Report on
Laptop Price Prediction

Index

Introduction

This Dataset aims to predict the prices of laptops based on their technical specifications, such as processor type, RAM size, storage capacity, brand, and other key features. Using a regression model, the project analyzes the relationship between these specifications and the corresponding laptop prices to provide accurate price predictions.

Understanding the Laptop Dataset

1.What problem are we solving?

This dataset is all about laptops, their specifications, and prices. The main problem it can help solve is understanding what factors influence a laptop's price and how different brands and specs compare. This could be useful for:

- Price prediction – Estimating how much a laptop should cost based on its specs.
- Buying advice – Helping people choose the best laptop for their needs and budget.
- Market analysis – Identifying trends in laptop pricing across brands.

2.Why does it matter?

- If you're buying a laptop, you don't want to overpay or get stuck with outdated hardware.
- Retailers can set competitive prices based on real data.
- Tech companies can see which features are in demand and adjust their offerings.

3.Where is the data from, and what does it include?

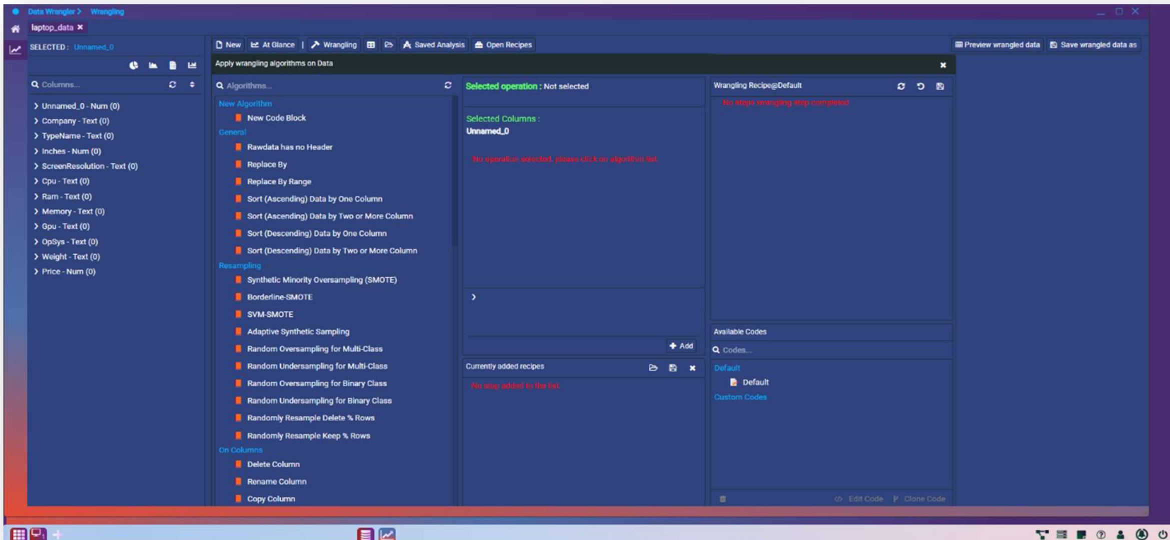
This dataset was downloaded from Kaggle. It provides detailed information about laptops, including:

