# ReCell – Refurbished Smartphones

## Supervised Learning - Foundations

June 6, 2023

# Contents / Agenda

- Executive Summary
- Business Problem Overview and Solution Approach
- EDA Results
- Data Preprocessing
- Model Performance Summary
- Appendix

# Executive Summary

- Summary of observations and conclusions
    - R-squared of the final model is 84.7%, indicates the model explains about 85% of the variation in the train data.  Model also is not underfitted.
    - MAE and RMSE train and training sets are comparable, indicate model is not overfitting.
    - MAE able to predict normalized_used_price within a mean error of 0.18 euros difference on the test data. The model is a good prediction as well as inferences purposes

# Business Problem Overview and Solution Approach

- Business problem overview:
  - Newly release refurbished phone has a higher price compared the older refurbished phone. It is important for company direction to decide whether it is worth to invest/refurbish/sell much older refurbished phone as the price may decrease quickly.
  - The company may need to focus more sales on the new release refurbished phone as it has higher re-sale value.
  - Smartphones brand Nokia, Samsung and Xiaomi give the high sale value compared with other brand.
  - Customer only looking for basic quality for refurbished smartphones, such as screen size, good quality of ram and internal memory, battery, camera quality and 4g technology availability, hence it is good to sell more of the excellent basic smartphones.

# Business Problem Overview and Solution Approach

- Solution approach/business improvement/recommendation
  - The company needs to collect/perform research for more data on the age of customers, genders and occupation to increase more sales.
  - Since the newly release refurbished phone has a higher price compared the older phone, company staffs need to improve more skill on smartphones refurbishment for readiness on new/advanced smartphone technologies.

