ReCell Project

Post-Grad Program in Data Science and Business Analytics

Date: 9/25/2022
By: Manik Chhabra

Contents / Agenda

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

Executive Summary

- The model can explain 84% of the variation in data and within 4.5% of the used devices price on the test data, which is exceptionally good
    - Hence, we can conclude that the model is good for prediction and interpretation
- If the **main_camera_mp** raises by one unit, then the normalized used device price increases by 0.0210, all other variables held constant
- If the **selfie_camera_mp** raises by one unit, then the normalized used device price increases by 0.0138, all other variables held constant
    - The company ReCell can buy higher megapixel cameras (selfie and main) to resell the used product at a higher price
- If the **ram** raises by one unit, then the normalized used device price increases by 0.0207, all other variables held constant
    - The company ReCell can buy more used products with higher ram to resell it at a higher price
- If the **weight** raises by one unit, then the normalized used device price increases by 0.0017, all other variables held constant
    - Buying phones with higher weight can increase the resell value of the phone
- If the **normalized_new_price** raises by one unit, then the normalized used device price increases by 0.4415, all other variables held constant
- According to the results from bivariate analysis, devices that use **4G** or **5G** network have higher **normalized used prices**; therefore the company ReCell can buy devices that are only **4G or 5G** to earn higher profits

Business Problem Overview and Solution Approach

- Business Problem
  - ReCell wants to hire a data scientist that can build a linear regression model to predict the price of a used phone/tablet and identify factors that significantly influence it
- The solution approach / methodology
  - The solution to the business problem is to first discover a relationship between the normalized used device price and the other variables in the dataset through exploratory data analysis
  - Second, preprocess the data before building the linear regression model
  - Test all 5 assumptions for linear regression modelling
  - Lastly build a model that has an exceptionally good fit and is not overfitting nor underfitting based on the comparison between the training and testing data

EDA Results

- Univariate Analysis
  - Variable - **Normalized Used Price**
    - According to **Figure 1,** the distribution for **normalized used price** is skewed to the left
    - The mean and median for the **normalized used price** is respectively 4.36 Euros and 4.41 Euros
    - Outliers exist on both sides of the Boxplot
  - Variable – **Normalized New Price**
    - According to **Figure 2,** the distribution for **normalized new price** appears evenly distributed
    - The mean and median for the **normalized new price** is respectively 5.23 Euros and 5.25 Euros
    - Outliers exist on both sides of the Boxplot

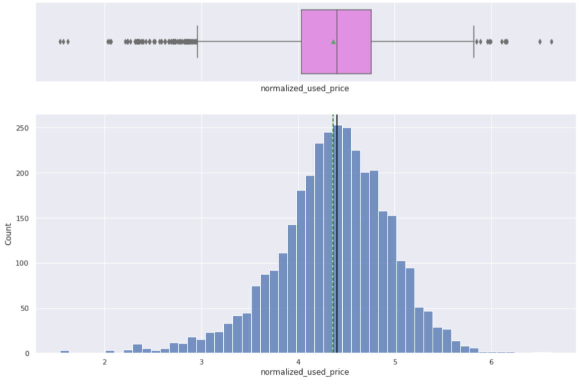

Figure 1

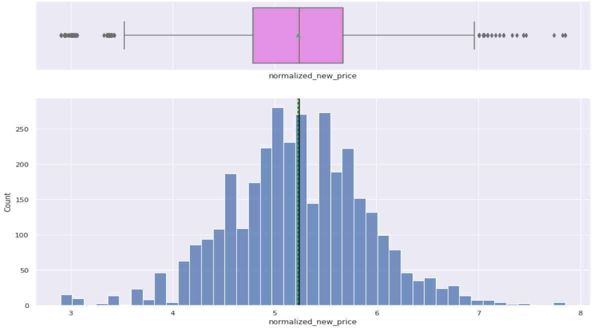

Figure 2

Link to Appendix slide on data background check

EDA Results

- Univarate Analysis
  - Variable – **Screen Size**
    - According to **Figure 3,** the distribution for **screen size** is skewed to the right
    - The mean and median for the **screen size** is respectively 13.71cm and 12.83 cm
    - Outliers exist on both sides of the Boxplot
  - Variable – **Main Camera MP**
    - According to **Figure 4,** the distribution for **main camera mp** is heavily skewed to the right
    - The mean and median for **main camera mp** is respectively 9.46 mp and 8.00 mp
    - Outliers exist beyond the max value of the boxplot

