Capstone Project

Mobile Price Range Prediction

BY – AMOGHA K

CONTENT

- Introduction
- Problem Statement
- Data Summary
- Data processing
- Exploratory Data analysis
- Machine learning algorithms
- Conclusion

Introduction

- Mobile phones are now an essential commodity for us to connect to the world or our loved ones, resulting in a huge amount of data being generated as a result of the massive number of mobile phones manufactured.

- People want more features and the best specifications in a phone, and they want them at a cheaper price.

- Mobile phones come in a variety of prices, features, and specifications. Price estimation and prediction are important parts of consumer strategy. Deciding on the correct price of a product is very important for its market success.

Problem Statement

- In the competitive mobile phone market companies want to understand sales data of mobile phones and factors which drive the prices. The objective is to find out some relation between features of a mobile phone (e.g.:- RAM, Internal Memory, etc) and its selling price.

- The problem statement is to predict the price range of mobile phones based on the features available (price range indicating how high the price is). Here is the description of target classes:

  0 - Low cost Phones

  1 - Medium cost phones

  2 - High cost phones

  3 - Very High cost phones

- The main objective of this project is to build a model which will classify the price range of mobile phones based on the specifications of mobile phones.

Data Summary

Independent variables:-

- Battery_power - Total energy a battery can store in one time measured in mAh
- Blue - Has bluetooth or not
- Clock_speed - speed at which microprocessor executes instructions
- Dual_sim - Has dual sim support or not
- Fc - Front Camera mega pixels
- Four_g - Has 4G or not
- Int_memory - Internal Memory in Gigabytes
- M_dep - Mobile Depth in cm
- Mobile_wt - Weight of mobile phone
- N_cores - Number of cores of processor

Data Summary

- Pc - Primary Camera mega pixels
- Px_height - Pixel Resolution Height
- Px_width - Pixel Resolution Width
- Ram - Random Access Memory in Mega Bytes
- Sc_h - Screen Height of mobile in cm
- Sc_w - Screen Width of mobile in cm
- Talk_time - longest time that a single battery charge will last when you are
- Three_g - Has 3G or not
- Touch_screen - Has touch screen or not
- Wifi - Has wifi or not

Dependent variables :-

- Price_range- This is the target variable with value of

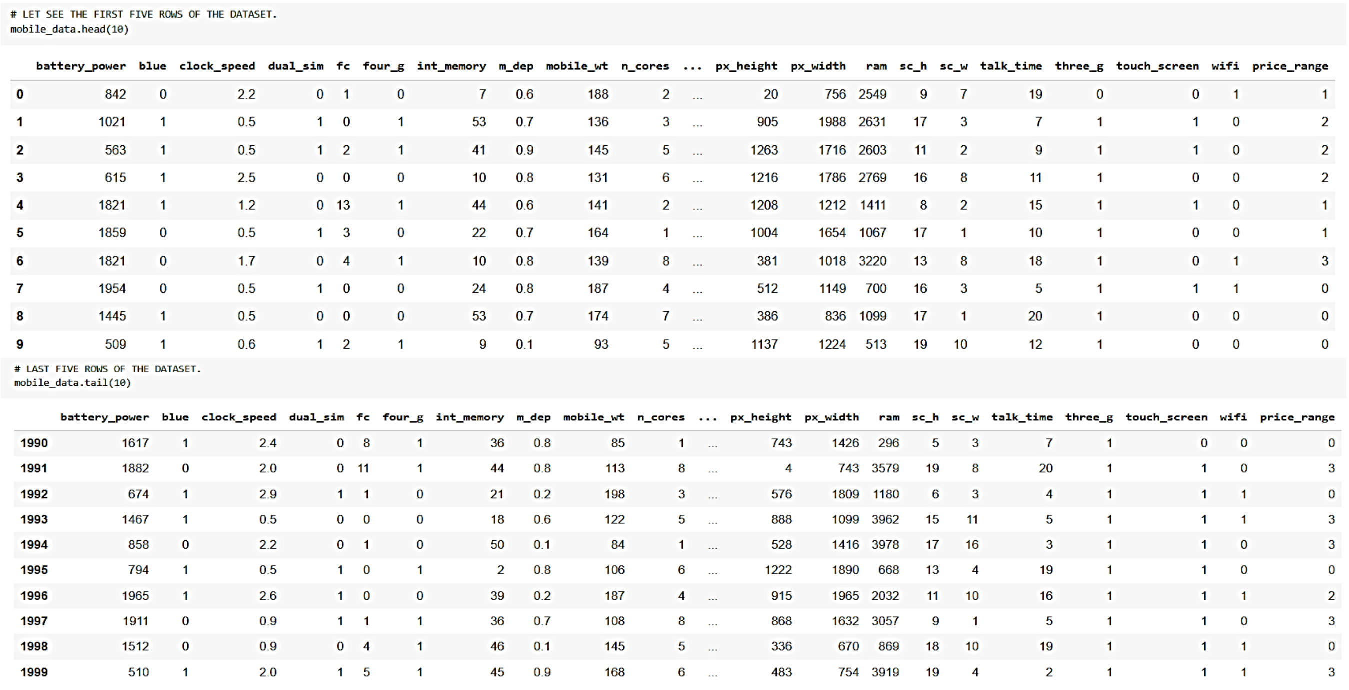
  0 - (low cost),

  1 - (medium cost),

  2 - (high cost)

  3 - (very high cost).

Data processing



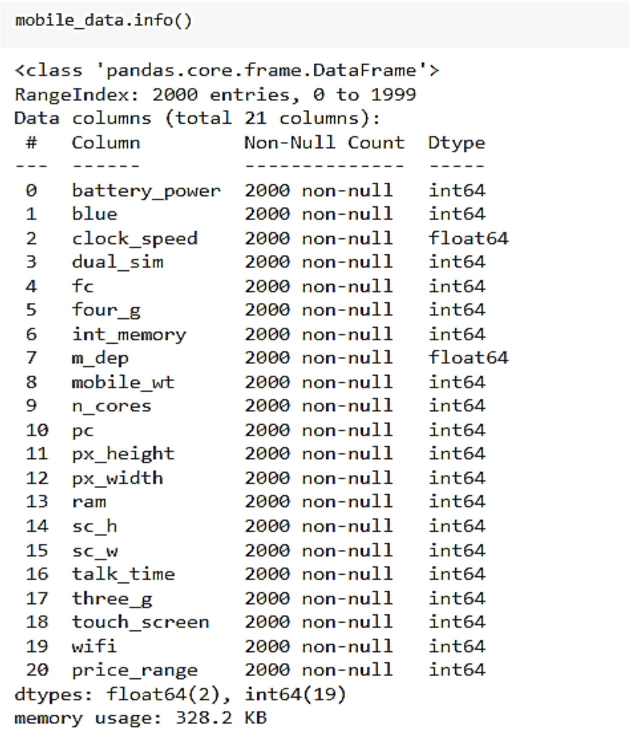
```
# LET SEE THE FIRST FIVE ROWS OF THE DATASET.
mobile_data.head(10)
```

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_height	px_width	ram	sc_h	sc_w	talk_time	three_g	touch_screen	wifi	price_range
0	842	0	2.2	0	1	0	7	0.6	188	2	...	20	756	2549	9	7	19	0	0	1	1
1	1021	1	0.5	1	0	1	53	0.7	136	3	...	905	1988	2631	17	3	7	1	1	0	2
2	563	1	0.5	1	2	1	41	0.9	145	5	...	1263	1716	2603	11	2	9	1	1	0	2
3	615	1	2.5	0	0	0	10	0.8	131	6	...	1216	1786	2769	16	8	11	1	0	0	2
4	1821	1	1.2	0	13	1	44	0.6	141	2	...	1208	1212	1411	8	2	15	1	1	0	1
5	1859	0	0.5	1	3	0	22	0.7	164	1	...	1004	1654	1067	17	1	10	1	0	0	1
6	1821	0	1.7	0	4	1	10	0.8	139	8	...	381	1018	3220	13	8	18	1	0	1	3
7	1954	0	0.5	1	0	0	24	0.8	187	4	...	512	1149	700	16	3	5	1	1	1	0
8	1445	1	0.5	0	0	0	53	0.7	174	7	...	386	836	1099	17	1	20	1	0	0	0
9	509	1	0.6	1	2	1	9	0.1	93	5	...	1137	1224	513	19	10	12	1	0	0	0

```
# LAST FIVE ROWS OF THE DATASET.
mobile_data.tail(10)
```

