Exploratory Data Analysis (EDA)

Prepared by:

Leonard Andrew D. Mesiera
DLL - Information Technology Professor 1

Introduction to Data Exploration

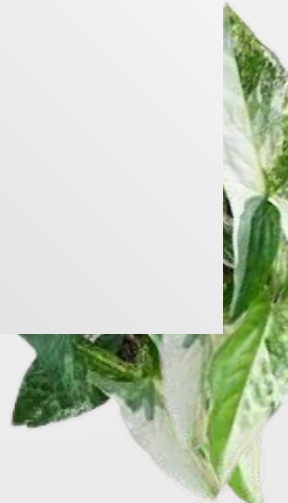
What is EDA?

What is EDA?

- A crucial step in data analysis to summarize main characteristics of a dataset.
- Helps uncover patterns, detect anomalies, and check assumptions using statistics and visualization.
- Prepares data for modeling by handling missing values and identifying trends.

- **Why is EDA Important?**

- Understands data distribution and relationships.
- Identifies data cleaning needs.
- Supports better decision-making before applying machine learning models.



EDA Workflow

1. **Load the Dataset**
2. **Understand the Structure** (Shape, Columns, Data Types)
3. **Handle Missing Values**
4. **Summary Statistics**
5. **Visualizations for Insights**
6. **Detect Outliers and Anomalies**
7. **Feature Correlation Analysis**

Summary Statistics

- Basic statistical overview:

```python
df.describe()
df.describe(include='all')  # Includes categorical variables
```

- Value counts for categorical data:

```python
df['category_column'].value_counts()
```

Loading the DataSet

- Load data from CSV, Excel, or other sources:

```python
df = pd.read_csv('data.csv')  # CSV file
df = pd.read_excel('data.xlsx')  # Excel file
```

- Display the first few rows:

```python
df.head()
```

- Check data structure:

```python
df.info()
df.shape  # (rows, columns)
```

Handling Missing Values

- Detect missing values:

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

- Fill missing values:

```python
df.fillna(method='ffill', inplace=True)  # Forward fill
df.fillna(method='bfill', inplace=True)  # Backward fill
df.fillna(df.mean(), inplace=True)  # Replace with mean (numerical columns)
```

- Drop missing values:

```python
df.dropna(inplace=True)
```

Aggregating Data

What is Data Aggregation ?

Broad Definition.

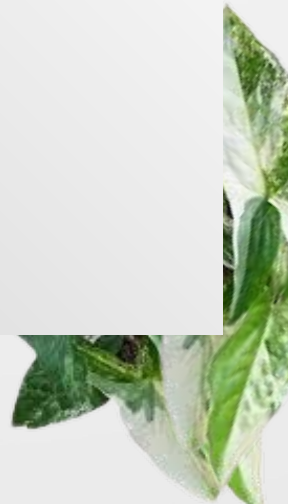
Data aggregation is the process of collecting, organizing, and summarizing data to create a simplified view. For example, instead of analyzing every sales transaction individually, you can group the data by month and calculate the total sales for each month.

What is Data Aggregation ?

Definition based on Data Analytics.

Definition: The process of combining multiple values into a summary metric (e.g., sum, mean, count).

- **Why is it useful?**
  - Helps detect trends and patterns.
  - Reduces dataset complexity for better visualization.
  - Essential for generating meaningful insights.
- **Common Aggregation Functions:**
  - sum(), mean(), median(), min(), max(), count(), std()



Why is it important to aggregate data ?

Why is it Important in Data Analytics?

1. **Simplifies Large Datasets** : It reduces complex, raw data into meaningful summaries (like averages, sums, or counts) that are easier to analyze.
2. **Identifies Trends and Patterns** : Aggregation helps spot trends over time, across locations, or within specific groups.
3. **Improves Decision-Making** : Summarized insights help businesses make informed decisions, like forecasting sales or optimizing resources.
4. **Supports Data Visualization** : Aggregated data is easier to visualize in charts or dashboards, enhancing comprehension.

Aggregation

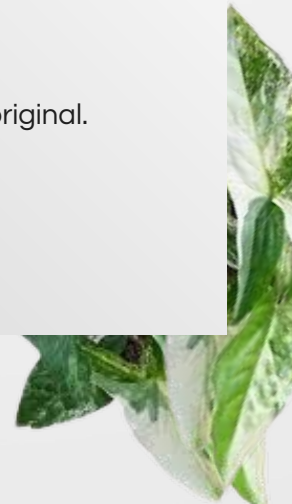
To start aggregating data in Pandas, the `groupby()` function is specifically designed to facilitate aggregation and other operations.

The `groupby()` function is a method of the Pandas `DataFrame` class. It works by:

1. **Splitting** the data into groups based on specified criteria.
2. **Applying** a function to each group independently.
3. **Combining** the results into a new data structure.

When applied to a DataFrame, `groupby()` returns a `GroupBy` object, which acts as the foundation for several data manipulation operations, including:

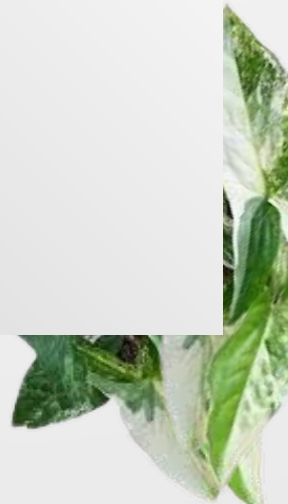
- **Aggregation** : Computing summary statistics (e.g., `mean`, `sum`, `count`) for each group.
- **Transformation** : Applying functions to each group and returning modified data with the same structure as the original.
- **Filtration** : Selecting groups that meet specific criteria or conditions.
- **Iteration** : Iterating over individual groups or their values for custom operations.



groupby()

groupby()

A pandas DataFrame method that groups rows of the data frame together based on their values at one or more columns, which allows further analysis of the groups

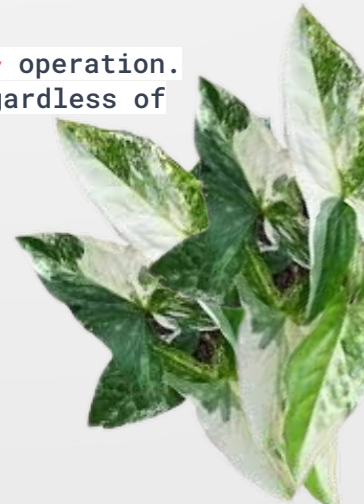


Built in Aggregate Function

Built-in aggregation functions

The previous examples demonstrated the mean(), min(), and size() aggregation functions applied to groupby objects. There are many available built-in aggregation functions. Some of the more commonly used include:

- count(): The number of non-null values in each group
- sum(): The sum of values in each group
- mean(): The mean of values in each group
- median(): The median of values in each group
- min(): The minimum value in each group
- max(): The maximum value in each group
- std(): The standard deviation of values in each group
- var(): The variance of values in each group
- size() : `.size` counts the total number of rows in each group after performing a `groupby` operation.
    a. Unlike `.count()`, which excludes missing (`NaN`) values, `.size` includes all rows, regardless of whether they contain missing values.



