ReCell Project

Supervised Learning - Linear Regression

Yair Brama – August 2024

Contents / Agenda

Executive Summary - Business Context

The rising potential of this comparatively under-the-radar market of refurbished and used cellular devices fuels the need for an ML-based solution to develop a dynamic pricing strategy. ReCell, a startup aiming to tap the potential in this market, has hired me as a data scientist. In this project we will analyze the data provided and build a linear regression model to predict the price of a used phone/tablet and identify factors that significantly influence it.
(See the data structure in the appendix part)

Performing Supervised Machine Learning – Linear Regression

The primary purpose of linear regression is to predict the value of the dependent variable (price of used devices) based on the values of the independent variables.

As it will show in the next slides, the model that is based on the given data can predict the price with an accuracy of ~81%, which meets our expectations for a reliable model

Executive Summary – Model Results

Metric	Train Value	Test Value	Description
R-squared	**0.816984**	**0.822131**	The training model is NOT underfitting, ~82% of the results can be predicted by the model
Adjusted R-squared	**0.81584**	**0.819518**	This value shows how well the model fits the data, considering the amount of variables
MAE	**0.194005**	**0.196185**	MAE indicates that our current model can predict the price within a mean error of ~0.2 units on the test data.
MAPE	**0.046678**	**0.047941**	MAPE of 4.8% shows we can predict the price of the test data within this range
RMSE	**0.249705**	**0.253286**	The train and test RMSE (as well as MAE) are comparable, so the model is not overfitting

Equation of linear regression

normalized price of used device = 1.3946020282806701 + 0.014677951801391 42 * (**main_camera_mp**) + 0.01171334063789519 * (**selfie_camera_mp**) + 0.018694083226400683 * (**ram**) + 9.434404942690962e-05 * (**battery**) + 0.475646014127734 6 * (**normalized_new_price**) + -0.021515391520670724 * (**years_since_release**) + 0.05985704728434499 * (**brand_name_Alcatel**) + 0.14081623761562923 * (**brand_name_Karbonn**) + 0.0645434807324212 * (**brand_name_Lenovo**) + 0.13940662078054894 * (**brand_name_Microsoft**) + 0.07148832358895618 * (**brand_name_Nokia**) + 0.07361505473160784 * (**brand_name_Xiaomi**) + -0.1454720059578327 * (**os_Others**) + -0.0763163720532658 * (**5g_yes**)

Solution Approach

Data initial review	EDA	Data Preprocessing	Model Building - Linear Regression	Check assumptions	Final Model Summary
Check for null values Check for duplicates Check data types	Univariate Analysis – Box plots, Histograms Multivariate Analysis - Check for correlation (Heatmap), visualize the correlated variables	Missing Value Imputation – Based on Median values Feature Engineering – Add field for number of years Create Dummy variables – For the categories fields – brands, OS	Split the dataset between training and test data Run the model and check the performance	No Multicollinearity Linearity of variables Independence of error terms Normality of error terms No Heteroscedasticity	Perform the final run Compare the training and the test sets

EDA Results - Data Overview

The data includes 3454 rows and 15 columns

There are 34 unique brands (e.g., Nokia, Apple), and 4 OS type (e.g., Android, iOS)

Data types:

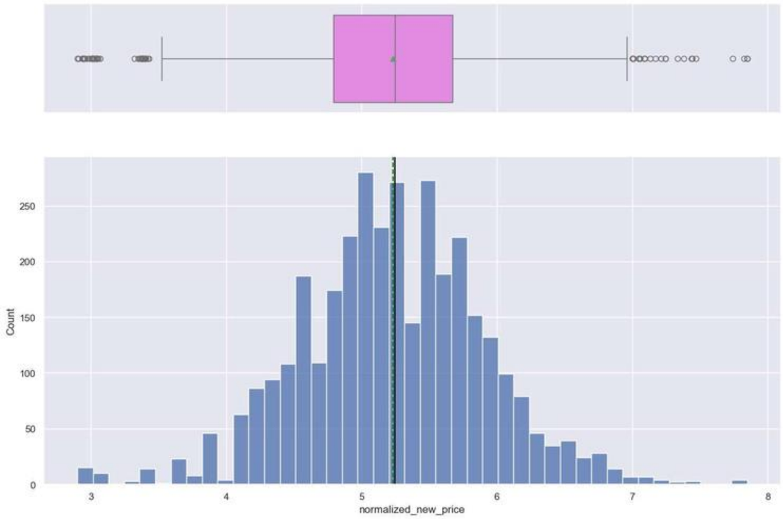
```
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   brand_name             3454 non-null   object
 1   os                     3454 non-null   object
 2   screen_size            3454 non-null   float64
 3   4g                     3454 non-null   object
 4   5g                     3454 non-null   object
 5   main_camera_mp         3275 non-null   float64
 6   selfie_camera_mp       3452 non-null   float64
 7   int_memory             3450 non-null   float64
 8   ram                    3450 non-null   float64
 9   battery                3448 non-null   float64
 10  weight                 3447 non-null   float64
 11  release_year           3454 non-null   int64
 12  days_used              3454 non-null   int64
 13  normalized_used_price  3454 non-null   float64
 14  normalized_new_price   3454 non-null   float64
dtypes: float64(9), int64(2), object(4)
memory usage: 404.9+ KB
```

Checking for missing values

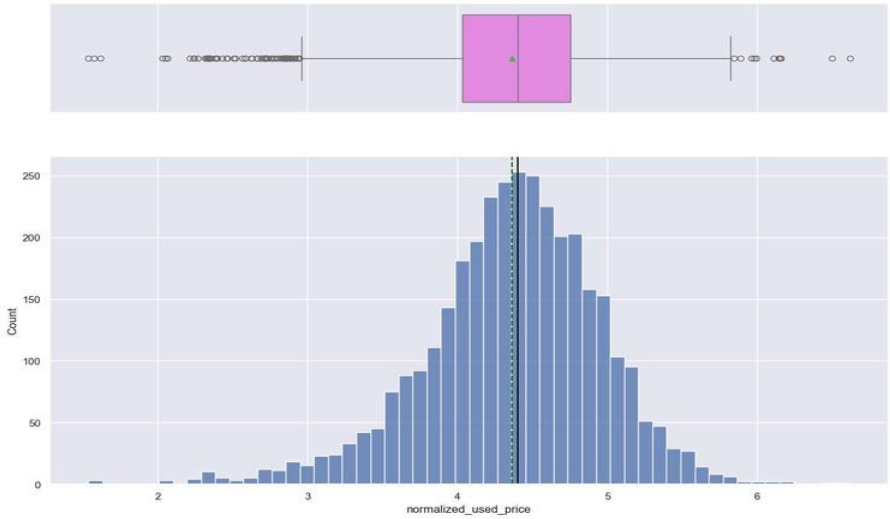
```
In [20]: data.isnull().sum() ## Complete the code to check duplicate entries in the data

Out[20]: brand_name                 0
         os                         0
         screen_size                0
         4g                         0
         5g                         0
         main_camera_mp           179
         selfie_camera_mp           2
         int_memory                 4
         ram                        4
         battery                    6
         weight                     7
         release_year               0
         days_used                  0
         normalized_used_price      0
         normalized_new_price       0
         dtype: int64
```

