Assignment Of Introduction To AI & ML

Name : Tarbundiya Prashant & Vatsal Vadgama

Roll No : 23BCE347 & 23BCE35

Q -1 :- Select a source from KAGGLE/GitHub/similar repository, and identify the Data Domain and Characteristics of the selected Dataset. Also, write a report with the following detail

a) Selection of data domain
=>Smart Phones

b) find the data characteristics (null, not null, unique values)

- Brand
- Modal Name
- Original Price & Discount price
- Colour
- Rating
- Storage
- Processor
- Review
- Rear & Front Camera
- Display size
- Battery type & capacity

d) Find the output/class label of the data set

- For this dataset the class label was “discounted Price”

e) Define the selection of fields (Data types)

- Categorical Data Types:
  Brand: This represents the brand of the smartphone (e.g., VIVO, APPLE, XIAOMI, REALME).
  Model: The model name of the smartphone.
  Colour: The color of the smartphone.
  Processor: The processor type of the smartphone.
- Numerical Data Types:
  Original Price: The original price of the smartphone (numeric).
  Discounted Price: The discounted price of the smartphone (numeric).
  Ratings: The average rating of the smartphone (numeric).
  Rating Count: The count of ratings given to the smartphone (numeric).
  Reviews: The number of reviews for the smartphone (numeric).
  Memory: The memory capacity of the smartphone (numeric).
  Storage: The storage capacity of the smartphone (numeric).
  Display Size: The size of the smartphone's display (numeric).
  Battery Capacity: The capacity of the smartphone's battery (numeric).

- Text Data Types:
  Rear Camera: Description of the rear camera setup.
  Front Camera: Description of the front camera setup.
  Battery Type: Type of battery used in the smartphone.
- Mixed Data Types:
  Battery Type: It could be considered categorical if we have a limited set of battery types (e.g., Lithium, Lithium Polymer), or it could be considered as text data if it includes other types.

Q-2 :- Load the dataset, drop all the null records and replace the NA values in the numerical column with the mean value of the field as per the class label and categorical columns with the mode value of the field as per the class label.

```
#Here we import the pandas and the numpy for working with the data set
import numpy as np
import pandas as pd

#here we read the csv file and import data from this
data =
pd.read_csv(r"C:\Users\prash\AppData\Local\Programs\Python\Python312\Scripts\flipkart_
smartphones.csv",header='infer')
print(data) #Dispaly the cintent of dataset
print(data.info()) #This is use for the viewing the content on the dataset
```

```
       brand             model               colour  original_price  \
0       VIVO      VIVO T1 44W            Starry Sky           19990
1     APPLE  APPLE IPHONE 11                 White           48900
2       VIVO      VIVO T1 44W       Midnight Galaxy           20990
3     XIAOMI       POCO M4 5G           Power Black           15999
4     XIAOMI         REDMI 10       Caribbean Green           14999
..       ...               ...                 ...             ...
831  REALME  REALME GT NEO 2              NEO Blue           38999
832  REALME  REALME GT NEO 2             NEO Black           38999
833  REALME  REALME GT NEO 2             NEO Black           34999
834  REALME   REALME X50 PRO              Rust Red           17999
835  XIAOMI      POCO M2 PRO   Two Shades of Black           17999

      discounted_price  ratings  rating_count  reviews  memory  storage  \
0                14499      4.5         87331     6044     4.0    128.0
1                47199      4.6        184191    10818     NaN    128.0
2                15999      4.4         51365     3750     6.0    128.0
3                11999      4.2         53448     4185     4.0     64.0
4                 9299      4.3        187787    12084     4.0     64.0
..                 ...      ...           ...      ...     ...      ...
831              35999      4.4          4430      628     NaN    256.0
832              35999      4.4          4430      628     NaN    256.0
833              31999      4.4         18194     2478     8.0    128.0
834              41999      4.4          8689     1232     8.0    128.0
835              17999      4.4        350744    33202     6.0     64.0

                     processor               rear_camera      front_camera  \
0     Qualcomm Snapdragon 680          50MP + 2MP + 2MP              16MP
1                A Bionic Chip              12MP + 12MP              12MP
2     Qualcomm Snapdragon 680          50MP + 2MP + 2MP              16MP
3      Mediatek Dimensity 700                50MP + 2MP               8MP
4     Qualcomm Snapdragon 680                50MP + 2MP               5MP
..                        ...                       ...              ...
831   Qualcomm Snapdragon 870          64MP + 8MP + 2MP              16MP
832   Qualcomm Snapdragon 870          64MP + 8MP + 2MP              16MP
833   Qualcomm Snapdragon 870          64MP + 8MP + 2MP              16MP
834                       NaN  64MP + 12MP + 8MP + 2MP    32MP + 8MP Dual
835  Qualcomm Snapdragon 720G   48MP + 8MP + 5MP + 2MP              16MP

     display_size  battery_capacity     battery_type
0           16.36           5000.0          Lithium
1           15.49              NaN              NaN
2           16.36           5000.0          Lithium
3           16.71           5000.0  Lithium Polymer
4           17.02           6000.0  Lithium Polymer
..            ...              ...              ...
831         16.81           5000.0              NaN
832         16.81           5000.0              NaN
833         16.81           5000.0              NaN
834         16.36           4200.0              NaN
835         16.94           5000.0          Lithium

[836 rows x 16 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 836 entries, 0 to 835
Data columns (total 16 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   brand             836 non-null    object
 1   model             836 non-null    object
 2   colour            831 non-null    object
 3   original_price    836 non-null    int64
 4   discounted_price  836 non-null    int64
 5   ratings           836 non-null    float64
 6   rating_count      836 non-null    int64
 7   reviews           836 non-null    int64
```

```
 8   memory           757 non-null    float64
 9   storage          820 non-null    float64
 10  processor        698 non-null    object
 11  rear_camera      836 non-null    object
 12  front_camera     707 non-null    object
 13  display_size     836 non-null    float64
 14  battery_capacity  778 non-null    float64
 15  battery_type     477 non-null    object
dtypes: float64(5), int64(4), object(7)
memory usage: 104.6+ KB
None
```

```
#making different copies of our csv file using copy to perform various functions on it and keeping original csv as it is.
data1 = data.copy()
data2 = data.copy()
data3 = data.copy()
```

In [3]:

```
print('The number of rows and columns in dataset are ',data.shape)

#using dropna function to delete the coloumns having null entries
data2 = data2.dropna(axis=1)
data3 = data3.dropna(axis=0)
print('The number of rows after NA values in coloumns are ',data2.shape)
print('The number of rows after NA values in rows are ',data3.shape)
```

Output :-

```
The number of rows and columns in dataset are  (836, 16)
The number of rows after NA values in coloumns are  (836, 9)
The number of rows after NA values in rows are  (456, 16)
```