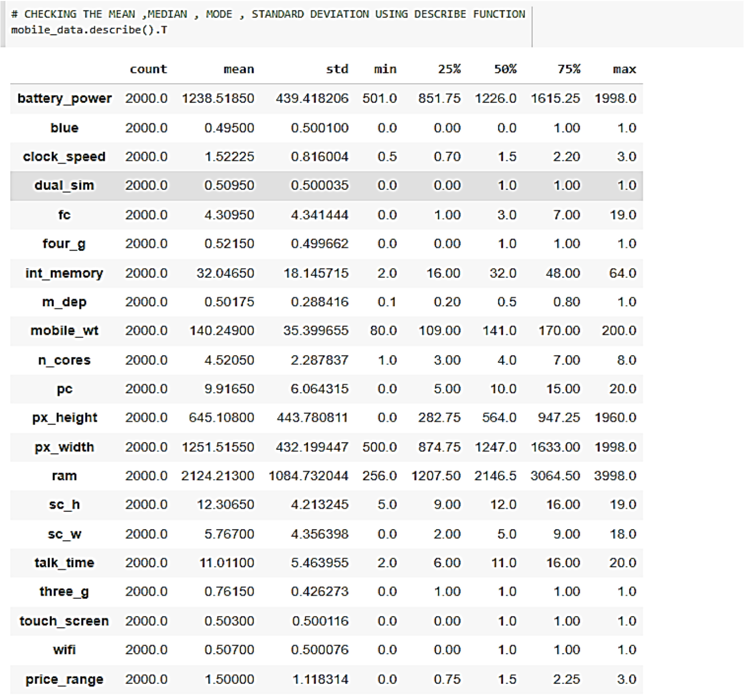
	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_height	px_width	ram	sc_h	sc_w	talk_time	three_g	touch_screen	wifi	price_range
1990	1617	1	2.4	0	8	1	36	0.8	85	1	...	743	1426	296	5	3	7	1	0	0	0
1991	1882	0	2.0	0	11	1	44	0.8	113	8	...	4	743	3579	19	8	20	1	1	0	3
1992	674	1	2.9	1	1	0	21	0.2	198	3	...	576	1809	1180	6	3	4	1	1	1	0
1993	1467	1	0.5	0	0	0	18	0.6	122	5	...	888	1099	3962	15	11	5	1	1	1	3
1994	858	0	2.2	0	1	0	50	0.1	84	1	...	528	1416	3978	17	16	3	1	1	0	3
1995	794	1	0.5	1	0	1	2	0.8	106	6	...	1222	1890	668	13	4	19	1	1	0	0
1996	1965	1	2.6	1	0	0	39	0.2	187	4	...	915	1965	2032	11	10	16	1	1	1	2
1997	1911	0	0.9	1	1	1	36	0.7	108	8	...	868	1632	3057	9	1	5	1	1	0	3
1998	1512	0	0.9	0	4	1	46	0.1	145	5	...	336	670	869	18	10	19	1	1	1	0
1999	510	1	2.0	1	5	1	45	0.9	168	6	...	483	754	3919	19	4	2	1	1	1	3

- To view a small sample of a Series or the DataFrame object, use the head() and the tail() methods



```
mobile_data.info()
```

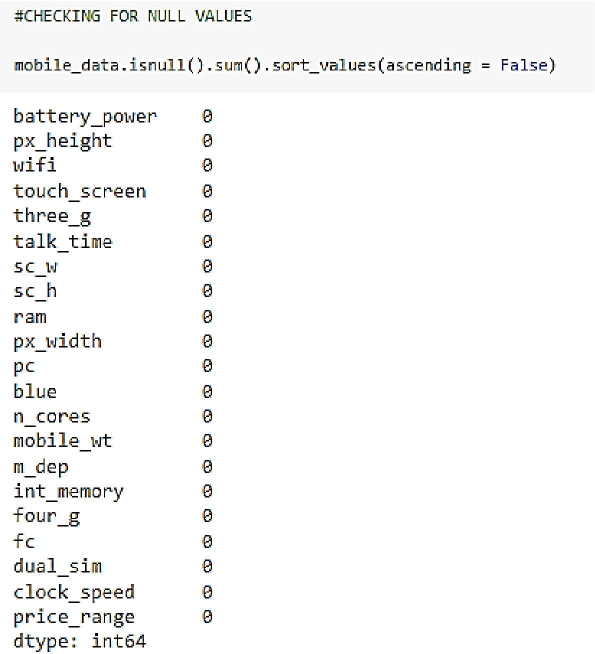
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   battery_power  2000 non-null   int64  
 1   blue           2000 non-null   int64  
 2   clock_speed    2000 non-null   float64
 3   dual_sim       2000 non-null   int64  
 4   fc             2000 non-null   int64  
 5   four_g         2000 non-null   int64  
 6   int_memory     2000 non-null   int64  
 7   m_dep          2000 non-null   float64
 8   mobile_wt      2000 non-null   int64  
 9   n_cores        2000 non-null   int64  
 10  pc             2000 non-null   int64  
 11  px_height      2000 non-null   int64  
 12  px_width       2000 non-null   int64  
 13  ram            2000 non-null   int64  
 14  sc_h           2000 non-null   int64  
 15  sc_w           2000 non-null   int64  
 16  talk_time      2000 non-null   int64  
 17  three_g        2000 non-null   int64  
 18  touch_screen   2000 non-null   int64  
 19  wifi           2000 non-null   int64  
 20  price_range    2000 non-null   int64  
dtypes: float64(2), int64(19)
memory usage: 328.2 KB
```



```
# CHECKING THE MEAN ,MEDIAN , MODE , STANDARD DEVIATION USING DESCRIBE FUNCTION
mobile_data.describe().T
```

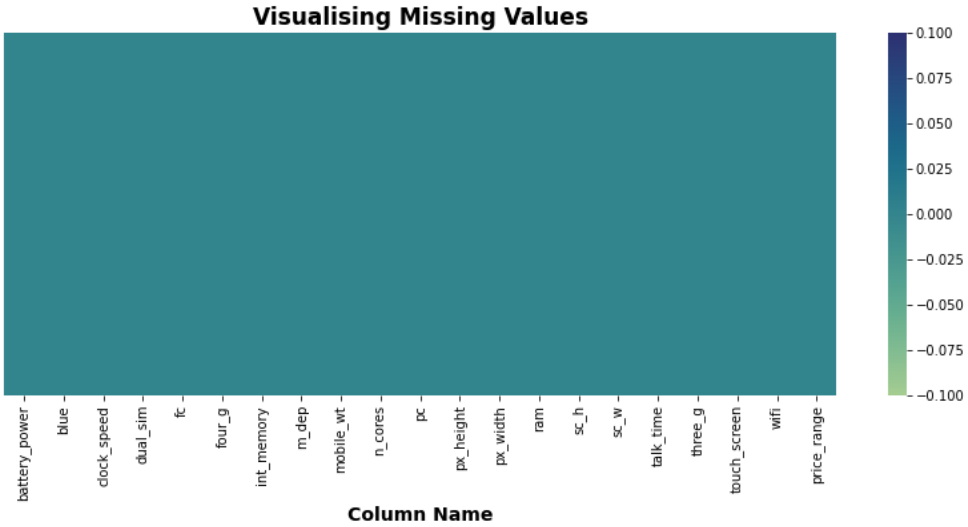
	count	mean	std	min	25%	50%	75%	max
battery_power	2000.0	1238.51850	439.418206	501.0	851.75	1226.0	1615.25	1998.0
blue	2000.0	0.49500	0.500100	0.0	0.00	0.0	1.00	1.0
clock_speed	2000.0	1.52225	0.816004	0.5	0.70	1.5	2.20	3.0
dual_sim	2000.0	0.50950	0.500035	0.0	0.00	1.0	1.00	1.0
fc	2000.0	4.30950	4.341444	0.0	1.00	3.0	7.00	19.0
four_g	2000.0	0.52150	0.499662	0.0	0.00	1.0	1.00	1.0
int_memory	2000.0	32.04650	18.145715	2.0	16.00	32.0	48.00	64.0
m_dep	2000.0	0.50175	0.288416	0.1	0.20	0.5	0.80	1.0
mobile_wt	2000.0	140.24900	35.399655	80.0	109.00	141.0	170.00	200.0
n_cores	2000.0	4.52050	2.287837	1.0	3.00	4.0	7.00	8.0
pc	2000.0	9.91650	6.064315	0.0	5.00	10.0	15.00	20.0
px_height	2000.0	645.10800	443.780811	0.0	282.75	564.0	947.25	1960.0
px_width	2000.0	1251.51550	432.199447	500.0	874.75	1247.0	1633.00	1998.0
ram	2000.0	2124.21300	1084.732044	256.0	1207.50	2146.5	3064.50	3998.0
sc_h	2000.0	12.30650	4.213245	5.0	9.00	12.0	16.00	19.0
sc_w	2000.0	5.76700	4.356398	0.0	2.00	5.0	9.00	18.0
talk_time	2000.0	11.01100	5.463955	2.0	6.00	11.0	16.00	20.0
three_g	2000.0	0.76150	0.426273	0.0	1.00	1.0	1.00	1.0
touch_screen	2000.0	0.50300	0.500116	0.0	0.00	1.0	1.00	1.0
wifi	2000.0	0.50700	0.500076	0.0	0.00	1.0	1.00	1.0
price_range	2000.0	1.50000	1.118314	0.0	0.75	1.5	2.25	3.0

- ✓ We can quickly determine the data type and null values in our dataframes by using the info() method.
- ✓ We don't have any object data type in our data set.
- ✓ The dataset has total of 21 columns and 2000 rows