EDA Results - Univariate Analysis

Normalized new price



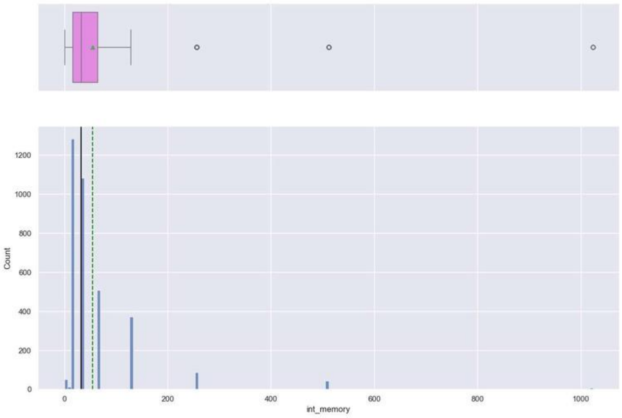
Normalized used price



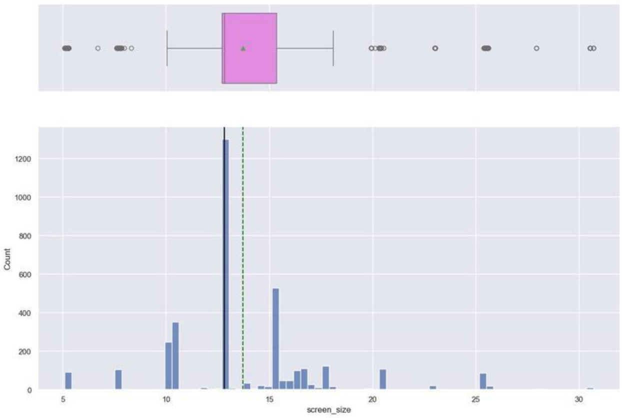
Observation – In this sampling observation, the used price and the new price seem to be distributed normally (visually), with outliers on both sides

EDA Results - Univariate Analysis – Cont.

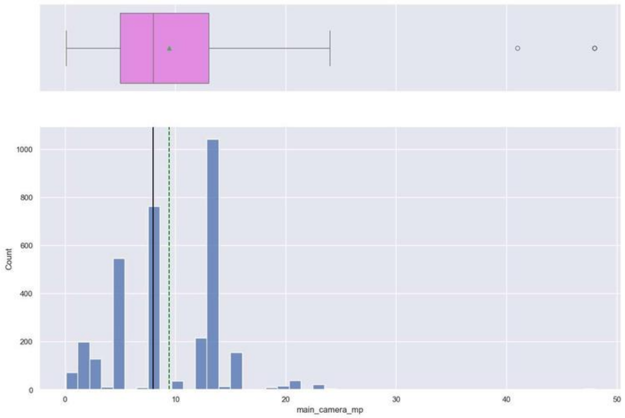
Internal Memory



Screen size



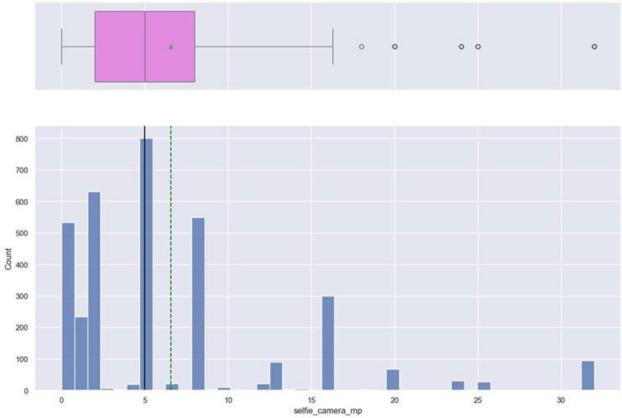
Main camera MP



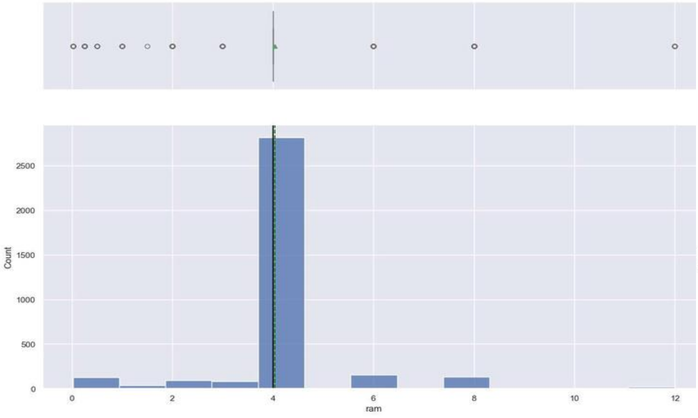
Observation – In this sampling observation, these variables look skewed to the right

EDA Results - Univariate Analysis – Cont.