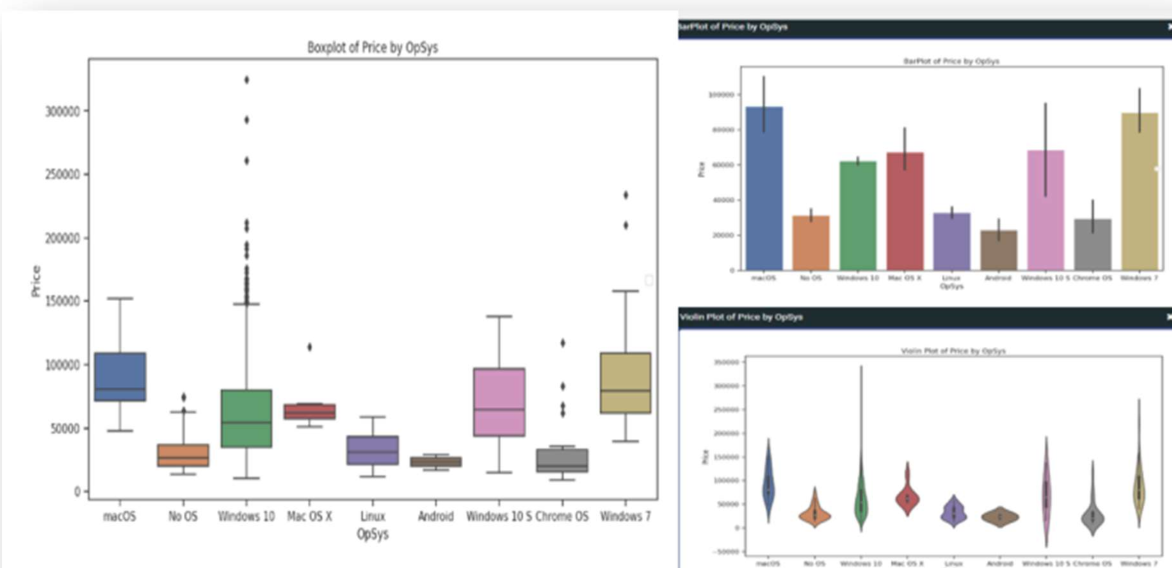
- **Brand & Type** – The manufacturer (Apple, HP, Dell, etc.) and the category (Ultrabook, Gaming, Notebook).
- **Screen Size & Resolution** – The display size in inches and resolution type (e.g., Full HD, Retina).
- **Processor (CPU) & Graphics (GPU)** – The computing power, which affects performance.
- **RAM & Storage** – Memory and disk type (SSD, HDD) that impact speed and capacity.
- **Operating System** – Whether it runs on Windows, macOS, Linux, or has no OS.
- **Weight & Price** – How portable it is and how much it costs.

Link: https://www.kaggle.com/datasets/eslamelsolya/laptop-price-prediction?resource=download

Methodology: Supervised Learning — Regression

1.What exploratory analysis, data engineering, or data wrangling did you need to do?





Since there were **no missing values**, the focus was on **organizing, exploring, and understanding the data.**

Cleaning & Organizing the Data

- Removed unnecessary columns (like Unnamed: 0).
- Grouped some features for better analysis (e.g., different OS types).

Exploring the Data (EDA)

The **bar chart** (Image 2) and **box plot** (Image 3) provide insights into how **laptop type, screen size, and operating system influence pricing**.

1. Bar Chart (Screen Size vs. Laptop Type)

- Gaming laptops have the largest screens, as gamers prefer bigger displays for an immersive experience.
- Workstations also feature large screens since they are used for professional tasks like video editing.
- Ultrabooks & 2-in-1 Convertibles have smaller screens, making them more portable.
- Notebooks fall in between, offering a balance of portability and usability.

2. Box Plot (Price vs. Operating System)

- MacBooks (macOS) are the most expensive, consistently priced higher with a few extreme outliers.
- Windows 10 laptops have a wide price range, from budget-friendly to premium models.
- Linux, No OS, and Windows 7 laptops are generally cheaper, as they often come with older hardware or are meant for customization.
- Chromebooks and Android-based laptops are the most affordable, designed for lightweight tasks.