Correlation v Causation

Understanding the Difference in Data Analysis

Definitions

- **Correlation**: A relationship between two variables (they move together).
  - Example: *Higher temperatures → More ice cream sales*
  - Does **not** imply one causes the other.
- **Causation**: One event directly causes another.
  - Example: *Lightning → Thunder*

Types of Correlation

Positive Correlation: Both variables increase or decrease together.

Negative Correlation: One variable increases while the other decreases.

No Correlation: One variable changes, but the other remains unchanged.

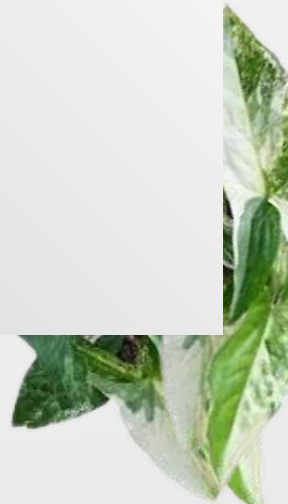
Why it Matters

Correlation **does not** always mean causation.

Misinterpreting data can lead to false conclusions.

Example: **Ice cream sales & hot weather**

- Higher temperatures **may** lead to more ice cream sales.
- Other factors (e.g., discounts) could also influence sales.



Case Study 1 - Cause of Disease

Early 1900s: Pellagra (disease) was linked to unsanitary conditions.

Later research: Pellagra was caused by **niacin deficiency**, not poor sanitation.

Lesson: Correlation (unsanitary living & pellagra) ≠ Causation.

Case Study 2 - Distribution of Aid

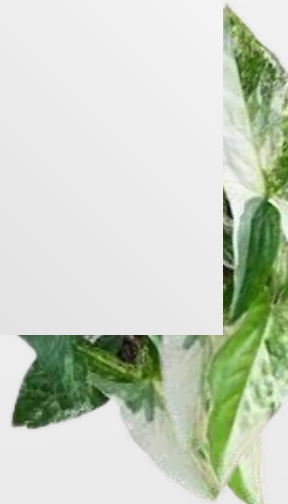
- Government agency observes that eligible users **visit but don’t sign up** for SNAP benefits.
- Possible correlations (e.g., frequent visits, quick exits) provide **clues**.
- **Solution**: Conduct a survey to determine the **true cause**.

Key take aways

Critically analyze any correlations.

Check context before concluding causation.

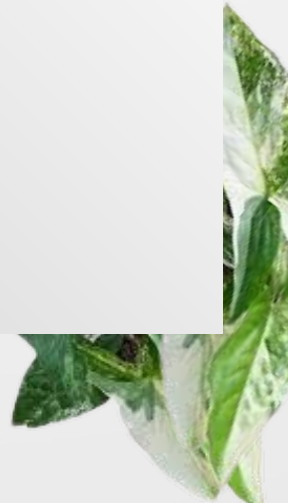
Know tool limitations—analytics show patterns, not direct causes.



Data Visualization

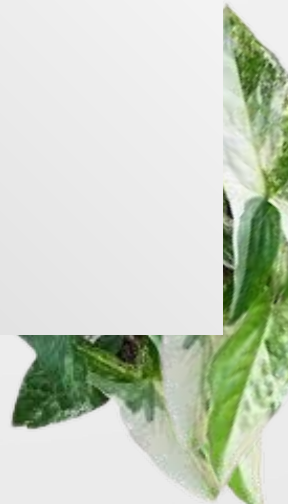
Data Visualization

data visualization literally means to visualize (make something visible) data. There are many different ways this can be done, and the most common of these will be covered in this course section.



Data Visualization

- **What is Data Visualization?**
  Graphical representation of data.
- **Why Does It Matter?**
  - Audiences may struggle with complex data.
  - Your role: Communicate analysis in a **meaningful, engaging, and easy-to-understand** way.
- **Why It Works:**
  - Eyes are drawn to **colors, shapes, and patterns**.
  - Visual elements tell a story that goes beyond numbers.

