DATA SCIENCE

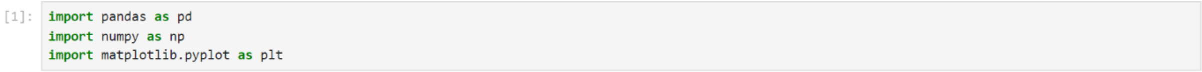
ASSIGNMENT – 1

Name: Divyakumar Patel (1226279)

Introduction:-

Data analysis is the process of inspecting, cleansing, transforming, and modeling data with the goal of discovering useful information, informing conclusions, and supporting decision-making. In this assignment, I gathered a smart phones data from Kaggle. After, I did data preprocessing using Pandas python library. Then, I did data modelling and analytics. In last, I present the data set in PDF histograms and CDF. Below are the steps :

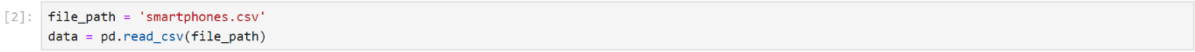
1. First install the libraries:



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

2. Loading the data set

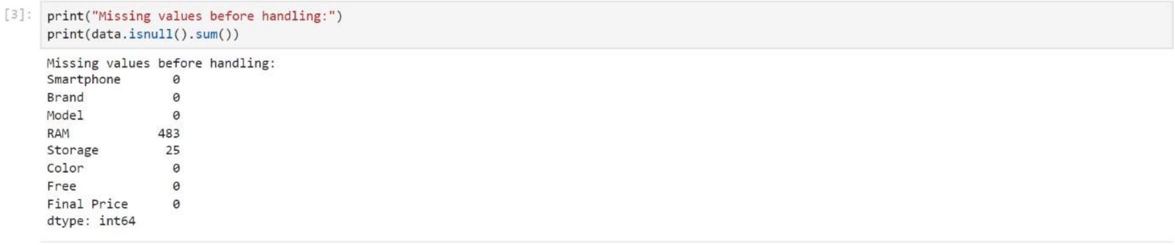
   I used smartphones dataset. The parameters of the data set is brand, model, RAM, storage, color, free, final price.



```
[2]: file_path = 'smartphones.csv'
     data = pd.read_csv(file_path)
```

3. Data Pre-Processing

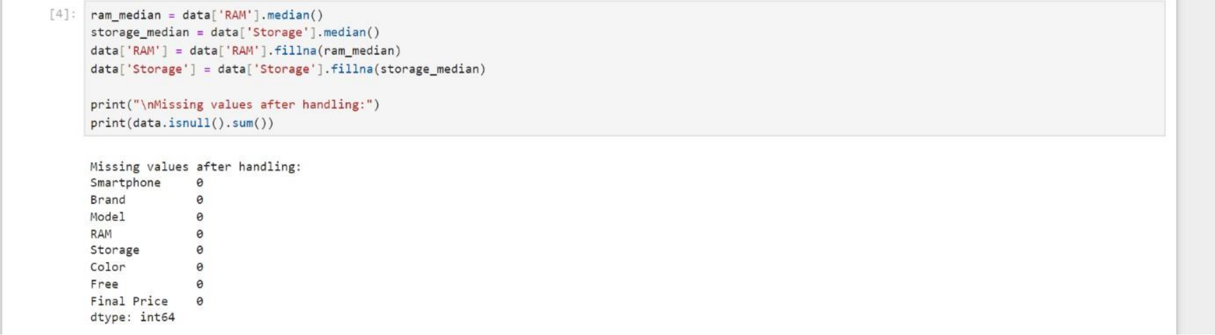
   The first prints the number of missing values for each coloum using “data.isnull().sum()”



```
[3]: print("Missing values before handling:")
     print(data.isnull().sum())
```

```
Missing values before handling:
Smartphone       0
Brand            0
Model            0
RAM            483
Storage         25
Color            0
Free             0
Final Price      0
dtype: int64
```

Missing values in the 'RAM' and 'Storage' columns are filled with their respective medians. This is a common technique to handle missing data without distorting the distribution too much.