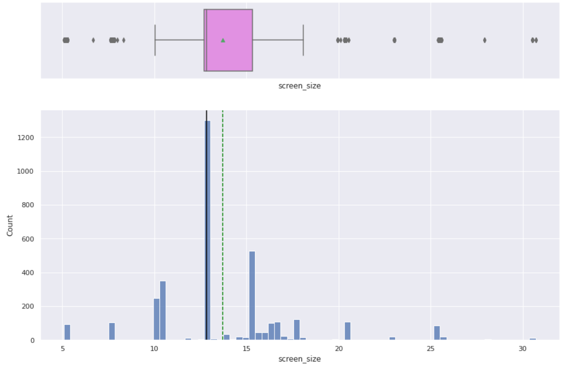

Figure 3

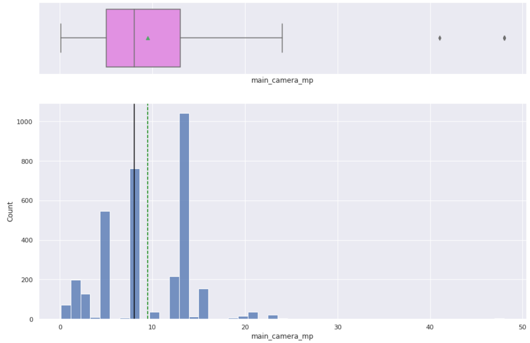

Figure 4

Link to Appendix slide on data background check

EDA Results

- Univariate Analysis
  - Variable – **Selfie Camera MP**
    - According to **Figure 5,** the distribution for **Selfie Camera MP** is heavily skewed to the right
    - The mean and median for the **Selfie Camera MP** is respectively 6.55 mp and 5 mp
    - Outliers exist beyond the max value of the boxplot
  - Variable – **Internal Memory**
    - According to **Figure 6,** the distribution for **Internal Memory** is heavily skewed to the right
    - The mean and median for **Internal memory** is respectively 54.57 GB and 32 GB
    - Outliers exist beyond the max value of the boxplot

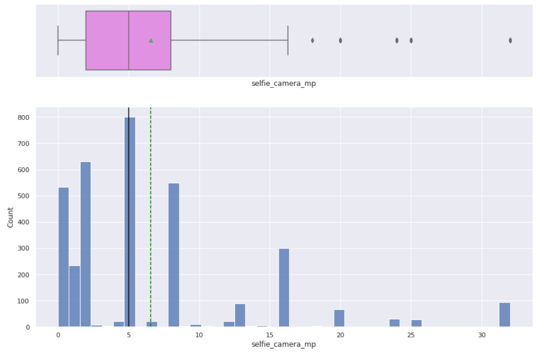

Figure 5

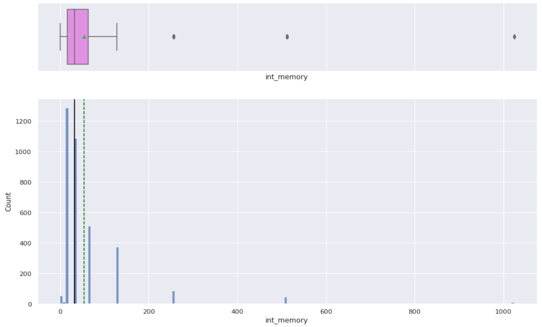

Figure 6

Link to Appendix slide on data background check

EDA Results

- Univariate Analysis
  - Variable – **Ram**
    - According to **Figure 7,** the distribution for **Ram** is unevenly distributed; A boxplot was not formed
    - The mean and median for the **Ram** is respectively 4.04 GB and 4 GB
  - Variable – **Weight**
    - According to **Figure 8,** the distribution for **Weight** is heavily skewed to the right
    - The mean and median for **Weight** is respectively 182.75 g and 160 g
    - Outliers exist beyond the max value of the boxplot

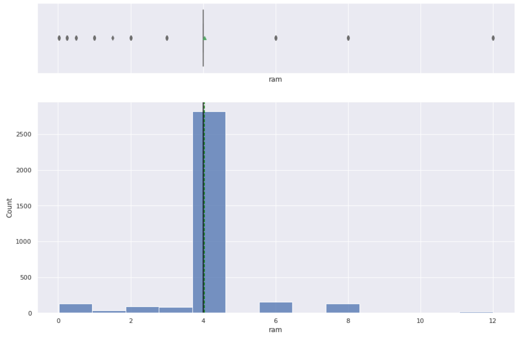

Figure 7

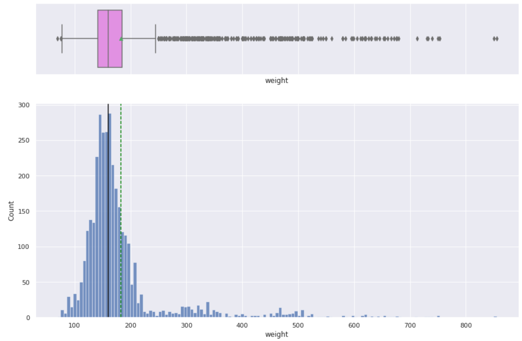

Figure 8

Link to Appendix slide on data background check

EDA Results

- Univariate Analysis
  - Variable – **Battery**
    - According to **Figure 9,** the distribution for **Battery** is heavily skewed to the right
    - The mean and median for the **Battery** is respectively 3133.40 mAh and 3000 mAh
    - Outliers exist beyond the max value of the boxplot
  - Variable – **Days Used**
    - According to **Figure 10,** the distribution for **Days Used** is skewed to the left
    - The mean and median for **Days Used** is respectively 674.87 and 690.50
    - Outliers do not exist for this variable