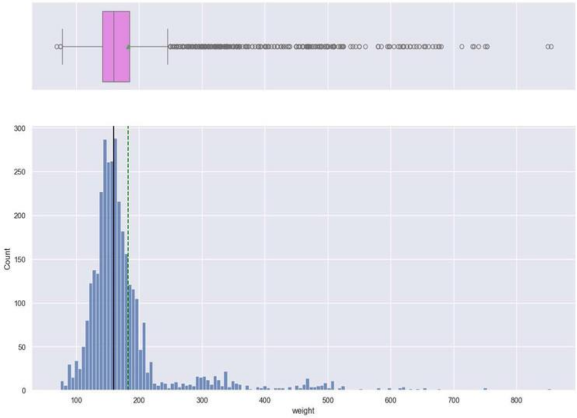
Selfie camera MP



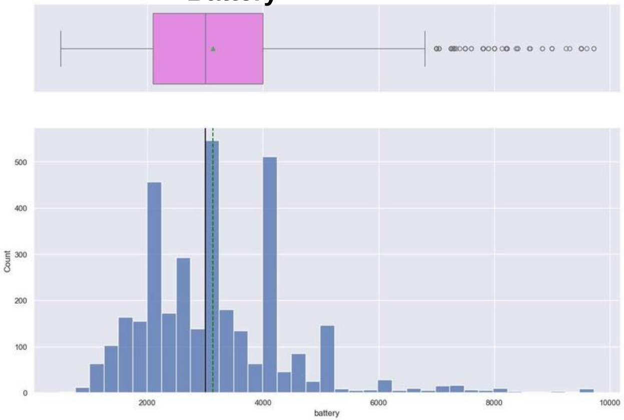
RAM



Weight



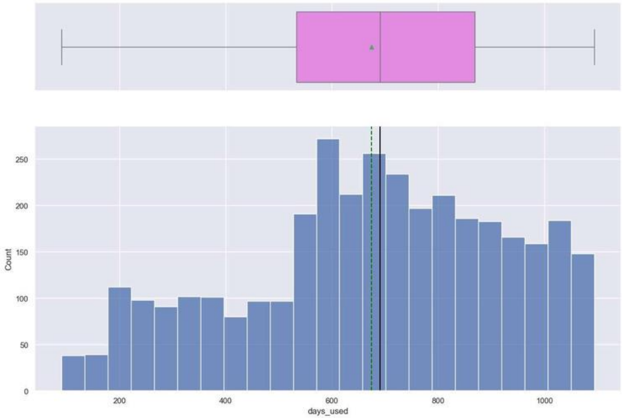
Battery



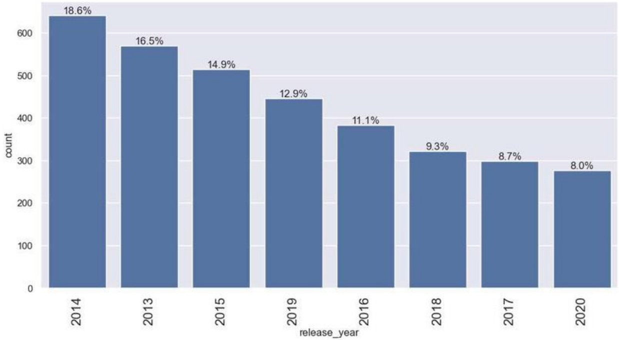
Observation – In this sampling observation, these variables look skewed to the right as well

EDA Results - Univariate Analysis – Cont.

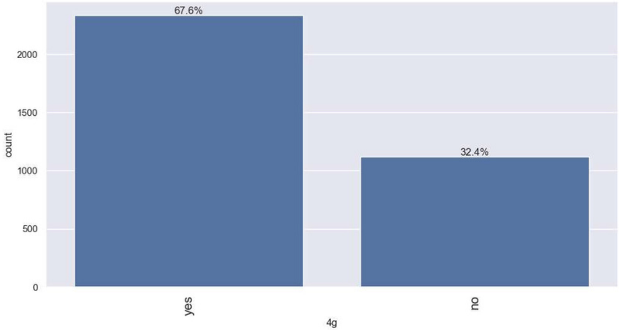
Days Used



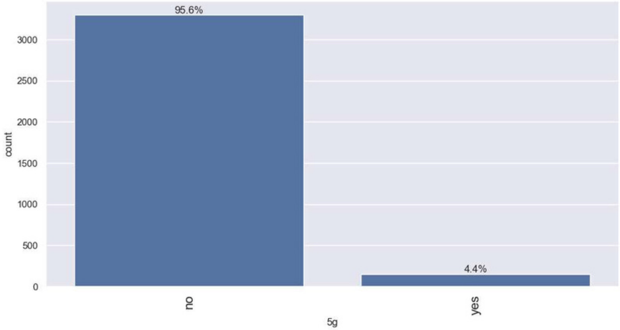
Release Year



4g



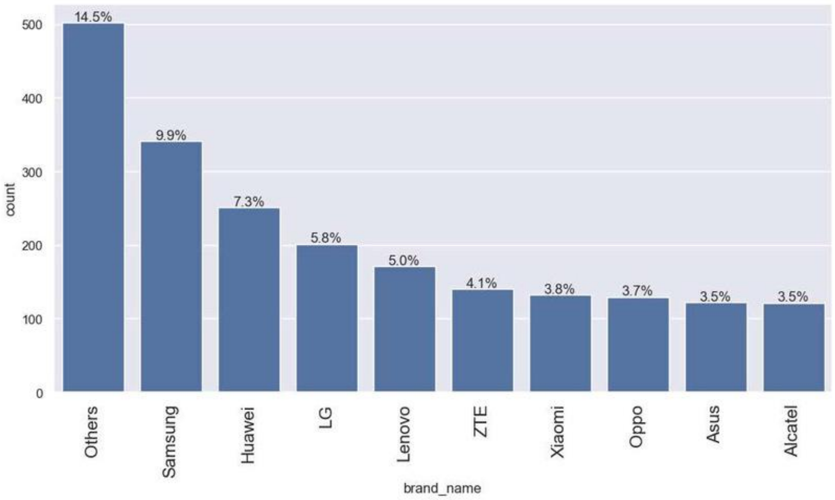
5g



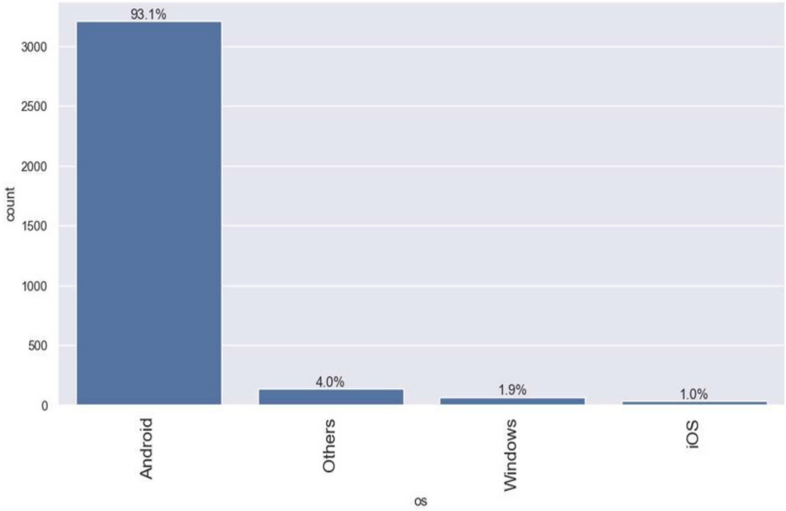
Observation – Not surprisingly, these variables show that there is more representation for older devices in this set, with more days used, and more 4g than 5g

EDA Results - Univariate Analysis – Cont.

