ISBN: 978-93-48620-52-1

FUNDAMENTALS OF DATA HANDLING & VISUALIZATION

Editors:
Dr. Sumit Chopra
Er. Navjot Kaur Basra
Er. Simran
Er. Debjit Mohapatra

Bhumi Publishing

www.bhumipublishing.com

Fundamentals of Data Handling and Visualization

(ISBN: 978-93-48620-52-1)

Editors

Dr. Sumit Chopra

Er. Navjot Kaur Basra

Er. Simran

Er. Debjit Mohapatra

Bhumi Publishing

April 2025

Title: Fundamentals of Data Handling and Visualization

Editors: Dr. Sumit Chopra, Er. Navjot Kaur Basra, Er. Simran, Er. Debjit Mohapatra

First Edition: April 2025

ISBN: 978-93-48620-52-1

Published by:

BHUMI PUBLISHING

Nigave Khalasa, Tal – Karveer, Dist – Kolhapur, Maharashtra, INDIA 416 207

E-mail: bhumipublishing@gmail.com

PREFACE

In today's data-driven world, the ability to interpret and communicate insights from data effectively is more vital than ever. As vast amounts of information are generated every moment, the challenge lies not just in data collection but in making sense of it and presenting it in a meaningful and accessible manner. Fundamentals of Data Handling and Visualization is designed to provide readers with a foundational understanding of the principles and practices that underpin effective data visualization.

This book brings together essential concepts ranging from the basics of data visualization to the use of advanced tools like Tableau, offering a comprehensive roadmap for students, researchers, and professionals across disciplines. Starting with an Introduction to Data Visualization, the text progresses to explore how data is mapped onto various aesthetic dimensions to uncover patterns and relationships. Readers will gain insights into how to represent trends and uncertainty, and how to apply the principle of proportional ink, ensuring clarity and honesty in representation.

Further, the book highlights the significance of color usage, a critical component in enhancing both readability and visual appeal. The latter half of the book delves into Tableau, a leading data visualization tool, guiding the reader from basic chart creation to advanced features and dashboard development. It also covers the art of storytelling through data, emphasizing how narrative techniques can transform data into compelling and persuasive stories. The final chapter reflects on emerging trends in the field, preparing readers for the evolving future of data visualization.

Each chapter is structured to blend theoretical concepts with practical applications, encouraging hands-on exploration and critical thinking. Whether you are a beginner stepping into the world of data or a practitioner looking to enhance your visualization skills, this book serves as a valuable resource to help you visualize data not just beautifully, but meaningfully.

We hope that this book not only equips readers with technical know-how but also inspires them to approach data visualization as both a science and an art.

- Editors

TABLE OF CONTENT

Sr. No.	Book Chapter and Author(s)	Page No.
1.	**Introduction to Data Visualization** Navjot Kaur Basra, Davinder Singh, Kiranjit Kaur	1 – 36
2.	**Mapping Data onto Aesthetics** Simran**,** Satyam Sharma, Gurpreet Kaur	37 – 86
3.	**Visualizing Trends and Uncertainty** Ramandeep Kaur, Ritu Rani, Sahib Singh, Navdeep Kaur	87 – 111
4.	**The Principle of Proportional Ink** Arshdeep Singh, Vikramjit Parmar, Ramandeep Kaur	112 – 144
5.	**Color Usage in Data Visualization** Shruti, Arshdeep Singh, Neharika Sharma, Sumit Chopra	145 – 154
6.	**Introduction to Tableau** Vikramjit Parmar, Debjit Mohapatra, Suruchi, Shruti	155 – 172
7.	**Creating Data Visualization in Tableau** Suruchi, Gagandeep Singh, Arshdeep Singh	173 – 189
8.	**Advanced Features in Tableau** Jeevanjot Singh Chagger, Manpreet Singh, Paramjit Kaur	190 – 204
9.	**Storytelling in Data Visualization** Navdeep Kaur, Navjot Kaur Basra, Sumit Chopra	205 – 216
10.	**Case Studies on Real-World Data Visualization** Jasmeet Kaur, Babita Sidhu, Jaskiran Ghotra	217 – 220
11.	**Future Trends in Data Visualization** Manpreet Kaur, Simran, Tarun Bhalla	221 – 224
12.	**Image Enhancement Techniques in Digital Image Processing** Bhoomi Gupta, Gagandeep Singh, Sumit Chopra	225 – 242
13.	**Multi-Pose Guided Virtual Try-On Systems (MPGVTOS)** Gurnoor Singh, Sumit Chopra, Gagandeep Singh	243 – 254
14.	**References**	255 – 258

Chapter 1

INTRODUCTION TO DATA VISUALIZATION

Navjot Kaur Basra[1], Davinder Singh[2], Kiranjit Kaur[3]

[1]GNA University, Phagwara

[2]LKCTC, Jalandhar

[3]IKG PTU Main Campus, Kapurthala

1.1 Definition:

Data visualization is a graphical representation of information and data. Data visualization uses visual components such as charts and graphs and maps to help individual users and organizations understand trends, elaborate patterns and detect outliers in their datasets. Ultimately, it aims to make complex data more digestible, comprehensible, and useful for decision-making [1].

Data visualization is not so much about making the data pretty as it is about making sure the data conveys information. It helps one understand distributions, identify correlations, and detect underlying trends that may not be obvious through exploring raw numeric data. For example, hospitals in the healthcare sector rely on data visualization to maintain patient health records and predict disease outbreaks. The dashboard (as shown in fig. 1.1) should offer real-time patient stats and assist medical staff in detecting patterns of disease progression to facilitate resource allocation[1].

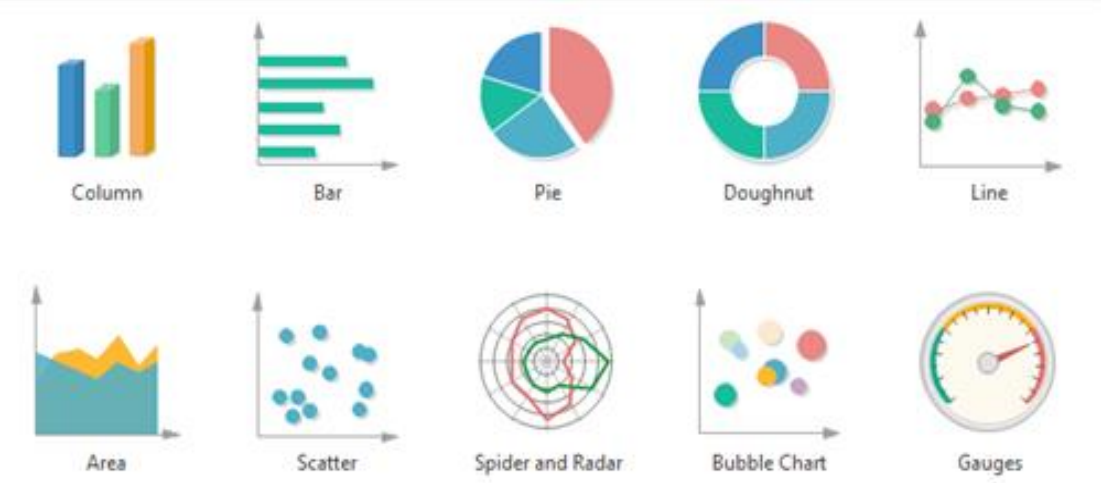


Fig. 1.1: Data Visualization

1.2 Methodology of Data Visualization:

Data visualization is a systematically structured process that facilitates proper processing, analysis, and abstract display. At least three main stages in the process all play an important role in the visualizing pipeline. A systematic way of visualization is taking unrefined data and shaping that into visual insights. The standard process comprises various phases through which data is transformed correctly so that it can exactly communicate the focal points in front of the target audience. Below is a concise overview of each of the steps in the methodology of data visualization (as shown in fig. 1.2) [2]:

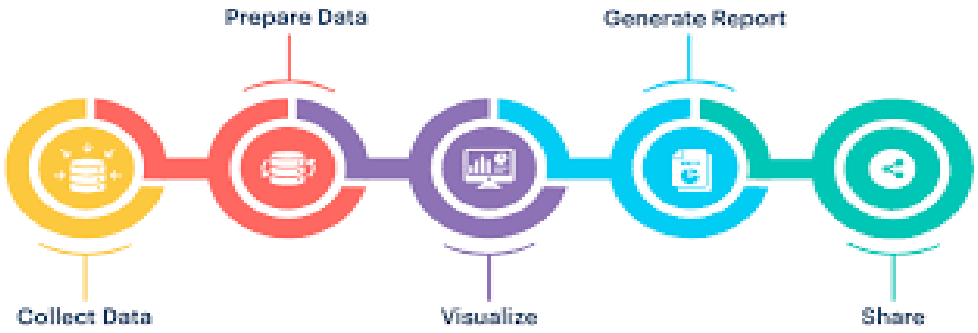


Fig. 1.2: Methodology of Data Visualization.

1.2.1 Understanding the data:

An understanding of the data is fundamental to all visualization. If there is no precise conviction on the nature of the underlying data within the dataset, any visualization created on top of it could potentially suggest wrong conclusions, or even worse, be pointless. This process comprises several key tasks [2]:

Identifying Data Types:

Data can be categorized into different types, each requiring specific handling and visualization methods:

1. **Categorical Data:** Such data would contain qualitative data of descriptive characteristics that cannot be quantitatively measured. This type includes data such as gender, race, and religion (as shown in fig. 1.3). Categorical datasets can further be divided into nominal and ordinal data. Nominal data categorized because of characteristics but not rank could include various colors, various types of fruit, etc. Ordinal data can be ordered into levels of education or ratings on a scale.

Categorical Data
Gender
Religion
Method of treatment
Type of teaching approach
Marital status
Qualifications
Native Language
Type of instruction
Problem-solving strategy used
Social classes

Fig. 1.3: Categorical data representation

2. **Numerical Data:**

Numerical data are those that contain quantitative data measured on a numerical scale. Age, height, weight, and income data types are the most common datasets (as shown in fig. 1.4). Further numerical datasets can be subdivided into discrete and continuous data. Discrete data are representable by integers and are countable, for example, the number of cars in a parking lot. Continuous data take on any value on a continuum; they can be presented as decimals or fractions-for example, temperature and weight.

Case	Attributes				Decision
	Length	Height	Width	Weight	Quality
1	4.8	1.2	1.6	0.8	high
2	4.8	1.4	1.8	0.8	high
3	4.8	1.4	1.8	0.8	high
4	4.4	1.4	1.6	1.0	medium
5	4.4	1.2	1.6	1.4	medium
6	4.2	1.2	1.8	1.4	low
7	4.2	1.8	1.8	1.4	low
8	4.2	1.8	1.8	1.4	low

Fig. 1.4: Presentation of numerical data.

3. Time-Series Data:

Aside from the fact that time-series data sets are used to observe changes over time, it may include sequential data like stock prices, weather patterns, and web traffic. Time-series data sets are arranged in such a way as to record progress through chronological order with a timestamp or date on each observation (as shown in fig. 1.5). Such a structure makes it possible to identify trends and patterns over time.

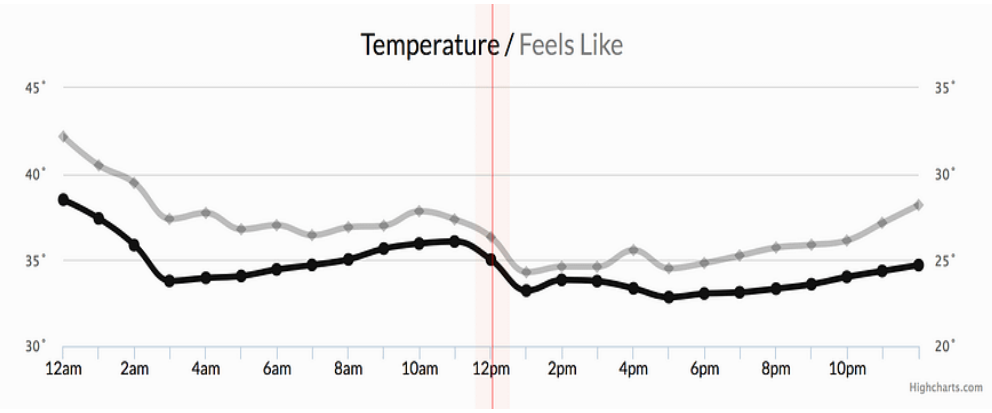


Fig. 1.5: Time series data.

4. Spatial Data:

Spatial datasets contain locational data, such as maps or GPS data. It may contain information about the locations of stores, the population density of a city (as shown in fig. 1.6), or the spread and incidence of certain diseases. Spatial datasets can have structured or unstructured forms, depending on their levels of detail and data format.

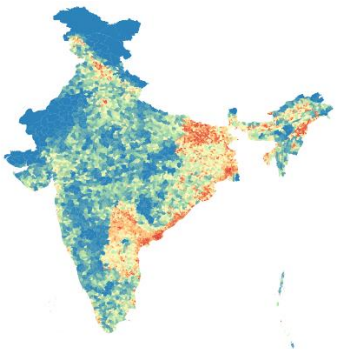


Fig. 1.6: Spatial data.

1.2.2 Determining the Source and Collection Method:

Knowing the source of data facilitates the assessment of reliability and accuracy. Data comes from different sources, such as:

- **Databases:** Simpler data stored in relational databases.
- **Real-time data:** Data acquired using an API or web service.
- **Manual Data Entry:** Data entered into spreadsheets, forms, and surveys.
- **Data from objects:** Data received from sensors and smart Internet-connected devices.

1.2.3 Accessing data quality:

Data Quality Assessment is a vital step in any research or data-related project, ensuring that the data collected is both accurate and reliable for analysis. Poor quality of data can lead to misleading conclusions and comparatively poorer performance of the model. Hence, this step is important (as shown in fig. 1.7).

Key Aspects of Data Quality Assessment:

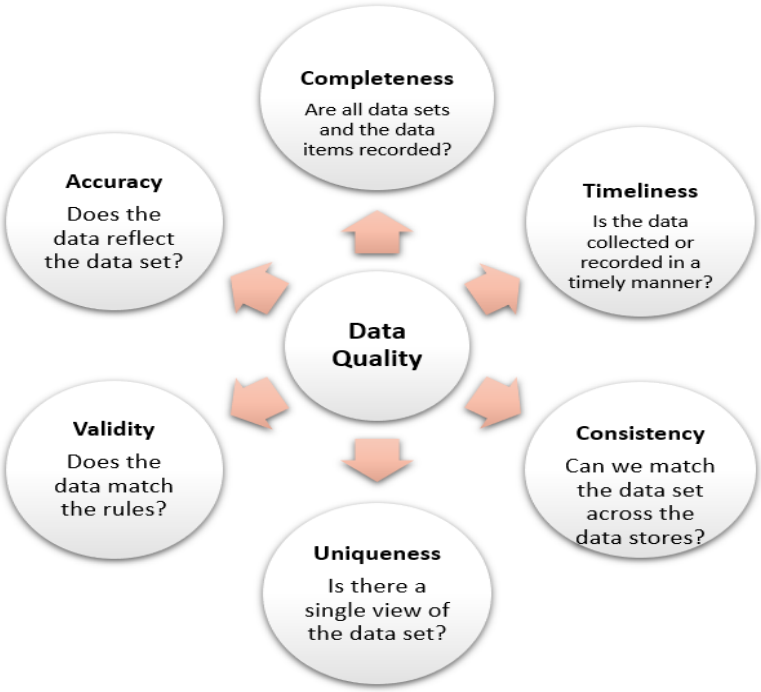


Fig. 1.7: Steps in accessing data quality

i. **Completeness:** Confirm that all the data fields in the dataset are present and have no missing data. Detect and address missing values by using imputation and deletion methods, among others.

ii. **Consistency:** Check with various data sources to ensure data is consistent over time.

iii. Check for variations in the way data has been input, especially with up-to-date formats and categorical labels.

iv. **Accuracy:** Data is checked systematically by comparing it with known or accepted sources or on some ground truth. Statistical techniques should be employed to search for outliers or anomalies, which may be evaluated by experts.

v. **Timeliness:** Analyze whether data are fresh and useful for the intended scholarly research or inquiry. The data should be checked to see if it is outdated and no longer useful.

vi. **Validity:** This checks whether the values in the dataset are within the limits defined by the rules or the specified formats (e.g., no alphabetical characters in numeric values). Constraints must be examined in terms of logical relationships between variables and acceptable age ranges.

vii. **Uniqueness:** Check for duplicates in records so they can be deleted to avoid redundancy. Make sure that wherever applicable, data represents a unique entity.

viii. **Integrity:** Relationships among different datasets must exist in the correct way (e.g., the maintenance of foreign key constraints in databases). Referential integrity checks ascertain that records that are referenced still exist and are valid.

1.2.4 Handling Missing or Inconsistent Data:

Handling any missing or inconsistent information forms a crucial part of preprocessing that makes any analysis results or prediction models reliable and accurate. Poor diagnosis may lead to batch effects that can bias results and yield inaccurate predictions or ineffective decision-making.

a) Identifying Missing or Inconsistent Data: Before handling, it is essential to identify missing or inconsistent values in the dataset. Some common indicators include:

- Null or NaN (Not a Number) values in datasets.
- Blank or empty fields in text-based datasets.
- Outliers or incorrect values, such as negative age values or inconsistent date formats.
- Mismatched categories, such as different spellings or formats of the same entry (e.g., "Male" vs "M").

b) **Methods for Handling Missing Data:** Certainly, data is missing for many reasons including but not limited to, technical issues, human error, privacy problems, data processing problems, or characteristics of the variable itself. Knowing what and why of these missing data guides you into different modes of handling the missing data and hence giving you a quality analysis.

Types of Missing Values: There are three main types of missing values [4]:

I. Missing Completely at Random (MCAR):

MCAR is a process of generating missing values during which the actual missingness of a variable occurs completely at random and is therefore unrelated to any other variables that have been collected in the dataset. In other words, there is nothing that associates whether a key value is truly missing or not with other variable values or characteristics of the observation itself [4].

II. Missing at Random (MAR):

MAR is a type of missing data where the probability of data being missed is affected by the existing independent variable but not influenced by the missing value of the variable itself. In other words, the mechanism for missingness is dependent on other observed values, meaning that it is not completely random [4].

III. Missing Not at Random (MNAR):

MNAR is the most complicated missing data phenomenon. It arises whenever a missing value itself relates to the probability that the actual data point is recorded. In other words, the reason behind the data missing is informative and essentially linked to the variable that is not complete [4].

c) Removing Missing Data:

- **Listwise Deletion (Complete Case Analysis):** Entire rows with any missing data will be removed. Suitable when the missing values are rare.
- **Risks:** Causes loss of information if the values are, however, significantly missing throughout the data set.
- **Column-wise Deletion:** Drop a column if it has large missing values. Useful where a column contains significantly missing values and is not too important.

d) Filling in Missing Values:

- **Mean/Median/Mode Imputation:** Substitute missing numerical values with the mean, median, or mode.
- **Example:** The average income of a data set is $50,000; therefore, missing values in the "Income" column should be replaced with $50,000. Best for numerical data with a normal distribution.
- **Forward Fill / Backward Fill:** Use previous or next known values to fill missing data in time-series datasets.
- **Interpolation:** Estimation of missing data using statistical or machine learning techniques.
- **K-Nearest Neighbours (KNN) Imputation:** Predict the missing values on the basis of other data points that are similar.
- **Regression Imputation:** Use regression models to predict the missing values using other available features.

e) Handling Inconsistent Data: Inconsistent data occurs when there are formatting issues, duplication, or logical errors.

- **Standardizing Formats:** Convert categorical data to consistent labels (e.g., "USA" vs. "United States"). Unify date formats (e.g., "01/02/2024" vs. "2024-02-01").
- **Correcting data entry errors:** Misspelled words or incorrect values are corrected using an automated tool or through manual review.
- **Handling duplicates:** Duplicated rows or entries are deleted.
- **Outlier detection and treatment:** Identify and deal with outliers using statistical methods such as Z-score or IQR.

1.2.5 Exploring and Understanding Data Relationships:

An examination of relationships between data constitutes the basic step of data analysis; they will accompany the patterns, dependencies, and insight that can support decision-making or lead to the improvement of machine learning models. During this step, statistical and visual techniques are used to analyse the interaction between different variables.

1. **Understanding Data Relationships:** Data relationships describe how variables influence one another. The key types of relationships include

i) Univariate Analysis (Single Variable Analysis): Univariate data consists of a single variable. Univariate data show that each observation or data point represents a single variable. In other words, it always describes some single attribute or quality of a single person or item in the dataset. The univariate data are very simple to analyze and are the simplest form of analysis in statistics [5].

Example: Consider the heights of the seven students in a class recorded in the preceding table. The only variable here is height and is not dealing with relationship or cause (as shown in Table 1).

Table 1: Univariate Data

Heights (in cm)	**164**	**167.3**	**170**	**174.2**	**178**	**180**	**186**

ii) Bivariate Analysis (Two Variable Relationships):

Bivariate data is a data set that entails two different variables, with the analysis of this type centering around the relationship or association between the two variables. An example of bivariate data is summer season temperature and ice cream sales. Supposing temperature and ice cream sales are the two variables of bivariate data (as shown in Table 2). Here, the relationship can be clearly seen from the table that temperature and sales are directly related and therefore interconnected, for as the temperature increases, sales also rise [5].

Table 2: Bivariate Data

Temperature	**Ice Cream Sales**
20	2000
25	2500
35	5000

iii) Multivariate Analysis (Multiple Variables):

In general terms, multivariate data could be described as datasets in which each observational or sample point consists of multiple variables or features. These variables may further represent different aspects, characteristics, or measurements related to the observed phenomenon. In general, three or more variables meet this condition under the broad definition of multivariate (as shown in Table 3).

For instance, an advertiser wants to compare the popularity of four ads on the website.

Table 3: Multivariate Data

Advertisement	**Gender**	**Click rate**
Ad1	Male	85
Ad3	Female	65
Ad2	Female	128
Ad1	Male	68
Ad3	Male	25

The click rate can be assessed for men and women, and relationships among variables can then be examined. It resembles bivariate except it has more than one dependent variable.

Difference between Univariate, Bivariate and Multivariate data:

Table 4: Difference between Univariate, Bivariate and Multivariate data

Univariate	Bivariate	Multivariate
It only summarizes a single variable at a time.	It only summarizes two variables	It only summarizes more than 2 variables.
It does not deal with causes and relationships.	It does deal with causes and relationships and analysis is done.	It does not deal with causes and relationships and analysis is done.
It does not contain any dependent variable.	It does contain only one dependent variable.	It is similar to bivariate, but it contains more than 2 variables.
The main purpose is to describe.	The main purpose is to explain.	The main purpose is to study the relationship among them.
An example of a univariate can be height.	An example of bivariate can be temperature and ice sales in summer vacation.	For example, suppose an advertiser wants to compare the popularity of four advertisements on a website. Then their click rates could be measured for both men and women and relationships between variables can be examined

Key Techniques for Exploring Data Relationships:

Exploratory analysis comprises methods of statistics and visualization that help to associate the other basic information patterns to investigate the relationship, dependence, and trends between variables. The key techniques are described in detail below:

Statistical Methods for Exploring Data Relationships:

1. **Correlation Analysis:** Correlation gauges the strength and direction of the relationship between two numerical variables. It indicates whether an increase in one variable is associated with an increase or a decrease in another [6].

   Types of Correlation:

a) **Positive Correlation (r > 0):** Both variables increase together (i.e., more time spent studying leads to higher exam scores).

b) **Negative Correlation (r < 0):** As one variable increases, the other variable decreases (e.g., more exercise results in a lower percentage of body fat).

c) **No Correlation (r ≈ 0):** No readily apparent relationship between the variables.

Common Correlation Coefficients:

a) **Pearson Correlation (r):** Measures linear relationships between two continuous variables.

Formula:

$$\mathbf{r} = \frac{\sum(\mathbf{X_i} - \overline{\mathbf{X}})(\mathbf{Y_i} - \overline{\mathbf{Y}})}{\sqrt{\sum}(\mathbf{X_i} - \overline{\mathbf{X}})^2 \ \sqrt{\sum}(\mathbf{Y_i} - \overline{\mathbf{Y}})^2}$$

Example: Height vs. Weight correlation.

b) **Spearman Rank Correlation (ρ):** Measures monotonic relationships (not necessarily linear). Suitable for ordinal data. Example: Ranking students based on exam scores and time spent studying [6].

Applications:

- It is used in financial analysis (e.g., stock price correlations).
- It helps in feature selection for machine learning.

2. **Covariance Analysis:** Covariance is a measure of how variables vary together. Covariance is not standardized: it treats variables on different scales, which means that its value can be easily affected by changes in scale.

- **Positive Covariance:** Variables move in the same direction.
- **Negative Covariance:** Variables move in opposite directions.
- **Zero Covariance:** No relationship.
- **Formula:**

$$\mathbf{cov(X, Y)} = \frac{\sum(\mathbf{X_i} - \overline{\mathbf{X}})(\mathbf{Y_i} - \overline{\mathbf{Y}})}{\mathbf{n}}$$

Where:

- Xi, Yi = Data points
- $\bar{X}$, $\bar{Y}$= Means of X and Y
- n = Number of observations

Example:

Covariance between revenue and advertising spend.

3. **Regression Analysis:** It establishes the relationship in predictive and inferential analyses between potentially dependent variables and independent variables.

Types of Regression:

Linear Regression: Linear regression constitutes a statistical method used to model the relationship between a dependent variable and one or more independent variables. It provides insights that are valuable from a prediction and data analysis standpoint.

Formula:

$$\mathbf{Y = \beta 0 + \beta 1 X + \epsilon}$$

Example: Predicting house prices based on square footage.

4. **Logistic Regression:** Logistic regression is a supervised machine learning algorithm used for classification tasks where the goal is to predict the probability that an instance belongs to a given class or not. Using the logistic regression model for binary classification and non-linear transformations of continuous variables is often done using a sigmoid function, which takes independent variables as inputs and gives outputs ranging from 0 to 1 as probabilities.

For example, we have two classes, Class 0 and Class 1 If the value of the logistic function for an input is greater than 0.5 (threshold value), then it belongs to Class 1; otherwise, it belongs to Class 0. It's referred to as regression because it is the extension of linear regression, but is mainly used for classification problems [7].

1.2.6 Visualization Methods for Exploring Data Relationships:

Scatter Plots: A scatter plot is a graph used in statistics and mathematics to plot data. The scatter plot, or scatter diagram or scatter chart, uses dots to represent the values of two different numerical variables. The position of each dot on the horizontal and vertical axes represents values for an individual data point (as shown in fig. 1.8).

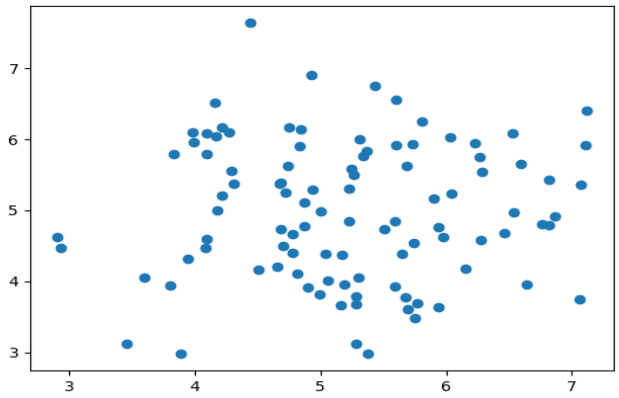


Fig. 1.8: Scatter Plot visualization

It is generally used to display the relationship between one independent variable and one dependent variable, with the independent variable plotted on the x-axis and the dependent variable on the y-axis, facilitating visualization of how the independent variable affects the dependent variable. These plots are known as Scatter Plot Graph or Scatter Diagram. Example: Plotting height vs. weight, Checking the relationship between advertising spend and sales revenue [8].

Applications of scatter plot: Some of the common uses of Scatter Plots are discussed below.

- **Correlation Analysis:** A scatter plot can be useful in the correlation analysis of two different variables. That means one can know if one variable correlate positively, negatively, or does not correlate with another variable.
- **Outlier Detection:** An outlier is a data point that is away from the other data. A scatter plot is a good way to bring these outliers out.
- **Cluster Identification:** In some instances, a scatter plot will show potential clusters or groupings of data points.

Heatmaps: Heatmaps are a very useful technique for the graphical representation of numerical data, where values are represented in colors. This helps recognize patterns, trends, and anomalies in large datasets. The most conventional color schemes are generally ones that reversibly span from the warm colors red to the cold colors blue where warm ones signify higher values more often than do the cool colors which denote lower values (as shown in fig. 1.9) Such visualizations make complex data set interpretation digestible with speed and in an intuitive way [8].

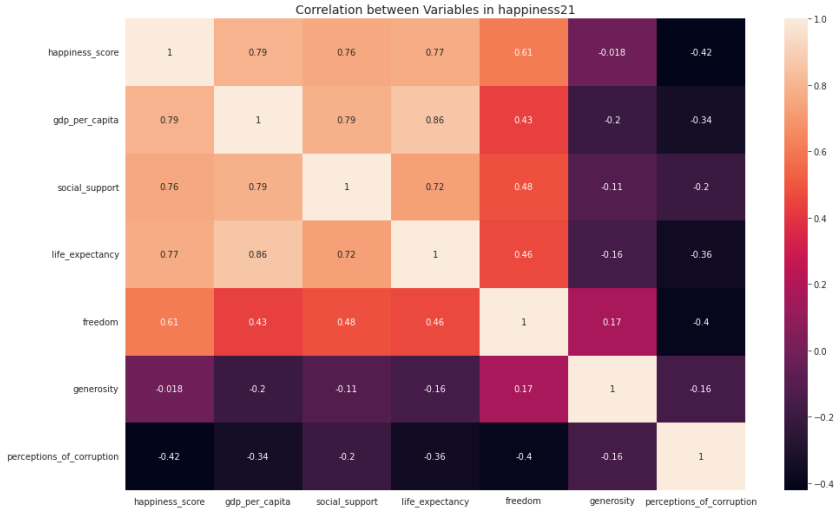


Fig. 1.9: Heatmaps Visualization

Applications of heatmaps: Heat maps are versatile and can be used in different scenarios:

i. Website optimization to study user behaviours and optimize the design.
ii. Financial analysis to visualize performance metrics and point out growing areas in need of improvement.
iii. Marketing to observe campaign performance and customer engagement.
iv. Scientific inquiry to analyze genetic data and other complex datasets.
v. Geographical analysis through graphical representations of spatial data, including population density, crime rate, or welfare patterns.
vi. Sports analytics to physically analyze players' movement, game strategy, or performance metrics.

Line graph: A line graph juxtaposes two variables from a visual perspective, demonstrated by the x-axis and y-axis. It illustrates the information by joining all the coordinates on a grid using a continuous line. There are three different types of line graphs [9]. They include:

i. The simple line graph- one single line is presented on the graph.
ii. The multiple line graph- more than one line on the same set of axes. It is used for the effective comparison of like items over the same period.
iii. The compound line graph-if the data can be divided into two or more types. This use of the term is to refer to a line graph indicated by a separate additional line further topmost. A compound line graph has lines drawn from the total to show the component part of the total. The top line represents the total and below it is a line representing part of the total. The distance present between two lines represents the size of each part (as shown in fig. 1.10).

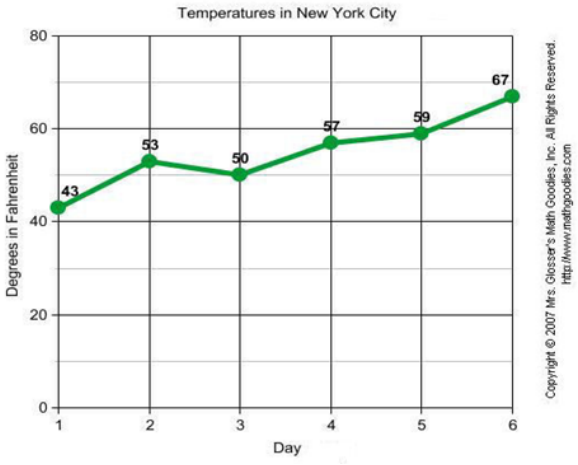


Fig. 1.10: Line Graph representation

Bar Graph: A bar graph shows the comparisons among categories. It can consist of two or more parallel vertical (or horizontal) bars (rectangles). A bar chart will serve to compare two or more values against a small set of results. There are two types of Bar Graphs [9]:

a) A vertical Bar Graph
b) A horizontal Bar Graph

Depending on the choice of the baseline, the vertical bar is based on the vertical axis while the horizontal bar is set on the horizontal axis. A bar graph showing the comparison of a student's marks successfully indicates the desired details (as shown in fig. 1.11).

It is possible, using bar graphs, to compare students' scores in different subjects as well as represent the scores of one pupils in each subject. It is also very easy to produce a bar graph for every student concerning all subjects.

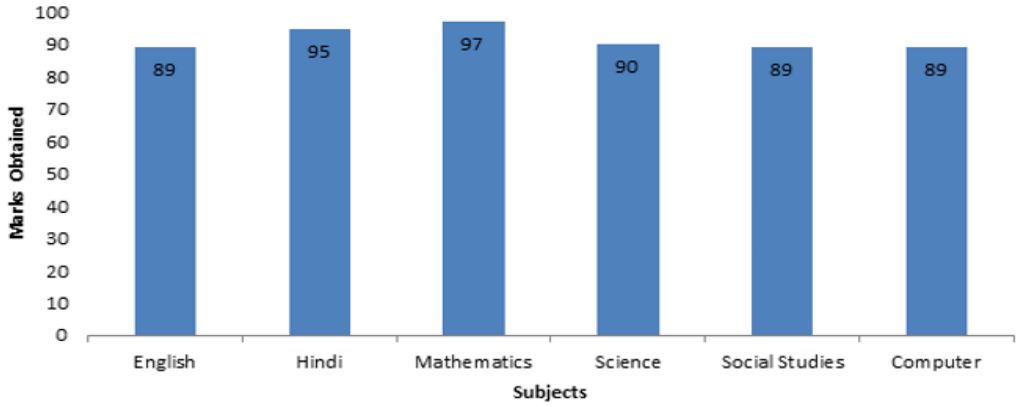


Fig. 1.11: Bar Graph representation

Advanced Techniques for Data Relationships:

a) **Principal Component Analysis (PCA):** Principal component analysis is a dimensionality reduction method widely used for dimensionality reduction in huge data dimensions. It is the process of transforming many variables into a few whiles still retaining most of the information in the large set.

Reducing the number of variables from a data set usually loses some accuracy, and therein lies the art of dimensionality reduction: sacrificing a little accuracy for simplicity (as shown in fig. 1.12). Smaller, easy-to-explore data sets lend themselves to easier and much faster visualization and, consequently, analysis of data points for machine learning algorithms that needn't process irrelevant variables [10].

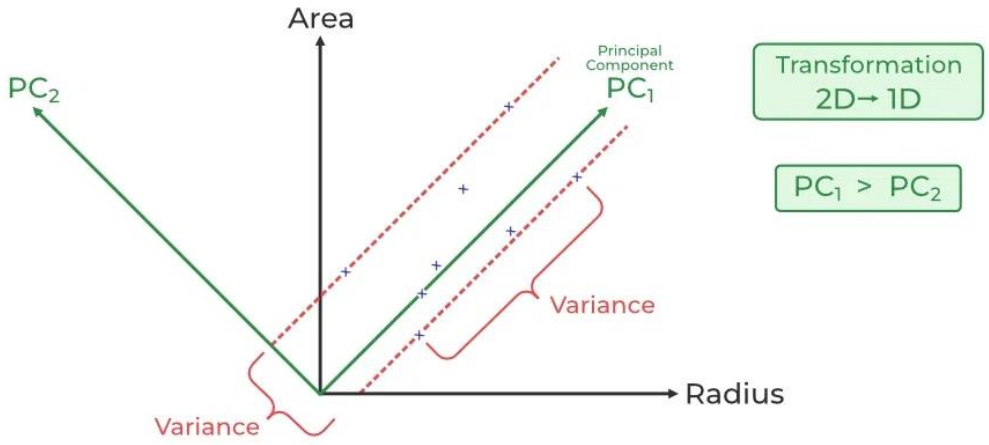


Fig. 1.12: Principal Component Analysis Diagram

b) Clustering (K-Means, Hierarchical Clustering): K-means is a method of cluster analysis that utilizes a previously decided number of clusters. It requires advanced knowledge of a specified number of clusters. K-means clustering considers partitioning of n observations into k clusters such that each observation belongs to the cluster with the nearest mean as the prototype of that cluster (as shown in fig. 1.13).

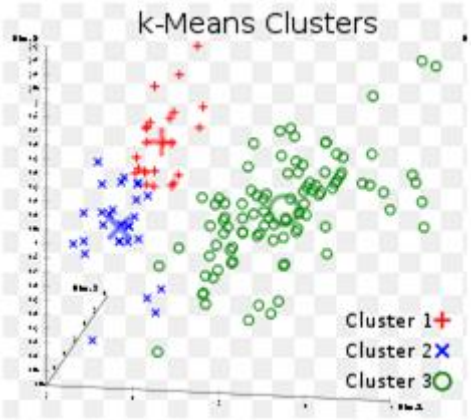


Fig. 1.13: K-means clustering visualization

The k-means clustering algorithm attempts to partition an anonymous dataset into a fixed-positive number (k) of clusters, specifying that a set of information-containing identities contained within the classes will be used. The k-means clusters typically select the Centroid initializations of the apex number of k. A Centroid is a point in a data cluster that serves as a centre.

- **Hierarchical clustering,** also known as hierarchical cluster analysis (HCA), is another method of cluster analysis that builds up a hierarchy of clusters without having a fixed number of clusters (as shown in fig. 1.14).

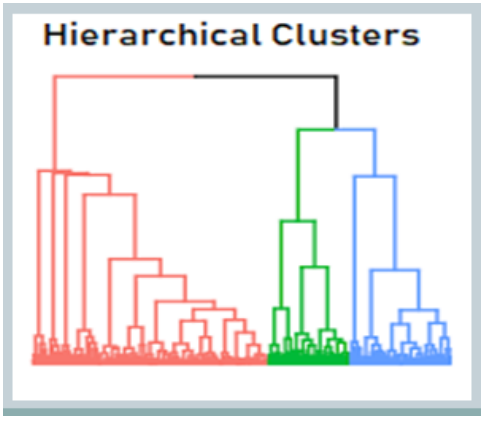


Fig. 1.14: Hierarchical clustering visualization

1.2.7 Preparing the data for visualization: Data preparation is the most important stage of the data analysis process. Proper lexical and structural organization and cleaning of the data guarantee that the corresponding visualizations are accurate, meaningful, and interpretable. This accordingly involves the cleansing process, the transposition of the specific data, and further formatting tailored to prioritize some of the more prominent insights.

a. Steps in Preparing Data for Visualization

1.Data Cleaning: Before visualizing data, it must be cleaned to remove errors, inconsistencies, and missing values.

a) **Handling Missing Data:** Remove missing values if they are insignificant and impute missing values using mean, median, or mode.

b) **Removing Duplicates:** Ensure each data entry is unique to avoid skewed visualizations.

c) **Correcting Data Errors:** Standardize inconsistent data formats (e.g., dates, currency, measurement units) and Fix spelling or categorization errors (e.g., "USA" vs "United States").

2. Data Transformation: Transforming data helps make it suitable for visual representation.

Key Techniques:

a) **Scaling and Normalization:** Rescale numerical values to a uniform range (e.g., 0 to 1) to avoid skewed charts. **Example:** Standardizing income values in thousands instead of full dollar amounts.

b) **Aggregation:** Summarize large datasets to make visualizations more readable. **Example:** Showing monthly sales instead of daily sales for better trend analysis.

c) **Binning (Grouping Data into Ranges):** Converts continuous data into categorical ranges. **Example:** Grouping ages into "18-25", "26-35", "36-45", etc., for bar charts.

d) **Feature Engineering:** Creating new relevant variables to enhance visualization. **Example:** Converting timestamps into "weekday vs. weekend" to analyse traffic patterns.

3. Formatting Data for Visualization Tools: Different visualization tools require data to be structured in a specific way.

a) Tidy Data Format

- Each row represents a single observation.
- Each column represents a variable.
- Each cell contains a single value.

Example:

Date	Product	Sales	Category
2024-02-01	A	500	Electronics
2024-02-01	B	300	Clothing

b) Pivoting & Melting Data:

- **Pivoting:** Converts long-format data into a wide format for comparison.
- **Melting:** Converts wide-format data into a long format for easier plotting.

4. Handling Categorical and Text Data: It Converts categorical values into numerical labels if needed. It Uses one-hot encoding for machine-learning-friendly visualizations. Shorten long text labels for better readability in plots.

5. Filtering and Sampling Data: It Uses filtering to focus on relevant subsets (e.g., the last 12 months of sales data). It Apply sampling if dealing with large datasets to speed up visualization.

6. Ensuring Data Consistency and Accuracy:

- Check for outliers that might distort charts.
- Verify that all values are in the expected range.
- Cross-check against sources to avoid misrepresentation.

b. Tools for Data Preparation:

- Excel & Google Sheets: Basic data cleaning and formatting.
- Python (Pandas, NumPy): Advanced data processing and transformation.
- SQL: Aggregating and structuring large datasets.
- Tableau, Power BI: Built-in data preparation tools for visualization.

After getting the data ready for a visual display, the next step is to get the visual display going by selecting an appropriate type of chart or graph and having it laid out to provide clarity and a true representation of the data. Above all else, the type of visual representation depends on the data type and

insights to be presented: charts for comparisons, line charts for trends, scatter plots for correlations, and maps for geographic data. The elements of good design are clear labeling, uniform colors, and avoidance of clutter to aid legibility. Interactivity as filters and tooltips, is a necessary engagement with a user in tools like Tableau and Power BI. The completed visuals are then validated against an established accuracy and effectiveness test to ensure that they convey the insight. To support data-driven decision-making, the visual will be incorporated into reports, dashboards, or presentations [10].

1.3 Seven Stages of Data Visualization

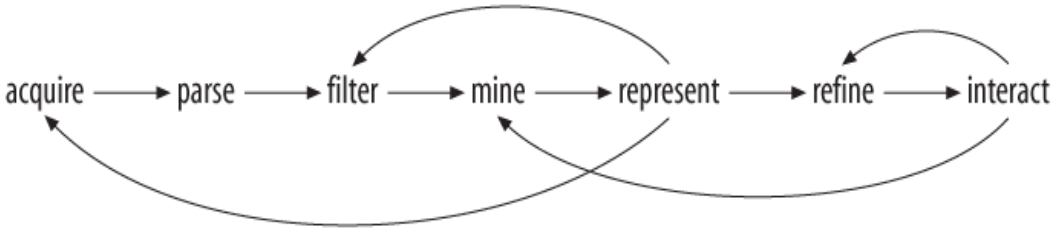


Fig. 1.15: Seven stages of data visualization

a) Acquire: The first stage in data visualization is the cornerstone of the entire process (as shown in fig 1.16). It involves retrieving raw data from various sources and evaluating it to make sure that it is good enough in both respect and quantity for the analysis. Without these datasets, even the best visualization techniques would fail to deliver meaningful results. It is the process of gathering raw data from diverse sources so that they may be visualized. An accurate and meaningful visualization can be created only from high-quality, relevant, and well-structured data [11].

Fig. 1.16: Acquiring data from various sources.

Objectives for Data Acquisition:

- **Identify the Right Data Source:** Make sure that the data source is both credible and pertinent.
- **Extract Data Efficiently**: Involve various tools and methods to collect the data.
- **Complete Data Assembly:** Draw the most critical variables about the analysis process.
- **Validate Data Accuracy:** Ensure missing values, errors, and inconsistencies are checked and corrected as early as possible.

Types of Data for Data Collection: Data can be classified into different types based on its structure and nature (as shown in fig 1.17):

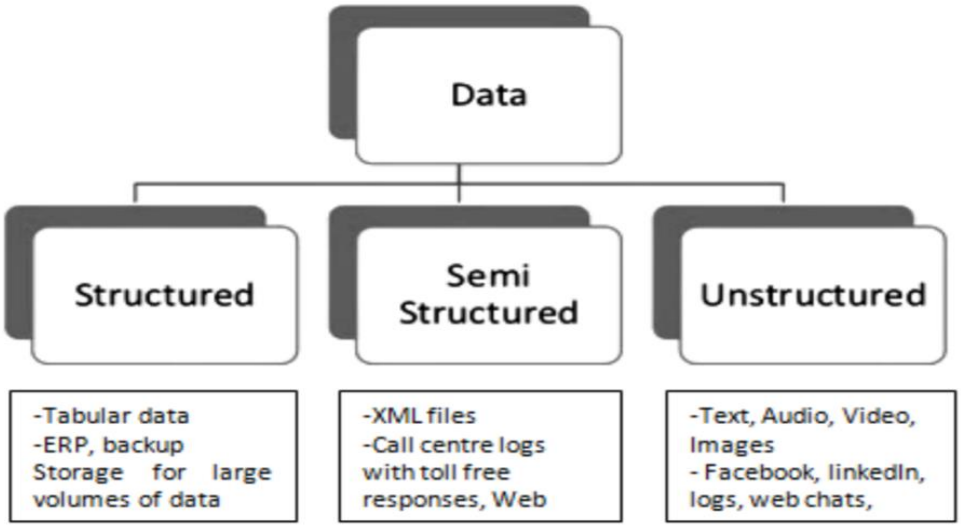


Fig. 1.17: Types of data

i. Structured Data:

Structured data refers to data that is highly organized and formatted to be stored in a predetermined format, usually in relational databases, spreadsheets, or data warehouses. It is structured according to a specific model of schema rows and columns, which facilitates data entry, search, retrieval, and analysis.

Features of Structured Data:

- **Predefined Format** – Arranged within tables of fixed columns and data types (for example, text, number, and date).
- **Easily Searchable –** Can be queried using SQL.
- **Highly Structured –** Stored in a relational database like MySQL, PostgreSQL, and Microsoft SQL Server.
- **Efficient Processing –** Fast retrieval is facilitated by access paths.
- **Scalable –** Can maintain structure while increasing by adding new rows or new columns.

ii. Semi-Structured Data:

Semi-structured data is data without a strict set of tabular formats, like structured data, yet still possesses some organizational properties via tags, markers, or metadata. This type of data lies between structured and unstructured data, hence has elements of both.

In contrast to structured data stored in relational databases, semi-structured data is usually stored in formats like JSON, XML, or NoSQL databases, depending on its flexibility concerning arrangement.

Features of Semi-Structured Data:

- **Partially Organized –** This term is often used interchangeably with loosely or partially organized data. This type of data usually contains some structure (for instance, tags, keys, or metadata), even if it does not fit neatly into tables.
- **Flexible schema:** This is a data structure that varies from record to record, unlike those that follow a rigid schema.
- **Easier to Process than Unstructured Data:** it requires slightly more effort than structured data. Still, it possesses certain identifiable features allowing it to be parsed and queried**.**

- **Uses Hierarchical or Graph-Based Storage:** structuring types, these could often be either JSON, XML, or NoSQL in nature.
- **Requires special querying methods:** one cannot query it in the conventional SQL mode; rather operations can be performed using tools like XPath, XQuery, or NoSQL query languages.

iii. Unstructured Data:

Unstructured data refers to information that has no defined model, format, or organization. Unstructured data is raw information and thus has to use advanced techniques for analysis including machine learning, NLP, and AI. In contrast to structured data, which typically sits in relational databases, or semi-structured data, like those bearing some organization (for instance, XML or JSON), analysis of unstructured data takes a more advanced form.

Features of Unstructured Data:

- **No Fixed Format:** This means lacking a predefined set of rules, thus making it impossible to store in traditional databases.
- **Diverse and Complex-** Information could be in the form of text, pictures, videos, or audio.
- **High volume and Growth-** Accounts over 80% of the world's store of data and is in constant growth.
- **Difficult to Query and Analyze-** Need sophisticated tools such as artificial intelligence, natural language processing, and deep learning to retrieve data that makes sense.
- **Dense with Information-** Such documents contain important indications concealed within the raw text, speech, or multimedia content.

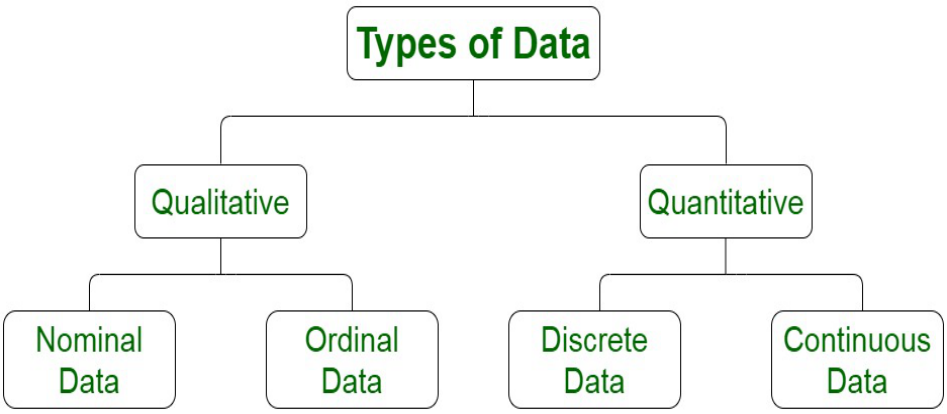


Fig. 1.18: Types of Data representation

iv. Qualitative Data:

Qualitative data, i.e., categorical data, are non-numerical information that represents the characteristics, labels, or descriptions. Such information forms the data categorization or classification basis but cannot be measured or expressed numerically meaningfully (as shown in fig 1.18). Unlike quantitative data, which deals with numbers and measurements, qualitative data references attribute patterns or classifications in a non-numerical fashion.

For example:

- A survey collecting feedback on customers' experiences (e.g., "Very Satisfied," "Neutral," "Dissatisfied").
- A dataset recording hair colors (e.g., "Black," "Brown," "Blonde").
- A company's job roles of employees (e.g., "Manager," "Engineer," "Salesperson").

Characteristics of Qualitative Data:

i. **Descriptive:** It refers to any information describing the qualities or characteristics and not measured in numerals.

ii. **Categorical:** Categorical value divides information or observations into pre-chosen groups or categories.

iii. **Non-numerical:** Non-numeric value means that no arithmetic operations, like addition or subtraction, can be performed on it.

iv. **Used for Classification:** The observations are grouped in meaningful or categorical groups.

v. **Can be Subjective:** Some qualitative data can be rather subjective; expressed as opinions or reviews, these may vary relative to any subjective interior interpretation.

Types of Qualitative Data:

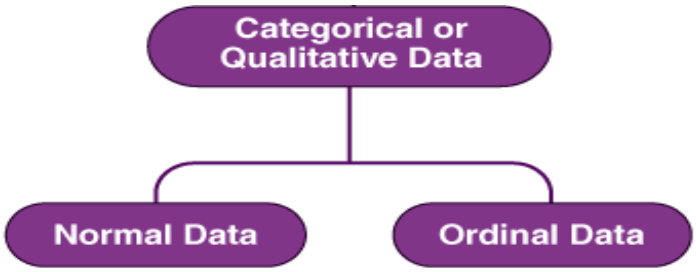


Fig. 1.19: Types of Qualitative data

a. Nominal Data:

Nominal values refer to available discrete units of variables or categories. You can think of them as simply "labels." This type of data is unordered. Meaning, it doesn't matter if you rearrange the order of its values; the meaning won't change (as shown in fig 1.20).

Example:

- **Colors:** Red, Blue, Green, Yellow
- **Countries:** USA, Canada, Germany, Australia
- **Car Brands:** Toyota, Ford, BMW, Honda
- **Blood Types:** A, B, AB, O

Fig. 1.20: Nominal Data Visualization

b. Ordinal Data:

Ordinal data is a type of qualitative (categorical) data that consists of categories ordered in a meaningful way, but it does not guarantee that the interval between these categories is constant or can be measured. This means ordinal data allows for comparison to the extent that one category can be said to be higher or lower than another, but the exact distance between the categories cannot be said to exist (as shown in fig 1.21). Example:

a) **Customer Satisfaction Ratings:** A survey asks customers to rate their satisfaction with a product:
   - Very Dissatisfied (1)
   - Dissatisfied (2)
   - Neutral (3)
   - Satisfied (4)
   - Very Satisfied (5)

   The responses have a clear ranking, but the difference between "Neutral" and "Satisfied" may not be equal to the difference between "Dissatisfied" and "Neutral".

b) **Education Levels:** Levels of education ranked in order:
   - High School
   - Bachelor's Degree
   - Master's Degree
   - PhD

   A PhD is ranked higher than a Master's, but the difference in knowledge or skill between each level is not exactly measurable.

c) **Movie Ratings:** When rating movies, people often use stars:
   ★ (1-star)
   ★★ (2-stars)
   ★★★ (3-stars)
   ★★★★ (4-stars)
   ★★★★★ (5-stars).

   A 5-star movie is better than a 3-star movie, but the difference between 2-star and 3-star might not be the same as between 4-star and 5-star.

d) **Economic Class**
   - Low Income
   - Middle Income
   - High Income

Fig. 1.21-1.24: Ordinal Data Example

v. Quantitative Data:

Quantitative data is nothing but numerical data that can be precisely measured, counted, and expressed in numbers. It shows quantities, which makes it possible to perform mathematical operations, including addition, subtraction, averaging, and the like.

Quantitative data answers questions like:

- How much?
- How many?
- How long?
- What is the numerical value of...?

For example:

- **Student test scores:** 85, 90, 78, 92
- **Company revenue:** $50,000, $75,000, $100,000
- **Daily temperature:** 25°C, 30°C, 28°C

Characteristics of Quantitative Data:

a) Numerical Representation: A way of showing things in numbers instead of words.

b) Quantifiable: Quantifiable in measurements like height, weight, income, and temperature.

c) Mathematical Operations: Subject to Arithmetic Operations like addition and subtraction or mean, etc.

d) Stats analysis: Mean, median, standard deviation, correlation, etc.

Types of Quantitative Data

Quantitative data is categorized into **two main types** (as shown in fig 1.25).:

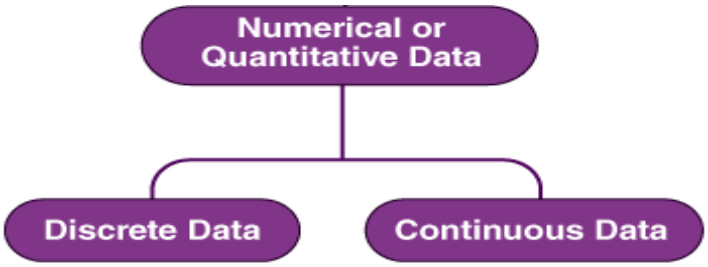


Fig. 1.25: Types of Quantitative Data

a. Discrete Data (Countable numbers):

Discrete data refers to a type of quantitative (numerical) data, capable of adopting countable, finite values. Values cannot be split into smaller parts; hence, discrete data is made up only of whole numbers (integers). Discrete data consists of fixed and countable values (as shown in fig 1.26). It doesn't contain decimal or fractional parts of values.

Example:

- Number of students in a classroom → 20, 25, 30 (but never 20.5 students).
- Number of cars in a parking lot → 10, 50, 100 (but never 10.3 cars).
- Number of books on a shelf → 5, 10, 15 (but never 7.8 books).

Representation of Discrete Data:

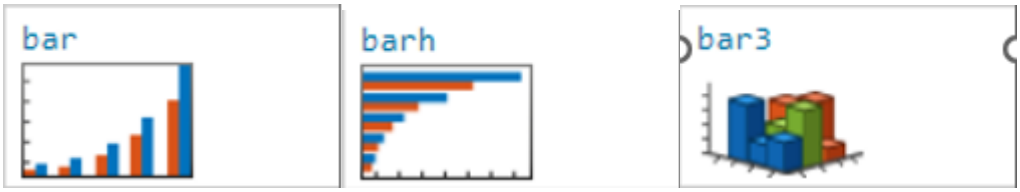


Fig. 1.26: Discrete Data representation in graph

b. Continuous Data (Measurable numbers):

Continuous data is a type of quantitative (numerical) data that can take any value within an interval. It is measurable rather than countable, so it could be represented in decimal or fraction form. Continuous data can be divided into smaller values indefinitely, and there are no fixed gaps between any two values (as shown in fig 1.27).

Example:

- **Height of a person** → 165.4 cm, 170.2 cm, 175.9 cm
- **Temperature** → 36.5°C, 37.8°C, 39.1°C
- **Speed of a car** → 55.6 km/h, 60.3 km/h, 72.8 km/h
- **Weight of an object** → 2.5 kg, 3.75 kg, 4.6 kg

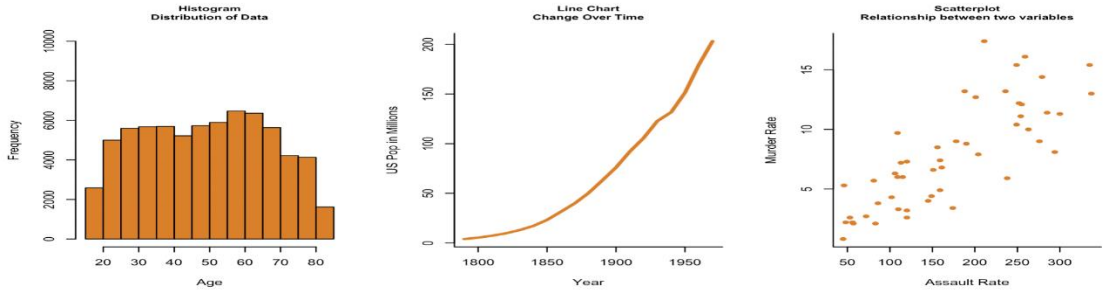


Fig. 1.27: Continuous Data representation in graph

Data Collection Methods:

Data for visualization can be collected from various sources depending on the objectives. It is mainly of two types (as shown in fig 1.28).:

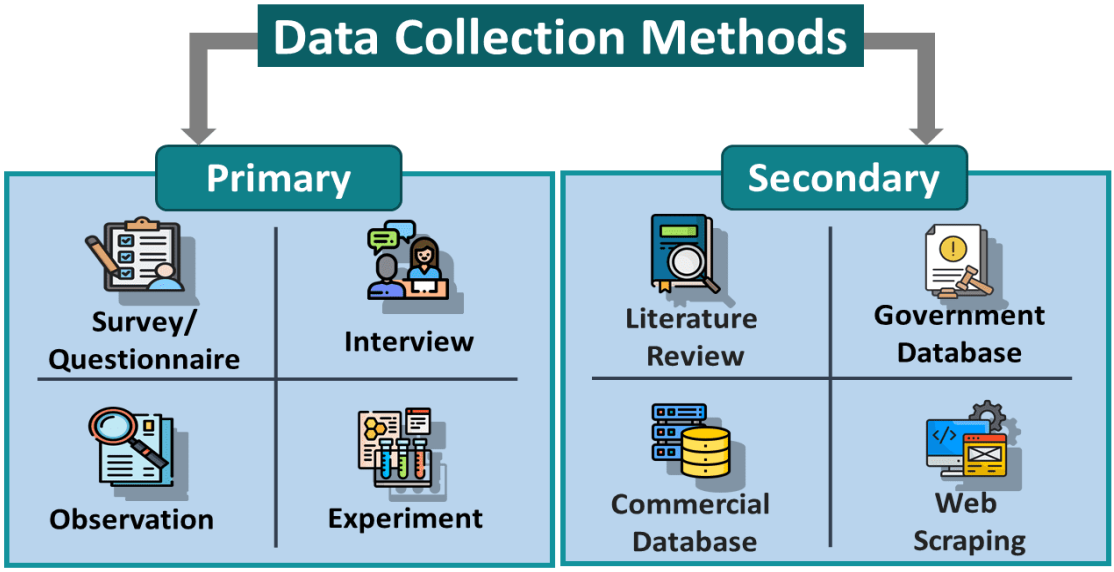


Fig. 1.28: Data collection methods

a) **Primary Source:**

This refers to a handful of originally collected data from the source. It provides the researcher with fresh or raw numerical information connected with the statistical study. To put it, primary sources of data give the researcher direct access to this topic of investigation, such as statistical data, art pieces, and the transcripts of interviews.

Manual Data Collection:

Manual data collection refers to collecting data by hand or without mechanical tools or software. This involves any human endeavour that is utilized by a person to record, enter, validate, or organize information extracted from sources ranging from surveys to observations, interviews, and written records. Manual data collection has high levels of human error and a lot of wasted time, and it is applied only when it is impossible to use automated procedures or when precision calls for human scrutiny.

Methods of Manual Data Collection:

1. Surveys & Questionnaires

- Data is collected by asking individuals specific questions through paper-based or digital forms.
- Examples: Customer satisfaction surveys, and employee feedback forms.

2. Observations

- Researchers manually record data by observing people, events, or processes.
- Examples: Tracking customer behaviour in a store, monitoring weather conditions.

3. Interviews & Focus Groups

- Information is collected by conducting in-person or virtual conversations.
- Examples: Job interviews, and market research discussions.

4. Data Entry from Documents

- Manually transcribing data from physical or digital sources into spreadsheets or databases.
- Examples: Entering patient records in hospitals, and compiling sales reports.

5. Direct Measurements

- Individuals record data from physical instruments or experiments.
- Examples: Taking temperature readings and manually counting foot traffic in a store.

b) Secondary Source:

The information presented in secondary sources is mostly a reprocessing of data obtained from primary sources by either institutions or agencies having already obtained them from primary research. This essentially means that the researcher does not have first-hand quantitative and raw information related to their study. Thus, it is only through secondary sources of data that one may interpret, describe, or synthesize information from primary sources. Examples include reviews, government websites that publish surveys/data, academic books, published journals, and articles.

While primary sources lend credibility that is greatly based on evidence, well-rounded research will require data collection from both primary and secondary sources.

Automated Data Collection:

Automated data collection describes the method of gathering and recording data by some software, scripts, sensors, or algorithms, without the intervention of human beings. Automated data collection is faster and more efficient and involves far fewer instances of human error than manual data collection. In industries like finance, healthcare, marketing, and IoT (Internet of Things), automated methods find their frequent application.

Methods of Automated Data Collection

a) **Web Scraping:** Extracting data from websites using automated tools or scripts.
   Examples:
   - Gathering product prices from e-commerce websites.
   - Extracting news headlines for sentiment analysis.
   - Tools Used: BeautifulSoup, Scrapy, Selenium.

b) **APIs (Application Programming Interfaces):** Fetching data from online services in real time.
   Examples:
   - Google Analytics API for website traffic insights.
   - Twitter API for collecting social media trends.
   - Tools Used: REST APIs, GraphQL, Postman.

c) **IoT & Sensor Data Collection:** Devices automatically capture and transmit data.
   Examples:
   - Smart home sensors measure temperature and humidity.
   - GPS trackers recording vehicle movements.
   - Tools Used: Arduino, Raspberry Pi, AWS IoT.

d) **Database & Log File Extraction:** Collecting structured data from databases without manual input.
   Examples:
   - Automatic data retrieval from SQL databases.
   - Extracting user activity logs from servers.
   - Tools Used: SQL queries, NoSQL databases, Logstash.

e) **Optical Character Recognition (OCR):** Converting printed or handwritten text into digital format.
   Examples:
   - Scanning and digitizing invoices or receipts.
   - Converting paper-based medical records into digital files.
   - Tools Used**:** Tesseract OCR, Google Vision API.

f) **AI & Machine Learning-Based Data Collection:** Using AI to analyze and collect unstructured data.
   Examples:
   - AI chatbots collecting customer feedback.
   - Machine learning models extracting insights from images or videos.
   - Tools Used: TensorFlow, OpenAI APIs, Natural Language Processing (NLP).

Example of Acquiring data:

A copy of the zip code listing can be found on the U.S. Census Bureau website, as it is frequently used for geographic coding of statistical data. The listing is a freely available file with approximately 42,000 lines, one for each of the codes, a tiny portion of which is shown in Figure 1.29.

00210	+43.005895	-071.013202	U	PORTSMOUTH	33	015
00211	+43.005895	-071.013202	U	PORTSMOUTH	33	015
00212	+43.005895	-071.013202	U	PORTSMOUTH	33	015
00213	+43.005895	-071.013202	U	PORTSMOUTH	33	015
00214	+43.005895	-071.013202	U	PORTSMOUTH	33	015
00215	+43.005895	-071.013202	U	PORTSMOUTH	33	015
00501	+40.922326	-072.637078	U	HOLTSVILLE	36	103
00544	+40.922326	-072.637078	U	HOLTSVILLE	36	103
00601	+18.165273	-066.722583		ADJUNTAS	72	001
00602	+18.393103	-067.180953		AGUADA	72	003
00603	+18.455913	-067.145780		AGUADILLA	72	005
00604	+18.493520	-067.135883		AGUADILLA	72	005
00605	+18.465162	-067.141486	P	AGUADILLA	72	005
00606	+18.172947	-066.944111		MARICAO	72	093
00610	+18.288685	-067.139696		ANASCO	72	011
00611	+18.279531	-066.802170	P	ANGELES	72	141
00612	+18.450674	-066.698262		ARECIBO	72	013
00613	+18.458093	-066.732732	P	ARECIBO	72	013
00614	+18.429675	-066.674506	P	ARECIBO	72	013
00616	+18.444792	-066.640678		BAJADERO	72	013

Fig. 1.29: Zip codes in the format provided by the U.S. Census Bureau.

b) Parse: Once you have the data, it must be parsed and transformed into a form that assigns each data component a purpose. The file must be split on each line into its components; in this case, it must be delimited at every tab character. Then, each component of the data must be transformed into a useful format. The composition of each line of the listing in the census is illustrated by Figure 1.30 and needs to be comprehended before we can deparse it and extract from it what we require [11].

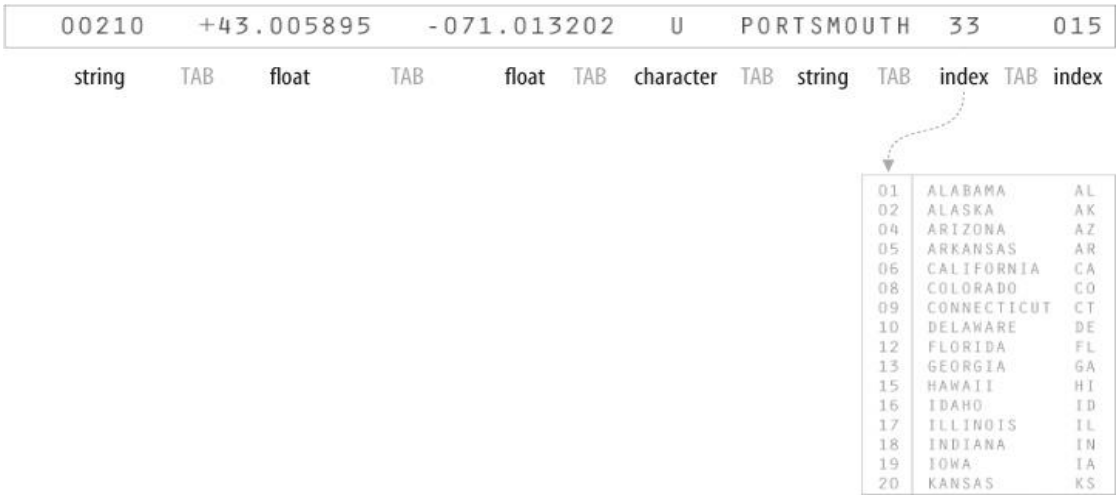


Fig. 1.30: Structure of acquired data.

Objectives of Parsing Data:

- Transform raw data into a structured format (e.g., JSON, CSV, SQL tables).
- Identify and label key variables for easier understanding.
- Standardize data types and formats (dates, numbers, text).
- Handle missing or inconsistent data to prevent errors in analysis.

Data parsing is said to occur when raw data is prepared in a manner that gives it structured form and meaning to analyze and visualize it. The process ensures the consistency, right format, and capability of such data to yield credible insights.

The operations performed in the field of data transformation include data cleaning, data filtering, data aggregating, data restructuring, and data normalization.

Steps in Data parsing:

1. Data Cleaning

- **Purpose:** Remove inconsistencies, duplicates, and errors.
- **Methods:**
    - Handle missing values (fill, interpolate, or remove).
    - Standardize data formats (date, currency, text).
    - Remove duplicate entries.

2. Data Integration

- **Purpose:** Combine multiple data sources into a unified dataset.
- **Methods:**
    - Merge databases (SQL JOIN, Pandas merge()).
    - Consolidate multiple Excel sheets or CSV files.
    - Integrate data from APIs or external sources.

3. Data Aggregation

- **Purpose:** Summarize detailed data for easier analysis.
- **Methods:**
    - Calculate totals, averages, or counts.
    - Group data by time periods (e.g., daily → monthly sales).
    - Aggregate data at different levels (e.g., city → country).

4. Data Normalization & Scaling

- **Purpose:** Standardize values across different scales for accurate visualization.
- **Methods:**
    - **Min-Max Scaling:** Normalize values between 0 and 1.
    - **Z-Score Normalization:** Convert values to standard deviations from the mean.
    - **Log Transformation:** Reduce the impact of large numerical differences.

5. Data Encoding & Formatting

- **Purpose:** Convert categorical data into numerical formats for analysis.
- **Methods:**
    - **One-Hot Encoding:** Convert categorical values into binary columns.
    - **Label Encoding:** Assign numerical values to categories (e.g., Male = 0, Female = 1).
    - **Reformat Dates & Times:** Convert time zones and standardize formats (YYYY-MM-DD).

6. Data Filtering & Selection

- **Purpose:** Remove unnecessary data to focus on relevant insights.
- **Methods:**
    - Remove outliers that skew visualizations.
    - Filter data based on conditions (e.g., exclude users with incomplete profiles).
    - Select key columns needed for visualization.

Tools for Data parsing:

Table 5: Tools used for data parsing

Tool	Usage
Python (Pandas, NumPy, SciPy)	Cleaning, filtering, aggregating, and transforming datasets.
SQL (Structured Query Language)	Data extraction, merging, and aggregation.
Excel & Google Sheets	Basic transformations like filtering, pivot tables, and formula-based modifications.
Tableau Prep	Preparing and transforming data before visualization in Tableau.
Power BI Data Transformations	Using Power Query to clean and reshape data.

c) Filter: It is the process of identifying, correcting and removing errors, inconsistencies and inaccuracies within raw data for visualization. Clean data provides clarity, dependability and significance to the insights derived when creating visual representations like charts, graphs and dashboards.

The need for data parsing arises in a data visualization pipeline due to the integrated existence of repeats, missing values or wrong information in real-time data, on account of which it is a crucial step for very good analysis [11].

Steps in Data Filtering for Data Visualization:

1. Removing Duplicate Data:

- **Issue:** Duplicate records can distort insights.
- **Solution:** Use tools like Python's **pandas. drop_duplicates()** or Excel's "Remove Duplicates" function to eliminate redundant entries.

2. Handling Missing Data:

- **Issue:** Missing values can lead to misleading visualizations.
- **Solution:**

i. Remove missing values if they are few and non-critical.
ii. Impute missing values using statistical methods (mean, median, or mode).
iii. Interpolate missing values in time-series data.

3. Correcting Inconsistent Data:

- **Issue:** Variations in formatting, spelling, or measurement units can create inconsistencies.
- **Solution:**

i. Standardize date formats (e.g., **YYYY-MM-DD** instead of **DD/MM/YYYY**).
ii. Ensure uniform capitalization (e.g., **"USA"** vs. **"usa"**).
iii. Convert measurement units (e.g., inches to centimeters).

4. Identifying & Removing Outliers:

- **Issue:** Extreme values can distort visual trends.
- **Solution:**

i. Use box plots to detect outliers.
ii. Apply z-score or IQR (**Interquartile Range**) methods to filter out anomalies.
iii. Consider domain knowledge before removing data points.

5. Fixing Data Entry Errors:

- **Issue:** Typos, incorrect values, and misplaced data affect accuracy.
- **Solution:**

i. Validate data entries before use.

ii. Cross-check data sources for correctness.

6. Standardizing Data Formats:

- **Issue:** Inconsistent formats make visual comparisons difficult.
- **Solution:**

i. Convert categorical values to a unified format (e.g., "Male"/"Female" vs. "M"/"F").

ii. Standardize date, currency, and numerical formats.

Tools for Data Filtering in Data Visualization

Table 6: Tools for data filtering

Tool	Usage
Python (Pandas, NumPy)	Data preprocessing and cleaning using scripts.
Excel & Google Sheets	Manual cleaning, removing duplicates, and standardizing formats.
SQL (Structured Query Language)	Filtering, correcting, and cleaning large databases.
OpenRefine	Cleaning messy data and standardizing records.
Tableau Prep	Preparing and cleaning data before visualization in Tableau.

d) Mine: The data mining step in data visualization involves analyzing the filtered and structured data to discover any patterns, trends, correlations, or anomalies. The statistical methods, machine learning algorithms, or other analytical techniques would thereby extract meaningful insights from the dataset. It involves mathematics, statistics, and data mining. The only thing that will be accomplished in this context is a very simple method of treatment program must, in a sense, sweep the data to find the minimum and maximum values for latitude and longitude, for the rational display of the data on a screen at an appropriate scale. In most cases, this will be far more complex than simply a pair of equations (as shown in fig 1.31) [11].

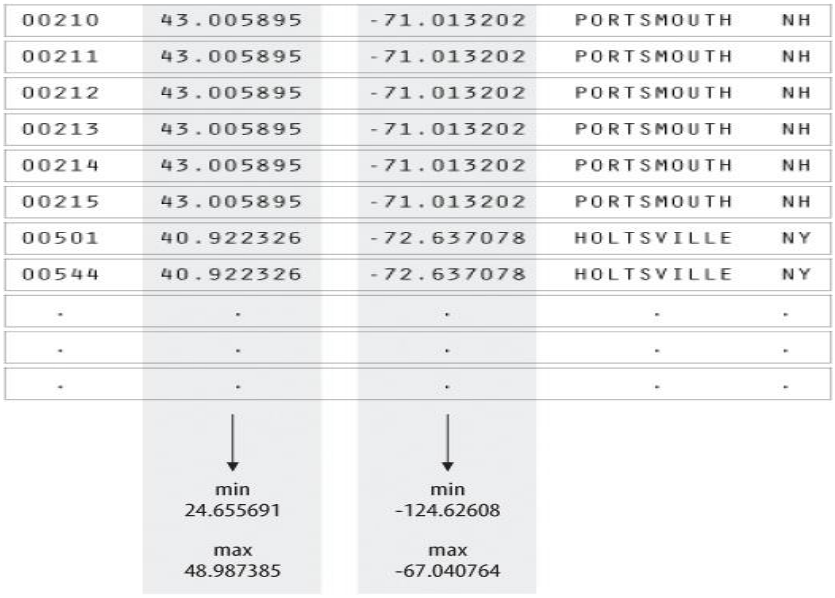


Fig. 1.31: Mining the data: Just compare values to find the minimum and maximum.

Objectives of Data Mining

1. Identify hidden patterns and trends in the data.
2. Find correlations and relationships between variables.
3. Detect anomalies or outliers that may indicate unusual events.

4. Summarize large datasets into key insights.
5. Prepare data for meaningful visualization.

e) Represent: This step is to set up a fundamental way in which a set of data is going to be expressed in lists, trees, etc. Here, since each zip code has a specific corresponding latitude and longitude point, it is possible to present its importance on a two-dimensional plot, with the minimum and maximum values of latitude and longitude chosen to extend the respective vertical and horizontal scales at both ends. This idea is illustrated in Fig 1.32.

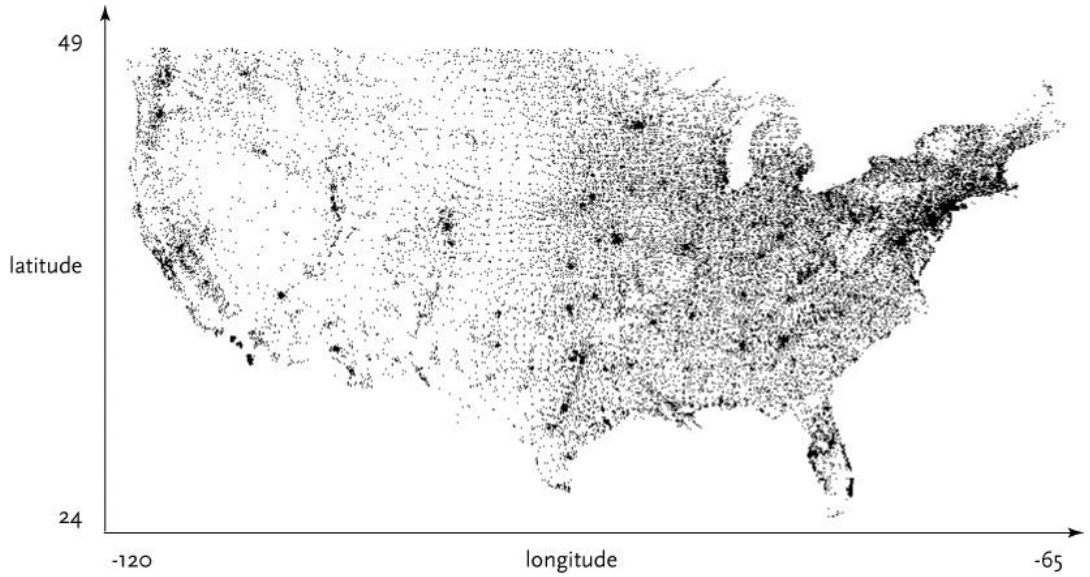


Fig. 1.32: Basic visual representation of zip code data.

The Represent stage is a pivot that tells you the single most critical decision in a visualization project and might cause you to reconsider previous stages. The way you represent the data might dictate the very first step (what data you acquire) and the third step (what specific pieces you extract).

f) Refine: At this step, graphical design techniques are applied further to define the representation by directing further emphasis on specific data (hierarchization) or modifying features (color) contributing to readability. In Fig 1.33, for example, hierarchy is established through the deep gray color background and presentation of selected points (all four-leading codes) as white and the deselected ones in medium yellow [11].

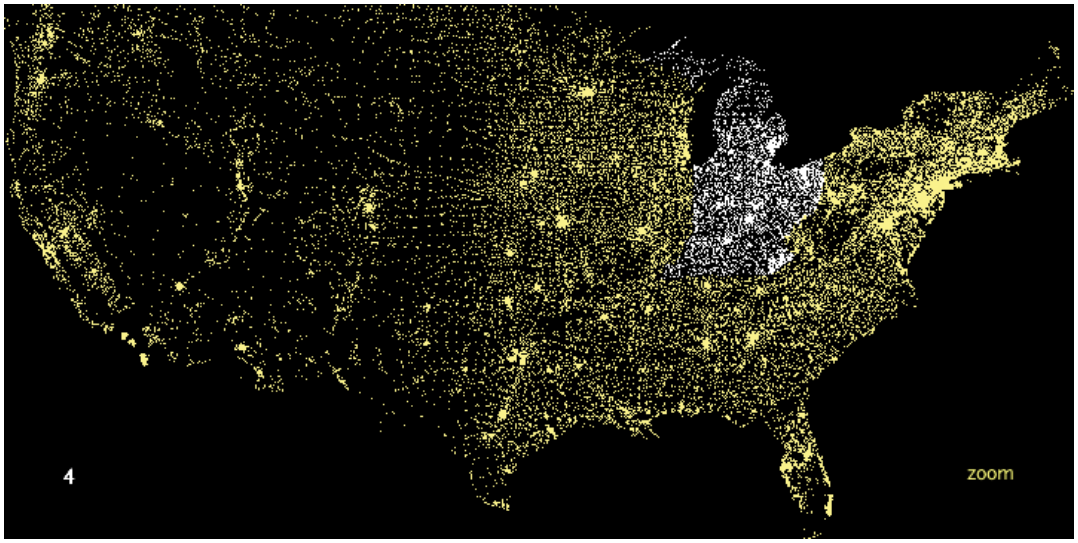


Fig. 1.33: Using color to refine the representation.

g) Interact: The subsequent phase of the process introduces interaction, allowing the user to manipulate or investigate the data. Interaction may include such things as the selection of a subset of the data or viewpoint change. As a second example of a phase influencing an earlier component of the process, this phase can influence the refinement step as well, since a change in viewpoint may necessitate that the data be designed differently.

In the Zip decode project, entering a number selects all zip codes that start with that number. Fig 1.34 and Fig 1.35 illustrate all the zip codes starting with zero and nine, respectively [11].

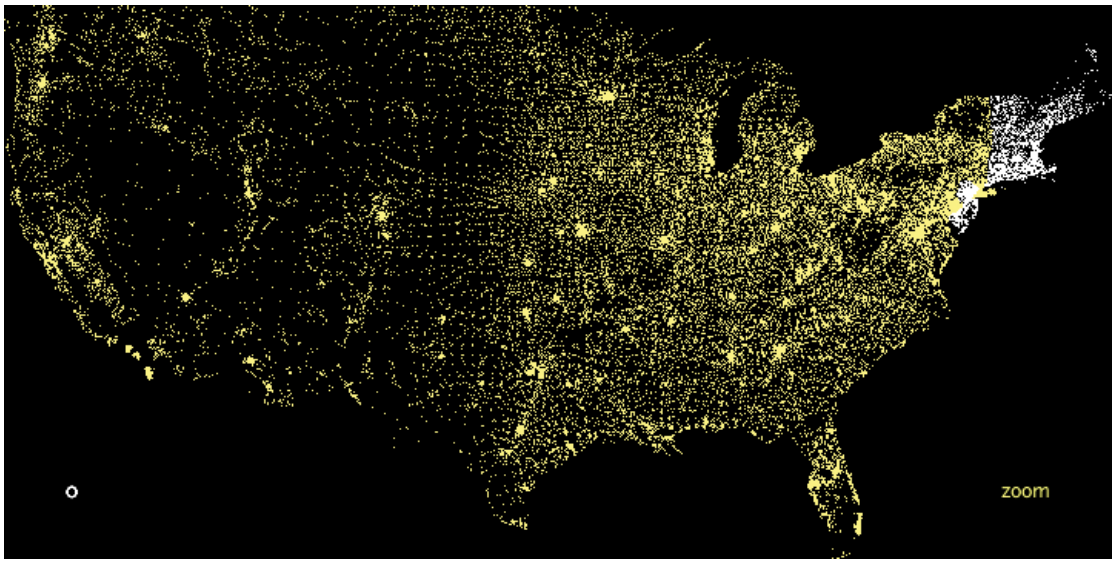


Fig. 1.34: The user can alter the display through choices (zip codes starting with 0).

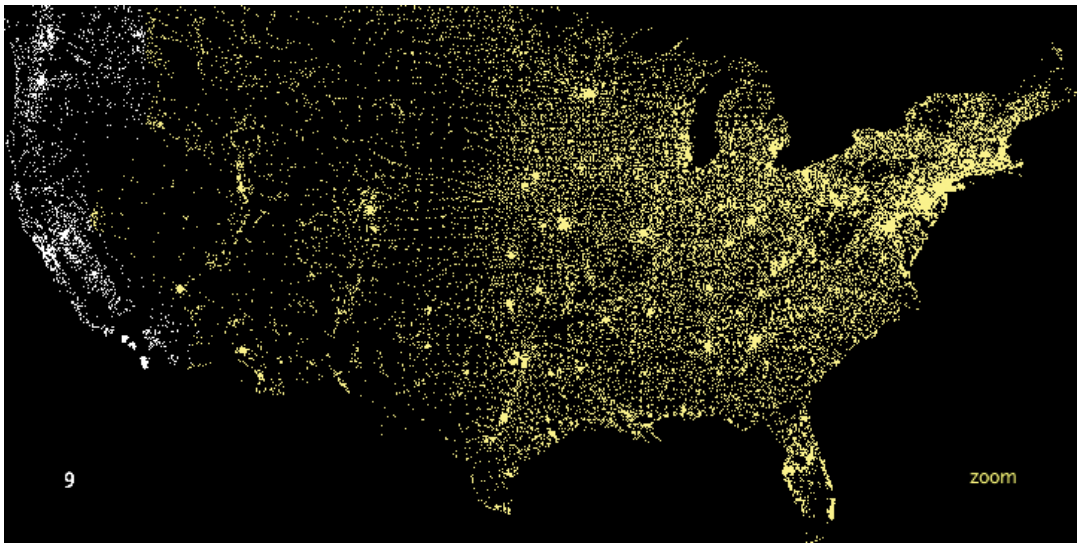


Fig. 1.35: The user can alter the display through choices (zip codes starting with 9).

Another addition to user interaction makes the users able to slide laterally through the display and cycle through various of the prefixes. By holding down the Shift key after inputting part or all of a zip code, users can overwrite the last number that has been typed in without pressing the Delete key to go backwards.

Typing is a very rudimentary type of interaction, yet it enables the user to quickly develop an impression of the structure of the zip code system. Just compare this sample application to the effort required to infer the same information from a table of city names and zip codes.

You can still type digits and observe what the area looks like when covered by the next group of prefixes. Fig 1.36 depicts the area under the two digits 02, Fig 1.37 depicts the three digits 021, and Fig 1.38 depicts the four digits 0213. At last, Fig 1.39 depicts what happens when you input a full zip code, 02139—a city name appears on the display [11].

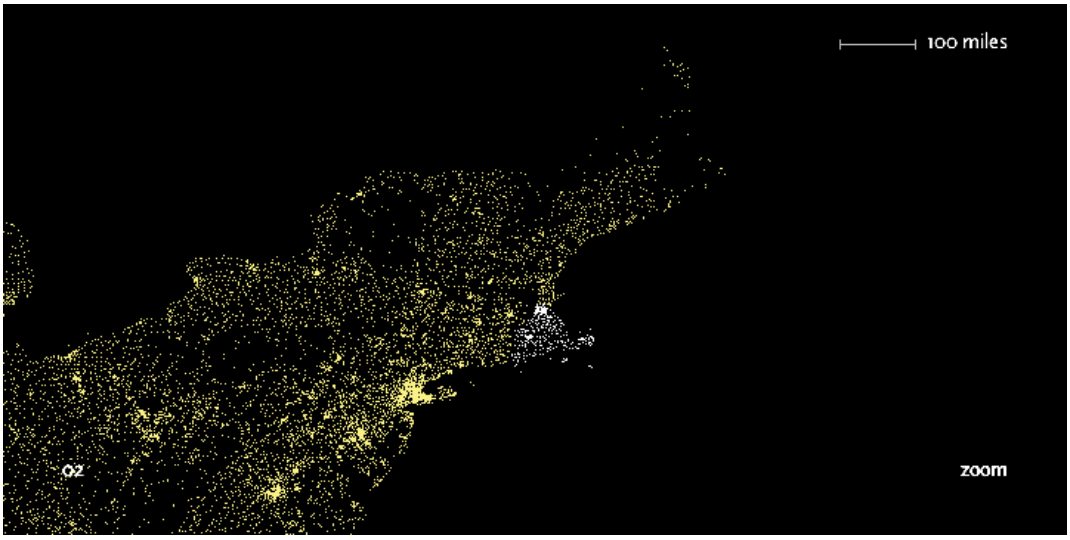


Fig. 1.36: Refining Zip Code Representation with Two Digits (02).