# EDA Results – Univariate Analysis

- After checking the data types and screen it, we may need to roughly screen all the information below into meaningful graphs.
- There is about 14 useful data to investigate the information.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3454 entries, 0 to 3453
Data columns (total 15 columns):
 #   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
```

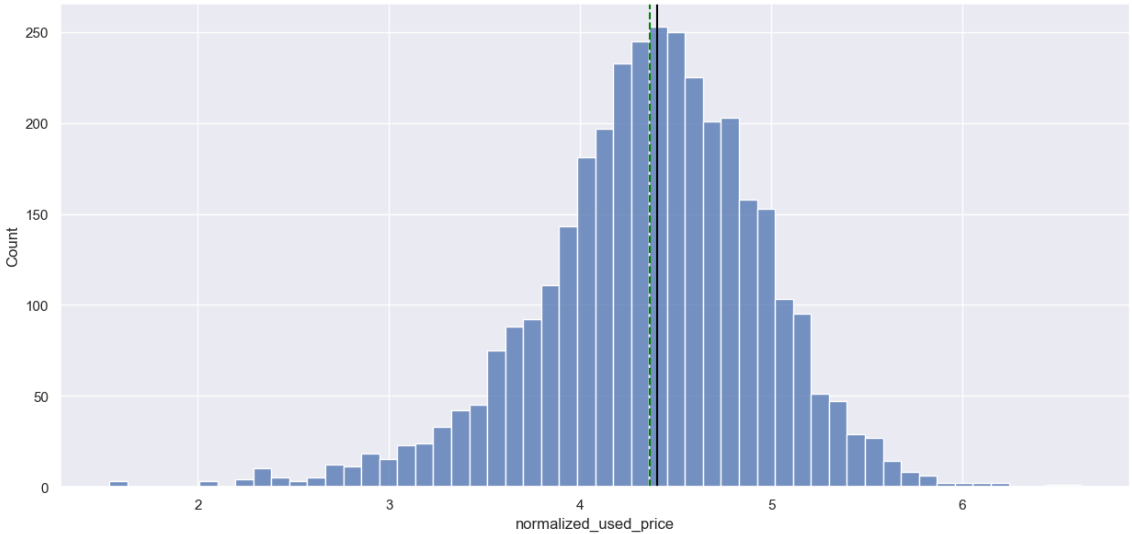
# EDA Results – Univariate Analysis

- Rough statistical summary for variables and categorical value as below:
  - Top famous refurbished smartphone is Android with brand_name Others
  - As 5g is probably quite new to the refurbished smartphones, 4g is the top requirement for market demand.
  - The maximum price for used_price is about 6.6 euros with maximum days 1094.

| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| screen_size | 3454.0 | 13.713115 | 3.805280 | 5.080000 | 12.700000 | 12.830000 | 15.340000 | 30.710000 |
| main_camera_mp | 3275.0 | 9.460208 | 4.815461 | 0.080000 | 5.000000 | 8.000000 | 13.000000 | 48.000000 |
| selfie_camera_mp | 3452.0 | 6.554229 | 6.970372 | 0.000000 | 2.000000 | 5.000000 | 8.000000 | 32.000000 |
| int_memory | 3450.0 | 54.573099 | 84.972371 | 0.010000 | 16.000000 | 32.000000 | 64.000000 | 1024.000000 |
| ram | 3450.0 | 4.036122 | 1.365105 | 0.020000 | 4.000000 | 4.000000 | 4.000000 | 12.000000 |
| battery | 3448.0 | 3133.402697 | 1299.682844 | 500.000000 | 2100.000000 | 3000.000000 | 4000.000000 | 9720.000000 |
| weight | 3447.0 | 182.751871 | 88.413228 | 69.000000 | 142.000000 | 160.000000 | 185.000000 | 855.000000 |
| release_year | 3454.0 | 2015.965258 | 2.298455 | 2013.000000 | 2014.000000 | 2015.500000 | 2018.000000 | 2020.000000 |
| days_used | 3454.0 | 674.869716 | 248.580166 | 91.000000 | 533.500000 | 690.500000 | 868.750000 | 1094.000000 |
| normalized_used_price | 3454.0 | 4.364712 | 0.588914 | 1.536867 | 4.033931 | 4.405133 | 4.755700 | 6.619433 |
| normalized_new_price | 3454.0 | 5.233107 | 0.683637 | 2.901422 | 4.790342 | 5.245892 | 5.673718 | 7.847841 |

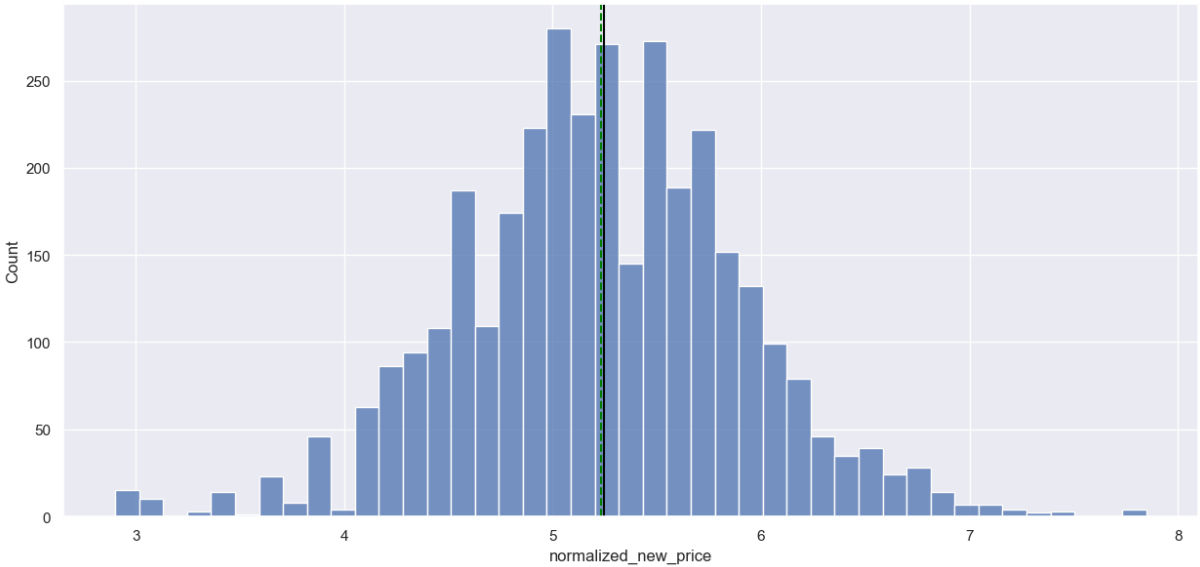
| | count | unique | top | freq |
|---|---|---|---|---|
| brand_name | 3454 | 34 | Others | 502 |
| os | 3454 | 4 | Android | 3214 |
| 4g | 3454 | 2 | yes | 2335 |
| 5g | 3454 | 2 | no | 3302 |

# EDA Results – Univariate Analysis: 1. normalized_used_price

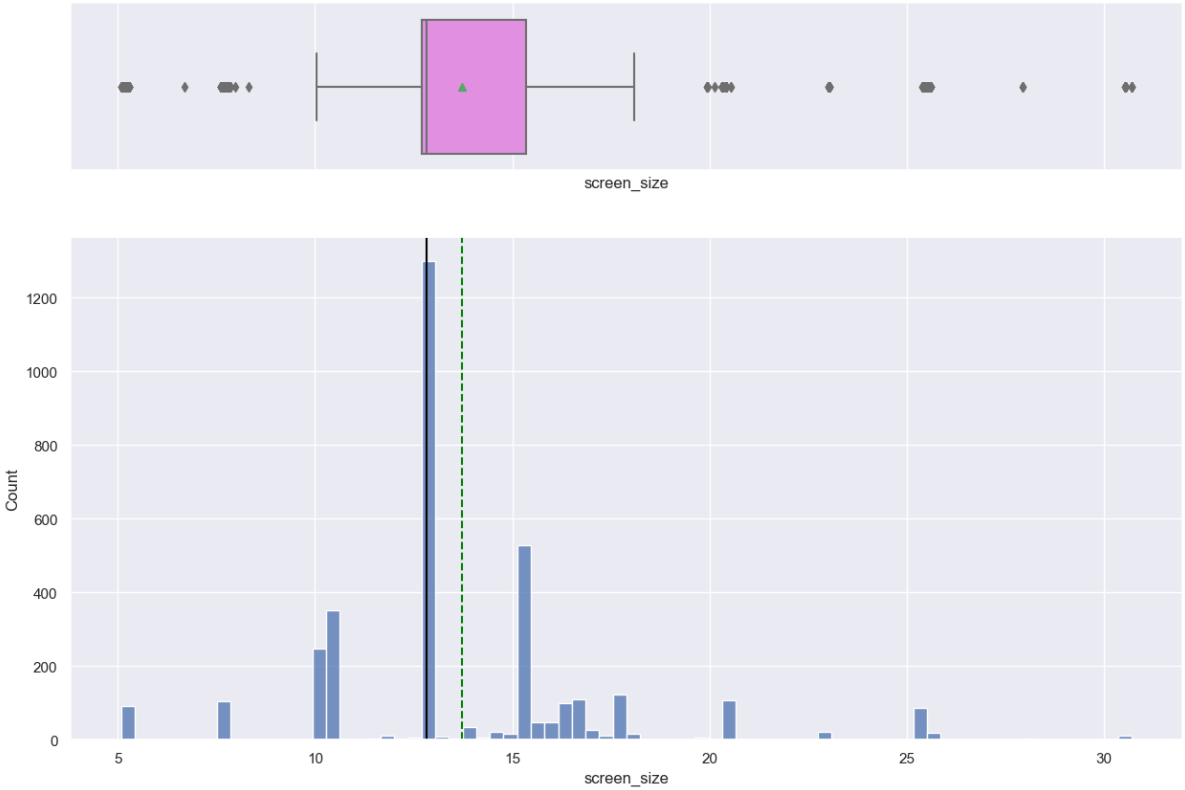

- Normalized_used_price has an outlier for both left and right.
- However, the histogram indicates quite good of bell shape curve and this is a good indication of data is normally distributed.

# EDA Results – Univariate Analysis: 2. normalized_new_price

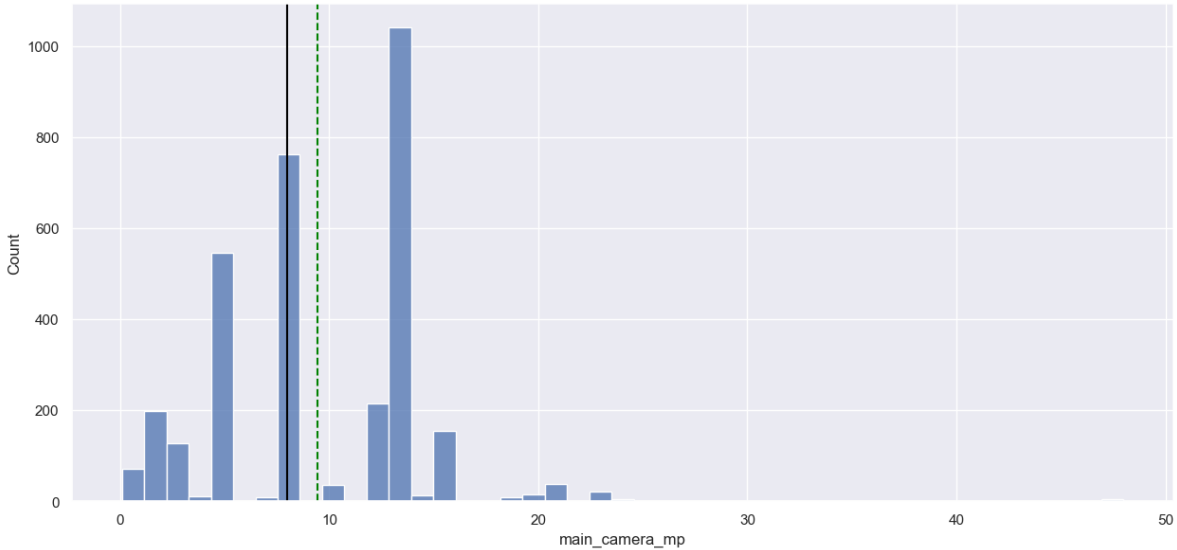

- Normalized_new_price also has an outlier for both left and right.
- Histogram also shows a good indication of data is normally distributed.

# EDA Results – Univariate Analysis: 3. screen_size



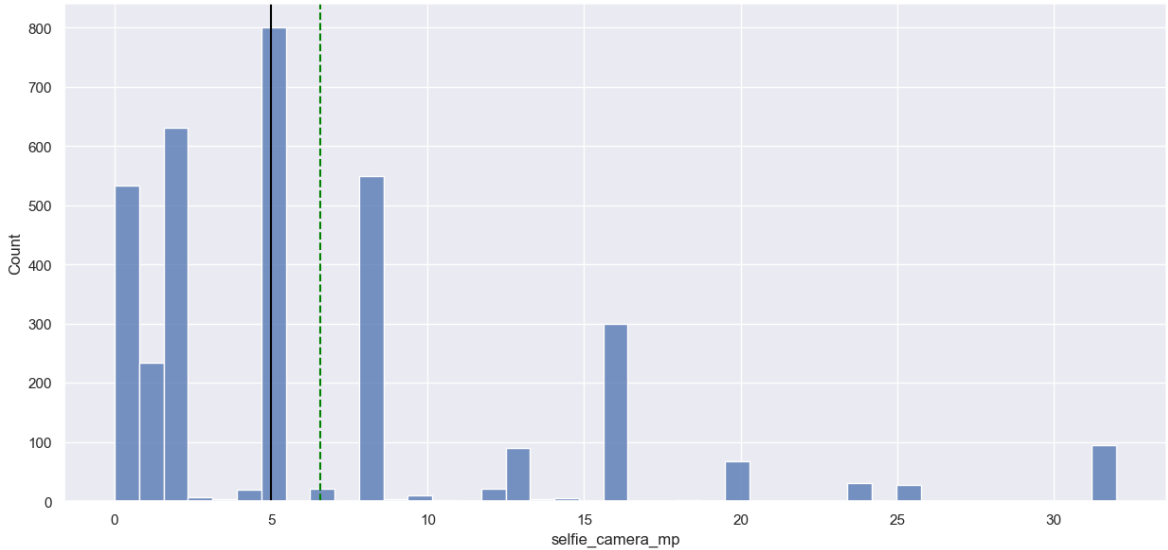

