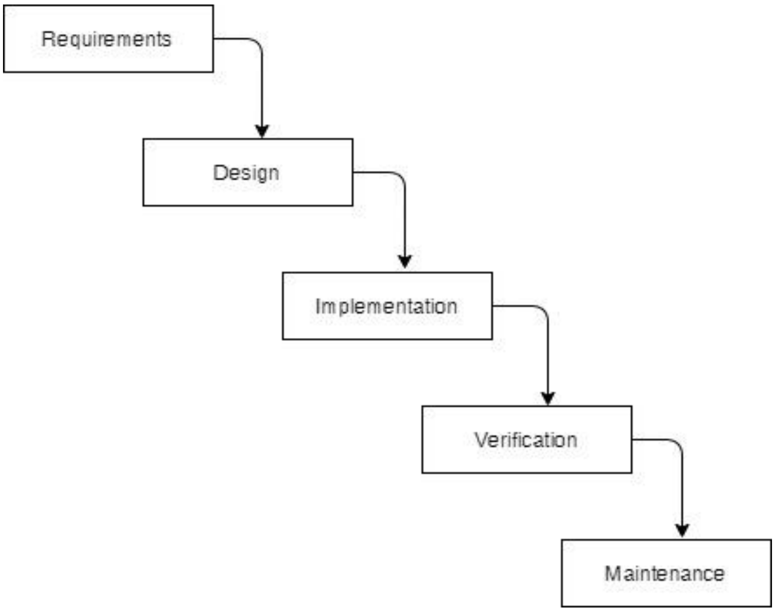


Figure 1: Waterfall Model for Laptop Price Prediction System

1.6 Report Organization

Chapter One

The built system's introduction is covered in this chapter. It has five subtopics: It includes five subtopics i.e. Introduction, Problem of statement, Objective, limitations. It describes the early stages of a software development.

Chapter Two

This chapter discusses the background study, an overview of relevant current systems, and advantages and disadvantages of each. It covers two subjects, namely background study and literature reviews. This chapter focuses on the background of the project as well as the reference you took to start this project.

Chapter Three

This chapter presents the System Analysis and Design including Requirement analysis and Feasibility Analysis. Systems analysis is the method through which a person studies a system so that an information system can be examined, modeled, and evaluated. Functional and non-functional analysis are the two components of requirement analysis. Technical, operational, and economic feasibility are every component of feasibility analysis.

Chapter Four

This Chapter presents the Implementation, Testing and debugging. Implementation includes the tools used and the details of how modules were implemented. The login form, register form is all tested using test cases and unit testing

Chapter Five

In Chapter five, Conclusion, Lesson learnt and Future Enhancements are briefly explained. In conclusion, we define the project's overall goal and objectives. The features that will be introduced but currently are not available are listed in Future Enhancements.

Chapter 2: Background Study and Literature Review

2.1 Background Study

The laptop market is quite dynamic the costs are affected by a number of factors, including design, added features like graphics cards and display quality, technical specs such as processor, RAM, and storage and also brand reputation. Laptops are the latest product of machine learning, which has become a potent tool for pricing prediction in a variety of industries, including real estate and autos. Accurate forecasts are made possible by machine learning algorithms that analyze enormous datasets to find patterns and links between a laptop's attributes and pricing. To create a strong model, though, a number of issues including assuring data quality, choosing pertinent characteristics, and taking price volatility into consideration must be resolved. This research investigates the use of machine learning to forecast laptop costs, perhaps benefiting customers and merchants in the process.

2.1.1 Fundamental Theories and General Concepts

Machine learning is a field of inquiry devoted to understanding and building methods that “learn" – that is, methods that leverage data to improve performance on some set of tasks. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide application, such as in medicine, email filtering, speech recognition, agriculture, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed task. [1]

Ensemble Learning: Ensemble learning refers to techniques that create multiple models (typically of the same type) and combine them to produce improved results. The fundamental idea behind ensemble methods is that a group of weak learners can be combined to form a strong learner, enhancing the model's accuracy and robustness. There are several types of ensemble methods, including bagging and boosting. Bagging, or Bootstrap Aggregation, is particularly relevant here as it is the foundational technique used in the Random Forest algorithm. [3]

2.2 Literature Review

Predicting price of laptops has been studied extensively in various researches. Listian discussed, in her paper written for Master thesis, that regression model that was built using decision tree & Random Forest Repressor can predict the price of a laptop that has been leased with better precision than multivariate regression or some simple multiple regression. This is on the grounds that Random Forest Algorithm is better in dealing with datasets with more dimensions and it is less prone to over fitting and under fitting. The weakness of this research is that a change of simple regression with more advanced Random Forest Algorithm regression was not shown in basic indicators like mean, variance or standard deviation. [2]

Ensemble methods, particularly Random Forest, have become a standard approach in predictive modeling due to their robustness and ability to handle diverse data types. Breiman, who introduced the Random Forest algorithm, demonstrated that this ensemble method significantly reduces the variance of predictions by averaging multiple decision trees trained on different subsets of data. His work has been cited extensively in subsequent research, underscoring the effectiveness of Random Forests in various predictive tasks, including price prediction. [4]

Decision Trees have been widely used in various price prediction tasks due to their interpretability and ease of implementation. Breiman introduced the concept of Classification and Regression Trees, which laid the foundation for many subsequent studies in price prediction. In a more recent study, Chen utilized decision trees to predict real estate prices, showing that these models are particularly effective when dealing with categorical and continuous variables. [5]

Moreover, Quinlan (1993) developed the C4.5 algorithm, an extension of decision trees that improved handling of continuous attributes. This work has been instrumental in the development of more advanced tree-based methods like Random Forests, which aggregate the predictions of multiple decision trees to enhance accuracy. [6]

Chapter 3: System Analysis and Design

3.1 System Analysis

This system is created through a series of steps, starting with the analysis of the requirements, followed by the design, implementation, testing, and maintenance. All of the functional and nonfunctional requirements are examined during the requirement analysis, and the system is then designed in accordance with the requirements. Following the design phase, the coding and development phase is started, and then the system is integrated and tested. If the testing is successful, the system is put into use; if not, some maintenance is carried out before the system is put into use.

3.1.1 Requirements Collection

For this system, requirements are basically identified through functional and nonfunctional requirements.

I. Functional Requirements.

For Users

1. The system allows users to predict the price of laptop.
2. The system displays the predicted price.

For System admin:

1. The system allows the admin to login and logout from the system.
2. The system allows the admin to manage user list.
3. The system allows the admin to manage the prediction history.

Use case

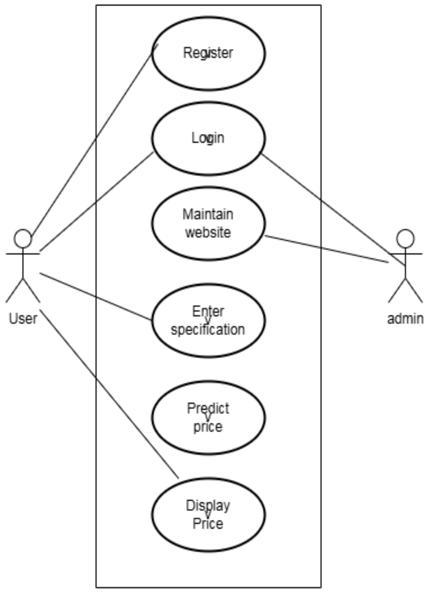


Figure 3. 1 Use case diagram for Laptop Price Prediction System

II. Non- Functional

Security

Admin is able to log into the system and have access to the user's database system and be protected by the user login screen that requires a username and password.

Availability

The system is available for 24 hours service as users can apply from anywhere and at any time.

Performance

The performance of the system is fast and accurate with the minimum use of data and memory as it will provide a fast response to the user's actions. The system is able to handle expected and unexpected errors and also large amounts of data at any given time.

Reliability

The system is reliable as it should perform function and run without a failure, and it have to be reliable due to importance of data and damages that can be caused by incorrect or incomplete data.

3.1.2 Feasibility Analysis

i. Technical Analysis

The factors for future project's technical feasibility are that the basic programming language which will be suitable for project is available and the libraries required for project will be capable of achieving the result that we are aiming for. It also depends on factors such as the available technology infrastructure and the specific requirements of the desired system.

ii. Operational Analysis

This system is including all the requirements used for the Laptop Price Prediction System and this system will be completely operational and can be successfully implemented and easy to use this system as it will be user-friendly.

iii) Economic Analysis

The system which we are going to develop is economically reasonable and cost effective. As all the tools and resources required are either open sources or free. After the completion of the system won't need to deploy any new hardware and software as the required software and hardware. The existing resource of the system can be used.

iv) Schedule Analysis

The system which we are going to develop will be completed within the scheduled time and should not surpass the scheduled time.

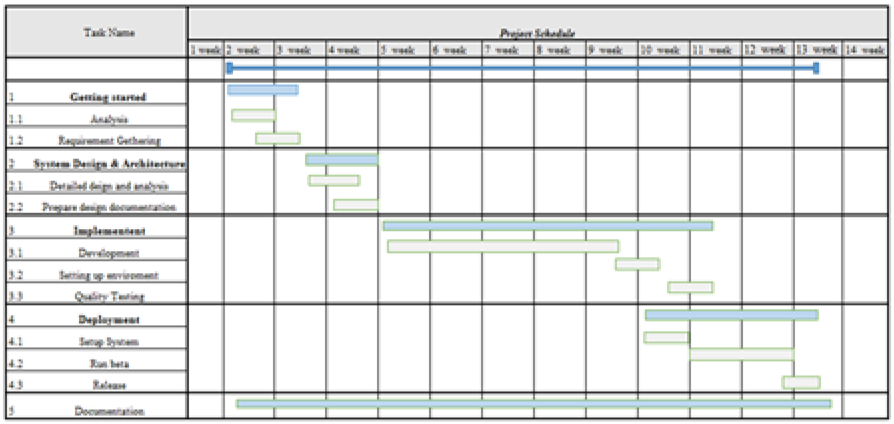


Figure 3. 2 Gantt chart for Laptop Price Prediction System

3.1.3 Data modeling (ER-Diagram)

There are three entities in our ER diagram. They are admin, users and predictions. The admin has one to many relationships with every entity. The users have one to many relations with predictions.

