LAPTOP PRICE PREDICTION

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

DATA SCIENCE WITH ML (UPGRAD)

Submitted by:

Muhammed Althaf K

12205672

Submitted to:

Karan Kumar Das

Subject Code: CSM355

Section: K22UG

Supervisor Certificate

This is to certify that the project report titled “Laptop Price Prediction” is a record of work carried out by Muhammed Althaf K under my guidance.

The work has been carried out as a part of the academic requirements for the course *ML Project* offered at Lovely Professional University and UpGrad.

Mr. Karan Kumar Das

Signature:

Acknowledgement

I would like to acknowledge my mentor Mr. Karan Kumar Das for his invaluable guidance and support during this project on Laptop Price Prediction. This project focuses on analyzing various laptop specifications to predict their market prices, using a real-world dataset that includes features such as processor type, RAM size, storage capacity, brand, operating system, and display characteristics.

I, Muhammed Althaf K, am grateful for the mentorship and assistance provided by my teacher throughout the course of this project.

Table Of Contents:

1. Abstract

In the modern digital age, laptops serve a vital role in professional, academic, and personal spheres. With a vast array of models available in the market, predicting the price of a laptop based on its specifications is a complex yet valuable task. This project focuses on developing a machine learning-based predictive system to estimate laptop prices using structured data containing features like brand, processor type, RAM, storage, screen characteristics, GPU, and operating system.

The project begins with comprehensive data preprocessing, including cleaning inconsistencies, handling missing values, converting categorical variables, and engineering new features. Derived variables such as pixels per inch (PPI), presence of touchscreen and IPS technology, and resolution-based metrics are included to enhance the model's predictive power.

Exploratory Data Analysis (EDA) uncovers key patterns and correlations. Features like RAM size, display quality (PPI), processor type, and brand significantly influence laptop pricing. Insights derived from visualizations helped guide the feature selection and transformation process.

Three machine learning models were applied and compared: Linear Regression, Random Forest Regressor, and K-Nearest Neighbors (KNN). These models were evaluated using R^2 score to measure goodness-of-fit, and Mean Absolute Error (MAE) to assess prediction accuracy. Random Forest performed particularly well, balancing bias and variance effectively, while KNN and Linear Regression provided baseline comparisons.

The final model delivers robust predictions and highlights the most influential features in determining laptop prices. The project concludes with a discussion on model performance, key findings, and recommendations for future improvements, such as expanding the dataset, incorporating deep learning methods, and deploying the model through a user-friendly interface.

2. Introduction

Laptops have become indispensable in the modern world, serving a broad spectrum of users—from students and working professionals to gamers and content creators. As a result, the laptop market is flooded with diverse models, each offering different specifications and performance capabilities. These variations create a challenge in pricing, as even small differences in hardware or brand reputation can significantly impact the cost. For consumers, this can lead to confusion, and for sellers or manufacturers, mispricing can result in lost sales or diminished profits.

Traditionally, laptop pricing is determined by manual market comparisons, brand positioning, and feature listings. However, this approach is subjective and may not always reflect true market trends or customer preferences. With the growing availability of structured product data and advancements in machine learning, it is now feasible to automate and optimize the pricing process through predictive modeling.

This project, titled **Laptop Price Prediction**, aims to build a machine learning model that can accurately predict the price of a laptop based on its technical specifications. The goal is to leverage data science to uncover complex relationships between features such as RAM, processor type, storage configuration, screen resolution, and GPU, and their impact on the final price.

By using historical data and implementing regression algorithms, the project attempts to learn these patterns and apply them to new, unseen data. Beyond predicting prices, the project also serves as an exercise in end-to-end machine learning workflow: from data acquisition and cleaning to feature engineering, model evaluation, and interpretation of results.

Ultimately, the project provides a practical solution for use cases in e-commerce, retail pricing, and digital platforms where price estimation is vital. It also demonstrates the power of machine learning in solving real-world problems with measurable and scalable outcomes.

3. Literature Review

Machine learning has increasingly become a powerful tool in predictive analytics, particularly in domains involving structured numerical and categorical data. The problem of price prediction—whether in real estate, automobiles, or consumer electronics—has been well explored in recent years, with multiple studies highlighting different modeling techniques and their relative effectiveness.