```
#CHECKING FOR NULL VALUES

mobile_data.isnull().sum().sort_values(ascending = False)
```

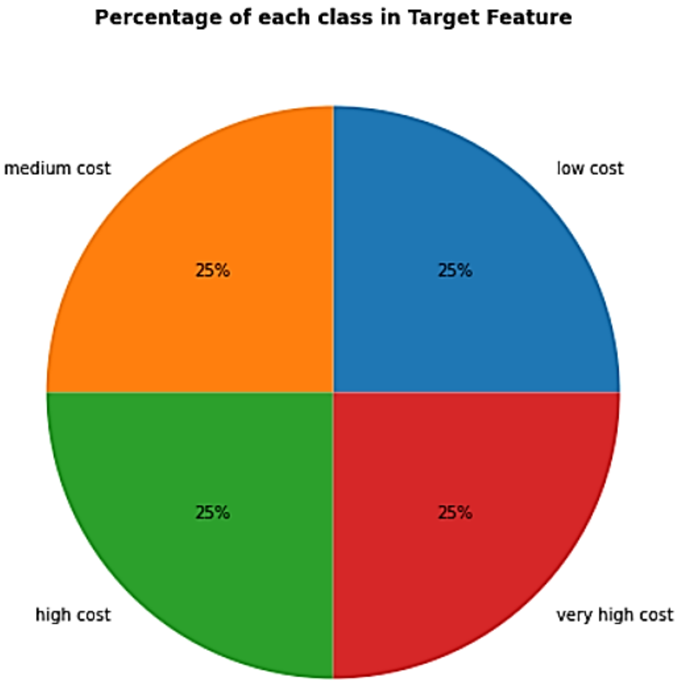
```
battery_power    0
px_height        0
wifi             0
touch_screen     0
three_g          0
talk_time        0
sc_w             0
sc_h             0
ram              0
px_width         0
pc               0
blue             0
n_cores          0
mobile_wt        0
m_dep            0
int_memory       0
four_g           0
fc               0
dual_sim         0
clock_speed      0
price_range      0
dtype: int64
```



- As we can see, in our mobile_data, we have zero null values. which means the data was well maintained.

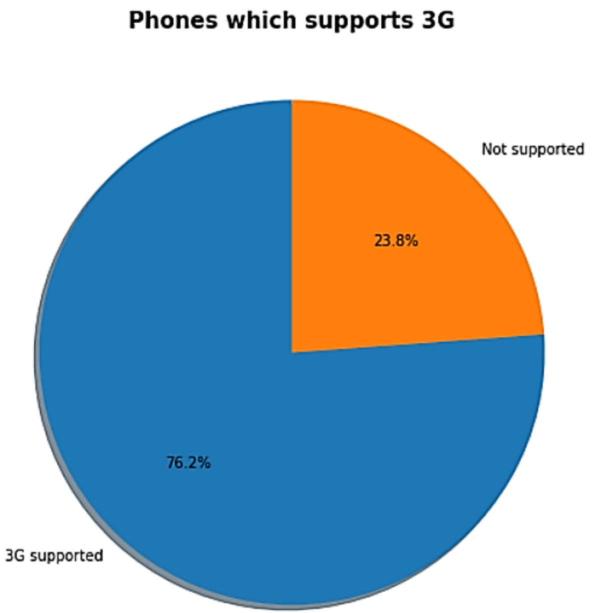
Exploratory Data analysis

PERCENTAGE OF EACH CLASS IN THE TARGET FEATURE

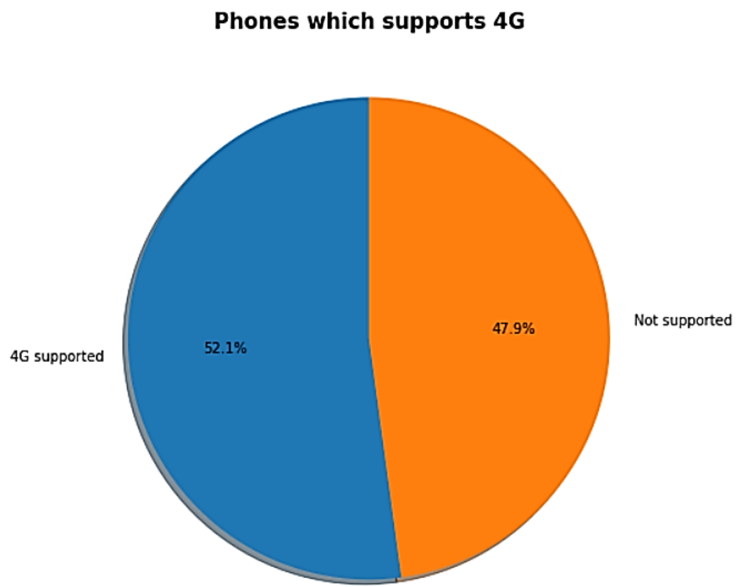


- ❖ Each class has an almost equal number of observations for each category. So our target feature is well balanced. The accuracy score will be the best evaluation metric for us to select the model.
- ❖ We can see that our target variable is equally distributed.
- ❖ There is no need for oversampling or undersampling because our data is balanced.

Percentage of phones which supports 3G

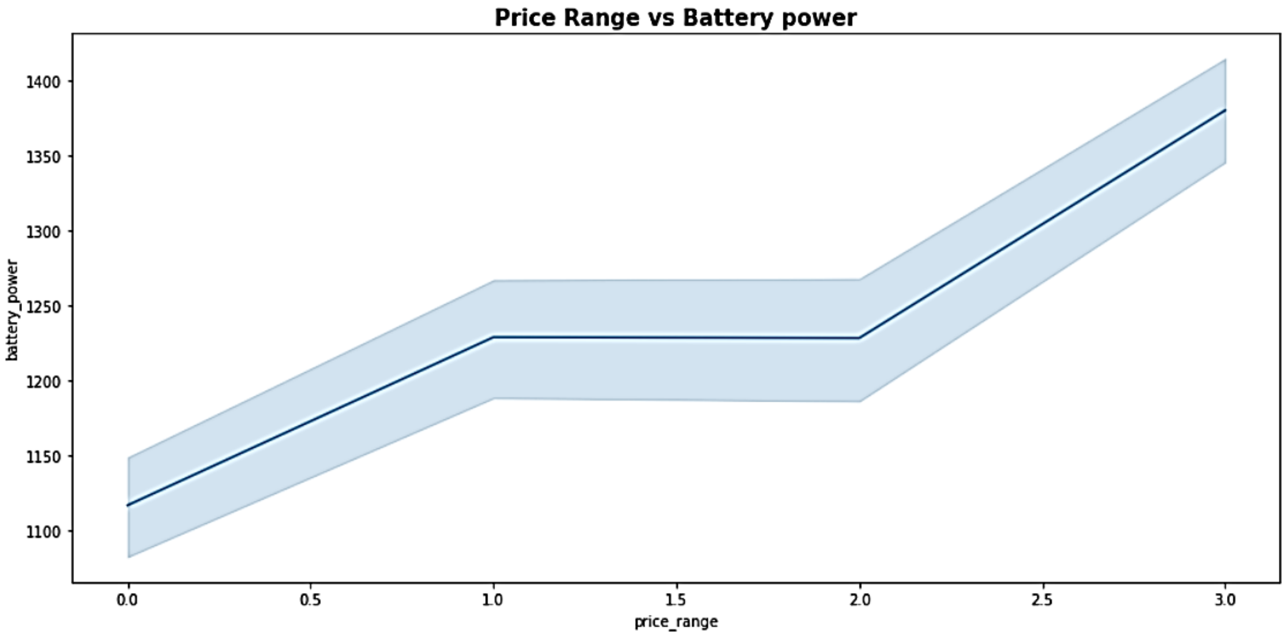


Percentage of phones which supports 4G



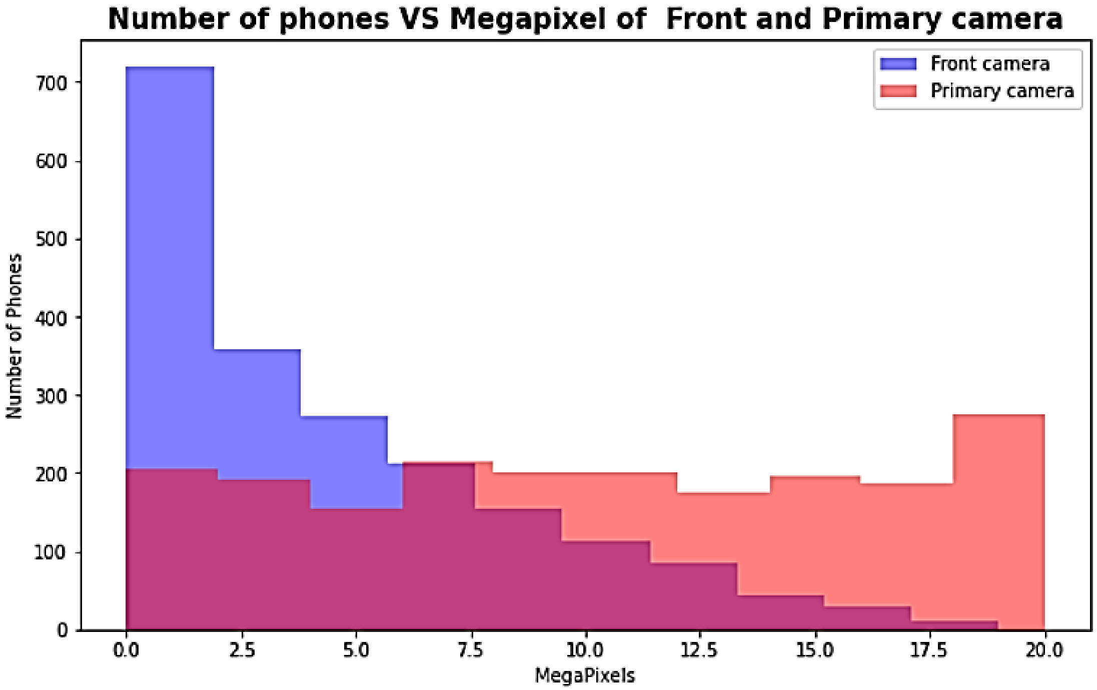
- According to the above graph, 76.2% of phones support 3G, while the remaining 23.8% do not. So 75% of our data has 3G support.
- According to the above graph, 52.1% of phones support 4G, while the remaining 47.9% do not.

Price Range VS Battery power



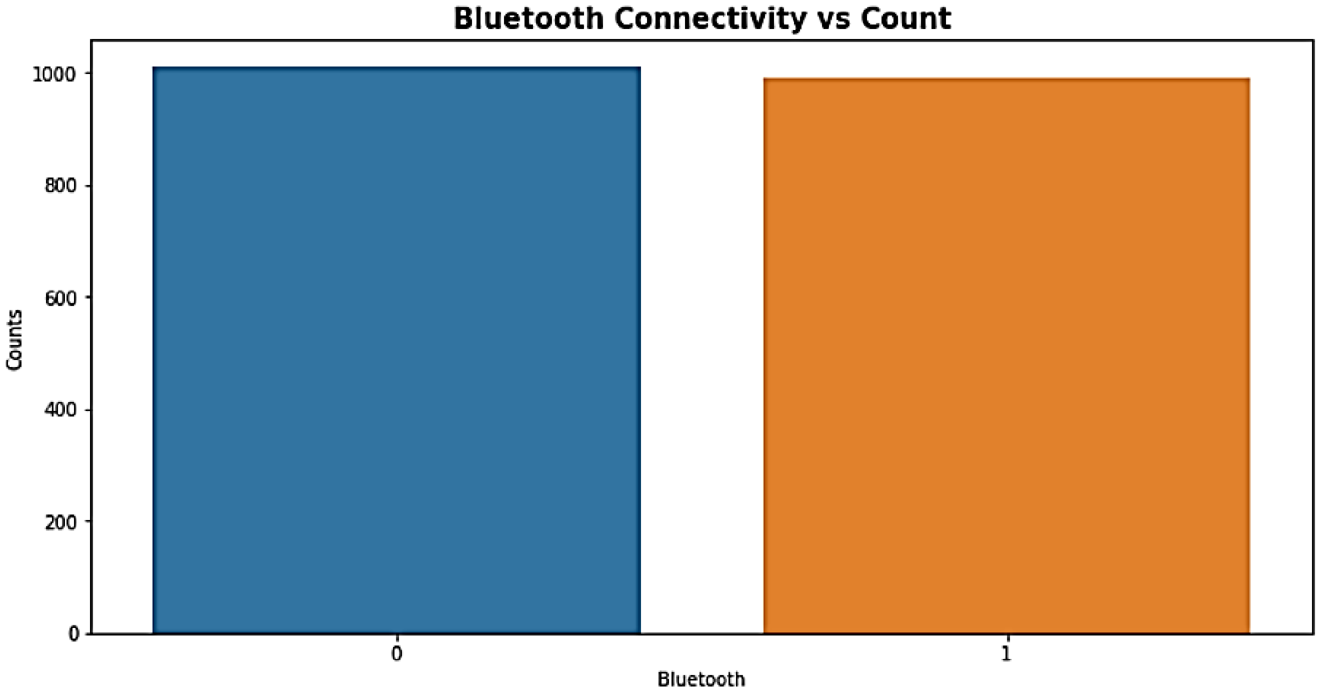
- ➢ Mobiles with battery power greater than 1300 mAh have a very high cost, and mobiles with battery power between 1200 and 1300 mAh fall into the medium and high cost categories.

Number of Phones vs Camera megapixels of front and primary camera



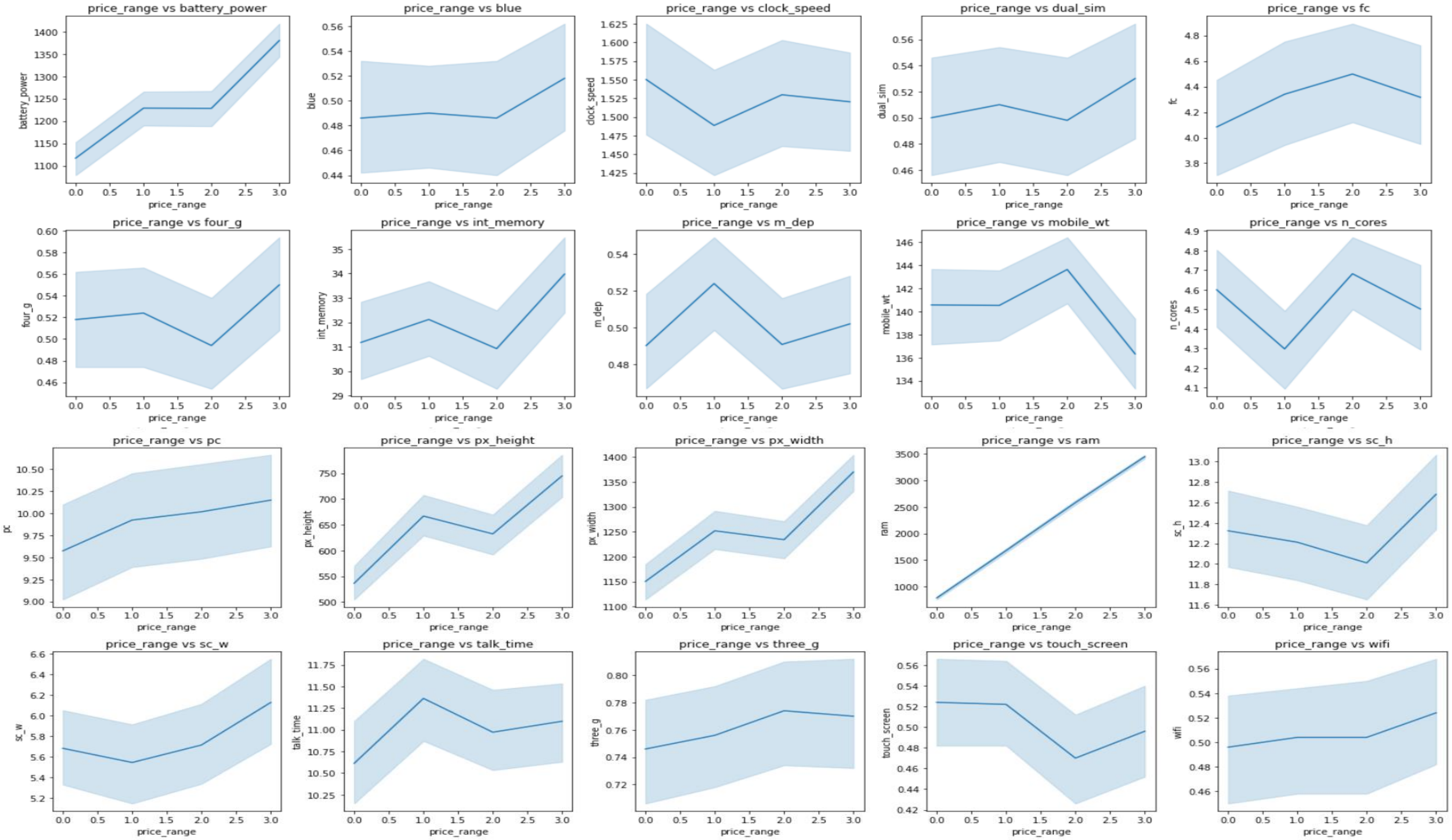