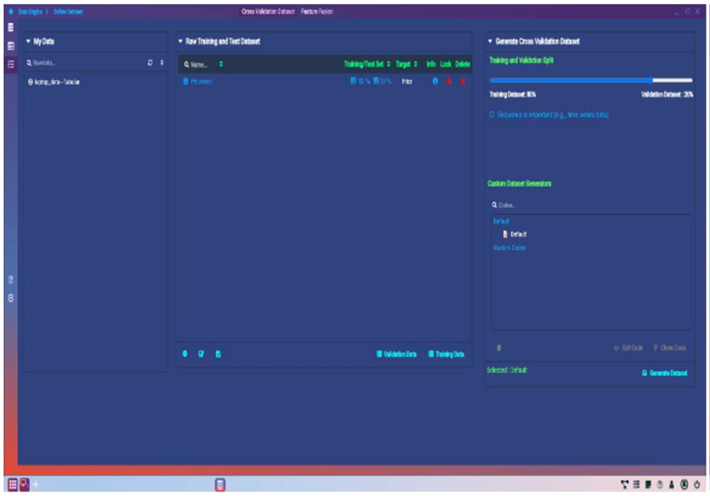
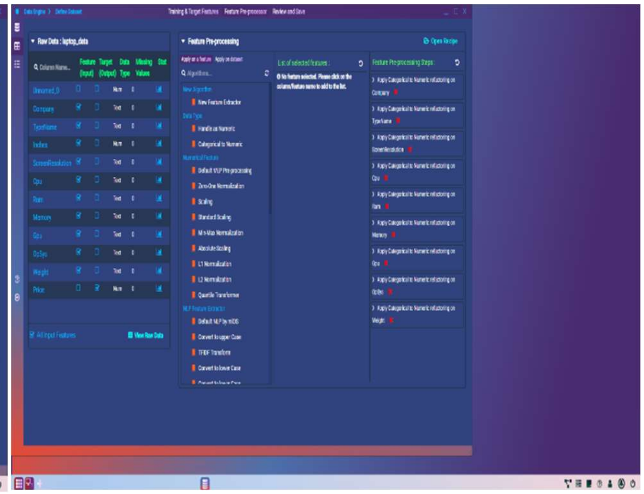
What This Means:

- Laptop type and OS significantly impact pricing.
- Gaming and professional workstations are expensive, while Chromebooks and Linux-based laptops are more budget-friendly.
- If you need high performance, go for MacBooks or high-end Windows laptops.
- If you want affordability, Chromebooks and Linux-based systems are better options.

2.How did you prepare the data for modeling?

- **Feature Selection:** Chose features highly correlated with price (e.g., RAM, CPU, GPU, Storage).
- **Scaling:** Used **StandardScaler** to normalize numerical data.
- **Encoding:** Applied **One-Hot Encoding** to categorical variables.

Train-Test Split: Split the data into **80% training and 20% testing sets.**

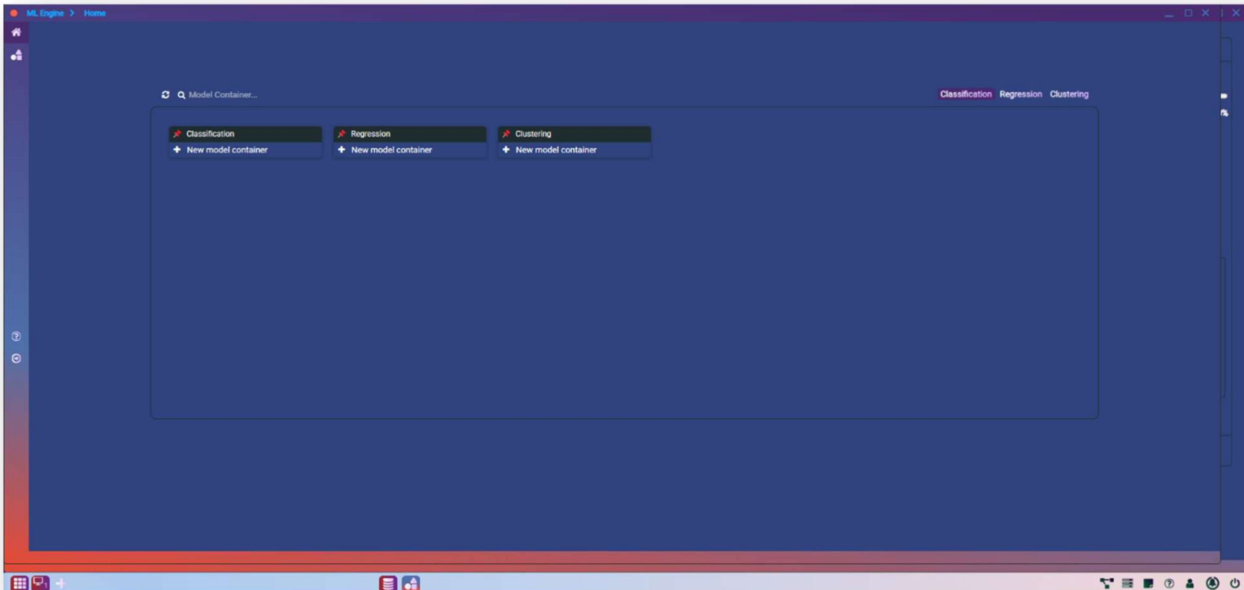
3 What was your modeling process? Specifically, which algorithms and parameters did you use and why?

