Estimating the Hardware-Based Price of Apple iPhones Using Machine Learning

Introduction

Some people perceive Apple iPhones as overpriced compared to other similar smartphones, suspecting an "Apple premium" beyond hardware costs. We aim to investigate this perception by trying to estimate the hardware-based price of Apple iPhones using ML. We analyze a dataset of smartphone specifications and prices to develop regression models that estimate what an iPhone would cost based on its hardware alone. The paper discusses the problem and dataset, methods used, and ML models used, presents results and evaluation metrics, and concludes with findings.

Problem Formulation

The problem is that we don't know what a standalone Apple iPhone model with certain components would cost if only take into consideration their hardware aspects (is there an Apple premium to be paid for the same hardware). This can be framed as a supervised regression problem where we try to predict the price of the smartphone based on technical components.

The dataset consists of 1359 data points with 22 columns that include information about technical aspects of the phone (such as Battery capacity (mAh), RAM, internal storage, rear camera MP, etc.) and the price of the phone in Indian Rupees (INR). The price will be converted to Euros and inflation-adjusted accordingly to reflect the current time's price.

The data used is from the 2022 Mobile Phone Specifications and Prices dataset, link to Kaggle. Data is scraped from the Gadgets360 website. It includes detailed hardware specifications and pricing from different smartphone manufacturers, see Table 3 for more information.

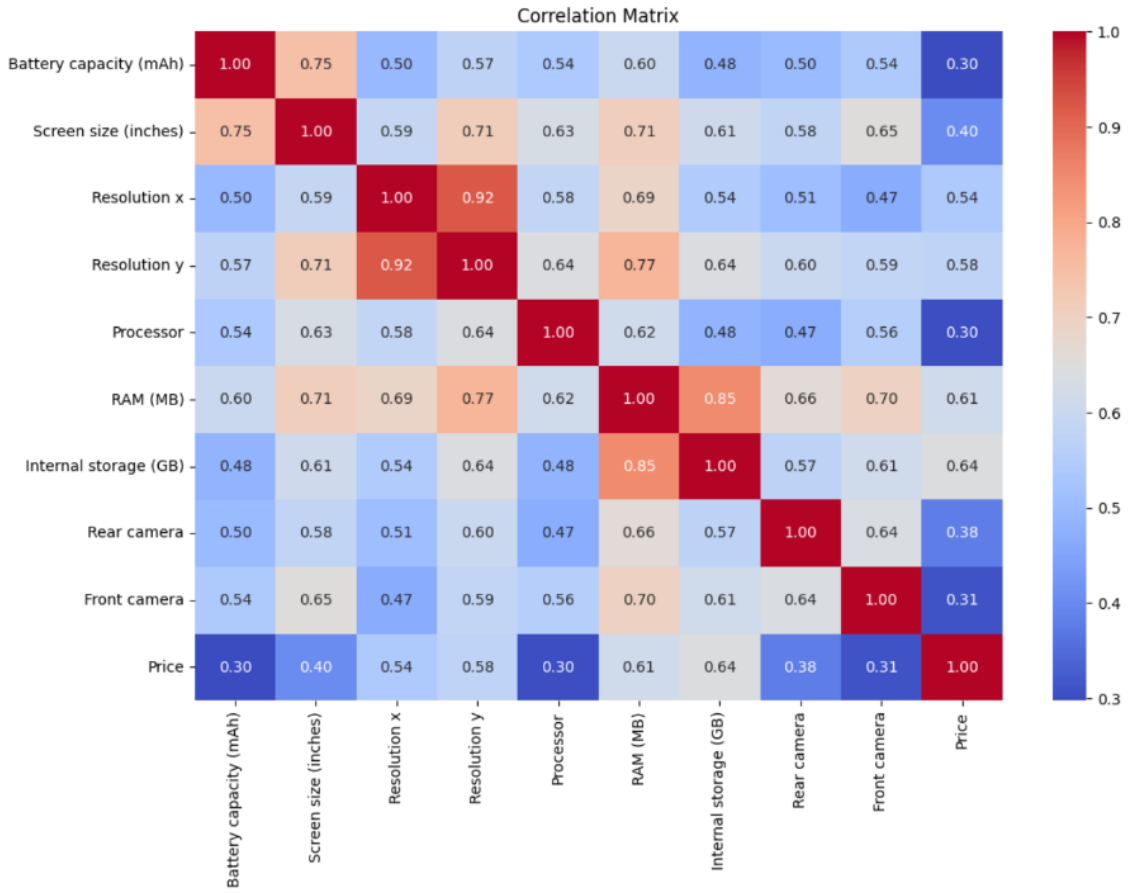

Figure 1: Feature correlation matrix

Methods

Data Preprocessing and Feature Selection

Initial inspection of features like name, brand, model, operating system Wi-fi, Bluetooth, GPS, number of SIMs, 3G, 4G/LTE, and touchscreen revealed a minimal correlation (< 0.10) with pricing; thus, these columns were removed from the consideration in our analysis. Additionally, all Apple phones were extracted for later testing purposes.

