Data Science CA2

Topic : EDA(Exploratory Data Analysis)
Dataset : Smartphone / Mobile Phone
Name : Jatin Dnyaneshwar Gangare
Class : SY.IT / A ROLL No : 6130

We will follow steps for EDA as follows:

1) Import Libraries
2) Know your dataset
3) Data Cleaning
4) Data Visualization
5) Conclusion

Introduction:

Smartphones have become an indispensable part of our lives, and the data they produce can give us important information on user preferences and behaviour. To learn more about the traits and usage habits of smartphone users, we will examine a dataset of smartphone usage in this EDA.

Dataset Overview:

Before diving into the analysis, let's take a quick look at the dataset:

Dataset Source: Kaggle.com

Dataset Name: smartphones.csv

Columns: 'model', 'price', 'rating', 'sim', 'processor', 'ram', 'battery', 'display', 'camera', 'card', 'os'.

Import Libraries :

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

We will use pandas library to import csv file of dataset

```
df = pd.read_csv(r"C:\Users\jatin\Downloads\smartphones.csv")
```

Know Your Dataset :

```
df.columns
```

```
Index(['model', 'price', 'rating', 'sim', 'processor', 'ram', 'battery',
       'display', 'camera', 'card', 'os'],
      dtype='object')
```

```
df.head()
```

	model	price	rating	sim	processor	ram	battery	display	camera	card	os
0	OnePlus 11 5G	₹54,999	89.0	Dual Sim, 3G, 4G, 5G, VoLTE, Wi-Fi, NFC	Snapdragon 8 Gen2, Octa Core, 3.2 GHz Processor	12 GB RAM, 256 GB inbuilt	5000 mAh Battery with 100W Fast Charging	6.7 inches, 1440 x 3216 px, 120 Hz Display wit...	50 MP + 48 MP + 32 MP Triple Rear & 16 MP Fron...	Memory Card Not Supported	Android v13
1	OnePlus Nord CE 2 Lite 5G	₹19,989	81.0	Dual Sim, 3G, 4G, 5G, VoLTE, Wi-Fi	Snapdragon 695, Octa Core, 2.2 GHz Processor	6 GB RAM, 128 GB inbuilt	5000 mAh Battery with 33W Fast Charging	6.59 inches, 1080 x 2412 px, 120 Hz Display wi...	64 MP + 2 MP + 2 MP Triple Rear & 16 MP Front ...	Memory Card (Hybrid), upto 1 TB	Android v12
2	Samsung Galaxy A14 5G	₹16,499	75.0	Dual Sim, 3G, 4G, 5G, VoLTE, Wi-Fi	Exynos 1330, Octa Core, 2.4 GHz Processor	4 GB RAM, 64 GB inbuilt	5000 mAh Battery with 15W Fast Charging	6.6 inches, 1080 x 2408 px, 90 Hz Display with...	50 MP + 2 MP + 2 MP Triple Rear & 13 MP Front ...	Memory Card Supported, upto 1 TB	Android v13
3	Motorola Moto G62 5G	₹14,999	81.0	Dual Sim, 3G, 4G, 5G, VoLTE, Wi-Fi	Snapdragon 695, Octa Core, 2.2 GHz Processor	6 GB RAM, 128 GB inbuilt	5000 mAh Battery with Fast Charging	6.55 inches, 1080 x 2400 px, 120 Hz Display wi...	50 MP + 8 MP + 2 MP Triple Rear & 16 MP Front ...	Memory Card (Hybrid), upto 1 TB	Android v12
4	Realme 10 Pro Plus	₹24,999	82.0	Dual Sim, 3G, 4G, 5G, VoLTE, Wi-Fi	Dimensity 1080, Octa Core, 2.6 GHz Processor	6 GB RAM, 128 GB inbuilt	5000 mAh Battery with 67W Fast Charging	6.7 inches, 1080 x 2412 px, 120 Hz Display wit...	108 MP + 8 MP + 2 MP Triple Rear & 16 MP Front...	Memory Card Not Supported	Android v13

```
df.tail()
```

	model	price	rating	sim	processor	ram	battery	display	camera	card	os
1015	Motorola Moto Edge S30 Pro	₹34,990	83.0	Dual Sim, 3G, 4G, 5G, VoLTE, Wi-Fi	Snapdragon 8 Gen1, Octa Core, 3 GHz Processor	8 GB RAM, 128 GB inbuilt	5000 mAh Battery with 68.2W Fast Charging	6.67 inches, 1080 x 2460 px, 120 Hz Display wi...	64 MP + 8 MP + 2 MP Triple Rear & 16 MP Front ...	Android v12	No FM Radio
1016	Honor X8 5G	₹14,990	75.0	Dual Sim, 3G, 4G, 5G, VoLTE, Wi-Fi	Snapdragon 480+, Octa Core, 2.2 GHz Processor	6 GB RAM, 128 GB inbuilt	5000 mAh Battery with 22.5W Fast Charging	6.5 inches, 720 x 1600 px Display with Water D...	48 MP + 2 MP + Depth Sensor Triple Rear & 8 MP...	Memory Card Supported, upto 1 TB	Android v11
1017	POCO X4 GT 5G (8GB RAM + 256GB)	₹28,990	85.0	Dual Sim, 3G, 4G, 5G, VoLTE, Wi-Fi, NFC, IR Bl...	Dimensity 8100, Octa Core, 2.85 GHz Processor	8 GB RAM, 256 GB inbuilt	5080 mAh Battery with 67W Fast Charging	6.6 inches, 1080 x 2460 px, 144 Hz Display wit...	64 MP + 8 MP + 2 MP Triple Rear & 16 MP Front ...	Memory Card Not Supported	Android v12
1018	Motorola Moto G91 5G	₹19,990	80.0	Dual Sim, 3G, 4G, 5G, VoLTE, Wi-Fi, NFC	Snapdragon 695, Octa Core, 2.2 GHz Processor	6 GB RAM, 128 GB inbuilt	5000 mAh Battery with Fast Charging	6.8 inches, 1080 x 2400 px Display with Punch ...	108 MP + 8 MP + 2 MP Triple Rear & 32 MP Front...	Memory Card Supported, upto 1 TB	Android v12
1019	Samsung Galaxy M52s 5G	₹24,990	74.0	Dual Sim, 3G, 4G, 5G, VoLTE, Wi-Fi	Octa Core Processor	8 GB RAM, 128 GB inbuilt	5000 mAh Battery with Fast Charging	6.5 inches, 1080 x 2400 px Display with Water ...	64 MP + 8 MP + 5 MP Triple Rear & 32 MP Front ...	Memory Card Supported, upto 1 TB	Android v12

Data Cleaning :

Here we will check null values in the columns

```
df.isnull().sum()
```

```
model          0
price          0
rating       141
sim            0
processor      0
ram            0
battery        0
display        0
camera         1
card           7
os            17
dtype: int64
```

```
df.duplicated().sum()
```

```
0
```

Here we will fill the null values in rating column

```
df['rating'].fillna(df['rating'].mean(), inplace=True)
df
```