In [4]:

```
#Put the value of mean and mode according to the column
na_color = data1.loc[:,"colour"].mode()[0]
print("The mode of colour column",na_color)
data1.loc[data1["colour"].isna(),"colour"] = na_color

na_memory = np.mean(data1.loc[~data1["memory"].isna(),"memory"])
print("The mean of memory column",na_memory)
data1.loc[data1["memory"].isna(),"memory"] = na_memory

na_storage = np.mean(data1.loc[~data1["storage"].isna(),"storage"])
print("The mean of storage column",na_storage)
data1.loc[data1["storage"].isna(),"storage"] = na_storage

na_processor = data1.loc[:,"processor"].mode()[0]
print("The mode of processor column",na_processor)
data1.loc[data1["processor"].isna(),"processor"] = na_processor

na_front_camera = data1.loc[:,"front_camera"].mode()[0]
print("The mode of front camera column",na_front_camera)
data1.loc[data1["front_camera"].isna(),"front_camera"] = na_front_camera

na_battery_capacity = np.mean(data1.loc[~data1["battery_capacity"].isna(),"battery_capacity"])
```

```
print("The mean of battery capacity column",na_battery_capacity)
data1.loc[data1["battery_capacity"].isna(),"battery_capacity"] = na_battery_capacity

na_battery_type = data1.loc[:,"battery_type"].mode()[0]
print("The mode of battery type column",na_battery_type)
data1.loc[data1["battery_type"].isna(),"battery_type"] = na_battery_type

print(data1.info())
print(data1)
```

```
The mode of colour column Black
The mean of memory column 5.225891677675033
The mean of storage column 108.45853658536585
The mode of processor column Qualcomm Snapdragon 680
The mode of front camera column  16MP
The mean of battery capacity column 4950.9318766066835
The mode of battery type column Lithium
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 836 entries, 0 to 835
Data columns (total 16 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   brand             836 non-null    object
 1   model             836 non-null    object
 2   colour            836 non-null    object
 3   original_price    836 non-null    int64
 4   discounted_price  836 non-null    int64
 5   ratings           836 non-null    float64
 6   rating_count      836 non-null    int64
 7   reviews           836 non-null    int64
 8   memory            836 non-null    float64
 9   storage           836 non-null    float64
 10  processor         836 non-null    object
 11  rear_camera       836 non-null    object
 12  front_camera      836 non-null    object
 13  display_size      836 non-null    float64
 14  battery_capacity  836 non-null    float64
 15  battery_type      836 non-null    object
dtypes: float64(5), int64(4), object(7)
memory usage: 104.6+ KB
None
```

```
      brand             model              colour  original_price  \
0      VIVO      VIVO T1 44W            Starry Sky           19990
1     APPLE  APPLE IPHONE 11                 White           48900
2      VIVO      VIVO T1 44W       Midnight Galaxy           20990
3    XIAOMI       POCO M4 5G           Power Black           15999
4    XIAOMI         REDMI 10       Caribbean Green           14999
..      ...               ...                 ...             ...
831  REALME  REALME GT NEO 2              NEO Blue           38999
832  REALME  REALME GT NEO 2             NEO Black           38999
833  REALME  REALME GT NEO 2             NEO Black           34999
834  REALME   REALME X50 PRO              Rust Red           17999
835  XIAOMI      POCO M2 PRO   Two Shades of Black           17999

     discounted_price  ratings  rating_count  reviews    memory  storage  \
0               14499      4.5         87331     6044  4.000000    128.0
1               47199      4.6        184191    10818  5.225892    128.0
2               15999      4.4         51365     3750  6.000000    128.0
3               11999      4.2         53448     4185  4.000000     64.0
4                9299      4.3        187787    12084  4.000000     64.0
..                ...      ...           ...      ...       ...      ...
831             35999      4.4          4430      628  5.225892    256.0
832             35999      4.4          4430      628  5.225892    256.0
833             31999      4.4         18194     2478  8.000000    128.0
834             41999      4.4          8689     1232  8.000000    128.0
835             17999      4.4        350744    33202  6.000000     64.0

                  processor              rear_camera      front_camera  \
0     Qualcomm Snapdragon 680         50MP + 2MP + 2MP              16MP
1              A Bionic Chip             12MP + 12MP              12MP
2     Qualcomm Snapdragon 680         50MP + 2MP + 2MP              16MP
3      Mediatek Dimensity 700               50MP + 2MP               8MP
4     Qualcomm Snapdragon 680               50MP + 2MP               5MP
..                        ...                      ...               ...
831   Qualcomm Snapdragon 870         64MP + 8MP + 2MP              16MP
832   Qualcomm Snapdragon 870         64MP + 8MP + 2MP              16MP
833   Qualcomm Snapdragon 870         64MP + 8MP + 2MP              16MP
834   Qualcomm Snapdragon 680  64MP + 12MP + 8MP + 2MP    32MP + 8MP Dual
835  Qualcomm Snapdragon 720G   48MP + 8MP + 5MP + 2MP              16MP

     display_size  battery_capacity     battery_type
0           16.36       5000.000000          Lithium
1           15.49       4950.931877          Lithium
2           16.36       5000.000000          Lithium
3           16.71       5000.000000  Lithium Polymer
4           17.02       6000.000000  Lithium Polymer
..            ...               ...              ...
831         16.81       5000.000000          Lithium
832         16.81       5000.000000          Lithium
833         16.81       5000.000000          Lithium
834         16.36       4200.000000          Lithium
835         16.94       5000.000000          Lithium

[836 rows x 16 columns]
```

Q-3 :- Perform statistical analysis on the selected dataset (count, sum, range, min, max, mean, median, mode, variance and Standard deviation).

#Here describe() use for the knowing the statistial value of each numerical column like ,count,mean,standard deviaton,minimum,maximum etc...

```
print("Stastical value of each column")
print(data1.describe())

#For printing the statistics value i used the for loop for the short code
import statistics as st
column = ['original_price', 'discounted_price', 'ratings', 'rating_count', 'reviews', 'memory',
'storage', 'display_size', 'battery_capacity']
for i in column:
    mode = st.mode(data1[i])
    sum = np.sum(data1[i])
    median = np.median(data1[i])
    variance = np.var(data[i])
    range = np.max(data[i]) - np.min(data[i])
    print("Statistial Value For Column :",i)
    print("Mode :", mode)
    print("Sum :", sum)
    print("Median :", median)
    print("Variance :", variance)
    print("Range :", range)
```

```
Stastical value of each column
       original_price  discounted_price     ratings  rating_count  \
count      836.000000        836.000000  836.000000  8.360000e+02
mean     26112.848086      21532.468900    4.272488  4.169612e+04
std      23781.117479      21363.273113    0.284083  9.084175e+04
min       4899.000000       3599.000000    0.000000  0.000000e+00
25%      13999.000000       9999.000000    4.200000  9.227500e+02
50%      17999.000000      14999.000000    4.300000  5.917500e+03
75%      25999.000000      22992.250000    4.400000  3.995500e+04
max     159900.000000     152999.000000    4.700000  1.171704e+06

             reviews      memory     storage  display_size  battery_capacity
count     836.000000  836.000000  836.000000    836.000000        836.000000
mean     3144.740431    5.225892  108.458537     16.525179       4950.931877
std      7524.617982    1.910066   68.508451      0.682189        484.308321
min         0.000000    1.000000    8.000000     10.160000       1600.000000
25%        78.000000    4.000000   64.000000     16.510000       5000.000000
50%       514.000000    5.225892  128.000000     16.660000       5000.000000
75%      3314.000000    6.000000  128.000000     16.760000       5000.000000
max    122453.000000    8.000000  512.000000     17.780000       7000.000000
Statistial Value For Column : original_price
Mode : 17999
Sum : 21830341
Median : 17999.0
Variance : 564865063.4661567
Range : 155001
Statistial Value For Column : discounted_price
Mode : 8999
Sum : 18001144
Median : 14999.0
Variance : 455843517.722717
Range : 149400
Statistial Value For Column : ratings
Mode : 4.3
Sum : 3571.8
Median : 4.3
Variance : 0.08060672832581671
Range : 4.7
Statistial Value For Column : rating_count
Mode : 4110
Sum : 34857957
Median : 5917.5
Variance : 8242353080.012915
Range : 1171704
Statistial Value For Column : reviews
Mode : 2
Sum : 2629003
Median : 514.0
Variance : 56552148.64434624
Range : 122453
Statistial Value For Column : memory
Mode : 4.0
Sum : 4368.845442536328
Median : 5.225891677675033
```

```
Variance : 4.024270175848836
Range : 7.0
Statistial Value For Column : storage
Mode : 128.0
Sum : 90671.33658536585
Median : 128.0
Variance : 4779.262914931588
Range : 504.0
Statistial Value For Column : display_size
Mode : 16.76
Sum : 13815.05
Median : 16.66
Variance : 0.46482472857191925
Range : 7.620000000000001
Statistial Value For Column : battery_capacity
Mode : 5000.0
Sum : 4138979.048843188
Median : 5000.0
Variance : 251739.1380327251
Range : 5400.0
```