- ❖ Based on our findings, most phones have low megapixels in the front camera.

Bluetooth VS Mobile Phones



- ✓ As we can see, roughly half of the devices have Bluetooth connectivity, while the other half do not.

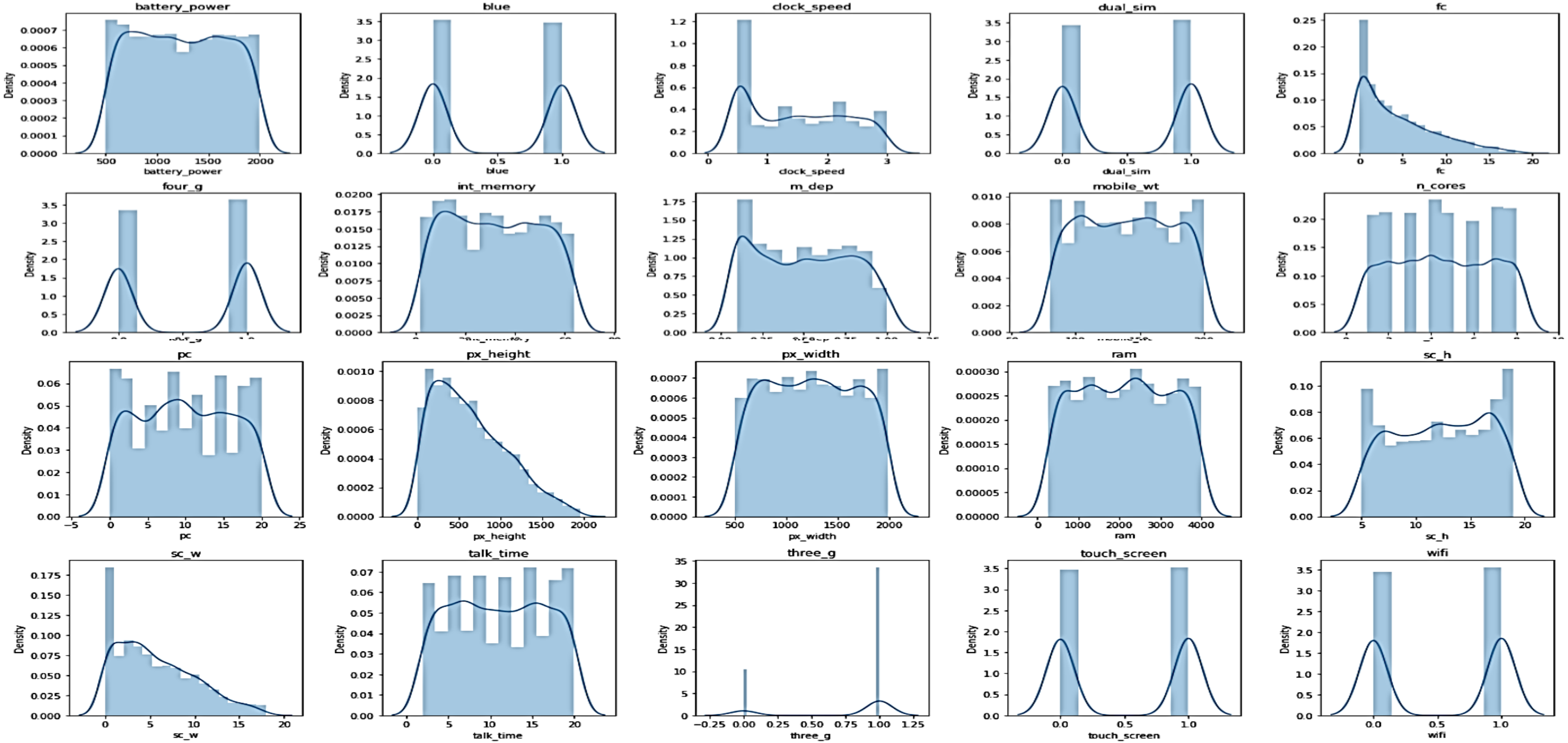
NUMIERICAL FEATURE WITH PRICE RANGE



NUMIERICAL FEATURE WITH PRICE RANGE - EXPLANATION

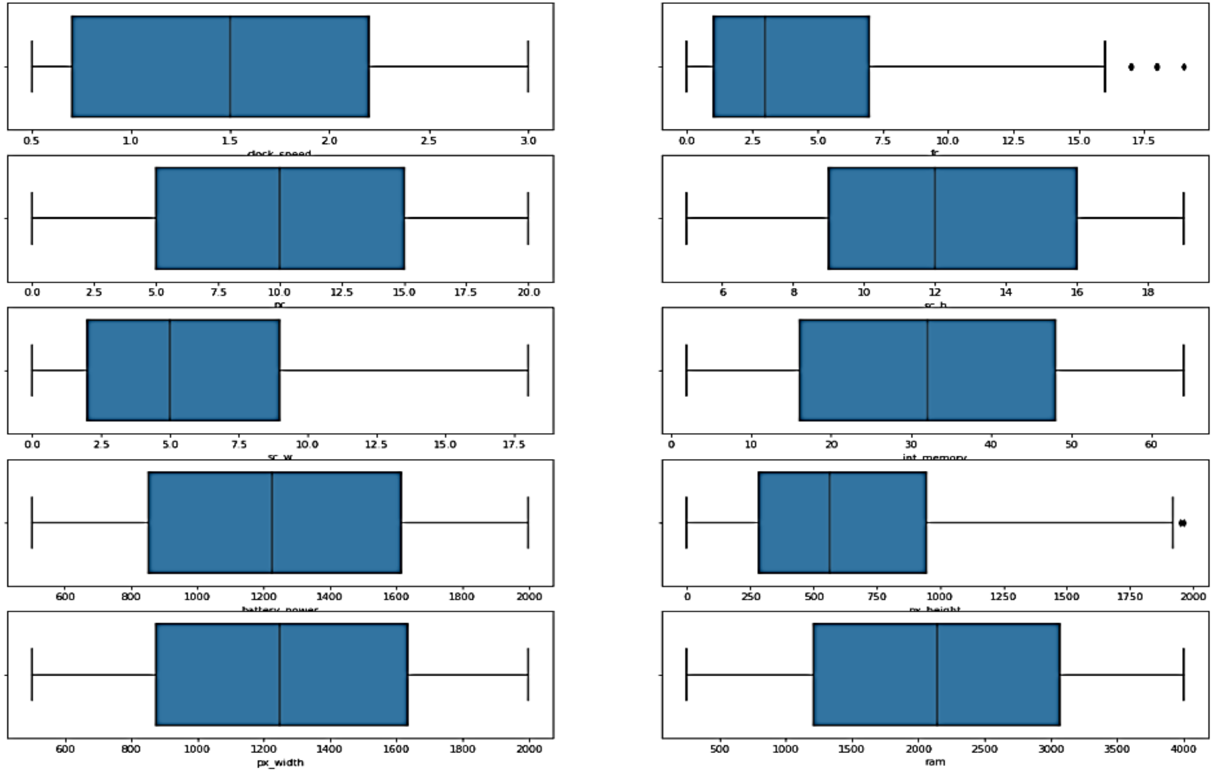
- The power range of class 1 and class 2 batteries is nearly identical. As battery power increases, the price also increases, which is quite obvious..
- Mobiles in a very high price range (Class 3) have less weight compared to other classes. That means as the weight of mobiles decreases, their price increases.
- Mobile phones with the largest screen height and width are extremely expensive. We can see in the linechart of sc_width and sc_height from class 2 that screen width and height start increasing with price. A similar case is with px_height and px_width. When the resolution of the screen increases, the price also increases.
- RAM has a clear relationship with price range.

Distribution of our each features



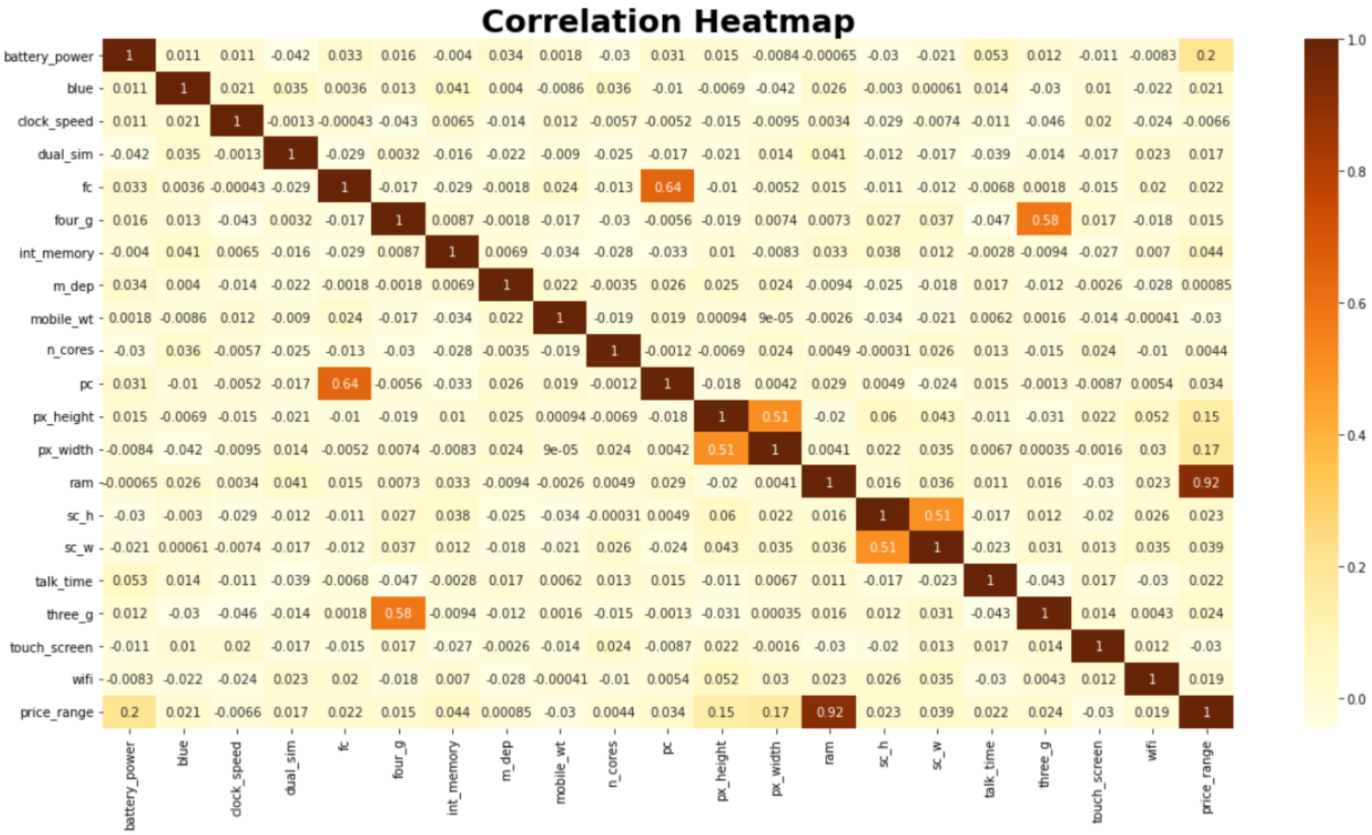
➢ Most of our features look similar to normal distribution and some features have multi mode because one of those columns has categories, which hold values like 0 or 1.

Outlier Detection



✓ As we see, there are not too many outliers. So we can move forward.

CORRELATION ANALYSIS



Correlation Heatmap

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	pc	px_height	px_width	ram	sc_h	sc_w	talk_time	three_g	touch_screen	wifi	price_range
battery_power	1	0.011	0.011	-0.042	0.033	0.016	-0.004	0.034	0.0018	-0.03	0.031	0.015	-0.0084	-0.00065	-0.03	-0.021	0.053	0.012	-0.011	-0.0083	0.2
blue	0.011	1	0.021	0.035	0.0036	0.013	0.041	0.004	-0.0086	0.036	-0.01	-0.0069	-0.042	0.026	-0.003	0.00061	0.014	-0.03	0.01	-0.022	0.021
clock_speed	0.011	0.021	1	-0.0013	-0.00043	-0.043	0.0065	-0.014	0.012	-0.0057	-0.0052	-0.015	-0.0095	0.0034	-0.029	-0.0074	-0.011	-0.046	0.02	-0.024	-0.0066