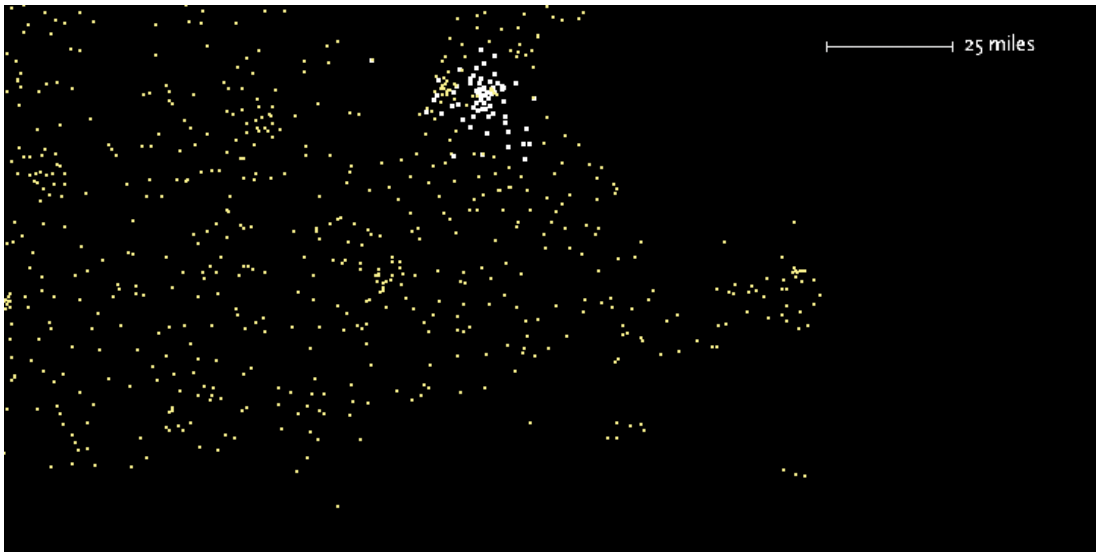


Fig. 1.37: Refining Zip Code Representation with Three Digits(021).

Furthermore, users can turn on a "zoom" capability that zooms in closer to each subsequent digit, with more detail appearing around the region and a steady rate of detail at every level. Since we've selected a map as our representation, we might include additional details of county and state lines or other geographic elements to enable viewers to equate the "data" space of zip code points with their knowledge of the local environment [11].

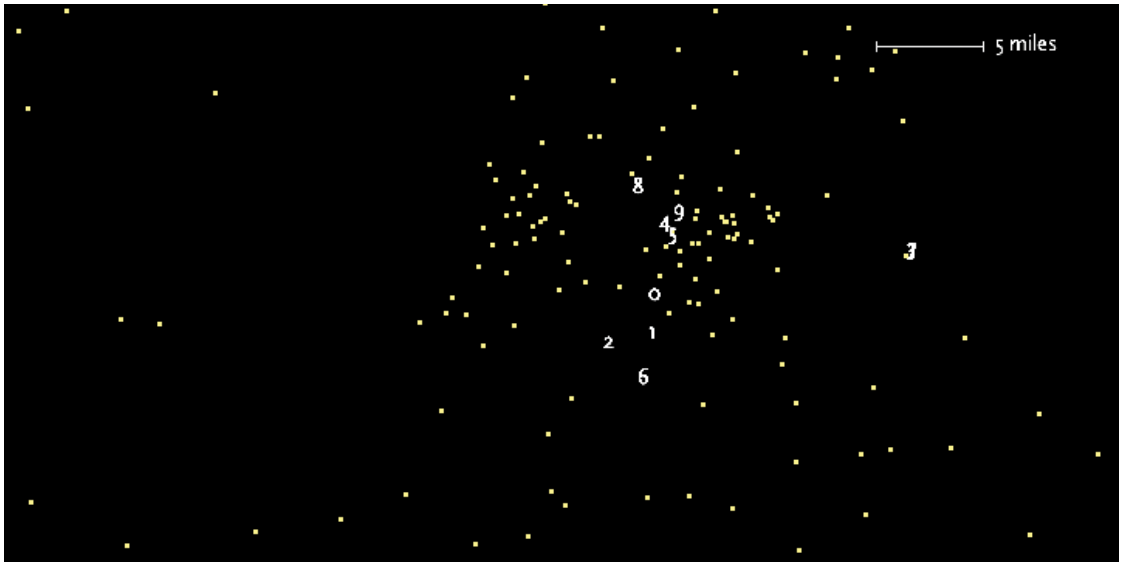


Fig. 1.38: Refining Zip Code Representation with Four Digits(0213).

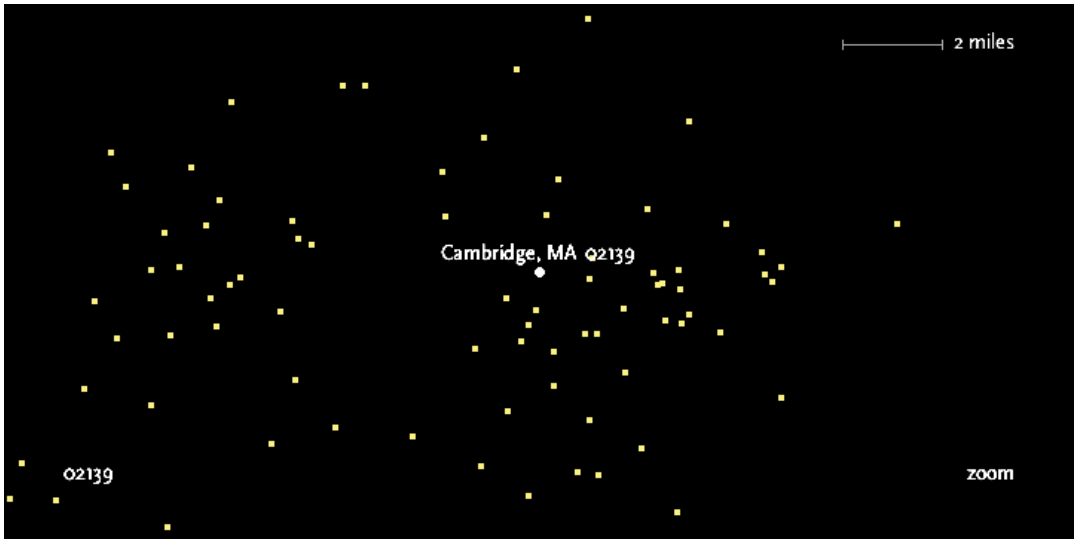


Fig. 1.39: Full Zip Code Representation (02139).

1.4 Data Visualization Tools and Techniques

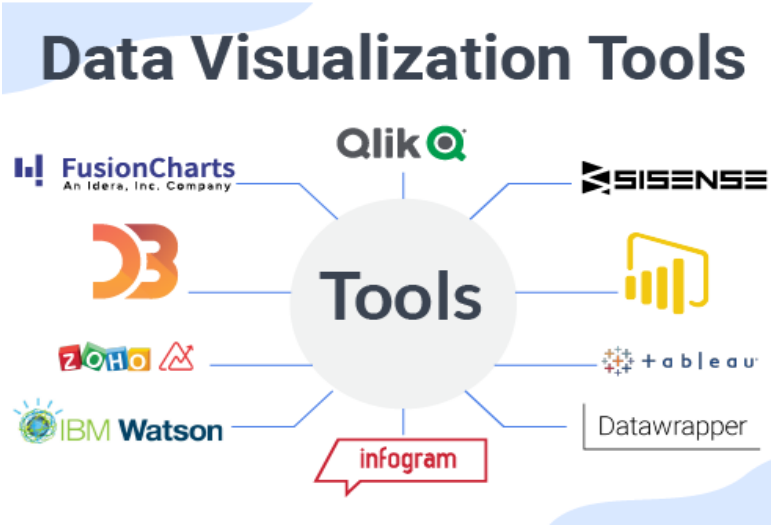


Fig. 1.40: Data visualization tools

1.4.1 Tableau:

Tableau is an excellent data visualization tool, a favourite of data analysts, scientists, statisticians, and others for visualizations, to gain clearer opinions from their computations. Tableau is widely known because it can ingest data and deliver the required data visualization output in the least amount of time possible. In other words, it can elevate your data into insights that can be used to drive actions in the future. It achieves all of this while granting the highest possible level of security, and it guarantees that they deal with you for security issues as soon as they arise or are brought to their attention by users.

Tableau is also capable of data preparation, cleansing, formatting, visualizations, sharing insights, and publishing to other end-users. Through data queries, you can gather insights from your visualizations, and these data are managed at an organizational scale using Tableau. In fact, according to many, it is like a lifesaver for Business Intelligence since one can handle data without having such high-end technical knowledge. Hence, Tableau may be utilized by individuals or in a mass for the more significant benefits of business teams and their organizations. Several organizations like Amazon, Lenovo, Walmart, Accenture, etc. already utilize Tableau. There are different Tableau products for different types of users, be they single-person-oriented or enterprise-oriented. Now, let's see what these may be in detail depicted in the fig 1.41 [12].

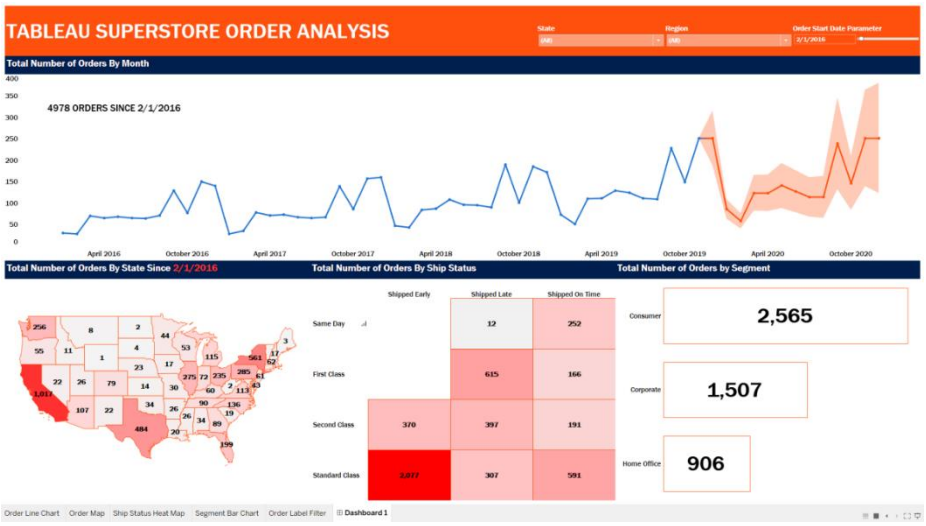


Fig. 1.41: Visual representation of Tableau

1.4.2 Power BI:

Microsoft Power BI is a business intelligence (BI) platform that provides nontechnical business users with tools for aggregating, analyzing, visualizing, and sharing data (as shown in fig 1.42). If users are familiar with Excel, they shouldn't find Power BI difficult to use. Moreover, since it is deeply integrated with other Microsoft products, it has become a versatile self-service tool that demands minimal upfront training.

Power BI is available for download on Windows 10 and 11 as Power BI Desktop or as mobile apps for Windows, Android, and iOS devices. Report Server is for those companies that must keep their data and reports on-premises. That version of Power BI requires an additional version of the desktop app - aptly named Power BI Desktop for Power BI Report Server [13].

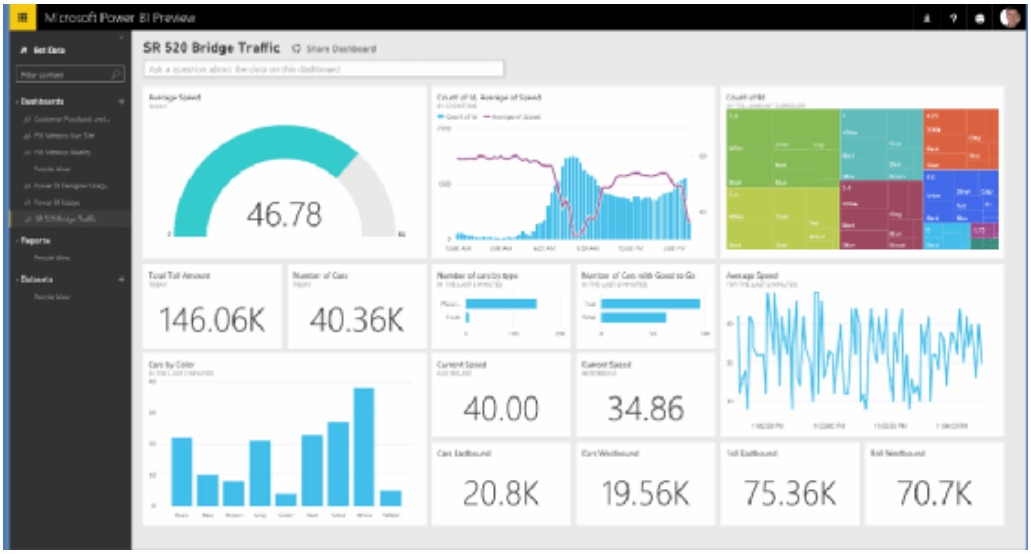


Fig. 1.42: Visualization of PowerBI dashboard

1.4.3 (a) Matplotlib & Seaborn:

Matplotlib is an extraordinary 2D array plotting visualization library for Python. It brings together the power of NumPy arrays with the rest of the SciPy stack to create a multi-platform data visualization library. The library was created in 2002 by John Hunter. So, let's try to touch on some of the advantages and features of Matplotlib [14].

- It compiles fast, runs smoothly because of numpy, and generally is easier to build.
- Was modified several times after having entered the public domain into a much better and more advanced library.
- The maintained graphical output is of such high quality and beauty with many users approaching.
- Easy to make basic and advanced charts.
- With a large community on board, the user/developer is greatly aided in fixing issues and debugging.

Example:

Bar plot using matplotlib: Find different types of bar plot to clearly understand the behaviour of given data.

Code:

```
# importing libraries
 import matplotlib.pyplot as plt
 import pandas as pd
 import numpy as np
 ts = pd.Series(np.random.randn(1000), index = pd.date_range(
                              '1/1/2000', periods = 1000))
 df = pd.DataFrame(np.random.randn(1000, 4), index = ts.index,
                                    columns = list('ABCD'))
 df3 = pd.DataFrame(np.random.randn(1000, 2),
              columns =['B', 'C']).cumsum()
```

```
df3['A'] = pd.Series(list(range(len(df))))
df3.iloc[5].plot.bar()
plt.axhline(0, color ='k')
plt.show()
```

Output of the code is shown in Fig 1.43

Output:

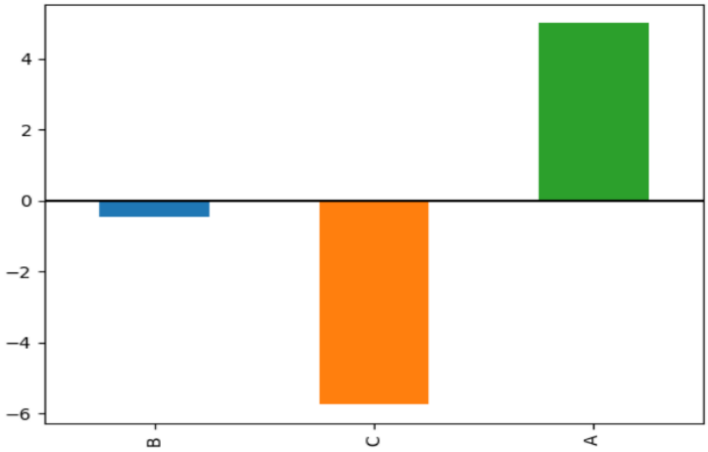


Fig. 1.43: Creating a graph using Matplotlib

(b) Seaborn is a library for statistical graphics in Python. Seaborn is built this way on top of matplotlib and is closely integrated with pandas data structures.

Seaborn helps you explore and understand your data. Its higher-level plotting functions work on dataframes and arrays that represent whole datasets. Internally, they perform the necessary semantic mapping and statistical aggregation to produce useful plots. Its dataset-oriented, declarative API allows you to concentrate on the meaning of the various elements of your plots rather than on the specifics of how to draw them [15].

- Built-in themes aid better visualization
- Statistical functions aiding better data insights
- Better aesthetics and built-in plots
- Helpful documentation with effective examples

Example:

Code:

```
 # Import seaborn
import seaborn as sns
# Apply the default theme
sns.set_theme()
# Load an example dataset
tips = sns.load_dataset("tips")
# Create a visualization
sns.relplot(data=tips, x="total_bill", y="tip", col="time", hue="smoker", style="smoker", size="size")
```

Output of the code is shown in Fig 1.44

Output:

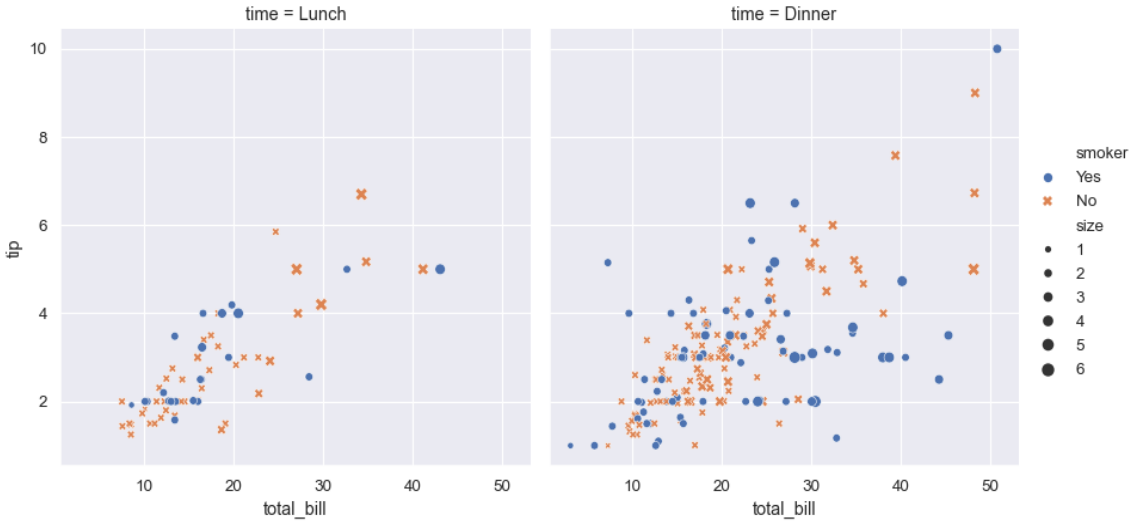


Fig. 1.44: Creating a graph using Seaborn.

1.4.4 D3.js: D3.js is a JavaScript library for manipulating documents based on data. It creates dynamic and interactive data visualizations in web browsers with a mixture of HTML, SVG, and CSS. It thus allows for the effective telling of data-driven stories and analyses.

Data visualization chooses to provide data in graphical and pictorial forms to work with even very complicated data and easy comprehensibility. The visualization provides a means of spotting patterns easily and allowing comparative analyses to facilitate decision-making with ease. There are very simple frameworks such as D3.js (as shown in fig 1.45), where the construction of these visual representations can profoundly change [16].

Fig. 1.45: Showing Candlestick Chart using D3.js

1.4.5 Excel: Microsoft Excel is a versatile spreadsheet application developed by Microsoft for data organization and financial analysis; used worldwide. Its important features include data entry, management, financial models, and charts. Due to its capability and flexibility, Excel has a special position in budgeting, forecasting, and analyzing as used in finance and accounting. It allows functions, formulas, and shortcuts to support more productivity and efficiency. Excel is an indispensable tool for finance, accounting, and other professionals needing data organization and analysis.

In other words, Microsoft Excel is the software created for organizing numbers and data with the help of spreadsheets. Normally, its analysis is common all over the world and is used by companies of different sizes to carry out financial analysis(as shown in fig 1.46) [17].

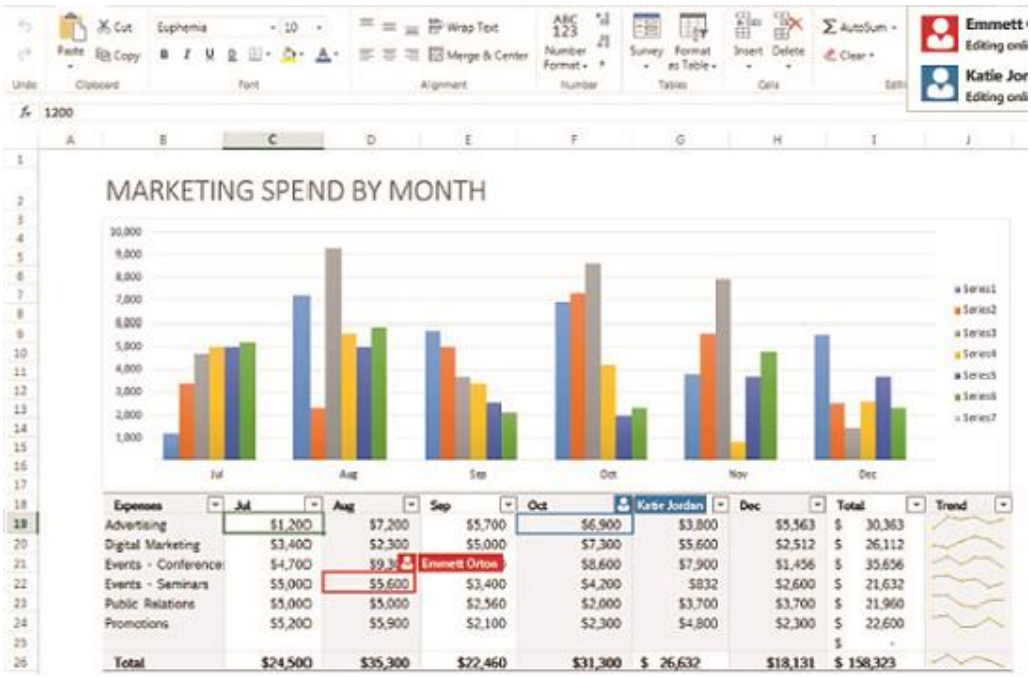


Fig. 1.46: Visualization of data using Microsoft Excel

Chapter 2

MAPPING DATA ONTO AESTHETICS

Simran[1], Satyam Sharma[2] and Gurpreet Kaur[3]

[1,2]GNA University, Phagwara

[3]Lovely Professional University, Jalandhar

2.1 Fundamentals of Data Mapping:

Mapping data onto aesthetics refers to how we visually encode data using different levels of representation in charts or graphics. Such visible properties (aesthetics) would better show the patterns, relationships, and insights available. Limited Key aesthetic mappings include position, color, size, shape, and transparency [18].

Data visualization is the process of taking raw data and systematically and logically converting them into visual elements that comprise the final graphic. From that perspective, these variants may look completely different; for example, when viewing a scatter plot, pie chart, and heatmap, one might think these visualizations have no similarities. However, all data visualizations can be described through a common language: a description of how the data values are "mapped" onto graphics which are usually rendered with ink on paper or colored pixels on the screen. The vital thought that should strike one is this: All data visualizations map data values into measurable attributes of the graphic produced. We shall refer to these attributes as aesthetics.

The aesthetics define every aspect attached to a graphical element; a few examples are included in Figure 2.1.

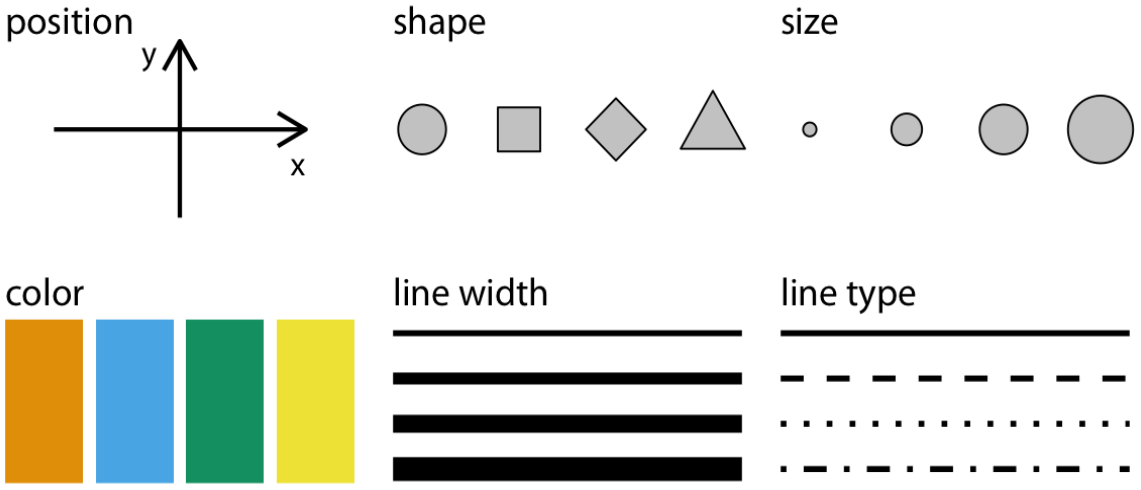


Fig. 2.1: Commonly employed aesthetics in data visualization include position, shape, size, color, line width, and line type. Some aspects can depict both continuous and discrete data (position, size, line width, and color), while others can, for the most part, represent only discrete data (shape and line type).

The position of a graphical element is very important, i.e., where the element is. In standard 2d graphics, we refer to positions through an x and y value; however, other coordinate systems, including one or three-dimensional visualizations, exist. Next, all graphical elements involve a certain shape, size, or color. Even

in a purely black-and-white drawing, graphical elements must have a certain color to, at the very least, be visible- for instance, black if the background is white and white if it is black. Finally, as far as we visualize the data using lines, they can display different widths or dash-dot patterns. New aesthetics may further emerge in diverse situations beyond the examples shown in Figure 1: for instance, for textual representation, we specify the font family, font face, and font size, or if graphical objects overlap, we might have to specify whether they are somewhat transparent.

2.1.1 Position: Positions refer to the arrangement of data points along one or more axes in a chart or graph. This is known to be one of the most fundamental ways to encode data visually, as the human mind effortlessly interprets spatial variances.

- **X-Axis (Horizontal):** It represents independent variables, such as time, categories, or sequences.
- **Y-Axis (Vertical):** It represents dependent variables, such as quantity, percents, or concrete numbers**.**
- **Z-Axis (Depth in 3D graphs):** One of the dimensions of information used in 3D visualization.

Types of Position:

a. Scatter Plot: Scatter plots are graphs that show the relationship between two variables in a dataset. They present data points on a two-dimensional plane or a Cartesian coordinate system. The independent variable or attribute is plotted on the X-axis, while the dependent variable is plotted on the Y-axis. These are often referred to as scatter graphs or scatter diagrams.

Scatter plots provide immediate reports on large amounts of data. It can make its case on the following points:

In the case of given points having an extremely large dataset. Each pair has a set of values. The data under consideration is in numeric form (as shown in fig 2.2).

Fig. 2.2: Displaying Scatter Plot

b. Line Chart (Position for Trends Over Time): Line graphs, or line charts and line plots, are used in graphing. It is a kind of graph that, pictorially, represents the data so that the raw data sets are easily understood. In the case of a line graph, the elastic line connects data points that are represented either with points or wedges. In other words, A line graph or line chart is a chart representing the data showing the interdependence of two or more variables about time. It is created by drawing data points together with straight-line segments (as shown in fig 2.3) [2].

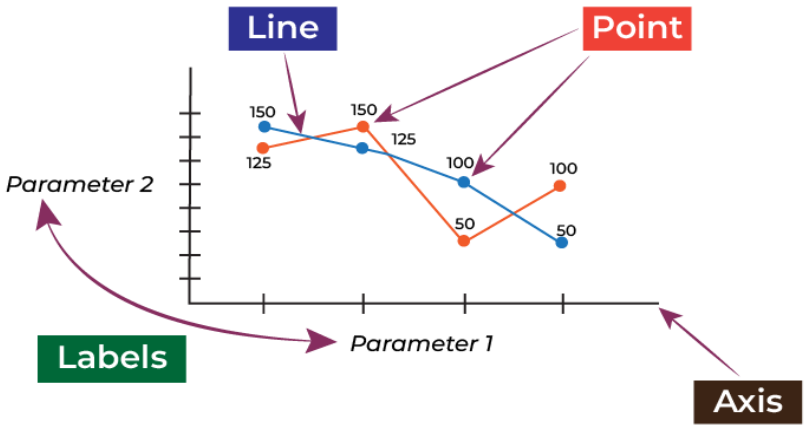


Fig. 2.3: Displaying Line Chart

Components of Line Graph:

The components of the line graph are the following:

- **Title:** It is simply the name of the graph plotted.
- **Axes:** The line graph has two axes i.e. X-axis and Y-axis.
- **Labels:** The label provided to the x-axis and y-axis.
- **Line:** It is the line segment that is used to join two or more points.
- **Point:** It is simply a point assigned at each segment.

c. **Bar Chart (Position for Categorical Comparison):** A bar chart or bar graph is a chart or graph representing explicit data in the form of rectangular bars. Briefly, a bar graph is a rectangular bar graph or a vertical or horizontal bar graph. A vertical bar chart is also referred to as a column chart. The bars vary in length based on the value since the bars are proportional to the value. A bar chart is a type of chart that presents data with rectangular parallel bars (as shown in fig 2.4) [18].

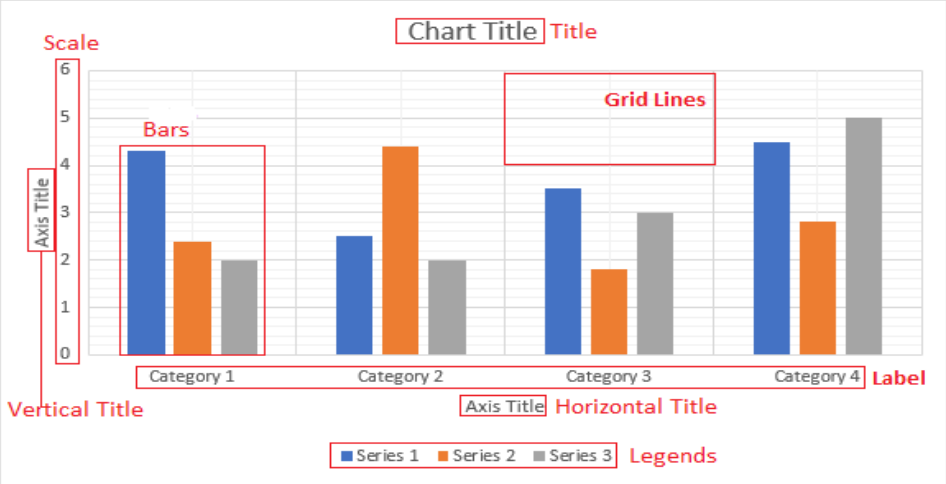


Fig. 2.4: Displaying Bar chart

Components of Bar Graph:

- **Chart Title:** It refers to the title of the bar chart. Here we can describe what is being represented in the chart.
- **Grid Lines:** The vertical and horizontal lines in gray color are called grid lines.

- **Bars:** A bar is about a value. It is either horizontal or vertical. The biggest bar is for the most significant value.
- **Axis Title:** A bar graph contains two titles, one vertical and the other horizontal. Both axis are connected. We can name the axis title for better understanding. Let's suppose the vertical axis shows expenses. So, we can name the vertical axis as Expenses (in rupees). The expenses could be of various types, so we can name types of expenses on the horizontal axis.
- **Labels:** We can also label the title of the horizontal axis. For instance, the categories of expenses can be labeled as medical, transport, office, etc.
- **Legends:** A legend indicates what a bar is showing. It is also referred to as the key of a chart. Take the following graph; if we put 2019 in the place of Series 1, then the blue bars of the graph indicate the year 2019 data.
- **Scale:** The scale is used to represent the vertical values. It can contain rupees, population, size, etc.

d. Bubble Chart (Position for Three Variables): A bubble chart (also referred to as a bubble plot) is an expansion of the scatter plot that one employs to examine associations between three numerical variables. In a bubble chart, one dot represents a single point of data, and for each point, the values of the variables are represented by dot size, horizontal position, and vertical position.

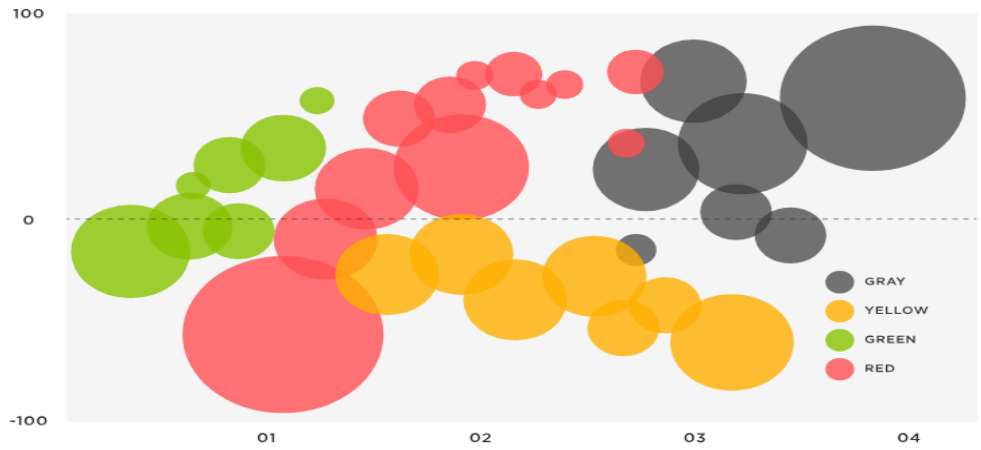


Fig. 2.5: Showing Bubble Chart

A bubble chart is a graph that displays three variables in terms of bubble size, color, and location (as shown in fig 2.5).

Like other graphs, the bubble chart has an x-axis and y-axis to display two variables, and the size of the bubbles shows the third variable. The larger the bubble, the greater the value of the third variable.

For instance, a bubble chart can be utilized to illustrate the correlation between age (x-axis), income (y-axis), and expense (size of bubble). The chart indicates that income increases as age increases, but so does the expense.

2.1.2 Color:

Color is a strong visual encoding of data that is used to distinguish categories, emphasize trends, and call attention to significant features. Color can be used to represent categories, numerical values, or intensities within a dataset.

Key Components of Color Mapping:

1. **Hue** → Different colors for categorical data.
2. **Saturation** → The intensity of a color (bold vs. faded).
3. **Brightness** → The lightness or darkness of a color (used in gradients).

Types of Color Mappings in Data Visualization:

1. **Categorical Color Mapping (Distinct Colors for Groups):**

   Categorical colors assist users in projecting non-numeric meaning onto objects in a visualization. They are created to be visually different from each other. The Spectrum categorical 6-color palette has been made distinguishable for color vision-deficient users (as shown in fig 2.6).

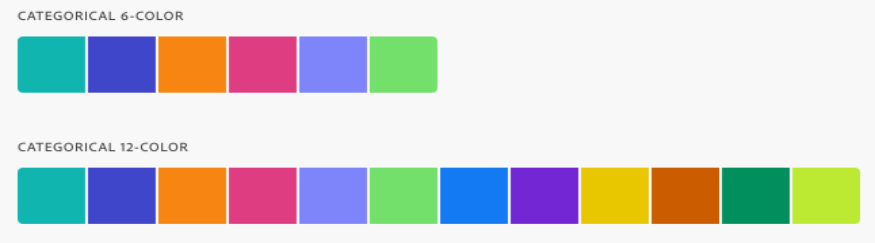


Fig. 2.6: Categorical Color Mapping

2. **Sequential Color Mapping (Gradient for Numerical Data):**

   Sequential colors have numerical meanings. These are a range of colors that transition from light to dark.

   Spectrum accommodates 5 palettes intended for the use of sequential data(as shown in fig 2.7):

   i. Viridis; ii. Magma iii. Rose
   iv. Cerulean v. Forest

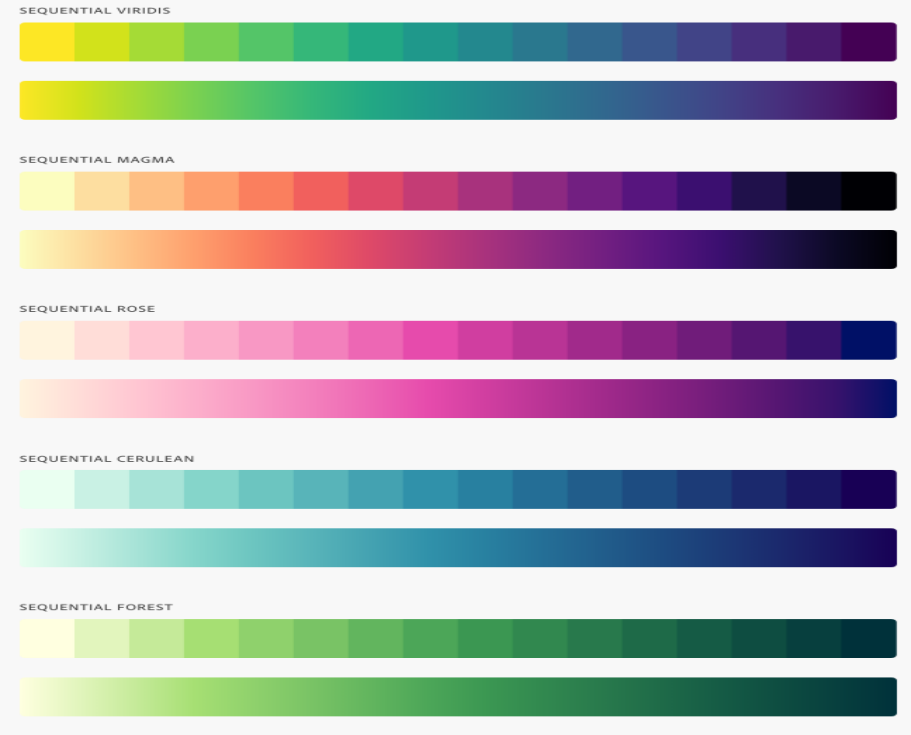


Fig. 2.7: Sequential Color Mapping

3. **Diverging:**

   Diverging colors also carry numeric significance. They come in handy when working with negative values or ranges that have two extremes and a baseline in the middle. Diverging palettes are a set of 2 gradations of colors that converge in the middle.

   Spectrum contains 3 palettes that are specifically meant to be used with diverging data (as shown in fig 2.9):

   - Orange-yellow-seafoam
   - Red-yellow-blue
   - Red blue

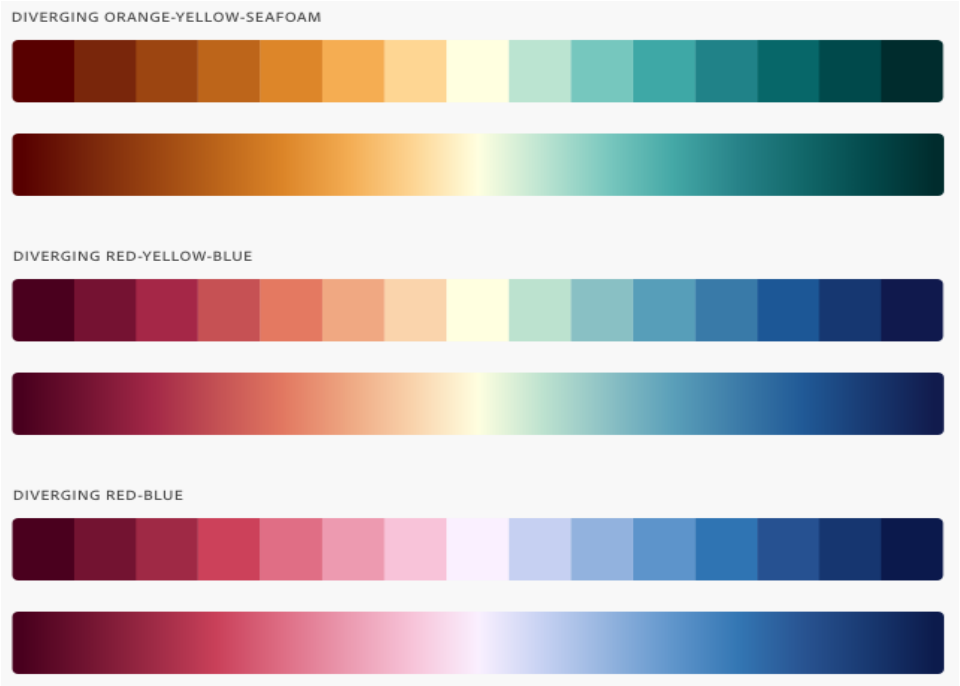


Fig. 2.8: Diverging Palette

2.1.3 Size:

Size is a visual encoding method that is employed to show quantitative (numerical) data by changing the area, length, or diameter of elements in a visualization. The larger the size, the higher the value, and the smaller the size, the lower the value. Size is convenient when comparing values but must be used with caution to preserve readability and prevent misinterpretation.

Example:

a. **Pie chart:** Pie chart is a well-known and visually intuitive data representation tool that makes it easy to comprehend complicated information briefly.

   Pie chart splits data into slices, every slice showing a portion of the total (as shown in fig 2.9), to enable easy comparison of various categories making it easier to consume complicated information in an easy, intuitive format.

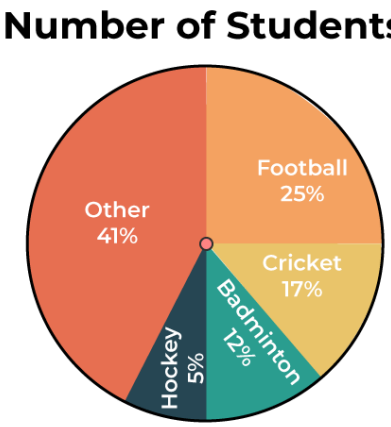


Fig. 2.9: Pie Chart Representation

Types of Pie Charts:

There are some variations or types of pie charts. some of the popular ones are:

1. **3D Pie Chart:** A 3D pie chart provides depth to the otherwise two-dimensional pie chart by graphing it in three dimensions.
2. **Doughnut Chart:** A doughnut chart is analogous to a pie chart but with a hole in the middle.
3. **Exploded Pie Chart:** In an exploded pie chart, a slice or slices are moved away from the rest of the pie to highlight their significance or to make them prominent.
4. **Nested Pie Chart:** This chart can also be called a multi-level pie chart or a hierarchical pie chart. A nested pie chart is composed of several rings of pie charts in which every ring symbolizes an alternative level in the data hierarchy.
5. **Ring Chart:** A ring chart is like a doughnut chart but made up of several rings rather than one. Each ring corresponds to a different data category, and the size of each segment in the ring is proportional to its share of the total.

2.1.4 Shape:

Shape is an important aesthetic mapping in visualization(as shown in fig 2.10). that assists in differentiating between various categories or groups of data. In contrast to size or color, the shape is only utilized for categorical data to visualize separate groups without suggesting any numerical connection. Shapes can be applied across types of visualizations, particularly scatter plots, legends, maps, and icon-based visualizations.

Fig. 2.10: Types of Shapes

2.1.5 Transparency

Transparency (alternatively referred to as opacity) is a visual property that governs the amount of visible or transparent an element in a graph or chart. You can control overlapping data points, highlight key information, and achieve depth in visualization by modifying transparency.

Transparency is specified on a scale of 0% (fully transparent) to 100% (completely opaque) (as shown in fig 2.11).

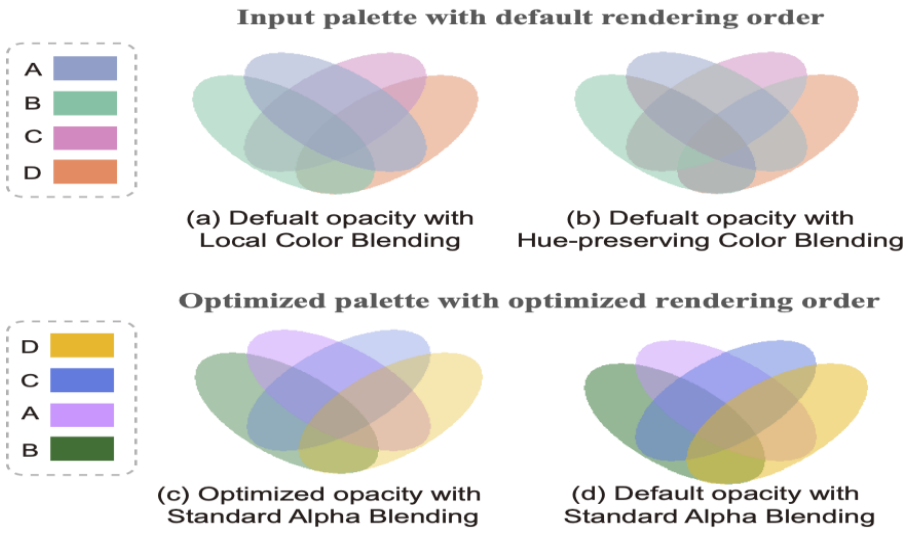


Fig. 2.11: Transparency in Data Visualization

2.2 Visualizing Data Types:

Data visualization is about converting intricate data into clearer and comprehensible visual formats. The kind of visualization depends on the type of data set, the audience, and the insights to be conveyed. The common types of visualizations below can be collated per their use cases.

The kinds of data in visual appearance treat four main aspects illustrated in fig 2.12:

1. **Nominal (Categorical) Data:** This does not have an ordinal ranking. For instance, colors, names, and gender.
2. **Ordinal Data:** Some ordinal rank exists which ordered categorical data without equal intervals. For instance, satisfaction ratings and education level.
3. **Discreet (Countable) Numerical Data:** This describes the whole numbers, wherein each has a unique value. For instance: the number of students, sales of products, etc.
4. **Continuous (Measured) Numerical Data:** Data which, in its interval ranges, has infinite possibilities. For instance: height, temperature, etc.

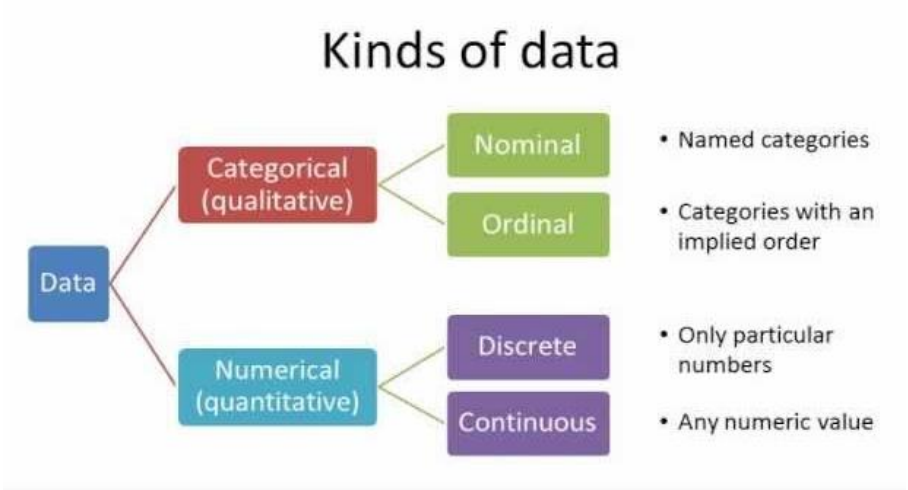


Fig. 2.12: Types of Data Categorization

1. **Nominal Data:**

Nominal values refer to available discrete units of variables or categories. You can think of them as simply "labels." This type of data is unordered. Meaning, it doesn't matter if you rearrange the order of its values; the meaning won't change. Example:

- **Colors:** Red, Blue, Green, Yellow
- **Countries:** USA, Canada, Germany, Australia
- **Car Brands:** Toyota, Ford, BMW, Honda
- **Blood Types:** A, B, AB, O

Techniques used for nominal data visualization:

a. Bar Charts: A bar chart, also known as a bar graph, gives a pictorial representation of numeric values for the numeric levels of a categorical feature in terms of length. The levels of categorical features and values are plotted on their axises in bar charts. Each level of the feature occupies its rectangular bar as illustrated in fig.2.13. The height of a bar reflects the magnitude of the desired percentage for this category. Bars are placed on the same baseline, allowing easy and accurate visual comparison of values.

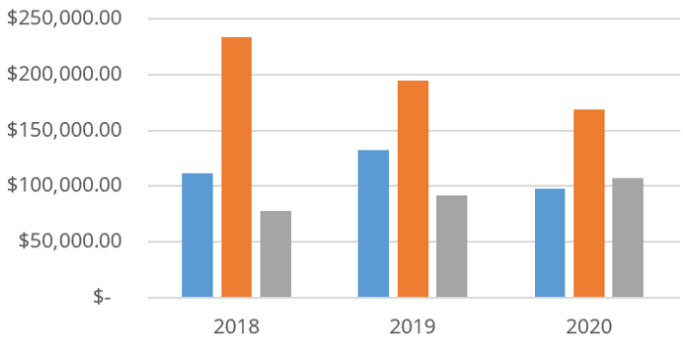


Fig. 2.13: Bar Chart for nominal data

b. Pie Charts: A pie chart is a way of showing information represented in a circular graph (as shown in fig 2.14) format where each slice of pie denotes a piece or proportion from the whole. All slices of pie sum up to a whole of 100 percent and 360 degrees.

A pie chart offers a very handy manner of visually depicting data to understand the distribution of values in any given data series. It is also referred to as a circle chart. The slices are usually represented as percentages to indicate their proportion to the whole. However, this is not mandatory; a pie chart can also be drawn by converting the data values directly into degrees. But percentages are also easier for a layman reader.

Fig. 2.14: Pie Chart for nominal data

c. Heatmaps: A heatmap is a colored grid that depicts values for a main variable of interest over two axis variables. The two-axis variables can be categorized into some ranges, like in bar charts or

histograms. The color of each cell indicates the value of the main variable in the range to which it corresponds.

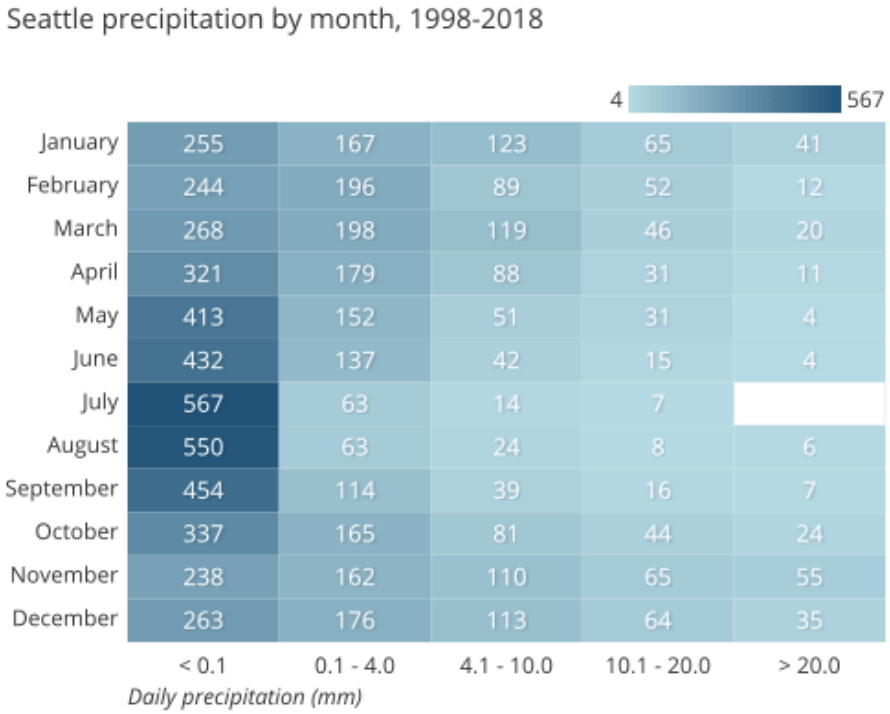


Fig. 2.15: Heatmap example

The heat map above in fig 2.15 demonstrates how precipitation is distributed from one day to another, segregated by month. This data was recorded in Seattle, Washington, over a period of eleven years. Each cell shows a numeric count, as in a data table, but also, with some coloring, scoring it such that higher counts are shown with darker coloring. On the heat map, it is possible to see that most of the days, dark in the left column, had no precipitation throughout the year. The color pattern in the cells throughout each month also shows that rains are more frequent in winter from November to March, declining in frequency during summer months, July and August.

2. Ordinal Data:

Ordinal data is a type of qualitative (categorical) data that consists of categories ordered in a meaningful way but does not guarantee that the interval between these categories is constant or can be measured. This means ordinal data allows for comparison to the extent that one category can be said to be higher or lower than another, but the exact distance between the categories cannot be said to exist. Example:

1. Customer Satisfaction Ratings

A survey asks customers to rate their satisfaction with a product:

- Very Dissatisfied (1)
- Dissatisfied (2)
- Neutral (3)
- Satisfied (4)
- Very Satisfied (5)

The responses have a clear ranking, but the **difference between "Neutral" and "Satisfied" may not be equal to the difference between "Dissatisfied" and "Neutral"**.

2. Education Levels

Levels of education ranked in order:

- High School
- Bachelor's Degree
- Master's Degree
- PhD

A PhD is ranked higher than a Master's, but the difference in knowledge or skill between each level is not exactly measurable.

Techniques used in ordinal data visualization:

a. Line Charts: Line graphs are suitable for representing quantitative data, while they are also suitable for ordinal qualitative data. They display changes in the value of the data through ordered values for a criterion.

Construction of a line graph generally involves plotting points corresponding to each ordered value for a criterion and then connecting these with lines as shown in fig 2.16:

Example: Customer Satisfaction Ratings Over Months

We have collected **customer satisfaction ratings** (ordinal data) over six months. The ratings are categorized as:

- **Very Unsatisfied (1)**
- **Unsatisfied (2)**
- **Neutral (3)**
- **Satisfied (4)**
- **Very Satisfied (5)**

The data shows the **average rating** per month:

Month	Average Satisfaction Rating (1-5)
January	3.2
February	3.5
March	3.8
April	4.0
May	4.2
June	4.5

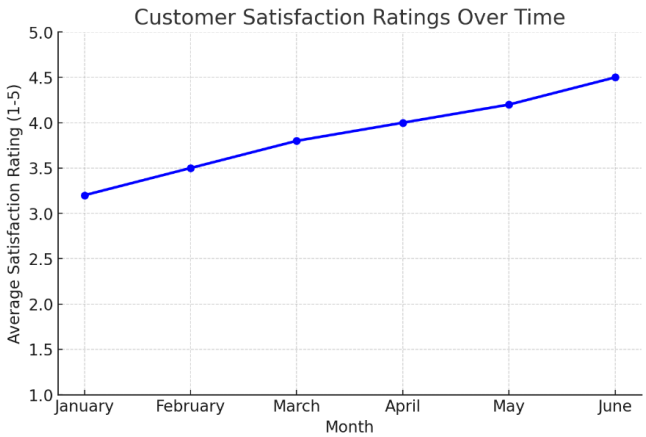


Fig. 2.16: Line Chart

1. **X-Axis (Months)** – Displays the months from January to June, showing the progression of time (as shown in fig 2.16).
2. **Y-Axis (Satisfaction Ratings)** – Represents ordinal data (customer satisfaction levels) on a scale from 1 (Very Unsatisfied) to 5 (Very Satisfied).
3. **Line Trend** – The line gradually increases, indicating an improvement in customer satisfaction over time.
4. **Markers (o)** – Each point on the graph represents the average rating for that month, making it easy to track changes.

b. Diverging Stacked Bar Charts: This type of chart is slightly more complex to display ordinal data, especially Likert survey results when responses vary from negative to positive (strongly disagree to strongly agree).

Example: Employee Job Satisfaction Survey

A company conducted a survey where employees rated their job satisfaction on a 5-point Likert scale as shown in fig 2.17:

1. Strongly Disagree
2. Disagree
3. Neutral
4. Agree
5. Strongly Agree

Below is the survey data for different departments:

Department	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
HR	5%	10%	15%	40%	30%
IT	10%	15%	20%	30%	25%
Marketing	8%	12%	25%	35%	20%
Finance	6%	9%	20%	40%	25%

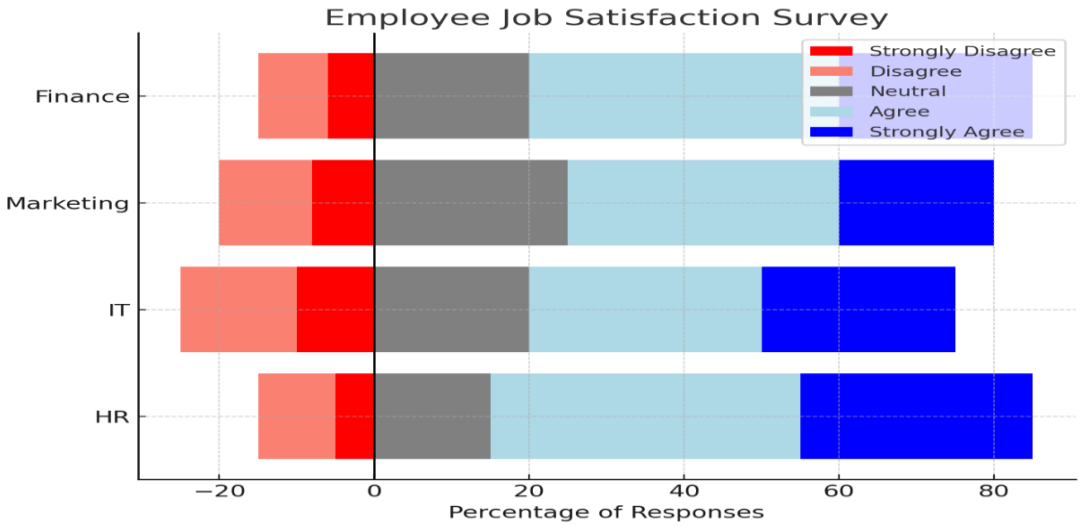


Fig. 2.17: Stacked Bar Chart

1. Negative Responses (Left Side in Red): Strongly Disagree & Disagree responses are displayed on the left, creating a diverging effect. This helps visually separate negative feedback from positive feedback.

2. Neutral Responses (Center in Gray): The neutral responses are placed at the center, acting as a reference point.

3. Positive Responses (Right Side in Blue): Agree & Strongly Agree responses are displayed on the right, showing positive sentiment.

4. Vertical Reference Line at Zero (Neutral Point): The vertical black line at zero divides negative and positive opinions.

5. Comparison Across Departments: HR and Finance have higher positive satisfaction (longer blue bars). IT and Marketing show a more balanced mix of satisfaction and dissatisfaction.

3. **Discrete Numerical Data Visualization:** Discrete numerical data have a countable set of distinct value types (e.g., number of students in a class, total customer complaints per month). The methods for visualization are particularly useful to evidence trends, comparisons, and distributions due to the presence of gaps between the discrete data values.

Techniques used for visualizing discrete data:

a) Column Chart (Vertical Version of Bar Chart): A vertical variant of a bar chart used to compare discrete numerical data across varying categories.

Example: Number of Products Sold Per Month:

A store tracks the number of electronic gadgets sold over **six months** as shown in fig 2.18:

Month	Units Sold
January	120
February	150
March	180
April	140
May	200
June	170

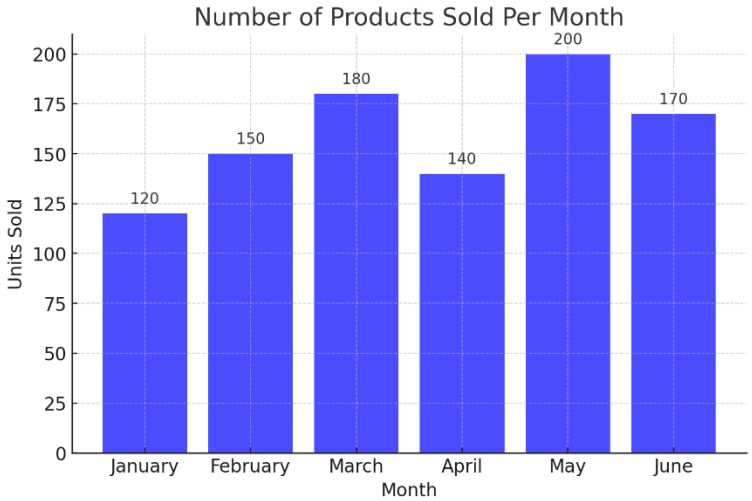


Fig. 2.18: Bar chart using discrete data

1. **X-Axis (Months)** – Shows the time (January to June).
2. **Y-Axis (Units Sold)** – Displays the number of products sold.
3. **Bar Heights** – Represent sales numbers for each month.
4. **Value Labels on Bars** – Help in reading exact sales figures.
5. **Gridlines** – Improve readability of sales differences.

b) Box Plot: A Box Plot talks about the distribution, spread, and outliers among the observations in a dataset. It summarizes five key statistics:

1. Minimum – The lowest number (without outliers).

2. First Quartile (Q1) – The lower quartile that corresponds to the 25th percentile.

3. Median (Q2) – The 50th percentile or the centre value.

4. Third Quartile (Q3) – The upper quartile that corresponds to the 75th percentile.

5. Maximum – The highest number (without outliers).

The whiskers extend to the highest and lowest values contained within a distance of 1.5×IQR (Interquartile Range). Outliers appear as individual points beyond the whiskers.

Example: Exam Scores of Students

A teacher records the exam scores of 20 students (as shown in fig 2.20):

[45, 50, 52, 55, 60, 61, 65, 67, 70, 72, 75, 78, 80, 83, 85, 88, 90, 92, 95, 98]

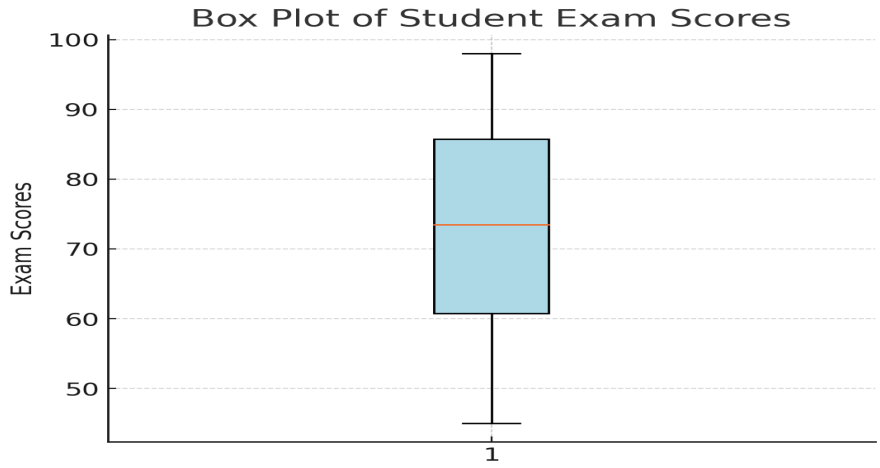


Fig. 2.19: Box Plot

1. Box (Light Blue Area) – Represents the Interquartile Range (IQR) (middle 50% of the data).

2. Line Inside the Box – The median (Q2) (middle value of the dataset).

3. Whiskers – Extend to the minimum and maximum values within 1.5×IQR.

4. Outliers (if any) – Appear as individual dots beyond the whiskers.

5. Y-Axis (Exam Scores) – Shows the range of student scores from 45 to 98.

4. **Continuous Numerical Data Visualization:** Continuous numerical data is data that is used in records where the variable can take any value in a given range (e.g., height, weight, temperature, time). Unlike discrete data, continuous data is infinitely divisible (e.g., 2.1 kg, 2.15 kg, 2.155 kg). Plotting methods involve employing appropriate visual presentation methods to represent the continuous data in a way that reflects distribution, trend, and pattern.

Techniques used for visualizing continuous numerical data:

a. Histogram: A histogram refers to a type of bar chart that provides a graphical representation of the distribution of continuous numerical data by grouping the data into intervals or bins. It gives an overall idea of the shape, dispersion, and central tendency of the data being displayed.

Example: Distribution of Student Test Scores

A teacher records test scores of 50 students ranging from 40 to 100(as shown in fig 2.20)

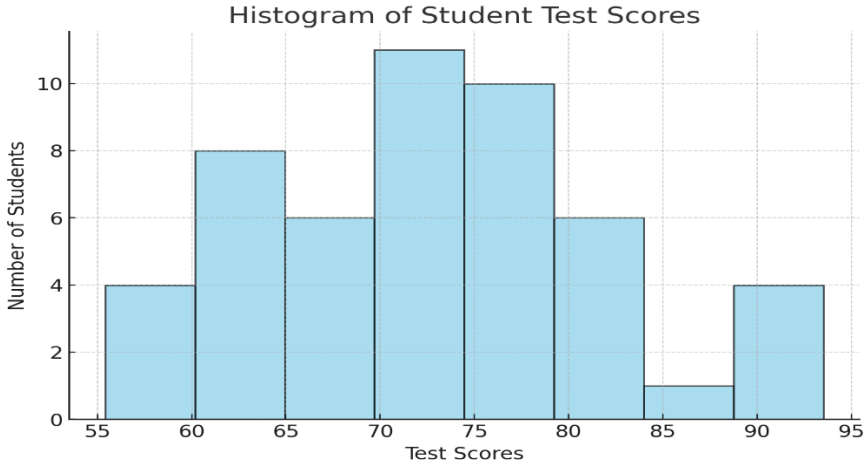


Fig. 2.20: Bar Chart using continuous data

1. **X-Axis (Test Scores)**: Divided into 8 bins (intervals) representing score ranges.
2. **Y-Axis (Number of Students)**: Shows how many students fall into each score range.
3. **Bar Heights**: Indicate the frequency of scores within each bin.
4. **Shape of Distribution**: The histogram helps identify patterns such as normal distribution, skewness, or outliers.

b. Density Plot: A density plot, also known as a KDE plot, is a smooth representation of the distribution of one continuous numerical variable. It is like a histogram, but instead of discrete bars, it will lay out a continuous curve. It helps to understand the shape, spread, and peaks of data distribution.

Example: Imagine we have a dataset containing the scores of 300 students in a mathematics exam (as shown in fig.2.21) Instead of using a histogram, we can use a density plot to visualize the distribution of scores smoothly.

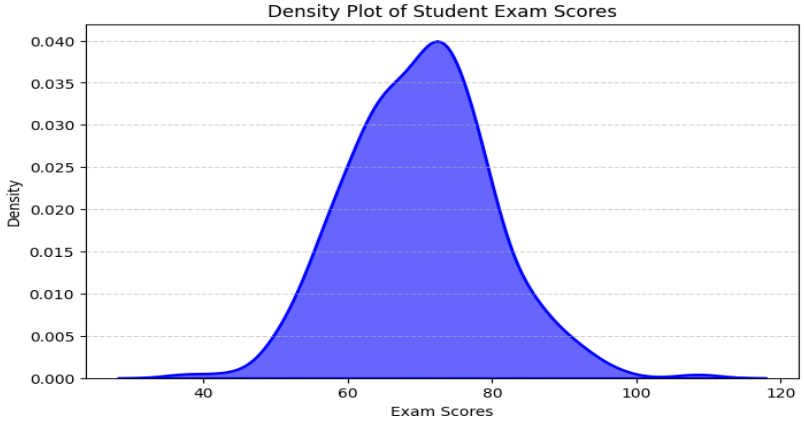


Fig. 2.21: Density Plot

1. The peak (mode) of the curve shows the most common score range.
2. The spread of the curve indicates the variability in scores.
3. The presence of multiple peaks could indicate different groups in the data (e.g., high and low performers).

2.1 Visualizing Amounts

Bar charts, pie charts, and stacked area charts are three common forms of representing amounts in data visualizations. They help compare quantities, proportions, and trends over time.

a. Bar Charts: A bar chart is one of the best methods to visualize amounts. It uses rectangular bars to represent categories, whose height (for a vertical bar chart) or length (for a horizontal bar chart) relates to the amount or frequency of that data.

Example: Monthly Sales Performance of a Retail Store

A retail store tracks its monthly sales revenue for the past six months to analyze trends and performance. The store's sales data is:

Month	Sales Revenue (in $)
January	25,000
February	30,000
March	28,000
April	35,000
May	40,000
June	38,000

Code:

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
# Generate a bar graph for Monthly Sales Performance of a Retail Store
# Sample data
months = ["January", "February", "March", "April", "May", "June"]
sales_revenue = [25000, 30000, 28000, 35000, 40000, 38000]  # Sales revenue in dollars
# Create the bar chart
plt.figure(figsize=(8, 5))
plt.bar(months, sales_revenue, color='blue')
# Labels and title
plt.xlabel("Months")
plt.ylabel("Sales Revenue ($)")
plt.title("Monthly Sales Performance of Retail Store")
# Show values on top of bars
for i, value in enumerate(sales_revenue):
    plt.text(i, value + 500, str(value), ha='center', fontsize=10)
# Show grid and display plot
plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.show()
```

Output of the code is shown in fig 2.22

Output:

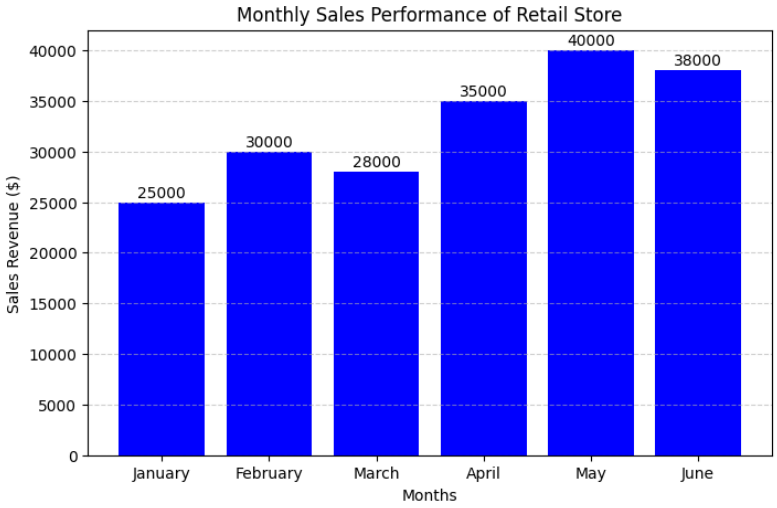


Fig. 2.22: Bar Graph

- The X-axis represents months, and the Y-axis represents sales revenue.
- Each bar's height shows the total sales amount for that month.
- The store owner can easily compare which months had the highest and lowest sales.
- The trend shows that sales increased over time, with a peak in May.

b. Pie Charts: A pie chart is a circular graph that has a number of slices to show portions of the whole. Each slice signifies a category's contribution to the whole. The bigger the slice, the greater the slice count towards that category.

Example: A company earns revenue from different sources:

Source	Revenue (%)
Product Sales	50%
Subscriptions	20%
Advertisements	15%
Partnerships	10%
Other	5%

Code:

```
# Data for company revenue sources
revenue_sources = ["Product Sales", "Subscriptions", "Advertisements", "Partnerships", "Other"]
revenue_percentages = [50, 20, 15, 10, 5]  # Percentage contribution
# Create a pie chart
plt.figure(figsize=(7, 7))
plt.pie(revenue_percentages, labels=revenue_sources, autopct='%1.1f%%', colors=['blue', 'green', 'red', 'purple', 'orange'], startangle=140)
# Title
plt.title("Company Revenue Sources Distribution")
# Display the pie chart
plt.show()
```

Output of the code is shown in fig 2.23

Output:

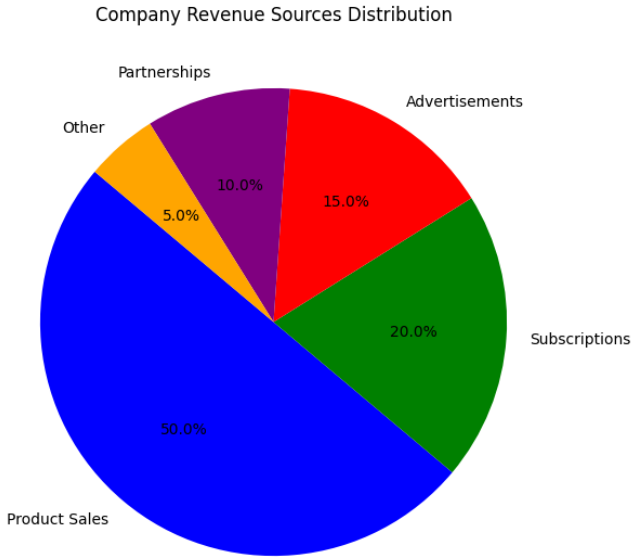


Fig. 2.23: Pie Chart

Here is the pie chart representing the Company Revenue Sources Distribution. Each slice shows the percentage contribution of different revenue sources. Product Sales (50%) is the largest segment, followed by Subscriptions (20%), Advertisements (15%), Partnerships (10%), and Other (5%).

c. **Stacked Area Charts:** An extended line chart where there are multiple data series stacked vertically upon one other and filling the spaces between with varying colors. This chart is used to show very clearly how much they accumulate over time and can indicate how different categories contribute to that total.

Example: A website tracks its traffic sources over six months:

Month	Organic	Paid Ads	Social Media	Direct	Referral
Jan	2000	1000	500	1500	800
Feb	2500	1200	600	1600	900
Mar	2700	1300	700	1800	1000
Apr	3000	1500	800	2000	1100
May	3200	1600	900	2200	1200
June	3500	1700	1000	2500	1300

Code :

```
months = ["Jan", "Feb", "Mar", "Apr", "May", "June"]
organic = [2000, 2500, 2700, 3000, 3200, 3500]
paid_ads = [1000, 1200, 1300, 1500, 1600, 1700]
social_media = [500, 600, 700, 800, 900, 1000]
direct = [1500, 1600, 1800, 2000, 2200, 2500]
referral = [800, 900, 1000, 1100, 1200, 1300]
# Create stacked area chart
plt.figure(figsize=(8, 6))
plt.stackplot(months, organic, paid_ads, social_media, direct, referral,
        labels=["Organic", "Paid Ads", "Social Media", "Direct", "Referral"],
```

```
        colors=["blue", "green", "red", "purple", "orange"], alpha=0.7)
# Labels and title
plt.xlabel("Months")
plt.ylabel("Website Traffic (Visitors)")
plt.title("Website Traffic Sources Over 6 Months")
plt.legend(loc="upper left")
# Display the stacked area chart
plt.show()
```

Output of the code is shown in fig 2.24

Output:

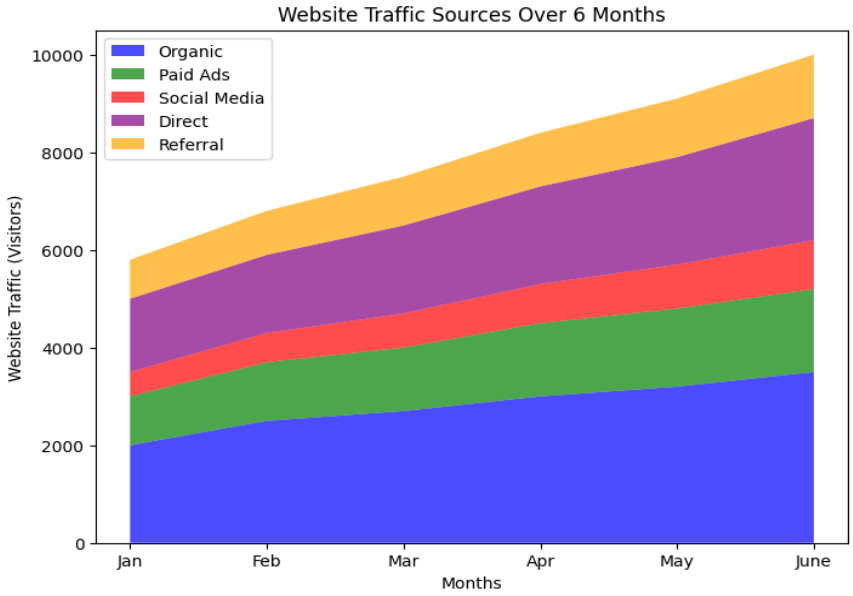


Fig. 2.24: Stacked Area Chart

2.3 Visualizing Distributions:

A distribution represents how values in a dataset are spread across a range. It helps in understanding patterns like central tendency, spread, and skewness of data. Two common methods to visualize distributions are [20]:

a. Histograms: A histogram is a bar graph of the frequency of a particular set of data contained in an interval called bins. A histogram differs from a bar chart in that it is always used for continuous numerical data, and bar charts are used for categorical data.

Example: A teacher collects math test scores from 53 students and makes a study of how the scores are distributed. The data is administratively grouped with the purpose of determining how many students fall into any one score range.

Score Range	Number of Students
40-50	3
50-60	8
60-70	12
70-80	15
80-90	10
90-100	5

Code:

```
# Data for exam scores distribution
score_ranges = ["40-50", "50-60", "60-70", "70-80", "80-90", "90-100"]
student_counts = [3, 8, 12, 15, 10, 5]
# Define bin edges to match the score ranges
bins = [40, 50, 60, 70, 80, 90, 100]
# Generate sample data points within each range
exam_scores=np.concatenate([np.random.randint(bins[i], bins[i+1], student_counts[i]) for i in range(len(bins)-1)])
# Create histogram
plt.figure(figsize=(8, 6))
plt.hist(exam_scores, bins=bins, color='blue', edgecolor='black', alpha=0.7)
# Labels and title
plt.xlabel("Exam Score Ranges")
plt.ylabel("Number of Students")
plt.title("Distribution of Exam Scores")
# Show grid
plt.grid(axis='y', linestyle='--', alpha=0.6)
# Display the histogram
plt.show()
```

Output of the code is shown in fig 2.25

Output:

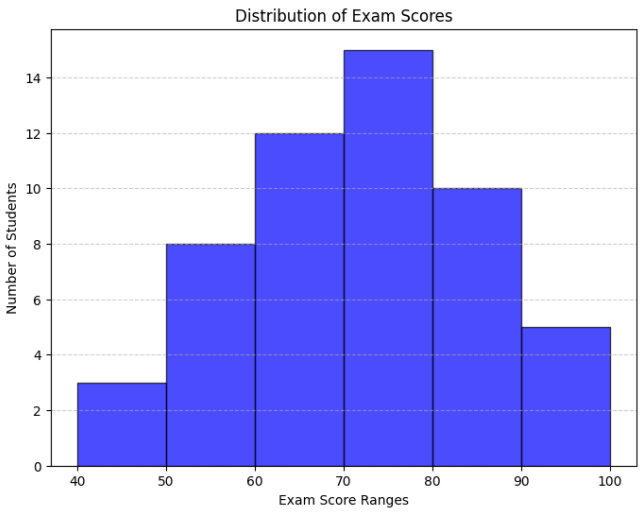


Fig. 2.25: Histogram

Here is the **histogram** representing the **exam scores distribution** of students.

- The **X-axis** represents the **score ranges** (40-50, 50-60, etc.).
- The **Y-axis** shows the **number of students** in each score range.
- The highest number of students scored between **70-80**, followed by **60-70** and **80-90**.
- The histogram helps identify the most common score range and overall data spread.

b. Density Plots: Density plots are the alternate smoothed version of the histograms and display the probability density function of the data, assigning higher probability densities to regions with higher frequency of observations. Such are used with the aid of kernel density estimation to better visualize the trend in the data.

Example: This company measures the refined outcome every day spent by a visitor on their website. A density plot would help give a fair visualisation of the most generally occurring time ranges (e.g., most users spend time between 2-5 min).

Code:

```
# Generate sample data: user session durations in minutes
np.random.seed(42)  # For reproducibility
session_durations = np.concatenate([
 np.random.normal(3, 1, 300),  # Most users stay around 3 minutes
 np.random.normal(7, 2, 150),  # Some users stay longer around 7 minutes
 np.random.normal(12, 3, 50)   # A few users stay much longer around 12 minute])
# Ensure no negative session durations
session_durations = session_durations[session_durations > 0]
# Create density plot
plt.figure(figsize=(8, 6))
sns.kdeplot(session_durations, fill=True, color="blue", alpha=0.6)
# Labels and title
plt.xlabel("Session Duration (Minutes)")
plt.ylabel("Density")
plt.title("Density Plot of Website Session Durations")
# Show grid
plt.grid(axis='y', linestyle='--', alpha=0.6)
# Display the density plot
plt.show()
```

Output of the code is shown in fig 2.26

Output:

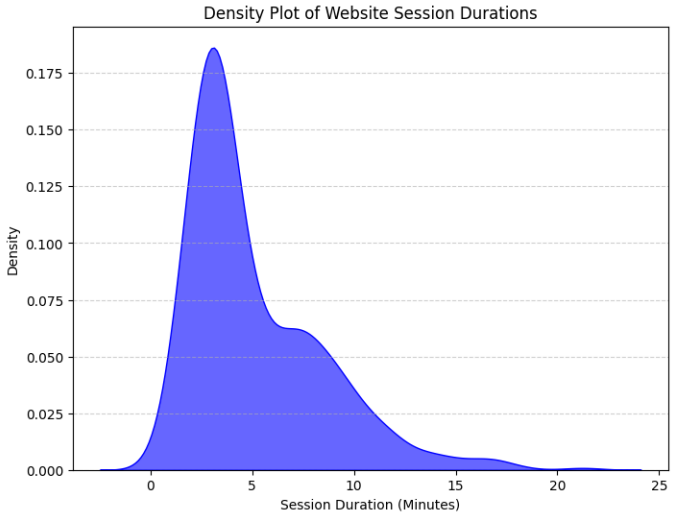


Fig. 2.26: Density Plot

a. The highest peak is around 3 minutes, meaning most users stay on the site for this duration.
b. A secondary peak of around 7 minutes suggests another group of users spends a bit more time.
c. A smaller peak of around 12 minutes indicates that a few users stay significantly longer.
d. The density gradually decreases after 15 minutes, showing that very few users stay beyond that.

2.4 Visualizing Propositions:

Visualizing propositions refers to the use of visual elements like graphs, charts or diagrams to present and convey some propositions, which can be statements that can be true or false. Instead of using complex computations and raw text, visualization techniques make it easier to understand logical structure, flow of the data and decision-making processes. several fields like data science, artificial intelligence, philosophy, mathematics, and business intelligence for an easier representation of logical rules, conditions, and dependencies [20].

1. Propositions and Logical Statements

To ensure data integrity, improve interpretability, and structure visual insights, propositions and logical statements are essential components of data visualisation. These components support the development of rules, the production of conclusions from graphical representations, and the validation of those findings.

a. Propositions: A proposition is a statement that is either true (T) or false (F) but not both. In data visualization, propositions are used to describe data patterns, relationships, and trends.

Propositions are declarative statements that have a particular truth value, which means that they are either true (T) or false (F), but not both at the same time. In fields like data visualisation, artificial intelligence, machine learning, and programming, propositions are essential components of logical reasoning, mathematical proofs, and computational logic.

The fundamental logical relations and norms can be represented by manipulating and combining proposition operators such as conjunction (∧), disjunction (∨), negation ¬, implication →, and biconditional ↔.

Examples of proposition in Data Visualization:

1. First proposition: "Product A's sales grew in Q4." (Verifiable with a line or bar chart).
2. The second proposition states that "country X has a larger population than country Y." (A choropleth map can be used for testing).
3. Proposition 3: "Missing values are present in the dataset." (Data profiling tools can be used for validation).
4. A statement is not a proposition if it cannot be given a clear truth value.

Logical Connectives in Data Visualization

Logical connectives are used to combine multiple propositions and create complex statements that influence decision-making in data visualization.

Table 2.1: Logical Connectivities

Logical Operator	Symbol	Meaning	Example in Data Visualization
Negation	¬P	NOT	"It is NOT true that sales increased in Q4."
Conjunction	P ∧ Q	AND	"Sales increased AND profits rose."
Disjunction	P ∨ Q	OR	"Sales increased OR marketing expenses decreased."
Implication	P → Q	if…then	"If website traffic increases, then sales increase."
Biconditional	P ↔ Q	if and only if	"Sales increase if and only if customer engagement increases."

Propositional logic's logical operators have a number of significant characteristics.

1. P ∧ Q ≡ Q ∧ P P ∨ Q ≡ Q ∨ P is commutative.
2. The second is associativity: (P ∧ Q) ∧ R ≡ P ∧ (Q ∧ R) (P ∨ Q) ∨ R ≡ P ∨ (Q ∨ R)
3. P ∧ (Q ∨ R) ≡ (P ∧ Q) ∨ (P ∧ R) is the distributivity.
4. Identity: P ∧ false ≡ P P ∧ true ≡ P
5. Domination: P ∧ false ≡ false P ∨ true ≡ true
6. Double Negation: P ≡ ¬ (¬P)
7. P ∧ P ≡ P P ∨ P ≡ P is Idempotence

Examples of Propositional Logic in Data Visualizations

1. Conditional Data Filtering & Highlighting

Use Case: Highlighting data points based on logical conditions.

- Example Proposition:
  "If sales are above $10,000, then mark the region as profitable."
  Logical Form: Sales > 10000 → Profitable

1. Import Required Libraries

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

2. Create Sample Data

```
data = {"Region": ["North", "South", "East", "West"],
    "Sales": [12000, 8000, 15000, 5000]}
df = pd.DataFrame(data)
```

Region	Sales
North	12000
South	8000
East	15000
West	5000

3. Apply Logical Condition for Coloring

```
colors = ['green' if sales > 10000 else 'red' for sales in df["Sales"]]
```

Color Assignment Example

Region	Sales	Color
North	12000	Green
South	8000	Red
East	15000	Green
West	5000	Red

4. Create and Customize the Bar Chart

```
plt.bar(df["Region"], df["Sales"], color=colors)
plt.xlabel("Region")
plt.ylabel("Sales")
plt.title("Sales Performance by Region")
plt.show()
```

Output of the code is shown in fig 2.27

Output:

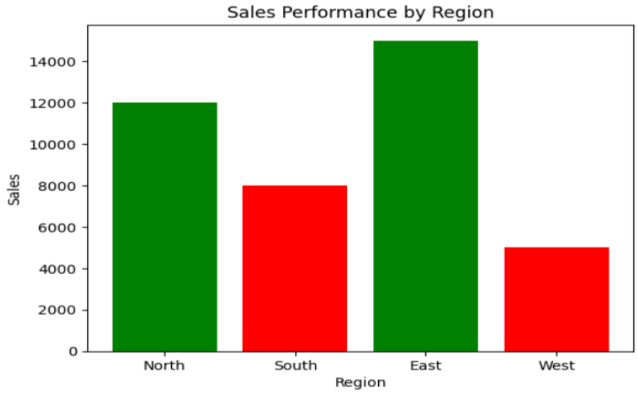


Fig. 2.27: Bar Chart

2. Logical Grouping in Clustering & Classification

Use Case: Clustering data into meaningful groups using logical statements.

- Example Proposition:

  "If a student scores above 90 in Math and Science, then they are in the 'Top Performer' category."

  Logical Form: (Math > 90 ∧ Science > 90) → Top Performer

  1. IMPORT REQUIRED LIBRARIES

  import seaborn as sns

  import pandas as pd

  import matplotlib.pyplot as plt

  2. CREATE SAMPLE DATA

  df = pd.DataFrame({ "Math": [85, 92, 78, 95, 88],"Science": [80, 95, 85, 97, 90], "Category": ["Average", "Top", "Average", "Top", "Average"]})

  3. CREATE SCATTER PLOT WITH LOGICAL GROUPING

  sns.scatterplot(x=df["Math"], y=df["Science"], hue=df["Category"], style=df["Category"], s=100)

  4. CUSTOMIZE THE CHART WITH LABELS & TITLES

  plt.xlabel("Math Score")

  plt.ylabel("Science Score")

  plt.title("Student Performance Clustering")

  plt.show()

  Output of the code is shown in fig 2.28

  Output:

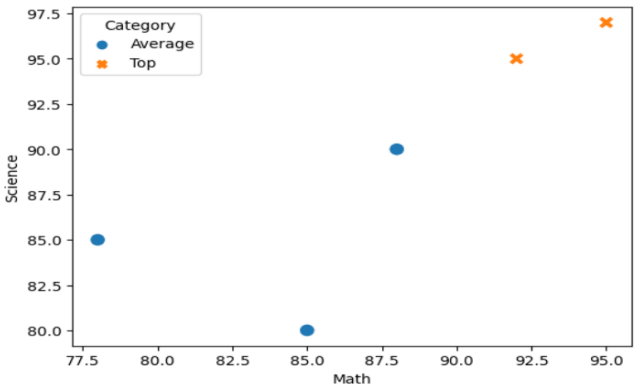


Fig. 2.28: Scatter Plot

b. Logical Statements

Logical statements are conditional words that affect the processing, classification, filtering, and display of data in data visualisation. By highlighting connections, patterns, and anomalies, these statements contribute to the increased significance of visualisations.

The following are common foundations for logical statements:

- Thresholds (e.g., indicating high vs. low numbers)
- Comparisons (e.g., condition-based data grouping)
- Boolean logic, such as colour coding using if-else situations
- Filtering (displaying only important information)

Examples of Logical Statements in Data Visualizations

Logical statements in data visualization are used to filter, categorize, highlight, or modify data presentation based on conditions. Here are some practical examples using Python and libraries like Matplotlib, Seaborn, and Pandas.

1. Conditional Formatting (Color Coding a Bar Chart)

Example: Highlighting sales performance based on a threshold.
Use Case: Differentiate between high and low-performing regions

```
import pandas as pd
import matplotlib.pyplot as plt
# Sample Data
data = {"Region": ["North", "South", "East", "West"],  "Sales": [12000, 8000, 15000, 5000]}
df = pd.DataFrame(data)
# Apply logical condition for coloring
colors = ['green' if sales > 10000 else 'red' for sales in df["Sales"]]
# Create Bar Chart
plt.bar(df["Region"], df["Sales"], color=colors)
plt.xlabel("Region")
plt.ylabel("Sales")
plt.title("Sales Performance by Region")
plt.show()
```

Output of the code is shown in fig 2.29

Output:

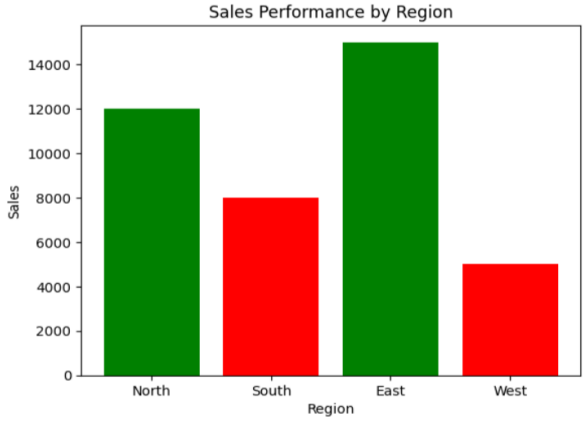


Fig. 2.29: Color Coding Bar Graph

2. Filtering Data in Visualizations

Example: Showing only high-sales regions

Use Case: Displaying only the top-performing regions

```
# Filter regions where sales > 10,000
high_sales_df = df[df["Sales"] > 10000]
# Create Bar Chart for high-sales regions
plt.bar(high_sales_df["Region"], high_sales_df["Sales"], color="blue")
plt.xlabel("Region")
plt.ylabel("Sales")
plt.title("High Sales Regions")
plt.show()
```

Output of the code is shown in fig 2.30

Output:

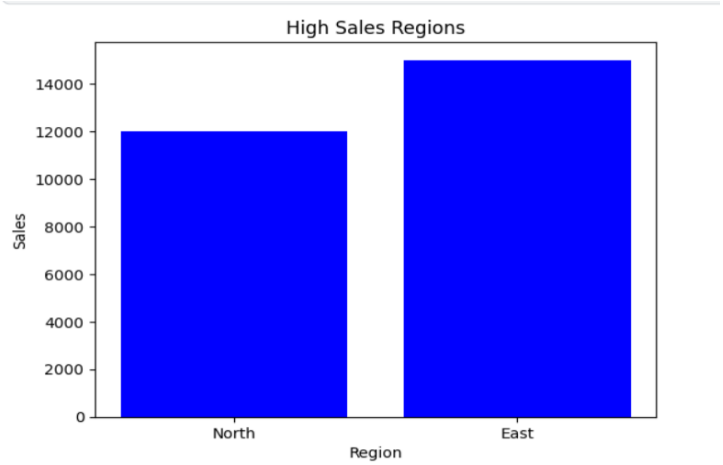


Fig. 2.30: Data Filteration

3. Venn Diagrams & Euler Diagrams

Venn Diagram:

A Venn diagram uses overlapping circles or other shapes to illustrate the logical relationships between two or more sets of items. They are commonly used to graphically organize or compare items involving two or three sets of a few elements or 3D presentations.

When to use a Venn diagram?

a. **To visualize the relationship between sets of items, such as commonalities and differences**

In the Kaggle notebook, we generated a Venn diagram to compare the features of three programming languages: Python, Java, and C++. The diagram visually represents the commonalities and differences between them.

1. **Library Installation & Import:**

```
!pip install matplotlib-venn
# Import required libraries
import matplotlib.pyplot as plt
from matplotlib_venn import venn3
```

- The matplotlib-venn package is installed and imported along with matplotlib.pyplot.
- This allows us to create and display the Venn diagram.

2. **Defining Sets of Features:**

```
python_features = {"OOP", "Dynamic Typing", "Interpreted", "Libraries"}
java_features = {"OOP", "Strong Typing", "JVM", "Enterprise"}
cpp_features = {"OOP", "Strong Typing", "Compiled", "Performance"}
```

- We define sets of characteristics unique to Python, Java, and C++.
- These sets contain elements that represent key properties of each language.

3. **Creating the Venn Diagram:**

```
plt.figure(figsize=(6,6))
venn = venn3([python_features, java_features, cpp_features],('Python', 'Java', 'C++'))
```

- Using the venn3() function, we plot the intersection and differences of the three languages.
- Overlapping areas indicate shared features, while non-overlapping sections represent unique features.

4. **Displaying the Diagram:**

```
# Add a title
plt.title("Feature Comparison of Python, Java, and C++")
plt.show()
```

- plt.show() is used to render the diagram.

Output of the code is shown in fig 2.31

Output:

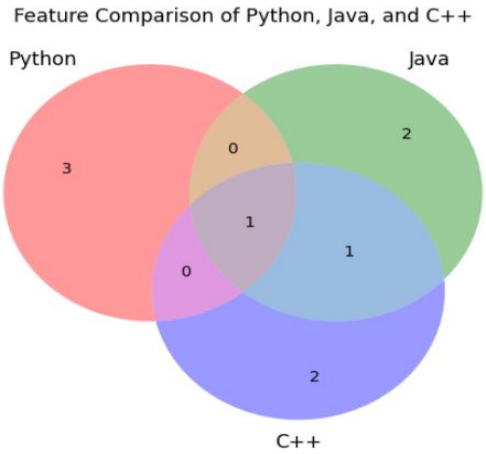


Fig 2.31: Venn Diagram

b. Comparing two or more choices and predicting probabilities

A Venn diagram is often used in predicting various occurrences and statistics, and several models for analysis of databases while these Venn diagrams can be used to a certain extent, they can however subject to several levels of degree commonalities and differences between data sets. They give various kinds of corrections toward predictions of probabilities of certain occurrences. This aspect gives predictive analytics to be performed over the datasets. Venn diagrams have been used to study the commonalities and differences among languages [21].

The Venn diagram visually represents the overlap between Apple devices (iPhone, iPad, iPod) and Samsung mobile phones based on shared features as shown in fig 2.32. The intersections highlight common functionalities, while non-overlapping areas show unique characteristics of each device.

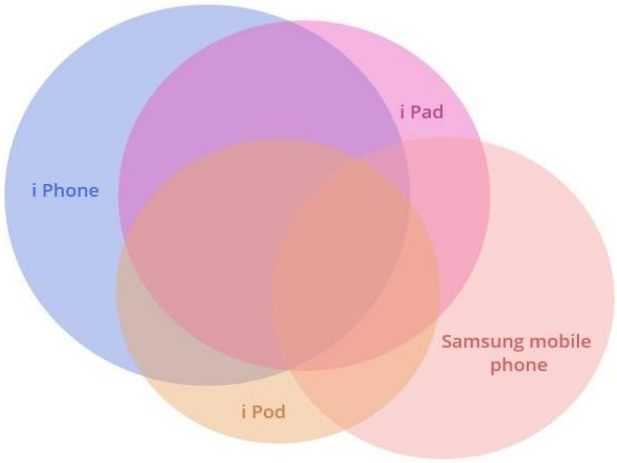


Fig. 2.32: Comparing two or more choices

c. To reason through the logic

Venn diagrams are often used to establish the validity of particular arguments and conclusions. In deductive reasoning, if the premises are true and the argument form is valid, the conclusion must also be true. The graphs provide an intuitive understanding for reasoning through the logic governing sentences or statements, such as knowing the Boolean logic behind words with "or" and "and" or having to do a word search.

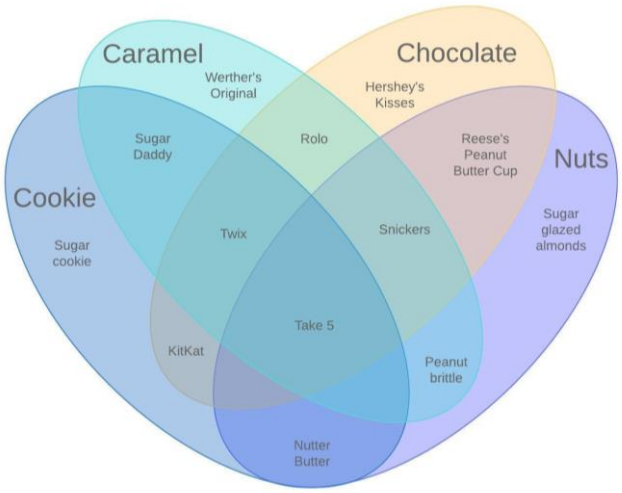


Fig. 2.33: Venn Diagram using logic

This Venn diagram categorizes different sweets based on their ingredients: Caramel, Chocolate, Cookie, and Nuts as shown in fig 2.33. The overlapping areas show candies that share multiple ingredients, with "Take 5" at the center, containing all four.

Euler Diagrams

An Euler diagram is another diagram that represents sets and their relationships. It's like a Venn diagram as both use circles to create the diagram. However, while a Venn diagram represents an entire set, an Euler diagram represents a part of a set. A Venn diagram shows an empty set by shading it out, whereas in an Euler diagram that area could simply be missing altogether [21].

In the Kaggle notebook, we have generated a Euler diagram to compare the three sets. The diagram visually represents the commonalities and differences between them.

1. Importing Required Libraries

```
import matplotlib.pyplot as plt
from matplotlib_venn import venn3
```

- matplotlib.pyplot is used for visualization and rendering the diagram.
- venn3 from matplotlib_venn helps us create a Venn diagram with three sets.

2. Defining the Sets and Their Intersections

The venn3 function takes a dictionary that specifies the sizes of the sets and their overlaps.