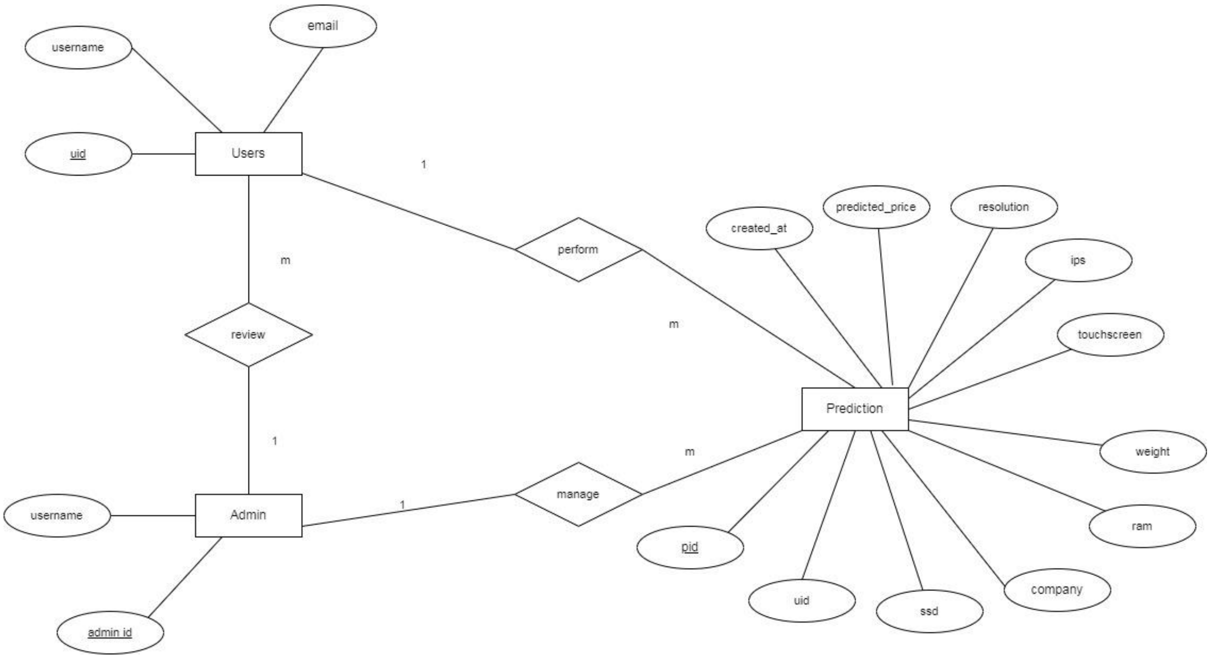


Figure 3. 3 Entity Relation diagram for Laptop Price Prediction System

3.1.4 Process modeling(DFD)

Data Flow Diagram consists of two levels of DFD which are context diagram and level 1 DFD. This are used to make the system.

Level 0: Context DFD

In Level 0 DFD, login request, prediction, registration is input for LPPS and user request for login and register and admin request for login. User will perform prediction by giving in some specification.

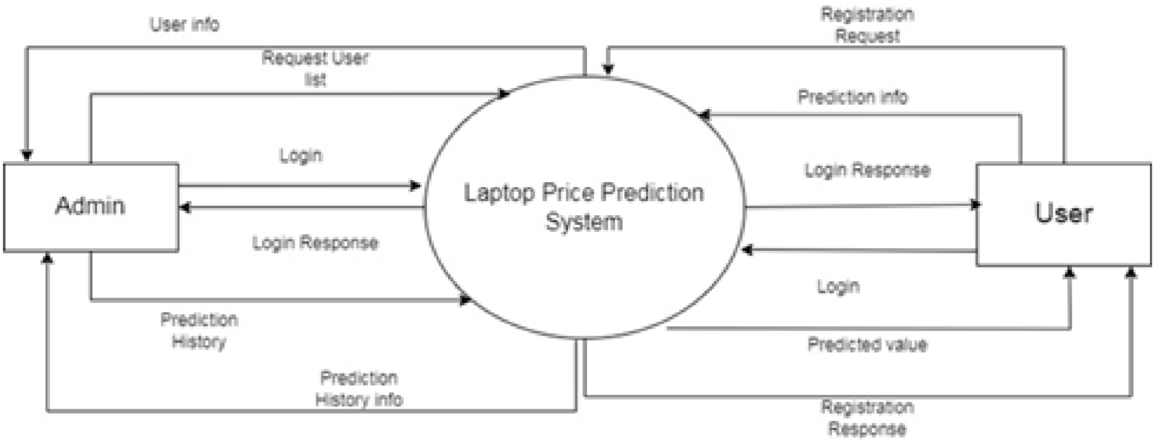


Figure 3. 4 Level 0 DFD for Laptop Price Prediction System

Level 1 DFD:

In level 1 DFD there are three processes where login is responsible for login of admin and users. Prediction is responsible for user prediction. Registration is responsible for user registration. There are three entities Admin, user and system and two data stores which are users and predictions.

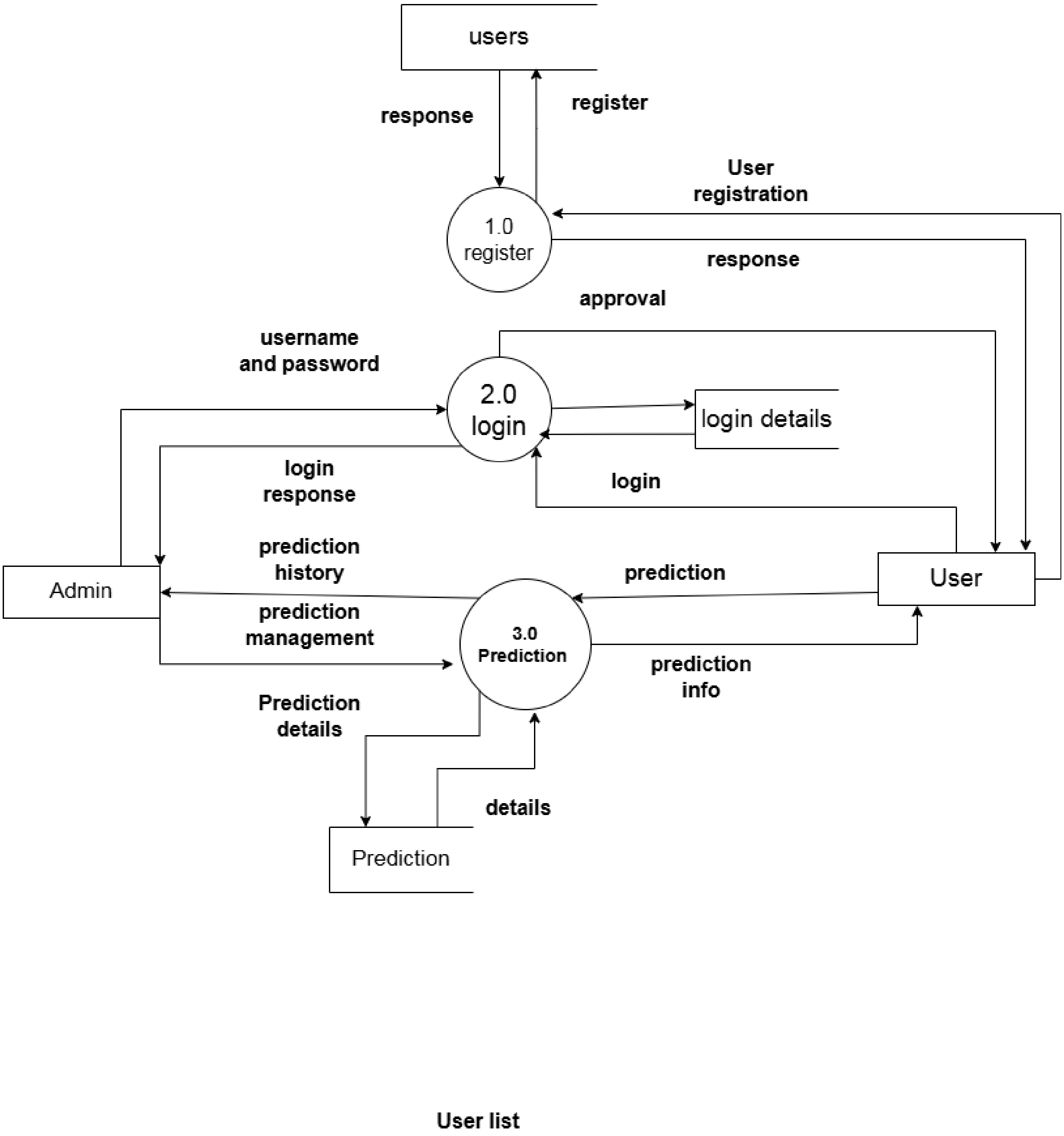


Figure 3. 5 Level 1 DFD for Laptop Price Prediction System

3.2 System Design

3.2.1 Architectural Design

In architectural design, LPPS system is connected to web server which is linked to its database and vice-versa.

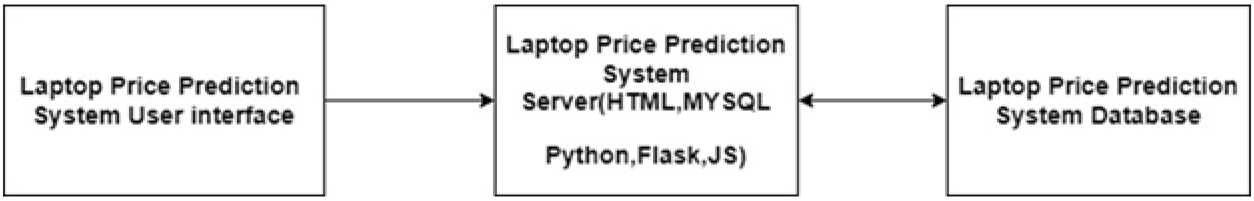

Laptop Price Prediction System User interface
Laptop Price Prediction System Server(HTML,MYSQL Python,Flask,JS)
Laptop Price Prediction System Database

Figure 3. 6 Architectural Design of Laptop Price Prediction System

3.2.2 Database Schema Design

In database schema, it shows relation link between tables.

