Laptop Price Prediction using Machine Learning

A Project Work-II Report

Submitted in partial fulfillment of requirement of the

Degree of

BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING

BY

Mitali Gupta (EN21CS301456)
Muskan Jain (EN21CS301486)

Under the Guidance of
Dr. Lokendra Singh

Department of Computer Science & Engineering
Faculty of Engineering
MEDICAPS UNIVERSITY, INDORE- 453331

Jan- June 2025

Report Approval

The project work **"Laptop Price Prediction using Machine Learning"** is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as prerequisite for the Degree for which it has been submitted.

It is to be understood that by this approval the undersigned do not endorse or approved any statement made, opinion expressed, or conclusion drawn there in; but approve the "Project Report" only for the purpose for which it has been submitted.

Internal Examiner

Name:

Designation

Affiliation

External Examiner

Name:

Designation

Affiliation

Declaration

We hereby declare that the project entitled **“Laptop Price Prediction using Machine Learning”** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in ‘Department of Computer Science and Engineering’ completed under the supervision of **Dr. Lokendra Singh, Associate Professor, Department of Computer Science and Engineering,** Faculty of Engineering, Medicaps University Indore is an authentic work.

Further, we declare that the content of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for the award of any degree or diploma.

Mitali Gupta

Muskan Jain

__/__/2025

<u>Certificate</u>

We, **Dr. Lokendra Singh** certify that the project entitled **“Laptop Price Prediction using Machine Learning”** submitted in partial fulfillment for the award of the degree of Bachelor of Technology by **Mitali Gupta (EN21CS301456), Muskan Jain (EN21CS301486)** is the record carried out by them under my guidance and that the work has not formed the basis of award of any other degree elsewhere.

Dr. Lokendra Singh

Department of Computer Science and Engineering

Medicaps University, Indore

Dr. Ratnesh Litoriya

Head of the Department

Computer Science & Engineering

Medicaps University, Indore

Acknowledgements

I would like to express my deepest gratitude to Honorable Chancellor, **Shri R C Mittal,** who has provided me with every facility to successfully carry out this project, and my profound indebtedness to **Prof. (Dr.) D K Patnaik,** Vice Chancellor, Medicaps University, whose unfailing support and enthusiasm has always boosted up my morale. I also thank **Prof. (Dr.) Pramod S Nair,** Dean, Faculty of Engineering, Medicaps University, for giving me a chance to work on this project. I would also like to thank my Head of the Department **Dr. Ratnesh Litoriya** for his continuous encouragement for the betterment of the project.

I express my heartfelt gratitude to my Internal Guide, **Dr. Lokendra Singh,** Associate Professor, Department of Computer Science and Engineering, MU, without whose continuous help and support, this project would ever have reached completion.

I would also like to thank my teammates who extended their kind support and help towards the completion of this project. It is their help and support, due to which we became able to complete the design and technical report.

Without their support this report would not have been possible.

Mitali Gupta (EN21CS301456)
Muskan Jain (EN21CS301486)
B.Tech. IV Year
Department of Computer Science & Engineering
Faculty of Engineering
Medicaps University, Indore

Abstract

The abstract provides a concise summary of the research, methodology, findings, and implications. Here's a breakdown of the Laptop Price Prediction using Machine Learning abstract:

- Purpose: The core aim of this project is to develop a machine learning-based system that predicts laptop prices based on key specifications such as brand, processor, RAM, storage, GPU, and display features. Accurate price prediction helps both consumers make informed decisions and retailers create competitive pricing strategies.

- Methodology: The project involves collecting a dataset of 1,300 laptops, cleaning and preprocessing the data , and transforming the features . Several regression-based machine learning models are trained and evaluated. Performance is measured using R^2 Score and Mean Absolute Error (MAE).

- Results: Among all tested models, Random Forest performed the best with an R^2 score of 0.90 and an MAE of 0.14, indicating high accuracy.

- Conclusion: The proposed system serves as a practical tool for both buyers and sellers, aiding in price estimation based on configuration. Future enhancements may include real-time market trend integration, price range prediction, and expansion to other electronic products.

- Contents

Table of Contents

List of Figures

Abbreviations

Abbreviation	Full Form
AI	Artificial Intelligence
KNN	K- Nearest Neighbour
CSV	Comma Separated Values
HDD	Hard Disk Drive
ML	Machine Learning
PKL	Pickle
SVM	Support Vector Machine
UI	User Interface
API	Application Programming Interface
IoT	Internet of Things
JSON	JavaScript Object Notation
SQL	Structured Query Language
DNN	Deep Neural Network

Chapter 1
Introduction

1.1 Introduction

In the modern era of digital transformation, laptops have become an essential part of both professional and personal life. With the vast and rapidly evolving landscape of laptop models, consumers often face the challenge of selecting the right device that fits their budget and requirements. Simultaneously, retailers and manufacturers are under pressure to price their products competitively while ensuring profitability. In such a dynamic and diverse market, predicting the fair price of a laptop based on its configuration and features has become increasingly complex.

The price of a laptop is influenced by numerous factors, including both tangible and intangible elements. Tangible specifications such as processor type (CPU), RAM size, storage capacity (HDD/SSD), display size and resolution, graphics card (GPU), presence of touchscreen or IPS panel, and the brand play a significant role in determining the cost. Intangible factors like brand reputation, market demand, and availability also contribute to price variation. Due to this wide range of variables and continuous technological advancements, traditional methods of manual comparison and static pricing often fall short.

Machine learning (ML) emerges as a powerful tool to address this pricing challenge. By analyzing historical data, ML models can learn complex relationships between laptop features and their market prices. This enables automated and highly accurate price prediction. The essence of machine learning lies in its ability to detect patterns, adapt to new data, and provide actionable insights without being explicitly programmed.

In this project, we have utilized a dataset consisting of 1,300 laptop records, which includes a diverse set of specifications. The data underwent thorough preprocessing, which involved removal of missing values, outlier detection, and elimination of duplicates to ensure quality and consistency. Exploratory Data Analysis (EDA) was performed to understand data distribution, identify key influencing factors, and visualize correlations.

The final model was integrated into a web-based application built using Streamlit, enabling users to input laptop specifications and receive an estimated price in real time. This system not only aids consumers in making cost-effective choices but also empowers retailers to adopt data-driven pricing strategies.

In conclusion, this project highlights the relevance and effectiveness of machine learning in solving real-world pricing challenges in the electronics market. It bridges the gap between technical specifications and price expectations, leading to more transparent and informed decision-making for both buyers and sellers.

1.2 Literature Review

Using XGBoost on 992 laptop records, Kafabihi (2024) demonstrated that price prediction is greatly impacted by RAM, processor brand, and storage kind. The study emphasized the significance of feature selection while confirming XGBoost's higher accuracy (R2: 0.84). Competitive pricing and consumer decision-making are facilitated by machine learning. Price prediction models for customers and producers are improved by using a variety of hardware features.[1]

Predicting laptop prices is a crucial task, particularly in cases where laptops are supplied straight from the manufacturer. Using supervised machine learning approaches, this forecast examines variables like a laptop's model, RAM, and CPU, frequently using multiple linear regression (R2: 0.81). Since anticipating a price class might often be more useful than guessing a specific price, recent research has also looked into predicting laptop price ranges.[2]

Using a dataset of 1320 samples, the study examines laptop price prediction using the Linear Regression, Random Forest, and XGBoost models. It determines that the XGBoost model has the best accuracy by comparing RMSE and R2 values (R2: 0.85, RMSE: 294.11). RAM, CPU, weight, and GPU are identified via feature importance analysis as the main determinants of price prediction. This study helps consumers make well-informed purchasing decisions and dealers plan competitive pricing.[3]

Previous research highlights how technology may increase system accuracy and efficiency, highlighting methods that lead to better results. Additionally, a number of research addresses the incorporation of contemporary methods like artificial intelligence and machine learning to automate and improve procedures. This study uses Gradient Boosting Regressor to predict the price of laptops with R2 score of 0.88 and MSE was 0.06.[4]

Several laptop price prediction solutions from Kaggle contests are included in the current system, which combines conventional machine learning techniques with novel concepts including neural networks, residual regression, and logit transform. However, because of the small dataset sizes, the findings of the laptop price fluctuation forecast are not always accurate and can have high standard deviations. By applying machine learning techniques to create a more precise laptop price prediction tool, this study seeks to address these problems. The methodology entails pre-processing the data, applying the Random Forest algorithm to train a model, and evaluating the accuracy of the model.[5]

Regression-based techniques are used in the majority of current price prediction studies to estimate a certain price value. However, predicting a pricing range is far more practical for many real-world applications. There is just one study on laptop price range prediction in the literature, despite the fact that there are numerous studies on laptop price prediction. Furthermore, the laptop price range prediction problem has seen very little testing of machine learning techniques. A dataset that was originally used for laptop price prediction was modified to be used for laptop price range prediction. Preprocessing techniques like data cleaning, feature engineering, and label encoding were used to optimize the dataset for laptop price range prediction giving R2 score of 0.70.[6]

1.3 Project Objectives

The primary aim of this project is to design and implement a robust machine learning-based system capable of accurately predicting the price of a laptop based on its technical specifications and key features. The following specific objectives outline the goals and deliverables of this project:

1. **Development of an Accurate Predictive Model**
   To build a reliable and efficient machine learning model that can estimate the price of a laptop with high accuracy. This model will utilize various input parameters such as processor type, RAM capacity, storage size, brand, screen size, and other relevant features to generate a price prediction that aligns closely with market trends.