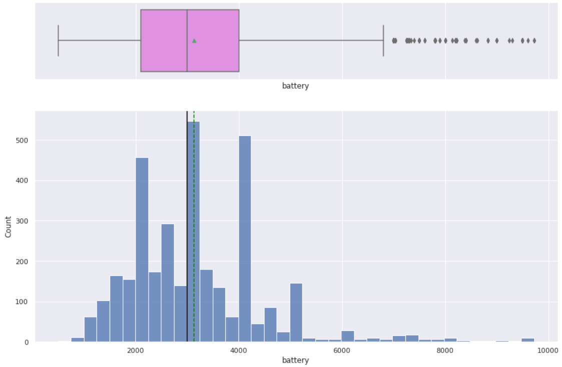

Figure 9

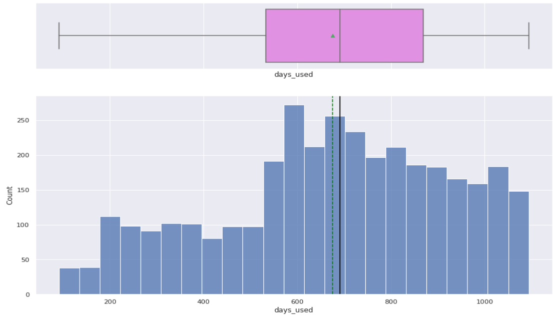

Figure 10

Link to Appendix slide on data background check

EDA Results

- Univariate Analysis
  - Variable – **Brand Name**
    - According to **Figure 11,** the top 3 brands in the data set (in terms of count) are **Others** (14.5%), **Samsung** (9.9%), and **Huawei** (7.3%).
  - Variable – **OS**
    - According to **Figure 12,** most of the phones/tablets in the dataset run on the **Android OS** (3214)
    - **iOS** phones/tablets have the lowest count (36) in the data set
  - Variable – **4G**
    - According to **Figure 13,** there are more than twice as much **4G** phones/tablets (2335) than non-**4G** phones/tablets (1119)
  - Variable – **5G**
    - According to **Figure 14,** 3302 phones/tablets are not run on **5G**
    - 152 phones/tablets are run on **5G**

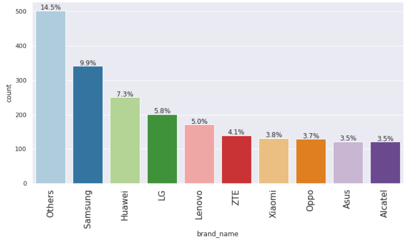

Figure 11

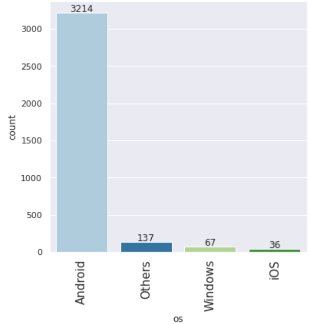

Figure 12

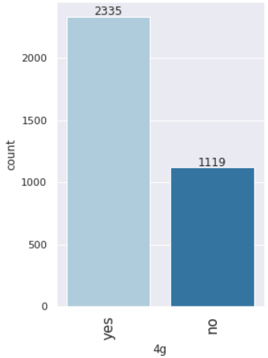

Figure 13

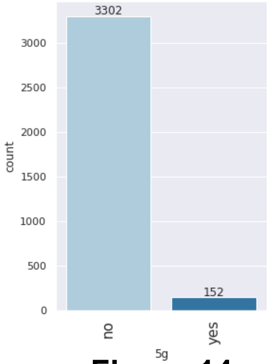

Figure 14

[Link to Appendix slide on data background check]

EDA Results

- Univariate Analysis
  - Variable – **Release Year**
    - According to **Figure 15,** the top 3 amount of phones/tablets in the data set were released in 2014 (642), 2013 (570) , and 2015 (515)
    - The least amount of phones in the dataset were released in 2020 (277).

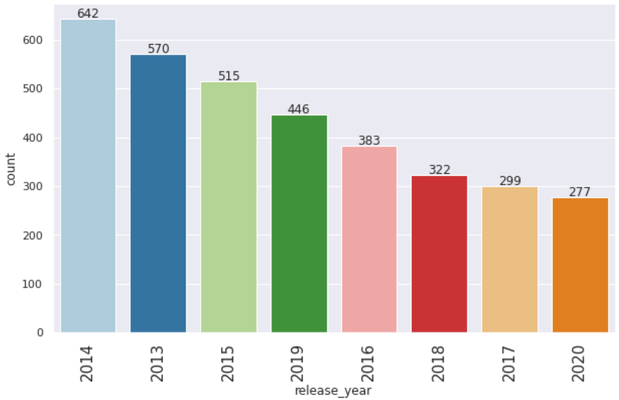

Figure 15

[Link to Appendix slide on data background check](#)