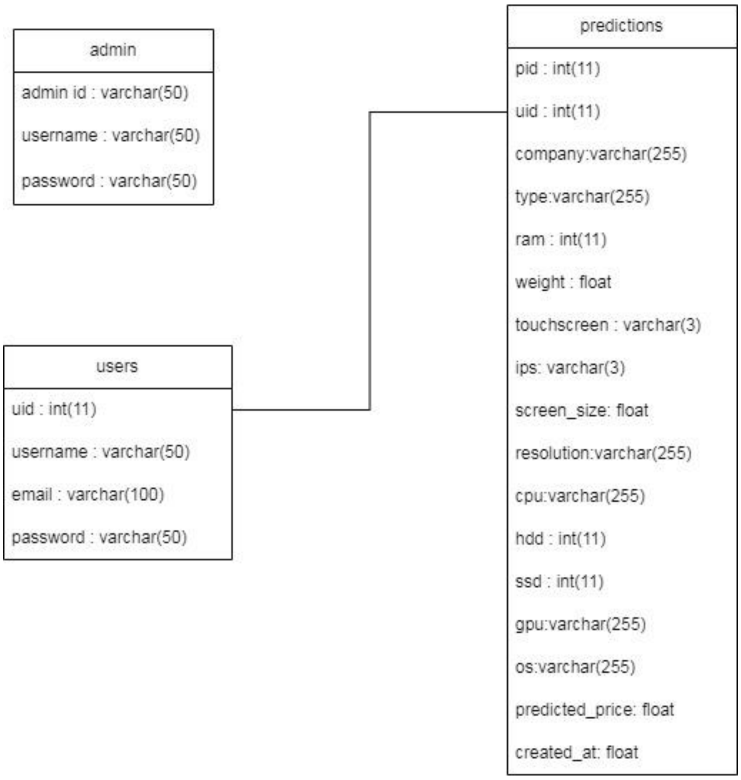


Figure 3. 7 Database Schema Design for Laptop Price Prediction System

3.2.3 Interface Design

In interface design it shows the interface of the website. It shows dashboard and various other operations.

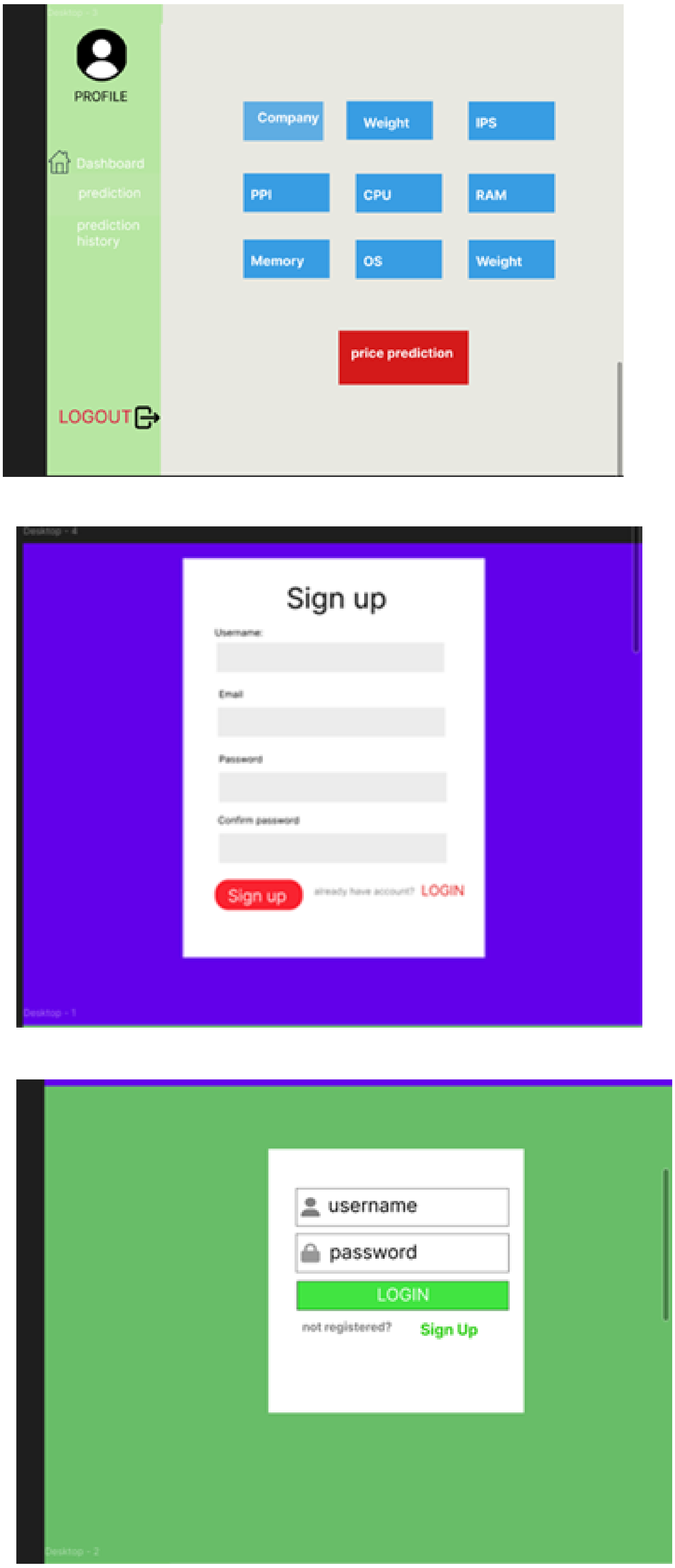
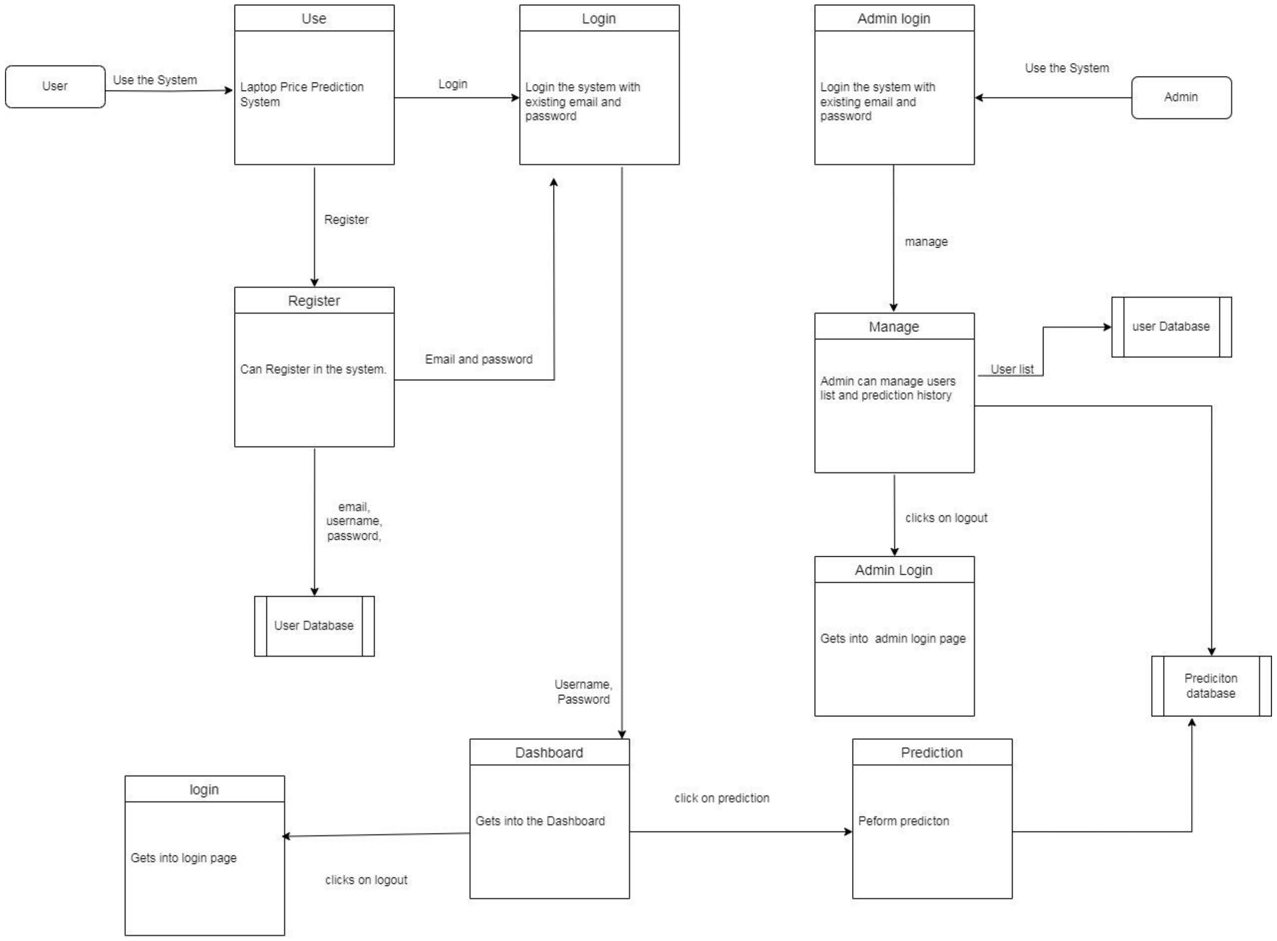


Figure 3. 8 Interface of Laptop Price Prediction System

3.2.4 Physical DFD



` **Figure 3. 9 Physical DFD for Laptop Price Prediction System**

3.3 Algorithm details

Random Forest Regression Algorithm

The Random Forest Regression algorithm is an ensemble learning method that builds multiple decision trees on various subsets of the dataset and averages their predictions. This approach helps improve accuracy, reduce overfitting, and create a more robust model compared to a single decision tree. In regression tasks, Random Forests are particularly effective because they can capture complex relationships within the data.

Steps of the Random Forest Regression Algorithm

1. **Draw Random Samples**:
   - For each tree in the forest, draw a bootstrap sample (random sample with replacement) from the training dataset. This sampling method means each tree sees a slightly different dataset.
2. **Build a Decision Tree for Each Sample**:
   - For each bootstrap sample, build a decision tree using a subset of features chosen randomly at each split. Limiting the number of features available at each split reduces correlation between trees and improves overall model robustness.
3. **Make Predictions with Individual Trees**:
   - Once all the trees are built, make predictions for a new input by passing it through each tree. Each tree outputs its prediction based on the input features.
4. **Aggregate Predictions**:
   - For regression, the final prediction is the average of all individual tree predictions. This averaging process smoothens the prediction, leading to more stable and accurate outputs.