2. **Data Collection and Preprocessing**
   To gather a comprehensive dataset from credible sources that includes a diverse range of laptop specifications and their corresponding prices. The raw data will undergo thorough preprocessing, including handling missing values, encoding categorical variables, feature selection, and normalization, to ensure it is clean and suitable for model training and testing.

3. **Exploration and Evaluation of Regression Algorithms**
   To investigate and implement various machine learning regression techniques such as Linear Regression, Decision Tree Regression, Random Forest, Gradient Boosting, and XGBoost. These algorithms will be evaluated based on performance metrics like R^2 score, Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) to determine the most effective model.

4. **Design of an Interactive User Interface**
   To develop a simple and intuitive front-end interface, possibly using web frameworks like Streamlit or Flask, where users can input laptop specifications manually. The system will process these inputs and return an estimated price in real-time, enhancing the accessibility and usability of the model for general users.

5. **Decision-Making Support for Customers and Retailers**
   To provide valuable insights to both consumers and retailers. Customers can use this tool to estimate fair prices before making purchases, while retailers can leverage the model for competitive pricing strategies, inventory planning, and customer targeting.

6. **Analysis of Market Trends for Enhanced Insights**
   To utilize the dataset to uncover patterns and trends in the laptop market, such as the influence of specific features on price changes over time. This analysis will not only aid in improving model predictions but also support strategic decision-making based on current and emerging market dynamics.

1.4 Scope of the Report

This project focuses on building a machine learning model to predict the price of laptops based on various features such as brand, processor, RAM, storage type, screen size, and operating system. The scope includes data collection, data cleaning, feature selection, model training, testing, and evaluation. The report also covers the comparison of different regression algorithms to determine the most effective model. This system can be useful for customers, retailers, and e-commerce platforms to estimate laptop prices and make informed decisions. However, the scope is limited to the available dataset and selected features, and may not cover all the external factors that affect pricing, such as market trends or seasonal discounts.

1.5 Research Design

The research design for the **Laptop Price Predictor** aims to develop a comprehensive system that utilizes a structured dataset to predict the price of laptops based on their specifications and features. The design involves several key components, including data collection, model development, evaluation metrics, and deployment strategy. Each phase of the research is carefully structured to ensure that the final system is robust, accurate, and user-friendly.

1. **Problem Definition and Objective:**
   The core objective of this research is to create a laptop price prediction system that analyzes various technical specifications—such as brand, processor type, RAM, storage, screen size, and operating system—to estimate the price of a laptop. The primary goal is to build predictive models, assess their accuracy, and validate their effectiveness using real-world data.

2. **Data Collection:**
   Data is collected from a variety of sources to ensure comprehensive and diverse input for training the models. Data preprocessing techniques will be employed to clean and normalize the data, ensuring that the models receive high-quality, consistent input. This will include handling missing values, scaling numerical data, and encoding categorical variables.
   - **Brand, Processor, RAM, Storage, Screen Size, and Operating System:** Data includes specifications and corresponding prices scraped from online platforms like Amazon, Flipkart, and Kaggle datasets.
     Data preprocessing techniques will be employed to clean and normalize the data, ensuring that the models receive high-quality, consistent input. This will include handling missing values, scaling numerical data, and encoding categorical variables.

3. **Model Development:**
   The next step involves selecting appropriate machine learning models for price prediction. These models include:

- **Support Vector Machine (SVM):** Used for regression tasks, particularly when data is high-dimensional.

- **Linear Regression:** A traditional model for predicting continuous variables, useful for its simplicity and interpretability.

- **K-Nearest Neighbor (KNN):** A non-parametric model used for predicting based on the similarity between data points.

- **Random Forest:** An ensemble method combining multiple decision trees to improve prediction accuracy.

Each model will be trained using the relevant dataset and evaluated using metrics like accuracy, Mean Squared Error (MSE), and R^2 score. The nest results were given by Random Forest algorithm.

4. Deployment and User Interface

Once the models are trained and validated, the final system will be deployed using a web-based platform built with tools like Streamlit. This allows users to input laptop specifications and receive real-time price predictions. The user interface will be designed to be intuitive, where users can enter details such as brand, processor type, RAM size, storage capacity, screen size, and operating system to get an estimated price.

The system will also integrate a backend that loads the pre-trained machine learning models stored in Pickle files. This will ensure efficient, real-time predictions without the need to retrain the models each time a user inputs data.

Chapter 2
Requirements Specification

2.1 User Characteristics

The primary users of the laptop price prediction system can be categorized into different groups.

1. **General Customers**: Users who wish to buy or sell laptops can use the system to estimate the fair market price based on specifications. They may not have technical knowledge, so the interface should be simple and intuitive.

2. **Retailers and E-commerce Platforms**: These users can leverage the system for setting competitive prices, managing inventory, and analyzing trends in customer preferences.

3. **Data Analysts and Researchers**: Individuals interested in analyzing pricing models, feature importance, and applying data science techniques for improving predictions.

4. **Developers and System Administrators**: Technical users who manage the backend, train models, update datasets, and ensure the system is running efficiently.

Most users are expected to have basic digital literacy, including the ability to use computers, fill out forms, and interpret simple visual outputs.

2.2 Functional Requirements

1. **User Input Interface**
   The system must provide a user-friendly front-end interface that enables users to manually input various laptop specifications. These include brand (e.g., Dell, HP, Lenovo), processor type (e.g., Intel i5, AMD Ryzen 7), RAM size (in GB), storage type and capacity (e.g., 512GB SSD, 1TB HDD), screen size (in inches), presence or absence of a dedicated graphics card, and operating system (e.g., Windows, Linux, macOS). This interface should validate input fields and ensure ease of use across all devices.

2. **Data Preprocessing**
   Before the input data is passed to the machine learning model, it must undergo preprocessing. This includes cleaning missing or null values, treating outliers to avoid skewed predictions, and converting categorical values (like brand and processor) into numerical formats through encoding techniques such as label encoding or one-hot encoding. The preprocessing should be efficient and executed automatically upon user input submission.

3. **Model Prediction**
   The system should use a pre-trained machine learning model to process the user's input and generate a predicted price for the laptop. This prediction should be calculated in real-time with minimal latency. The model should be capable of interpreting different combinations of features and returning a price that reflects current market trends.

4. **Model Training Module**
   The backend should include a model training module that enables administrators or developers to retrain the model with updated datasets. This allows the system to stay relevant with changing market conditions and newly released laptop configurations. The retraining process should support tasks like model validation, evaluation, and saving updated models using formats such as .pkl (Pickle).

5. **Result Display**
   The predicted laptop price should be displayed in a clear, understandable, and visually appealing format. The system may also include additional visualizations, such as bar charts or price range comparisons, to show how the predicted price compares to average prices for similar configurations in the market.

6. **Error Handling**
   The system should be able to detect and respond to incomplete or invalid input data. For instance, if a user leaves out a required field or selects an unsupported value, the system must provide meaningful error messages or prompts to help the user correct the input. This ensures smooth and uninterrupted operation of the application.

