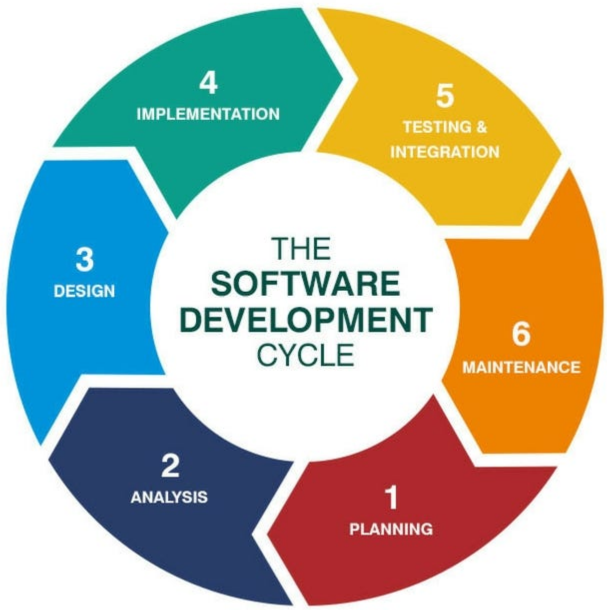
Project Development Guidelines

Software Development Life Cycle (SDLC)

The Software Development Life Cycle (SDLC) is a systematic process for planning, creating, testing, deploying, and maintaining software applications. It provides a structured and standardized approach to software development that helps ensure the quality, efficiency, and effectiveness of the final product. There are several models of SDLC, each with its own set of stages and activities.



Here is a general overview of the typical stages in the SDLC:

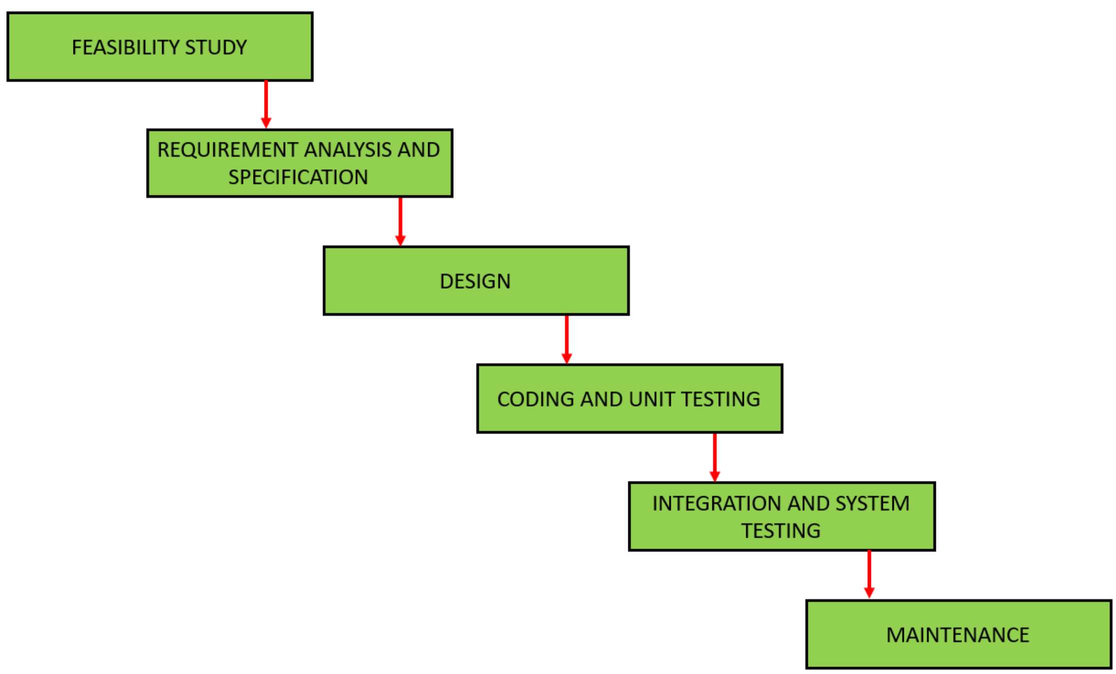
1. **Planning:**
   - Define the project scope, objectives, and requirements.
   - Identify constraints, risks, and resources.
   - Develop a project plan outlining timelines, milestones, and deliverables.
2. **Feasibility Study:**
   - Evaluate the technical, economic, and operational feasibility of the project.
   - Assess potential risks and challenges.
   - Decide whether to proceed with the project or not.
3. **System Design:**
   - Create a high-level design of the system architecture.
   - Specify system components and their relationships.
   - Define data structures, interfaces, and algorithms.
4. **Implementation (Coding):**
   - Write code based on the detailed design specifications.
   - Follow coding standards and best practices.
   - Conduct code reviews to ensure quality and consistency.

5. **Testing:**
    - Develop and execute test cases to ensure the software meets requirements.
    - Identify and fix bugs and issues.
    - Perform various testing types, such as unit testing, integration testing, system testing, and user acceptance testing.
6. **Deployment:**
    - Release the software to the production environment.
    - Ensure a smooth transition from development to production.
    - Provide user training and support.
7. **Maintenance and Support:**
    - Address issues and bugs discovered post-deployment.
    - Make updates and enhancements based on user feedback.
    - Provide ongoing support and maintenance.

It's important to note that these stages can be executed in a sequential manner (as in the Waterfall model) or iteratively and incrementally (as in Agile methodologies). Some common SDLC models include:

1. Waterfall Model

- It is the fundamental model of the software development life cycle. This is a very simple model.



- The waterfall model is not in practice anymore, but it is the basis for all other SDLC models. Because of its simple structure, the waterfall model is easier to use and provides a tangible output.

- In the waterfall model, once a phase seems to be completed, it cannot be changed, and due to this less flexible nature, the waterfall model is not in practice anymore.

2. Agile Model

- The agile model was mainly designed to adapt to changing requests quickly. The main goal of the Agile model is to facilitate quick project completion.

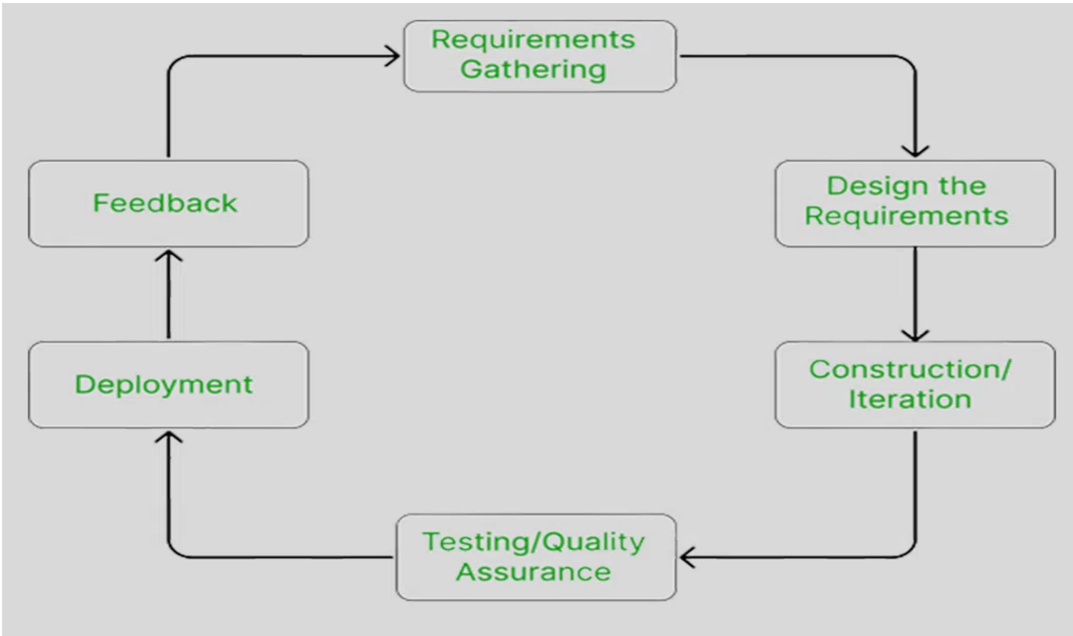
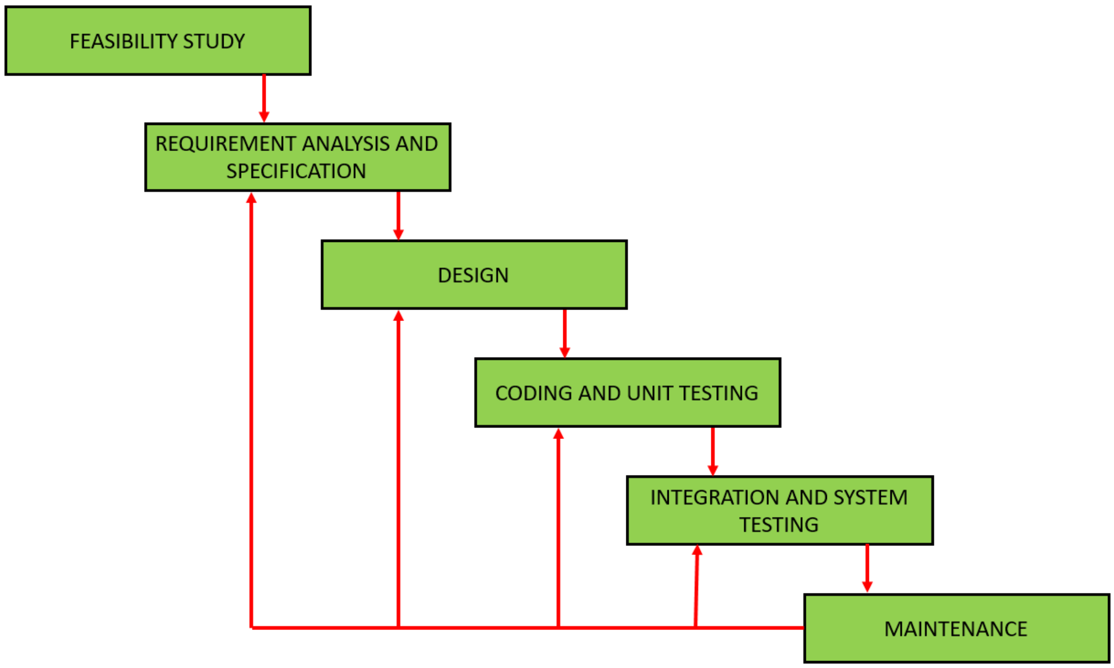


Fig:- Steps in Agile SDLC Model

- The agile model refers to a group of development processes. These processes have some similar characteristics but also possess certain subtle differences among themselves.