Pseudocode

```
Input: Training data (X, y), Number of trees (T), Maximum depth (max_depth)

Initialize an empty list of trees

For t = 1 to T:
    Draw a bootstrap sample from the training data (X_sample, y_sample)
    Train a DecisionTree on (X_sample, y_sample) with max_depth
    Add the trained tree to the list of trees

For each test sample x:
    Initialize an empty list of predictions
    For each tree in the list of trees:
        Predict the target value using the tree
        Append the prediction to the list of predictions
    Average the predictions to get the final prediction for x

Output: Final predictions for all test samples
```

Chapter 4: Implementation and testing

4.1 Implementation

Data was gathered in the first phase. This step was a crucial time in the project's growth. The project's whole physical design has been translated into functional computer code. The chapter before spoke about a lot of the technology and tools that were used to create the system.

4.1.1 Tools Used

This chapter discusses the tools used to create the front and back ends of this project.

Front-end

HTML5, CSS3, and JavaScript are used for developing the front end.

HTML5 (Hyper Text Markup Language)

HTML is utilized in our project to build webpages and gives us with the general skeletal structure of the webpage. HTML is our project's major display language since it allows us to expose the structure of our page in the browser, which allows us to debug simply and effectively.

CSS3 (Cascading Style Sheets)

In our project, CSS is utilized to style the HTML content. It is utilized to make our website responsive and to make the website's design clean.

JavaScript

JavaScript is used to make our webpages interactive similarly JavaScript is also used for form validation.

Backend

The Backend is assembled using Python and Flask. Flask is a micro web framework written in Python. MySQL is used to design databases.

Python

Python is a versatile programming language used for various purposes, including data processing, automation, and machine learning. Python is used to implement the machine learning model, perform data analysis, and manage backend logic.

Flask

Flask is a lightweight web framework that makes it easy to create web applications. It handles routing, request handling, and rendering HTML templates, allowing you to build web interfaces and APIs quickly.

MySQL

The primary utilization of MySQL is as a database. It is a place where data is stored. With the aid of databases, data remains intact regardless of whether the device is switched off. MySQL is mostly used for creating multiple tables for different purposes and store various data.

XAAMP

To fulfill the requirements of the project, XAAMP serves as a local server and MySQL is used as a local server and database.

4.1.2 Implementation Details of Modules

The proposed approach is composed up of two components which are user module and admin module.

4.1.2.1 User Module:

In the Laptop Price Prediction System, the User Registration & Login module enables users to sign up and log in using their email credentials. It interacts with the backend for authentication and user management. Upon successful registration or login, the user's information is securely stored in a MySQL database, ensuring data integrity and security. The module manages user sessions directly through traditional session handling, allowing users to access their dashboards and prediction history without needing to log in repeatedly.

4.1.2.2 Admin Module:

The admin is provided a username and password for the admin dashboard. Admin can access the dashboards by entering the proper username and password. Users and their prediction are handled by the admin. The admin module has a dashboard where the admin may manage users and predictions and log out of the system by pressing the logout button.

4.1.2.3 Prediction Module

The Prediction Module in the Laptop Price Prediction System estimates laptop prices based on key specifications like model, RAM, storage, GPU, CPU, display type, and touchscreen capability. Users input these details via a form, and the module uses a custom machine learning algorithm to analyze the specifications and predict a likely price range. The results are presented clearly, enabling users to make informed decisions on laptop purchases and pricing in the competitive tech market.

4.2 Testing

We tested a complete software product using the requirements document to check how it behaves and make sure it works well. It's important to test an app or website thoroughly before releasing it. This led to creating specific tests for this website. We used different methods to test it, which are explained below.

4.2.1 Test Case for Unit Testing

Table 1 Users Registration Test Table

ID	Test Case Description	Test Data	Expected Result	Actual Result	Pass/Fail
P_REG	User forgets to fill a particular field	Username: ayush123 Email:ayush123@gmail.com Password: ayush123 Confirm Password:	Display message that instruct to fill out the empty field	Outcome as expected	Pass
P_REG1	User enters incomplete username User enters	Username: ayush Email:ayush123@gmail.com Password: ayush1234 Confirm: Password:ayush1234	Display message that instruct to fill out username properly Display	Outcome as expected	Pass
P_REG2	Users enter email in wrong format	Username; ayush123 Email: ayush123.com Password:ayush123 Confirm Password: ayush123	Display message that instruct to fill out email correctly	Outcome as expected	Pass
P_REG3	Users enters different password in password and confirm password field	Username: ayush123 Email: ayush123@gmail.com Password: ayush123 Confirm Password: ayus123	Display message that instruct to fill out Password correctly	Outcome as expected	Pass

Table 2 Users Login Test Table

ID	Test Case Description	Test Data	Expected Result	Actual Result	Pass/Fail
U_LOG	User enters wrong username	Username; ayush123 Password; ayush1234	Login Failed	As expected	Pass
U_LOG1	User enters wrong password	Username; ayush1234 Password; ayush123	Login Failed	As Expected	Pass

Table 3 Admin Login Test Table

ID	Test Case Description	Test Data	Expected Result	Actual Result	Pass/Fail
A_LOG	Admin enters Wrong Username	Username : ayush1 password: ayush123	Login Failed	As expected	Pass
A_LOG1	Admin enters Wrong Password	Username : Ayush123 Password: ayush1234	Login Failed	As expected	Pass

Table 4 Prediction Test

ID	Test Case Description	Test Data	Expected Result	Actual Result	Pass/Fail
P_TEST	Laptop price prediction	Brand: Apple Type: Ultrabook Ram(GB):8 Weight:1.37 Touchscreen: no IPS Display: no Screen Size:13.3 Resolution: 2560x1600 CPU Brand: Intel core i5 HDD (GB):0 SSD (GB):128 GPU Brand: Intel OS Type: mac	Price of the laptop RS 69108	Price of the actual laptop RS 71378	Pass
P_TEST1	Laptop price prediction	Brand: HP Type: Notebook Ram(GB):8 Weight:1.86 Touchscreen: no IPS Display: no Screen Size:15.6 Resolution: 1920x1080 CPU Brand: Intel core i5 HDD (GB):0 SSD (GB):256 GPU Brand: Intel OS Type: no os	Price of the laptop RS 34226	Price of the actual laptop RS 30636	Pass

4.2.2 Test case for System Testing

Check system behavior.

- If the website starts successfully and has all the necessary features and pages.
- If the user can register and login to the site
- If the key features like log-in and log-out work as intended,
- If SESSION and COOKIE are working as intended.

Chapter 5: Conclusion and Future Recommendations

5.1 Lesson Learnt /Outcome

The completion of the Laptop Price Prediction System project has successfully achieved its primary objectives. This system allows users to input various specifications of laptops and receive an accurate price estimate based on a machine learning model. The project involved building a Random Forest model trained on a large dataset of laptops with varying specifications, including processor, RAM, storage, GPU, and display features. Through this project, we gained practical experience in data preprocessing, feature selection, model training, and deploying a machine learning model via a web interface. The project also highlighted the importance of handling real-world data complexities, tuning model parameters, and ensuring a user-friendly interface for seamless interaction.

5.2 Conclusion

The Laptop Price Prediction System offers a valuable tool for users, allowing them to make informed decisions about laptop pricing based on specifications. Built with Flask as the backend framework, the system leverages HTML, CSS, and JavaScript for a responsive and interactive front end. MySQL is used as the database to store user registration data, and the Random Forest model is integrated for price predictions. The system provides an easy-to-use interface that can be operated even by users with minimal technical knowledge. The accuracy of the predictions and user-friendly design contribute to a better shopping experience and assist in market analysis. This system demonstrates the potential of machine learning to transform decision-making in e-commerce and product pricing.

5.3 Future Recommendation

Several enhancements can improve the functionality and user experience of the Laptop Price Prediction System. Adding online payment options could transform it into a complete e-commerce platform. Incorporating real-time pricing data and periodic model retraining would ensure prediction accuracy aligns with market trends. User account features like password reset, profile management, and prediction history could enhance engagement, while a more modern, responsive design would improve accessibility across devices. Supporting multiple locations with localized pricing would make the system adaptable to different regions. Additional features, like budget-based laptop recommendations and upgrade suggestions, would increase its usefulness. Finally, experimenting with advanced models like Gradient Boosting or Neural Networks could further optimize prediction performance.

References

[1] S. V. Pandey M. A Random forest algorithm pertaining to the student performance analysis and prediction. International Journal of Computer Applications [Accessed: 08 22 2024]

[2] M. Listiani. (2009). Support vector regression analysis for price prediction in a car leasing application (Doctoral dissertation, Master thesis, TU Hamburg-Harburg) [Accessed: 08 22 2024]

[3] Smith, J., Brown, C., & Davis, M. (2019). An evaluation of regression models for predicting electronics prices. Computational Intelligence and Neuroscience, 2019, Article 123456. [Accessed: 08 22 2024]

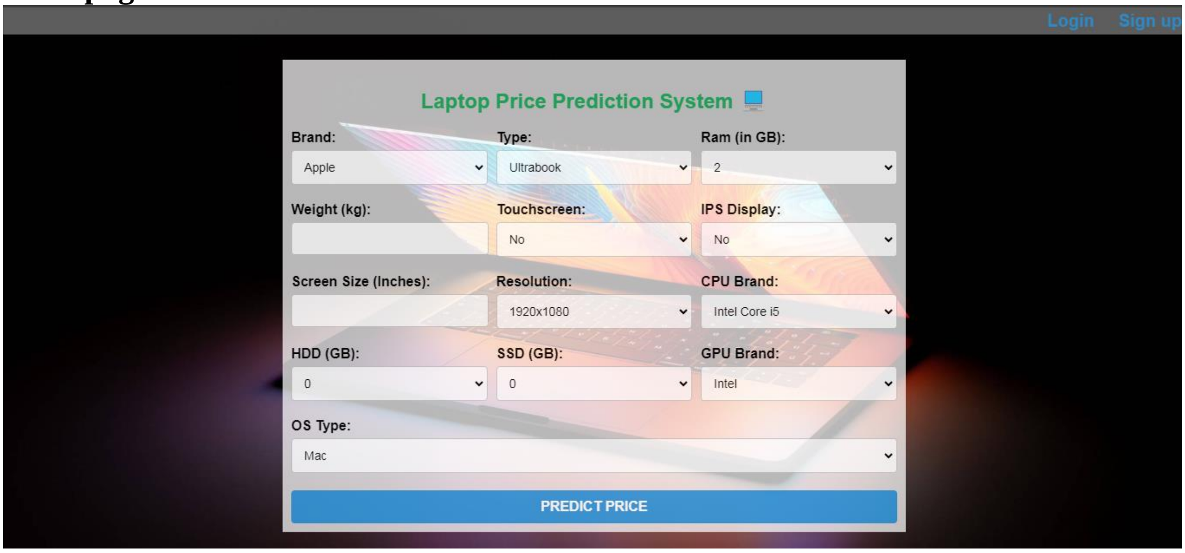
[4] Z.-H. Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC. [Accessed: 08 22 2024]