2.3 Dependencies

The Laptop Price Predictor relies on the following external systems and resources:

1. **Machine Learning Libraries**:
   - **Scikit-learn**: A Python library for machine learning, used to implement algorithms like Support Vector Machine (SVM), Logistic Regression, K-Nearest Neighbor (KNN), and Random Forest.
2. **Data Storage**:
   - **SQL/NoSQL Database**: Used to store user inputs, health data, and predictions. It should support secure and efficient data retrieval.
   - **Cloud Storage**: To store large datasets, model files (in **Pickle** or **H5** formats), and processed health reports.
3. **Web Framework and Deployment**:
   - **Streamlit**: A Python framework for building the web application, allowing easy integration of machine learning models into a web-based interface.

2.4 Non-Functional Requirements

1. **Performance**
   The system must be capable of delivering real-time price predictions with minimal processing delay. Especially for lightweight inputs—such as typical laptop configurations—the response time should be within a few seconds. This ensures a smooth user experience and keeps the application efficient even on systems with limited computational resources.

2. **Usability**
   The application must offer a clean, visually appealing, and easy-to-navigate user interface. Both technical users (e.g., developers, analysts) and non-technical users (e.g., customers or retailers) should find it intuitive. Clear instructions, tooltips, input validation, and responsive design should be incorporated to enhance accessibility and user satisfaction.

3. **Reliability**
   The system must be stable and capable of functioning consistently without unexpected crashes, hangs, or bugs. It should handle edge cases and unusual inputs gracefully, and the prediction engine must return valid results under a wide range of configurations. Logging and monitoring tools should be used to detect and address issues promptly.

4. **Maintainability**
   The codebase should follow clean coding practices, be modular in structure, and include proper documentation for functions, modules, and logic. This ensures that future updates—such as the integration of new datasets, enhancement of machine learning models, or UI redesign—can be implemented smoothly by developers without introducing technical debt.

5. **Portability**
   The application should be designed for easy deployment across various platforms. It should be functional in different environments including local development systems, cloud-based platforms like Heroku or AWS, and as a standalone desktop application if required. The use of standard frameworks and minimal environment dependencies will support cross-platform compatibility.

6. **Security**
   If the system handles user data—especially in cases where login features or usage history tracking is implemented—it must ensure data protection through appropriate authentication mechanisms. Encryption of sensitive data, secure API communication (HTTPS), and basic data privacy principles must be applied to prevent data breaches or misuse.

7. **Scalability**
   The system should be scalable to accommodate growth in data volume or user traffic. As the application gains more users or incorporates larger datasets for training and prediction,

it should maintain its performance and responsiveness. Scalable design principles like modular architecture, use of cloud infrastructure, and database optimization should be applied to support future expansion.

2.5 Hardware Requirements

- **RAM**:
  A minimum of **4 GB of RAM** is required for optimal performance. This is sufficient to handle the basic operations of the system, such as data preprocessing, running machine learning models, and providing real-time price predictions without significant delays. For larger datasets or more complex operations, upgrading to 8 GB or higher would further enhance performance.

- **Processor**:
  The system requires at least an **Intel Core i3 processor** or an equivalent model, such as AMD Ryzen 3, to ensure smooth operation. This processor should be capable of running the application's machine learning algorithms and web interface efficiently. Higher-end processors (Intel i5/i7 or Ryzen 5/7) are recommended for handling more intensive computations or large datasets.

- **Storage**:
  The system should have **250 GB of storage**, either through a **HDD** (Hard Disk Drive) or **SSD** (Solid State Drive). An SSD is preferred due to its faster read/write speeds, which will improve overall system performance, especially during data loading, model training, and handling multiple requests. The storage capacity should be sufficient to store datasets, machine learning models, and application files without running into space limitations.

- **Operating System**:
  The application is designed to be cross-platform, compatible with multiple operating systems including **Windows 10**, **macOS**, or **Linux**. Users can run the system on any of these platforms, ensuring accessibility across a wide range of devices. For optimal compatibility, the latest updates and patches should be installed on the operating system.

- **Display**:
  The system should be capable of running on a display with at least **720p resolution** (HD), which ensures a clear and readable user interface. However, higher resolutions like 1080p or higher are recommended for users who want better visual clarity, especially when working with charts, graphs, or detailed data visualizations in the user interface.

- **Internet**:
  A **stable internet connection** is required for downloading necessary libraries, packages, and software dependencies during the initial setup. It is also necessary for uploading data, accessing external databases, or using cloud-based platforms for deployment or model training.

Chapter 3
Design

The design of the Laptop Price Prediction system is structured to ensure modularity, efficiency, and ease of use. The system is divided into several components that work together to achieve accurate price predictions based on user inputs. The goal is to design a system that is scalable, user-friendly, and capable of producing reliable results with minimal input. The design is based on a layered architecture that separates concerns and simplifies development and maintenance.

3.1 Algorithm

The core of the Laptop Price Predictor is built around machine learning algorithms designed to accurately predict the price of a laptop based on its specifications. The selected algorithms include Support Vector Machine (SVM), Linear Regression, K-Nearest Neighbor (KNN), and Random Forest. Each of these models is chosen for its ability to handle diverse features and provide accurate price predictions based on the input laptop configuration.

The algorithm works by first collecting input from users, such as brand, processor type, RAM size, storage capacity, screen size, graphics card, and operating system. These inputs are processed and passed through a series of preprocessing steps, such as data cleaning, normalization, and feature selection. Missing values, outliers, or inconsistent data are handled appropriately to ensure that the models function with high accuracy.

Once the data is preprocessed, it is used to train the machine learning models. For each algorithm, the model is trained using relevant data specific to laptop configurations and their corresponding prices. For example, the model might use features like processor type, RAM, storage, and brand to predict the price. The dataset will include various laptop configurations, and the model will learn patterns in the data that link these configurations to the price.

The algorithm relies on an iterative training process, where the model's parameters are adjusted to minimize prediction errors and improve accuracy. After training, the models are validated using a separate test dataset to assess their generalization ability and ensure they are not overfitted to the training data. Once validated, the model is ready to predict the price of new laptops based on the specifications provided by the user.

3.2 Function- Oriented Design

The machine learning component includes the following stages:

- Data Collection: Data was collected from reliable online sources, including e-commerce websites and open-source datasets.The dataset contains detailed specifications of laptops such as brand, processor, RAM, storage, GPU, and screen resolution.It also includes the corresponding market prices for accurate prediction.A total of 1,300 laptop entries were gathered for model training and testing.The data was checked for completeness and relevance before use.This dataset serves as the foundation for building the prediction model..

- Data Preprocessing: Before training the machine learning model, the dataset undergoes thorough preprocessing to ensure the quality and consistency of the data. The raw dataset, collected from reliable online sources, often contains missing values, duplicates, and inconsistencies that can affect model performance. These issues are handled by removing duplicate records, filling or dropping missing values, and standardizing categorical fields such as brand names, processor types, and operating systems. This preprocessing stage plays a crucial role in improving the accuracy and efficiency of the model.

- Feature Selection: Features that strongly influence laptop prices are selected to optimize the model. These include brand, RAM, processor, SSD, HDD, touchscreen, and GPU. Feature selection helps in reducing model complexity and improving accuracy.

- Model Training: Algorithms like Linear Regression, Decision Tree Regression, and Random Forest Regression are trained on the datasetThe cleaned data is split into training and testing sets.Various algorithms like Linear Regression, Random Forest, and XGBoost are trained.Each model is evaluated using R^2, MAE, and MSE metrics.Random Forest performs the best with the highest accuracy and lowest error.The final model is saved and integrated into a web app for real-time predictions.

- Model Evaluation: after training the models, their performance was assessed using evaluation metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared (R^2) value. MAE measures the average magnitude of the errors between predicted and actual prices, while MSE penalizes larger errors more heavily. The R^2 score indicates how well the model explains the variation in laptop prices. These metrics are essential for comparing the effectiveness of different algorithms. Based on the results, the Random Forest model showed the best performance with the highest R^2 value and the lowest error scores, making it the most reliable model for deployment.

- Model Deployment: Once the best model is identified, it is saved using a serialization technique such as joblib or pickle. A web application is built using Streamlit and Python to provide a user-friendly interface for predictions. The model is integrated into the backend and deployed on a cloud platform like Heroku for easy public access. The deployment allows real-time predictions and can be scaled as needed.

- Security and Future Design Considerations**:** The system can be further enhanced by incorporating security features such as input validation, protection against code injection, and encrypted API communication. It is also designed to be modular so that new models or features can be added in the future without major changes to the architecture.