Here we have checked that how many null values does camera colum have

```
df['camera'].isnull().sum()
```

```
1
```

Here we will drop the the row in which the null values are occuring

```
dropedcol=df.drop(['card','os','display'],axis=1)
```

```
df1=dropedcol.dropna()
```

Here we will remove unnecessary symbols and characters

price - has unneccesary '₹' validity price - has ',' between numbers validity

```
df1_copy = df1.copy()
df1_copy['price'] = df1_copy['price'].str.replace('₹', '').str.replace(',', '')
df1_copy
```

Here we first create 3 different column in which we will fill Boolean values with the help data from Sim column

```
# Create new columns with initial values as False
df1_copy[['has_5g', 'has_NFC', 'has_IR_Blaster']] = False

# Update the new columns based on the 'sim' column
df1_copy['has_5g'] = df1_copy['sim'].str.contains('5G')
df1_copy['has_NFC'] = df1_copy['sim'].str.contains('NFC')
df1_copy['has_IR_Blaster'] = df1_copy['sim'].str.contains('IR Blaster')

# Drop the original 'sim' column
df1_copy.drop('sim', axis=1, inplace=True)
df1_copy
```

Here we will split model column and differentiate brand and model

```
# Split the 'model' column into 'brand' and 'model'
df1_copy[['brand', 'model']] = df1_copy['model'].str.split(n=1, expand=True)
df1_copy
```

Here we will split Processor column and differentiate processor name, number of code and processor speed

```
# Split the 'processor' column into 'processor_name', 'core', and 'processor_speed'
df1_copy[['processor_name', 'core', 'processor_speed']] = df1_copy['processor'].str.split(', ', expand=True)

# Drop the original 'processor' column
df1_copy.drop('processor', axis=1, inplace=True)+
df1_copy
```

Here we will take only the name of processor and discard other metadata

```
# Extract only the first word of the processor name
df1_copy['processor_name'] = df1_copy['processor_name'].str.split().str[0]
```

Here we will clean battery_capacity column and by keeping only numerical data and extracting boolean data and storing it in another column

```
# Extract only numeric part from the 'battery' column as 'battery_capacity'
df1_copy['battery_capacity'] = df1_copy['battery'].str.extract(r'(\d+)', expand=False)

# Convert the 'battery_capacity' column to numeric, setting NaN values to 0
df1_copy['battery_capacity'] = pd.to_numeric(df1_copy['battery_capacity'], errors='coerce').fillna(0).astype(int)

# Create a boolean column 'supports_fast_charging'
df1_copy['supports_fast_charging'] = df1_copy['battery'].str.contains('Fast Charging', case=False)

# Drop the original 'battery' column
df1_copy.drop('battery', axis=1, inplace=True)
```

Here we will split ram column into ram and rom as it contains data for both

```
df1_copy[['ram', 'rom']] = df1_copy['ram'].str.extract(r'(\d+)\s*GB RAM, (\d+)\s*GB inbuilt', expand=True)

# Convert 'ram' and 'rom' columns to numeric, handling NaN values
df1_copy['ram'] = pd.to_numeric(df1_copy['ram'], errors='coerce')
df1_copy['rom'] = pd.to_numeric(df1_copy['rom'], errors='coerce')
```

Here we will only keep numerical data

```
# Rename the 'processor_speed' column
df1_copy.rename(columns={'processor_speed': 'processor_speed (GHz)'}, inplace=True)

# Remove non-digit characters from the 'processor_speed (GHz)' column values
df1_copy['processor_speed (GHz)'] = df1_copy['processor_speed (GHz)'].replace('[^\d.]', '', regex=True)
```

Here we will fill null values in ram and com column with median() function
And in processor_speed(GHz) column with pad method

```
# Impute missing values with median for 'ram' and 'rom'
df1_copy['ram'].fillna(df1_copy['ram'].median(), inplace=True)
df1_copy['rom'].fillna(df1_copy['rom'].median(), inplace=True)

# Impute missing values with pad method
df1_copy['processor_speed (GHz)'].fillna(method='pad', inplace=True)
```

Here we will convert True/False values to boolean (0/1)

```
# Convert 'True' and 'False' to 1 and 0 for specified columns
columns_to_convert = ['has_5g', 'has_NFC', 'has_IR_Blaster','supports_fast_charging']
df1_copy[columns_to_convert] = df1_copy[columns_to_convert].astype(int)
```

At last we will export our cleaned data to a different csv file

```
#To Export Cleaned Data in csv format
df1_copy.to_csv("C:\\Users\\jatin\\Documents\\Data Science\\Final Ca2\\Smartphone_cleaned.csv")
```

Data Visualization :

Importing all necessary Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Here the data to be used is already cleaned

```
df = pd.read_csv(r"C:\Users\jatin\Documents\Data Science\Final Ca2\Smartphone_cleaned.csv")
```

- Exploring data

```
df.shape
```

```
(1019, 13)
```

```
df.columns
```

```
Index(['model', 'price', 'rating', 'ram', 'has_5g', 'has_NFC',
       'has_IR_Blaster', 'brand', 'processor_name', 'processor_speed (GHz)',
       'battery_capacity', 'supports_fast_charging', 'rom'],
      dtype='object')
```

```
df.head()
```

	model	price	rating	ram	has_5g	has_NFC	has_IR_Blaster	brand	processor_name	processor_speed (GHz)	battery_capacity	supports_fast_charging	rom
0	11 5G	54999	89.0	12	1	1	0	OnePlus	Snapdragon	3.2	5000	1	256
1	Nord CE 2 Lite 5G	19989	81.0	6	1	0	0	OnePlus	Snapdragon	2.2	5000	1	128
2	Galaxy A14 5G	16499	75.0	4	1	0	0	Samsung	Exynos	2.4	5000	1	64
3	Moto G62 5G	14999	81.0	6	1	0	0	Motorola	Snapdragon	2.2	5000	1	128
4	10 Pro Plus	24999	82.0	6	1	0	0	Realme	Dimensity	2.6	5000	1	128

```
df.tail()
```

	model	price	rating	ram	has_5g	has_NFC	has_IR_Blaster	brand	processor_name	processor_speed (GHz)	battery_capacity	supports_fast_charging	rom
014	Moto Edge S30 Pro	34990	83.0	8	1	0	0	Motorola	Snapdragon	3.00	5000	1	128
015	X8 5G	14990	75.0	6	1	0	0	Honor	Snapdragon	2.20	5000	1	128
016	X4 GT 5G (8GB RAM + 256GB)	28990	85.0	8	1	1	1	POCO	Dimensity	2.85	5080	1	256
017	Moto G91 5G	19990	80.0	6	1	1	0	Motorola	Snapdragon	2.20	5000	1	128
018	Galaxy M52s 5G	24990	74.0	8	1	0	0	Samsung	Unisoc	2.20	5000	1	128