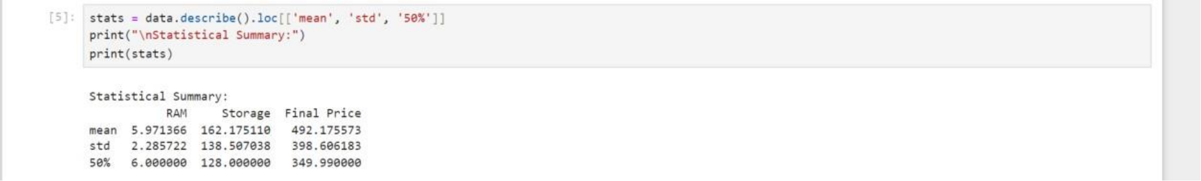
```
[4]: ram_median = data['RAM'].median()
     storage_median = data['Storage'].median()
     data['RAM'] = data['RAM'].fillna(ram_median)
     data['Storage'] = data['Storage'].fillna(storage_median)

     print("\nMissing values after handling:")
     print(data.isnull().sum())
```

```
Missing values after handling:
Smartphone     0
Brand          0
Model          0
RAM            0
Storage        0
Color          0
Free           0
Final Price    0
dtype: int64
```

4. Statistical Analysis

A summary of statistics (mean, standard deviation, and median) for all numerical columns in the dataset is calculated using “data.describe()” and displayed.



```
[5]: stats = data.describe().loc[['mean', 'std', '50%']]
     print("\nStatistical Summary:")
     print(stats)
```

```
Statistical Summary:
           RAM     Storage  Final Price
mean  5.971366  162.175110   492.175573
std   2.285722  138.507038   398.606183
50%   6.000000  128.000000   349.990000
```

5. Visualization

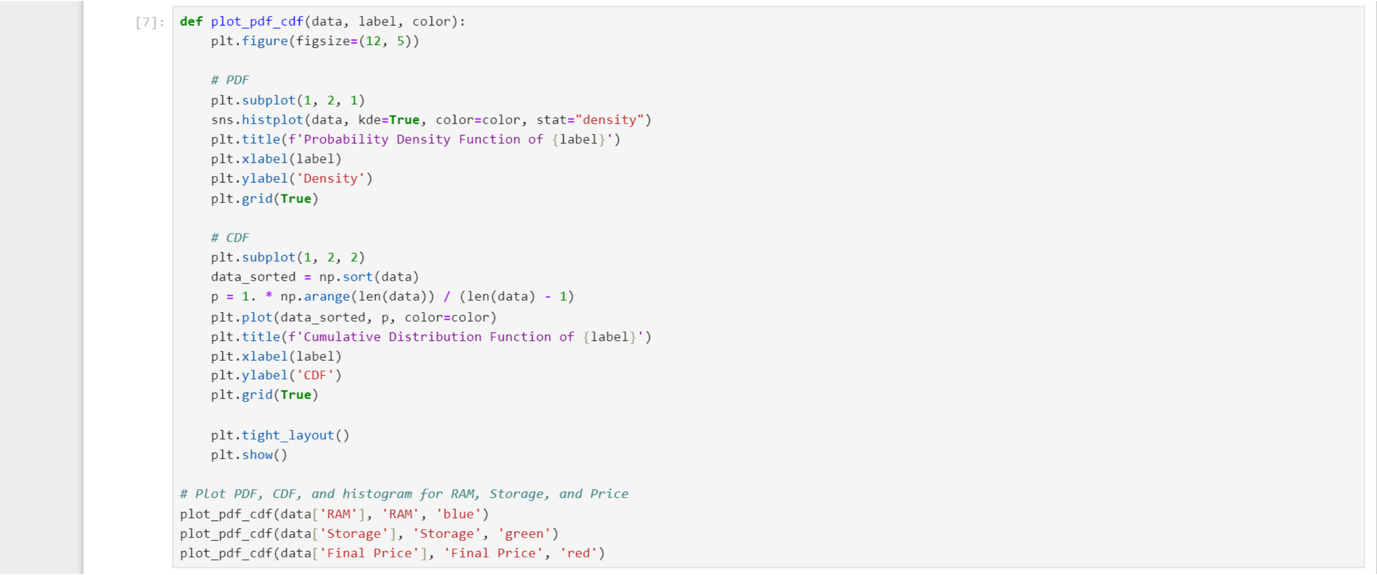
PDF:

Each histogram sets the figure size, specifies the number of bins, chooses a color, and includes grid lines for better readability. The histograms help in understanding how the data is spread across different values. Three histograms are created to visualize the distributions of 'RAM', 'Storage', and 'Final Price'.