- Screen_size also has an outlier for both left and right.
- The histogram shows data is sparse and widely distributed.

# EDA Results – Univariate Analysis: 4. main_camera_mp

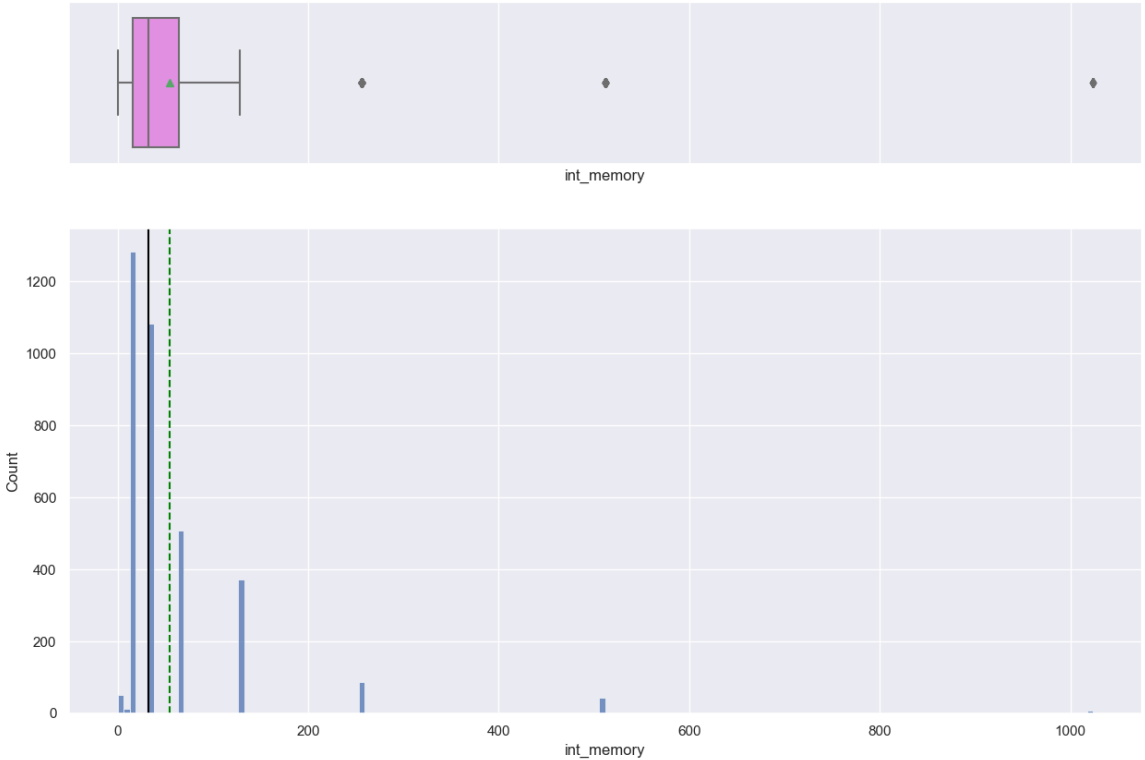

- Main_camera_mp has some outlier at the right side.
- The histogram shows data is widely distributed.

# EDA Results – Univariate Analysis: 5. selfie_camera_mp

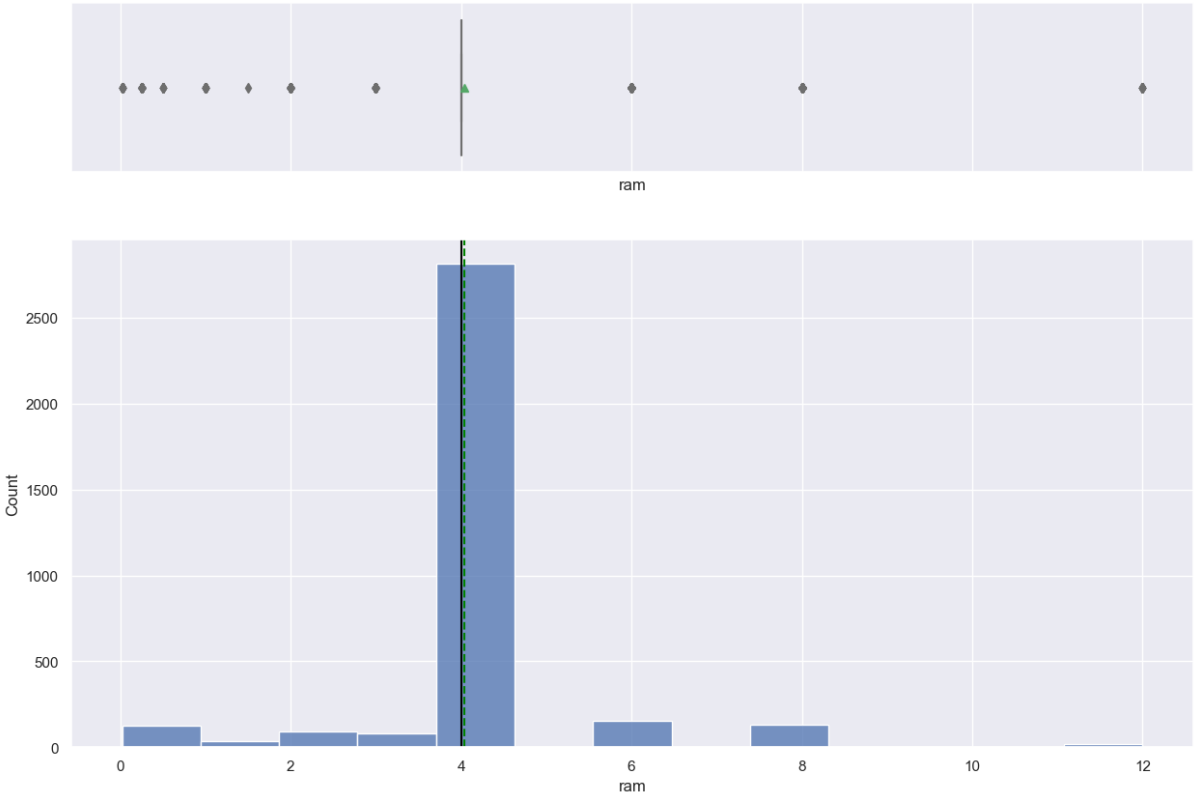

- Selfie_camera_mp also has some outlier at the right side.
- The histogram shows data is widely distributed and a little bit skewed at the right side.

# EDA Results – Univariate Analysis: 6. int_memory



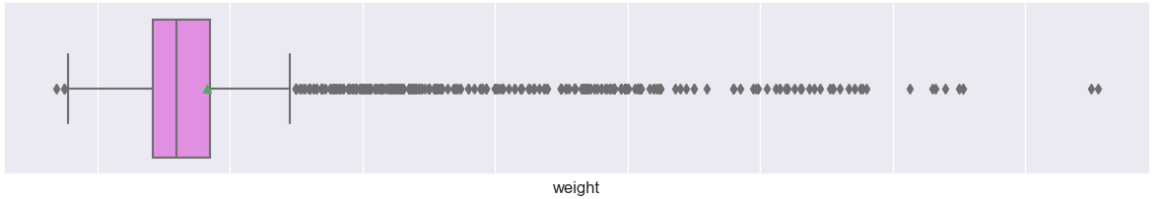

- Int_memory also has some outlier at the right side.
- The histogram shows data is a little bit skewed at the right side.

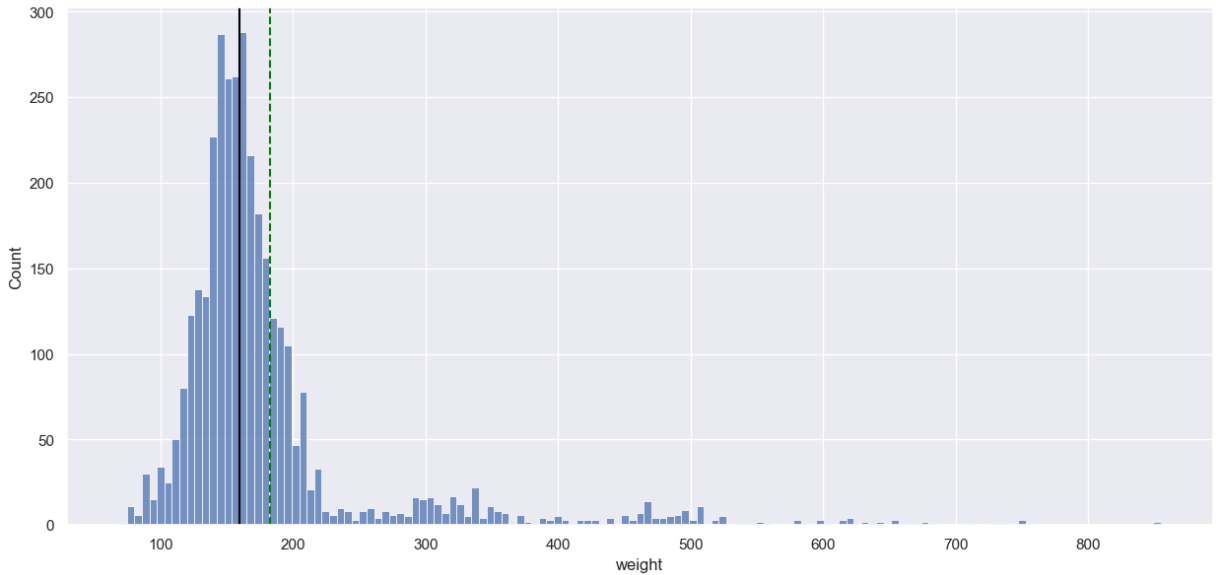
# EDA Results – Univariate Analysis: 7. ram




- ram also has some outlier at the left and right side.
- The histogram shows data widely distributed, with 4GB is the highest frequency for ram.

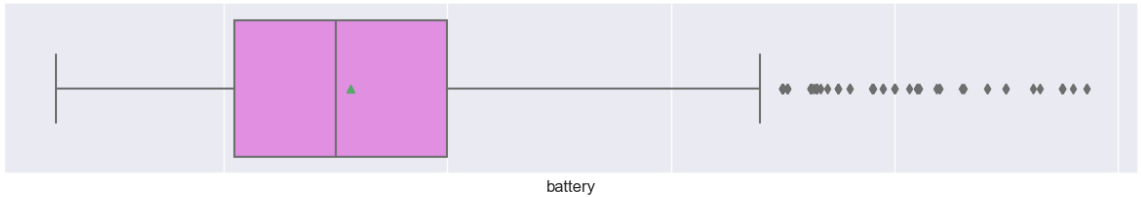
# EDA Results – Univariate Analysis: 8. weight

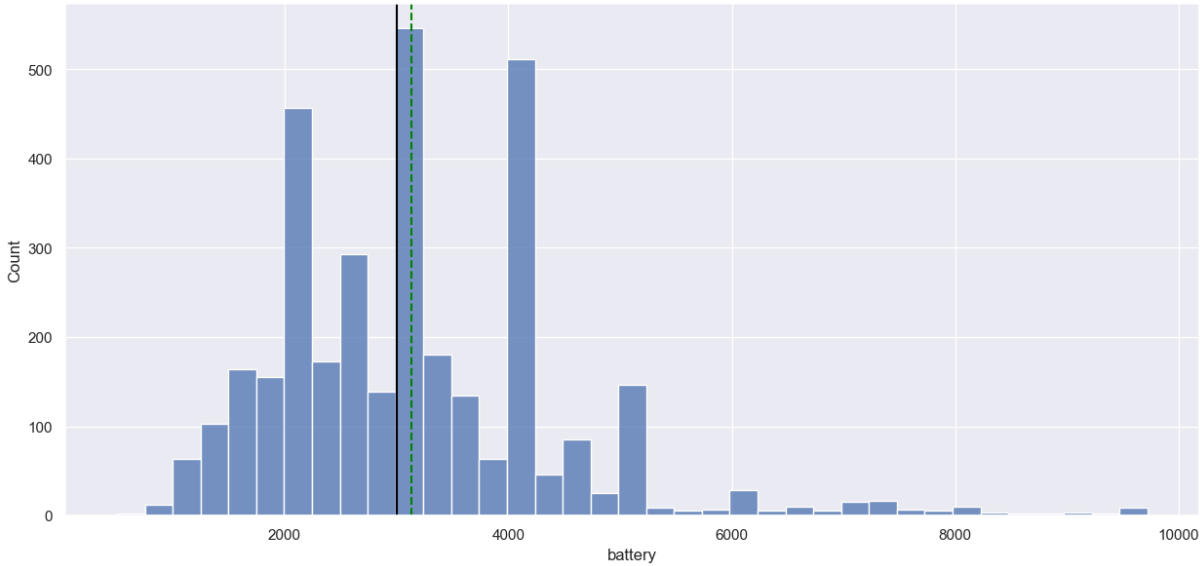



- Smartphone weight has a lot of outlier at the right side.
- The histogram shows data is rightly skewed.

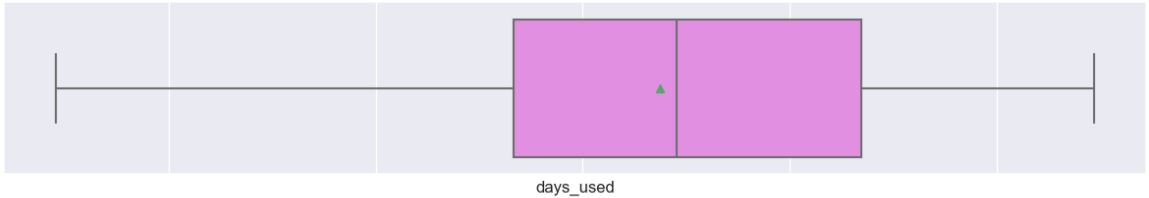
# EDA Results – Univariate Analysis: 9. battery

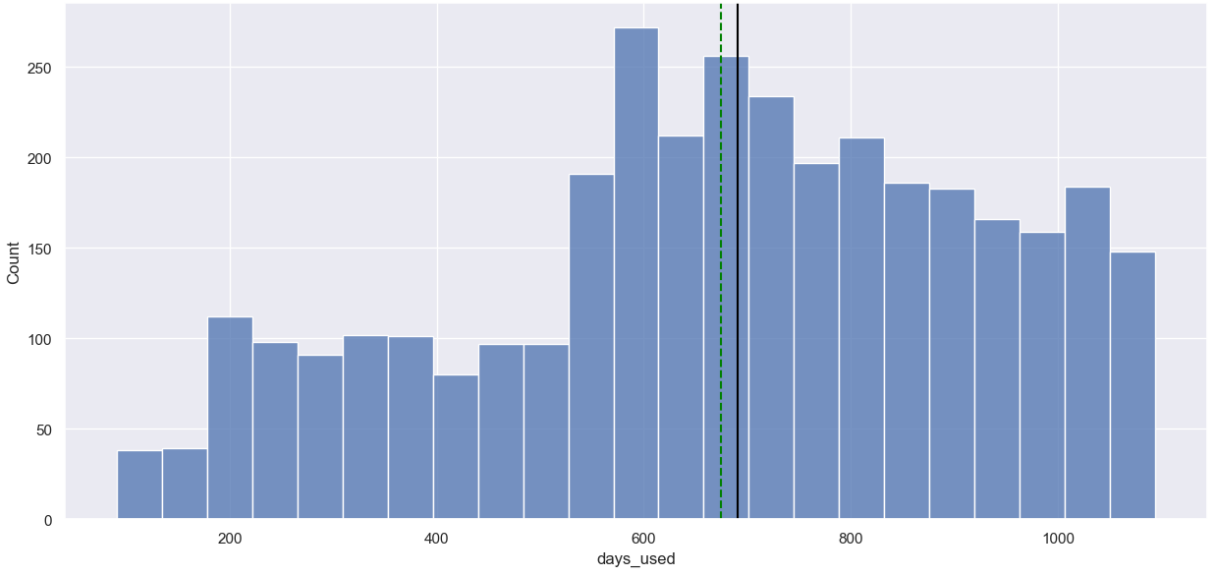



- Smartphone battery has a lot of outlier at the right side.
- The histogram shows data is rightly skewed.

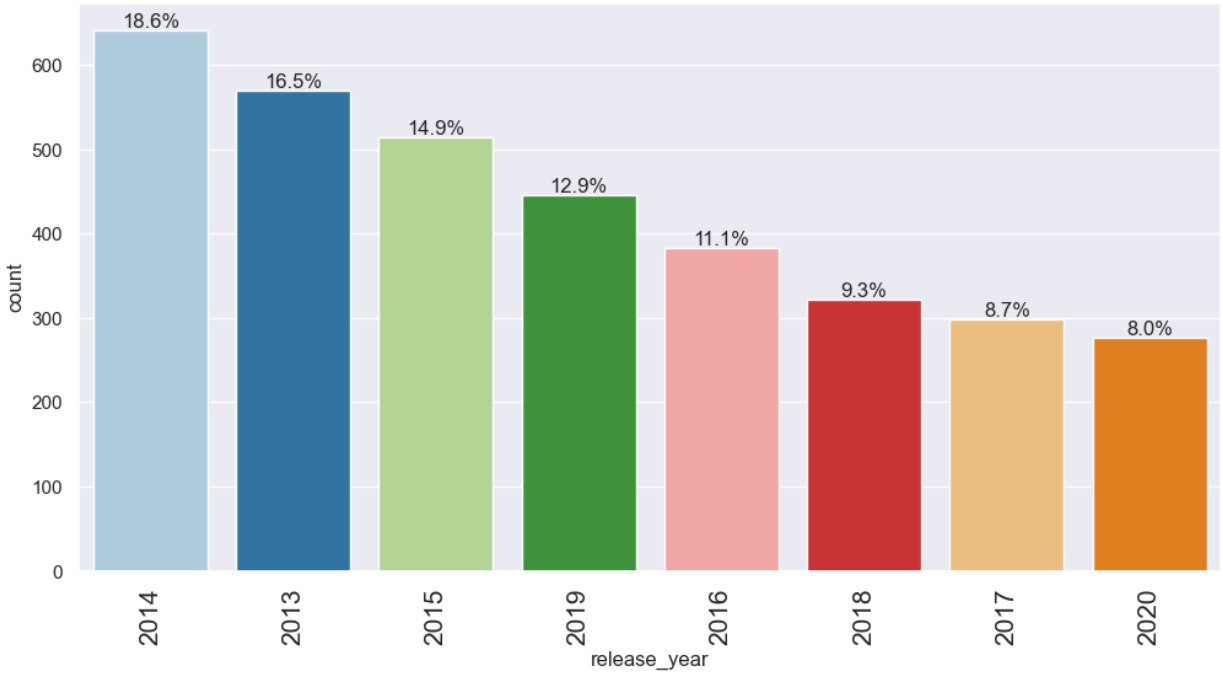
# EDA Results – Univariate Analysis: 10. days_used



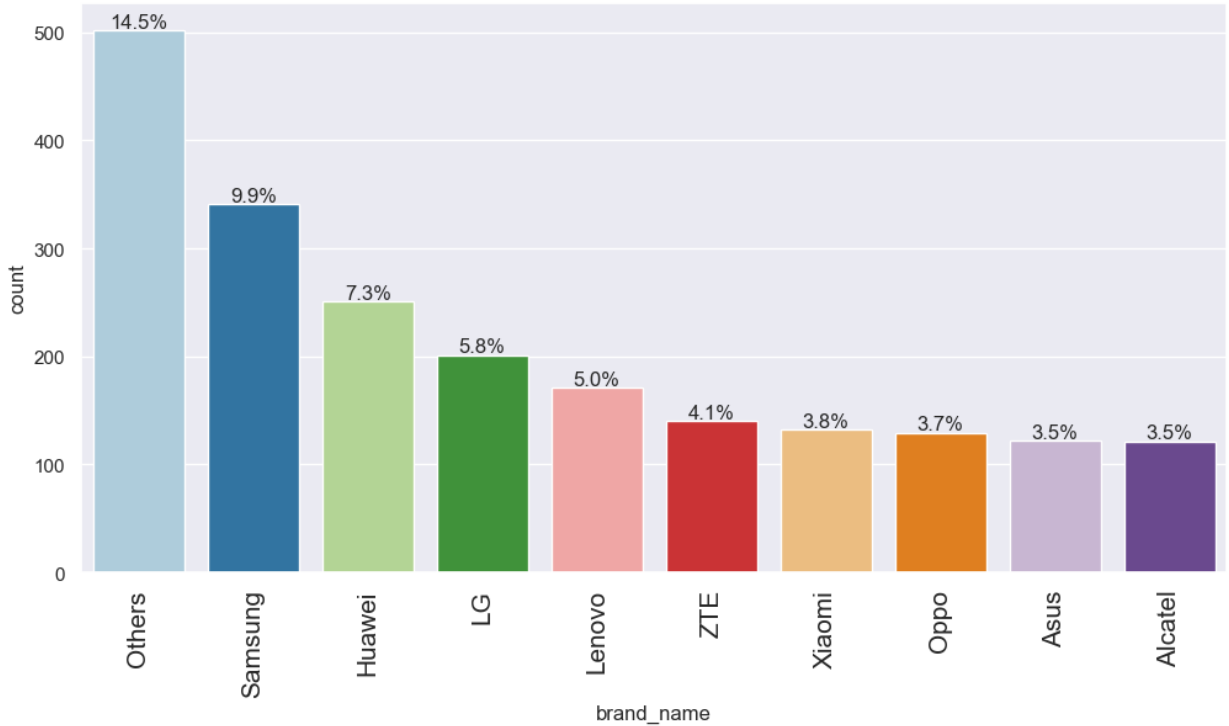

- Smartphone days_used has no outlier but having a wide range of data with average 675 days
- The histogram shows data is widely distributed with slightly skewed to the left side.

# EDA Results – Univariate Analysis: 11. release_year



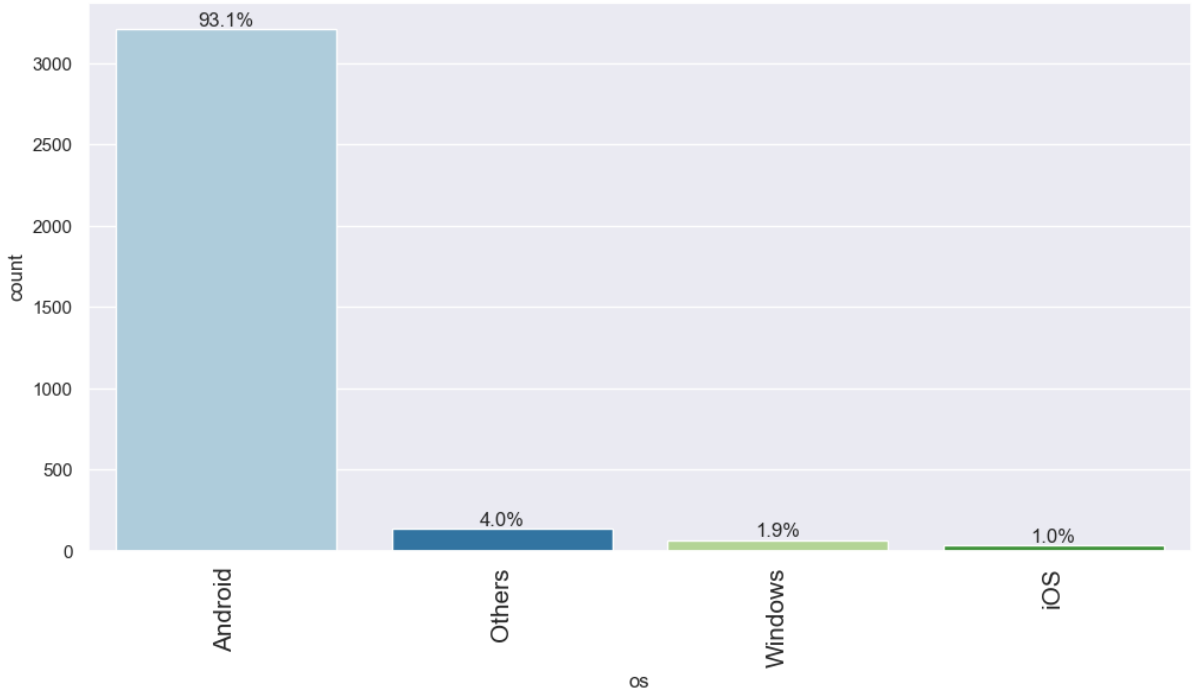

- 2014 release_year is the highest on refurbished smartphone with contribution of 18.6%, followed by 2013 (16.5%) and 2015 (14.9%)

# EDA Results – Univariate Analysis: 12. brand_name



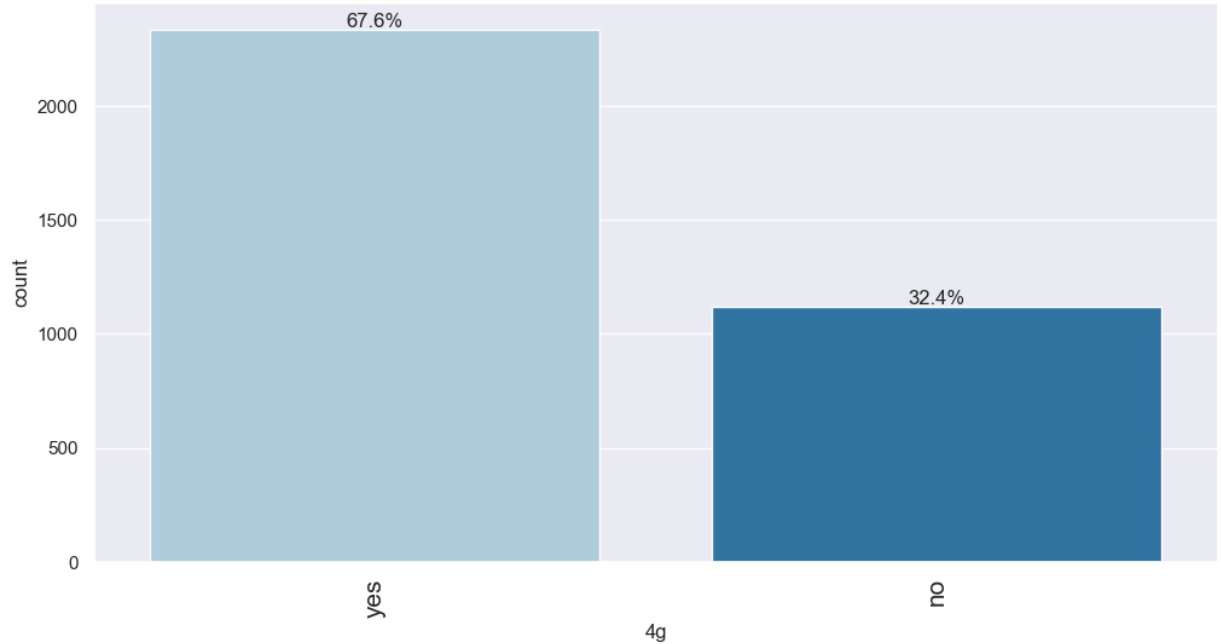

- Brand_name Others shows the highest contribution, estimated 14.5% of refurbished smartphones, followed by Samsung (9.9%) and Huawei (7.3%)

# EDA Results – Univariate Analysis: 13. os



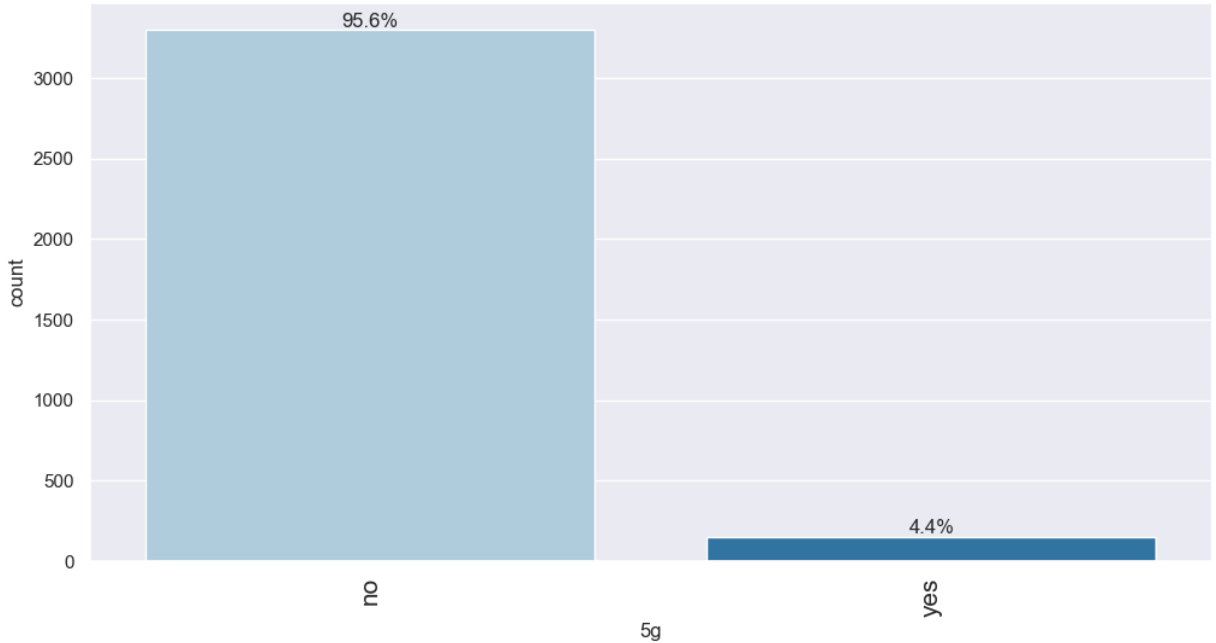

- Android dominates the most contributed os for refurbished smartphones, which is 93.1%, while other os may not give high contribution.
- Nevertheless, Others is contributed about 4% and Windows contributed about 1.9%

# EDA Results – Univariate Analysis: 14. 4g



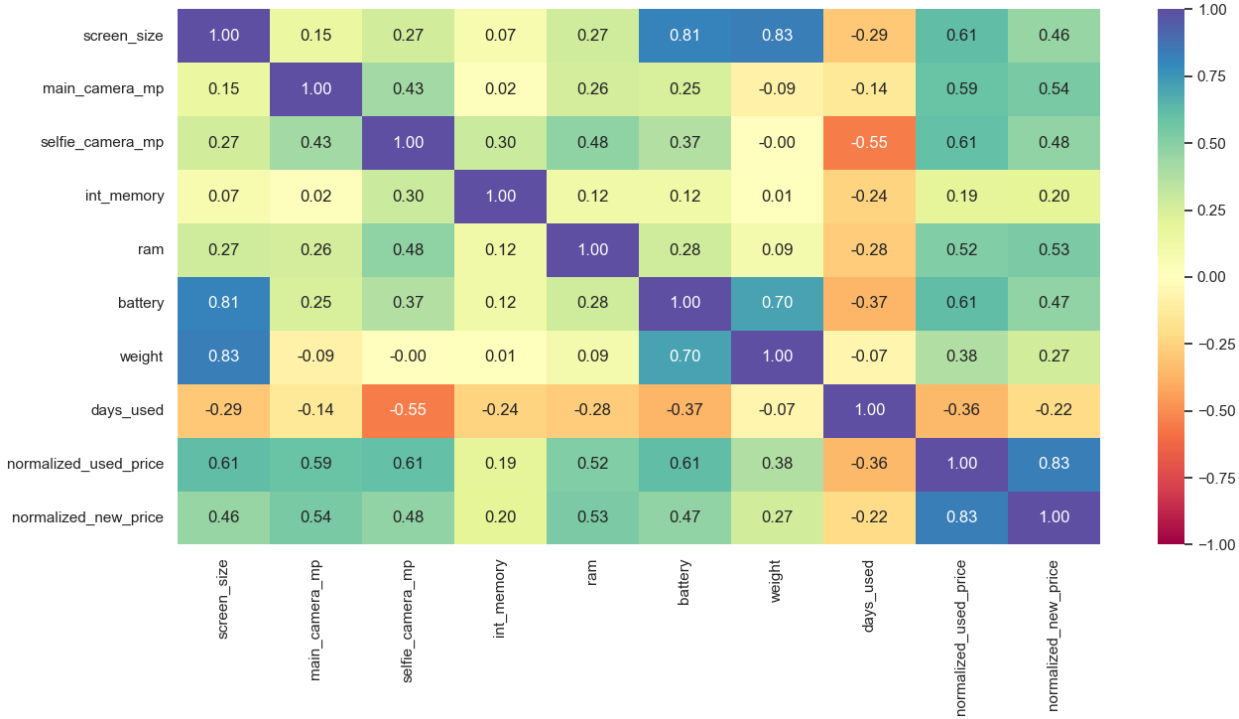

- Refurbished smartphones contains 4g function is having about 67.6% while 32.4% has no 4g function.

# EDA Results – Univariate Analysis: 15. 5g



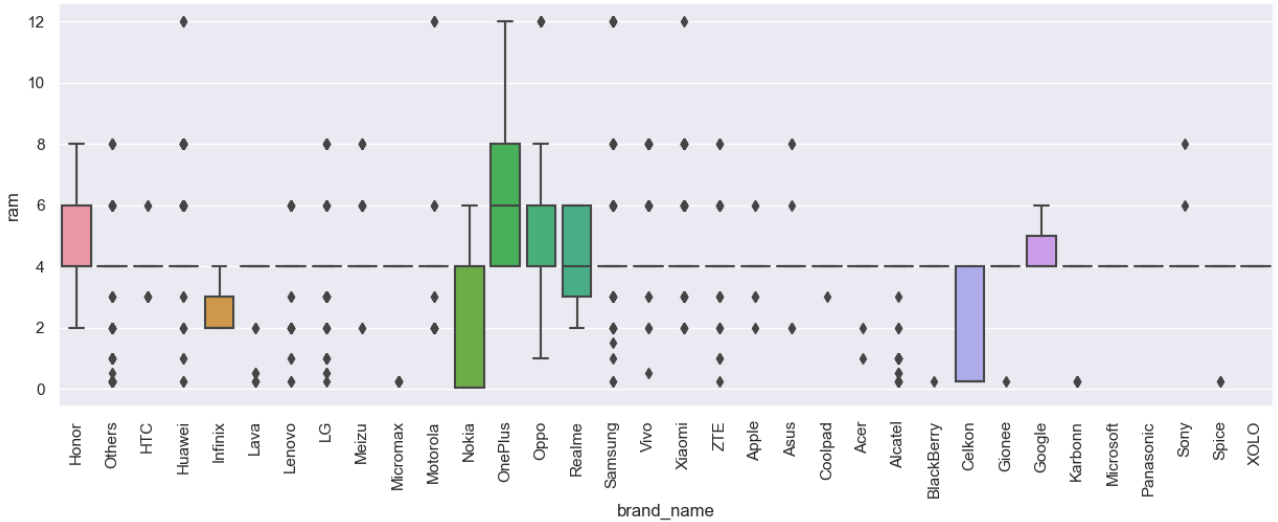