EDA Results

- Bivariate Analysis
  - Heat Map
    - According to **Figure 16,** ‘Selfie camera mp’, ’Screen_size’, and ‘Battery’ are moderately positively correlated with ‘normalized_used_price’ – 0.61 correlation
    - ‘Main_camera_mp’ is moderately positively correlated with ‘normalized_used_price’ – 0.59
    - ‘Normalized_used_price’ and ‘normalized_new_price’ are highly positively correlated with each other – 0.83
  - **Release year** vs **Normalized Used Price** – **Figure 17**
    - A strong positive correlation exists between **release year** and **normalized used price** from 2013-2018
      - Used phone prices increase with the newer years
    - After 2018, there was a weaker positive correlation(2019- 2020 – no correlation existed) between **release year** and **normalized used price**

Figure 16

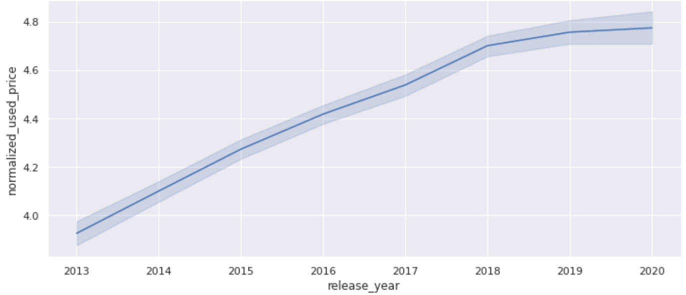

Figure 17

Link to Appendix slide on data background check

EDA Results

- Bivariate Analysis
  - **4G** vs **Normalized Used Price** – **Figure 18**
    - According to the Boxplot, normalized used tablets/phones prices are higher for tablets/phones that use 4G network (~4.5 euros) vs non-4G network (~4 euros)
  - **5G** vs **Normalized Used Price** – **Figure 19**
    - According to the Boxplot, normalized used tablets/phones prices are higher for tablets/phones that use 5G network (~5.2 euros) vs non-5G network (~4.5 euros)

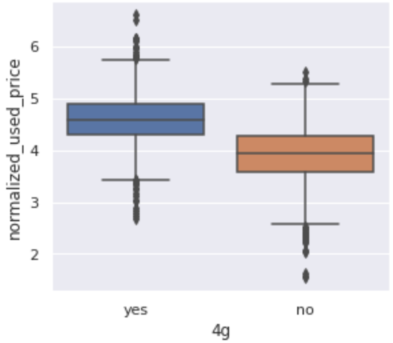

Figure 18

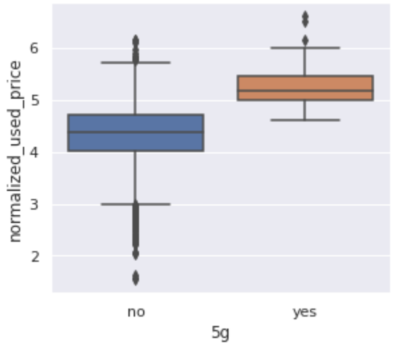

Figure 19

[*Link to Appendix slide on data background check*]

Data Preprocessing

- Duplicate value check
- Missing value treatment
- Outlier check (treatment if needed)
- Feature engineering
- Data preparation for modeling

Data Preprocessing

- Duplicate value check
  - No duplicate values exist in the dataset
- Missing Value Treatment
  - Used the code below to check for missing values in the dataset across all columns; Missing value is showing below per column:

```
# checking for missing values
df1.isnull().sum() ## Complete the code to check missing values in all the columns
```

```
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
  - After 3 rounds of imputing the missing values with the column medians grouped by ‘brand_name’ and ‘release_year’, by ‘brand_name’, and no grouping, no missing values appeared for each column after round 3

Round 1

```
cols_impute = ["main_camera_mp",
    "selfie_camera_mp",
    "int_memory",
    "ram",
    "battery",
    "weight"]

for col in cols_impute:
  df1[col] = df1[col].fillna(value=df1.groupby(["release_year","brand_name"])[col].transform("median"))

# checking for missing values

df1.isnull().sum()
 ## Complete the code to check missing values after imputing the above columns
```

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

Round 2

```
cols_impute = [
    "main_camera_mp",
    "selfie_camera_mp",
    "battery",
    "weight",
]

for col in cols_impute:
    df1[col] = df1[col].fillna(
        value=df1.groupby(['brand_name'])[col].transform("median")
    ) ## Complete the code to impute the missing values in cols_impute with median by grouping the data on brand name

# checking for missing values
df1.isnull().sum() ## Complete the code to check missing values after imputing the above columns
```

```
brand_name                 0
os                         0
screen_size                0
4g                         0
5g                         0
main_camera_mp            10
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
```

Round 3

```
df1["main_camera_mp"] = df1["main_camera_mp"].fillna(df1["main_camera_mp"].median())