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1019 entries, 0 to 1018
Data columns (total 13 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   model                   1019 non-null   object
 1   price                   1019 non-null   int64
 2   rating                  1019 non-null   float64
 3   ram                     1019 non-null   int64
 4   has_5g                  1019 non-null   int64
 5   has_NFC                 1019 non-null   int64
 6   has_IR_Blaster          1019 non-null   int64
 7   brand                   1019 non-null   object
 8   processor_name          1019 non-null   object
 9   processor_speed (GHz)   1019 non-null   float64
 10  battery_capacity        1019 non-null   int64
 11  supports_fast_charging  1019 non-null   int64
 12  rom                     1019 non-null   int64
dtypes: float64(2), int64(8), object(3)
memory usage: 103.6+ KB
```

- Statistical Analysis

```
df.describe()
```

	price	rating	ram	has_5g	has_NFC	has_IR_Blaster	processor_speed (GHz)	battery_capacity	supports_fast_charging	
count	1019.000000	1019.000000	1019.000000	1019.000000	1019.000000	1019.000000	1019.000000	1019.000000	1019.000000	1019.000
mean	31400.738960	78.258248	6.526006	0.538763	0.385672	0.156035	2.425123	4607.045142	0.821394	135.772
std	39177.240556	6.874998	2.685927	0.498740	0.486993	0.363067	0.463674	1363.108817	0.383211	84.020
min	99.000000	60.000000	1.000000	0.000000	0.000000	0.000000	1.100000	0.000000	0.000000	8.000
25%	12489.500000	75.000000	4.000000	0.000000	0.000000	0.000000	2.050000	4500.000000	1.000000	64.000
50%	19980.000000	78.258248	6.000000	1.000000	0.000000	0.000000	2.300000	5000.000000	1.000000	128.000
75%	34999.000000	83.000000	8.000000	1.000000	1.000000	0.000000	2.840000	5000.000000	1.000000	128.000
max	650000.000000	89.000000	18.000000	1.000000	1.000000	1.000000	3.220000	22000.000000	1.000000	512.000

```
df.isnull().sum()
```

```
model                     0
price                     0
rating                    0
ram                       0
has_5g                    0
has_NFC                   0
has_IR_Blaster            0
brand                     0
processor_name            0
processor_speed (GHz)     0
battery_capacity          0
supports_fast_charging    0
rom                       0
dtype: int64
```

```
df.duplicated().sum()
```

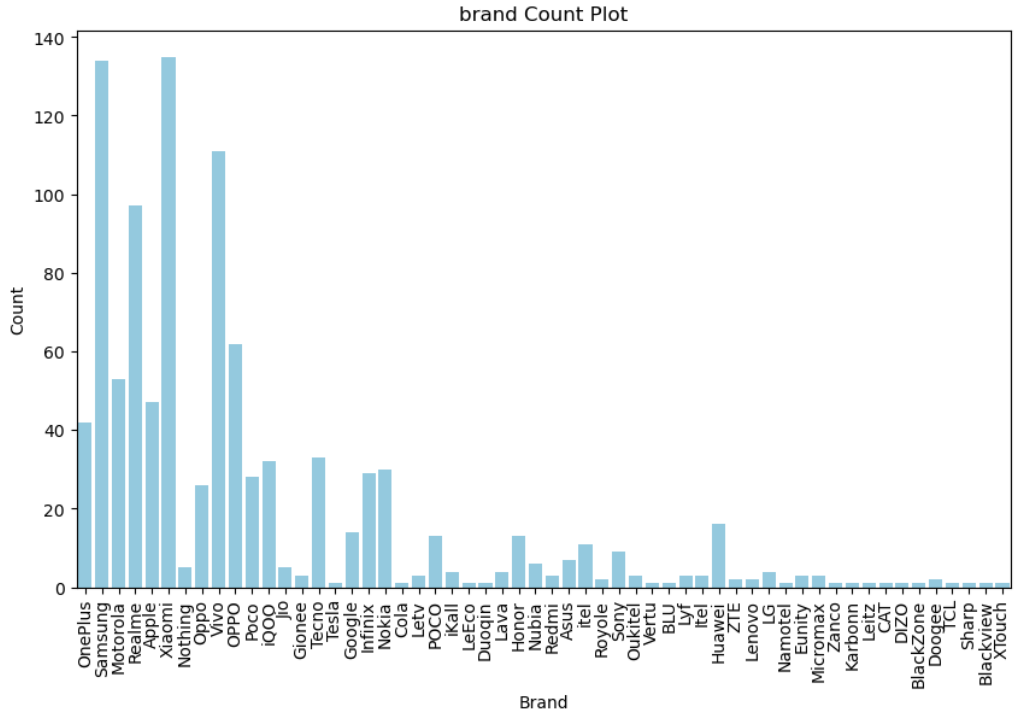
```
0
```