```
venn = venn3(subsets={'100': 10, '010': 8, '001': 6,  '110': 4, '101': 3, '011': 2, '111': 1},
set_labels=('A', 'B', 'C'))
```

Each key in the dictionary represents a combination of set membership, where:

- '100' means only Set A (10 elements).
- '010' means only Set B (8 elements).
- '001' means only Set C (6 elements).
- '110' represents the intersection of Set A and Set B (4 elements).
- '101' represents the intersection of Set A and Set C (3 elements).
- '011' represents the intersection of Set B and Set C (2 elements).
- '111' represents the intersection of all three sets (1 element).

3. Adding a Title to the Diagram

```
plt.title("Euler Diagram Approximation with 3 Sets")
```

4. Displaying the Diagram

```
plt.show()
```

Output of the code is shown in fig 2.34

Output:

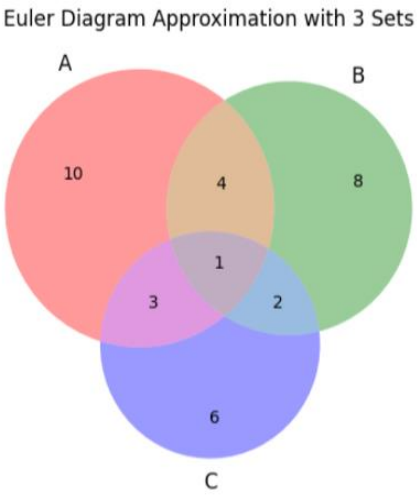


Fig. 2.34: Euler Diagram

This program uses the matplotlib_venn library to create a Venn diagram, which is a simplified form of an Euler diagram, visualizing the relationships between three sets.

Venn diagram VS Euler diagram

Table 2.2: Difference between Venn Diagram and Euler Diagram

Feature	Venn Diagram	Euler Diagram
Overlap	Shows all possible overlaps between sets with shared regions.	May not show all overlaps, focusing only on relevant ones.
Completeness	Displays all possible relationships between sets, even if some are empty.	Can be partial, showing only relevant relationships without all intersections.
Expressiveness	Limited to basic set operations like union, intersection, and difference	More expressive, able to represent complex relationships and dependencies.
Complexity	Becomes cluttered and harder to interpret with more than three sets.	Handles more sets and complex relationships in a clearer, often simpler way.

2.5 Truth Table

A truth table is a table that shows all the possible combinations of truth values (true or false) for a set of statements or propositions. Each row of the table represents a different scenario, and each column represents a different statement or proposition. The table also shows the truth value of a compound statement or proposition that is formed by combining the statements or propositions with logical operators, such as and, or, not, if-then, and if and only if. For example, the following table shows the truth values of the statements p, q, and p and q [22].

p	q	p and q
T	T	T
T	F	F
F	T	F
F	F	F

2.5.1 Truth Tables in Data Visualization:

A truth table is defined as a mathematical table used to fare the truth implies of logical propositions depending on variable values. Truth tables can peek into data visualization in the following ways:

To Visualize in Binary: Values setting to true or false on all combinations of logical propositions in a table/grids format: these could include AND, OR, or NOT.

Interactive Diagrams: Frameworks that represent truth values set up Portuguese dynamials are executionable and visible to the user through both screen interfaces, displaying the user-set logical outcome.

Venn Diagrams: These could stand for truths that show Venn's diagrams where intersection across the set could imply logical relationships of conjunction (AND), disjunction (OR), or negation (NOT).

In this notebook, we will examine Boolean logic operations through a truth table and represent it visually with a heatmap.

1. Importing Required Libraries

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

- pandas is used for creating and managing tabular data (DataFrames).
- seaborn is a visualization library that helps in creating aesthetically pleasing plots.
- matplotlib.pyplot is used for rendering the plots.

2. Defining Boolean Values for A and B

We define the possible truth values for two Boolean variables, A and B:

A = [True, True, False, False]

B = [True, False, True, False]

Since Boolean values can be either True (1) or False (0), we enumerate all possible combinations for two variables.

3. Constructing the Truth Table

```
# Create a DataFrame for the truth table
truth_table = pd.DataFrame({
   'A': A,
   'B': B,
   'A AND B': [a and b for a, b in zip(A, B)],
   'A OR B': [a or b for a, b in zip(A, B)],
   'NOT A': [not a for a in A],
   'NOT B': [not b for b in B]
})
```

- A AND B: Logical AND operation (True only if both A and B are True).
- A OR B: Logical OR operation (True if either A or B is True).
- NOT A: Negation of A (inverts the value of A).
- NOT B: Negation of B (inverts the value of B).

4. Displaying the Truth Table

```
print(truth_table)
```

5. Visualizing the Truth Table with a Heatmap:

```
plt.figure(figsize=(8, 4))
sns.heatmap(truth_table[['A AND B', 'A OR B', 'NOT A', 'NOT B']].astype(int),
 annot=True, cmap='Blues', cbar=False)
plt.title('Truth Table Visualized as a Heatmap')
plt.show()
```

- plt.figure(figsize=(8, 4)): Sets the figure size for better readability.
- truth_table[['A AND B', 'A OR B', 'NOT A', 'NOT B']].astype(int): Converts Boolean values (True/False) into integers (1/0) for visualization.
- sns.heatmap(..., annot=True, cmap='Blues', cbar=False):
    - annot=True displays values inside the heatmap.
    - cmap='Blues' assigns a blue color gradient.
    - cbar=False removes the color bar for simplicity.

Output of the code is shown in fig 2.35

Output:

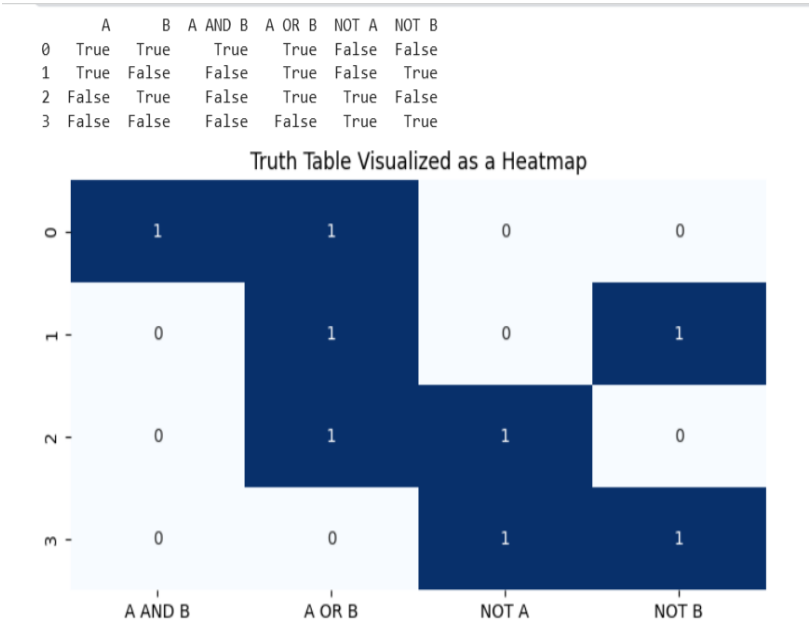


Fig. 2.35: Heatmap

2.6 Argument mapping:

Argument mapping is an analytical and visual process that allows individuals to break down complex arguments into their constituent parts. Argument maps are a type of data visualization used to represent the structure of logical reasoning, debates, or discussions. They help break down complex arguments into a visual format, showing how different premises, conclusions, and objections relate to one another [23].

Basic Techniques for Argument Mapping:

8. Box and Arrow Diagrams: Use boxes to denote various components of an argument, such as claims and premises, and arrows to indicate logical flow or dependency.
9. Color Coding: Use different colors for claims, premises, and objections to visually differentiate them.
10. Labeling: Label each component as P1 for the first premise, C for the main claim, and O for objections.

In a Kaggle Notebook, an argument map is a graphical illustration of first-order reasoning using NetworkX and Matplotlib. By connecting the main proposition that "we should eat healthy" with supporting arguments, it helps to take some of the heat out of complex reasoning.

1. Importing Required Libraries

import networkx as nx

import matplotlib.pyplot as plt

- networkx is a Python library used for graph creation and analysis.
- matplotlib.pyplot is used for visualizing the graph.

2. Creating a Directed Graph

G = nx.DiGraph()

- DiGraph() creates a **directed graph**, meaning that edges have a specific direction.

3. Adding Nodes (Claims and Supporting Arguments)

```
G.add_node("We should eat healthy")  # Main claim
G.add_node("Healthy food improves energy")  # Supporting reason 1
G.add_node("Healthy food prevents diseases")  # Supporting reason 2
```

4. ADDING EDGES (LOGICAL CONNECTIONS)

```
G.add_edge("We should eat healthy", "Healthy food improves energy")
G.add_edge("We should eat healthy", "Healthy food prevents diseases")
```

5. VISUALIZING THE ARGUMENT MAP

```
plt.figure(figsize=(6, 4))
pos = nx.spring_layout(G, seed=42)  # Layout for better visualization
nx.draw(G, pos, with_labels=True, node_size=2000, node_color="lightgreen",
 edge_color="black", font_size=10, font_weight="bold", arrows=True)
plt.title("Simple Argument Map")
plt.show()
```

Output of the code is shown in fig 2.36

Output:

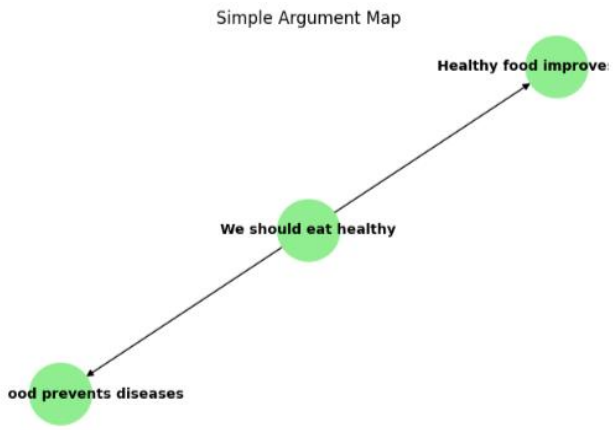


Fig. 2.36: Simple Argument Map

2.7 Casual Diagram

A causal diagram is a graphical representation of a data generating process. Causal diagram consists of nodes and arrows. Nodes represent variables taken into consideration; arrows represent the immediate cause and effect among nodes. It helps illustrate how different factors influence each other in a system, often using Directed Acyclic Graphs.

This code below creates a causal diagram using NetworkX in Kaggle, representing the relationships between exercise, a healthy diet, weight control, and overall health.

1. Importing Required Libraries

```
import networkx as nx
import matplotlib.pyplot as plt
```

2. Creating a Directed Graph

```
G = nx.DiGraph()
```

3. Adding Nodes

```
G.add_node("Exercise")
G.add_node("Healthy Diet")
G.add_node("Better Health")
```

G.add_node("Weight Control")

4. Adding Edges (Causal Relationships)

G.add_edge("Exercise", "Better Health")
G.add_edge("Healthy Diet", "Better Health")
G.add_edge("Exercise", "Weight Control")
G.add_edge("Weight Control", "Better Health")

- These edges represent cause-and-effect relationships:
- "Exercise" → "Better Health" → Exercise directly improves health.
- "Healthy Diet" → "Better Health" → A good diet contributes to better health.
- "Exercise" → "Weight Control" → Exercise helps maintain weight.
- "Weight Control" → "Better Health" → Weight control further improves health.

5. Visualizing the Causal Diagram

plt.figure(figsize=(8, 6))
pos = nx.spring_layout(G, seed=42) # Layout for better visualization
nx.draw(G, pos, with_labels=True, node_size=3000, node_color="lightblue", edge_color="black", font_size=12, font_weight="bold", arrows=True, arrowsize=10)
plt.title("Causal Diagram: Factors Affecting Health")
plt.show()
Output of the code is shown in fig 2.37

Output:

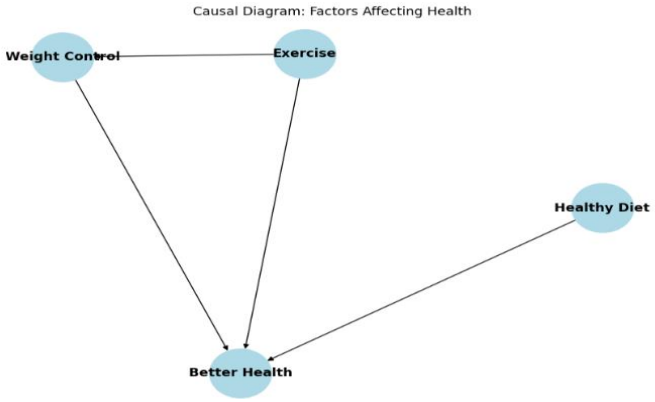


Fig. 2.37 Casual Diagram

2.8 Visualizing Associations (Two or More Quantitative Variables)

Many datasets comprise two or more quantitative variables, and we may be interested in their interrelationships. For example, consider one dataset comprising quantitative measurements of various animals, such as the animals' height, weight, length, and daily energy demand. To plot the relationship between two such variables, such as height and weight, we usually employ a scatterplot. If we want to display several variables at once, a bubble chart, a scatterplot matrix, or a correlogram may be appropriate. Finally, very high-dimensional datasets may sometimes be reduced in dimensionality through methods such as principal components analysis [21].

a. Scatter Plot

A scatter plot is a data visualization which displays at least two numerical variables for a study. Each point indicates an observation for the given dataset, based on the values for the two variables.

Use Cases of Scatter Plots:

- Understanding correlation (positive, negative, or no correlation).

- Spotting outliers in data.
- Identifying clusters or patterns in datasets.

Example: The given code is to use a scatter plot to show the link between sepal width and length in the Iris dataset. The dataset is first loaded from a CSV file, and then Seaborn and Matplotlib are used to build a scatter plot in kaggle notebook.

1. Importing Required Libraries

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

2. Loading the Dataset

```
file_path = '/kaggle/input/practica/iris.csv'  # Ensure the file path is correct
df = pd.read_csv(file_path)
```

3. Creating a Scatter Plot

```
plt.figure(figsize=(8, 5))
sns.scatterplot(data=df, x="sepal_length", y="sepal_width", hue="species", style="species", palette="deep")
```

- plt.figure(figsize=(8, 5)) → Sets the figure size to **8x5 inches** for better visibility.
- sns.scatterplot(...) → Creates a scatter plot using Seaborn:
  - data=df → Uses the **Iris dataset**.
  - x="sepal_length", y="sepal_width" → Plots **sepal length** (x-axis) against **sepal width** (y-axis).
  - hue="species" → Colors each point based on the **species** category.
  - style="species" → Uses different **marker styles** for each species.
  - palette="deep" → Uses a distinct color palette for better differentiation.

4. Adding Labels and Title

```
plt.xlabel("Sepal Length (cm)")
plt.ylabel("Sepal Width (cm)")
plt.title("Scatter Plot of Sepal Length vs Sepal Width (Iris Dataset)")
```

5. Displaying the Plot

```
plt.show()
```

Output of the code is shown in fig 2.38

Output:

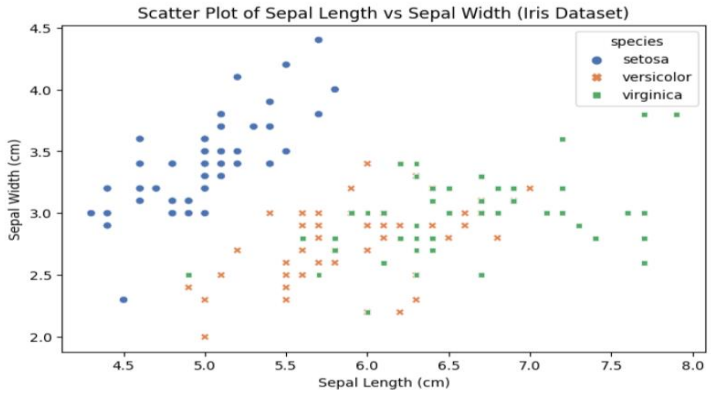


Fig. 2.38: Scatter Plot

The scatter plot visualizes the relationship between **sepal length** and **sepal width** in the **Iris dataset**, differentiating species using colors and markers.

b. Regression Line

A regression line is a line which is used to describe the behavior of a set of data. In other words, it gives the best trend of the given data. Regression lines are useful in forecasting procedures. Its purpose is to describe the interrelation of dependent variables with one or many independent variables such as X.

The equation derived from the regression line proves to be a guiding tool an analyst uses to forecast the dependent variable future behaviors using different independent variable values.

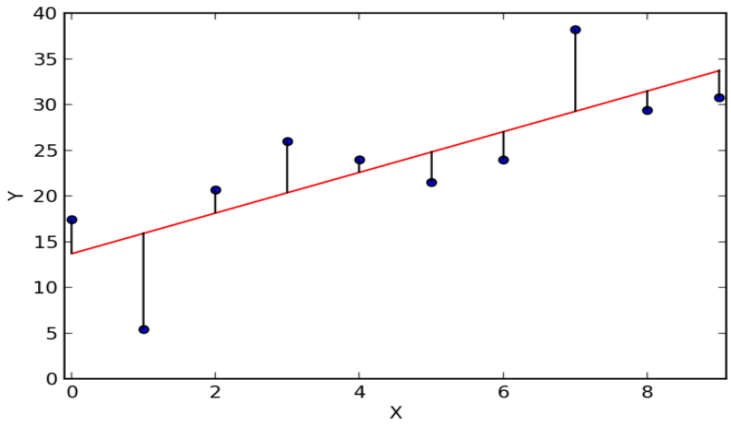


Fig. 2.39: Regression Line

Regression Line Formula: $y = a + bx + u$

Multiple Regression Line Formula: $y= a + b_1x_1 +b_2x_2 + b_3x_3 +\ldots+ b_tx_t + u$

Regression analysis is the basic statistical technique for modeling the relationship between variables (as shown in fig 2.39). It tells you about trend predictions, understanding data behaviours, and assisted data-driven decision-making. Visualizations assume an important role in the early interpretation of regression analysis results. Graphical representation of regression models enables us to view the relationships in a much more simplified form.

Where is linear regression used?

Regression models are heavily relied upon in the fields of finance and business. Various financial analysts employ linear regressions to forecast stock prices and commodity prices and to value many other securities. Various firms employ regression analysis for forecasting sales, inventories, and many other variables.

1. Importing Required Libraries

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
```

- sklearn.linear_model.LinearRegression → Implements linear regression for fitting a straight line to the data.

2. Defining Sample Data

```
X = np.array([1, 2, 3, 4, 5, 6]).reshape(-1, 1)  # Independent variable
Y = np.array([2, 4, 5, 4, 5, 7])  # Dependent variable
```

- reshape (-1, 1) ensures X is a 2D array, which is required for Scikit-Learn models.

3. Fitting the Linear Regression Model

```
model = LinearRegression()
model.fit(X, Y)
Y_pred = model.predict(X)
```

- LinearRegression() initializes a linear regression model.
- fit(X, Y) trains the model to find the best-fitting line for the data.
- predict(X) generates predicted Y values (i.e., regression line values).

4. Plotting the Data and Regression Line

```
plt.scatter(X, Y, color='blue', label="Actual Data")
plt.plot(X, Y_pred, color='red', label="Regression Line")
```

- plt.scatter(X, Y, color='blue') → Plots the actual data points in blue.
- plt.plot(X, Y_pred, color='red') → Draws the best-fit regression line in red.

5. Adding Labels and Title

```
plt.xlabel("X")
plt.ylabel("Y")
plt.title("Regression Line Example")
plt.legend()
plt.show()
```

- **legend()** → Adds a legend to differentiate the actual data points and the regression line.

Output of the code is shown in fig 2.40

Output:

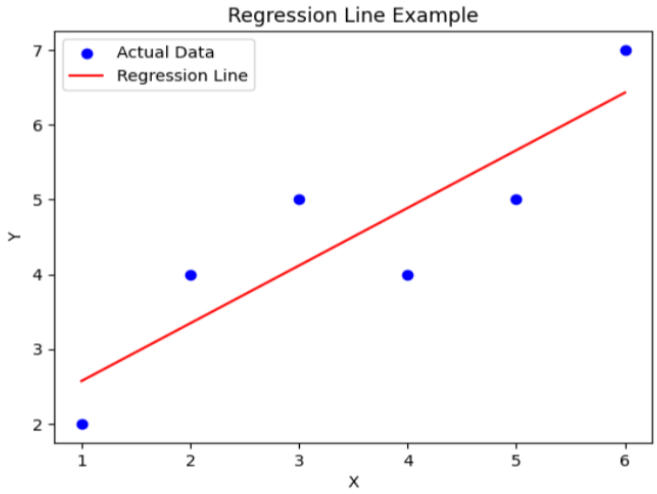


Fig. 2.40: Graph using Regression Line Model

c. Heatmaps & Correlation Matrices

A heatmap is a graphical representation of the correlation coefficients between two or more variables. The colors on a heatmap help to better understand the relationships and dependencies within a dataset strength and direction of correlations.

In a heatmap of correlation:

1. Positive Correlation:

- Positive Correlation occurs when an increase in one variable is associated with an increase in another variable.

- The correlation coefficient is positive and indicates a positive linear relationship.
- The value of the correlation coefficient runs from 0 to +1.
  Examples: Time spent studying and exam scores. Temperature and ice cream sales.

2. Negative Correlation:

- Negative Correlation occurrs when an increase in one variable results in a decrease in another variable.
- The correlation coefficient is negative and indicates a negative linear relationship.
- The value of the correlation coefficient runs from 0 to -1.
  Examples: Exercise and body weight or fuel efficiency and speed.

3. Zero Correlation:

- This refers to zero correlation or no correlation, indicating that there exists no linear relationship between two variables.
- The correlation coefficient is close to 0, indicating there is no systematic association.
  Examples: The height of a person and the number of books read per month may have zero correlation.

Libraries for Creating Heatmaps

Multiple Python libraries can generate heatmaps. Below, we discuss the most used:

Seaborn is a Python library built on matplotlib, enabling data visualization. A means of presenting data in a statistical graph format, an attractive and informative blend of storytelling. Two clusters of related data are delineated differently through a color palette; the method termed heat mapping is one of the supporting motifs invoked by seaborn. The central interest of this article lies in correlation heat-mapping with some tips on how, through seaborn along with pandas and matplotlib, you can do that for any dataframe.

Like any another Python libraries, seaborn can be easily installed using pip:

- pip install seaborn

This library is a part of Anaconda distribution and usually works just by import if Anaconda supports your IDE, but it can be installed too by the following command:

- conda install seaborn

Let's plot the heatmap for X, Y and Z using seaborn

1. Importing Required Libraries

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

2. Creating a Sample Dataset

```
data = {'X': [1.5, 2, 3.9], 'Y': [2.4, 4.5, 6.3], 'Z': [3.4, 2.6, 8.9]}
df = pd.DataFrame(data)
```

- A dictionary (data) is defined with three numerical variables (X, Y, and Z).
- pd.DataFrame(data) converts this dictionary into a pandas DataFrame (df).

3. Computing the Correlation Matrix

```
autocorrelation_matrix = df.corr()
```

- .corr() calculates the correlation coefficients between the variables in df.
- The result is a correlation matrix, where:

- Values range from -1 to 1.
- 1 means perfect positive correlation.
- 0 means no correlation.
- -1 means perfect negative correlation.

4. Creating a Heatmap for Visualization

sns.heatmap(autocorrelation_matrix, annot=True, cmap='coolwarm', vmin=-1, vmax=1)

- sns.heatmap(...) generates a heatmap of the correlation matrix.
  - annot=True → Displays correlation values inside the heatmap.
  - cmap='coolwarm' → Uses the Cool-Warm color scheme (blue for negative, red for positive correlations).
  - vmin=-1, vmax=1 → Defines the color scale from -1 to 1 for consistency.

5. Adding a Title and Displaying the Plot

plt.title('Autocorrelation Matrix Heatmap')

plt.show()

Output of the code is shown in fig 2.41

Output:

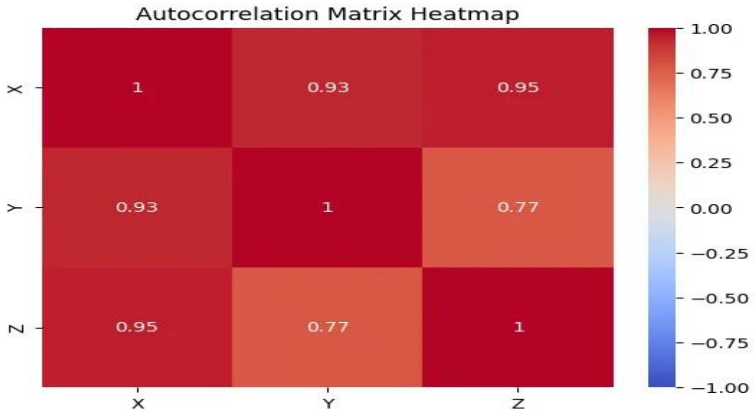


Fig. 2.41: Matrix Heatmap

d. Bubble Charts

A bubble chart is a powerful visualization tool that extends the capabilities of a standard scatter plot by incorporating additional dimensions of data. Here's a structured breakdown:

Core Components

1. **Axes**:
   - Horizontal (X-axis): Represents one numerical variable (e.g., GDP per capita).
   - Vertical (Y-axis): Represents a second numerical variable (e.g., Life Expectancy).
2. **Bubble Size**:
   - A third numerical variable is encoded in the size of each bubble (e.g., Population). Larger bubbles indicate higher values, allowing comparisons across three metrics simultaneously.
3. **Additional Dimensions (Optional)**:
   - Color: Can represent a fourth variable, either categorical (e.g., Region) or numerical (using gradients).

- Animation: Shows temporal changes, illustrating how data evolves over time (e.g., decades).

Example Use Cases of Bubble Charts

- Market Analysis: One can represent different companies according to revenue (X-axis), profit (Y-axis), and market share (bubble size).
- Population Studies: Showing the countries based on GDP (X-axis), literacy rate (Y-axis), population size (bubble size).
- Health Data: Showing the average BMI (X-axis), life expectancy (Y-axis), and pollution levels (bubble size) in different cities.
- Financial Analysis: Comparing the stock performance of price (X-axis), growth rate (Y-axis), and trading volume (bubble size).

Example: Let's Create a bubble chart using matplotlib.pyplot in a kaggle notebook using python.

1. Import the Required Library

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

2. Define the Data

```
products = ['A', 'B', 'C', 'D', 'E']
revenue = [50, 70, 90, 30, 60]  # X-axis (Revenue in $1000s)
profit = [10, 25, 30, 8, 15]  # Y-axis (Profit in $1000s)
sales_volume = [200, 450, 300, 150, 400]  # Bubble size (Units sold)
```

3. Scale the Bubble Size

```
bubble_size = [s * 2 for s in sales_volume]
```

- **Sales volume is scaled** by multiplying each value by 2.
- This ensures bubbles are **proportionate and visible** on the chart.

4. Create the Bubble Chart

```
plt.figure(figsize=(8, 5))
plt.scatter(revenue, profit, s=bubble_size, alpha=0.5,  c=['red', 'blue', 'green', 'purple', 'orange'], edgecolors="black")
```

- revenue → Plots revenue values on the **X-axis**
- profit → Plots profit values on the **Y-axis**.
- s=bubble_size → Sets **bubble sizes** based on sales volume.
- alpha=0.5 → Makes bubbles slightly transparent (so overlapping is visible).

5. Add Labels to Each Bubble

```
for i, product in enumerate(products):
   plt.text(revenue[i], profit[i], product, fontsize=12)
```

6. Add Axis Labels and Title

```
plt.xlabel("Revenue ($1000s)")
plt.ylabel("Profit ($1000s)")
plt.title("Bubble Chart: Revenue vs Profit vs Sales Volume")
```

7. Add Grid Lines

```
plt.grid(True, linestyle="--", alpha=0.5)
```

- Adds a **grid** with dashed ("--") lines and **50% transparency (alpha=0.5)** for readability.

8. Display the Chart

plt.show()

Output of the code is shown in fig 2.42

Output:

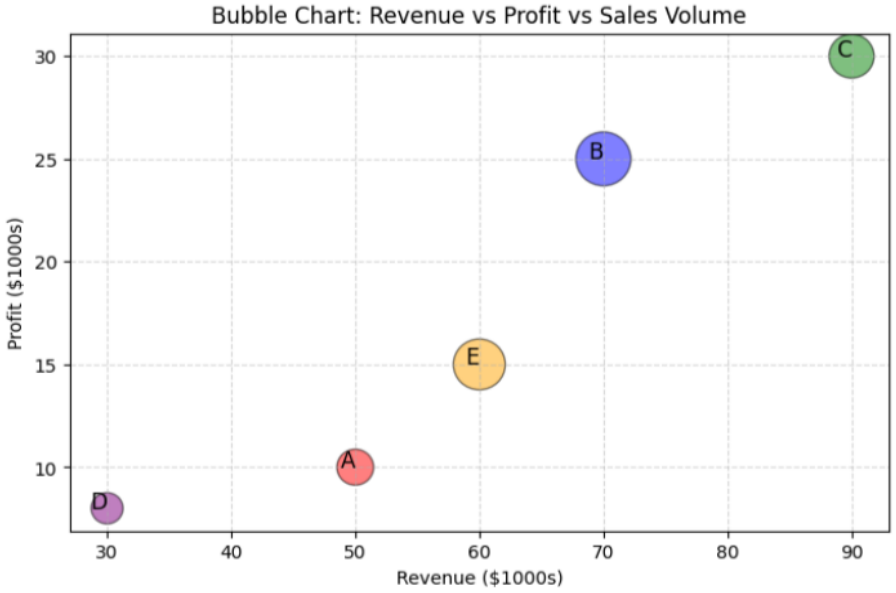


Fig. 2.42: Bubble Chart

e. Pair Plots

Pair Plots in Seaborn is a data analysis application developed from Matplotlib that uses its power by providing very beautiful plots exactly according to the analytical process.

Here's the breakdown of the Pair Plot code step by step:

1. Import Required Libraries

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

2. Create a Sample Dataset

```
data = {
   'GDP': [50000, 40000, 30000, 20000, 10000],
   'Life_Expectancy': [80, 78, 75, 70, 65],
   'Literacy_Rate': [99, 97, 95, 90, 85]
}
df = pd.DataFrame(data)
```

- A dictionary (data) is created with three numerical variables:
  - GDP (Gross Domestic Product per capita).
  - Life Expectancy (average lifespan in years).
  - Literacy Rate (percentage of people who can read and write).
- pd.DataFrame(data) converts the dictionary into a DataFrame, which is required for Seaborn.

3. Create the Pair Plot

```
sns.pairplot(df)
plt.show()
```

Output of the code is shown in fig 2.43

Output :

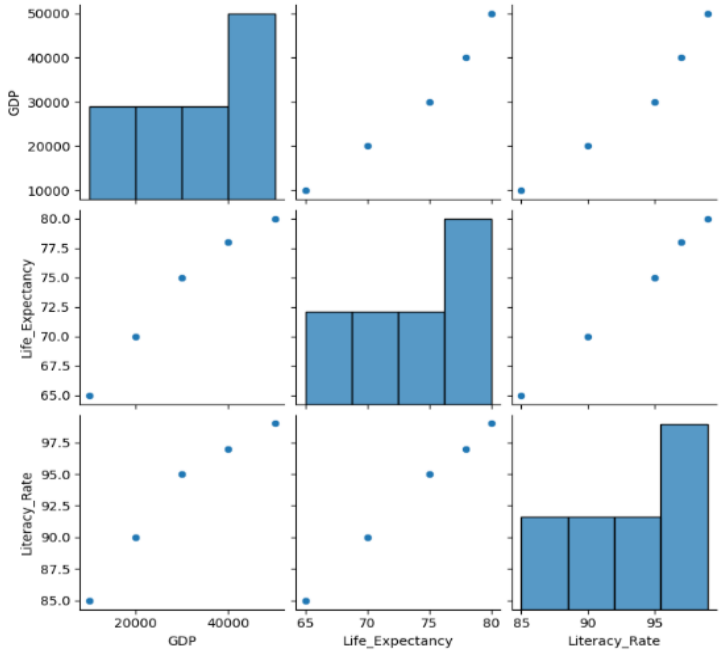


Fig. 2.43: Pair Plot

Difference in Bubble Charts and Pair Plots:

Table 2.3 Difference between bubble chart and pair plot.

FEATURE	BUBBLE CHART	PAIR PLOT
DEFINITION	A Scatter plot is usually a standard way to represent relationship between Two variables and is easier to interpret with focus on position of points across the x & y axis	A grid of scatter plots showing pairwise relation- ships between all numerical variables in a dataset.
USED FOR	Comparing three variables (X, Y, and bubble size) in a dataset	Exploring relationships bet-ween multiple variables in a dataset.
NUMBER OF VARIABLES	3 (X-axis, Y-axis, and Bubble Size).	More than 2 (plots relation- ships between all numerical variables).
BEST FOR	Business analytics, sales comparisons, financial data, and economic trends.	Data exploration, correlation analysis, and feature selec-ion in machine learning.
LIBRARIES USED	Matplotlib	Seaborn
VISUALIZATION TYPE	Single chart with bubbles of different sizes and colors.	Multiple scatter plots arranged in a grid.
EXAMPLE	Showing revenue, profit, and market share of different companies.	Exploring relationships between GDP, Literacy Rate, Life Expectancy, etc.

2.9 Time-Series Analysis (Trends & Seasonality):

A time series is simply a set of data that is tabulated over some time, probably at uniform intervals. The most generally known series of time data includes stock prices or foreign exchange rates of financial information. Time series data are also capable of identifying meteorological information or corresponding information like sales figures [24].

Time series can thus be either univariate or multivariate:

- Univariate time series data are concerned with time data based on one specific variable. Examples include the price of a likely stock or the number of new cases of a disease seen in a day.
- Multivariate time series refers to time data based on different (independent) variables; a good example of multivariate time series data would include weather data (which may include temperature, humidity, and precipitation).
- Time series are plotted over time, and the analyses include the application of statistical methods. But time series analysis forms the basis for future forecasting, giving a lot of insight into understanding complex data.

There are two principal typologies of time series compositional data:

- **Continuous data:** This form of data, observed at equal intervals, can be represented in graphs as a straight line-for instance, a thermometer-readout data point.
- **Discrete data:** This form recalls the observation of individual data points collected for specified instances and possible point representation in graphs-for instance, survey informer.

Creating Time Series analysis using line chart:

Example: 1. Import Libraries

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

2. Create Time Series Data

```
dates = pd.date_range(start="2023-01-01", periods=10, freq='D')
sales = [100, 120, 130, 125, 140, 145, 160, 180, 190, 200]
```

- pd.date_range() generates dates starting from January 1, 2023, for 10 days.

3. Plot the Line Chart

```
plt.plot(dates, sales, marker='o', linestyle='-', color='b', label="Sales Trend")
```

- plt.plot(x, y, options) → Creates the line chart.
- marker='o' → Marks each data point with a circle.
- linestyle='-' → Draws a solid line connecting points.
- color='b' → Sets the line color to blue.
- label="Sales Trend" → Adds a legend label.

4. Add Labels, Title, and Grid

```
plt.xlabel("Date")  # X-axis label
plt.ylabel("Sales ($)")  # Y-axis label
plt.title("Daily Sales Trend")  # Chart title
plt.legend()  # Show legend
plt.grid(True)  # Add a grid
```

5. Display the Chart

```
plt.show()
```

Output of the code is shown in fig 2.44

Output:

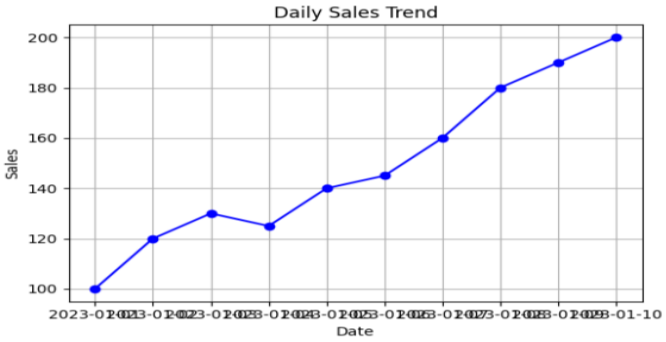


Fig. 2.44: Line chart for time series

Creating Time Series analysis using bar chart:

1. Import the Seaborn Library

import seaborn as sns

import matplotlib.pyplot as plt # Required for display

2. Define the Data

months = ["Jan", "Feb", "Mar", "Apr", "May"]

revenue = [5000, 7000, 6500, 8000, 9000]

- **Months** → The time periods for the X-axis.
- **Revenue** → The values for the Y-axis.

3. Create the Bar Chart Using Seaborn

sns.barplot(x=months, y=revenue)

- sns.barplot(x, y) → Creates a bar chart where:
  x=months → Months are on the X-axis.
  y=revenue → Revenue is on the Y-axis.
- Seaborn automatically styles the bars and calculates confidence intervals.

4. Add Labels and Title

plt.xlabel("Month")

plt.ylabel("Revenue ($)")

plt.title("Monthly Revenue")

5. Display the Chart

plt.show()

Output of the code is shown in fig 2.45

Output:

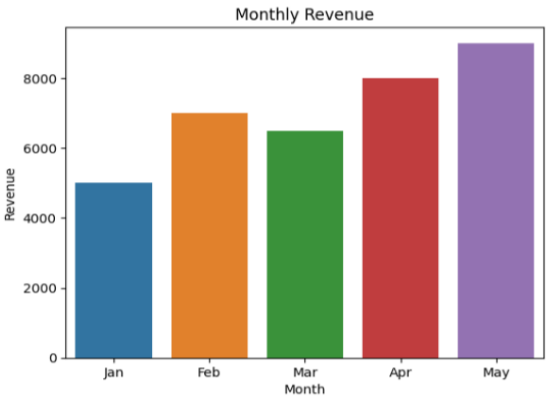


Fig. 2.45: Bar chart using Seaborn

2.10 Advanced Data Visualizations

Advanced data visualization is a branch of data visualization that involves representations of complex datasets in visual formats using sophisticated tools and techniques. Apart from basic graphs and charts, advanced visualization methods include interactive dashboards, heat maps, 3D plots, and real-time data feeds. This level of interaction allows users to explore data in more detail and with greater intuitiveness, facilitating the identification of trends, patterns, and anomalies. These techniques provide businesses with sophisticated insights, leading to informed decision-making [25].

Advanced Data Visualization Techniques

1. **Interactive Dashboards**

   An interactive dashboard is a dynamic tool that permits the user to manipulate data and see different views in real time. It often combines multiple data visualizations into a single interface, allowing users to drill in for further detail, filter data sets, and possibly monitor KPIs. This interactivity allows for greater insight into the data and more informed business decisions.

2. **Geospatial Visualization**

   Geospatial visualization is a process in which certain data points are placed on a geographical location, allowing the reader to observe the relationship and patterns of data in space. It is pertinent to organizations that operate in multiple regions or those that want to get a deeper understanding of location-based data such as sales territories, customer demographics, or supply chain visualization. Companies can see trends that would otherwise not be seen, thus making informed location-based strategic decisions.

3. **Heat Maps & Density Maps**

   Heat maps and density maps require the usage of color gradients representing data values, which occupy two-dimensional space. These maps visualize the intensity of data points available in an area or on some measurable scale: website clicks, customer activity, Usage of a resource, etc. The heat map shows the concentration of the information point, allowing the business to understand hotspots that would help in resource optimization.

2.11 Visualizing Geospatial Data

Geospatial Visualization is the process of displaying geographic data on maps to analyze patterns, relationships, and trends. It is commonly used in GIS (Geographic Information Systems), urban planning, climate studies, and business analytics.

Analysing geospatial data enables us to explore and find commonalities and relationships among the items in our geographically modelled world. Information on the distance between two items, the shortest path between them, the state of the area we monitor, and the terrain's height and land relief are all provided by the components of geospatial analysis. A 2D or 3D model of a specific area can then be made using this information. Business and public infrastructure decision-making are aided by geospatial analysis. For instance, it can be used to determine if ambulances can reach any location within a specified emergency response time or how they travel across a metropolis [25].

Through the use of maps, charts, and spatial analysis methods, geospatial visualisations turn location-based data into insightful understandings. Additional geospatial visualisation types beyond the fundamental ones are listed below:

1. **Choropleth Maps:** A map in which regions (states, districts, and countries) are shaded in various colours according to a data value (such as GDP, COVID-19 cases, or population density).

They are Ideal for comparing information from different geographical areas. Example: A global map with a colour gradient displaying GDP by nation .

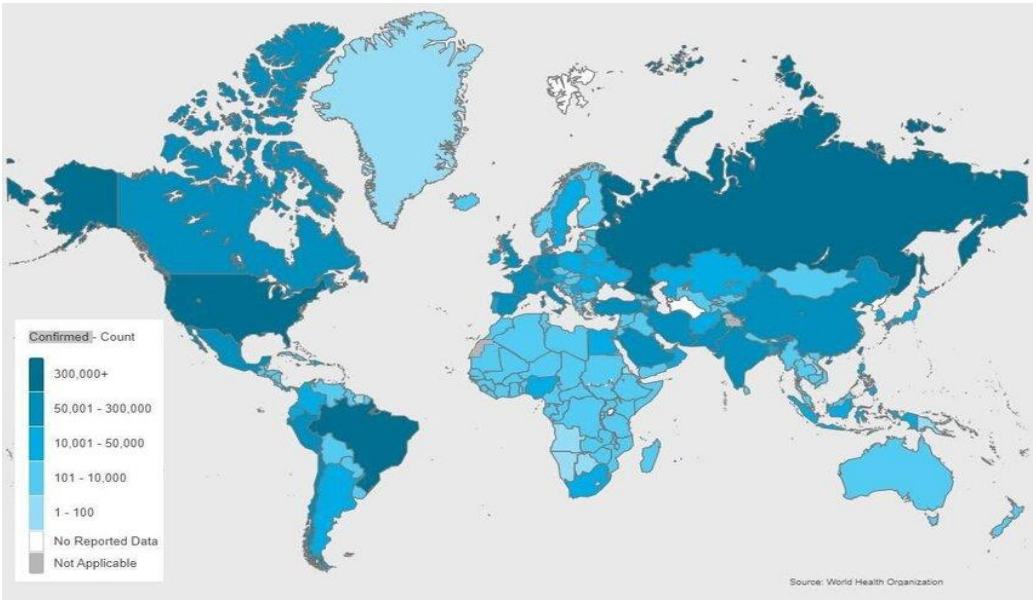


Fig. 2.46: Choropleth map representing the number of COVID-19 cases worldwide, on 2020/06/01.

2. **Point Map:** A point map is the simplest way of seeing geospatial data. It involves plotting a point anywhere on the map that corresponds to the variable you are measuring (for example, a landmark such as a hospital).

   This technique is effective in rehabilitating distribution and density patterns of an object, but it needs an accurate collection or geocoding of location data to pinpoint each location directly on the map as illustrated in the fig 2.47. The point technique can be unwieldy for a large-scale map, because points may certainly overlap at some zoom level.

Fig. 2.47: Point map

3. Proportional symbol map

Like the point map, this shows a feature at a particular location represented as a circle or some other shape. At the same time, it can convey several other variables from one point using its size and/or color (e.g., population and/or average age) as depicted in fig 2.48.

The proportional symbol maps allow integrating several variables at the same time. However, they share the same limitation that point maps have: the effort to capture too many data points on a large-scale map, particularly across relatively small geographic regions, can result in an overlap.

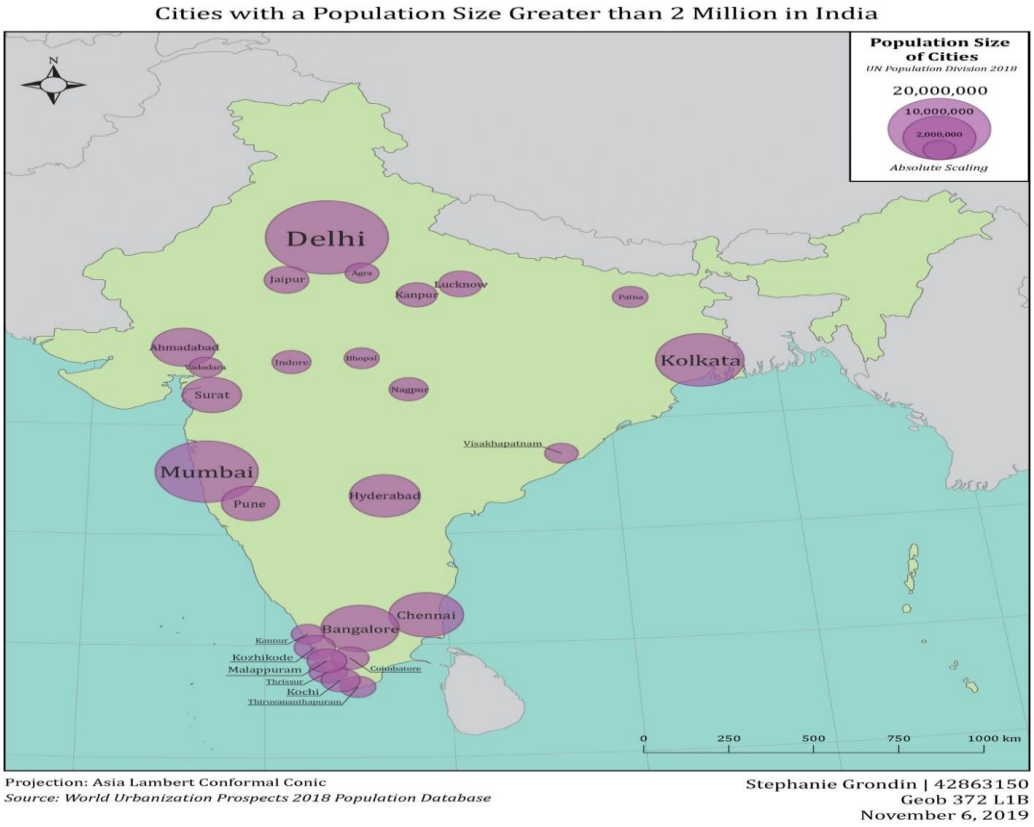


Fig. 2.48: Symbol map

4. **Cluster Map:** A cluster map is a sort of data visualization that groups similar data points together on account of certain characteristics. It is widely used in geospatial analysis, machine learning, and statistical data visualization to highlight patterns and relationships among data.

Types of Cluster Maps

1. Geospatial Cluster Map

Used to visualize clusters of geographic points, such as crime hotspots, disease outbreaks, and business locations.

Example: Heatmaps showing areas with the high concentration of events.

Libraries: folium, matplotlib, seaborn, geopandas.

2. Hierarchical cluster map (dendrograms)

Used in unsupervised learning to show relationships among the data points.

Example: Seaborn's clustermap that organizes rows and columns based on similarity.

Libraries: seaborn, scipy.cluster.hierarchy.

3. Heatmap with clustering

Heatmaps with clustering visualize relationships in data (for example, genetic data or correlation matrices).

Example: Clustering customers based on buying behavior.

Library used: seaborn.clustermap.

Creating Hierarchical cluster map in kaggle using python :

Install Necessary Libraries

!pip install seaborn pandas matplotlib scipy

- Ensures required libraries (seaborn, pandas, matplotlib, scipy) are installed.

Import Required Libraries

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from scipy.spatial.distance import pdist, squareform
from scipy.cluster.hierarchy import linkage, dendrogram
```

- seaborn: Used to create the clustermap.
- matplotlib.pyplot: Handles plotting.
- pandas: Creates and manipulates the dataset.
- numpy: Generates random data.
- scipy: Computes hierarchical clustering.

Generate a Random Dataset

```
data = pd.DataFrame(np.random.rand(10, 5), columns=[f'Feature_{i}' for i in range(1, 6)])
```

- Creates a 10×5 matrix with random values between 0 and 1.
- Columns are labeled Feature_1 to Feature_5.

Compute Distance and Linkage Matrix

```
distance_matrix = pdist(data, metric='euclidean')
linkage_matrix = linkage(distance_matrix, method='ward')
```

- Computes pairwise Euclidean distances between data points.
- Applies Ward’s method to form hierarchical clusters.

Generate a Clustermap

```
sns.clustermap(data, method='ward', cmap='coolwarm', standard_scale=1)
```

- Creates a heatmap with hierarchical clustering.
- method='ward': Uses Ward’s method for clustering.
- cmap='coolwarm': Uses a red-blue color gradient.
- standard_scale=1: Normalized data across columns.

Display the Plot

```
plt.show()
```

Output of the code is shown in fig 2.49

Output:

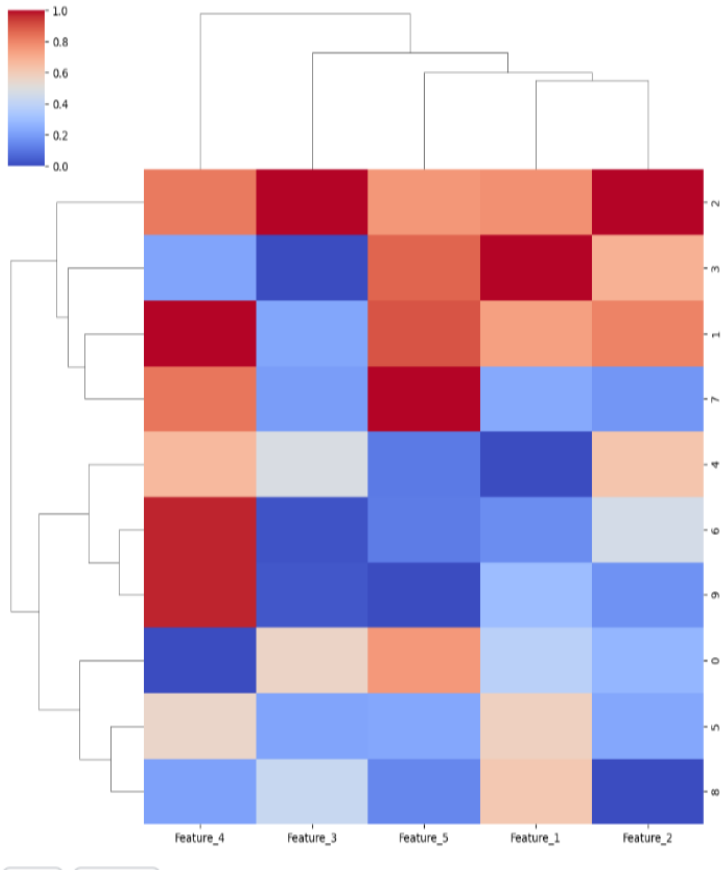


Fig. 2.49: Cluster map

Applications of Geospatial Visualization:

Geospatial visualization comprises a broad application across various domains to analyze spatial data easily, aiding in decision-making. Some of the most important applications include:

1. Urban Planning and Smart Cities

- It helps design cities through the visualization of population density, infrastructure, and transport networks.
- Used in zoning, land-use planning, and resource allocation.
- Example: GIS-based mapping to optimize public transport routes and green spaces.

2. Disaster Management and Risk Assessment

- Monitor and predict natural disasters (earthquakes, floods, wildfires).
- Guide emergency responders to distribute resources efficiently.
- Example: Real-time heat maps for detecting wildfire spread and evacuation planning.
- 3. Environmental Monitoring and Climate Change
- Monitor deforestation, air pollution, water quality, and carbon footprint.
- Analyze climate change effects using satellite imagery and weather patterns.
- Example: NASA's Earth Observatory maps to monitor glacier melting and global warming trends.

3. Epidemiology and Public Health

- Modeling the spread of physical diseases, which could include the tracking of diseases like COVID-19.

- Includes coverage maps, healthcare facilities, and hotspots/pockets of outbreaks.
- Example: Runtime map of COVID-19 global spread made by John Hopkins University.

4. Agriculture and Precision Farming

- Monitors soil health, irrigation, and crop yields from satellite imagery and drones.
- Allows for the preservation of fertilizers and water on farms.
- Example: NDVI (Normalized Difference Vegetation Index) mapping to assess plant health.

5. Crime Mapping and Law Enforcement

- Identify trends and hotspots for better policing strategies.
- Useful in predictive policing by evaluating past crime trends.
- Example: CompStat system used by the NYPD for crime analysis.

6. Business and Market Analytics

- Helps analyze customer behavior based on location-based sales data.
- Helps select new store locations based on food traffic and competition.
- Example: Starbucks uses geospatial analytics to locate profitable sites.

7. Transportation and Logistics

- Optimizes route planning, traffic flow, and fleet tracking.
- Used in ride-sharing apps, such as Uber and Lyft, and real-time GPS navigation, like Google Maps.
- Example: Amazon's logistics mapping for smooth warehouse deliveries.

Chapter 3

VISUALIZING TRENDS AND UNCERTAINTY

Ramandeep Kaur[1], Ritu Rani[2], Sahib Singh[3] and Navdeep Kaur[4]

[1,3,4]GNA University, Phagwara

[2]Rayat Bahra Group of Institutions and Nanotechnology, Hoshiarpur

3.1 Visualizing Trends:

A key component of data analysis and one of our most dependable methods for making decisions is visualising trends and patterns. It enables us to transform complicated data into insights by simplifying it. Visualizing data trends and patterns opens up a potent tool for comprehending the world we live in. Our brains are hardwired to interpret visual information. It should come as no surprise that humans typically absorb visual information much more quickly than word or numerical data [24].

3.2 Smoothing in Trend Visualization

"A statistical technique used to reduce short-term variations in data by averaging or weighting observations, thereby revealing underlying patterns and trends with greater clarity."

Smoothing is a fundamental technique in data visualization and time-series analysis, aimed at reducing noise and enhancing the clarity of underlying trends. It is particularly useful in datasets where short-term fluctuations may obscure long-term patterns. By applying mathematical transformations, smoothing helps analysts and researchers focus on meaningful insights without the distraction of random variations [24].

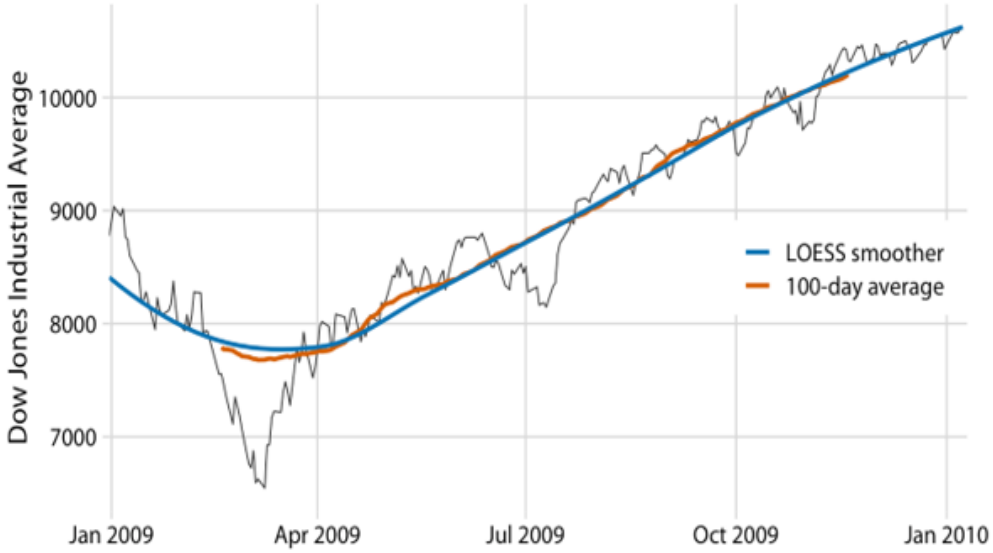


Fig. 3.1: Smoothening in trend visualization

The image above (fig 3.1) illustrates trend smoothing in the Dow Jones Industrial Average (DJIA) during 2009 using two different techniques:

- 100-Day Moving Average (Brown Line): A rolling average that reduces short-term fluctuations by averaging stock prices over a fixed period. This method helps to highlight trends but may lag behind actual market movements.
- LOESS Smoother (Blue Line): A locally weighted regression method that provides a more flexible and adaptive trend line, capturing both short-term and long-term variations in the data.

3.2.1 Types of Smoothing

1. Moving Average Smoothing

In data visualization, Moving Average Smoothing is a very common algorithm for understanding trends in time-series data. What it does is calculate the mean of data points within a fixed window, hence smoothing out the fluctuations and depicting the inherent trend. This method is particularly useful in finance, economics, climate science, and demand forecasting, where the raw data is likely to have much short-term volatility.

Concept of Moving Average Smoothing

A moving average essentially smooths out data by substituting each value with the mean of values in neighboring positions in a fixed window. This serves to minimize noise while preserving the general shape of the dataset. On the basis of the manner in which weights are distributed to previous values, moving averages may be categorized into various forms.

Example: The following code demonstrates how to apply Moving Average Smoothing on a time-series dataset using python:

1. Data Generation

```
dates = pd.date_range(start="2023-01-01", periods=100, freq='D')
data = pd.DataFrame({'Date': dates, 'Value': (pd.Series(range(100)) +
pd.Series(range(100)).rolling(5).mean().fillna(0))})
```

- Generates 100 sequential dates starting from January 1, 2023.
- Creates a synthetic time-series dataset with fluctuating values.
- Uses.rolling(5).mean() to introduce some initial variation.

2. Applying Moving Averages

```
data['SMA_7'] = data['Value'].rolling(window=7).mean()  # 7-day Simple Moving Average
data['SMA_14'] = data['Value'].rolling(window=14).mean()  # 14-day Simple Moving Average
```

- Computes 7-day and 14-day simple moving averages (SMA).
- The .rolling(window).mean() function calculates the average of the previous n values, smoothing short-term fluctuations.

3. Visualizing the Data

```
plt.figure(figsize=(12,6))
sns.lineplot(x='Date', y='Value', data=data, label='Original Data', color='black', alpha=0.5)
sns.lineplot(x='Date', y='SMA_7', data=data, label='7-Day SMA', color='blue')
sns.lineplot(x='Date', y='SMA_14', data=data, label='14-Day SMA', color='red')
```

- Original data (black line): Represents the raw, unprocessed time-series data.
- 7-Day SMA (blue line): Captures short-term trends while reducing noise.
- 14-Day SMA (red line): Provides a smoother long-term trend, ideal for identifying overall movement.

4. Enhancing the Visualization

```
plt.title('Moving Average Smoothing in Time-Series Data')
plt.xlabel('Date')
plt.ylabel('Value')
plt.legend()
plt.show()
```

Output of the code is shown in fig 3.2

Output:

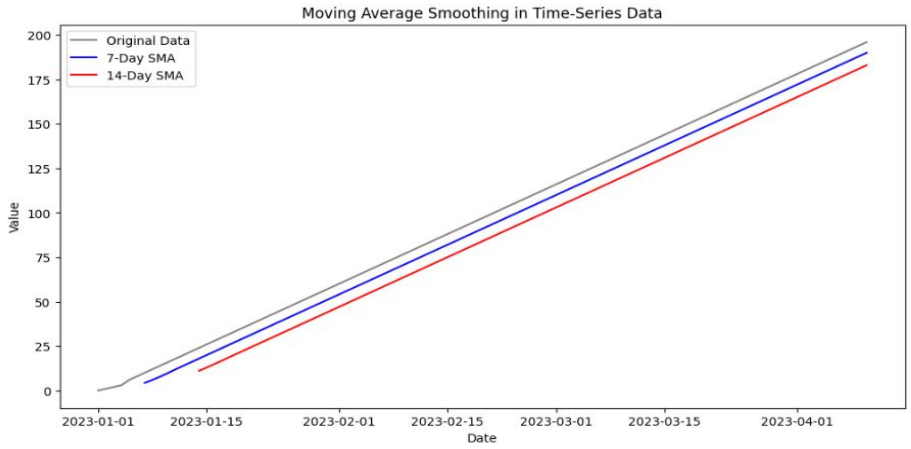


Fig. 3.2: Moving averages

The graph will display:

1. Black Line: The original data with random fluctuations.
2. Blue Line (7-Day SMA): A smooth curve that follows short-term patterns.
3. Red Line (14-Day SMA): A more stable curve, highlighting the long-term trend.

Types of Moving Averages

1. Simple Moving Average (SMA):

- Averages an equal number of past observations of fixed weights.
- Applied to analysis of stock market to observe long-term patterns.

2. Weighted Moving Average (WMA):

- Puts heavier emphasis on the newer observations in order to increase sensitivity to recent trends.
- Preferably used to forecast inventories and sales.

3. Exponential Moving Average (EMA):

- Takes averages of exponentially declining weights with previous values, assigning higher relative importance to recent observations.
- Often applied to finance for smoothing stock price changes.

4. Cumulative Moving Average (CMA):

- Averages all previous data points right through to the present point, re-calculating whenever fresh data is added.
- Applied in long-term trend analysis.

Applications of Moving Average Smoothing

- Financial Markets: Studies of stock trends and forecasting price movements.
- Weather Forecasting: Determining climate patterns on various time scales.
- Economic Analysis: Examining GDP growth rates and inflation trends.
- Demand Forecasting: Forecasts product sales based on past data.

The Importance of Smoothing in Trend Analysis

Smoothing is widely used in various fields, including finance, climate science, and economic forecasting. It allows researchers to:

- **Identify long-term patterns** by reducing short-term noise.
- **Improve predictive accuracy** in forecasting models.
- **Make data-driven decisions** by understanding overall movements rather than reacting to temporary fluctuations.

For example, in stock market analysis, smoothing techniques help investors distinguish between daily price variations and genuine long-term growth trends. Similarly, in climate studies, smoothing is applied to temperature records to observe global warming patterns over decades.

By using appropriate smoothing techniques, data visualization becomes more insightful, guiding better decision-making across multiple domains.

2. Exponential smoothing

Time series methods assume a forecast is the sum of all past observations or lags. Exponential smoothing gives more weight to the most recent observations and decreases exponentially as the observations get further back in time; with the assumption the future will be like the recent past. The term "exponential smoothing" means each demand observation is assigned an exponentially decreasing weight.

- This captures the general pattern and can be extended to include trends and seasonality to make precise time series forecasts from past data.
- Gives a bit of long-term forecast errors.
- Works well with smoothing when time series parameters change slowly over time.

Types of Exponential Smoothing

1. Simple or Single Exponential smoothing

Simple smoothing is a method of forecasting time series using univariate data without a trend or seasonality. You need only one parameter, which is also referred to as alpha (α*α*) or smoothing factor so as to check how much the impact of past observations should be minimized.the weight to be given to the current data as well as the mean estimate of the past depends on the smoothing parameter (α*α*). A smaller value of a implies more weight on past prediction and vice-versa. The range of this parameter is typically 0 to 1.

The formula for simple smoothing is:

st=αxt+(1–α)st−1=st−1+α(xt–st−1)*st*=*αxt*+(1–*α*)*st*−1=*st*−1+*α*(*xt*–*st*−1)

where,

- st*st* = smoothed statistic (simple weighted average of current observation xt)
- st−1*st*−1 = previous smoothed statistic
- α*α* = smoothing factor of data; 0<α<10<*α*<1
- t*t* = time period

2. Double Exponential Smoothing

Double exponential smoothing, also known as the Holt's trend model, or second-order smoothing, or Holt's Linear Smoothing is a smoothing method used to predict the trend of a time series when the data does not have a linear trend but does not have a seasonal pattern. The fundamental idea behind double exponential smoothing is to use a term that can take into account the possibility that the series will show a trend.

Double exponential smoothing requires more than just an alpha parameter. It also requires a beta (b) factor to control the decay of the effect of change in the trend. The smoothing method supports both additive and multiplicative trends.

The formulas for Double exponential smoothing are:

st=αxt+(1–α)(st−1+bt−1)*st*=*αxt*+(1–*α*)(*st*−1+*bt*−1)bt=β(st–st−1)+(1–β)bt−1*βt*=*β*(*st*–*st*−1)+(1–*β*)*bt*−1

where,

- bt*bt* = trend at time t

- $\beta\beta$ = 0<β<10<β<1

The following Python code demonstrates how to apply Exponential Smoothing using statsmodels to smooth time-series data effectively.

1. Generating Time-Series Data

```
np.random.seed(42)
dates = pd.date_range(start="2023-01-01", periods=100, freq='D')
values = np.cumsum(np.random.randn(100)) + 50  # Creating a random walk trend
data = pd.DataFrame({'Date': dates, 'Value': values})
```

- Generates 100 time-series data points from January 1, 2023.
- Uses a random walk model to simulate a real-world dataset (e.g., stock prices, sales trends).

2. Applying Exponential Smoothing

```
data['EWMA_0.2'] = data['Value'].ewm(alpha=0.2).mean()  # Smoothing factor 0.2
data['EWMA_0.5'] = data['Value'].ewm(alpha=0.5).mean()  # Smoothing factor 0.5
```

- ewm(alpha=α).mean() applies Exponentially Weighted Moving Average (EWMA).
- α (smoothing factor) determines the weight of recent values:
    - α = 0.2 (blue line): More smoothing, long-term trends.
    - α = 0.5 (red line): Less smoothing, reacts faster to changes.

3. Visualizing the Results

```
plt.figure(figsize=(12,6))
sns.lineplot(x='Date', y='Value', data=data, label='Original Data', color='black', alpha=0.5)
sns.lineplot(x='Date', y='EWMA_0.2', data=data, label='EWMA (α=0.2)', color='blue')
sns.lineplot(x='Date', y='EWMA_0.5', data=data, label='EWMA (α=0.5)', color='red')
```

- Original Data (black line): Represents the raw time-series data.
- EWMA (α=0.2, blue line): A smoother trend capturing long-term movements.
    - EWMA (α=0.5, red line): A more responsive line that adjusts quickly to new changes.

Output of the code is shown in fig 3.3

Output:

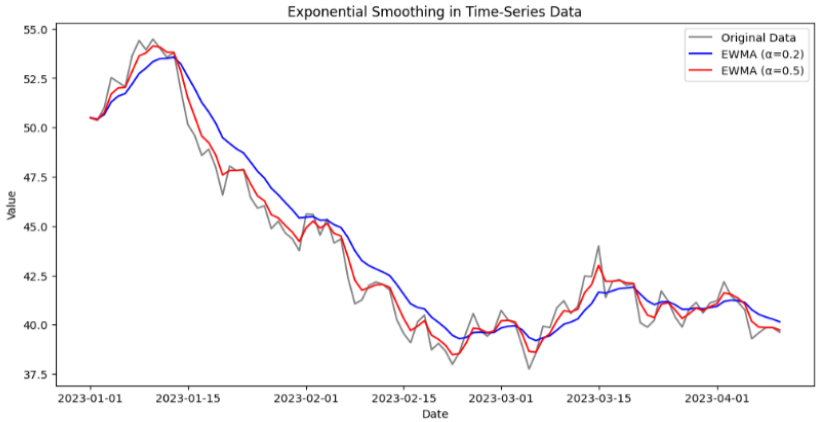


Fig. 3.3: Exponential Smoothening

Exponential Smoothing is widely used in:

- Stock price trend analysis
- Sales forecasting and demand prediction
- Weather and climate pattern analysis

- Economic and financial modeling

3. Showing Trends with a Defined Functional Form

In data visualization, trends can be represented by specific functional forms, which involve a mathematical equation that explicitly describes the connection between variables. By using these functions, fluctuations in the data can be smoothed out and fundamental patterns can be emphasized. This method is especially useful when the data shows a consistent relationship, like linear, polynomial, logarithmic, or exponential trends. By applying a functional form to the data, we can predict future values, examine long-term trends, and better understand data behavior. This technique is commonly used in fields such as economics, finance, climate research, and machine learning.

Example: Polynomial Trend Fitting in Python

To illustrate how a defined functional form can be applied, consider the following example, where we fit linear and polynomial trends to a dataset.

1. Generating Time-Series Data

```
np.random.seed(42)
x = np.arange(1, 51)  # 50 time points
y = 2.5 * x + np.random.normal(scale=10, size=x.shape)  # Linear trend with noise
```

- Creates 50 time points (x) with a synthetic linear trend (y = 2.5x + noise).
- Random noise is added to simulate real-world fluctuations.

2. Fitting Functional Trend Lines

```
linear_fit = np.poly1d(np.polyfit(x, y, 1))  # Linear model (1st-degree polynomial)
poly_fit = np.poly1d(np.polyfit(x, y, 2))    # Quadratic model (2nd-degree polynomial)
```

- Linear fit (degree = 1): Models a straight-line trend.
- Quadratic fit (degree = 2): Captures curvature for better trend representation.

3. Visualizing Trends

```
sns.scatterplot(x=x, y=y, label="Original Data", color="black", alpha=0.6)
plt.plot(x_pred, y_linear, label="Linear Trend", color="red")
plt.plot(x_pred, y_poly, label="Polynomial Trend (Quadratic)", color="blue")
```

- Scatterplot (black points) represents the raw data.
- Red line (Linear Trend) shows a constant rate of change.
- Blue line (Quadratic Trend) captures a curved relationship, offering a better fit.

Output of the code is shown in fig 3.4

Output:

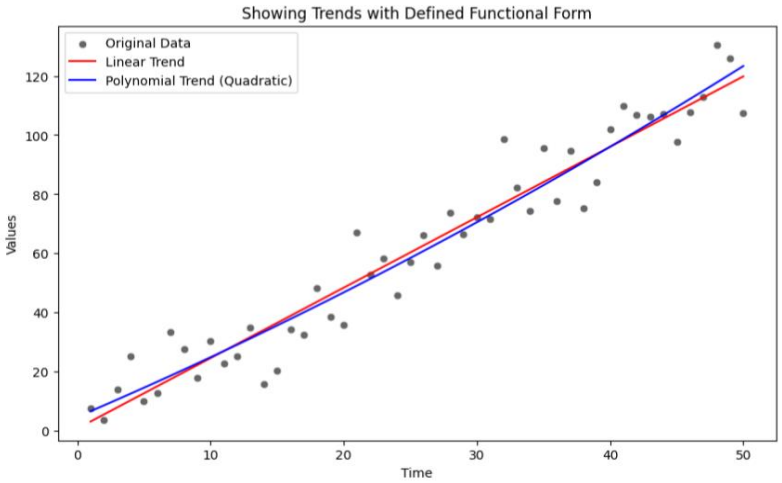


Fig. 3.4: Trends with functional form

Real-World Applications

- Stock Market Analysis: Capturing long-term growth trends in financial data.
- Climate Change Studies: Modeling temperature variations over decades.
- Sales Forecasting: Identifying seasonal patterns in business performance.

3.3 Detrending & Time-Series Decomposition

A lot of datasets often yield considerable amounts of data that mask the inner insights intended for a given analysis, characteristically in time-series. Essentially, detrending refers to the removal of long-term developments taking away such unwanted trends that obscure short-term variations of interest, whereas time-series decomposition breaks down a dataset into a limited set of core components [25]:

Trend Component - The movement of data, such as population growth or increasing global temperatures, occurring over a long time

Seasonal Component - Repeated patterns, for example, higher ice cream sales in summers, increased online shopping before holidays, etc.

Residual or Noise Component - Random fluctuations that are devoid of any apparent pattern.

Moving averages: This technique smooths the dataset inserting general trends by taking averages using sliding windows.

Differencing: It removes trends in time-series by only subtracting the two consecutive time values from each other, which makes the given dataset stationary and more suitable for various analytical methods.

STL Decomposition (Seasonal-Trend Decomposition Using Loess): A composition analysis technique well-known for separating time-series data into trend, seasonal, and residual components to produce clearer insights.

For example, economic data often tend to be detrended prior to analysis.GDP growth, inflation rates, and employment levels vary over time, and by removing enduring trends, an analyst may seize clear microeconomic cycles on short-term analyses or seasonal effects.

1. Generating Synthetic Time-Series Data:

```
time = pd.date_range(start='2020-01-01', periods=100, freq='D')
trend = np.linspace(10, 50, 100)  # Linear increasing trend
seasonality = 5 * np.sin(2 * np.pi * time.dayofyear / 30)  # Monthly seasonality
noise = np.random.normal(scale=3, size=100)  # Random fluctuations
data = trend + seasonality + noise  # Combine components
```

- Trend Component: A linear increase in values over time.
- Seasonal Component: A repeating sinusoidal wave simulating monthly patterns.
- Noise Component: Random fluctuations mimicking real-world uncertainty.

2. Performing Time-Series Decomposition:

```
decomposition = seasonal_decompose(df['Value'], model='additive', period=30)
```

- Uses additive decomposition, which assumes that the time-series is composed of:
    - Trend: Long-term growth or decline.
    - Seasonality: Recurring patterns.
    - Residual (Noise): Unexplained variations.

3. Visualizing the Decomposed Components:

```
plt.plot(decomposition.trend, label="Trend Component", color='red')
plt.plot(decomposition.seasonal, label="Seasonal Component", color='blue')
```

plt.plot(decomposition.resid, label="Residual Component", color='green')

Output of the code is shown in fig 3.5

Output:

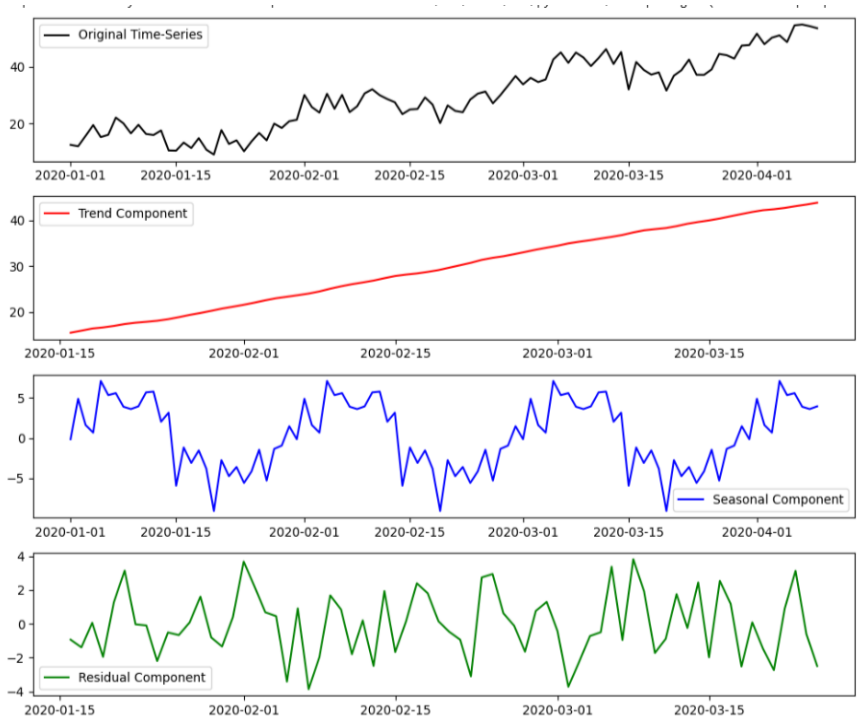


Fig. 3.5: Trend lines, seasonality & residual fluctuations

i. Trend Line: Shows a steady increase, indicating growth over time.
ii. Seasonal Pattern: Displays a monthly cyclic behaviour, useful for identifying seasonal variations.
iii. Residual Fluctuations: Help detect unexpected anomalies or outliers.

3.4 Visualizing Geospatial Data

In the modern era of data-driven decision-making, geospatial data visualization plays a crucial role in transforming raw location-based information into meaningful insights. By leveraging maps, layers, and advanced visual encoding techniques, geospatial visualization enables analysts to uncover patterns, trends, and relationships that might otherwise remain hidden in traditional dataset [24].

One of the fundamental strengths of geospatial visualization is its ability to provide contextual awareness—allowing users to see how data varies across different regions, detect clusters, and analyze spatial dependencies. Whether in urban planning, environmental monitoring, epidemiology, or logistics, mapping techniques such as choropleth maps, heatmaps, cartograms, and spatial interpolation help in effectively representing large-scale spatial information. Advancements in Geographic Information Systems (GIS), remote sensing, and machine learning have further expanded the capabilities of geospatial visualization. Interactive tools allow users to zoom, filter, and overlay multiple data layers, providing a multidimensional perspective of the data.

3.4.1 Projections:

In the field of data visualization, projections are mathematical techniques used to represent a three-dimensional (3D) world on a two-dimensional (2D) plane. Due to the Earth's spherical shape, flattening it onto a map inevitably causes distortions in area, shape, distance, or direction. Various types of projections are employed to balance these distortions based on the visualization's purpose.

There are different types of projections in mapping:

1. Mercator Projection: This type preserves angles and directions but distorts area, making regions near the poles appear larger. It is commonly used in navigation maps due to its accurate representation of direction.

2. Robinson Projection: A compromise projection that balances size and shape distortions. It is used in world maps to offer a visually pleasing global representation.

3. Equal-Area Projection (e.g., Mollweide, Albers): These projections maintain area proportions but distort shape and distance. They are often utilized in population density maps or environmental studies.

4. Azimuthal Projection: This projection maintains accurate distances from a central point. It is commonly used in aeronautical charts to show great-circle distances accurately.

By selecting the appropriate projection, cartographers and data analysts ensure that spatial relationships are depicted in a way that optimally supports the analysis and communication of insights.

1. Installing Required Libraries

!pip install geopandas matplotlib cartopy

- GeoPandas: Used to handle and visualize geospatial data.
- Matplotlib: Helps in plotting the transformed maps.
- Cartopy: Provides additional support for handling map projections (useful for more advanced cartography).

2. Loading the World Map Dataset

```
import geopandas as gpd
import matplotlib.pyplot as plt
# Load the world map dataset
world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))
```

3. Defining Projections to Apply

```
projections = {
"Mercator Projection": "EPSG:3395",
"Robinson Projection": "ESRI:54030",
 "Mollweide Projection": "ESRI:54009",
 "Azimuthal Projection": "ESRI:54032"
        }
```

- A dictionary is created to store different projection systems (CRS - Coordinate Reference Systems).
- The projections included are:
- Mercator (EPSG:3395) – Preserves angles but distorts area (widely used in web mapping).
- Robinson (ESRI:54030) – A compromise projection balancing shape and size (used in world maps).
- Mollweide (ESRI:54009) – Equal-area projection, best for population or climate maps.
- Azimuthal (ESRI:54032) – Shows distances accurately from a central point.

4. Creating Subplots to Compare Projections

```
fig, axes = plt.subplots(2, 2, figsize=(12, 8))
```

- A 2x2 grid of subplots is created to display all four projections side by side.
- The figsize=(12,8) ensures the maps are displayed clearly.

5. Applying Projections and Plotting the Maps

```
for ax, (title, proj) in zip(axes.flat, projections.items()):
```

```
world_projected = world.to_crs(proj)  # Convert projection
 world_projected.plot(ax=ax, edgecolor="black", color="lightblue")
 ax.set_title(title)
 ax.axis("off")
```

6. Adjusting Layout and Displaying the Maps

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

Output of the code is shown in fig 3.6

Output:

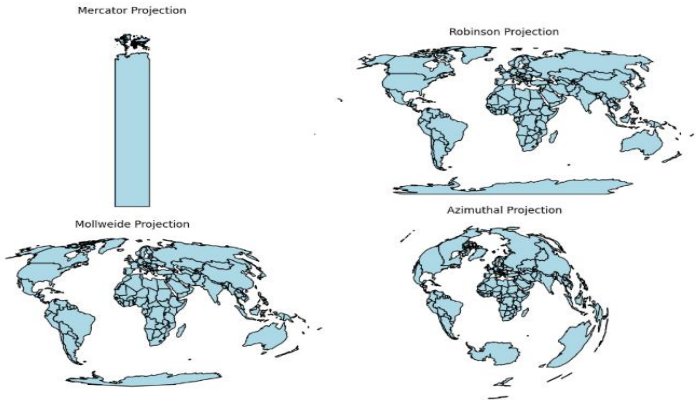


Fig. 3.6: Projections

3.4.2 Layers:

Data visualization combines layers into beautiful, complex, multi-dimensional representations of data. A layer in visualization is a distinct aspect of information that further constitutes meaning, depth, and clarity to a specified chart, graph, or map. The composition of multiple layers can build up rich, insightful representations of data that enhance interpretation and storytelling.

Layers are indispensable within all types of visualization, be it geospatial mapping, statistical graphics, or interactive dashboards. Each layer is designed for a purpose—be it context, annotation, or comparison—in order to allow analysts, researchers, and decision-makers to derive meaningful insight.

Types of Layers in Data Visualization.

Base Layer:

A base layer is the base of any visualization and provides context; it serves as a backdrop against which more data may be added.

Example:

- In a geospatial visualization, the base layer can provide the map; this could be a world map used to plot country-level data.
- In the bar chart, the base layers are the axes, gridlines, and labels that serve to provide structure.

Data Layers:

As the visualization will illustrate, this layer represents the actual data points.

Example:

- In a line chart, the main data layer is the trend line that connects the data points.
- In a choropleth map, the data layer shows a color-coded region based on such values as, for instance, population density.

Feature Layers:

Feature layers are additional informational overlays on primary data. These layers might represent features on a map, such as geographic, political, or specific markers.

Example:

- A weather map might represent a feature layer with city names, country borders, or rivers.
- In the scatter plot, a feature layer is trend lines or clusters.

In Kaggle, you can use GeoPandas, Folium, and Matplotlib to create a map with multiple layers.