Q-3 :- Display all the unique value counts and unique values of all the columns of the dataset

```
#hare we print the unique value and length of each column
unique_brand = data1.loc[:,"brand"].unique()
print("Unique Value For Brand Column :",unique_brand)
print("Length Of Unique Value :",len(unique_brand))

unique_model = data1.loc[:,"model"].unique()
print("Unique Value For Model Column :",unique_model)
print("Length Of Unique Value :",len(unique_model))

unique_original_price = data1.loc[:,"original_price"].unique()
print("Unique Value For Original Price Column :",unique_original_price)
print("Length Of Unique Value :",len(unique_original_price))

unique_discounted_price = data1.loc[:,"discounted_price"].unique()
```

```
print("Unique Value For Discount Price Column :",unique_discounted_price)
print("Length Of Unique Value :",len(unique_discounted_price))

unique_ratings = data1.loc[:,"ratings"].unique()
print("Unique Value For Ratings Column :",unique_ratings)
print("Length Of Unique Value :",len(unique_ratings))

unique_rating_count = data1.loc[:,"rating_count"].unique()
print("Unique Value For Rating Count Column :",unique_rating_count)
print("Length Of Unique Value :",len(unique_rating_count))

unique_reviews = data1.loc[:,"reviews"].unique()
print("Unique Value For Review Column :",unique_reviews)
print("Length Of Unique Value :",len(unique_reviews))

unique_memory = data1.loc[:,"memory"].unique()
print("Unique Value For Memory Column :",unique_memory)
print("Length Of Unique Value :",len(unique_memory))

unique_storage = data1.loc[:,"storage"].unique()
print("Unique Value For Storage Column :",unique_storage)
print("Length Of Unique Value :",len(unique_storage))

unique_processor = data1.loc[:,"processor"].unique()
print("Unique Value For Processor Column :",unique_processor)
print("Length Of Unique Value :",len(unique_processor))

unique_rear_camera = data1.loc[:,"rear_camera"].unique()
print("Unique Value For Rear Camera Column :",unique_rear_camera)
print("Length Of Unique Value :",len(unique_rear_camera))
```

```
unique_front_camera = data1.loc[:,"front_camera"].unique()

print("Unique Value For Front Camera Column :",unique_front_camera)

print("Length Of Unique Value :",len(unique_front_camera))


unique_display_size = data1.loc[:,"display_size"].unique()

print("Unique Value For Display Size Column :",unique_display_size)

print("Length Of Unique Value :",len(unique_display_size))


unique_battery_capacity = data1.loc[:,"battery_capacity"].unique()

print("Unique Value For Battery Capacity Column :",unique_battery_capacity)

print("Length Of Unique Value :",len(unique_battery_capacity))


unique_battery_type = data1.loc[:,"battery_type"].unique()

print("Unique Value For Battery Type Column :",unique_battery_type)

print("Length Of Unique Value :",len(unique_battery_type))
```

```
Unique Value For Brand Column : ['VIVO' 'APPLE' 'XIAOMI' 'INFINIX' 'MOTOROLA' 'REALME' 'SAMS
UNG' 'OPPO'
 'ONEPLUS' 'GOOGLE' 'NOTHING' 'I KALL' 'IQOO' 'MICROMAX' 'ITEL' 'TECNO'
 'NOKIA' 'LAVA' 'MARQ']
Length Of Unique Value : 19
Unique Value For Model Column : ['VIVO T1 44W ' 'APPLE IPHONE 11 ' 'POCO M4 5G ' 'REDMI 10 '
'POCO C31 '
 'INFINIX HOT 20 PLAY ' 'REDMI 9I SPORT ' 'MOTOROLA G52 '
 'MOTOROLA G62 5G ' 'REALME C30 ' 'POCO M4 PRO ' 'MOTOROLA G32 '
 'SAMSUNG GALAXY F13 ' 'APPLE IPHONE 14 ' 'POCO M4 PRO 5G '
 'MOTOROLA G31 ' 'SAMSUNG GALAXY F23 5G ' 'INFINIX NOTE 12I '
 'INFINIX HOT 12 ' 'INFINIX SMART 6 HD ' 'MOTOROLA E40 ' 'MOTOROLA E13 '
 'REALME C30 - LOCKED WITH AIRTEL PREPAID ' 'POCO X4 PRO 5G '
 'SAMSUNG GALAXY F04 ' 'OPPO A17K ' 'REALME 10 PRO+ 5G ' 'REDMI NOTE 10S '
 'MOTOROLA G82 5G ' 'POCO C50 ' 'VIVO T1X ' 'REDMI 10A '
 'REDMI NOTE 11 SE ' 'ONEPLUS NORD CE 2 LITE 5G ' 'REALME C33 '
 'GOOGLE PIXEL 6A ' 'REALME NARZO 50 ' 'OPPO F19 PRO+ 5G ' 'OPPO K10 5G '
 'MOTOROLA G22 ' 'INFINIX HOT 20 5G ' 'REALME 10 PRO 5G '
 'APPLE IPHONE 12 ' 'REALME C35 ' 'REALME C30S ' 'REDMI 10 PRIME '
 'REDMI 9I ' 'REDMI NOTE 12 PRO 5G ' 'MOTOROLA G42 ' 'REDMI 9 ACTIV '
 'MOTOROLA E32 ' 'SAMSUNG GALAXY M13 ' 'REDMI NOTE 11 ' 'OPPO F21 PRO '
 'INFINIX HOT 12 PRO ' 'APPLE IPHONE 13 ' 'REALME 9I 5G '
 'INFINIX NOTE 12 5G ' 'SAMSUNG GALAXY A03 CORE ' 'OPPO A17 '
 'VIVO T1 5G ' 'REALME 10 ' 'REDMI A1 ' 'MOTOROLA G72 ' 'POCO X5 PRO 5G '
 'MOTOROLA G82 5G' 'REALME C31 ' 'OPPO RENO8T 5G ' 'OPPO F19 ' 'VIVO Y16 '
 'SAMSUNG GALAXY M33 5G ' 'SAMSUNG GALAXY A14 5G ' 'VIVO Y01 ' 'VIVO Y21 '
 'VIVO Y02 ' 'INFINIX NOTE 12 PRO 5G ' 'MOTOROLA EDGE 30 '
 'NOTHING PHONE ' 'INFINIX SMART 6 ' 'SAMSUNG GALAXY M13 5G ' 'OPPO K10 '
 'VIVO T1 PRO 5G ' 'REALME C11 2021 ' 'MOTOROLA EDGE 20 5G ' 'VIVO Y35 '
```

'SAMSUNG GALAXY S21 FE 5G ' 'INFINIX ZERO 5G 2023 ' 'I KALL Z19PRO '
'APPLE IPHONE 14 PLUS ' 'VIVO Y15S ' 'IQOO Z6 44W ' 'MICROMAX IN 2C '
'INFINIX NOTE 12 PRO ' 'SAMSUNG GALAXY A03 ' 'REDMI 10A SPORT '
'ITEL VISION3 ' 'REALME NARZO 50A PRIME ' 'OPPO A77S ' 'INFINIX NOTE 12 '
'TECNO POVA 3 ' 'INFINIX SMART 6 PLUS ' 'REDMI NOTE 10 PRO '
'SAMSUNG GALAXY M04 ' 'SAMSUNG GALAXY A23 5G ' 'REALME C21 '
'REDMI NOTE 11S ' 'INFINIX ZERO 20 ' 'IQOO Z6 LITE 5G ' 'VIVO Y33S '
'REDMI 11 PRIME 5G ' 'INFINIX HOT 12 PLAY ' 'VIVO Y75 '
'SAMSUNG GALAXY M32 PRIME EDITION ' 'REDMI NOTE 12 PRO+ 5G '
'VIVO Y100 5G ' 'REDMI NOTE 12 5G ' 'REDMI 10 POWER ' 'OPPO F19S '
'MOTOROLA G60 ' 'REDMI 9A SPORT ' 'TECNO SPARK 9 ' 'POCO M5 '
'TECNO CAMON 19 ' 'INFINIX ZERO 5G 2023 TURBO '
'MOTOROLA EDGE 30 FUSION ' 'SAMSUNG GALAXY F22 ' 'REALME C20 '
'TECNO POP 6 PRO ' 'GOOGLE PIXEL 7 ' 'REALME 9 PRO 5G '
'TECNO POP 5 LTE ' 'VIVO Y33T ' 'REALME NARZO 50I ' 'NOKIA G11 PLUS '
'VIVO Y56 5G ' 'OPPO F21S PRO 5G ' 'REALME 9 ' 'IQOO NEO 6 5G '
'VIVO Y22 ' 'REDMI K50I 5G ' 'MOTOROLA G40 FUSION ' 'POCO C3 '
'ITEL VISION 3 ' 'SAMSUNG GALAXY S23 5G ' 'INFINIX HOT 11 ' 'POCO F4 5G '
'VIVO Y21T ' 'REDMI NOTE 10T 5G ' 'SAMSUNG GALAXY Z FLIP3 5G '
'MOTOROLA G51 5G ' 'OPPO A76 ' 'REALME NARZO 50 PRO 5G '
'XIAOMI 11LITE NE ' 'ITEL A23S ' 'INFINIX HOT 11S ' 'REDMI GO '
'SAMSUNG GALAXY A52 ' 'REDMI NOTE 9 ' 'REDMI 7A ' 'REALME GT 2 '
'SAMSUNG GALAXY S22 5G ' 'OPPO A12 ' 'OPPO A55 ' 'SAMSUNG GALAXY A13 '
'REALME 8I ' 'REDMI NOTE 11 PRO PLUS 5G ' 'OPPO A15S ' 'MI 11 LITE '
'VIVO V23E 5G ' 'TECNO SPARK 8 ' 'SAMSUNG GALAXY A73 5G '
'SAMSUNG GALAXY A04E ' 'VIVO V21 5G ' 'VIVO Y53S ' 'OPPO F21S PRO '
'REALME NARZO 50 5G ' 'REALME 9 5G ' 'SAMSUNG GALAXY A53 ' 'REALME 8 '
'REALME C21Y ' 'SAMSUNG GALAXY A23 ' 'INFINIX SMART 5A '
'NOKIA C01 PLUS ' 'REALME C25_Y ' 'ONEPLUS 10R 5G ' 'ITEL A49 '
'REDMI NOTE 11T 5G ' 'IQOO Z6 PRO 5G ' 'REALME 9 5G SE '
'TECNO SPARK 8C ' 'APPLE IPHONE 14 PRO ' 'SAMSUNG GALAXY F42 5G '
'I KALL Z5 ' 'REDMI A1+ ' 'POCO F3 GT 5G ' 'MOTOROLA EDGE 30 ULTRA '
'OPPO A77 ' 'ITEL VISION1 ' 'REALME GT NEO 3 ' 'REALME NARZO 30 '
'LAVA BLAZE PRO ' 'TECNO SPARK GO 2022 ' 'APPLE IPHONE 14 PRO MAX '
'ITEL A27 ' 'OPPO A16 ' 'LAVA Z2 MAX ' 'VIVO Y21A ' 'REALME 9 PRO+ 5G '
'INFINIX NOTE 12 TURBO ' 'OPPO A57 ' 'SAMSUNG GALAXY M53 5G '
'REDMI 11 PRIME ' 'MOTOROLA G30 ' 'XIAOMI 12 PRO 5G ' 'REALME 8 5G '
'IQOO Z6 5G ' 'SAMSUNG GALAXY M11 ' 'REDMI 9A ' 'LAVA Z2 '
'REDMI NOTE 10 PRO MAX ' 'SAMSUNG GALAXY A04S ' 'TECNO SPARK 8T '
'REALME NARZO 30 PRO 5G ' 'REDMI NOTE 11 PRO ' 'OPPO A16E '
'VIVO V23 PRO 5G ' 'REALME C15 ' 'VIVO V25 5G ' 'MOTOROLA E7 POWER '
'VIVO Y21G ' 'TECNO POP 5 PRO ' 'SAMSUNG GALAXY A33 ' 'OPPO A54 '
'MOTO G71 5G ' 'APPLE IPHONE 12 MINI ' 'SAMSUNG GALAXY F12 '
'TECNO POVA NEO ' 'REALME C25S ' 'TECNO SPARK 8 PRO ' 'REALME 9I '
'LAVA Z2C ' 'INFINIX SMART HD 2021 ' 'LAVA BEU '
'MARQ BY FLIPKART M3 SMART ' 'APPLE IPHONE 13 PRO ' 'INFINIX NOTE 11S '
'INFINIX NOTE 11 ' 'REALME 6 PRO ' 'SAMSUNG GALAXY M32 ' 'VIVO Y15C '
'OPPO A31 ' 'POCO M3 PRO 5G ' 'SAMSUNG GALAXY S20 FE 5G '
'TECNO SPARK 8P ' 'OPPO A74 5G ' 'INFINIX HOT 11 2022 ' 'POCO F1 '
'OPPO A 78 5G ' 'MOTOROLA EDGE 20 FUSION 5G ' 'REALME NARZO 30 5G '
'VIVO Y01A ' 'VIVO X80 ' 'REDMI K50I ' 'TECNO SPARK GO 2021 '
'REALME C11 ' 'REDMI NOTE 7S ' 'INFINIX ZERO 5G '
'SAMSUNG GALAXY S23 PLUS 5G ' 'SAMSUNG GALAXY F41 ' 'OPPO A78 5G '
'REALME GT NEO 3T ' 'I KALL 401 PRO DARK BLUE ' 'OPPO F21SPRO '
'POCO M2 PRO ' 'MICROMAX BHARAT 2 PLUS ' 'REDMI 6 ' 'REALME 3I '
'POCO M2 RELOADED ' 'MOTO C PLUS ' 'SAMSUNG GALAXY S23 ULTRA 5G '
'OPPO RENO8 PRO 5G ' 'SAMSUNG GALAXY A04 ' 'POCO X2 '
'OPPO A74 5G BLACK ' 'I KALL Z20 PRO ' 'MI 11X PRO 5G '
'REDMI NOTE 9 PRO ' 'OPPO A96 ' 'REDMI 8 ' 'IQOO 9T 5G '
'REALME X50 PRO ' 'NOKIA C21 PLUS ' 'SAMSUNG GALAXY S22 PLUS 5G '

