**Apple Sales Analysis Project  
  
This project aims to analyze sales data for iPhones to uncover trends, insights, and patterns that enhance our understanding of customer behavior and product performance. We will utilize data manipulation and visualization libraries to explore various aspects of the dataset, including pricing, discounts, customer ratings, and reviews.  
Key Questions  
The analysis will address the following questions:  
What are the top 10 highest-rated iPhones on Flipkart in India?  
How many ratings do the highest-rated iPhones on Flipkart have?  
Which iPhone has the highest number of reviews on Flipkart?  
What is the relationship between the sale price of iPhones and the number of ratings on Flipkart?  
What is the relationship between the discount percentage and the number of ratings for iPhones on Flipkart?  
Can we identify the least and most expensive iPhones in the Indian market and their specifications?   
  
# Importing necessary libraries  
import pandas as pd # For data manipulation and analysis using DataFrames.  
  
import numpy as np # For numerical computing and handling arrays.  
  
import plotly.express as px # For easy and quick interactive visualizations.  
  
import plotly.graph\_objects as go # For creating customizable and complex visualizations.  
  
# Loading the Dataset  
  
In this section, we will load the dataset containing information about various Apple products, specifically iPhones. This dataset is in CSV (Comma-Separated Values) format, which is commonly used for storing structured data. Using the Pandas library, we will read this CSV file into a DataFrame, allowing us to manipulate and analyse the data efficiently.  
  
# Loading the dataset  
df = pd.read\_csv('apple\_products.csv') # Reads the CSV file into a DataFrame named 'df' for further analysis.  
  
  
# Displaying the Dataset  
  
In this step, we will print the contents of the DataFrame to the console. This allows us to visually inspect the data and understand its structure, including the columns and the first few rows. This is an essential step in data analysis to verify that the dataset has been loaded correctly.  
  
  
# Displaying the dataset  
print(df) # Outputs the entire DataFrame to the console for inspection.**

**# Previewing the Dataset**

**Here, we will use the `head()` method to display the first few rows of the DataFrame. This is helpful for quickly checking the structure and contents of the dataset without overwhelming ourselves with too much data. By default, `head()` shows the first five rows, but this can be adjusted if needed.**

**# Previewing the dataset**

**print(df.head()) # Displays the first five rows of the DataFrame for a quick overview.**

**# Viewing the Last Rows of the Dataset**

**In this step, we will use the `tail()` method to display the last few rows of the DataFrame. This is useful for checking how the dataset ends, which can help identify any potential issues or patterns toward the conclusion of the data. By default, `tail()` also shows the last five rows.**

**# Viewing the last rows of the dataset**

**print(df.tail()) # Displays the last five rows of the DataFrame for a quick overview.**

**# Generating Descriptive Statistics**

**In this step, we will use the `describe()` method to generate summary statistics for the numerical columns in the DataFrame. This includes metrics such as count, mean, standard deviation, minimum, maximum, and quartiles. Descriptive statistics provide a quick overview of the data distribution and can help identify any anomalies or trends.**

**# Generating descriptive statistics**

**print(df.describe()) # Outputs summary statistics for numerical columns in the DataFrame.**

**# Checking for Missing Values**

**In this step, we will use the `isnull().sum()` method to check for any missing values in the DataFrame. This is an important step in data cleaning, as missing values can affect the quality of our analysis. By identifying columns with null values, we can decide how to handle them—whether to fill them, drop them, or take other appropriate actions.**

**# Checking for missing values**

**df.isnull().sum() # Returns the count of missing values in each column of the DataFrame.**

**# Conclusion on Missing Values**

**#After checking the dataset for missing values, we found no null values in any columns.**

**#This indicates that the data is complete and requires no additional cleaning for missing entries.**

**#A complete dataset is essential for accurate analysis, as it ensures that our insights and findings are based on all available data.**

**# Top 10 Highest-Rated iPhones on Flipkart India?**

**# Identifying the Top 10 Highest-Rated iPhones**

**In this section, we will identify the top 10 highest-rated iPhones from the dataset. We will sort the DataFrame by the "Star Rating" column in descending order to prioritize higher ratings. After sorting, we will select the top 10 entries and display the names of these products.**

**# Identifying the top 10 highest-rated iPhones**

**highest\_rated = df.sort\_values(by=["Star Rating"], ascending=False) # Sorts the DataFrame by "Star Rating" in descending order.**

