Project Development Guidelines

- 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.

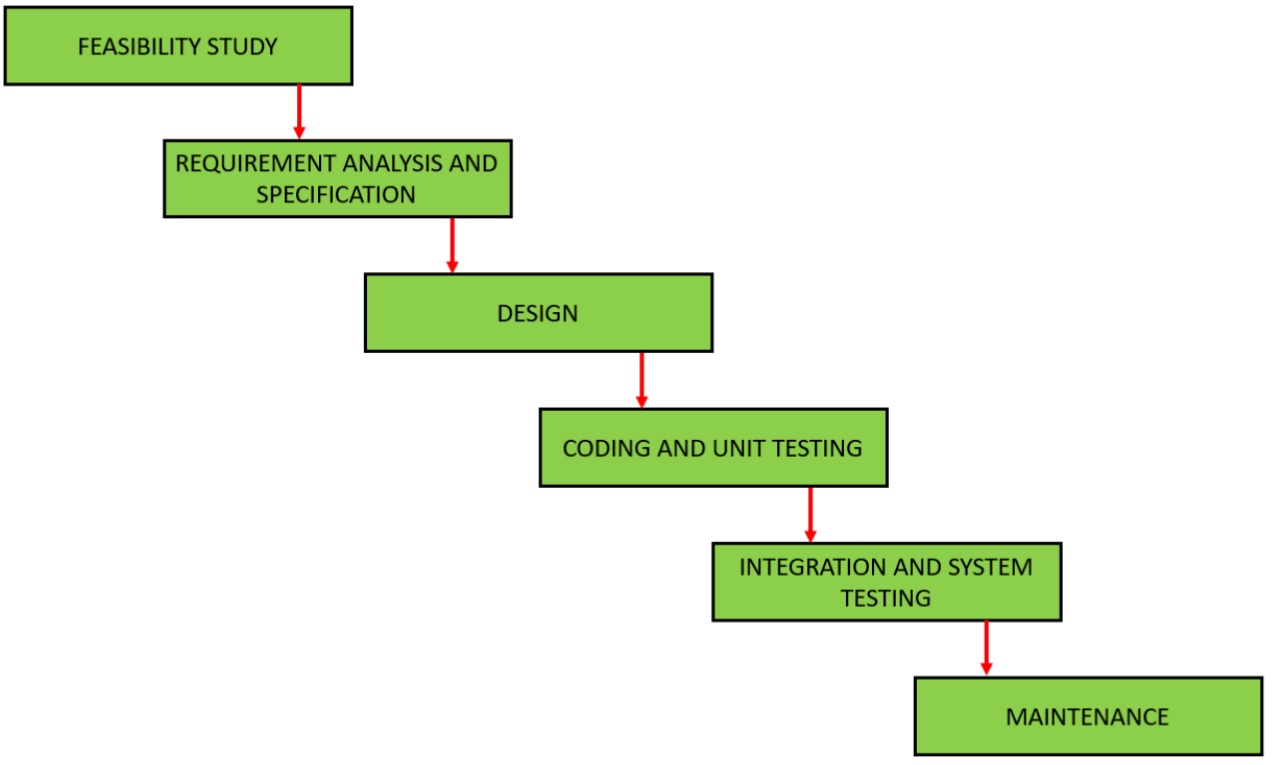
Here is a general overview of the typical stages in the SDLC:

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.

