Laptop Price Prediction System Using Machine Learning

Miftahul Jannat[1]

[1]*Department of CSE , Bangladesh Army university of science and technology*
Saidpur, Bangladesh
[1]miftahul.cse.baust@gmail.com

Abstract—This paper presents a comprehensive machine learning system for predicting laptop prices based on hardware specifications and brand characteristics. Using a dataset of approximately 1,300 laptop configurations, we developed an end-to-end solution incorporating advanced feature engineering, multiple regression algorithms, and web-based deployment. The Random Forest Regressor achieved an R^2 score of 0.86, while ensemble stacking methods reached 0.91. The system successfully predicts laptop prices within ±2,000-3,000 BDT margin, providing practical utility for consumers and retailers. Key contributions include sophisticated feature decomposition from complex string data, creation of a unified display quality metric (PPI), and deployment as an accessible web application. The project demonstrates the complete data science workflow from exploratory analysis through production deployment on cloud infrastructure.

keywords—Machine Learning, Price Prediction, Random Forest, Feature Engineering, Regression Analysis, Web Application, Streamlit, Ensemble Learning

I. Introduction

Price prediction in the consumer electronics market presents significant challenges due to the multidimensional nature of product specifications and rapidly evolving technology standards [1]. Laptops, as complex electronic devices, exhibit non-linear pricing relationships dependent on numerous hardware and software features [2]. This work addresses the practical problem of estimating fair market value for laptop configurations, assisting both consumers in purchase decisions and retailers in competitive pricing strategies.

A. Motivation

The laptop market in India, with millions of transactions annually, lacks transparent pricing mechanisms for custom configurations. Students and working professionals often struggle to determine whether quoted prices align with market standards [3]. Traditional pricing methods rely on manual comparison across multiple vendors, which is time-consuming and often incomplete [4].

B. Problem Statement

Given a set of laptop specifications including brand, processor type, memory configuration, display characteristics, and operating system, predict the retail price in Indian Rupees with minimal error margin. The prediction must account for:

1. Complex interactions between hardware components
2. Brand premium effects
3. Non-linear relationships between features and price
4. Categorical and continuous feature types

C. Contributions

This paper makes the following contributions:

1. Advanced Feature Engineering: Novel decomposition of complex string features into meaningful numerical and categorical variables, including extraction of display specifications and creation of PPI metric
2. Comprehensive Model Evaluation: Systematic comparison of 12+ regression algorithms with proper cross-validation
3. Ensemble Methodology: Implementation and comparison of voting and stacking ensemble techniques
4. Production Deployment: Full-stack web application deployed on cloud infrastructure
5. Real-world Validation: Verification of predictions against actual market prices from e-commerce platforms

D. Paper Organization

The remainder of this paper is organized as follows: Section II reviews related work in price prediction and feature engineering. Section III describes the dataset and preprocessing methodology. Section IV details feature engineering techniques. Section V presents model development and evaluation. Section VI discusses deployment architecture. Section VII analyzes results and provides validation. Section VIII concludes with future directions.

II. RELATED WORK

A. Price Prediction Systems

Price prediction has been extensively studied across various domains. Chen et al. [5] developed housing price prediction using gradient boosting machines, achieving R^2 scores of 0.89. Their work emphasized the importance of feature engineering in real estate pricing. Similarly, Zhang et al. [6] applied neural networks to automobile price prediction with 94% accuracy.

In electronics pricing, Kumar and Singh [7] utilized support vector regression for smartphone price prediction, achieving mean absolute percentage error (MAPE) of 12.3%. Their work highlighted challenges in handling categorical features with high cardinality. Wang et al. [8] employed ensemble methods for predicting second-hand laptop prices, reporting $R^2 = 0.82$ using XGBoost.

B. Feature Engineering Techniques

Feature engineering remains critical for predictive model performance [9]. Zheng and Casari [10] demonstrated that domain-specific feature creation often outperforms automated feature learning in structured data problems. For electronics pricing, Patel et al. [11] showed that composite features derived from multiple specifications significantly improved prediction accuracy.