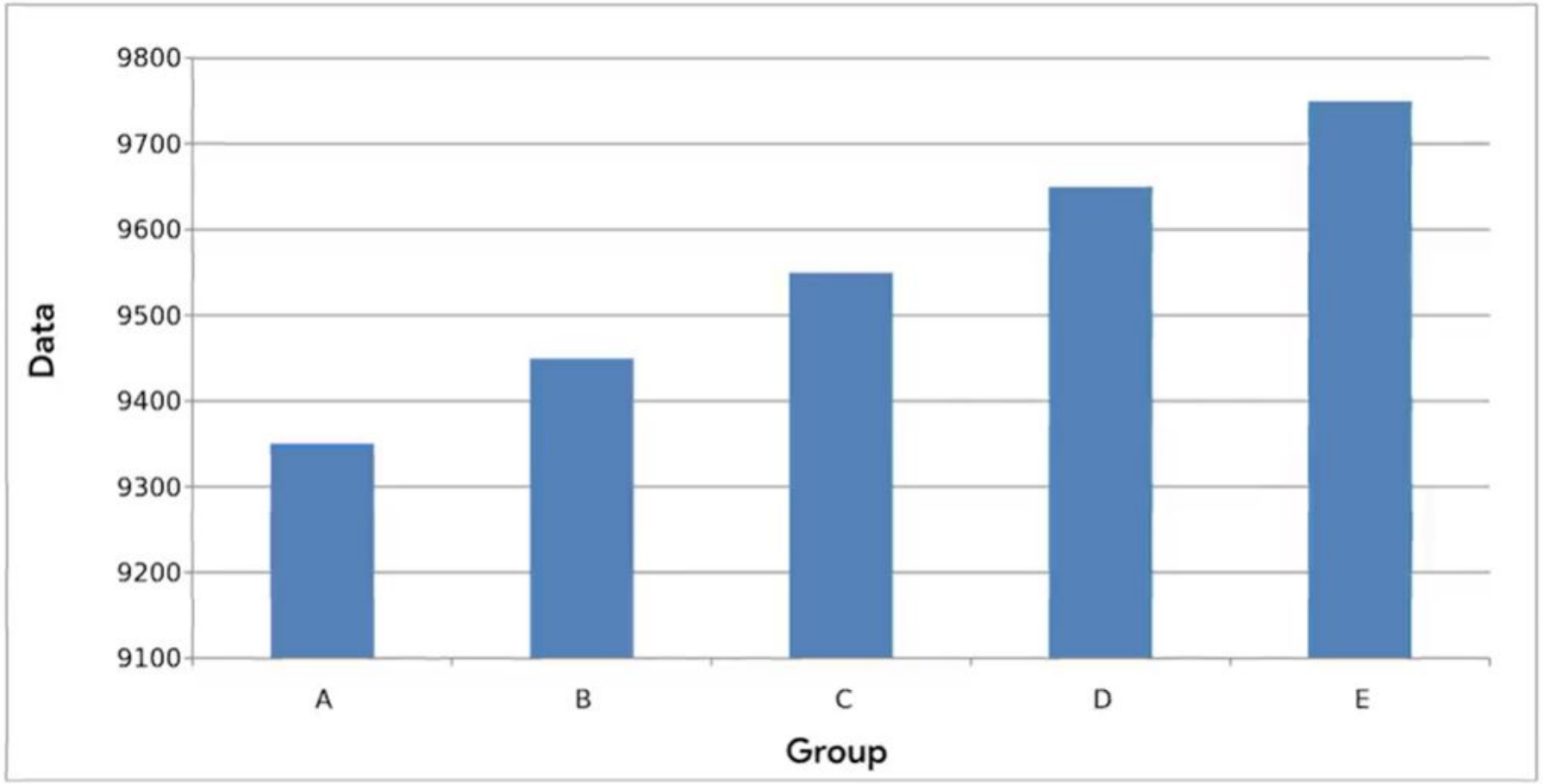


Bar graphs

Use size contrast to compare two or more values

Bar Chart

A *bar chart* uses either horizontal or vertical bars to display discrete, numerical comparisons across categories. One axis shows the categories being compared, while the other axis represents discrete values. Bar charts differ from histograms in that they do not display continuous developments over a period of time.