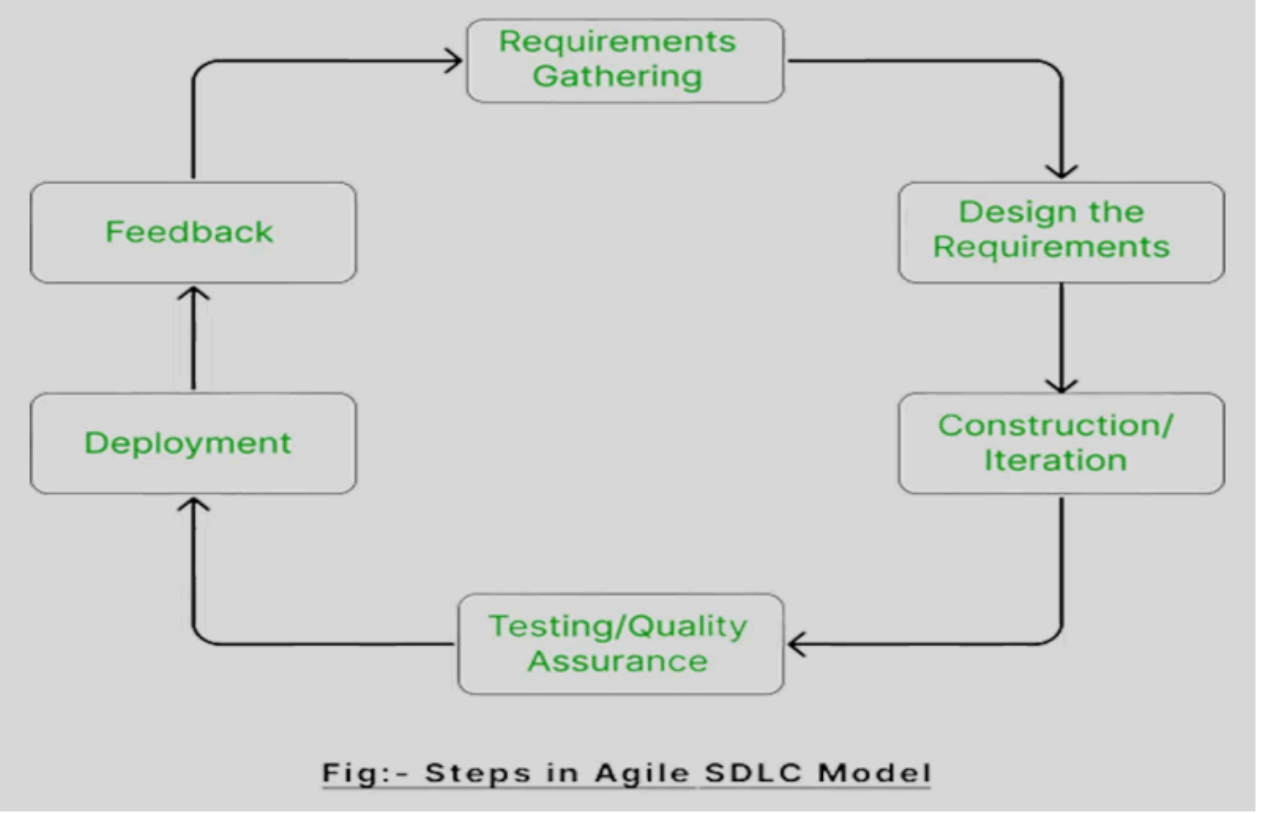
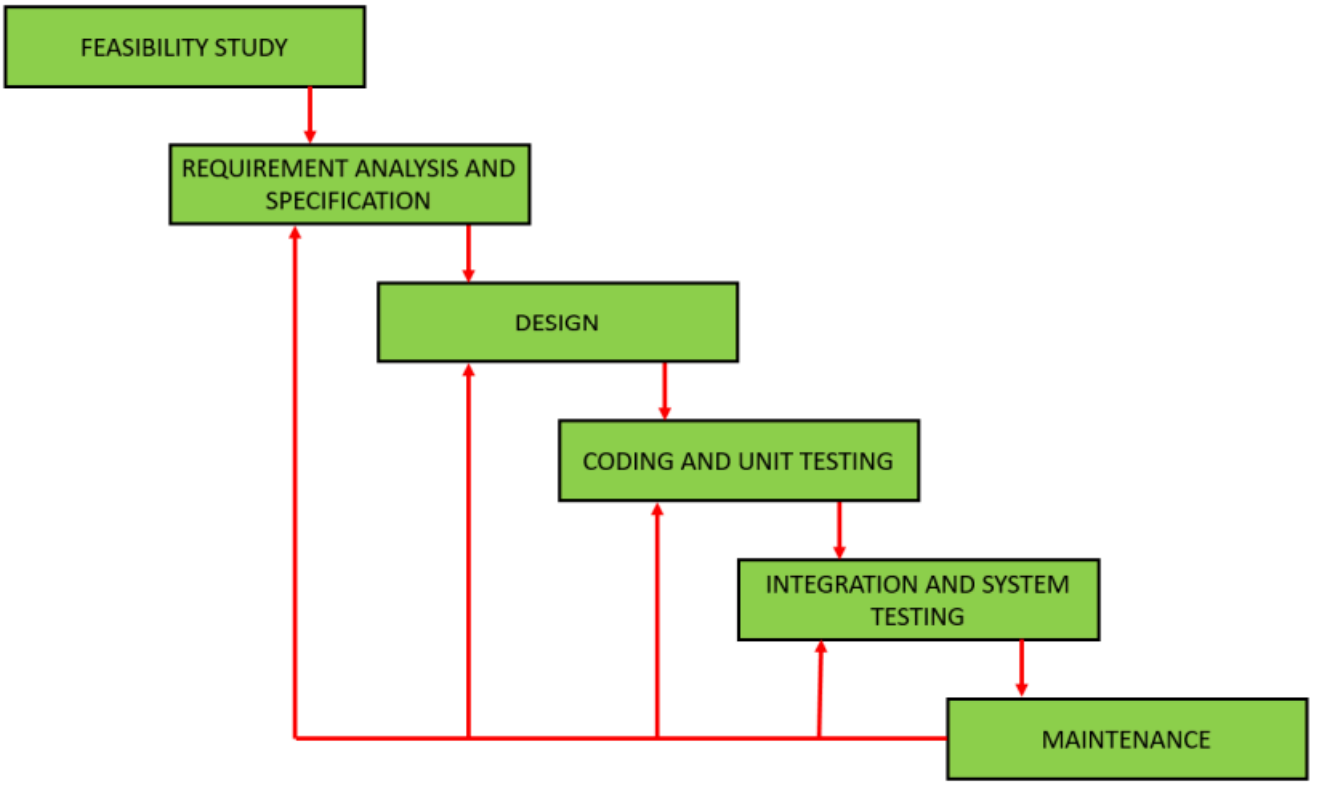