dual_sim	-0.042	0.035	-0.0013	1	-0.029	0.0032	-0.016	-0.022	-0.009	-0.025	-0.017	-0.021	0.014	0.041	-0.012	-0.017	-0.039	-0.014	-0.017	0.023	0.017
fc	0.033	0.0036	-0.00043	-0.029	1	-0.017	-0.029	-0.0018	0.024	-0.013	0.64	-0.01	-0.0052	0.015	-0.011	-0.012	-0.0068	0.0018	-0.015	0.02	0.022
four_g	0.016	0.013	-0.043	0.0032	-0.017	1	0.0087	-0.0018	-0.017	-0.03	-0.0056	-0.019	0.0074	0.0073	0.027	0.037	-0.047	0.58	0.017	-0.018	0.015
int_memory	-0.004	0.041	0.0065	-0.016	-0.029	0.0087	1	0.0069	-0.034	-0.028	-0.033	0.01	-0.0083	0.033	0.038	0.012	-0.0028	-0.0094	-0.027	0.007	0.044
m_dep	0.034	0.004	-0.014	-0.022	-0.0018	-0.0018	0.0069	1	0.022	-0.0035	0.026	0.025	0.024	-0.0094	-0.025	-0.018	0.017	-0.012	-0.0026	-0.028	0.00085
mobile_wt	0.0018	-0.0086	0.012	-0.009	0.024	-0.017	-0.034	0.022	1	-0.019	0.019	0.00094	9e-05	-0.0026	-0.034	-0.021	0.0062	0.0016	-0.014	-0.00041	-0.03
n_cores	-0.03	0.036	-0.0057	-0.025	-0.013	-0.03	-0.028	-0.0035	-0.019	1	-0.0012	-0.0069	0.024	0.0049	-0.00031	0.026	0.013	-0.015	0.024	-0.01	0.0044
pc	0.031	-0.01	-0.0052	-0.017	0.64	-0.0056	-0.033	0.026	0.019	-0.0012	1	-0.018	0.0042	0.029	0.0049	-0.024	0.015	-0.0013	-0.0087	0.0054	0.034
px_height	0.015	-0.0069	-0.015	-0.021	-0.01	-0.019	0.01	0.025	0.00094	-0.0069	-0.018	1	0.51	-0.02	0.06	0.043	-0.011	-0.031	0.022	0.052	0.15
px_width	-0.0084	-0.042	-0.0095	0.014	-0.0052	0.0074	-0.0083	0.024	9e-05	0.024	0.0042	0.51	1	0.0041	0.022	0.035	0.0067	0.00035	-0.0016	0.03	0.17
ram	-0.00065	0.026	0.0034	0.041	0.015	0.0073	0.033	-0.0094	-0.0026	0.0049	0.029	-0.02	0.0041	1	0.016	0.036	0.011	0.016	-0.03	0.023	0.92
sc_h	-0.03	-0.003	-0.029	-0.012	-0.011	0.027	0.038	-0.025	-0.034	-0.00031	0.0049	0.06	0.022	0.016	1	0.51	-0.017	0.012	-0.02	0.026	0.023
sc_w	-0.021	0.00061	-0.0074	-0.017	-0.012	0.037	0.012	-0.018	-0.021	0.026	-0.024	0.043	0.035	0.036	0.51	1	-0.023	0.031	0.013	0.035	0.039
