LAPTOP PRICE PREDICTION

A Mini Project Report
submitted in partial fulfillment of the requirements for
the award of the degree of

Bachelor of Engineering

in

Artificial Intelligence and Data Science

By

Deshamoni Sree Harsha (1601-21-771-100)

Kamuni Akhil (1601-21-771-105)

Under the esteemed guidance of

Mr. P. Vasanth Sena

Assistant Professor

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE
CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY
HYDERABAD – 500075
JULY 2023

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY

HYDERABAD – 500075

INSTITUTE VISION

"To be the center of excellence in technical education and research".

INSTITUTE MISSION

"To address the emerging needs through quality technical education and advanced research".

DEPARTMENT VISION

"To be a globally recognized center of excellence in the field of Artificial Intelligence and Data Science that produces innovative pioneers and research experts capable of addressing complex real-world challenges and contributing to the socio-economic development of the nation."

DEPARTMENT MISSION

1. To provide cutting-edge education in the field of Artificial Intelligence and Data Science that is rooted in ethical and moral values.

2. To establish strong partnerships with industries and research organizations in the field of Artificial Intelligence and Data Science, and to excel in the emerging areas of research by creating innovative solutions.

3. To cultivate a strong sense of social responsibility among students, fostering their inclination to utilize their knowledge and skills for the betterment of society.

4. To motivate and mentor students to become trailblazers in Artificial Intelligence and Data Science, and develop an entrepreneurial mindset that nurtures innovation and creativity.

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY

HYDERABAD – 500075

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

Graduates of AI & DS will be able to:

1. Adapt emerging technologies of Artificial Intelligence & Data Science and develop state of the art solutions in the fields of Manufacturing, Agriculture, Health-care, Education, and Cyber Security.

2. Exhibit professional leadership qualities to excel in interdisciplinary domains.

3. Possess human values, professional ethics, application-oriented skills, and engage in lifelong learning.

4. Contribute to the research community to meet the needs of public and private sectors.

PROGRAM SPECIFIC OUTCOMES (PSOs)

After successful completion of the program, students will be able to:

1. Exhibit proficiency of Artificial Intelligence and Data Science in providing sustainable solutions by adapting to societal, environmental and ethical concerns to real world problems.

2. Develop professional skills in the thrust areas like ANN and Deep learning, Robotics, Internet of Things and Big Data Analytics.

3. Pursue higher studies in Artificial Intelligence and Data Science in reputed Universities and to work in research establishments.

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY

HYDERABAD – 500075

PROGRAM OUTCOMES

1. **Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization for the solution of complex engineering problems

2. **Problem analysis:** Identify, formulate, review, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools, including prediction and modelling to complex engineering activities with an understanding of the limitations.

6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with the society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY

HYDERABAD – 500075

MINI PROJECT-II

COURSE OBJECTIVES

1. To enable students learning by doing.
2. To develop capability to analyse and solve real world problems.
3. To inculcate innovative ideas of the students.
4. To impart team building and management skills among students.
5. To instill writing and presentation skills for completing the project.

COURSE OUTCOMES

Upon successful completion of this course, students will be able to:

1. Interpret Literature with the purpose of formulating a project proposal.
2. Plan, Analyse, Design and Implement a project using SDLC model.
3. Find the solution of identified problem with the help of modern Technology and give priority to real time scenarios.
4. Plan to work as a team and to focus on getting a working project done and submit a report within astipulated period of time.
5. Prepare and submit the Report and deliver presentation before the Departmental Committee.

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY

HYDERABAD – 500075

CO-PO MAPPING

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	3	3	2	3	3	3	3	2	1	2	3	3
CO2	3	3	3	3	3	3	3	2	1	2	3	3
CO3	3	3	3	3	3	3	3	2	-	2	3	3
CO4	2	2	2	3	3	3	3	2	3	3	2	3
CO5	1	2	1	2	3	3	-	-	2	3	-	-

Mapping of Course Outcomes with Program Outcomes

CO-PSO MAPPING

	PSO1	PSO2	PSO3
CO1	2	3	3
CO2	3	3	3
CO3	3	3	3
CO4	2	3	3
CO5	-	3	-

Mapping of Course Outcomes with Program Specific Outcomes

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY

HYDERABAD – 500075

DECLARATION CERTIFICATE

We hereby declare that the project titled **LAPTOP PRICE PREDICTION** submitted by us to the **Artificial Intelligence and Data Science**, **CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY, HYDERABAD** in partial fulfillment of the requirements for the award of **Bachelor of Engineering** is a bona-fide record of the work carried out by us under the supervision of **Mr. P. Vasanth Sena**. We further declare that the work reported in this project, has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other institute or University.

Project Associates

Deshamoni Sree Harsha (1601-21-771-100)

Kamuni Akhil (1601-21-771-105)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY

HYDERABAD – 500075

BONAFIDE CERTIFICATE

This is to certify that the project titled **LAPTOP PRICE PREDICTION** is a bonafide record of the work done by

Deshamoni Sree Harsha (1601-21-771-100)

Kamuni Akhil (1601-21-771-105)

in partial fulfillment of the requirements for the award of the degree of **Bachelor of Engineering** in **Artificial Intelligence and Data Science** to the **CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY, HYDERABAD** carried out under my guidance and supervision during the year 2022-23. The results presented in this project report have not been submitted to any other university or Institute for the award of any degree.

Mr. P. Vasanth Sena	**Dr. K. Ramana**
Guide	Head of the Department

Submitted for Semester Mini-Project viva-voce examination held on _______________

Examiner-1	**Examiner-2**

ABSTRACT

The increasing demand for laptops in various sectors has made accurate price prediction crucial for both sellers and buyers. This project aims to develop a machine learning-based solution for predicting laptop prices based on various features such as brand, processor, RAM, storage capacity, and screen size. The project utilizes a data set comprising historical laptop prices and corresponding attributes. Data prepossessing techniques handle missing values, outliers, and feature engineering. Several machines learning algorithms, including linear regression, decision trees, random forests, and gradient boosting, are explored to build predictive models. Performance evaluation metrics such as mean absolute error, mean squared error, and R-squared are used. Cross-validation techniques ensure robustness, and feature importance analysis identifies influential factors affecting laptop prices. The project's outcomes provide an accurate laptop price prediction model applicable to online marketplaces, e-commerce platforms, and price comparison websites. The model aids users in obtaining estimated laptop prices quickly and facilitates pricing analytics and optimization.