'XIAOMI 11I 5G ' 'ITEL A23 PRO ' 'ITEL A25 ' 'REDMI NOTE 5 PRO '
'REDMI 8A DUAL ' 'SAMSUNG GALAXY S20 FE ' 'ITEL VISION3 TURBO ' 'MI 10T '
'XIAOMI 11T PRO 5G HYPERPHONE ' 'TECNO SPARK 7 ' 'REALME GT NEO 2 ']
Length Of Unique Value : 304
Unique Value For Original Price Column : [19990 48900 20990 15999 14999 11999 23990 9999
10999 43900
17999 24999 9299 8499 16999 19999 79900 13999 23999 18999
12999 22690 8999 21999 11499 27999 25999 17990 16990 43999
29990 18990 22999 20999 10499 64900 69900 28999 38999 18499
31999 30999 42999 28990 7999 34999 74999 89900 13990 9499
8690 13499 14499 21990 25990 39999 29999 37999 99900 36999
33999 22990 30990 59900 49999 59999 6599 7500 95999 32999
21499 4899 16499 85999 10990 47490 32990 39990 5799 6799
27990 26999 9190 44999 159900 8498 11990 85666 25490 18490
69999 119900 129900 6999 8299 45999 34990 139900 15990 8449
84999 17499 8248 9850 23895 41990 17490 33990 74900 8240
15499 8948 9909 24298 7777 149900 10595 9976 74990 41999
66999 35999 9145 116999 22929 26499 9498 149999 52999 38990
109900 11389 101999 79990 65999 35899]
Length Of Unique Value : 136
Unique Value For Discount Price Column : [14499 47199 15999 11999 9299 7499 17999 8199
6499 40999
10999 16999 6749 5749 10499 11099 12999 71999 12099 11499
15499 9499 13999 9999 8499 5799 8999 7999 5299 16499
25999 17499 19999 6999 21499 8585 19985 29999 11534 11745
17990 13499 14999 18999 19922 47999 7673 56999 20499 9699
24999 6249 11044 12680 11989 20999 61999 13099 19975 6600
12499 15990 6275 6489 14990 14495 12380 16771 8751 22999
12630 27999 19990 35999 12639 13990 23999 18499 34999 8299
6299 80999 7478 81999 19998 21999 14699 16239 5999 18634
8149 10989 5899 10977 16990 10869 12490 8966 15824 19980
15900 13976 19888 13525 28499 26999 90999 12890 32999 12263
17290 16964 16965 7284 7095 7189 9990 13977 15700 51999
39999 11500 6799 6700 57099 16390 6464 16790 6870 9094
25490 30300 23290 12628 11570 7299 14180 6880 6933 6777
79999 11244 14535 69999 16490 21890 18990 4699 41999 52149
12899 15199 87999 21900 12910 20990 9449 23990 14394 15490
6911 33499 5850 7199 5699 25990 18757 10890 23800 18700
24599 8888 38999 152999 7899 10990 52499 11990 54999 110999
122999 5199 42999 9149 10260 6990 6865 6569 8749 29990
127999 12990 7799 20995 13489 17899 7285 28999 23680 13690
16994 10489 17489 9450 7746 6769 27990 17692 12700 14774
18300 57890 38990 14399 12895 17099 8699 27989 10980 55999
8140 10247 8799 10790 9798 9850 6350 10670 23579 15380
27499 5960 5499 10799 5169 10699 21990 12788 12090 28989
9949 17490 31390 11400 9239 17089 8981 24990 19595 26480
12340 7689 66999 9135 17989 94999 19428 31999 34439 33999
5175 5149 20880 3599 132999 11180 22990 23700 36999 8549
124999 11463 45999 101999 9370 37599 10285 14819 17397 14450
10749 49499 4199 4889 46990 8490 17800 50010 27699 7100
37999 47490 33500 20390]
Length Of Unique Value : 294
Unique Value For Ratings Column : [4.5 4.6 4.4 4.2 4.3 4.1 4.7 3.8 4. 3.4 3.9 3.5 3. 3.6 0. 3.3 3.7]
Length Of Unique Value : 17
Unique Value For Rating Count Column : [87331 184191 51365 53448 187787 202727 5798
8298 236239
134382 10106 4375 38925 83016 150352 27486 122678 27454
16010 24991 47939 34024 157763 26886 1003 18601 22440
70754 1351 12436 39955 73220 2792 687 5792 84168
17160 7048 1528 4238 78071 6435 39454 47110 19530