CDF:

A function named plot_cdf is defined to plot CDFs for given data. The CDF plots for 'RAM', 'Storage', and 'Final Price' are generated by:

- Sorting the data and calculating the percentile rank for each value.
- Plotting these values to see the cumulative distribution, which helps in understanding the probability of a variable falling below a certain value.



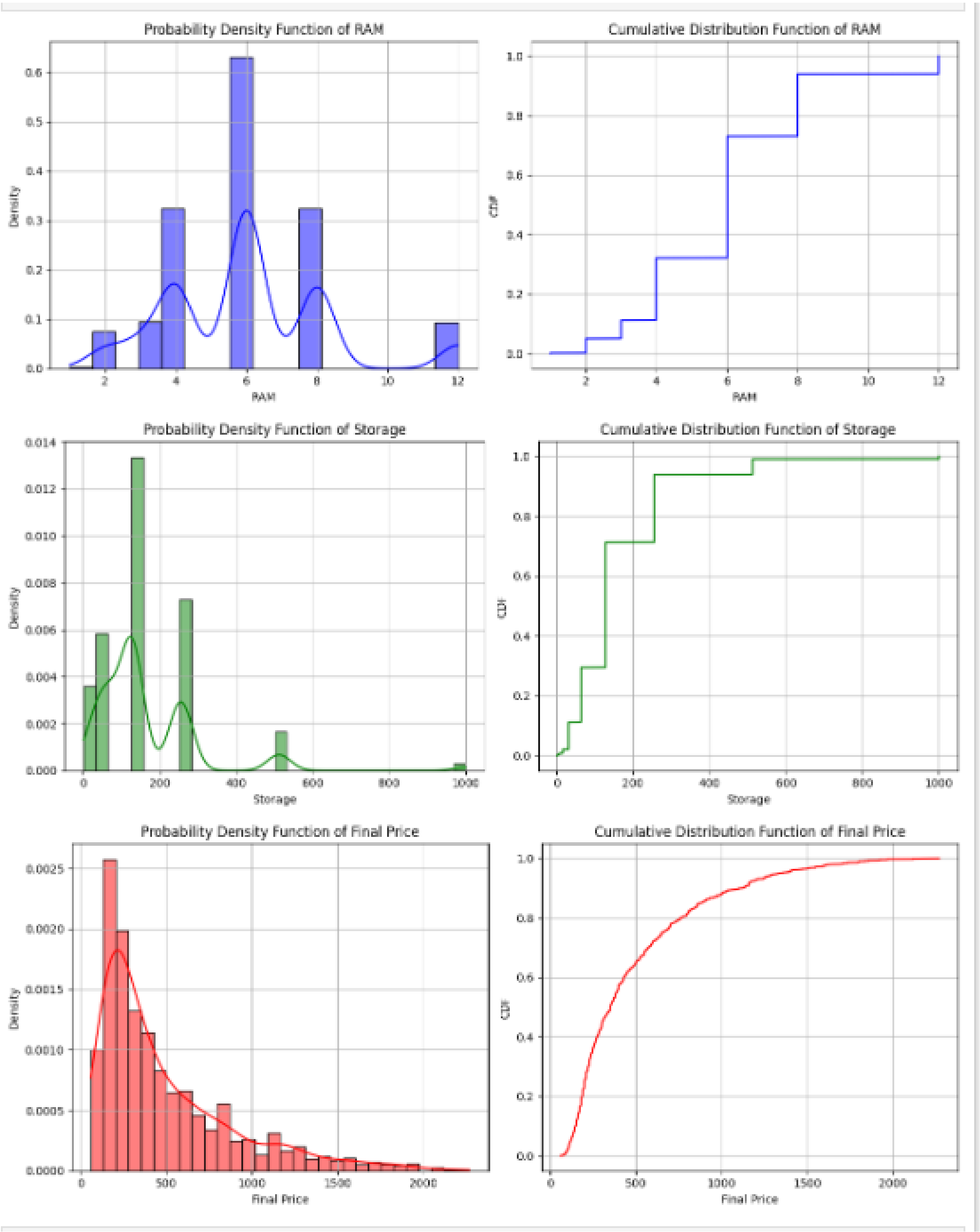
```
[7]: def plot_pdf_cdf(data, label, color):
         plt.figure(figsize=(12, 5))

         # PDF
         plt.subplot(1, 2, 1)
         sns.histplot(data, kde=True, color=color, stat="density")
         plt.title(f'Probability Density Function of {label}')
         plt.xlabel(label)
         plt.ylabel('Density')
         plt.grid(True)

         # CDF
         plt.subplot(1, 2, 2)
         data_sorted = np.sort(data)
         p = 1. * np.arange(len(data)) / (len(data) - 1)
         plt.plot(data_sorted, p, color=color)
         plt.title(f'Cumulative Distribution Function of {label}')
         plt.xlabel(label)
         plt.ylabel('CDF')
         plt.grid(True)

         plt.tight_layout()
         plt.show()

     # Plot PDF, CDF, and histogram for RAM, Storage, and Price
     plot_pdf_cdf(data['RAM'], 'RAM', 'blue')
     plot_pdf_cdf(data['Storage'], 'Storage', 'green')
     plot_pdf_cdf(data['Final Price'], 'Final Price', 'red')
```