```
df.nunique()
```

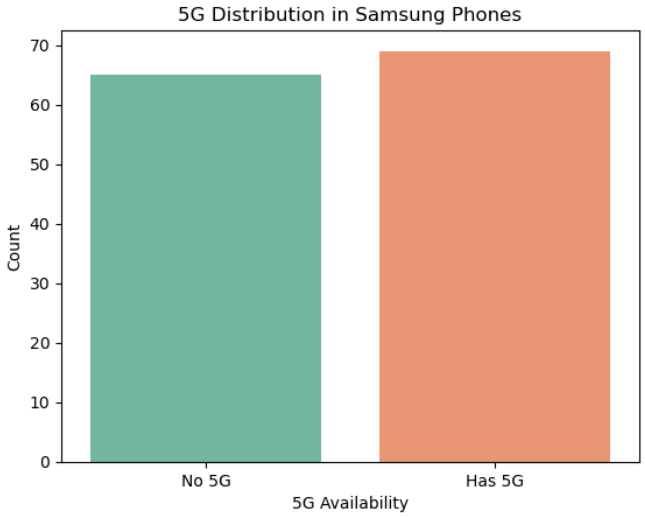
```
model                     1005
price                      411
rating                      31
ram                          9
has_5g                       2
has_NFC                      2
has_IR_Blaster               2
brand                       56
processor_name              16
processor_speed (GHz)       36
battery_capacity           106
supports_fast_charging       2
rom                          7
dtype: int64
```

- Univariate Analysis

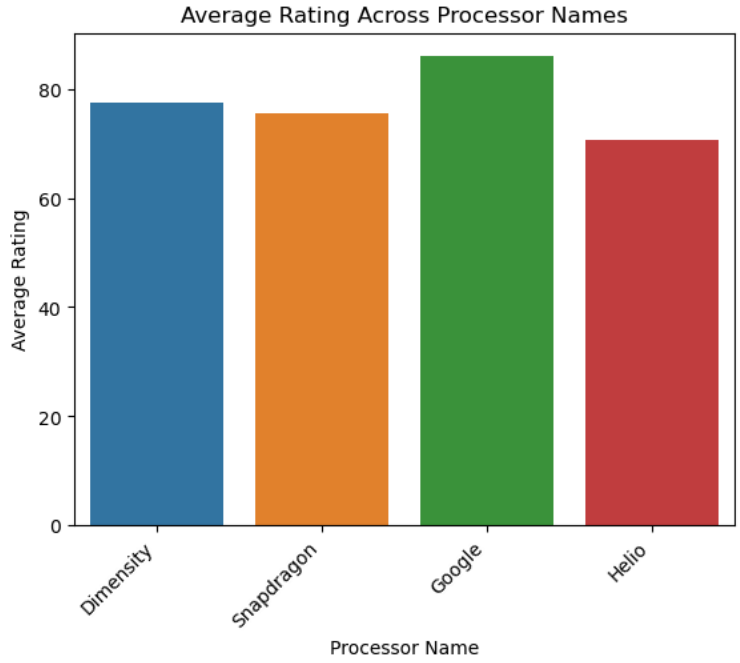
```
# Bar chart for brand distribution
plt.figure(figsize=(10, 6))
sns.countplot(x=df['brand'], color='skyblue')
plt.xticks(rotation=90)
plt.title("brand Count Plot")
plt.xlabel("Brand")
plt.ylabel("Count")
plt.show()
```



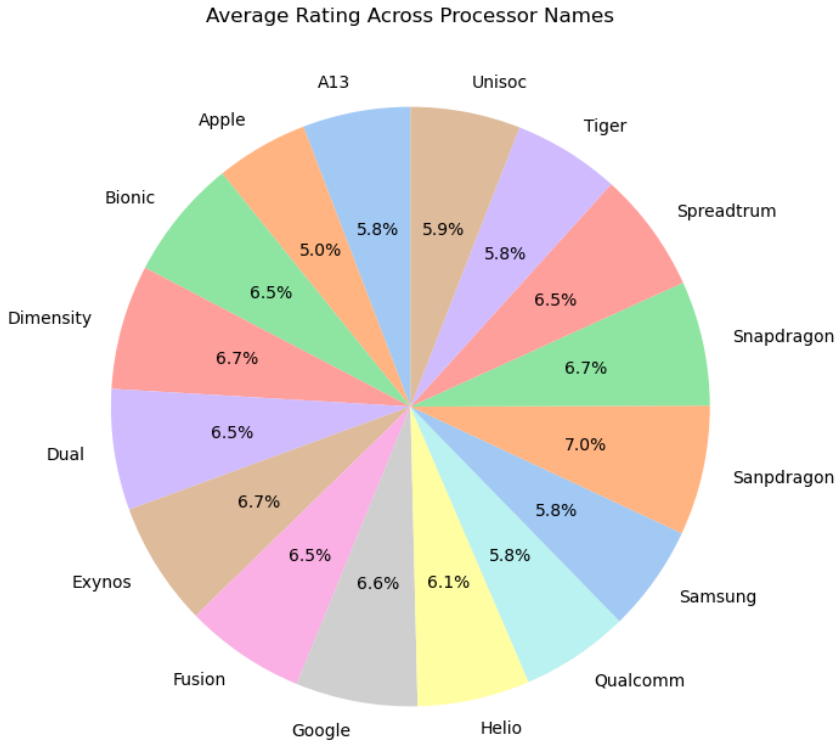

```
samsung_df = df[df['brand'] == 'Samsung']

# Bar plot for 5G distribution in Samsung phones
sns.countplot(x='has_5g', data=samsung_df, palette='Set2')
plt.xlabel('5G Availability')
plt.ylabel('Count')
plt.title('5G Distribution in Samsung Phones')
plt.xticks([0, 1], ['No 5G', 'Has 5G'])
plt.show()
```



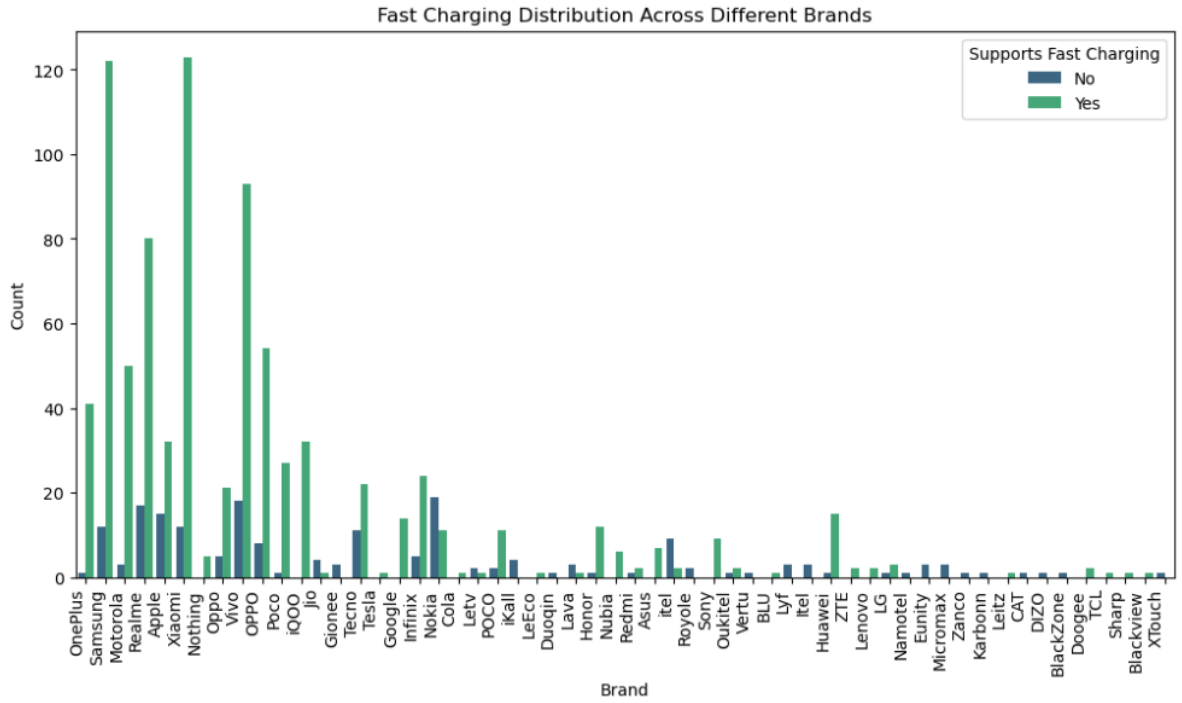
```
sns.barplot(x='processor_name', y='rating', data=df.sample(10), errorbar=None)
plt.xlabel('Processor Name')
plt.ylabel('Average Rating')
plt.title('Average Rating Across Processor Names')
plt.xticks(rotation=45, ha='right')
plt.show()
```



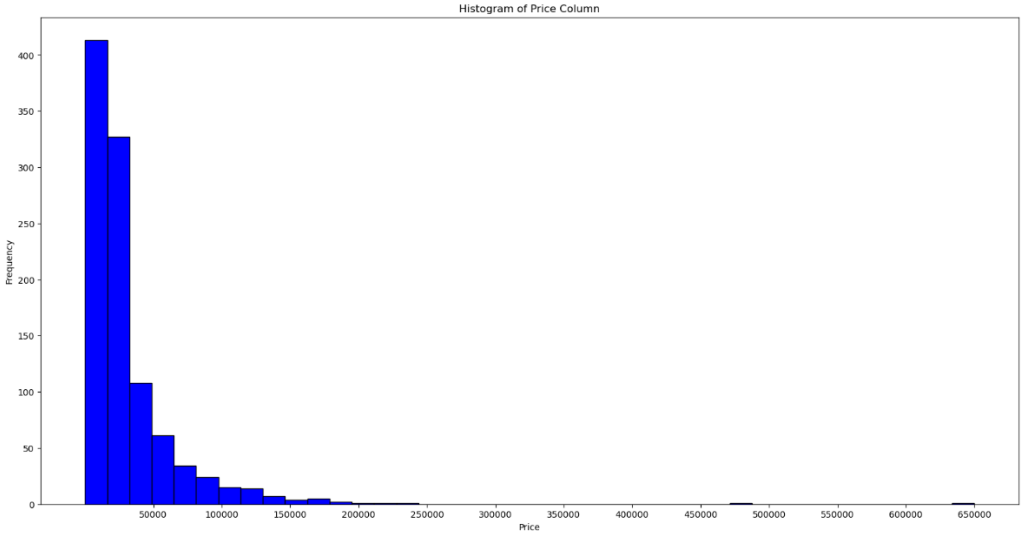