```
# Install necessary libraries !pip install folium
import folium
# Create a base map centered at a specific location
m = folium.Map(location=[20, 0], zoom_start=2)
# Add a polygon layer (Example: Highlight a country)
folium.Polygon(locations=[[28, 77], [37, -122], [51, -0.1]],  # Example coordinates (Delhi, SF, London)
 color="blue",
 fill=True,
 fill_color="lightblue",
 fill_opacity=0.5
).add_to(m)
# Add a marker layer for key cities
folium.Marker([28.6139, 77.2090], popup="New Delhi").add_to(m)  # India
folium.Marker([37.7749, -122.4194], popup="San Francisco").add_to(m)  # USA
folium.Marker([51.5074, -0.1278], popup="London").add_to(m)  # UK
# Add a circle layer (Example: Highlight population zone)
folium.Circle(
  location=[34.0522, -118.2437],  # Los Angeles
 radius=500000,  # 500 km radiuscolor="red", fill=True,
 fill_color="pink")
fig.show().
```

Output of the code is shown in fig 3.7

Output:

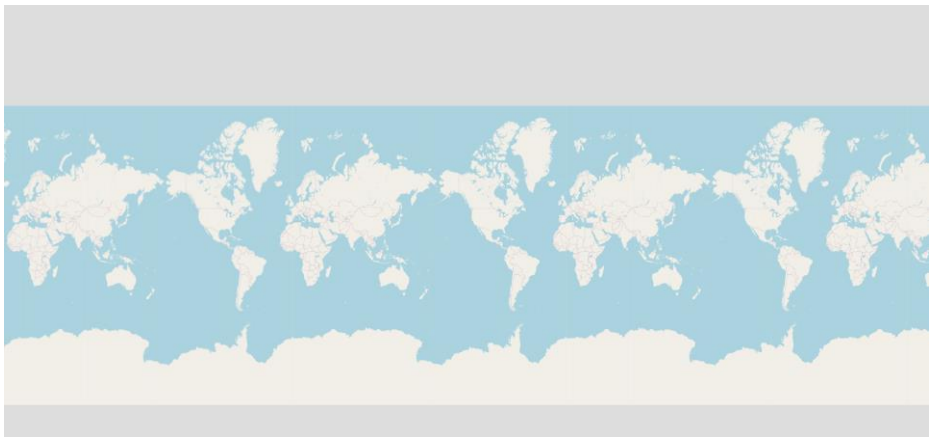


Fig. 3.7: Layers

3.4.3 Choropleth Maps

A choropleth is a kind of thematic map on which areas are shaded or patterned in proportion to the value of the statistical variable being displayed, in this case, population density or per capita income. The choropleth gives a very good way to see how a particular measure varies across some land area, or how much variation there is in a region.

Types of Choropleth Maps

Choropleth maps are powerful geospatial visualization tools that use color gradients to indicate data values across geographic regions. These maps are widely used in demographics, economics, public health, and environmental studies. Based on classification methods and data representation techniques, choropleth maps can be categorized into several types.

1. Sequential Choropleth Maps:

It uses a single color that goes from light to dark (or vice-versa) to show progression from low to high values.

Example: Mapping population density in which lighter shades indicate lower density and darker shades indicate higher density.

2. Diverging Choropleth Maps:

It Uses contrasting colors to show positive and negative deviation from a central reference point (a zero or an average value).

Example:

Election results where blue denotes one party and red includes another party, with neutral tones used where differences are marginal.

3. Binned Choropleth Maps (Classed Choropleth):

Data falls into discrete classes or categories (for example, five equal intervals), with each range distinguished by a particular color.

Example:

Air pollution levels: safe, moderate, unhealthy, hazardous.

Creating a choropleth map using Plotly in Kaggle:

Step 1: Importing Required Libraries

```
import pandas as pd
import plotly.express as px
```

Step 2: Creating the Dataset

```
data = {
   'Country': ['United States', 'China', 'Japan', 'Germany', 'India'],
   'GDP': [21.43, 14.34, 5.08, 3.84, 2.87],  # in Trillions
}
```

Step 3: Converting the Data into a DataFrame

```
df = pd.DataFrame(data)
```

Step 4: Creating the Choropleth Map

```
fig = px.choropleth(df, locations="Country",  # country names
locationmode="country names",  # specify country names mode
color="GDP",  # column for color coding
hover_name="Country",  # add hover information
```

```
color_continuous_scale="Viridis",  # color scale
title="Choropleth Map of Countries by GDP")
```

Step 5: Displaying the Map

```
fig.show().
```

This code generates an interactive choropleth map where GDP for nation-states is displayed on it. Each country is colored based on its GDP, with a color scale for the range. Hovering on a particular country causes its name and GDP value to appear upon the cursor. The interactive map can be employed to zoom and pan Output of the code is shown in fig 3.8.

Output:

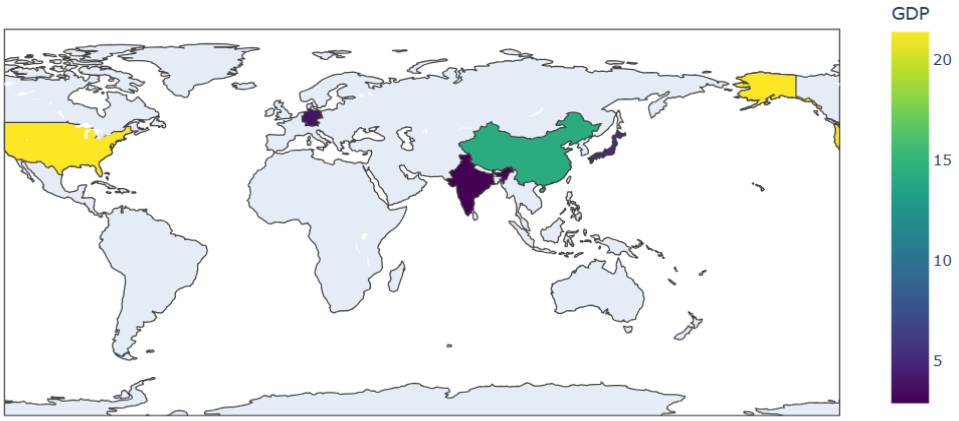


Fig. 3.8: Choropleth map

3.4.4 Cartogram Maps: A cartogram is said to be a thematic map that substitutes for land area or distances a mapping variable like travel time, population, or GNP. As the map geometry conveys the information regarding an alternative variable, the map is usually distorted, sometimes vastly so. Cartograms have long been used primarily to display emphasis and minimize the kinds of natural distortions that can affect our perception of real geography.

Types of Cartograms

Non-Contiguous Cartograms: In this type, each region is resized independently while maintaining its shape and position. For example, a population cartogram where countries are expanded or contracted without overlapping.

Contiguous Cartograms: This type distorts the map as a whole while preserving connectivity between regions. Shapes are altered, such as in an election results map where states grow or shrink based on votes received.

Dorling Cartograms: Instead of distorting shapes, geographic areas are represented as circles sized according to the data variable. For instance, a wealth distribution cartogram where larger circles represent higher GDP per capita.

Applications of Cartograms:

-Demographic Studies: Displaying population density variations.

- Economic Analysis: Representing GDP, trade volumes, or income distribution.

- Public Health Mapping: Visualizing disease spread, healthcare access, or vaccination rates.

- Environmental Studies: Showing deforestation rates, carbon emissions, or water scarcity impact.

Example: World Population Cartogram Transforming a world map into a cartogram where countries are resized based on their population would highlight large nations like China and India. Less populated

countries would shrink significantly, visually emphasizing global population distribution, less apparent in traditional maps.

By effectively using cartograms, data visualization practitioners can present complex spatial data.

```
# Install necessary libraries
!pip install geopandas matplotlib cartopy
import geopandas as gpd
import matplotlib.pyplot as plt
import numpy as np
# Load a world map shapefile
world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))
# Example: Adding a hypothetical population dataset
world['Population'] = world['pop_est']
world['Cartogram_Size'] = np.log1p(world['Population'])  # Apply transformation for better visualization
# Plot the cartogram
fig, ax = plt.subplots(figsize=(12, 6))
world.plot(ax=ax, column='Cartogram_Size', cmap='Blues', legend=True, edgecolor='black')
# Add titles and labels
plt.title("World Population Cartogram", fontsize=14)
plt.xlabel("Longitude")
plt.ylabel("Latitude")
# Show the plot
plt.show()
```

Output of the code is shown in fig 3.9.

Output:

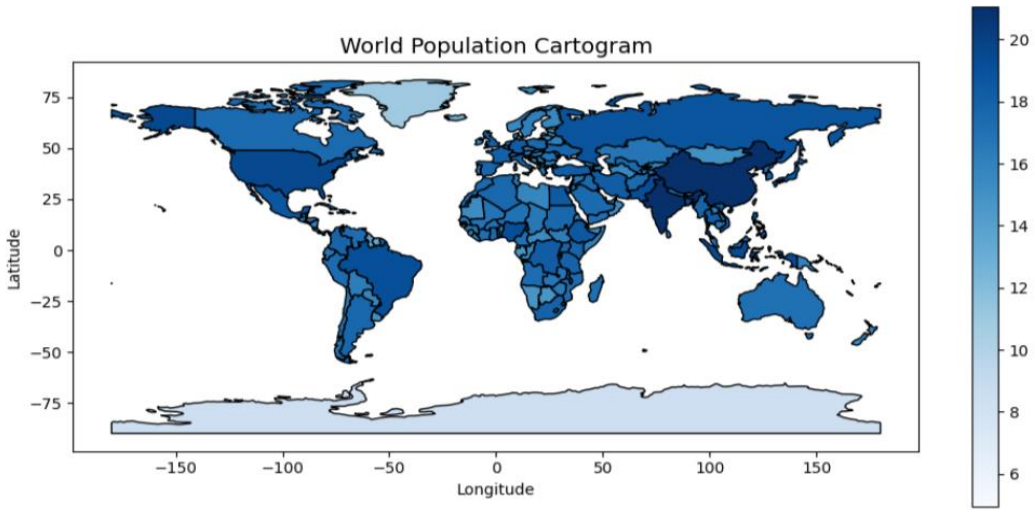


Fig. 3.9: Cartogram

3.5 Visualizing Uncertainty & Trends: Uncertainty visualization is not a new topic. It has been talked about a hundred years ago with a majority of uncertainty visualizations about summary statistics and statistical inference except for scientific research. Uncertainty in the scientific community is much more complex. Some of them pertain to uncertainty in data quality, others treat the visualization of three (or more) dimensions [25].

The visual representation of uncertainty can help you and your audience understand the quality, reliability, and variability of your data and make informed decisions based upon them. Uncertainty may come from various places, including errors in the measurements, bias of sampling, missing data, model assumptions, or future scenarios. With that representation, uncertainty can express the range of possible outcomes with a range of confidence or probabilities. Such a visual mapping reveals an overall state of agreements and/or disagreements, and risks and opportunities concerning your data.

The roles of uncertainty and trends are vital in data analysis as they affect decision-making and predictions. When examining economic indicators, climate trends, or stock market values, grasping the effects of uncertainty on trends is critical. In the following sections, we delve into important techniques for managing and illustrating uncertainty in data.

Techniques for Visualizing Uncertainty:

Uncertainties with respect to the data can arise out of several origins. These include measurement inaccuracies, limitations associated with the data, and underlying assumptions made with regard to the model. While deterministic models impose fixed values regarding the data, the actuality of data is usually probabilistic.

To visualize uncertainty, we can use several techniques:

1. **Confidence Interval:** This is the probable range of values within which a data point is likely to fall and is usually expressed numerically, e.g., 95% confidence. Shaded regions around trend lines in plots represent this concept.
2. **Error Bars:** A graphical representation commonly used in scatter plots or bar charts to indicate the variation in data points, thereby providing a direct indication of uncertainty.
3. **Violin and box plot:** Used to display distributions of data along with median, quartiles and potential outliers; this enables a useful comparison between datasets containing inherent variability.
4. **Predictive uncertainty visualization:** This allows for other model projections to issue forth on top of each other, so you see the range in which a future trend is plausible in machine learning or forecasting. This enables a reasonable understanding of risk for a decision-maker.

How to Choose a Technique:

When deciding on a technique for visualizing uncertainty, one should think about the type and level of uncertainty you intend to convey: Is it in a single variable, a relationship, a distribution, a prediction, or a comparison? The type and scale of your data must also be borne in mind: Is it categorical, numerical, temporal, spatial, or multidimensional? Is it continuous, discrete, or ordinal? The goal and context of your visualization should be considered for any visual: Will it be exploratory, explanatory, or persuasive? Who is your audience, what is their background, and what do they care about? How will your visualization be displayed and how will users access it? Other considerations include the number of data points. In climate science, for example, projections of temperature in future decades typically have banded uncertainty attached, indicating the range of different scenarios for modeling. The same principle is also applied in stock market forecasting, where projected stock prices will be presented to include potential upper and lower bounds[26].

Visualizing uncertainty and trends is a very fundamental part of data analysis that cuts a very serious rate of misleading insights or over-determined conclusions. Analysts can effectively utilize confidence intervals, error bars, and predictive uncertainty methods for communicating the range or spread of possible outcomes.

Limitations and challenges: The visualization of uncertainty itself has limitations. For instance, many times, a data set may not have error estimates; some uncertainty measures have no universally standard definitions or methodology, whereas some uncertainty comparisons may not have any valid or meaningful baselines. Some concepts of uncertainty are not in accord with general common sense or your audiences' expectations; some uncertainty visualizations fail to convey the intended meaning or impression of your data, some uncertainty decisions are not clear for purposes of proper decisions, and it is not possible to reach an agreed ethical standpoint concerning the issue in the mind of some policymakers, given that they expect different outcomes anyway. Also, some uncertainty situations may not correspond to or may be relative to your purpose or context; some uncertainty perceptions may not be welcome or useful for your audience or stakeholders, and rightfully, some uncertainty manipulations may not be honest or requisite for your data and impact.

3.5.1 Framing Probabilities as Frequencies:

Framing probabilities in terms of frequencies is a highly effective means of facilitating the understanding of uncertainty, especially for non-experts. Rather than saying there was a 20% chance of rain, it might be relatively easier to frame it as a frequency, such as, "Out of 10 similar rainy days, it rained 2 days."

This lends itself to improved decision-making through the contextualization of uncertainty in terms of real-world experiences. Researchers have found that people understand probabilistic information much better framed in terms of actual occurrences than in terms of abstraction.

For example:

Before talking about uncertainty visualization, we must first define what uncertainty should exactly mean. We probably find it easy to intuitively grasp the concept of uncertainty in the context of future events. I flip the coin and wait for the result. The outcome is uncertain. But there is also what may be called uncertainty in terms of the past. If yesterday, I looked from my kitchen window exactly two times, namely at 8 am and again at 4 pm, and at 8 am saw a red car parked across the street but not at 4 pm, I know it must have left sometime in the eight-hour gap, but I don't know exactly at what time: 8:01, 9:30 am, 2:00 pm, or anytime during that eight-hour delay.

Mathematically, we embrace the notion of chances to deal with uncertainty. Definitions of probability are rather complex, and this discourse extends far beyond this book. However, we can successfully reason on probabilities without indulging in all mathematical complexities. In many practically relevant problems, discussing probabilities as relative frequencies suffices. Suppose you conduct a fair random trial, let's say flip a coin or roll a die, to check a particular outcome, heads, or roll a six. Suppose you treat this outcome as success and every other outcome as failure. Then the probability of success will, approximately, be given by the fraction of times you expect to see that outcome if you were to repeat this random trial artlessly many times. For example, if an outcome has a probability of 10%, that means that one could expect to see that result in about one case for every ten trials made. Framing probabilities in terms of frequencies helps with risk communication by giving audiences a handle on weighing various possible outcomes. It is particularly useful in places of application such as health care, finance, and weather forecasting where uncertainty lingers in decision-making [26].

Visualizing individual probabilities can be difficult. How does one visualize the winning chance in a lottery, or winning a fair die? In both cases, the probability is a single number. One could take that number as some kind of amount and show it by using any of the techniques discussed in Chapter 6-a bar graph, or a dot plot, to name just a few. But this would not be very helpful. Most people do not have an

intuitive understanding of how a probability value translates into experienced reality and showing that value as either a bar or a dot placed on a line won't assist that endeavour.

To make the concept of probability more concrete, we might develop a graph that combines a focus on the frequency aspect along with the random trial element's uncertainty, perhaps created by randomly arranging squares of different colors. I have employed this methodology, as shown in Figure 16.1, to visualize three separate probabilities: the probability of success of 1%, 10 %, and 40 %. To read this figure, one should imagine that you have been assigned to randomly choose a dark square without knowing beforehand which of the squares will be dark or light. (If you will, imagine you are to select a square with your eyes closed.) You are quite likely to understand that it is not probable to select the one dark square in the case of 1% chance as shown in fig 3.10. It is also not likely in the case of 10% chance to randomly choose a dark square. However, this would not look so bad in the case of a 40% chance. Discrete outcome visualization is the mode of visualization in which we show potential specific outcomes, whereas frequency framing is the visualization of a probability in the form of a frequency. We address the probabilistic nature of an outcome in such a way that perceived frequencies of outcomes are easily understood.

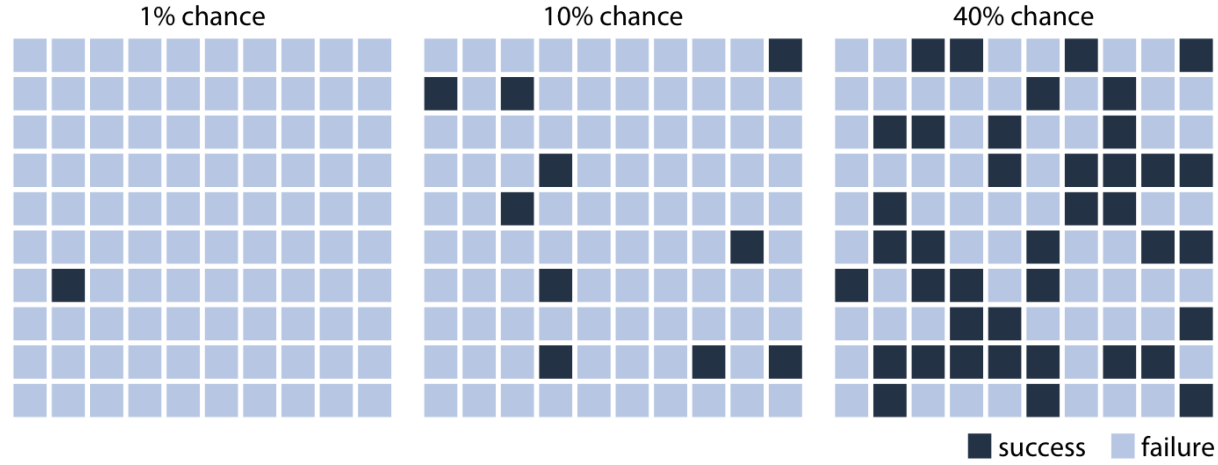


Fig. 3.10: Framing probabilities as frequencies

Visual techniques to illustrate frequency-based probabilities would include:

a. Dot Plots: Dot plots have the advantage of providing a direct visualization for presenting probabilities as frequency counts. They are arranged as individual dots to represent occurrences in the total. For instance, if the probability is said to be "20 out of 100," then the dot plot would show 100 dots with 20 of them depicted differently to symbolize the occurrence of the event.

Dot plots are useful to:

Compare probabilities: Present several categories side by side to compare probability.

Make probability interpretable: Make abstract probabilities concrete through visual representation.

Risk communication: Help people understand uncertainties in medical outcomes, financial risks, and weather forecasts.

Dot plots work well for very small datasets or categorical variables. They are very intuitive and make it easier to compare uncertainty and probability distributions.

Code:

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

```
# Probabilities and corresponding frequencies
probabilities = [0.1, 0.25, 0.5, 0.75, 0.9]
frequencies = (np.array(probabilities) * 100).astype(int)
# Dot plot
plt.scatter(probabilities, frequencies, color='red', s=100)
plt.xlabel('Probability')
plt.ylabel('Frequency (out of 100)')
plt.title('Dot Plot of Probabilities as Frequencies')
plt.grid(True)
plt.show()
```

Output of the code is shown in fig 3.11

Output:

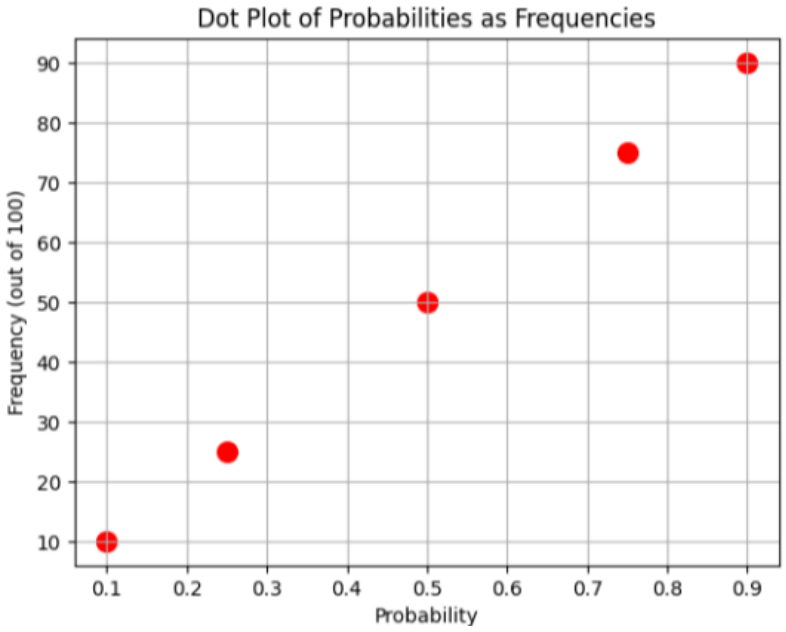


Fig. 3.11: Dot plot

b. Icon Arrays: Icon arrays represent proportions, using simple pictograms or icons. The designs help people grasp probabilities by providing a series of identical icons, where a part depicts the outcome of interest. For example, it might be an icon array with 100 figures to reflect a 30% probability, with 30 highlighted differently from the remaining 70.

Icon arrays fit the need for:

Communicating Medical Risks: The use of human icons with affected individuals rendered in different colors facilitates the understanding of conflicts where major risks are concerned.

Comparison of Proportions: This allows one to easily put the various probabilities side by side.

Simplification of Certain Situations: Using visual metaphors increases understanding with nontechnical audiences.

Icon arrays help bridge the abyss separating statistical uncertainty and human intuition by substituting vague numbers with vivid symbols.

Code:

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

```
def plot_icon_arrays(probabilities, size=10):
fig, axes = plt.subplots(1, len(probabilities), figsize=(len(probabilities) * 3, 3))
for ax, p in zip(axes, probabilities):
grid = np.random.rand(size, size) < p
ax.imshow(grid, cmap='gray', aspect='equal')
ax.set_xticks([])
ax.set_yticks([])
ax.set_title(f'{p*100:.0f}%')
plt.show()
```

Output of the code is shown in fig 3.12

Output:

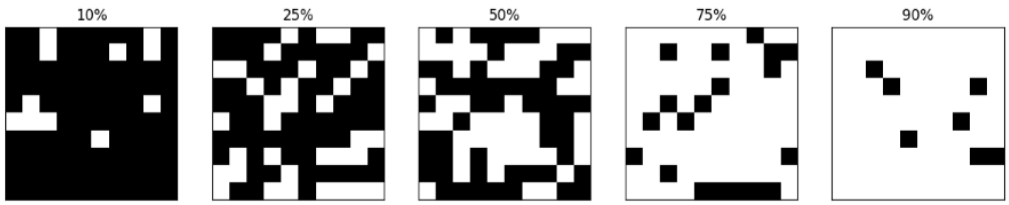


Fig. 3.12: Icon arrays

c. Histogram Bins: Histogram bins are the backbone of histograms representing the distribution of a dataset. A histogram is a series of bars to represent the distribution of data points into intervals, known as bins. The height of each bar reflects the number of observations within that bin.

How to Choose the Right Number of Bins?

Choosing the correct number of bins can represent the shape of the distribution most effectively. Methods to determine bin size include the following:

Sturges' Rule: It gives several bins. The value is calculated from the number of data points. Formula:

$$\mathbf{k = 1 + \log_2(n)}$$

Scott's Rule: It selects the bin width using the data variance as a criterion that is optimized for a normally distributed dataset.

Freedman-Diaconis Rule: It uses IQR to get bin width, so it is quite robust concerning skewed distributions.

Uses of Histogram Bins: It help the distribution analysis of data and incites those patterns from it.

Illuminating outlier values outside the main distribution.

Probabilistic density estimation provides intuition about the underlying probability distributions used for statistical modeling.

With careful choice of bin size, histograms proficiently summarize large datasets and disclose important trends.

Code:

```
import numpy as np
import matplotlib.pyplot as plt
def plot_icon_array(probabilities, size=10):
fig, axes = plt.subplots(1, len(probabilities), figsize=(len(probabilities) * 2, 2))
for ax, p in zip(axes, probabilities):
ax.imshow(np.random.rand(size, size) < p, cmap='gray')
```

```
ax.set_xticks([])
ax.set_yticks([])
ax.set_title(f'{p*100:.0f}%')
plt.show()
def plot_histogram(data, bins=5):
plt.hist(data, bins=bins, edgecolor='black')
plt.show()
plot_histogram(np.random.randn(100), bins=10)
```

Output of the code is shown in fig 3.13

Output:

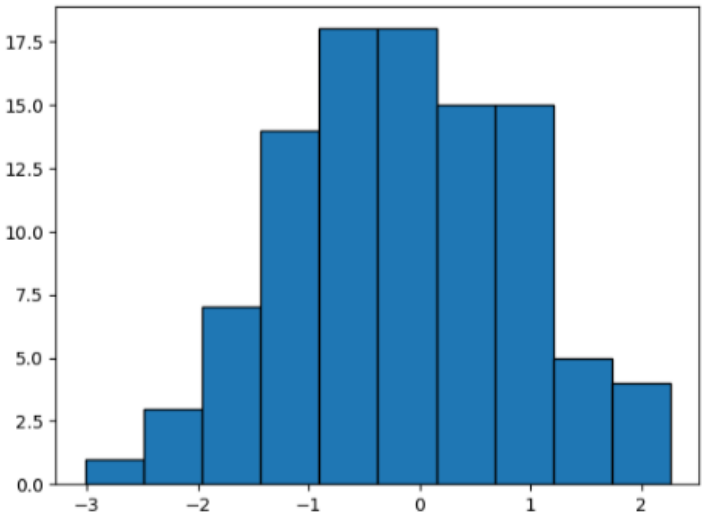


Fig. 3.13: Histogram bins

3.5.2 Visualizing the Uncertainty of Point Estimates:

Point estimates are single-value predictions based on sample data, and these are inherently subject to some uncertainty. The visual representation of this uncertainty is important to ensure the correct interpretation of results and to avoid misleading conclusions. A few key methods available for representing uncertainty in point estimates are the following [26]:

a. Confidence Intervals:

A confidence interval (CI) shows a range in which the actual value of a population parameter lies with a certain probability level (e.g., 95%) is explained in terms of sampling variability and describes the plausible range of values for the precision of estimation, and the controversy is attributed to the large interval which shows uncertainty.

An alternative way of saying this is instead of saying that the average income of the population is something like $50,000, you would say that you are 95% confident that it lies between $48,000 and $52,000. So, if you were to repeat the experiment many times, then you would expect 95% of them to yield intervals containing the true mean.

Code:

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats
def plot_confidence_interval(data, confidence=0.95):
```

```
mean, sem = np.mean(data), stats.sem(data)
margin = sem * stats.t.ppf((1 + confidence) / 2., len(data)-1)
plt.hist(data, bins=10, edgecolor='black', alpha=0.7)
plt.axvline(mean, color='red', linestyle='dashed', label='Mean')
plt.axvline(mean - margin, color='blue', linestyle='dashed', label=f'{confidence*100:.0f}% CI')
plt.axvline(mean + margin, color='blue', linestyle='dashed')
plt.legend()
plt.show()
```

Output of the code is shown in fig 3.14

Output:

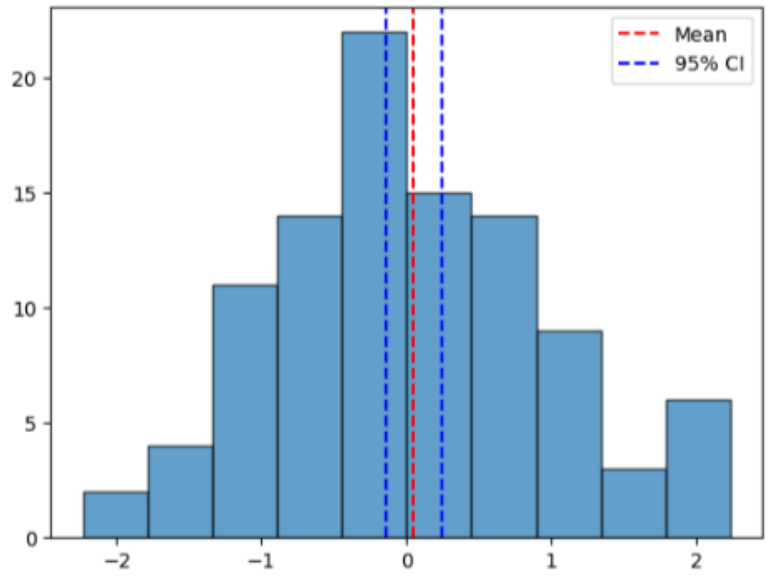


Fig. 3.14: Graph showing confidence interval

b. Error Bars:

Error bars are, in essence, graphical representations of variability in the data. They span from the estimated value to show the possible variations in measurement. They are often used in bar charts, line graphs, and scatterplots.

The types of error bars are:

Standard error (SE): This represents how much variability there is for the given estimate, given the random sampling nature of data collection.

Standard deviation (SD): This is the amount by which the data are dispersed; that is, how far different values fall from the mean value.

Confidence intervals (CI): A much wider interval that seems to embrace all potential values as discussed in the previous confidence interval method.

Code:

```
import numpy as np
import matplotlib.pyplot as plt
x = np.arange(5)
y = np.random.rand(5) * 10
errors = np.random.rand(5)
plt.errorbar(x, y, yerr=errors, fmt='o', capsize=5, color='blue')
```

plt.show()

Output of the code is shown in fig 3.15

Output:

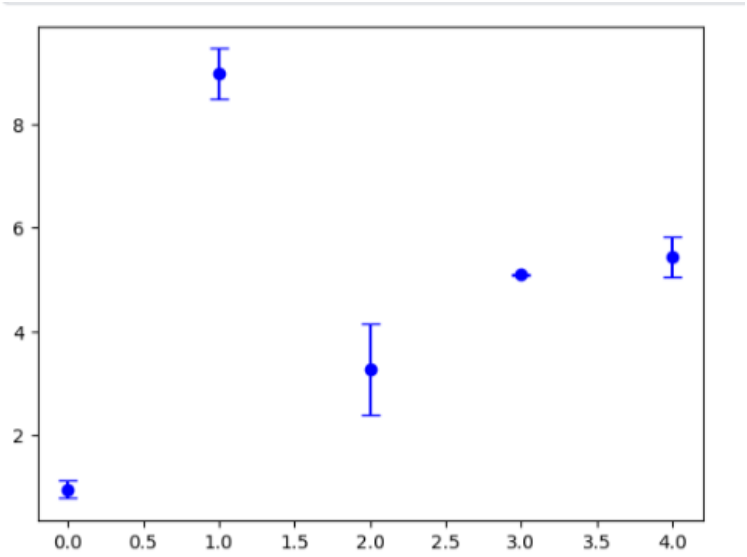


Fig. 3.15: Graph showing error bars

c. Violin Plots: Violin plots combine the boxplots and kernel density estimation to provide richer insights about the data distribution. It shows:

The median and interquartile range (IQR) (like box-prop).

The density of the data across different values, reveals distribution asymmetry and multimodal patterns.

Violin plots are particularly suited for the comparison of multiple datasets and for visualizing uncertainty.

Code:

```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
# Generate sample data
np.random.seed(42)
data = [np.random.normal(loc=mean, scale=1.0, size=100) for mean in [5, 6, 7]]
# Create a violin plot
plt.figure(figsize=(6,4))
sns.violinplot(data=data)
# Label the plot
plt.xlabel("Categories")
plt.ylabel("Values")
plt.title("Violin Plot Example")
# Show the plot
plt.show()
```

Output of the code is shown in fig 3.16

Output:

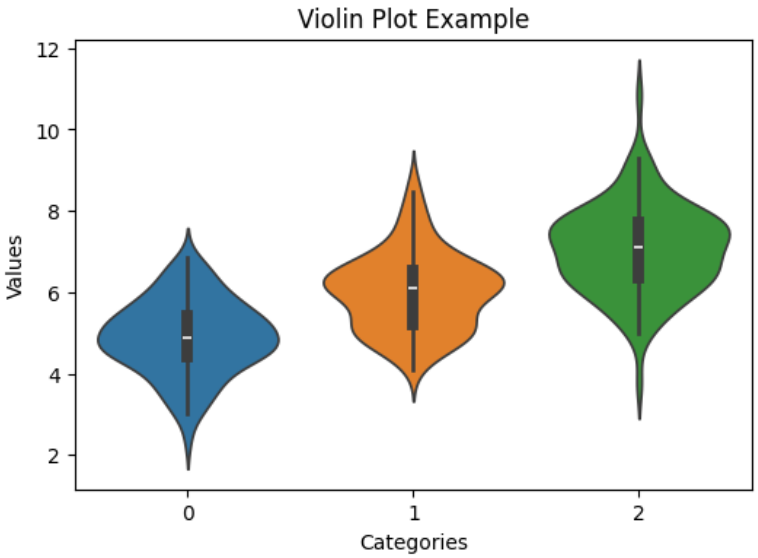


Fig. 3.16: Violin plot

3.5.3 Visualizing the Uncertainty of Curve Fits:

Curve fitting is a method of modelling the relationship between variables by an appropriate function on the recorded data points. All the fitted models have, however, some uncertainty due to noise, limited data, or wrong model assumptions. Visualization of this uncertainty is useful for establishing the reliability of models and therefore aids in decision-making [26].

a. Confidence and Prediction Intervals:

Confidence intervals estimate the interval which will include the true regression line. By using a confidence interval with a smaller width, it can be inferred that the model is more certain of the prediction.

Prediction intervals cover the range in which individual future observations are likely to be found. These are usually wider than confidence intervals due to added uncertainty regarding the model and variability in the data as shown in fig 3.17.

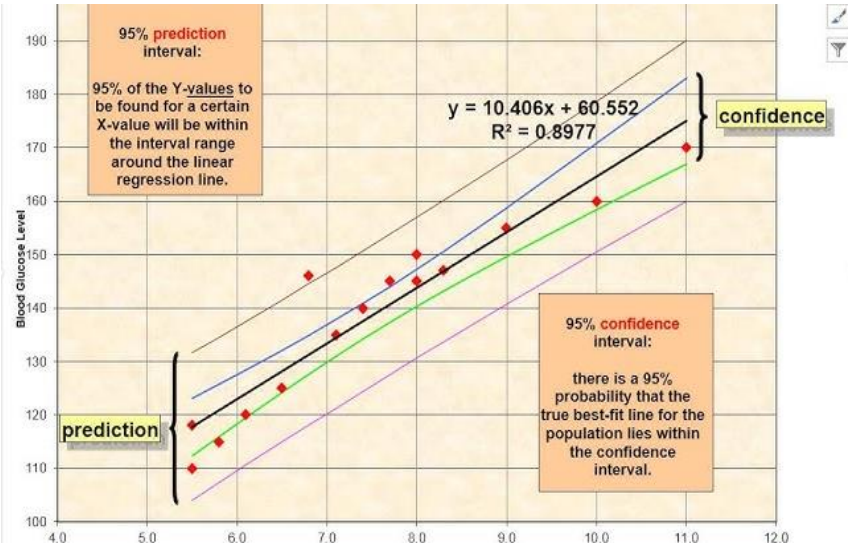


Fig. 3.17: The concepts of confidence intervals and prediction intervals are explained in the context of a linear regression model.

Here is a breakdown:

1. Regression Line (Black Line):

This is the best-fit line obtained by linear regression using the equation:

$$y=10.406x+60.552$$

It predicts the relationship between the independent variable (X-axis) and the dependent variable (Y-axis, "Blood Glucose Level").

2. Confidence Interval (Blue Lines):

The 95% confidence interval refers to the range that is expected to contain the true regression line 95% of the time.

Put differently, if the same experiment were repeated many times, then, 95% of the estimated best-fit lines would fall within the blue lines.

The confidence interval is smaller than the prediction interval because, for the former, there is only uncertainty in estimating the regression line and no variability in individual data points.

3. Prediction Interval (Purple Lines):

The prediction interval gives a region in which we an expect new individual data points to fall in 95% of the cases.

This interval is wider because it carries two sources of uncertainty:

- The uncertainty in estimating the regression line.
- The natural variability in the data (i.e., individual data points will be more spread out than just the mean trend).

4. Data Points (Red Dots):

These are the observed data points. Some fall within the confidence interval and some extend into the wider prediction interval.

3.6 Hypothetical Outcome Plots: Hypothetical Outcome Plots convey uncertainty visually in a plurality of possible outcomes instead of seeing uncertainty only numerically as with static properties like confidence intervals. Its way of allowing viewers to derive an intuition about variability and the results is beautiful.

Rather than showing uncertainty via error bars or confidence bands, HOPs allow for animation or moving sequential frames, where each frame editable presents a reasonable simulation of outcomes, based solely on the underlying probability distribution. Watching many such simulations gives the viewer a feel of possible oscillations and, importantly, which direction things are likely to go.

3.6.1 Advantage of HOPs:

Better Understanding of Uncertainty: By illustrating multiple plausible outcomes, they provide users with a better grip on the range of variation than a static representation.

More Engaging Visualization: Active animations capture attention and stimulate interaction with the data.

Useful for Decision-Making: Stakeholders can gauge how robust predictions are by how often extreme cases happen.

3.6.2 Applications of HOPs:

Forecasting and Risk Analysis: Using HOPs allows the illustration of uncertain scenarios that can be, for instance, a forecasting tool for financial markets, weather predictions, as well as, for medical prognosis.

Experimental Results Interpretation: Scientists or engineers can use HOPs to illustrate uncertainty in experimental data.

Machine Learning Model Evaluation: HOPs visualize the variability in the modeled predictions that can be attributed to randomness in the training data or tuning of parameters.

Generate Hypothetical Outcome Plots:

Simulate Multiple Outcomes: Take a sample from the probability distribution of interest.

Generate Consecutive Frames: Each frame represents a single plausible outcome.

Animate the Results: The sequential display of frames illustrates the range of uncertainty.

Interpret the Patterns: Observe variability and frequency of extreme outcomes.

Chapter 4

THE PRINCIPLE OF PROPORTIONAL INK IN DATA VISUALIZATION

Arshdeep Singh[1], Vikramjit Parmar[2] and Ramandeep Kaur[3]

[1,2,3]GNA University, Phagwara

4.1 Introduction to the Principle of Proportional Ink:

The **Principle of Proportional ink**, introduced by Edward Tufte in "*The Visual Display of Quantitative Information",* states that the magnitude of data should be proportional to the amount of ink used to represent the data [27].

Many graphic elements in various visualization scenarios represent data values. For example, draw a beam on a bar diagram that starts at 0 and ends with the data value that represents. In this case, the data values are coded not only at the end of the rod but also at the height or length of the rod. Drawing bars starting with different values as zeros conveys inconsistent information on the length of the rod and rod endpoints. Such numbers are internally inconsistent as they show two different values with the same graphic element. Compare this to a scenario in which data values are visualized at some point. In this case, the value is coded only at the point of the point, not the size or shape of the point. Using graphic elements such as bars, rectangles, and shaded areas in any form will cause similar issues. Or other elements with visual extensions defined that are compatible with the displayed data values consistently or compatible with the displayed data. In all these cases, you need to make sure there are no conflicts. This principle prevents false government and ensures clarity of appropriate goals and visual communication. This concept was called the principle of proportional ink [27].

In this chapter, we are going to discuss various visualization techniques that adhere to this principle, including visualizations along linear and logarithmic axes, direct area visualizations, and methods for handling overlapping data points.

Why it is important:

1. Avert Misleading Data explication:

If ink usage does not match the data values, readers can misinterpret the meaning and importance of a part icular data set.

Example: A simple pie chart where slices are proportional to the total without any unnecessary effects as illustrated in fig 4.1.

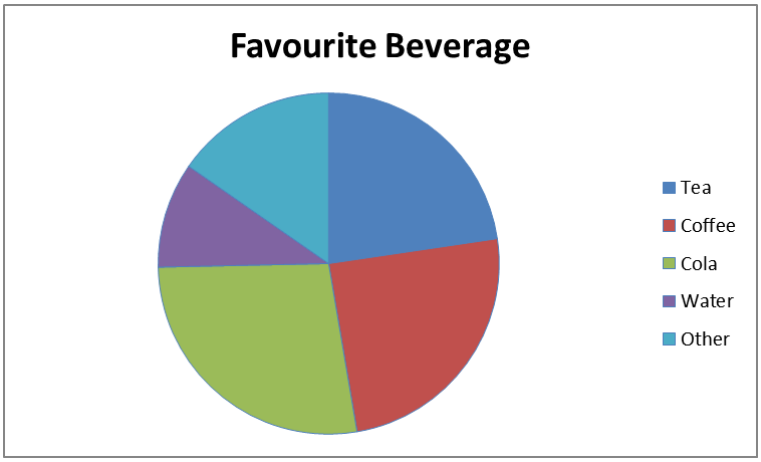


Fig. 4.1: Pie chart of favourite beverages

2. Enhances data readability and clarity: It is easy to comprehend the insights when the visualization is well-designed and it minimizes unnecessary ink usage. **Example:** Removing non-data ink such as extra lines, shading, and unnecessary labels makes the data clearer as shown in fig 4.2.

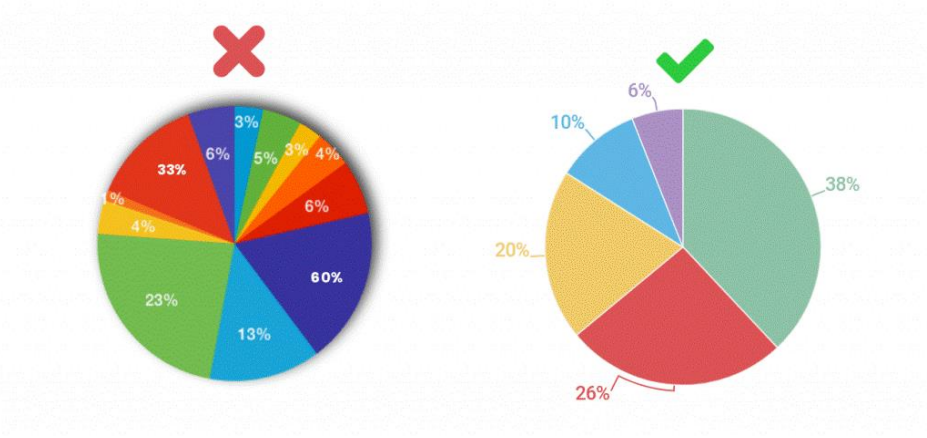


Fig. 4.2: Showing well defined pie chart

3.Maintains integrity: Data should be presented in such a way that it maintains its true meaning and prevents exaggeration or understatement. **Example:** A simple, thin-line graph which represents the changes in data accurately as shown in fig 4.3.

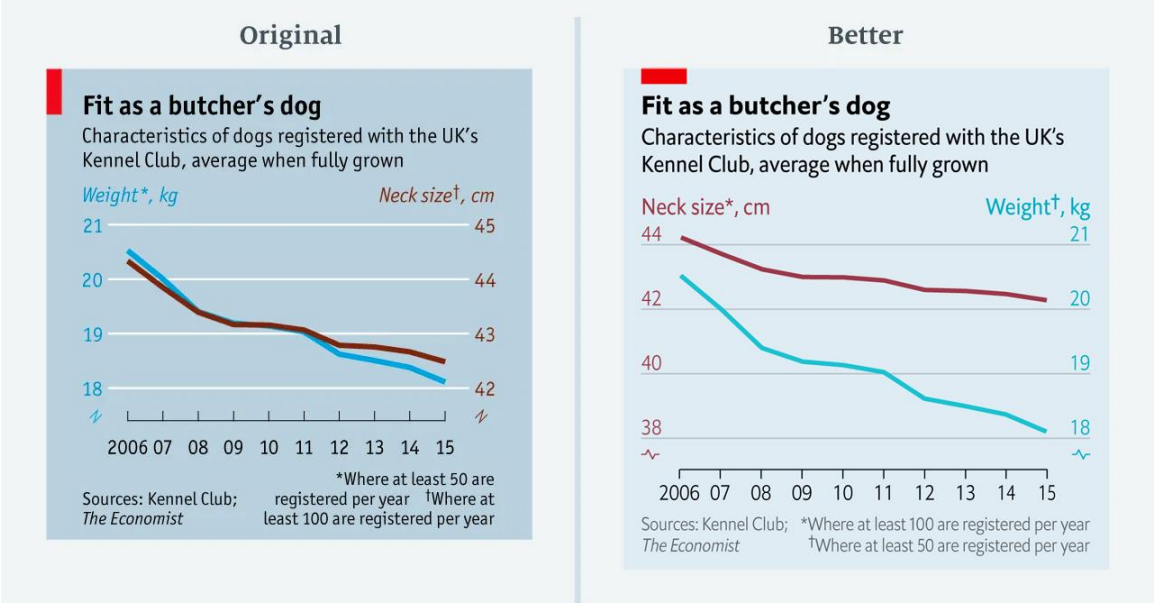


Fig. 4.3: Showing which graph is easy to understand

4.2 Key Aspects of Proportional Ink in Data Visualization:

Here are some of the key aspects mentioned below:

1.Link should be directly proportional to data values: The amount of link should directly correspond to the magnitude of data.

Example: A 2D bar chart where bar heights correctly correspond to data values as shown in the fig 4.4.

	India	US	UK	France
Potato	5	3	5	2
Tomato	11	2	6	8
Okra	5	7	8	6
Cucumber	3	5	7	8
Carrot	3	7	6	9
Cauliflower	2	4	6	8
Cabage	1	7	5	2

Fig. 4.4: Example showing 2d graphs

2.Avoid unnecessary visual elements: Charts should mainly focus on the data representation not on unnecessary decorations. **Example:** A simple, clean chart with only important elements is better than a chart with shadows and gradients, which don't add any value to the understanding as shown in the fig 4.5.

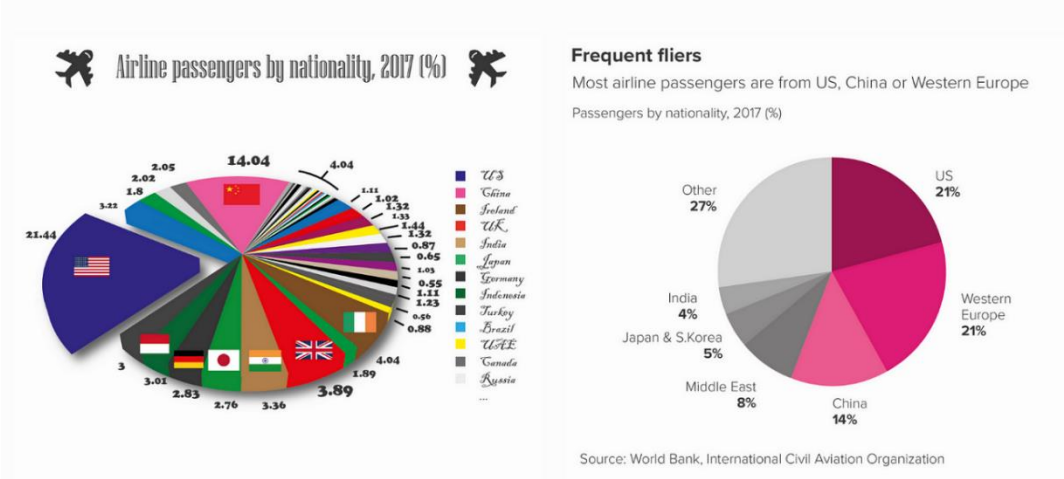


Fig. 4.5: Right way to draw a pie chart

3. Avoid distorting data representation: Some charts distort proportions which may lead to false comparisons. Charts should represent data accurately without any exaggeration. **Example:** Some 3D pie charts make some slices look bigger just due to perspective as compared to A flat pie chart where all slices reflect actual proportions [28].

4.3 Problems and Abuses of Proportional Link:

Although the Principle of Proportional Ink is so crucial, numerous visualizations in popular culture and research break it. Several abuses include:

a.) Truncated Axes: By beginning the y-axis at a non-zero value, a bar or line graph may inflate small variations, making trends look more spectacular than they do.

b.) Unnecessary 3D Effects: Three-dimensional bar charts and pie charts tend to twist proportions, making it more difficult for the audience to actually compare values.

- **Misleading Area Representations:** Misjudged circles, squares, or other forms can give a misleading impression of relative magnitudes.
- **Overuse of Decorative Elements:** Too much color, patterns, or backgrounds can hide data relationships instead of clarifying it.

To avoid such pitfalls, data visualization designers should aim for clarity and accuracy rather than aesthetic appeal. By observing best practices and sticking to the Principle of Proportional Ink, they can ensure that visual data representations provide their intended value without deceiving the audience [28].

4.4 Virtualization along linear axis:

Linear axes are most used to represent the data in case of dealing with quantities that have a proportional relationship. When the Principle of Proportional Ink is applied to the linear axis visualizations:

i. The height of bars in a bar chart should be directly proportional to the values they represent.
ii. There should be proportional distances maintained between points in the Line Charts to ensure accuracy.
iii. Uniform spacing should be maintained In the Scatter Plots to accurately represent relationships.
iv. Ensuring proper scaling.
v. Avoiding unnecessary embellishments helps preserve the integrity of the data visualization.

i. Bar Charts:

Bar charts are often used to compare categories or groups. The length or height of the graph represents the value of the category as shown in the fig 4.6, and the bar must be proportional to accurately reflect the differences.

Example:

Imagine a bar diagram showing the populations of different cities:

City A: 5,00,000

City B: 10,00,000

City C: 15,00,000

Imagine a bar diagram showing the populations of different cities:

City A: 500,000

City B: 1,000,000

City C: 1,500,000

City C bar is three times the bar of City A and City B is twice the length of City A correctly representing the populations of the cities.

Common pitfall: If the Y-axis starts at 40000, the visual difference between A and B appears to be more important than it is. So always launch the y-axis with 0 and make sure bar length is proportional to the data value.

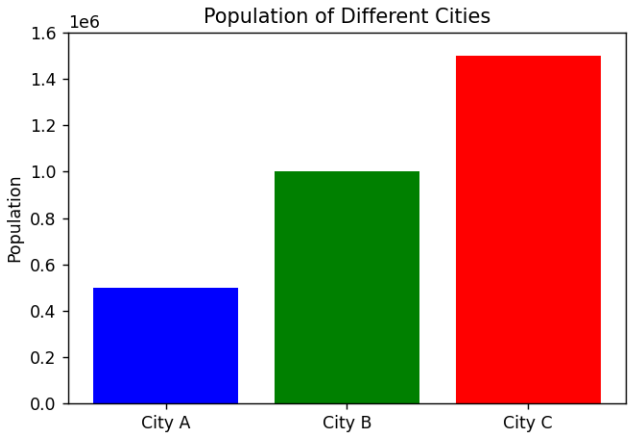


Fig. 4.6: Bar graph

Benefits of Bar Charts

1. Simple to Read & Understand: Bar charts employ rectangular bars of varying heights, making visual comparison of data simple. **Example:** A bar chart representing the number of students in various classes (Class A, Class B, Class C) easily indicates which class contains the largest number of students.

2. Suits Comparisons Well: Bar charts clearly highlight differences for comparing several categories. **Example:** A bar chart of five products' sales data enables a company to identify which product performs best.

3. Supports Large Data Sets: Bar charts can display a large variety of values, so they support both small and large data sets. **Example:** A bar chart of 10 countries' population sizes can be viewed and comprehended at one glance.

4. Supports Positive & Negative Values: A bar chart can indicate gains and losses by employing bars that go above and below the x-axis. **Example:** A firm's profit and loss over a number of months can be represented by bars going up (profit) or down (loss).

5. Flexible (Can Be Vertical or Horizontal): Bar charts can be oriented vertically or horizontally to suit the space available and make them more readable. **Example:** A horizontal bar chart can be substituted in place of a vertical one when category names are too long.

Limitations of Bar Charts

1. Can Get Cluttered with Too Many Categories: If there are too many bars, the chart becomes messy and difficult to read.

Example: A bar chart showing the sales comparison of 50 various types of shoes would be confusing and impossible to understand.

2. Not Suitable for Continuous Data: Bar charts are best used for categorical data, not continuous data such as temperature or time trends.

Example: A bar chart would not be ideal to illustrate how temperature changes over the course of a day (use a line chart instead).

3. Can Be Deceptive if Scales Are Manipulated: When the y-axis does not begin at zero or when the bars vary in width, the chart will mislead the audience.

Example: A bar chart of company profits could overstate differences if the y-axis begins at 50 rather than 0.

4. Difficulty in Comparing Complex Data Sets: When data sets have more than one variable, bar charts get complex and difficult to understand.

Example: A bar chart comparing the revenue, profit, and expenses of 10 companies over various years may be too complex.

5. Limited for Showing Trends Over Time: Whereas bar graphs can indicate movement over time, line graphs tend to be most effective at picking up trends.

Example: The use of a line graph over a bar graph would be ideal for illustrating how monthly temperatures change over one year.

ii. Line Charts:

Line charts are suitable for representing the data over continuous intervals such as time. These data points are connected by lines, and the position of each data point along the y-axis corresponds to its value. These are best to display trends over time.

Example: A line chart showing a company's income figures over five years illustrated in the fig 4.7 :

- Year 1: $200,000
- Year 2: $400,000
- Year 3: $600,000
- Year 4: $800,000
- Year 5: $1,000,000

The vertical positions of the data points must correspond proportional to the income figures, which results in the same vertical distance as an increase in revenue. To maintain proportionality and avoid misleading representations, it's advisable to start the y-axis at zero otherwise it can exaggerate the perceived growth or decline.

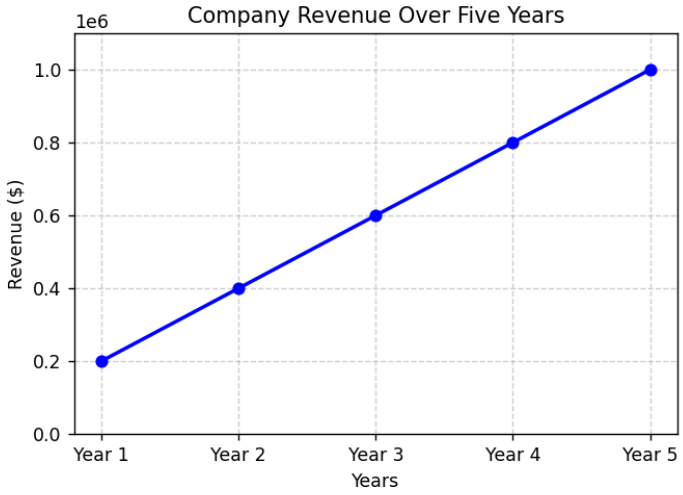


Fig. 4.7: Line chart

Advantages of Line Charts

1. **Best for Showing Trends Over Time:**
   - Line charts are great for displaying how data changes over time. Unlike bar charts, which show individual values, line charts connect data points to reveal overall trends.
   - **Example:** Suppose you track daily temperatures for a month. A line chart will show a clear pattern of how temperatures rise and fall over time, helping you identify warming and cooling trends.
2. **Easy to Identify Patterns and Relationships:**
   - Since line charts show continuous data points connected by lines, they make it easy to see upward and downward trends, cycles, or sudden changes.
   - **Example:** A business owner tracking sales over a year can easily spot seasonal peaks, like increased sales in December due to holiday shopping, and dips in other months.
3. **Useful for Comparing Multiple Data Sets:**
   - Line charts can include multiple lines, allowing easy comparisons between different data sets on the same graph.
   - **Example***:* A fitness coach tracking the weight loss progress of three clients can use a line chart with three different lines, showing who is losing weight faster and who needs more support.
4. **Takes Up Less Space Than Other Charts:**
   - Unlike bar charts, which require a separate bar for each data point, line charts use a single line to represent a dataset, making them more compact and efficient.

- **Example:** If a company wants to show how its revenue, expenses, and profit have changed over five years, a single line chart can display all three trends clearly, whereas using a bar chart would require a lot more space.

5. **Can Show Small Changes Clearly:**

- Because line charts use a continuous line rather than separate bars, even small fluctuations in data are visible.
- **Example***:* If a scientist tracks the daily growth of a plant, a line chart will show even the slightest changes, while a bar chart might not represent small variations as effectively.

Disadvantages of Line Charts

1. **Not Ideal for Large Data Sets with Too Many Data Points:**
   - If a line chart has too many data points, it can become cluttered and difficult to read. When there are too many overlapping lines, the chart loses its effectiveness.
   - **Example***:* If you try to plot the daily stock prices of 50 companies over a year on a single chart, the lines will overlap, making it impossible to distinguish individual trends.
2. **Can Be Misleading if Scales Are Not Properly Set:**
   - If the vertical axis (y-axis) scale is adjusted incorrectly, a line chart can exaggerate or minimize changes in data, leading to misinterpretation.
   - **Example:** If a company shows sales growth using a line chart but starts the y-axis at a high number instead of zero, it might make a small increase in sales look much larger than it actually is.
3. **Not Suitable for Showing Exact Values:**
   - Since line charts focus on overall trends rather than individual data points, they are not the best choice when you need to highlight exact values.
   - **Example:** If a teacher wants to compare the test scores of students in different subjects, a bar chart would be better than a line chart, as it clearly displays individual scores.
4. **Difficult to Interpret When Data Fluctuates Too Much:**
   - If data points have frequent sharp rises and drops, the line may look chaotic and hard to analyze. This can make it difficult to determine if there is a meaningful trend.
   - **Example:** If you track the number of website visitors every minute throughout a day, the line may go up and down rapidly, making it hard to identify useful trends.
5. **Limited Use for Categorical Data:**
   - Line charts are best for continuous data and are not suitable for categories that do not follow a natural order.
   - **Example:** If a company wants to compare customer satisfaction scores for five different products, a bar chart would be better than a line chart because customer ratings do not follow a continuous progression like time or temperature [29].

iii. Scatter plots:

Scatter plots represent the relationship between two continuous variables, Where each point's position along the x and y axes represents its values for the two variables by plotting data points on a Cartesian plane.

Example: A scatter plot (fig 4.9) illustrating the relationship between exam scores and hours studied:

X-axis: Represents hours studied

Y-axis: Represents exam Score

Each point represents hours of study and the corresponding exam score of a student. The pattern and the distribution of points can reveal correlations, trends, or outliers.

Common Pitfall: There can be chances of overplotting when the data points overlap each other, which can make it difficult to understand the data distribution. In the case of overlapping techniques, such as jittering (adding small random variations) can be employed to improve clarity.

Advantages of Scatter Plots:

1. **Excellent to Identify:**

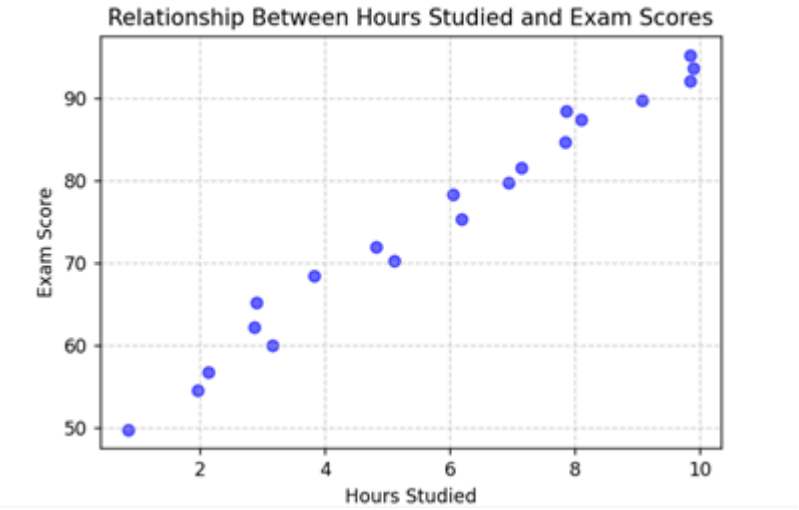


Fig. 4.8: Scatter plot

2. **Relationships between Two Variables:**

- Scatter plots assist in displaying if two variables are positively related, negatively related, or not related at all.
- **Example:** A teacher might use a scatter plot to illustrate students' study time vs. exam scores. If the points trend upwards, it indicates more studying results in better scores.

2. Displays Patterns and Trends Clearly:

- Scatter plots assist in identifying patterns, including clusters, linear trends, or exponential increases.
- **Example:** A marketing department monitoring advertising expenditures vs. customer buys can observe whether increased spending actually translates to increased sales. If points are all over the place, it indicates ads aren't affecting sales [29].

3. Assists in Identifying Outliers:

- Outliers are data points that do not conform to the overall pattern and may reflect errors or exceptional cases.
- **Example:** A doctor researching patient age versus blood pressure may discover an 18-year-old with very high blood pressure, which may be a sign of a rare condition to look into.

4. Can Handle Large Data Sets Without Cluttering:

- Scatter plots can show thousands of data points without getting too cluttered, unlike bar charts or line charts.

- **Example:** A scientist tracking global temperatures over the last 100 years can plot thousands of data points to identify long-term climate trends.

5. Can Show Non-Linear Relationships:

- Scatter plots are not limited to straight-line trends; they also reveal curved, exponential, or other complex relationships between variables.
- **Example:** In economics, the relationship between tax rates and revenue might not be a simple straight line but a curve showing diminishing returns.

Disadvantages of Scatter Plots

1. Cannot Display Exact Data Values Conveniently:

- Because every point only displays one value, scatter plots cannot display exact numbers like bar charts or tables.
- **Example:** If a company wishes to compare sales per month between various years, a bar chart may be more suitable because it displays exact values more conveniently.

2. Does Not Work Well with Categorical Data:

- Scatter plots need numerical data for both axes, so they are not a good choice for categories such as "Male vs. Female" or "Product Type A vs. Product Type B."
- **Example:** If a store wishes to compare customer satisfaction ratings for five brands, a bar chart would be preferable to a scatter plot.

3. Hard to Interpret When Data Points Overlap Too Much:

- If too many points are on top of each other, it is difficult to discern separate data points and trends.
- Example: A hospital plotting patient weight against cholesterol might have many data points piled on top of each other, making it difficult to interpret. Jittering (shifting points slightly) or transparency can remedy this.

2. Does Not Always Prove Causation:

- Scatter plots indicate correlation but not causation—just because two variables vary together doesn't mean one causes the other.
- **Example:** A study could indicate that ice cream sales and drowning rates go up at the same time, but it does not imply that eating ice cream leads to drowning. The actual reason is that both occur more during the summer.

3. Can Be Misleading if Scales Are Not Chosen Properly:

If the scales on the axes are adjusted, the correlation between variables can seem stronger or weaker than it is.

Example: A business might modify the scale on a scatter graph of ad spending to sales such that the graph could appear to suggest that spending a bit more results in a dramatic increase in sales even though the actual effect is small.

4.5 Visualization along logarithmic axes:

Logarithmic axes are practical for showcasing data that spans multiple orders of magnitude. When a logarithmic scale is used, the spacing is determined by a logarithmic function rather than a linear one between the values. In a logarithmic scale, each unit increase on the axis corresponds to a multiplication of a constant factor (commonly 10) of the previous value. For example, the intervals can be 1, 10, 100, 1,000, and so on illustrated in fig 4.9. This scaling is advantageous when processing data covering a wide

range, as it compresses larger values and expands smaller ones, which allows for more comprehensive visualization [29].

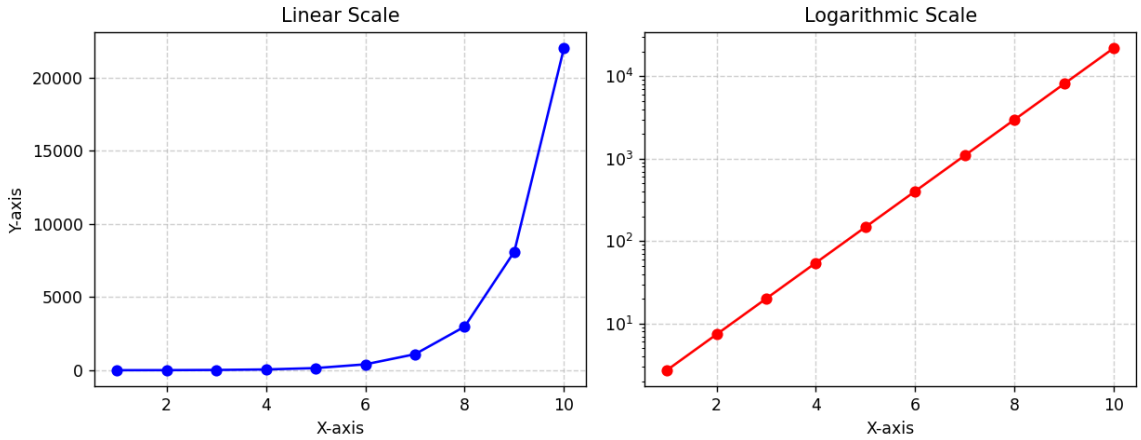


Fig. 4.9: Showing linear & logarithmic scale

To maintain the Principle of Proportional Link in logarithmic visualizations:

Must ensure that the proportionality is maintained about the logarithmic scale as shown in fig 4.10.

Fig. 4.10: Logarithmic visualization

Description: This line chart demonstrates how logarithmic scales can effectively represent data that spans multiple orders of magnitude, adhering to the Principle of Proportional Ink.

- Do not use linear-scaled visuals as sometimes they can be misleading.
- It indicates the log axis to prevent inappropriate interpretation.

Log visualization is usually used in areas that indicate index growth, frequency distribution and areas such as finance, physics, biology etc.

Applications of logarithmic scales:

1. Financial Data: Stock prices and market indices often record exponential growth or decline. Plotting these on a logarithmic scale can show changes in percentage over time, providing clearer insights into growth patterns.

Example: Visualizing the growth of an investment portfolio over decades. On a linear scale, early growth appears insignificant compared to the youngest. However, the logarithmic scale reveals consistent growth rates throughout the period as depicted in fig 4.11.

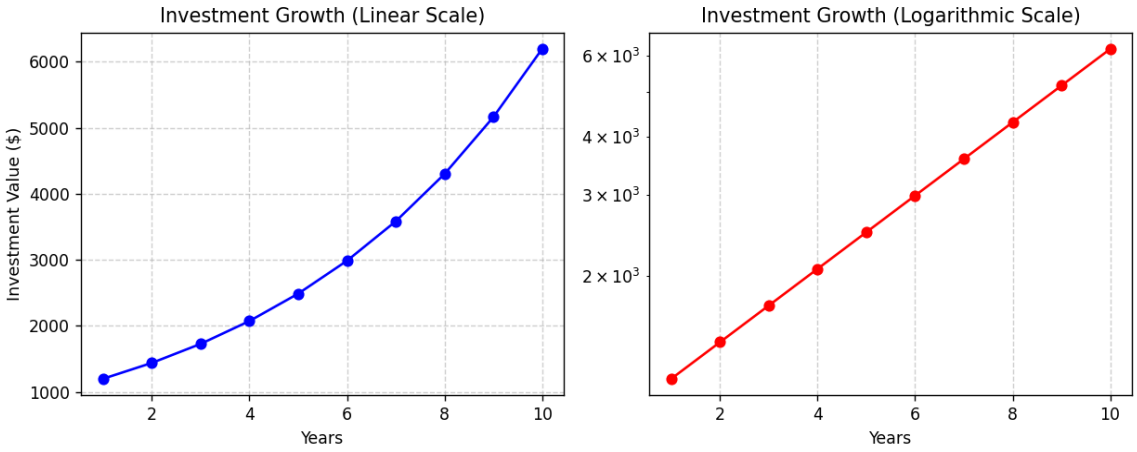


Fig. 4.11: Graphs for finance

2. Scientific Data: Measurements in fields such as astronomy, geology, and physics often include a wide range of areas. Logarithmic scales are used to present sizes such as seismic size (judge scale) or sound intensity (decibels).

Example: A logarithmic scale on the magnitude axis allows for a clearer representation of the distribution of both small and large earthquakes as shown in fig 4.12.

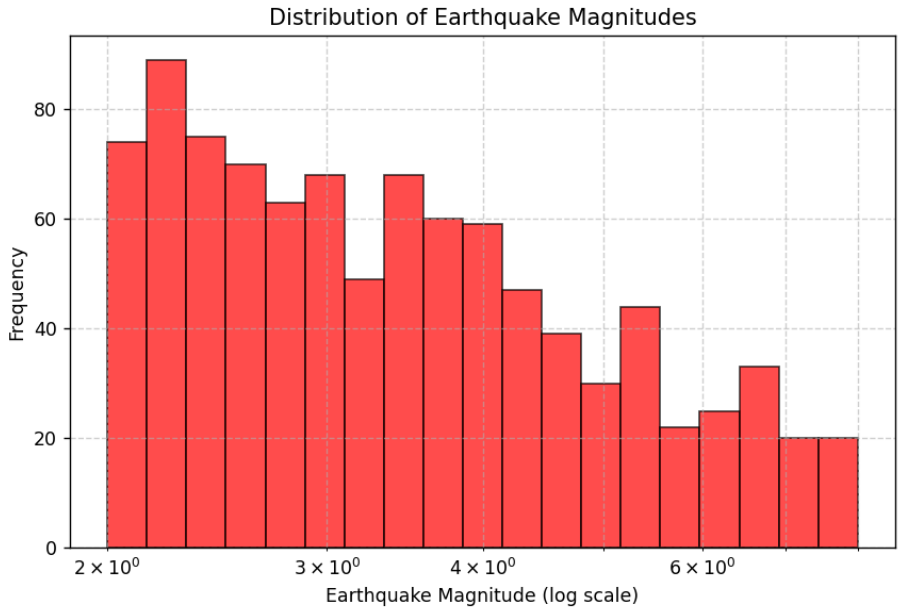


Fig. 4.12: Logarithmic scale for scientific data

3. Biological Data: In biology, phenomena such as bacterial growth or the spreading of diseases can become exponential. Logarithmic scales effectively make these rapid changes to visualization.

Example: At a linear level, the early stages of growth may appear negligible, but the logarithmic scales emphasize the exponential increases more clearly as illustrated in fig 4.13.

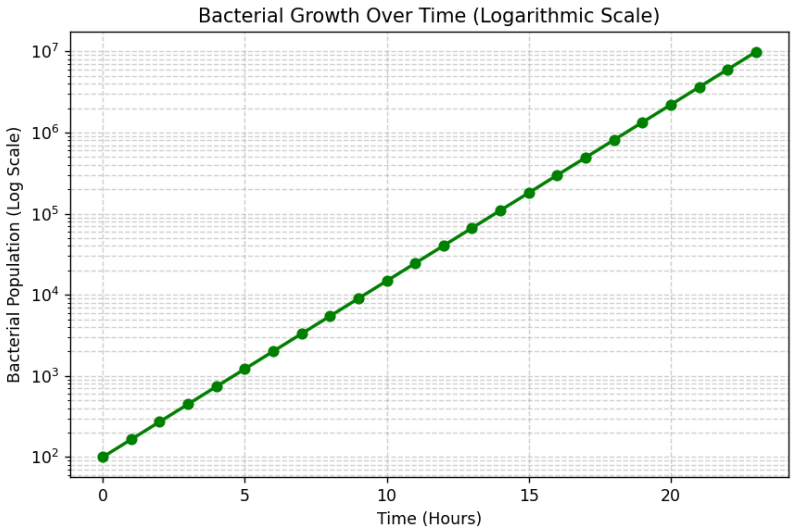


Fig. 4.13: graph showing biological data

4. Audio and Sound Measurement:

The decibel (dB) scale is a logarithmic scale for measuring sound intensity because the human ear responds to loudness logarithmically, not linearly.

This enables a large range of sound levels, from a whisper to a jet engine, to be represented usefully.

Example: A rock concert at 120 dB is not as loud as a conversation at 60 dB but a million times more intense as a pressure wave. Logarithmic scaling renders these differences intuitive as shown in fig 4.14.

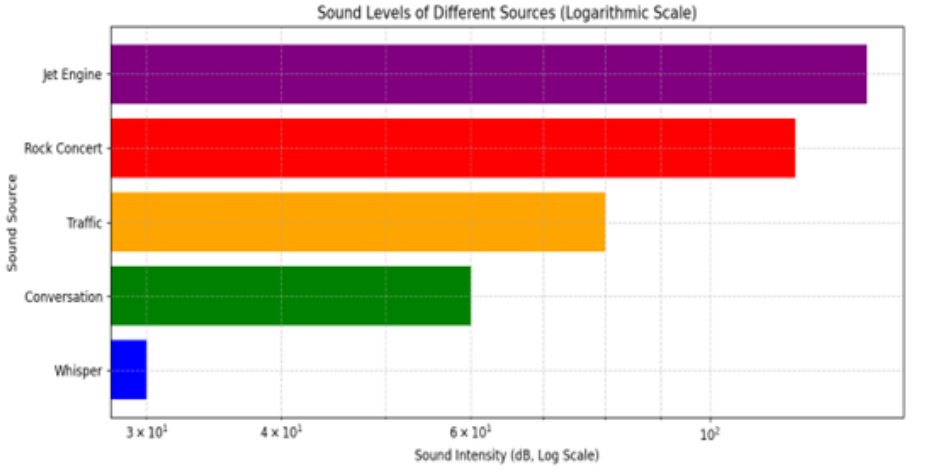


Fig. 4.14: Logarithmic scale for sound measurement

5. pH Scale in Chemistry:

The pH scale for measuring acidity or alkalinity of a solution is logarithmic and each unit represents a tenfold increase in hydrogen ion concentration.

This scale aids in distinguishing weak from strong acids or bases.

Example:

Household ammonia has a pH of 11, and lemon juice has a pH of 2. Ammonia is not slightly less acidic; it is 100 million times less acidic than lemon juice as shown in fig 4.15. The use of the logarithmic scale facilitates the perception of this drastic difference.

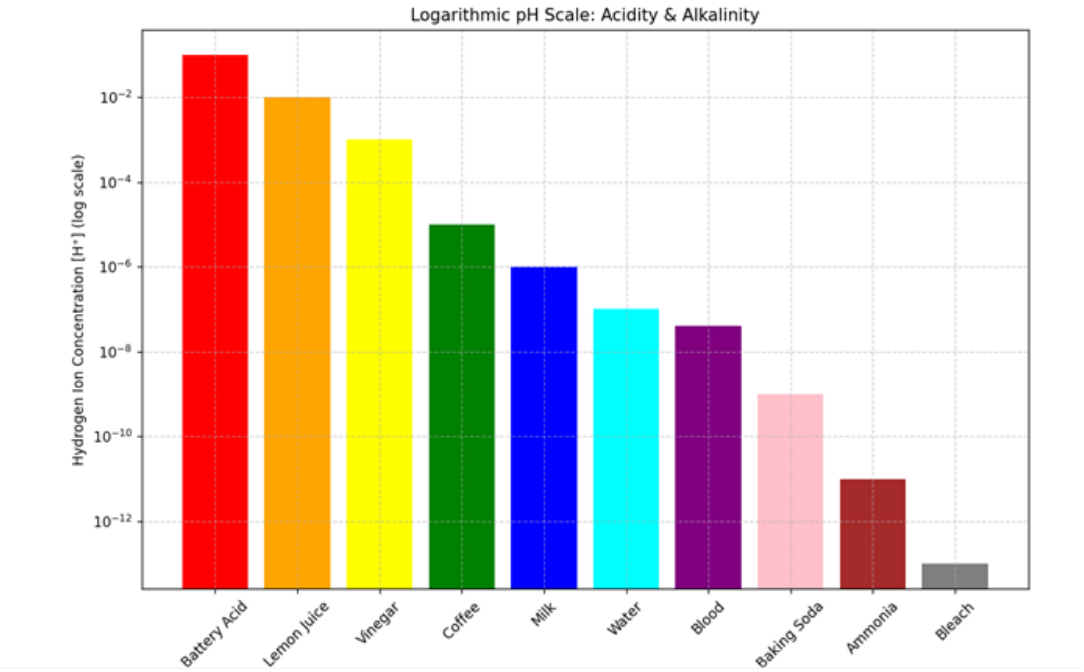


Fig. 4.15: Bar graph using logarithmic scale

4.6 Direct Area Visualization:

Direct area visualizations, including bubble charts and tree maps, represent data values with the help of areas rather than lengths as illustrated in fig 4.16. This approach may influence human insight of area to convey quantitative information effectively, sticking to the Principle of Proportional Ink. To adhere to the Principle of Proportional Ink [30] :

- Data values represented should be directly proportional to the area of each graphical element.
- Avoid misleading distortions, such as stretching or skewing.
- To ensure accurate representation, proper scaling methods such as square root scaling for circles should be used.

Months	Percentage of sales	Quarter	Sales(in billion)
April	0.10	4	90.20
August	0.65	3	27.50
December	0.05	3	92.00
February	0.50	4	75.30
January	0.80	2	65.80
July	0.35	5	33.90
June	0.55	8	53.30
March	0.40	6	80.50
May	0.75	7	42.30
November	0.55	9	89.90
October	0.50	8	73.30
September	0.30	6	65.20
Total	**5.50**	**65**	**789.20**

Fig. 4.16: Direct area visualization

Common types of direct area visualizations include area charts, bubble charts, and tree maps.

i. Area Charts: The area chart provides graphical representations of quantitative data. This fills the area between the axis and the line to display the volume.

Example: The area charts can explain how direct, forwarding social media contributes to the total visits that emphasize trends and percentages as illustrated in fig 4.17

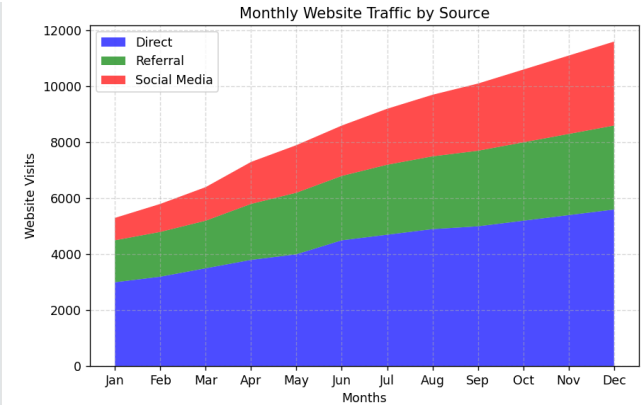


Fig. 4.17: Area chart

Applications of area chart:

1. Financial and stock market analysis:

Utilized to present cumulative revenue, market trends, and stock prices over time.