Brand Name

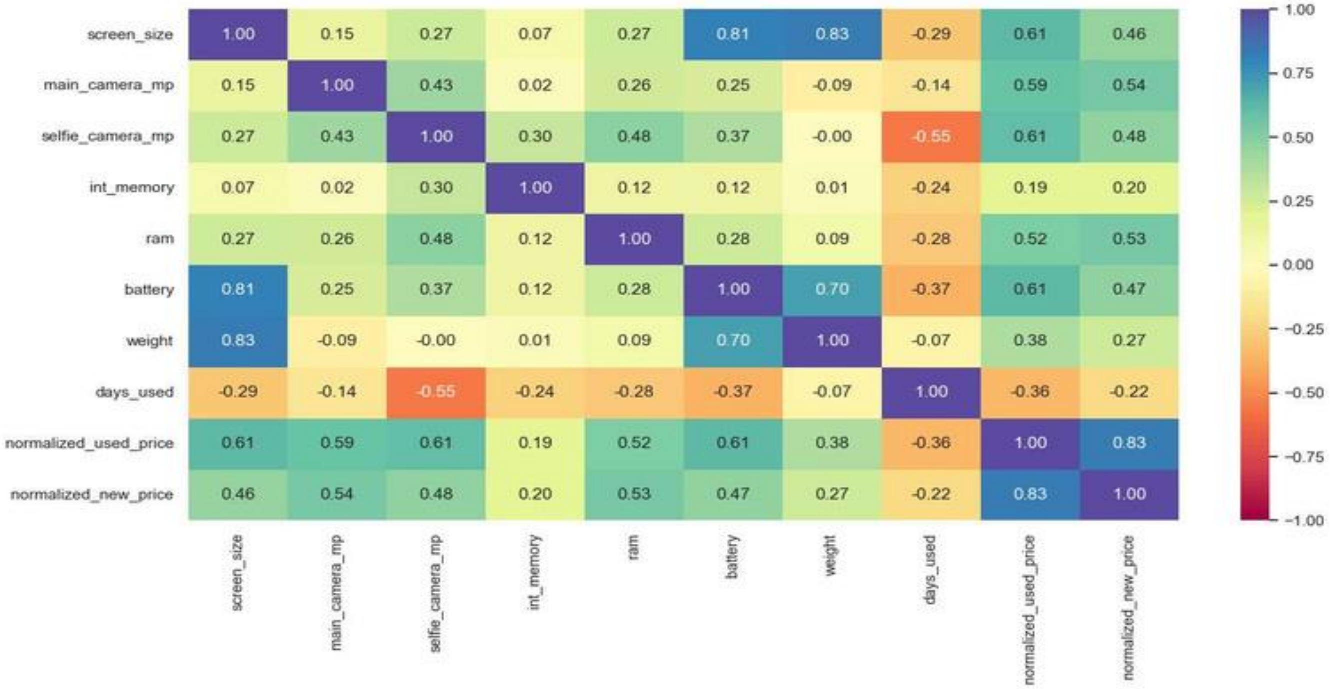


OS



Observation – These 2 variables are most likely correlated, as the OS and the brand usually go together. Note that there is a small representation of iPhones, which can indicate that these devices are not resold like Android devices, or there isn't enough data

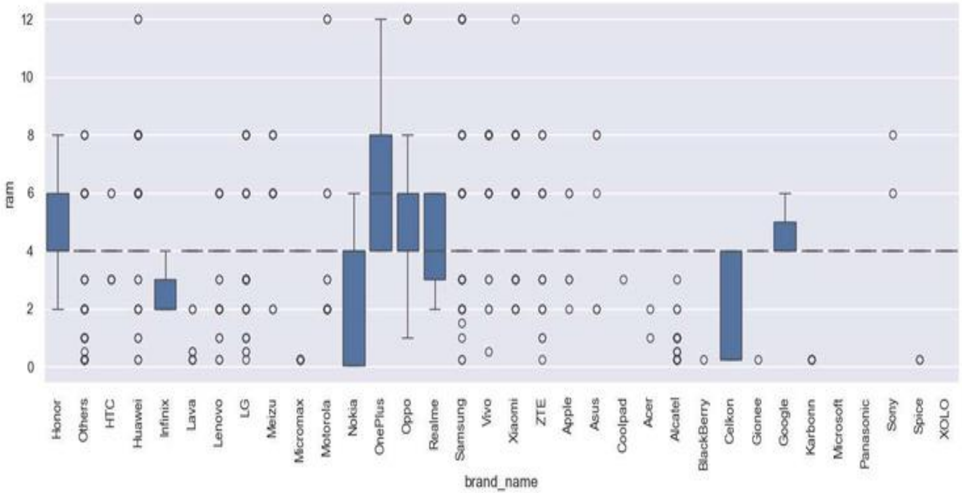
Multivariate Analysis – Heatmap/Correlation



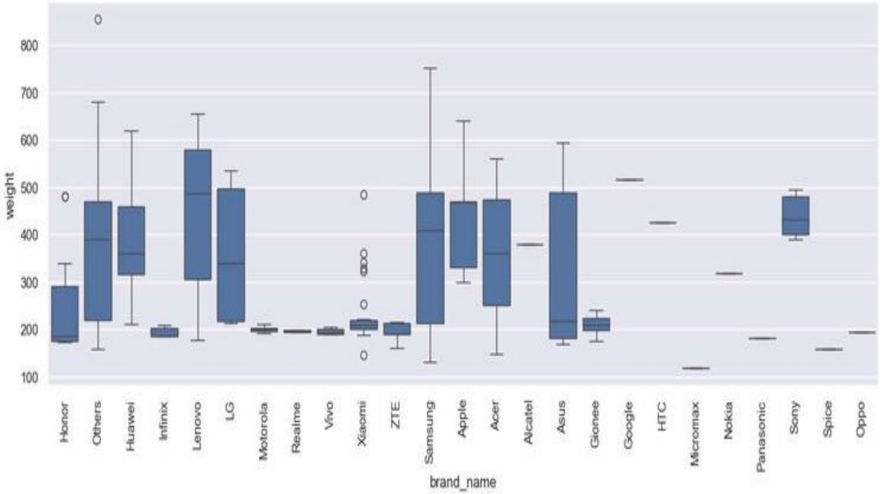
Observation – We can see a stronger correlation between screen size, battery and weight. Also, there is a strong correlation between the new price and the old price