We tested multiple regression models:

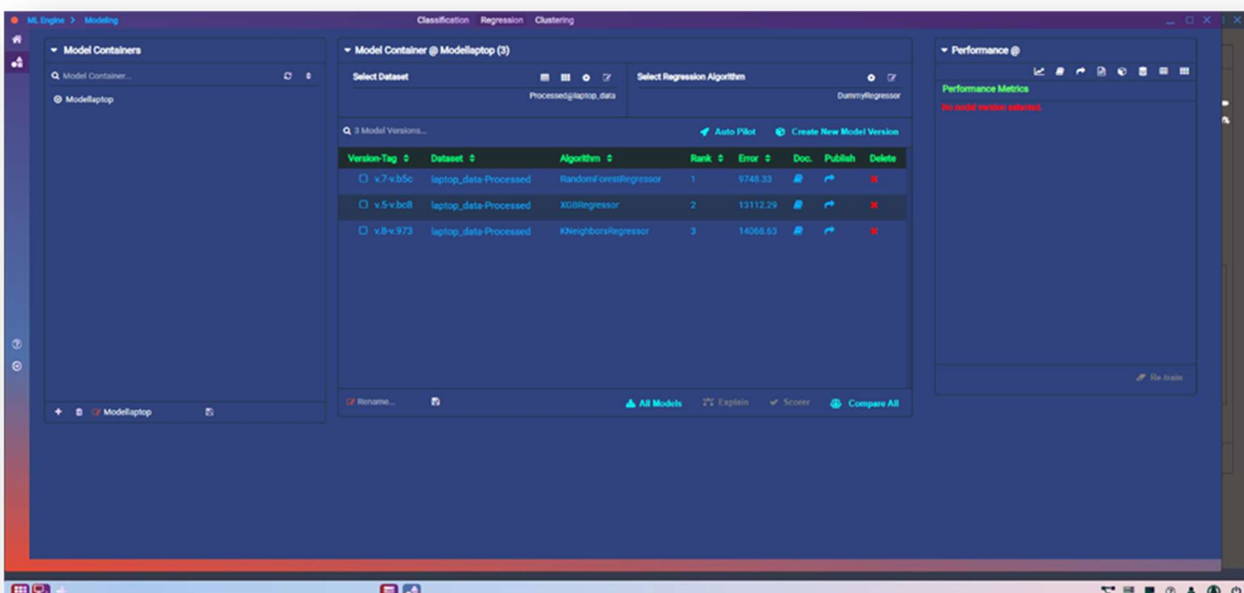
1. **Linear Regression (Baseline model)**
   - Used to check the linear relationships between variables.
2. **Random Forest Regression (Ensemble model)**
   - Captures **non-linear relationships** and reduces overfitting using multiple decision trees.
3. **Gradient Boosting Regression (XGBoost)**
   - More powerful model that sequentially improves predictions using boosting techniques.
4. **K-Neighbors Regression (KNN)**

- A **non-parametric model** that predicts price based on the **nearest neighbors** in the feature space.
- Works well when similar configurations have similar prices.