[5] L. Breiman. Random forests. *Machine Learning* [Accessed: 08 22 2024]

[6] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers.
[Accessed: 08 22 2024]

APPENDICES

Homepage



Laptop Price Prediction System

Brand: Apple

Type: Ultrabook

Ram (in GB): 2

Weight (kg):

Touchscreen: No

IPS Display: No

Screen Size (Inches):

Resolution: 1920x1080

CPU Brand: Intel Core i5

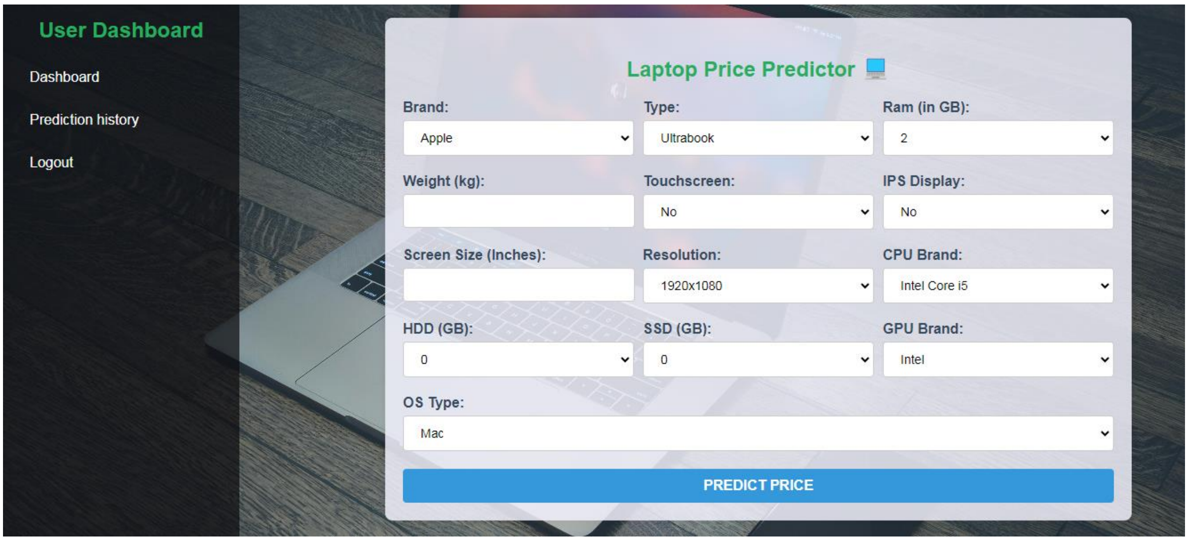
HDD (GB): 0

SSD (GB): 0

GPU Brand: Intel

OS Type: Mac

User Dashboard



Laptop Price Predictor

Brand: Apple

Type: Ultrabook

Ram (in GB): 2

Weight (kg):

Touchscreen: No

IPS Display: No

Screen Size (Inches):