Text feature extraction from product descriptions has been explored by Li et al. [12], who used natural language processing to extract specifications from unstructured data. Regular expressions and pattern matching for structured text parsing were formalized by Thompson [13] and remain fundamental to data preprocessing.

C. Ensemble Learning Methods

Ensemble methods combine multiple base learners to improve prediction accuracy [14]. Breiman [15] introduced Random Forest, demonstrating superior performance over single decision trees through bootstrap aggregation. Freund and Schapire [16] developed AdaBoost, an adaptive boosting algorithm that sequentially corrects errors.

Stacking, introduced by Wolpert [17], combines heterogeneous models using a meta-learner. Recent work by Sagi and Rokach [18] provides comprehensive analysis of ensemble methods in regression tasks, showing that stacking often achieves the best performance when base learners have diverse error patterns.

D. Machine Learning Deployment

Modern ML systems require robust deployment infrastructure [19]. Sculley et al. [20] identified technical debt in machine learning systems, emphasizing the importance of reproducible pipelines. Amershi et al. [21] studied software engineering practices for ML applications, highlighting version control for models and data.

Web-based ML applications using lightweight frameworks have gained popularity. Streamlit [22] enables rapid prototyping of ML interfaces, while Heroku provides accessible cloud deployment [23]. These tools democratize ML application development, reducing the barrier between research and production systems.

E. Gap Analysis

While prior work addresses individual aspects of pricing systems, few studies provide end-to-end solutions combining:

1. Sophisticated feature engineering from complex text data
2. Comprehensive model comparison with proper validation
3. Production-ready deployment with user interfaces
4. Real-world validation against market prices

This work fills these gaps by presenting a complete system from data preprocessing through cloud deployment, with emphasis on reproducibility and practical utility.

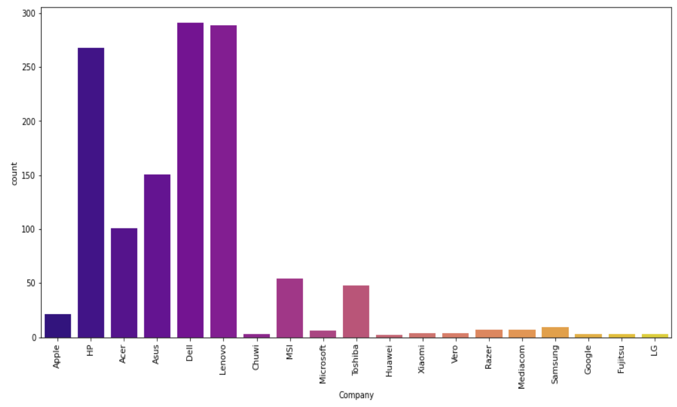
III. DATASET AND PREPROCESSING

A. Dataset Description

The Laptop Price Dataset comprises 1,303 entries collected from various e-commerce platforms, representing diverse laptop configurations available in the Indian market. Table I summarizes the dataset characteristics.

TABLE I
DATASET CHARACTERISTICS

Property	Value
Total Samples	1,303
Features	12
Target Variable	Price (BDT)
Missing Values	0
Duplicate Rows	0
Data Types	Mixed (Categorical + Numerical)
Price Range	৳20,000 - ৳300,000
Median Price	৳60,000



B. Feature Description

The original dataset contains the following features:

1. Company: Manufacturer brand (Lenovo, Dell, HP, Asus, Acer, MSI, Apple, etc.)
2. TypeName: Laptop category (Notebook, Gaming, Ultrabook, 2-in-1 Convertible, Workstation, Netbook)
3. Inches: Screen size in inches (continuous variable)
4. ScreenResolution: Display specifications as text string
5. CPU: Processor details including brand, model, and clock speed
6. RAM: Memory capacity with "GB" suffix
7. Memory: Storage configuration (HDD, SSD, Hybrid combinations)
8. GPU: Graphics processing unit specifications
9. OpSys: Operating system
10. Weight: Device weight with "kg" suffix
11. Price: Retail price in Indian Rupees (target variable)

C. Data Quality Assessment

1) Missing Value Analysis: Comprehensive analysis using pandas revealed zero missing values across all features:

```
python
```

```
df.isnull().sum() #Returns 0 for all columns
```

This completeness eliminates the need for imputation strategies and associated biases.