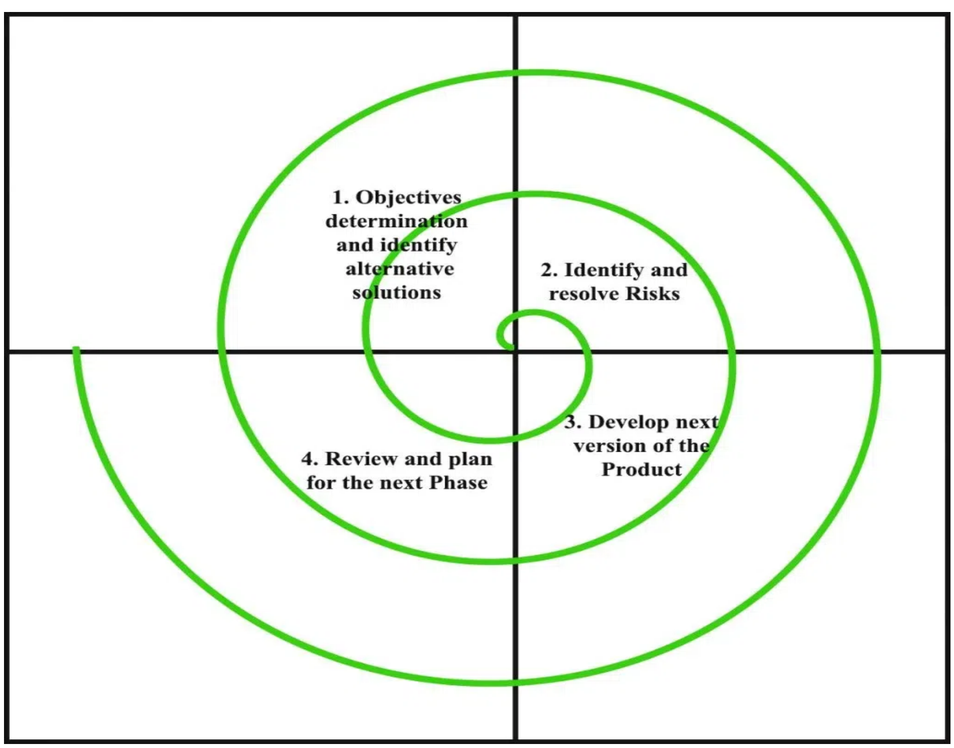
3. Iterative Model

- In the iterative model, each cycle results in a semi-developed but deployable version; with each cycle, some requirements are added to the software, and the final cycle results in the software with the complete requirement specification.



4. Spiral Model

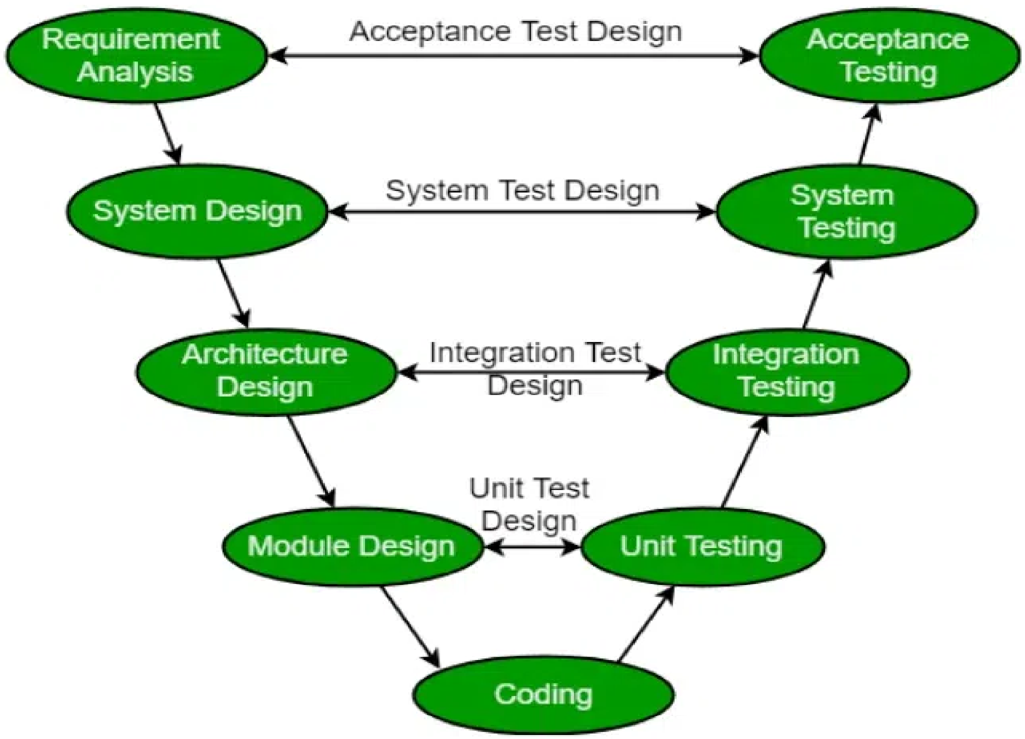
- The spiral model is one of the most crucial SDLC models that provides support for risk handling.



- It has various spirals in its diagrammatic representation; the number of spirals depends upon the type of project.
- Each loop in the spiral structure indicates the Phases of the Spiral model.

5. V-Shaped Model

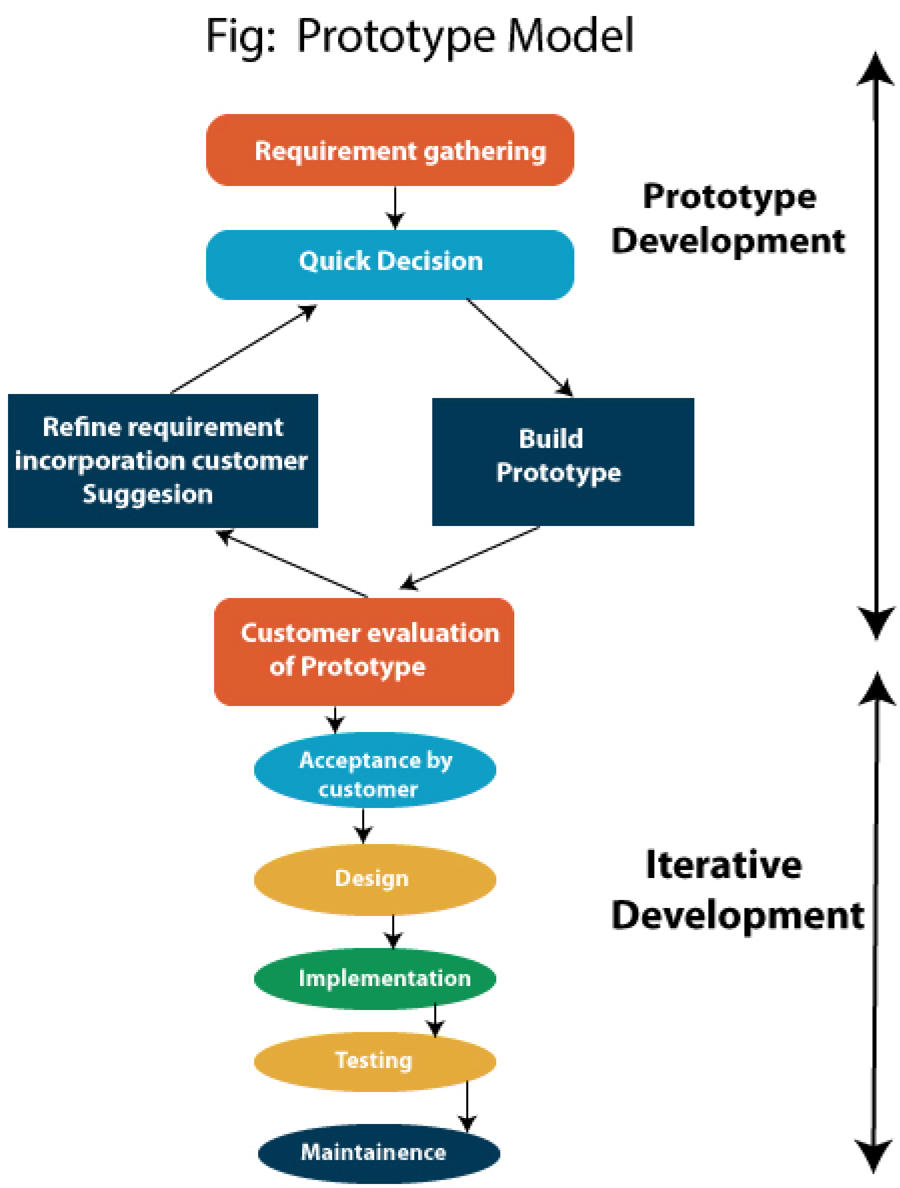
- The V-shaped model is executed in a sequential manner in V-shape. Each stage or phase of this model is integrated with a testing phase.