Example: Trends in a company's stock prices as shown in fig 4.18

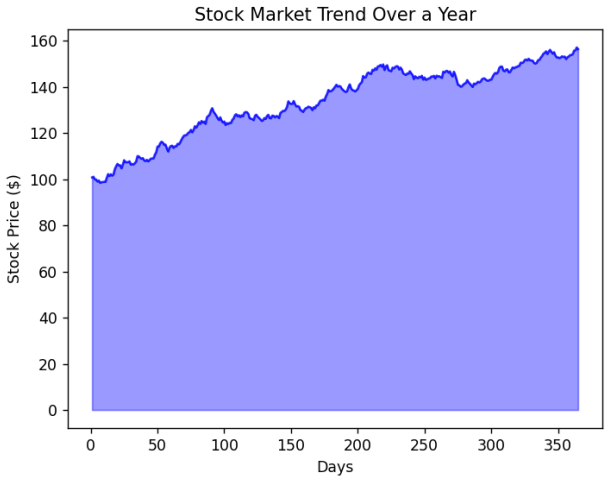


Fig. 4.18: Area chart for stock analysis

1. Environment and climate change data:

Monitors temperature fluctuations, pollution levels, and consumption of resources over some time.

Example: Global temperature rise over a year as depicted in fig 4.19.

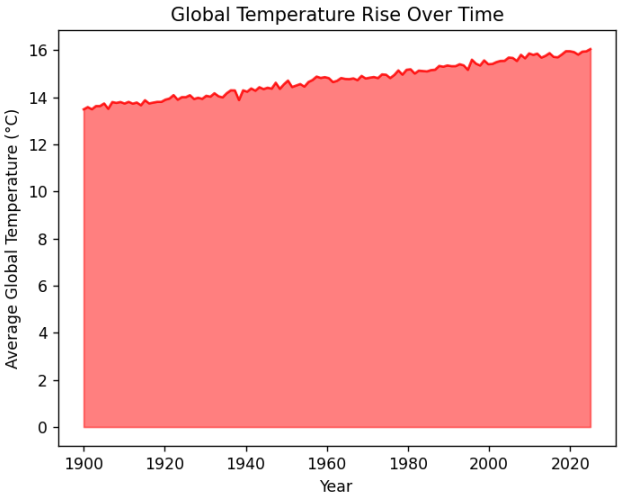


Fig. 4.19: area chart for global temperature measurement

3. Economic Growth and country representation:

Bubble charts give insights into GDP, population, and economic growth over time.

Example: Comparing countries based on GDP, population, and literacy rate as shown in fig 4.20.

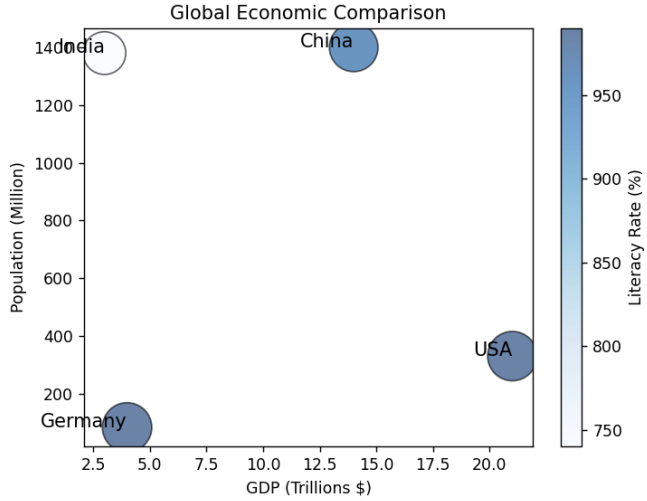


Fig. 4.20: Bubble chart showing economic growth

4. Environmental Science and Climate Change:

Used in monitoring levels of pollution, carbon emission, and temperature increase worldwide.

Example: A comparison of carbon emissions, air quality index (AQI), and forest cover as shown in fig 4.21.

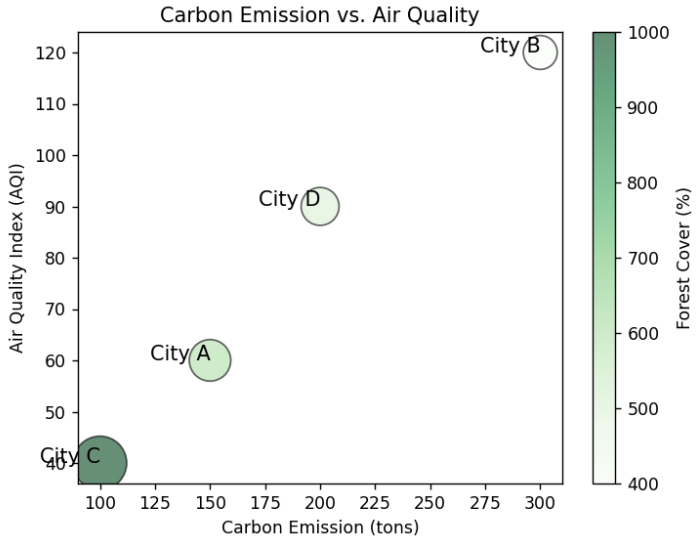


Fig. 4.21: Bubble chart showing Climate changes

ii. Bubble Charts

Bubble charts represent data with circles (bubbles), where the position refers to two variables, and the area (or diameter) of the bubble represents a third variable. This type of visualization is effective in simultaneously displaying relationships between three variables.

Example: Analysis of sales services, in which each bubble represents a product. The X-axis shows the number of units sold, the Y-axis displays the profit margin, and the bubble size shows the total turnover as illustrated in fig 4.22.

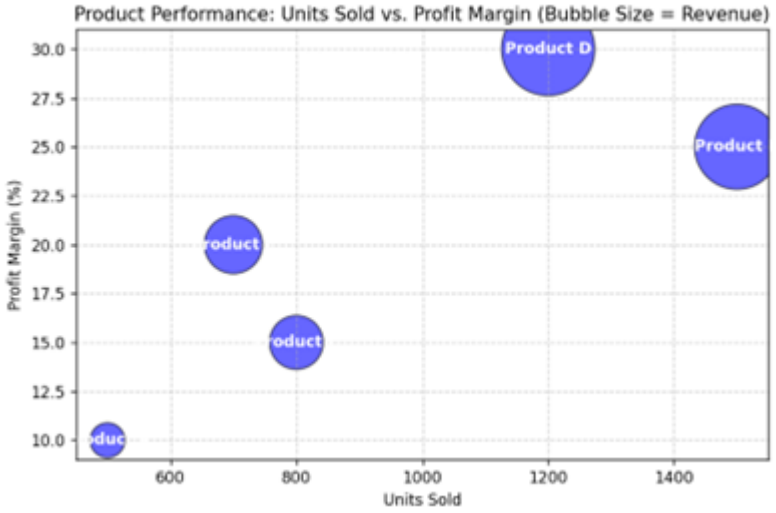


Fig. 4.22: Bubble chart

APPLICATIONS OF BUBBLE CHART:

1. Business and market analysis:

Bubble charts are used extensively in business intelligence to compare market trends, financial information, and performance metrics.

Example: Comparing company revenue, employee count, and market share as illustrated in fig 4.23.

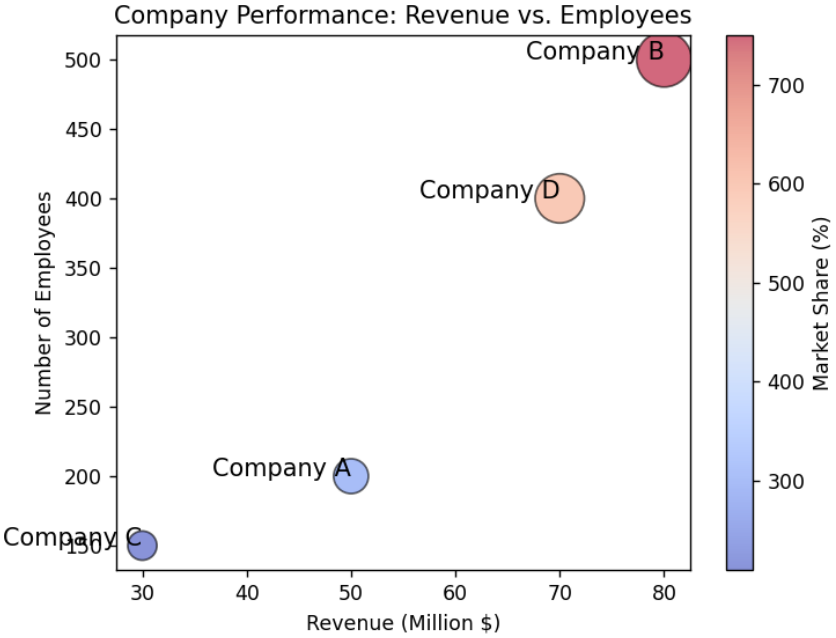


Fig. 4.23: Bubble chart showing company performance

2. Healthcare and epidemiology:

Bubble charts visualize disease transmission, population health indicators, or healthcare resource allocation.

Example: COVID-19 case mapping, hospital bed capacity, and death rate as shown in fig 4.24.

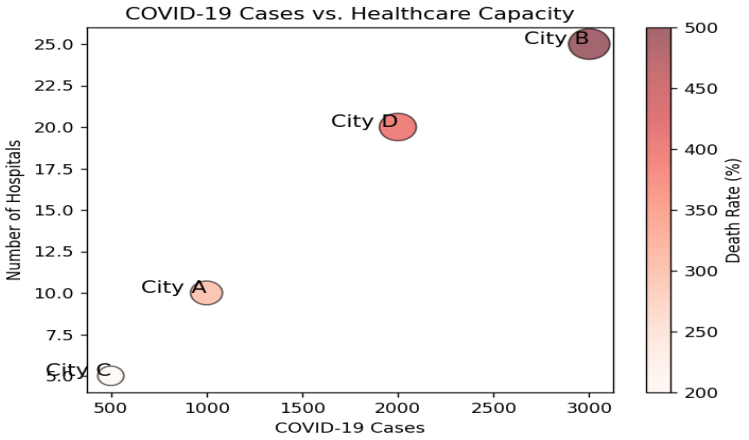


Fig. 4.24: Bubble chart showing COVID-19 cases vs healthcare capacity

Applications of Direct Area Visualization:

1. Population Distribution and Density Maps:

Direct area visualization is extensively applied to map population distribution over areas. Rather than applying for mere numbers, choropleth maps or bubble maps show areas of varying colors or sizes to depict population density [30].

Example: A world map with the population of nations where bigger circles signifies more populous cities such as New York, Tokyo, and Delhi, and smaller circles signify less populous areas.

2. Tree maps of Financial Data and Budget Analysis:

Tree maps are a form of hierarchical data in which the size of every block is equal to a given value. It is helpful in financial analysis when firms or industries are represented based on their revenue or market capitalization.

Example: Stock market tree map with each company's market value represented as a rectangle, with the larger rectangles used for large firms such as Apple or Microsoft and the smaller rectangles for less recognized stocks.

3. Infographics and Data Journalism:

Newspapers, magazines, and the internet frequently make use of area-based visualizations to simplify difficult data. These are commonly found in election results, income distribution graphs, and economic reports.

Example: An election results infographic in which the area of each state is proportionally sized according to the number of electoral votes as shown in fig 4.25, so it is visually obvious how various states contribute to the outcome.

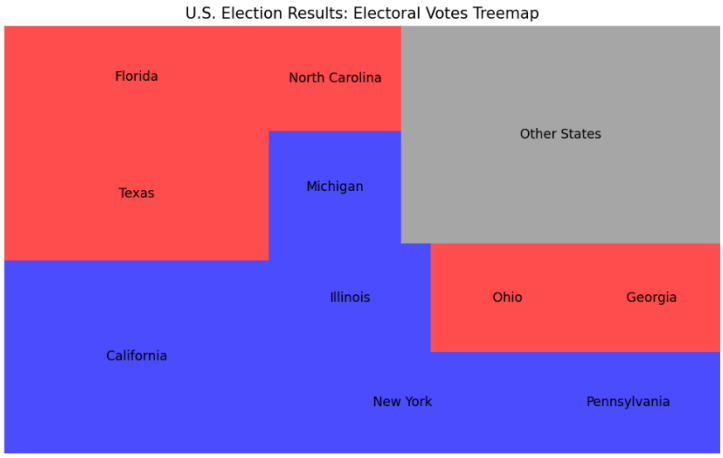


Fig. 4.25: Tree map showing election results

4. Bubble Charts for Comparing Multiple Variables:

Bubble charts present three dimensions of data: X-axis, Y-axis, and bubble size. They are frequently employed in economics, healthcare, and environmental science to illustrate relationships between various variables.

Example: Global health bubble chart with each nation appearing as a bubble; GDP per capita on the x-axis, life expectancy on the y-axis, and bubble size for population.

5. Epidemiology and Disease Outbreak Visualization:

Direct area visualization in monitoring the outbreak of diseases such as COVID-19, Ebola, or the flu. Circular or shaded map areas indicate areas of infection, facilitating policymakers' planning of interventions.

Example:

A COVID-19 heatmap (shown in fig 4.26) in which red-colored areas have higher infection rates, and blue or green-colored areas represent lower case numbers.

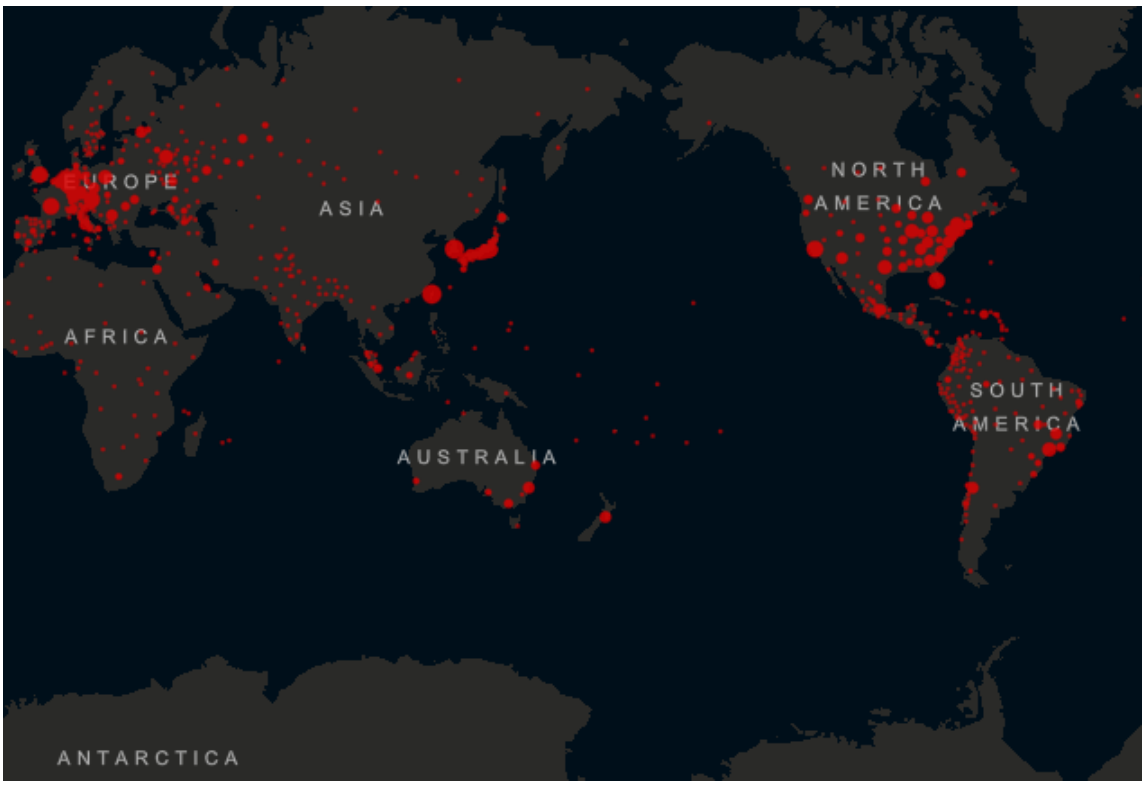


Fig. 4.26: Heatmap depicting COVID-19 cases

4.7 Handling Overlapping Points:

Overlapping data points in visualizations can conceal the essential details and give a false account of distributions. The problem of overlapping points can be effectively handled using techniques such as partial transparency, jittering, 2D histograms, and Contour lines.

i.Partial Transparency: For the better visibility of the data points, the opacity of the points can be reduced which allows a clear view of dense regions where data points overlaps as shown in fig 4.27.

Example:

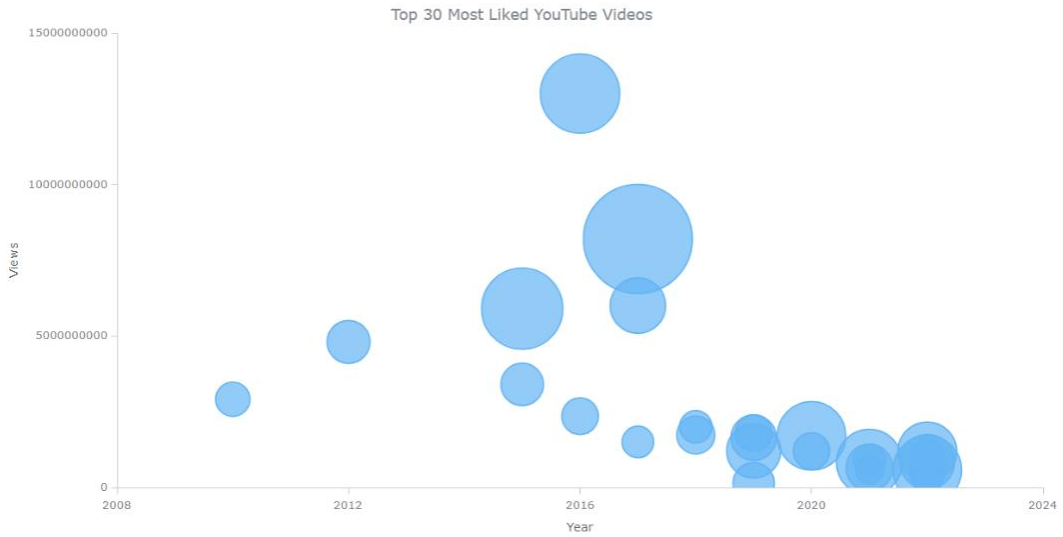


Fig. 4.27: Partial transparency

Why is Partial Transparency Needed?

When visualizing large datasets, particularly in scatter plots, heatmaps, and density plots, overlapping points will hide information and mislead the viewer. When points are fully opaque, the visualization will fail to represent high-density clusters well enough, leading the visualization to appear as if there are fewer data points than there are.

Example: Imagine a scatter plot of customer purchases at a shopping mall. If many customers are purchasing items from a similar store at the same time, their points would be at the same place. If there is no transparency, you might see only a few points while hundreds of purchases overlap. If you use partial transparency, overlaps are visible, and it becomes clear which stores have the maximum number of customer visits [30].

How Partial Transparency Works:

Reducing the Opacity of Data Points: Instead of each point being fully solid, its transparency is adjusted so that it is half-transparent (i.e., 50% opaque). This means that when points overlap, they merge visually rather than covering each other completely.

Accumulation Effect: When there are many overlapping points in a region, the density of color accumulates, which visually conveys that the region has a large quantity of data points. More transparent regions will be lighter in color, and denser regions will be darker or more saturated, hence easier to separate out as patterns.

Uses of Partial Transparency in Data Visualization:

1. Scatter Plots with Big Data:

Opacity decrease helps reveal clusters rather than hide them behind overlapping points in huge datasets with thousands of points.

Example: A scatter plot of student scores in an exam across different schools can contain overlapping values. With partial transparency, it is easier to notice which scores occur more often as illustrated in fig 4.28.

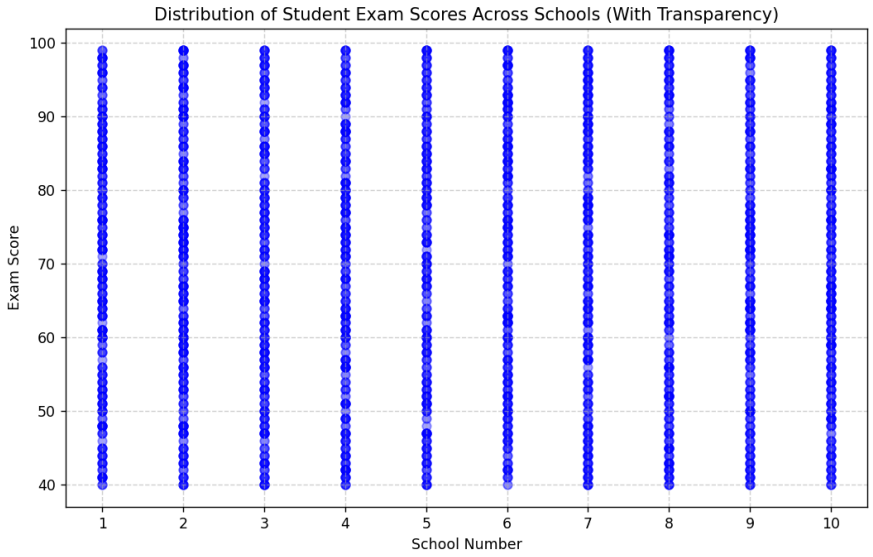


Fig. 4.28: Scatter plot presenting students' scores in exams

2. Geographical Data and Heatmaps:

When plotting GPS user locations, traffic flow, or COVID-19 infections, transparency is helpful in identifying high-risk regions better.

Example: A heatmap of New York City taxi pickup locations is improved by partial transparency, as high-demand areas are more saturated as shown in fig 4.29.

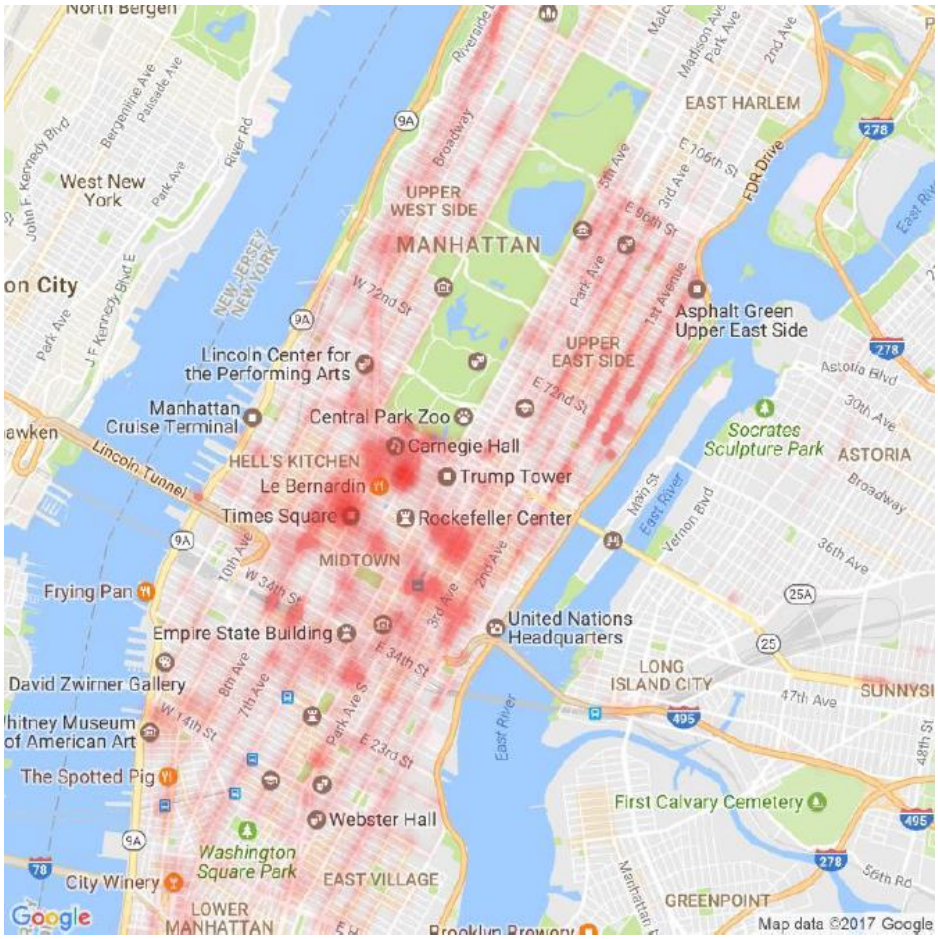


Fig. 4.29: Heatmap showing taxi pickup in Newyork

2. Astronomy and Scientific Data:

In astronomical observation, there can be thousands of stars, galaxies, or other objects in the same region of space.

Example: A scatter plot of galaxy clusters in a star cluster uses transparency so that tight clusters are visible rather than presented as one single patch as illustrated in fig 4.30.

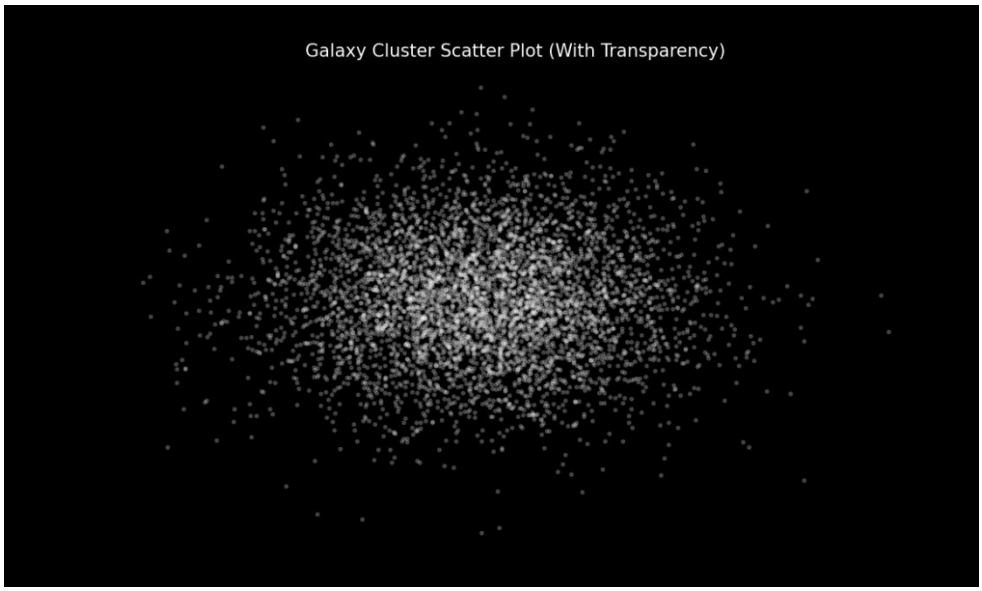


Fig. 4.30: Galaxy cluster with transparency

3. Financial Market Analysis:

The data of the stock market typically relates to presenting huge amounts of trade data, and here transparency can draw attention to trading times or top stocks.

Example: A stock price fluctuation chart shown in fig 4.31 for a year for a few companies can contain the same level of prices on certain days. Partial transparency allows for easier comparison of different stocks.

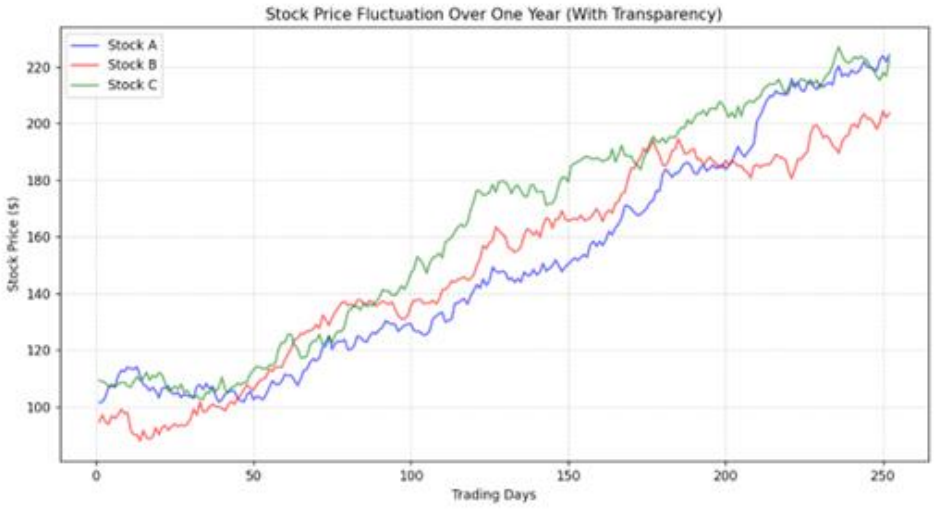


Fig. 4.31: Stock price fluctuation chart

4. Overlapping Line Charts in Time Series Analysis:

When several time series are shown on the same graph, partial transparency distinguishes overlapping trends without being cluttered.

Example: Stock trends of several companies for a year as illustrated in fig 4.32.

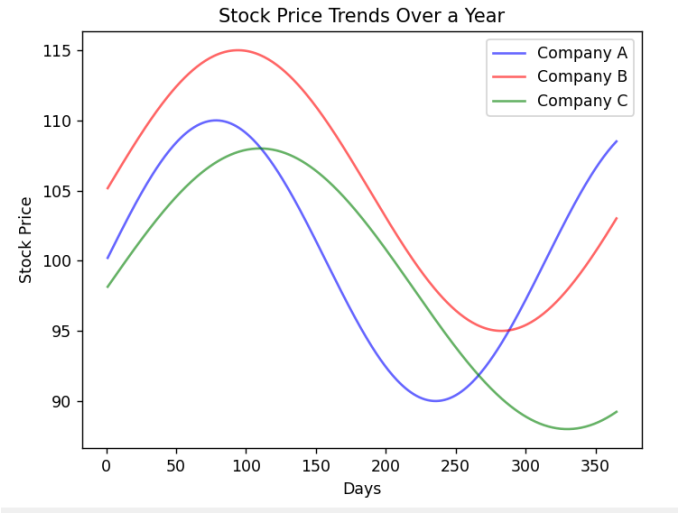


Fig. 4.32: Stock price trends over year

5. Visualization of Density in Survey Responses:

There may be overlapping answers in survey findings. Particular transparency assists in discriminating between the common answers.

Example: Age vs level of satisfaction survey findings as illustrated in fig 4.33.

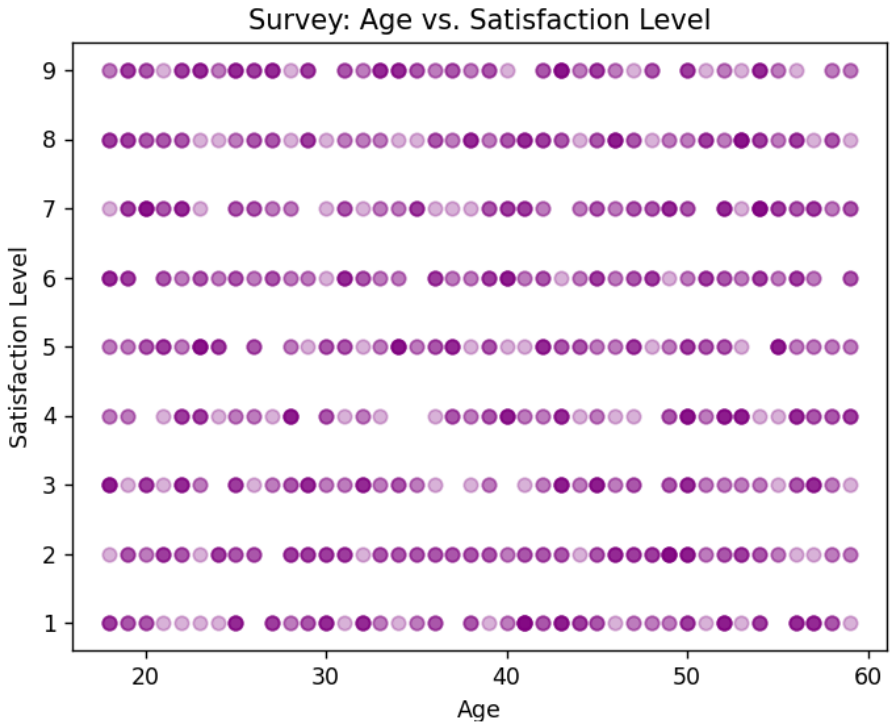


Fig. 4.33: Partial transparency differentiating common answers

6. Analysis of climate and environmental data:

Transparency in visualizations of climate assists in determining seasonal patterns and outliers.

Example: Temperature fluctuations over various years as shown in fig 4.34.

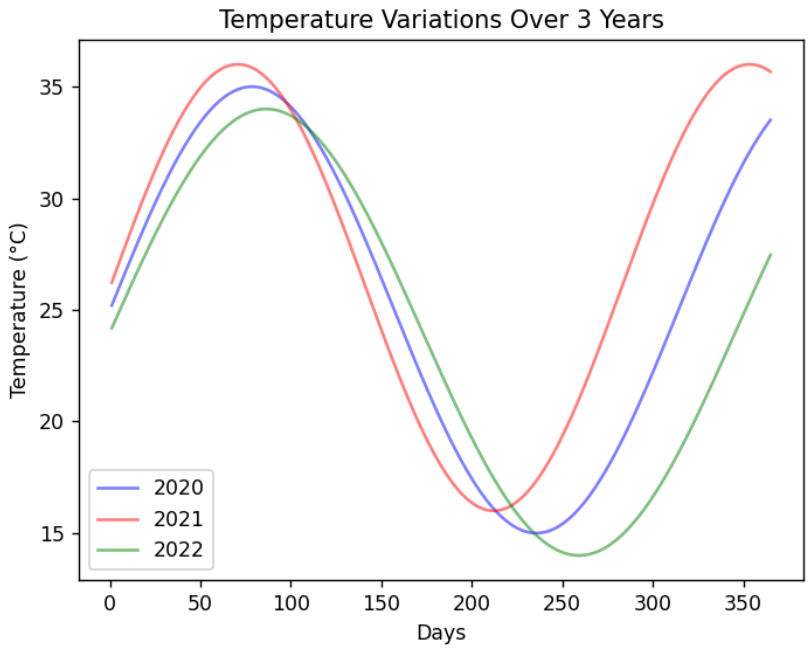


Fig. 4.34: temperature fluctuations over 3 years using partial transparency

8. Urban Planning and traffic flow:

In traffic pattern research, transparency assists in visualizing various overlapping routes.

Example: Transparency heatmap of NYC taxi rides as shown in fig 4.35.

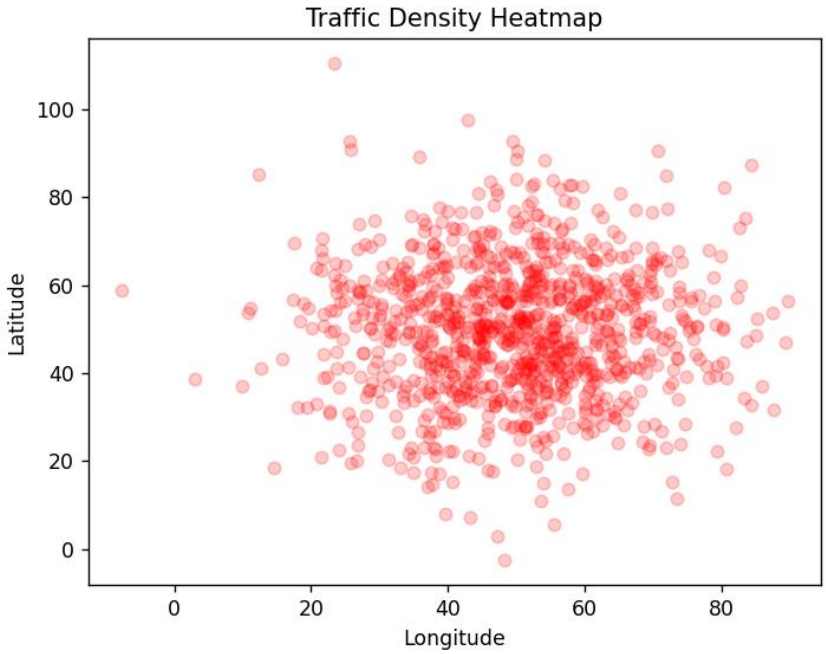


Fig. 4.35: Heatmap showing traffic density using partial transparency

ii. Jittering: Slight random variations are introduced to the positioning of the points to separate overlapping values, making patterns more observable.

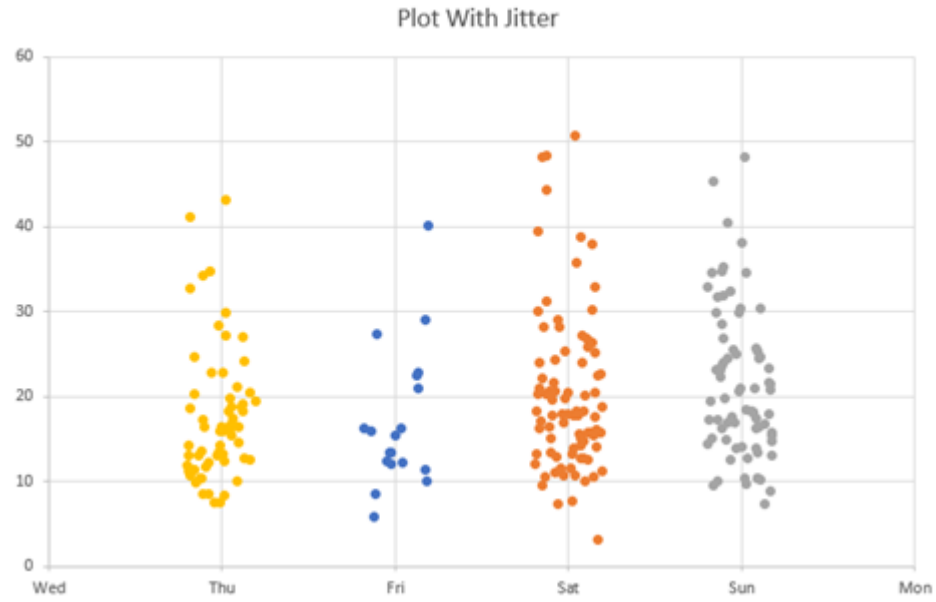


Fig. 4.36: Plot using jittering

Why is Jittering Needed?

When data points have the same value or very close values as shown in fig 4.36, they get plotted at the exact same location, causing them to overlap. This leads to data loss, where only a few visible points appear, even if the actual dataset contains many more [31].

Example: Assuming you had taken a survey and a lot of people gave the same rating (for example 5 out of 5 stars). If you plotted these on a scatter plot, all of the points for a 5 would be directly on top of each other. This can make it look like few people gave that rating when actually some did. Introducing a small amount of jitter simply shuffles the points around so that the quantity of ratings awarded is easy to see [31].

How Jittering Works

Adding Random Offsets: Jittering adds small, random offsets to points that would otherwise be plotted at the same location.

- Maintains Data Integrity: The changes are cosmetic and never alter the dataset values underneath.
- Prevents Overlapping Data: The small amount of movement separates each point cleanly without stacking.

Jittering is generally used in scatter plots, strip plots, violin plots, and dot plots, where overlapping values exist.

Applications of Jittering in Data Visualization

1. **Survey Data and Rating Systems:**

   Surveys tend to collect categorical data, e.g., user ratings between 1 and 5. Jittering is used to separate responses that would otherwise be the same.

   Example: If 200 users gave a product a rating of 5 out of 5, all 200 markers would cluster at the same location. Jittering scatters them slightly, and the density is more visible as seen in fig 4.37.

Fig. 4.37: Survey ratings visualization with jittering

2. Scatter Plots with Discrete Values:

When points are limited to certain integer values, they coincide. Jittering scatters them for the sake of clarity.

Example: A scatter plot of study hours vs. test scores of students can have numerous students with 80 scores and different study hours. Jittering distorts these points to represent their actual distribution as shown in fig 4.38.

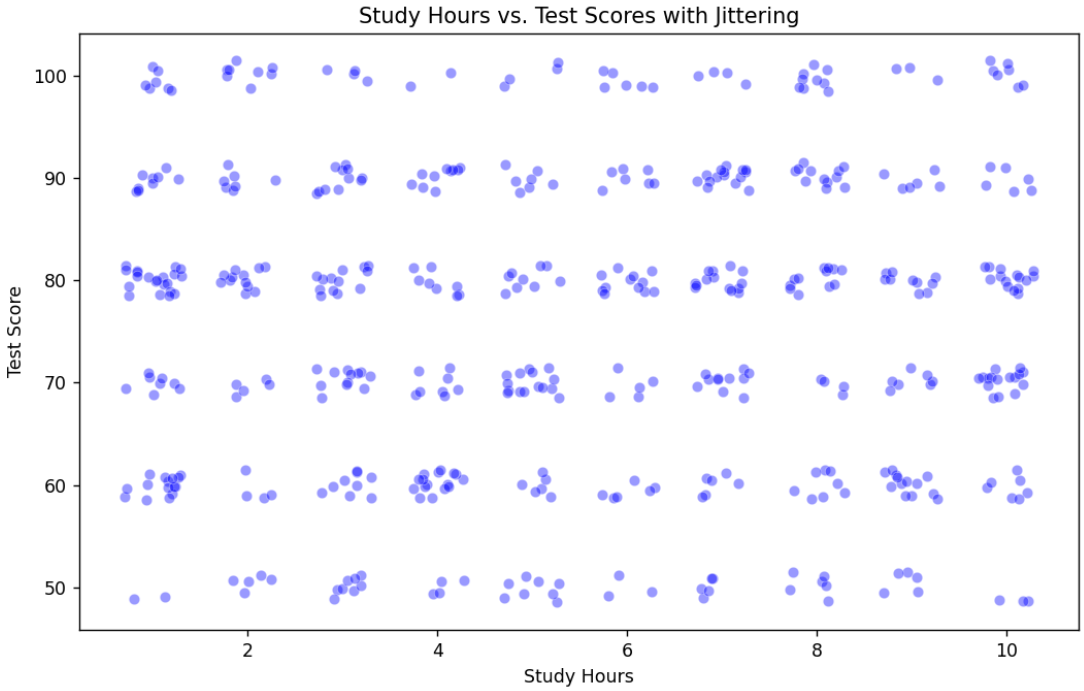


Fig. 4.38: Scatter plot using jittering

3. Medical and Clinical Research Data:

Data sets in health studies can have the same value for different subjects. Jittering prevents useful data from being lost.

Example: Blood pressure vs. age study might find that many patients share the same blood pressure. Jittering separates individual cases as illustrated in fig 4.39.

Fig. 4.39: Jittering showing clinical research data

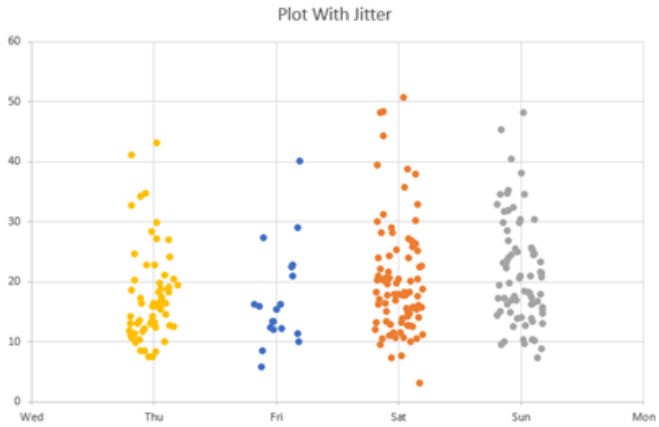


Fig. 4.40: Plot with jittering

4.8 2D Histograms:

2D histograms are used when there are lots of data points where scatterplot cannot perform really effectively as shown in fig 4.41 due to overplotting in chart. These are commonly used in heatmaps and for summarizing dense scatter plots by dividing the data space into bins.

The below picture is an example of a heatmap. The intensity of the color represents the magnitude of the count of samples from the dataset that fall into the corresponding bins of the histogram.

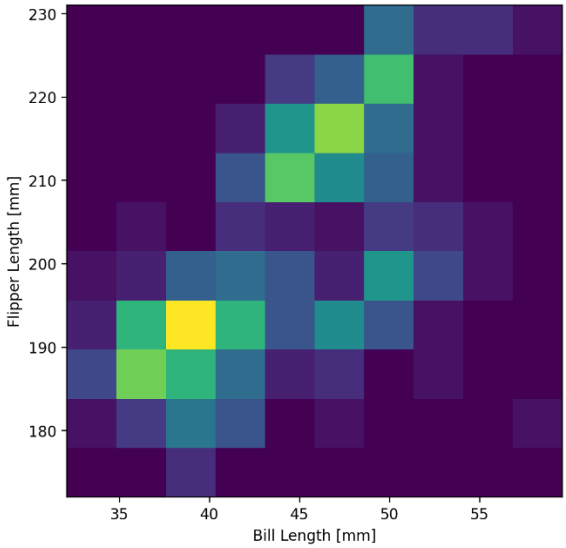


Fig. 4.41: 2D histogram representation

Why Not Use a 2D Histogram Rather Than a Scatter Plot?

1. Works Excellent with Large Data Sets:

Overlaid points conceal significant details in scatter plots, and data concentration is hard to discern.2D histograms reduce the data into bins, which makes areas of concentration easier to spot.

For example, if you are examining climate and have millions of readings of temperature vs. humidity, a 2D histogram makes it easier to identify trends.

2. **More Visible Data Density:**

   Instead of plotting individual points, a 2D histogram bunches up similar values close together, and dense regions stand out.

Example: Customer analysis: if you examine the interaction between spending and customer age, a 2D histogram reveals how much is spent by each age group without the clutter of individual points.

3. **Better Visualization for Continuous Variables:**

   Suits best when both x and y variables are continuous and have large values.

   Example: A 2D histogram can, in finance, be employed to display the interaction between stock returns and market volatility by defining the most frequent return-volatility pairs.

How Does a 2D Histogram Work?

Divide the Data Space into Bins:

Equal-sized intervals (bins) are divided along the x-axis and y-axis.

Each bin represents a rectangular area over an interval of values.

Count the Number of Points in Each Bin:

Rather than graphing all the points, the histogram accumulates the number of points within each bin.

Coloring the Bins by Density:

The denser the bin, the darker or taller the bin in the plot.

Applications of 2D Histograms

1. **Heatmaps of Data Density:**

   2D histograms find extensive applications in heatmaps, where color luminance is used to represent data density.

   Example: In geodata, heatmaps show population density as shown in fig 4.42 in urban areas with the assistance of 2D histogram methods.

Fig. 4.42: Heatmaps shows population density

2. Scientific Research and Astronomy:

2D histograms are employed to graph galaxy distributions, space temperature gradients, and black hole radiation patterns in astronomy.

Example: A 2D histogram of star brightness vs. temperature is used to determine the type of stars (as shown in fig 4.43)

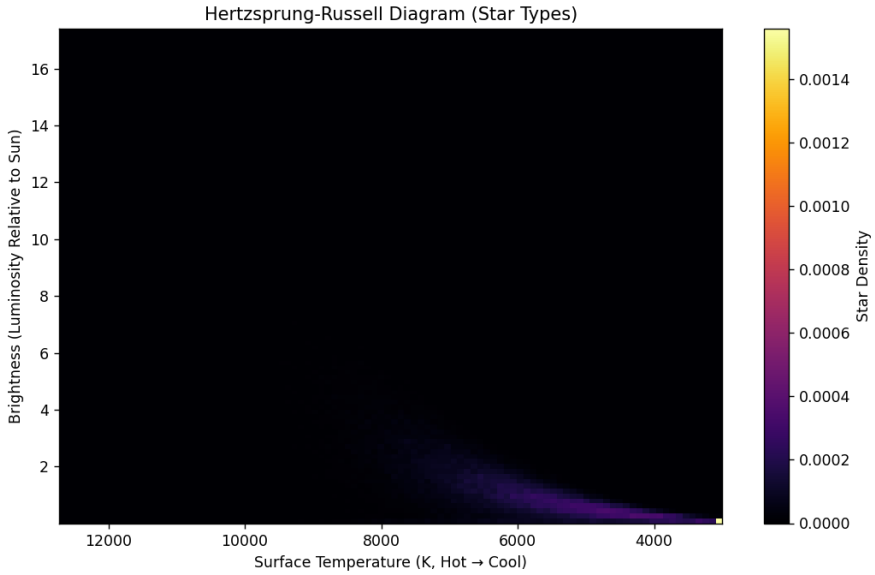


Fig. 4.43: 2D histogram of star brightness

3. Traffic Analysis and Smart Cities:

2D histograms assist with traffic flow pattern analysis, accidents, and speed of vehicles.

Example: Traffic study of a city may utilize a 2D histogram to illustrate peak congestion time vs. place as illustrated in fig 4.44.

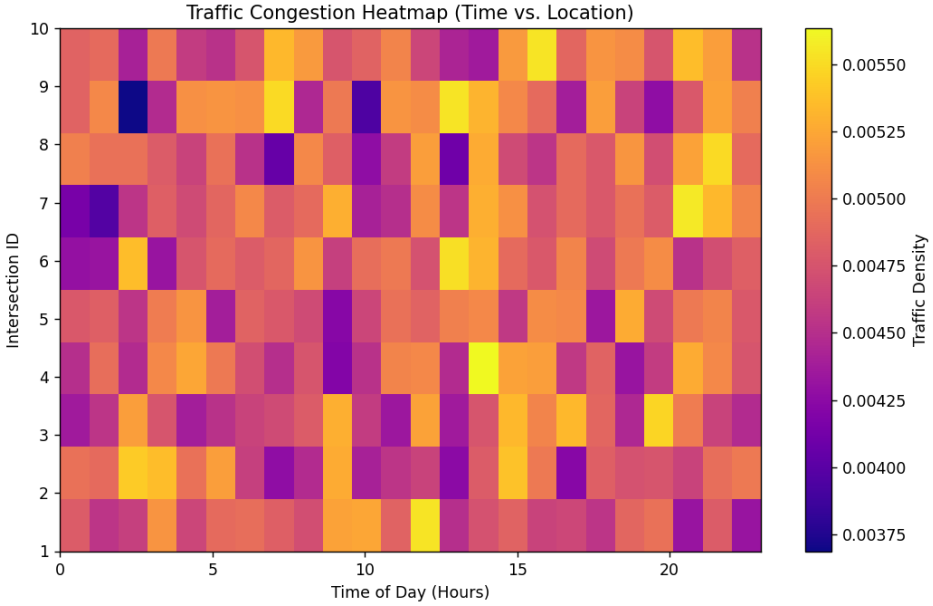


Fig. 4.44: 2D histogram illustrating peak congestion

4. Stock Market & Finance Analysis:

2D histograms assist in the analysis of stock market trend correlations.

Example: volume vs. stock price changes shown in fig 4.45

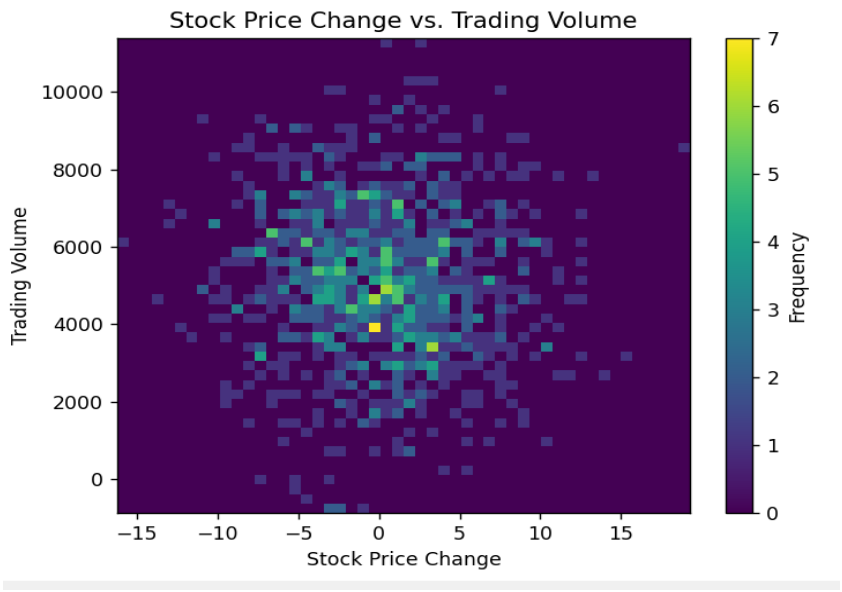


Fig. 4.45: 2D histogram assisting stock market

4. Market and Customer Behaviour Analysis:

2D histograms facilitate the visualization of customer shopping behaviours in terms of various metrics, including age vs spending patterns.

Example: Plotting customer spending vs age distribution as shown in fig 4.46.

Fig. 4.46: 2D histogram shows customer shopping behaviour

b. Contour Lines: These are used to represent the density in scatter plots and also geographical data visualizations. They help to reduce the area of high concentrations and is used with heatmaps for effective images if density changes as shown in fig 4.47. They determine the cluster of the data set with overlapping points.

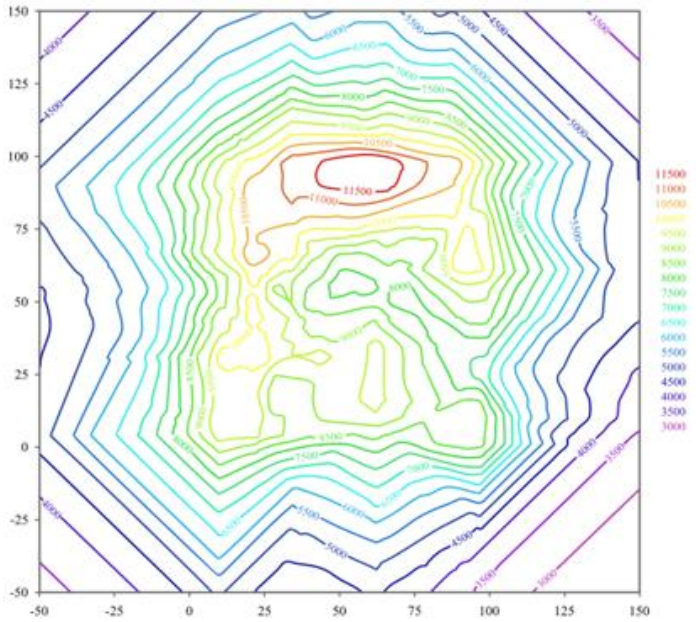


Fig. 4.47 Contour lines

Why Contour Lines?

1. **Captions Dense Scatter Plots:**
   On very large datasets containing thousands of points, contour lines avoid overplotting and improve the visibility of clusters.
   Example: When plotting house prices vs. square footage for a whole city, contour lines will show the most prevalent price ranges without filling the chart with thousands of separate points.
2. **Identifies Clusters and Trends:**
   Contour lines aggregate data points into areas of equal density, making trends easier to read.
   Example: In climate science, contour lines are used to create maps of temperature gradients between regions.
3. **Improves Heatmaps for Density Visualization:**
   Heatmaps use color saturation to display density, but contour lines provide sharp boundaries to segment high-density regions.
   Example: A map of COVID-19 infection can use contour lines above a heatmap to distinctly indicate the areas with most concentrated cases.
4. **Improves Legibility of Geographical Maps:**
   Contour lines are used to show elevation, pressure, or other geographical change on topographical maps.
   Example: Contour lines on trekking and military maps indicate steepness of land. The tighter the lines are together, the more steeply the land.

How Contour Lines Work

1. **Data Density is Calculated**
   The set of data is examined to ascertain where points are most dense. Such locations are attributed varying levels of density.
2. **Contours are Rendered Depending on Density:**
   Lines are employed to link points of equal density, creating closed curves enclosing high-density areas. The thinner the lines, the denser the area.

3. Color or Shading is Provided (Optional):

Contour maps tend to utilize color gradient or shading for enhanced interpretation.

Applications of Contour Lines

1. Geographical and Topographic Mapping:

Used to graph height, sea depth, and topography.

Example: A mountain range contour map informs hikers about the steepest and flattest areas as shown in fig 4.48.

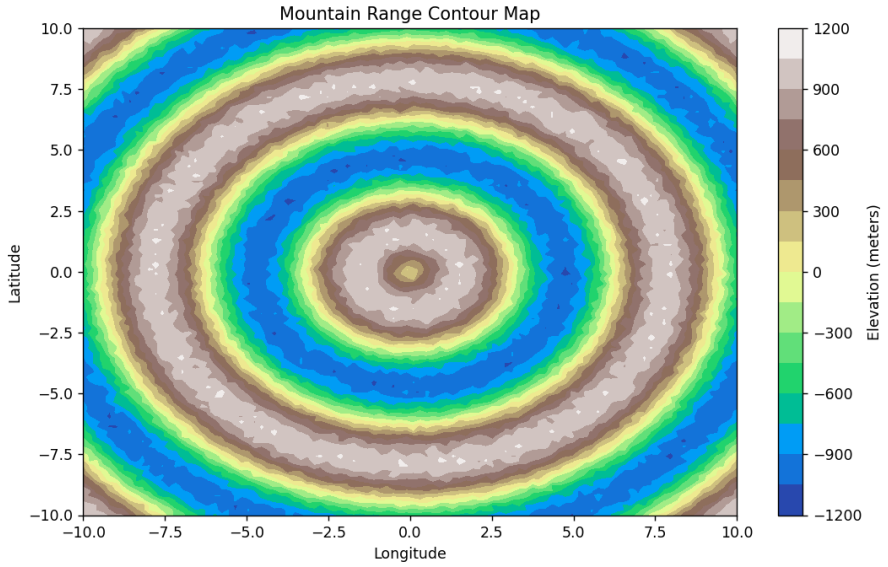


Fig. 4.48: Contour Map

2. Meteorology and Climate Science:

Contour lines assist in monitoring temperature, atmospheric pressure, and storm vigor.

Example: Contour weather mapping reveals high and low-pressure zones, which can forecast storm tracks as shows in fig 4.49

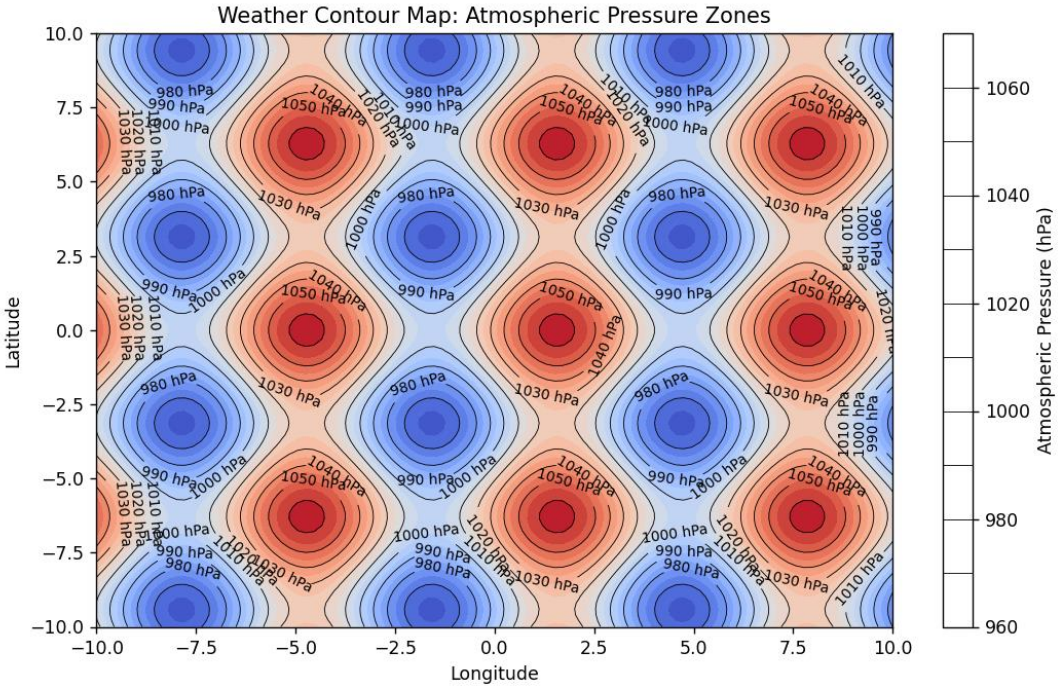


Fig. 4.49: Weather contour map

3. Scatter Plot Density Representation:

Plotting numerous points, contour lines clump together the overlapping data, showing major trends.

Example: An education level vs. population income scatter plot with contour lines illustrates the most frequent income ranges in fig 4.50.

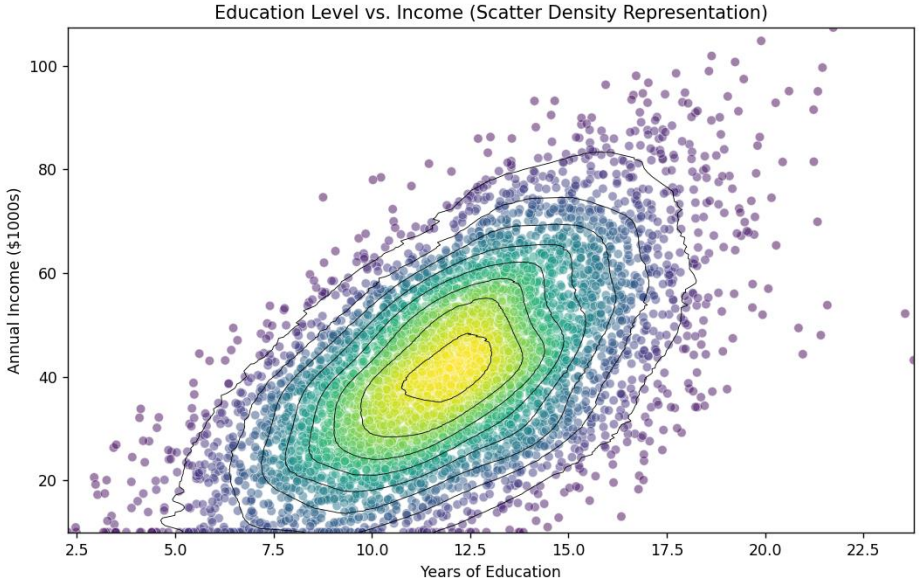


Fig. 4.50: Scatter Density representation

4. **Environmental science and pollution analysis:**
   Utilized for tracking air pollution, deforestation, and climate change patterns. Example: Contour plot of air pollution concentration over a city.
5. **Sports Analytics & Performance Visualization:**
   Used to analyze player movements, ball trajectories, and scoring heatmaps in sports.
   Example: Heatmap of player movement in a soccer match as shown in fig. 4.51

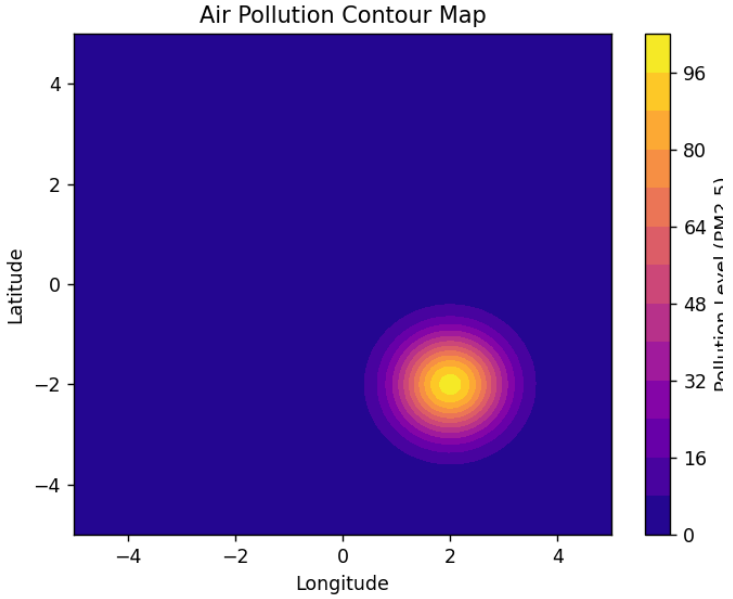


Fig. 4.51: Air pollution contour map

It is applied to study player movement, ball path, and scoring heatmaps in sports.
Example: Player movement heatmap in a soccer game as shown in fig 4.42

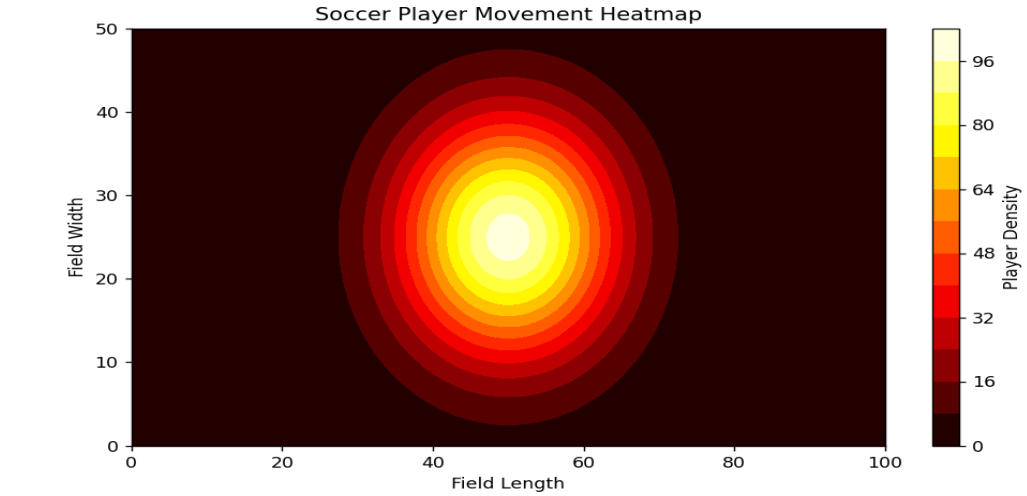


Fig. 4.42: Soccer player movement heatmap

Chapter 5

COLOR USAGE IN DATA VISUALIZATION

Shruti[1], Arshdeep Singh[2], Neharika Sharma[3] and Sumit Chopra[4]

[1,2,4]GNA University, Phagwara

[3]Rayat Bahra Group of Institutions and Nanotechnology, Hoshiarpur

5.1 The Role of Color in Communication

Color plays a crucial role in visual communication. It deepens comprehension, captures interest, conveys feelings, and improves accessibility. In areas like branding, user interfaces, data visualization, and printed materials, color directs perception and influences choices.

As much as color can stir emotions, it also has the strength to confuse, misinterpret, and leave those who should otherwise access materials inaccessible. It is perhaps one of the most potent weapons in the arsenal of a designer and data visualizer. But it is only as good as it is well applied. To draw color appropriately, you have to understand the merits as well as the demerits that come with it [32].

Perhaps one of the biggest advantages is that color carries meaning. Accordingly, it should enhance reading more effectively by drawing attention to the important parts when used correctly. For example, if the report is stressed in a different color, this can help people see critical points more quickly. Also, colors show the distinctions between different categories, groups, or trends. For example, in data visualization, you can give a specific color to different segments of the bar chart or the slice of a pie chart so that users can easily understand the patterns and relationships without reading comprehensive labels. The most common and well-illustrated example of this effective color use is the system of traffic lights. Red shows "stop," yellow signifies "warning," green indicates "go." This universal applicability of color means that a driver simply recognizes and reacts to a signal without anything needing to be added. Similarly, in consistent patterns, the user can understand more and improve the interaction when applying color consistently in design and data presentation.

5.2 Explaining Pitfalls in Data Visualization:

5.2.1 Pitfall 1: Encoding Too Much Information with Color

The Persuasive Use of Color in Design: Color in design, data visualization, or user interface is an important tool to convey meaning, grab attention, and aid understanding. When applied correctly, color helps the understanding of information; when wrongly applied—laying too much inference on it or using colors unrelated to the topic or excessive colors—color clouds meaning, reduces readability, and makes the data or design fail in the first place. With such a vast assortment of colors embedded into visualization, there seems to be a feeling that such colors obstruct understanding and instead generate visual clutter in view of the viewer.

One of the major sins in color usage is jamming too many elements into one chart or design as shown in fig 5.1. The moment too many colors are used, it detracts the focus from key insights that are really important for the users. A very clear example of this mistake comes in pie charts where differentiation is attempted between numerous categories having separate colors for each. Introducing such a plethora of colors to the chart tends to further concentrate on the data rather than clarifying it, confusing the observer instead [32].

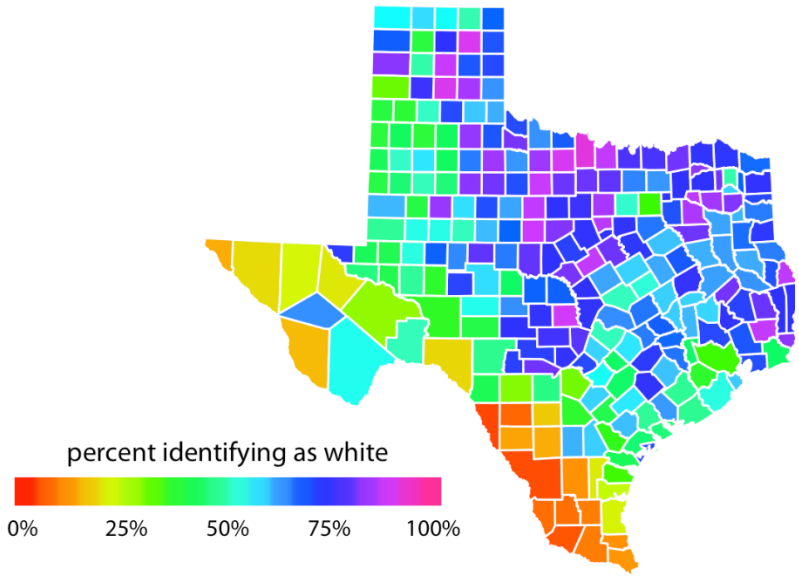


Fig. 5.1: Encode too much color in graph

A fundamental problem connected with excessive color usage in visual representation is that it impedes value comparison. If anyone attempts a value comparison having a wide palette of colors in their charts or graphs, they are likely to focus less and less on finding relationships that matter when distractions keep protruding among the clutter of hues. It is the distractive palette of colors that may be taking away their attention, preventing them from arriving at useful information. Having a simple structure with few carefully chosen colors works best for creating clarity by drawing attention to the key aspects of the data and away from any extraneous distractions. To complicate matters further, the very abundance of color serves to prolong the time spent interpreting the presentation as linking labels to colors. Users cannot rid themselves of wanting to refer to the legend for mapping every color whenever a chart displays somewhat similar colors. That undoubtedly adds to cognitive burden and decelerates value retrieval. By judiciously using few colors, users can intuitively match them to categories using little thought without correlating with color legend excessively. A major problem arises when the variations of the two colors are too subtle to be distinguished. When different shades of blue or green are used to represent completely different colors, persons with color vision deficiencies and those viewing it on a low-resolution monitor might miss the distinction completely. Hence, the critical cues may be lost just because of the chosen colors not providing sufficient contrast. To aid accessibility and clear interpretation by all users, designers must opt for colors that are distinct and provide a higher contrast. Thus while color is a great asset when enhancing data visualization, it is more of a strategy when one comes to usage [32]

Certain websites or apps have lots of different colors for buttons, text, and backgrounds as illustrated in fig 5.2. If each part of the website is a different color, people can find it difficult to know what's most important.

If too many colors are applied with no particular intent, the visual design ends up being overwhelming. Users find it hard to direct their attention towards key items, resulting in an annoying experience. Cluttered visuals may cause distraction from the central content and complicate navigation. When colors are not used consistently, users might struggle to comprehend their meaning or function. For instance, using varying colors for the same functions or actions can cause confusion, and it becomes more challenging for users to anticipate interactions and navigate effectively.

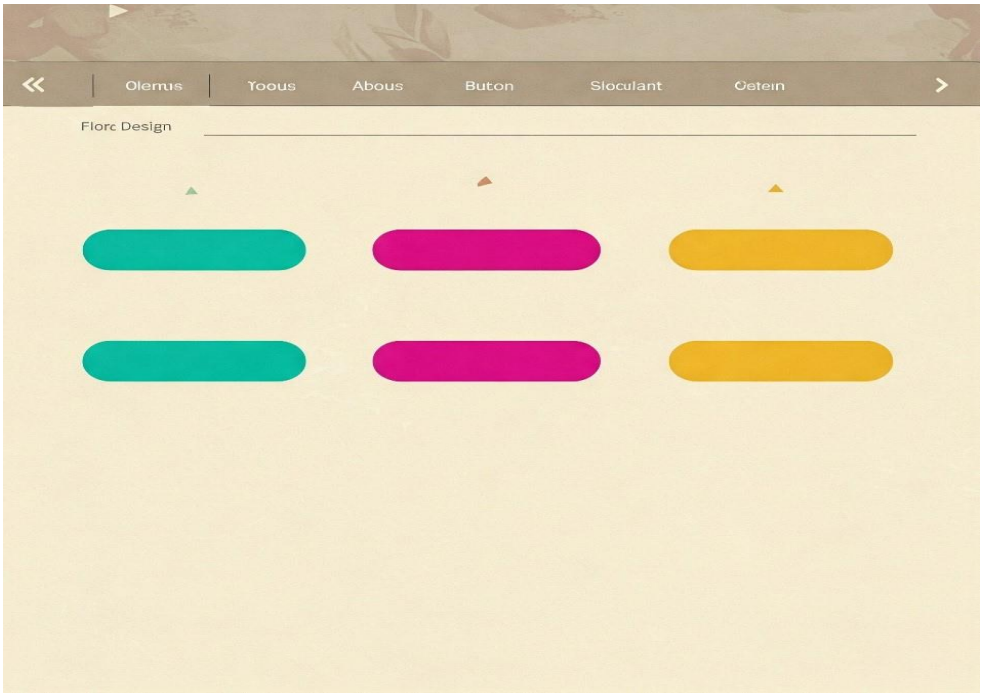


Fig. 5.2: Different colors for buttons, text, and backgrounds

An organized color scheme facilitates intuitive navigation. Bad color selections, like low contrast between the background and text, can render content less readable, particularly for people with visual disabilities. Eye-stressing colors like bright or too saturated ones lower readability. Furthermore, neglecting color blindness makes important information unavailable to a part of users. Good contrast and careful use of colors improve readability and provide inclusivity.

5.2.2 Pitfall 2: Irrelevant Colors

Color is intrinsically important in thinking about data interpretation; misapplying it according to the normal understandings often results in confusion and misinterpretation. Irrelevant or misleading coloring may distort meaning as well and thus hinder the viewer's ability to understand and reach conclusions based on trends. Rather than improving clarity, misapplication can in fact lead to more unnecessary mental lifting, since users must second-guess what it is that they see instead of absorbing information readily.

Typical example of an irrelevant color application is mixed-up color associations. There are certain colors which enjoy widespread acceptance in many cultures and industries to mean a particular thing. For example, green would commonly represent such positive results as profits, growth, and success, whereas red usually indicates caution, loss, or negative trends. By inverting these associations, it confuses viewers who are conditioned with the normal color coding. Sample this: a financial report, balances, revenues shown in red, and deficits in green. This is the exact opposite of the most commonly recognized standard in which green indicates profit and red loss. Picture in their minds what may happen during data interpretation by financial analysts or decision makers as they go through such report; whereas, their brains have been wired to associate red with loss in financial performance and green with positive growth [33]. Thus, this reversal forces viewers to take an extra step to reorient themselves to the color scheme, resulting in unnecessary delays in analysis and a higher chance for making mistakes. Misuse of colors can also influence the emotional perception of the data. Every color has psychological meanings; the color blue synonymous with calmness and stability; yellow represents caution and warning. Therefore,

representing data using an alarming color red for neutral information is likely to cause panic and alarm in the absence of one. Similarly, using subtle pastels in reporting very discordant data, such as risk assessment reports and emergency response dashboards, will not highlight seriousness for proper attention.

Hence, without aligning one's color choices with conventional meaning and expectations of the audience, clarity and effectiveness in data visualization will not be achieved. Following common color conventions should therefore enable designers and analysts to not only aid comprehension and minimize cognitive overload but also allow viewers to grasp the insights with minimal confusion.

Fig. 5.3: Earnings are shown in red color while loss in green.

If a design or interface does not conform to established conventions, the users will be confused about how to use it. Design consistency allows users to anticipate functionality, and when they are broken, it can create frustration and confusion as shown in fig 5.3. For instance, if commonly used symbols or colors are used in a manner different from normal, the users will misinterpret their meaning. It leads to misinterpretation of information: Inadequate design decisions, like deceptive visualizations, confusing labels, or inconsistent data representation, may lead to users misinterpreting major insights. This is particularly undesirable in graphs and charts, where incorrect scaling, deceptive colors, or perplexing legends can lead to inaccurate conclusions being drawn. Intuitive and precise presentation is needed to effectively present accurate information. When a chart or graphic is not clearly and simply designed, users must take additional time to learn how to read and understand it. Confusing or cluttered graphics, ambiguous legends, or irregular formatting add cognitive load, making it more difficult for users to get the critical points of the information quickly. A neat chart ought to convey insights without demanding a lot of effort from the reader [34].

Some people utilize color on things that do not need it. For example, if all of the text on a page is in different colors without a real reason, then it is distracting. When design elements, like too many colors or ornaments, are added without purpose, they don't help make the message more understandable. Rather

than helping people understand better, they become distractions that do not make the content clearer or more effective. Each design decision must serve a functional purpose to enhance communication. A messy or disorganized design may give a website, presentation, or document an unprofessional look. Inconsistent application of colors, fonts, or graphics can provide an amateurish look, decreasing credibility. Well-organized, well-balanced design reflects attention to detail and strengthens the overall user experience. It can cause readability issues, especially on some backgrounds: Insufficient contrast between text and background colors may result in difficult reading, especially for visually impaired users. For instance, light-colored text on a white background or dark-colored text on a very dark background will cause eye strain. Providing adequate contrast and accessible color schemes enhances readability and user experience for all users.

It is best to avoid irrelevant and meaningless colors in design and even data visualization by advocating best-practice measures for clarity and meaning. The best of these includes one of the most effective methods: complying with color conventions. The reality is that some colors are universally accepted as having specific meanings—for example, red for warnings or errors; green for successful operations; and blue for hyperlinks. As a result, by adhering to the expected use of colors, information can be stumbled across within an environment as intuitively as possible. Another element is that color should be used only where it is really needed. Too many unnecessary colors can introduce unnecessary 'clutter' and elsewhere make it harder for users to see the more important things. Color should be for emphasizing significant data points, differentiating categories, or guiding user attention. Any color which does not seem to serve a function would better be left out. Finally, there's consistency in color application too. In particular, similar items must always be coded in the same way for a logical flow of coherence. So blue for hyperlinks in one page shouldn't necessarily be the same blue coded for error messages in another. All done in a consistent way, users will realize how easy it is for them to navigate interpretive documents without going into too much trouble.

5.2.3 Pitfall 3: Employing Nonmonotonic Color Scales to Represent Data Values

Color is a critical data visualization tool that allows the audience to rapidly understand complicated information. However, the misuse of color scales can result in misinterpretation, visual disorientation, and data distortion. One of the most frequent errors is the utilization of nonmonotonic color scales—scales that are not smooth, perceptual, and therefore make it hard to interpret data values correctly. There are two types of color scales i.e. Monotonic Color Scale and Nonmonotonic Color Scale. A monotonic color scale is a logical, perceptually smooth progression that allows viewers to easily relate increasing or decreasing color intensity to a corresponding change in data values. A nonmonotonic color scale, however, does not exhibit a smooth or consistent progression. Rather, it:

Using lots of colors without systematically applying them can create a chaotic and nonsensical visual experience. When colors are placed in a random manner as opposed to a logical one, users have difficulty discerning patterns and meaning. A lack of consistency complicates information being read quickly and efficiently. Sudden changes in brightness or saturation can be visually jarring and distracting. High-contrast color transitions, unless used intentionally, can make text harder to read and comprehend. In charts or graphs, these kinds of changes might mislead people by creating spurious points of emphasis that don't accurately represent the significance of the data. In the case of irregular distribution of colors over a design or data visualization, some information may automatically end up being emphasized more than others. This distorts the perceived meaning of data points and can lead to erroneous interpretation. A

balanced color scheme ensures emphasis is obtained consciously and consistently, guiding users to the appropriate conclusions.

Fig. 5.4: The rainbow color scale is non-monotonic

The Nonmonotonic Color Scales shown in fig 5.4 are Problematic because it produce visual aberrations that mislead the interpretation of the data. Human eyes do not see all colors with the same intensity. Some colors, for example, yellow, are brighter than others, like blue, even though they might be showing the same data value. For Example: A rainbow color scale heatmap can misleadingly imply that yellow areas are "higher" than neighboring colors, even if the values are equal. Utilize perceptually uniform color scales like Viridis, Inferno, or Cividis that provide equal visual weighting. If nonmonotonic scales jump from one color to another abruptly (e.g., green → yellow → red), they create artificial visual boundaries that are not present in the data. For instance: In a weather map depicted with a rainbow scale, the smooth temperature gradient can be represented as having "hot zones" where the color changes suddenly although the transition might be gradual. Employ a single-hue sequential gradient that smoothes its transition from light to dark and does not include sudden changes. Nonmonotonic scales can mislead trends, particularly when audience members believe equal color distances imply equal numerical differences. For Example: If the blue-green-yellow-red scale is used to portray air pollution, individuals will be likely to presume that the distance between blue and green is identical to the distance between yellow and red, although the numerical difference is much greater. Solution: Employ logarithmic or linear scaling that properly demonstrates the underlying relations of the data. Both nonmonotonic color scales typically depend upon red-green and blue-yellow transition, which for colorblind users are hard to distinguish. For Example: A red → white → green diverging scale can become illegible with red-green color blindness. Select colorblind-safe palettes (e.g., Cividis or Grayscale), and always include alternative encodings (patterns, symbols, or annotations). To make sure that your visual data are communicated clearly and accessibly, color scales are best employed according to data visualization conventions. As the right color scale adds value to the visualization by letting the audience easily comprehend patterns, trends, and relationships in the data, it is necessary to choose appropriately [34].

Perceptually uniform color scales such as Viridis, Inferno, Cividis, and Magma should arguably be among the most significant considerations. Designed to achieve balanced and perceptually consistent change in perception across the full range, these color scales differ from the arbitrary choice of colors which do not convey any meaning; these perceptually uniform scales scale up in diffractive failure across the whole range of colors any one can perceive just as differences in data values. Simply put, the more color-uniform a color scheme is, the more intuitive and accessible visualizations become to everyone, including people with color blindness. For ordered or continuous data, it would be best to apply a single-hue sequential color scale. This entails using a gradient that goes from light to dark within one color family, for example, light blue to dark blue in showing population density. This logical application therefore allows no room for confusion that 2 or 3 colors (say blue, green, and yellow) might present, which could be misinterpreted as categorical data rather than continuous. With diverging scales, one must ensure that

it is justifiable to declare a true and meaningful midpoint to the data. In other words, any situation where the information depicts positive-negative (e.g., profits-losses, temperature differences, or election outcome), diverging color schemes such as red-white-blue would indicate the differences. Nuisance use of diverging colors can, however, give rise to the wrong impression, especially when the midpoint bears no operational weight.

5.2.4 Pitfall 4: Failing to Design for Color-Vision Deficiency:

Color plays an essential role in design, data visualization, and user experience. It enhances 3readability, boosts engagement, and conveys meaning quickly. However, color perception varies among individuals. Those with color vision deficiency (CVD), commonly known as color blindness, may struggle to differentiate between certain colors. In spite of the widespread nature of color blindness, numerous designers, developers, and researchers do not consider it, which leads to inaccessible designs, deceptive data visualizations, and poor user experience.

Color vision deficiency (CVD) is a condition that causes individuals to have difficulty differentiating between specific colors as a result of how their eyes perceive light. About 300 million people globally are affected, which includes 1 in every 12 men and 1 in every 200 women. It affects designs as most designers take it for granted that color is perceived by all people in the same manner, resulting in visual elements that are impossible—or at least very hard—for colorblind people to understand. Overlooking color accessibility can lead to unclear warnings, alerts, and UI elements [35].

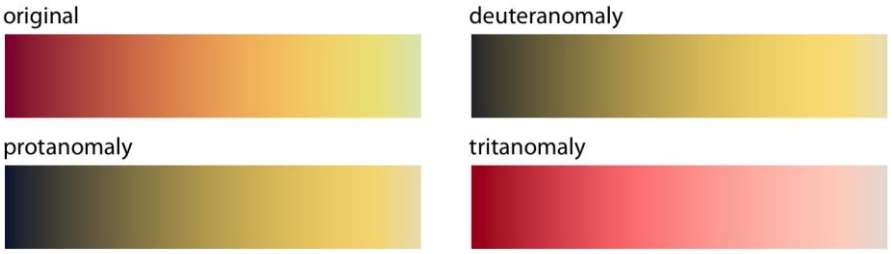


Fig. 5.5: Color-vision deficiency (CVD) simulation of the sequential color scale Heat

The designing for color blindness matters because using only color to convey crucial information may create accessibility problems since some users will not be able to comprehend the message. Though color is a strong visual device, it must always be paired with additional design elements to maintain clearness and inclusiveness. For example: Take an online form where the errors are merely highlighted by a red outline or red text. A user who is completing the form may come across a field that becomes red when the user enters incorrect data. But the problem is that not everyone views colors in the same manner. Individuals with red-green color deficiency, the most prevalent of the color vision deficiencies, might not be able to tell the red error indicator apart from other items on the page. This would lead to confusion, as they would either not know an error has been made or have difficulty finding it on the form. Also, in low-light environments or on low-contrast screens, red marks may not be easily distinguishable enough to clearly get the point across. But the solution is to make sure that the error messages are understood by everyone, designers need to include more than one visual cue in addition to color. Placing an exclamation point or warning symbol next to the error field makes the problem stand out even for users who are unable to see red. Showing a message like "Invalid email format" gives a concise description of the issue. Applying a dotted or bold underline rather than simply altering the color makes the problem noticeable in a non-color-based manner.

To define categories in graphs and charts through color alone creates a problem for accessibility, especially for people who have color vision deficiency. The chart should be designed in a way that everybody, irrespective of their capacity for color perception or not, has no difficulty at all interpreting data. For example: A pie chart with red, green, and blue used for different categories and no labels, patterns, or other distinguishing markings. Users need to distinguish among the sections just by color. But the problem is that individuals who are red-green color blind (the most prevalent type) won't be able to tell red from green sections and thus interpret the chart effectively. Even those with typical color vision might struggle in dim lighting or on displays with low color contrast. If a chart is displayed in grayscale, all colors can be represented as varying shades of gray, rendering the data virtually impossible to interpret. Users will have to refer to an external legend, which means they will have to keep switching their attention back and forth, which raises cognitive load and slows down interpretation. Rather than depending on color alone, fill areas of the chart with striped, dotted, or crosshatched patterns. This enables users to distinguish areas even in grayscale or color-blind viewing environments. Simply labeling every section of a chart, as opposed to utilizing a distinct legend, allows the user to perceive the data without color-matching. In line graphs or bar charts, employing a variety of different shapes (i.e., triangles, circles, squares) for data points allows them to be distinguished without reliance on color. For digital charts, making accessibility settings available to enable users to alter colors or enhance contrast can improve readability for a variety of audiences.