Resolution: 1920x1080

CPU Brand: Intel Core i5

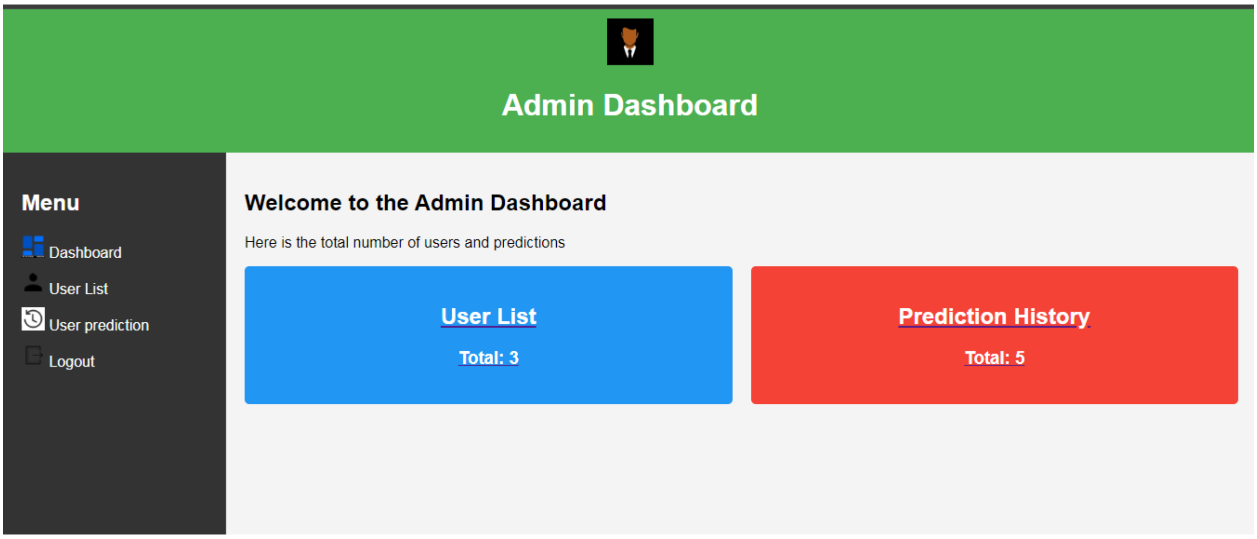
HDD (GB): 0

SSD (GB): 0

GPU Brand: Intel

OS Type: Mac

Admin Dashboard



Welcome to the Admin Dashboard

Here is the total number of users and predictions

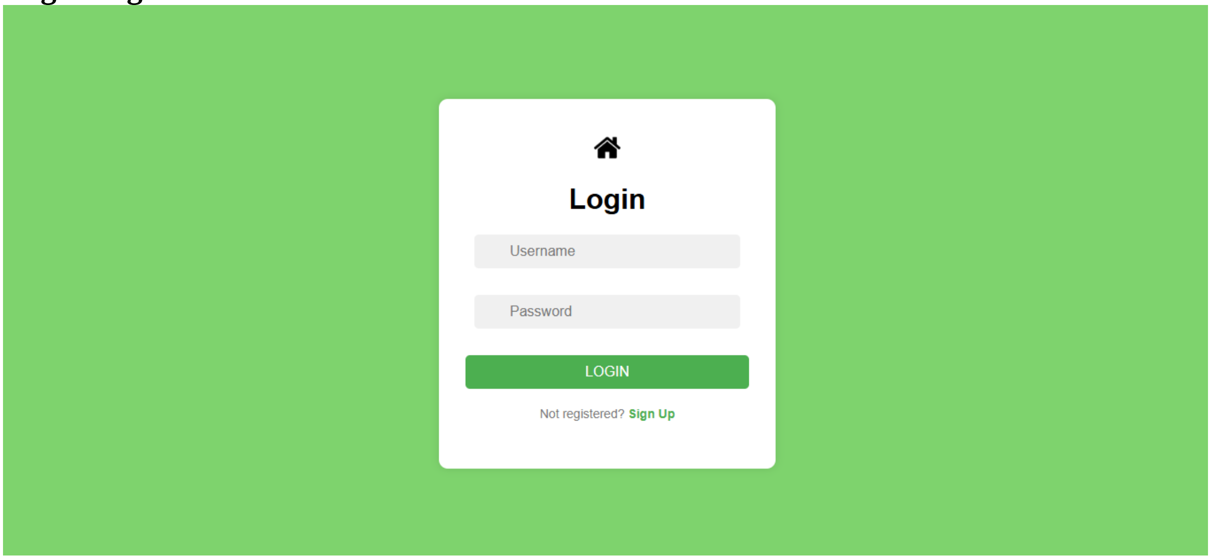
User List

Total: 3

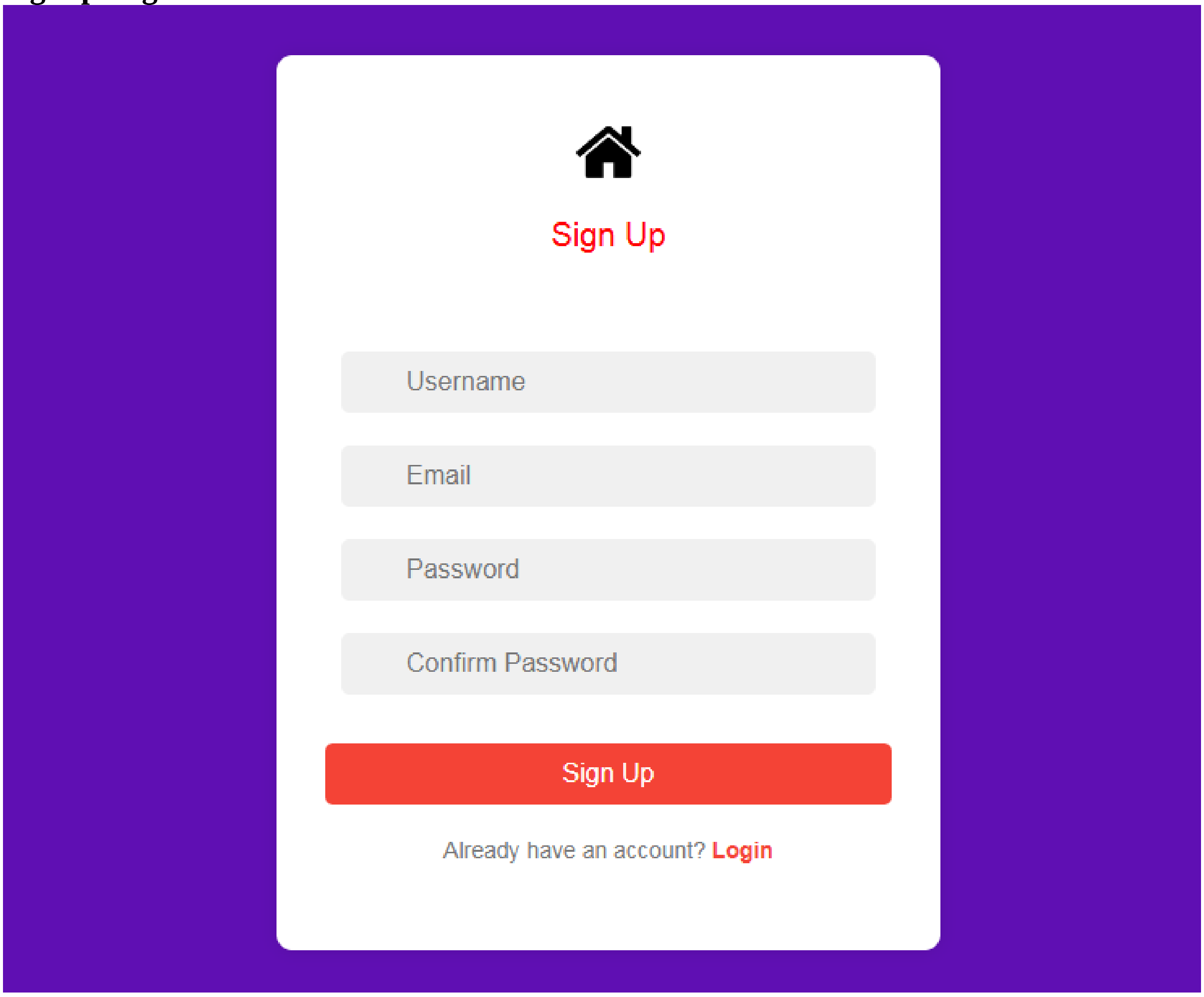
Prediction History

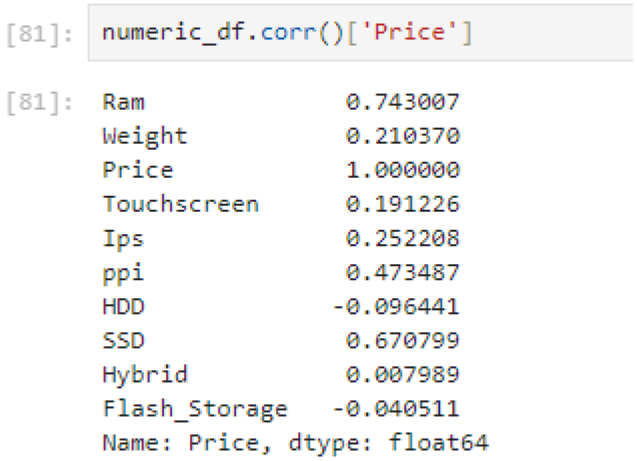
Total: 5

Login Page



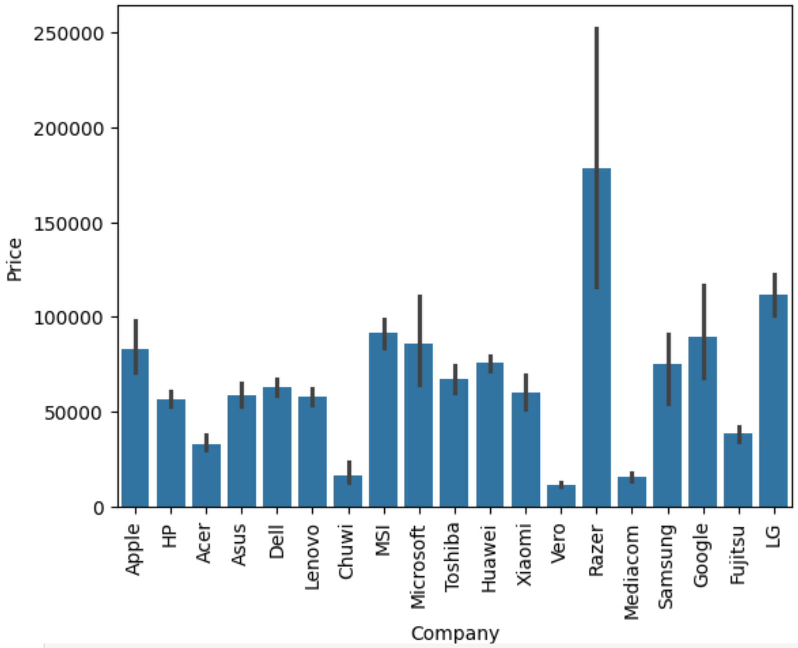
Signup Page



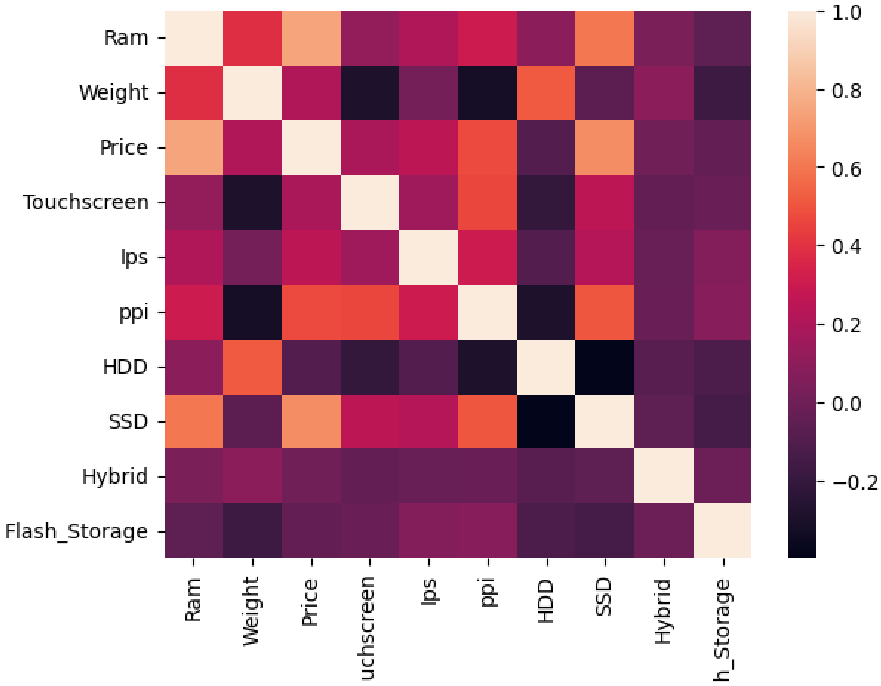


```
[81]: numeric_df.corr()['Price']
```

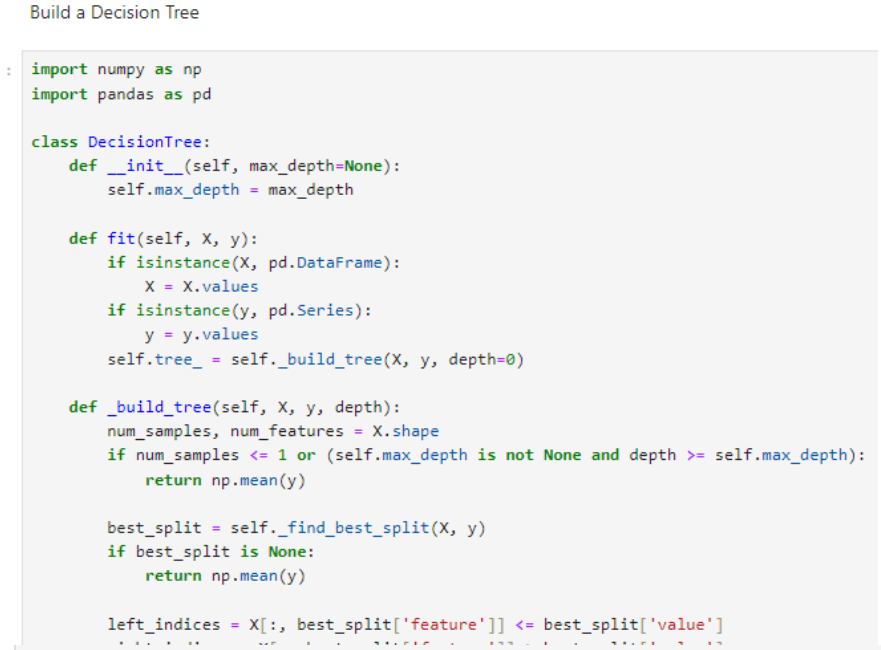
```
[81]: Ram              0.743007
      Weight           0.210370
      Price            1.000000
      Touchscreen      0.191226
      Ips              0.252208
      ppi              0.473487
      HDD             -0.096441
      SSD              0.670799
      Hybrid           0.007989
      Flash_Storage   -0.040511
      Name: Price, dtype: float64
```



```
[83]: <Axes: >
```