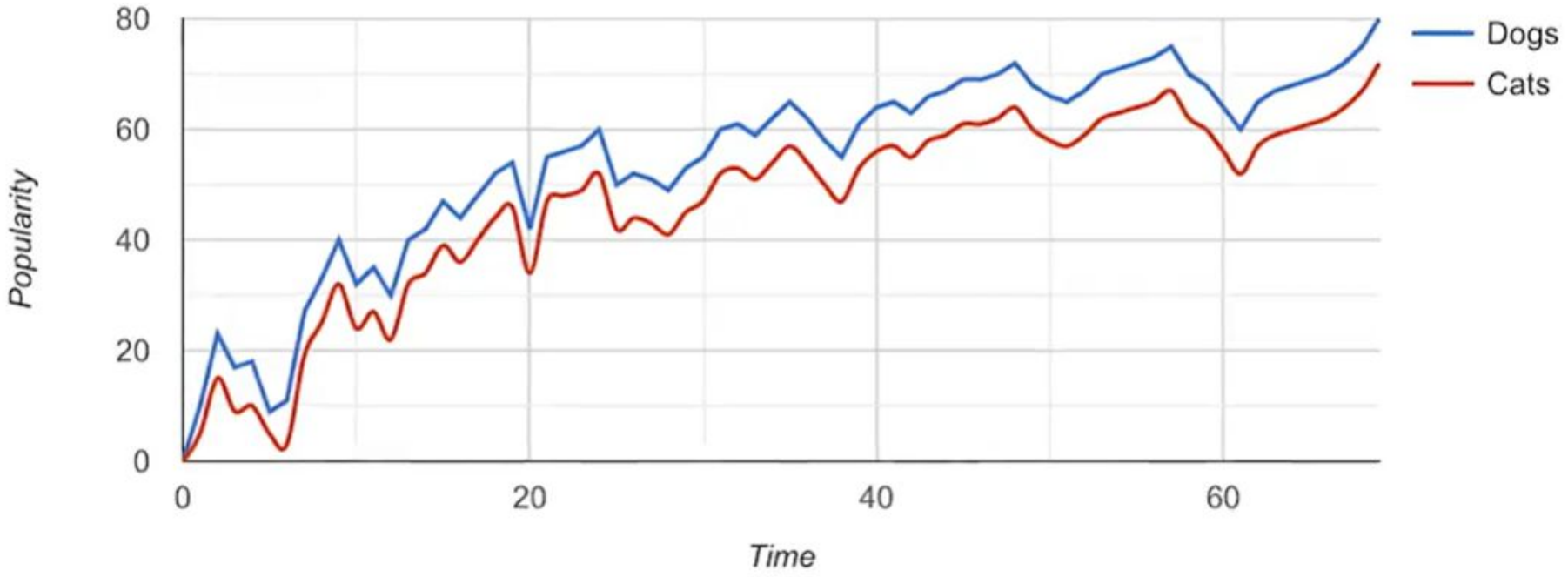

9800
9700
9600
9500
9400
9300
9200
9100
Data
A
B
C
D
E
Group

Line graphs

Help your audience understand shifts or changes in your data

Line Chart

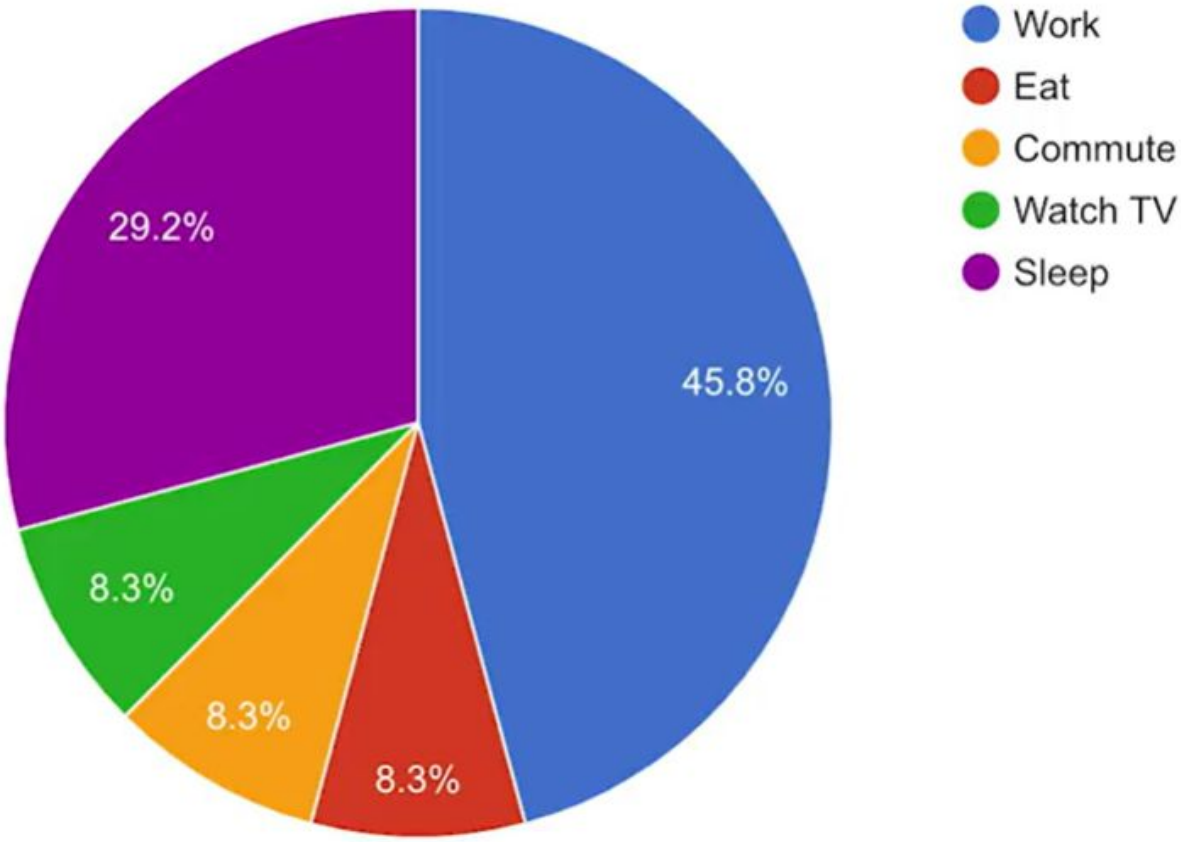
A *line chart* is used to display quantitative data over a period of time – allowing for a better understanding of trends. It is the most common method to display statistics. Line charts are also called *line plots* or *line graphs*. They look like this:


80
60
40
20
0
Popularity
0
20
40
60
Time
Dogs
Cats

Pie charts

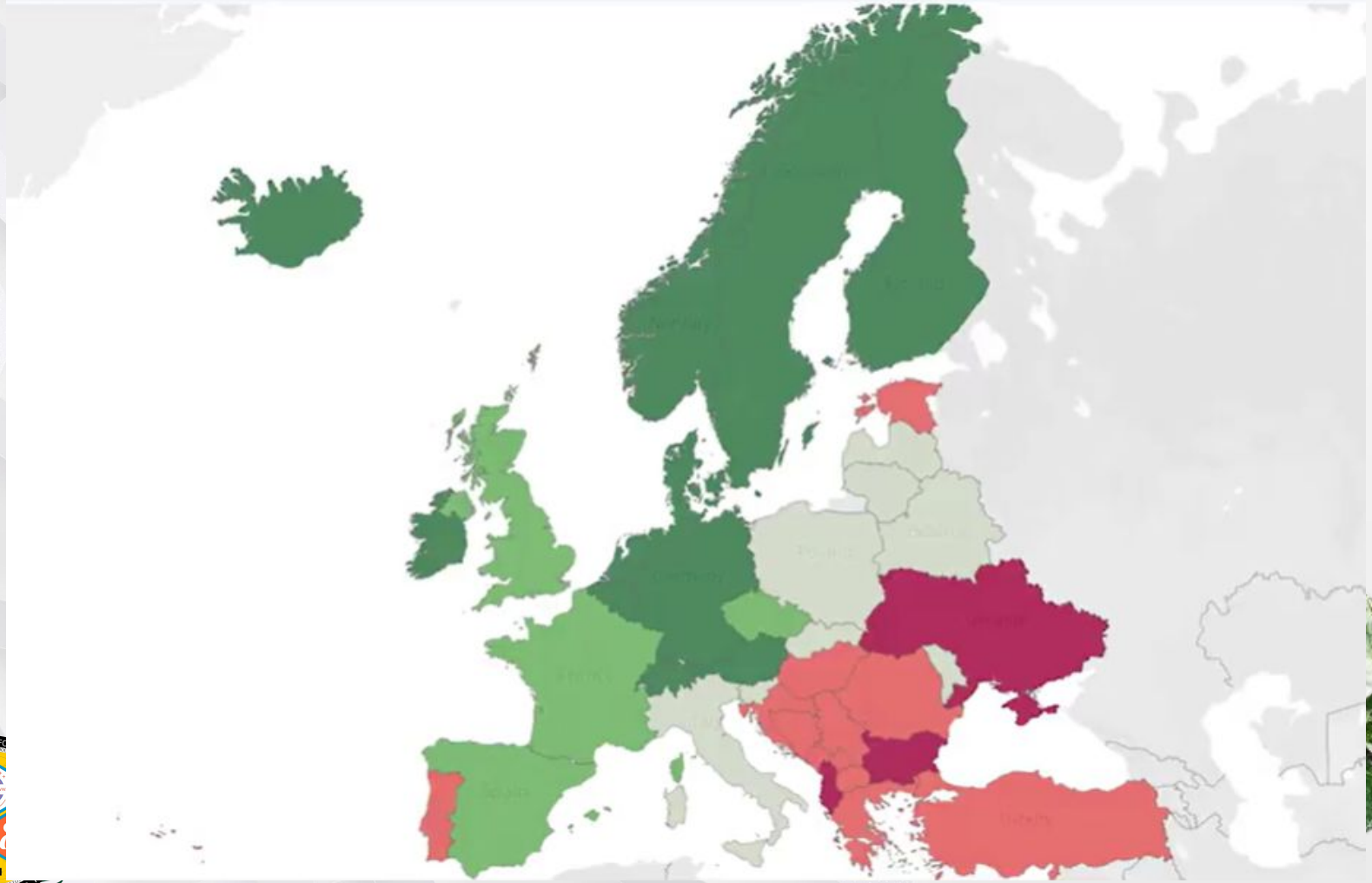
Show how much each part of something makes up the whole