```
# Calculate the average rating for each processor name
processor_avg_rating = df.groupby('processor_name')['rating'].mean()

# Plotting a pie chart for average ratings across processor names
plt.figure(figsize=(8, 8))
plt.pie(processor_avg_rating, labels=processor_avg_rating.index, autopct='%1.1f%%', startangle=90, colors=sns.color_palette('past
plt.title('Average Rating Across Processor Names')
plt.show()
```



```
# Bar plot for fast charging distribution across different brands
plt.figure(figsize=(12, 6))
sns.countplot(x='brand', hue='supports_fast_charging', data=df, palette='viridis')
plt.xlabel('Brand')
plt.ylabel('Count')
plt.title('Fast Charging Distribution Across Different Brands')
plt.legend(title='Supports Fast Charging', labels=['No', 'Yes'])
plt.xticks(rotation=90, ha='right')
plt.show()
```



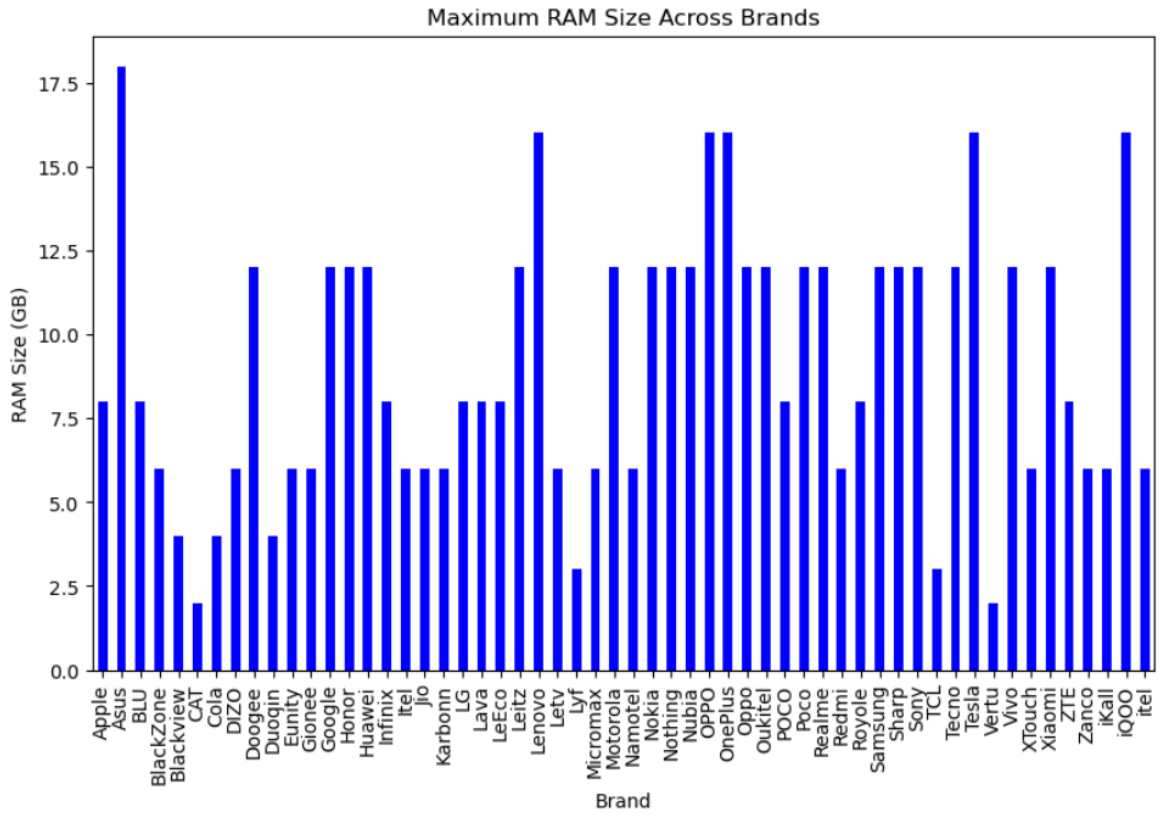

```
# Histogram for the "price" column
plt.figure(figsize=(20, 10))
y=[50000,100000,150000,200000,250000,300000,
   350000,400000,450000,500000,550000,600000,650000]
plt.hist(df['price'],40, color='blue', edgecolor='black')