talk_time	0.053	0.014	-0.011	-0.039	-0.0068	-0.047	-0.0028	0.017	0.0062	0.013	0.015	-0.011	0.0067	0.011	-0.017	-0.023	1	-0.043	0.017	-0.03	0.022
three_g	0.012	-0.03	-0.046	-0.014	0.0018	0.58	-0.0094	-0.012	0.0016	-0.015	-0.0013	-0.031	0.00035	0.016	0.012	0.031	-0.043	1	0.014	0.0043	0.024
touch_screen	-0.011	0.01	0.02	-0.017	-0.015	0.017	-0.027	-0.0026	-0.014	0.024	-0.0087	0.022	-0.0016	-0.03	-0.02	0.013	0.017	0.014	1	0.012	-0.03
wifi	-0.0083	-0.022	-0.024	0.023	0.02	-0.018	0.007	-0.028	-0.00041	-0.01	0.0054	0.052	0.03	0.023	0.026	0.035	-0.03	0.0043	0.012	1	0.019
price_range	0.2	0.021	-0.0066	0.017	0.022	0.015	0.044	0.00085	-0.03	0.0044	0.034	0.15	0.17	0.92	0.023	0.039	0.022	0.024	-0.03	0.019	1

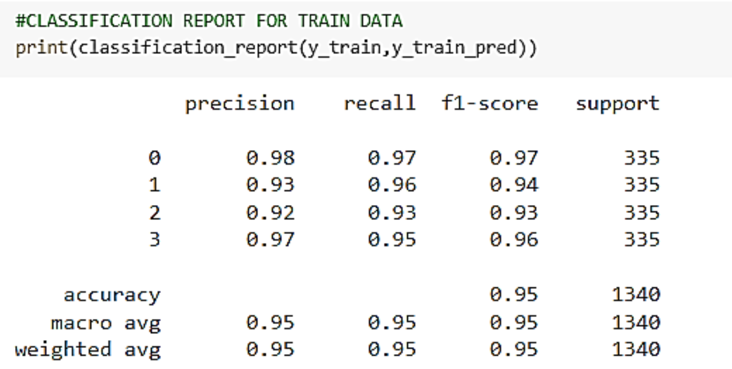
CORRELATION ANALYSIS - EXPLANATION

- RAM and price_range show a high correlation, which is a good sign because it signifies that RAM will be a major deciding factor in estimating the price range.
- There is some collinearity in feature pairs ('pc', 'fc') and ('px_width', 'px_height'). Both correlations are justified since there are good chances that if the front camera of a phone is good, the back camera will also be good.
- If a mobile phone supports 4G, it has to be compatible with 3G as well, because 4G is the latest generation that came after 3G. Thus, a phone with a 4G feature should support 3G as well.
- Battery_power also has a positive correlation with the price range. Generally, mobile phones with high prices come with good battery power.
- sc_h and sc_w are positively correlated.

Machine learning algorithms

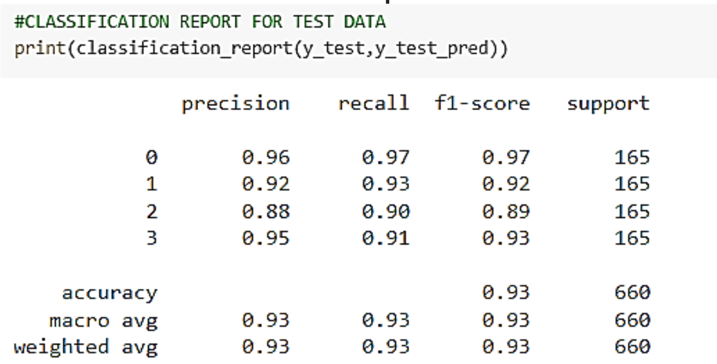
1. KNN Classifier:-

K Nearest Neighbor algorithm falls under the Supervised Learning category and is used for classification (most commonly) and regression. It is a versatile algorithm also used for imputing missing values and resampling datasets. As the name (K Nearest Neighbor) suggests it considers K Nearest Neighbors (Data points) to predict the class or continuous value for the new Datapoint.