3.3 System Design

3.3.1 Use Case Diagram

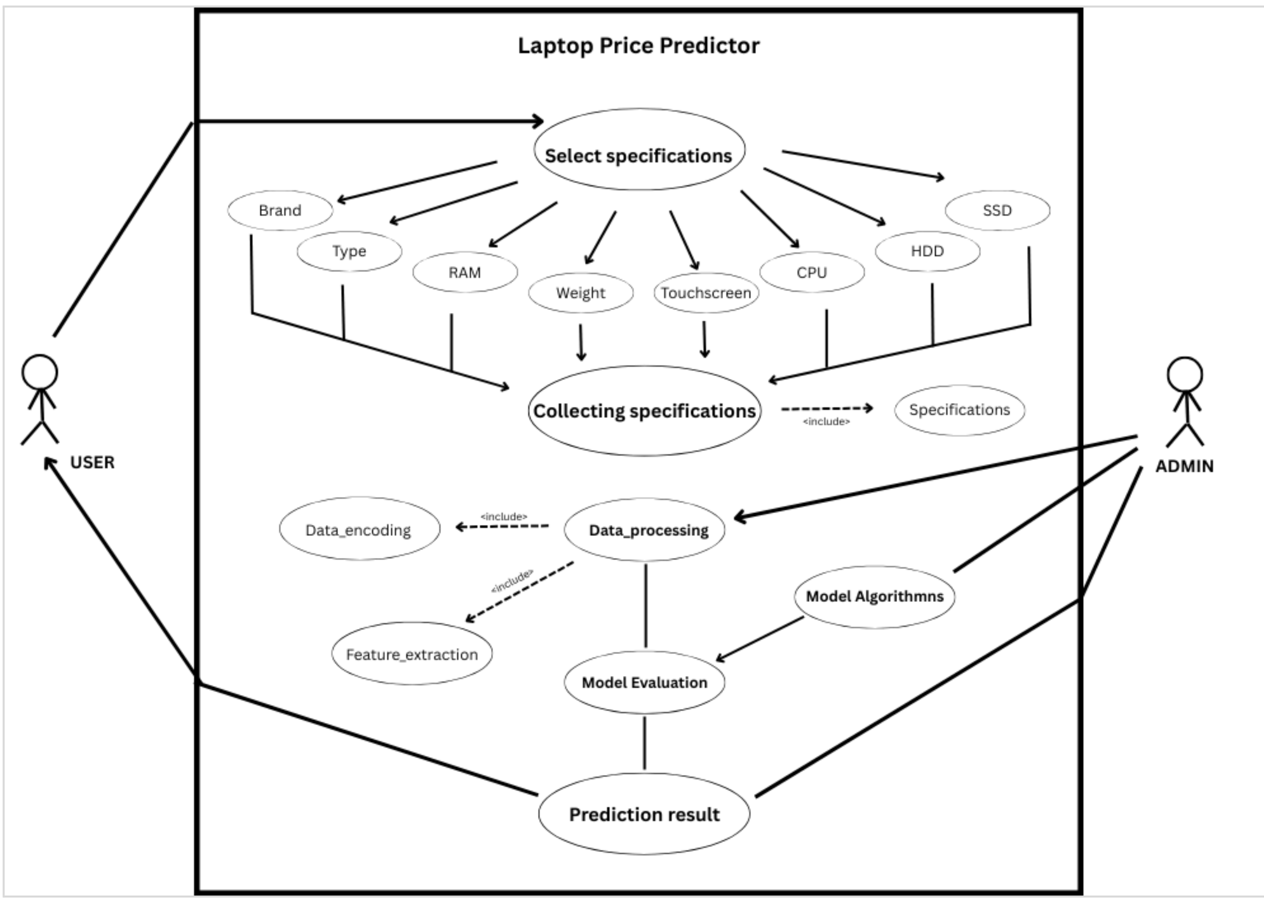

Figure 3.1 Use Case diagram

3.3.2 Flow Chart

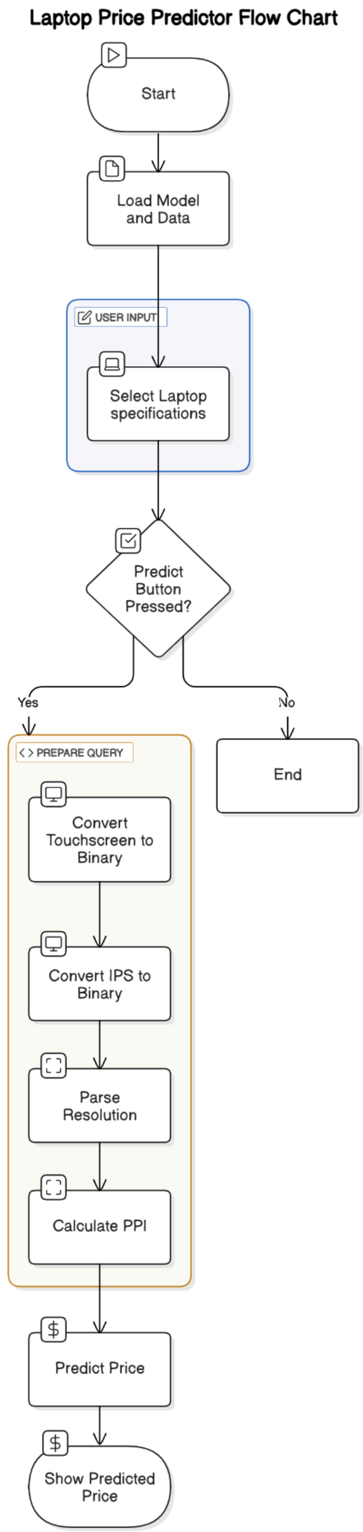

Figure 3.2 Flow diagram

3.3.3 Activity Diagram

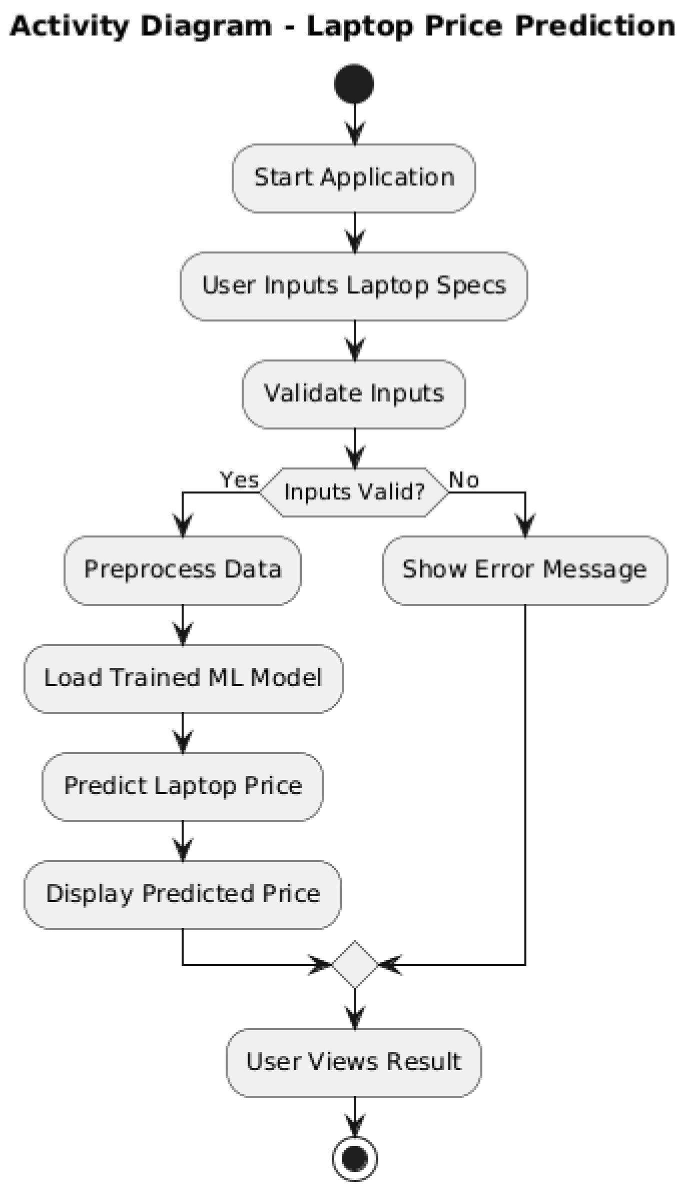

Figure 3.2 Activity Diagram

Chapter 4
Implementation, Testing and Maintenance

4.1 Introduction to Languages, IDEs, Tools and Technologies Used for Implementation

- **Machine Learning Models**
    a. Support Vector Machine (SVM):
       Support Vector Machine (SVM) is a robust supervised learning algorithm used for classification and regression tasks. SVM is highly effective for problems with high-dimensional data and can be used to understand complex, non-linear relationships between laptop features (such as processor type, RAM, storage) and their prices. It is particularly useful for high-dimensional, sparse datasets.
    b. Linear Regression: Linear Regression is a statistical model used for predicting continuous outcomes based on one or more predictor variables. In the context of laptop price prediction, linear regression estimates the relationship between laptop features and its price
    c. K-Nearest Neighbor (KNN): It is a simple yet effective algorithm that classifies data points by looking at the closest training samples in the feature space. It is a non-parametric and instance-based learning algorithm, which makes predictions based on the proximity of input features to the stored instances.
    d. Random Forest:It's an ensemble learning method that creates multiple decision trees during training and combines their results to improve model accuracy and reduce overfitting. It is a versatile algorithm that can handle both classification and regression tasks. Random Forest is particularly valuable in the laptop price prediction task due to its ability to handle complex interactions between features (e.g., processor, RAM, brand) and its robustness to noisy data.

- **Technology Stack**
    1. Streamlit: Streamlit is an open-source Python library that facilitates the rapid development of interactive web applications. It is used to create user-friendly interfaces where users can input laptop specifications and receive predicted prices in real-time. Streamlit makes it easy to deploy machine learning models and share results seamlessly with users through dynamic and visually appealing applications.

    2. HTML (HyperText Markup Language): HTML is the standard language used to create webpages and structure content on the web. It is essential for defining the layout, design, and elements such as text, buttons, and forms that allow users to interact with the laptop price prediction tool.

3. PKL (Pickle) File: Pickle is a Python library used for serializing and deserializing Python objects, including machine learning models. It enables the storage of trained models as files, which can later be loaded back into memory for predictions without needing to retrain the model. This makes the system efficient and fast, as it allows for real-time price predictions on new laptop configurations.
4. CSV (Comma Separated Values) File: CSV is a popular file format used to store tabular data. Each row in a CSV file represents a record, and each column represents an attribute or feature. In the laptop price prediction system, CSV files are used to store the dataset of laptop features and their corresponding prices, which are used for training the machine learning models.

4.2 Implementation

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

from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.15,random_state=2)
X_train

import sys
!{sys.executable} -m pip install xgboost==1.3.3

from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder
from sklearn.metrics import r2_score,mean_absolute_error
from sklearn.linear_model import LinearRegression,Ridge,Lasso
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor,
AdaBoostRegressor, ExtraTrees Regressor
from sklearn.svm import SVR
from xgboost import XGBRegressor

step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse_output=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step2 = LinearRegression()

pipe = Pipeline([
    ('step1',step1),
    ('step2',step2)