- After every development phase, a testing phase is associated with it, and the next phase will start once the previous phase is completed, i.e., development & testing. It is also known as the verification or validation model.

<u>6. Prototype</u>

- The prototype model requires that before carrying out the development of actual software, a working prototype of the system should be built.
- A prototype is a toy implementation of the system.
- A prototype usually turns out to be a very crude version of the actual system, possibly exhibiting limited functional capabilities, low reliability, and inefficient performance as compared to actual software.

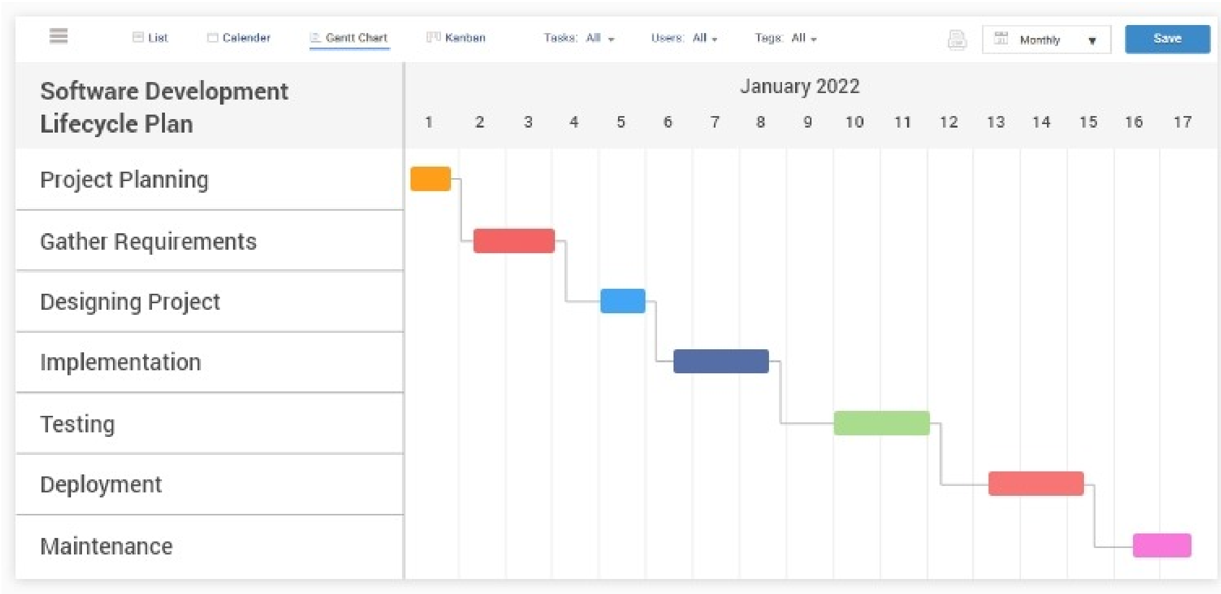


Project Timeline
A schedule for your entire project from inception to completion. It will break your entire project into smaller tasks and milestones, with a deadline assigned to each

Project timelines give an opportunity to:
1. Organize their tasks
2. Show when in the project the tasks start
3. View task deadlines
4. Link dependent tasks
5. Break the project into phases
6. Identify team members assigned to a task

You can use any tool or available software or even Excel for creating a project timeline.

Example:



<u>Steps to be followed</u>

1. Research on the type of real world projects that you can make by using Python
2. Decide on your project scenario
3. Select your project development model
4. Decide if it will be an individual project or a group project
5. Define the components to be created for your project
6. Create a project timeline with roles and responsibilities
7. Start development as per the timeline and your chosen project development model

Scenario:

Product Sales Analysis and Visualisations using Python

Libraries used [Pandas, Numpy, Matplotlib, Seaborn]

Introduction

Every modern company that engages in online sales or maintains a specialized e-commerce website now aims to maximize its throughput in order to determine precisely what their clients need in order to increase their chances of sales.

What is Sales Analysis?

For each product sold by your business, it is recommended that you perform a product sales analysis to compare the profit contribution of different products. Product sales analysis is a judgment on the market performance of a product.

How to perform Sales Analysis?

Product sales data analysis provides a wealth of intelligence about your Product's sales strategy, the performance of your team, and much more. It’s a competitive advantage you can’t afford to miss out on. So let’s get started with the basics.

Purpose of Analysis
The purpose of a product analysis report can be broadly broken down into three major facets:
1. Internal Analysis: which focuses on how the business can better improve, tweak and market your product.

2. External Analysis: which focuses on your potential customers, analyzing how you can convince them that your product is worth buying, and why they should choose it over a similar competitor’s product.

3. Cost Analysis: which focuses on the end-to-end costs involved from manufacturing to sale — allowing you to analyze where you can potentially cut costs while still maintaining the quality of your product.

Things to consider before doing a Sales Analysis

i.) Understanding the Business Model

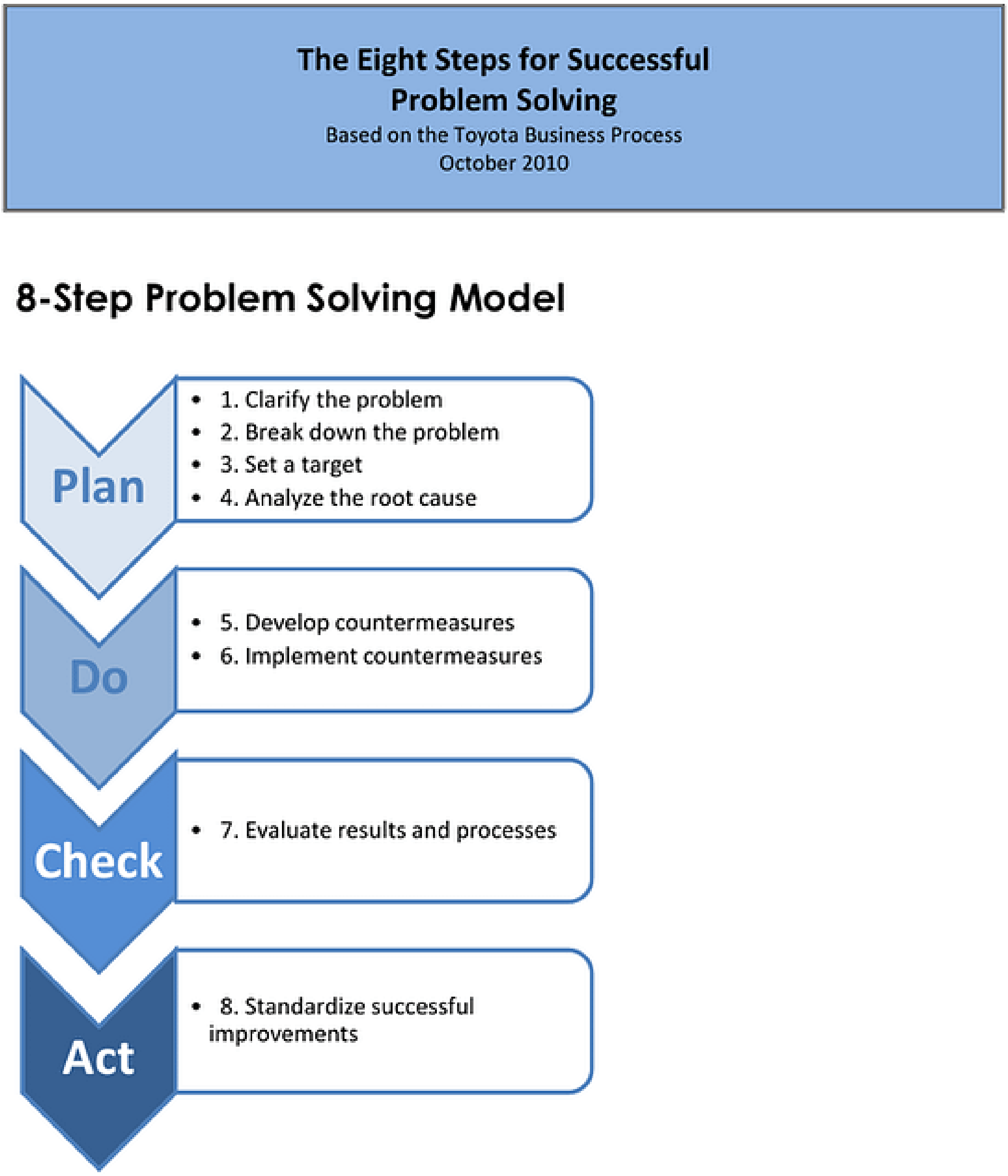
Business model refers to a company's plan for making a profit. It identifies the products or services the business plans to sell, its identified target market, and any anticipated expenses.

ii.) Problem we are trying to solve (Problem Analysis)

Problem analysis is the process of understanding real-world problems and user's needs and proposing solutions to meet those needs. The goal of problem analysis is to gain a better understanding of the problem being solved before developing a solution.