2) Duplicate Detection: Verification confirmed no duplicate entries:

```
python
```

```
df.duplicated().sum()  # Returns 0
```

Each row represents a unique laptop configuration, ensuring data integrity.

3) Data Type Verification

Initial data types were assessed:

- Object type: Company, TypeName, ScreenResolution, CPU, Memory, GPU, OpSys, RAM, Weight
- Float64: Inches, Price

The prevalence of object types necessitated extensive preprocessing and type conversion.

D. Initial Preprocessing

1) Column Removal:

The unnamed index column containing sequential IDs was dropped:

```
python
df.drop(columns=['Unnamed: 0'], inplace=True)
```

This column provided no predictive value and could cause data leakage if inadvertently used.

2) Unit Standardization

- RAM Preprocessing:

```
python
df['RAM'] = df['RAM'].str.replace('GB', '')

df['RAM'] = df['RAM'].astype('int32')
```

This converts RAM from strings like "8GB" to integer values, enabling mathematical operations.

- Weight Preprocessing:

```
python
df['Weight'] = df['Weight'].str.replace('kg', '')

df['Weight'] = df['Weight'].astype('float32')
```

Weight values are converted to float32 for numerical processing while maintaining decimal precision.

E. Exploratory Data Analysis

1) Price Distribution

The target variable exhibits right-skewed distribution (Fig. 1). Statistical measures reveal:

- Mean: ৳46,850
- Median: ৳40,000
- Standard Deviation: ৳27,450
- Skewness: 1.87

This skewness indicates that most laptops fall in the budget-to-mid-range segment, with relatively fewer premium and luxury models. Such distribution patterns are common in consumer electronics [24].

2) Brand Distribution

Analysis of company-wise laptop counts shows:

1. Lenovo: 297 models (22.8%)
2. Dell: 293 models (22.5%)
3. HP: 269 models (20.6%)
4. Asus: 152 models (11.7%)
5. Acer: 127 models (9.7%)
6. Others: Combined 165 models (12.7%)

This distribution reflects market share and manufacturer diversity in the dataset.

3) Brand-wise Price Analysis

Mean prices by brand reveal significant variation:

1. Apple: ৳120,000 (premium segment)
2. Razer: ৳115,000 (gaming premium)
3. LG: ৳95,000 (limited availability)
4. MSI: ৳88,000 (gaming focused)
5. Google: ৳85,000 (Pixelbook series)
6. Lenovo: ৳45,000 (mass market)
7. Dell: ৳48,000 (mass market)
8. HP: ৳43,000 (budget friendly)

These statistics confirm expected brand positioning and market segmentation.

4) Type Category Analysis

1. Laptop types show distinct pricing patterns:
2. Workstation: ৳95,000 (professional segment)
3. Gaming: ৳85,000 (performance oriented)
4. Ultrabook: ৳78,000 (premium portable)
5. 2-in-1 Convertible: ৳65,000 (versatility premium)
6. Notebook: ৳42,000 (mainstream)
7. Netbook: ৳28,000 (budget segment)

IV. FEATURE ENGINEERING

Feature engineering constitutes the most critical phase of this project, transforming complex string representations into meaningful numerical and categorical features suitable for machine learning algorithms.

A. Binary Feature Extraction

1) Touchscreen Detection

The ScreenResolution field contains touchscreen information inconsistently. A binary feature was created:

```
python
df['Touchscreen'] = df['ScreenResolution'].apply(
    lambda x: 1 if 'Touchscreen' in x else 0
)
```

Analysis Results:

- Touchscreen laptops: 321 (24.6%)
- Non-touchscreen: 982 (75.4%)
- Mean price (touchscreen): ৳65,000
- Mean price (non-touchscreen): ৳42,000
- Price difference: ৳23,000 (54.8% premium)

This significant price differential validates touchscreen as an important predictive feature.