```

```
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))

step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse_output=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step2 = SVR(kernel='rbf',C=10000,epsilon=0.1)

pipe = Pipeline([
    ('step1',step1),
    ('step2',step2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))

step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse_output=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step2 = RandomForestRegressor(n_estimators=290,
                              random_state=3,
                              max_samples=0.85,
                              max_features=0.35,
                              max_depth=30)

pipe = Pipeline([
    ('step1',step1),
    ('step2',step2)
])

pipe.fit(X_train,y_train)
```

```
y_pred = pipe.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))

import pickle

pickle.dump(df,open('df.pkl','wb'))
pickle.dump(pipe,open('pipe.pkl','wb'))

import streamlit as st
import pickle
import numpy as np
import pandas as pd


# import the model
pipe = pickle.load(open('pipe.pkl','rb'))
df = pickle.load(open('df.pkl','rb'))

st.title("Laptop Price Predictor")

# brand
company = st.selectbox('Brand',df['Company'].unique())

# type of laptop
type = st.selectbox('Type',df['TypeName'].unique())

# Ram
ram = st.selectbox('RAM(in GB)',[2,4,6,8,12,16,24,32,64])

# weight
weight = st.number_input('Weight of the Laptop')

# Touchscreen
touchscreen = st.selectbox('Touchscreen',['No','Yes'])

# IPS
ips = st.selectbox('IPS',['No','Yes'])

# screen size
screen_size = st.slider('Scrensize in inches', 10.0, 18.0, 13.0)

# resolution
```

```
resolution = st.selectbox('Screen Resolution',['1920x1080','1366x768','1600x900','3840x2160','3200x1800','2880x1800','2560x1600','2560x1440','2304x1440'])

#cpu
cpu = st.selectbox('CPU',df['Cpu brand'].unique())

hdd = st.selectbox('HDD(in GB)',[0,128,256,512,1024,2048])

ssd = st.selectbox('SSD(in GB)',[0,8,128,256,512,1024])

gpu = st.selectbox('GPU',df['Gpu brand'].unique())

os = st.selectbox('OS',df['os'].unique())

if st.button('Predict Price'):
    # query
    ppi = None
    if touchscreen == 'Yes':
        touchscreen = 1
    else:
        touchscreen = 0

    if ips == 'Yes':
        ips = 1
    else:
        ips = 0

    X_res = int(resolution.split('x')[0])
    Y_res = int(resolution.split('x')[1])
    ppi = ((X_res ** 2) + (Y_res ** 2)) ** 0.5 / screen_size
    query = np.array([company, type, ram, weight, touchscreen, ips, ppi, cpu, hdd, ssd, gpu, os])

    #query=np.array(query,dtype=object)
    query = query.reshape(1, 12)
    st.title("The  estimated price of your laptop -> " + str(int(np.exp(pipe.predict(query)[0])))+"/-")
```

4.3 Testing and Techniques

Linear regression

[161]:

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse_output=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step2 = LinearRegression()

pipe = Pipeline([
    ('step1',step1),
    ('step2',step2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2 score 0.8072993943141318
MAE 0.21019955789651018
```

Figure 4.3 Linear Regression

SVM

[166]:

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse_output=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step2 = SVR(kernel='rbf',C=10000,epsilon=0.1)

pipe = Pipeline([
    ('step1',step1),
    ('step2',step2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2 score 0.8083317557480397
MAE 0.202303601210246
```

Figure 4.4 Support Vector Machine

Decision Tree

[165]:

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse_output=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step2 = DecisionTreeRegressor(max_depth=8)

pipe = Pipeline([
    ('step1',step1),
    ('step2',step2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2 score 0.8276821607938187
MAE 0.18735161150763366
```

Figure 4.5 Decision Tree

Random Forest

[167]:

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse_output=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step2 = RandomForestRegressor(n_estimators=290,
                              random_state=3,
                              max_samples=0.85,
                              max_features=0.35,
                              max_depth=30)