# checking for missing values
df1.isnull().sum() ## Complete the code to check missing values after imputing the ab
```

```
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
```

Data Preprocessing

- Outlier Check
  - Outliers exist for most of the variables except ‘release_year’ and ‘days_used’
  - However, they will not be treated as they are proper values

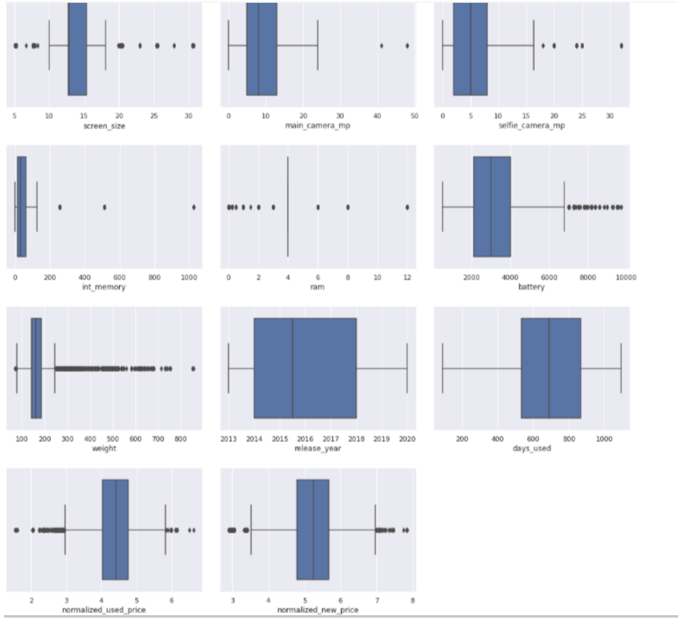

Figure 20

Data Preprocessing

- Feature Engineering
    - ‘years_since_release’ column was created from the ‘release_year’ column
    - ‘Release_year’ was then subsequently removed
    - In **Figure 21,** is a statistical summary of the new column, ‘years_since_release’:
        - Number of years since release varies from 1-8.
        - Mean is 5 years

Figure 21

```
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

- Data Preparation for Modeling
    - Purpose is to predict the normalized price of used devices
    - Defined the dependent and independent variables in **Figure 22**
        - y (dependent variable) = ‘normalized used price’ values
        - X (independent variable) = all the predictor variables that are not ‘normalized used price’
    - Added the intercept to the data in **Figure 23**
    - Encoded categorical features of the dataset in **Figure 24**
        - Dummy values were created for the categorical variables, ‘brand_name’, ‘os,’ ‘4g’, and ‘5g’ before building the model

Figure 22

```
## Complete the code to define the dependent and independent variables
X = df1.drop('normalized_used_price',axis=1)
y = df1['normalized_used_price']

print(X.head())
print()
print(y.head())
```

Figure 23

```
# let's add the intercept to data
X = sm.add_constant(X)
```

Figure 24

```
# creating dummy variables
X = pd.get_dummies(
    X,
    columns=X.select_dtypes(include=["object", "category"]).columns.tolist(),
    drop_first=True,)  ## Complete the code to create dummies for independent features
X.head()
```

	const	screen_size	main_camera_mp	selfie_camera_mp	int_memory	ram	battery	weight	days_used	normalized_new_price	...	brand_name_Spice	brand_name_Vivo	brand_name_XOLO	brand_name_Xiaomi	brand_name_ZTE	os_Others	os_Windows	os_iOS	4g_yes	5g_yes
0	1.0	14.50	13.0	5.0	64.0	3.0	3020.0	146.0	127	4.715100	...	0	0	0	0	0	0	0	0	1	0
1	1.0	17.30	13.0	16.0	128.0	8.0	4300.0	213.0	325	5.519018	...	0	0	0	0	0	0	0	0	1	1
2	1.0	16.69	13.0	8.0	128.0	8.0	4200.0	213.0	162	5.884631	...	0	0	0	0	0	0	0	0	1	1
3	1.0	25.50	13.0	8.0	64.0	6.0	7250.0	480.0	345	5.630961	...	0	0	0	0	0	0	0	0	1	1
4	1.0	15.32	13.0	8.0	64.0	3.0	5000.0	185.0	293	4.947837	...	0	0	0	0	0	0	0	0	1	0

5 rows × 49 columns

Data Preprocessing

- Data Preparation for Modeling
    - Splitting the data between test and train data to evaluate the model built on the train data in **Figure 25**
        - Data was split in 70:30 ratio for train to test data

Figure 25

```
# splitting the data in 70:30 ratio for train to test data

x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=1
) ## Complete the code to split the data into train and test in specified 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
```

Model Performance Summary

- Overview of ML model and its parameters
- Summary of most important factors used by the ML model for prediction
- Summary of key performance metrics for training and test data in tabular format for comparison

Link to Appendix slide on model assumptions

Model Performance Summary

- Overview of ML model (final) and its parameters, **Figure 26**
  - The Adjusted R^2 is 0.838, which is an exceptionally good model
    - The Adjusted R^2 values generally range from 0 to 1, where a higher value suggests a better fit, considering certain conditions are met
  - The constant coefficient, also known as the Y-intercept, is 1.50
    - To further explain, if the coefficients for the predictor variables are 0, then the output would be the constant coefficient
  - The coefficients of the predictor variables (14 total) is highlighted in **Figure 26**
    - For example, if the coefficient for 'main_camera_mp' increases by 1 unit and all other coefficients are constant, then the y (output) would change by 0.02
  - All features have a p-value of less than 0.5, which is why they were kept in the model