Best Practices for Color-Blind-Friendly Design:

Color palettes that are accessible to the general public, including people with color vision deficits (CVD) shown in fig 5.5, should be used when developing data visualizations. Data is misconstrued because most traditional color schemes are difficult for colorblind people to distinguish. Color palettes created especially for accessibility are offered by visualization programs like Tableau, Matplotlib, and D3.js to promote diversity.

Scientific data visualization becomes an important aspect for making complex data more understandable so that researchers and professionals can detect patterns, trends, and outliers. Visualization should ensure that the data is not only good-looking but also correct, readable, and available to everyone using it. A major aspect of data visualization is the selection of color schemes because incorrect choice of color schemes can cause misinterpretation and inaccessibility. It is here that Viridis, a perceptually uniform colormap, has found itself an indispensable resource for scientific visualization. Scientific visualizations have historically used the rainbow color scale, based on a sequence of colors from violet through to red. Although it is perhaps the most intuitive-looking choice, the rainbow scale has severe limitations. It is not perceptually uniform, i.e., certain transitions of color look more extreme than others, even though the data behind them is changing uniformly. This can skew the view of data relationships and result in false conclusions. The rainbow scale is also a problem for color vision deficient people, especially those with red-green color blindness, since some colors become impossible to distinguish. Viridis, created as a better alternative, is a colorblind-friendly, perceptually uniform, and high-contrast colormap. It is a smooth gradient from dark blue to light yellow, so all transitions are perceived evenly. In contrast to the rainbow colormap, which produces artificial visual discontinuities, Viridis has uniform luminance progression, making it simpler to perceive subtle data variations. This makes it particularly valuable in scientific applications like heatmaps, geospatial maps, and medical imaging, where precise color reproduction is critical to making accurate conclusions [35].

One of the most important benefits of Viridis is that it is accessible. It is created to be readable by people with different types of color blindness, so that scientific information is not excluded. In addition, it is very readable when printed in black and white, which makes it a useful option for research articles, where reproduction of color is not always guaranteed. Its simplicity, perceptual similarity, and easiness to produce make it particularly suitable for visualizing data with the assurance of presenting accurate meaning and being read easily by as wide an audience as possible. Choosing the correct color scheme, though, is not always simple. Inaccurate color use can result in misinterpretation, unavailability for colorblind users, and poor communication of important findings. To counter these problems, ColorBrewer, a cartography tool created by cartographer Cynthia Brewer, offers optimized color palettes specially made for mapping and data visualization. It is a collection of well-designed palettes of colors that improve readability, accessibility, and perceptual correctness. The palettes find extensive use in geographic information systems (GIS), thematic maps, and statistical graphics [36].

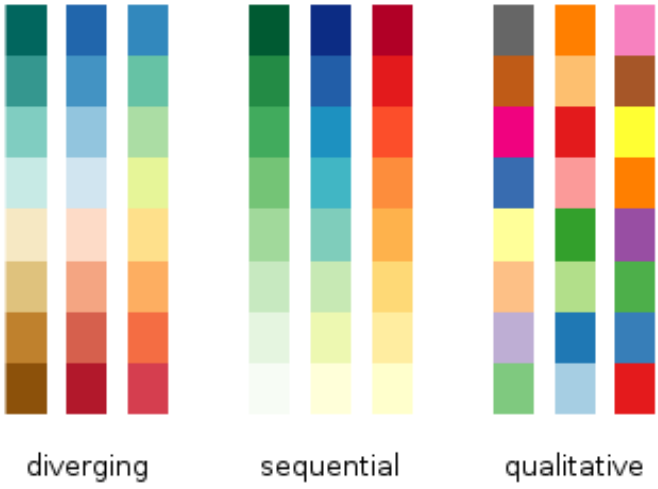


Fig. 5.6: Color Map from colorbrewer

The tool is divided into three broad categories of color schemes that are designed for the representation of different types of data. Sequential Color Schemes are applied to data that has a continuous progression, like elevation maps or temperature gradients. These schemes often go from light to dark shades of one color (e.g., light blue to dark blue) as shown in fig 5.6, making it easy for users to intuitively perceive increasing or decreasing values. Diverging Color Schemes are best for data with a significant midpoint, like temperature anomalies (positive and negative) or election outcomes (gains and losses). These schemes employ two contrasting colors that come together at a neutral center, allowing one to more easily observe where data values diverge. Qualitative Color Schemes for categorical data, where colors should be different but not ordered. These are typically applied in political maps, land-use maps, or region-based visualizations, so that various categories can be easily differentiated. One of the strongest strengths of ColorBrewer is that it has colorblind-friendly versions included. Most standard colormaps, particularly those based upon red-green differences, are problematic for color vision deficiency sufferers. ColorBrewer offers versions that guarantee to have high contrast and legibility for every user, so it is the default recommendation for government organizations, researchers, and GIS analysts. Aside from accessibility, ColorBrewer also guarantees the effectiveness of color schemes in any media format, whether displayed on a screen, printed in black and white, or projected. This flexibility is what makes it

an effective instrument in cartography, demography, environmental mapping, and urban planning, where intuitive and unambiguous visual communication is critical [36].

Dependence on color alone to communicate information in charts, graphs, and maps can lead to accessibility problems, particularly for people with color vision deficiencies. To enhance clarity and inclusivity, designers and data visualizers need to add other elements like labels, patterns, and icons. These add emphasis to the message, making visual data easier to understand for a wider audience. Labels are one of the easiest methods of enhancing data visualization. By inserting text annotations directly into a chart or map, users can instantly comprehend the meaning of each part without needing to consult a legend elsewhere. For instance, in a pie chart, inserting percentage values and category names inside the segments obviates the need to correlate colors with an external key, accelerating understanding. In the same manner, labeling points directly in a scatter plot can render trends more obvious, avoiding ambiguity due to overlapping hues. Patterns offer another non-color-related method for distinguishing parts within a visualization. This works well for grayscale prints or where people have difficulty perceiving color differences. In bar graphs, for example, striped, dotted, or crosshatched patterns may be employed in place of (or in addition to) color to differentiate among various data groups. In geographical maps, land use types (urban, forest, and water bodies) may be symbolized with textured overlays such that each category can be identified even in the absence of color.

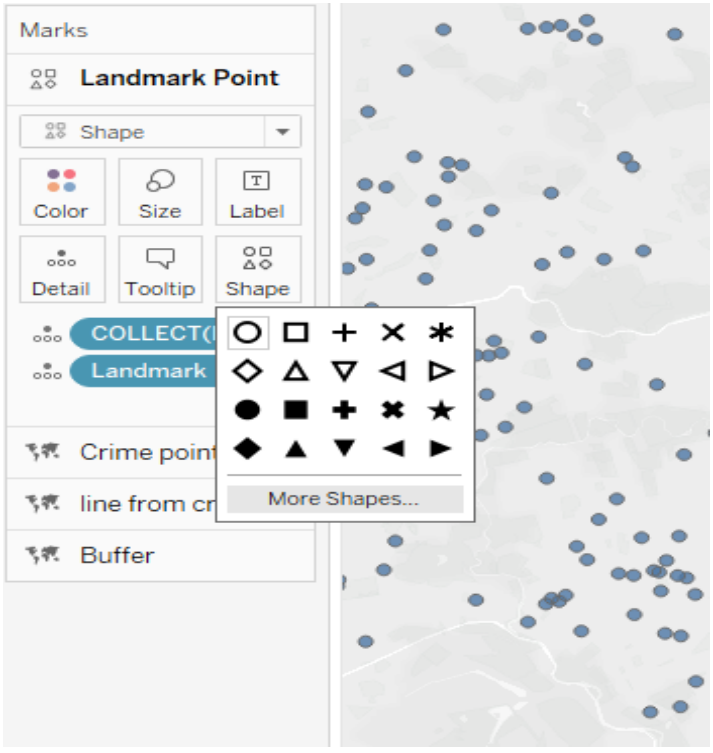


Fig. 5.7: Custom icons on map

Icons provide an intuitive means of augmenting data representation. Small graphic symbols can stand in or be used in addition to color coding as shown in fig 5.7, making visual information more understandable briefly. For instance, on a weather map, a sun symbol for sunny and a cloud for cloudy weather allows the forecast to be comprehensible even in black and white. Similarly, in financial reporting, using upward or downward arrows, in addition to red-green color coding, allows users to easily identify positive or negative trends [37].

Chapter 6

INTRODUCTION TO TABLEAU

Vikramjit Parmar[1], Debjit Mohapatra[2], Suruchi[3] and Shruti[4]

[1,2,3,4]GNA University, Phagwara

6.1 What is Tableau?

Tableau is a data visualization and business intelligence software that enables users to interactively analyze and visualize data. It makes it easy for non-technical users to develop dashboards, charts, and reports using a drag-and-drop feature as shown in fig 6.1. Tableau has several data sources that can be supported, such as Excel, SQL databases, cloud services, and big data systems [38].

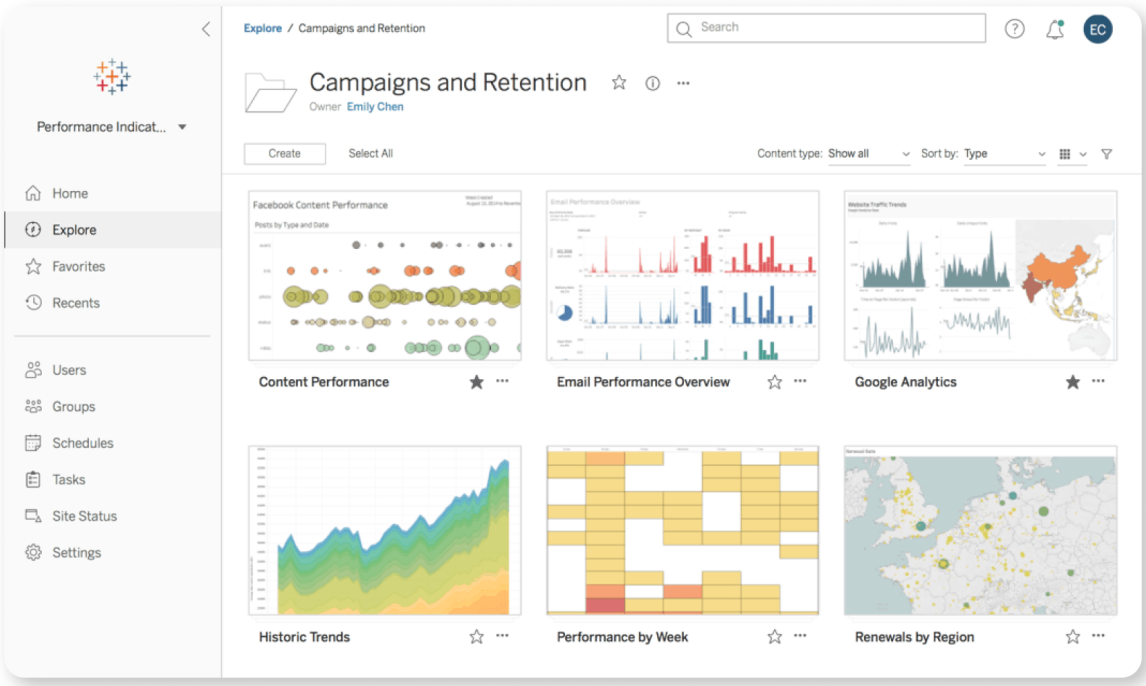


Fig. 6.1: Tableau dashboard

6.1.1 Key Features of Tableau:

1. Interactive Dashboards:

Tableau can generate interactive dashboards that allow users to filter, highlight, and analyze data interactively. Unlike static reports, interactive dashboards allow users to click on different elements, drill down into specific data points, and apply real-time filters to get more insights. For instance, a sales group can interact with a dashboard to break revenue data into regions, product category, or time frames and thereby view trends and patterns more clearly. This level of interactivity enhances decision-making since users are able to examine data from various perspectives without the need to produce several reports.

2. Real-Time Data Analysis:

Tableau has real-time connections to multiple data sources, enabling companies to analyze data in real-time. Through connections to databases, cloud environments, and APIs, Tableau ensures that dashboards and reports are constantly updated with the most recent data. This is particularly useful in industries where real-time data is critical, such as finance, healthcare, and supply chain management. For example,

an online store is able to track website traffic, sales performance, and customer behavior in real time, allowing them to make decisions on inventory, pricing, or marketing strategy immediately. This real-time capability allows organizations to stay agile and respond instantly to shifts [38].

3. Advanced Analytics & AI Integration:

Tableau also has advanced analytics features that are integrated, such as statistical modeling, forecasting, and insights based on AI.

All aspects of trend analysis, predictive forecasting, and cluster algorithms are able to utilize using non-coding skills. AI and machine learning models also become integrated using Tableau, such that entities make use of auto-detect anomaly, prediction model, and sentiment analysis across enormous datasets. As an example, a retail venture powered through AI can anticipate purchase patterns in the customer segment and adjust promotions using their predictive insight. With that, AI incorporation builds upon data-driven insight with revealing probable patterns one does not at first view through raw analysis.

4. Mobile-Friendly Design:

Tableau ensures that reports and dashboards are fully responsive and mobile-ready, i.e., accessible and consumable across multiple devices, e.g., tablets and smartphones.

Contrary to some other report-focused reporting programs requiring distinct mobile-consumption settings, Tableau dynamically sizes up the visuals for the availability of multiple display screens. This is most handy for executives and field agents that need access to essential business data when away from the workstation. For example, a sales regional manager can see actual time sales performance as well as customer patterns right on their mobile phone in order for them to make sound decisions even outside of their workplace.

1. Overview of Tableau Environment:

Prior to using Tableau for data visualization and analysis, it is important to set up the environment correctly. The initial setup process is important in ensuring seamless functionality, maximum performance, and effective management of data. Although Tableau has a user-friendly and intuitive interface, incorrect installation or configuration can result in performance problems, compatibility issues, and challenges in managing large datasets.

The installation process varies based on the Tableau product being used, either Tableau Desktop, Tableau Server, or Tableau Cloud. Users need to ensure that their system supports the required hardware and software requirements such as the availability of enough processing power, memory, and storage capacity, particularly when dealing with large data sets.

Moreover, correct data connection configuration is crucial. Tableau accommodates multiple data sources, including relational databases, cloud platforms, and spreadsheets. Configuring secure and optimized connections to these data sources allows data to be loaded, processed, and visualized effectively. By installing and configuring Tableau with care, users can optimize its performance, minimize possible errors, and produce smooth data-driven visualizations for enhanced decision-making.

2. Minimum System Requirements:

Before installing Tableau, make sure your system has the required hardware and software specifications.

Table 6.1: Operating System and Windows 10 (64-bit) or later/macOS 10.14 or later

Operating System	Windows 10 (64-bit) or later/macOS 10.14 or later
Processor	2 GHz or faster (multi-core recommended)
RAM	Minimum 8GB (16GB+ recommended for larger datasets)
Storage	At least 5GB of free disk space
Internet	Required for activation and cloud-based features
Database Connectivity	ODBS/JDBC for databases like MySQL, SQL Server, PostgreSQL

3. Setting Up Tableau:

The installation of Tableau is quite simple, but proper setup is critical for flawless functioning. Be it Windows or Mac, the process of installation of Tableau Desktop and Tableau Public remains more or less the same with minor variations in file running and directory configuration.

The initial step is to download Tableau from the website (tableau.com). Go to the Download page and select the proper version—Tableau Desktop for business reporting and analysis, or Tableau Public, a free version with limited features. To continue with the download, users are required to log in or register for a Tableau account. After downloading, installation is initiated by executing the installer file. In Windows, users are supposed to double-click the .exe file, whereas for Mac, they have to open the .dmg file and move the Tableau app into the Applications folder. Upon opening the installer, users have to read and agree to the End User License Agreement (EULA) to continue. Second, users have the option of selecting the install location. For Windows users, they can choose to install Tableau in a custom location if necessary, but Mac users directly install it inside the Applications folder by default. After installation, the software automatically asks users to log in. For Tableau Desktop, the user must insert their license key to activate it. For Tableau Public, the user will be asked to sign in with their Tableau account credentials so that they could access cloud-enabled visualization capabilities. Once logged in, the ultimate step is finishing up the configuration. When Tableau is opened, the Start Page is displayed, where users can navigate through available features and link to data sources like Excel files, databases, or live cloud-based data. With installation and configuration now done, users can start creating interactive dashboards and meaningful visualizations, making Tableau an effective tool for data-driven decision-making [39].

6.1.2 Navigation in Tableau:

1. Introduction to table navigation:

Tableau offers a simple and intuitive user interface for visualizing data as shown in fig 6.2. Effective navigation of Tableau is important to accomplish activities like importing data, creating visualizations, and managing dashboards. This chapter discusses Tableau's main components, their functionalities, and how users can use the interface smoothly.

2. Elements of the Tableau Interface:

The Home Page: The Start Page is the initial screen to be displayed upon launching Tableau. Recent workbooks, sample datasets, and several data connection options are readily available. Elements that constitute the Start Page: The Connect Panel allows users to connect to databases, cloud apps, and Excel, among others. You can find the most recent Tableau workbooks you opened under "Open/Recent Workbooks."

3. Navigation and Interface of Workbook:

Once you have opened a workbook or have created a new one, Tableau shows you its primary workspace, which includes a number of significant components.

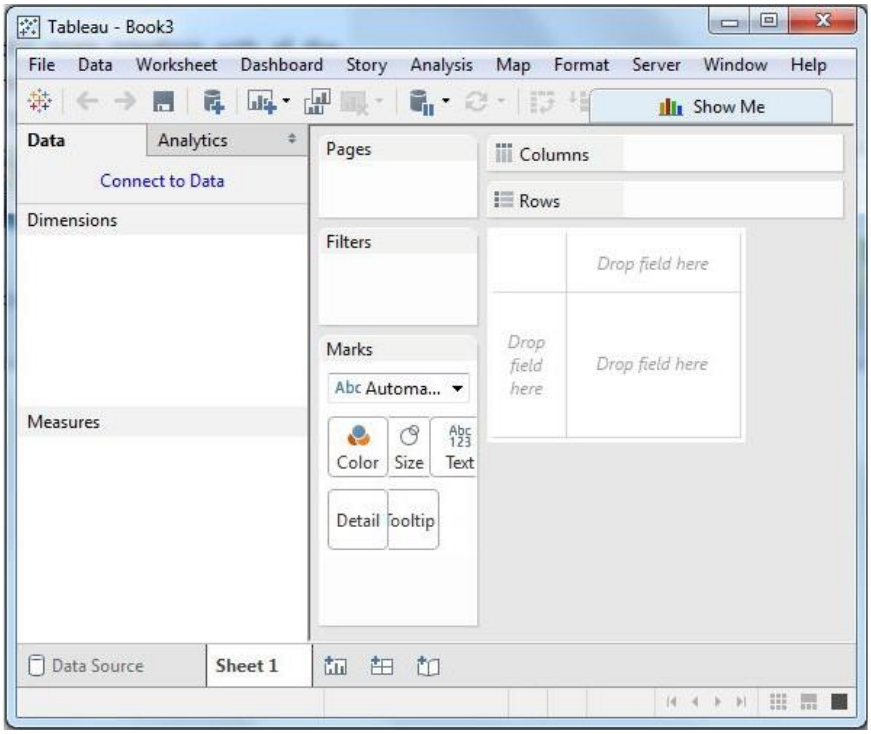


Fig. 6.2: Navigation in Tableau

1. Menu Bar:

Placed at the top, the Menu Bar holds drop-down menus for some of Tableau's functions.

Table 6.2: Menu bar

File	Open, save, print and export workbooks
Data	Connect, edit, or refresh data sources
Dashboard	Create and edit dashboards
Analysis	Add calculations, filters and reference lines
Map	Set up map settings and layers
Help	Use Tableau documentation and tutorials
Format	Change fonts, colors, and alignment
Server	Publish or download workbooks from tableau server
Worksheet	Work with single worksheets (rename, copy, clear)

2. Toolbar:

The Toolbar, which is situated beneath the bar menu, offers easy access to often-used features.

Table 6.3: Toolbar

New Dashboard	Create a new dashboard
Undo/Redo	Reverse or redo last action
Save	Save Workbook
Show Me	Opens Chart suggestions based on highlighted data
Formatting	Customize fonts, color and border Styling
New Worksheet	Add a new Worksheet
Zoom	Adjust visualization’s zoom level

3. The Data Pane:

Data Pane displays Dimensions (categorical fields) and Measures (numerical fields). Your data fields are separated into two categories by the Data Pane's most prominent part: Dimensions: Typically qualitative, category data such as names, dates, or geographic data. Blue icons are typically used to represent dimensions. Measures: Numerical, quantifiable information that can be summed and analyzed. Green icons usually illustrate measures.

Tableau applies this section to tell you which fields are employed to categorize and group your data (dimensions) and which fields can be counted or measured (measures).

4. Worksheets, Dashboards, and Stories:

On the bottom row, Tableau has three chief components for visualization creation:

Worksheets:

A sheet (or a "worksheet") is the base unit in Tableau, where a single graph, chart, or table enabling the user to analyze a given detail of data. A single worksheet is supposed to be working with one view, like a bar chart, line graph, or scatter plot. One of its strong points is interactivity, allowing users to apply filters, sort data, and emphasize important insights in the sheet directly. A workbook may include several worksheets, and users can analyze different views of the same data before combining them into dashboards or reports. Because worksheets are extremely flexible, users can test different visualization techniques before settling on their data presentation. To add a new worksheet, the users can just click the \"+\" icon at the bottom of the screen.

Dashboards:

A dashboard comprises more than one worksheet presented together on one screen as shown I fig 6.3, giving a wide overview of insights in the data. Dashboards allow users to aggregate findings across various worksheets, facilitating easy comparison and correlation of data. Dashboards also have interactive capabilities like filters, dropdowns, and highlight action, allowing users to browse and analyze data interactively. With a flexible layout, dashboards can be resized, rearranged, and formatted to enhance readability. Additionally, when connected to a live data source, dashboards update automatically, ensuring real-time data visualization. Users can fine-tune the layout by using the Dashboard Layout Panel, which provides customization options for adjusting component sizes and positions [40].

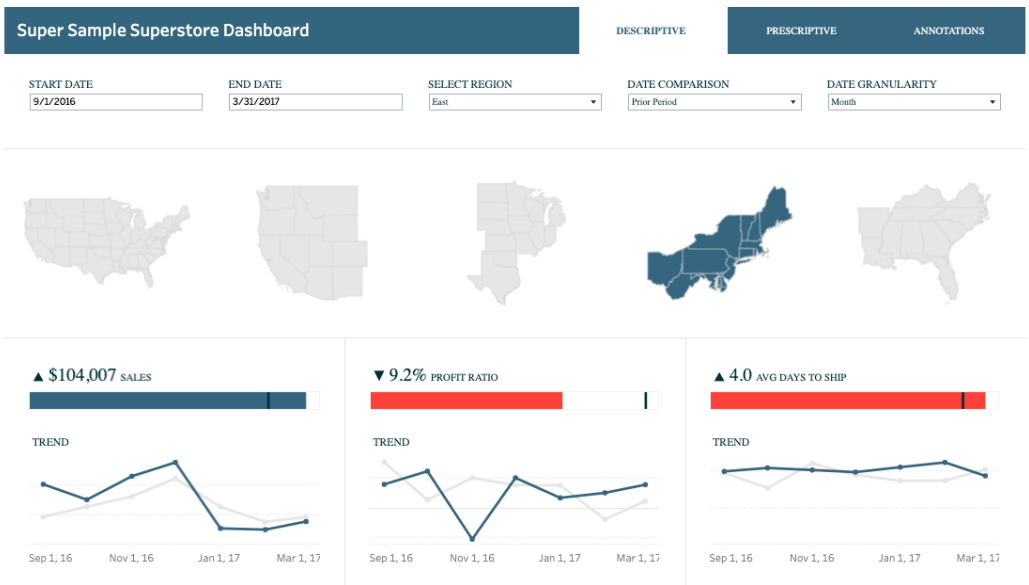


Fig. 6.3: Dashboards in tableau

Stories:

A Tableau story is a designed series of worksheets and dashboards, viewed as a slideshow to convey insight in a methodical, step-by-step order. This aspect is especially applicable for data storytelling, where the results are formatted chronologically to take an audience through a tale. Every point in the story (or slide) has a dashboard or worksheet, serving to break down involved data into manageable insight. Stories also have interactive components, which allow users to investigate various parts of the data within the story. Users can create a story by dragging and dropping worksheets or dashboards into the Story workspace, with a seamless and directed flow of information.

6.1.3 File and Data Types in tableau:

Types of Files and Data in Tableau:

Tableau's file and data types: Tableau is a robust data visualization tool that works with a wide range of file and data formats. To properly process, visualize, and analyze data, they must be understood. This section covers the kinds of files that Tableau can handle, the kinds of data that Tableau can use, and the best ways to handle them.

6.1.3.1 File types in Tableau:

Workbooks, data sources, extracts, and packaged data are all arranged into several file formats by Tableau. These formats are divided into two categories: those specific to Tableau and those for external data sources.

1. File Types Particular to Tableau:

Tableau uses its file extensions to store and share data visualizations effectively.

Table 6.4: File types particular to Tableau

File Type	Extension	Purpose
Tableau Data Source	.tds	Saves connection information but not the data
Tableau Packaged data source	.tdsx	Saves both connection information and pulled data
Tableau Workbook	.twb	Saves visualization but not data
Tableau Preferences	.tps	Saves custom color palettes for consistent styling
Tableau Workbook	.twb	Stores visualization but not data
Tableau Map Source	.tms	Saves customized geographic maps
Tableau Data Extract	. hyper/.tde	Compressed, optimized snapshot of data performance
Tableau packaged Workbook	.twbx	Saves both visualization and data for convenience

2. Formats of External Data Sources:

There are many different external data sources that Tableau may connect to. Typical file formats include the following:

Table 6.5: Formats of external data sources

File Type	Extension	Purpose
Comma-Separated Values(CSV)	.csv	Common text-based tabular format
Microsoft Access Database	.mdb, .accdb	Links access database files
Microsoft Excel	.xls,.xlsx	Connect to Excel Wrokbooks
Text files	.txt	Uses tab or other delimiters for tabular data
Google Sheets	Online	Link to cloud spreadsheets

Data Types in Tableau:

1. Primary Datatypes:

Table 6.6: Primary datatypes

Datatype	Description	Example
Date	Date values without time	"16-02-2001"
Date & time	Contains both data and time	"16-02-2001 12:30 P.M"
Boolean	Trues/False values used for logical operations	"In Stock: True/False"
String(text)	Categorical data represented as text	"Product name", "Category"
Geographic (Location)	Geographic fields like Country, lattitude	"USA", "New York", 28.718° N
Number(Whole/Decimal)	Numerical values(both whole and decimal)	200, -25, 3.2

2. Automatic Data Type Identification:

For proper analysis, you might have to change data types sometimes. With data transformation or calculated fields, Tableau provides you with the facility to alter the data type of a field.

Example1: Conversion of String to date:

If data filed contains a string("21-02-2001"), you can convert to date using:

DATEPARSE("yyyy/MM/dd", [Date_String])

Example 2: Conversion of string to number

If numeric field is stored as text(eg. "254"), you can convert to number using:

INT([String_Field])

3. Discrete and Continuous Data:

Discrete data is data that falls within definite groups or categories. It cannot be meaningfully divided. In Tableau, it is generally represented in blue. Examples are customer names (e.g., "John," and "Michael") and product categories (e.g., "Furniture" and "Clothing"), which are all discrete data examples. Discrete data is categorical or countable in nature. There are no meaningful values in between (e.g., "half an employee ID" is not a valid value). It is utilized for filtering and grouping data. Continuous data is measurable values which may have any value in a range. It is numeric and may be in fractional or decimal form. In Tableau, it is generally shown in green color. It is always numeric and measurable. There are infinite possible values in a range. It can be subdivided (for example, time can be subdivided as hours, minutes, and seconds). Examples of Continuous Data: Age (such as 18, 21.5, 30.7), Time (such as 12:00 A.M, 2:02 P.M) .

6.1.3.2 Data Source:

In Tableau, a data source is the basis for producing visualizations. It is where the raw data is located, which Tableau processes, analyzes, and interprets into valuable insights. A data source can be from manually constructed datasets, Excel files, cloud services, or enterprise databases. The ability of Tableau to work with various data sources is what makes it a formidable business intelligence and data analysis tool [40].

Type of Connection:

Tableau provides the capability to connect to data in two manners: Live Connection and Extract Connection. Live Connection gets data directly from the source in real time so that visualizations can be up to date with the latest updates. This is ideal for situations where data constantly changes, e.g., analyzing stocks or observing live sales. But performance relies on network efficiency and database speed. Conversely, an Extract Connection builds a static snapshot of the data, keeping it locally for performance optimization. Extracts enhance processing efficiency by minimizing repeated database queries, thus being perfect for use with big datasets or where real-time updates are not needed.

Relationships and Joins:

Tableau allows users to handle several tables by building relationships or joins among them. Relationships offer a dynamic and adaptable method of linking data without first defining join conditions, enabling Tableau to choose the optimal approach depending on visualization requirements. In contrast, joins define tables to be merged using Inner, Left, Right, or Full Outer Joins, which supports more organized data combination. These methods are necessary when joining data from multiple sources, like associating customer profiles with transaction records for thorough analysis.

Data Types:

Tableau classifies data fields by their type so that they can be processed and visualized appropriately. Typical data types are text (string), numeric (integer or decimal), date & time, boolean (true/false), and geographical data. Data types are important to know for proper calculations, sorting, and filtering. For instance, date fields allow time-series analysis, and numerical data allows aggregations such as sum, average, or count.

Aggregations and Filters:

Prior to the construction of visualizations, data may be preprocessed to be clear and relevant. Aggregations condense data into totals by employing operators like SUM, AVG, MIN, MAX, or COUNT in order to detect trends and patterns. Rather than representing raw transactional data, aggregations may represent important insights, including total revenue by location or mean sales by month. Filters, however, narrow down datasets by removing irrelevant or undesired data points. Users can utilize filters by date ranges, numeric thresholds, categorical choices, or custom criteria to narrow down particular insights, for example, to show only the best-selling products or to filter data for a specific time interval.

6.1.4 Custom Data View:

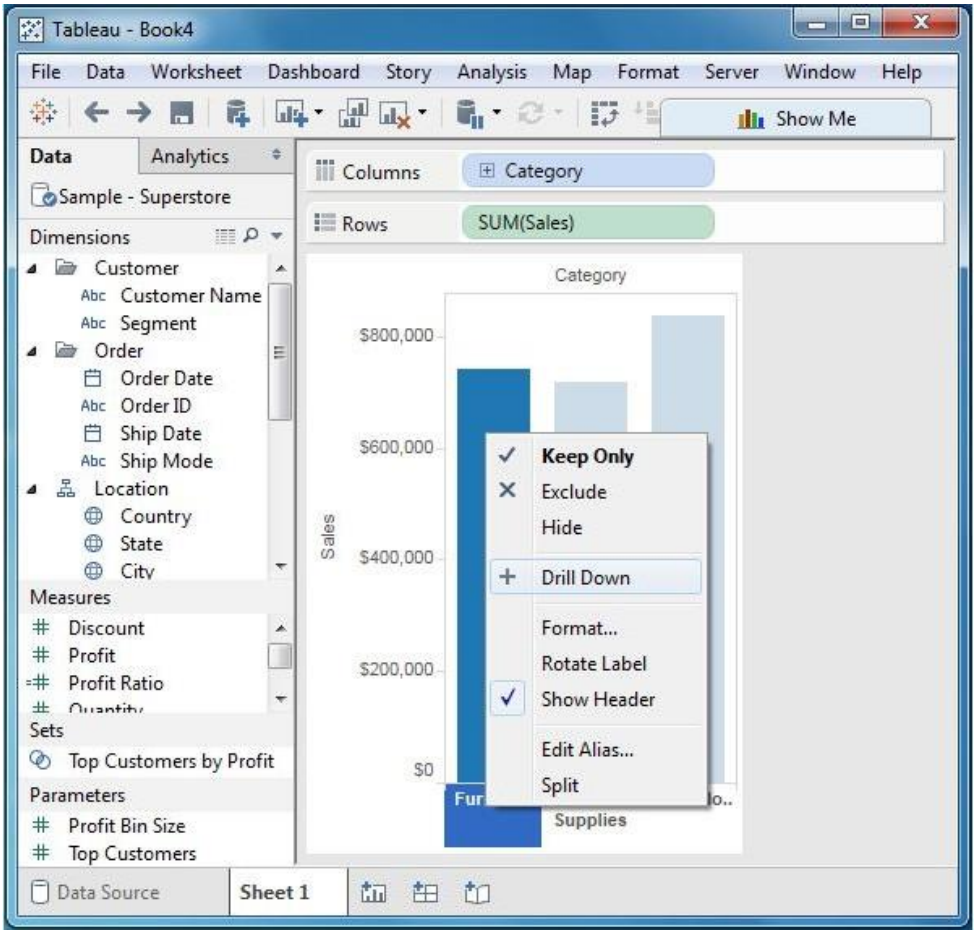


Fig. 6.4: Custom Data View

Custom Data View feature enables you to alter, restructure, and customize the data source view prior to utilizing it for visualization. By using filtering, renaming, creation of calculated fields, definition of data types, and organization of data connections, it enables users to personalize and refine data as illustrated in fig 6.4.

A Custom Data View in Tableau enables users to alter and refine a dataset prior to developing visualizations. The process entails cleaning and formatting data, combining several tables by joining or connecting them, eliminating unwanted data, designing calculated fields for bespoke measures, and field renaming or modifying data types. By pre-refining the dataset, users make sure that the data is thoroughly structured and geared for effective analysis.

Why a Custom Data View is Necessary?

Building a Custom Data View assists in eliminating unnecessary data clutter, making visualizations more effective by loading only necessary information. It also improves dashboard performance, with only the necessary data being processed. Data integrity is enhanced by eliminating inconsistencies and filtering out errors, and the dataset's flexibility is increased through the application of custom SQL queries and calculated fields. These modifications make the dataset more dynamic and flexible for advanced analysis.

Example of a Custom Data View in Tableau:

Suppose there is a sales dataset that has fields that do not pertain to sales such as "Customer Phone Number" or "Order Processing Time." Rather than handling a messy dataset, you have the ability to hide unimportant fields, set filters on a given year of data, and add a new field for the profit percentage. Through such cleansing of the dataset, it will become tidy, more efficient, and more revealing, thus informing sounder decisions.

6.1.5 Extracting data:

Data extraction in Tableau refers to the method of pulling a smaller portion of data from a large dataset and saving it in a specific extract file format. This practice alleviates pressure on live databases, enhances performance, and facilitates easier offline access. By storing a compressed version of the data, extracts allow Tableau to work with massive datasets effectively while speeding up queries and interactions.

Data extraction offers several advantages:

Data extraction provides loads of benefits when used in Tableau, chief among which is performance. Since Tableau does not request data from live databases each time a query is executed, this intrinsic feature positively impacts the performance of working with extracted data. The better the performance, the faster the query execution, easier the loading of dashboards, and seamless interaction enjoyed by users with their visualizations. This advantage is an important aspect when working with more extensive datasets or complex calculations.

Offline availability is another key benefit. When a lot of extracted data is available, users can utilize their datasets without the availability of an internet connection or a live database. This is most helpful to professionals who need to analyze data while on the move or during remote work assignments. With locally saved extracts, users can create reports and further insights without worrying about constant access to a live database connection.

Data extraction adds to information security from another angle. Organizations can restrict access to sensitive data by storing it in a local repository. Extracts can be shared with users authorized to access that particular data without giving them access to the entire live database. This lowers the chances of a security breach.

Another advantage extends to streamlining and organizing data. Extracts have only relevant data needed to maintain a tidy dashboard and free of clutter. This way, destructured data is cleaned, thus making it easier for users to analyze data trends and obtain useful insights.

In addition, the practice aids in reducing the load on servers. Since Tableau holds extracted data locally, the multiple queries sent simultaneously from many users to the live database are spared. This ease loads on database servers and helps keep the system efficient. Any organization that relies on real-time analytics can hugely benefit from extracts by ensuring the availability of live data sources for core operations, while for reporting and analysis, Tableau users can use extracts as depicted in the fig 6.5.

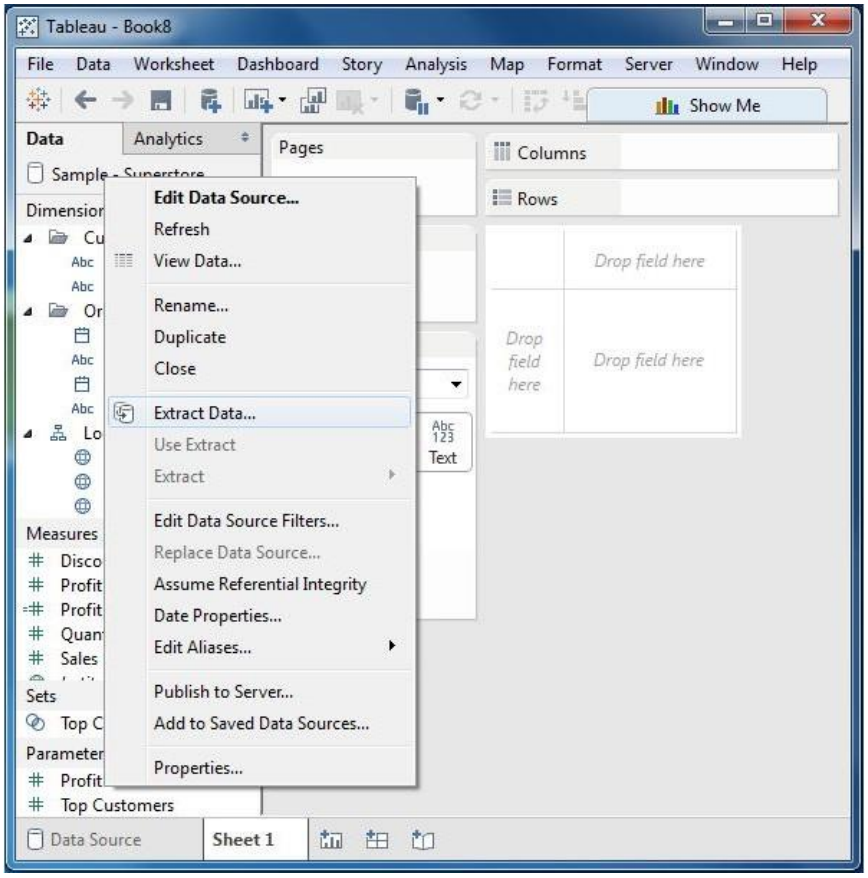


Fig. 6.5: Extracting of data

How to Extract Data from Tableau?

Extracting data from Tableau is a process essential for every user to optimize performance, enhance analysis, and work offline with the data. Data extraction is one of the procedures in Tableau that retrieves data from various sources and saves it in an optimized format for better visibility efficiency and dashboard performances. It is a very useful process especially when dealing with large datasets as the extraction can hasten query execution and reduce dependence on live connections.

Step 1: Connect to the Data Source:

To start the data extraction process, open Tableau and connect to the data source. This can be done by going to File > Connect to Data once Tableau is up and running. Users can choose to connect to a whole variety of data sources including local files such as Excel or CSV, anything from cloud-based storage, or any kind of relational database like SQL, or even enterprise-level data warehouses. Select the source wisely to ensure that the extracted data is as relevant and reliable as possible towards the data analysis planned.

Step 2: Choose the Extract Mode:

Now that the data connection is established, it's time to decide whether to conduct all operations using a Live Connection or an Extract. A live connection will continuously pull real-time data from the source, which can be exhausting in terms of resources. Instead, choosing Extract will allow Tableau to save a snapshot of the data locally and improve the speed of any query, thus lessening the load from accessing the original data source. To toggle from the Live feature to an extract mode, just select Extract from the Data Source column.

Step 3: Set the Preference for Extraction:

The extraction preferences before finalizing are to configure extraction settings to optimize performance. Clicking Edit on the extract settings allows users to set filters, aggregation, and limits on data to be retrieved. Extract filters can be to include only the required records, which can also help avoid extraneous data extraction. For example, in a dataset that contains sales records spanning multiple years, a filer can be applied for sales records in 2023 and thereafter, reducing the extracted size but still very relevant.

Step 4: Save and Extract:

After setting the extraction parameters, the extract file needs to be saved in a location. Clicking on Extract Data will begin the extraction process and save the data in Tableaus extract formats of .hyper or .tde. The .hyper format is the latest and more optimized version of extract file types, promising high performance and good handling of large datasets. The extracted file acts like a stand-alone dataset and can be used within Tableau without having to keep a constantly updated connection with the original data source.

Step 5: Open Tableau and Import the Extracted Data:

As the extracted file in the course of the data extraction process, users will be able to import it into Tableau. Develop dashboards, do visualizations, and perform in-depth exploration of the data using the extracted dataset. Working with a

6.2 Field Operations:

Field manipulation is one of the essentials in Tableau for preparing data for analysis and visualization. Organizing, resizing, and managing field are effectively made possible using these operations. In tableau, fields are collectively referred to as columns which are classified as different types based on their nature and function.

Dimensions are qualitative data that can categorize and segment data from the set. These include dimensions like product name, category, country, and the like. They are non-mathematical, but they help organize the data and group it. However, measures consist of the quantitative values which either can be aggregated or can be used in mathematical operations on them. With measures, we can take quantity, profit, and sales. These will generate our key performance indicators as well as statistics.

Besides these elements, there are also calculated fields, which allow a user to plug into a formula using existing fields as the basis for customizing a calculation to suit a specific application. This feature is handy when a user needs further transformation or business logic to be applied but does not want to change the dataset or the source of data in the background. Another area of Tableau field operations is aggregation. Fields that have been processed through sum, average, count, minimum, or maximum operations above are known as aggregated fields. The advantages of aggregations are that they can summarize huge data sets into smaller, readable segments.

6.2.1 Regular Field Operations in Tableau:

Field operations are very important in tailoring, augmenting, and restructuring data to provide the best possible visualization.

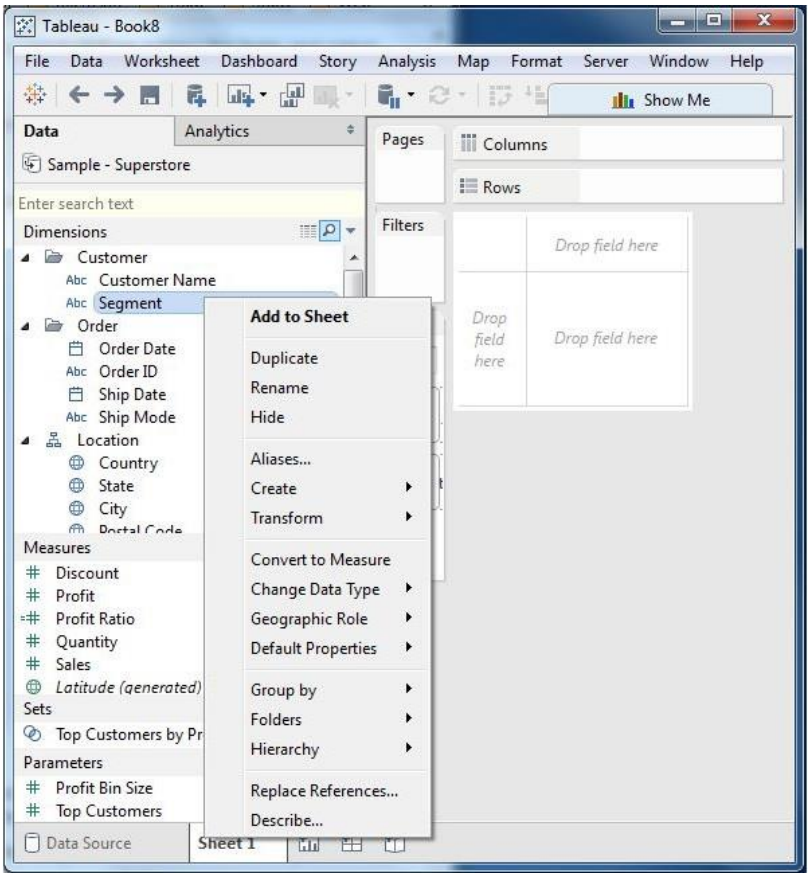


Fig. 6.6: Field Operations

1. Field renaming:

By default, automatic names for fields are generated based on the available data. In the Data Pane, right-click a field and click Rename to rename it. For example, renaming Cust_Name to Customer Name provides more clarity as shown in fig 6.6.

2. Altering the Data Types:

Occasionally human intervention is necessary, although Tableau often identifies data types automatically. As an example, Order Date should be altered from String to Date. Sales ID: Conversion of String to Numeric.

To alter the data type: Click the icon for the data type beside the field to choose the correct type.

Calculations and Field aggregations:

Aggregation is the method in which data is summarized into SUM, AVG, COUNT, MAX etc.

Table 6.7: Calculations and field aggregations

Type	Description
SUM ()	Sums up all the Values
AVG ()	Calculate average data values
MIN ()	Return the smallest value
MAX ()	Return the greatest Value
COUNT ()	Count number of records

6.3 Metadata in tableau:

Metadata in Tableau is the descriptive and structural data regarding a dataset that assists users in organizing and understanding their data effectively. It consists of field names or column headers, which identify the labels for various data attributes. These names are important for providing clarity and organization while handling datasets. Every field in Tableau is given a data type, e.g., String, Number, Date, or Boolean, that dictates how the data is computed and rendered. Choosing the right data type is important because it impacts calculations, filtering, and sorting in the visualization.

Metadata also involves field properties, which determine if a field is of type Dimension or Measure and whether it is Discrete or Continuous. Dimensions hold categorical data like names or locations, while Measures store numeric values to be calculated. In the same manner, Discrete fields are used as distinct individual values, whereas Continuous fields are stored as a range, often applied in graphs and time-series analysis. It also includes calculated fields and aliases, through which users can derive new derived fields against existing data. Hierarchies and relationships also expand metadata by specifying relationships between tables and defining structured drill-downs, e.g., drilling down sales data into region, country, and city. Metadata in Tableau can be edited by users through renaming fields, data type modification, proper formatting, and organizing the dataset for performance and analysis. These improvements make data clean, well-structured, and optimized for effective visualizations.

6.3.1 Editing metadata in tableau:

1. Renaming Fields:

The data source may have field names with ambiguous labels, spaces, or abbreviations. Renaming fields makes them easier to use and comprehend.

To change a field's name:

Navigate to the Data Pane in Tableau > Right-click on the field you wish to rename > Select Rename and input the new field name.

Example:

Change	cust_id	→	Customer	ID
Change	ord_dt	→	Order	Date

This enhances the readability and user-friendliness of the dashboard.

2. Converting Data Types:

Tableau performs automatic data types, but certain modifications may be required to carry out correct analysis. To modify a data type:

Open the Data Pane > Select the data type icon beside the field name > Select the proper data type from the list.

Example:

Change 20240101 (string) to Date Format (YYYY-MM-DD).

Modify the Discount from Integer to Decimal to calculate exactly.

Incorrect data types can cause analysis errors (e.g., analyzing Order Date as text rather than a date).

3. Changing the Default Properties:

Tableau allows you to change the default properties of fields, such as currency type, number format, and aggregate. To change the aggregation by default:

Select a Measure field (like Sales) with a right-click > Select Aggregation under Default Properties > Select SUM, AVG, COUNT, MIN, MAX, and so forth.

To change the format of a number: Right-click on a numeric field (e.g., Profit) > Click Default Properties → Number Format > Select Currency, Percentage, Decimal, etc.

Example:

Alter Profit to Currency Format ($1,000.00).

Alter Discount to Percentage Format (15%).

4. Custom Labels, or Naming Aliases:

Aliases enable the renaming of certain field values for better readability. For instance:

California, New York, Texas can be used in place of CA, NY, TX. It is possible to rename "Male" and "Female" to "M" and "F" for better brevity. To make aliases: Select a field (like State) using a right-click > Select the option for Aliases > For each value, enter a unique name. This enhances the clarity and quality of the data shown in dashboards.

5. Creating Structures:

Hierarchies organize fields systematically to facilitate drill-down analysis. A sample of a geographical hierarchy is: Area → Nation → State → City. Below are steps to create a hierarchy: You may drag one field, like Country, onto another field, e.g., Region. Tableau will ask you to build a hierarchy. Insert more levels and give a name to the hierarchy. Users are now able to dynamically.

6. Dividing and Combining Domains:

Occasionally, a field contains multiple pieces of information that need to be separated.

For example: "John Doe" is a client that may be separated into "John" as the first name and "Doe" as the last name. To divide a field:

In the Data Pane, right-click on the field. Select the Transform → Split option. Split 1 and Split 2 are new fields that Tableau will generate. Calculated fields can also be used to merge (concatenate) fields. Example: Merging the First Name and Last Name

6.4 Tableau Worksheets: A Comprehensive Analysis

The foundational components of Tableau, where users perform data visualization and analysis, are worksheets. These serve as a workspace for users to merge data fields, implement filters, and apply various analytical methods to create distinct visualizations.

Understanding the Definition and Core Concepts of Tableau Worksheets:

In Tableau, a worksheet represents a single space for visualization that consists of the following elements:

- The main visualization area
- Cards and shelves for adding fields (Columns, Rows, Filters, etc.)
- The Data pane showing the available fields as shown in fig 6.7.
- The Marks card for controlling visual elements
- The Analytics window for advanced calculations

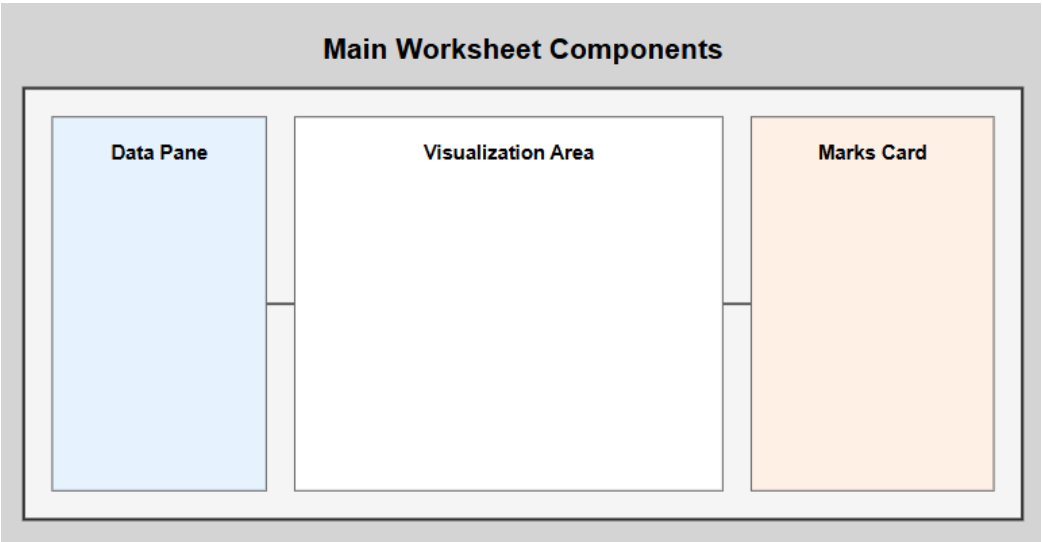


Fig. 6.7: Main Worksheet Components

1. Understanding the Visualization Space in Tableau:

The primary visualization area in Tableau is the centre of the worksheet, where you visualize charts, graphs, or maps. This is an empty canvas for transforming raw data into meaningful information. The data presentation based on how various fields are placed on different shelves uniquely shapes the corresponding view that breathes life into the story of its data.

Besides being an exhibition floor, this visualization space is also an interactive space where the user interacts with data points. Instead of passive images, Tableau makes data objects interactive—for instance, the user might click a bar in a bar chart to filter the related data or hover over a point in a scatter plot to glean detailed insights. All these interactions make data exploration more intuitive and user-friendly. It provides many visualization options, including bar graphs, line charts, scatter plots, maps, pie charts, etc. With this flexibility, a user may opt for the best visual form for his dataset and the kind of insight he wants to explore. Such flexibility is one of the strongholds of Tableau, enabling personalized and meaningful data storytelling. Another strong feature of Tableau's visualization space is the immediacy with which it responds to changes made by the user or underlying data. Filtering, changing calculations, or working with a dashboard visually refreshes the Tableau views right away in a seamless and dynamic manner. With this real-time flexibility, end users are given instant feedback, thus enabling continuous iteration through their data exploration.

2. Shelves:

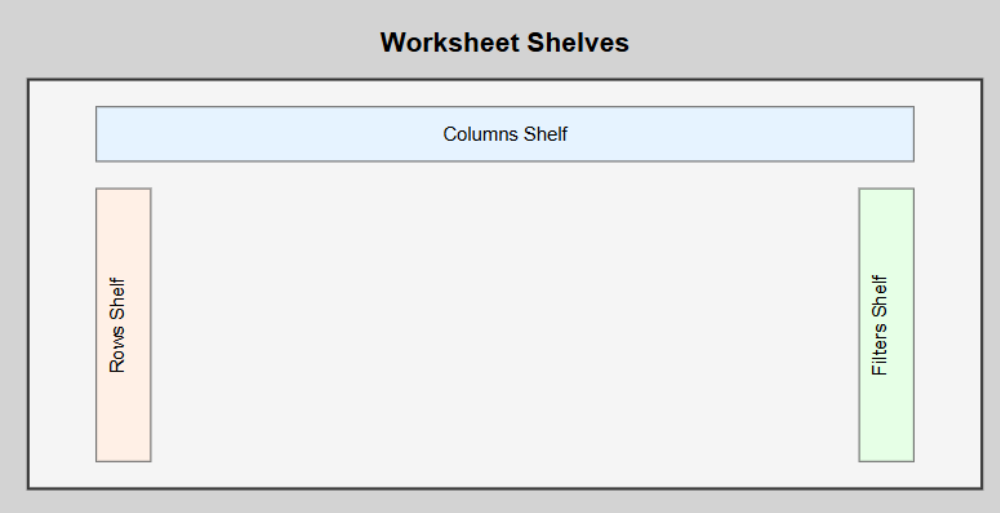


Fig. 6.8: Worksheet Shelves

Shelves in Tableau are of the utmost importance as they format and organize field data for easy visualizations. Shelves are horizontal panels onto which the user drags data fields from the Data Pane; they determine how data will be formatted in a chart or graph as illustrated in the fig 6.8.

Hence, the Columns Shelf will be the source for the horizontal axis, while the Rows Shelf will determine the vertical axis of the visualization, dictating the general orientation of the chart. Filtering is another mechanism through which data can be narrowed and managed. Users can limit what is visible on the visualization based on these specified conditions. Therefore, only information of interest shows up and thus provides clarity and concentration. The Pages Shelf conveniently divides data into categories or segments and produces separate views that can easily be navigated back and forth. This ability is particularly useful when looking for trends over time or differences between groups. The Details Shelf adds more pieces of information that may not be immediately visible in the main view but can be

accessed via interactivity, such as in tooltips or drill-downs. By carefully placing data fields on these shelves, users can customize their visualizations, add interactivity, and significantly improve the way they present data in Tableau.

3. Marks Card:

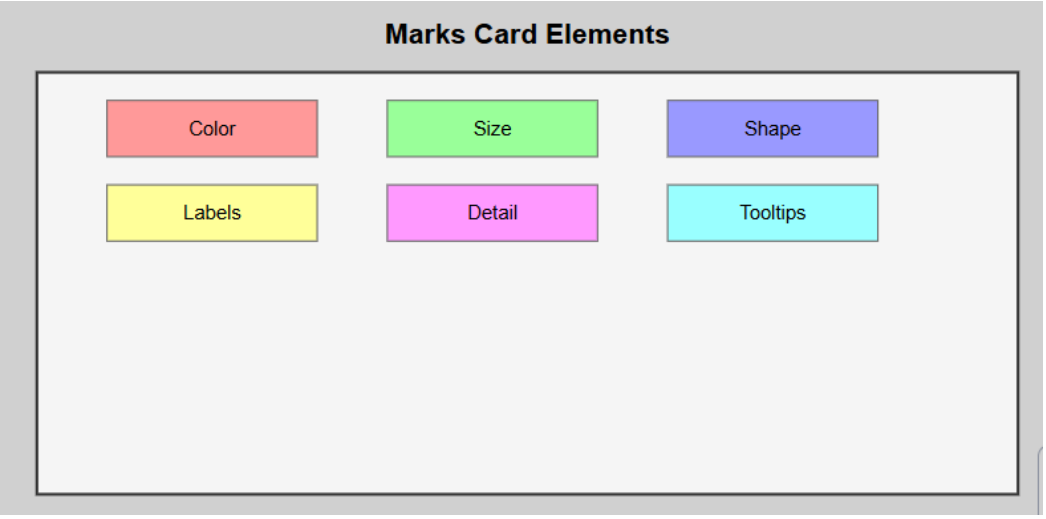


Fig. 6.9: Mark Card Elements

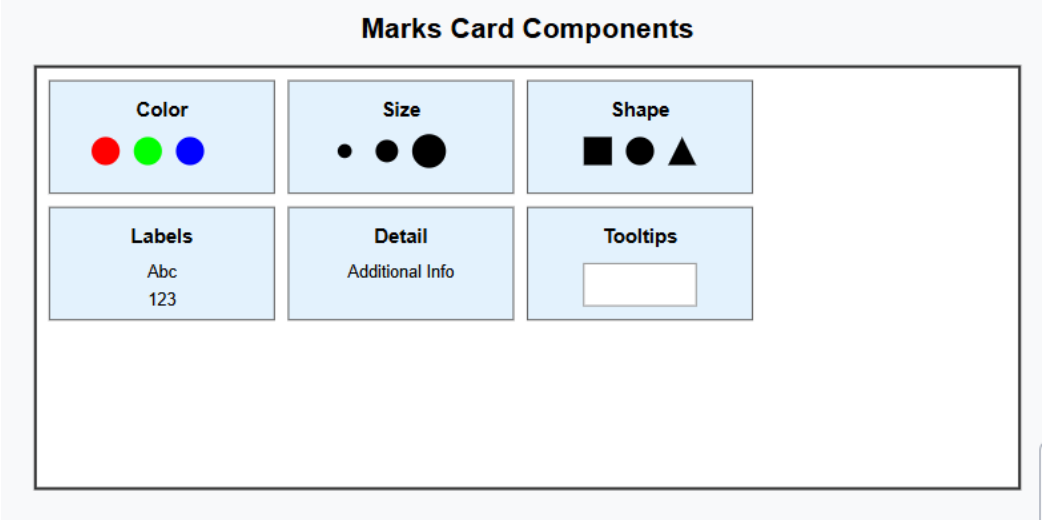


Fig. 6.10: Mark Card Components

The Marks card is a highly effective tool within Tableau for controlling the data point appearance and behaviour within a visualization. It offers several points of customization so that users can go deeper with their insights and improve storytelling with their data. One of its most important features is the Mark Type, which allows users to select how data is visually portrayed, for example, circles, squares, bars, or lines. This determines the overall appearance of the visualization. Color is also a vital feature that enables users to define different colors for data points based on dimensions or measures, which makes it simple to tell apart categories or accentuate differences in data. The Size option scales marks according to a measure and assists in depicting different magnitudes of data. The Shape feature allows the application of various symbols to classify data points, and is of special use in distinguishing among groups as shown in fig 6.10.

For further details, the Detail function enables users to add more information to tooltips without overloading the primary visualization. The Label feature assists in showing data values on the chart itself,

improving legibility. Finally, Tooltips offer the ability to tailor the pop-up text that appears when marks are hovered over, providing users with easy access to precise information without overloading the visual design.

4. Data Pane:

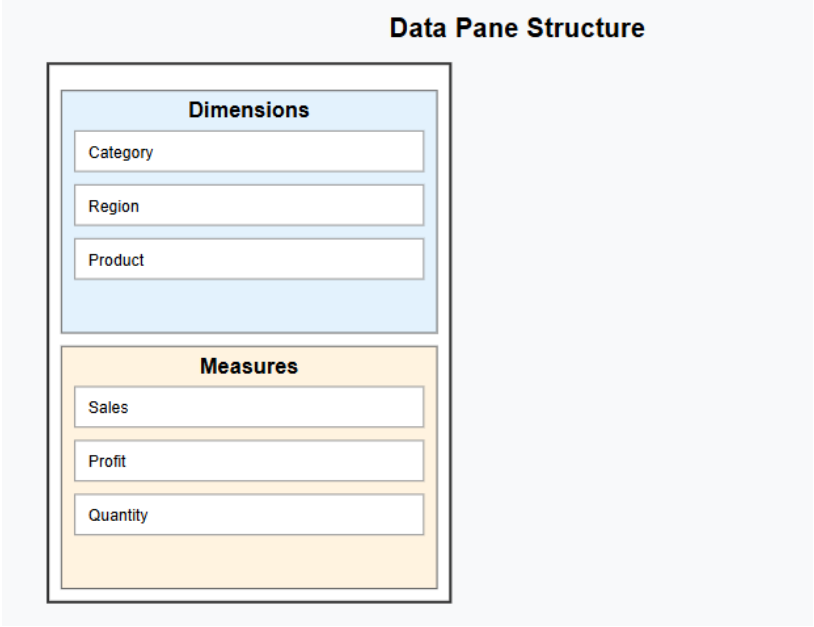


Fig. 6.11: Data Pane Structure

The Data Pane of Tableau is the major working window to access all the fields of a connected data source. It would display all columns from the data set and would thus prove to be the first step toward visualization. Fields have different values and measures and are therefore needed for organizing data, which is analyzable by users.

In Tableau, fields are categorically classified into two broad categories: Dimensions and Measures as illustrated in fig 6.11. Dimensions are categorical data such as product names, customer segments and dates. Measures are considered to be numerical values such as sales revenue and profit considered for the purpose of calculation and aggregation. The Data Pane organizes intuitive exploration and visualization of data through easy dragging and dropping of fields onto shelves and cards. It allows datasets to be kept to scale using a systematic structure, resulting in less cumbersome and effective data manipulation and insightful analytics.

6.4.1 Creating a Tableau Worksheet:

A Tableau worksheet is the best environment for users who build visualizations and handle data fields. The worksheet includes critical components, including mark cards, Columns, and Rows shelves, enabling users to manipulate data into a meaningful chart, graph, or table. To create a new worksheet, first, open Tableau and connect it to your desired data source. Once you connect, head to the bottom of the interface and click on the New Worksheet tab, which resembles a blank sheet icon. This opens up a new worksheet, where you can begin developing visualizations. After opening a worksheet, all available fields from the dataset are shown in the Data Pane on the left. Users can easily drag and drop fields into the Columns and Rows shelves to layout the visualization. The Marks card allows for further customization through color, size, and labels.

Creating a Tableau worksheet is an embodiment of an interactive-oriented process, enabling users to investigate and analyze data with insightful perspectives wherein clear-cut and comprehensive visual forms are created as illustrated in the fig 6.12.

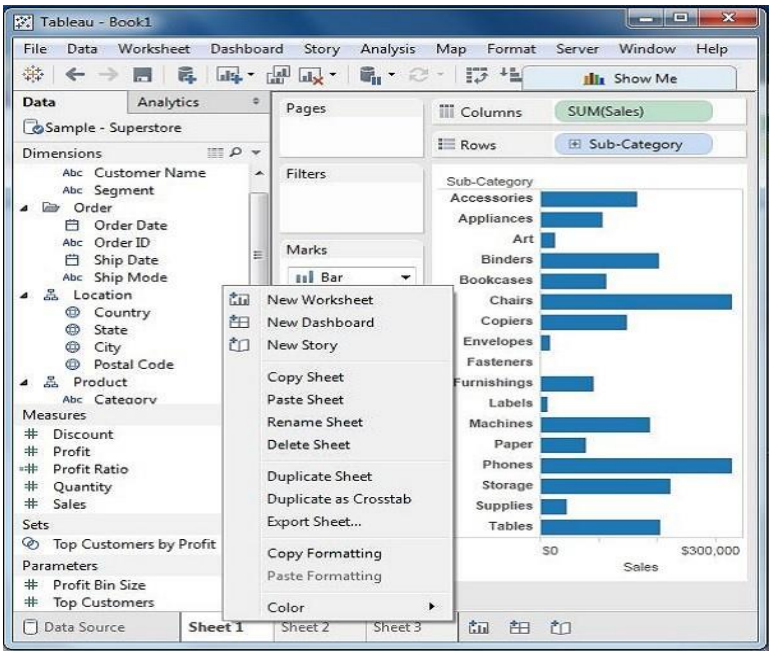


Fig. 6.12: Creating a Worksheet

Customizing Worksheet Visualizations:

Tableau gives users plenty of options for customizing visualizations to improve readability, clarity, and engagement in data presentation. Another way to modify the visualization is to change its chart type. The "Show Me" panel in the top right corner gives several choices of charts, such as bar charts, line charts, scatter plots, and heat maps. Users can use the drag-and-drop interface to switch between formats and find the one that communicates their data effectively. Filters are another major customization option that refines data. When users drag a field onto the Filters Shelf, they can control which data points appear in the visualization, thus focusing on specific trends or comparative aspects. Filters assist in segmenting the data based on a condition, such as a date range, a product category, or a numerical threshold. The Marks card gives additional enhancements to the visualization by giving users the ability to change color, size, labels, and tooltips, among other things. For example, colors could be changed to distinguish one class from another; size of marks could highlight differences of importance in the data, and labels could present only relevant values.

Users can right-click on axis labels and choose Format to customize labels and formatting. This allows them to modify fonts, colors, number formats, and alignment, thereby making sure that the visualization is appealing and easy to interpret.

Chapter 7

CREATING DATA VISUALIZATION IN TABLEAU

Suruchi[1], Gagandeep Singh[2] and Arshdeep Singh[3]

[1,2,3]GNA University, Phagwara

7.1 Building Common Charts:

7.1.1 Bar Chart:

Bar charts are also known as bar graphs; these have been used since the advent of data visualization at the dawn of civilization, in universities to display categorical data in forms of bars, usually rectangular. It represents the two categories, which are usually segregated in their fields; the height or length of each rectangular section is proportional to the numerical value they are representing. It can either be horizontal or vertical depending on how the two fields are categorized on their axes. Bar charts are very simple yet extremely effective for displaying a comparison of various data sets. It allows fast interpretation of trends and spotting patterns and comparison of values through various categories. From analyzing an enterprise into business analytics, in financial reporting and into research, bar charts are intuitively depicted into a way of interpreting data.

Creating Bar chart in Tableau:

Connect to your source data in Tableau at first. Then you will drag your categorical field, such as "Product Category" or "Region," into Rows or Columns. After that, grab your measure (for instance, a numeric field like "Sales" or "Profit") and drag it to the other shelf. Next, you'll select the appropriate type of visual from the Marks card, that is here "Bar." Your result will be automatically adapted into bar charts by Tableau, representing categories with their numerical values. Further changes can be made, such as applying color, size, filter, or adding labels, to improve understandability.

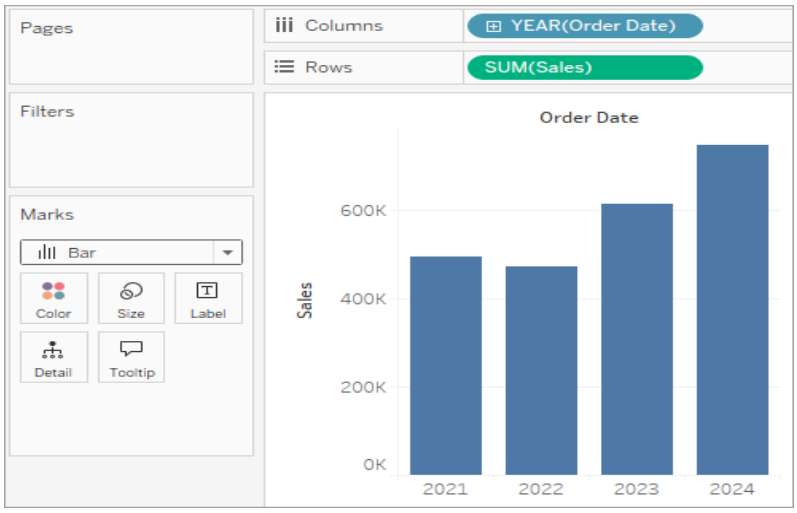


Fig. 7.1: Bar Chart in Tableau

This bar chart (as shown in fig 7.1) illustrates the trend of total sales over four years, from 2021 to 2024, based on the order date. The X-axis is used to represent the years, and the Y-axis is used to represent the sum of sales from 0K to 600K. The bars are used to represent the sum of sales of a given year. The graph clearly shows an increasing trend in sales throughout the years, with a dramatic rise from 2023 to 2024. The simplicity of the bar chart enables one to grasp the total sales performance over this period at a

glance. The chart can also be filtered by order date so the analysis can focus on relevant data. Other custom options are available with the Marks card in the shape of color, size, labels, details, and tooltips.

Comparison is one of the chief applications of bar charts. Bar charts are good for making comparisons between a large number of groups or categories, be it sales revenue across different products or regions. Assuming a business needs to evaluate the overall revenue of sales for various product categories in a given year, the product categories would be denoted on the X-axis, either on a less aggregated level, such as furniture, electronics, and office supplies, for example, or on a very aggregated level, such as large, small, or medium. The different heights of the bars, representing the income brought in by that category, clearly show the area of performance comparison across different groups. Distribution Analysis, the next important application, serves to show how one variable is distributed among various categories. One specific instance could be that the bar chart can show the distribution of students' grades in a class where each bar represents a grade range, and its height shows the number of students who received that grade. This makes it simple to identify patterns such as clustering around certain scores or perhaps the presence of outliers.

Finally, if time periods are treated as categories, bar charts can be employed for Trend Analysis. Rather than the classic line graphs, bar charts can show trends over time by placing different time intervals-such as months, quarters, or years-on the X-axis and corresponding values plotted on the Y-axis. This would aid in registering all the sales, revenue, website traffic, or customer behavior concerning changes in the different time periods.

7.1.2 Line Chart:

A line chart, or line graph, is a graphical representation of data that plots information as a sequence of data points joined by straight lines. The charts are used to display the trends, patterns, and changes in a continuous interval and are best employed for the analysis of temporal data. The x-axis is often employed to plot time or a sequence, while the y-axis is employed to plot the measured values.

Data preparation:

Aligning the correct time-series data first before plotting a line graph. Date or time fields should be formatted correctly, such as "YYYY-MM-DD" or "Month-Year," to allow for proper trend analysis. Check for missed values or other discrepancies that can disrupt visualization. Clean data for unobstructed trend representation without misleading gaps found on the chart.

Drawing the Line Chart in Tableau:

Connect the data source to Tableau and open the new worksheet to begin. From there, drag the date or time field into the Columns shelf, which refers to the dimension defining the x-axis, time, and the numeric measure-or indicator at y-ax at which value over time is represented, such as sales, profit or revenue.

Tableau may choose the default automatic visualization type. Go to the Marks card, click the dropdown menu, and choose "Line" to change to a line chart. In situations where the date field has several levels (e.g., year, quarter, month, day), drill down or aggregate the data at other levels using Tableau's date hierarchy.

Improving the Line Chart

For multi-measure comparisons such as adding multiple numeric fields either to the Rows shelf or Marks card will render each appearing in different lines within the same graph. One may improve colors to delineate lines by dragging the measure or category to Color shelf in the Marks card. Markers may be

added at select data points to highlight significant values such as a drop or peak within a month. An additional avenue for forecasting is adding trendlines or forecasting features to the data within Tableau, which anticipates the new or emerging trends by analyzing historical information. Such primary enhancements for decision-making significantly improve thorough insights.

Add annotations to highlight noteworthy points:

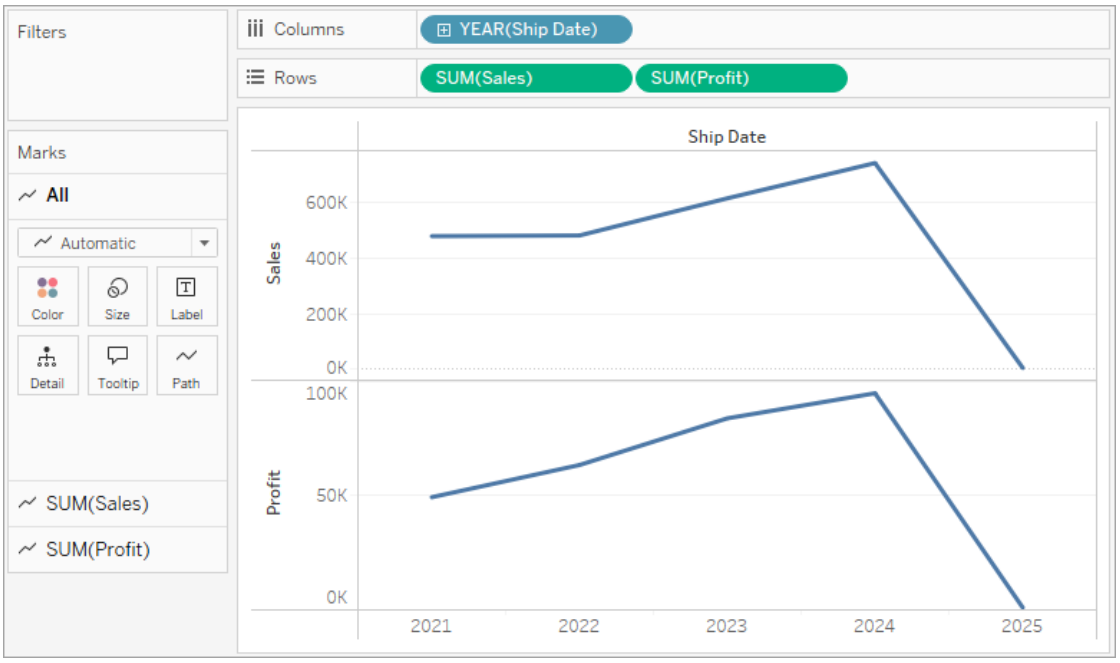


Fig. 7.2: Line Chart in Tableau

The fig 7.2 shows a dual-axis line chart charting Sales and Profit against Years on the basis of Ship Date. This is how it is divided:

The X Axis (Columns) indicates the Year, which is derived from the Ship Date field. For 5 years, 2021 - 2025, this chart lends itself to a specific analysis of trends over time. When time-based data are plotted on the X-axis, it is easy to track business changes in the metrics over the years. The Y-Axis (Rows - Left) shows Sum of Sales from 0K to 600K. It helps to track patterns of sales annually, such as increasing, decreasing, or keeping steady. The Y-Axis (Rows - Right) associated with that is Sum of Profits, whose values range from 0K to 100K. The scale used for profit is different from that of sales, so a different Y-axis is used to present images without hindrance from the one associated with sales. The chart has two lines: one for the sales measure and another for the profit measure. They will depict how those metrics trend over the period of 5 years so that their movement can be compared easily. Filters restrict the dataset based on Ship Date, excluding irrelevant time-based data in the visualization. The process will hone the analysis to a defined period. In Marks Card, it identifies the type of chart as a Line Chart and offers modification options. Any analyst can modify the Color, Size, Label, Detail, Tooltip, and Path to improve the readability and usability of the visualization. It is dual-axis in nature, meaning that within the scope of the same chart, two different measures (Sales and Profit) will be shown on different scales. This can try to keep both visible and comparable for any reader but avoid overshadowing one by another.

This is generally the case when doing a Trend Analysis, which shows temporally changing variables. For example, you can see changes in temperature, fluctuations in stock market trends, online traffic, or businesses over time. It makes clear sense within that visualization style, providing additional understanding of how things are evolving. The chart is also into Time Series Data very well. Here, one

reads time intervals on the X-axis, while corresponding values appear on the Y-axis; that is the most common way to assess trends, try to review predicted future movements, and spend time identifying seasonality in data.

7.1.3 Pie Chart:

Imagine a pie chart as cutting up a real pie at dinner with your family - everyone receives a slice, and you can immediately tell who received the larger slices! That's precisely what pie charts do with information. The underlying premise of pie charts is that they can represent part-to-whole relationships in a visually engaging format. By representing data as relative portions of a pie, pie charts take advantage of our natural feel for relative sizes and angles and enable readers to better understand the way particular portions relate to a whole set of data. Such graphic representation works particularly well if the goal is to highlight relative sizes of multiple categories of one variable.

In data visualization, pie charts play a unique role of emphasizing the part-whole relationship rather than attempting to make precise numerical comparisons.

Creating Pie Charts in Tableau (The Practical Way):

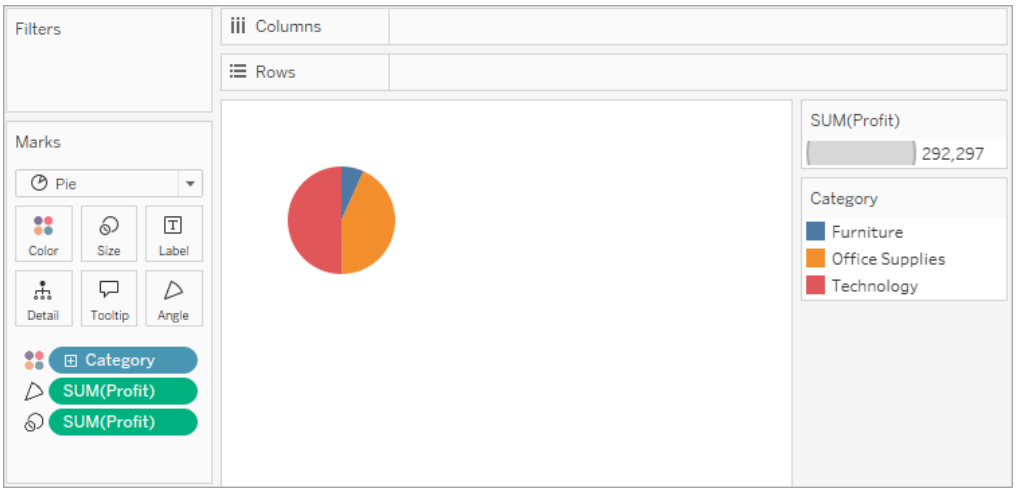


Fig. 7.3: Pie Chart in Tableau

Fig 7.3 is an representation of a pie chart of Sum of Profit by Categories:

Step-by-step explanation follows:

1. Data Preparation:

Before plotting a pie chart, verify that the data source is appropriately connected to Tableau. The visualization needs two fields: one category dimension (e.g., "Category") and one measure for their numeric values (e.g., "Profit"). These two will form the basis for slicing the pie and determining the sizes of these slices.

2. Plotting the Pie Chart:

With the data ready, select the "Category" field and drag it down onto the Color shelf on the Marks card. This divides the pie chart graphically with sections for different categories. Now, measure Sum of Profit will be dragged into the Angle shelf of the Marks card, determining the corresponding size of each slice based on profit values.

Since Tableau might present data in an alternate form, it is sometimes really essential to change the Mark type altogether. On the Marks card, click the dropdown showing "Automatic" or possibly some other visualisation type and select "Pie". This way, you can ensure Tableau is treating it as a pie.

3. Customizing the Pie Chart:

The pie chart can be improved for understandability and visual appeal by applying various customizations. While dragging the "Category" dimension to the Label shelf would ensure that category names are shown on slices, placing Sum of Profit on the Label shelf would ensure profit values are visible. Further formatting can be done by right-clicking on the label and selecting "Format" to refine text size, alignment, etc. The default colors assigned to each category by Tableau could be customized by clicking the Color shelf in Marks card and selecting "Edit Colors". This customization will allow better visualization to distinguish between categories. Tooltips, which display detailed information by hovering over the slice, can also be modified by clicking on the Tooltip shelf and editing the text to provide extra insights. Further, the Size option in the Marks card allows altering the overall size of the pie chart itself, which ideally aids in visual unequal weighting.

4. Interpreting the Pie Chart:

Examining the pie chart after creation and customization provides some useful information. For example, if using sales data by category, the pie chart paints a picture of how profits behave across the groups of Furniture, Office Supplies, and Technology. The size of each slice depicts the measure of profit given to each category and allows for straightforward comparison. The absolute profit value (for example, SUM(Profit) = 292,297) is normally shown, helping the reader comprehend company performance status at a glance.

7.1.4 Scatter Plot:

A scatter chart, or scatter plot or scatterplot, is a visualization tool that illustrates the relationship between two quantitative variables. Every point on the chart is a single data point whose position is defined by its coordinates on two axes (x and y). Scatter charts are great at showing patterns, correlations, clusters, and outliers in datasets and are extremely useful for statistical analysis and research in many disciplines. Scatter plots are the only type of chart that can visually show hundreds or thousands of separate data points on one chart at a time and give us such valuable insights into data relationships.

Creating Scatter Charts in Tableau:

For starters, drag the first measure to the Columns shelf, which defines the x-axis, then drag the second measure to the Rows shelf, which sets the y-axis. That sets up a two-dimensional space that plots data points based on these measures. Tableau may not immediately grant you a scatter plot but rather keep any mark type by default. Hence, click the dropdown box on the Marks card to select "Circle" or "Shape" so that the individual data points are clearly displayed. Adjust point size or opacity if required for an effective view, especially with many data points.

Enhanced scatter plot features for analysis:

Adding a trend line aids in the identification of patterns and correlations. This can be done from the Analytics pane by dragging a Trend Line onto the scatter plot, facilitating the visualization of the strength and direction of the relationship between the two variables. A scatter chart can also use color, size, or labels to express additional variables. By dragging a categorical field onto the Color shelf, data points are differentiated by groups for easy observation of clusters. If a third numerical variable is to be represented, sizing the points differently will reflect the differences in magnitude. Labeling specific data points, such as outliers or highlighted trends, can give more context and improve readability.

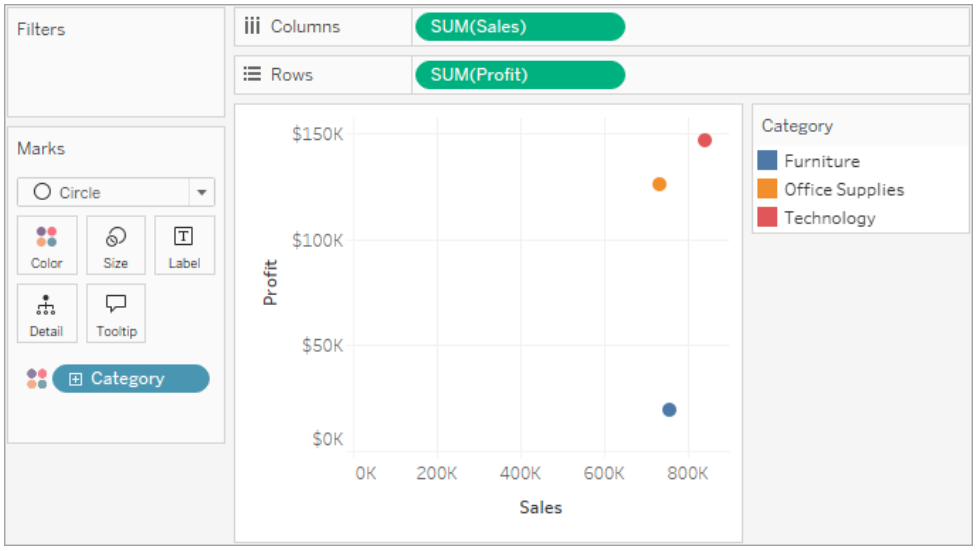


Fig. 7.4: Scatter Plot in Tableau

The graph shows (fig 7.4) a scatter plot that visualizes columns Sales and Profit, with a segmenting by Category. This is the breakdown:

Visualizations in tableau have been built using various keys, whose combination gives it the structure and appearance of a chart. These elements are useful in organizing the data and interpreting it better. The X-axis (Columns) in this visualization represents Sum of sales. The X-axis ranges from 0K to 800K, indicating total sales values distributed across different data points. This helps understand the contribution of a category to sales. On the Y-Axis (Rows), we see the Sum of Profit, which indicates a range of 0K-150K. We can compare sales and profit as it gives us insights into categories that have the highest profit level against their sales level. Marks are the data points we see as circles that denote an individual category. This illustration shows how Sales and Profit connect making it easy to perceive patterns and correlation in the data set. The Color of the marks is by the dimension of the category, where we distinguish different product categories. The blue mark represents Furniture, orange from Office Supplies, and red refers to Technology. Such identification straightaway helps one tell the difference between categories. Thus, the Filters are not applicable in this visualization; hence, the chart will show the whole data set without restriction. This is to ensure that nothing from the available data is denied from being displayed on the chart for analysis. The Marks Card provides the options to customize the view. For this example, it uses the Circle chart type, while the other options such as Color, Size, Label, Detail, and Tooltip are set either by default or customized to present better options. Such options provide a much more refined interactive view. Last, there is the Legend, which is more likely located to the right side of the chart and shows all the different categories with respective colors. This will be helpful reference material for viewers to be able to easily interpret the chart and understand how the data is distributed at a glance.

7.1.5 Bubble Chart:

Bubble chart is a three-dimensional plot that is an extension of the scatter plot idea by the inclusion of a third variable based on bubble sizes (circles). Scatter plots represent relationships between two variables as x and y coordinates, while bubble charts bring in a new dimension through variation in bubble sizes, hence useful to represent three quantitative variables at a time. Creating a bubble chart requires two numeric variables-first for the x-axis and second for the y-axis. These measures determine the location of

all the bubbles on the chart. The third numeric measure is used for bubble sizing, and it is imperative that this value is positive in order to portray it properly.

If desired, a categorical dimension can be placed on the Color shelf to distinguish groups within the data and assist in the visual identification of patterns or categories. Labels can also be added for direct identification of the bubbles, which becomes especially helpful in accentuating some important data points.

Technical Considerations in Tableau:

Bubble charts are numerical representations; hence, all measures used to plot the x-axis and y-axis should be continuous about the categorical options. This allows Tableau to plot them correctly in the coordinate system. The measure used for bubble size should also be numerical with values accepted only for positives, thereby disallowing any representation on the size of the bubble for negatives. Well-formatted and well-scaled data is also a very crucial factor. Inconsistent scaling can, at times, make the bubbles appear much larger or smaller than other bubbles, thus skewing the interpretation of the data. Tableau allows for customization of bubble appearance and client interface for better readability.

Best Practices for Implementation:

Before being able to create an effective bubble chart, it is very important to prepare and clean the data. If the measures used for bubble size differ markedly in scale, standardizing the values may help to retain proportionality. The removal of extreme outliers is important in avoiding distortion, as one single extremely large value can make all the other bubbles look too small. It is equally important to ensure that all measures are on a compatible scale for proper representation. In some cases, transformation of the data could be done, for example normalizing the values or log scaling, so that visualization will be clearer.