Fig:- Steps in Agile SDLC Model

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

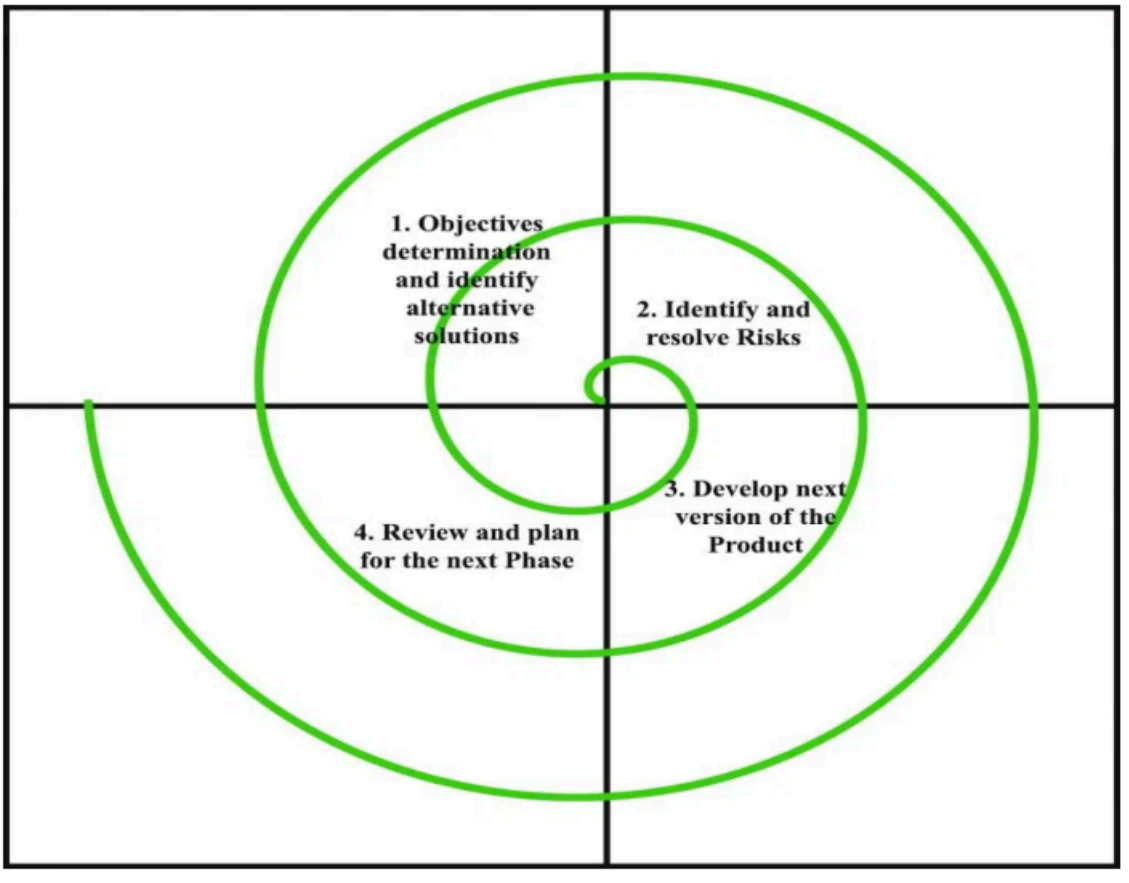
3. Iterative Model

- 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

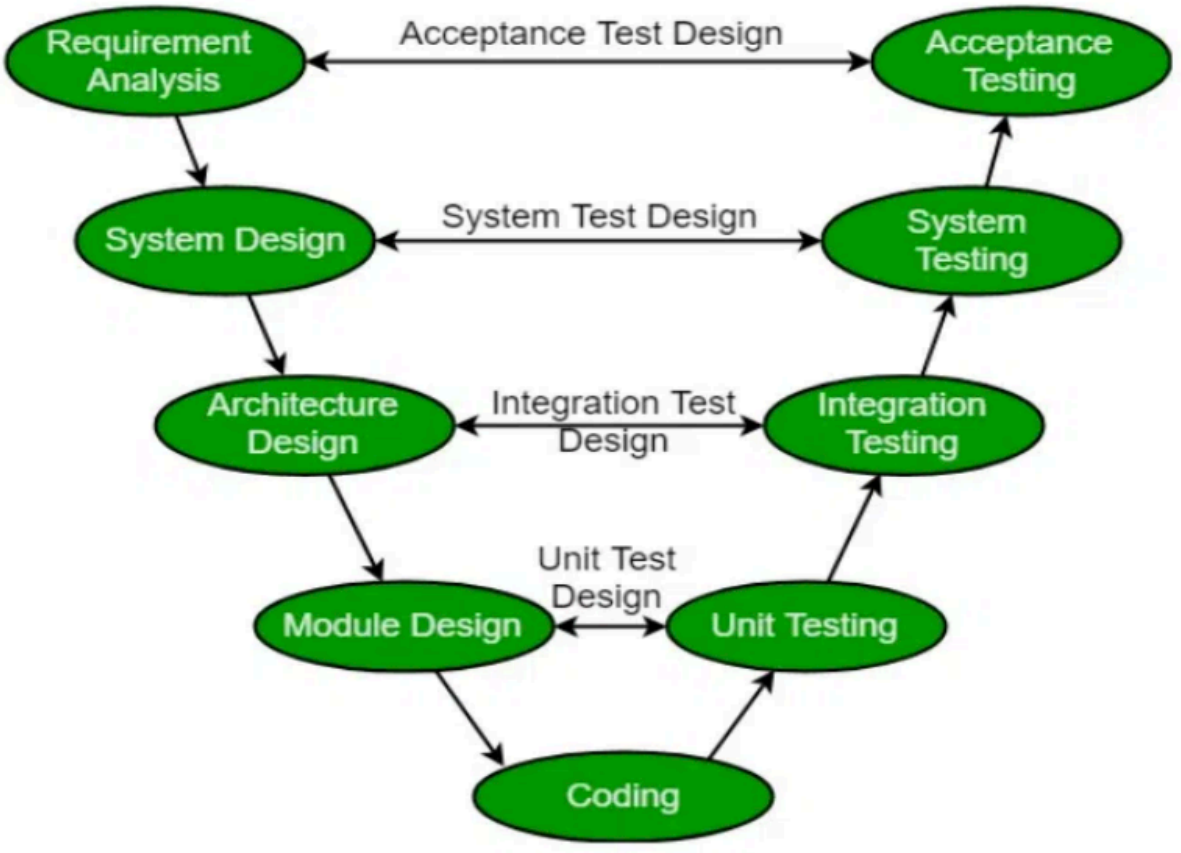
- The spiral model is one of the most crucial SDLC models that provides support for risk handling.



- 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.

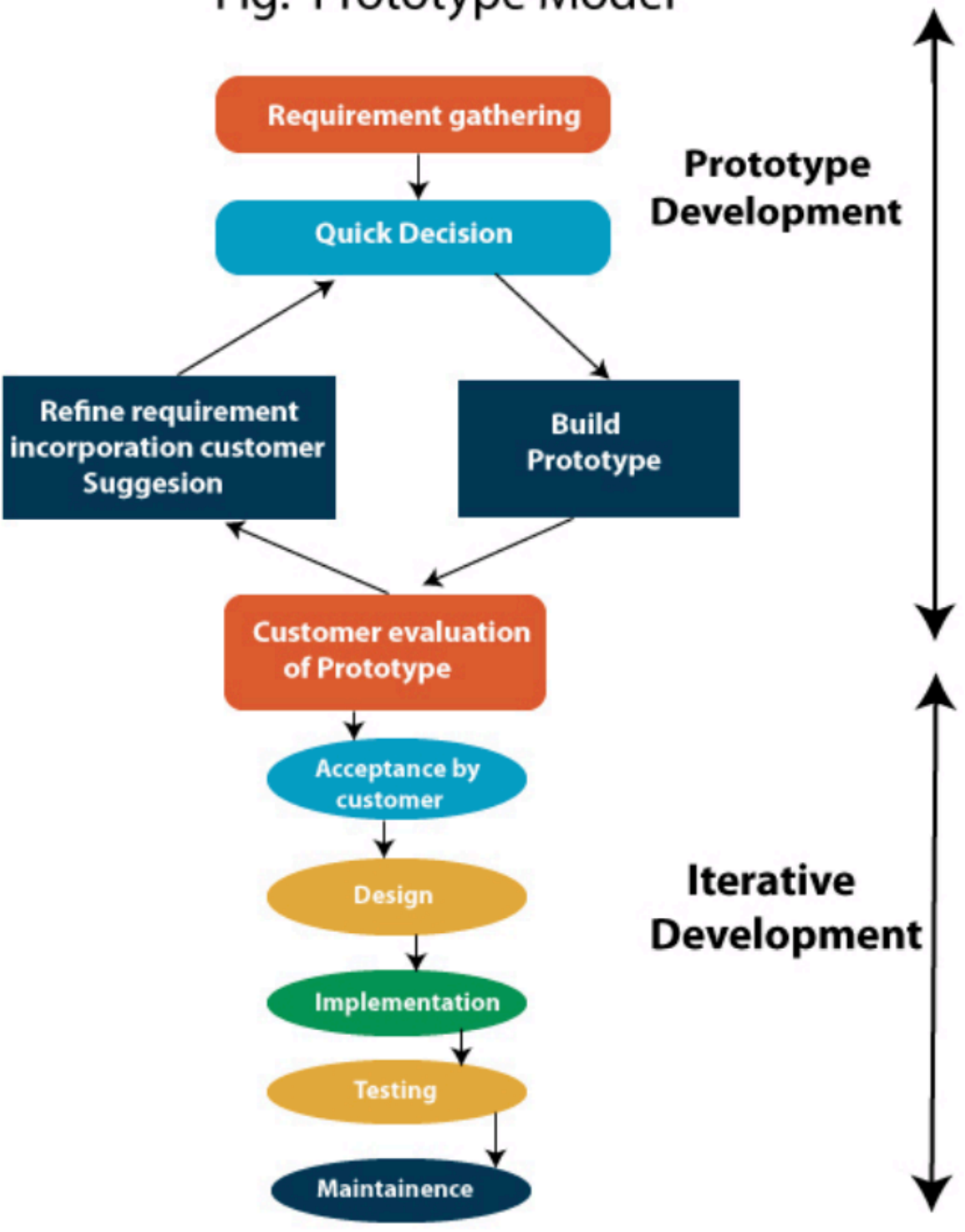


- After every development phase, a testing phase is associated with it, and the next phase will start once the previous phase is completed, i.e., development & testing. It is also known as the verification or validation model.