Figure 26

```
                            OLS Regression Results
==============================================================================
Dep. Variable:     normalized_used_price   R-squared:                       0.839
Model:                               OLS   Adj. R-squared:                  0.838
Method:                    Least Squares   F-statistic:                     895.7
Date:                   Sun, 25 Sep 2022   Prob (F-statistic):               0.00
Time:                           06:33:50   Log-Likelihood:                 80.645
No. Observations:                   2417   AIC:                            -131.3
Df Residuals:                       2402   BIC:                            -44.44
Df Model:                             14
Covariance Type:               nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
const                     1.5000      0.048     30.955      0.000       1.405       1.595
main_camera_mp            0.0210      0.001     14.714      0.000       0.018       0.024
selfie_camera_mp          0.0138      0.001     12.858      0.000       0.012       0.016
ram                       0.0207      0.005      4.151      0.000       0.011       0.030
weight                    0.0017      6e-05     27.672      0.000       0.002       0.002
normalized_new_price      0.4415      0.011     39.337      0.000       0.419       0.463
years_since_release      -0.0292      0.003     -8.589      0.000      -0.036      -0.023
brand_name_Karbonn        0.1156      0.055      2.111      0.035       0.008       0.223
brand_name_Samsung       -0.0374      0.016     -2.270      0.023      -0.070      -0.005
brand_name_Sony          -0.0670      0.030     -2.197      0.028      -0.127      -0.007
brand_name_Xiaomi         0.0801      0.026      3.114      0.002       0.030       0.130
os_Others                -0.1276      0.027     -4.667      0.000      -0.181      -0.074
os_iOS                   -0.0900      0.045     -1.994      0.046      -0.179      -0.002
4g_yes                    0.0502      0.015      3.326      0.001       0.021       0.080
5g_yes                   -0.0673      0.031     -2.194      0.028      -0.127      -0.007
==============================================================================
Omnibus:                      246.183   Durbin-Watson:                   1.902
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              483.879
Skew:                          -0.658   Prob(JB):                     8.45e-106
Kurtosis:                       4.753   Cond. No.                     2.39e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.39e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[Link to Appendix slide on model assumptions]()

Model Performance Summary

- Summary of most important factors used by the ML model for prediction, **Figure 26**
  - The equation for the ML model would be:
    - **‘normalized_used_price’** = 1.5 + 0.0210**(‘main_camera_mp’)** + 0.0138**(‘selfie_camera_mp’)**+0.0207**(‘ram’)**+0.0017 **(‘weight’)**+0.4415**(‘normalized_new_price’)-0**.0292(’**years_since_release**’)+0.1156**(‘brand_name _Karbonn)**-0.0374**(‘brand_name_Samsung**’)-0.0670**(‘brand_name_Sony’)**+0.0801**(‘brand_name_ Xiaomi**’)-0.1276**(‘os_Others**’)-0.0900**(‘os_iOS’)**+0.0502**(‘4g_yes**’)-0.0673**(‘5g_yes’)**
    - The above features are the most important factors used by the ML model for prediction

Figure 26

```
                            OLS Regression Results
==============================================================================
Dep. Variable:     normalized_used_price   R-squared:                   0.839
Model:                               OLS   Adj. R-squared:              0.838
Method:                    Least Squares   F-statistic:                 895.7
Date:                   Sun, 25 Sep 2022   Prob (F-statistic):           0.00
Time:                           06:33:50   Log-Likelihood:             80.645
No. Observations:                   2417   AIC:                        -131.3
Df Residuals:                       2402   BIC:                        -44.44
Df Model:                             14
Covariance Type:               nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
const                     1.5000      0.048     30.955      0.000       1.405       1.595
main_camera_mp            0.0210      0.001     14.714      0.000       0.018       0.024
selfie_camera_mp          0.0138      0.001     12.858      0.000       0.012       0.016
ram                       0.0207      0.005      4.151      0.000       0.011       0.030
weight                    0.0017      6e-05     27.672      0.000       0.002       0.002
normalized_new_price      0.4415      0.011     39.337      0.000       0.419       0.463
years_since_release      -0.0292      0.003     -8.589      0.000      -0.036      -0.023
brand_name_Karbonn        0.1156      0.055      2.111      0.035       0.008       0.223
brand_name_Samsung       -0.0374      0.016     -2.270      0.023      -0.070      -0.005
brand_name_Sony          -0.0670      0.030     -2.197      0.028      -0.127      -0.007
brand_name_Xiaomi         0.0801      0.026      3.114      0.002       0.030       0.130
os_Others                -0.1276      0.027     -4.667      0.000      -0.181      -0.074
os_iOS                   -0.0900      0.045     -1.994      0.046      -0.179      -0.002
4g_yes                    0.0502      0.015      3.326      0.001       0.021       0.080
5g_yes                   -0.0673      0.031     -2.194      0.028      -0.127      -0.007
==============================================================================
Omnibus:                      246.183   Durbin-Watson:                   1.902
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              483.879
Skew:                          -0.658   Prob(JB):                    8.45e-106
Kurtosis:                       4.753   Cond. No.                     2.39e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.39e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Link to Appendix slide on model assumptions