- Refurbished smartphones contains 5g function is dominates about 95.6% while 4.4% has no 5g function.

# EDA Results – Bivariate Analysis: 1. Correlation check



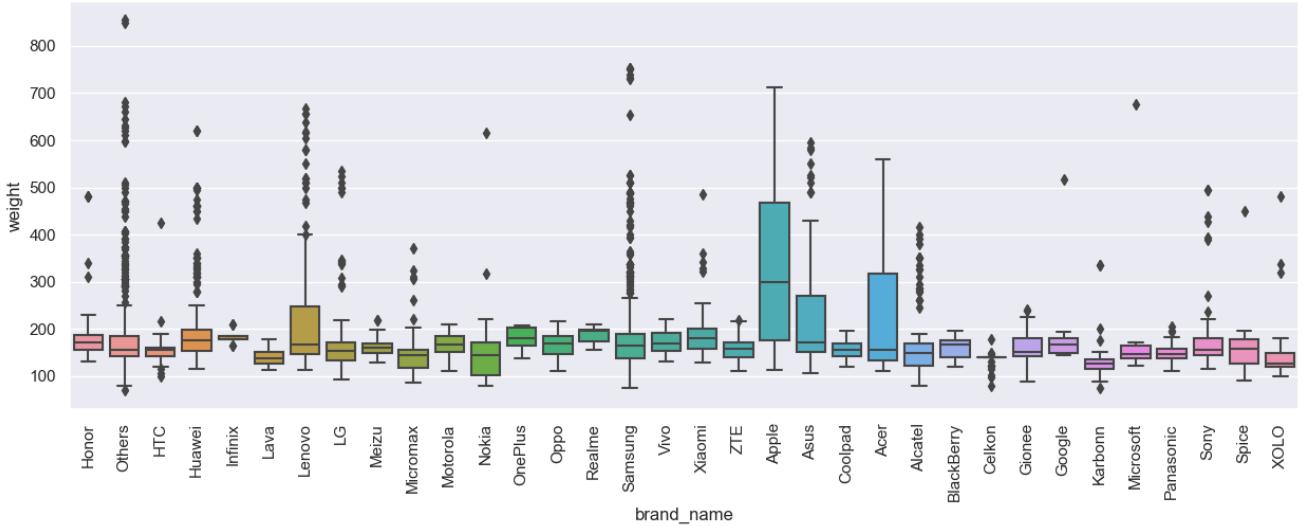

- Normalized_used_price is high correlated with normalized_new_price. This may indicates when the normalized_new_price is increase, the normalized_used_price is also increasing.
- Other high correlated item is smartphones battery and its weight.

# EDA Results – Bivariate Analysis: 2. Amount of RAM varies across brands



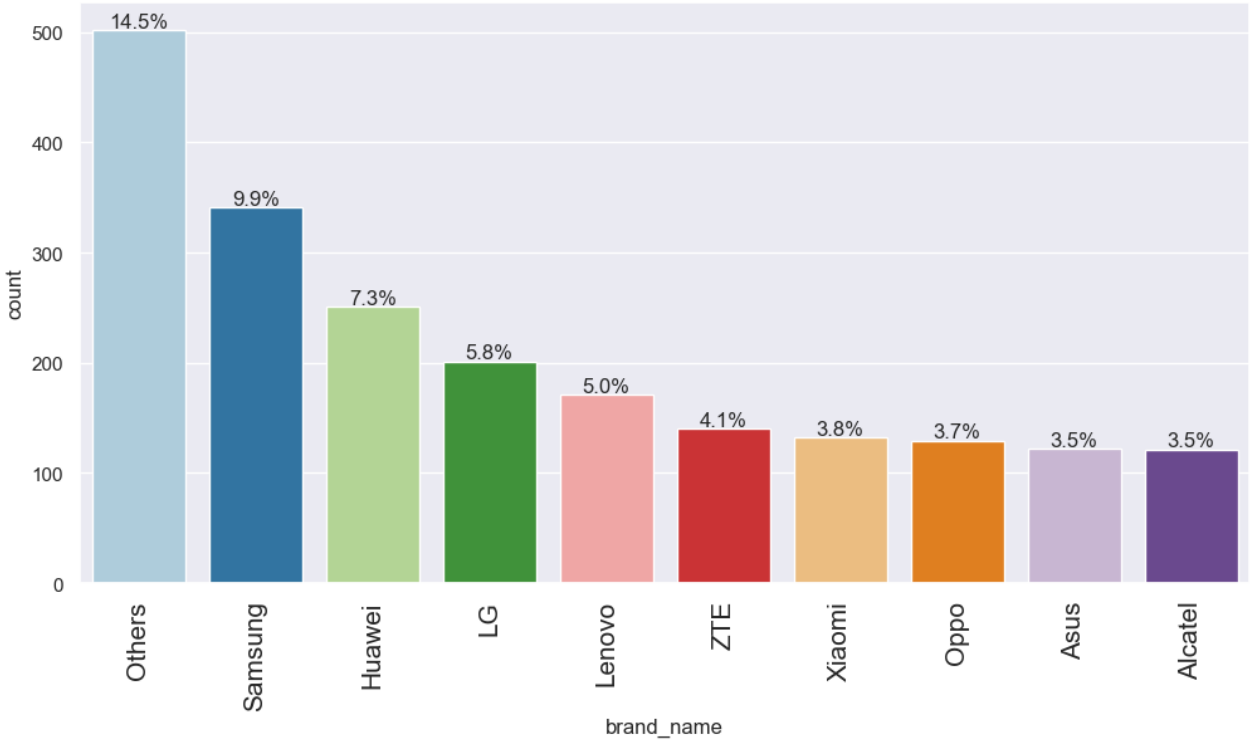

- Most brands are concentrated on below of 4GB of ram, while some brand also shows they are also having more of 4GB of ram on refurbished smartphones.
- 3 brand_name is having minimum of 4GB of ram is OnePlus, Google and Sony.

# EDA Results – Bivariate Analysis: 3. Devices which offer a large battery



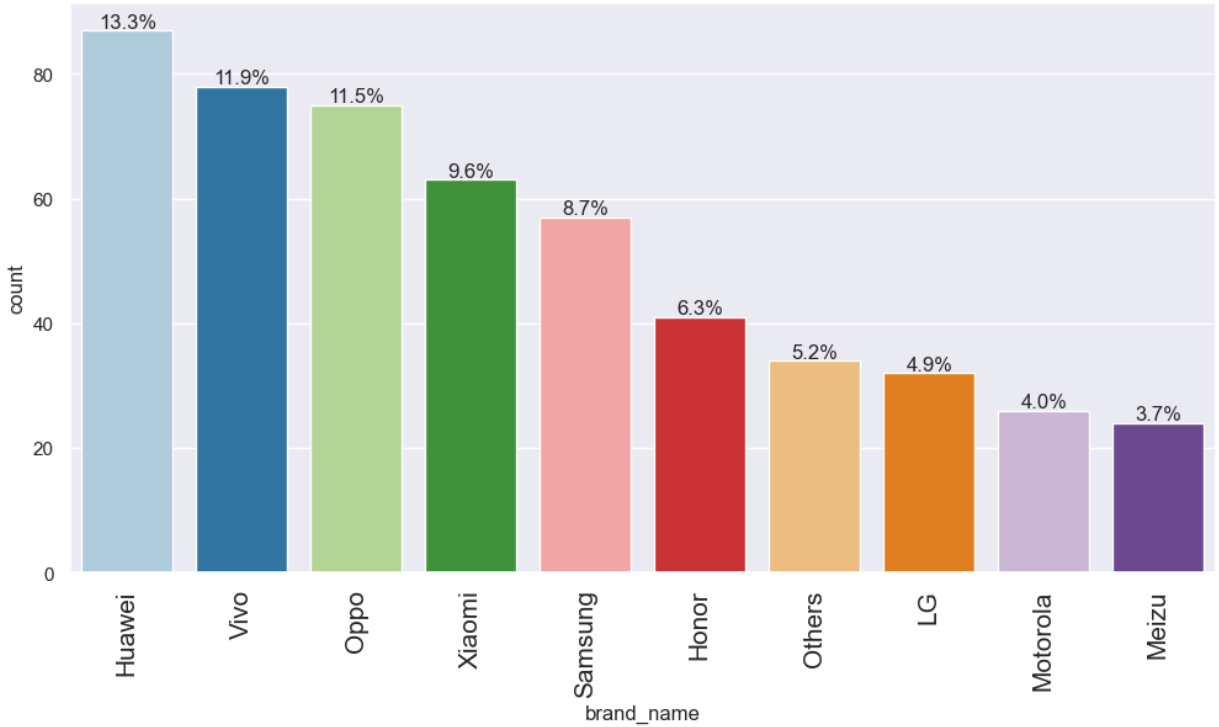

- All brands are having around similar weight of smartphones, with frequency ranging 100-200g.
- However, some smartphones brand is seen having more outliers/more weight than 200g, with brand Apple is having a most wide range compared with other brands.

# EDA Results – Bivariate Analysis: 4. Devices with large screen primarily for entertainment



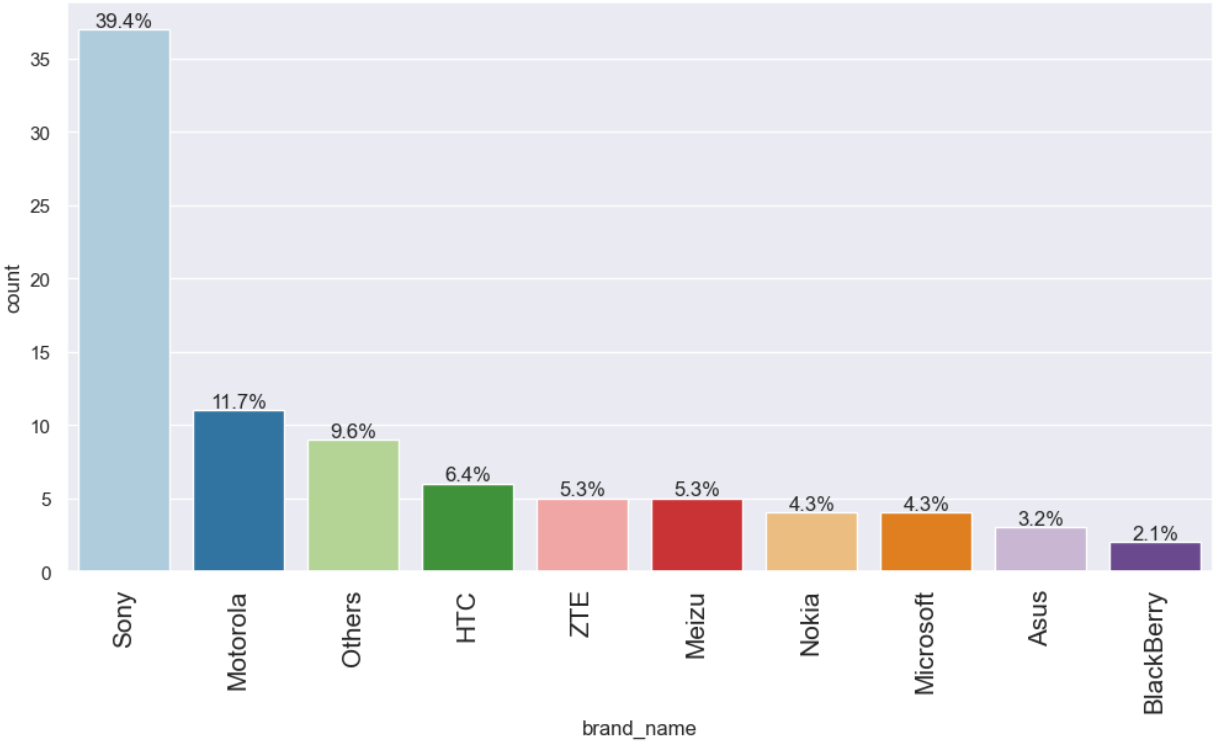

- Others smartphones dominate highest contribution with largest screen (14.5%).
- Samsung and Huawei is followed closely with each contribution is 9.9% and 7.3% respectively.

# EDA Results – Bivariate Analysis: 5. Good front camera to click cool selfies



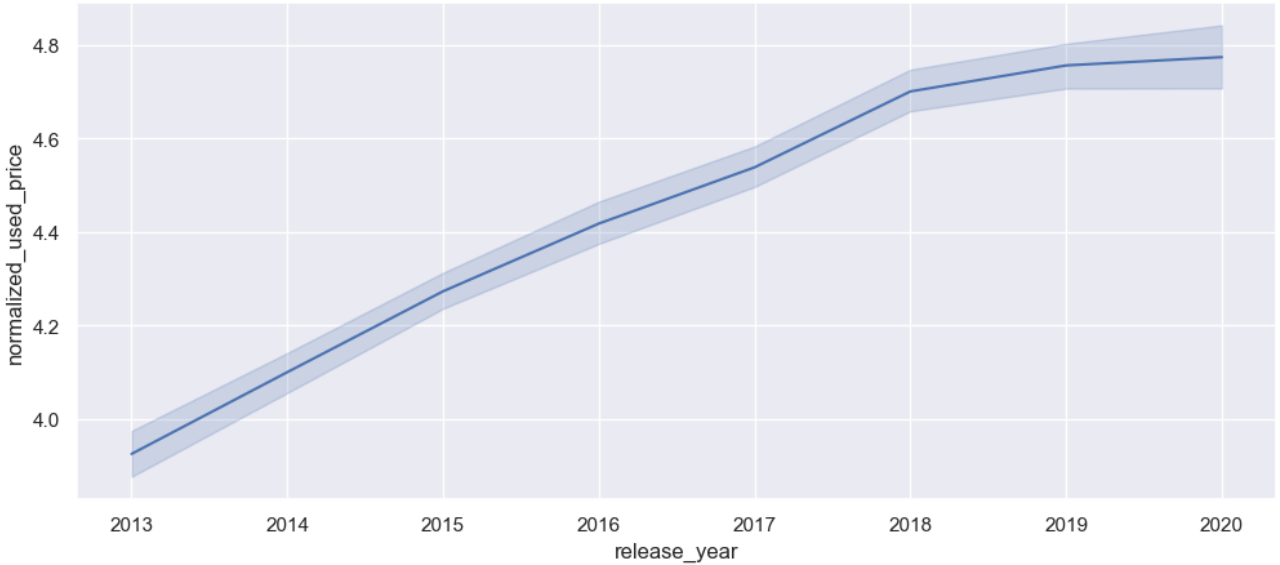

- Huawei becomes the highest contributor for good front camera, which is 13.3%, followed closely by Vivo (11.9%) and Oppo (11.5%)

# EDA Results – Bivariate Analysis: 6. Good rear camera to click cool pictures



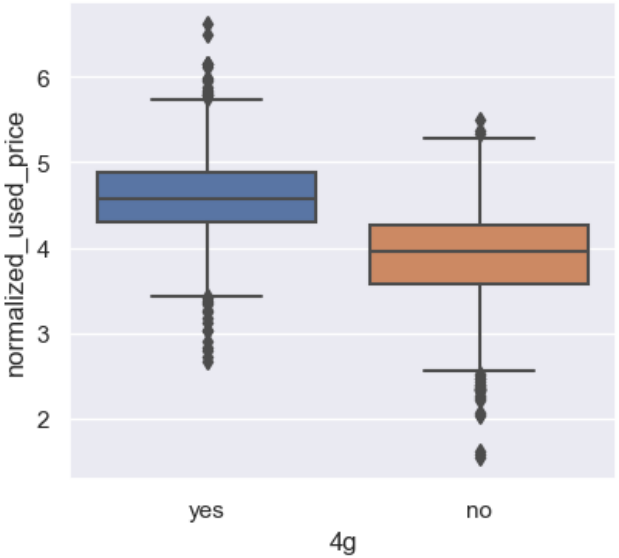

- Sony becomes the highest contributor for good rear camera, which is 39.4%, followed with wide gap by Motorola (11.7%) and Others (9.6%)

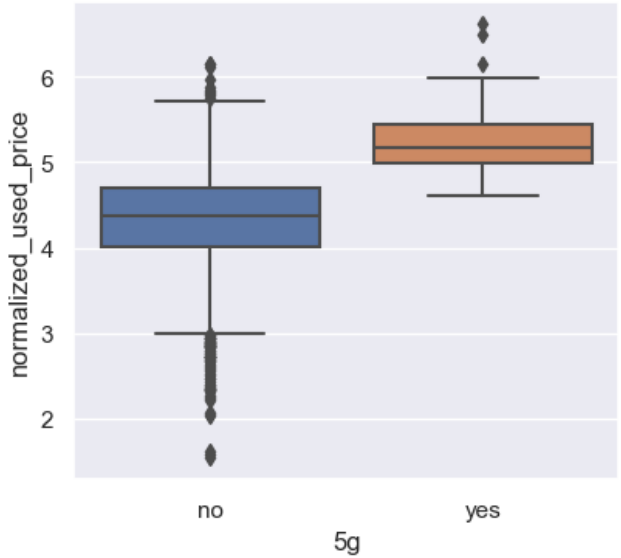
# EDA Results – Bivariate Analysis: 7. Price of used devices across the years




- Refurbished smartphones release_year is increasing with the increasing of normalized_used_price. It is keep increasing since the baseline of 2013.

# EDA Results – Bivariate Analysis: 8. Phones and tablets offering 4G and 5G networks




- Normalized_used_price can be seen increasing with the adaptable of new technology on 4g and 5g.
- The average used price of smartphones having 4g is estimated about 4.6 euros while 5g is 5.2 euros.

# Data Preprocessing – Check for duplicate and missing value

- Duplicate value check

  data.duplicated().sum()

  - No value was duplicated from data given

- Checking for missing value

  - Missing value from lowest to highest quantity:

    - Selfie_camera_mp, int_memory, ram, battery, weight, main_camera_mp