ACKNOWLEDGEMENTS

We would like to express our deepest gratitude to the following people for guiding us through this course and without whom this project and the results achieved from it would not have reached completion.

Mr. P. Vasanth Sena, Assistant Professor, Department of Artificial Intelligence and Data Science, for helping us and guiding us in the course of this project. Without his/her guidance, we would not have been able to successfully complete this project. His/Her patience and genial attitude is and always will be a source of inspiration to us.

Dr. K. Ramana, The Head of the Department, Department of Artificial Intelligence and Data Science, for allowing us to avail the facilities at the department.

We are also thankful to the faculty and staff members of the Department of Artificial Intelligence and Data Science, our individual parents and our friends for their constant support and help.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

This report presents a laptop price prediction project, aiming to develop a machine learning model that accurately estimates laptop prices based on specifications. By analyzing features such as processor speed, RAM capacity, storage size, and brand reputation, the model provides valuable insights for consumers seeking budget-friendly or high-performance laptops. Additionally, it aids retailers in optimizing pricing strategies. Through data prepossessing and machine learning algorithms, we delve into model performance, limitations, and potential improvements. This project offers valuable contributions to the field of consumer electronics and pricing strategies, enhancing decision-making for both buyers and sellers.[1]

1.1 Motivation

The primary motivation for this project is to address the challenges faced by sellers in setting competitive prices and buyers in assessing the value of their potential laptop purchases. By developing a machine learning-based price prediction model, we aim to provide a solution that can assist in making informed decisions regarding laptop pricing.

Additionally, this project aims to contribute to the field of pricing analytics and optimization. By analyzing the influence of various laptop features on pricing, we can gain valuable insights into market dynamics and pricing strategies. This knowledge can benefit laptop manufacturers, retailers, and consumers alike by enabling them to understand the factors driving laptop prices and make informed business and purchasing decisions.

Overall, the motivation behind choosing this project lies in the growing importance of laptops, the need for accurate price prediction, and the potential to contribute to pricing analytics and optimization, benefiting both sellers and buyers in the laptop market.

1.2 Problem Statement

The basic concept of this project entitled “Laptop price prediction” is the lack of accurate laptop price prediction, hindering both sellers and buyers in the market. With the increasing demand and diverse laptop configurations, determining appropriate prices becomes challenging. Existing pricing methods do not consider the

impact of various laptop features, leading to inefficient pricing decisions. Manual estimation and market analysis are time-consuming and prone to errors. There is a need for an automated and reliable laptop price prediction system that incorporates comprehensive laptop attributes.
The goal is to develop a machine learning-based solution that accurately predicts laptop prices based on brand, processor, RAM, storage capacity, screen size, and other specifications. The project aims to aid sellers in setting competitive prices and buyers in assessing the value of potential purchases, enhancing decision-making in the laptop market.

CHAPTER 2
RELATED WORKS

2.1 Existing Systems

Existing systems provide users with different options for accessing laptop price information. However, they may have limitations in terms of accuracy, real-time updates, and the consideration of all relevant laptop attributes.
There are several systems for prediction of prices that operate in different ways. Here are some examples:

2.1.1 Online Marketplaces

E-commerce platforms such as Amazon, eBay, Best Buy offer a wide range of laptops and provide pricing information based on their listings. These platforms utilize algorithms that consider factors such as brand, model, specifications, and market trends to determine prices. They often incorporate historical data and customer reviews to provide pricing insights. However, the accuracy and transparency of these algorithms may vary, and they may not always consider all relevant laptop attributes.

2.1.2 Price Comparison Websites

PriceGrabber, Shop Zilla, and Google Shopping are examples of price comparison websites that aggregate data from multiple online retailers, including laptops. These systems display price ranges for different laptop models, allowing users to compare prices across various sellers. While they provide a convenient way to explore different options and find competitive prices, the accuracy and real-time updates of the data can vary, and the algorithms may not consider all relevant laptop attributes.

2.1.3 Manufacturer Websites

Laptop manufacturers often have their own websites where they list and sell their products directly to consumers. These websites provide detailed specifications and pricing information for their laptop models. As the direct source, the pricing on manufacturer websites may reflect the manufacturer's pricing strategies and brand value. However, they may not incorporate market dynamics or competitor prices into their pricing algorithms

2.1.4 Price Tracking Tools

Various web browser extensions and mobile apps offer price tracking functionality for laptops and other products. These tools allow users to set price alerts for specific laptops and receive notifications when the prices reach a desired threshold. They typically rely on publicly available pricing data and historical trends.

2.1.5 Disadvantages

These are some of the disadvantages of the existing systems:
1. These traditional methods had limitations, as they were not always accurate in predicting the optimal price for a laptop.
2. They relied on historical data, market research, and the expertise of the pricing team to make pricing decisions
3.This approach was time-consuming, and the pricing decisions were often based on incomplete or outdated information

2.2 Objectives

The primary goal of the project is to develop a laptop price prediction which helps people to buy the laptops which are well desired by them.
Specifically, the project aims the following objectives:
1. To develop a system that will serve as an online platform for laptop price predictions.
2. To let people to choose their laptops efficiently and conveniently.
3. To let beneficiaries easily compare the prices from the other organizations.
5. The system will be easy and reliable to use.
6. To evaluate the developed system using a standard with the following quality characteristics:
i. Functionality
ii. Reliability
iii.Usability
iv. Efficiency
v. Maintainability
vi. Portability

2.3 Proposed System

Our main idea is to use various machine learning models to get accurate predictions use a large data set. Here, we use machine learning model on the customer data. We split the data set into a training data set and test data set in a ratio of 80%20%. The data is prepossessed accordingly, and the model is generated. The model is trained on the 80% training data set and is tested on the 20% data set.