Multivariate Analysis – RAM and Weight across brands

Brand Name and RAM relation



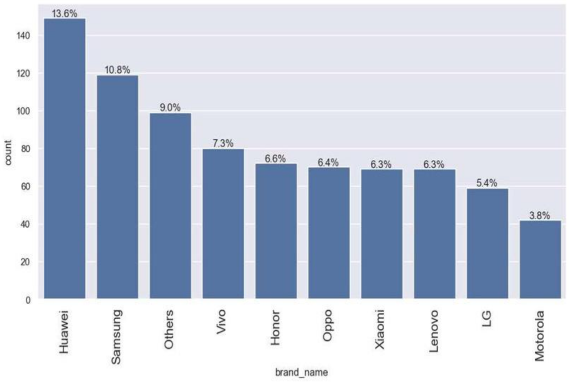
Brand Name and weight relation where battery > 4500



Observation – There is no relation between the RAM and the weight within the same brand

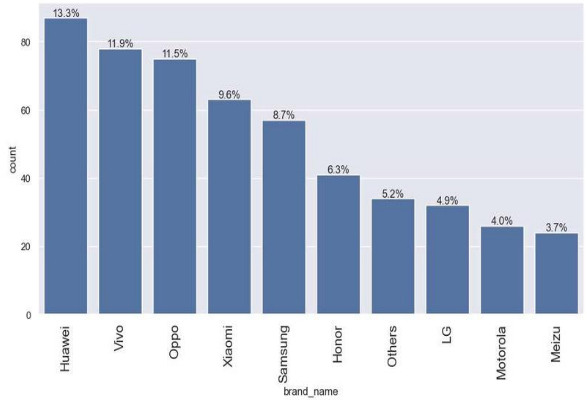
Multivariate Analysis – Screen size and cameras across brands

Brand Name and screen size > 6 * 2.54

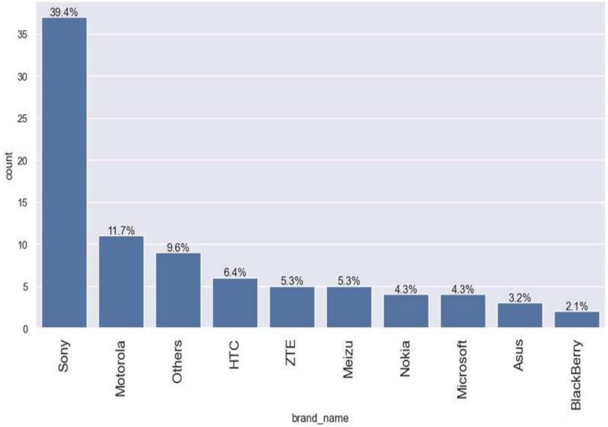


Observation – We can see that the similar brands have the stronger front cameras and a bigger screen.
However, different brands have stronger rear camera (the main camera)