```
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
```

# Data Preprocessing – Missing value treatment

- Missing value treatment – by using group median on release_year, brand_name and main_camera_mp
- Re-confirm missing value treatment after imputation with median treatment

```
brand_name             0
os                     0
screen_size            0
4g                     0
5g                     0
main_camera_mp         0
selfie_camera_mp       0
int_memory             0
ram                    0
battery                0
weight                 0
release_year           0
days_used              0
normalized_used_price  0
normalized_new_price   0
dtype: int64
```

- No missing value after treatment

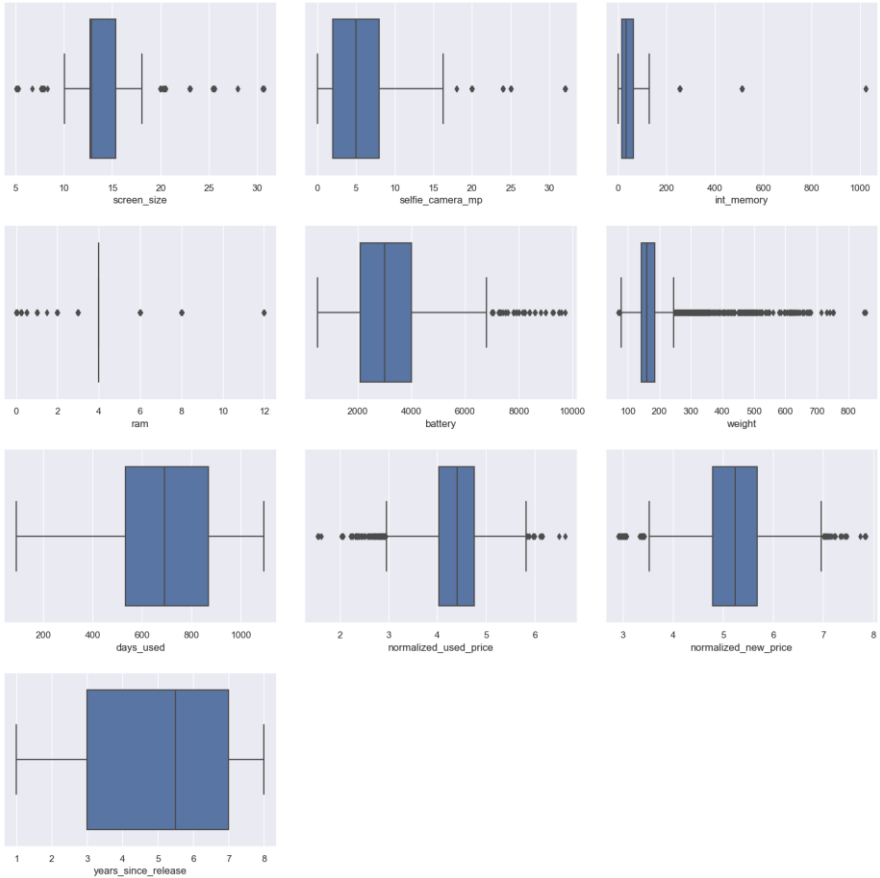
# Data Preprocessing – Feature Engineering

- Now we need to create new column for years_since_release column, and take 2021 year as baseline. df.head() as below:

| | brand_name | os | screen_size | 4g | 5g | main_camera_mp | selfie_camera_mp | int_memory | ram | battery | weight | days_used | normalized_used_price | normalized_new_price | years_since_release |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Honor | Android | 14.50 | yes | no | 13.0 | 5.0 | 64.0 | 3.0 | 3020.0 | 146.0 | 127 | 4.307572 | 4.715100 | 1 |
| 1 | Honor | Android | 17.30 | yes | yes | 13.0 | 16.0 | 128.0 | 8.0 | 4300.0 | 213.0 | 325 | 5.162097 | 5.519018 | 1 |
| 2 | Honor | Android | 16.69 | yes | yes | 13.0 | 8.0 | 128.0 | 8.0 | 4200.0 | 213.0 | 162 | 5.111084 | 5.884631 | 1 |
| 3 | Honor | Android | 25.50 | yes | yes | 13.0 | 8.0 | 64.0 | 6.0 | 7250.0 | 480.0 | 345 | 5.135387 | 5.630961 | 1 |
| 4 | Honor | Android | 15.32 | yes | no | 13.0 | 8.0 | 64.0 | 3.0 | 5000.0 | 185.0 | 293 | 4.389995 | 4.947837 | 1 |

```
count    3454.000000
mean        5.034742
std         2.298455
min         1.000000
25%         3.000000
50%         5.500000
75%         7.000000
max         8.000000
Name: years_since_release, dtype: float64
```

# Data Preprocessing – Outlier check




- There are a lot of outlier data can be found on the checked item.
- However, we will not remove it to prevent from information loss during predicting the model.

# Data Preprocessing – Data preparation for modeling

- Output for prediction: normalized price of used device
  - Define dependant and independent variables, (data.head()) below
  - Encode categorical feature
  - Split the data into train and test data, by define the dependant and independent variable
  - Build linear regression using train data and check its performance

```
0    4.307572
1    5.162097
2    5.111084
3    5.135387
4    4.389995
Name: normalized_used_price, dtype: float64
```

# Data Preprocessing – Creating dummy variables

- data.head() – contains 5 rows, 89 columns

| | const | screen_size | selfie_camera_mp | int_memory | ram | battery | weight | days_used | normalized_new_price | years_since_release | ... | main_camera_mp_20.1 | main_camera_mp_12.6 | main_camera_mp_16.3 | main_camera_mp_22.6 | main_camera_mp_19.0 | main_camera_mp_21.5 | main_camera_mp_21.2 | main_camera_mp_8.1 | main_camera_mp_1.2 | main_camera_mp_22.5 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1.0 | 14.50 | 5.0 | 64.0 | 3.0 | 3020.0 | 146.0 | 127 | 4.715100 | 1 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **1** | 1.0 | 17.30 | 16.0 | 128.0 | 8.0 | 4300.0 | 213.0 | 325 | 5.519018 | 1 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **2** | 1.0 | 16.69 | 8.0 | 128.0 | 8.0 | 4200.0 | 213.0 | 162 | 5.884631 | 1 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **3** | 1.0 | 25.50 | 8.0 | 64.0 | 6.0 | 7250.0 | 480.0 | 345 | 5.630961 | 1 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **4** | 1.0 | 15.32 | 8.0 | 64.0 | 3.0 | 5000.0 | 185.0 | 293 | 4.947837 | 1 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

# Data Preprocessing – splitting test and train data

- Split data into 70:30 ratio

```
print("Number of rows in train data =", x_train.shape[0])
print("Number of rows in test data =", x_test.shape[0])
```