6. Prototype

- 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.

Fig: Prototype 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

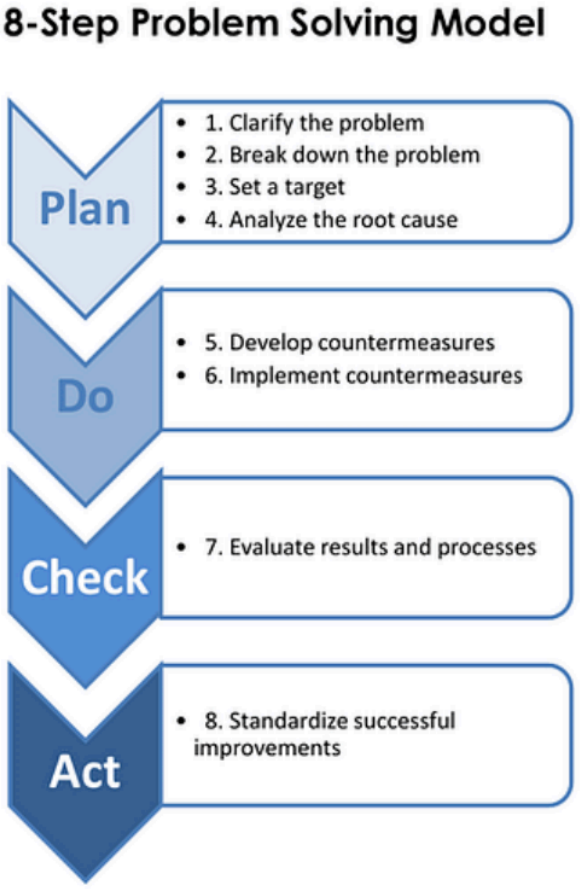
Business model refers to a company's plan for making a profit. It identifies the products or services the business plans to sell, its identified target market, and any anticipated expenses.

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



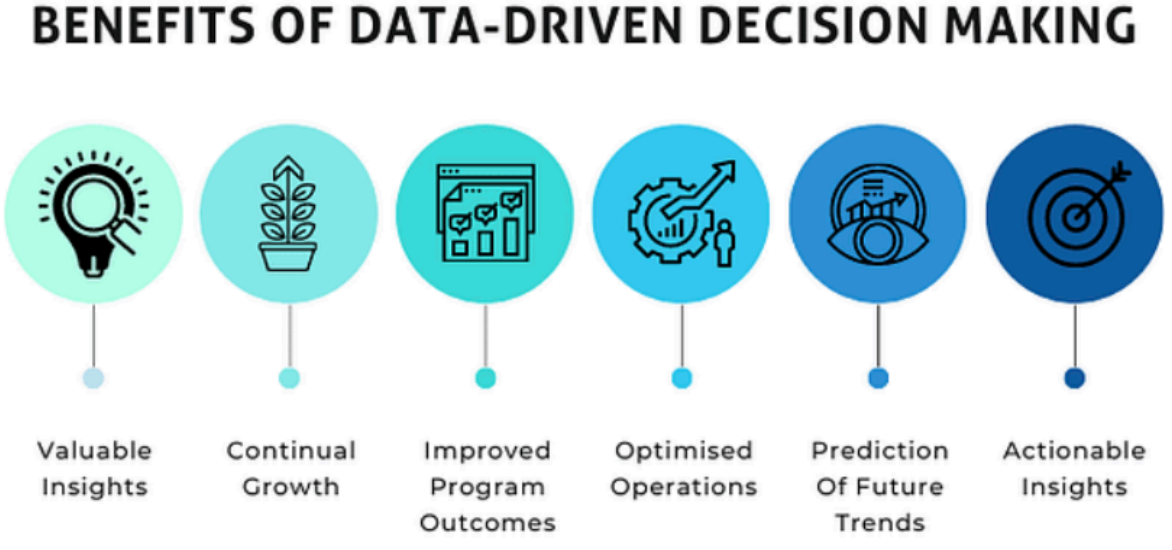
iii.) How is it and how is it going to be consumed by the consumer?
Understanding how consumers will use the output of your model will allow you to create features targeted to them. For example, are you building models that serve internal users and influence company strategy, or are you building models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

```python
# Importing the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Importing the dataset
dataset = pd.read_csv('/Users/arunkumarsharma/Documents/cleaned.csv')

# list of first five rows
print(dataset.head())

# list of last five rows
print(dataset.tail())
```

```python
# shape
print(dataset.shape)
```

```
(45166, 12)
```

```python
# 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.
print(dataset.info())
```

```
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
dtypes: int64(9), object(3)
memory usage: 4.1+ MB
None
```

to get a better understanding of the dataset,
we can also see the statistical summary of the dataset.

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

```
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
```