Probability Density Function of RAM

Cumulative Distribution Function of RAM

Probability Density Function of Storage

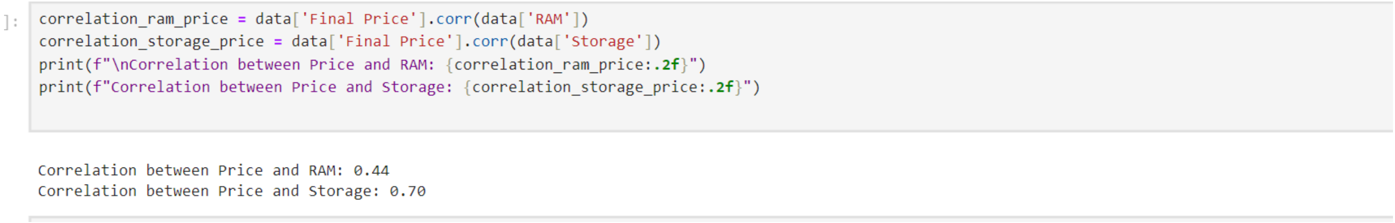
Cumulative Distribution Function of Storage

Probability Density Function of Final Price

Cumulative Distribution Function of Final Price

6. Correlation Analysis:

Now, we calculate and interpret the Pearson correlation coefficients between price and RAM, and price and storage to quantify the strength and direction of these relationships.

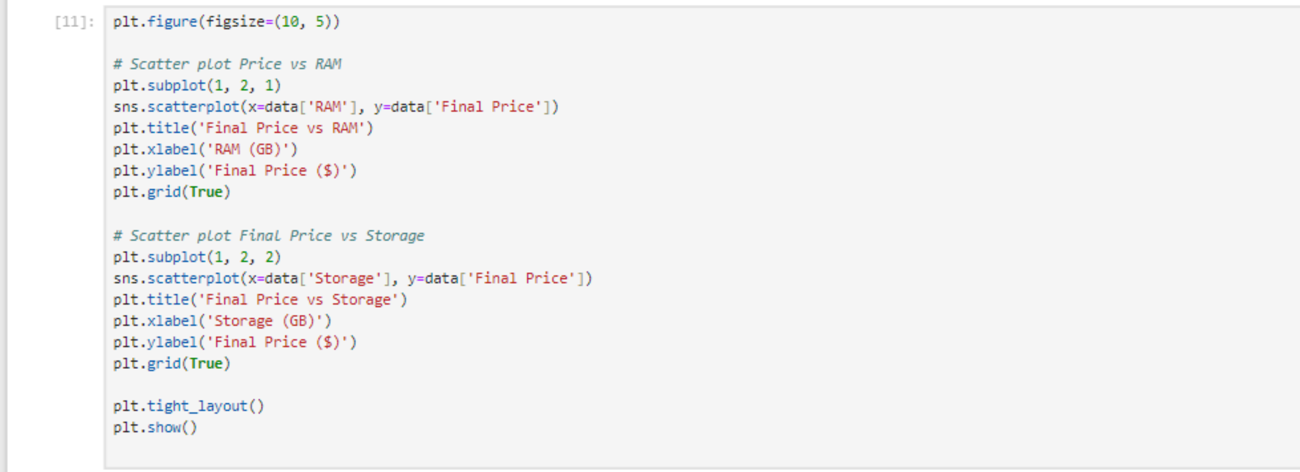


```
correlation_ram_price = data['Final Price'].corr(data['RAM'])
correlation_storage_price = data['Final Price'].corr(data['Storage'])
print(f"\nCorrelation between Price and RAM: {correlation_ram_price:.2f}")
print(f"Correlation between Price and Storage: {correlation_storage_price:.2f}")
```

```
Correlation between Price and RAM: 0.44
Correlation between Price and Storage: 0.70
```