pipe = Pipeline([
    ('step1',step1),
    ('step2',step2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2 score 0.901127388722254
MAE 0.14983570578931865
```

Figure 4.6 Random Forest

Chapter 5

Results and Discussions

The primary objective of this study was to develop an accurate model for predicting laptop prices based on various specifications, and to evaluate the performance of multiple machine learning algorithms in achieving this goal. The models tested included Linear Regression, Ridge Regression, Decision Tree, Random Forest, XGBoost, Gradient Boosting, Support Vector Machine (SVM), and K-Nearest Neighbors (KNN). The evaluation metrics used for model comparison were R^2 score and Mean Absolute Error (MAE).

The results from the evaluation showed that the Random Forest model was the most effective in predicting laptop prices. It achieved an R^2 score of 0.90 and an MAE of 0.14, making it the highest-performing model compared to the others. XGBoost and Gradient Boosting followed closely, with R^2 scores of 0.87 and 0.88, respectively. On the other hand, models like Linear Regression and Ridge Regression produced lower R^2 scores of 0.80 and 0.81, indicating their relative ineffectiveness for this particular task.

In addition to model performance, an essential aspect of this project was the exploration of feature importance. The analysis indicated that RAM, CPU type, GPU, and storage type (SSD vs. HDD) were the most significant factors influencing laptop prices. Larger RAM sizes and more powerful CPUs, such as Intel Core i7 or i9, were consistently linked to higher prices. Similarly, laptops with dedicated graphics cards and Solid-State Drives were also priced higher, reflecting their greater value in gaming, graphic design, and performance-intensive tasks.

To provide a user-friendly solution, a web application was built using Streamlit, which allows users to input laptop specifications and receive real-time price predictions. The application was designed to be simple and intuitive, making it accessible to both consumers and retailers. The model, powered by Random Forest, was deployed on Heroku, ensuring that users could easily access the tool from anywhere. This real-time prediction system has proven to be beneficial for consumers looking to make informed purchasing decisions and for retailers optimizing their pricing strategies.

Despite the strong performance of the Random Forest model, there are limitations to consider. One of the main limitations is that the model is only as good as the data it is trained on. Therefore, keeping the dataset up to date is crucial to maintaining the model's accuracy. Future iterations of this system could integrate real-time data from e-commerce platforms to further improve its relevance.

Overall, the project successfully demonstrated the power of machine learning in addressing the challenge of laptop price prediction. The Random Forest model provided the best results, and the development of a web-based application made the solution both practical and accessible. Moving forward, the accuracy and functionality of the system can be enhanced by incorporating more detailed features, real-time data, and possibly expanding the model to predict prices for other electronics.

5.1 User interface Representation

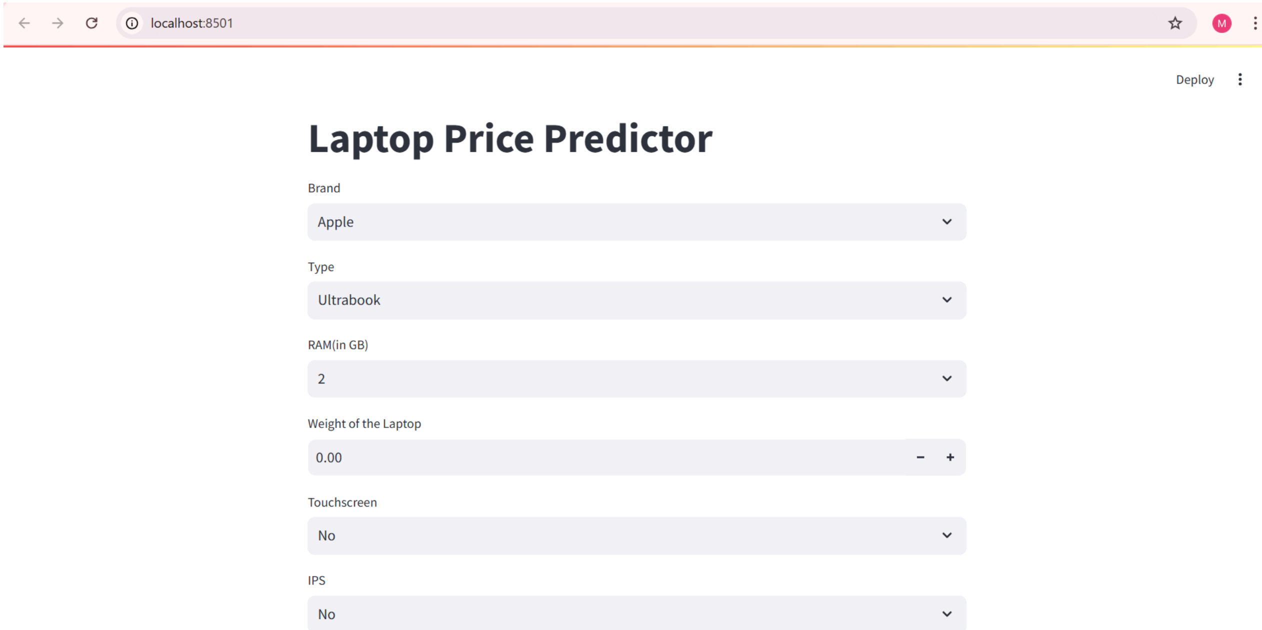

Figure 5.1 User Interface

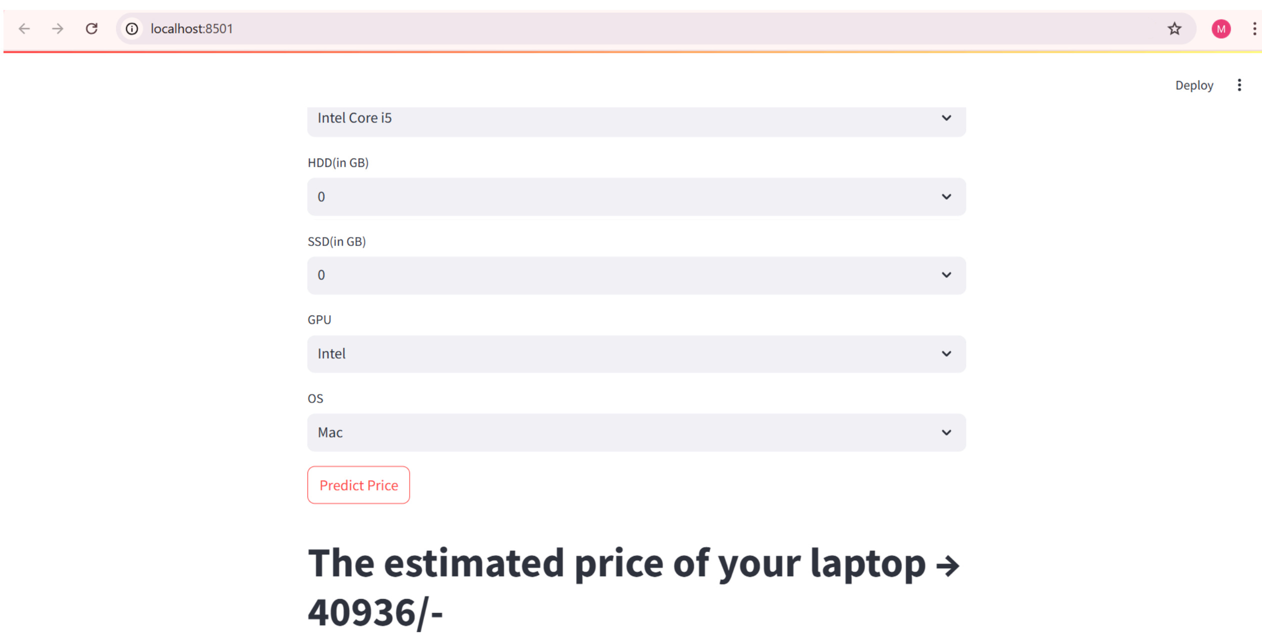

Figure 5.2 User Interface

5.2 Back ends Representation

```
import streamlit as st

import pickle

import numpy as np

import pandas as pd


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

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

st.title("Laptop Price Predictor")


company = st.selectbox('Brand',df['Company'].unique())     # brand

type = st.selectbox('Type',df['TypeName'].unique())   # type of laptop

ram = st.selectbox('RAM(in GB)',[2,4,6,8,12,16,24,32,64]) # Ram

weight = st.number_input('Weight of the Laptop')  # weight

touchscreen = st.selectbox('Touchscreen',['No','Yes'])  # Touchscreen

ips = st.selectbox('IPS',['No','Yes']) # IPS

screen_size = st.slider('Scrensize in inches', 10.0, 18.0, 13.0)  # screen size

resolution = st.selectbox('Screen
Resolution',['1920x1080','1366x768','1600x900','3840x2160','3200x1800','2880x1800','2560x1600'
,'2560x1440','2304x1440'])   # resolution

cpu = st.selectbox('CPU',df['Cpu brand'].unique())   #cpu

hdd = st.selectbox('HDD(in GB)',[0,128,256,512,1024,2048])

ssd = st.selectbox('SSD(in GB)',[0,8,128,256,512,1024])

gpu = st.selectbox('GPU',df['Gpu brand'].unique())

os = st.selectbox('OS',df['os'].unique())

if st.button('Predict Price'):
```

```
ppi = None

if touchscreen == 'Yes':

    touchscreen = 1

else:

    touchscreen = 0

if ips == 'Yes':

    ips = 1

else:

    ips = 0


X_res = int(resolution.split('x')[0])