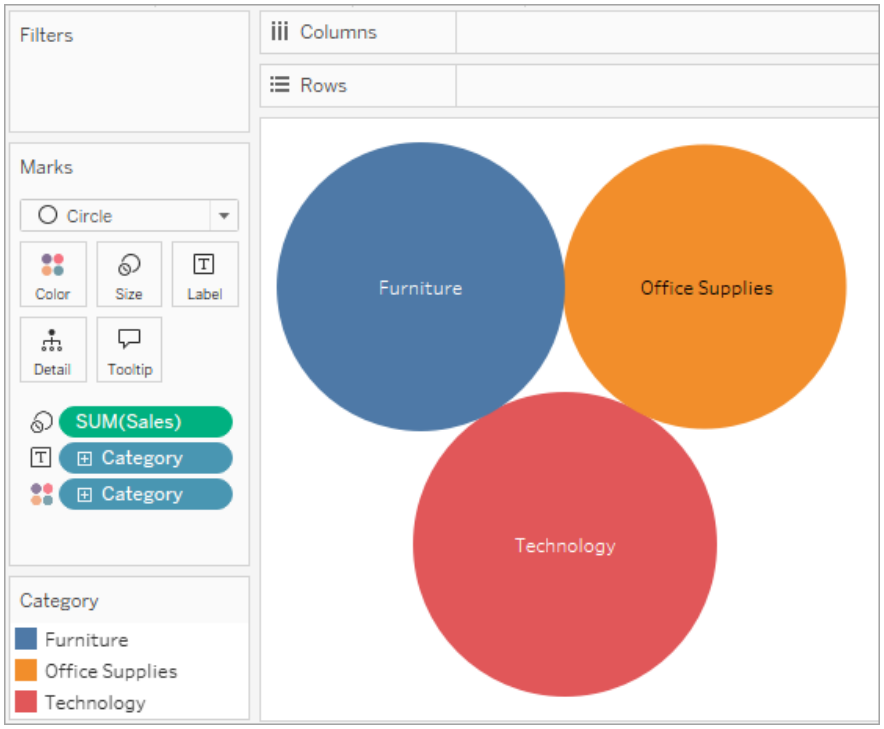


Fig. 7.5: Bubble Chart in Tableau

The above image (fig 7.5) is a bubble chart showing the Sum of Sales by various Categories. See these step-by-step instructions:

1. Data Preparation:

Initial consideration was given to connecting the data source in Tableau for purposes of mapping information. After establishing the connection, the following essential fields must be chosen for visualization: a categorical dimension (like "Category") that specifies each bubble, and a numerical measure (like "Sales") for determining the size of the bubble.

2. Creating the Bubble Chart:

Begin with dragging the "Category" dimension to the Color shelf of the Marks card. This means that a different color will be assigned to the bubbles based on different categories, thus enhancing clarity and differentiation. Next, drag the measure "Sum of Sales" onto the Size shelf on the Marks card, which will size each bubble proportionately based on total sales for that category. Dragging the Category dimension to the Label shelf will label each bubble with the name of its category. It is only necessary to check that the proper mark type is set to "Circle" since Tableau will automatically assign a default chart type. Choose "Circle" from the drop-down on the Marks card, and Tableau will take care of the rest.

3. Customizing:

Many customization options are available in Tableau to enhance the readability and presentation of the bubble chart. You can define the category colors according to your preferences by clicking on the Color shelf and selecting “Edit Colors.” Bubbles can be made smaller or larger according to your liking by editing the size on the Size shelf.

Tooltip information is also encouraged, as it facilitates better interactivity with the visualization and allows further information to display when hovering over a bubble. The indicator "Sum of Sales" automatically goes into the Tooltip shelf; however, further editing can be performed on the text and formatting features if you were to click on Tooltip and edit from there.

7.1.6 Gantt Chart:

A Gantt chart is a bar chart on the horizontal axis used specifically to display project timelines and progress across time. A Gantt chart is named after Henry Gantt and represents activities or tasks along the y-axis and their respective time periods along the x-axis. Each bar represents a task's duration, with the bar's position and length indicating when the task starts, how long it lasts, and when it ends."

Creating Gantt Charts in Tableau:

To create a Gantt chart in Tableau, project data should be connected. The date field, Start Date, would be dragged to the Columns shelf to form the timeline. The Task/Activity field would then be dragged onto the Rows shelf to see all project tasks. The type of mark will be changed to Gantt Bar in the Marks card. Starting Date is added to the Columns shelf, while Duration or End Date is added to the size shelf in order to denote the length of each task bar. Further customization can be achieved by changing bar color according to various categories or statuses.

In addition to this, for enhanced functionality, we might differentiate tasks from one another with color based on categories or progress. Such color coding will assist in distinguishing between various groups of tasks. Any dependencies between tasks can also be shown in order to show how phases of a project are linked. While tracking task progress, the percentage completion indicator may be added. Important deadlines can be highlighted using milestone markers while hierarchical classification can be adopted to group tasks that are related to one another for clarity.

Great usage of the Gantt chart provides a clearer picture in terms of how projects should be managed; in short, a real-time visual for tracking deadlines and executions while ensuring the employees know about the tasks at hand.

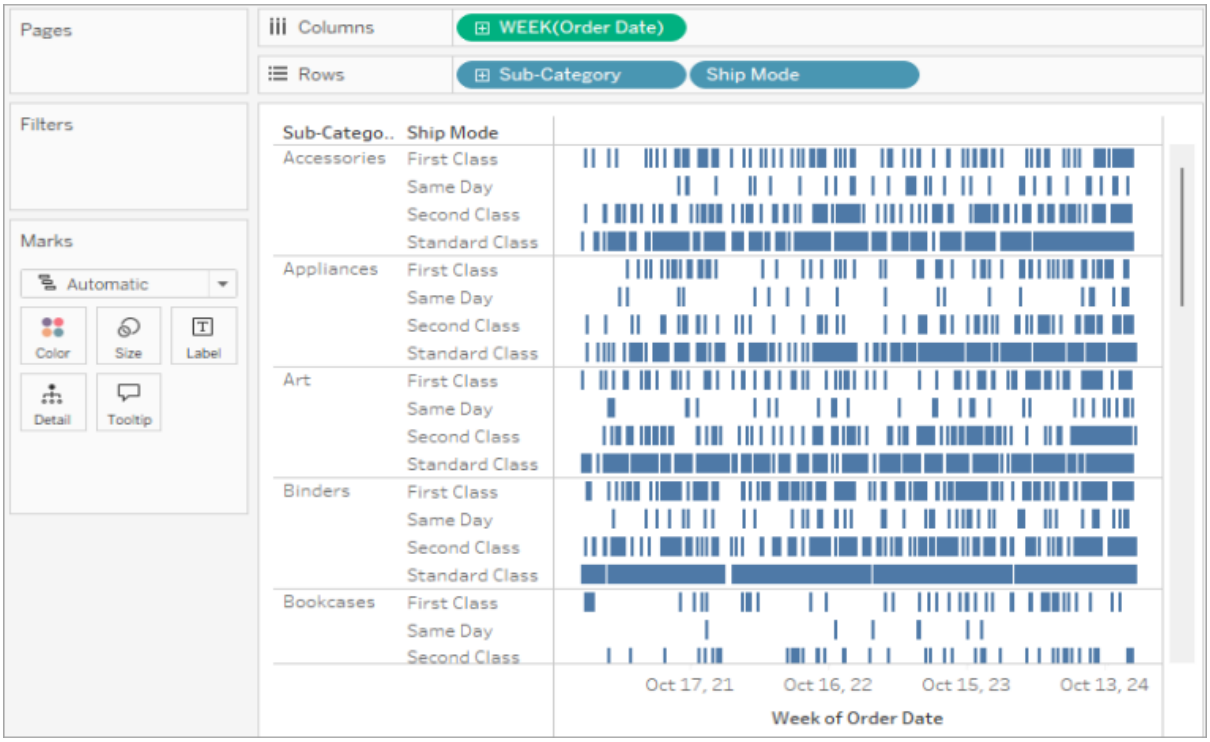


Fig. 7.6: Gantt Chart in Tableau

Fig 7.6 presents a Gantt chart visualizing the Sub-Category of products and their Ship Mode over Weeks based on the Order Date. Here's a breakdown:

Key Elements:

This Gantt chart gives a visualization with its different elements contributing to organizing and presenting data.

The X-Axis or Columns show the Week of the Order Date, thereby marking the timeline from 17 October 2021 to 13 October 2024. This enables the follow-up of the order in sequence and duration over the entire studied period. Organized therefore on a time-based axis, tracking enables easy identification of patterns and trends in order processing. The Rows show an intersection of Ship Mode and Sub-Category. Each row corresponds to a unique combination of a shipping mode and a product sub-category, and every moment of these two attributes appearing side by side will be represented distinctly. Thus this intersection helps analyze how different shipping modes are inclined toward various product categories. Gantt bars are displayed as horizontal bars whereby each bar signifies an order or job. The length of the order oriented horizontal bars signifies the weeks that an order was active. This pictorial representation would help the understanding of how long different orders lasted and how they were spread over time.

Colors are applied to the bars in ensuring that Ship Modes and Sub-Categories can easily be distinguished. Even through color differentiation, the easy identification of patterns and relationships with different shipping methods and product categories adds to the insight of the chart. There are no filters applied to this chart and so every aspect of the data is being brought forth without any limitations. This will ensure that the users see all available data points without any beforehand applications of constraints.

Marks Card gives relevant options for customizing the appearance of the Gantt chart. A checkbox permits the classification of the chart as automatic, whereby Tableau will select the most appropriate chart type with respect to the structure of the data. It thus permits further customization in the areas of Color, Size, Label, Detail, and Tooltip, making the visualization interactive and informative.

7.1.7 Histogram:

The image presents a Gantt chart visualizing the Sub-Category of products and their Ship Mode over Weeks based on the Order Date. Here's a breakdown:

Creating Tableau Histograms:

An easy way to build a histogram in Tableau is to pick "Histogram" from the Show Me menu or by dragging a measure (numeric field) to the Columns shelf. You also could right-click on the measure, select Create → Histogram, whereby Tableau will automatically create bins so as to group the data into ranges. The height of each bar in a histogram reflects the frequency count of values inside a bin. To edit the bins, simply click on the binned field in the Data Pane and select Edit. You can change the bin width to control for data grouping and the number of bins to make the representation better. Choosing a reasonable bin width ensures that the histogram presents salient features of the dataset pattern without overgeneralizing or overcomplicating the view.

The result of the fine-tuning of the histogram bin sizes and the number of bins leads to a more specific and insightful representation of the data distribution. It ensures that any trends, outliers, and general spread of the data come into the light for easy analysis and interpretation of its patterns.

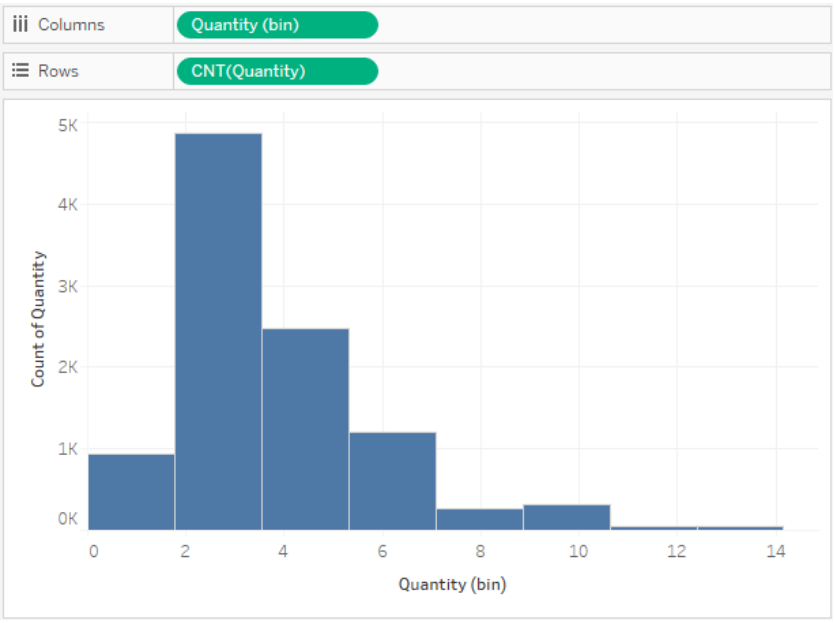


Fig. 7.7: Histogram Chart in Tableau

Fig 7.7 is an representation of a histogram of the distribution of the quantity value.

The breakdown is as follows:

Key Elements:

All these components go a long way in organizing, presenting, and directing data interpretation. The X-axis (Columns) denotes Quantity (bin) which has been determined into binned specified range intervals of quantity values. This helps comprehend the range intervals by combining similar quantity values together so that it can be easily viewed and observed of patterns that occur in the dataset. By segmenting the data

into bins, end users can compare how they are able to analyze the frequency distribution of various quantity values. The Y-axis (Rows) depicts Count per Quantity, CNT(Quantity), which is all the data points that represent the frequency within that bin range. It provides an indication of the amounts per quantities visible in the dataset. The bars in the histogram each represent respective bins, and the height of the bar is proportional to the number of data points (orders) that fall within that specific bin range. Thus, taller bars have a large number of orders in that specific quantity range while shorter bars signify few.

This chart has no applied filters, meaning it holds all the datasets with no restriction. This gives the full view of quantity distribution over all available data points. The additional customization offered by the Marks Card. The type of the chart is Automatic so that Tableau can determine the best styling according to the data set. It can further be customized by the user with Size, Label, Detail, Color, and Tooltip making it readable and capable of highlighting key insight. Such flexibility will allow end-users to modify how the histogram is presented, making it more informative and good-looking.

7.1.8 Waterfall Chart:

Waterfall charts are complex visuals aimed at showing the cumulative effect of sequential trades in increase with those in decrease affecting an initial value, leading to a final total. Such an instrument is particularly useful for analyzing changes in financial data, performance metrics, or any scenario where intermediate contributions affect an overall result. Key in a waterfall chart is tracking changes at every stage in the dataset. In essence, it portrays how much each of a number of individual positive and negative contributions influences the overall trend. Generally, increases are designated one color (for example, green) while decreases are called out in a contrasting color (here it might be red) thus allowing the viewer to assess the contributions of each flow easily.

The other very important feature comes with respect to the starting and finishing points that quickly walk you through the steps from the initial value to the final cumulative result. Each step on the chart represents a category or time period within that category through which each stage affects the total. For instance, a waterfall chart that contains sales information displays how different product subcategories affect the total revenue, thereby informing the company about factors affecting either profitability or loss.

Fig 7.8 presents a waterfall chart visualizing the cumulative Sum of Sales across different Sub-Categories. The break can be seen below:

In this waterfall chart visualization, the changes in sales across different sub-categories can easily be interpreted using several key features. The X-Axis (Columns) represents the Sub-Category, presenting all the different product categories along it. Each sub-category corresponds to a different point along the axis thus making comparisons easy as to the sales changes among different categories. The Y-axis (Rows) depicts the Sales Running Sum, which is the grand total running process through each sub-category. The running sum can hence track how the sales evolve under different categories giving an indication of the trends and contributions. The Gantt Bars in the chart finally present the individual changes in sales for every sub-category. Their sizes are determined by the magnitude of the change, while the direction they take indicates whether such a change is an increase or decrease for the sub-category in question. Color differentiates such increases and decreases in sales. Here, small bars in green represent an effect on sales in terms of improving sales and thus positively contributing to the running sum. This makes it so easy to see at a glance. There are not any individually defined filters in the chart. As a result, all sub-categories are included without restriction, thus allowing for a complete view of sales patterns across various products. The Marks Card is another feature that gives more possible options for the enhancement of the

visualization. This is defined by the type of the chart, which is Gantt Bar. Customization may be made at Color, Size, Label, Detail, and Tooltip extend to better presentation of the data. Such changes can help with readability and insights.

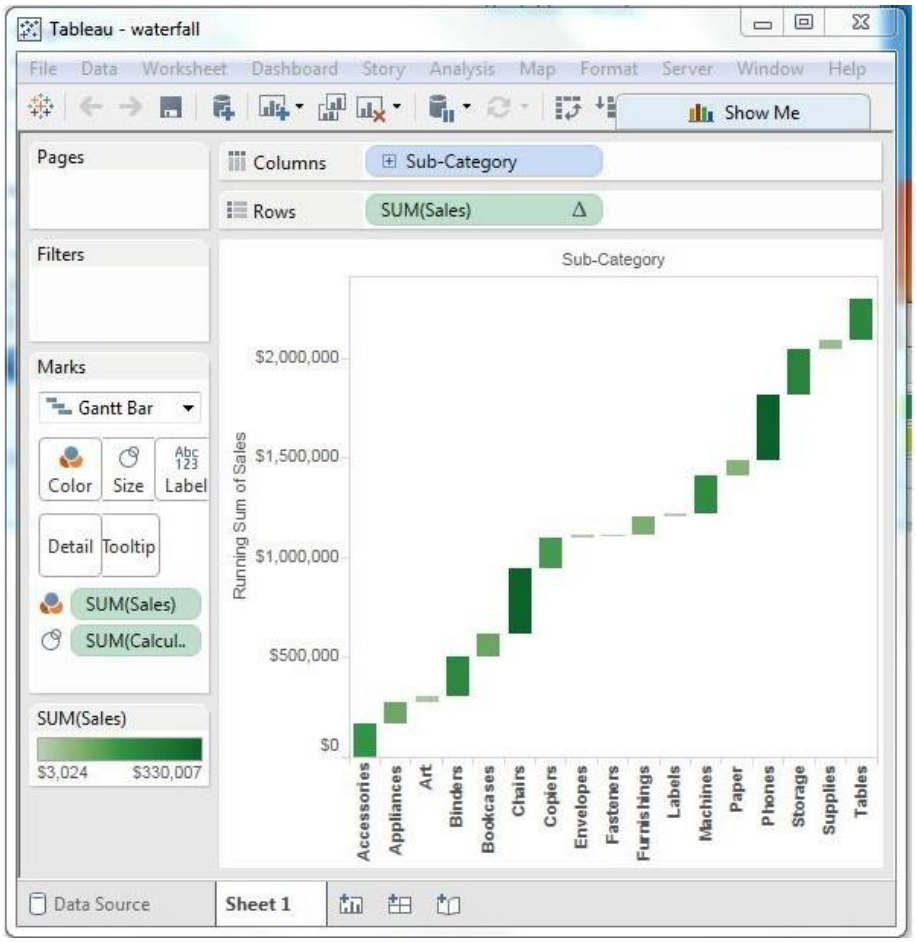


Fig. 7.8: Waterfall Chart in Tableau

The most important thing in the waterfall chart is Delta, a field that has been calculated. The triangle symbol (Δ) next to "SUM(Sales)" on the Rows shelf signifies that a computed field is being used to measure the variation in sales by category. This calculated field breaks individual contributions down to produce the overall running sum to be visualized effectively. This means that the Delta makes it possible for the chart to reflect increased and decreased sums accurately so that it assists the analysis of each sub-category's influence in total sales.

7.2 Dashboard:

Tableau dashboard is a collection of several visualizations such as charts, graphs, and tables in one view. The dashboard allows individuals to interact with data, see trends, and make intelligent decisions. Dashboards provide a single-point view of data from different worksheets, where users can compare and analyze important metrics in one glance.

A Tableau dashboard acts as a dynamic interface whereby information is presented through various charts, graphs, and tables in one view, thus providing a complete overview of the data. The dashboard allows plausibility for the user to analyze and interpret any given information basically through features such as interactivity, layout customization, and storytelling.

Key Features of Tableau Dashboards:

Tableau dashboards give the functionalities involved in interactive filtering, one of the best features. Users can apply filters or parameters to change what data is shown dynamically, allowing analysis to be further refined according to specific criteria by drilling down to particular aspects of an entire dataset, for instance, viewing sales figures for a specific region or filtering data by time periods. Another very important functionality is known as dashboard actions, allowing end-users to interact with the charts and tables. A single action, such as clicking on a data point within one visualization-type (like bars in a bar chart), can cause other visualizations to update. Tableau has URL actions to enable launching of external links, which is a good way of giving further context or providing resources. Custom-designed dashboards for different devices are offered by Tableau for better usability so that the dashboards can be customized for desktops, tablets, and mobile devices. This further ensures that the dashboards respond and adjust to various screen sizes while providing the user with good accessibility and readability. Tableau also enables storytelling with dashboards for advanced reporting and presentations, whereby multiple dashboards can be pulled together into a coherent narrative. This feature is particularly useful for giving step-by-step analyses that communicate insights and trends with ease.

7.2.1 Building a Tableau Dashboard:

In the process of building a Tableau dashboard, data is imported from various sources such as Excel, SQL databases, or CSV files. Once connected, the requisite datasets are loaded into Tableau's workspace and becomes ready for visualization.

Individual worksheets are created with each carrying a particular type of visualization, like bar graphs, line graphs, or pie charts. These worksheets are representations of specific aspects of the data, for instance, "Sales by Region" or "Profit Trends" which will later be assembled into a dashboard. Once visualizations are ready, the next step is to create a brand-new dashboard by going to the "Dashboard" option in Tableau. Following this, individual worksheets are dragged and dropped into the dashboard workspace and arranged into a layout of their own choosing. Users can resize and position the elements according to their analysis requirements.

To do this effectively, some interactive elements like filters, parameters, and dashboard actions could be added for better usability. These features increase interactivity, allowing users to dynamically explore the data from different angles. The formatting will also be applied to enhance clarity and maintain a consistent design.

Types of Tableau Dashboards:

1. Dashboards for operations:

Summary of the real-time insight dashboards include ongoing business processes; it seems that they do not only make it possible for users to see activities but also performance metrics and key indicators in an instant for daily activity performance tracking. Indeed, these dashboards are essential requirements for managers' and executives' activities of tracking all progress, bottlenecks identification and action realization and thereby promoting any rising change effort. A good example of this would be sales operations' daily orders, revenue, customer activity, and sales performance by region-all critical dimensions in support of surfacing sales trends, identifying the best-selling products, and readjusting sales strategies. In a facility, one could also see an operational dashboard for monitoring production rates, machine downtime, and efficiency in supply chains, providing an organization with the flexibility in responding to imminent problems.

Most operational dashboards have KPIs such as the amount of order fulfilment, inventory levels, and employee productivity indicators to enable businesses to plan ahead in their operations. Data is frequently refreshed, sometimes even in real-time, to ensure that users benefit from the latest information.

2. Dashboards for analysis:

Analytical dashboards provide long-term analysis of performance, historical trends, and predictive insights. These dashboards are for a deep-dive investigation into various views that help businesses detect patterns that may not be immediately seen in the routine operations of the organization. In the case of a financial analysis dashboard, years of revenue growth, trends in profit and loss, and budget variances could all be tracked as illustrated in fig 7.9. Here, financial analysts will assess how the company is doing, recognize spending patterns, and estimate future financial results.

In a similar fashion, a customer retention dashboard could track customer churn rates, levels of engagement, and satisfaction ratings for marketing teams as they refine their strategies to enhance customer loyalty. Besides, analytical dashboards would allow businesses to analyze trends like seasonal sales fluctuations, the effectiveness of marketing campaigns, or employee performance over time. Because analytical dashboards work on historical data, they usually employ advanced visualizations, statistical models, and forecasting tools to get even deeper insights. These dashboards enable decision-makers to build strategic plans based not on gut feeling but on systematic evidence.

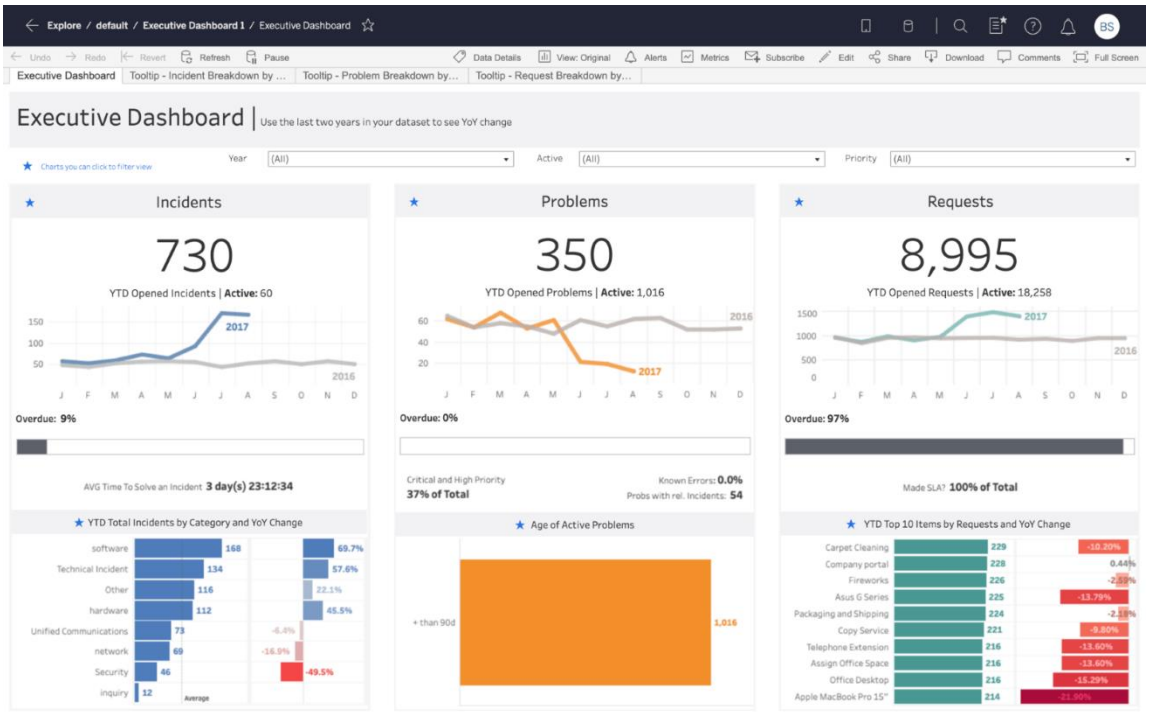


Fig. 7.9: Analytics Dashboard

3. Dashboards for strategic planning:

Strategic planning dashboards, according to their definition, are dashboards that facilitate managers and executives' ability to monitor their KPIs and measure the success of the business. This view will, of course, stay extremely high level without looking deeply into numbers and taking an overview of the organization for the purpose of its financial health, operational efficiencies, market trends, etc. With this motivation of consolidating important data, decision-makers identify strengths, weaknesses, and opportunities.

As an example, a performance dashboard for a company might contain metrics concerning income, expenditure, and newly acquired clients every quarter. This enables an executive to know the sources of revenue, manage costs well, and analyze the acquisition of customers. Most strategic planning dashboards could be monthly-or yearly-interval refresh rates. So, companies can input accurate data and draw data-driven decisions according to their long-term targets.

4. Strategic Dashboards:

Where executive dashboards exhibit KPIs at a high level, those for strategy by middle management are concerned with efficiencies at all levels of operation. Thus, these dashboards offer extensive views into individual departmental performances to ensure all that managers have the relevant views and points necessary for actions in day-to-day decision making.

An example is that a customer service dashboard could have different customer satisfaction scores, the status of a ticket, market shares (as shown in fig 7.10) and the time it takes to respond to inquiries. Stores these types of data will give managers the picture through which they can gauge the trends in customer inquiries, identify problems in service delivery, and, at the same time, better distribute resources. It is when either response times grow longer or customer satisfaction scores take a dip that the dashboard provides instant visibility, which allows teams to take corrective actions in real-time.

The dashboards also complement workflow optimization and resource planning for targets to be set by managers, performance evaluation of staff, and strategy deployment for improvement. Integrating interactive elements like filters, drill-down reports, and alerts increases usability and will typically allow a focus on a particular area of concern. With such applications in strategic dashboards, smoother operation and higher productivity would follow and subsequently build a basis for increased customer satisfaction.

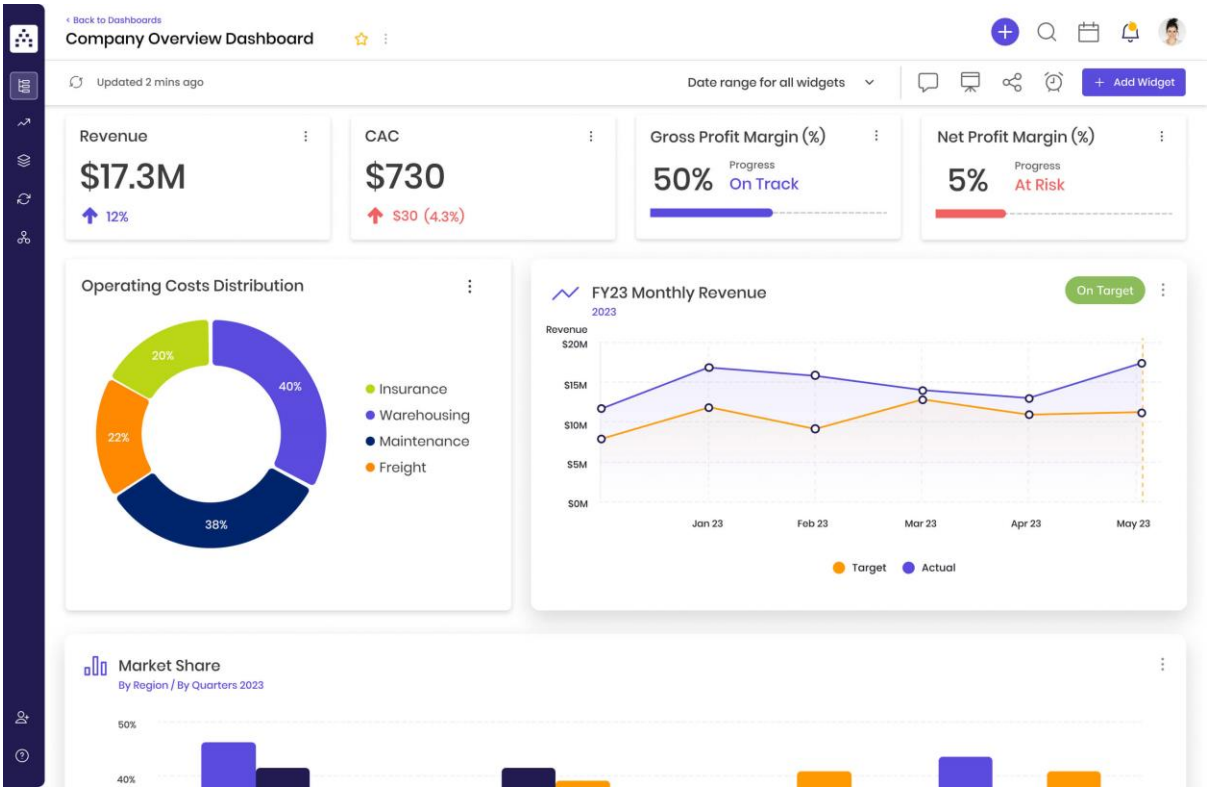


Fig. 7.10: Strategic dashboard

7.3 Formatting:

Formatting also works for aesthetics. It will beautify and clarify your visualizations. Formatting is all about setting adjustments of colors, fonts, borders, alignments, and other visual components in improving the manner dashboards are read and appear professionally. With proper formatting features, reports can look good and be straightforward to interpret. Proper formatting gives a whole structure and a lively way of presenting data insight as shown in fig 7.11, thus being easy for users to analyze trends, comparing values, and making decisions based on data.

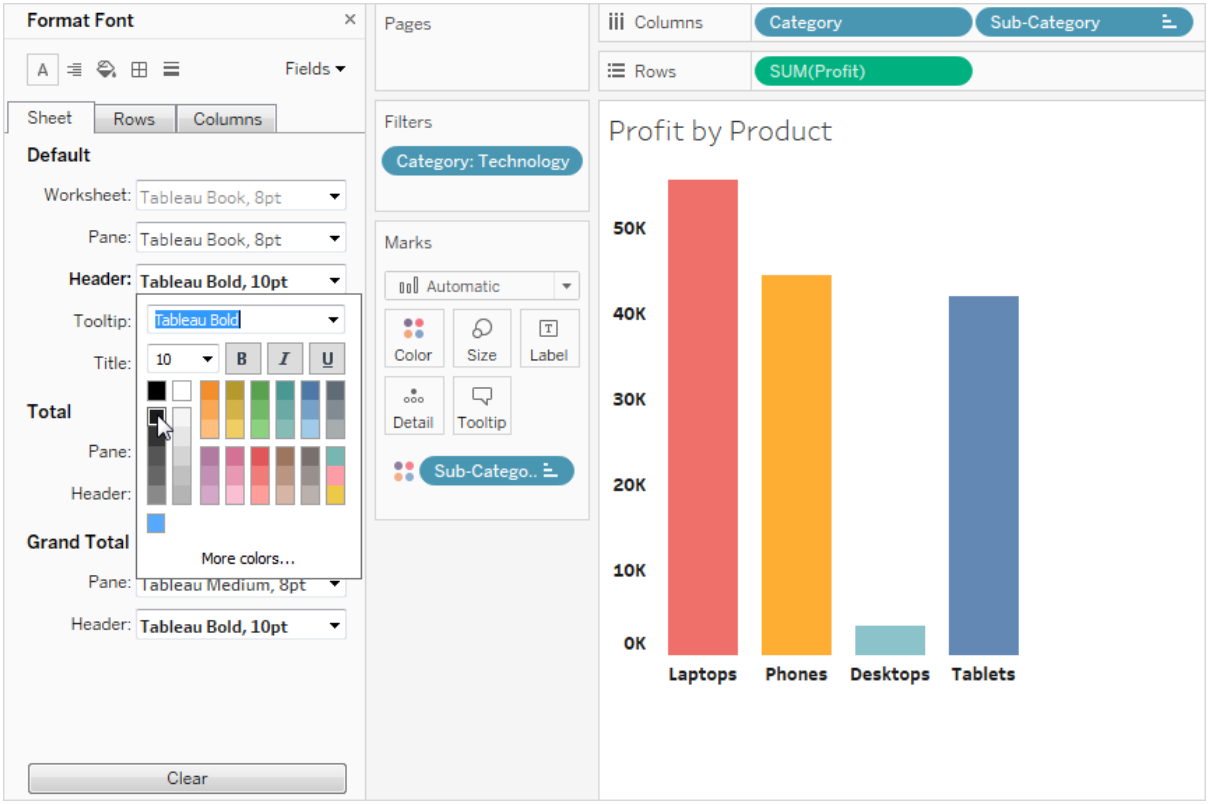


Fig. 7.11: Formatting in tableau

Worksheet Level Formatting in Tableau:

Different tools are available in Tableau to customize the visualization at worksheet level. A user can change a few configurations enhance their presentable and readable data. Part of the worksheet formatting involves font type adjustments, gridlines, shading, and other things to make the visualization neat and orderly. Font customization allows adjustments in text size, style, and alignment ensuring consistency and readability.

Another important aspect of formatting is number formatting via which the numerical values can be displayed in the relevant form, like in the format of currency, percentage, or scientific notation. In addition, users can rename headers and modify axes and scales for intuitive and user-friendly data understanding. Gridlines and divisions can also be modified or eliminated to improve the overall appearance of the chart. Keeping the dashboard background and design simple also brings up a quite professional polished look.

Steps to Format a Worksheet in Tableau:

To format a worksheet, one must select which worksheet they'd like to apply the formatting to. When that worksheet is opened in Tableau, users must click the Format menu to find various things they want to

modify. In this Menu, the incorporation of shading, borders, fonts, and any other display characteristics would enhance the whole visualization aspect from what their work would entail.

Font style and size modification are among the key worksheet formatting features. Borders and gridlines can be added or removed for a more structured and visually appealing workspace. Customizing background color is also an important aspect of making improvements regarding contrast as well as highlighting crucial data points.

Text Formatting in Tableau:

Text format is an aspect in designing through Tableau. One can personalize tooltips, axis labels, titles, and captions. Proper text makes data easy to read and interpretable when one needs to draw in key insights. To be able to format text, one needs to select a specific text object, whether it be tooltips, titles, or labels. One can simply right-click on the text to get to the Format option and thereafter change alignment with font size, color, and style to design with users' preferences.

For instance, if axis labels have been made bold and enlarged, they can be easy for viewers to gather meaning out of. Further, it is also necessary to use consistent font styles across the dashboards and worksheets so that everything looks uniform and professional.

Number Formatting in Tableau:

Good number formatting is important as far as manipulating numerical data is concerned, so one presents it very understandably. Number formatting in Tableau is in various forms such as decimal, currency, percentages, or even scientific notation, depending on the data

Chapter 8

ADVANCED FEATURES IN TABLEAU

Jeevanjot Singh Chagger[1], Manpreet Singh[2] and Paramjit Kaur [3]

[1,2]Sant Baba Bhag Singh University, Jalandhar

[3]Rayat Bahra Institute of Engineering and Nanotechnology, Hoshiarpur

8.1 Data Blending and Combining Datasets in Data Visualization:

Data visualization may involve the merging of data from various sources to present an accurate and meaningful representation. Merging datasets and data blending can be done to make this happen. Understanding the complexity of both processes is needed to make it an accurate, uniform, and meaningful analysis. The step-by-step definition of each term is shown below:

8.1.1 Data Blending:

Data blending in Tableau is a method for combining information from several unrelated sources. Data blending links data using a common dimension in Tableau, as opposed to data joining, which necessitates that tables have a shared field at the database level. Data blending is very helpful when dealing with data from various sources, for example, merging data from MySQL and Excel. It allows users to merge aggregated data from various sources without the hassle of database-level joins. Data blending prevents duplicate records or inconsistencies compared to the common joins that could lead to this. It maintains data integrity by keeping the sources distinct but providing easy integration. Further, it is helpful when the correspondence between datasets is poorly organized for joining, providing a flexible mechanism to combine and analyze heterogeneous data sources without changing the original database schema.

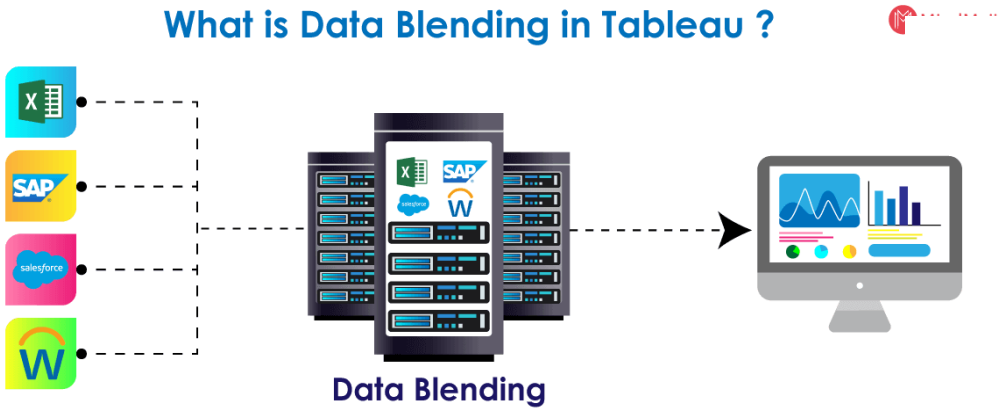


Fig. 8.1: Data Blending

In contrast to data joining, in which datasets are permanently merged into one structure, data blending keeps each dataset intact and combines them on a shared field (key) as depicted in the fig 8.1. The blending is done in real-time by the visualization tool during analysis.

Why is Data Blending Important?

i. Organizations will tend to gather information from various sources (databases, APIs, spreadsheets, etc.).
ii. Data sources can be of different granularities (i.e., monthly or daily sales).
iii. Not every data set has to be joined physically; some are better connected dynamically.
iv. Enables real-time analysis without modifying original data sets.

Key Features of Data Blending:

i. **Non-destructive:** The original datasets remain unchanged.
ii. **Dynamic Integration:** Data sources are linked in real-time.
iii. **Maintains Granularity:** Blending works even when datasets have different levels of detail.
iv. **Flexible:** Works with structured (databases, Excel) and unstructured (APIs, logs) data.

How Data Blending Works?

Data blending refers to the real-time combination of data from multiple sources within visualization. Data joining, where data sets are permanently combined, differs from data blending, as data blending does not combine the sources but combines them at the visualization level on a shared key.

1. Understanding the Process of Data Blending:

Step 1: Determine Primary and Secondary Sources:

Primary Data Source: The main data source used in the visualization.

Secondary Data Source: The second set of data about the primary source.

The main source determines the structure of the visualization, and the secondary source has extra information.

For example:

i. **Primary Source:** Sales transactions from a firm's database.
ii. **Secondary Source:** Google Analytics website traffic data.
iii. **Common Key**: Date.

Step 2: Identify the Common Key (Blending Key):

There must be a common key to combine data appropriately. The common key should be present in both data sets. Common keys can be:

i. **Date** → Used to compare daily visits to the website with daily sales.
ii. **Customer ID** → Utilized to link customer buying with customer demographics.
iii. **Product Code** → Used to reconcile sales amounts with quantities available.

The common key ensures that data points are correctly matched between the two datasets.

Step 3: Import the Datasets into a Visualization Platform:

Blending data is performed in data visualization tools such as Tableau, Power BI, Looker, and Google Data Studio. The tools allow:

i. Import multiple data sources into a project.
ii. Allocate a secondary and primary data source.
iii. Create the key for merging to link the datasets.

Example in Tableau:

i. Load Sales data as the initial source.
ii. Load Website Traffic Data as the Secondary Source.
iii. Make "Date" the key for blending.
iv. Tableau automatically links matching records.

Step 4: Matching and Aggregating Data:

As data blending is dynamic, certain rules hold:

i. The primary source powers the visualization, i.e., only information available in the primary source will be shown.
ii. The secondary source is collected based on the common key.

iii. Aggregations like SUM, AVERAGE, and COUNT are used in the event of multiple matches in the secondary source.

Example of Aggregation During Blending:

Date	Sales Revenue (Primary Source)	Website Visits (Secondary Source)
01-Mar	$5,000	1,000 visitors
02-Mar	$3,200	800 visitors
03-Mar	$4,500	950 visitors

Here, website traffic data is **aggregated by date** before blending with sales revenue.

Step 5: Visualization of the Blended Data:

Once the datasets are combined, they are utilized to build charts, dashboards, and reports.

For instance:

i. **Line Chart:** Sales revenue vs website visits over time.

ii. **Scatter Plot:** A review of how web traffic relates to sales.

iii. **Bar Chart:** Illustrating average sales revenue per visitor interaction.

Visualization software does blend on the fly, so the data is separate but shown together.

Example Use Cases for Data Blending:

The relationship between the datasets is not ideally structured for joining. In Tableau, data blending employs a primary-secondary model where:

Primary Data Source: The main dataset that forms the basis of the visualization. Secondary Data Source: The additional dataset that is linked to the primary source via a shared field (e.g., Customer ID, Date, Product Name).

i. Lets consider a scenario, there is two different datasets which stored in separate sources:

Table 1: Sales Data (Primary Data Source- MySQL Database)

Order Id	Date	Product	Sales	Country
001	March 1	Earphones	$50	UK
002	March 2	Phone	$300	Canada
003	March 3	Laptop	$150	USA
004	March 4	Speakers	$200	UK

Table 2: Customer Data (Secondary data Source-Excel File)

Customer Id	Name	Country	Customer Segment
C101	Robin	UK	Consumer
C102	John	Canada	Small Business
C103	James	Itlay	Consumer
C104	Michael	USA	Enterprise

ii. **Blending Process in Tableau:**

In Tableau, data blending allows users to merge data from more than one source to construct a unified view for analysis. It all starts with the primary data source: the main dataset. For instance, the user connects to the sales data that lies in the MySQL database. This will be the base for various visualizations and analysis. After that, an auxiliary data source is connected. This secondary dataset may come from various sources, including Excel, Google sheets, or other databases. For this example, customer data

taken from the Excel file has been leveraged as the auxiliary data source. With both data sources now connected, Tableau looks for a common field between the two data sources to establish a potential relationship. Here, that common field is "Country," which exists in Sales Data (MySQL) and Customer Data (Excel). Tableau recognizes this relationship and shows the common field with an icon for linking in the pane of data. The icon confirms that Tableau has successfully recognized a linking field for blending data between the two sources.

Now that the relationship has been established, users may begin to develop their visualization. It is very straightforward. Users drop fields from their primary datasource (Sales Data) into the Tableau worksheet in order to generate charts, tabulations, or any other visual forms. After the main dataset is in place, users may proceed to add fields from the secondary source, and Tableau will carry out the blending of dependent data automatically based on the common country element. This tight integration ensures that the data from different sources meaningfully combine so that metrics can be analyzed across the datasets. Data blending is highly applicable in Tableau, especially when dealing with data from bachelor databases, working on different levels of granularity, or cross-source comparisons. Hence, this must-have feature enables the users to derive key insights in a more effortless manner and take data-driven decisions.

Output: Blended data (After data blending in Tableau)

Country	Total sales	Customer Segment
UK	$250	Consumer
Canada	$300	Small business
Italy	-	Consumer
USA	$150	Enterprise

Observations:

a. Itlay is not included in merged data as it does not occur in the sales data.
b. Aggregation precedes blending; thus the sales are added up by country.
c. Two sets of data from different sources (SQL + Excel) are combined harmoniously without database-level joining.

ii. **Marketing Analytics:** Comparing Facebook ad impressions with e-commerce sales.
iii. **Retail Analysis:** Merging website traffic with in-store purchase data.
iv. **Finance & Banking:** Blending stock market data with external financial news.
v. **Healthcare Analytics:** Integrating patient medical records with wearable device data.

iii. **Limitations of Data Blending:**

a. **Performance Issues:** Bringing together large datasets can be slow for dashboards.
b. **Requires Shared Key:** Blending is impossible without a shared key.
c. **Limited to Visualization Tools:** Blending is temporary and cannot be utilized outside the realm of visualization.
d. **No Full Outer Join Support:** Data merging is only possible in case there exists a common key in the parent table unlike database joins.

8.1.2 Data Joining in tableau:

Combining datasets is the process of bringing together numerous different sources of data into one dataset to study and graph. It is a process that can be used by companies dealing with data from varied sources, shapes, and kinds but which have to research them as a whole.

In Tableau, data joining is a crucial step that allows you to combine data from different tables using a common field (key column). It helps to combine different data sources, improve databases, and create valuable insights from a variety of data points. Joining is especially useful when dealing with relational databases, spreadsheets, or various data tables containing related information separately.

Let's take the following example:

Orders Table (Order ID, Customer ID, Order Date, Sales)

Customer Table (Customer ID, Customer Name, Region), respectively.

Joining these tables based on Customer ID, we are able to develop a full dataset with order details and customer details.

Why Combine Datasets?

Organizations have a higher probability of having discrete data from assorted sources like databases, spreadsheets, APIs, and cloud stores. Integrating these data collections assists in

i. **Holistic Analysis:** Combining customer purchases with site visits for greater insight.
ii. **Comparative Studies:** Comparing company sales and market trends.
iii. **Shattering Data Silos:** Integrating fragmented data for informed decision-making.
iv. **Enhanced Reporting:** Creating standardized reports with multiple sources of data.

Methods of Combining Datasets:

Merging data sets is necessary for creating comprehensive insights from several data sets. The technique you use depends on the data structure, the data sets' relationship, and the analysis type you wish to perform.

The four primary methods by which datasets are combined are as follows:

1. **Appending (Union): Stacking similar datasets together**

   Appending (also referred to as union) merges datasets by placing rows one above the other when both datasets share the same structure (column names and data types).

   Use Case:

   i. Combining sales information across different stores or time periods.
   ii. Consolidating customer lists from sources.

Example:

Dataset 1 (North Region Sales)

Date	Sales	Region
Jan-01	500	North
Jan-02	600	North

Dataset 2 (South Region Sales)

Date	Sales	Region
Jan-01	450	South
Jan-02	550	South

Final Combined Table (After Appending)

Date	Sales	Region
Jan-01	500	North
Jan-02	600	North
Jan-01	450	South
Jan-02	550	South

Key Considerations:

i. Column names and types must match in both datasets.
ii. If columns differ, tools like Power BI, Tableau, or SQL allow column alignment.

2. Joining (Merge): Combining datasets based on shared keys:

Joining combines data horizontally by matching rows on a common key (e.g., Customer ID, Date, Product Code).

Use Case:

i. Blending customer information with transaction history.
ii. Connecting employees to departments

Example:

Dataset 1: Customer Information

Customer ID	Name	Country
101	John	USA
102	Alice	UK

Dataset 2: Order Details

Customer ID	Order ID	Order Amount
101	A001	$500
102	A002	$400

Final Combined Table (After Joining on Customer ID)

Customer ID	Name	Country	Order ID	Order Amount
101	John	USA	A001	$500
102	Alice	UK	A002	$400

8.1.2.1 Types of Joins:

Joins in Tableau enable the merging of multiple tables based on a shared column. This is necessary when dealing with relational databases or structured data that has related data split across several tables. Tableau has four join types as shown in the fig 8.2 that define how two tables' records are merged.

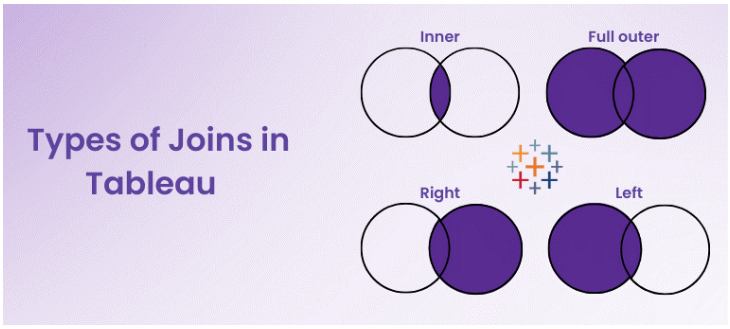


Fig. 8.2: Types of Joins in Tableau

1. Inner Join:

An Inner Join will only return records that share common values in both tables. It will never return any data that lacks a common value in the second table.

How It Works:

i. If Table A is customer orders and Table B is customer information, an Inner Join will give us only customers who have orders.

ii. Any customers present in Table B but without orders in Table A will be excluded.

Example:

Table A (Orders):

Order ID	Customer ID	Order Amount
101	C1	$500
102	C2	$300
103	C3	$700

Table B (Customers):

Customer ID	Customer Name	Country
C1	John Doe	USA
C2	Alice Smith	Canada
C4	Robert King	UK

Inner Join Output (Only Matching Records):

Order ID	Customer ID	Order Amount	Customer Name	Country
101	C1	$500	John Doe	USA
102	C2	$300	Alice Smith	Canada

Note: Customer C4 is missing from the result because there is no matching order in Table A. Similarly, Order 103 (C3) is missing because there is no matching customer in Table B.

2. Left Join:

A Left Join is used to fetch all records from the left table (Table A) and only the corresponding records from the right table (Table B). In cases where there is no match, NULL values will be returned for the columns coming from the right table.

Mechanism of Functioning:

If Table A has all the sales orders and Table B has customer details, then there will be a Left Join where all the orders are there, even if there is incomplete customer data.

Example:

Table A (Orders):

Order ID	Customer ID	Order Amount
101	C1	$500
102	C2	$300
103	C3	$700

Table B (Customers):

Customer ID	Customer Name	Country
C1	John Doe	USA
C2	Alice Smith	Canada
C4	Robert King	UK

Left Join Output (All Records from Table A, Matched from Table B):

Order ID	Customer ID	Order Amount	Customer Name	Country
101	C1	$500	John Doe	USA
102	C2	$300	Alice Smith	Canada
103	C3	$700	NULL	NULL

Note: Customer C3 is included in the result, even though it does not exist in Table B. Tableau fills the missing values with NULL.

3. Right Join:

A Right Join brings back all the records from the right table (Table B) and only those records from the left table (Table A) that correspond. Where there is no corresponding record, Tableau puts NULL values in the columns from the left table.

Mechanism of Action: In the case where Table A contains sales orders and Table B contains customers, a Right Join ensures the presence of all customers either individually or otherwise.

Example:

Table A (Orders):

Order ID	Customer ID	Order Amount
101	C1	$500
102	C2	$300
103	C3	$700

Table B (Customers):

Customer ID	Customer Name	Country
C1	John Doe	USA
C2	Alice Smith	Canada
C4	Robert King	UK

Right Join Output (All Records from Table B, Matched from Table A):

Order ID	Customer ID	Order Amount	Customer Name	Country
101	C1	$500	John Doe	USA
102	C2	$300	Alice Smith	Canada
NULL	C4	NULL	Robert King	UK

Note: Customer C4 appears in the result, even though they have no orders in Table A. The missing values are represented as NULL.

4. Full Outer Join:

Full Outer Join returns all of the records from the two tables, with or without matching. NULL values are provided if there is no match for a record in the other table.

How It Works:

If Table A contains orders and Table B contains customers, then a Full Outer Join contains all orders and all customers with non-matching entries on NULLs.

Example:

Table A (Orders):

Order ID	Customer ID	Order Amount
101	C1	$500
102	C2	$300
103	C3	$700

Table B (Customers):

Customer ID	Customer Name	Country
C1	John Doe	USA
C2	Alice Smith	Canada
C4	Robert King	UK

Full Outer Join Output (All Records from Both Tables):

Order ID	Customer ID	Order Amount	Customer Name	Country
101	C1	$500	John Doe	USA
102	C2	$300	Alice Smith	Canada
103	C3	$700	NULL	NULL
NULL	C4	NULL	Robert King	UK

Note: Both unmatched orders and customers appear in the result with NULL values.

To Conduct Data Joins in Tableau:

1. Connect to Your Data Source: Open Tableau and connect to the database or spreadsheet of your choice.
2. In the Workspace, move tables. You can drag a table to the Data Pane area. Next to it, set the second table.
3. Specify the conditions for joining: Based on shared fields, Tableau will automatically suggest a join condition. If required, manually modify the Join Key (such as Customer ID).
4. Select the Join Type: Pick from Inner, Left, Right, or Full Outer Join according to your needs.
5. Preview the Joined Data: Check to ensure the join is functioning as expected. Investigate any NULL values, missing entries, or duplicates if present.

3. Blending: Dynamically combining datasets from different sources:

Blending integrates data dynamically into a visualization tool instead of merging it physically.

Use Case:

i. Combining real-time marketing metrics (Google Analytics) and in-house sales reports
ii. Integration of diverse database sources without changing existing data

Example:

A business analyst would like to examine the impact of Google Ads expenditure on sales.

i. Ad Data (Google Analytics): Contains Clicks, Impressions, and Cost per Click.
ii. Sales Data (SQL Database): Order Amount and Customer Information.
iii. Shared Key: Date

Data Blending allows the two sources to remain separate but see them together.

Key Considerations:

i. Primary dataset determines the layout of visualization.
ii. The secondary dataset is pre-aggregated before blending.
iii. Only matching data from the primary source is used.

4. Aggregation & Transformation: Granularity change before joining:

Certain data sets need to be aggregated or rescaled before merging because they have differing levels of granularity (e.g., monthly vs. daily sales).

Use Case:

i. Converting the transaction data into monthly aggregations before the merge.
ii. Data cleaning and standardization for greater accuracy.

Example of Aggregation Before Merging:

Daily Sales Data (Before Aggregation)

Date	Sales
Jan-01	100
Jan-02	200
Jan-03	150

After Aggregating to Monthly Sales:

Month	Total Sales
Jan	450

Aggregation ensures meaningful data comparison before merging.

Examples for Combining Datasets:

i. **Offer a Single Key for Joining:** Joining data requires a common field.
ii. **Search for Duplicates:** Unnecessary duplicates can lead to miscomputations.
iii. **Manage Missing Data:** Determine if to impute, filter, or remove missing values.
iv. **Optimize Performance:** Dashboards are slowed down by joining large datasets; use pre-aggregations.
v. **Verify the Combined Data:** Always verify combined datasets.

Limitations:

i. **Schema Mismatch:** Column headers and data types may vary between data sets.
ii. **Performance Issues:** Merging huge datasets slows down analysis tools.
iii. **Granularity Conflicts:** Combining daily and monthly data can produce misleading conclusions.
iv. **Data Quality Problems:** Flawed keys (e.g., date formats) can lead to improper joins.

8.2 Forecasting:

With forecasting in Tableau, it is possible to forecast future values based on historic data. Tableau uses complex statistical methods in making the prediction and showing them in your chart as shown in fig 8.3. Forecasting is particularly valuable for time-based data, such as being able to see probable future trends and make data-based decisions.

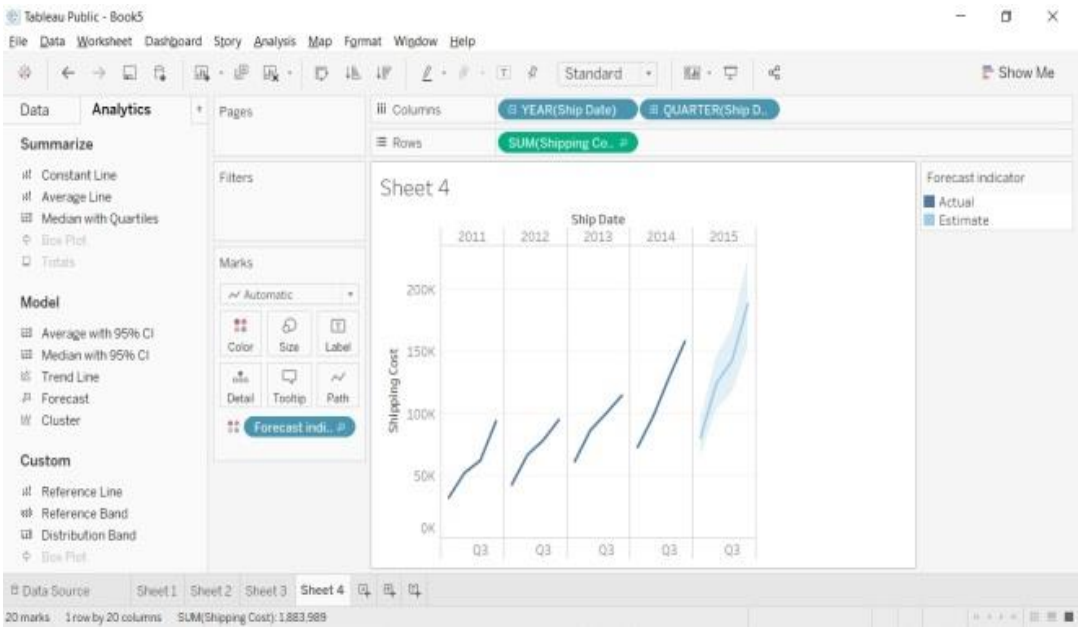


Fig. 8.3: Forecasting

How to Create a Forecast?

1. Choose the Worksheet: You can begin by accessing the worksheet containing the time series data you wish to analyze.

2. Introducing a Forecast: Navigate to the Analytics pane and pull the "Forecast" option into your view. Tableau will create an automatically generated forecast using the existing data.

3. Adjust the Forecast: Right-click in the forecast area and choose "Edit" to alter the forecast model. You can adjust the forecast duration, select between types of models (e.g., additive or multiplicative), and whether to include or exclude periods.

4. Display the Forecast: The forecast will appear as a long line or band on your chart, indicating the predicted values for subsequent periods. Tableau also shows confidence intervals to reflect the uncertainty in the predictions.

Applications

• Sales Forecasting: Projecting future sales amounts by examining past sales records to guide inventory management and marketing approaches. • Revenue Estimation: Anticipating future revenue by examining past financial outcomes.

• Demand Prediction: Estimating future demand for goods or services to aid in production and supply chain planning.

Example: Estimating monthly sales for the upcoming year based on sales data from the last three years.

Trend Lines

Trend Lines in Tableau serve to illustrate the overall trajectory of data points within a visualization. They assist in recognizing patterns and trends over time or among different categories. Based on statistical models, trend lines can be incorporated into various chart types, including scatter plots and line charts as illustrated in the given fig 8.4.

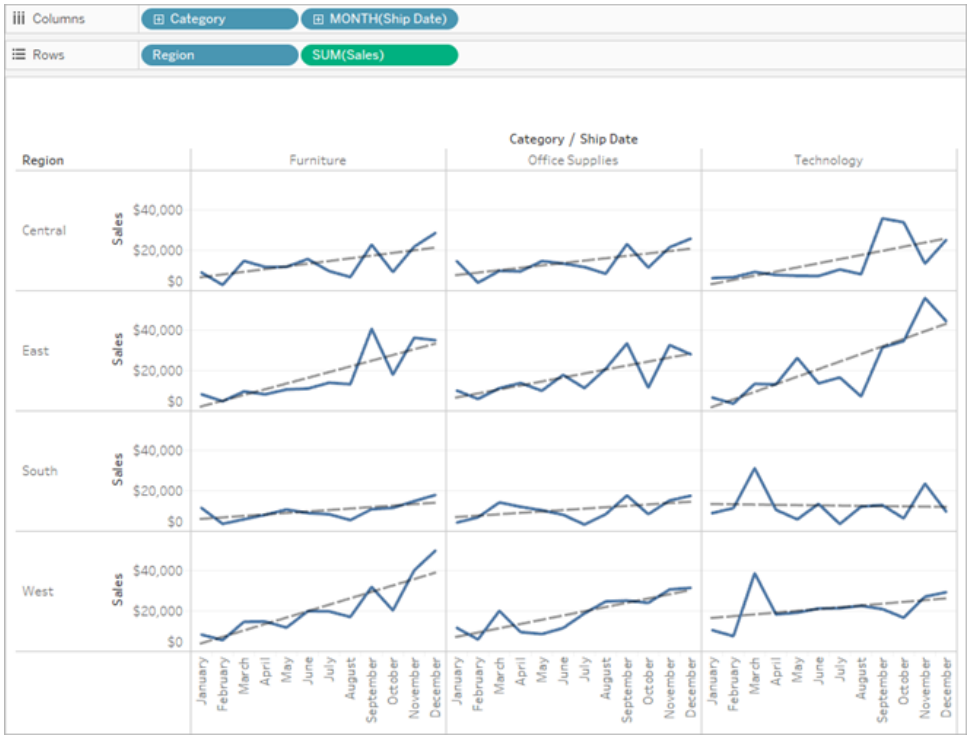


Fig. 8.4: Trend Lines

How to Create a Trend Line in Tableau?

1. Choose the Worksheet: Launch the worksheet containing the information you wish to examine.

2. Add Trend Line: Select the "Trend Line" option in the Analytics pane and drag it onto the screen. Tableau will automatically generate a trend line from the data points.

3. Change Trend Line: Right-click on the trend line and choose "Edit" to alter the model. You can choose the type of trend line (e.g., linear, logarithmic, polynomial), change the line thickness and color, and show statistical information like the R-squared and p-value.

Applications and Example

• **Trend Analysis:** Recognizing long-term trends in metrics like sales, website traffic, and stock prices.

• **Pattern Identification:** Spotting seasonal trends, cycles, and anomalies within the data. • Performance Assessment: Evaluating the performance of various factors over time. Example: Introducing a trend line to a scatter plot to investigate the correlation between advertising expenditures and sales income.

Visualization methods for different domains:

Employ visualization methods across various fields: Tableau, a leading data visualization tool, enables dynamic and effective visualizations for numerous sectors including business, healthcare, finance, education, and others. Given the distinctiveness of each industry's data, target audience, and goals, a tailored visualization strategy is required for each.

In this discussion, we will explore how Tableau is applied in different industries, featuring real-world examples, recommended practices, and a variety of visualization techniques.

1. Analytics for Healthcare:

Visualization's Significance in Healthcare:

Massive volumes of patient data, medical histories, and disease information are produced by the healthcare sector. It helps in:

•Tracking patterns in patient health is made easier with data visualization.

• Forecasting disease epidemics.
• Assessing the performance of hospitals.
Techniques for Healthcare Visualization:
1. Patient Information Dashboard: Bar graphs and pie charts display the patient distribution by disease type, age, and gender.
2. Hospital Bed Occupancy Rate: Regions with high patient occupancy rates are depicted by a filled map or heatmap.
3. COVID-19 Trend Analysis: Line graphs show how infection rates have been rising and falling over time.
A Use Case Example:
• A bar chart that displays the number of patients visited each day in week in hospital to analyze patient admission trends as shown in fig 8.5.
• A map chart to track disease outbreaks throughout several regions.
• A bubble chart that shows the success rates of various diseases' treatments.

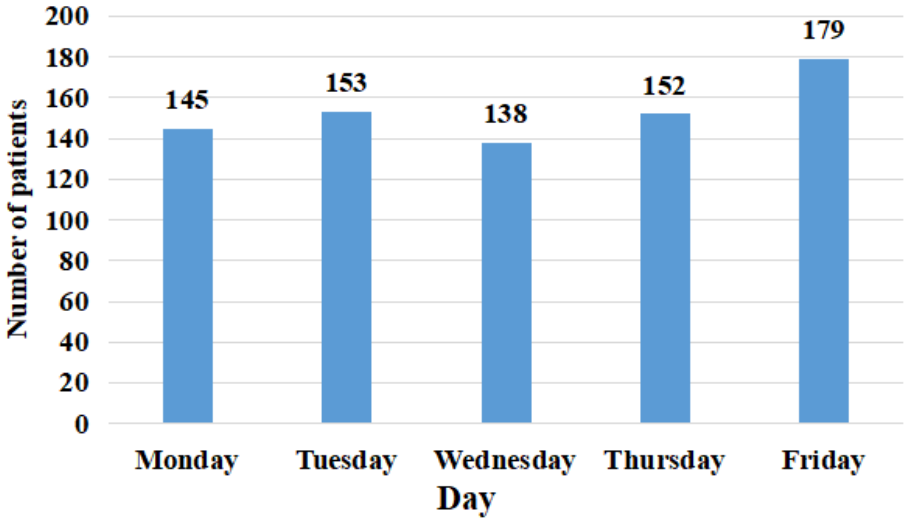


Fig. 8.5: Bar graph representing relation of number of patients and days of week

1. Education and E-Learning Analytics:
Educational institutions make use of data visualization to track levels of engagement, enrollment rates, and student achievement. It helps in:
•Monitoring the progress of students.
•Identifying areas in which students struggle.
•Measuring teacher effectiveness.
Frequent Educational Visualizations:
1. Student Performance Dashboard: Bar charts display grades for various subjects.
2. Trends in Enrollment Over Time: Admissions over the year are represented by line graphs.
3. Drop-Out Rate Analysis: Pie charts represent the reason for dropouts.
For example, Use Case: A university utilizes Tableau to analyze the performance of the students by:
• Making a bar graph representing passed and failed students in various subjects as shown in fig 8.6.
• Making a line graph to compare attendance trends.
• Utilizing a scatter plot to find associations between study hours and exam results.

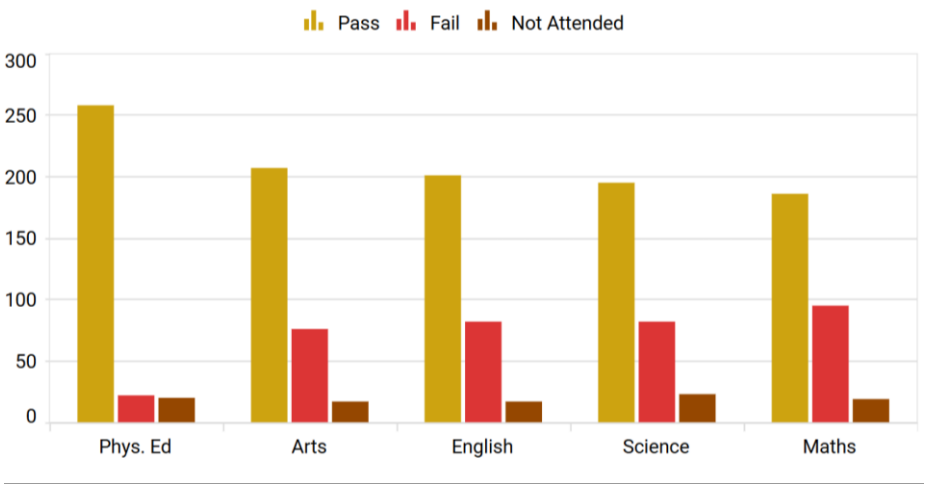


Fig. 8.6: Bar Graph of data of total students' performance

3. Analytics for Business and Sales

Typical Business Sales Performance Visualizations:

1. **The dashboard** incorporates a mix of line charts, bar charts, and key performance indicators (KPIs) to track growth trends, profit margins, and revenue from sales.

2. By showing variation in sales across regions and customer tastes, **heatmaps** can be employed to illustrate market segmentation.

3. **Tree maps** may be employed to visually display profitability analysis and illustrate the way profits are allocated among various products or geographic areas.

A retail organization uses Tableau to compare monthly sales for different regions, as an example of a use case shown in fig 8.7. They use :

• A line chart to see trends in sales seasons.

• A revenue comparison bar chart by state.

• A scatter plot that illustrates how marketing expenditures relate to one another.

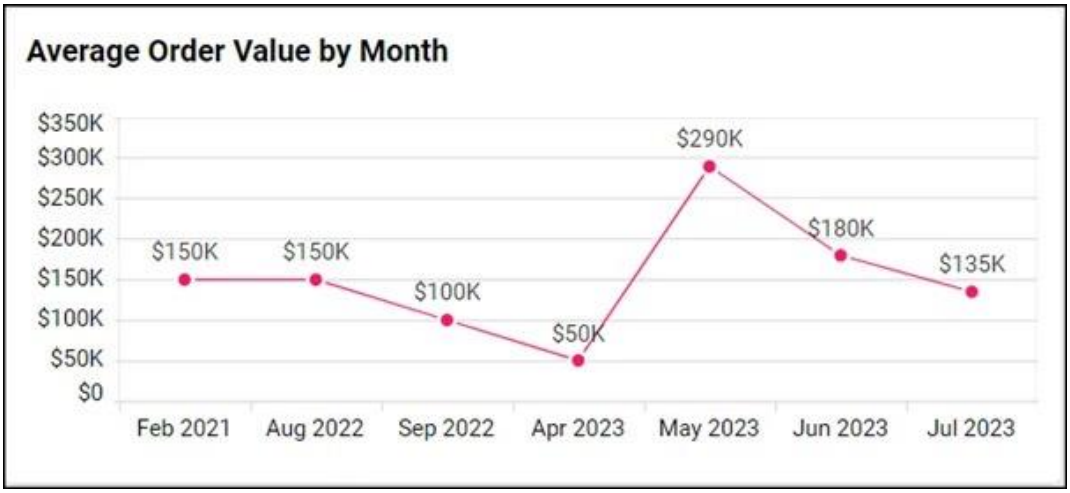


Fig. 8.7: Line chart representing sales in different years

4. Logistics and Supply Chain Analytics:

In supply chain management, visualization increases delivery efficiency, reduces expenses, and optimizes logistics. It is employed for:

• Tracking the time of shipment and delivery.

• Inventory level management for storage facilities.
• Forecasts of demand and supply variability.
Typical Supply Chain Visualizations:

1. **Sankey diagrams** are used to show how goods move from suppliers to consumers in a supply chain.
2. **Delivery Route Optimization**: Delivery routes and delays are shown on geographic maps.
3. Inventory Stock Levels: Current warehouse inventory is shown versus demand in bar charts.

A Use Case Example shown in fig 8.8:
Tableau is used by a delivery service to optimize deliveries:
• Delivery timings are tracked by location using a map representation.
• Warehouse inventory levels are displayed in a bar chart.

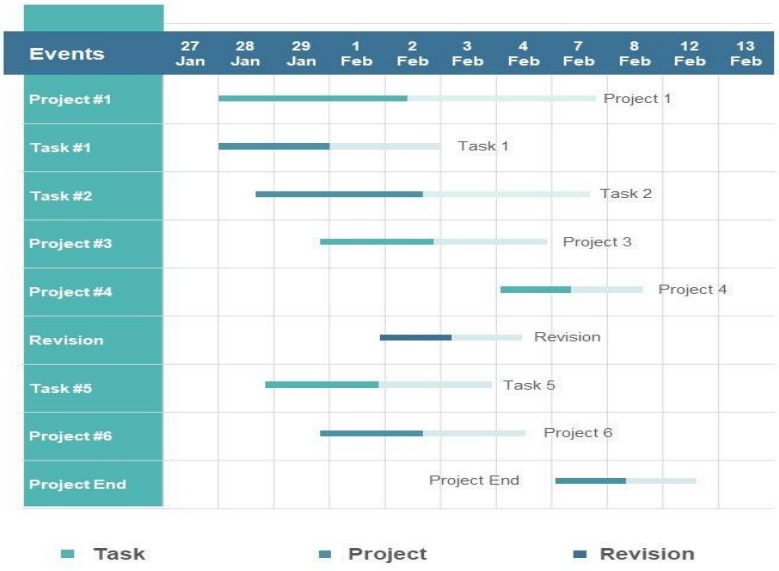


Fig. 8.8: Gantt chart of schedules transportation

Chapter 9

STORYTELLING WITH DATA VISUALIZATION

Navdeep Kaur[1], Navjot Kaur Basra[2] and Sumit Chopra[3]

[1,2,3]GNA University, Phagwara

Data visualization is more about storytelling and less about graphs and charts. Having the ability to transform dense data into short, compelling, and actionable insights is a value. The principles of good data storytelling are the topic of this chapter, followed by case studies on applying visualization techniques in different domains and good data storytelling as shown in the given fig.9.1.

Fig. 9.1: Storytelling using Data Visualization

9.1 Designing Effective Data Stories:

9.1.1 What is Data Storytelling?

Data storytelling is the process of combining data, visuals, and storytelling to present insights in an effective way. A good data story answers critical questions, leads the audience to conclusions, and employs well-crafted narratives and visuals.

Fig. 9.2: What is Data Storytelling?

9.1.2 Key Elements of a Data Story:

Data storytelling is a strong method that integrates data, graphics, and narrative to present insights effectively as depicted in the above fig 9.2. It serves to convert raw data into actionable stories that can be used to inform decision-making. The following are the essential components of data storytelling in data visualization, detailed as follows:

i. **Data:** At the core of data storytelling is the data itself. Without high-quality, reliable data, even the most beautiful story has no credibility.

Best Practices for Data Usage in Storytelling:

a. Ensure Data Accuracy: Cross-validate sources to prevent misinformation.
b. Use Relevant Data: Utilize only the data points that add to the story.
c. Provide Context: Raw figures are misleading; always provide their context.
d. Reveal Trends and Patterns: Data storytelling must emphasize relationships, trends, and key findings instead of mere numbers.

A narrative based on incorrect or incomplete data may result in misinterpretation and poor decision-making.

ii. **Visuals:** Data visualization is very important in telling stories because the human mind absorbs visuals 60,000 times faster than written words. Nicely designed graphics aid in presenting complex data more easily.

Selecting the Best Visualization:

a. Bar Charts: Best utilized when comparing distinct categories.
b. Line Charts: Best for presenting trends over some time.
c. Pie Charts: These are utilized to represent proportions but must be utilized judiciously.
d. Heatmaps: Good for detecting patterns in big datasets.
e. Scatter Plots: Assist in demonstrating relationships and correlation between two variables.

Guidelines for Effective Visualization:

a. Keep it Simple: Refrain from clutter and extraneous design components.
b. Use Color Judiciously: Apply contrasting colors to emphasize crucial aspects, and make it accessible (e.g., colorblind-friendly color palettes).
c. Be Consistent: Employ similar scales, labels, and formats for a consistent appearance.
d. Include Labels and Annotations: Highlight axes, data points, and trends with clear labels for enhanced understanding.

Good visual storytelling decreases cognitive load and makes the message readily understandable

iii. **Narrative:** An effective narrative gives meaning and context to data. It takes the reader through the findings and makes sure they get the message from the numbers.

Elements of a Great Narrative:

It gives the background on the data and why it is significant.

Conflict or Challenge: States the problem or question the data is trying to solve.

Insights and Findings: Includes the main conclusions, backed up by data visuals.

Ends with something actionable or suggestive.

A strong narrative aid in making data more relatable and ensuring the audience can relate to the message.

9.1.3 Principles for Effective Data Storytelling: Design Principles for effective data visualization are given in the figure 9.3.

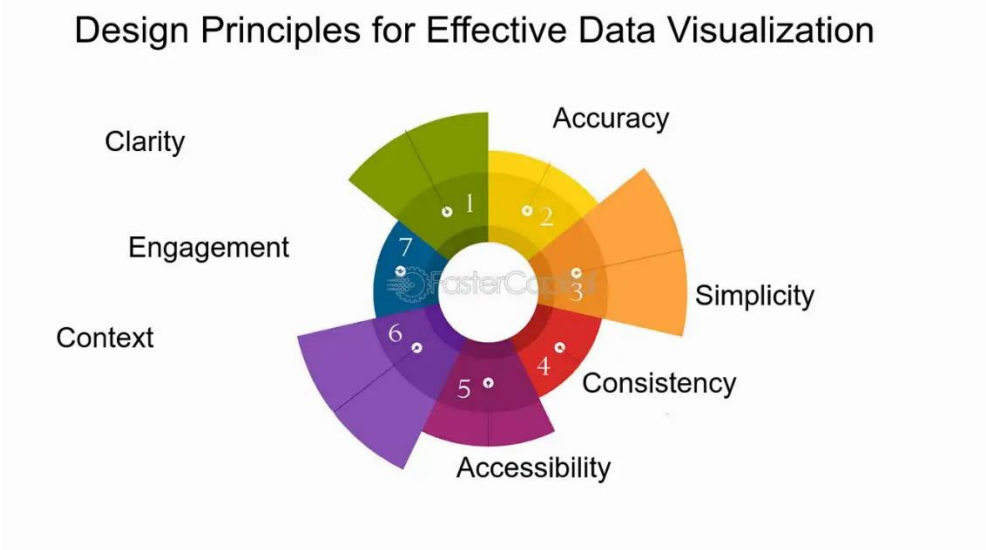


Fig. 9.3: Design Principles

i. Start with a Clear Purpose:

i. Establish the tone of the narrative: Narrative, descriptive, or expository?
ii. Identify the target audience (executives, analysts, general public).

ii. Know Your Audience:

i. What is the audience's level of knowledge?
ii. What conclusions will they make from this information?

iii. Organize the Story:

i. Begin: Indicate the problem or question.
ii. Middle: Display the data with analysis.
iii. End: Key takeaways and call to action.

iv. Have Good Visualization Skills:

i. Trends & comparisons → Line graphs, bar charts.
ii. Proportions → Pie charts, stacked bars.
iii. Relationships → Scatter plots, bubble charts.

v. It should be concise and easy:

i. Avoid unnecessary complexity in charts.
ii. Highlight the main points to avoid information overload.

vi. Highlight Key Takeaways:

i. Employ annotating, coloring, and labelling to direct attention.
ii. Remove distracting items that are not adding anything.

9.2 Applying Visualization Methods to Different Domains:

Data visualization is a key tool in the majority of fields, assisting professionals in understanding complex data sets, recognizing patterns, and making sound decisions. Certain visualization methods are employed

by industries based on the nature of data they have and the purpose. The following are key areas where visualization plays a significant role, with the right methods and examples.

1. Business & Finance:

Companies and financial institutions are dependent on data visualization to track performance, monitor financial trends, and make improved decision-making. Executives, investors, and analysts require transparent, real-time insights to evaluate business health and market dynamics.

Key Visualization methods:

i. **Line Charts:** It employed to illustrate trends in expenses, revenues, or stock prices over some time.
ii. **Bar Charts:** It compares financial performance between various periods or business segments.
iii. **Pie Charts:** This illustrates the percentage of revenue contributed by different products or services.
iv. **Heatmaps:** They determine the most profitable and underperforming areas or business segments.
v. **Dashboard Reports:** It combines several visualizations for an overall business overview.

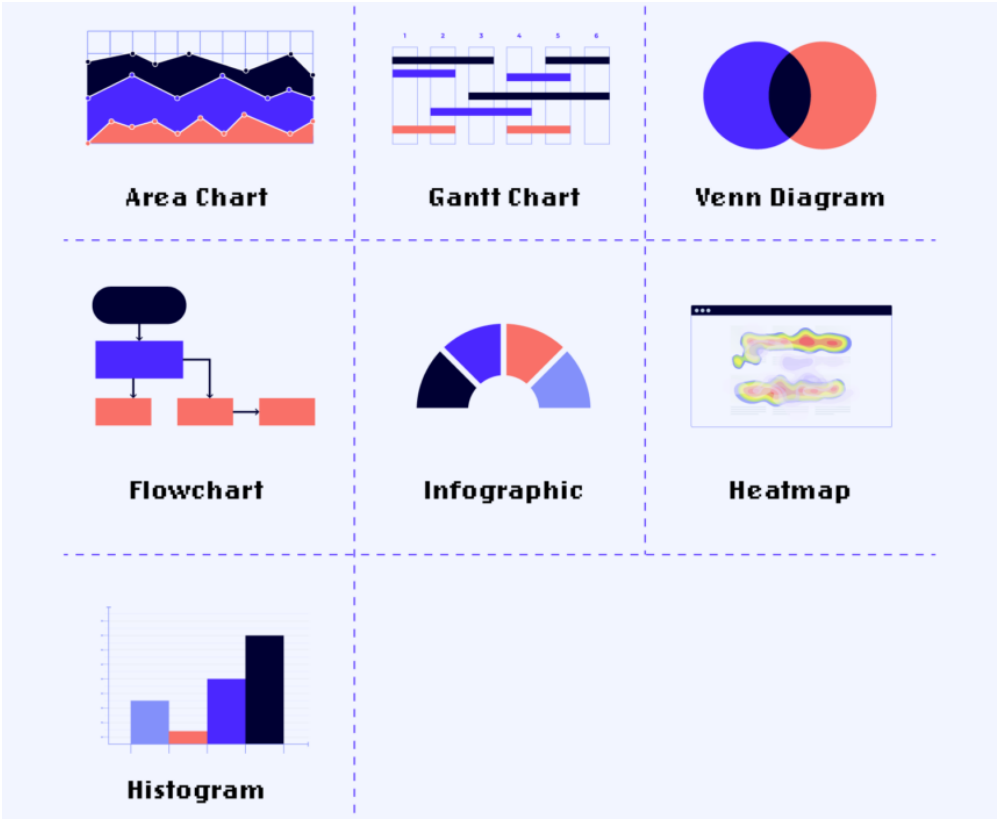


Fig. 9.4: Visualization methods for business & finance

Example:

A global company employs an interactive business intelligence dashboard to monitor revenue, costs, and profitability by region. The dashboard emphasizes financial KPIs using bar charts and line graphs as shown in the fig 9.4, enabling executives to make fast, smart decisions.

2. Healthcare & Epidemiology:

The medical industry applies data visualization to monitor outbreaks of diseases, compare patient demographics, and enhance treatment plans using various visualization methods like shown in the fig 9.5.

Strong visualizations assist policymakers, scientists, and healthcare workers in spotting trends and in the effective distribution of resources.

Major Visualization Techniques:

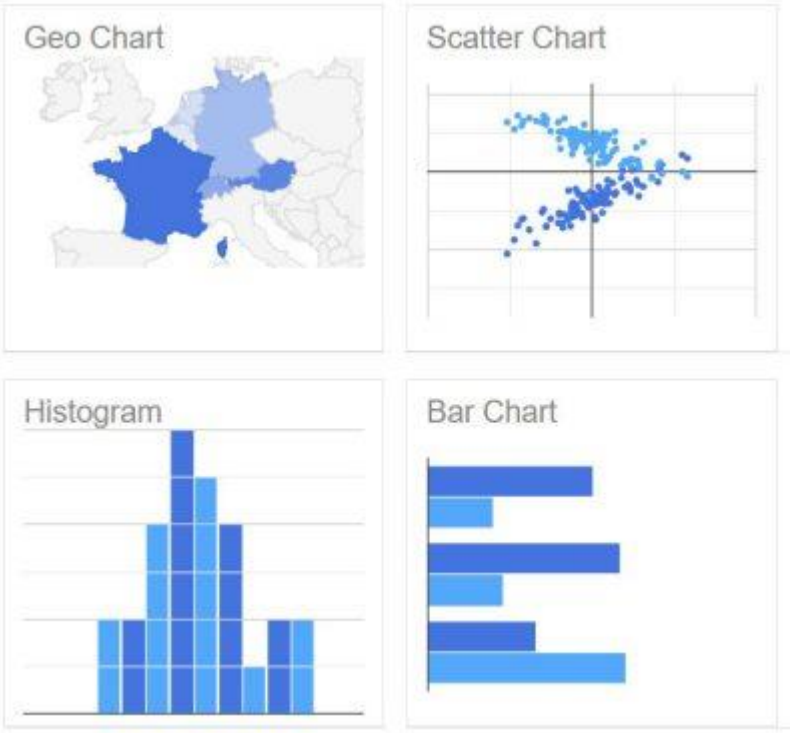


Fig. 9.5: Visualization methods for healthcare

i. **Line Charts:** It follows the spread of disease (e.g., new daily COVID-19 cases).
ii. **Scatter Plots:** It examines the correlation between risk factors (e.g., age and severity of disease).
iii. **Bubble Charts:** It contrasts the rates of disease prevalence in various regions or populations.
iv. **Geospatial Maps (Choropleth Maps):** It displays disease outbreaks geographically.
v. **Histograms:** These present patient age range or hospital admission patterns.

Example:

During the COVID-19 pandemic, Johns Hopkins University's interactive dashboard employed geospatial maps and time-series graphs to map the virus's global spread. The tool enabled policymakers and the public to monitor real-time infection rates and health responses.

3. Marketing & Consumer Analytics:

Marketing experts use data visualization to monitor consumer behaviour, measure campaign success, and optimize ad strategies. Through visual analytics, companies can enhance customer interaction and achieve maximum return on investment (ROI).

Visualization techniques used:

i. **Funnel Charts:** It monitors the conversion of visitors into customers.
ii. **Sankey Diagrams:** They illustrate customer journeys and determine drop-off points in sales funnels.
iii. **Heatmaps:** They display user interaction on websites with high-engagement areas.
iv. **Word Clouds:** It draws important themes from social media conversations and customer reviews**.**
v. **Pie Charts:** It divides customer demographics by age, income, or interests.

Example:

A digital marketing firm examines website heatmaps to see where visitors click most, thereby enhancing webpage design and user experience.

4. Social & Political Sciences:

Social scientists and political analysts use data visualization to study public sentiment, voting behaviour, and demographic changes. These insights are crucial for policy-making, electoral strategies, and media analysis.