7. Hypothesis statement

By testing these hypotheses, we aim to understand if and how the hardware specifications (RAM and storage) influence the pricing of smartphones, thereby providing insights into the factors that contribute to the pricing strategies of smartphone manufacturers.

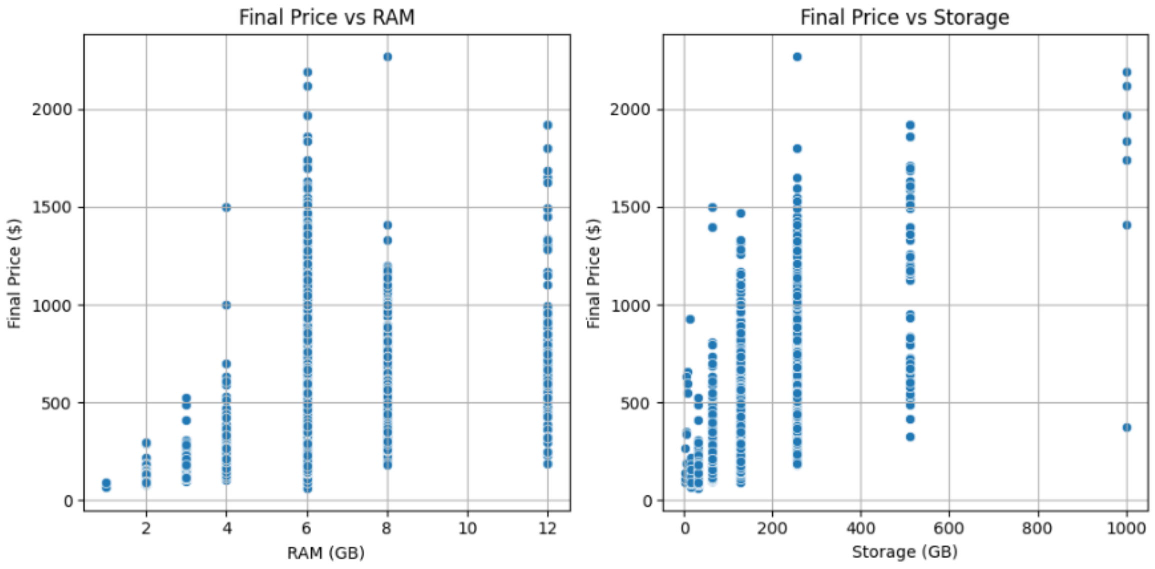


```
plt.figure(figsize=(10, 5))

# Scatter plot Price vs RAM
plt.subplot(1, 2, 1)
sns.scatterplot(x=data['RAM'], y=data['Final Price'])
plt.title('Final Price vs RAM')
plt.xlabel('RAM (GB)')
plt.ylabel('Final Price ($)')
plt.grid(True)

# Scatter plot Final Price vs Storage
plt.subplot(1, 2, 2)
sns.scatterplot(x=data['Storage'], y=data['Final Price'])
plt.title('Final Price vs Storage')
plt.xlabel('Storage (GB)')
plt.ylabel('Final Price ($)')
plt.grid(True)

plt.tight_layout()
plt.show()
```



8. Saving the Processed Data

Finally, the processed data is saved to a new CSV file named 'processed_smartphones.csv'. This includes all the modifications and imputations done to the dataset. data.to_csv('processed_smartphones.csv', index=False