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.

```
print(dataset.nunique())
```

```
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

Some of the reasons are listed below:

1. Past data might get corrupted due to improper maintenance.
2. Observations are not recorded for certain fields due to some reasons.
3. There might be a failure in recording the values due to human error.
4. The user has not provided the values intentionally.

check for missing values(Checking sum of Null Values)

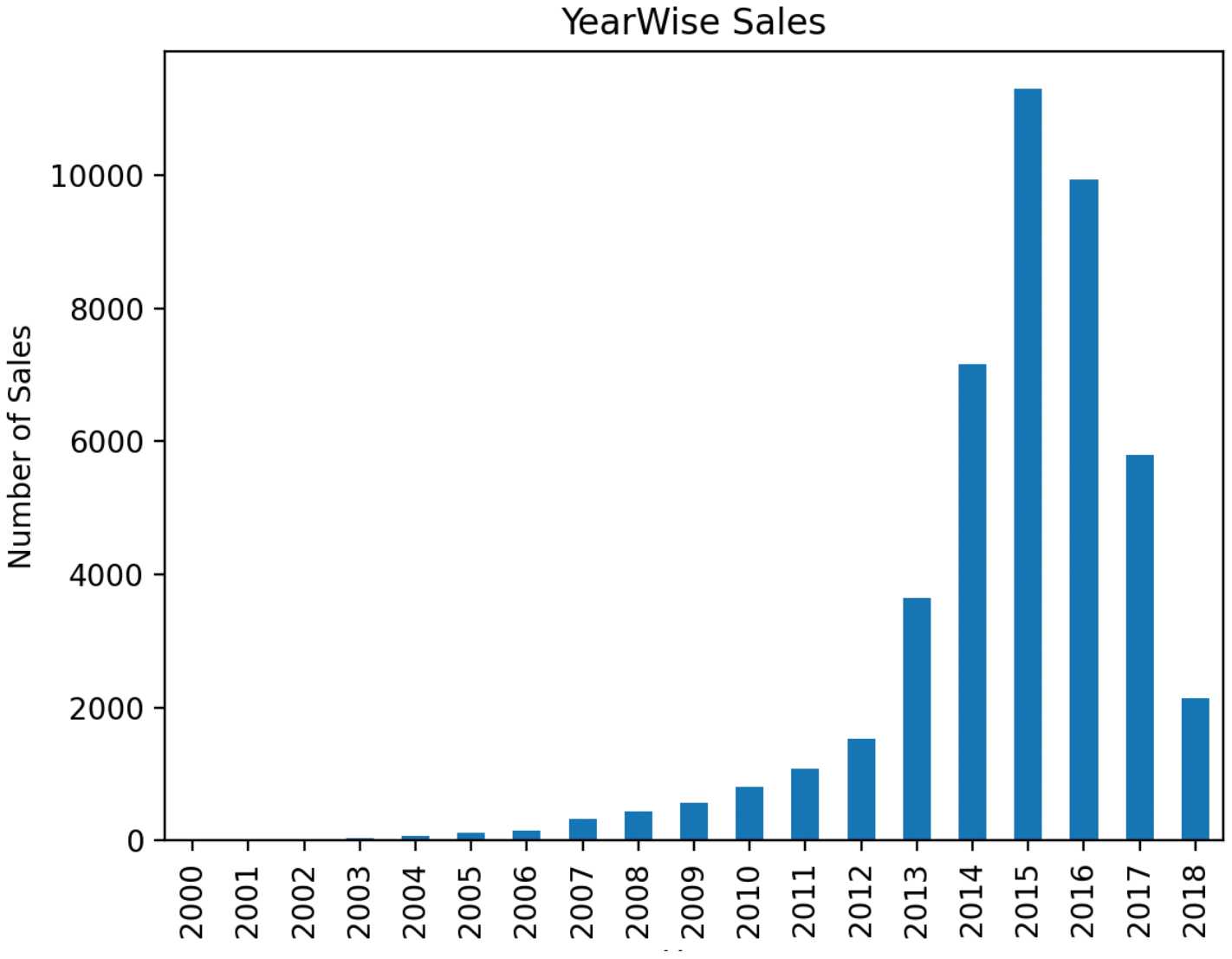
```
print(dataset.isnull().sum())
```

```
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
dtype: int64
```

Finding Answers with the Data Using Visualizations

i.) What was the best year of sales?

```
# what was the best year of sales
dataset.groupby('year')['amount'].count().plot(kind='bar',title='YearWise Sales')