My Daily Activities



Maps

Help organize data geographically



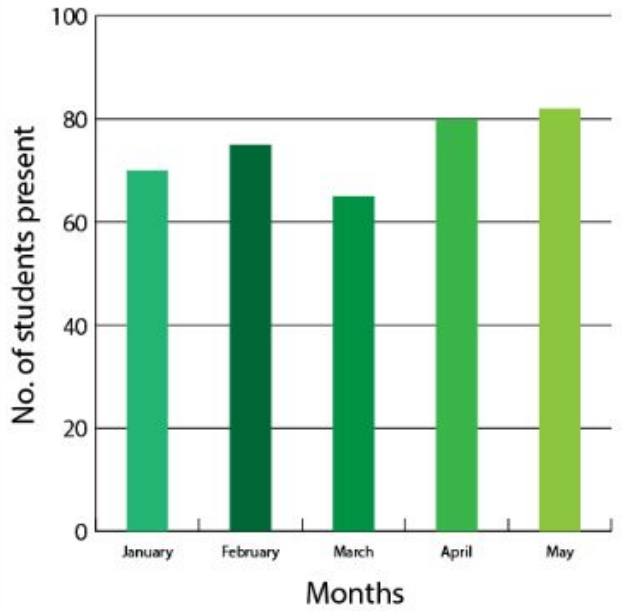
Histogram

A chart that shows how often data values fall into certain ranges

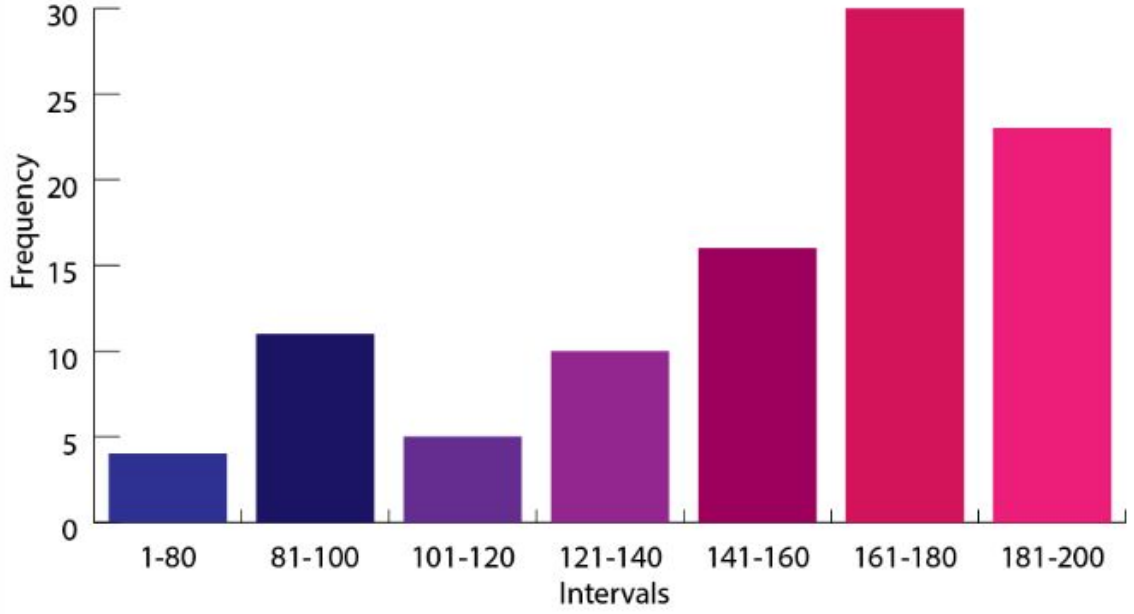
Histogram

A *histogram* is a diagram that is made of vertical rectangles used to show the frequency of variables and other data. It is similar to a bar graph.