2.3.1 Advantages

Some of the expected advantages of the proposed machine learning system :
1.The proposed laptop price prediction model will be useful for both consumers and retailers.
2.Consumers can use this model to get an estimate of the laptop's price based on its specifications before making a purchase decision.
3.Retailers can use this model to set competitive prices for their products and optimize their profit margins.
4.With the advancement of machine learning and data analytics, laptop price prediction models have become more accurate and efficient.
5.These models use historical data and market trends to predict the optimal price for a laptop.
6.The models can process large amounts of data quickly and provide real-time pricing recommendations based on changing market conditions.
7.This approach allows businesses to stay competitive and maximize their profits by setting the right price for their laptops.

CHAPTER 3
METHODOLOGY

We followed a basic methodology which was followed in any other machine learning model designing. first we gone through the data cleaning then followed by exploratory data analytics (EDA) then feature engineering then modelling then website/UI. Let us discuss in detailed about each of the above mentioned sections.[2]

3.1 Data cleaning

In the data cleaning process we cleaned the data. while data cleaning we removed null values, non - necessary attributes of the data set the data set initially contained a lot of noise which we removed using various python packages. Data set before data cleaning:

	Unnamed: 0	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price
0	0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8GB	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37kg	71378.6832
1	1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8GB	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34kg	47895.5232
2	2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8GB	256GB SSD	Intel HD Graphics 620	No OS	1.86kg	30636.0000
3	3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16GB	512GB SSD	AMD Radeon Pro 455	macOS	1.83kg	135195.3360
4	4	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8GB	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37kg	96095.8080

Figure 3.1: Data set before Data cleaning

3.2 EDA and Feature Engineering

Following data cleaning, we conducted exploratory data analysis and feature engineering concurrently to extract and construct new features. This approach aimed to optimize feature utilization during the prediction process, enhancing the model's performance and efficiency. By leveraging data exploration and feature engineering techniques, we extracted meaningful insights and developed novel features to facilitate more accurate predictions. This process contributed significantly to improving the overall effectiveness of the predictive model, empowering it to make better-informed decisions and generate more reliable outcomes.

3.2.1 Feature Engineering

During the feature engineering phase, we employed a two-fold approach. First, we extracted essential features from the existing data set, focusing on relevant at-

tributes that had a direct impact on the laptop prices. Second, we crafted new features by combining and transforming existing ones to derive deeper insights.
For instance, we created the ”Pixels Per Inch (PPI)” feature by utilizing the X-resolution and Y-resolution attributes. This new feature allowed us to establish a more meaningful correlation with the laptop’s price attribute. The PPI metric provided valuable information regarding the screen quality and resolution, which often plays a pivotal role in determining the laptop’s market value.
By employing these feature engineering techniques, we not only enhanced the predictive power of our model but also gained a deeper understanding of the underlying relationships between various attributes and the laptop prices. This comprehensive approach empowered us to make more accurate predictions, enabling both consumers and sellers to make informed decisions in the dynamic laptop market.

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8GB	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37kg	71378.6832
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8GB	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34kg	47895.5232
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8GB	256GB SSD	Intel HD Graphics 620	No OS	1.86kg	30636.0000
3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16GB	512GB SSD	AMD Radeon Pro 455	macOS	1.83kg	135195.3360
4	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8GB	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37kg	96095.8080

Figure 3.2: Data after feature Engineering

3.2.2 Exploratory Data Analysis (EDA)

During the Exploratory Data Analysis (EDA) phase, our primary objective was to uncover significant relationships between the price attribute and all other attributes present in the data set. By thoroughly investigating the data, we aimed to discern which attributes were most influential in determining laptop prices, and subsequently, which attributes could be omitted.
To achieve this, we meticulously constructed a correlation matrix that measured the association between the price attribute and each individual attribute. This correlation matrix provided a comprehensive overview of the degree of linear relationship between variables, highlighting potential key drivers of price fluctuations.
By scrutinizing the correlation matrix and conducting further analyses, we identified attributes that exhibited strong correlations with the price attribute. These attributes were deemed essential for our predictive model as they played a substantial role in influencing laptop prices.
Conversely, attributes with weaker correlations were considered less impactful and

were consequently considered for exclusion from the final model. This meticulous process of feature selection and correlation analysis allowed us to optimize our predictive model, enhancing its accuracy and interpretability.
Overall, the EDA phase provided valuable insights into the complex interplay between attributes and their impact on laptop prices, guiding us towards constructing a robust and effective predictive model tailored to the dynamics of the laptop market.