We additionally calculated a new feature 'Resolution ratio', from the resolutions x and y components because x and y resolutions have such a strong correlation. This will be used instead of both components separately.

Given a somewhat heavy distribution towards cheaper phones in our initial dataset (visualized in the Price Distribution graph), we needed to do something regarding this disparity. We have employed a balancing via down-sampling of lower-priced phones and up-sampling of higher-priced ones. Sampling down 20% low-priced phone data points and upsizing high-priced entries by four times. A graph is generated of the new distribution (Price Distribution after Balancing). It is important to note that this solution is not perfect but satisfactory.

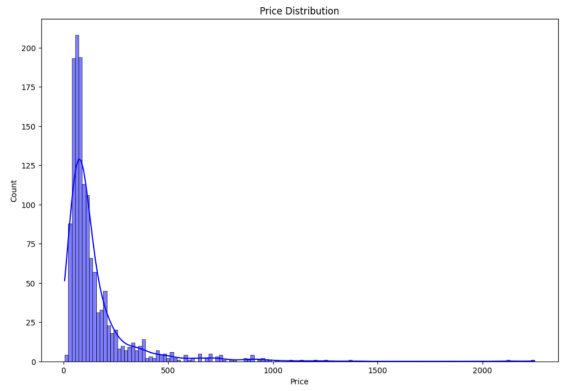

Figure 2: Original Price Distribution

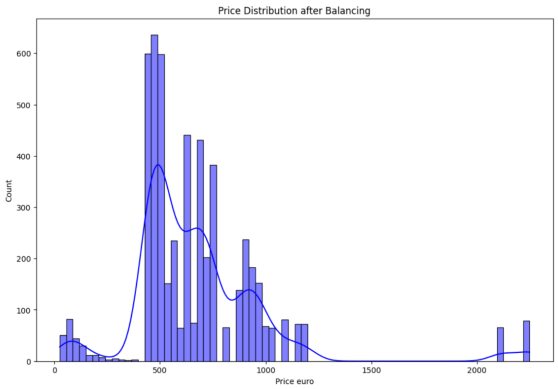

Figure 3: After Balancing

Model Validation Strategy

The data set has been divided into three parts: training, validation and test set. We are employing a 70-20-10 split. The training set is comprised of 3745 (70%) data points. Validation set 1070 (20%) data points. And test set 535 (10%) data points.

The splitting has been conducted randomly using `train_test_split` provided in scikit-learn. The choice for this kind of setup was made because we want to prioritize the model training dataset size. Testing and validation datasets are still large enough to improve and generalize the model so that we can counteract overfitting.

Model 1: Linear Regression

The linear regression model is used to model linear relationships in data. We visualized all of our feature variables against price and found a small linear correlation in the data. This makes sense intuitively because better technical aspects should cost more.

The YouTube channel LearnChemE gives a great technical overview of the math behind the method in their video called 'Matrix Approach to Multiple Linear Regression'. The video states that in linear regression, we try to find weights β so that:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \cdots + \beta_k x_k$$

$$y = X\beta \quad \text{(matrix form)}$$

The data points won't perfectly fit the linear model so we need to find the β values that minimize the sum of squares error $(y - X\beta)^T (y - X\beta)$. The minimization is calculated by solving for β in:

$$X^T X\beta = X^T y$$

$$\beta = (X^T X)^{-1} X^T y$$

Model 2: Multi-layer Perceptron (MLP) Regression with Squared Error Loss Function

To model the complex, non-linear relationships between smartphone technical specifications and their corresponding prices, we use an MLP regression model. Unlike linear regression, which assumes a linear relationship between input features and the target variable, MLPs can approximate any continuous function and can capture intricate patterns within the data [1].

The MLP architecture has an input layer, four hidden layers each containing 200 neurons, and an output layer. The hidden layers allow the network to learn hierarchical feature representations, improving its ability to model non-linearity between the features and the target variable.

Activation functions play an important aspect of non-linearity in the network. In this model, the Rectified Linear Unit (ReLU) activation function was employed for the hidden layers. ReLU is defined as:

$$f(x) = \begin{cases} 0, & x \leq 0 \\ x, & x > 0 \end{cases}$$

Training the MLP involves forward propagation and backward propagation. According to [1], the weight updates for a neuron can be expressed as:

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}$$

Mean Squared Error

The squared error is natural to use in our application because it is used already in the fitting process. According to Jung's book:

$$\frac{1}{n} \sum_{i=0}^{n} (y_i - h(X_i))^2$$

Coefficient of Determination, R^2

Residual sum of squares:

$$SS_{res} = \sum_i (y_i - f_i)^2$$

Total sum of squares:

$$SS_{tot} = \sum_i (y_i - \bar{y})^2$$

Coefficient of determination:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

Results

Table 1: Model performance

Models	Val MSE	Test MSE	Val R^2	Test R^2
Linear regression	72391.07	65929.51	0.37	0.38
Polynomial regression (degree 2)	24969.26	25059.11	0.78	0.76
Multilayer perceptron	7788.76	6003.74	0.93	0.94
Decision tree	2818.04	718.25	0.98	0.99
Random forest	3037.06	792.56	0.97	0.99

As can be seen from the table, our best results are obtained from the decision tree and random forest. The multilayer perceptron also performs extremely well.