Brand Name and front camera > 8 mp

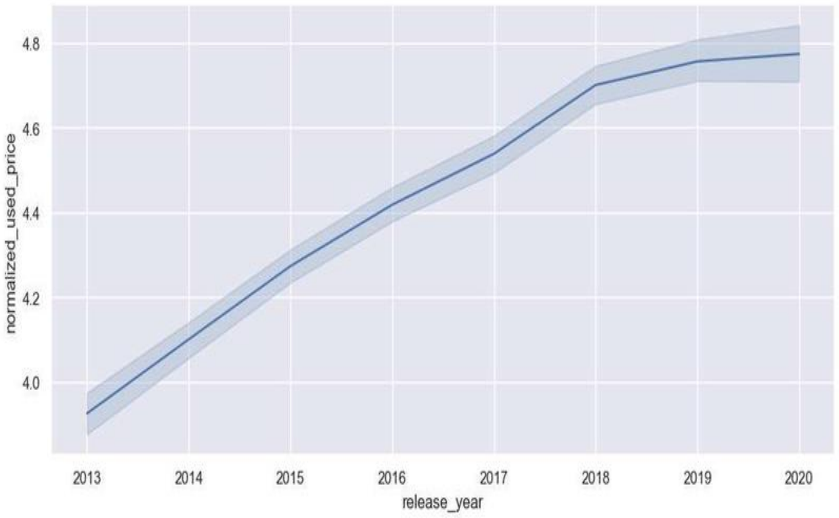


Brand Name and weight rear camera > 16 mp

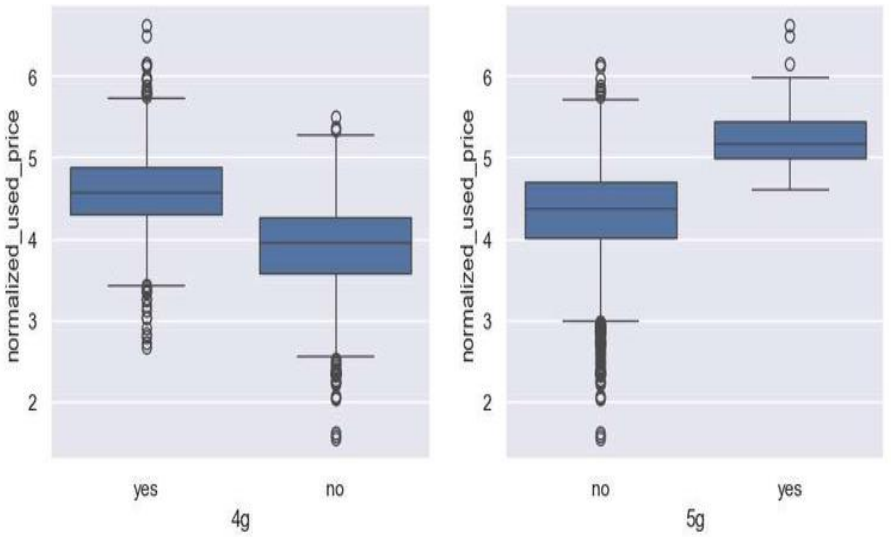


Multivariate Analysis – Prices over time and in relation to 4/5g

Price of used devices



Price of used 4g and 5g



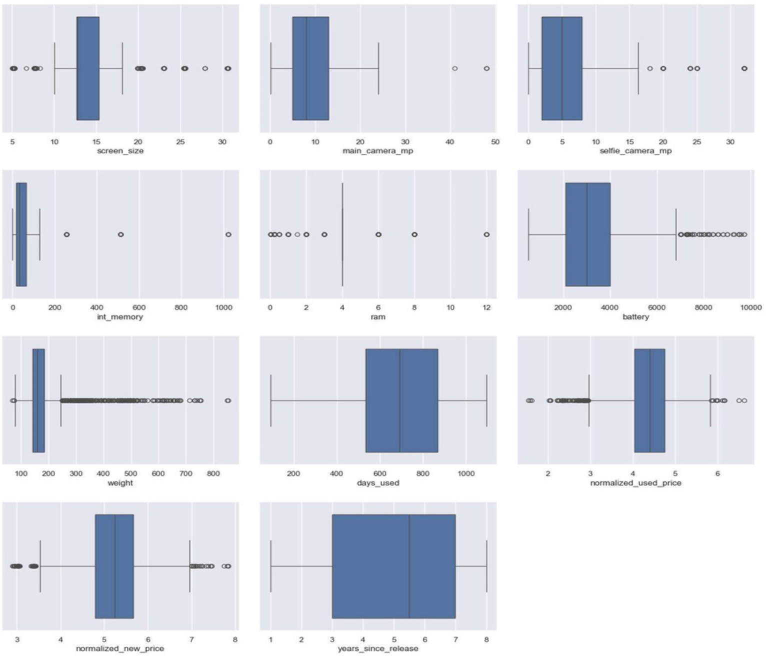
Observation – As expected, newer (but used) devices cost more, and 5g is more expensive than 4g that is more expensive than non-4g

Data Preprocessing – Missing Value Imputation

	initial missing data	After adding the median based on release year and brand	After adding the median based based brand name only	Final State – After adding the median of the entire column of main_camera_mp
brand_name	0	0	0	0
os	0	0	0	0
screen_size	0	0	0	0
4g	0	0	0	0
5g	0	0	0	0
main_camera_mp	179	179	10	0
selfie_camera_mp	2	2	0	0
int_memory	4	0	0	0
ram	4	0	0	0
battery	6	6	0	0
weight	7	7	0	0
release_year	0	0	0	0
days_used	0	0	0	0
normalized_used_price	0	0	0	0
normalized_new_price	0	0	0	0
dtype: int64				

Data Preprocessing – Feature Engineering and Outlier Check

- **New variable was added, to simplify the computation of years.**
  - Let's create a new column `years_since_release` from the `release_year` column.
  - We will consider the year of data collection, 2021, as the baseline.
  - We will drop the `release_year` column.



Outliers Check - Most of the variables have outliers, however we will move on with considering them as part of the model

Model Performance Summary – Process Stages

Final Model Run

Run the final model and check the residuals

Remove variables

One by one, check the impact of removing the variables below

Check Assumptions

Check VIF > 5 and P values > 0.05 of the variables

Data Preperation

Create train set (70%) and test set (30%)

Dummy Variables

Dummy variables for brand name and OS

Initial Model Run

Run the model on the training set and check the results

Model Performance Summary – Initial Run Performance

Training Performance

	RMSE	MAE	R-squared	Adj. R-squared	MAPE
0	0.229884	0.180326	0.844886	0.841675	0.043268

Test Performance

	RMSE	MAE	R-squared	Adj. R-squared	MAPE
0	0.238358	0.184749	0.842479	0.834659	0.045017

Observation – The results show a strong performance and ability to predict the independent variable (normalized price of used devices).
Now we need to check the assumptions of linear regression.

Model Performance Summary – Checking Linear Regression assumptions

- **NO MULTICOLLINEARITY**

The independent variables must not be highly correlated with one another, as this can lead to unstable and unreliable coefficient estimates.

- **LINEARITY**

The relationship between the independent variables and the dependent variable must be linear.

- **INDEPENDENCE**

The observations in the dataset must be independent of one another, meaning they are not influenced by any other observations in the dataset.

- **NORMALITY**

The residuals must be normally distributed with a mean of zero.

- **HOMOSCEDASTICITY**

The variance of the residuals (the difference between the observed values and the predicted values) must be constant across all levels of the independent

Model Performance Summary – No Multicollinearity

Index	Feature	VIF
0	const	227.744081
1	screen_size	7.677290
2	main_camera_mp	2.285051
3	selfie_camera_mp	2.812473
4	int_memory	1.364152
5	ram	2.282352
6	battery	4.081780
7	weight	6.396749
8	days_used	2.660269
9	normalized_new_price	3.119430
10	years_since_release	4.899007
11	brand_name_Alcatel	3.405693
12	brand_name_Apple	13.057668
13	brand_name_Asus	3.332038
14	brand_name_BlackBerry	1.632378
15	brand_name_Celkon	1.774721
16	brand_name_Coolpad	1.468006
17	brand_name_Gionee	1.951272

Index	Feature	VIF
18	brand_name_Google	1.321778
19	brand_name_HTC	3.410361
20	brand_name_Honor	3.340687
21	brand_name_Huawei	5.983852
22	brand_name_Infinix	1.283955
23	brand_name_Karbonn	1.573702
24	brand_name_LG	4.849832
25	brand_name_Lava	1.711360
26	brand_name_Lenovo	4.558941
27	brand_name_Meizu	2.179607
28	brand_name_Micromax	3.363521
29	brand_name_Microsoft	1.869751
30	brand_name_Motorola	3.274558
31	brand_name_Nokia	3.479849
32	brand_name_OnePlus	1.437034
33	brand_name_Oppo	3.971194
34	brand_name_Others	9.711034
35	brand_name_Panasonic	2.105703
36	brand_name_Realme	1.946812
37	brand_name_Samsung	7.539866
38	brand_name_Sony	2.943161
39	brand_name_Spice	1.688863
40	brand_name_Vivo	3.651437
41	brand_name_XOLO	2.138070

Index	Feature	VIF
42	brand_name_Xiaomi	3.719689
43	brand_name_ZTE	3.797581
44	os_Others	1.859863
45	os_Windows	1.596034
46	os_iOS	11.784684
47	4g_yes	2.467681
48	5g_yes	1.813900

Observation – The highlighted variables have VIF > 5, therefore we will try to remove them one by one and see if it will clear the correlated variables and what will be the impact on the model performance

Model Performance Summary – No Multicollinearity – Cont.