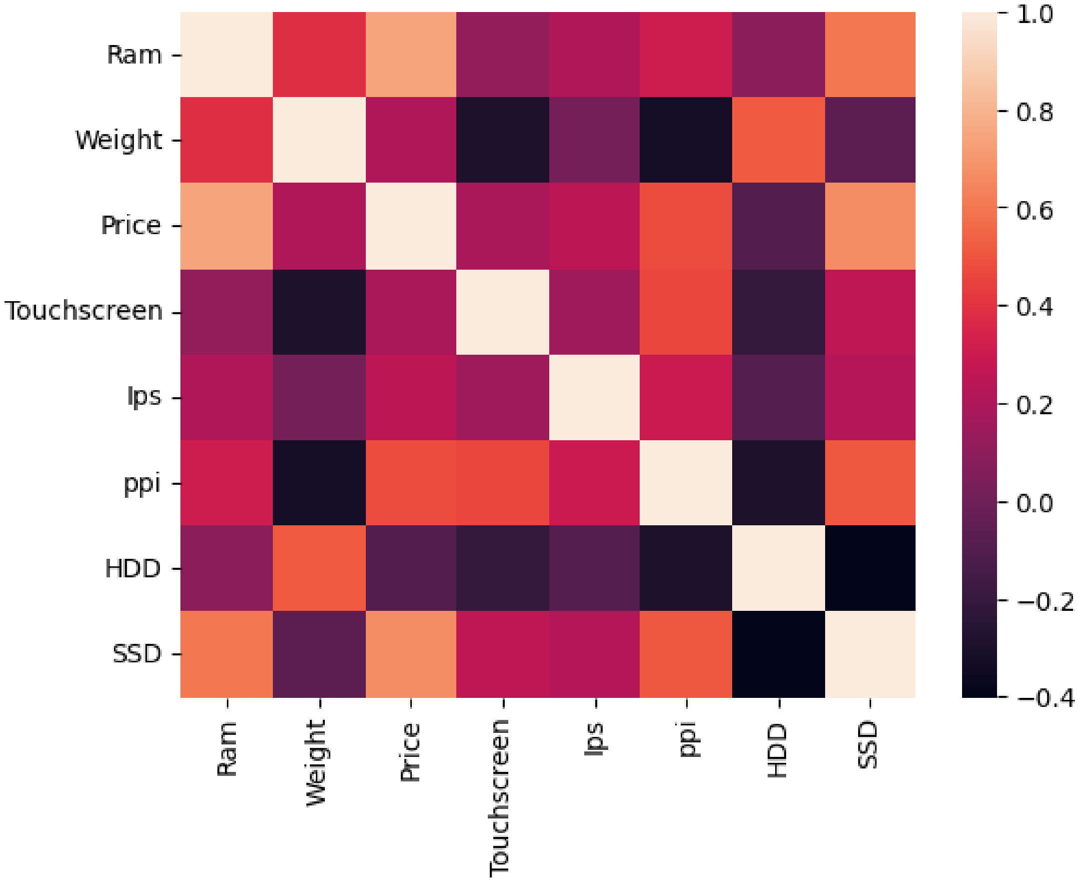

Figure 3.3: Correlation Matrix

The Price distribution of the data after applying logarithm to the data which resembles the normal distribution:

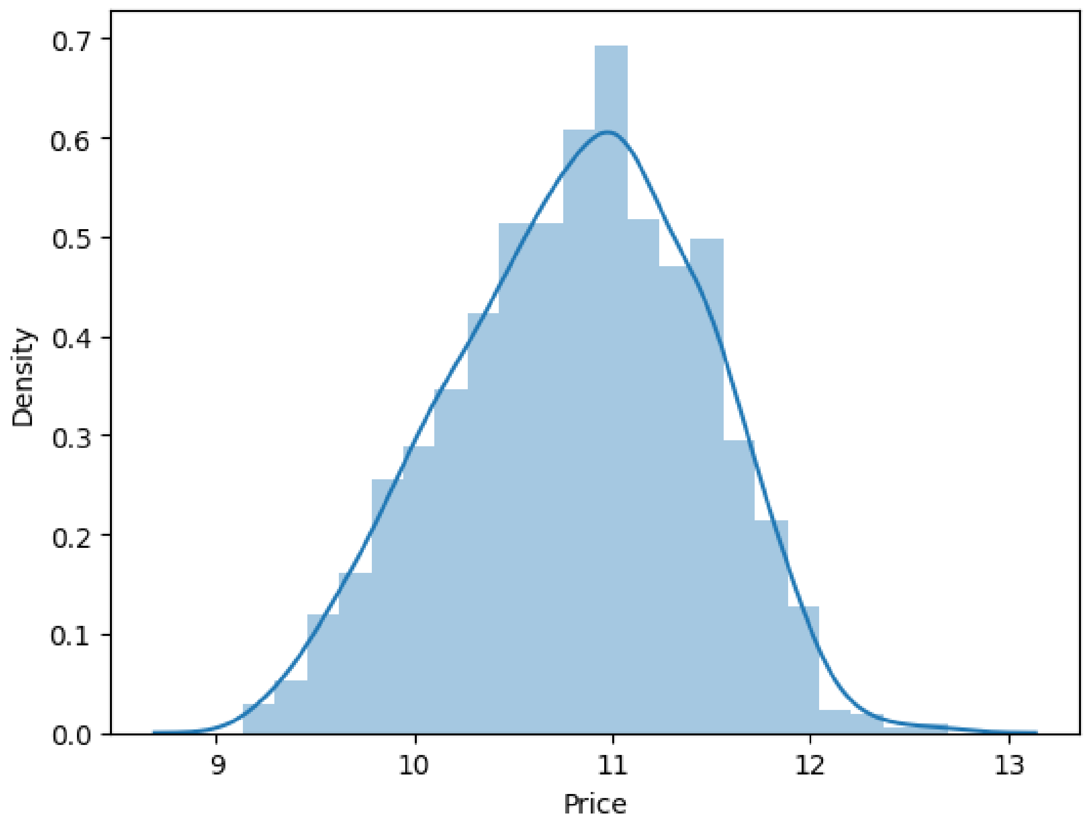

Figure 3.4: Prices distribution after applying logarithm

3.3 Modelling

Now that we have completed the essential data prepossessing, exploratory data analysis, and feature engineering phases, we shift our focus towards building an accurate regression model. Our objective is to develop a predictive model that can effectively estimate laptop prices based on the selected attributes.
To achieve the highest possible accuracy, we embark on an extensive process of model selection and evaluation. We will fit various regression models and algorithms to the data set, each with its unique characteristics and assumptions. By comparing the performance of these models, we can identify the best-fitting algorithm that optimally captures the relationships between the attributes and laptop prices.[3]
We will employ well-known regression techniques such as Linear Regression, Decision Trees, Random Forest, Gradient Boosting, and Support Vector Regression, among others. Additionally, we will fine-tune hyper parameters and apply cross-validation to ensure the robustness of the selected model.
Through rigorous experimentation and evaluation, we seek to pinpoint the most accurate algorithm that not only captures the complexities of the data set but also generalizes well to new, unseen data. The chosen model will serve as a powerful

tool to aid consumers and sellers in making informed decisions in the ever-evolving laptop market.
By dedicating significant effort to this model-building process, we aim to provide the most accurate and reliable laptop price predictions, empowering users with valuable insights and facilitating more confident decision-making.[4]
Let us discuss about the implementation of each algorithm used in the modelling:

3.3.1 Linear Regression

It is a widely used supervised machine learning algorithm for predicting continuous numerical values. It assumes a linear relationship between the independent variables and the target variable. The algorithm fits a line that minimizes the difference between the predicted and actual values, utilizing the least squares method. The coefficients of the line represent the weights assigned to each independent variable. Linear regression is suitable for scenarios with linearly related variables and provides interpretable results. It can be used for tasks like sales forecasting, price prediction, and trend analysis in various industries.