29306 13407 15538 28057 61218 29477 850 4011 5932
9577 1841 186979 169098 4730 59027 12411 577272 14132
9137 5670 2416 5042 897 2494 774 396 2568
7129 36500 190651 7639 40920 14950 2729 1727 14815
63968 18256 3424 803 4176 10081 983 11341 46544
1761 25035 1084 7862 4590 89 924 61796 1275
14580 200 1418 3694 16249 19595 8266 6616 30140
279 103802 7982 234364 8964 1670 36083 65 189
4110 4457 214 3241 3742 3895 859 3156 1070
3941 674 5416 797 211 198 263 25033 167235
471 206885 11620 42313 4596 365 71737 147339 2281
1433 54 12159 1288 253 112223 1862 51215 244
2098 46 68 715 5404 10073 133640 16255 6199
25145 773 3191 86 168 12428 2621 102798 296873
48 152 2703 1565 81684 645 758 2576 122289
6 9732 3 359 61332 24775 1010 684 1043
32941 398759 331 135 503 51 14450 16145 12034
1559 31669 1071 1876 17217 360 2808 62 1616
92 49358 167688 2150 123832 388968 5990 110 75143
748 419 329 70085 172736 5903 5645 6022 2141
919 1392 56396 93 14 17150 18983 245 31
38692 48229 1369 95767 121725 90474 831 78844 4349
46866 65488 587 5747 300 3061 2488 26850 103
1226 756 454 10434 67536 478 1734 738 62267
368 4709 1112 345 25705 21 12 76491 228
818 440 254314 769 656 807 7397 3846 301
1867 35 29755 7857 2202 55 38 36307 144
75848 106194 690 10653 21534 333 16833 26 425
28455 1161 3346 755 3786 125111 6847 83852 2134
422 827 4062 22647 125841 31257 298148 50767 26434
106 44860 66389 10938 469 1421 11981 10807 61057
1776 4421 6058 1149 29727 53487 691 95 1601
25 1129 17054 244108 0 19791 84648 832 42
741 326 826 124865 459210 422214 121931 10272 10
393417 17 2904 12124 7524 400 52 32 350744
6917 559353 80034 66978 56326 365540 43 33 83
2268 655 13 60874 1803 450 26761 973 942649
1639 8689 114 2366 1448 179 37768 4783 1096
134 1171704 57135 927 3645 496 1487 271179 4430
18194]
Length Of Unique Value : 397
Unique Value For Review Column : [6044 10818 3750 4185 12084 11672 483 486 13755
8481
1217 442 2527 4466 13531 2666 7131 3250 686 1816
4182 3859 12636 2123 125 1333 1267 6695 155 1014
3495 7392 111 15 656 5715 1763 909 96 535
4429 377 2459 3685 818 3682 1403 993 2899 4533
3466 76 250 471 801 128 12739 9781 167 2566
863 33908 1226 1023 630 120 335 82 164 63
34 188 431 2893 10468 564 1930 1646 237 70
936 5079 1818 245 57 218 1427 109 1146 1841
196 1801 68 640 403 3 119 2928 81 1090
9 86 395 1401 2444 1114 753 2071 7606 764
10135 1303 3525 8 13 294 23 271 274 348
66 372 77 432 43 392 49 17 11 21
2495 12790 26 11043 836 3963 269 27 5 3906
7086 176 212 2 1104 8237 146 6659 242 4
39 641 555 1697 454 1885 44 240 25 10
33 1196 456 9876 14013 12 425 103 6856 52
182 4876 1 676 4451 1876 45 35 138 3704

```
 26507    53     7  1165  1380  1723   107  2424    85   144
  2113   406   168     6  5250 13179   261  9424 28128   785
    16  4514    65    20  4884 10847   493   306   611   189
    62   184  6064  1704  1615  2191  3121   152  8396  9954
  3087    58  5387   376  2234  6019    78    31   452   256
   231  2216   161   998  3116   129  3499   762   213    18
  2164     0  6058  9538    46    55   681   275   140  3314
   989   106  3550  6016  8989   859  1289    56  1271    84
   382    32   408  7187   677  7443    91   229  2610 10101
  3811 23492  2940    14  1569  3240  4304   215   121  1580
  1256  6545   172   444   597  1866  4495    48    75  1398
 30022  2662  7375   136    54  7409 29491 34039 10687  1546
 42174   259  1413   851 33202   752 40463  6607  4602  3840
 53824    73  6988  1830    94 66962   185  1232   266   158
  4638   390   102 122453  4170    89   460 28996   628  2478]
Length Of Unique Value : 330
Unique Value For Memory Column : [4.         5.22589168 6.         8.         3.         2.
 1.        ]
Length Of Unique Value : 7
Unique Value For Storage Column : [128.          64.          32.         256.         108.45853659
  16.         512.           8.        ]
Length Of Unique Value : 8
Unique Value For Processor Column : ['Qualcomm Snapdragon 680' 'A Bionic Chip' 'Mediatek Dimen
sity 700'
 'MediaTek Helio G35' 'MediaTek G37' 'MediaTek Helio G25' 'MediaTek G35'
 'Qualcomm Snapdragon 695 5G' 'Unisoc T612' 'Mediatek Helio G96'
 'Exynos 850' 'A Bionic Chip,  Core' 'Mediatek Dimensity 810'
 'Mediatek Helio G85' 'Qualcomm Snapdragon 750G' 'Meditek Helio G37'
 'Mediatek Helio A22' 'UNISOC T700' 'Unisoc T606' 'Mediatek Helio P35'
 'Mediatek Dimensity 1080 5G' 'Mediatek Helio G95'
 'Mediatek Helio A22, Upto 2.0 GHz' 'Mediatek Helio G95 Octa Core'
 'Google Tensor' 'MediaTek Dimensity 800U' 'Mediatek Helio G37'
 'Dimensity 810' 'A Bionic Chip with Next Generation Neural Engine'
 'Unisoc Tiger T616' 'Unisoc SC9863A/ Unisoc SC9863A1' 'Helio G88'
 'Mediatek Dimensity 1080' 'Mediatek Helio G35' 'Unisoc T616'
 'Mediatek Dimensity 810 5G' 'Qualcomm Snapdragon 695'
 'Mediatek Helio G99 Octa Core' 'Mediatek Helio G99'
 'Qualcomm Snapdragon 778G' 'Qualcomm Snapdragon 662'
 'SEC S5E8535 (Exynos 1330)' 'MediaTek P22'
 'Qualcomm Snapdragon 778G Plus' 'Qualcomm Snapdragon 778G+'
 'Qualcomm Snapdragon 778G 5G Mobile Platform' 'Octa-core'
 'Mediatek Dimensity 920' 'Octa Core' 'Unisoc T610' 'Unisoc UMS9230'
 'MediaTek Helio G88' 'Turbo Snapdragon 695' 'Mediatek Helio G25'
 'Qualcomm Snapdragon 732G' 'MediaTek Helio P35'
 'Qualcomm Snapdragon 695 (SM6375)' 'Mediatek G99' 'MediaTek Helio G80'
 'Mediatek G96' 'Mediatek Dimensity 900' 'MediaTek Helio G37'
 'MediaTek Helio G85' 'Qualcomm Snapdragon 888 +' 'Unisoc SC9863A'
 'Google Tensor G2' 'SC9863A' 'Qualcomm SM6225 Snapdragon 680 4G (6 nm)'
 'Mediatek Helio G70' 'Unisoc 9863A' 'Qualcomm Snapdragon 8 Gen 2'
 'MediaTek Helio G70' 'Qualcomm Snapdragon 870'
 'Qualcomm Snapdragon 888 Octa-Core' 'Qualcomm Snapdragon 480 Pro'
 'Dimensity 920 5G' 'Mediatek Helio G88' 'Qualcomm Snapdragon 425'
 'Qualcomm Snapdragon 720G' 'Qualcomm Snapdragon 439'
 'Qualcomm Snapdragon 888' 'Exynos Octa Core' 'MediaTek Helio G96'
 'MediaTek Helio G95' 'Octa-core(EXYNOS)' 'MediaTek Helio A20'
 'Unisoc T618' 'Quad Core' 'Snapdragon® 778G' 'MediaTek Dimensity 700'
 'MediaTek Dimensity 1200' 'Qualcomm Snapdragon 8+ Gen 1'
 'Unisoc SC9863A Octa Core' 'Mediatek Dimensity 8100'
 'Mediatek G37 Octa Core' 'Helio A20' 'MediaTek Helio Quad Core'
 'Mediatek Helio P22' 'Snapdragon 662' 'Snapdragon@ 8 Gen 1'
```