In the domain of electronic goods pricing, several research efforts have employed regression-based approaches due to their interpretability and suitability for continuous output variables. Linear Regression (LR) is among the most widely used methods in early studies, often serving as a baseline. While simple and fast, LR can struggle with multicollinearity and non-linear relationships, which are common in real-world datasets.

To overcome these limitations, ensemble methods such as Random Forest (RF) have been introduced. Random Forests are particularly effective in capturing complex interactions between features and are robust to overfitting due to their use of bootstrapped aggregation (bagging). Multiple studies in real estate and e-commerce domains have shown RF outperforming linear models, particularly in datasets with mixed types of features (e.g., numeric and categorical).

K-Nearest Neighbors (KNN) regression has also been explored in similar applications due to its non-parametric nature. KNN predicts values based on the proximity of data points in feature space, which can be intuitive but computationally expensive and sensitive to the choice of distance metric and feature scaling. It performs well in lower-dimensional datasets but may suffer in high-dimensional spaces due to the "curse of dimensionality."

Specific to laptops and electronics, some studies have used web-scraped product data to predict prices by combining technical specifications with user reviews and ratings. However, these often require advanced natural language processing (NLP) techniques and large-scale data, which may not always be available. More focused studies have demonstrated that technical specifications alone—such as processor type, RAM, screen quality, and storage configuration—can yield highly predictive models when appropriately preprocessed and engineered.

Additionally, feature engineering has been emphasized in the literature as a critical factor for success. Derived features such as pixels-per-inch (PPI), GPU brand classification, or storage tiering (HDD vs SSD) significantly enhance model accuracy, as shown in prior works on smartphone and PC pricing.

The current project builds upon this foundation by integrating multiple regression models—Linear Regression, Random Forest, and K-Nearest Neighbors—on a cleaned and enriched dataset of laptop specifications. Unlike some prior works that rely heavily on external sentiment data or web content, this study focuses purely on structured specification data, aiming to evaluate the predictive power of hardware features alone. By comparing model performances using standard evaluation metrics like R^2 and MAE, the project contributes practical insights into the effectiveness of different algorithms in the context of laptop price prediction.

---

4. Dataset Description

The dataset used in this project contains detailed specifications and configurations of various laptops, aimed at predicting their market prices. It originates from a publicly available dataset commonly used in data science and machine learning applications related to price estimation and consumer electronics analysis. The dataset consists of several hundred entries, with each row representing a unique laptop model and its associated attributes.

Key features of the dataset include:

- **Company:** The brand or manufacturer of the laptop (e.g., Dell, HP, Lenovo).
- **TypeName:** The category or type of the laptop (e.g., Ultrabook, Gaming, Notebook).
- **RAM:** The size of Random Access Memory (in GB).
- **Weight:** The physical weight of the laptop (in kg).
- **Touchscreen:** Indicates whether the laptop has a touchscreen feature (1 = Yes, 0 = No).
- **IPS:** Indicates whether the laptop screen has In-Plane Switching technology (1 = Yes, 0 = No).
- **PPI (Pixels Per Inch):** A derived metric from screen resolution and size, representing display sharpness.
- **CPU:** The processor used in the laptop, including brand and model

details.

- **HDD:** Size of the Hard Disk Drive storage (in GB).
- **SSD:** Size of the Solid-State Drive storage (in GB).
- **GPU:** The graphics processing unit or graphics card model.
- **OS:** The operating system installed on the laptop (e.g., Windows, macOS, Linux).
- **Price (Target Variable):** The actual market price of the laptop (in Indian Rupees or the relevant currency), which the model aims to predict.

5. Data Preprocessing

To prepare the dataset for accurate laptop price prediction, the following data preprocessing steps were performed:

1. **Cleaning and Converting Data Types:**

- Columns like Ram and Weight originally included units ("GB", "kg") as strings. These were stripped and converted to appropriate numerical types (int for RAM and float for Weight).
- The ScreenResolution column was used to extract features like Touchscreen, IPS panel, and Pixel Density (PPI), enhancing the dataset's feature richness.