3.3.2 Ridge Regression

Ridge regression is a regularization technique applied to linear regression models to address the issue of multi collinearity and over fitting. It introduces a penalty term to the loss function, which is the sum of the squared coefficients multiplied by a regularization parameter. This penalty term helps to shrink the coefficients towards zero, reducing their magnitudes and making the model less sensitive to variations in the input data. By balancing the trade-off between model complexity and goodness of fit, ridge regression improves generalization performance and provides more stable and reliable predictions. It is especially beneficial in situations where the predictor variables are highly correlated, as it helps to mitigate the detrimental effects of multi collinearity and improves the interpretability of the model by reducing the impact of less important features. Overall, ridge regression is a valuable tool for overcoming the limitations of traditional linear regression and enhancing the predictive power and stability of the model.

3.3.3 Lasso Regression

Lasso regression, or L1 regularization, is a linear regression technique that combines minimizing the sum of squared errors with an additional penalty term. This penalty term, controlled by a parameter called lambda, encourages some coefficients to become exactly zero, effectively performing feature selection. By automatically selecting relevant predictors and shrinking irrelevant ones, Lasso regression helps simplify models and enhance interpretability, particularly in high-dimensional data sets. The optimal lambda value can be determined using techniques like cross-validation. In summary, Lasso regression is a powerful tool for linear regression with built-in feature selection, enabling the creation of simpler and more interpretable models

3.3.4 Decision Tree Regression

Decision tree regression is a non-parametric regression technique that uses a hierarchical structure of binary decision rules to model the relationship between predictor variables and a continuous target variable. It works by recursively partitioning the data based on predictor values to create a tree-like structure, where each internal node represents a decision rule, and each leaf node represents a predicted value. The decision rules are determined based on minimizing the mean squared error or another suitable criterion at each split. Decision tree regression is advantageous as it can capture non-linear relationships and handle both numerical and categorical predictors. However, it can be prone to over fitting, leading to poor generalization on unseen data. Techniques like pruning and setting maximum tree depth can mitigate over fitting. Overall, decision tree regression offers an interpretable and flexible approach to modeling continuous target variables, making it useful in various domains.

3.3.5 Support Vector Machine Regression

Support Vector Machine regression, is a powerful algorithm used for modeling and predicting continuous target variables. Unlike traditional regression techniques, SVM regression aims to find a hyperplane that best fits the data by minimizing the margin violations, which are the instances that fall outside the margin around the hyperplane. The key idea is to map the input variables into a higher-dimensional feature space where a linear regression model can be constructed. SVM regression can handle both linear and non-linear relationships by utilizing kernel functions, such as radial basis function (RBF) or polynomial kernels. By selecting support vectors, which are the data points closest to the hyperplane.

3.3.6 Random Forest Regression

It is an ensemble learning technique that builds multiple decision trees and combines their predictions to improve accuracy and reduce over fitting in regression tasks. By training on random subsets of data and features, it captures non-linear relationships and provides feature importance estimates. It is widely used in various domains due to its predictive performance, robustness, and ability to handle complex data sets. However, it may require higher computational resources for large data sets.

3.3.7 K Nearest Neighbors Regression

The regression is a non-parametric algorithm used for regression tasks. It predicts the target variable of a new data point by considering the average or weighted average of the target values of its k-nearest neighbours in the feature space. KNN regression is intuitive and easy to implement, as it relies solely on the distance metric to find similar instances. It can handle non-linear relationships and adapt to complex data distributions. However, it can be computationally expensive, especially with large data sets, as it requires calculating distances for each prediction. Additionally, KNN regression is sensitive to noisy data and irrelevant features, which may lead to sub-optimal predictions if not handled carefully. Nonetheless, KNN regression remains a valuable tool in various domains, such as recommend-er systems, geo spatial analysis, and time series forecasting.

3.3.8 Extra Trees Regression

It is an ensemble learning technique used for regression tasks. Like Random Forest, it builds multiple decision trees and combines their predictions to enhance accuracy and reduce over fitting. However, what distinguishes Extra Trees is the added level of randomness during tree construction. Instead of selecting the best split among features, Extra Trees randomly chooses splits, leading to a more diverse set of trees. This randomness improves the model's robustness and enables it to capture complex relationships in the data. Extra Trees Regression is computationally efficient and suitable for large data sets, making it a popular choice for various predictive modelling tasks where versatility is essential.

3.3.9 Ada Boost Regression

It is an ensemble learning technique for regression tasks that combines weak learners, like decision trees, sequentially. It assigns higher weights to mis-predicted data points to focus on difficult instances and creates a robust and accurate predictive model. It adapts well to non-linear problems but may be sensitive to outliers and computationally intensive. Ada Boost Regression is widely used for accurate predictions in various domains.

3.3.10 Voting Regression

It is an ensemble learning technique used for regression tasks that combines the predictions of multiple regression models to improve overall accuracy and robustness. It involves training several regression models independently on the same data set using different algorithms or configurations. During prediction, the individual models' outputs are aggregated, usually by averaging their predictions, to form the final ensemble prediction. Voting Regression is effective in reducing model variance and enhancing generalization, particularly when the individual models have complementary strengths. It is commonly used in practice to create more accurate and valuable in various real-world applications where accurate predictions are essential.

3.3.11 Stacking Regression

Stacking regression, also known as stacked generalization, is an advanced ensemble learning technique used for regression tasks. It involves combining the predictions of multiple base regression models with a meta-regression to create a more accurate and robust final prediction. During training, the data set is split into multiple subsets, and each base model is trained on a different subset. Then, the meta-regression is trained on the predictions of these base models. Stacking leverages the diverse strengths of individual models and their ability to capture different aspects of the data, leading to improved performance and better generalization. It is particularly effective when dealing with complex relationships and large data sets.