Some important suggestions for creating problem trees

- Involve stakeholders who can contribute relevant technical and local knowledge
- Complete several problem tree exercises with different stakeholder groups, to help determine different perspectives and differing priorities
- Recognize that the process is as important as the product. The exercise should be presented as a learning experience for all those involved, and as an opportunity for different views and interests to be presented and discussed. However, don't expect from all stakeholders complete agreement about the problems and their relative importance
- Recognize that the product (the problem tree diagram) should provide a simplified but nevertheless robust version of reality
- Aim for simplicity. If the exercise is too complicated, it is likely to be less useful in providing direction to subsequent steps in the analysis



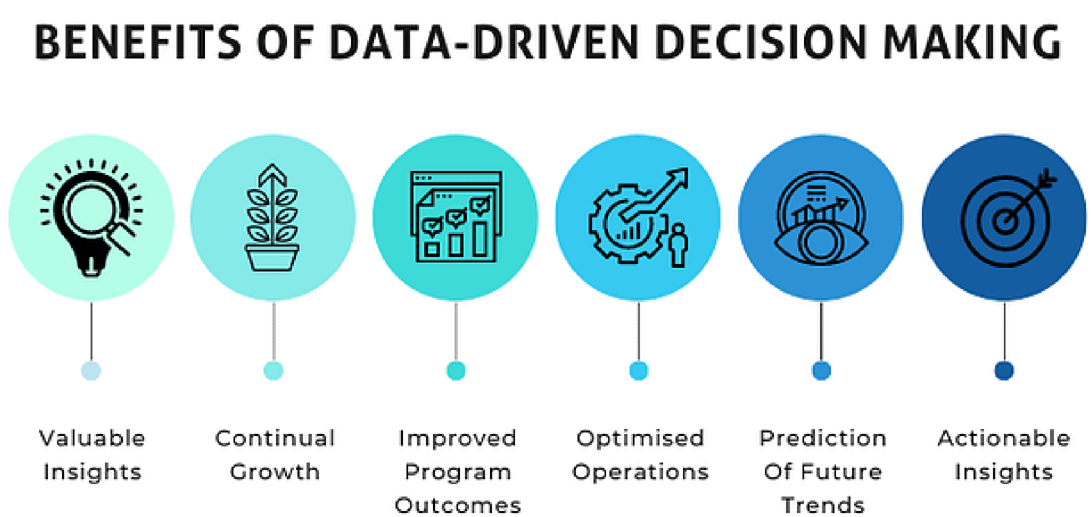
iii.) How is it and how is it going to be consumed by the consumer?
Understanding how consumers will use the output of your model will allow you to create features targeted to them. For example, are you building models that serve internal users and influence company strategy, or are you building models that are customer-facing.

iv.) The economic impact of this project

It is essential in the permitting process to show decision makers the benefits a project will have on a product (e.g., revenue increase , sales etc.). Alternatively, the report may be used to illustrate the economic impact on the company if a product was to be done away with.

v.) What type of decisions will our data drive?

Data-driven decision-making (sometimes abbreviated as DDDM) is the process of using data to inform your decision-making process and validate a course of action before committing to it.



vi.) Target in mind to quantify success of the project

Measuring the success of a project once it's brought to completion is a valuable practice. It provides a learning opportunity for future undertakings, and the opportunity to assess the true effectiveness of the project. In order to have a holistic view, objective and subjective criteria need to be considered.

Overview
We are going to consider a dataset of electronics sales data at Amazon. It contains user ratings for various electronics items sold, along with the category of each item and time of sale.

We will use Python libraries (Pandas, Numpy, Matplotlib & Seaborn) to analyze and answer business questions for sales data. The data contains hundreds of thousands of electronics store purchases broken down by month, product type, cost, purchase address, etc.
The dataset can be downloaded here.
https://github.com/AnudipAE/DANLC/blob/master/cleaned.csv

In this analysis, we will be using Jupyter Notebook.

STEP 1:

Exploratory Data Analysis [EDA]

This is the process by which we shall critically perform initial investigations of the data we have to discover patterns, to spot anomalies, test hypotheses and to check assumptions with the help of summary statistics and graphical representations.

It is how we get to understand the data we have and gather many insights from it. It is more of making sense of the data we have before working with it.

```
# Importing the libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# visualization

import seaborn as sns

# Importing the dataset

dataset =
pd.read_csv('https://raw.githubusercontent.com/AnudipAE/DANLC/master/cleaned.cs
v')

# list of first five rows

dataset.head()
```

Output:

index	item_id	user_id	rating	timestamp	gender	category	brand	year	month	quantity	unitprice	amount
0	7	131	4	36692	Female	Home Audio	Philips	2000	6	5	6360	31800
1	19	231	5	36891	Female	Camera	Canon	2000	12	10	9955	99550
2	14	233	5	36893	Female	Camera	Kodak	2001	1	9	7639	68751
3	14	257	5	36926	Female	Camera	Kodak	2001	2	7	5097	35679
4	14	269	5	36952	Female	Camera	Kodak	2001	3	10	6472	64720

To take a look at the first five rows we use the pandas function " .head()". Similarly ".tail()" returns the last five observations of the data set.

```
# list of last five rows

dataset.tail()
```

Output:

```
# list of last five rows

dataset.tail()
```

	item_id	user_id	rating	timestamp	gender	category	brand	year	month	quantity	unitprice	amount
45161	7828	1157458	5	43341	Female	Headphones	Bose	2018	8	7	5925	41475
45162	8624	1157504	5	43342	Female	Headphones	Pyle	2018	8	7	9717	68019
45163	9513	1157527	5	43344	Male	Headphones	Mpow	2018	9	8	9197	73576
45164	9125	1157555	3	43348	Female	Headphones	EldHus	2018	9	10	8848	88480
45165	9478	1157632	1	43374	Female	Headphones	Etre Jeune	2018	10	7	7717	54019

To know the total number of rows and columns in the data set we use ".shape" as shown below.

```
# shape

dataset.shape
```

Output:

```
# shape

dataset.shape
```

```
(45166, 12)
```

Inference:
Dataset comprises 45166 Rows and 12 columns.

It is also a good practice to know the columns and their corresponding data types, along with finding whether they contain null values or not.

```
# It is also a good practice to know the columns and their corresponding data types
# along with finding whether they contain null values or not.

dataset.info()
```

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45166 entries, 0 to 45165
Data columns (total 12 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   item_id    45166 non-null  int64 
 1   user_id    45166 non-null  int64 
 2   rating     45166 non-null  int64 
 3   timestamp  45166 non-null  int64 
 4   gender     45166 non-null  object
 5   category   45166 non-null  object
 6   brand      45166 non-null  object
 7   year       45166 non-null  int64 
 8   month      45166 non-null  int64 
 9   quantity   45166 non-null  int64 
 10  unitprice  45166 non-null  int64 
 11  amount     45166 non-null  int64 
```

Inference:
No Variable column has null/missing values
We can see that the dataset contains 12 columns and 45166 rows.

The columns are as follows:

1. item_id
2. user_id
3. rating
4. timestamp
5. gender
6. category
7. brand
8. year
9. month
10. quantity
11. unitprice
12. amount

The data types of the columns are as follows:

1. item_id int64
2. user_id int64
3. rating int64
4. timestamp int64
5. gender object
6. category object
7. brand object
8. year int64
9. month int64
10. quantity int64
11. unitprice int64

12. amount	int64

We can see that the columns User ID and Rating are of int64 data type, while the columns Product ID and Category are of object data type there are no null values in the dataset. The column Timestamp is of int64 data type.

The column Product ID is of object data type, but it is actually a string, the column Category is of object data type, but it is actually a string.

To get a better understanding of the dataset, we can also see the statistical summary of the dataset using the function ".describe()".

This includes count, mean, median (or 50th percentile) standard variation, min-max, and percentile values of columns as shown below.

```
# to get a better understanding of the dataset,

# we can also see the statistical summary of the dataset.

dataset['rating'].describe()
```

Output:

```
count    45166.000000
mean         4.218594
std          1.221118
min          1.000000
25%          4.000000
50%          5.000000
75%          5.000000
max          5.000000
Name: rating, dtype: float64
```

Inference:
The statistical summary of the dataset gives us the following information:

1. The mean rating is 4.2
2. The minimum rating is 1
3. The maximum rating is 5.
4. The standard deviation of the ratings is 1.22
5. The 25th percentile of the ratings is 4.
6. The 50th percentile of the ratings is 5.
7. The 75th percentile of the ratings is 5.

We can also see the number of unique users and items in the dataset.

```
# We can also see the number of unique users and items in the dataset.

dataset.nunique()
```

Output:

```
item_id        1892
user_id       40401
rating            5
timestamp      4179
gender            2
category         10
brand            50
year             19
month            12
quantity          6
unitprice      5001
amount        19611
dtype: int64
```

Dealing With Missing Values

There can be multiple reasons why certain values are missing from the data. Reasons for the missing data from the dataset affect the approach of handling missing data. So it’s necessary to understand why the data could be missing.

Some of the reasons are listed below:
Past data might get corrupted due to improper maintenance.
Observations are not recorded for certain fields due to some reasons.
There might be a failure in recording the values due to human error.
The user has not provided the values intentionally.

```
# check for missing values

dataset.isnull().sum()
```

Output:

```
item_id      0
user_id      0
rating       0
timestamp    0
gender       0
category     0
brand        0
year         0
month        0
quantity     0
unitprice    0
amount       0
```

Image: Checking sum of Null Values

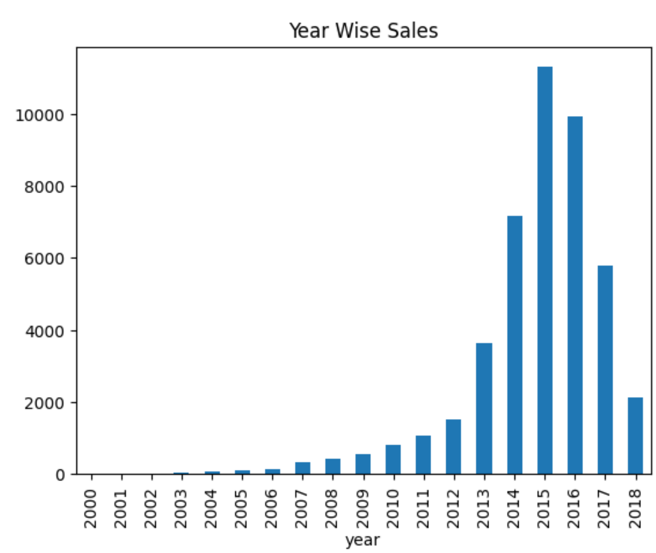
<u>Finding Answers with the Data Using Visualizations</u>

To make it easier to understand, we are going to use matplotlib and seaborn that we earlier imported to visualize our results with simple bar charts. This will make it easier to answer questions that might arise from the data set.

i.) What was the best year of sales?