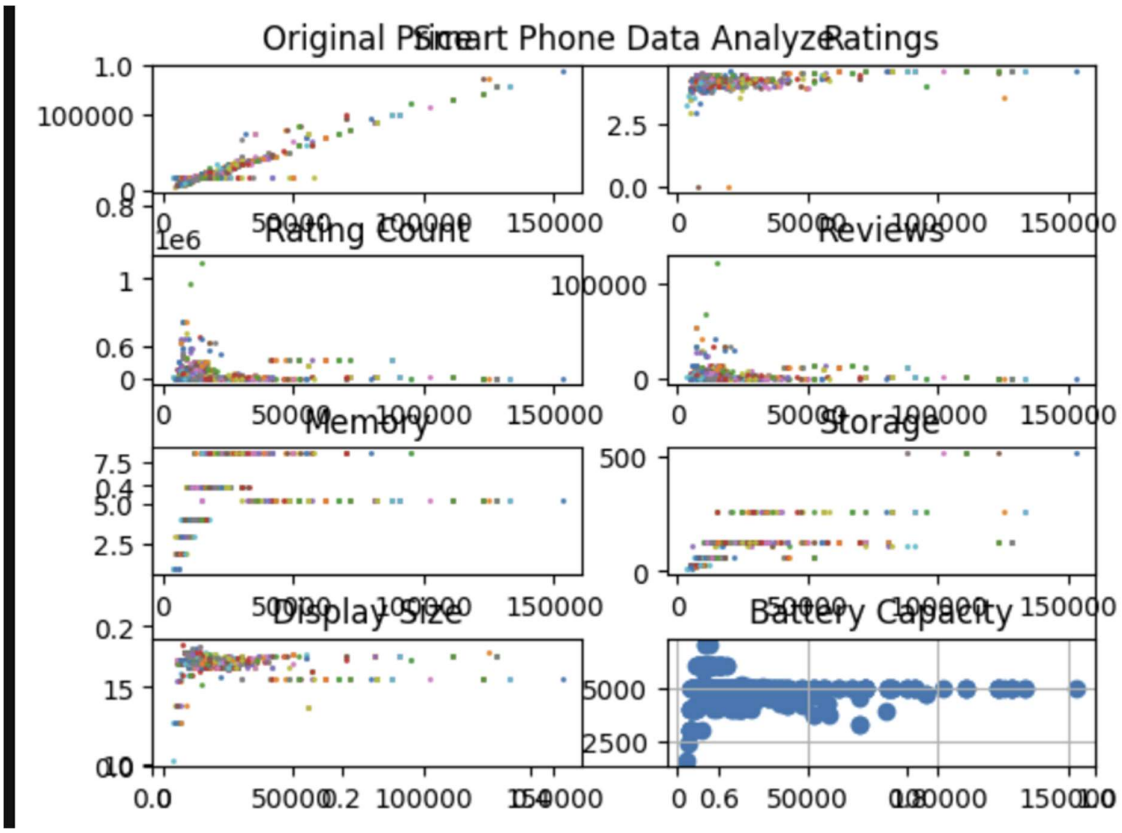
'MediaTek Dimensity 700 (MT6833)'
'Qualcomm Snapdragon (SDM450-F01) Octa Core' 'Mediatek Dimensity 1200'
'Mediatek MT6769 Helio G70' 'Helio A22' 'Exynos 1280' 'Helio G25'
'Qualcomm Snapdragon 680 (SM6225)' 'Snapdragon 720G' 'Unisoc 9832E'
'MediaTek Helio P35 Octa Core' 'UniSoc T610' 'Qualcomm Snapdragon 845'
'Mediatek Dimensity 9000' 'Qualcomm Snapdragon 660 AIE' 'Exynos 9611'
'Mediatek' 'SC9832' '2.0 GHz Mediatek P22 Octacore'
'MediaTek Helio P60 Octa Core 2.0 GHz'
'Mediatek MTK6737 Quad Core 1.3Ghz' 'Mediatek Dimensity 8100 Max'
'MTK Helio G35' 'Qualcomm Snapdragon 730G' 'Qualcomm® Snapdragon™ 720G'
'Qualcomm Snapdragon 8 Gen 1' 'Unisoc 9832E Quad Core'
'Cortex A53 architecture (Quad core)' 'Qualcomm Snapdragon 636'
'Qualcomm Snapdragon 865' 'Helio A25']
Length Of Unique Value : 131
Unique Value For Rear Camera Column : ['50MP + 2MP + 2MP ' '12MP + 12MP ' '50MP + 2MP ' '13M
P + 2MP + 2MP '
'13Mp + AI Lens ' '13MP ' '50MP + 8MP + 2MP ' '8MP ' '64MP + 8MP + 2MP '
'50MP + 5MP + 2MP ' '50MP + 8MP ' '50 MP + 2 MP + QVGA '
'50 MP + 2 MP Depth Lens + AI Lens ' '50MP' '48MP + 2MP + 2MP '
'13MP + 2MP ' '108MP + 8MP + 2MP ' '64MP + 8MP + 2MP + 2MP '
'8MP Dual Camera ' '13MP' '64MP' '50MP + 0.3MP ' '12.2MP + 12MP '
'48MP + 8MP + 2MP + 2MP ' '48MP + 2MP ' '50MP + 8MP + 2MP + 2MP '
'50MP + AI Lens ' '108MP + 2MP ' '50MP + 2MP + 0.3MP '
'50MP (OIS) + 8MP + 2MP ' '50 MP + Depth Lens ' '8MP'
'50MP + 0.3MP + 0.3MP ' '5MP' '13MP + 2MP + 0.3MP ' '108MP + 2MP + 2MP '
'108MP + 2MP (Depth) + 2MP (Macro) ' '50MP + 50MP + 2MP ' '50MP + 50MP '
'8 MP + Depth Lens ' '108MP + 8MP + 16MP ' '12MP + 12MP + 8MP (OIS) '
'50MP + 50MP + 2MP + 2MP ' '108MP + 2MP Depth Lens + AI lens ' '50MP '
'50MP + 2MP Depth + AI Lens ' '64MP + 8MP + 5MP + 2MP ' '50MP + 5MP '
'108MP' '108MP + 13MP + 2MP ' '13MP + Depth Lens ' '200MP + 8MP + 2MP '
'64MP (OIS) + 2MP + 2MP ' '48MP'
'50MP + 2MP Depth Sensor + 2MP Macro Sensor ' '64MP + 2MP '
'50MP + 13MP + 2MP ' '50MP + 12MP ' '64MP + 2MP + 2MP '
'64MP + 64MP + 8MP + 2MP ' '50MP + 10MP + 12MP '
'48MP Primary Camera + 2MP Macro Lens + 2MP Depth Sensor '
'64MP + 8MP + 5MP ' '2MP' '50 MP + 2 MP + AI Lens '
'64MP + 12MP + 5MP + 5MP ' '12MP ' '50MP + 12MP + 10MP '
'50MP + 5MP + 2MP + 2MP ' '16MP ' '108MP + 12MP + 5MP + 5MP '
'8MP + Depth Sensor ' '5MP ' '64MP ' '48MP + 12MP + 12MP + 12MP '
'64MP + 5MP + 2MP ' '200MP + 50MP + 12MP ' '8MP + 0.3MP '
'50 MP + 2 MP Depth Lens + AI lens ' '13MP + 2MP' '13MP + 5MP + 2MP '
'108MP ' '48MP + 8MP + 2MP ' '13MP + 8MP + 2MP + 2MP '
'48MP + 8MP + 5MP + 2MP ' '48MP + 5MP + 2MP + 2MP '
'13MP + Digital Camera ' '12MP + 12MP + 12MP '
'50 MP + 2 MP Depth Lens + 2 MP Macro Lens ' '64MP + 12MP + 8MP + 2MP '
'2MP ' '12MP + 2MP + 2MP ' '32MP' '13 MP + 2 MP Depth Lens '
'64MP + 64MP + 2MP + 2MP ' '12MP + 5MP ' '12MP' '50MP + 12MP + 12MP '
'48MP + 5MP ' '48 MP + 13 MP Portrait Lens + 2 MP Bokeh Lens '
'13MP + 8MP + 5MP + 2MP ' '200MP + 10MP + 12MP + 10MP '
'50MP + 2MP + AI Lens ' '108MP + 8MP + 5MP ' '12MP + 2MP '
'12MP + 12MP + 8MP ' '64MP + 13MP + 5MP ']
Length Of Unique Value : 107
Unique Value For Front Camera Column : [' 16MP' ' 12MP' ' 8MP' ' 5MP' ' 13MP' ' 32MP' ' 16MP + 16
MP Dual'
' 60 MP with OIS' ' 44MP' ' 10.8MP' ' 8MP + 2MP Dual' ' 16MP + 2MP Dual'
' 20MP' ' 10MP' ' 32MP + 2MP Dual' ' 2MP' ' 60MP' ' 8MP + 8MP Dual'
' 50MP + 8MP Dual' ' 50MP' ' 16MP + 8MP Dual' ' 16MP Dual'
' 20MP + 2MP Dual' ' 32MP + 8MP Dual']
Length Of Unique Value : 24

```
Unique Value For Display Size Column : [16.36 15.49 16.71 17.02 16.59 17.32 16.76 16.64 16.51 16.
33 16.26 16.94
 16.66 16.56 16.74 15.6  17.07 16.43 16.54 17.22 17.53 16.97 16.21 17.27
 16.   16.55 16.81 12.7  13.84 15.8  15.46 17.78 17.09 13.72 15.44 17.65
 16.48 15.7  15.9  10.16 16.69 15.21]
Length Of Unique Value : 42
Unique Value For Battery Capacity Column : [5000.        4950.93187661 6000.        4410.        4310.
 4500.        4800.        4020.        4700.        4000.
 7000.        5020.        4050.        4980.        4400.
 4270.        5080.        3900.        5200.        3300.
 4250.        3020.        3000.        3700.        4230.
 5065.        4610.        4600.        4300.        4060.
 1600.        4520.        4200.        5050.        5160.
 2400.       ]
Length Of Unique Value : 36
Unique Value For Battery Type Column : ['Lithium' 'Lithium Polymer' 'Lithium Ion']
Length Of Unique Value : 3
```