**highest\_rated = highest\_rated.head(10) # Selects the top 10 highest-rated iPhones.**

**print(highest\_rated['Product Name']) # Displays the names of the top 10 highest-rated iPhones.**

**# How many ratings do the highest rating iPhones on Flipkart have?**

**# Visualizing the Ratings of Top 10 Highest-Rated iPhones**

**In this section, we will visualize the number of ratings for the top 10 highest-rated iPhones using a bar chart. We will extract the product names and their corresponding counts to create a clear and informative visualization. This helps us understand which iPhones not only have high ratings but also have significant customer feedback, indicating their popularity.**

**# Counting the number of ratings for the highest-rated iPhones**

**iphones = highest\_rated['Product Name'].value\_counts() # Counts occurrences of each product name in the highest-rated DataFrame.**

**label = iphones.index # Extracts the product names to use as labels for the bar chart.**

**counts = highest\_rated["Number Of Ratings"] # Gets the number of ratings for each of the highest-rated iPhones.**

**# Creating a bar chart to visualize the ratings with increased bar height**

**figure = px.bar(**

**highest\_rated,**

**x=label, # X-axis: Product Names**

**y=counts, # Y-axis: Number of Ratings**

**title='Number of Ratings for Top 10 Highest-Rated iPhones', # Chart title**

**labels={'x': 'iPhone Model', 'y': 'Number of Ratings'}, # Axis labels**

**color='Number Of Ratings', # Color based on the number of ratings**

**text=counts, # Display count on top of each bar**

**text\_auto=True, # Automatically display text on the bars**

**template='plotly\_dark' # Use a dark template for better aesthetics**

**)**

**# Adjusting layout to increase bar height and width**

**figure.update\_layout(**

**bargap=0.1, # Decrease the gap between bars to make them appear longer**

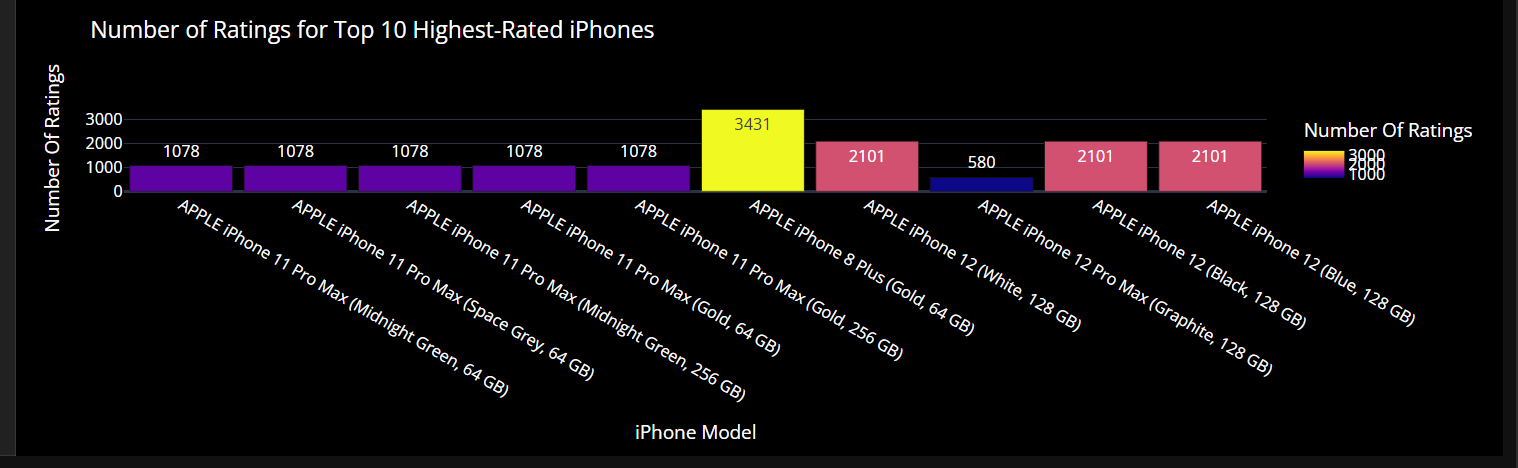
**height=600, # Increase the height of the figure for better visibility**

**plot\_bgcolor='black', # Set background color of the plot area to black**

**paper\_bgcolor='black', # Set background color of the paper to black**

**font\_color='white' # Set font color to white for visibility against dark background**

**)**

**figure.show() # Displays the bar chart**

**# Visualizing the Number of Reviews for Top 10 Highest-Rated iPhones**

**In this section, we will visualize the number of reviews for the top 10 highest-rated iPhones using a bar chart. We will extract the product names and their corresponding review counts to create a clear and informative visualization. This helps us understand the level of customer engagement and feedback for these popular iPhone models.**

# Counting the number of ratings for the highest-rated iPhones

iphones = highest\_rated['Product Name'].value\_counts() # Counts occurrences of each product name in the highest-rated DataFrame.

label = iphones.index # Extracts the product names to use as labels for the bar chart.

counts = highest\_rated["Number Of Ratings"] # Gets the number of ratings for each of the highest-rated iPhones.