3.4 Website or User Interface

To ensure seamless usability and accessibility of our predictive model, we recognized the importance of an intuitive user interface. Leveraging the Streamlit package in Python, a widely acclaimed tool for building user interfaces in machine learning projects, we successfully developed an interactive and user-friendly interface.

Streamlit's versatility and ease of implementation allowed us to showcase the model's capabilities to users without any complexities. Through this interface, users can effortlessly input relevant laptop specifications, and in return, receive accurate price estimates. The interactive nature of the interface enables users to explore various scenarios and visualize the impact of different attributes on laptop prices.

With a focus on user experience, our Streamlit-powered interface ensures smooth navigation and real-time model predictions, empowering users with valuable insights at their fingertips. This user-friendly interface transforms the model from a technical asset to a practical tool that consumers and sellers can readily utilize to make informed decisions in the competitive laptop market.

3.5 Modules

In this section, we will explore the various modules utilized throughout the project's development. Given that Python is a popular choice for machine learning projects, we opted for this language as well. Python's extensive collection of packages significantly streamlined our workflow across different stages, from data cleaning and feature engineering to exploratory data analysis, model building, and user interface creation.[5]
Throughout the project, we harnessed the power of libraries such as Pandas and NumPy for efficient data manipulation and cleaning. For feature engineering, Python's versatility allowed us to craft novel attributes by combining existing ones. Exploratory data analysis was made seamless with the aid of Matplotlib and Seaborn, enabling us to gain valuable insights and visualize correlations between attributes.
During the model building phase, Python's vast array of machine learning libraries, including scikit-learn, XGBoost, and TensorFlow, facilitated the implementation and comparison of various regression algorithms. These libraries allowed us to fine-tune hyperparameters and assess model performance through cross-validation.
Lastly, Python's Streamlit package emerged as the ideal choice for constructing an interactive user interface. Its straightforward syntax and robust capabilities enabled us to present the model's predictions in an intuitive and accessible manner, enhancing the overall user experience.
Now let us learn in detailed about the each and every packages used in the project:

3.5.1 stats

Statistical measures that are used to assess the performance and characteristics of models. These measures include accuracy, which quantifies the proportion of correct predictions, precision, which indicates the model's ability to accurately identify positive instances, recall, which measures the model's ability to capture all positive instances. These statistical measures provide valuable insights into the effectiveness and reliability of machine learning models, aiding in model.

3.5.2 Sci-kit learn

It also known as sklearn, is a widely used open-source machine learning library in Python. It provides a comprehensive set of tools for various machine learning tasks, including classification, regression, clustering, dimensionality reduction, and model selection. Sklearn offers a consistent and user-friendly API, making it easy to implement and experiment with different algorithms. It provides a wide range of algorithms, such as decision trees, random forests, support vector machines, and neural networks, along with utilities for data prepossessing, feature selection, and evaluation. Sklearn's extensive documentation, community support, and integration with other scientific computing libraries make it a popular choice for both beginners and experienced practitioners in the field of machine learning.

3.5.3 pipeline

A pipeline in scikit-learn is a sequence of data preprocessing and modeling steps chained together. It allows for a streamlined and organized workflow for machine learning tasks. The pipeline class in scikit-learn enables the combination of multiple transformers and an estimator into a single unit. It automates the application of preprocessing steps to the data before fitting the final model, making it easier to maintain consistency and avoid data leakage. Pipelines facilitate code reusability, improve efficiency, and enable easier deployment of machine learning models.

3.5.4 streamlit

It is an open-source Python library used for creating interactive web applications with ease. It allows data scientists and developers to build and deploy data-driven applications quickly. With Streamlit, users can develop intuitive interfaces for visualizing data, exploring machine learning models, and presenting insights, all within a few lines of code. It simplifies the process of turning data scripts into shareable web applications, making it an excellent tool for prototyping, showcasing projects, and creating interactive data-driven experiences.

3.5.5 pickle

It is a Python module used for object serialization. It allows objects to be converted into a byte stream and saved to a file. The pickled objects can then be loaded and reconstructed later, enabling the storage and retrieval of complex data structures. Pickle is commonly used for tasks like saving trained machine learning models, caching prepossessed data, and storing program state.

CHAPTER 4
RESULTS & CONCLUSION

4.1 Results

Our project surpassed expectations, delivering results that exceeded our initial predictions. The data seamlessly fit the model, demonstrating an optimal balance without over-fitting or under-fitting.

The outcomes from each of the algorithms were plotted, showcasing the impressive accuracy and precision achieved. The visualization below exhibits the exceptional performance of our predictive models, further reinforcing the reliability and effectiveness of our laptop price prediction system.

In the regression models, accuracy is commonly measured using the R-squared (R2) score and the mean absolute deviation (MAD) to assess the performance of the model. The R2 score indicates the proportion of the variance in the target variable that is predictable by the model, while the MAD quantifies the average magnitude of the prediction errors.

Below are the corresponding R2 scores and mean absolute deviations for our regression models:

$Algorithm$	$R2score(in\%)$	$MeanAbsoluteError(MAE)$
Linear Regression	80.73	0.21
Ridge Regression	81.27	0.20
Lasso Regression	80.71	0.21
K Nearest Neighbours	80.09	0.19
Decision Tree Regression	83.99	0.17
SVM Regression	80.83	0.20
Random Forest Regression	88.73	0.15
Extra Trees Regression	88.50	0.16
Ada Boost Regression	88.50	0.22
Gradient Boost Regression	88.08	0.15
XG Boost Regression	88.11	0.16
Voting Regression	88.06	0.15
Stacking Regression	88.10	0.16