Q-4 :- Draw applicable plots to visualise data using the subplot concept on the dataset. (scatter plot/ line graph/ histogram etc.)

```
#For plot graph we import the library of matplotlib

import matplotlib.pyplot as plt

plt.title('Smart Phone Data Analyze')

#here we used the subplot method

plt.subplot(4,2,1)

plt.plot([data1["discounted_price"]],[data1["original_price"]],marker='o',markersize=1)

plt.title('Original Price')


plt.subplot(4,2,2)

plt.plot([data1["discounted_price"]],[data1["ratings"]],marker='o',markersize=1)

plt.title('Ratings')


plt.subplot(4,2,3)

plt.plot([data1["discounted_price"]],[data1["rating_count"]],marker='o',markersize=1)

plt.title('Rating Count')


plt.subplot(4,2,4)

plt.plot([data1["discounted_price"]],[data1["reviews"]],marker='o',markersize=1)

plt.title('Reviews')
```

```
plt.subplot(4,2,5)
plt.plot([data1["discounted_price"]],[data1["memory"]],marker='o',markersize=1)
plt.title('Memory')

plt.subplot(4,2,6)
plt.plot([data1["discounted_price"]],[data1["storage"]],marker='o',markersize=1)
plt.title('Storage')

plt.subplot(4,2,7)
plt.plot([data1["discounted_price"]],[data1["display_size"]],marker='o',markersize=1)
plt.title('Display Size')

plt.subplot(4,2,8)
plt.scatter([data1["discounted_price"]],[data1["battery_capacity"]])
plt.title('Battery Capacity')

plt.subplots_adjust(hspace=0.5)
plt.grid()
plt.show()
```

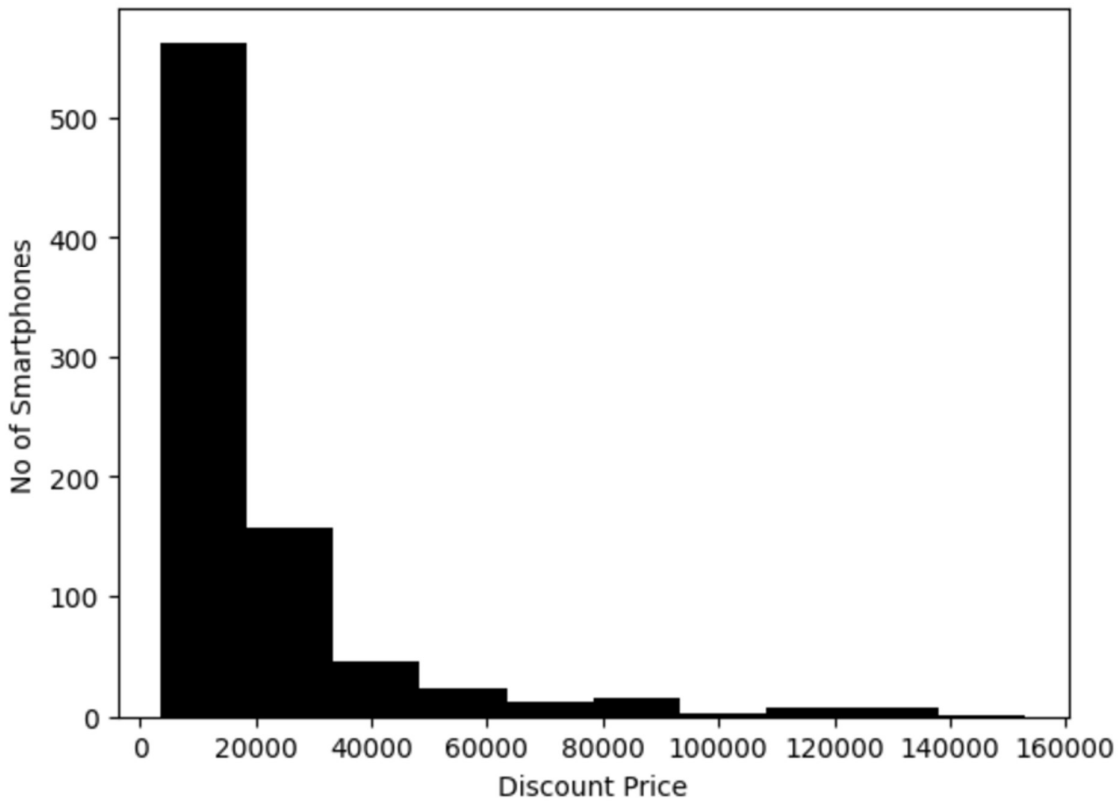


#here we plot the histogram of Discount price

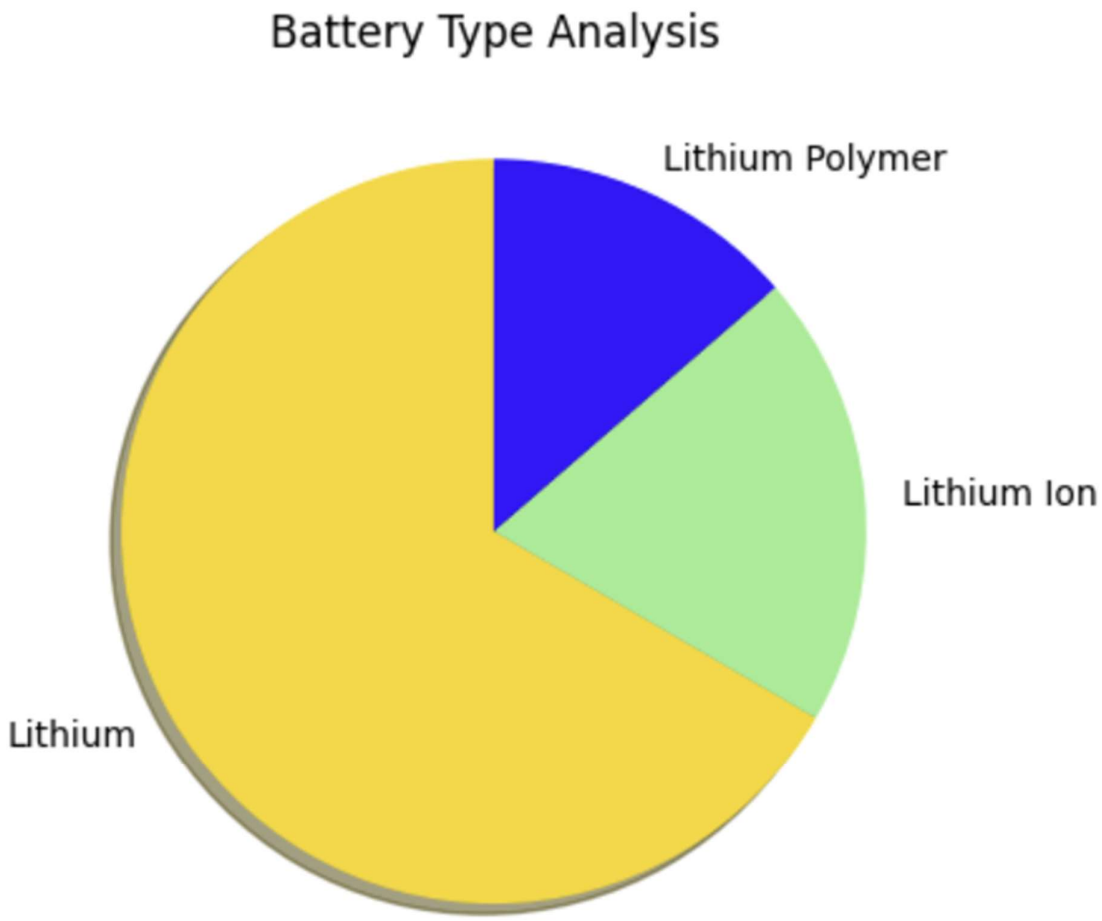
plt.hist(data1['discounted_price'],color='k')

plt.ylabel("No of Smartphones")

plt.xlabel("Discount Price")




```
#here we plot the pie chart of Battery Type
maritial = data1["battery_type"].value_counts()

label = maritial.index
counts = maritial.values
colors = ['gold','lightgreen','blue']
plt.title('Battery Type Analysis')

plt.pie(counts, labels = label, startangle = 90,shadow=True, colors=colors)
plt.show()
```



Q-5 :- Train the model of the K-nearest Neighbors Classifier/Regressor with 80% of the data and predict the class label for the rest 20% of the data. Evaluate the model with all appropriate measures.