# Creating a bar chart to visualize the ratings with increased bar height and details

figure = px.bar(

highest\_rated,

x=label, # X-axis: Product Names

y=counts, # Y-axis: Number of Ratings

title='Number of Ratings for Top 10 Highest-Rated iPhones', # Chart title

labels={'x': 'iPhone Model', 'y': 'Number of Ratings'}, # Axis labels

color='Number Of Ratings', # Color based on the number of ratings

text=counts, # Display count on top of each bar

text\_auto=True, # Automatically display text on the bars

template='plotly\_dark' # Use a dark template for better aesthetics

)

# Adjusting layout to increase bar height and add more details

figure.update\_traces(texttemplate='%{text}', textposition='outside') # Show text outside the bars for clarity

figure.update\_layout(

bargap=0.1, # Decrease the gap between bars to make them appear longer

height=600, # Increase the height of the figure for better visibility

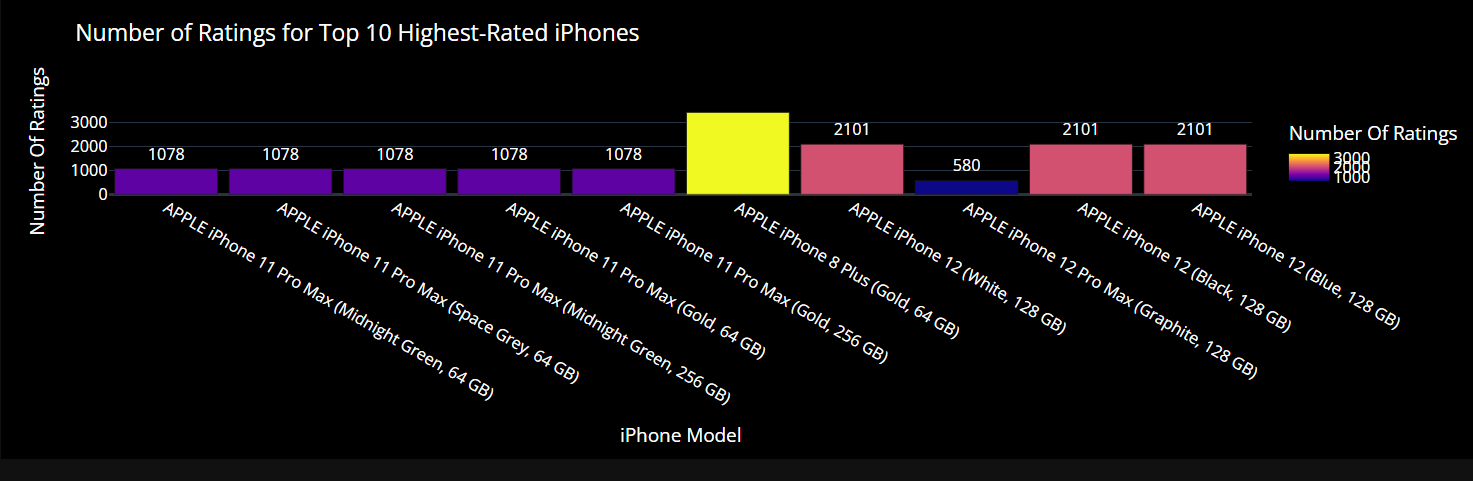
plot\_bgcolor='black', # Set background color of the plot area to black

paper\_bgcolor='black', # Set background color of the paper to black

font\_color='white' # Set font color to white for visibility against dark background