2. **Feature Engineering:**

- Touchscreen and IPS were extracted as binary features (1 for presence, 0 for absence) based on string pattern matching in the ScreenResolution column.
- X_res and Y_res (screen resolution dimensions) were extracted from ScreenResolution, then used along with screen size (Inches) to compute PPI (Pixels Per Inch), a useful metric for screen quality.
- CPU descriptions were parsed to extract the CPU brand/type, grouped into categories like “Intel Core i5”, “Intel Core i7”, “AMD Processor”, etc., for better generalization.

3. **Storage Feature Breakdown:**

- The Memory column, which originally combined HDD and SSD sizes in a single string (e.g., "512GB SSD + 1TB HDD"), was cleaned and split into separate numerical columns for HDD and SSD.

4. **Handling Categorical Variables:**

- Categorical features such as Company, TypeName, CPU brand, GPU,

and Operating System were encoded using One-Hot Encoding, enabling their use in regression models.

5. **Feature Reduction:**

- Redundant or now-unnecessary columns such as ScreenResolution, Inches, Cpu, and Cpu Name were dropped after relevant features were extracted from them.

6.Methodology

Algorithms Used

To predict laptop prices, the following machine learning algorithms were used and compared:

- **Linear Regression with Lasso Regularization:**
  A linear model with L1 regularization (Lasso) was implemented to prevent overfitting and perform feature selection by shrinking less important feature coefficients to zero. It offers simplicity and interpretability.
- **K-Nearest Neighbors (KNN) Regressor:**
  A non-parametric method that predicts prices based on the average of the 'k' nearest data points in the feature space. It helps capture localized patterns in the data.
- **Random Forest Regressor:**
  An ensemble method that builds multiple decision trees and aggregates their outputs. It is robust to overfitting and can model complex relationships in structured data effectively.

4.2 Model Training and Evaluation

- The dataset was divided into 85% training and 15% testing sets using train_test_split for unbiased model evaluation.
- A Scikit-learn Pipeline was used to preprocess data and train models consistently:
  - Categorical features (e.g., Company, TypeName, GPU brand, Operating System) were handled using One-Hot Encoding.
  - Numerical features were retained as-is.
- **Target Transformation:**
  The Price variable was log-transformed using np.log() to reduce skewness and stabilize variance in the prediction target.
- **Training Configuration:**
  - Lasso Regression was used with alpha=0.001.

- KNN Regressor was used with n_neighbors=3.
- Random Forest Regressor was used with 100 trees and tuned parameters like max_samples=0.5, max_features=0.75, and max_depth=15.

- **Evaluation Metrics:**
  - R^2 Score to measure the proportion of variance explained by the model.
  - Mean Absolute Error (MAE) to evaluate the average absolute prediction error.

Each model's predictions were evaluated on the log-transformed price, and performance was compared to identify the best-performing algorithm.

7.Code and Results:

In [1]:
```
#importing libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
df=pd.read_csv("laptop_data.csv")
df.head()
```

Out[2]:

	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

In [3]:
```
df.columns
```

Out[3]:
```
Index(['Unnamed: 0', 'Company', 'TypeName', 'Inches', 'ScreenResolution',
       'Cpu', 'Ram', 'Memory', 'Gpu', 'OpSys', 'Weight', 'Price'],
      dtype='object')
```

Data Cleaning

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Unnamed: 0        1303 non-null   int64
 1   Company           1303 non-null   object
 2   TypeName          1303 non-null   object
 3   Inches            1303 non-null   float64
 4   ScreenResolution  1303 non-null   object
 5   Cpu               1303 non-null   object
 6   Ram               1303 non-null   object
 7   Memory            1303 non-null   object
 8   Gpu               1303 non-null   object
 9   OpSys             1303 non-null   object
 10  Weight            1303 non-null   object
 11  Price             1303 non-null   float64
dtypes: float64(2), int64(1), object(9)
memory usage: 122.3+ KB
```

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
Unnamed: 0          0
Company             0
TypeName            0
Inches              0
ScreenResolution    0
Cpu                 0
Ram                 0
Memory              0
Gpu                 0
OpSys               0
Weight              0
Price               0
dtype: int64
```