Model Performance Summary

- Summary of key performance metrics for training and test data in tabular format in **Figure 27**
  - As a result, the adjusted R^2 (for the training data) is 0.84 (able to explain 84% of the variation in data), therefore the model is not underfitting
  - The train and test RMSE and MAE are low and comparable, so the model is not overfitting either
  - MAE suggests that the model can predict used phone prices within a mean error of 0.18 on the test data
  - MAPE (Mean Absolute Percentage Error) of 4.5 on the test data means that we can predict within 4.5% of the used phone prices
  - As mentioned previously, the model is neither underfitting nor overfitting since the adjusted R-squared values for Training and Test performance are within 5% of each other

Figure 27

Training Performance

	RMSE	MAE	R-squared	Adj. R-squared	MAPE
0	0.229884	0.180326	0.844886	0.841675	4.326841

Test Performance

	RMSE	MAE	R-squared	Adj. R-squared	MAPE
0	0.238358	0.184749	0.842479	0.834659	4.501651

Link to Appendix slide on model assumptions

APPENDIX

Data Background and Contents

- The data set consists of information about the attributes of used/refurbished phones and tablets
  - There are 3454 rows of cell phones/tablets and 15 columns
    - The columns consist of :
      - ‘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’

First 5 rows of the dataset

	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

Data Background and Contents

- No duplicate values exist in the data

- Missing values do exist.
  - Under the **5g** column, there are 179 missing values
  - Under the **main_camera_mp**, there are 2 missing values
  - Under the **selfie_camera_mp**, there are 4 missing values
  - Under the **int_memory** column, there are 4 missing values
  - Under the **ram** column, there are 6 missing values
  - Under the **battery** column, there are 7 missing values

- In regards to the types of variables in the data set, there are 9 float types, 2 integer types, and 4 object types

- Statistical summary of the numerical values:
  - The range for the normalized used price (the target variable) is from 1.54 – 6.62 Euros
    - The mean and median for the normalized used price is respectively 4.36 and 4.41 Euros
  - The range for the normalized new price is 2.90 – 7.85 Euros
    - The mean and median for the normalized new price is respectively 5.23 and 5.25 Euros
  - The release year for the phones/tablets varies from 2013-2020
  - The phones/tablets weight varies from 69-855 grams
  - The battery varies from 900 – 9720 mAh

Data Background and Contents

- Statistical summary of the numerical values
  - The ram of the phones/tablets varies from 0.20 – 12 GB
  - The internal memory of the phones/tablets varies from 0.01 – 1024 GB
  - Resolution of the front camera of the phones/tablets varies from 0 – 32 MP
  - Resolution of the rear camera of the phones/tablets varies from 0.08 – 48 MP
  - The screen size of the phones/tablets varies from 5.08 – 30.71 CM

Model Assumptions

- Tests conducted for checking model assumptions and the results obtained
  - **Test for Multicollinearity**
    - 48 features were utilized to test for multicollinearity
    - The dummy variables were excluded
    - ‘Screen_size’ and ‘weight’ were the two columns that had VIFs >5 according to **Figure 28**
      - If VIF is greater than 5 for a predictor variable, then that variable should be dropped
    - To remove multicollinearity
      - each column that has VIF>5 is dropped one by one,
      - look at the adjusted R^2 and RMSE (in **Figure 29**) to discover the variable making the least change in adjusted R^2 after being dropped

Figure 28

```
checking_vif(x_train)  ## Complete the code to check VIF on train data
```

	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
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

Figure 29

```
col_list = ['screen_size','weight'] ## Complete the code to specify the columns with high VIF

res = treating_multicollinearity(x_train, y_train, col_list) ## Complete the code to check the effect
res
```

	col	Adj. R-squared after_dropping col	RMSE after dropping col
0	screen_size	0.838381	0.234703
1	weight	0.838071	0.234928

Model Assumptions

- Please mention the tests conducted for checking model assumptions and the results obtained
  - **Test for Multicollinearity**
    - To remove multicollinearity
      - ‘Screen size’ was dropped since its adjusted R^2 changed the least
    - After removing ‘Screen size’ and checking the VIF for the new training set, all predictor variables had VIF<5:

VIF after dropping screen_size

	feature	VIF
0	const	202.673906
1	main_camera_mp	2.281835
2	selfie_camera_mp	2.809009
3	int_memory	1.362043
4	ram	2.282350
5	battery	3.842989
6	weight	2.993855
7	days_used	2.648929
8	normalized_new_price	3.077650
9	years_since_release	4.730315