)

figure.show() # Displays the bar chart



**# What is the relationship between the Sales Price of iPhones and the number of ratings on Flipkart?**

# Create a scatter plot to visualize the relationship

figure = px.scatter(

data\_frame=df, # DataFrame containing our data

x="Number Of Ratings", # X-axis: Number of Ratings

y="Sale Price", # Y-axis: Sales Price

size='Discount Percentage', # Size of points based on Discount Percentage

color='Brand', # Color points by Brand for better insights

trendline='ols', # Add a trendline (Ordinary Least Squares) to show the overall trend

title="Relationship Between Sales Price and Number of Ratings", # Title of the plot

labels={

'Number Of Ratings': 'Number of Ratings', # X-axis label

'Sale Price': 'Sales Price (in USD)', # Y-axis label

'Discount Percentage': 'Discount (%)' # Legend label for size

},

hover\_name='Product Name', # Show product name when hovering over points

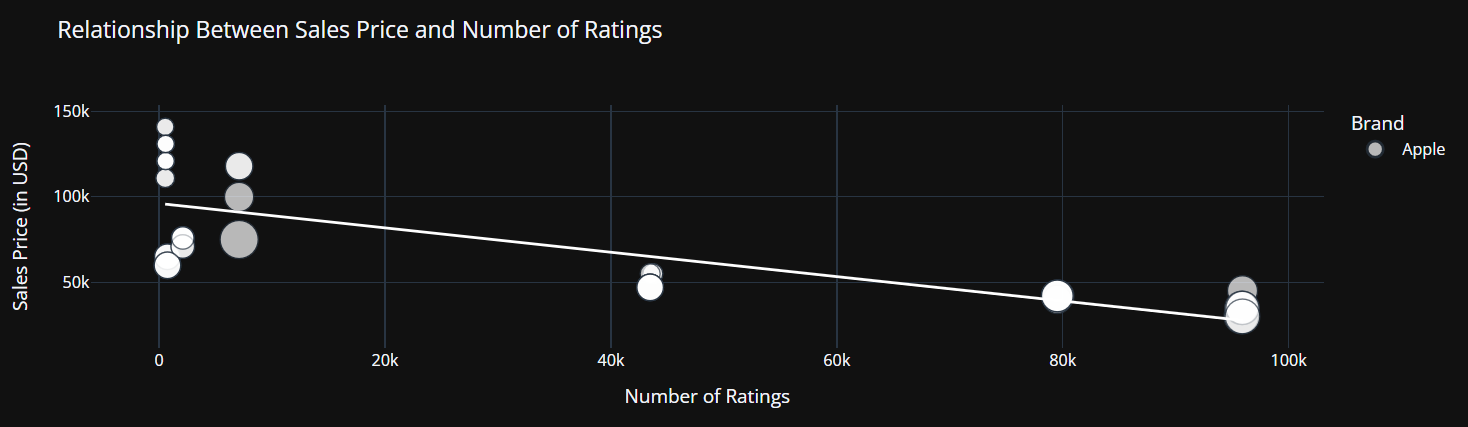
template='plotly\_dark', # Use a dark template for better aesthetics

color\_discrete\_sequence=['white'] # Change point color to white for visibility against a dark background

)

# Show the plot

figure.show() # Display the interactive plot



**# What is the relationship between Discount Percentage and the number of ratings on Flipkart?**

# Creating a scatter plot to visualize the relationship between discount percentage and number of ratings

figure = px.scatter(

data\_frame=df, # DataFrame containing the data

x="Number Of Ratings", # X-axis: Number of Ratings

y="Discount Percentage", # Y-axis: Discount Percentage

size="Sale Price", # Corrected size to use 'Sale Price' instead of 'Sales Price'

trendline="ols", # Add a trendline to show the general relationship (Ordinary Least Squares)

title="Relationship Between Discount and Ratings of iPhones", # Title of the plot

template='plotly\_dark' # Use a dark theme for better aesthetics

)