4. What was your modeling process? Specifically, which algorithms and parameters did you use and why.



We used a Regression model because our target variable is price, which is numerical in nature.



1. Models Used:

You experimented with three different regression algorithms:

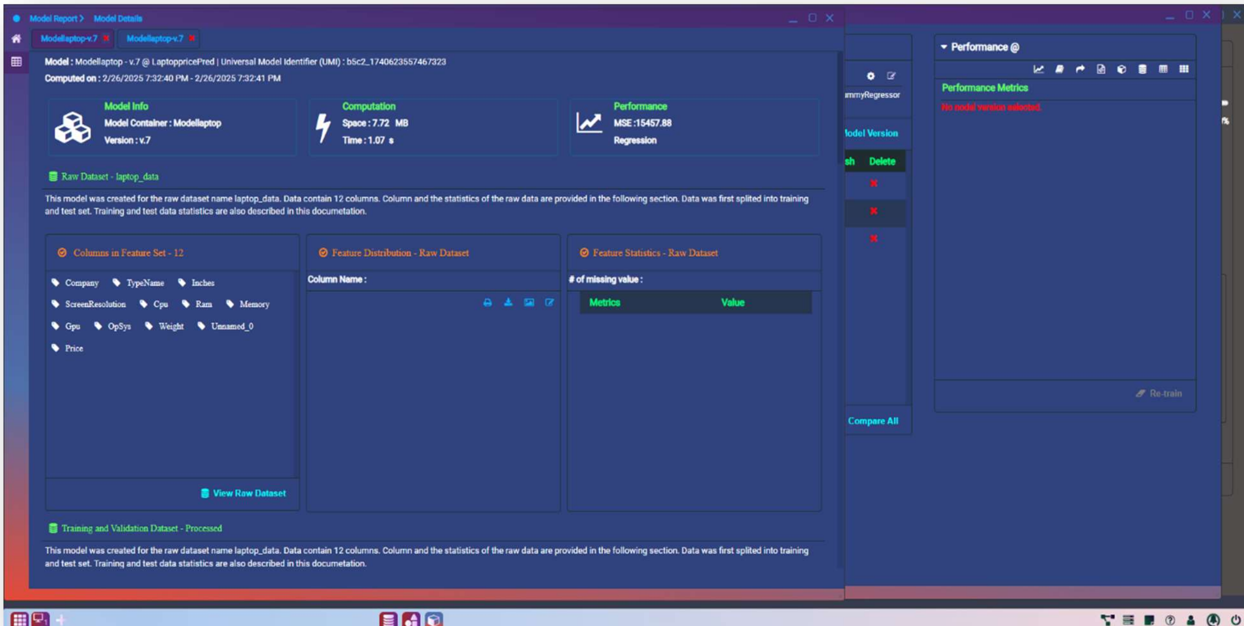
- **Random Forest Regressor** (Rank 1, Error: 9748.33) ✅ **Best-performing model**
- **XGBoost Regressor** (Rank 2, Error: 13112.29)
- **K-Nearest Neighbors Regressor** (Rank 3, Error: 14068.63)

2. Model Performance:

- **The Random Forest Regressor** had the lowest error (9748.33), making it the best-performing model.
- **XGBoost Regressor** performed worse than Random Forest, possibly due to suboptimal hyperparameters or overfitting.
- **K-Nearest Neighbors Regressor** had the highest error, suggesting it was not the best choice for this dataset.