Y_res = int(resolution.split('x')[1])

ppi = ((X_res ** 2) + (Y_res ** 2)) ** 0.5 / screen_size

query = np.array([company, type, ram, weight, touchscreen, ips, ppi, cpu, hdd, ssd, gpu, os])


query=np.array(query,dtype=object)

query = query.reshape(1, 12)

st.title("The  estimated price of your laptop -> " + str(int(np.exp(pipe.predict(query)[0])))+"/-")
```

5.3 Snapshots of CSV file

jupyter laptop_data.csv Last Checkpoint: last month

File Edit View Settings Help

Delimiter: ,

		Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Me
1	0	Apple	Ultrabook	13.3	Retina Display 2560x1600	Intel Core i5 2.3GHz	8GB	128GB
2	1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8GB	128GB Flash Sto
3	2	HP	Notebook	15.6	Full HD 1920x1080	tel Core i5 7200U 2.5GHz	8GB	256GB
4	3	Apple	Ultrabook	15.4	Retina Display 2880x1800	Intel Core i7 2.7GHz	16GB	512GB
5	4	Apple	Ultrabook	13.3	Retina Display 2560x1600	Intel Core i5 3.1GHz	8GB	256GB
6	5	Acer	Notebook	15.6	1366x768	MD A9-Series 9420 3GHz	4GB	500GB
7	6	Apple	Ultrabook	15.4	Retina Display 2880x1800	Intel Core i7 2.2GHz	16GB	256GB Flash Sto
8	7	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8GB	256GB Flash Sto
9	8	Asus	Ultrabook	14.0	Full HD 1920x1080	tel Core i7 8550U 1.8GHz	16GB	512GB
10	9	Acer	Ultrabook	14.0	Panel Full HD 1920x1080	tel Core i5 8250U 1.6GHz	8GB	256GB
11	10	HP	Notebook	15.6	1366x768	tel Core i5 7200U 2.5GHz	4GB	500GB
12	11	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i3 6006U 2GHz	4GB	500GB
13	12	Apple	Ultrabook	15.4	Retina Display 2880x1800	Intel Core i7 2.8GHz	16GB	256GB
14	13	Dell	Notebook	15.6	Full HD 1920x1080	Intel Core i3 6006U 2GHz	4GB	256GB
15	14	Apple	Ultrabook	12.0	Retina Display 2304x1440	Intel Core M m3 1.2GHz	8GB	256GB
16	15	Apple	Ultrabook	13.3	Retina Display 2560x1600	Intel Core i5 2.3GHz	8GB	256GB
17	16	Dell	Notebook	15.6	Full HD 1920x1080	tel Core i7 7500U 2.7GHz	8GB	256GB
18	17	Apple	Ultrabook	15.4	Retina Display 2880x1800	Intel Core i7 2.9GHz	16GB	512GB
19	18	Lenovo	Notebook	15.6	Full HD 1920x1080	tel Core i3 7100U 2.4GHz	8GB	1TB

Figure 5.3 CSV Snapshot

92	91	Acer	Notebook	15.6	Full HD 1920x1080	tel Core i7 8550U 1.8GHz	8GB	1TB
93	92	HP	Notebook	13.3	Full HD 1920x1080	tel Core i7 8550U 1.8GHz	16GB	512GB
94	93	Dell	Gaming	15.6	Full HD 1920x1080	l Core i5 7300HQ 2.5GHz	8GB	256GB
95	94	Asus	Ultrabook	14.0	Full HD 1920x1080	tel Core i7 7500U 2.7GHz	8GB	256GB
96	95	Acer	2 in 1 Convertible	13.3	/ Touchscreen 1920x1080	tel Core i5 8250U 1.6GHz	8GB	256GB
97	96	Dell	Notebook	15.6	Full HD 1920x1080	tel Core i7 7500U 2.7GHz	8GB	1TB
98	97	Dell	Notebook	15.6	Full HD 1920x1080	Intel Core i3 6006U 2GHz	4GB	256GB
99	98	Asus	Notebook	15.6	Full HD 1920x1080	tel Core i3 7100U 2.4GHz	6GB	256GB
100	99	HP	Gaming	15.6	Panel Full HD 1920x1080	l Core i7 7700HQ 2.8GHz	12GB	128GB SSD + 1TB
101	100	HP	Notebook	15.6	Full HD 1920x1080	tel Core i7 7500U 2.7GHz	8GB	256GB
102	101	HP	Notebook	15.6	Full HD 1920x1080	-Series E2-9000e 1.5GHz	4GB	500GB
103	102	Dell	Notebook	15.6	Full HD 1920x1080	tel Core i5 8250U 1.6GHz	8GB	1TB
104	103	HP	Ultrabook	13.3	Panel Full HD 1920x1080	tel Core i7 7500U 2.7GHz	8GB	256GB
105	104	Microsoft	Ultrabook	13.5	Touchscreen 2256x1504	tel Core i5 7200U 2.5GHz	8GB	256GB
106	105	HP	Notebook	14.0	Panel Full HD 1920x1080	tel Core i5 7200U 2.5GHz	6GB	256GB
107	106	Lenovo	Notebook	15.6	Full HD 1920x1080	Intel Core i3 6006U 2GHz	4GB	1TB
108	107	Asus	Ultrabook	14.0	Panel Full HD 1920x1080	tel Core i7 7500U 2.7GHz	8GB	256GB
109	108	MSI	Gaming	15.6	Full HD 1920x1080	l Core i7 7700HQ 2.8GHz	16GB	256GB SSD + 1TB
110	109	Lenovo	Notebook	14.0	Panel Full HD 1920x1080	tel Core i7 7500U 2.7GHz	8GB	256GB
111	110	Asus	Ultrabook	15.6	Full HD 1920x1080	tel Core i7 8550U 1.8GHz	16GB	256GB SSD + 1TB

Figure 5.4 CSV Snapshot

		Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Me
154	153	MSI	Gaming	17.3	Full HD 1920x1080	l Core i7 7700HQ 2.8GHz	16GB	256GB SSD + 1TB
155	154	HP	Ultrabook	14.0	Full HD 1920x1080	tel Core i5 7500U 2.7GHz	4GB	256GB
156	155	HP	Notebook	15.6	Full HD 1920x1080	tel Core i5 8250U 1.6GHz	6GB	256GB
157	156	Lenovo	2 in 1 Convertible	14.0	/ Touchscreen 1920x1080	tel Core i3 7100U 2.4GHz	4GB	256GB
158	157	Asus	2 in 1 Convertible	13.3	/ Touchscreen 1920x1080	tel Core i5 8250U 1.6GHz	8GB	256GB
159	158	Dell	2 in 1 Convertible	15.6	/ Touchscreen 1920x1080	tel Core i7 8550U 1.8GHz	8GB	256GB
160	159	Toshiba	Notebook	15.6	1366x768	tel Core i3 6006U 2.2GHz	4GB	500GB
161	160	Asus	Notebook	15.6	1366x768	MD A9-Series 9420 3GHz	4GB	1TB
162	161	Acer	Notebook	17.3	Panel Full HD 1920x1080	tel Core i5 8250U 1.6GHz	4GB	256GB
163	162	Dell	Notebook	15.6	Full HD 1920x1080	tel Core i5 8250U 1.6GHz	8GB	256GB
164	163	Lenovo	Gaming	15.6	Panel Full HD 1920x1080	l Core i7 7700HQ 2.8GHz	16GB	256GB
165	164	Acer	Notebook	15.6	1366x768	Dual Core N3350 1.1GHz	4GB	1TB
166	165	MSI	Gaming	15.6	Panel Full HD 1920x1080	l Core i7 7700HQ 2.8GHz	16GB	256GB SSD + 1TB
167	166	Acer	Notebook	15.6	1366x768	Quad Core N4200 1.1GHz	4GB	1TB
168	167	Dell	Gaming	15.6	Full HD 1920x1080	l Core i7 7700HQ 2.8GHz	8GB	128GB SSD + 1TB
169	168	Acer	Notebook	17.3	Panel Full HD 1920x1080	tel Core i5 8250U 1.6GHz	8GB	256GB
170	169	HP	Notebook	13.3	Panel Full HD 1920x1080	tel Core i5 8250U 1.6GHz	4GB	500GB
171	170	Huawei	Ultrabook	13.0	Panel Full HD 2160x1440	tel Core i5 7200U 2.5GHz	8GB	256GB
172	171	HP	Notebook	17.3	Panel Full HD 1920x1080	tel Core i5 7200U 2.5GHz	6GB	2TB
173	172	Lenovo	Notebook	15.6	1366x768	D A6-Series 9220 2.9GHz	4GB	500GB

Figure 5.5 CSV Snapshot

jupyter laptop_data.csv Last Checkpoint: last month

File Edit View Settings Help

Delimiter: ,

		Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Me
1285	1284	Lenovo	2 in 1 Convertible	14.0	/ Touchscreen 1920x1080	tel Core i7 6500U 2.5GHz	4GB	128GB
1286	1285	Lenovo	2 in 1 Convertible	13.3	/ Touchscreen 3200x1800	tel Core i7 6500U 2.5GHz	16GB	512GB
1287	1286	Lenovo	Notebook	14.0	1366x768	Dual Core N3050 1.6GHz	2GB	64GB Flash Sto
1288	1287	HP	Notebook	15.6	1366x768	tel Core i7 6500U 2.5GHz	6GB	1TB
1289	1288	Asus	Notebook	15.6	1366x768	Dual Core N3050 1.6GHz	4GB	500GB
1290	1289	Asus	Ultrabook	13.3	Panel Full HD 1920x1080	ntel Core M 6Y30 0.9GHz	8GB	512GB
1291	1290	Dell	Notebook	15.6	1366x768	tel Core i3 7100U 2.4GHz	6GB	1TB
1292	1291	Acer	Notebook	15.6	1366x768	Dual Core N3060 1.6GHz	4GB	500GB
1293	1292	Dell	Notebook	15.6	1366x768	Dual Core N3050 1.6GHz	2GB	500GB
1294	1293	Lenovo	Notebook	15.6	Panel Full HD 1920x1080	l Core i7 6700HQ 2.6GHz	8GB	1TB
1295	1294	HP	Notebook	15.6	Full HD 1920x1080	D A9-Series 9410 2.9GHz	6GB	1.0TB H
1296	1295	Dell	Notebook	15.6	1366x768	tel Core i7 7500U 2.7GHz	8GB	1TB
1297	1296	HP	Netbook	11.6	1366x768	Dual Core N3060 1.6GHz	2GB	32GB Flash Sto
1298	1297	Asus	Notebook	15.6	1366x768	tel Core i7 6500U 2.5GHz	4GB	500GB