```
Number of rows in train data = 2417
Number of rows in test data = 1037
```

# Data Preprocessing – Building linear regression

- Result

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | normalized_used_price | R-squared: | 0.850 |
| Model: | OLS | Adj. R-squared: | 0.845 |
| Method: | Least Squares | F-statistic: | 159.2 |
| Date: | Wed, 31 May 2023 | Prob (F-statistic): | 0.00 |
| Time: | 11:35:56 | Log-Likelihood: | 163.90 |
| No. Observations: | 2417 | AIC: | -159.8 |
| Df Residuals: | 2333 | BIC: | 326.6 |
| Df Model: | 83 | | |
| Covariance Type: | nonrobust | | |

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| const | 1.6815 | 0.080 | 20.959 | 0.000 | 1.524 | 1.839 |
| screen_size | 0.0238 | 0.003 | 6.865 | 0.000 | 0.017 | 0.031 |
| selfie_camera_mp | 0.0137 | 0.001 | 12.146 | 0.000 | 0.012 | 0.016 |
| int_memory | 0.0002 | 7.09e-05 | 2.259 | 0.024 | 2.11e-05 | 0.000 |
| ram | 0.0223 | 0.005 | 4.274 | 0.000 | 0.012 | 0.032 |

# Data Preprocessing – Model performance check

- Training performance

| | RMSE | MAE | R-squared | Adj. R-squared | MAPE |
|---|---|---|---|---|---|
| 0 | 0.226107 | 0.17736 | 0.849942 | 0.844202 | 4.247918 |

- Testing performance

| | RMSE | MAE | R-squared | Adj. R-squared | MAPE |
|---|---|---|---|---|---|
| 0 | 0.226107 | 0.17736 | 0.849942 | 0.844202 | 4.247918 |

- Both data on training and testing is comparable.  Data is not overfitting.

# Checking Linear Regression Assumptions – 1. Checking multicollinearity

- Checking for VIF
  - Found more than 80 data of VIF, 10 of them is higher than 5 and some of them contain NaN – os_iOS, brand_name_Others, brand_name_Huawei, brand_name_Apple, main_camera_mp_1.0, main_camera_mp_41.0, main_camera_mp_20.1, main_camera_mp_12.6, brand_name_Infinix and brand_name_Samsung

| | feature | VIF |
|---|---|---|
| 0 | const | 293.713519 |
| 1 | screen_size | 8.072753 |
| 2 | selfie_camera_mp | 2.897750 |
| 3 | int_memory | 1.434004 |
| 4 | ram | 2.375248 |
| 5 | battery | 4.285870 |
| 6 | weight | 6.972076 |
| 7 | days_used | 2.751173 |
| 8 | normalized_new_price | 3.464667 |
| 9 | years_since_release | 5.260309 |
| 10 | brand_name_Alcatel | 3.436328 |
| 11 | brand_name_Apple | 13.339101 |
| 12 | brand_name_Asus | 3.377646 |
| 13 | brand_name_BlackBerry | 1.748348 |
| 14 | brand_name_Celkon | 1.835217 |
| 15 | brand_name_Coolpad | 1.481877 |
| 16 | brand_name_Gionee | 1.978906 |
| 17 | brand_name_Google | 4.619072 |
| 18 | brand_name_HTC | 3.562150 |
| 19 | brand_name_Honor | 3.410125 |
| 20 | brand_name_Huawei | 6.200095 |
| 21 | brand_name_Infinix | inf |
| 22 | brand_name_Karbonn | 1.588284 |
| 23 | brand_name_LG | 4.949061 |
| 24 | brand_name_Lava | 1.738618 |
| 25 | brand_name_Lenovo | 4.646126 |
| 26 | brand_name_Meizu | 2.224413 |
| 27 | brand_name_Micromax | 3.411905 |
| 28 | brand_name_Microsoft | 1.941682 |
| 29 | brand_name_Motorola | 3.387973 |
| 30 | brand_name_Nokia | 3.587112 |
| 31 | brand_name_OnePlus | 1.451525 |
| 32 | brand_name_Oppo | 4.054459 |
| 33 | brand_name_Others | 9.908586 |
| 34 | brand_name_Panasonic | 2.143026 |
| 35 | brand_name_Realme | 1.983867 |
| 36 | brand_name_Samsung | 7.745273 |
| 37 | brand_name_Sony | 3.584545 |
| 38 | brand_name_Spice | 1.719226 |
| 39 | brand_name_Vivo | 3.725951 |
| 40 | brand_name_XOLO | 2.168078 |
| 41 | brand_name_Xiaomi | 3.838877 |
| 42 | brand_name_ZTE | 3.967755 |
| 43 | os_Others | 2.183287 |
| 44 | os_Windows | 1.808120 |
| 45 | os_iOS | 11.918108 |
| 46 | 4g_yes | 2.593308 |
| 48 | main_camera_mp_8.0 | 1.762953 |
| 49 | main_camera_mp_5.0 | 2.175234 |
| 50 | main_camera_mp_10.5 | 1.241867 |
| 51 | main_camera_mp_3.15 | 1.559747 |
| 52 | main_camera_mp_<bound method NDFrame._add_nume... | inf |
| 53 | main_camera_mp_2.0 | 1.793736 |
| 54 | main_camera_mp_16.0 | 1.174057 |
| 55 | main_camera_mp_0.3 | 1.687279 |
| 56 | main_camera_mp_12.0 | 1.363908 |
| 57 | main_camera_mp_14.5 | 1.140618 |
| 58 | main_camera_mp_48.0 | 1.043227 |
| 59 | main_camera_mp_3.0 | 1.047813 |
| 60 | main_camera_mp_21.0 | 1.063097 |
| 61 | main_camera_mp_1.3 | 1.406044 |
| 62 | main_camera_mp_13.1 | 1.028512 |
| 63 | main_camera_mp_24.0 | 1.022395 |
| 64 | main_camera_mp_0.08 | 1.037308 |
| 65 | main_camera_mp_20.7 | 1.156673 |
| 66 | main_camera_mp_23.0 | 1.192150 |
| 67 | main_camera_mp_1.0 | NaN |
| 68 | main_camera_mp_18.0 | 1.067228 |
| 69 | main_camera_mp_12.2 | 4.057327 |
| 70 | main_camera_mp_12.3 | 1.320221 |
| 71 | main_camera_mp_20.0 | 1.052247 |
| 72 | main_camera_mp_20.2 | 1.015961 |
| 72 | main_camera_mp_20.2 | 1.015961 |
| 73 | main_camera_mp_4.0 | 1.094703 |
| 74 | main_camera_mp_12.5 | 1.034819 |
| 75 | main_camera_mp_10.0 | 1.066051 |
| 76 | main_camera_mp_6.5 | 1.019133 |
| 77 | main_camera_mp_6.7 | 1.227826 |
| 78 | main_camera_mp_41.0 | NaN |
| 79 | main_camera_mp_20.1 | NaN |
| 80 | main_camera_mp_12.6 | NaN |
| 81 | main_camera_mp_16.3 | 1.021789 |
| 82 | main_camera_mp_22.6 | 1.011516 |
| 83 | main_camera_mp_19.0 | 1.176157 |
| 84 | main_camera_mp_21.5 | 1.033344 |
| 85 | main_camera_mp_21.2 | 1.059116 |
| 86 | main_camera_mp_8.1 | 1.137137 |
| 87 | main_camera_mp_1.2 | 1.043345 |
| 88 | main_camera_mp_22.5 | 1.013937 |

# Checking Linear Regression Assumptions – 2. Removing multicollinearity

- 10 with high VIF – drop os_iOS

| | col | Adj. R-squared after_dropping col | RMSE after dropping col |
|---|---|---|---|
| **0** | os_iOS | 0.844667 | 0.230093 |
| **1** | brand_name_Others | 0.844664 | 0.230096 |
| **2** | brand_name_Huawei | 0.844653 | 0.230104 |
| **3** | brand_name_Apple | 0.844631 | 0.230121 |
| **4** | main_camera_mp_1.0 | 0.844603 | 0.230141 |
| **5** | main_camera_mp_41.0 | 0.844603 | 0.230141 |
| **6** | main_camera_mp_20.1 | 0.844603 | 0.230141 |
| **7** | main_camera_mp_12.6 | 0.844603 | 0.230141 |
| **8** | brand_name_Infinix | 0.844603 | 0.230141 |
| **9** | brand_name_Samsung | 0.844589 | 0.230151 |

# Checking Linear Regression Assumptions – 3. Reconfirm multicollinearity

- Duplicate value check

| | feature | VIF |
|---|---|---|
| 0 | const | 293.219202 |
| 1 | screen_size | 8.003282 |
| 2 | selfie_camera_mp | 2.889453 |
| 3 | int_memory | 1.433948 |
| 4 | ram | 2.362290 |
| 5 | battery | 4.281403 |
| 6 | weight | 6.956962 |
| 7 | days_used | 2.750427 |
| 8 | normalized_new_price | 3.463698 |
| 9 | years_since_release | 5.259441 |
| 10 | brand_name_Alcatel | 3.436325 |
| 11 | brand_name_Apple | 2.202002 |
| 12 | brand_name_Asus | 3.377637 |
| 13 | brand_name_BlackBerry | 1.743071 |
| 14 | brand_name_Celkon | 1.835215 |
| 15 | brand_name_Coolpad | 1.481821 |
| 16 | brand_name_Gionee | 1.978761 |
| 17 | brand_name_Google | 4.618790 |
| 18 | brand_name_HTC | 3.561738 |
| 19 | brand_name_Honor | 3.409920 |
| 20 | brand_name_Huawei | 6.199815 |
| 21 | brand_name_Infinix | inf |
| 22 | brand_name_Karbonn | 1.588028 |
| 23 | brand_name_LG | 4.948910 |
| 24 | brand_name_Lava | 1.738443 |
| 25 | brand_name_Lenovo | 4.646126 |
| 26 | brand_name_Meizu | 2.224413 |
| 27 | brand_name_Micromax | 3.411482 |
| 28 | brand_name_Microsoft | 1.941576 |
| 29 | brand_name_Motorola | 3.387969 |
| 30 | brand_name_Nokia | 3.583185 |
| 31 | brand_name_OnePlus | 1.451477 |
| 32 | brand_name_Oppo | 4.054341 |
| 33 | brand_name_Others | 9.908202 |
| 34 | brand_name_Panasonic | 2.142948 |
| 35 | brand_name_Realme | 1.983694 |
| 36 | brand_name_Samsung | 7.744479 |
| 37 | brand_name_Sony | 3.584325 |
| 38 | brand_name_Spice | 1.718615 |
| 39 | brand_name_Vivo | 3.725889 |
| 40 | brand_name_XOLO | 2.167914 |
| 41 | brand_name_Xiaomi | 3.838508 |
| 42 | brand_name_ZTE | 3.967182 |
| 43 | os_Others | 2.031078 |
| 44 | os_Windows | 1.807009 |
| 45 | 4g_yes | 2.592217 |
| 46 | 5g_yes | 1.926578 |
| 47 | main_camera_mp_8.0 | 1.762742 |
| 48 | main_camera_mp_5.0 | 2.174966 |
| 49 | main_camera_mp_10.5 | 1.241863 |
| 50 | main_camera_mp_3.15 | 1.559727 |
| 51 | main_camera_mp_<bound method NDFrame._add_nume... | inf |
| 52 | main_camera_mp_2.0 | 1.790517 |
| 53 | main_camera_mp_16.0 | 1.173522 |
| 54 | main_camera_mp_0.3 | 1.677736 |
| 55 | main_camera_mp_12.0 | 1.362789 |
| 56 | main_camera_mp_14.5 | 1.140377 |
| 57 | main_camera_mp_48.0 | 1.043162 |
| 58 | main_camera_mp_3.0 | 1.047747 |
| 59 | main_camera_mp_21.0 | 1.063045 |
| 60 | main_camera_mp_1.3 | 1.400436 |
| 61 | main_camera_mp_13.1 | 1.028511 |
| 62 | main_camera_mp_24.0 | 1.022394 |
| 63 | main_camera_mp_0.08 | 1.036240 |
| 64 | main_camera_mp_20.7 | 1.156651 |
| 65 | main_camera_mp_23.0 | 1.192129 |
| 66 | main_camera_mp_1.0 | NaN |
| 67 | main_camera_mp_18.0 | 1.066858 |
| 68 | main_camera_mp_12.2 | 4.057217 |
| 69 | main_camera_mp_12.3 | 1.320106 |
| 70 | main_camera_mp_20.0 | 1.052235 |
| 71 | main_camera_mp_20.2 | 1.015957 |
| 72 | main_camera_mp_4.0 | 1.094701 |
| 73 | main_camera_mp_12.5 | 1.034751 |
| 74 | main_camera_mp_10.0 | 1.066001 |
| 75 | main_camera_mp_6.5 | 1.019125 |
| 76 | main_camera_mp_6.7 | 1.227689 |
| 77 | main_camera_mp_41.0 | NaN |
| 78 | main_camera_mp_20.1 | NaN |
| 79 | main_camera_mp_12.6 | NaN |
| 80 | main_camera_mp_16.3 | 1.021683 |
| 81 | main_camera_mp_22.6 | 1.011503 |
| 82 | main_camera_mp_19.0 | 1.176141 |
| 83 | main_camera_mp_21.5 | 1.033341 |
| 84 | main_camera_mp_21.2 | 1.059108 |
| 85 | main_camera_mp_8.1 | 1.136492 |
| 86 | main_camera_mp_1.2 | 1.043312 |
| 87 | main_camera_mp_22.5 | 1.013923 |

# Checking Linear Regression Assumptions – 4. Dropping high P-value

- High P-value after dropping, as below:
  - 'screen_size', 'selfie_camera_mp', 'int_memory', 'ram', 'battery', 'weight', 'normalized_new_price', 'years_since_release', 'brand_name_Nokia', 'brand_name_Samsung', 'brand_name_Xiaomi', '4g_yes', 'main_camera_mp_8.0', 'main_camera_mp_5.0', 'main_camera_mp_3.15', 'main_camera_mp_2.0', 'main_camera_mp_16.0', 'main_camera_mp_0.3', 'main_camera_mp_48.0', 'main_camera_mp_1.3', 'main_camera_mp_0.08', 'main_camera_mp_23.0', 'main_camera_mp_4.0', 'main_camera_mp_10.0', 'main_camera_mp_6.7', 'main_camera_mp_8.1'

# Checking Linear Regression Assumptions – 5. Re-check Modeling and its performance

- Re-build a new model, and re-check its training and test performance

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | normalized_used_price | R-squared: | 0.847 |
| Model: | OLS | Adj. R-squared: | 0.846 |
| Method: | Least Squares | F-statistic: | 509.9 |
| Date: | Wed, 31 May 2023 | Prob (F-statistic): | 0.00 |
| Time: | 13:20:56 | Log-Likelihood: | 142.48 |
| No. Observations: | 2417 | AIC: | -231.0 |
| Df Residuals: | 2390 | BIC: | -74.61 |
| Df Model: | 26 | | |
| Covariance Type: | nonrobust | | |

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| const | 1.7385 | 0.058 | 29.979 | 0.000 | 1.625 | 1.852 |
| screen_size | 0.0240 | 0.003 | 7.635 | 0.000 | 0.018 | 0.030 |
| selfie_camera_mp | 0.0140 | 0.001 | 13.088 | 0.000 | 0.012 | 0.016 |
| int_memory | 0.0001 | 6.77e-05 | 2.023 | 0.043 | 4.17e-06 | 0.000 |
| ram | 0.0182 | 0.004 | 4.104 | 0.000 | 0.009 | 0.027 |
| battery | -1.493e-05 | 7.05e-06 | -2.118 | 0.034 | -2.87e-05 | -1.11e-06 |
| weight | 0.0011 | 0.000 | 8.331 | 0.000 | 0.001 | 0.001 |
| normalized_new_price | 0.4031 | 0.011 | 35.417 | 0.000 | 0.381 | 0.425 |
| years_since_release | -0.0157 | 0.004 | -4.258 | 0.000 | -0.023 | -0.008 |
| brand_name_Nokia | 0.0909 | 0.031 | 2.938 | 0.003 | 0.030 | 0.152 |
| brand_name_Samsung | -0.0356 | 0.016 | -2.199 | 0.028 | -0.067 | -0.004 |
| brand_name_Xiaomi | 0.0690 | 0.025 | 2.718 | 0.007 | 0.019 | 0.119 |
| 4g_yes | 0.0495 | 0.015 | 3.291 | 0.001 | 0.020 | 0.079 |
| main_camera_mp_8.0 | -0.1172 | 0.014 | -8.636 | 0.000 | -0.144 | -0.091 |
| main_camera_mp_5.0 | -0.1937 | 0.018 | -11.056 | 0.000 | -0.228 | -0.159 |

- Training performance

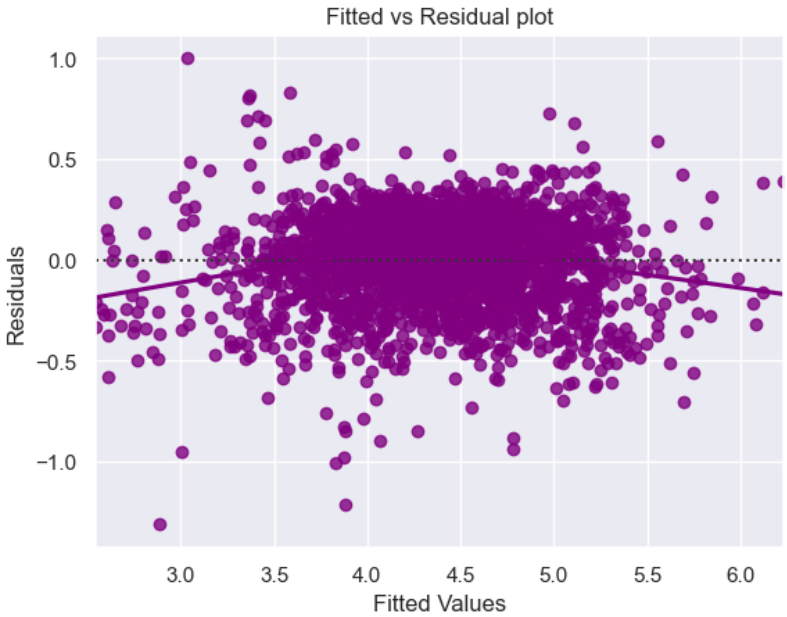
| | RMSE | MAE | R-squared | Adj. R-squared | MAPE |
|---|---|---|---|---|---|
| 0 | 0.22812 | 0.179195 | 0.847258 | 0.845532 | 4.289722 |

- Testing performance

| | RMSE | MAE | R-squared | Adj. R-squared | MAPE |
|---|---|---|---|---|---|
| 0 | 0.238456 | 0.184816 | 0.842349 | 0.83813 | 4.488782 |

# Checking Linear Regression Assumptions – 6. Test for linearity and independence

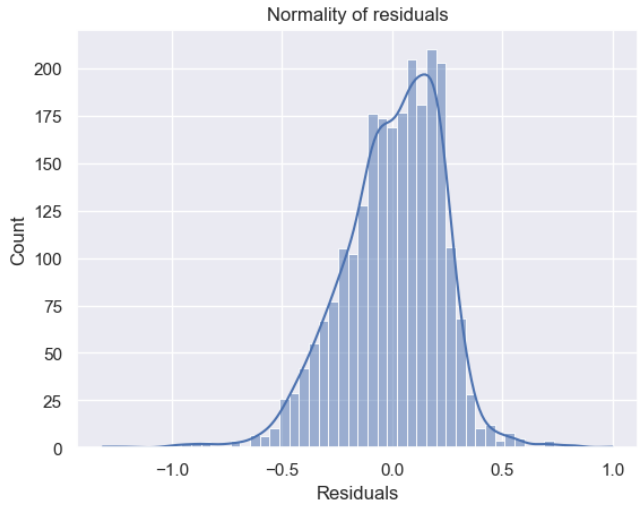
- Plot fitted values vs residuals below indicates no specific pattern of residuals were observed.
- Therefore, the assumptions of linearity and independence are met.

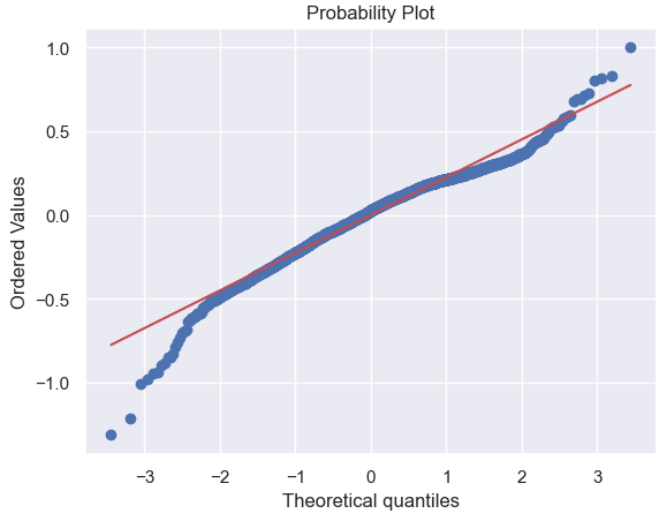



| | Actual Values | Fitted Values | Residuals |
|---|---|---|---|
| **3026** | 4.087488 | 3.873294 | 0.214194 |
| **1525** | 4.448399 | 4.597455 | -0.149055 |
| **1128** | 4.315353 | 4.303790 | 0.011563 |
| **3003** | 4.282068 | 4.209339 | 0.072729 |
| **2907** | 4.456438 | 4.484329 | -0.027891 |

# Checking Linear Regression Assumptions – 7. Test for normality

- Plot q-q graph and test its normality using Shapiro-Wilk's test
  - $H_0$: Residuals are follow normal distribution
    $H_a$: Residuals are not follow normal distribution
- Shapiro result (statistic = 0.9729295969009399, P-value = 5.526429619280507e-21)
- Based on the result, the residuals have formed some of bell shape curve pattern. P-value<0.05, the residuals are not normal and looking at plot, it is not normal at the tail only. We can accept it as being close to normal distribution.



# Checking Linear Regression Assumptions – 8. Test for homoscedasticity

- Test using goldfeldquandt test
  - #$H_0$: Residuals are homoscedastic

    #$H_a$: Residuals are heteroscedastic
- F statistic, 1.0621029651017098, P-value = 0.1502301471062937
- Based on the result, P-value>0.05, and therefore, the residuals are homoscedastic.

| | Actual Values | Residuals |
|---|---|---|
| **1995** | 4.566741 | 4.437539 |
| **2341** | 3.696103 | 3.982614 |
| **1913** | 3.592093 | 3.702040 |
| **688** | 4.306495 | 4.134388 |
| **650** | 4.522115 | 5.145306 |
| **2291** | 4.259294 | 4.411313 |
| **40** | 4.997685 | 5.458849 |
| **1884** | 3.875359 | 4.066433 |
| **2538** | 4.206631 | 4.006378 |
| **45** | 5.380450 | 5.270153 |

# Final Model – 9. Modeling and its performance

- Overview of final ML model and its parameters – training and performance data is comparable, data is not suffering from overfitting

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | normalized_used_price | R-squared: | 0.847 |
| Model: | OLS | Adj. R-squared: | 0.846 |
| Method: | Least Squares | F-statistic: | 509.9 |
| Date: | Wed, 31 May 2023 | Prob (F-statistic): | 0.00 |
| Time: | 13:38:43 | Log-Likelihood: | 142.48 |
| No. Observations: | 2417 | AIC: | -231.0 |
| Df Residuals: | 2390 | BIC: | -74.61 |
| Df Model: | 26 | | |
| Covariance Type: | nonrobust | | |

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| const | 1.7385 | 0.058 | 29.979 | 0.000 | 1.625 | 1.852 |
| screen_size | 0.0240 | 0.003 | 7.635 | 0.000 | 0.018 | 0.030 |
| selfie_camera_mp | 0.0140 | 0.001 | 13.088 | 0.000 | 0.012 | 0.016 |
| int_memory | 0.0001 | 6.77e-05 | 2.023 | 0.043 | 4.17e-06 | 0.000 |
| ram | 0.0182 | 0.004 | 4.104 | 0.000 | 0.009 | 0.027 |
| battery | -1.493e-05 | 7.05e-06 | -2.118 | 0.034 | -2.87e-05 | -1.11e-06 |
| weight | 0.0011 | 0.000 | 8.331 | 0.000 | 0.001 | 0.001 |
| normalized_new_price | 0.4031 | 0.011 | 35.417 | 0.000 | 0.381 | 0.425 |
| years_since_release | -0.0157 | 0.004 | -4.258 | 0.000 | -0.023 | -0.008 |
| brand_name_Nokia | 0.0909 | 0.031 | 2.938 | 0.003 | 0.030 | 0.152 |
| brand_name_Samsung | -0.0356 | 0.016 | -2.199 | 0.028 | -0.067 | -0.004 |
| brand_name_Xiaomi | 0.0690 | 0.025 | 2.718 | 0.007 | 0.019 | 0.119 |
| 4g_yes | 0.0495 | 0.015 | 3.291 | 0.001 | 0.020 | 0.079 |
| main_camera_mp_8.0 | -0.1172 | 0.014 | -8.636 | 0.000 | -0.144 | -0.091 |
| main_camera_mp_5.0 | -0.1937 | 0.018 | -11.056 | 0.000 | -0.228 | -0.159 |
| main_camera_mp_3.15 | -0.2646 | 0.030 | -8.716 | 0.000 | -0.324 | -0.205 |
| main_camera_mp_2.0 | -0.2889 | 0.027 | -10.711 | 0.000 | -0.342 | -0.236 |

- Training performance

| | RMSE | MAE | R-squared | Adj. R-squared | MAPE |
|---|---|---|---|---|---|
| 0 | 0.22812 | 0.179195 | 0.847258 | 0.845532 | 4.289722 |

- Testing performance

| | RMSE | MAE | R-squared | Adj. R-squared | MAPE |
|---|---|---|---|---|---|
| 0 | 0.238456 | 0.184816 | 0.842349 | 0.83813 | 4.488782 |

# Model Performance Summary

- Comparing both training and testing parameters, all of them are similar.
- Final equation listed as below:

```
x_train_final.columns
```

```
Index(['const', 'screen_size', 'selfie_camera_mp', 'int_memory', 'ram',
       'battery', 'weight', 'normalized_new_price', 'years_since_release',
       'brand_name_Nokia', 'brand_name_Samsung', 'brand_name_Xiaomi', '4g_yes',
       'main_camera_mp_8.0', 'main_camera_mp_5.0', 'main_camera_mp_3.15',
       'main_camera_mp_2.0', 'main_camera_mp_16.0', 'main_camera_mp_0.3',
       'main_camera_mp_48.0', 'main_camera_mp_1.3', 'main_camera_mp_0.08',
       'main_camera_mp_23.0', 'main_camera_mp_4.0', 'main_camera_mp_10.0',
       'main_camera_mp_6.7', 'main_camera_mp_8.1'],
      dtype='object')
```

```
x_test_final.columns
```

```
Index(['const', 'screen_size', 'selfie_camera_mp', 'int_memory', 'ram',
       'battery', 'weight', 'normalized_new_price', 'years_since_release',
       'brand_name_Nokia', 'brand_name_Samsung', 'brand_name_Xiaomi', '4g_yes',
       'main_camera_mp_8.0', 'main_camera_mp_5.0', 'main_camera_mp_3.15',
       'main_camera_mp_2.0', 'main_camera_mp_16.0', 'main_camera_mp_0.3',
       'main_camera_mp_48.0', 'main_camera_mp_1.3', 'main_camera_mp_0.08',
       'main_camera_mp_23.0', 'main_camera_mp_4.0', 'main_camera_mp_10.0',
       'main_camera_mp_6.7', 'main_camera_mp_8.1'],
      dtype='object')
```

normalized_used_price =
1.738545028306485 + 0.024049975384125753 * ( screen_size ) + 0.013989086272071936 * ( selfie_camera_mp ) + 0.00013694306297642295 * ( int_memory ) + 0.01816562860308736 * ( ram ) + -1.4925025305305476e-05 * ( battery ) + 0.0010752501859532682 * ( weight ) + 0.40307398052427745 * ( normalized_new_price ) + -0.015711837734562098 * ( years_since_release ) + 0.09093570766879824 * ( brand_name_Nokia ) + -0.035603489917088366 * ( brand_name_Samsung ) + 0.06895891693898151 * ( brand_name_Xiaomi ) + 0.049523278235525756 * ( 4g_yes ) + -0.11718247339081946 * ( main_camera_mp_8.0 ) + -0.1936561415392915 * ( main_camera_mp_5.0 ) + -0.2646330074418731 * ( main_camera_mp_3.15 ) + -0.2888872520038558 * ( main_camera_mp_2.0 ) + 0.1044097032237572 * ( main_camera_mp_16.0 ) + -0.4966417366223834 * ( main_camera_mp_0.3 ) + 0.2914834717876478 * ( main_camera_mp_48.0 ) + -0.5172022623394124 * ( main_camera_mp_1.3 ) + -0.48259611555877635 * ( main_camera_mp_0.08 ) + 0.2391718880381255 * ( main_camera_mp_23.0 ) + -0.30885878430487645 * ( main_camera_mp_4.0 ) + -0.238554776149014 * ( main_camera_mp_10.0 ) + -0.3081639343426148 * ( main_camera_mp_6.7 ) + -0.24584064612219225 * ( main_camera_mp_8.1 )

# Model Performance Summary

- Both of training and performance data is comparable, thus we can conclude our final model is able to predict 84.7% of the variation in the train data.
- RMSE value for both training and performance data is comparable, so our model is not suffering from overfitting.
- MAE value indicates that our current model is able to predict normalized_used_price within a mean error 0.18 euros on the test data.
- Therefore, we can conclude our final model is a good prediction as well as inferences purposes.

# APPENDIX

# Data Background and Contents

- data.head()

| | brand_name | os | screen_size | 4g | 5g | main_camera_mp | selfie_camera_mp | int_memory | ram | battery | weight | release_year | days_used | normalized_used_price | normalized_new_price |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Honor | Android | 14.50 | yes | no | 13.0 | 5.0 | 64.0 | 3.0 | 3020.0 | 146.0 | 2020 | 127 | 4.307572 | 4.715100 |
| 1 | Honor | Android | 17.30 | yes | yes | 13.0 | 16.0 | 128.0 | 8.0 | 4300.0 | 213.0 | 2020 | 325 | 5.162097 | 5.519018 |
| 2 | Honor | Android | 16.69 | yes | yes | 13.0 | 8.0 | 128.0 | 8.0 | 4200.0 | 213.0 | 2020 | 162 | 5.111084 | 5.884631 |
| 3 | Honor | Android | 25.50 | yes | yes | 13.0 | 8.0 | 64.0 | 6.0 | 7250.0 | 480.0 | 2020 | 345 | 5.135387 | 5.630961 |
| 4 | Honor | Android | 15.32 | yes | no | 13.0 | 8.0 | 64.0 | 3.0 | 5000.0 | 185.0 | 2020 | 293 | 4.389995 | 4.947837 |
| 5 | Honor | Android | 16.23 | yes | no | 13.0 | 8.0 | 64.0 | 4.0 | 4000.0 | 176.0 | 2020 | 223 | 4.413889 | 5.060694 |
| 6 | Honor | Android | 13.84 | yes | no | 8.0 | 5.0 | 32.0 | 2.0 | 3020.0 | 144.0 | 2020 | 234 | 3.878259 | 4.518958 |
| 7 | Honor | Android | 15.77 | yes | no | 13.0 | 8.0 | 64.0 | 4.0 | 3400.0 | 164.0 | 2020 | 219 | 4.729421 | 5.188726 |
| 8 | Honor | Android | 15.32 | yes | no | 13.0 | 16.0 | 128.0 | 6.0 | 4000.0 | 165.0 | 2020 | 161 | 4.886054 | 5.299916 |
| 9 | Honor | Android | 16.23 | yes | no | 13.0 | 8.0 | 128.0 | 6.0 | 4000.0 | 176.0 | 2020 | 327 | 4.773224 | 5.073610 |

# Data Background and Contents

- data.shape – 3454 rows, 15 columns
- data.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3454 entries, 0 to 3453
Data columns (total 15 columns):
 #   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
```

# Data Preparation for Modeling

- We want to predict the normalized_price of the used smartphones
- Below is after we treated all missing value.
- We need to encode the categorical features and split data into train and test data
- x.head()

| | brand_name | os | screen_size | 4g | 5g | main_camera_mp | selfie_camera_mp | int_memory | ram | battery | weight | days_used | normalized_new_price | years_since_release |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Honor | Android | 14.50 | yes | no | 13.0 | 5.0 | 64.0 | 3.0 | 3020.0 | 146.0 | 127 | 4.715100 | 1 |
| **1** | Honor | Android | 17.30 | yes | yes | 13.0 | 16.0 | 128.0 | 8.0 | 4300.0 | 213.0 | 325 | 5.519018 | 1 |
| **2** | Honor | Android | 16.69 | yes | yes | 13.0 | 8.0 | 128.0 | 8.0 | 4200.0 | 213.0 | 162 | 5.884631 | 1 |
| **3** | Honor | Android | 25.50 | yes | yes | 13.0 | 8.0 | 64.0 | 6.0 | 7250.0 | 480.0 | 345 | 5.630961 | 1 |
| **4** | Honor | Android | 15.32 | yes | no | 13.0 | 8.0 | 64.0 | 3.0 | 5000.0 | 185.0 | 293 | 4.947837 | 1 |

# First Model

- First model after perform all initial necessary treatments

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | normalized_used_price | R-squared: | 0.850 |
| Model: | OLS | Adj. R-squared: | 0.845 |
| Method: | Least Squares | F-statistic: | 159.2 |
| Date: | Wed, 07 Jun 2023 | Prob (F-statistic): | 0.00 |
| Time: | 08:34:13 | Log-Likelihood: | 163.90 |
| No. Observations: | 2417 | AIC: | -159.8 |
| Df Residuals: | 2333 | BIC: | 326.6 |
| Df Model: | 83 | | |
| Covariance Type: | nonrobust | | |