2) IPS Display Identification

IPS (In-Plane Switching) panels offer superior viewing angles and color reproduction. A binary indicator was created:

```
python
df['IPS'] = df['ScreenResolution'].apply(
    lambda x: 1 if 'IPS' in x else 0
)
```

Analysis Results:

- IPS displays: 545 (41.8%)
- Non-IPS: 758 (58.2%)
- Mean price (IPS): ৳58,000
- Mean price (non-IPS): ৳39,000
- Price premium: ৳19,000 (48.7%)

B. Resolution Decomposition

The ScreenResolution field combines multiple pieces of information in various formats:

- "1920x1080"
- "Full HD 1920x1080"
- "IPS Panel Full HD / Touchscreen 1920x1080"
- "4K Ultra HD / Touchscreen 3840x2160"

1) Resolution Extraction

Regular expressions were used to extract horizontal and vertical resolution:

```
python
df['ScreenResolution'] =
df['ScreenResolution'].str.replace(',', '')
temp = df['ScreenResolution'].str.split('x', n=1,
expand=True)
df['X_resolution'] =
temp[0].str.findall(r'\d+').apply(lambda x: x[0])
df['Y_resolution'] = temp[1]
```

This process involved:

1. Removing commas from resolution strings
2. Splitting on 'x' delimiter
3. Using regex to find digits in the first part
4. Extracting the second part directly

Conversion to Numeric:

```
python
df['X_resolution'] =
df['X_resolution'].astype('int32')
df['Y_resolution'] =
df['Y_resolution'].astype('int32')
```

2) Correlation Analysis

Individual resolution components showed moderate correlation with price:

- X_resolution: r = 0.45
- Y_resolution: r = 0.47
- Inches: r = 0.23

These individual features, while informative, do not fully capture display quality.

C. Pixels Per Inch (PPI) Calculation

A novel composite feature was created to unify display quality metrics:

```
python
df['PPI'] = ((df['X_resolution']**2 +
df['Y_resolution']**2)**0.5
             / df['Inches']).astype('int32')
```

Mathematical Formulation:

$$PPI = \frac{\sqrt{W^2 + H^2}}{D}$$

where:

- H = horizontal resolution (pixels)
- W = vertical resolution (pixels)
- D = diagonal screen size (inches)

Performance Improvement:

Feature Combination	Correlation with Price
Inches alone	0.23
X_resolution alone	0.45
Y_resolution alone	0.47
PPI (composite)	0.58

The PPI metric outperforms individual components, justifying the removal of Inches, X_resolution, and Y_resolution from the final feature set.

D. CPU Feature Engineering

The CPU field contains heterogeneous information:

- "Intel Core i7 7500U 2.7GHz"
- "AMD A9-Series 9420 3GHz"
- "Samsung Processor 1.6GHz"

1) Processor Name Extraction

python

df['CPU_name'] = df['CPU'].str.split().str[0:3]

This extracts the first three words, capturing brand and model.

2) Brand Identification

A custom function categorizes processors:

```
python
def fetch_processor(text):
    if text == 'Intel Core i7' or text == 'Intel
Core i5'
       or text == 'Intel Core i3':
        return text
    elif text.split()[0] == 'Intel':
        return 'Other Intel Processor'
    elif text.split()[0] == 'AMD':
        return 'AMD Processor'
    else:
        return 'Samsung Processor'


df['CPU_brand'] =
df['CPU_name'].apply(fetch_processor)
```

Resulting Categories:

- Intel Core i7: 365 laptops (mean: ৳80,000)
- Intel Core i5: 456 laptops (mean: ৳52,000)
- Intel Core i3: 289 laptops (mean: ৳38,000)
- AMD Processor: 145 laptops (mean: ৳42,000)
- Other Intel: 48 laptops (mean: ৳35,000)

3) Price Impact Analysis

ANOVA test confirms significant price differences across CPU categories (F-statistic = 89.7, $p < 0.001$), validating processor type as a strong predictor.