# Show the graph
plt.xlabel("Year")
plt.ylabel("Number of Sales")
plt.show()
```



Year	Amount
2000	500
2001	550
2000	525
2002	600
2001	700
2000	450

↑ dataset

df.groupby("year")

2000 → 500, 525, 450

2001 → 550, 700

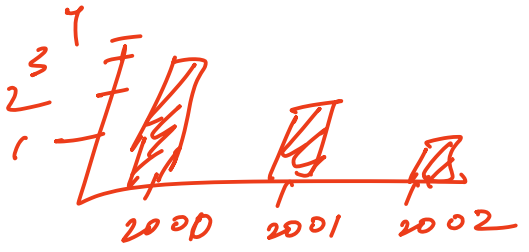
2002 → 600

↳ groupby

2000 → 3

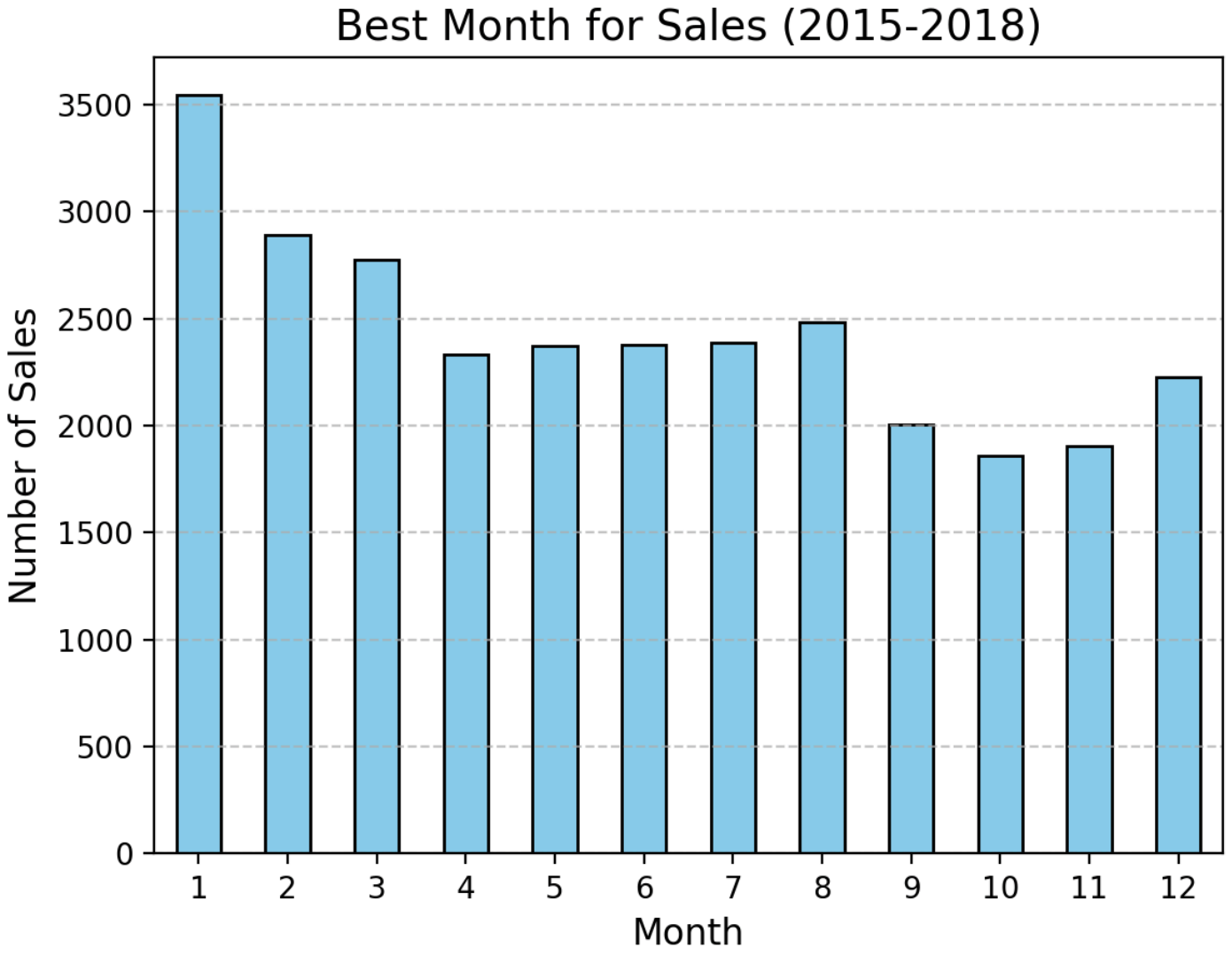
2001 → 2 ⇒ count()

2002 → 1



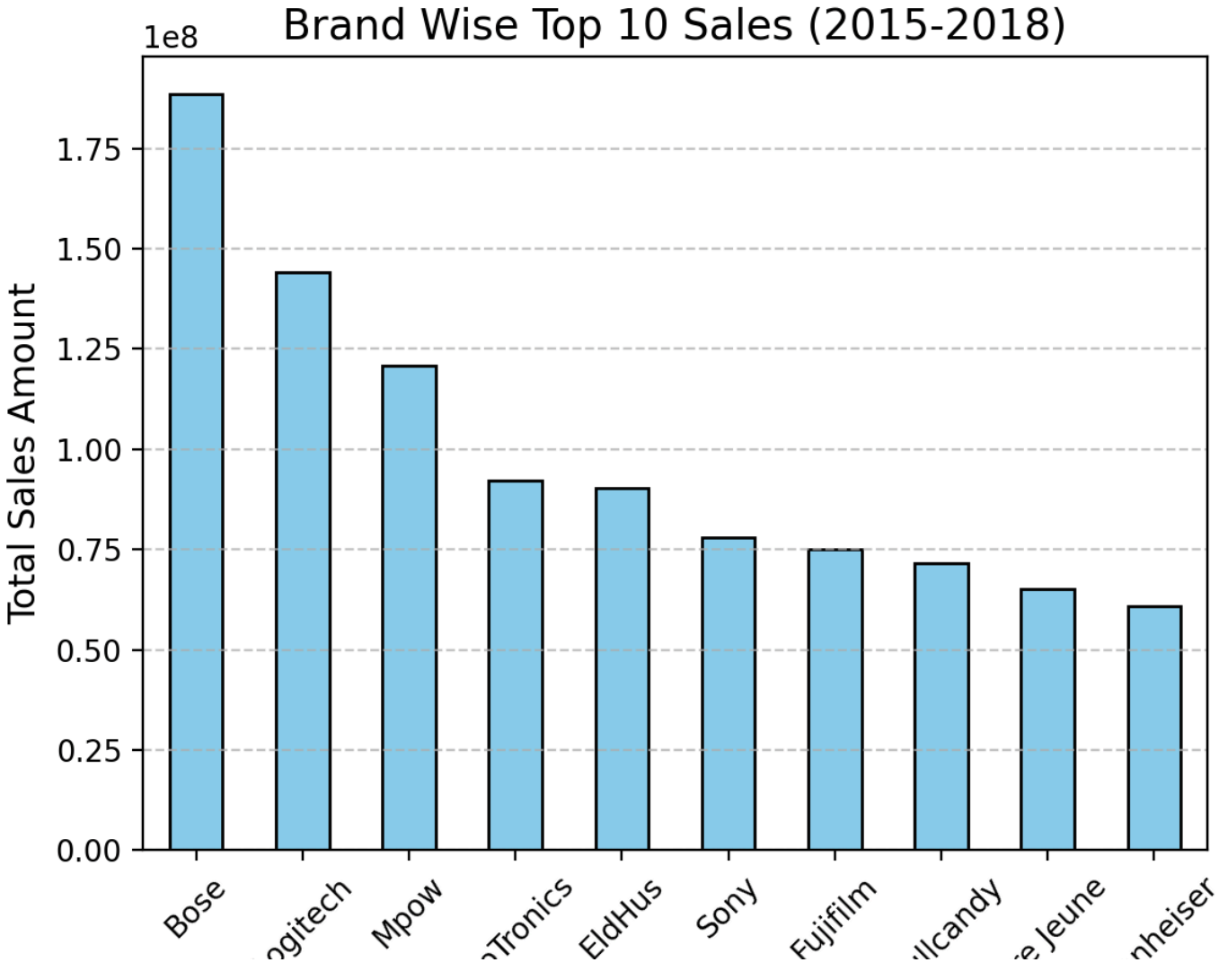
ii.) Which was the best month for sales between 2015 to 2018

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

# Load the dataset
dataset = pd.read_csv('/Users/arunkumarsharma/Documents/cleaned.csv')

# Filtering data for 2015-2018
dataset_2015_2018 = dataset[(dataset['year'] >= 2015) & (dataset['year'] <= 2018)]

# Grouping data by month and counting sales
dataset_2015_2018.groupby('month')['rating'].count().plot(kind='bar', color='skyblue', edgecolor='black')

# Enhancing the plot
plt.xlabel( xlabel: "Month", fontsize=12)
plt.ylabel( ylabel: "Number of Sales", fontsize=12)
plt.title( label: "Best Month for Sales (2015-2018)", fontsize=14)
plt.xticks(rotation=0)  # Keeps month labels straight
plt.grid(axis='y', linestyle='--', alpha=0.7)  # Adds a grid for better readability
plt.show()
```



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

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

# Load the dataset
dataset = pd.read_csv('/Users/arunkumarsharma/Documents/cleaned.csv')

# Filtering data for 2015-2018
dataset_2015_2018 = dataset[(dataset['year'] >= 2015) & (dataset['year'] <= 2018)]

# Grouping by brand and summing sales amount
top_brands = dataset_2015_2018.groupby('brand')['amount'].sum().sort_values(ascending=False).head(10)