```
# what was the best year of sales

dataset.groupby('year')['amount'].count().plot(kind='bar',title='Year
Wise Sales')
```

Output:

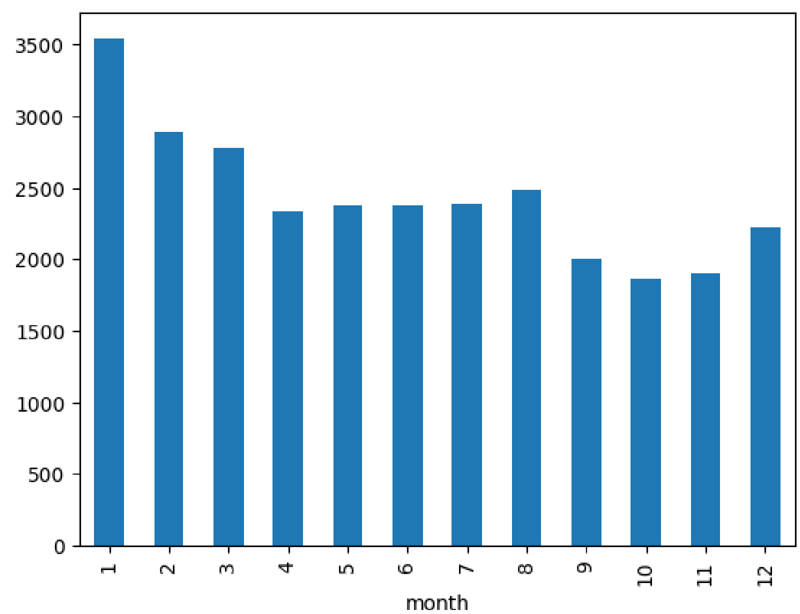


Inference:
From the graph we just plotted we can see that year 2015 had the best sales out of all years.
There was a steady increase of sales from the year 2007 to 2015 then a slight decline in 2016. That decline in sales was big in the following years of 2017 and 2018.

ii.) Which was the best month for sales between 2015 t0 2018

```
# We can see that the year 2015  to 2018 had the best sales.

# what was the best month of sales
dataset_2015_2018 = dataset[(dataset['year'] >= 2015) & (dataset['year']
<= 2018)]

dataset_2015_2018.groupby('month')['rating'].count().plot(kind='bar')
```

Output:



Inference:
January was the month when most sales were made across the product categories and over the years.

iii.) What brand sold the most in the highest selling year(2015 to 2018)

```
# what brand sold the most in 2015 to 2018

dataset_2015_2018 = dataset[(dataset['year'] >= 2015) & (dataset['year']
<= 2018)]
dataset_2015_2018.groupby('brand')['amount'].sum().sort_values(ascending
=False).head(10)\
.plot(kind='bar',title='Brand Wise Top 10 Sales 2015 to
2018',y='amount')
```

Output:

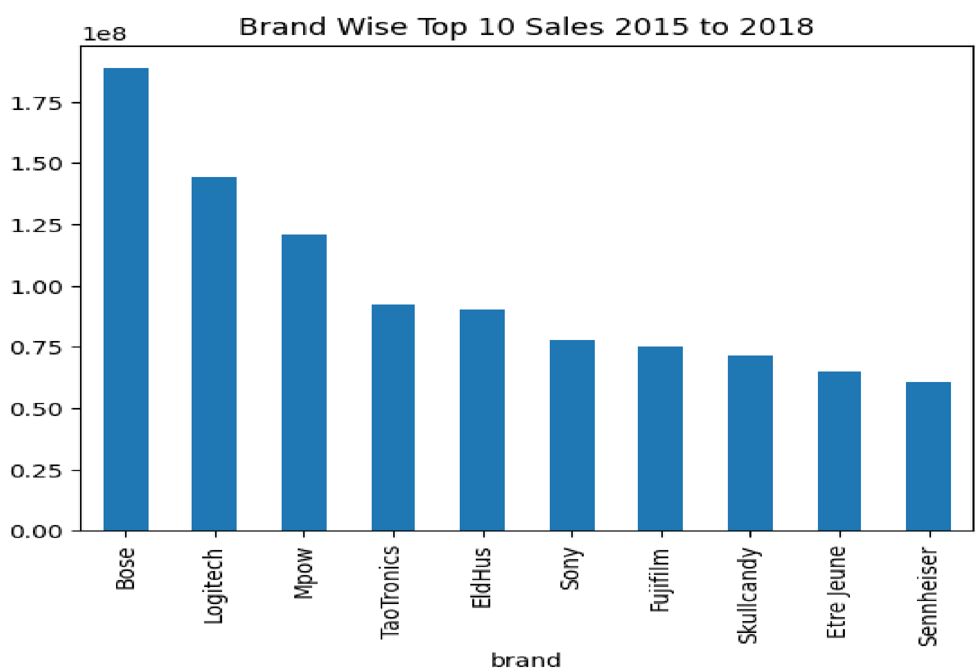


Image: Best selling Brand

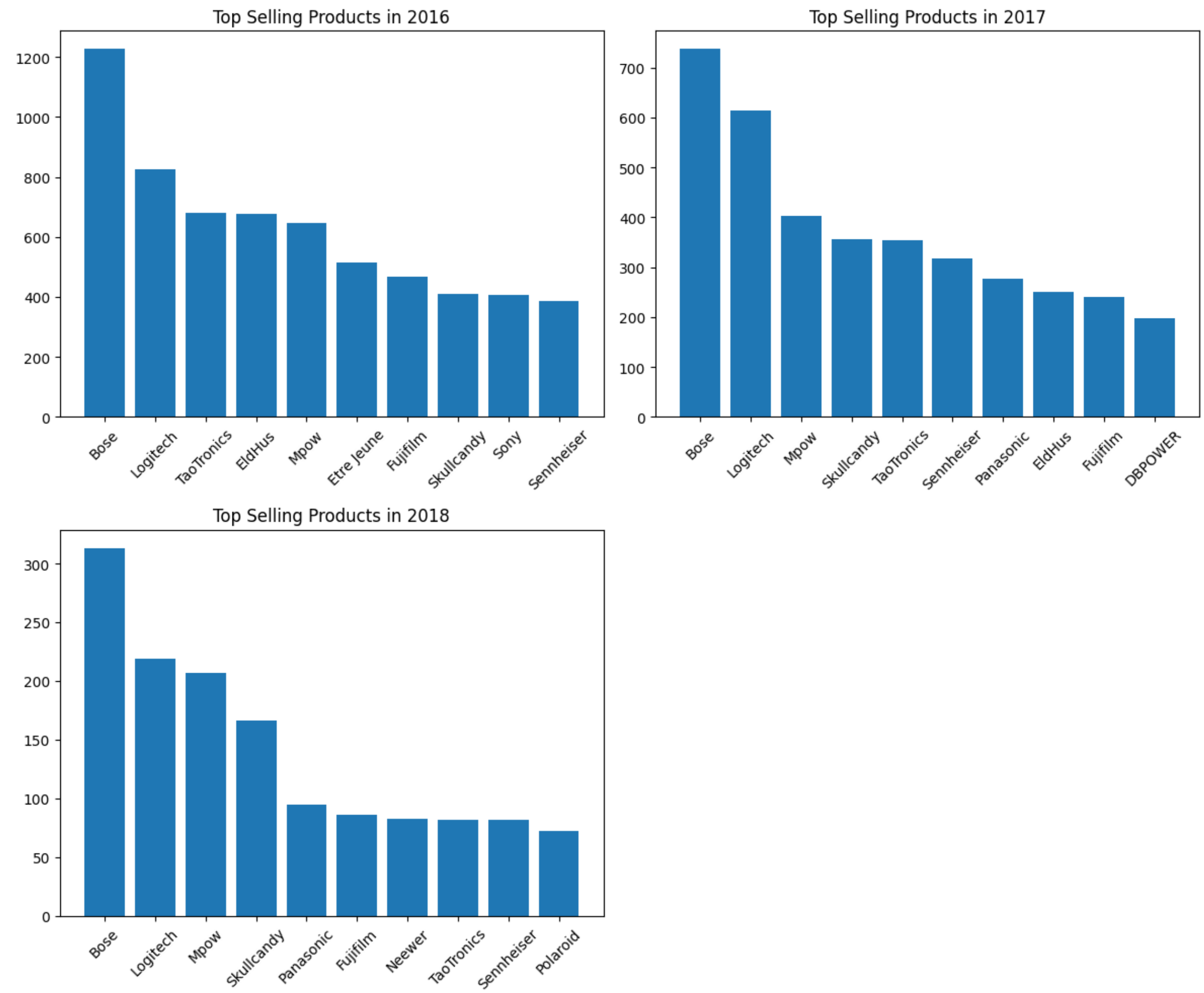
Inference:
Bose was the brand with the most sales in 2015 to 2018 followed by Logitech.

iv.) What products sold the most in the three years 2016, 2017 & 2018