Column	Adj. R-squared after dropping col	RMSE after dropping col
brand_name_Apple	0.841809	0.232201
brand_name_Huawei	0.841808	0.232201
brand_name_Others	0.841806	0.232203
os_iOS	0.841795	0.232211
brand_name_Samsung	0.841774	0.232227
screen_size	0.838381	0.234703
weight	0.838071	0.234928

Results – The highlighted variables were removed from the model and the model performance remained high

Model Performance Summary – Dropping high p-value variables

We will drop the predictor variables having a p-value greater than 0.05 as they do not significantly impact the target variable.
Since sometimes p-values change after dropping a variable. So, we'll not drop all variables at once.
Instead, we will do the following:

- Build a model, check the p-values of the variables, and drop the column with the highest p-value.
- Create a new model without the dropped feature, check the results

Model Performance Summary – Second Run Results

Results – The second iteration of the model, without the high VIF and the high P-values variables, show strong results, so we will continue working with it

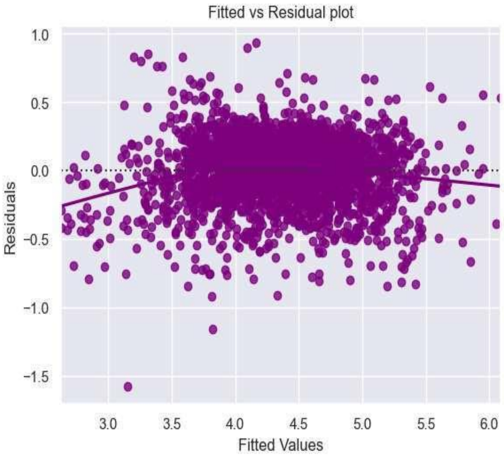
Train performance:

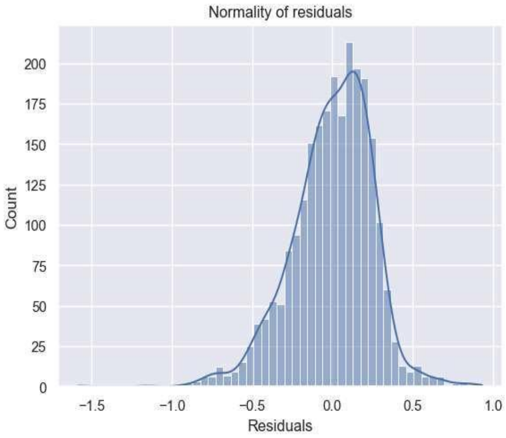
RMSE	MAE	R-squared	Adj. R-squared	MAPE
0.249705	0.194005	0.816984	0.81584	0.046678

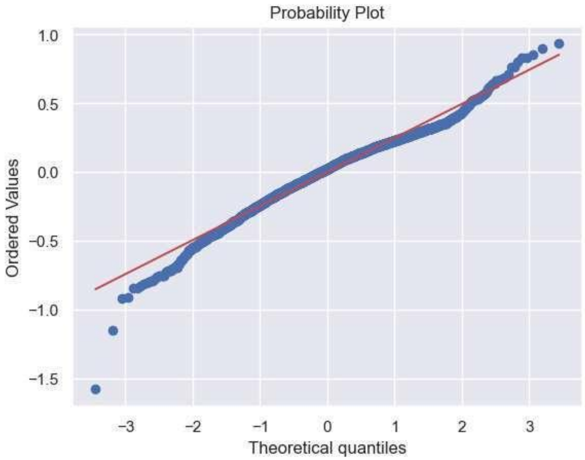
Test performance:

RMSE	MAE	R-squared	Adj. R-squared	MAPE
0.253286	0.196185	0.822131	0.819518	0.047941

Model Performance Summary – Linearity and Independence







Observation – By plotting, we see random relation between the fitted and the residuals and a visible normal distribution of the residuals. Even the Q-Q comparison, shows close enough to normal.

Model Performance Summary – Linearity and Independence – Cont.

Last test will be the Shapiro Test:

ShapiroResult(statistic=0.9790064070477261, **pvalue=2.080160689360709e-18**)

Since p-value < 0.05, the residuals are NOT normal as per Shapiro test. Strictly speaking - the residuals are not normal. However, as an approximation, we might be willing to accept this distribution as close to being normal

Model Performance Summary – Homoscedasticity

We will test for homoscedasticity by using the ‘goldfeldquandt’ test:

Result:

```
('F statistic', 1.000504951705391 4), ('p-value', 0.4965187310899025)
```

Since we got a p-value greater than 0.05, we can say that the residuals are homoscedastic. (The null hypothesis was that they are homoscedastic, and we fail to reject it)

Model Performance Summary – Final Model Performance

Training Set Performance:

RMSE	MAE	R-squared	Adj. R-squared	MAPE
0.249705	0.194005	0.816984	0.81584	0.046678

Test Set Performance:

RMSE	MAE	R-squared	Adj. R-squared	MAPE
0.253286	0.196185	0.822131	0.819518	0.047941

Summary – After adjusting the training and test sets to meet the assumptions, the model keeps its strong performance and shows R-squared of 82% which satisfies our expectations.

APPENDIX

Full Data Set Dictionary

The 2021 data contains the different attributes of used/refurbished phones and tablets.