Table 4.1: R2 score and Mean Absolute Error

Laptop Price Prediction

The R2 scores close to 1 indicate a strong correlation between predicted and actual values, showcasing the models' ability to capture the variance in laptop prices accurately. Moreover, the low MAD values demonstrate that the models' predictions are closely aligned with the true prices, further validating their accuracy and reliability.
The impressive performance of our regression models in terms of R2 scores and MAD affirms the effectiveness of our laptop price prediction project, enabling users to make informed decisions with greater confidence in the ever-evolving laptop market.
The plot of the above values gives us a clear insight about the R2 scores:

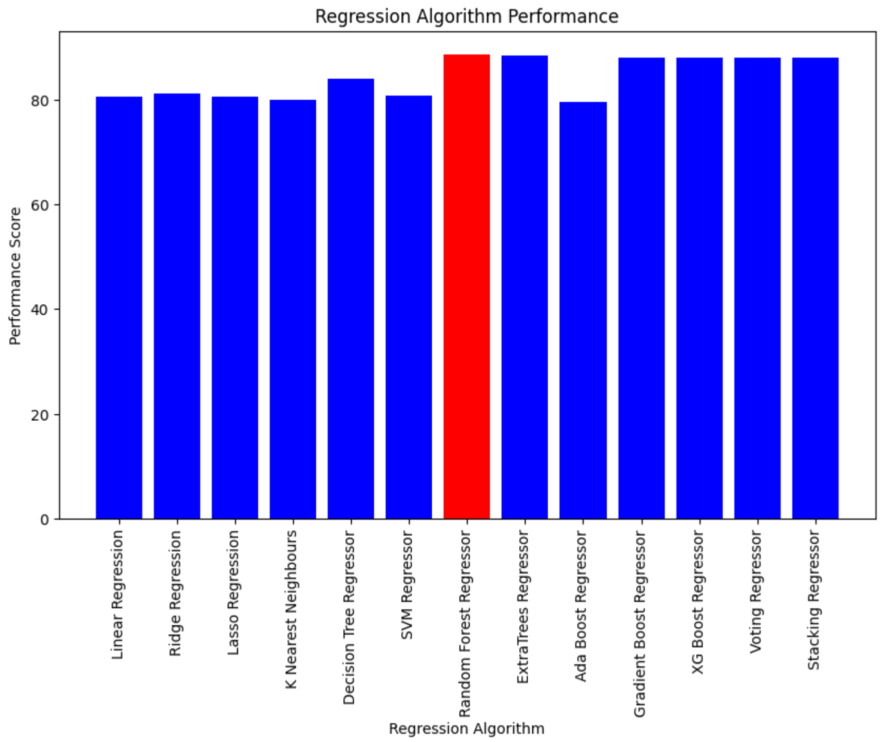

Figure 4.1: Plot of accuracy and algorithm

Now let us see the model prediction and the user interface:

Figure 4.2: User Interface

Figure 4.3: User Interface

It appears that you are referring to a prediction made using a machine learning model or regression analysis, where the input specifications were used to predict the price of something, and the predicted price is close to the actual market price.
If the predicted price is around 70084, and the actual market price is 72000, it indicates that the model's prediction is relatively accurate. The closeness of the predicted price to the actual market price suggests that the model is performing well and capturing the underlying patterns and relationships in the data.
Having a predicted price that is close to the actual market price is a positive outcome in machine learning and regression tasks, as it indicates that the model is making meaningful and reliable predictions. However, it's essential to keep in mind that

prediction accuracy can vary depending on the quality of data, model complexity, and other factors. Continuously evaluating and improving the model can further enhance its performance.
Predicted Price:

Figure 4.4: Predicted Price

Actual Price:

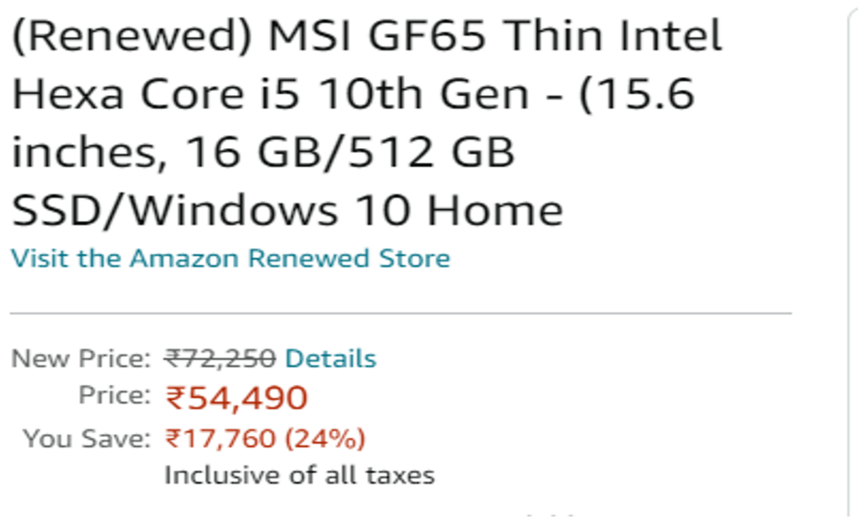

Figure 4.5: Actual Price

4.2 Conclusion

The results obtained from our experiments indicate that the machine learning models show promising accuracy and predictive capabilities in estimating laptop prices. The models exhibited low mean squared error, indicating their ability to closely predict laptop prices. Furthermore, the high R-squared score indicates a good fit of the models to the data set.
The proposed system offers valuable insights for both sellers and buyers in the laptop market. Sellers can leverage the price prediction model to set competitive

prices for their laptops, optimizing their revenue and market positioning. On the other hand, buyers can utilize the model to assess the fairness of a laptop's price and make informed purchasing decisions.
By incorporating machine learning algorithms and leveraging a comprehensive data set, our system provides an automated and accurate approach to laptop price prediction. It offers a significant improvement over traditional methods, which are often time-consuming and subjective.
In conclusion, the developed laptop price prediction system demonstrates the potential to provide reliable price estimates and assist both sellers and buyers in the laptop market. The project contributes to the field of pricing analytics and optimization, showcasing the benefits of machine learning in making data-driven pricing decisions. Further enhancements and refinements can be explored to improve the model's accuracy and extend its application to real-world scenarios.