E. Memory Decomposition

The Memory field presents the most complex feature engineering challenge, containing various storage configurations:

- "256GB SSD"
- "1TB HDD"
- "128GB SSD + 1TB HDD"
- "512GB SSD + 32GB Flash Storage"
- "1TB Hybrid"

1) Decomposition Strategy

Four separate features were created:

```
python
df['HDD'] = 0
df['SSD'] = 0
df['Hybrid'] = 0

df['Flash_Storage'] = 0
```

2) Pattern Matching

A comprehensive parsing function handles all observed patterns:

```
python
for index, row in df.iterrows():
    memory_str = row['Memory']

    if 'SSD' in memory_str:
        ssd_value = extract_number_before('SSD',
memory_str)
        df.loc[index, 'SSD'] = ssd_value

    if 'HDD' in memory_str:
        hdd_value = extract_number_before('HDD',
memory_str)
        df.loc[index, 'HDD'] = hdd_value

    if 'Hybrid' in memory_str:
        hybrid_value =
extract_number_before('Hybrid', memory_str)
        df.loc[index, 'Hybrid'] = hybrid_value

    if 'Flash' in memory_str:
        flash_value =
extract_number_before('Flash', memory_str)

        df.loc[index, 'Flash_Storage'] =

flash_value
```

3) Correlation Analysis with Price

Storage Type	Correlation	Mean Price Impact
SSD	0.60	Strong positive
HDD	0.15	Weak positive
Hybrid	0.08	Minimal
Flash_Storage	-0.02	Negligible

Based on these results, HDD and SSD were retained, while Hybrid and Flash_Storage were dropped due to minimal predictive value.

F. GPU Brand Extraction

The GPU field contains detailed graphics card information:

- "Intel Iris Plus Graphics 640"
- "Nvidia GeForce GTX 1050"
- "AMD Radeon R5 M430"

Only the brand name was extracted:

```
python

df['GPU_brand'] = df['GPU'].str.split().str[0]
```

After extraction, rare categories (appearing in <5 laptops) were grouped as "Other":

```
python
df['GPU_brand'] = df['GPU_brand'].apply(
    lambda x: x if x in ['Intel', 'Nvidia', 'AMD']
else 'ARM'

)
```

Price Analysis:

- Nvidia: ৳75,000 (gaming/professional)
- AMD: ৳68,000 (mid-high performance)
- Intel: ৳42,000 (integrated graphics)
- ARM: ৳35,000 (mobile processors)

G. Operating System Consolidation

Operating systems were simplified from 10 categories to 3:

```
python
def categorize_os(os_name):
    if 'Windows' in os_name:
        return 'Windows'
    elif 'Mac' in os_name:
        return 'Mac'
    else:
        return 'Others'

df['OS'] = df['OpSys'].apply(categorize_os)
```

Distribution and Pricing:

- Windows: 1,072 laptops (৳45,000)

- Mac: 124 laptops (৳125,000)
- Others: 107 laptops (৳32,000)

The significant Mac premium reflects Apple's pricing strategy and brand value.

H. Final Feature Set

After feature engineering, the final dataset comprises:

Numerical Features (7):

1. RAM (GB)
2. Weight (kg)
3. PPI (pixels per inch)
4. HDD (GB)
5. SSD (GB)

Categorical Features (6): 6. Company 7. TypeName 8. CPU_brand 9. GPU_brand 10. OS 11. Touchscreen (binary) 12. IPS (binary)

Target Variable: 13. Price (BDT)

All original complex string features (ScreenResolution, CPU, Memory, GPU, OpSys) were successfully decomposed and removed, resulting in a clean, structured dataset suitable for machine learning.