# Adjusting the layout for the dark theme

figure.update\_layout(

plot\_bgcolor='black', # Set the background color of the plot area to black

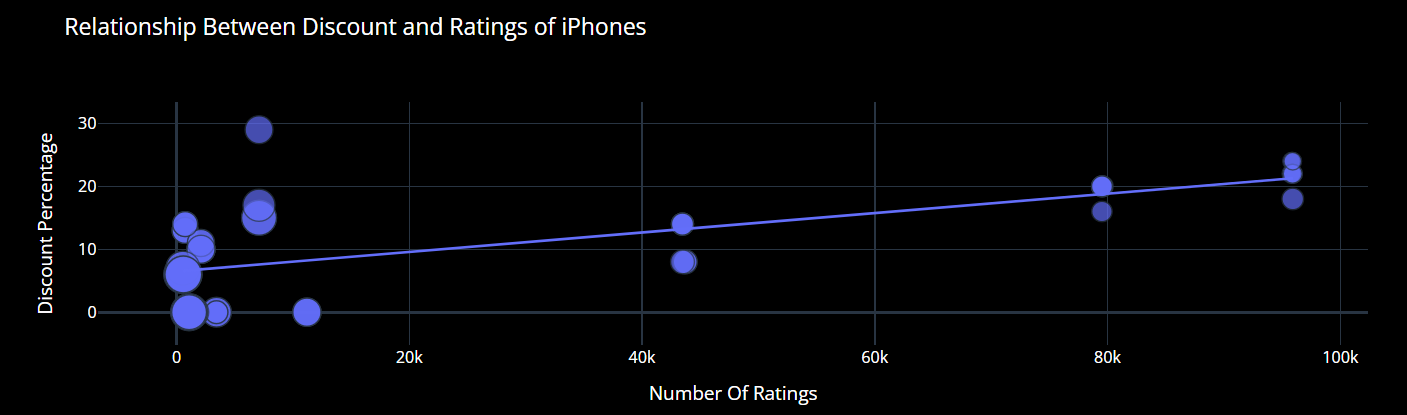
paper\_bgcolor='black', # Set the background color of the entire figure to black

font\_color='white' # Set the font color to white for visibility on a dark background

)

# Display the scatter plot

figure.show()



**# Identifying the Most and Least Expensive iPhones**

**In this section, we identify the most expensive and least expensive iPhones based on the 'Sale Price' from our dataset. This helps in understanding the price range and variations in iPhone models.**

# Find the most expensive iPhone by locating the row with the maximum 'Sale Price'

most\_expensive = df.loc[df['Sale Price'].idxmax()]

# Find the least expensive iPhone by locating the row with the minimum 'Sale Price'

least\_expensive = df.loc[df['Sale Price'].idxmin()]

# Display the most expensive iPhone details

print("Most Expensive iPhone is: \n")

print(most\_expensive)

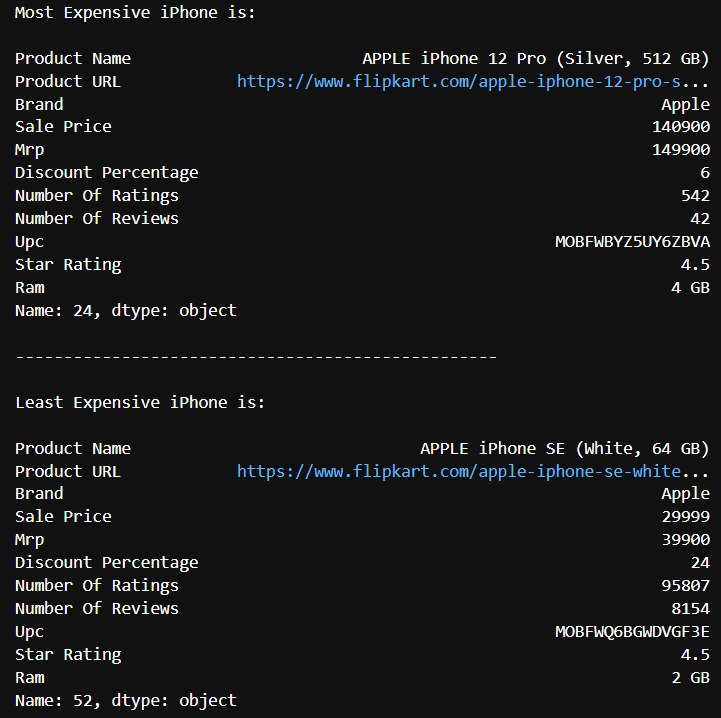
# Add space between the outputs for clarity

print("\n" + "-"\*50 + "\n") # Print a separator line for clarity

# Display the least expensive iPhone details

print("Least Expensive iPhone is: \n")

print(least\_expensive)



## Conclusion

The Apple Sales Analysis Project utilizes data manipulation and visualization libraries, such as \*\*Pandas\*\*, \*\*NumPy\*\*, and \*\*Matplotlib\*\*, to analyze iPhone sales data on Flipkart in India. Key requirements include:

- \*\*Identifying top-rated iPhones\*\* and their reviews.

- \*\*Analyzing relationships\*\* between sale price, ratings, and discount percentages.

- \*\*Determining the least and most expensive iPhones\*\* along with their specifications.

The insights gained from this analysis will enhance understanding of customer behavior and inform marketing strategies, ultimately optimizing sales performance in the smartphone market.