# Plot the top 10 selling brands
top_brands.plot(kind='bar', color='skyblue', edgecolor='black')


plt.xlabel( xlabel: "Brand", fontsize=12)
plt.ylabel( ylabel: "Total Sales Amount", fontsize=12)
plt.title( label: "Brand Wise Top 10 Sales (2015-2018)", fontsize=14)
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



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

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

dataset = pd.read_csv('/Users/arunkumarsharma/Documents/cleaned.csv')

#Creating a Grid of Subplots (2x2 Layout)
fig, axs = plt.subplots( nrows: 2,  ncols: 2, figsize=(12, 10))
```

✅ plt.subplots(2, 2) → Creates a 2-row, 2-column grid for subplots.
✅ figsize=(12, 10) → Defines the size of the entire figure.
✅ axs is an array of subplots where:

axs[0,0] = Top left
axs[0,1] = Top right
axs[1,0] = Bottom left
axs[1,1] = Bottom right (unused in this case)

Finding & Plotting the Top 10 Brands for Each Year

🔷 Plot for 2016

```
top_selling_2016 = dataset[dataset['year'] == 2016].groupby('brand')['rating'].count().sort_values(ascending=False)
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 for better readability

plt.show()
```

✅ Filters 2016 data, groups by brand, and counts the number of sales (rating).
✅ Selects top 10 brands using .head(10).
✅ Plots a bar chart in top-left subplot (axs[0,0]).

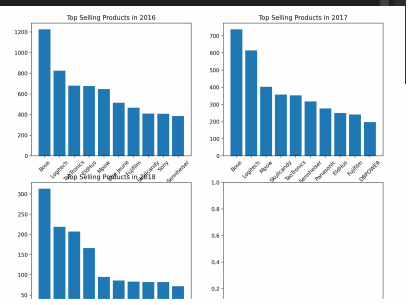
.bar(x, y)

This plots a bar chart, where:
x = top_selling_2018.index → Brand names (X-axis).
y = top_selling_2018 → Sales count (ratings) (Y-axis).

brand	count
A	10
B	8
C	6
D	5
:m	1

🔷 Plot for 2017 and 2018

```
top_selling_2016 = dataset[dataset['year'] == 2016].groupby('brand')['rating'].count().sort_values(ascending=False).he
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 for better readability

top_selling_2017 = dataset[dataset['year'] == 2017].groupby('brand')['rating'].count().sort_values(ascending=False).he
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)

top_selling_2018 = dataset[dataset['year'] == 2018].groupby('brand')['rating'].count().sort_values(ascending=False).he
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)
```

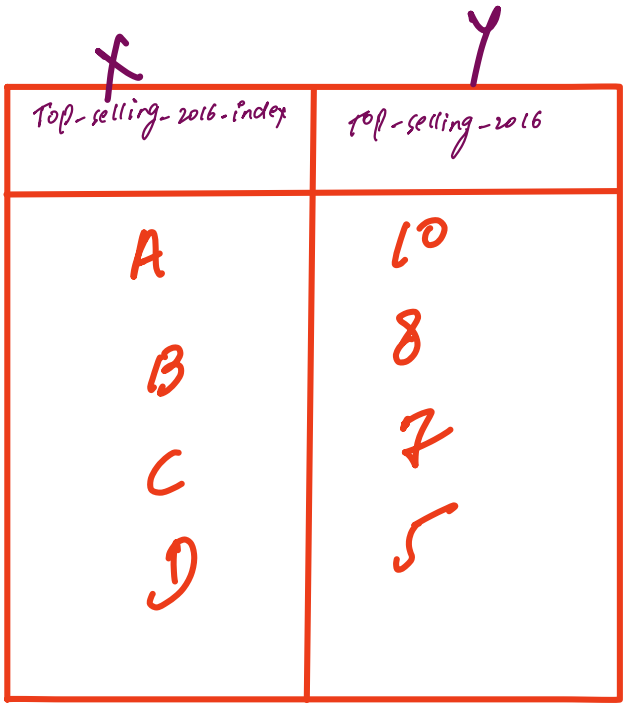


Hiding the Unused Subplot

```
axs[1, 1].axis('off')
```

Adjusting Layout for Better Appearance

```
plt.tight_layout()
```



-bar(x,y)

	Amt
A	500
B	550
C	450
A	300
B	960
A	700
C	200

A → 500, 300, 700 } 1500

B → 550, 960 } 1510

C → 450, 200 } 650

---

① filter data for 2016

ds[ds['year'] == 2016]

② group by brand

.groupby('brand')

③ ['rating'].count

2016

	Amt	grading	Id	
A	1	5	1	
	2	2	2	
	3	3	5	④
	4	5	7	
B	7	2	9	
	6	3	7	③
	4	5	1	

C → 2

D → 11

E → 20

④ . sort values

E → 20

D → 11 ⇒ head(10)

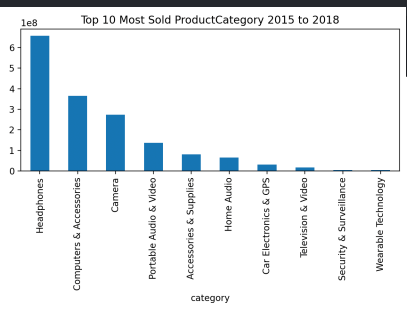
A → 4

B → 3

C → 11

v.) 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 ProductCategory 2015 to 2018'))

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

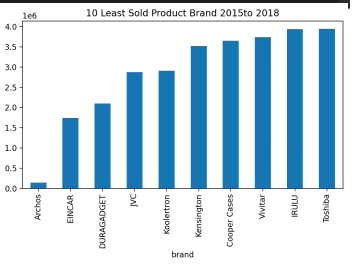


vi.)What product by category sold the least between 2015 to 2018?

```
dataset2015_2018 = dataset[(dataset['year'] >= 2015) & (dataset['year'] <= 2018)]

dataset2015_2018.groupby('category')['amount'].sum().sort_values(ascending=True).head(10).plot(kind='bar',title='10 Least Sold Product Brand2015 to 2018')
```

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
```

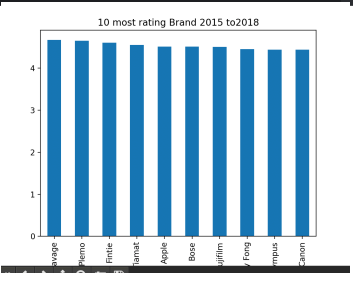


viii.) Ratings Distribution

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

dataset = pd.read_csv('/Users/arunkumarsharma/Documents/cleaned.csv')