plt.xticks(y)
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.title('Histogram of Price Column')
plt.show()
```



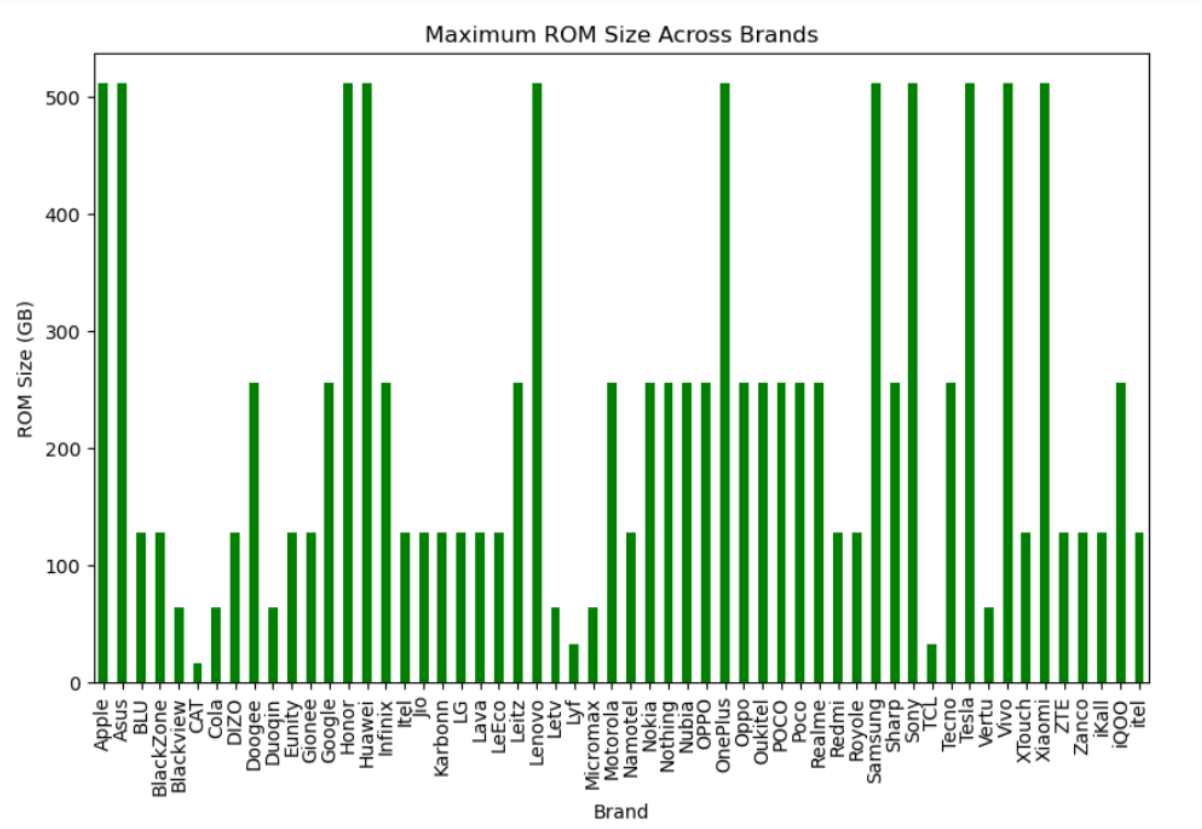
- Bivariate Analiysis

Here we will compare Ram and Rom size across the brands

```
max_values_df = df.groupby('brand')[['ram', 'rom']].max()

# Plotting RAM
fig, ax_ram = plt.subplots(figsize=(10, 6))
max_values_df['ram'].plot(kind='bar', ax=ax_ram, color='blue')
ax_ram.set_title('Maximum RAM Size Across Brands')
ax_ram.set_xlabel('Brand')
ax_ram.set_ylabel('RAM Size (GB)')

# Plotting ROM
fig, ax_rom = plt.subplots(figsize=(10, 6))
max_values_df['rom'].plot(kind='bar', ax=ax_rom, color='green')
ax_rom.set_title('Maximum ROM Size Across Brands')
ax_rom.set_xlabel('Brand')
ax_rom.set_ylabel('ROM Size (GB)')

plt.show()
```

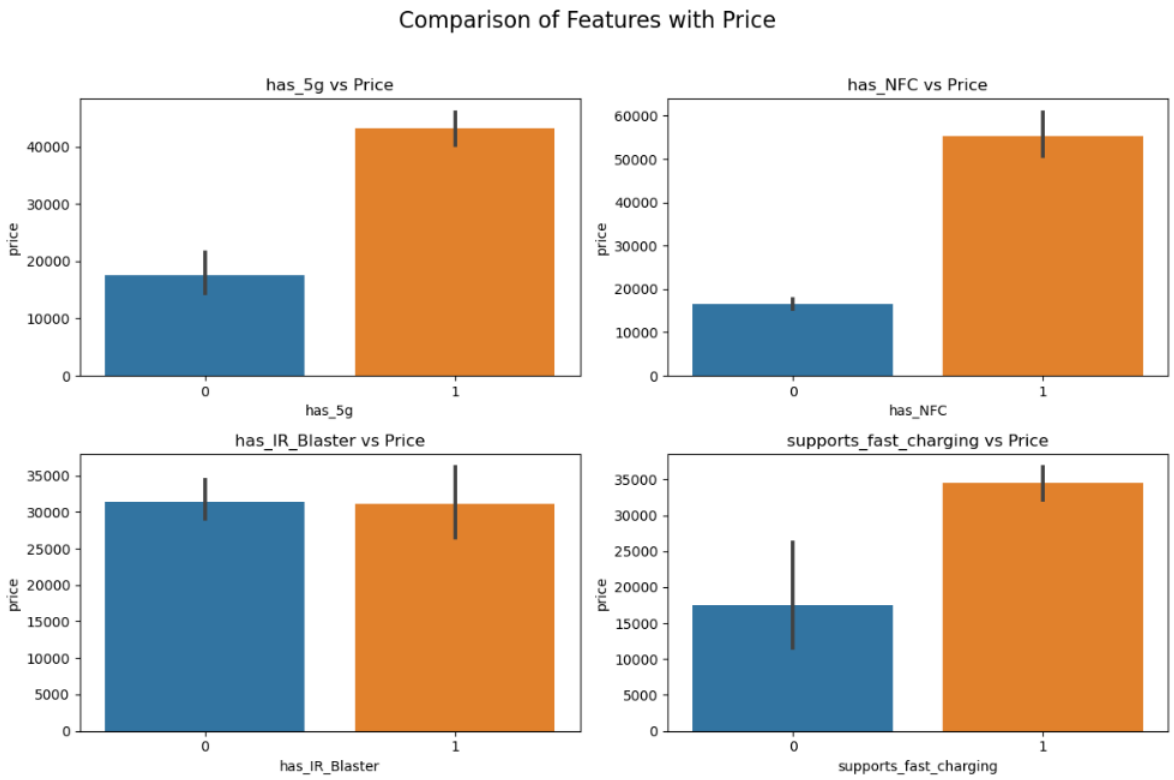


6130 Jatin Gangare SY.IT A



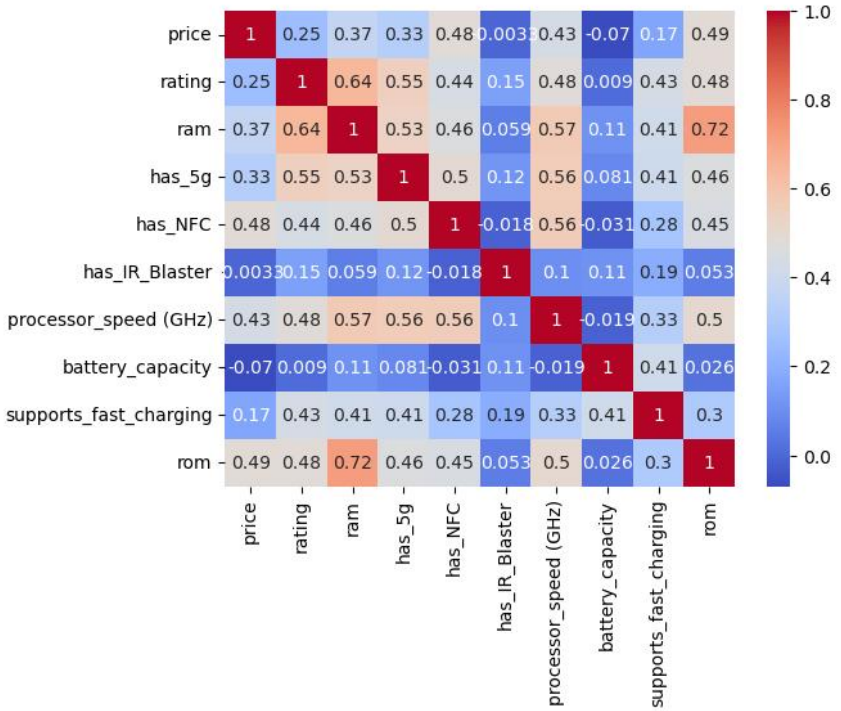
Here we will analyse the price of smartphones which has features like 5g, NFC, IR Blaster and Fast charging support