```
# Create subplots with 2 rows and 2 columns
fig, axs = plt.subplots(2, 2, figsize=(12, 10))
```

```
# Plot for 2016
top_selling_2016 = dataset[dataset['year'] ==
2016].groupby('brand')['rating'].count().sort_values(ascending=False).he
ad(10)
axs[0, 0].bar(top_selling_2016.index, top_selling_2016)
axs[0, 0].set_title('Top Selling Products in 2016')
axs[0, 0].tick_params(axis='x', rotation=45)  # Rotate x-axis labels

# Plot for 2017
top_selling_2017 = dataset[dataset['year'] ==
2017].groupby('brand')['rating'].count().sort_values(ascending=False).he
ad(10)
axs[0, 1].bar(top_selling_2017.index, top_selling_2017)
axs[0, 1].set_title('Top Selling Products in 2017')
axs[0, 1].tick_params(axis='x', rotation=45)  # Rotate x-axis labels

# Plot for 2018
top_selling_2018 = dataset[dataset['year'] ==
2018].groupby('brand')['rating'].count().sort_values(ascending=False).he
ad(10)
axs[1, 0].bar(top_selling_2018.index, top_selling_2018)
axs[1, 0].set_title('Top Selling Products in 2018')
axs[1, 0].tick_params(axis='x', rotation=45)  # Rotate x-axis labels

# Hide the empty subplot
axs[1, 1].axis('off')

# Adjust layout for better appearance
plt.tight_layout()

# Show the plots
plt.show()
```

Output:



Inference:
There has been one consistent Brand product with the most sales in the 3 years and it is Bose.
The second most sold brand's products have been Logitech.

- 2016 (Bose and Logitech)
- 2017 (Bose and Logitech)
- 2018 (Bose and Logitech)

v.) What product by category sold the most between 2015 to 2018?

```
# # What product by category sold the most between 2015 to 2018?
dataset2015_2018 = dataset[(dataset['year'] >= 2015) & (dataset['year']
<= 2018)]
```

```
dataset2015_2018.groupby('category')['amount'].sum().sort_values(ascending=False).head(10).plot(kind='bar',title='Top 10 Most Sold Product Category 2015 to 2018')
```

Output:

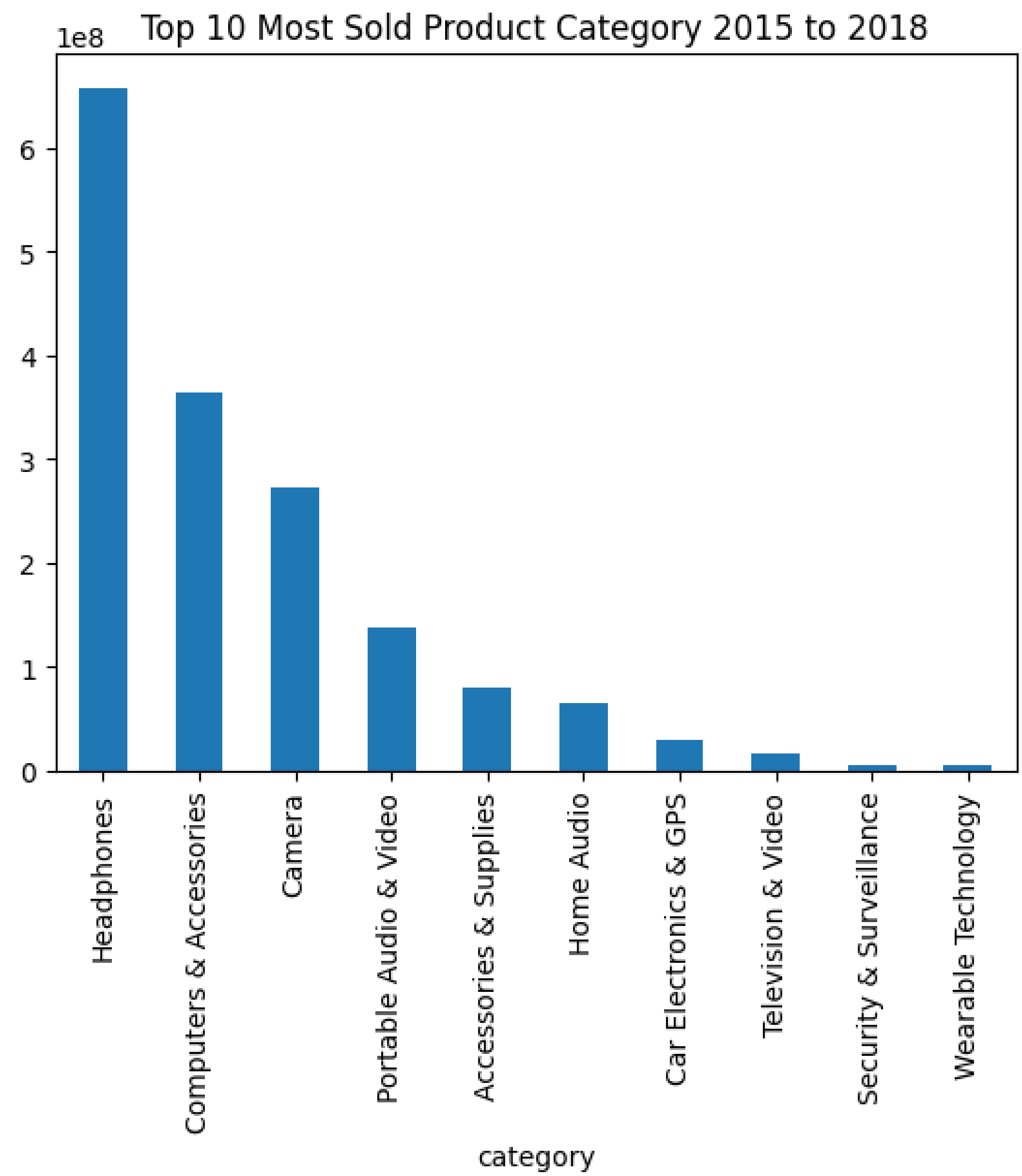


Image: Product by Category that sold the most

Inference:
We can see that the category of Headphones sold the most, computers and accessories were sold the second most while cameras sold the third most .

vi.)What product by category sold the least between 2015 to 2018?

```
# What product by brand name sold the least between 2015 to 2018?
dataset2015_2018 = dataset[(dataset['year'] >= 2015) & (dataset['year']
<= 2018)]
dataset2015_2018.groupby('category')['amount'].sum().sort_values(ascendi
ng=True).head(10).plot(kind='bar',title='10 Least  Sold Product Brand
2015 to 2018')
```

Output:

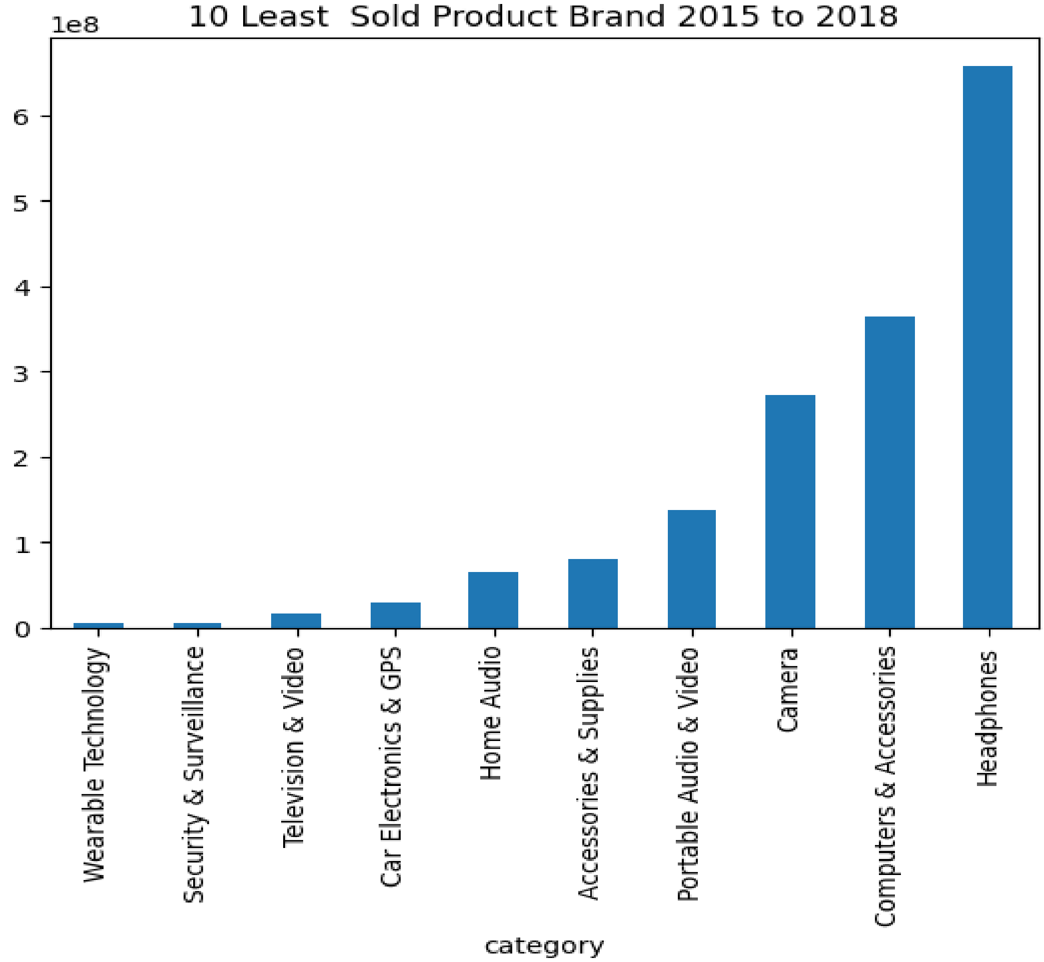


Image: Product by Category that sold the least

Inference:
We can see that the category of Wearable Technology sold the least followed closely by Security and Surveillance.

vii.) What product by brand name sold the least between 2015 to 2018?