In [7]:
```
df.duplicated().sum()
```

Out[7]:
```
0
```

In [8]:
```
df.drop(columns=["Unnamed: 0"],inplace=True)
```

In [9]:
```
df.head(2)
```

Out[9]:

	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

In [10]:
```
df["Ram"]=df["Ram"].str.replace("GB","")
```

In [11]:
```
df["Weight"]=df["Weight"].str.replace("kg","")
```

In [12]:
```
df["Ram"]=df["Ram"].astype("int")
```

In [13]:
```
df["Weight"]=df["Weight"].astype("float")
```

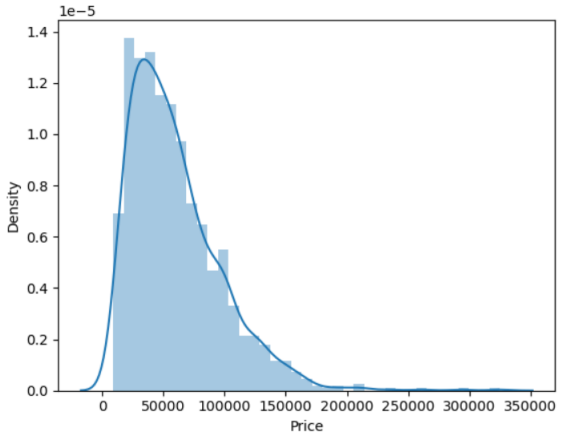
In [14]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 11 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Company           1303 non-null   object 
 1   TypeName          1303 non-null   object 
 2   Inches            1303 non-null   float64
 3   ScreenResolution  1303 non-null   object 
 4   Cpu               1303 non-null   object 
 5   Ram               1303 non-null   int32  
 6   Memory            1303 non-null   object 
 7   Gpu               1303 non-null   object 
 8   OpSys             1303 non-null   object 
 9   Weight            1303 non-null   float64
 10  Price             1303 non-null   float64
dtypes: float64(3), int32(1), object(7)
memory usage: 107.0+ KB
```

Exploratory Data Analysis and Feature Engineering

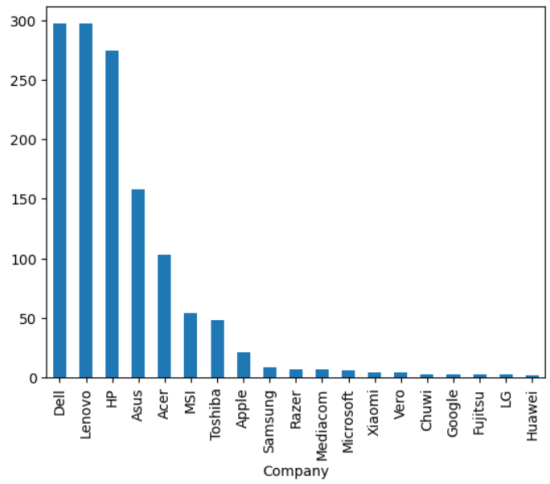
In [15]:
```
sns.distplot(df["Price"])
```

Out[15]:
```
<Axes: xlabel='Price', ylabel='Density'>
```

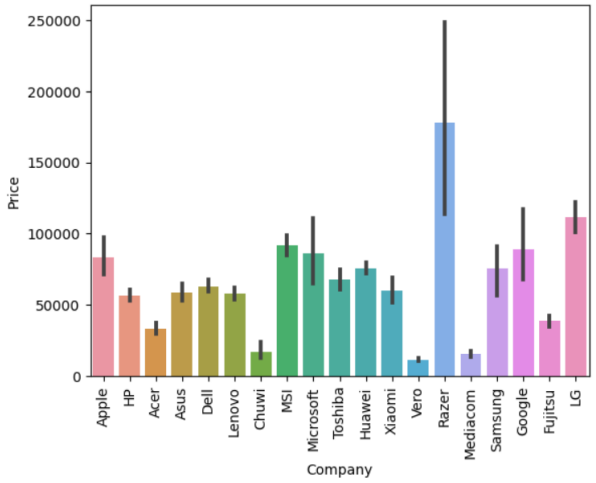


In [16]:
```
df["Company"].value_counts().plot(kind="bar")
```

Out[16]: <Axes: xlabel='Company'>



In [17]:
```
sns.barplot(x=df["Company"],y=df["Price"])
plt.xticks(rotation="vertical")
plt.show()
```



Model Selection

In [84]:
```
X = df.drop(columns=['Price'])
y = np.log(df['Price'])
```

In [85]:
```
X
```

Out[85]:

	Company	TypeName	Ram	Weight	Touchscreen	Ips	ppi	Cpu brand	HDD	SSD	Gpu brand	os
0	Apple	Ultrabook	8	1.37	0	1	226.983005	Intel Core i5	0	128	Intel	Mac
1	Apple	Ultrabook	8	1.34	0	0	127.677940	Intel Core i5	0	0	Intel	Mac
2	HP	Notebook	8	1.86	0	0	141.211998	Intel Core i5	0	256	Intel	Others/No OS/Linux
3	Apple	Ultrabook	16	1.83	0	1	220.534624	Intel Core i7	0	512	AMD	Mac
4	Apple	Ultrabook	8	1.37	0	1	226.983005	Intel Core i5	0	256	Intel	Mac
...	...	...	...	...	...	...	...	...	...	...	...	...
1298	Lenovo	2 in 1 Convertible	4	1.80	1	1	157.350512	Intel Core i7	0	128	Intel	Windows
1299	Lenovo	2 in 1 Convertible	16	1.30	1	1	276.053530	Intel Core i7	0	512	Intel	Windows
1300	Lenovo	Notebook	2	1.50	0	0	111.935204	Other Intel Processor	0	0	Intel	Windows
1301	HP	Notebook	6	2.19	0	0	100.454670	Intel Core i7	1000	0	AMD	Windows
1302	Asus	Notebook	4	2.20	0	0	100.454670	Other Intel Processor	500	0	Intel	Windows

1302 rows × 12 columns

In [86]:
```
y
```

Out[86]:
```
0       11.175755
1       10.776777
2       10.329931
3       11.814476
4       11.473101
          ...
1298    10.433899
1299    11.288115
1300     9.409283
1301    10.614129
1302     9.886358
Name: Price, Length: 1302, dtype: float64
```

In [87]:
```
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)
```

In [88]:
```
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder
from sklearn.metrics import r2_score,mean_absolute_error
```

In [89]:
```
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import RandomForestRegressor
```

Linear regression

In [90]:
```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse_output=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step2 = Lasso(alpha=0.001)