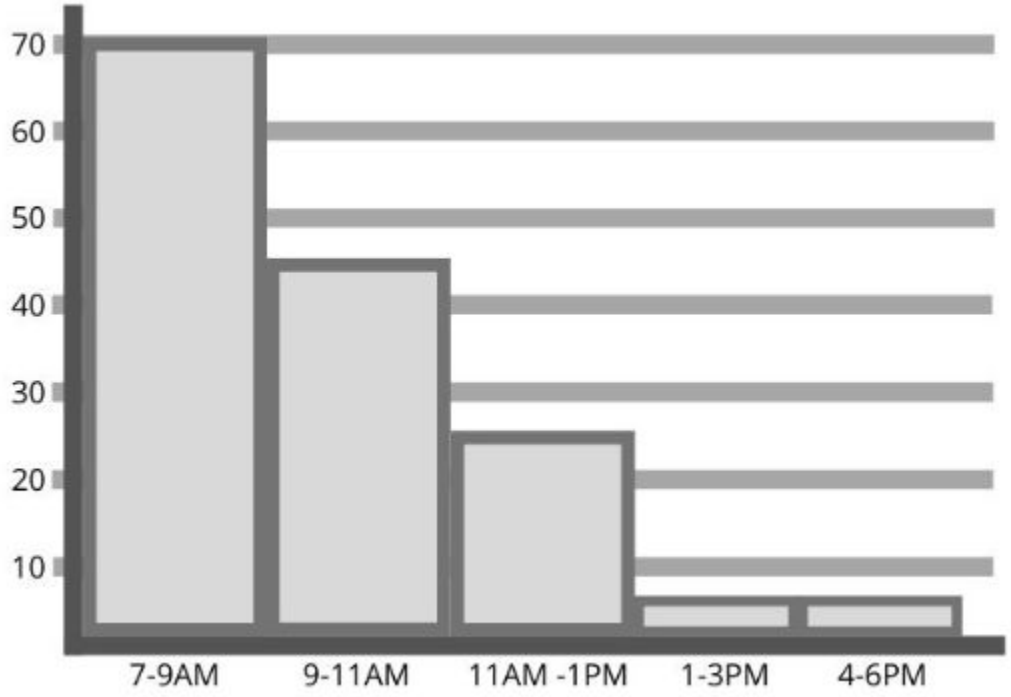
Bar Graph



Histogram



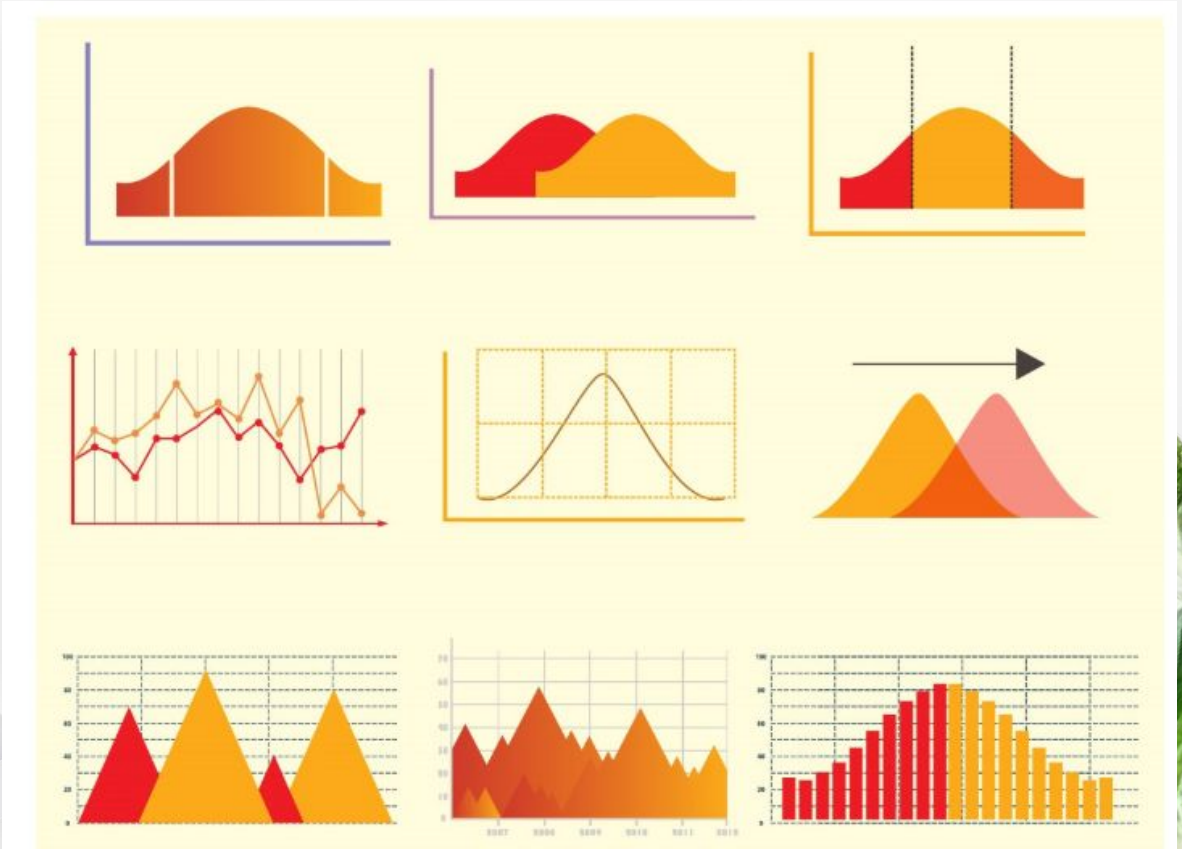
A histogram is commonly used to show the frequency distribution of data. For example, in the previous dot plot of coffee sold by the hour, we can section the times from 7-9, 9-11, 11-1, and so on.



Histogram

Statistical Distribution plots

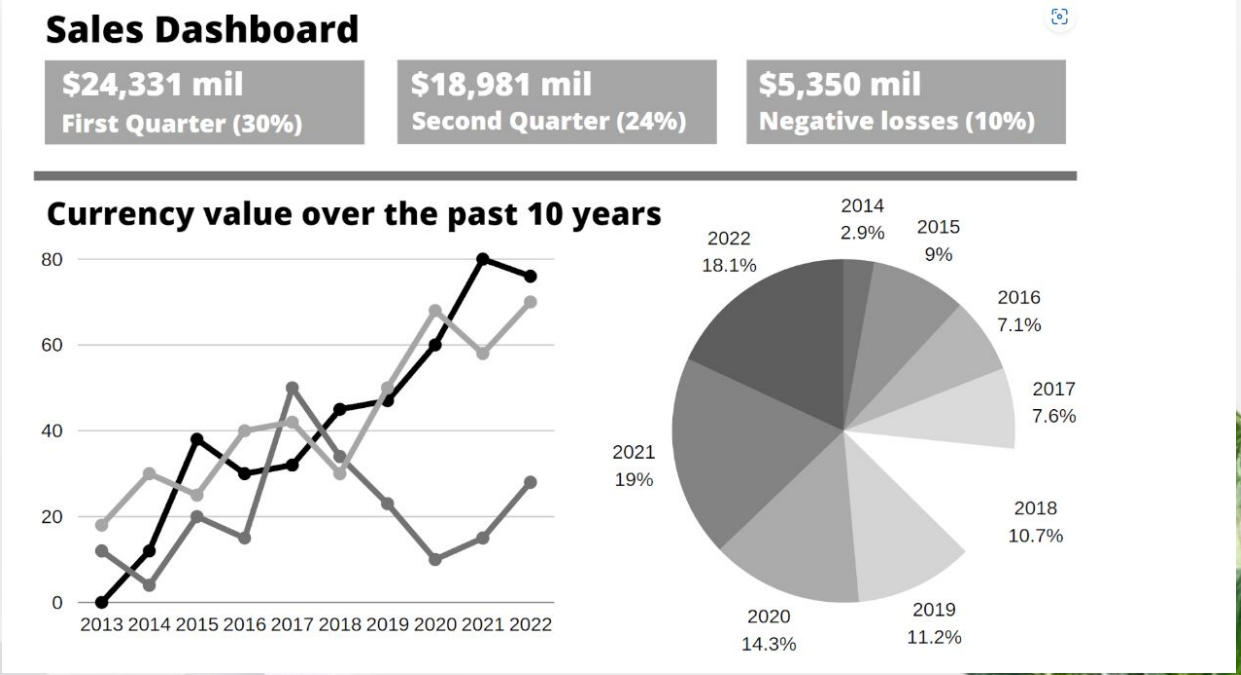
The *distribution* of a statistical data set shows us all the possible values or intervals of data, as well as how often each piece of data occurs and is observed. There are various methods of statistical distribution.



Dashboard

Dashboard

A *dashboard* is a data visualization tool that can be customized for viewing the overall data at a glance.



INTRODUCTION TO DATA VISUALIZATION WITH MATPLOTLIB

What is Matplotlib?

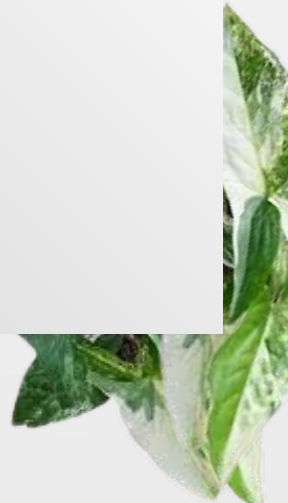
- Matplotlib is a powerful Python library for creating static, animated, and interactive visualizations.
- It is widely used in data science, machine learning, and research.
- Supports a variety of plot types: line plots, bar charts, scatter plots, histograms, etc.