```
#CLASSIFICATION REPORT FOR TRAIN DATA
print(classification_report(y_train,y_train_pred))
```

```
              precision    recall  f1-score   support

           0       0.98      0.97      0.97       335
           1       0.93      0.96      0.94       335
           2       0.92      0.93      0.93       335
           3       0.97      0.95      0.96       335

    accuracy                           0.95      1340
   macro avg       0.95      0.95      0.95      1340
weighted avg       0.95      0.95      0.95      1340
```



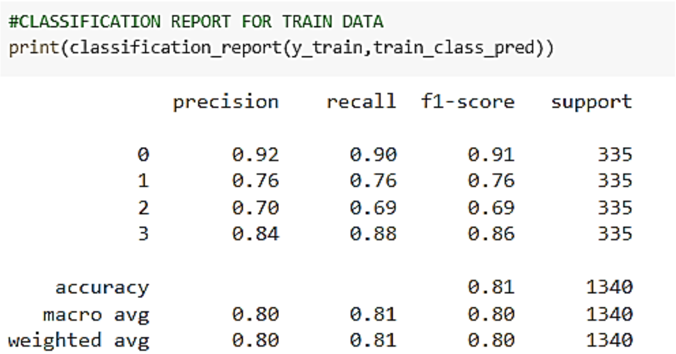
```
#CLASSIFICATION REPORT FOR TEST DATA
print(classification_report(y_test,y_test_pred))
```

```
              precision    recall  f1-score   support

           0       0.96      0.97      0.97       165
           1       0.92      0.93      0.92       165
           2       0.88      0.90      0.89       165
           3       0.95      0.91      0.93       165

    accuracy                           0.93       660
   macro avg       0.93      0.93      0.93       660
weighted avg       0.93      0.93      0.93       660
```

We achieved 95% accuracy by implementing the KNN algorithm. This model performs well on data.

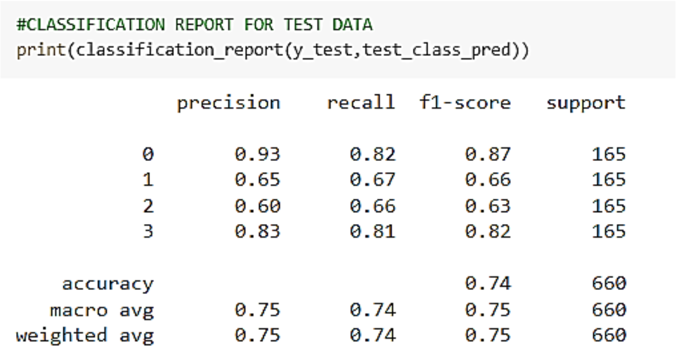
2.Logistic Regression

Logistic regression is a statistical method that is used for building machine learning models where the dependent variable is dichotomous: i.e. binary. Logistic regression is used to describe data and the relationship between one dependent variable and one or more independent variables. The independent variables can be nominal, ordinal, or of interval type.