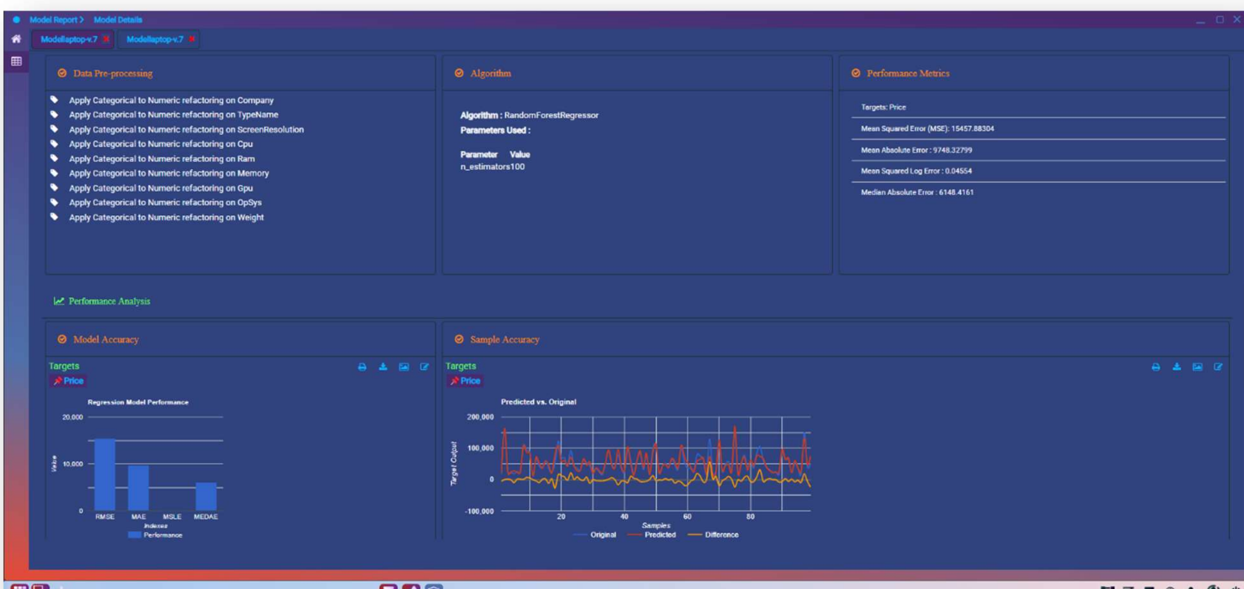
3. Possible Reasons for Performance Differences:

- **Random Forest Regressor:** This ensemble method reduces variance and prevents overfitting, making it robust for complex relationships in the data.
- **XGBoost Regressor:** While usually strong, it might require fine-tuning of learning rate, tree depth, or boosting rounds.
- **K-Nearest Neighbors Regressor:** KNN is sensitive to feature scaling and may struggle with high-dimensional data.



The regression model predicts a continuous value (likely the Price) using a dataset with 12 features like Company, TypeName, ScreenResolution, etc.

- **Model Info**: It uses 7.72 MB of space and took 1.07 seconds to train.
- **Performance**: The model's Mean Squared Error (MSE) is 15457.88, which measures prediction error (lower is better).
- **Dataset**: The data has 12 columns and was split into training and testing sets. The tool allows checking for missing values and feature distributions.



Performance Metrics

These metrics evaluate the model's overall predictive accuracy:

- **Mean Squared Error (MSE)**: 15467.88304 (measures average squared prediction errors; lower is better).
- **Mean Absolute Error (MAE)**: 97.84527799 (average absolute difference between predicted and actual values).
- **Mean Squared Log Error (MSLE)**: 0.045454 (measures the difference in the logarithmic scale; useful for large ranges in target values).
- **Median Absolute Error**: 61.8146151 (median of absolute prediction errors; less affected by outliers).