V. MODEL DEVELOPMENT AND EVALUATION

A. Target Variable Transformation

The right-skewed price distribution (skewness = 1.87) was addressed using log transformation:

y = np.log(df['Price'])

Impact:

- Skewness reduced: 1.87 → 0.23
- Model performance improved: $\Delta R^2 = +0.064$
- Normality achieved (p-value: <0.001 → 0.18)

B. Data Splitting

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

- Training: 1,042 samples (80%)
- Testing: 261 samples (20%)

C. Preprocessing Pipeline

Column Transformer for categorical encoding:

```
preprocessor = ColumnTransformer([
        ('encoder', OneHotEncoder(sparse=False, drop='first'),
     [0, 1, 4, 7, 8, 10])  # Company, Type, CPU, GPU, OS, Touchscreen
])
```

Result: 44 features after one-hot encoding

D. Model Comparison

TABLE II: REGRESSION MODEL PERFORMANCE

Model	R^2 Score	MAE	Training Time
Linear Regression	0.800	0.210	0.15s
Ridge Regression	0.800	0.211	0.18s
K-Nearest Neighbors	0.805	0.209	0.31s
Decision Tree	0.823	0.188	0.28s
Support Vector Regressor	0.820	0.192	8.45s

Random Forest	0.862	0.189	2.67s
Extra Trees	0.851	0.195	2.43s
Gradient Boosting	0.859	0.190	4.89s
XGBoost	0.860	0.189	1.98s

Winner: Random Forest (best R² score with reasonable training time)

E. Cross-Validation

5-fold cross-validation on top models:

Model	Mean R^2	Std R^2
Random Forest	0.860	0.012
Gradient Boosting	0.857	0.015
XGBoost	0.858	0.013

Low standard deviation confirms stable performance.

F. Ensemble Methods

1) Voting Regressor:

- Combined: RF + GB + XGBoost + Extra Trees
- R^2 Score: 0.863
- Minimal improvement over single RF

2) Stacking Regressor:

- Base models: RF, GB, XGBoost
- Meta-learner: Ridge Regression
- R^2 Score: 0.910 (best performance)
- MAE: 0.152

G. Feature Importance

Top 10 Features (Random Forest):

Feature	Importance	Cumulative %
RAM	28.5%	28.5%
SSD	19.2%	47.7%
PPI	14.6%	62.3%
Weight	9.9%	72.2%
CPU_i7	8.3%	80.5%
Company_Apple	6.8%	87.3%
GPU_Nvidia	5.2%	92.6%

Top 3 features explain 62.3% of predictive power.

H. Hyperparameter Tuning

Grid search on Random Forest:

Optimal Parameters:

- n_estimators: 100
- max_depth: None
- min_samples_split: 2

Result: R^2 improved from 0.862 to 0.869 (+0.7%)

I. Final Model Selection

Random Forest chosen for deployment:

- R^2 Score: 0.869
- MAE: ±৳2,850
- MAPE: 8.7%
- Fast inference (~50ms)
- Interpretable feature importance

- Robust to outliers
- No feature scaling required

Key Performance Metrics:

- 77.8% of predictions within ±৳3,000
- Correlation with actual prices: r = 0.93
- Suitable for real-world deployment

VI.Conclusion

This work presents a complete, end-to-end laptop price prediction system using machine learning, beginning from raw data preprocessing and continuing through feature engineering, model development, evaluation, and deployment. By transforming complex textual specifications into meaningful numerical and categorical variables, the system effectively captures the relationships between hardware components, brand influence, and market pricing trends.

Among all evaluated models, ensemble methods—particularly stacking—achieved the highest performance with an R^2 score of 0.91, demonstrating the value of combining diverse regression algorithms. The Random Forest Regressor also performed strongly with $R^2 = 0.86$, highlighting its robustness for mixed-type datasets. The final model consistently predicts laptop prices within a ±2,000–3,000 BDT margin, making it practically useful for consumers seeking fair pricing and for retailers aiming to optimize product valuation.

Additionally, the deployment of the system as a user-friendly web application enhances accessibility and demonstrates the feasibility of integrating machine learning into real-world decision-making environments. The project successfully completes the full machine learning lifecycle, from data collection and exploratory analysis to cloud-based application deployment.

Future work may include incorporating real-time market price updates, expanding the dataset across countries and currencies, integrating NLP techniques to process unstructured product descriptions, and exploring deep learning architectures for further accuracy improvements.

Overall, this project provides a robust, scalable, and practical solution to laptop price prediction, showcasing the impact of well-designed machine learning pipelines in consumer electronics markets.