Name	Description
brand_name	Name of manufacturing brand
os	OS on which the device runs
screen_size	Size of the screen in cm
4g	Whether 4G is available or not
5g	Whether 5G is available or not
main_camera_mp	Resolution of the rear camera in megapixels
selfie_camera_mp	Resolution of the front camera in megapixels
int_memory	Amount of internal memory (ROM) in GB
ram	Amount of RAM in GB
battery	Energy capacity of the device battery in mAh
weight	Weight of the device in grams
release_year	Year when the device model was released
days_used	Year when the device model was released
normalized_new_price	Normalized price of a new device of the same model in euros
normalized_used_price	This will be the independent variable

Model Performance Summary – Initial Run Full Results

OLS Regression Results

Dep. Variable:	normalized_used_price	R-squared:	0.845
Model:	OLS	Adj. R-squared:	0.842
Method:	Least Squares	F-statistic:	268.7
Date:	Sun, 25 Aug 2024	Prob (F-statistic):	0.00
Time:	12:21:20	Log-Likelihood:	123.85
No. Observations:	2417	AIC:	-149.7
Df Residuals:	2368	BIC:	134.0
Df Model:	48		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.3156	0.071	18.454	0.000	1.176	1.455
screen_size	0.0244	0.003	7.163	0.000	0.018	0.031
main_camera_mp	0.0208	0.002	13.848	0.000	0.018	0.024
selfie_camera_mp	0.0135	0.001	11.997	0.000	0.011	0.016
int_memory	0.0001	6.97e-05	1.651	0.099	-2.16e-05	0.000
ram	0.0230	0.005	4.451	0.000	0.013	0.033
battery	-1.689e-05	7.27e-06	-2.321	0.020	-3.12e-05	-2.62e-06
weight	0.0010	0.000	7.480	0.000	0.001	0.001
days_used	4.216e-05	3.09e-05	1.366	0.172	-1.84e-05	0.000
normalized_new_price	0.4311	0.012	35.147	0.000	0.407	0.455
years_since_release	-0.0237	0.005	-5.193	0.000	-0.033	-0.015
brand_name_Alcatel	0.0154	0.048	0.323	0.747	-0.078	0.109
brand_name_Apple	-0.0038	0.147	-0.026	0.980	-0.292	0.285
brand_name_Asus	0.0151	0.048	0.314	0.753	-0.079	0.109
brand_name_BlackBerry	-0.0300	0.070	-0.427	0.669	-0.168	0.108
brand_name_Celkon	-0.0468	0.066	-0.707	0.480	-0.177	0.083
brand_name_Coolpad	0.0209	0.073	0.287	0.774	-0.122	0.164
brand_name_Gionee	0.0448	0.058	0.775	0.438	-0.068	0.158
brand_name_Google	-0.0326	0.085	-0.385	0.700	-0.199	0.133
brand_name_HTC	-0.0130	0.048	-0.270	0.787	-0.108	0.081
brand_name_Honor	0.0317	0.049	0.644	0.520	-0.065	0.128
brand_name_Huawei	-0.0020	0.044	-0.046	0.964	-0.089	0.085

	coef	std err	t	P>|t|	[0.025	0.975]
brand_name_Infinix	0.1633	0.093	1.752	0.080	-0.019	0.346
brand_name_Karbonn	0.0943	0.067	1.405	0.160	-0.037	0.226
brand_name_LG	-0.0132	0.045	-0.291	0.771	-0.102	0.076
brand_name_Lava	0.0332	0.062	0.533	0.594	-0.089	0.155
brand_name_Lenovo	0.0454	0.045	1.004	0.316	-0.043	0.134
brand_name_Meizu	-0.0129	0.056	-0.230	0.818	-0.123	0.097
brand_name_Micromax	-0.0337	0.048	-0.704	0.481	-0.128	0.060
brand_name_Microsoft	0.0952	0.088	1.078	0.281	-0.078	0.268
brand_name_Motorola	-0.0112	0.050	-0.226	0.821	-0.109	0.086
brand_name_Nokia	0.0719	0.052	1.387	0.166	-0.030	0.174
brand_name_OnePlus	0.0709	0.077	0.916	0.360	-0.081	0.223
brand_name_Oppo	0.0124	0.048	0.261	0.794	-0.081	0.106
brand_name_Others	-0.0080	0.042	-0.190	0.849	-0.091	0.075
brand_name_Panasonic	0.0563	0.056	1.008	0.314	-0.053	0.166
brand_name_Realme	0.0319	0.062	0.518	0.605	-0.089	0.153
brand_name_Samsung	-0.0313	0.043	-0.725	0.469	-0.116	0.053
brand_name_Sony	-0.0616	0.050	-1.220	0.223	-0.161	0.037
brand_name_Spice	-0.0147	0.063	-0.233	0.816	-0.139	0.109
brand_name_Vivo	-0.0154	0.048	-0.318	0.750	-0.110	0.080
brand_name_XOLO	0.0152	0.055	0.277	0.782	-0.092	0.123
brand_name_Xiaomi	0.0869	0.048	1.806	0.071	-0.007	0.181
brand_name_ZTE	-0.0057	0.047	-0.121	0.904	-0.099	0.087
os_Others	-0.0510	0.033	-1.555	0.120	-0.115	0.013
os_Windows	-0.0207	0.045	-0.459	0.646	-0.109	0.068
os_iOS	-0.0663	0.146	-0.453	0.651	-0.354	0.221
4g_yes	0.0528	0.016	3.326	0.001	0.022	0.084
5g_yes	-0.0714	0.031	-2.268	0.023	-0.133	-0.010

Model Performance Summary – Final Results

OLS Regression Results

Dep. Variable:	normalized_used_price	R-squared:	0.817
Model:	OLS	Adj. R-squared:	0.816
Method:	Least Squares	F-statistic:	765.9
Date:	Sun, 25 Aug 2024	Prob (F-statistic):	0.00
Time:	13:43:32	Log-Likelihood:	-76.051
No. Observations:	2417	AIC:	182.1
Df Residuals:	2402	BIC:	269.0
Df Model:	14		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.3946	0.051	27.469	0.000	1.295	1.494
main_camera_mp	0.0147	0.001	10.594	0.000	0.012	0.017
selfie_camera_mp	0.0117	0.001	10.374	0.000	0.009	0.014
ram	0.0187	0.005	3.479	0.001	0.008	0.029
battery	9.434e-05	4.99e-06	18.911	0.000	8.46e-05	0.000
normalized_new_price	0.4756	0.011	42.607	0.000	0.454	0.498
years_since_release	-0.0215	0.003	-6.208	0.000	-0.028	-0.015
brand_name_Alcatel	0.0599	0.028	2.109	0.035	0.004	0.116
brand_name_Karbonn	0.1408	0.058	2.414	0.016	0.026	0.255
brand_name_Lenovo	0.0645	0.023	2.783	0.005	0.019	0.110
brand_name_Microsoft	0.1394	0.070	1.996	0.046	0.002	0.276
brand_name_Nokia	0.0715	0.033	2.157	0.031	0.007	0.136
brand_name_Xiaomi	0.0736	0.028	2.676	0.007	0.020	0.128
os_Others	-0.1455	0.030	-4.805	0.000	-0.205	-0.086
5g_yes	-0.0763	0.033	-2.348	0.019	-0.140	-0.013

Happy Learning !