```
# What product by brand name sold the least between 2015 to 2018?
dataset2015_2018 = dataset[(dataset['year'] >= 2015) & (dataset['year']
<= 2018)]
dataset2015_2018.groupby('brand')['amount'].sum().sort_values(ascending=
True).head(10).plot(kind='bar',title='10 Least  Sold Product Brand 2015
to 2018')
```

Output:

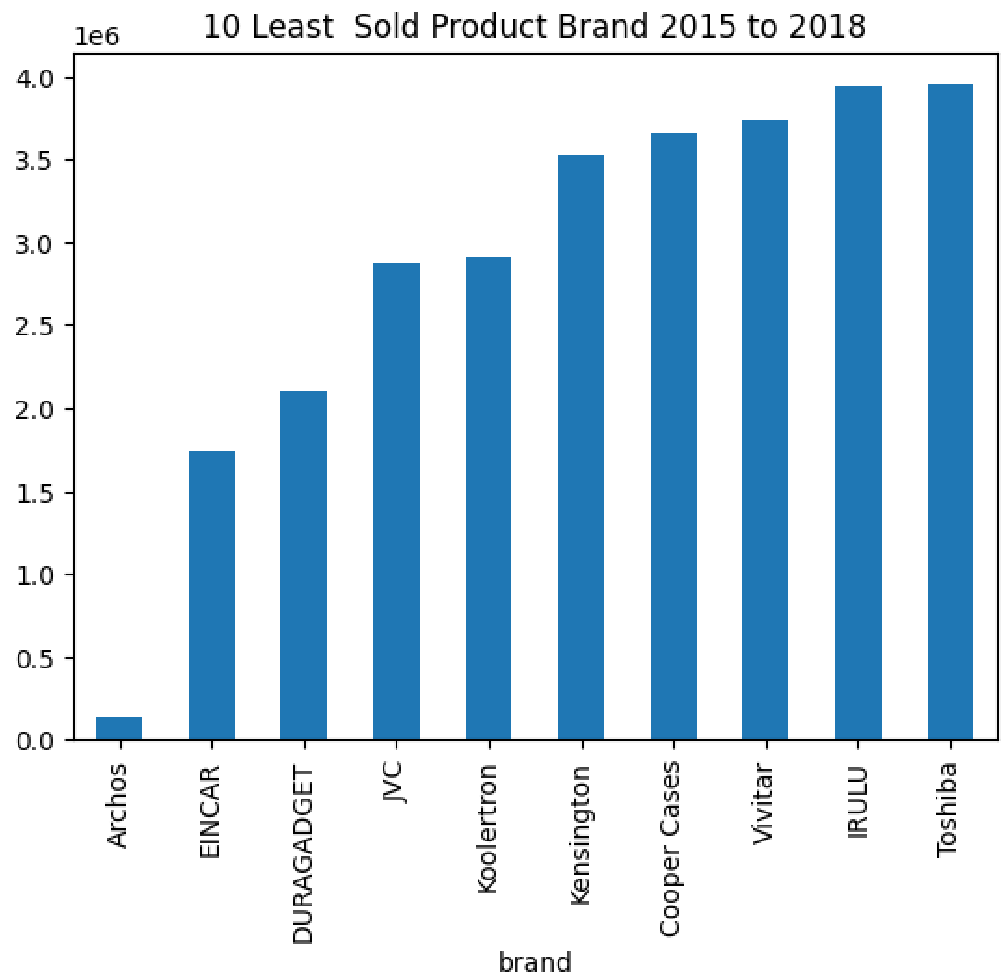


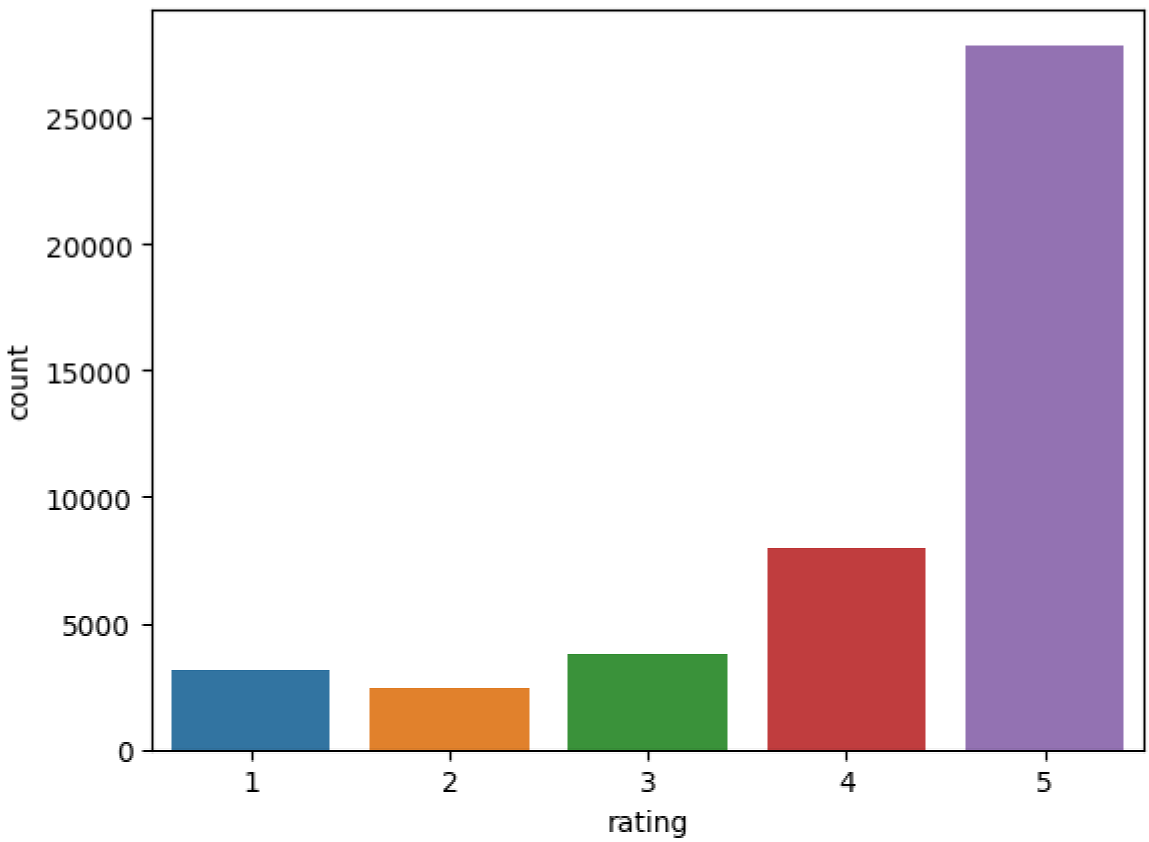
Image: Product by brand name sold the least

Inference:
Archos sold the least followed closely with EINCAR.

viii.) Ratings Distribution

```
# # the distribution of ratings

sns.countplot(x='rating', data=dataset)
```

Output:



Inference:
Most Products were rated 5

ix.) Best rated brands