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

plt.show()
```

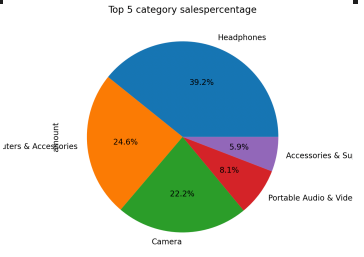
ix.) Best rated brands

```
# 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 mos

plt.show()
```



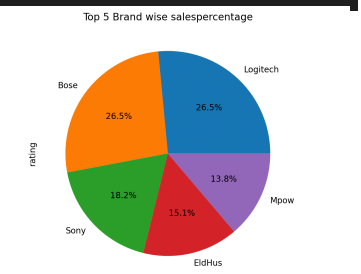
x) Top 5 category sales percentage

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



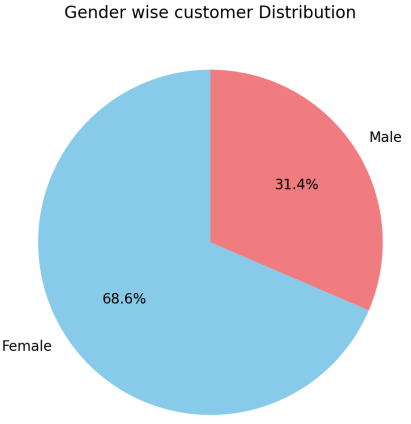
xi) Brand wise sales percentage

```
# 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 salespercentage'))
```



xii) Gender wise customer distribution

```
# Gender wise customer distribution
gender_distribution = dataset['gender'].value_counts()
# print(gender_distribution)
# Female    30963
# Male      14203
plt.pie(gender_distribution, labels=gender_distribution.index,
autopct='%1.1f%%', startangle=90, colors=['skyblue', 'lightcoral'])
plt.title('Gender wise customer Distribution')
```



Q. What was the best-selling product in each year?

```
dataset = pd.read_csv('/Users/arunkumarsharma/Documents/cleaned.csv')

for year in dataset['year'].unique():
    top_product = dataset[dataset['year'] == year].groupby('brand')['amount'].sum().idxmax()
    print(f'Top product in {year}: {top_product}')
```

.dataset['year'].unique() → Finds all unique years in the dataset.
.groupby('product_name')['amount'].sum() → Sums up the total sales amount for each product.
.idxmax() → Finds the product with the highest sales.

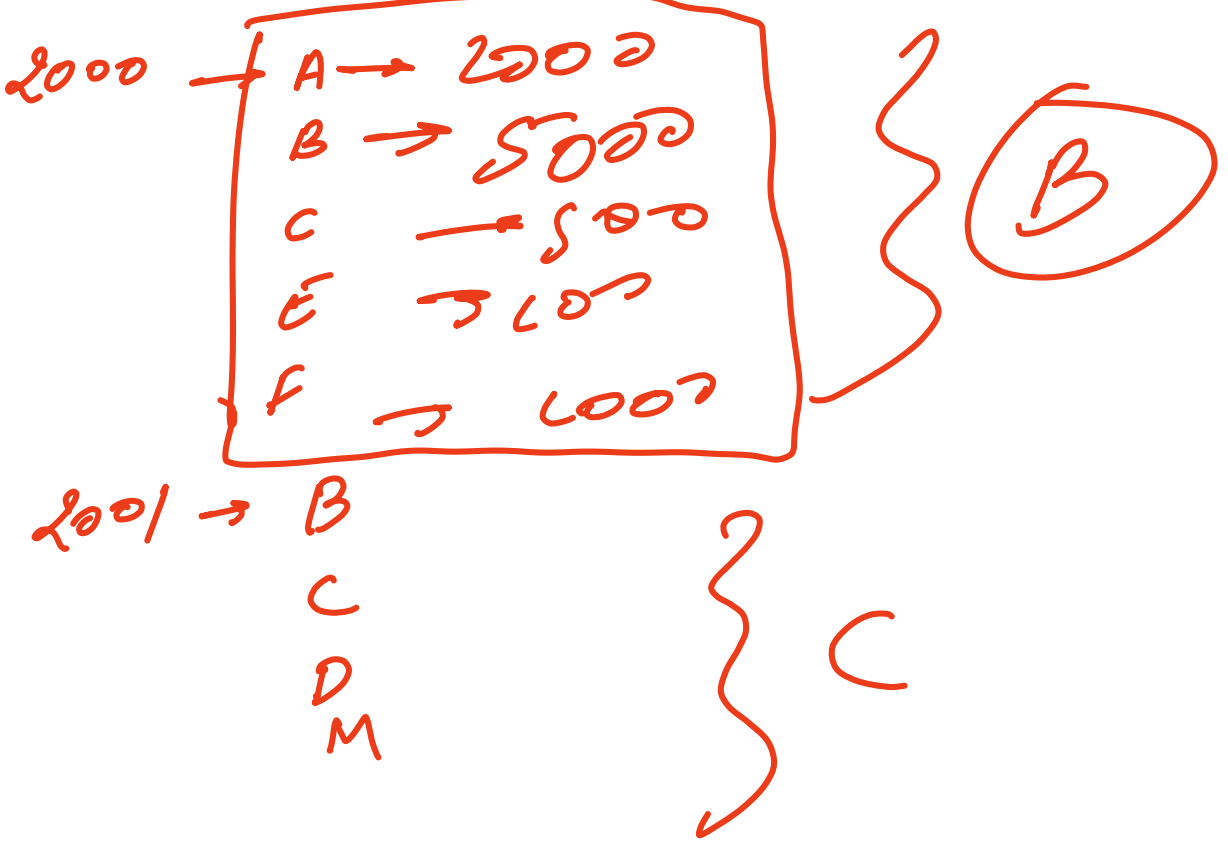
```
dataset = pd.read_csv('/Users/arunkumarsharma/Documents/cleaned.csv')

for year in dataset['year'].unique():  # Loop through each unique year
    top_product = dataset[dataset['year'] == year]  # Filter data for that year
    top_product = top_product.groupby('brand')['amount'].sum()  # Sum sales for each product
    top_product = top_product.idxmax()  # Find the product with the highest sales
    print(f'Top product in {year}: {top_product}')  # Print the result
```

Q. What is the sales distribution across different cities?

```
dataset = pd.read_csv('/Users/arunkumarsharma/Documents/cleaned.csv')

dataset.groupby('city')['amount'].sum().sort_values(ascending=False).head(10).plot(kind='bar', title='Top 10 Cities b
plt.xlabel('City')
plt.ylabel('Total Sales')
plt.show()
```

Q. Which payment method is most preferred by customers?

```
dataset = pd.read_csv('/Users/arunkumarsharma/Documents/cleaned.csv')

dataset['payment_method'].value_counts().plot(kind='pie', autopct='%1.1f%%', title='Payment Method Distribution')
plt.show()
```


2000
A → 2000
B → 5000
C → 500
E → 10
F → 1000
B
2001 → B
C
D
M
C