```
# Assuming df is your DataFrame
fig, axes = plt.subplots(2, 2, figsize=(12, 8))
fig.suptitle('Comparison of Features with Price', fontsize=16)

# Bar plots for each feature against price
sns.barplot(x='has_5g', y='price', data=df, ax=axes[0, 0])
axes[0, 0].set_title('has_5g vs Price')

sns.barplot(x='has_NFC', y='price', data=df, ax=axes[0, 1])
axes[0, 1].set_title('has_NFC vs Price')

sns.barplot(x='has_IR_Blaster', y='price', data=df, ax=axes[1, 0])
axes[1, 0].set_title('has_IR_Blaster vs Price')

sns.barplot(x='supports_fast_charging', y='price', data=df, ax=axes[1, 1])
axes[1, 1].set_title('supports_fast_charging vs Price')

# Adjust layout
plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()
```

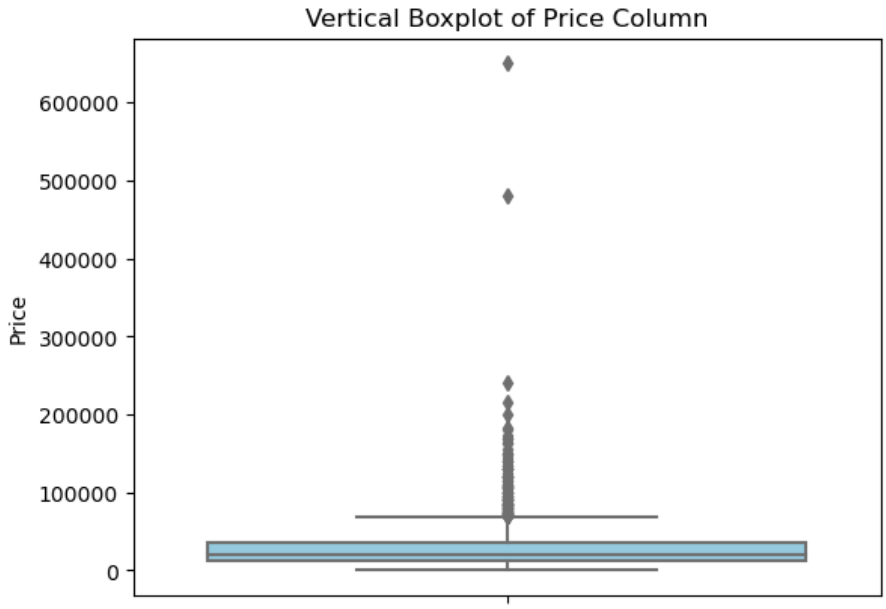


Here we correlate all numerical values

```
numeric_columns = df.select_dtypes(include=['float64', 'int64']).columns
correlation_matrix = df[numeric_columns].corr()
# Plot the correlation matrix heatmap
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.show()
```

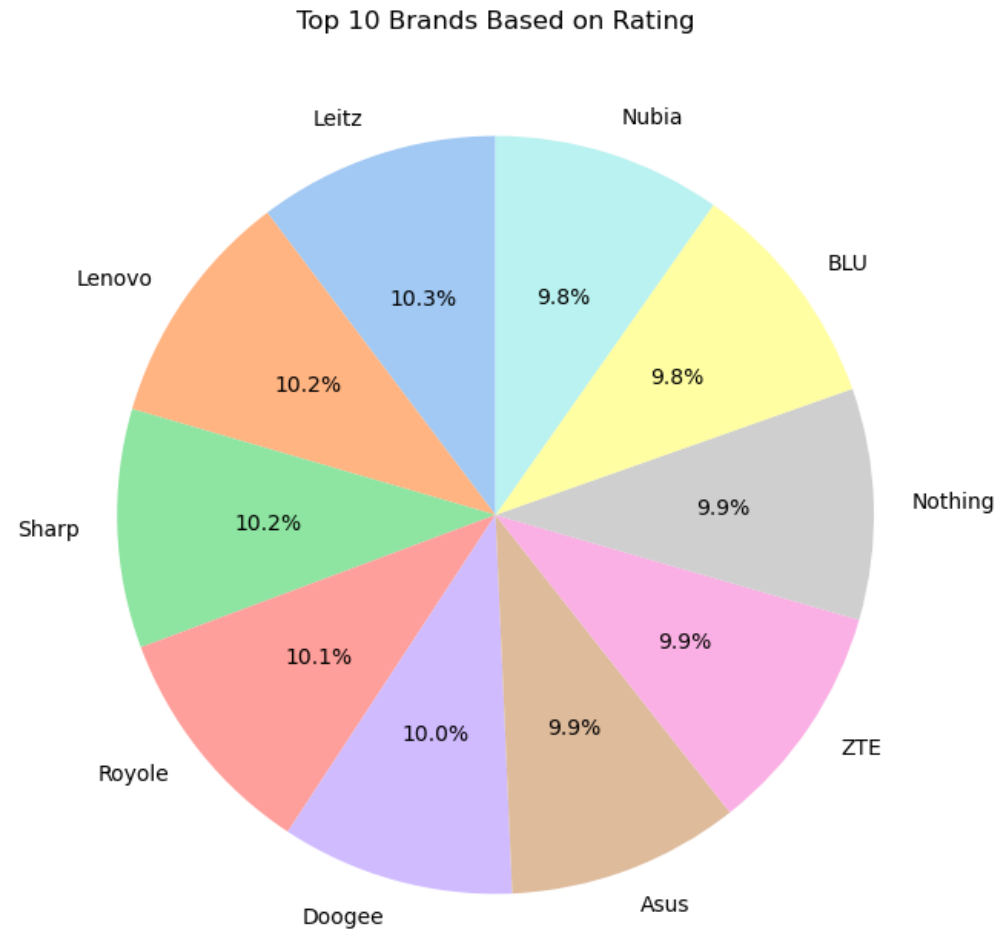