Model Assumptions

- Please mention the tests conducted for checking model assumptions and the results obtained
    - **Dropping p-values (not a part of the 5 assumptions for linear regression)**
        - Predictor variables that had a p-value >0.05 were dropped as they did not significantly have an impact on the target variable
        - After dropping the p-values, the OLS model was run again and the results are to the right
        - As you can see, the p-values were all below 0.05 in the Regression results

Figure 30

OLS Regression Results

Dep. Variable:	normalized_used_price	R-squared:	0.839
Model:	OLS	Adj. R-squared:	0.838
Method:	Least Squares	F-statistic:	895.7
Date:	Sun, 25 Sep 2022	Prob (F-statistic):	0.00
Time:	06:30:04	Log-Likelihood:	80.645
No. Observations:	2417	AIC:	-131.3
Df Residuals:	2402	BIC:	-44.44
Df Model:	14		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.5000	0.048	30.955	0.000	1.405	1.595
main_camera_mp	0.0210	0.001	14.714	0.000	0.018	0.024
selfie_camera_mp	0.0138	0.001	12.858	0.000	0.012	0.016
ram	0.0207	0.005	4.151	0.000	0.011	0.030
weight	0.0017	6e-05	27.672	0.000	0.002	0.002
normalized_new_price	0.4415	0.011	39.337	0.000	0.419	0.463
years_since_release	-0.0292	0.003	-8.589	0.000	-0.036	-0.023
brand_name_Karbonn	0.1156	0.055	2.111	0.035	0.008	0.223
brand_name_Samsung	-0.0374	0.016	-2.270	0.023	-0.070	-0.005
brand_name_Sony	-0.0670	0.030	-2.197	0.028	-0.127	-0.007
brand_name_Xiaomi	0.0801	0.026	3.114	0.002	0.030	0.130
os_Others	-0.1276	0.027	-4.667	0.000	-0.181	-0.074
os_iOS	-0.0900	0.045	-1.994	0.046	-0.179	-0.002
4g_yes	0.0502	0.015	3.326	0.001	0.021	0.080
5g_yes	-0.0673	0.031	-2.194	0.028	-0.127	-0.007

Omnibus:	246.183	Durbin-Watson:	1.902
Prob(Omnibus):	0.000	Jarque-Bera (JB):	483.879
Skew:	-0.658	Prob(JB):	8.45e-106
Kurtosis:	4.753	Cond. No.	2.39e+03

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.39e+03. This might indicate that there are strong multicollinearity or other numerical problems.

Model Assumptions

- Please mention the tests conducted for checking model assumptions and the results obtained
  - **Test for Linearity and Independence**
    - Since no pattern exists in **Figure 31** (Fitted Values vs Residuals graph), the model is linear and the residuals are independent
    - Otherwise, if a pattern was discovered, the model would be non-linear and residuals dependent

Figure 31

Model Assumptions

- Please mention the tests conducted for checking model assumptions and the results obtained
    - **Test for Normality**
        - Based on the results from the distribution of residuals (**Figure 32**), the data resembles a normally distributed curve
        - The Q-Q plot of residuals in **Figure 33** suggests that the residuals mostly follow a straight line except for the tails
        - Based on the results from the Shapiro Wilk's test **(Figure 34),** the residuals are not normal since the p value is less than 0.05. In strict terms, the residual values are not normal but they are close to a normal distribution. Therefore, the assumption is satisfied.

Figure 32

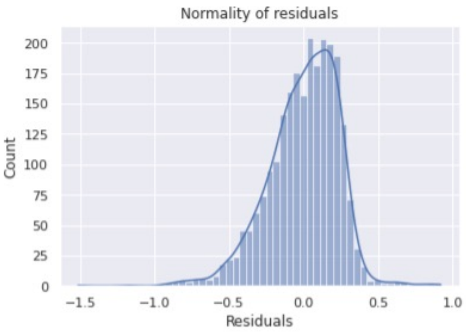

Figure 33

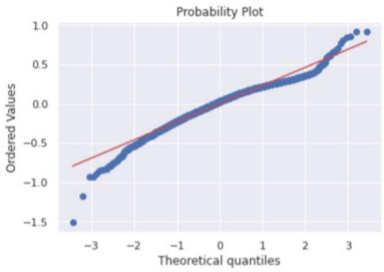

Figure 34

```
[ ] stats.shapiro(df_pred['Residuals']) #

ShapiroResult(statistic=0.9676972031593323, pvalue=6.995328206686811e-23)
```

Model Assumptions

- Please mention the tests conducted for checking model assumptions and the results obtained
  - **Test for homoscedasticity**
    - Goldfeldquandt test was used to test for homoscedasticity in **Figure 35**
    - Since the p-value is greater than 0.05, the residuals are homoscedastic

Figure 35

```
import statsmodels.stats.api as sms
from statsmodels.compat import lzip

name = ["F statistic", "p-value"]
test = sms.het_goldfeldquandt(df_pred["Residuals"], x_train3)
lzip(name, test)
```

```
[('F statistic', 1.008750419910676), ('p-value', 0.4401970650667301)]
```

Great Learning

Happy Learning !