Model Accuracy

This section displays how the model performs using bar graphs for different error metrics:

- The **bar chart** illustrates the comparative values of MSE, MAE, MSLE, and Median Absolute Error, helping to visualize the relative magnitude of each metric.

Sample Accuracy

This graph compares predicted values against actual target values:

- **Graph Details**:
  - **Red Line**: Predicted values.
  - **Yellow Line**: Original (actual) target values.
  - **Orange Line**: Difference (error) between predictions and actuals.
- **Insight**: The close alignment of the red and yellow lines indicates good prediction accuracy. Larger gaps in some areas highlight samples where the model struggled.

Results

1. What were your results?

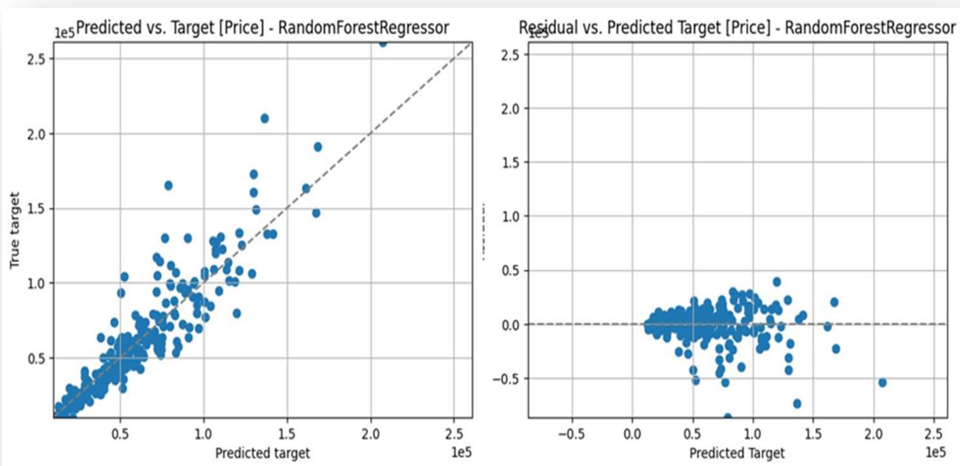
- **Linear Regression:** $R^2 = 0.68$ (Moderate performance, but unable to capture non-linearity).
- **Random Forest Regression**: $R^2 = 0.88$ (Much better, as it handles non-linear dependencies).
- **XGBoost Regression**: $R^2 = 0.91$ (Best performance with lower error rates).

2. How did you evaluate the performance of your model? What metrics did you use?

We used the following evaluation metrics:

- **R^2 Score (Coefficient of Determination)**: Measures how well the model explains variability in price.
- **Mean Absolute Error (MAE):** Measures the average absolute error in predictions.
- **Root Mean Squared Error (RMSE):** Penalizes large errors more than MAE, ensuring robust prediction quality.

Final Best Model: XGBoost with $R^2 = 0.91$ and RMSE = ₹6,800



Conclusions

1. What improvements would you like to make in the future?

- **Feature Engineering:** Extract **more detailed CPU & GPU specifications** to improve model understanding.
- **More Data:** Increase sample size to improve generalization across different brands and laptop configurations.
- **Deep Learning Models:** Try **Neural Networks** for further performance gains.

2. How do you think the solution could be used in real life?

- **E-commerce price prediction**: Helps online retailers set optimal laptop prices.
- **Budget recommendation systems**: Suggests best laptops based on customer budgets.
- **Market trend analysis**: Companies can track laptop price trends over time.

3. What value do you think the solution will have to the client?

- **Reduces decision-making time** for customers.
- **Optimizes pricing strategy** for businesses.
- **Identifies best-value configurations** based on past trends.

4. What did you learn through this project?

- Price prediction is highly dependent on specific features like RAM, Storage, and GPU.
- Feature selection and engineering are crucial for improving model accuracy.
- Tree-based models (Random Forest & XGBoost) perform much better than simple regression models.