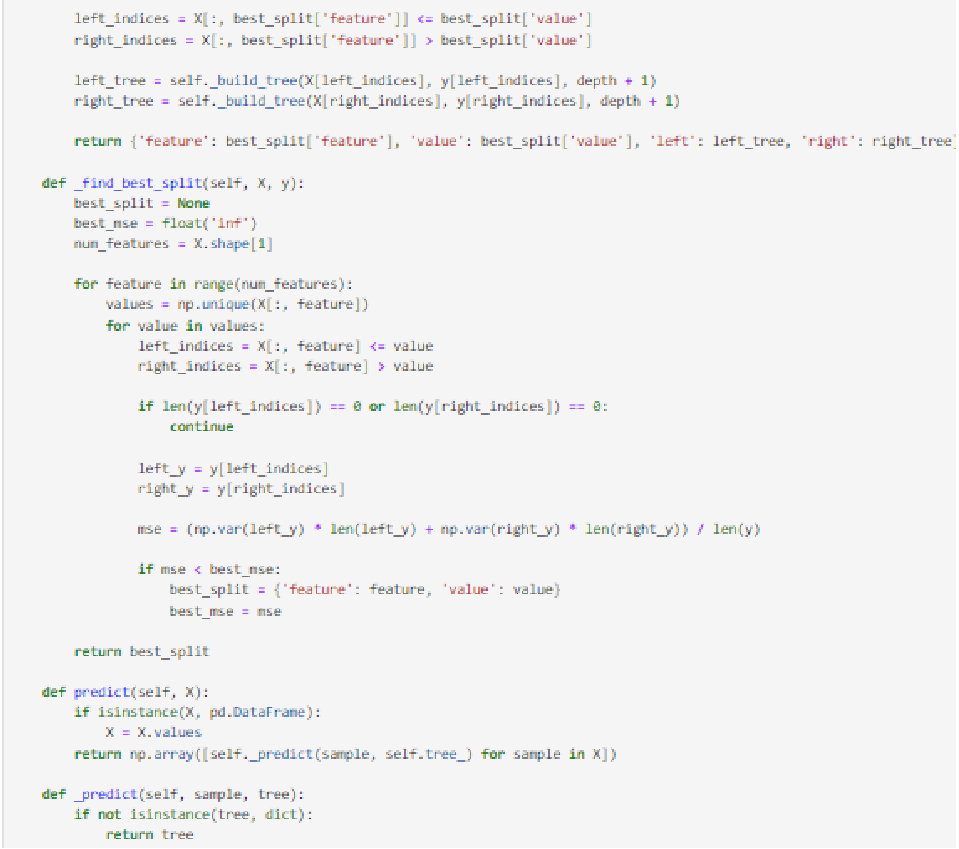


Decision Tree



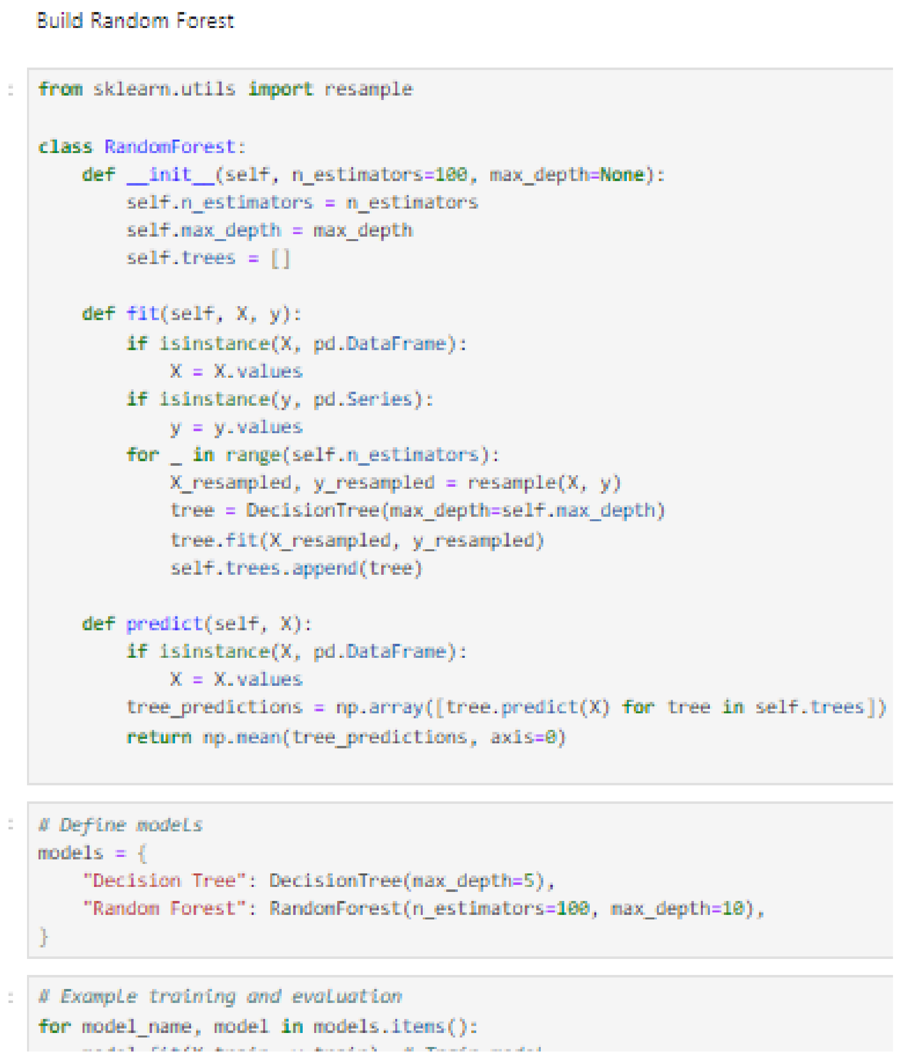
Build a Decision Tree

```
import numpy as np
import pandas as pd

class DecisionTree:
    def __init__(self, max_depth=None):
        self.max_depth = max_depth

    def fit(self, X, y):
        if isinstance(X, pd.DataFrame):
            X = X.values
        if isinstance(y, pd.Series):
            y = y.values
        self.tree_ = self._build_tree(X, y, depth=0)

    def _build_tree(self, X, y, depth):
        num_samples, num_features = X.shape
        if num_samples <= 1 or (self.max_depth is not None and depth >= self.max_depth):
            return np.mean(y)

        best_split = self._find_best_split(X, y)
        if best_split is None:
            return np.mean(y)

        left_indices = X[:, best_split['feature']] <= best_split['value']
```



```
        left_indices = X[:, best_split['feature']] <= best_split['value']
        right_indices = X[:, best_split['feature']] > best_split['value']

        left_tree = self._build_tree(X[left_indices], y[left_indices], depth + 1)
        right_tree = self._build_tree(X[right_indices], y[right_indices], depth + 1)

        return {'feature': best_split['feature'], 'value': best_split['value'], 'left': left_tree, 'right': right_tree

    def _find_best_split(self, X, y):
        best_split = None
        best_mse = float('inf')
        num_features = X.shape[1]

        for feature in range(num_features):
            values = np.unique(X[:, feature])
            for value in values:
                left_indices = X[:, feature] <= value
                right_indices = X[:, feature] > value

                if len(y[left_indices]) == 0 or len(y[right_indices]) == 0:
                    continue

                left_y = y[left_indices]
                right_y = y[right_indices]

                mse = (np.var(left_y) * len(left_y) + np.var(right_y) * len(right_y)) / len(y)

                if mse < best_mse:
                    best_split = {'feature': feature, 'value': value}
                    best_mse = mse

        return best_split

    def predict(self, X):
        if isinstance(X, pd.DataFrame):
            X = X.values
        return np.array([self._predict(sample, self.tree_) for sample in X])

    def _predict(self, sample, tree):
        if not isinstance(tree, dict):
            return tree
```

Random Forest



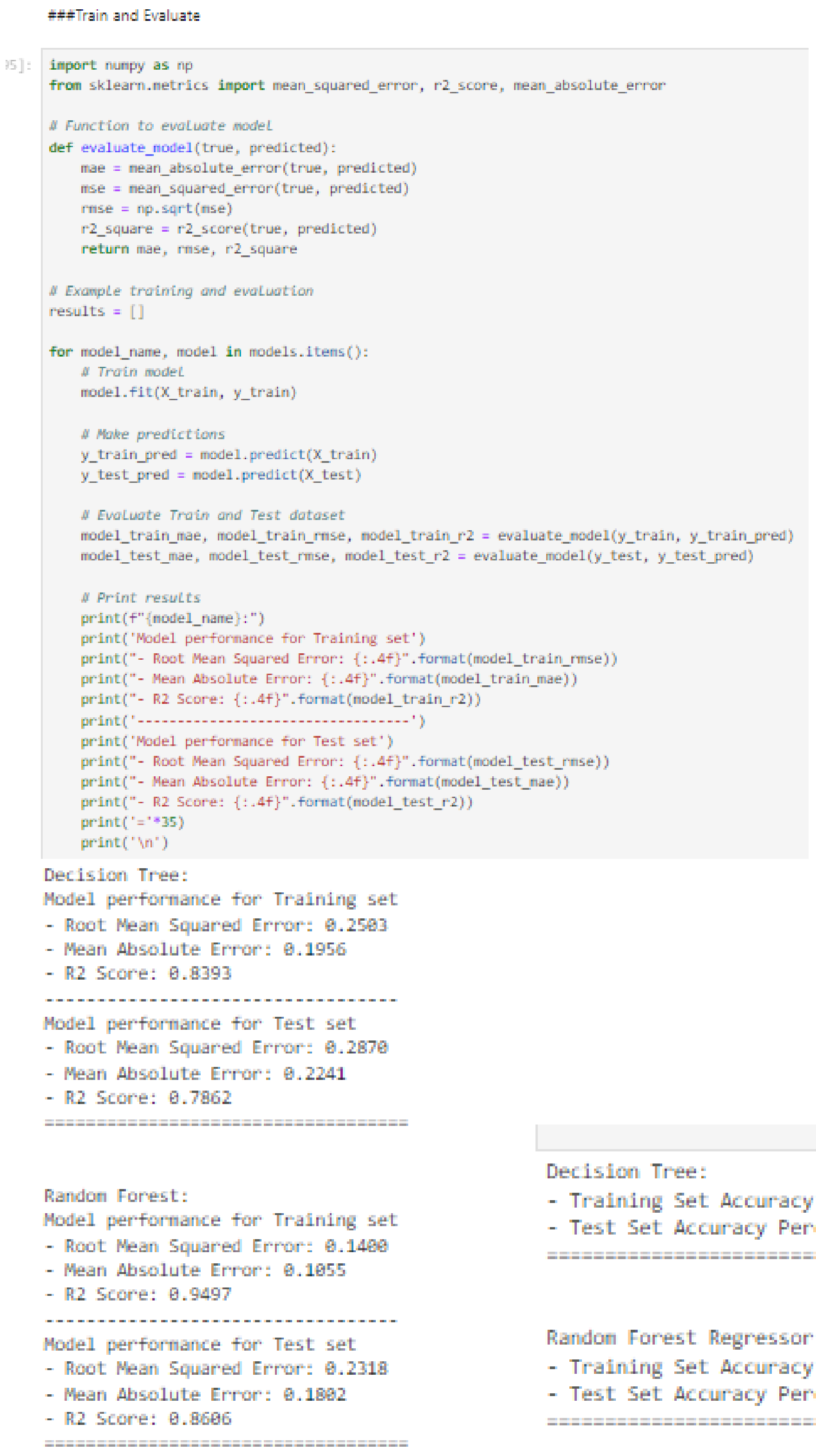
Build Random Forest

```python
from sklearn.utils import resample

class RandomForest:
    def __init__(self, n_estimators=100, max_depth=None):
        self.n_estimators = n_estimators
        self.max_depth = max_depth
        self.trees = []

    def fit(self, X, y):
        if isinstance(X, pd.DataFrame):
            X = X.values
        if isinstance(y, pd.Series):
            y = y.values
        for _ in range(self.n_estimators):
            X_resampled, y_resampled = resample(X, y)
            tree = DecisionTree(max_depth=self.max_depth)
            tree.fit(X_resampled, y_resampled)
            self.trees.append(tree)

    def predict(self, X):
        if isinstance(X, pd.DataFrame):
            X = X.values
        tree_predictions = np.array([tree.predict(X) for tree in self.trees])
        return np.mean(tree_predictions, axis=0)
```

```python
# Define models
models = {
    "Decision Tree": DecisionTree(max_depth=5),
    "Random Forest": RandomForest(n_estimators=100, max_depth=10),
}
```

```python
# Example training and evaluation
for model_name, model in models.items():
```