Why use Matplotlib ?

- Easy to use and highly customizable.
- Integrates seamlessly with NumPy and Pandas.
- Supports publication-quality figures.
- Extensive community support and documentation.

Anatomy of Matplotlib

- **Figure** : The entire canvas where plots are drawn.
- **Axes** : The area where data is plotted (can have multiple axes).
- **Title** : Describes the plot.
- **X-axis and Y-axis Labels** : Describe the dimensions of the data.
- **Legend :** Explains different elements in the plot.
- **Ticks and Tick Labels** : Mark points on the axes.



Setting up Matplotlib

Install Matplotlib using pip:

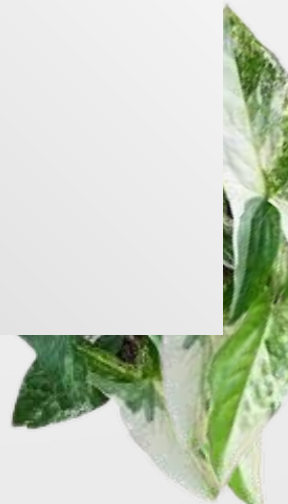
```
pip install matplotlib
```

Import Matplotlib in Python

```
import matplotlib.pyplot as plt
```

Basic workflow:

- Create data.
- Create a plot.
- Customize the plot.
- Display the plot.



Example 1 - Line Chart

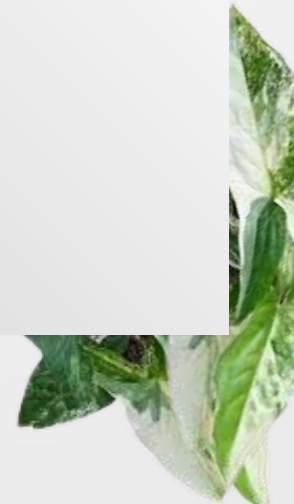
```
import matplotlib.pyplot as plt

# Data
x = [1, 2, 3, 4, 5]
y = [2, 4, 6, 8, 10]

# Create plot
plt.plot(x, y)

# Add labels and title
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Line Plot Example")

# Show plot
plt.show()
```



Example 2 - Bar Chart

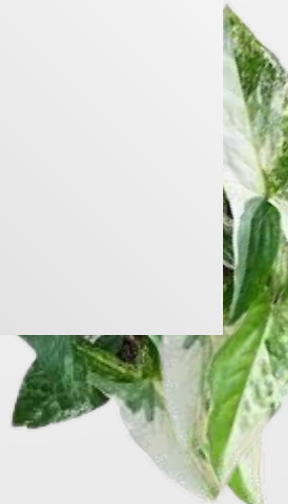
```
import matplotlib.pyplot as plt

# Data
categories = ['A', 'B', 'C', 'D']
values = [5, 7, 3, 8]

# Create bar chart
plt.bar(categories, values, color='skyblue')

# Add labels and title
plt.xlabel("Categories")
plt.ylabel("Values")
plt.title("Bar Chart Example")

# Show plot
plt.show()
```



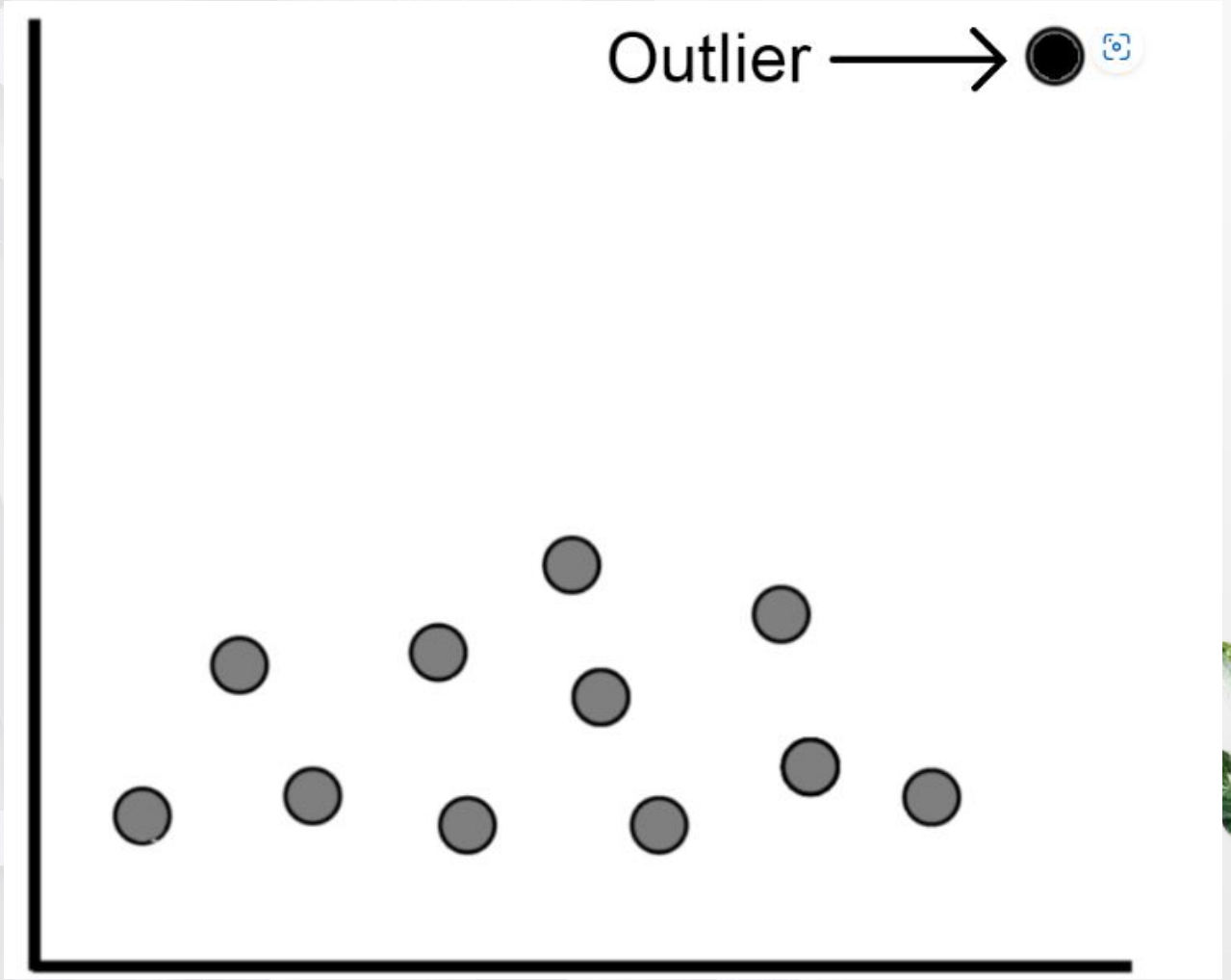
Gaps and outliers

Gaps and Outliers

A *gap* in data indicates one of two things:

- There was no data to begin with, or
- It wasn't recorded correctly and, therefore, it is rendered useless.