#Here first we done the trainig & testing and then we apply the classification

#Here we classify the Batttery type

from sklearn.model_selection import train_test_split

selected_columns = ['original_price', 'discounted_price', 'ratings', 'rating_count', 'reviews', 'memory','display_size']

X = data1[selected_columns].astype(float)

Y = data1['battery_capacity']

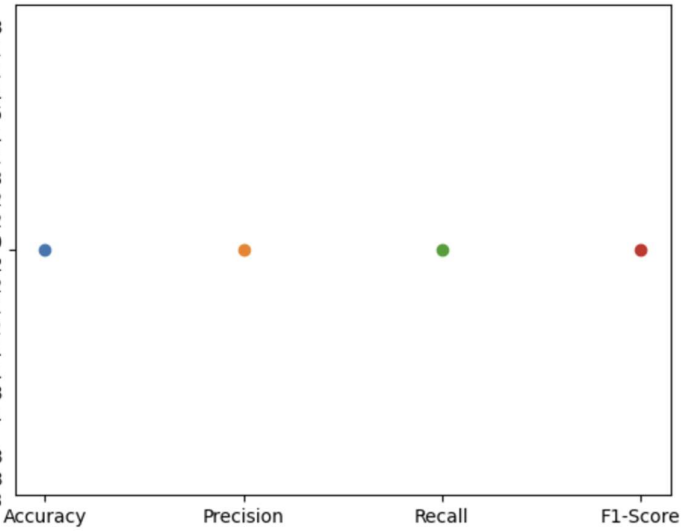
```
X = X.values.tolist()
X = np.array(X,dtype=float)
Y = np.array(Y,dtype=int)

X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.2)
from sklearn.metrics import accuracy_score,classification_report
from sklearn.neighbors import KNeighborsClassifier

k = int(input("Enter the number of nearest neighbors to be used, i.e. k :"))

model = KNeighborsClassifier(n_neighbors=k, weights='distance')

model.fit(X_train,Y_train)

pred = model.predict(X_test)

accuracy = accuracy_score(Y_test,pred)
cr=classification_report(Y_test,pred)
plt.scatter(['Accuracy'],[cr],label='Accuracy')
plt.scatter(['Precision'],[cr],label='Precision')
plt.scatter(['Recall'],[cr],label='Recall')
plt.scatter(['F1-Score'],[cr],label='F1-score')

print("Accuracy : ",accuracy)
print(classification_report(Y_test,pred))
```

Accuracy : 0.8511904761904762

	precision	recall	f1-score	support
3000	1.00	0.50	0.67	2
3020	0.00	0.00	0.00	1
3900	0.00	0.00	0.00	1
4000	0.67	0.50	0.57	4
4020	1.00	1.00	1.00	1
4200	0.00	0.00	0.00	2
4230	0.00	0.00	0.00	1
4250	0.00	0.00	0.00	2
4400	1.00	1.00	1.00	1
4500	1.00	0.73	0.84	11
4610	0.00	0.00	0.00	0
4700	0.00	0.00	0.00	1
4800	1.00	1.00	1.00	1
4950	0.92	1.00	0.96	11
4980	0.00	0.00	0.00	2
5000	0.88	0.97	0.92	115
5020	0.00	0.00	0.00	2
5080	0.00	0.00	0.00	1
6000	0.75	0.75	0.75	8
7000	0.00	0.00	0.00	1
accuracy			0.85	168
macro avg	0.41	0.37	0.39	168
weighted avg	0.81	0.85	0.82	168

---

	precision	recall	f1-score	support
3000	1.00	0.67	0.80	3
3300	1.00	1.00	1.00	1
3700	1.00	1.00	1.00	1
3900	0.00	0.00	0.00	1
4000	1.00	0.20	0.33	5
4060	0.00	0.00	0.00	1
4200	1.00	1.00	1.00	1
4230	0.33	0.33	0.33	3
4250	0.00	0.00	0.00	2
4400	0.00	0.00	0.00	2
4500	0.86	0.60	0.71	10
4700	1.00	1.00	1.00	2
4950	0.92	0.92	0.92	12
4980	0.50	1.00	0.67	1
5000	0.84	0.95	0.89	112
5020	0.00	0.00	0.00	1
5065	0.00	0.00	0.00	1
6000	0.50	0.50	0.50	8
7000	0.00	0.00	0.00	1
accuracy			0.82	168
macro avg	0.52	0.48	0.48	168
weighted avg	0.79	0.82	0.79	168

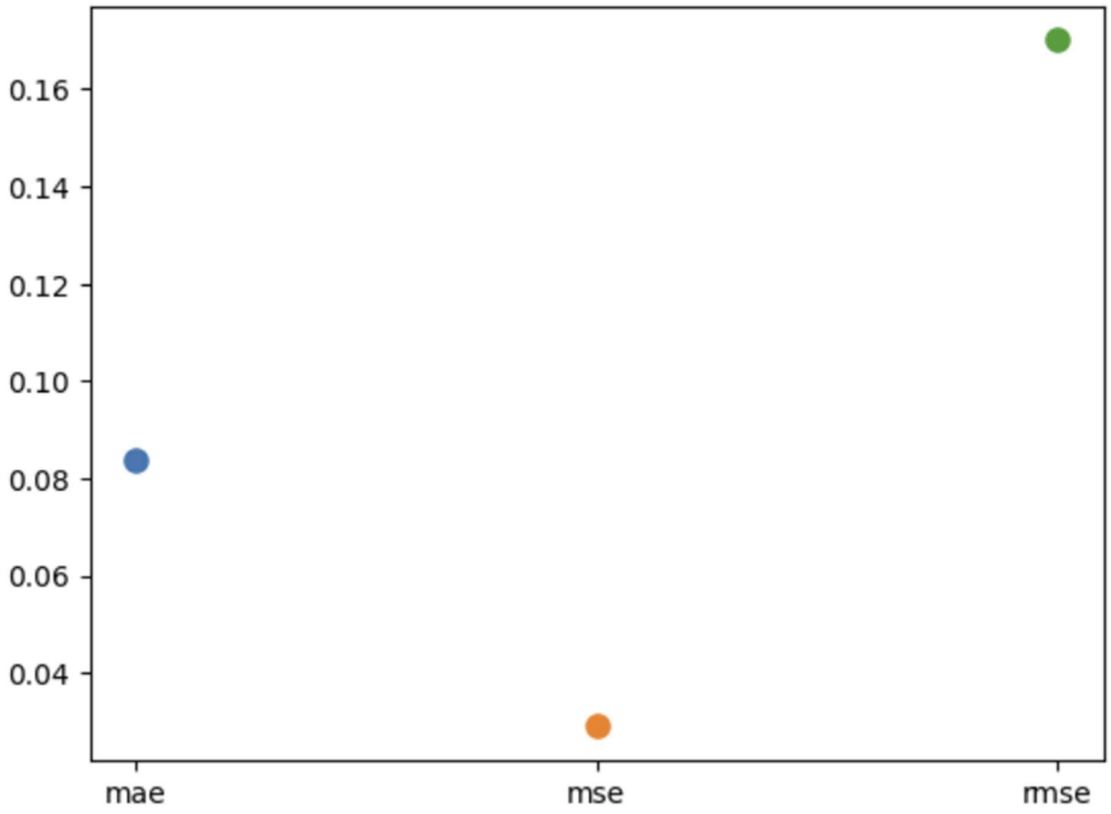


#here we do the regrasssion of battery capacity

```
from sklearn.model_selection import train_test_split

from sklearn.neighbors import KNeighborsRegressor

from sklearn.metrics import mean_absolute_error ,mean_squared_error



selected_columns = ['original_price', 'discounted_price', 'ratings', 'rating_count', 'reviews',
'memory','battery_capacity']


X = data1[selected_columns].astype(float)

Y = data1['display_size']


X = X.values.tolist()

X = np.array(X,dtype=float)

Y = np.array(Y,dtype=float)


X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.2)


model = KNeighborsRegressor(n_neighbors=k, weights='distance')


model.fit(X_train,Y_train)


pred = model.predict(X_test)


mae = mean_absolute_error(Y_test,pred)

mse = mean_squared_error(Y_test,pred)

rmse = np.sqrt(mse)


plt.plot(['mae'],[mae],marker='o',markersize=8,label='MAE')

plt.plot(['mse'],[mse],marker='o',markersize=8,label='MSE')

plt.plot(['rmse'],[rmse],marker='o',markersize=8,label='RMSE')
```

```
print("Using Sklearn:")
print("Mean Absolute Error (MAE) :",mae)
print("Mean Squared Error (MSE) :",mse)
print("Root Mean Squared Error (RMSE) :",mae)
```

```
Using Sklearn:
Mean Absolute Error (MAE) : 0.1366522036779201
Mean Squared Error (MSE) : 0.0810485613118826
Root Mean Squared Error (RMSE) : 0.1366522036779201
```



Conclusion:- Here the data of smartphones is used for the several operation.

- We use the classification for the Battery size
- We use the Regression for the Display size
- By doing this assignment we understand the problem of handling the data, interpretation of data.