```
# What is the most rated brand name  between 2015 to 2018?
dataset2015_2018 = dataset[(dataset['year'] >= 2015) & (dataset['year']
<= 2018)]
dataset2015_2018.groupby('brand')['rating'].mean().sort_values(ascending
=False).head(10).plot(kind='bar',title='10 most rating Brand 2015 to
2018')
```

Output:

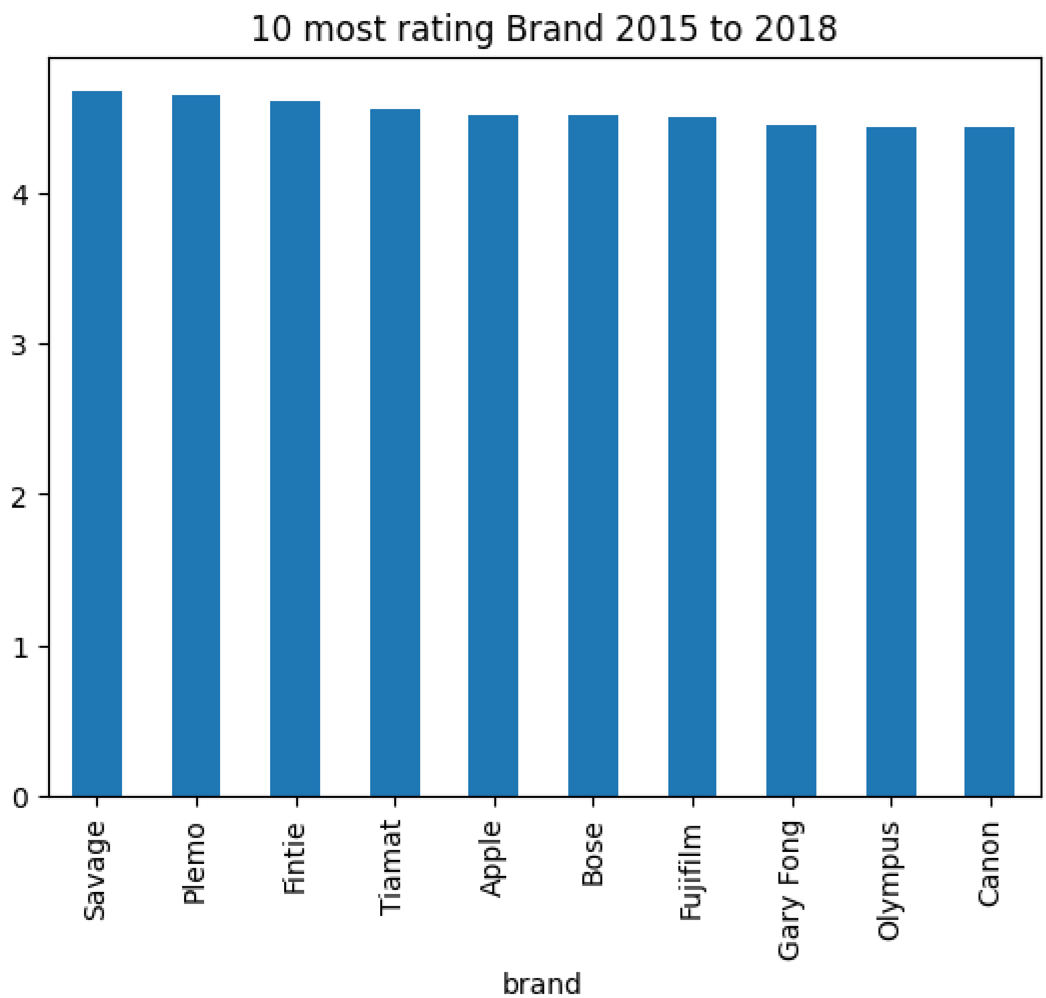


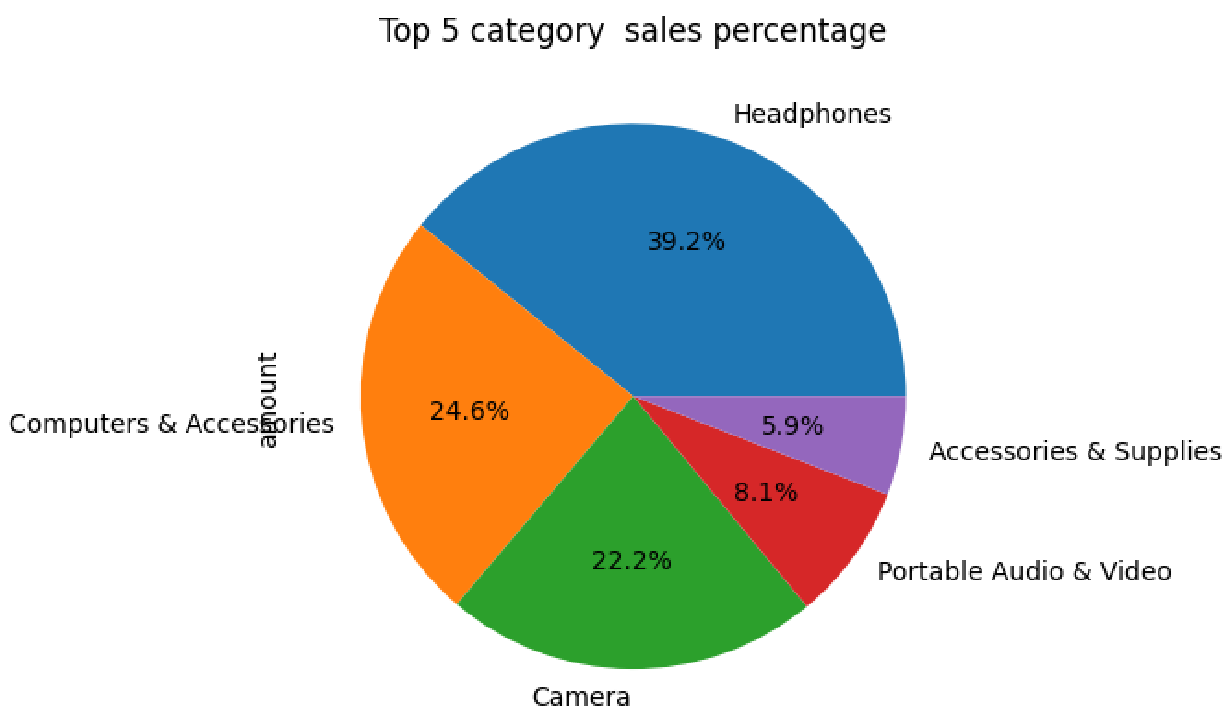
Image: Best brands by rating

Inference:
Savage and Plemo were the brands with the highest ratings.

x) Top 5 category sales percentage

```
# category percentage sales

dataset.groupby('category')['amount'].sum().sort_values(ascending=False)
.head(5).plot(kind='pie', autopct='%1.1f%%',title='Top 5 category  sales
percentage')
```

Output:

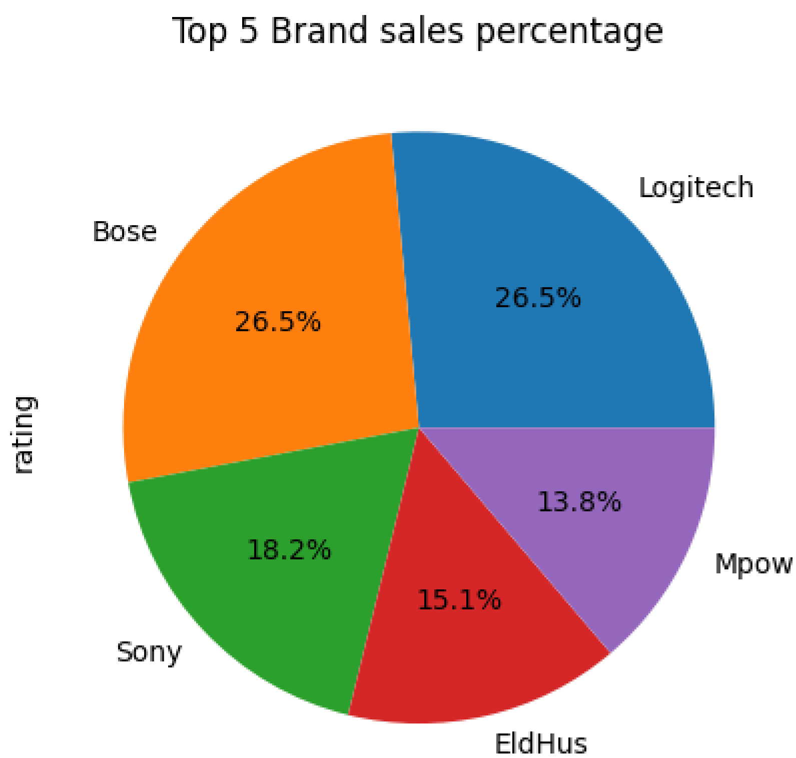


Inference:
Headphones sales % is the highest followed by Computers & Accessories.

xi) Brand wise sales percentage

```
# brand wise sales percentage

dataset.groupby('brand')['rating'].count().sort_values(ascending=False).
head(5).plot(kind='pie', autopct='%1.1f%%',title='Top 5 Brand wise sales
percentage')
```

Output:

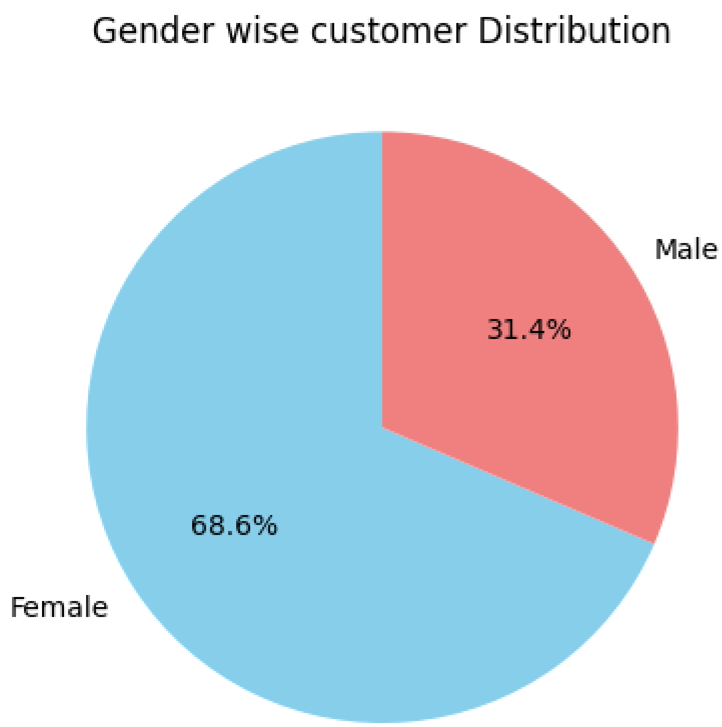


Inference:
Bose and Logitech sales % is the highest followed by Sony.

xii) Gender wise customer distribution

```
# Gender wise customer distribution
gender_distribution = dataset['gender'].value_counts()
plt.pie(gender_distribution, labels=gender_distribution.index,
autopct='%1.1f%%', startangle=90, colors=['skyblue', 'lightcoral'])
plt.title('Gender wise customer Distribution')
plt.show()
```

Output:



Inference:
Most of the customers are in Female categories.

Conclusion:

- 2015 was the best year in terms of sales and profit
- Headphones was the category with most sales followed closely with Computer and Accessories while the least sales were made in the Category Security & Surveillance.
- There has been a steady rise in sales from 2007 to 2015 and a sharp decline from 2016 to 2018.
- The brand name Bose sold the most followed by Logitech.
- The brand Archos sold the least followed closely with EINCAR..
- Most products were rated 5.
- Best rated brands were Savage and Plemo.

The above analysis should help you to understand and explore further on the reasons behind the popularity and/or poor sales of the products. With this foresight a company can make decisions whether to continue production/sales of a specific product for the future.