An *outlier* is a data point that is far different from the other data points, creating a large gap in the statistic. The outlier has the potential to cause the statistic to be inaccurate.

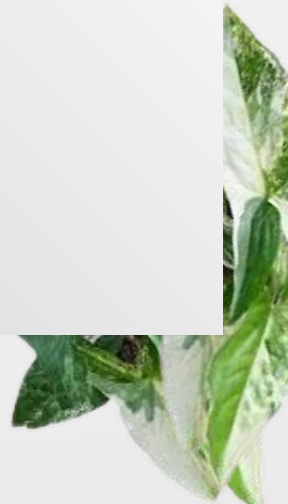

Outlier

Reference

1. Data visualization catalog
    a. https://datavizcatalogue.com/#google_vignette
2. 30 of the best Visualizations
    a. https://visme.co/blog/best-data-visualizations/
3. Information is Beautiful
    a. https://informationisbeautiful.net/wdvp/gallery-2019/
4. Data Studio Gallery
    a. https://lookerstudio.google.com/gallery?category=visualization

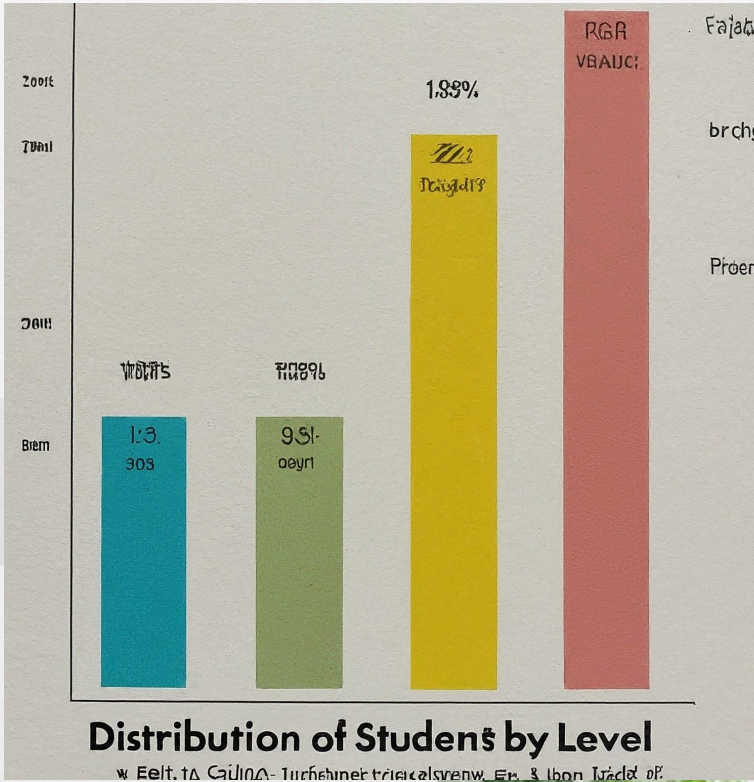
Visualizing Data

- **Key Role of a Data Analyst:**
  Communicate findings in a way that **resonates with your audience**.
- **Why Data Visualization Matters:**
  - Simplifies complex or monotonous information.
  - A **valuable skill** for effective communication.
- **Your Goal:**
  - Help the audience **"have a conversation" with the data**.
  - Use visuals to **draw them into the discussion**.
- **Example Use Case:**
  Visualizing large datasets, like the **flow of goods between countries**.



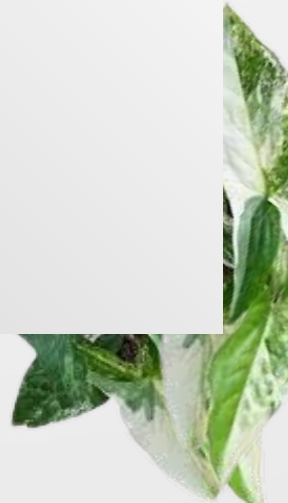
Seaborn

Unveiling Advance Visualization



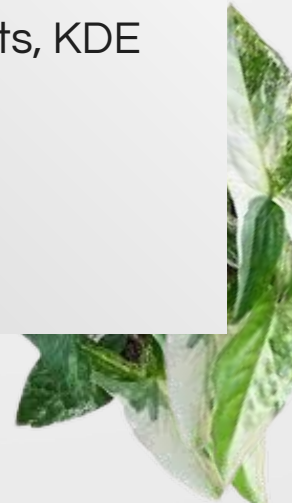
Why use SEABORN?

- Effortless creation of various plot types (scatter plots, bar charts, violin plots, etc.)
- Built-in themes and styles for a polished look
- Easy customization for a touch of personalization
- Seamless integration with pandas DataFrames



Exploring Data with Seaborn

- **Informative visualizations:** Create clear and concise plots that effectively communicate insights from your data.
- **Relationships uncovered:** Explore connections between variables using scatter plots and correlation matrices.
- **Distributions visualized:** Understand how data is spread out with histograms, violin plots, and box plots.
- **Comparisons made easy:** Compare categories or groups with bar charts, KDE plots, and joint plots.



What is EDA ?

- EDA is the initial investigation of data.
- It's like getting to know someone before becoming best friends with them.
- We use various techniques to understand the data's characteristics and relationships.

Why is EDA important ?

- Builds a strong foundation for further analysis.
- Helps us make informed decisions about data cleaning and modeling.
- Uncovers hidden insights and patterns that might be missed otherwise.

EDA in Jupyter Notebook

- Jupyter Notebook is a fantastic platform for EDA.
- It allows us to combine code, data, and visualizations seamlessly.
- We can import libraries like pandas and seaborn for data manipulation and visualization.

Steps of EDA in Jupyter Notebook

- Import libraries (pandas, seaborn, etc.)
- Load the data into a pandas DataFrame.
- Check data types and identify missing values.
- Explore data distribution with summary statistics and visualizations (histograms, boxplots, scatter plots).
- Analyze relationships between variables (correlation analysis).

EDA

- EDA is the foundation of successful data analysis.
- By understanding our data, we can make informed decisions.
- Jupyter Notebook provides a powerful platform for EDA.

Feature Engineering

Feature Engineering