1299	1298	Lenovo	2 in 1 Convertible	14.0	/ Touchscreen 1920x1080	tel Core i7 6500U 2.5GHz	4GB	128GB
1300	1299	Lenovo	2 in 1 Convertible	13.3	/ Touchscreen 3200x1800	tel Core i7 6500U 2.5GHz	16GB	512GB
1301	1300	Lenovo	Notebook	14.0	1366x768	Dual Core N3050 1.6GHz	2GB	64GB Flash Sto
1302	1301	HP	Notebook	15.6	1366x768	tel Core i7 6500U 2.5GHz	6GB	1TB
1303	1302	Asus	Notebook	15.6	1366x768	Dual Core N3050 1.6GHz	4GB	500GB

Figure 5.6 CSV Snapshot

Chapter 6

Summary & Conclusion

This project aimed to develop a machine learning-based system to predict laptop prices based on user-defined specifications such as brand, CPU, RAM, storage, and display size. A dataset of 1,300 laptop records was carefully cleaned, preprocessed, and analyzed. Various machine learning algorithms, including Linear Regression, Ridge Regression, Decision Trees, Random Forest, XGBoost, and Gradient Boosting, were employed to predict the price of laptops. The models were evaluated using R^2 score and Mean Absolute Error (MAE). Among the algorithms tested, Random Forest outperformed the others with an R^2 score of 0.90 and an MAE of 0.14, making it the most accurate model for laptop price prediction.

The final model was deployed in a user-friendly web application built using Streamlit, which allows users to input laptop specifications and receive an estimated price instantly. Consumers can make informed buying decisions, while retailers can optimize their pricing strategies based on the predicted prices.

The Laptop Price Prediction system successfully demonstrates how machine learning can be applied to estimate laptop prices based on specifications such as brand, processor, RAM, storage, GPU, and screen size. By collecting and preprocessing a dataset of 1,300 laptops, and training various regression models, the project identified Random Forest as the most accurate algorithm. The integration of this model into a user-friendly web application allows real-time price predictions, which can assist both consumers and retailers in making informed decisions. Overall, the system proves to be an effective and practical tool for price estimation in the electronics market.

This study demonstrates the potential of machine learning to automate and enhance the decision-making process in the consumer electronics market. By analyzing historical pricing data, the system identifies key factors influencing laptop prices and provides accurate predictions, making it a valuable tool for both consumers and businesses.

Chapter 7

Future Scope

The current implementation of the laptop price prediction system is robust, but it can be significantly enhanced in the future to provide a broader range of functionality and improved accuracy. One promising direction is the integration of real-time market data from e-commerce platforms such as Amazon, Flipkart, or Best Buy. This would enable the system to reflect ongoing price fluctuations, seasonal offers, and new product launches, thereby making the price prediction more dynamic and practical for both consumers and retailers.

Another potential enhancement is the ability to classify laptops into defined price ranges or tiers—such as budget, mid-range, and premium—rather than predicting a specific price point. This range-based approach is often more aligned with user expectations and can help simplify the decision-making process, especially for first-time or non-technical buyers. Additionally, including time-series forecasting would allow users to anticipate future pricing trends, enabling smarter decisions about when to buy a device.

From a technical standpoint, future versions of the model can incorporate deep learning algorithms, particularly neural networks, which have the potential to identify complex patterns in large datasets more effectively than traditional models. As the dataset grows, deep learning can further enhance predictive accuracy and robustness. Moreover, the system can be extended beyond laptops to include other consumer electronics like smartphones, tablets, smartwatches, and desktops, turning the platform into a comprehensive product price estimation system.

To increase global usability, the application could offer multilingual support and would further broaden its accessibility to non-English-speaking users. Additionally, implementing AI-powered virtual assistants or chatbots could simplify user interactions, allowing them to input their preferences via text or voice and receive recommendations or price estimates conversationally.

In summary, the future scope of this project is vast and promising. By incorporating advanced machine learning techniques, expanding device compatibility, enabling personalization, and making the tool more interactive and accessible, this system can evolve into an intelligent, full-fledged pricing assistant. It would not only serve individuals looking to make informed purchases but also support businesses in optimizing sales strategies and staying competitive in a fast-changing technological marketplace.

Chapter 8

References

References

1. R. J. Kafabihi, “Predicting Laptop Prices Based on Specifications Using Machine Learning Techniques: An Empirical Study,” IEESE International Journal of Science and Technology (IJSTE), vol. 13, no. 1, pp. 14–22, 2024.

2. B. Prajna, A. L. Sri, A. Sahiti, A. B. Teja, and A. Arzoo, “LAPTOP PRICE PREDICTION USING MACHINE LEARNING,” 2023. [Online]. Available: www.ijnrd.org

3. P. Tian, “Research On Laptop Price Predictive Model Based on Linear Regression, Random Forest and Xgboost,” 2024.

4. L. Akkouh, “Machine Learning-Based Price Prediction for Laptops.” [Online]. Available: https://www.researchgate.net/publication/377721653 P. Manegopale, K. Nerpagar, S. Sawant, M. Shinde, and K. Chandarkar, “Laptop Prediction & Comparison using Machine Learning,” 2023.

5. Y. Karakuş and T. T. Bilgin, “International Journal of Multidisciplinary Studies and Innovative Technologies Laptop Price Range Prediction with Machine Learning Methods,” ijmsit Research Article, vol. 2022, no. 1, pp. 40–45, doi: 10.36287/ijmsit.8.1.7.

6. V. Surjuse, S. Lohakare, A. Barapatre, and A. Chapke, “Laptop Price Prediction using Machine Learning,” International Journal of Computer Science and Mobile Computing, vol. 11, no. 1, pp. 164–168, Jan. 2022, doi: 10.47760/ijcsmc.2022.v11i01.021.

7. K. S. Priya, “Linear Regression Algorithm in Machine Learning through MATLAB,” Int J Res Appl Sci Eng Technol, vol. 9, no. 12, pp. 989–995, Dec. 2021, doi: 10.22214/ijraset.2021.39410.

8. L. Breiman, “Random Forests,” 2001.

9. Y. C. Chen, H. C. Su, S. M. Huang, C. H. Yu, J. H. Chang, and Y. L. Chiu, “Immune cell profiles and predictive modeling in osteoporotic vertebral fractures using XGBoost machine learning algorithms,” BioData Min, vol. 18, no. 1, Dec. 2025, doi: 10.1186/s13040-025-00427-y.