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| const | 1.6815 | 0.080 | 20.959 | 0.000 | 1.524 | 1.839 |
| screen_size | 0.0238 | 0.003 | 6.865 | 0.000 | 0.017 | 0.031 |
| selfie_camera_mp | 0.0137 | 0.001 | 12.146 | 0.000 | 0.012 | 0.016 |
| int_memory | 0.0002 | 7.09e-05 | 2.259 | 0.024 | 2.11e-05 | 0.000 |
| ram | 0.0223 | 0.005 | 4.274 | 0.000 | 0.012 | 0.032 |
| battery | -1.753e-05 | 7.39e-06 | -2.373 | 0.018 | -3.2e-05 | -3.04e-06 |
| weight | 0.0011 | 0.000 | 7.982 | 0.000 | 0.001 | 0.001 |
| days_used | 2.496e-05 | 3.11e-05 | 0.803 | 0.422 | -3.6e-05 | 8.6e-05 |
| normalized_new_price | 0.4146 | 0.013 | 32.372 | 0.000 | 0.390 | 0.440 |
| years_since_release | -0.0213 | 0.005 | -4.560 | 0.000 | -0.031 | -0.012 |
| brand_name_Alcatel | 0.0093 | 0.047 | 0.196 | 0.844 | -0.084 | 0.102 |
| brand_name_Apple | -0.1129 | 0.147 | -0.766 | 0.444 | -0.402 | 0.176 |
| brand_name_Asus | 0.0033 | 0.048 | 0.070 | 0.944 | -0.090 | 0.097 |
| brand_name_BlackBerry | -0.0768 | 0.072 | -1.067 | 0.286 | -0.218 | 0.064 |
| brand_name_Celkon | -0.0499 | 0.067 | -0.747 | 0.455 | -0.181 | 0.081 |
| brand_name_Coolpad | 0.0089 | 0.073 | 0.123 | 0.902 | -0.133 | 0.151 |
| brand_name_Gionee | 0.0425 | 0.058 | 0.738 | 0.461 | -0.070 | 0.155 |
| brand_name_Google | 0.0562 | 0.157 | 0.359 | 0.720 | -0.251 | 0.364 |
| brand_name_HTC | -0.0112 | 0.049 | -0.230 | 0.818 | -0.107 | 0.084 |
| brand_name_Huawei | -0.0225 | 0.045 | -0.502 | 0.616 | -0.110 | 0.065 |
| brand_name_Infinix | 0.0192 | 0.046 | 0.413 | 0.680 | -0.072 | 0.110 |
| brand_name_Karbonn | 0.0772 | 0.067 | 1.155 | 0.248 | -0.054 | 0.208 |
| brand_name_LG | -0.0206 | 0.045 | -0.455 | 0.649 | -0.110 | 0.068 |
| brand_name_Lava | 0.0197 | 0.062 | 0.317 | 0.752 | -0.102 | 0.142 |
| brand_name_Lenovo | 0.0340 | 0.045 | 0.751 | 0.453 | -0.055 | 0.123 |
| brand_name_Meizu | -0.0198 | 0.056 | -0.353 | 0.724 | -0.130 | 0.090 |
| brand_name_Micromax | -0.0254 | 0.048 | -0.533 | 0.594 | -0.119 | 0.068 |
| brand_name_Microsoft | 0.0642 | 0.089 | 0.720 | 0.472 | -0.111 | 0.239 |
| brand_name_Motorola | -0.0037 | 0.050 | -0.074 | 0.941 | -0.102 | 0.094 |
| brand_name_Nokia | 0.0781 | 0.052 | 1.498 | 0.134 | -0.024 | 0.180 |
| brand_name_OnePlus | 0.0680 | 0.077 | 0.881 | 0.378 | -0.083 | 0.219 |
| brand_name_Oppo | 0.0051 | 0.048 | 0.107 | 0.915 | -0.089 | 0.099 |
| brand_name_Others | -0.0125 | 0.042 | -0.297 | 0.767 | -0.095 | 0.070 |
| brand_name_Panasonic | 0.0428 | 0.056 | 0.768 | 0.443 | -0.067 | 0.152 |

# First Model

```
brand_name_Realme
0.0156      0.062      0.253      0.800      -0.105      0.136
brand_name_Samsung
-0.0477      0.043     -1.098      0.272      -0.133      0.037
brand_name_Sony
-0.0476      0.055     -0.862      0.389      -0.156      0.061
brand_name_Spice
-0.0184      0.063     -0.291      0.771      -0.142      0.106
brand_name_Vivo
-0.0279      0.049     -0.575      0.565      -0.123      0.067
brand_name_XOLO
-7.906e-05      0.055     -0.001      0.999      -0.107      0.107
brand_name_Xiaomi
0.0650      0.048      1.342      0.180      -0.030      0.160
brand_name_ZTE
-0.0027      0.048     -0.057      0.955      -0.097      0.091
os_Others
0.0241      0.035      0.683      0.495      -0.045      0.093
os_Windows
-2.594e-05      0.048     -0.001      1.000      -0.093      0.093
os_iOS
0.0272      0.146      0.186      0.852      -0.259      0.313
4g_yes
0.0479      0.016      2.975      0.003       0.016      0.080
5g_yes
-0.0665      0.032     -2.065      0.039      -0.130     -0.003
main_camera_mp_8.0
-0.1088      0.015     -7.331      0.000      -0.138     -0.080
main_camera_mp_5.0
-0.1803      0.019     -9.565      0.000      -0.217     -0.143
```

```
main_camera_mp_10.5
0.0275      0.055      0.501      0.616      -0.080      0.135
main_camera_mp_3.15
-0.2440      0.032     -7.688      0.000      -0.306     -0.182
main_camera_mp_<bound method NDFrame._add_numeric_operations.<locals>.median of 0
1       13.0
2       13.0
3       13.0
4       13.0
        ...
3449    13.0
3450    13.0
3451    13.0
3452    13.0
3453    13.0
Name: main_camera_mp, Length: 3454, dtype: float64>      0.0192      0.046      0.413
main_camera_mp_2.0
-0.2751      0.029     -9.569      0.000      -0.331     -0.219
main_camera_mp_16.0
0.1076      0.025      4.345      0.000       0.059      0.156
main_camera_mp_0.3
-0.4714      0.044    -10.704      0.000      -0.558     -0.385
main_camera_mp_12.0
0.0076      0.024      0.319      0.750      -0.039      0.055
main_camera_mp_14.5
-0.0145      0.078     -0.187      0.852      -0.167      0.138
main_camera_mp_48.0
0.2757      0.118      2.344      0.019       0.045      0.506
main_camera_mp_3.0
-0.1121      0.136     -0.824      0.410      -0.379      0.155
```

```
main_camera_mp_13.1
0.1410      0.165      0.854      0.393      -0.183      0.465
main_camera_mp_24.0
0.0398      0.134      0.296      0.767      -0.224      0.303
main_camera_mp_0.08
-0.4712      0.234     -2.010      0.045      -0.931     -0.011
main_camera_mp_20.7
0.1023      0.083      1.237      0.216      -0.060      0.264
main_camera_mp_23.0
0.2705      0.073      3.720      0.000       0.128      0.413
main_camera_mp_1.0
-1.266e-16    1.01e-16     -1.257      0.209    -3.24e-16    7.09e-17
main_camera_mp_18.0
0.0682      0.238      0.287      0.774      -0.398      0.535
main_camera_mp_12.2
-0.1628      0.175     -0.928      0.354      -0.507      0.181
main_camera_mp_12.3
0.0540      0.108      0.500      0.617      -0.158      0.266
main_camera_mp_20.0
0.1257      0.096      1.303      0.193      -0.064      0.315
main_camera_mp_20.2
0.0289      0.232      0.125      0.901      -0.426      0.484
main_camera_mp_4.0
-0.2934      0.098     -2.981      0.003      -0.486     -0.100
main_camera_mp_12.5
0.1250      0.117      1.067      0.286      -0.105      0.355
main_camera_mp_10.0
-0.2221      0.119     -1.868      0.062      -0.455      0.011
```

# First Model

```
main_camera_mp_6.5
-0.0930      0.164     -0.566      0.572      -0.415       0.229
main_camera_mp_6.7
-0.2939      0.128     -2.303      0.021      -0.544      -0.044
main_camera_mp_41.0
2.739e-17   2.24e-17      1.225      0.221   -1.65e-17    7.12e-17
main_camera_mp_20.1
-3.054e-17   9.68e-17     -0.315      0.752    -2.2e-16    1.59e-16
main_camera_mp_12.6
7.875e-17   4.99e-17      1.577      0.115   -1.91e-17    1.77e-16
main_camera_mp_16.3
0.3742      0.233      1.608      0.108      -0.082       0.830
main_camera_mp_22.6
-0.1479      0.232     -0.639      0.523      -0.602       0.306
main_camera_mp_19.0
-0.0679      0.102     -0.666      0.506      -0.268       0.132
main_camera_mp_21.5
0.1234      0.234      0.527      0.598      -0.335       0.582
main_camera_mp_21.2
0.2065      0.168      1.233      0.218      -0.122       0.535
main_camera_mp_8.1
-0.1930      0.123     -1.572      0.116      -0.434       0.048
main_camera_mp_1.2
-0.1055      0.235     -0.449      0.654      -0.567       0.356
main_camera_mp_22.5
0.1759      0.232      0.759      0.448      -0.279       0.630
==============================================================================
Omnibus:                      205.856   Durbin-Watson:                   1.906
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              389.471
Skew:                          -0.578   Prob(JB):                     2.68e-85
Kurtosis:                       4.591   Cond. No.                     4.21e+19
==============================================================================
```

- Training performance

| | RMSE | MAE | R-squared | Adj. R-squared | MAPE |
|---|---|---|---|---|---|
| 0 | 0.226107 | 0.17736 | 0.849942 | 0.844202 | 4.247918 |

- Testing performance

| | RMSE | MAE | R-squared | Adj. R-squared | MAPE |
|---|---|---|---|---|---|
| 0 | 0.226107 | 0.17736 | 0.849942 | 0.844202 | 4.247918 |

# Final Model

- Final model establish after we complete checking and qualify for all assumptions:
  - No multicollinearity – remove multicollinearity (VIF>10)
  - Dropping high P-value>0.05
  - Check for linearity and test for independence
  - Test for normality
  - Test for homosdecascity

# Final Model

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | normalized_used_price | R-squared: | 0.847 |
| Model: | OLS | Adj. R-squared: | 0.846 |
| Method: | Least Squares | F-statistic: | 509.9 |
| Date: | Wed, 07 Jun 2023 | Prob (F-statistic): | 0.00 |
| Time: | 09:26:13 | Log-Likelihood: | 142.48 |
| No. Observations: | 2417 | AIC: | -231.0 |
| Df Residuals: | 2390 | BIC: | -74.61 |
| Df Model: | 26 | | |
| Covariance Type: | nonrobust | | |

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| const | 1.7385 | 0.058 | 29.979 | 0.000 | 1.625 | 1.852 |
| screen_size | 0.0240 | 0.003 | 7.635 | 0.000 | 0.018 | 0.030 |
| selfie_camera_mp | 0.0140 | 0.001 | 13.088 | 0.000 | 0.012 | 0.016 |
| int_memory | 0.0001 | 6.77e-05 | 2.023 | 0.043 | 4.17e-06 | 0.000 |
| ram | 0.0182 | 0.004 | 4.104 | 0.000 | 0.009 | 0.027 |
| battery | -1.493e-05 | 7.05e-06 | -2.118 | 0.034 | -2.87e-05 | -1.11e-06 |
| weight | 0.0011 | 0.000 | 8.331 | 0.000 | 0.001 | 0.001 |
| normalized_new_price | 0.4031 | 0.011 | 35.417 | 0.000 | 0.381 | 0.425 |
| years_since_release | -0.0157 | 0.004 | -4.258 | 0.000 | -0.023 | -0.008 |
| brand_name_Nokia | 0.0909 | 0.031 | 2.938 | 0.003 | 0.030 | 0.152 |
| brand_name_Samsung | -0.0356 | 0.016 | -2.199 | 0.028 | -0.067 | -0.004 |
| brand_name_Xiaomi | 0.0690 | 0.025 | 2.718 | 0.007 | 0.019 | 0.119 |
| 4g_yes | 0.0495 | 0.015 | 3.291 | 0.001 | 0.020 | 0.079 |
| main_camera_mp_8.0 | -0.1172 | 0.014 | -8.636 | 0.000 | -0.144 | -0.091 |
| main_camera_mp_5.0 | -0.1937 | 0.018 | -11.056 | 0.000 | -0.228 | -0.159 |
| main_camera_mp_3.15 | -0.2646 | 0.030 | -8.716 | 0.000 | -0.324 | -0.205 |
| main_camera_mp_2.0 | -0.2889 | 0.027 | -10.711 | 0.000 | -0.342 | -0.236 |
| main_camera_mp_16.0 | 0.1044 | 0.024 | 4.360 | 0.000 | 0.057 | 0.151 |
| main_camera_mp_0.3 | -0.4966 | 0.040 | -12.287 | 0.000 | -0.576 | -0.417 |
| main_camera_mp_48.0 | 0.2915 | 0.115 | 2.529 | 0.012 | 0.065 | 0.518 |
| main_camera_mp_1.3 | -0.5172 | 0.062 | -8.331 | 0.000 | -0.639 | -0.395 |
| main_camera_mp_0.08 | -0.4826 | 0.231 | -2.086 | 0.037 | -0.936 | -0.029 |
| main_camera_mp_23.0 | 0.2392 | 0.067 | 3.581 | 0.000 | 0.108 | 0.370 |
| main_camera_mp_4.0 | -0.3089 | 0.094 | -3.271 | 0.001 | -0.494 | -0.124 |
| main_camera_mp_10.0 | -0.2386 | 0.116 | -2.060 | 0.040 | -0.466 | -0.011 |
| main_camera_mp_6.7 | -0.3082 | 0.120 | -2.564 | 0.010 | -0.544 | -0.072 |
| main_camera_mp_8.1 | -0.2458 | 0.116 | -2.113 | 0.035 | -0.474 | -0.018 |

| | | | |
|---|---|---|---|
| Omnibus: | 208.965 | Durbin-Watson: | 1.911 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 394.419 |
| Skew: | -0.586 | Prob(JB): | 2.25e-86 |
| Kurtosis: | 4.594 | Cond. No. | 1.69e+05 |

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.69e+05. This might indicate that there are strong multicollinearity or other numerical problems.

- Training performance

| | RMSE | MAE | R-squared | Adj. R-squared | MAPE |
|---|---|---|---|---|---|
| 0 | 0.22812 | 0.179195 | 0.847258 | 0.845532 | 4.289722 |

- Testing performance

| | RMSE | MAE | R-squared | Adj. R-squared | MAPE |
|---|---|---|---|---|---|
| 0 | 0.238456 | 0.184816 | 0.842349 | 0.83813 | 4.488782 |

# Final Equation

normalized_used_price =

1.738545028306485 + 0.024049975384125753 * ( screen_size ) + 0.013989086272071936 * ( selfie_camera_mp ) + 0.00013694306297642295 * ( int_memory ) + 0.01816562860308736 * ( ram ) + -1.4925025305305476e-05 * ( battery ) + 0.0010752501859532682 * ( weight ) + 0.40307398052427745 * ( normalized_new_price ) + -0.015711837734562098 * ( years_since_release ) + 0.09093570766879824 * ( brand_name_Nokia ) + -0.035603489917088366 * ( brand_name_Samsung ) + 0.06895891693898151 * ( brand_name_Xiaomi ) + 0.049523278235525756 * ( 4g_yes ) + -0.11718247339081946 * ( main_camera_mp_8.0 ) + -0.1936561415392915 * ( main_camera_mp_5.0 ) + -0.2646330074418731 * ( main_camera_mp_3.15 ) + -0.2888872520038558 * ( main_camera_mp_2.0 ) + 0.1044097032237572 * ( main_camera_mp_16.0 ) + -0.4966417366223834 * ( main_camera_mp_0.3 ) + 0.2914834717876478 * ( main_camera_mp_48.0 ) + -0.5172022623394124 * ( main_camera_mp_1.3 ) + -0.48259611555877635 * ( main_camera_mp_0.08 ) + 0.2391718880381255 * ( main_camera_mp_23.0 ) + -0.30885878430487645 * ( main_camera_mp_4.0 ) + -0.238554776149014 * ( main_camera_mp_10.0 ) + -0.3081639343426148 * ( main_camera_mp_6.7 ) + -0.24584064612219225 * ( main_camera_mp_8.1 )

Great Learning

# Happy Learning !