```
#Boxplot for the "price" column
sns.boxplot(y='price', data=df, color='skyblue', orient='v')
plt.ylabel('Price')
plt.title('Vertical Boxplot of Price Column')
plt.show()
```



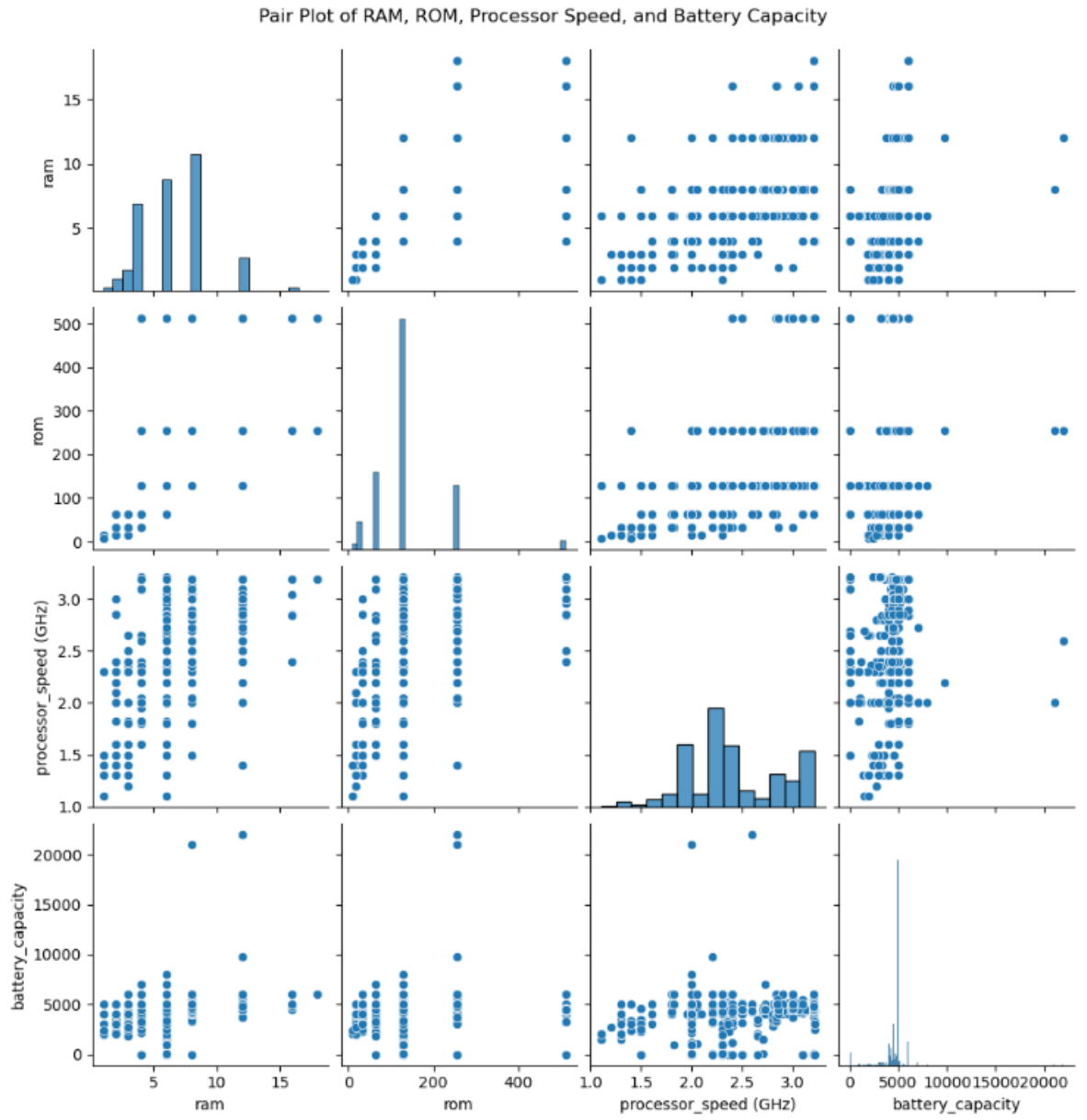
Here we compare top 10 rated brands

```
# Calculate the average rating for each brand
brand_avg_rating = df.groupby('brand')['rating'].mean().sort_values(ascending=True)
top_10_brands = brand_avg_rating.nlargest(10)

# Pie chart for the top 10 brands based on rating
plt.figure(figsize=(8, 8))
plt.pie(top_10_brands, labels=top_10_brands.index, autopct='%1.1f%%', startangle=90, colors=sns.color_palette('pastel'))
plt.title('Top 10 Brands Based on Rating')
plt.show()
```



- Multivariate Analysis

```
selected_columns = ['ram', 'rom', 'processor_speed (GHz)', 'battery_capacity']

# Create a pair plot
sns.pairplot(df[selected_columns])
plt.suptitle('Pair Plot of RAM, ROM, Processor Speed, and Battery Capacity', y=1.02)
plt.show()
```



Conclusion :

1. Data Quality:

The dataset has undergone a thorough cleaning process, ensuring that there are no missing values or inconsistencies in the key columns such as model, price, rating, etc.
Outliers or unusual values in the price, rating, and other numerical columns have been addressed, enhancing the overall reliability of the dataset.

2. Patterns and Trends:

The dataset reveals a diverse range of smartphone models, spanning various brands and specifications.
A positive correlation between price and certain features such as RAM, battery capacity, and processor speed is observed, indicating that higher-priced phones tend to have better hardware specifications.

3. Data Relationships:

There is a correlation between RAM size and overall rating, suggesting that smartphones with larger RAM capacities are generally better-rated by users.
The presence of 5G capability seems to be associated with higher prices, reflecting the market trend of pricing 5G-enabled devices at a premium.

4. Insights and Findings:

Smartphones with NFC and IR blaster features appear to be less common, potentially offering a market opportunity for brands looking to differentiate their products.
As the result smartphones having there features have high prices except for “Xiaomi” which provides feature like “IR_Blaster” is midrange Smartphones.
The positive correlation between battery capacity and fast-charging support highlights the importance of long-lasting batteries and quick charging technology to users.

5. Limitations:

The dataset might not capture the latest smartphone models or recent market trends, especially if there have been significant releases after the data collection period.
External factors such as user preferences, marketing strategies, and regional variations are not explicitly considered in this dataset.

6. Recommendations:

Brands looking to target a wider audience may consider incorporating NFC and IR blaster features, as these seem to be less common but potentially attractive to certain users.
Giving best features in midrange(i.e. under 25000) smartphone will definitely increase the sales as we can see in the brands like “Samsung” and ‘Xiaomi”.
Continuous monitoring of market trends and regular updates to the dataset will enhance its relevance for strategic decision-making.

7. Visualizations:

Utilize scatter plots to visually represent the relationships between price and features like RAM, battery capacity, and processor speed.
Bar charts can be used to display the distribution of smartphones across brands and the prevalence of specific features like 5G, NFC, and IR blaster.

8. Overall Implications:

Smartphone manufacturers can use the insights from this dataset to inform product development, pricing strategies, and marketing efforts.
The dataset provides valuable information for consumers looking to make informed decisions based on their preferences and budget constraints.
Brands can optimize their pricing strategy by considering the observed correlations between price and features. This can help in setting competitive yet profitable price points for different smartphone configurations.
The dataset allows brands to assess their market positioning based on factors such as price, features, and user ratings.