The linear regression and polynomial regression models perform poorly because there is no obvious linear or polynomial tendencies in the data. When printing the prices to the console predicted by the linear and polynomial models, you notice that the model tends to predict too high for cheap phones and too low for expensive phones.

When comparing the predicted prices to the actual prices, the multilayer perceptron performs better on new data than the decision tree and random forest models. The reason for this is that the tree and forest models are bad at predicting values outside the training data set. In other words, the two models are overfitted and when introducing our final test data set of Apple products, the models performed extremely poorly.

The best model according to our code is multilayer perceptron. The picture shows the Apple training set and the results of the MLP. The model performs well because MLP is good at predicting nonlinear relationships in data.

The model makes sense. Apple is known to have extremely high-profit margins on their new phones [2]. Profit margins in new phones can close to 100%. When the technology is getting old, Apple cuts down on prices so the cheaper phones don't have so huge markups.

Conclusion

In this paper, we have aimed to estimate the hardware price of Apple iPhones by using machine learning models. The MLP model performed best throughout our testing capturing nonlinear relationships within the data, test MSE 6003.74 and $R^2 = 0.94$.

MLP model predicted somewhat consistently lower prices for Apple iPhone compared to the actual market price, this supports the hypothesis that there is an "Apple premium"—the additional costs buyers pay for Apple products beyond hardware value.

Despite the efforts we have made in the research we see that there is much room for improvement. The preprocessing and feature selection made on the data are not

Table 2: MLPRegressor results sorted by decending actual price.

	Actual Price	Predicted Price	Price Difference
0	1373.622240	707.195336	666.426904
16	1245.126240	522.174790	722.951450
3	899.459150	503.555725	395.903425
7	899.459150	334.648610	564.810540
1	808.239480	388.587047	419.652793
4	770.963150	508.216271	262.746879
6	603.931200	341.012679	262.918521
2	584.643950	661.08365	-76.359914
5	475.422350	280.648613	194.773738
9	449.723150	464.485565	-14.762415
8	359.775950	291.648578	68.127372
10	333.961104	251.061495	82.899609
13	308.377550	114.125487	194.252063
12	295.527950	446.592259	-151.064309
11	218.403350	319.927724	-101.497374
14	205.580750	256.367783	-50.787033
15	109.208750	197.873444	-88.665594

necessarily the most optimal approaches made, this affects the model's performance. In the dataset, there is an imbalance towards cheaper phones, however, this could be due to the natural state of the market that is just reflected in the dataset. The balancing of the dataset could have potentially introduced biases that we are not aware of.

Random forest and decision tree showed low errors on the test set, however, they performed poorly when predicting the price of the phones, indicating a lack of generalization and possible overfitting. Linear regression performed poorly on this specific task.

For a future approach, it would be important to refine the dataset to include more recent and diverse data points, which possibly include incorporating international pricing and a wider range of high-end smartphone models. Enhancing the feature selection process by identifying and including more relevant technical specifications, for example, PCA could be performed, or creating composite features could improve model accuracy. It would be also important to try other ML methods such as SVR.

Note on AI usage

ChatGPT was utilized to assist with variable naming conventions and to deepen our understanding of key concepts. However, all the information provided is a direct reflection of the authors' knowledge and understanding of the subject matter.

References

References

[1] R. Kruse, C. Borgelt, F. Klawonn, C. Moewes, M. Steinbrecher, and P. Held, *Computational Intelligence: A Methodological Introduction*. London: Springer, 2013.

[2] G. Cuofano, "How much profit does apple make per iphone?" FourWeekMBA, 2024, accessed: 2024-10-08. [Online]. Available: https://fourweekmba.com/how-much-profit-does-apple-make-per-iphone/

Appendix

Table 3: Dataset Feature Overview

Feature Name	**Feature Description**
Name (String)	Name of Phone
Brand (String)	Brand Name
Model (String)	Model of the Phone
RAM (Integer)	RAM available in MB
Internal storage (Float)	Storage in GB
Rear camera (Float)	Resolution in MP
Battery capacity (Integer)	Capacity in mAh
Front camera (Float)	Resolution in MP
Screen size (Float)	Screen size in inches
Operating system (String)	OS on phone
Touchscreen (Yes/No)	Has touchscreen
Wi-Fi (Yes/No)	Has Wi-Fi
Resolution x (Integer)	Resolution width
Bluetooth (Yes/No)	Has Bluetooth
Resolution y (Integer)	Resolution height
GPS (Yes/No)	Has GPS
Processor (Integer)	No. of cores
Number of SIMs (Integer)	SIM card slots
3G (Yes/No)	Has 3G
4G/LTE (Yes/No)	Has 4G/LTE
Price (Integer)	Price in INR