```
#CLASSIFICATION REPORT FOR TRAIN DATA
print(classification_report(y_train,train_class_pred))
```

```
              precision    recall  f1-score   support

           0       0.92      0.90      0.91       335
           1       0.76      0.76      0.76       335
           2       0.70      0.69      0.69       335
           3       0.84      0.88      0.86       335

    accuracy                           0.81      1340
   macro avg       0.80      0.81      0.80      1340
weighted avg       0.80      0.81      0.80      1340
```



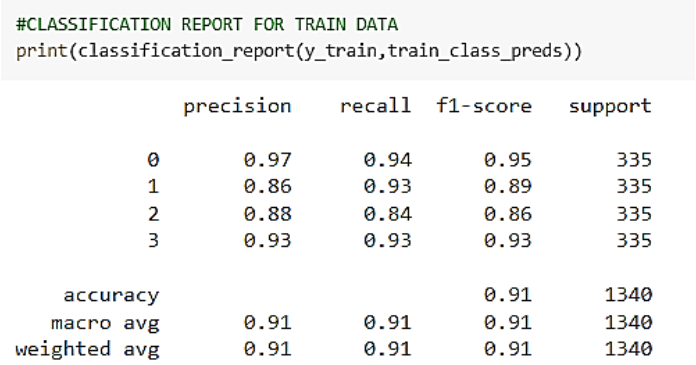
```
#CLASSIFICATION REPORT FOR TEST DATA
print(classification_report(y_test,test_class_pred))
```

```
              precision    recall  f1-score   support

           0       0.93      0.82      0.87       165
           1       0.65      0.67      0.66       165
           2       0.60      0.66      0.63       165
           3       0.83      0.81      0.82       165

    accuracy                           0.74       660
   macro avg       0.75      0.74      0.75       660
weighted avg       0.75      0.74      0.75       660
```

- Logistic regression has given an accuracy of 80%.

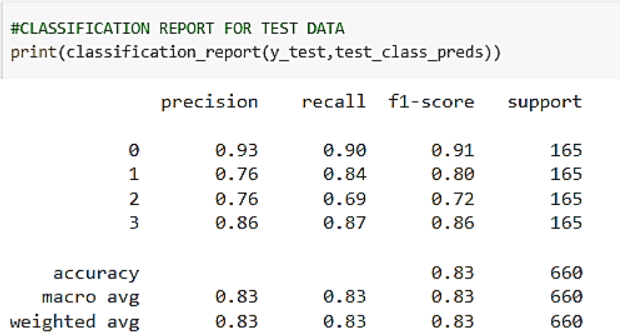
3.Decision Tree Classifier

Decision Tree is a Supervised learning technique that can be used for both classification and regression problems, but it is mostly preferred for solving classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules, and each leaf node represents the outcome.



```
#CLASSIFICATION REPORT FOR TRAIN DATA
print(classification_report(y_train,train_class_preds))
```

```
              precision    recall  f1-score   support

           0       0.97      0.94      0.95       335
           1       0.86      0.93      0.89       335
           2       0.88      0.84      0.86       335
           3       0.93      0.93      0.93       335

    accuracy                           0.91      1340
   macro avg       0.91      0.91      0.91      1340
weighted avg       0.91      0.91      0.91      1340
```



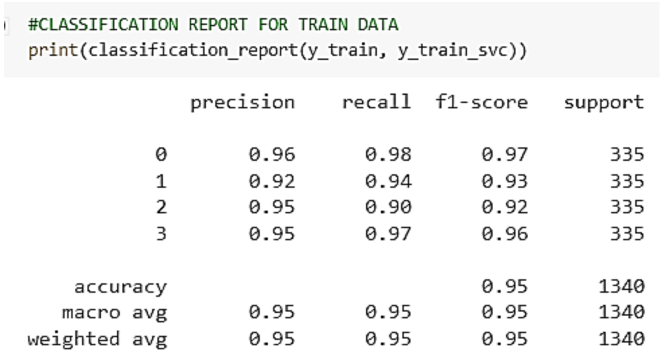
```
#CLASSIFICATION REPORT FOR TEST DATA
print(classification_report(y_test,test_class_preds))
```

```
              precision    recall  f1-score   support

           0       0.93      0.90      0.91       165
           1       0.76      0.84      0.80       165
           2       0.76      0.69      0.72       165
           3       0.86      0.87      0.86       165

    accuracy                           0.83       660
   macro avg       0.83      0.83      0.83       660
weighted avg       0.83      0.83      0.83       660
```

- With the decision tree algorithm, we got 91% accuracy, which is good, and the model is performing well on the data.

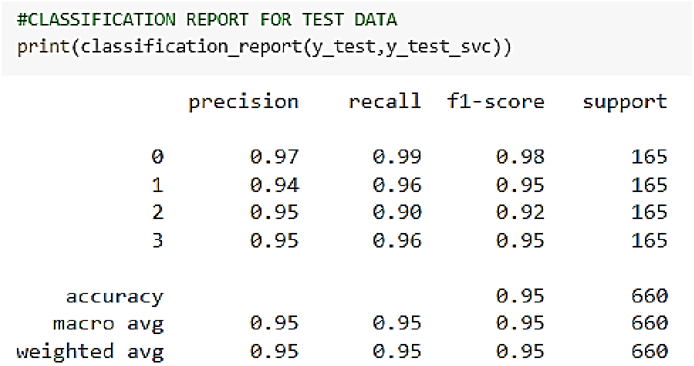
4.Support Vector Machine(SVM)

- Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.



```
#CLASSIFICATION REPORT FOR TRAIN DATA
print(classification_report(y_train, y_train_svc))
```

```
              precision    recall  f1-score   support

           0       0.96      0.98      0.97       335
           1       0.92      0.94      0.93       335
           2       0.95      0.90      0.92       335
           3       0.95      0.97      0.96       335

    accuracy                           0.95      1340
   macro avg       0.95      0.95      0.95      1340
weighted avg       0.95      0.95      0.95      1340
```



```
#CLASSIFICATION REPORT FOR TEST DATA
print(classification_report(y_test,y_test_svc))
```

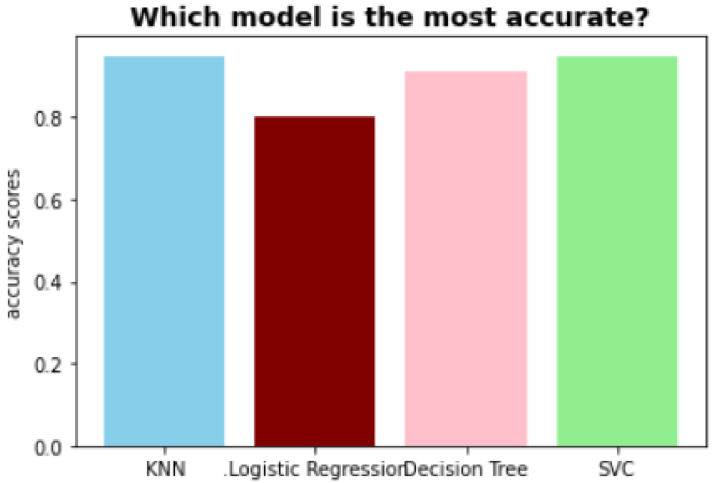
```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       165
           1       0.94      0.96      0.95       165
           2       0.95      0.90      0.92       165
           3       0.95      0.96      0.95       165

    accuracy                           0.95       660
   macro avg       0.95      0.95      0.95       660
weighted avg       0.95      0.95      0.95       660
```

- The SVM algorithm has given 95% accuracy, which is similar to the KNN algorithm. As a result, SVM performs well on the data set.

Conclusion

- We started with data understanding, data wrangling, and basic EDA, where we found the relationships, trends between price range and other independent variables.
- Mobiles with battery power greater than 1300 mAh have a very high cost, and mobiles with battery power between 1200 and 1300 mAh fall into the medium and high cost categories.
- The 75% of the devices has feature called three_g.
- In the analysis of the categorical features like blue (bluetooth), dual sim, touch screen, and 4G, we saw that 50% of the devices have these features and 50% don't.
- We can tell from the analysis that when ram is high, the price will be higher. As the higher the ram, the higher the price.
- Our target variable is well balanced. There is no class imbalance seen.
- RAM, battery power, pixels played more significant role in deciding the price range of mobile phone
- The "price range" of the given dataset has an equal distribution of the total number of phones in each of the price ranges with 500 numbers.

- During multivariate analysis, in the correlation heatmap, we get to see that "ram" is highly correlated with "price range," So that "ram" has a high impact on price prediction.



- KNN, logistic regression, decision tree, and support vector machine were implemented, and all modules are performing well on the data.
- As a result, we can tell that support vector machine and KNN are performing well on data compared to other models with 0.95 accuracy.

THANK YOU!