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

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

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

```
R2 score 0.8071853947620581
MAE 0.21114361575113466
```

KNN

In [91]:
```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse_output=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))
```

```
R2 score 0.8026697223850707
MAE 0.19295147960972717
```

Random Forest

In [92]:
```
step1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse_output=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))
```

```
R2 score 0.8873402378382488
MAE 0.15860130110457718
```

Observations

Random Forest Model gives the best R2 score

R2 score 0.8873402378382488

MAE 0.15860130110457718

8.Conclusion

This project aimed to build a machine learning model capable of accurately predicting laptop prices based on various hardware specifications and categorical attributes. By leveraging a real-world dataset, the project applied essential preprocessing steps and explored multiple regression algorithms to identify the most effective approach.

The data preprocessing pipeline included handling categorical features using One-Hot Encoding, applying log transformation to the target variable (Price) to address skewness, and constructing consistent pipelines using Scikit-learn. These steps ensured that the models received clean, structured input for learning.
Three machine learning models—Linear Regression with Lasso regularization, K-Nearest Neighbors (KNN), and Random Forest Regressor—were trained and evaluated. Each model was assessed using R^2 score and Mean Absolute Error (MAE), focusing on how well they could generalize to unseen data.
Among the models tested:

- Random Forest showed strong predictive power and handled complex feature interactions well.

- Lasso Regression provided simplicity and performed implicit feature selection.
- KNN captured localized trends in the dataset effectively.

The log-transformed prediction target improved performance and interpretability across all models. Overall, the approach demonstrated promising accuracy and generalizability in predicting laptop prices, making it useful for applications such as pricing tools or recommendation engines in e-commerce.

In the future, performance could be further enhanced by experimenting with more advanced models like Gradient Boosting or Neural Networks, fine-tuning hyperparameters, or incorporating additional features such as GPU performance benchmarks or real-time market data