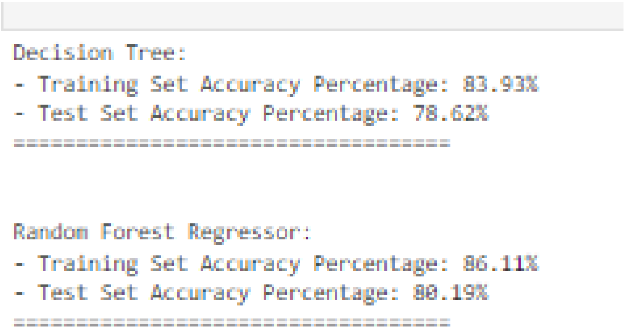
Train and Evaluate



###Train and Evaluate

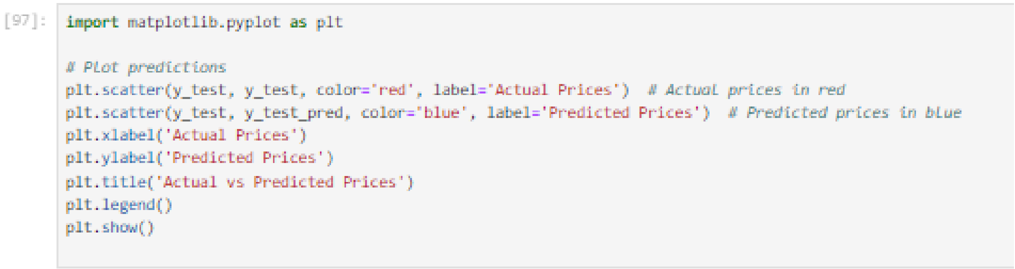
```python
import numpy as np
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error

# Function to evaluate model
def evaluate_model(true, predicted):
    mae = mean_absolute_error(true, predicted)
    mse = mean_squared_error(true, predicted)
    rmse = np.sqrt(mse)
    r2_square = r2_score(true, predicted)
    return mae, rmse, r2_square

# Example training and evaluation
results = []

for model_name, model in models.items():
    # Train model
    model.fit(X_train, y_train)

    # Make predictions
    y_train_pred = model.predict(X_train)
    y_test_pred = model.predict(X_test)

    # Evaluate Train and Test dataset
    model_train_mae, model_train_rmse, model_train_r2 = evaluate_model(y_train, y_train_pred)
    model_test_mae, model_test_rmse, model_test_r2 = evaluate_model(y_test, y_test_pred)

    # Print results
    print(f"{model_name}:")
    print('Model performance for Training set')
    print("- Root Mean Squared Error: {:.4f}".format(model_train_rmse))
    print("- Mean Absolute Error: {:.4f}".format(model_train_mae))
    print("- R2 Score: {:.4f}".format(model_train_r2))
    print('----------------------------------')
    print('Model performance for Test set')
    print("- Root Mean Squared Error: {:.4f}".format(model_test_rmse))
    print("- Mean Absolute Error: {:.4f}".format(model_test_mae))
    print("- R2 Score: {:.4f}".format(model_test_r2))
    print('='*35)
    print('\n')
```

```
Decision Tree:
Model performance for Training set
- Root Mean Squared Error: 0.2503
- Mean Absolute Error: 0.1956
- R2 Score: 0.8393
----------------------------------
Model performance for Test set
- Root Mean Squared Error: 0.2870
- Mean Absolute Error: 0.2241
- R2 Score: 0.7862
===================================


Random Forest:
Model performance for Training set
- Root Mean Squared Error: 0.1400
- Mean Absolute Error: 0.1055
- R2 Score: 0.9497
----------------------------------
Model performance for Test set
- Root Mean Squared Error: 0.2318
- Mean Absolute Error: 0.1802
- R2 Score: 0.8606
===================================
```

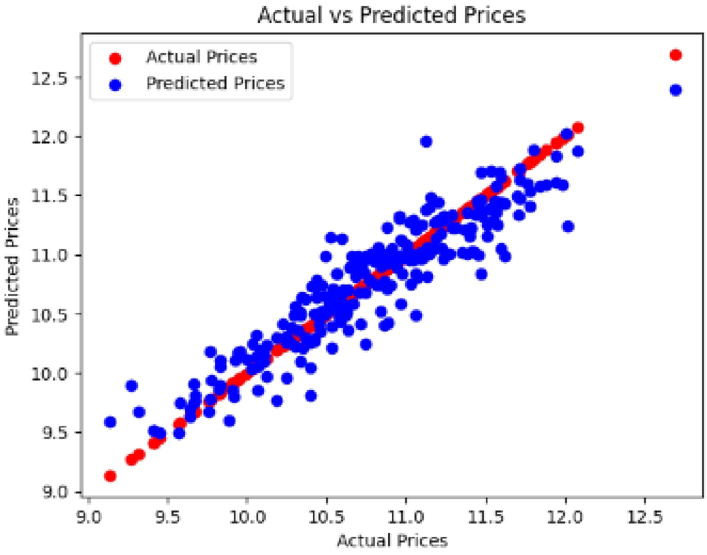


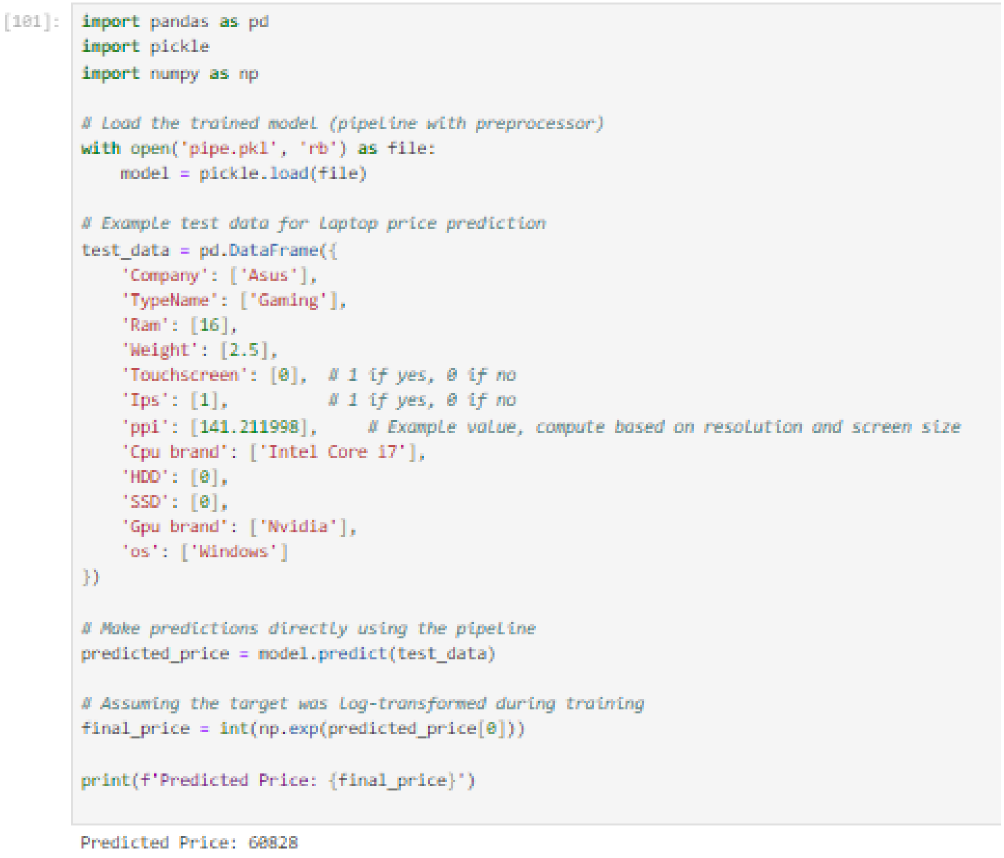
```
Decision Tree:
- Training Set Accuracy Percentage: 83.93%
- Test Set Accuracy Percentage: 78.62%
===================================


Random Forest Regressor:
- Training Set Accuracy Percentage: 86.11%
- Test Set Accuracy Percentage: 80.19%
===================================
```

Actual VS Predicted Price



```
[97]: import matplotlib.pyplot as plt

      # Plot predictions
      plt.scatter(y_test, y_test, color='red', label='Actual Prices')  # Actual prices in red
      plt.scatter(y_test, y_test_pred, color='blue', label='Predicted Prices')  # Predicted prices in blue
      plt.xlabel('Actual Prices')
      plt.ylabel('Predicted Prices')
      plt.title('Actual vs Predicted Prices')
      plt.legend()
      plt.show()
```





```
[101]: import pandas as pd
       import pickle
       import numpy as np

       # Load the trained model (pipeline with preprocessor)
       with open('pipe.pkl', 'rb') as file:
           model = pickle.load(file)

       # Example test data for laptop price prediction
       test_data = pd.DataFrame({
           'Company': ['Asus'],
           'TypeName': ['Gaming'],
           'Ram': [16],
           'Weight': [2.5],
           'Touchscreen': [0],  # 1 if yes, 0 if no
           'Ips': [1],          # 1 if yes, 0 if no
           'ppi': [141.211998],      # Example value, compute based on resolution and screen size
           'Cpu brand': ['Intel Core i7'],
           'HDD': [0],
           'SSD': [0],
           'Gpu brand': ['Nvidia'],
           'os': ['Windows']
       })

       # Make predictions directly using the pipeline
       predicted_price = model.predict(test_data)

       # Assuming the target was log-transformed during training
       final_price = int(np.exp(predicted_price[0]))

       print(f'Predicted Price: {final_price}')

       Predicted Price: 60828
```