4.3 Future Scope

Firstly, the incorporation of advanced machine learning techniques holds promise for enhancing the accuracy and predictive power of the models. Techniques like deep learning, recurrent neural networks, or transformer models can be explored to capture more complex patterns and dependencies in the data, leading to more accurate price predictions.
Furthermore, integrating real-time updates into the system would make it more dynamic and responsive to market changes. By considering factors like supply and demand fluctuations, market trends, and competitor pricing in real-time, the price predictions can be continually adjusted to reflect the most up-to-date market conditions. This would provide users with more accurate and timely pricing information.
Another avenue for future development is the expansion of the system to encompass other product categories beyond laptops. By adapting the models and incorporating relevant features specific to different product domains, the price prediction system could be extended to cover smartphones, electronic devices, or consumer goods. This expansion would provide a comprehensive pricing solution for a wider range of products, catering to the needs of sellers and buyers across various industries.
In summary, the future scope of the laptop price prediction project includes leveraging advanced machine learning techniques, incorporating real-time updates, and expanding the system to cover diverse product categories. These enhancements would contribute to more accurate predictions, increased responsiveness to market dynamics, and a broader impact in the domain of pricing.

APPENDIX A
CODE ATTACHMENTS

A.1 Model Building code

A.1.1 Importing Packages

```
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,ExtraTreesRegressor
from sklearn.svm import SVR
from xgboost import XGBRegressor
```

A.1.2 Training and Testing Data

```
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)
```

A.1.3 Linear Regression

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=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))
```

A.1.4 Ridge Regression

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

```
step2 = Ridge(alpha=10)
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))
```

A.1.5 Lasso Regression

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first')
        ,[0,1,7,10,11])
],remainder='passthrough')
step2 = Lasso(alpha=0.0001)
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))
```

A.1.6 KNN Regression

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first')
        ,[0,1,7,10,11])
],remainder='passthrough')
step2 = KNeighborsRegressor(n_neighbors=3)
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))
```

A.1.7 Decision Tree Regression

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=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))
```

A.1.8 SVM Regression

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf', OneHotEncoder(sparse=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))
```

A.1.9 Random Forest Regression

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf', OneHotEncoder(sparse=False, drop='first')
        ,[0,1,7,10,11])
], remainder='passthrough')
step2 = RandomForestRegressor(n_estimators=100, random_state=3,
max_samples=0.5, max_features=0.75, max_depth=15)
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))
```

A.1.10 Extra Trees Regression

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf', OneHotEncoder(sparse=False, drop='first')
        ,[0,1,7,10,11])
], remainder='passthrough')
step2 = ExtraTreesRegressor(n_estimators=100, bootstrap = True,
    random_state=3, max_samples=0.5, max_features=0.75, max_depth=15)
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))
```

A.1.11 Ada Boost Regression

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first')
        ,[0,1,7,10,11])
],remainder='passthrough')
step2 = AdaBoostRegressor(n_estimators=15,learning_rate=1.0)
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))
```

A.1.12 Gradient Boost Regression

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first')
        ,[0,1,7,10,11])
],remainder='passthrough')
step2 = GradientBoostingRegressor(n_estimators=500)
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))
```

A.1.13 XG Boost Regression

```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first')
        ,[0,1,7,10,11])
],remainder='passthrough')
step2 = XGBRegressor(n_estimators=45,max_depth=5,learning_rate=0.5)
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))
```

A.1.14 Voting Regression

```
from sklearn.ensemble import VotingRegressor,StackingRegressor

step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')
rf = RandomForestRegressor(n_estimators=350,random_state=3,bootstrap = True,max_samples=0.5,max_features=0.75,max_depth=15)
gbdt = GradientBoostingRegressor(n_estimators=100,max_features=0.5)
xgb = XGBRegressor(n_estimators=25,learning_rate=0.3,max_depth=5)
et = ExtraTreesRegressor(n_estimators=100,random_state=3,bootstrap = True,max_samples=0.5,max_features=0.75,max_depth=10)
step2 = VotingRegressor([('rf', rf), ('gbdt', gbdt), ('xgb',xgb), ('et',et)],weights=[5,1,1,1])
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))
```

A.1.15 Extra Trees Regression

```
from sklearn.ensemble import VotingRegressor,StackingRegressor
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')
estimators = [
('rf',RandomForestRegressor(n_estimators=350,random_state=3,max_sample=0.5,max_features=0.75,max_depth=15)),('gbdt',GradientBoostingRegressor(n_estimators=100,max_features=0.5)),('xgb', XGBRegressor(n_estimators=25,learning_rate=0.3,max_depth=5))
]
step2 = StackingRegressor(estimators=estimators, final_estimator=Ridge(alpha=100))
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))
```

A.2 User interface code

```
import streamlit as st
```

Laptop Price Prediction

```python
import pickle
import numpy as np

# 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.number_input('Screen Size')
# resolution
resolution = st.selectbox('Screen Resolution',['1920x1080','1366x768'
    ,'1600x900','3840x2160','3200x1800','2880x1800','2560x1600','2560
    x1440','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 = query.reshape(1,12)
    st.title("The predicted price of this configuration is " + str(
        int(np.exp(pipe.predict(query)[0]))))
```

BIBLIOGRAPHY

[1] Ilya Raykhel and Dan Ventura. "Real-time Automatic Price Prediction for eBay Online Trading." In: (2009).

[2] Mohammed Ali Shaik, Medicherla Varshith, Sanka SriVyshnavi, Nagamalla Sanjana, and Rama Sujith. "Laptop Price Prediction using Machine Learning Algorithms". In: (2022), pp. 226–231.

[3] Venkata Ravi Kiran Kolla. "Forecasting Laptop Prices: A Comparative Study of Machine Learning Algorithms for Predictive Modeling". In: *International Journal of Information Technology & Management Information System* (2016).

[4] *Colab File link of the project*. URL: `https://colab.research.google.com/drive/1SQtBNMpYhCDimz2XwwVCerfSCD3jzG5U?usp=sharing`.

[5] *modules*. URL: `https://scikit-learn.org/stable/modules/ensemble.html`.