Key Visualization Methods:

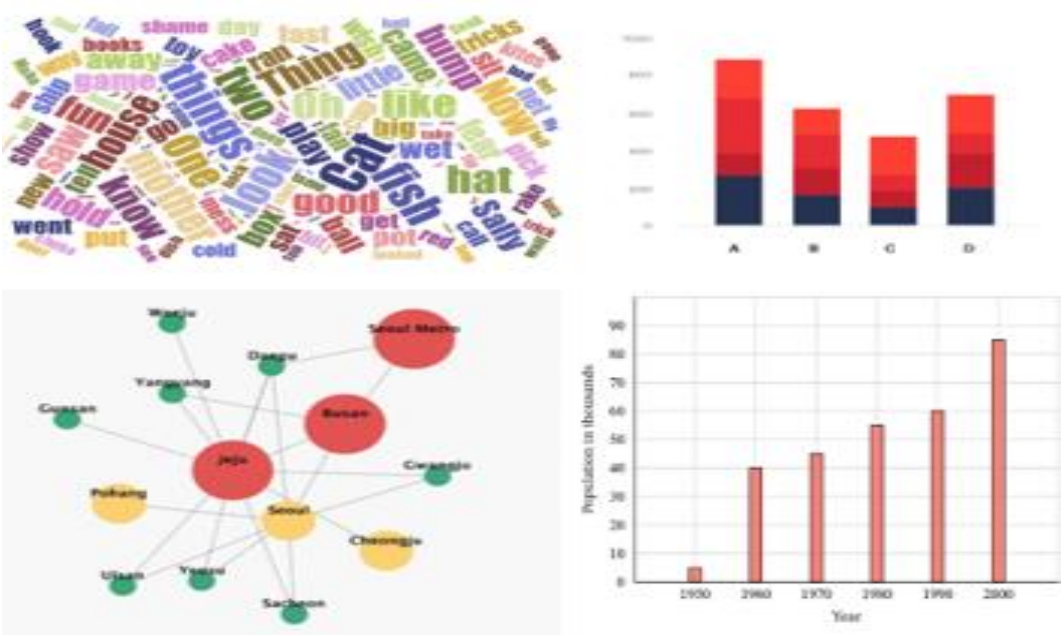


Fig. 9.6: Visualization for Social & Political Science

i. **Choropleth Maps**: It displays election results by state, district, or county.
ii. **Bar Charts**: It compares survey results on social issues across different populations.
iii. **Word Clouds**: It summarizes key themes from public debates, speeches, or news articles.
iv. **Network Graphs**: They show relationships between political entities, influencers, or social media users.
v. **Stacked Bar Charts**: They Visualize demographic distributions (e.g., education levels of voters).

Example:

During a presidential election, news agencies like The New York Times use interactive maps to display real-time voting results, allowing users to explore voting patterns across different regions using above mentioned graphs shown in fig 9.6.

5. Engineering & Scientific Research:

Scientists and engineers apply data visualization to examine experiment outcomes, track system behaviour, and improve designs. Large datasets need to be shown in a straightforward, relevant manner so that research and development can be enabled.

Major Visualization Techniques:

i. **Scatter Plots:** It illustrates variable relationships (e.g., material strength as a function of temperature).
ii. **Box Plots:** It illustrates statistical distributions of experimental data.

iii. **Time-Series Graphs:** It examines system behaviour over time (e.g., sensor values within an IoT network).

iv. **3D Surface Plots:** These visualize intricate physical or chemical processes.

v. **Histograms:** They visualize frequency distributions of experimental data.

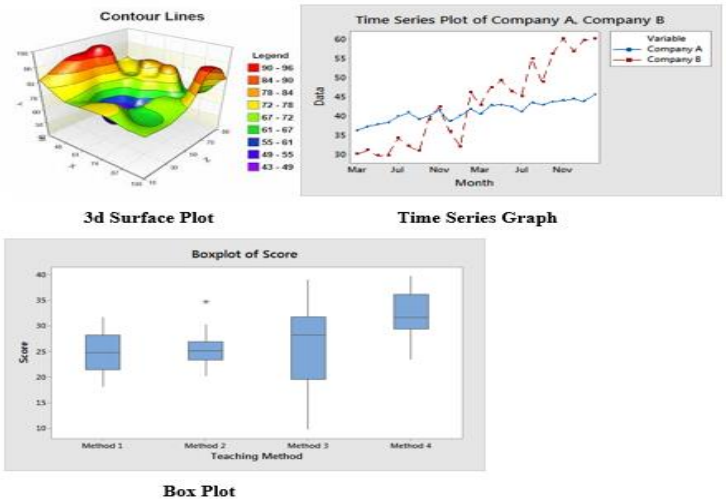


Fig. 9.7: Visualization for Engineering

Example:

A climate scientist plots global temperature change over the last century with line graphs and heatmaps and boxplots as shown in fig 9.7 to show the effects of climate change.

9.3 Design an Interactive data visualization storyboard for real-time data:

With the current big data age, interactive data visualization is indispensable in deriving sensible insights from giant datasets. Well-crafted storytelling allows users to spot trends, engage with information, and make quick decisions based on facts. Below is an interactive data visualization storyboard with the aim of analyzing Netflix data using director, release year, budget, language, IMDb rating, genre, and other attributes.

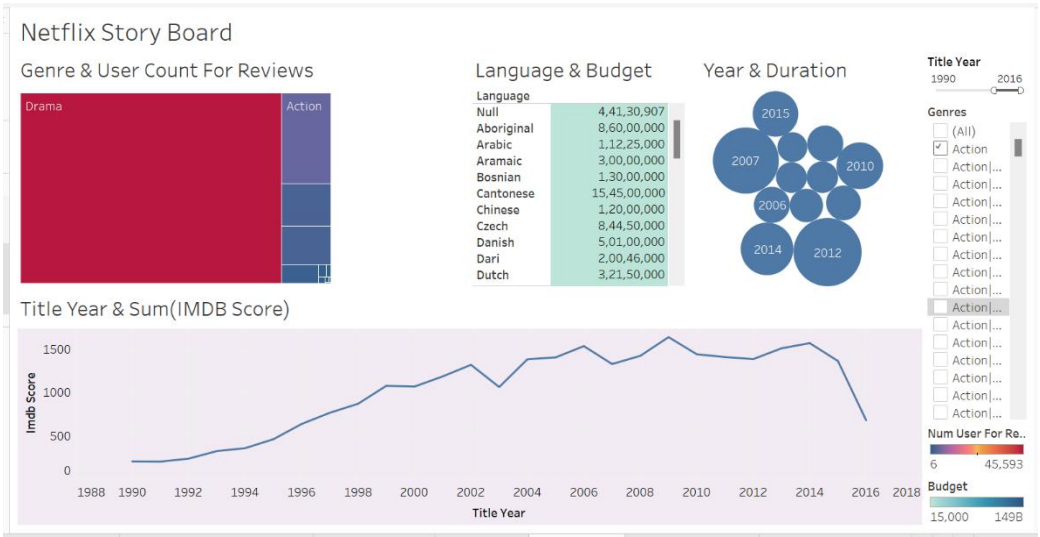


Fig. 9.8: Netflix Story Board in Tableau

Storyboard Objectives:

The objective of the interactive visualization is to allow users to have a fun and informative means of viewing Netflix content trends, determining the best genres, evaluating budget patterns, analyzing ratings

from IMDb, and knowing how different directors and languages affect the content on the streaming service. The dashboard will be user-friendly, dynamic, and easy on the eyes.

Data visualization plays a vital role in contemporary digital interfaces, with the ability to facilitate users' interaction with and exploration of large volumes of data. Netflix, being a top international streaming platform, utilizes interactive data visualization methods to optimize user experience and data discovery. This chapter discusses Netflix's real-time capabilities, mechanisms of user interaction, and visual components, highlighting in fig 9.8; how this drive an intuitive and interactive data-centric interface.

Real-time features and user interactions:

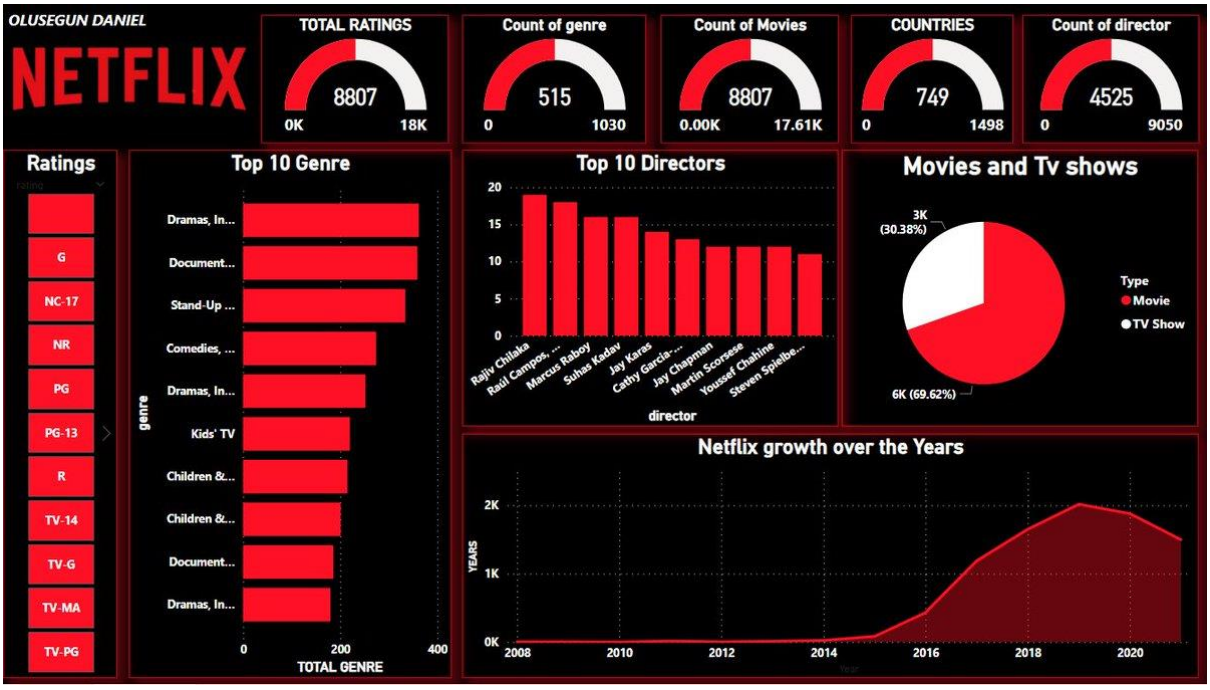


Fig. 9.9: Netflix visualization using different graphs

1. Filters and Dropdown Menus for Dynamic Data Filtering:

Netflix uses filters and dropdown menus to enable users to dynamically filter datasets according to their interests. This feature makes it possible for users to browse through particular categories like:

i. **Genre:** Action, Drama, Comedy, Thriller, etc.
ii. **Year of Release:** Search for films and television shows by particular time ranges.
iii. **IMDb Rating:** Choose content on the basis of user ratings and critic scores.
iv. **Language and Country of Production:** Filter by language preferences or regional content.

These filtering tools offer personalized and pertinent results so that users are able to examine and explore content effectively as illustrated in the fig 9.9.

2. Hover Tooltips for Contextual Information:

Hover tooltips improve data visualization by offering extra information when users hover over objects in the interface. Hover tooltips normally show:

i. The quantity of films or television programs related to a given data point.
ii. Title, genre, and rating of the movie on a mouse-over a data marker.
iii. Other metadata, e.g., release date, director, or runtime.

Through hover tooltips, Netflix enhances usability and understanding without overloading users with too much information on the primary display.

3. Real-Time Data Updates:

Netflix constantly updates and refreshes its information in real-time to keep it accurate and current. This feature is especially valuable for:

i. **Trending content:** The most recent popular television shows and films are updated according to viewership metrics.

ii. **Genre rankings:** The ranking of genres is automatically modified by the system based on recent viewing patterns.

iii. **IMDb ratings:** Any fluctuation in IMDb ratings is updated dynamically within the visualization.

Real-time updates guarantee that the latest information is always accessible, promoting users' and analysts' decision-making.

4. Drill-Down Capability for In-Depth Exploration:

Netflix also features drill-down functionality, enabling users to click on visual components to get additional data. The functionality is applied(shown in fig 9.10):

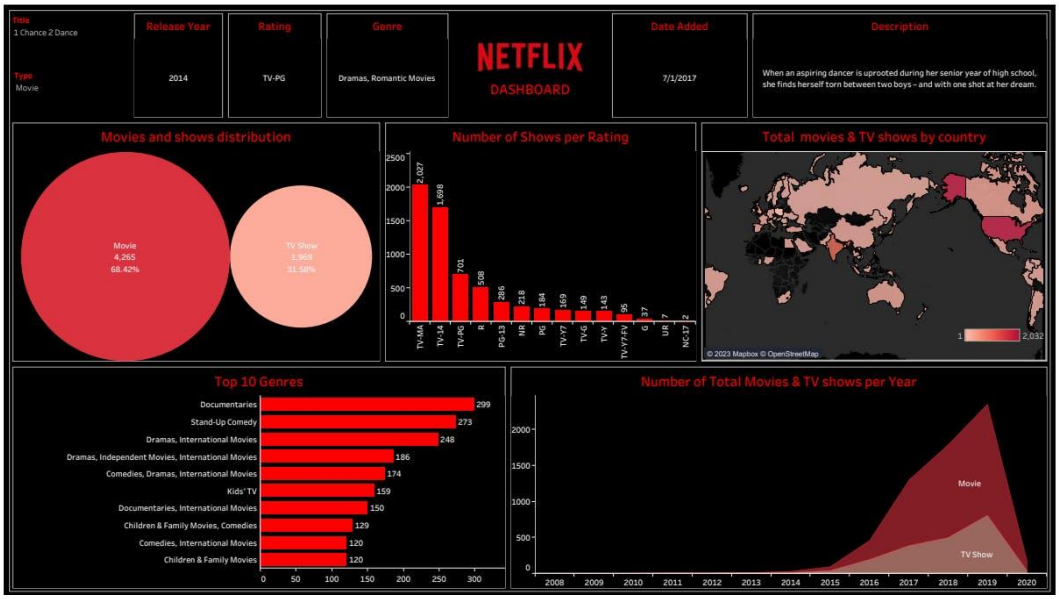


Fig. 9.10: Netflix visual components of data visualization

i. **Geospatial maps:** A click on a nation reveals the number of movies or TV shows produced there.

ii. **Trend charts by genre:** Users can view genre-based trends by choosing specific genres or years.

iii. **Best-performing content:** A click on a particular show or film gives additional information regarding its popularity, rating, and reviews.

Drill-down functionality supports a tiered data exploration strategy, where users can move from macro-level observations to in-depth analysis.

9.4 Netflix's Key Visual Elements for Data Representation:

1. Key metrics, or KPI cards:

Netflix employs Key Performance Indicator (KPI) cards to summarize high-level summary statistics in a readable format. Such KPI cards illustrated in fig 9.11 show key data points including:

i. Total movies and TV shows on the platform.

ii. Most watched genre, based on viewing statistics.

iii. Top three languages, which refer to the most commonly available content languages.

iv. Average IMDb rating for all titles accessible, giving the quality of the content.

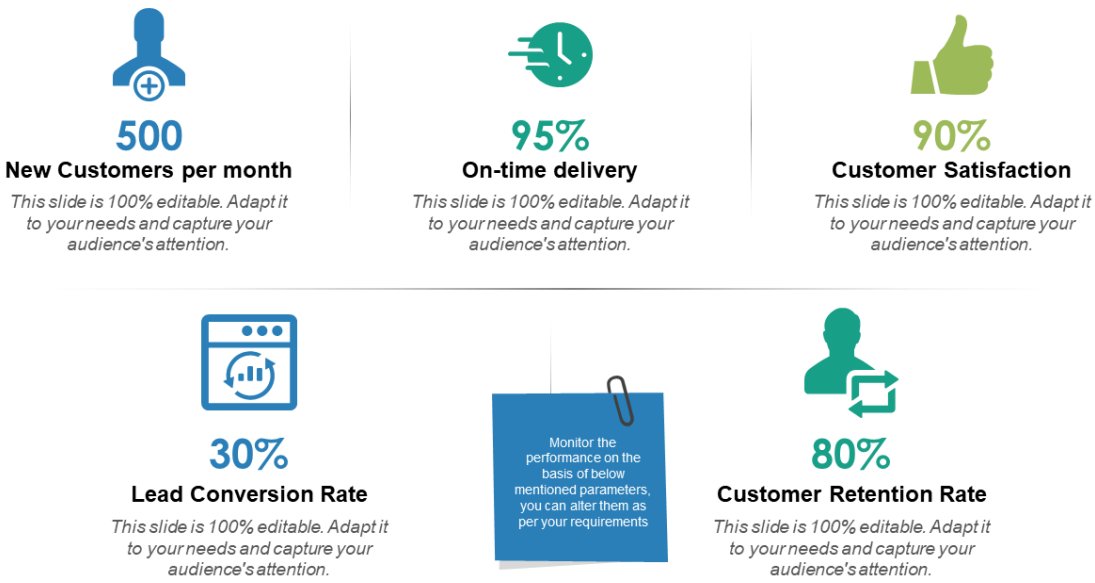


Fig. 9.11: KPI cards

KPI cards provide an instant glance at key measures, enabling people to understand the key information.

2. Interactive World Map for Geospatial Analysis

Netflix's interactive world map depicts the worldwide distribution of its movie and TV show productions visually. This map allows users to:

i. Identify the locations of film and TV show productions.
ii. Click on a country to see the total number of movies or TV shows produced there.
iii. Examine geographic content trends to enable users to discover country-by-country entertainment economies.

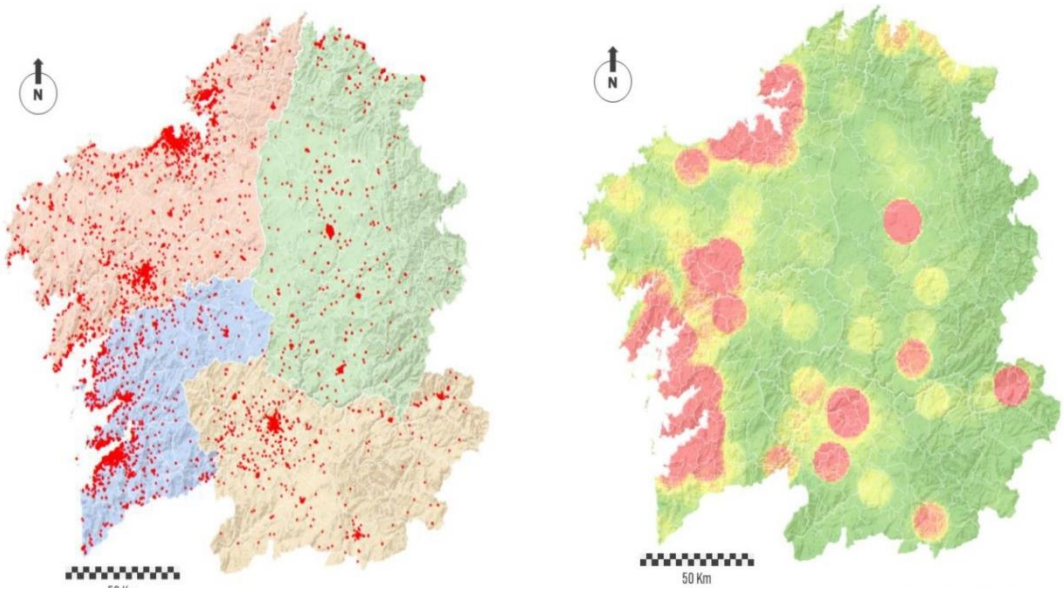


Fig. 9.12: Geospatial Analysis

Fig 9.12 represents geospatial visualization offers a natural means of investigating worldwide media trends and regional production centres.

3. Content Trend by Year (Line Chart):

Netflix uses line charts to demonstrate the trends of content production across time. The visualization emphasizes:

i. The number of movies and series released each year to monitor the growth of the industry.
ii. Filtering options so that users can analyze trends in terms of:
    - Director: Display content created by directors.
    - Genre: Evaluate which genres are on the rise or decline.
    - Language: Examine content availability in various languages.
    - IMDb Rating: Review the development of highly rated content.

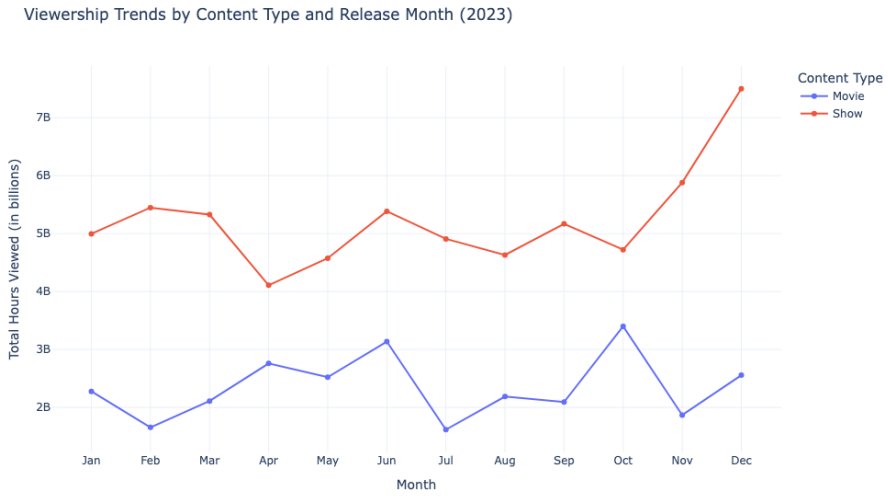


Fig. 9.13: Trends using line graph

By allowing interactive filtering, this chart (as shown in fig 9.13) enables monitoring content trends by multiple parameters, allowing valuable insights to be gained regarding industry trends.

Netflix's interactive data visualization platform is built to give real-time insights, dynamic filtering, and profound data exploration. With functionalities like dropdown filters, hover tooltips, real-time refresh, and drill-down, the users can efficiently analyze content trends and find worthwhile insights. The important visual aspects like KPI cards, interactive maps, and trend charts make data storytelling easy and provide better access to complicated information.

By combining these sophisticated data visualization methods, Netflix provides smooth, informative, and engaging user experience, creating a standard for data-driven websites.

Interactive Maps and Geospatial Representation:

One of the most characteristic aspects of NYT's election visualizations is its interactive geospatial maps, which present election results at various levels, including:

- State level (for presidential and gubernatorial elections)
- County level (for detailed voting patterns)
- District level (for congressional elections)

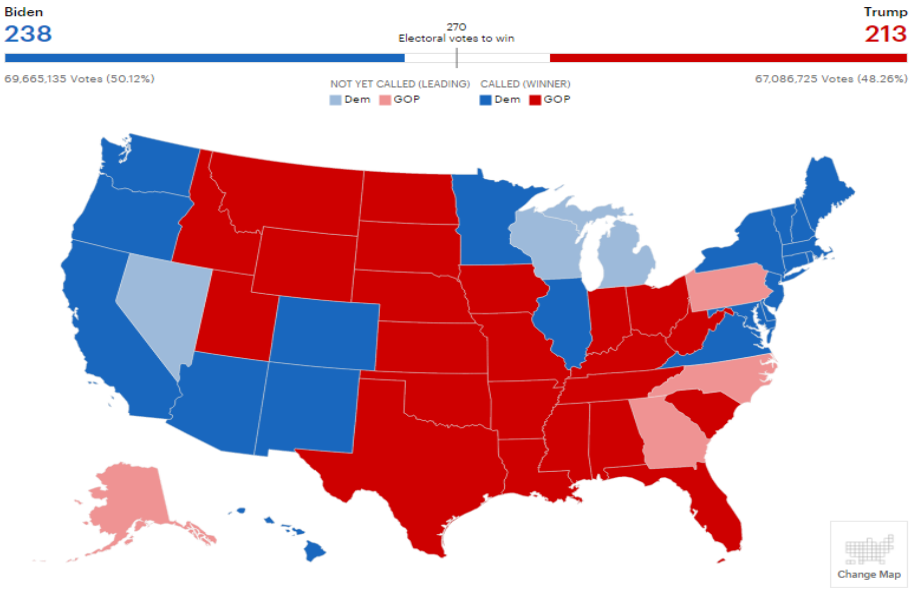


Fig. 9.14: Election data visualization

These are color-coded, usually blue for the Democrats and red for Republicans, to mirror real-time vote totals as shown in fig 9.14. Interactive features enable the viewer to move their cursor over or click on an area to access detailed figures, including overall votes, percentage of margin, and past voting behaviour. This helps to better present election trends and allows the reader to make comparisons between current and previous elections.

One of the strongest aspects of NYT's geospatial visualizations is their capacity to display electoral changes across time. An example is that users can look at how particular counties or states have politically changed from the past elections to the present, giving indications of swing states as well as changes in voting behaviour.

Chapter 10

CASE STUDIES ON REAL-WORLD DATA VISUALIZATION

Jasmeet Kaur[1], Babita Sidhu[2] and Jaskiran Ghotra[3]

[1]GNA University, Phagwara

[2]LKCTC, Jalandhar

[3]Guru Nanak institute of engineering & Management, Naushehra

10.1 Case Study 1: Google Flu Trends (Public Health Data Visualization):

Google Flu Trends (GFT) was a comprehensive public health visualization project initiated by Google in 2008 with the goal of monitoring and predicting flu epidemics in real time. The mechanism used Google flu-related search terms as a barometer of the prevalence of the flu in varying geographic locations. Using analysis of millions of global search queries, Google attempted to deliver lead indications of flu activity in a complementary effort alongside other methods applied by public health authorities such as the Centers for Disease Control and Prevention (CDC). Visualization from GFT stood out the most as it projected flu activity through an interactive heat map on which densely searched regions in darker color intensity were the pointers toward impending flu activity which is shown in the Fig 10.1

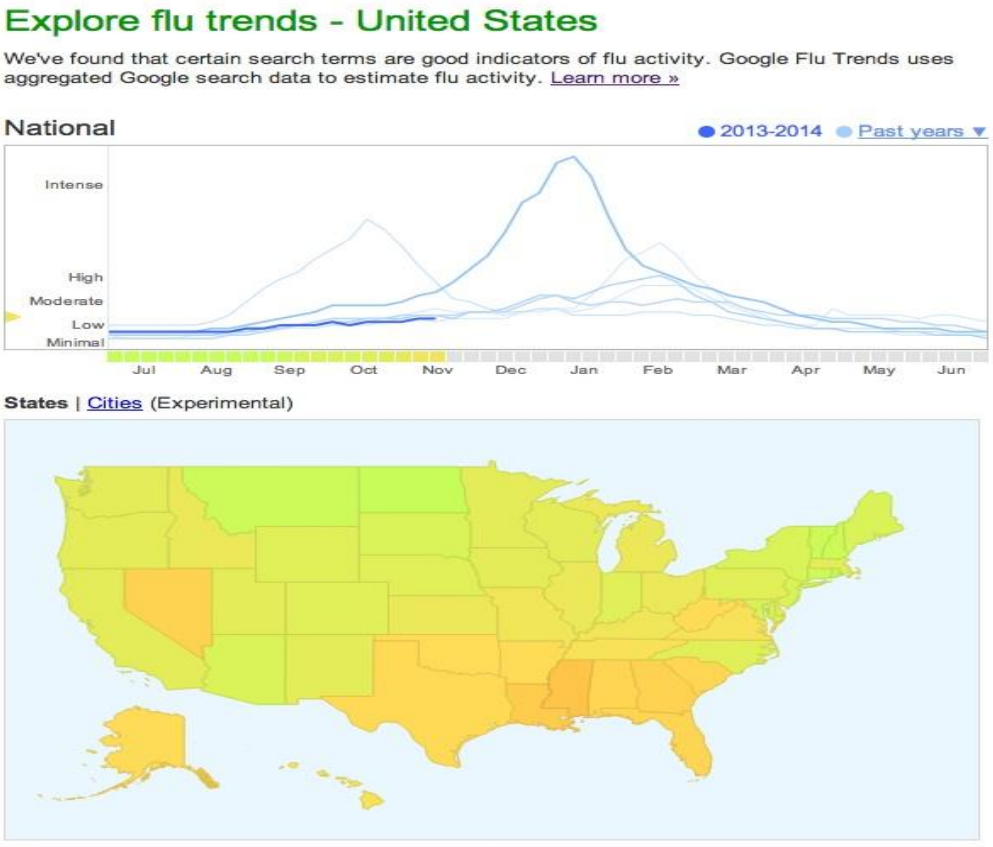


Fig. 10.1: Public health data visualization

The biggest strength of Google Flu Trends was its velocity and real-time monitoring feature. Conventional flu surveillance depended on hospital reports and laboratory confirmation, which resulted in delayed identification of outbreaks. With GFT, there were real-time insights offered based on Internet search patterns, enabling policymakers and health workers to act preemptively to flu patterns.

Nevertheless, with all its innovative spirit, the project did encounter some daunting challenges. By 2013, studies revealed that GFT systematically overestimated flu activity, primarily because of variations in search patterns shaped by media coverage, seasonal factors, and algorithmic biases. The absence of integration with clinical data sources further compromised its predictive power. Due to this, Google shut down the project in 2015, focusing on collaborative models with well-established public health institutions.

For all its flaws, Google Flu Trends is an early trailblazer of public health visualization and the power of big data analysis for disease surveillance. It illustrated how real-time digital trails were able to improve epidemic surveillance, and how future developments in AI-based health forecasting could be considered next. The case also underscores the need for data validation, interdisciplinary collaboration, and careful interpretation of big data trends to ensure that predictive models are not only visually appealing but also scientifically sound [41].

10.2 Case Study 2: New York Times – Election Data Visualization:

Election data visualization is important in the improvement of public knowledge of electoral results, voter patterns, and political environments. The New York Times (NYT) is well known for its creative and interactive election data visualizations that offer real-time information, comparative historical data, and analytical insights on presidential, congressional, and local elections. The NYT method of visualizing election data combines statistical precision, engaging narration, and interactive usability to provide sophisticated electoral information broadly and engagingly [42].

Interactive Maps and Geospatial Representation:

One of the most characteristic aspects of NYT's election visualizations is its interactive geospatial maps, which present election results at various levels, including:

- State level (for presidential and gubernatorial elections)
- County level (for detailed voting patterns)
- District level (for congressional elections)

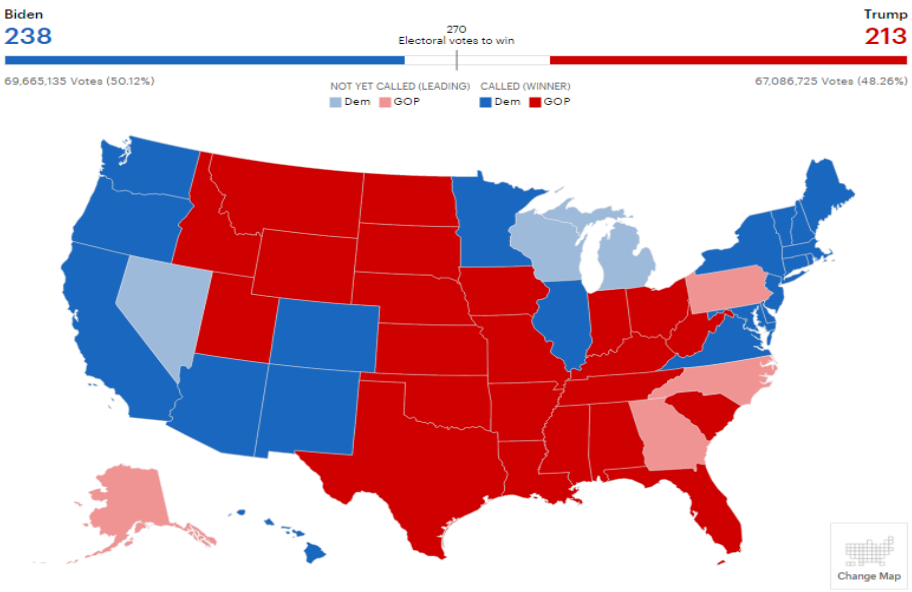


Fig. 10.2: Election data visualization

These are color-coded, usually blue for the Democrats and red for Republicans, to mirror real-time vote totals as shown in Fig 10.2. Interactive features enable the viewer to move their cursor over or click on an area to access detailed figures, including overall votes, percentage of margin, and past voting behaviour. This helps to better present election trends and allows the reader to make comparisons between current and previous elections.

One of the strongest aspects of NYT's geospatial visualizations is their capacity to display electoral changes across time. An example is that users can look at how particular counties or states have politically changed from the past elections to the present, giving indications of swing states as well as changes in voting behavior.

Real-Time Vote Tracking and Uncertainty Modeling:

Throughout elections, the NYT regularly refreshes its visualizations to match the most up-to-date vote tallies and projections. Vote progress bars that indicate the share of ballots counted and live updates on projected victors are included in their system. One innovation of their election coverage is the "election needle", a live forecasting gauge that indicates predictions using arriving vote counts.

The election needle moves dynamically from one political party to another with the arrival of new data, providing users with an estimate in real time of which candidate has the best chance of winning. But this functionality has also caused controversy, since sharp fluctuations in predictions can be confusing or create unnecessary anxiety among viewers. Nonetheless, the needle is a valuable illustration of how statistical uncertainty can be graphically illustrated in election prediction.

Historical Comparisons and Trend Analysis:

In addition to real-time data, the NYT provides historical context by allowing users to compare election results from different years. Their platform includes side-by-side electoral maps, showing how party control has changed over multiple election cycles. They also offer trend lines and bar charts that illustrate shifts in voter preferences based on factors like:

i. Demographics (race, gender, age, and education).
ii. Urban vs. rural voting patterns.
iii. Key policy issues influencing voter decisions.

This longitudinal analysis helps users understand broader political realignments, such as the increasing suburban support for Democrats or the Republican dominance in rural areas. By providing historical comparisons, the NYT allows readers to see elections not as isolated events but as part of a broader political evolution.

User Engagement and Customization:

To make election data more accessible, the NYT offers customization features that allow users to tailor their analysis based on specific interests. Readers can filter results by state, county, or district, explore presidential, Senate, and House races separately, and use search functionality to find specific races or candidates.

This interactive experience ensures that the visualization serves both casual readers and political analysts, providing different levels of detail depending on the user's needs. The ability to drill down into granular data enables deeper insights into local election dynamics, which can often be overshadowed by national trends.

Impact and Challenges:

The New York Times' election visualizations have gained international acclaim for their timeliness, clarity, and accuracy, serving as the gold standard for election data journalism. Millions use the visualizations to inform themselves about election outcomes, making the NYT arguably the most trusted source of information during election seasons.

Their methodology has also been met with challenges, such as

i. **Statistical uncertainty interpretation:** Many users are perplexed by live projections, such as the election needle.
ii. **Mitigating misinformation risks:** Precluding the possibility of misleading narratives being generated through early vote counts.
iii. **Preventing bias in portrayal:** Selecting the color palette, emphasizing data, and geographical priority can affect opinions.

In spite of these difficulties, the NYT's election visualizations are still a gold standard of political data visualization, providing a potent tool for voters, journalists, and analysts.

The New York Times' election data visualization is an example of the potential of interactive storytelling in journalism. Through the integration of real-time vote tracking, geospatial mapping, historical comparisons, and customizable options, they offer a complete and understandable picture of election results. Their effort showcases the significance of accuracy, interactivity, and transparency in data visualization, simplifying complex electoral data for better understanding and analysis. Going forward, advancements in machine learning and AI-driven analytics could further fine-tune election forecasting models and improve the way we visualize and interpret political data in the digital age.

Chapter 11

FUTURE TRENDS IN DATA VISUALIZATION

Manpreet Kaur[1], Simran[2] and Tarun Bhalla[3]

[1,2]GNA University, Phagwara

[3]Anand College of Engineering and Management, Kapurthala

Data visualization has changed dramatically over the past few years, fueled by technological advancements, artificial intelligence, and growing data complexity. As companies continue to produce enormous volumes of data, more advanced, interactive, and real-time visualization methods have become a necessity. This chapter discusses future trends in data visualization, with emphasis on the use of AI, interactive and real-time visualization, and best practices for data scientists and analysts.

11.1 The Role of AI in Data Visualization:

Artificial intelligence (AI) is revolutionizing data visualization by streamlining processes, improving insights, and making data more accessible. The use of AI in data visualization offers several major advancements:

1. Automated Data Analysis and Storytelling:

Visualization tools that are powered by artificial intelligence (AI) can examine datasets automatically, recognize patterns, and produce insightful results. These tools facilitate the interpretation of intricate data by delivering transparent visualizations without the need for extensive technical knowledge.

- Dashboards that are AI-powered offer relevant visualizations based on attributes of datasets.
- Natural Language Processing (NLP) allows people to pose questions in English and be provided with instantaneous visual insight.

2. Predictive and Prescriptive Analytics:

AI aids in visualization through the use of predictive analytics, which predicts trends based on past information. Further, prescriptive analytics provides a recommendation for action using real-time data.

- Machine learning algorithms model future possibilities for visualization to inform decision-making.
- Anomaly detection through AI brings attention to aberrant trends or outliers without any intervention.

3. Augmented Analytics:

Augmented analytics enabled by AI automates data preparation, insight generation, and explanation, allowing organizations to make quicker and better-informed decisions.

- AI identifies causations and correlations in big data.
- Smart data summarization reduces big amounts of data into simple-to-consume visuals.

4. Personalized and Adaptive Visualizations:

AI enables dynamic and user-specific visualization that adapts to user behaviour and prefer-ences.

- AI-powered recommendation engines recommend optimal visual forms to various users.
- Adaptive dashboards alter their layout depending on the user's requirements and areas of interest.

11.2 The Future of Interactive and Real-Time Visualization:

With growing data availability and computational power, interactive and real-time data visualization is becoming imperative. Organizations and companies need dashboards that not only display static insights but also enable dynamic exploration and real-time data updates.

1. Real-Time Data Processing:

The future of data visualization lies in real-time processing, where visualizations update dynamically as data streams in.

- Businesses use real-time dashboards to track KPIs, stock prices, and customer interactions.
- IoT (Internet of Things) devices enable real-time monitoring of sensor data.
- Social media sentiment analysis tools update live trends and reactions.

2. Immersive and 3D Visualization:

Virtual reality (VR) and augmented reality (AR) make new types of immersive data visualization possible.

- 3D data models allow for the visualization of complicated datasets, like geospatial analysis as shown in fig 10.1.
- AR overlays provide a way for users to interact with data in a real-world context.
- Virtual environments support data exploration in teams.

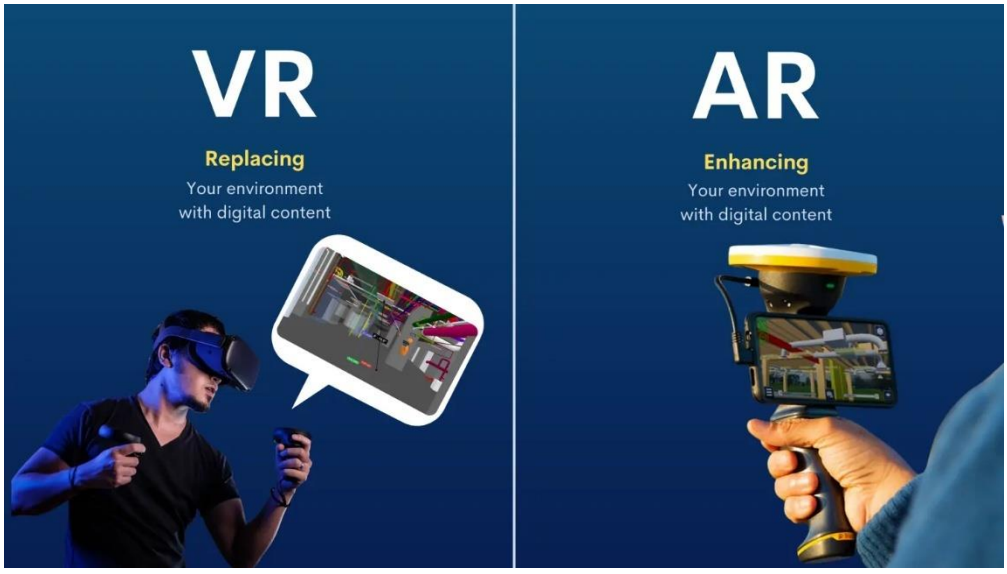


Fig. 11.1: Virtual Reality & Augmented Reality

3. Gesture and Voice-Enabled Interactions:

With growing advancements in machine learning and AI, new interfaces with data visualization are being explored, such as:

- Voice filtering and querying datasets (e.g., "Display sales trends in the last quarter").
- Immersive controls using gestures to zoom, pan, and choose data points as shown with the results in fig 10.2.

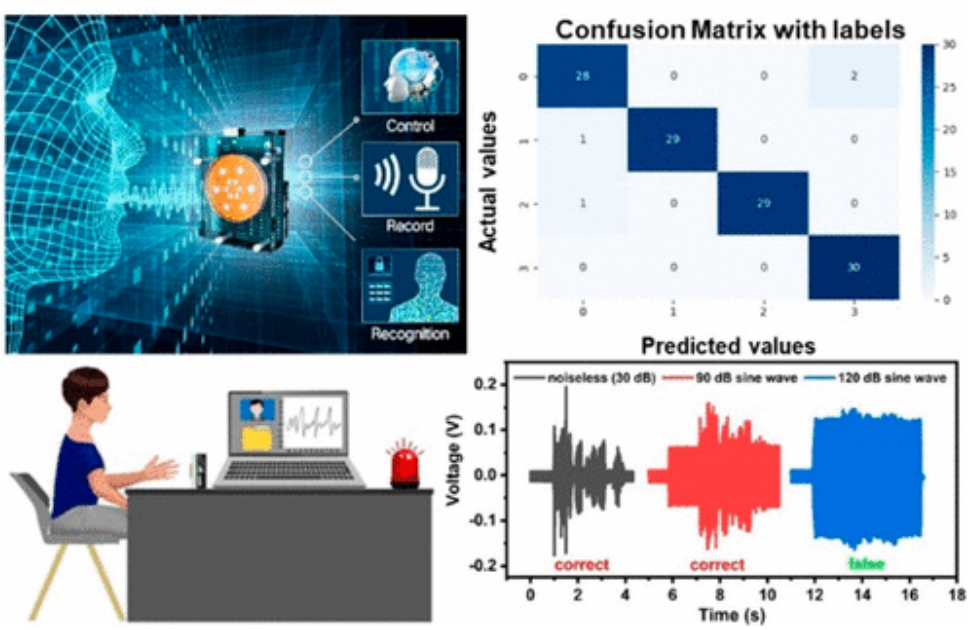


Fig. 11.2: Gesture & voice enabled interactions

4. Blockchain and Data Transparency

As concerns for data security and integrity grow, blockchain technology is increasingly being used in data visualization systems to make them transparent and trustworthy.

- Blockchain-based dashboards authenticate and show tamper-proof records of transactions.
- Decentralized data visualization prevents data from being manipulated by a single party.

11.3 Best Practices for Data Analysts and Scientists:

Data scientists and analysts need to adhere to a systematic process to ensure that their visualizations convey insights effectively while being clear and accurate. The following are key best practices that need to be embraced to make data visualization more effective:

1. Choosing the Right Visualization for the Right Data:

Picking the right kind of visualization is paramount to making sure that the data is well communicated. Various kinds of data need various visual representation methods.

- **Bar charts** work best when comparing discrete categories.
- **Line charts** are good for viewing trends over time.
- **Heatmaps** come in handy for finding correlations in big datasets.
- **Scatter plots** serve to visualize relationships between two variables.
- Avoid excessive or deceptive visual components that can misrepresent the data's interpretation.

2. Ensuring Data Accuracy and Integrity:

Deceptive or false data can lead to bad decisions. It is important to confirm that the data being visualized is accurate and trustworthy.

- Authenticate data sources before developing visualizations.
- Compare insights against different datasets for confirmation.
- Utilize clear methodologies when processing data to avoid errors and misrepresentation.

3. Enhancing User Interactivity:

Interactive functionalities provide data visualizations with an engaging and interactive experience, allowing people to interact with data dynamically.

- Add filtering, sorting, and drill-down capabilities to help users zoom into certain details.
- Utilize hover tips to give people more contextual information without messing up the main view.
- Provide customizability options so people can customize dashboards according to their requirements.

4. Optimizing for Mobile and Cross-Platform Compatibility:

Data visualizations must be accessible on various devices and screen sizes.

- Make dashboards and reports responsive and perform well on mobile devices.
- Select lightweight visualization libraries that are compatible with multiple screen resolutions and operating systems.
- Minimize loading times to avoid data rendering delays.

5. Incorporating AI-Powered Insights:

AI-powered analytics can enrich data visualization by delivering richer insights and predictive analytics.

- Use machine learning models to identify patterns and trends.
- Implement automated alerts to inform users of anomalies or substantial data changes.
- Utilize AI-driven recommendations to provide optimal visualization methods for datasets.

6. Promoting Ethical and Inclusive Data Visualization:

Ethics have a crucial part in ensuring that data visualizations are unbiased and inclusive for all users.

- Steer clear of biased representations that might mislead the audience or misinterpret results.
- Utilize colorblind-friendly palettes to make the visualizations accessible for visually impaired users.
- Offer alternative descriptions of visual information to support differently abled users.
- Be transparent by explicitly listing assumptions, sources of data, and any limitations.

11.4 Conclusion:

The future of data visualization is on the cusp of radical transformation by AI, real-time processing, and immersive technology. As businesses and sectors keep adopting sophisticated visualization methods, the work of data analysts and scientists will become even more vital in ensuring accuracy, reliability, and ethical usage of data.

By embracing AI-driven automation, interactive narration, and adaptive visual composition, professionals can craft more informative and interactive data visualizations. Further, as ethical and accessibility considerations take centre stage, the emphasis must be placed on crafting inclusive and transparent visualizations that reach various audiences.

Finally, the secret to the future of data visualization is balancing technology with human insight. With the help of cutting-edge tools and best practices, data professionals can guarantee that visualizations are effective decision-making, communication, and knowledge-discovery instruments for domains ranging from business to science.

Chapter 12

IMAGE ENHANCEMENT TECHNIQUES IN DIGITAL IMAGE PROCESSING

Bhoomi Gupta[1], Gagandeep Singh[2] and Sumit Chopra[3]

[1,2,3]GNA University, Phagwara

12.1 Introduction:

Digital Image Processing (DIP) is now an essential tool for contemporary technological and scientific endeavours. A wide range of methods is employed to carry out operations on digital images, with the overall aim of improving image quality or extracting useful information as shown in Fig.12.1. In the majority of fields of medical diagnostics, aerospace, industrial inspection, surveillance, and remote sensing, use of DIP is not only central but also fast changing with advances in computational capabilities and algorithmic advancements [43].

Of all the DIP operations, image improvement is of fundamental importance. It is the operation of transforming an image in such a way that it is suitable for a specific application or to improve its visual impact. The improvement techniques are not intended to add new information to the image, but rather to render existing information accessible to both human observers and machines.

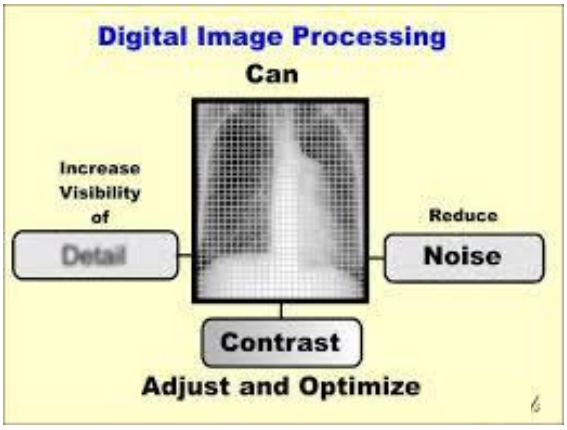


Fig. 12.1: Representing Digital Image Processing

Among its core techniques, **image enhancement** is pivotal in improving the quality, clarity, and interpretability of images. It aims to emphasize important features, reduce noise, and optimize contrast. This is especially crucial in applications such as satellite imaging, underwater photography, pathology, and biomedical research where original image quality may be compromised [43].

Image enhancement methods are typically classified into three broad categories:

- Spatial Domain Methods
- Frequency Domain Methods
- Fuzzy Domain Methods

Each of these categories serves distinct purposes and is suitable for specific contexts. This chapter presents a comprehensive review of these methods, their principles, strengths, limitations, and relevance to real-world applications.

12.1.1 The Purpose of Image Enhancement:

Image enhancement is a fundamental operation of digital image processing that is devoted to improving image quality and its usefulness. Images captured by various acquisition systems, such as digital cameras,

scanners, medical imaging devices, or satellites, are prone to different degradations. They might be too faint, defocused, noisy, non-homogeneous in contrast, or sensed in poor capture conditions. Those degradations can hide crucial information and have an effect on human and machine perception of an image [51].

In order to correct these issues, image enhancement techniques are applied to alter or enhance the image in a way that it becomes more suitable for visual examination or further computing processing. Improvement objectives can be categorized into three broad categories:

i. **Improving Human Visual Perception:** One of the primary goals of image enhancement is to produce an image that is more perceptually informative or more visually pleasing to the human eye. Human eyes are naturally contrast, brightness, edge, and color distribution sensitive. The enhancement algorithms try to emphasize these perceptual features so that observers can better interpret or understand the image contents as given in Fig.12.2 [42].

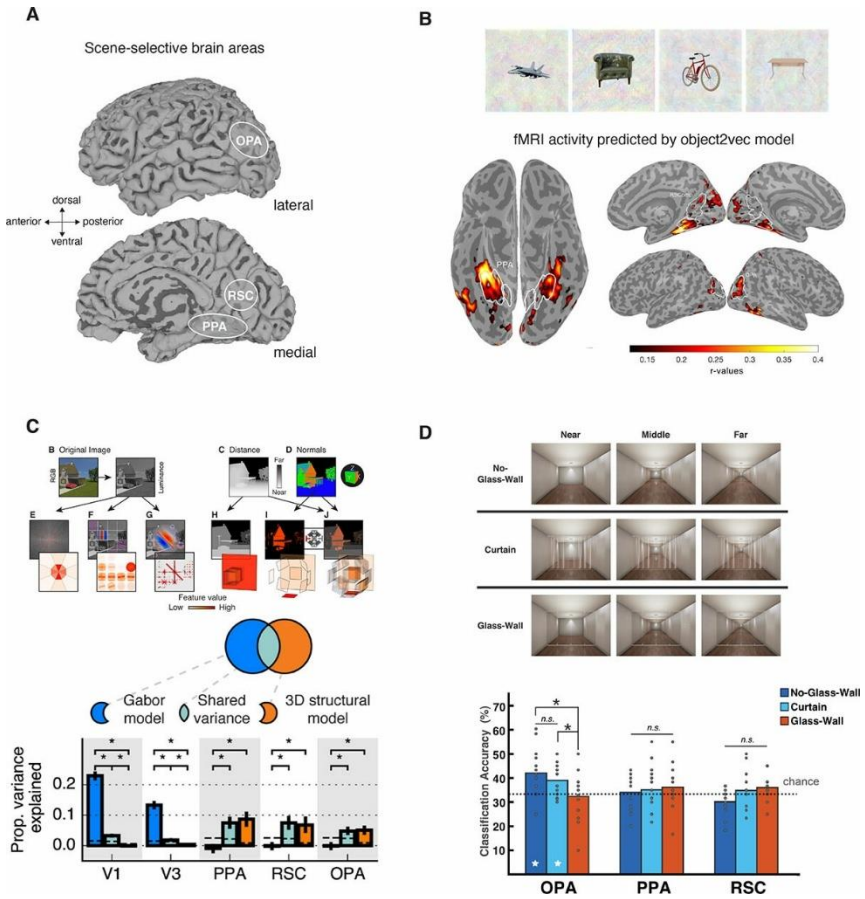


Fig. 12.2: Improving Human Visual Perception

Example Applications:

- Medical imaging contrast enhancement enables radiologists to distinguish more clearly between tissues of different densities. For example, a contrast between a normal tissue and a possible tumour can be made more distinguishable through methods such as histogram equalization or adaptive contrast enhancement.
- Facial feature enhancement or license plate number enhancement of low-light surveillance videos can assist criminal investigations in forensic analysis.
- In astro photography, normally imperceptible nebulae or stars can be made brighter by adjusting brightness and lowering background noise.

Lastly, the goal is to provide an image that promotes intuitive comprehension, correct judgment, and improved visual attractiveness.

ii. **Facilitating Automated Analysis:** As computer vision systems and AI algorithms increasingly become part of image analysis, images need to be improved in ways that maximize their fitness for automated analysis. Machines contrast images on the basis of numerical features, including pixel intensities, gradients, textures, and shapes, rather than subjective visual attractiveness [42].

Enhancement in this aspect entails:

- Noise reduction to prevent false alarms.
- Improving edge sharpness to enable object detection or segmentation.
- Improving feature perception to enable pattern recognition or clustering.

Example Applications:

- In industrial automation, sophisticated imaging enables the detection of micro-cracks, misalignment, or contamination on products on a production line.
- Deblurring and contrast enhancement improve vision systems in autonomous vehicles to enable them to recognize road signs, lane markings, or pedestrians better.
- In agriculture, improved drone images are utilized to estimate the health of crops or detect pest-infested regions through spectral imaging.

By improving the clarity and coherence of visual information, image enhancement enhances the accuracy, efficiency, and reliability of machine-dependent tasks.

iii. **Highlighting Specific Features:** In certain uses, one is not attempting to enhance the entire image as a composite but to enhance selectively certain individual features which are of greatest usefulness in the current application. These can be:
    - Edge enhancement to make edges and outlines more prominent.
    - Sharpening texture for enhancing surface contrasts.
    - Region-of-interest (ROI) brightening to bring out a particular region of interest in the image and decrease the background.

Such targeted improvement is particularly valuable in circumstances where:

- Small things (i.e., hairline breaks on X-rays or tiny imperfections on a circuit board) are worth examining.
- Some of the regions possess diagnostic or decision-critical information that must be separated and developed to facilitate accurate assessment.

Methods commonly employed for feature-specific enhancement are:

i. **Unsharp masking:** Unsharp masking is a common and old image improvement algorithm that is employed to render an image sharper by enhancing its edges. Interestingly, it renders the output image sharper than the original one, and its "unsharp" designation is due to the analog-photographic origin of this technique, given in Fig.12.3 [44].

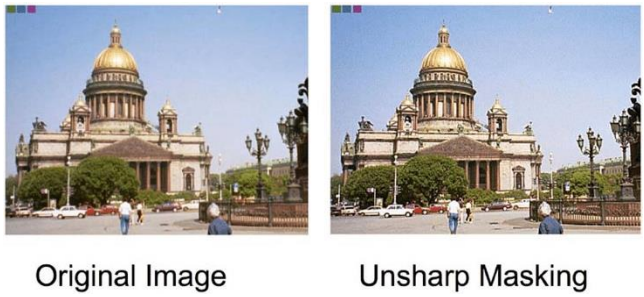


Fig. 12.3: Showing Unsharp Masking

How it Works:

i. **Blur the original image** using a smoothing filter such as a Gaussian blur.

ii. **Subtract the blurred image** from the original image to obtain the **mask**, which highlights edges and fine details.

iii. **Add the mask back** to the original image with a scaling factor to enhance sharpness.

Mathematical Expression:

Let:

- I be the original image,
- I_b be the blurred version,
- $M=I-I_b$ be the unsharp mask.

Then, the enhanced image I' is:

$$I'=I+k\cdot(I-Ib)$$

Where k is a gain factor controlling the sharpening intensity.

Applications:

- Medical imaging (e.g., enhancing X-ray or MRI images)
- Satellite imaging
- General photography (in software like Photoshop)

ii. **High-pass filtering:** High-pass filtering is an image processing method by which high-frequency components (edges and details) are allowed to pass and low-frequency components (smooth regions or background) are reduced, as shown in Fig. 12.4. It is used to accentuate abrupt intensity changes, typically occurring at edges [44].

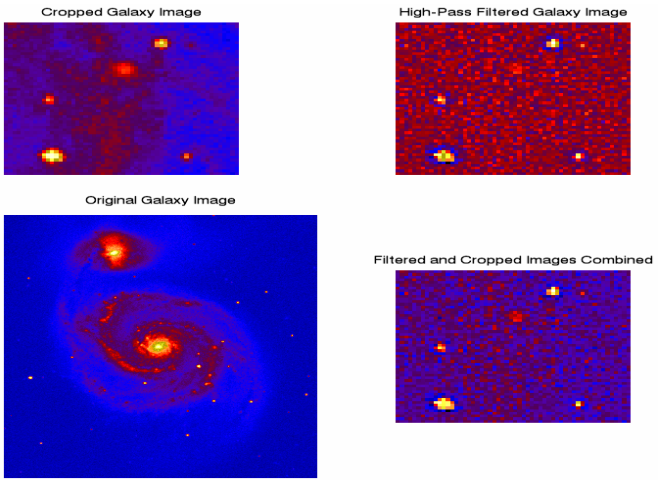


Fig. 12.4: Representation of High-Pass Filtering

How it Works:

- A convolution kernel (mask) is applied to the image where the center pixel has a **large positive value,** and the surrounding pixels have **negative or zero values**.
- This emphasizes regions with large intensity differences, effectively sharpening the image.

Example Kernel:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Effect:

- Enhances borders and edges
- Reduces the smooth areas
- Can increase noise if not used carefully

Applications:

- Edge detection
- Text recognition in documents
- Feature enhancement in satellite or microscopic images

iii. Gradient-based methods: Gradient-based improvement techniques are concerned with edge detection by calculation of the first-order derivatives (gradients) of the image intensity function. The magnitude of the gradient represents the strength of an edge, whereas the direction of the gradient indicates the orientation of the edge, as shown in Fig.12.5.

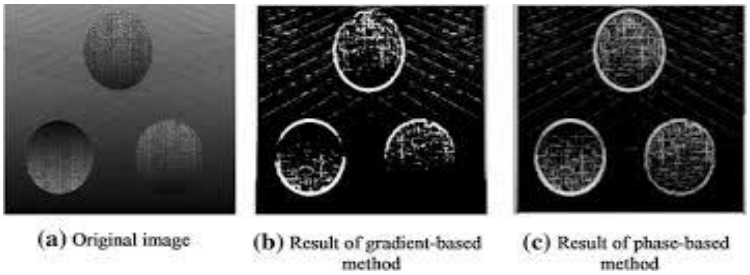


Fig. 12.5: Gradient Based Methods

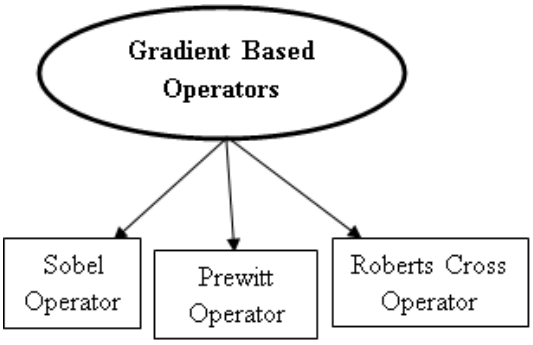
How it Works:

The gradient of an image I(x,y) is calculated as:

Gradient magnitutde: **G(x,y)** = $\sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}$

Gradient direction: $\boldsymbol{\theta(x,y) = tan^{-1}\left(\frac{\partial I/\partial y}{\partial I/\partial x}\right)}$

Popular operators used to approximate the gradient:



- **Sobel Operator:** Sobel operator, or Sobel–Feldman operator or Sobel filter in certain cases, is applied in computer vision and image processing, especially in edge detection algorithms where it generates an image with the edges highlighted. The Sobel operator is a discrete differentiation operator that computes an approximation of the gradient of the image intensity function. It is both differentiation and smoothing, and therefore more resistant to noise than the more primitive operators**.**

  The Sobel–Feldman operator relies on convolving the image with a small, separable, and integer-valued filter in the horizontal and vertical directions and is thus extremely cheap computationally. The gradient approximation it generates is, however, extremely coarse, especially for high-frequency changes in the image.

  Formulation: The operator uses two 3×3 kernels that are convolved with the input image to calculate approximations to the derivatives – a horizontal and a vertical. If A is the input image, and Gx and Gy are two images which at each point have the horizontal and vertical derivative approximations respectively, the calculations are as follows:

$$\mathbf{G}_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * \mathbf{A}$$

$$\mathbf{G}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * \mathbf{A}$$

Where $*$ is the 2-dimensional signal processing convolution operation.

The origin of the operator, Sobel, displays alternative signs for these kernels. It defined the operators as neighbourhood masks (i.e. correlation kernels) and are thus reversed from what is defined here as convolution kernels and assumed the vertical axis rises upwards rather than downwards as in image processing these days, and thus the vertical kernel is inverted [45].

Since the Sobel kernels can be decomposed as the products of an averaging and a differentiation kernel, they compute the gradient with smoothing. For example, $\mathbf{G_x}$ and $\mathbf{G_y}$ can be written as:

$$\mathbf{G}_x = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * \left(\begin{bmatrix} 1 & 0 & -1 \end{bmatrix} * \mathbf{A} \right)$$

$$\mathbf{G}_y = \begin{bmatrix} +1 \\ 0 \\ -1 \end{bmatrix} * \left(\begin{bmatrix} 1 & 2 & 1 \end{bmatrix} * \mathbf{A} \right)$$

The *x*-coordinate is defined here as increasing in the "right" direction, and the *y*-coordinate is defined as increasing in the "down" direction. At each point in the image, the resulting gradient approximations can be combined to give the gradient magnitude, using Pythagorean addition:

$$\mathbf{G} = \sqrt{\mathbf{G}_x{}^2 + \mathbf{G}_y{}^2}$$

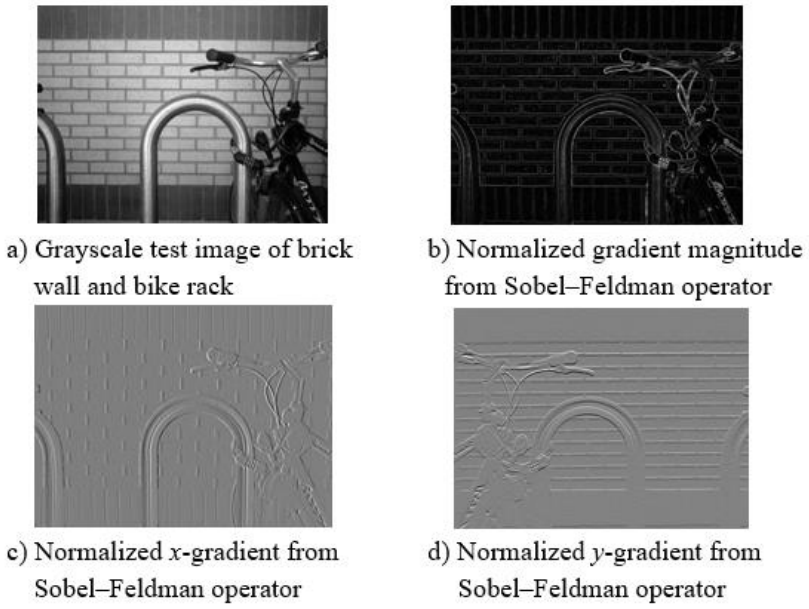
Using this information, we can also calculate the gradient's direction:

$$\mathbf{\Theta} = \operatorname{atan2}(\mathbf{G}_y, \mathbf{G}_x)$$

where, for example, $\mathbf{\Theta}$ is 0 for a vertical edge, which is lighter on the right side.

Example:

The output of the Sobel–Feldman operator is a 2D gradient map at each point. It can be processed and displayed as if it were an image, with the areas of high gradient (the probable edges) as white lines. The next images demonstrate this by displaying the Sobel–Feldman operator calculation on a basic image given in Fig.12.6 [46].



a) Grayscale test image of brick wall and bike rack

b) Normalized gradient magnitude from Sobel–Feldman operator

c) Normalized *x*-gradient from Sobel–Feldman operator

d) Normalized *y*-gradient from Sobel–Feldman operator

Fig. 12.6: Showing the formulation of Sobel Operator

The images below illustrate the change in the direction of the gradient on a grayscale circle. When the sign of $\mathbf{G_x}$ and $\mathbf{G_y}$ are the same the gradient's angle is positive, and negative when different. In the example below, the red and yellow colors on the edge of the circle indicate positive angles, and the blue and cyan colors indicate negative angles. The vertical edges on the left and right sides of the circle have an angle of 0 because there is no local change In $\mathbf{G_y}$. The horizontal edges at the top and bottom sides of the circle have angles of $-\pi/2$ and $\pi/2$, respectively, because there is no local change in $\mathbf{G_x}$. The negative angle for top edge signifies the transition is from a bright to dark region, and the positive angle for the bottom edge signifies a transition from a dark to bright region. All other pixels are marked as black due to no local change in either $\mathbf{G_x}$ or $\mathbf{G_y}$, and thus the angle is not defined. Since the angle is a function of the ratio of $\mathbf{G_y}$ to $\mathbf{G_x}$ pixels with small rates of change can still have a large angle response [46]. As a result, noise can have a large angle response which is typically undesired. When using gradient angle information for image processing applications effort should be made to remove image noise to reduce this false response as shown in Fig.12.7.

Grayscale image of a black Circle with a white background

The direction of the Sobel operator's Gradient

Fig. 12.7: Illustrate the change in the direction of the gradient on a grayscale circle

- **Prewitt Operator:** The Prewitt Operator is an old edge detection technique used in digital image processing to detect edges by estimating the first-order spatial derivatives of an image. Functionally, it seeks to highlight edges in the horizontal and vertical directions to look for steep intensity changes, which will usually be object boundaries or features [47].

  The Prewitt operator offers a less complex alternative to the Sobel operator and is valued for its computational elegance and ease of implementation, especially in contexts where real-time processing is crucial, as shown in Fig.12.8.

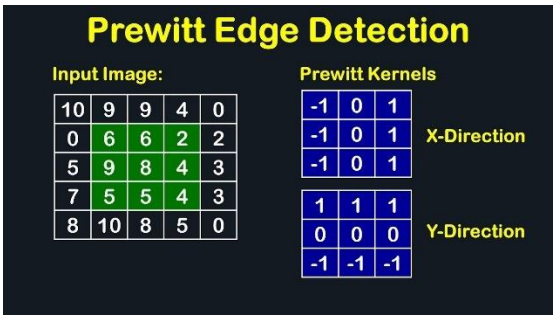


Fig. 12.8: Prewitt Operator

Principle of Operation:

The Prewitt operator works by convolving the image with **two 3×3 kernels (masks)**, each designed to detect changes in **intensity in a specific direction**:

- G_x detects changes along the **horizontal direction** (edges with vertical orientation)
- G_y detects changes along the **vertical direction** (edges with horizontal orientation)

These filters approximate the derivatives $\partial I/\partial x$ and $\partial I/\partial y$, where I represents the image intensity function.

i. Filter Kernels:

The 3×3 convolution masks used in the Prewitt operator are defined as follows:

Horizontal Gradient Kernel (G_x):

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

This kernel emphasizes vertical edges (changes from left to right in the image).

Vertical Gradient Kernel (G_y):

$$G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

This kernel emphasizes horizontal edges (changes from top to bottom).

ii. Computing the Gradient Magnitude:

Once the image is convolved with G_x and G_y, the **gradient magnitude** at each pixel is computed using:

$$G = \sqrt{G_x^2 + G_y^2}$$

For faster computation, especially in hardware-constrained systems, the magnitude is often approximated as:

$$G \approx |G_x| + |G_y|$$

The gradient direction (**θ**) can also be calculated using:

$$\boldsymbol{\theta = tan^{-1}\left(\frac{G_y}{G_x}\right)}$$

This provides the orientation of the edge at each pixel.

Advantages:

1. **Simple and efficient:** It uses coefficient, which are computationally in expensive.
2. **Directional Sensitivity:** It can detect bot horizontal and vertical edges independently.
3. **Useful for Real-Time Systems:** Due to its low computational complexity, it is useful in real-time systems.

Limitations:

1. **Less accurate edge localization:** Compared to Sobel, it may produce less defined edges.
2. **No smoothening component:** Unlike the Sobel operator, the Prewitt filter does not give extra weight to the center row or column, making it **more sensitive to noise**.
3. **Limited Diagonal Detection:** It is less effective at detecting diagonal edges unless additional filters are designed.

Applications:

1. **Edge detection in document scanning:** For locating lines, text boundaries and margins.
2. **Computer Vision Systems:** For detecting object boundaries and feature extraction in real-time.
3. **Medical Image Processing:** Identifying region contours or organ boundaries.
4. **Traffic and Surveillance Cameras:** Basic feature extraction under constrained resources.

Example:

Consider a 3×3 section of a grayscale image:

$$\begin{bmatrix} \mathbf{100} & \mathbf{100} & \mathbf{100} \\ \mathbf{150} & \mathbf{150} & \mathbf{150} \\ \mathbf{200} & \mathbf{200} & \mathbf{200} \end{bmatrix}$$

Applying the $\mathbf{G_y}$ filter (vertical gradient) would yield a strong response because there is a significant change in intensity from top to bottom (from 100 to 200), indicating a horizontal edge.

- **Roberts Cross Operator:** Roberts Cross Operator is one of the oldest and most straightforward edge detection digital image processing algorithms. It is utilized for the first-order derivative approximation of the image intensity function gradient. The process is highly efficient for the detection of diagonal direction (45°) edges and is also computationally simple due to its compact kernel size as shown in Fig.12.9.

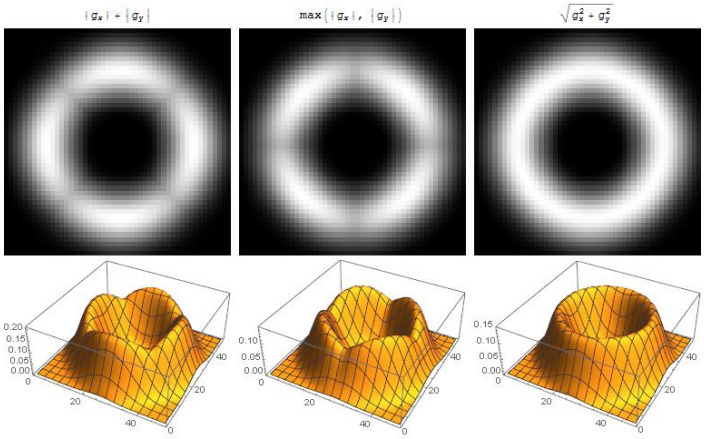


Fig. 12.9: Representing Robert Cross Operator

It was created by Lawrence Roberts in 1963, this operator is still studied for its historical importance and for its ability to detect edges at low computational expense [48].

Working Principle:

The Roberts Cross Operator does edge detection through the computation of the difference between diagonally neighbouring pixel values within a 2×2 image neighbourhood. This small window gives the operator very low overhead and is very fast, suitable for hardware-constrained systems or real-time systems.

Two convolution masks (kernels) are employed to compute gradients in orthogonal diagonal directions:

i. Robert Cross Kernal: Let I(x,y) be the intensity at pixel (x, y). The two gradient approximations are computed using the following 2×2 masks:

Gradient in the X-direction (G_x):

$$G_x = \begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix}$$

Gradient in the y-direction (G_y):

$$G_y = \begin{bmatrix} 0 & +1 \\ -1 & 0 \end{bmatrix}$$

Each kernel is applied by placing the top-left corner of the kernel on a pixel and computing the sum of the products of the overlapping values. These kernels effectively compute the intensity differences across the diagonals.

ii. Gradient Magnitude and Direction:

Once the two directional gradient G_x and G_y are calculated, the magnitude of the gradient at each pixel is determined using:

$$G = \sqrt{G_x^2 + G_y^2}$$

For faster computation, an approximately may be used:

$$G \approx |G_x| + |G_y|$$

The direction (orientation) of the edge can also be calculated as:

$$\theta = tan^{-1}\left(\frac{G_y}{G_x}\right)$$

Applications:

- Edge enhancement in object detection
- Contour extraction in segmentation
- Pre-processing in computer vision pipelines

Advantages:

- Simple and fast
- Effective in highlighting contours and transitions

Disadvantages:

- Sensitive to noise (can be mitigated with smoothing)
- Only detects intensity-based changes, not textures

iv. Morphological processing: Morphological processing is a collection of non-linear image processing operations that work on images in terms of shapes. It is strongest with binary images but can be applied to grayscale images [48]. Morphological operations search the image with a structuring element (a small shape) to extract or suppress specific structures, as shown in Fig.12.10.

Fig. 12.10: Showing Morphological Image Processing

Basic Operations:

- **Dilation**: Expands object boundaries and fills small holes.
- **Erosion**: Shrinks object boundaries and removes small objects.
- **Opening**: Erosion followed by dilation; removes small noise.
- **Closing**: Dilation followed by erosion; fills small holes.

Advanced Operations:

- **Top-hat Transform**: Enhances bright objects on dark backgrounds.
- **Bottom-hat Transform**: Enhances dark objects on bright backgrounds.
- **Morphological Gradient**: Difference between dilation and erosion; emphasizes edges.

Applications:

- Character recognition
- Noise removal in document scans
- Detection of cracks or defects in industrial inspection
- Shape analysis in biomedical imaging

Advantages:

- Good at maintaining geometric integrity
- Effective in noise removal, shape refinement, and structure enhancement

It should be kept in mind that the performance of image enhancement is context-dependent and varies with the application of the image. Thus, the right enhancement method is essential to meet the requirement.

12.1.2 Spatial Domain Methods:

Spatial domain techniques operate directly on pixel values. The transformed image ***g(x,y)*** is obtained by applying an operator ***T*** to the original image,

$$f(x,y): g(x,y) = T[f(x,y)]$$

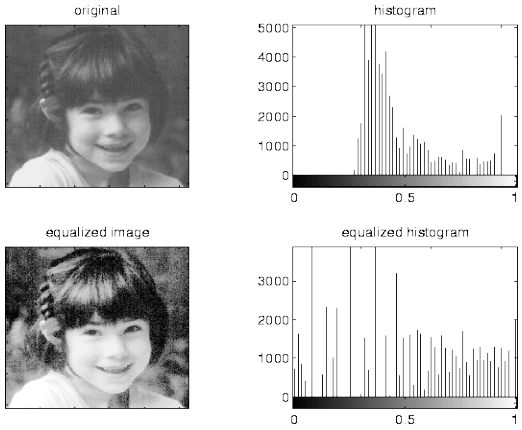


Fig. 12.11: Spatial Domain Methods

These methods are simple to implement and computationally efficient. They include operations like filtering, smoothing, and sharpening. The filtering process often involves applying a template or mask (e.g., 3x3) that defines how surrounding pixels influence the central pixel, as shown in Fig.12.11[42].

12.1.2.1 Smoothing Filters:

Smoothing filters are used to reduce image noise and eliminate small variations in pixel intensity that may not be part of the desired features. These filters are crucial in pre-processing tasks where the objective is to create a more uniform background or prepare the image for further processing, such as segmentation or edge detection.

There are three common types of smoothing filters:

- **Average (Mean) Filter:** This filter replaces each pixel value with the average value of its neighbouring pixels. It is effective for reducing random noise but can blur edges and fine details. The basic form of an average filter uses a uniform kernel where each weight is equal, and the sum of the weights is normalized to one as shown in Fig. 12.12 and the results in Fig.12.13.

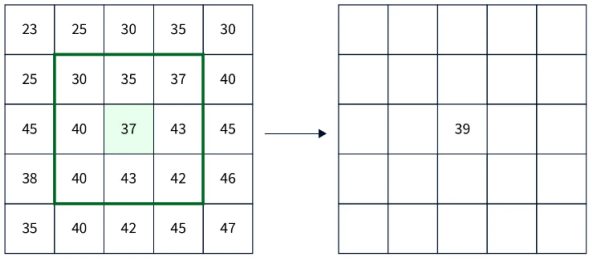


Fig. 12.12: Average Filter

Value at (3,3) = (30+35+37+40+37+43+40+43+42)/9 = 39

Kernel 3X3

$$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} \quad \text{or} \quad 1/9 \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Kernel 5X5

$$1/25 \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Fig. 12.13: Showing Average Filtering 5X5 kernel

- **Gaussian Filter:** This filter uses a kernel with values derived from the Gaussian function. It assigns higher weights to central pixels and progressively lower weights to distant ones. This approach produces a smoother, more natural blurring effect and is more effective at preserving edge information compared to the average filter as shown in Fig.12.14.
  5X5 Gaussian Kernel

$$\frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

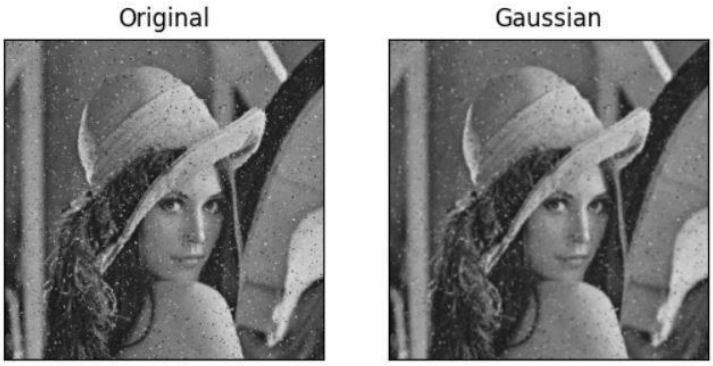


Fig. 12.14: Gaussian Filtering 5X5 Kernel

- **Adaptive Filter:** Unlike fixed-kernel filters, adaptive filters adjust their behaviour based on local image characteristics. They apply stronger smoothing in noisy areas and minimal filtering in areas with significant detail. Adaptive filters use statistical measures like local variance or intensity range to adapt the kernel response, preserving important features while reducing unwanted noise.

Smoothing filters are often applied as a precursor to other image processing operations to ensure consistency and to minimize the impact of noise on subsequent analysis. However, care must be taken to balance noise reduction and detail preservation, as excessive smoothing can lead to information loss.

12.1.2.2 Sharpening Filters:

Sharpening filters are used to enhance the edges and fine details in an image by increasing the contrast between adjacent pixels with differing intensities. Unlike smoothing filters, which aim to suppress high-frequency components (noise), sharpening filters accentuate these components to make image features more pronounced.

Sharpening is particularly useful in applications requiring detailed visual analysis, such as medical imaging, satellite photography, and industrial inspection. It is commonly achieved by applying derivatives to detect rapid changes in intensity [43].

There are two major types of sharpening techniques:

- **First-Order Derivative Filters:** These include gradient-based methods like the Sobel, Prewitt, and Roberts Cross operators, which calculate the rate of change in intensity. They are effective for edge detection by highlighting regions with strong gradients as discussed above in Gradient-Based Methods.
- **Second-Order Derivative Filters:** The most notable is the Laplacian operator, which calculates the second derivative of the image intensity. This operator responds to regions of rapid intensity change and is commonly used in conjunction with Gaussian smoothing to reduce noise sensitivity as given in Fig. 12.15.

The Laplacian is given by: $\boldsymbol{\nabla^2 f(x,y)} = \frac{\partial^r f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$

A common 3x3 Laplacian mask is:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Fig. 12.15: Representing the Laplacian Operator

12.1.2.3 Contrast Stretching:

Contrast stretching is a simple yet effective image enhancement technique used to improve the dynamic range of grayscale images. In many real-world applications, images often suffer from poor contrast due to suboptimal lighting conditions or sensor limitations. This leads to an image whose pixel values are confined within a narrow range of intensities, making visual interpretation difficult [46].

The primary objective of contrast stretching is to remap the intensity values so that the full range of display levels (e.g., 0 to 255 for 8-bit images) is utilized. This increases the overall contrast, making details in both dark and bright regions more visible shown in Fig.12.16.

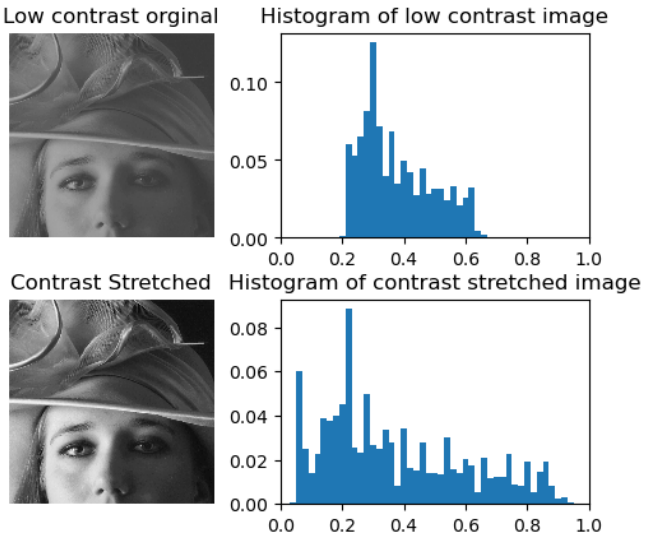


Fig. 12.16: Representing Contrast Stretching

Mathematically, contrast stretching can be expressed as:

$$I'(x,y) = \frac{I(x,y) - I_{\min}}{I_{\max} - I_{\min}} \times (N_{\max} - N_{\min}) + N_{\min}$$

Where:

- ***I(x,y)*** is the original image intensity at location (x,y).
- I_{min}, I_{max} are the minimum and maximum intensity values in the input image.
- N_{min}, N_{max} define the new desired range (e.g., 0 to 255).
- ***I'(x,y)*** is the enhanced pixel intensity.

This transformation stretches the input intensity values linearly across the desired output range. As a result, features that were previously indistinguishable due to low contrast become more prominent.

However, contrast stretching is a global technique, which means that the same transformation is applied across the entire image. While this works well in many cases, it may not be effective if contrast variation is localized. In such cases, local contrast enhancement techniques or adaptive histogram equalization may yield better results [46].

12.1.3 Frequency Domain Methods:

Frequency domain methods enhance images by modifying their frequency content rather than directly manipulating individual pixel values. This is accomplished by transforming the image into the frequency domain using mathematical tools like the **Fourier Transform**, processing the transformed data, and then converting it back into the spatial domain , shown in Fig.12.17.

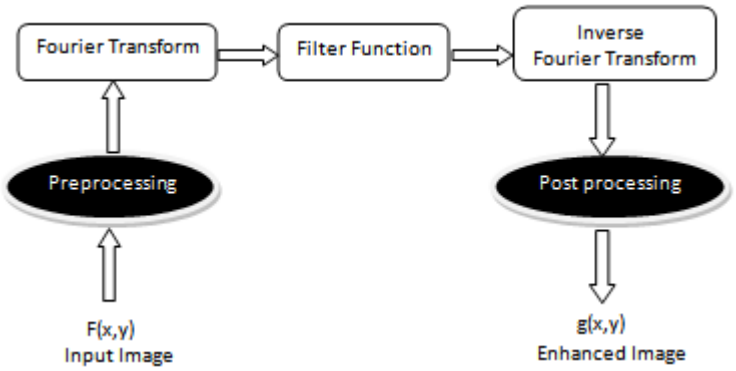


Fig. 12.17: Representing Frequency Domain Methods

12.1.3.1 Concept of Frequency in images:

In image processing, **low-frequency components** represent smooth regions with gradual intensity variations, while **high-frequency components** correspond to sharp edges, fine details, and noise. Frequency-based enhancement allows for selective amplification or suppression of these components based on the desired outcome [47].

12.1.3.2 Process Steps:

The typical process in frequency domain enhancement involves the following steps:

i. Fourier Transform: Convert the spatial domain image into the frequency domain using the Discrete Fourier Transform (DFT) or Fast Fourier Transform (FFT). This yields a spectrum representing the amplitude and phase of sinusoidal components.

ii. Apply Frequency Filters:

- **Low-pass filters** preserve smooth regions and suppress edges and noise.
- **High-pass filters** emphasize edges and fine details by suppressing low-frequency content.

- **Band-pass filters** isolate a specific frequency range.

Common filter designs include Ideal, Butterworth, and Gaussian filters.

iii. Inverse Fourier Transform: Convert the filtered frequency spectrum back to the spatial domain using the Inverse FFT (IFFT), producing the enhanced image.

12.1.3.3 Mathematical Foundation:

Let ***f(x,y)*** represent the original image. The 2D Fourier transform ***F(u,v)*** is given by:

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(ux/M+vy/N)}$$

After applying a filter ***H(u,v)*** , the filtered image in the frequency domain becomes:

$$G(u,v) = H(u,v) \cdot F(u,v)$$

Then, the inverse transform retrieves the spatial image:

$$g(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} G(u,v) e^{j2\pi(ux/M+vy/N)}$$

12.1.3.4 Advantages and Limitations:

Advantages:

- Effective in enhancing periodic patterns and textures.
- Useful for global processing and frequency-specific noise reduction.
- Convolution operations in the spatial domain convert to simple multiplications in the frequency domain, reducing computation.

Limitations:

- Computational overhead, especially for large images.
- May cause artifacts if not implemented carefully (e.g., ringing effects).
- Less intuitive compared to spatial methods.

Frequency domain techniques are particularly valuable in applications such as satellite image enhancement, fingerprint recognition, and optical system analysis, where periodic features and noise filtering are essential [50].

12.1.4 Fuzzy Domain Methods:

Fuzzy domain methods offer an intelligent and human-like approach to image enhancement, particularly effective in environments where pixel characteristics are ambiguous or uncertain. Inspired by fuzzy logic principles, these techniques allow for the gradual classification of pixel intensities instead of rigid binary distinctions, making them ideal for tasks involving complex image textures, noisy data, or low-contrast conditions shown in Fig.12.18.

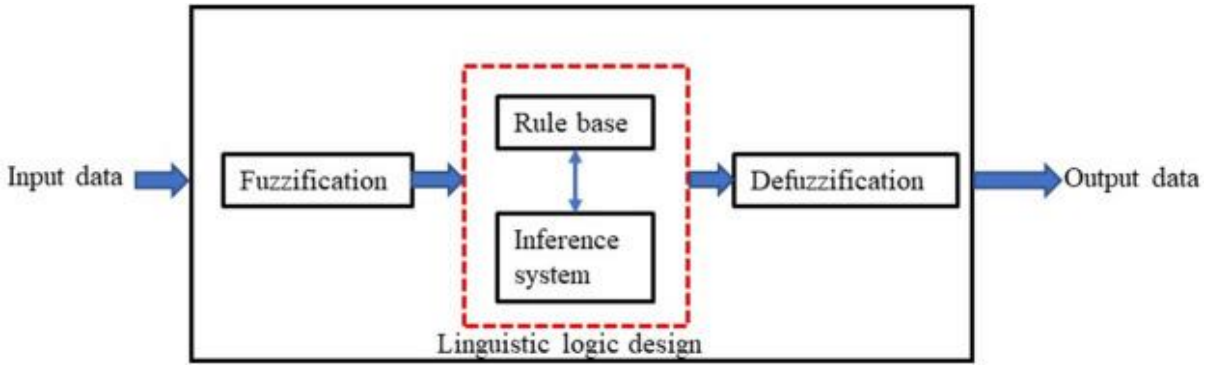


Fig. 12.18: Showing Fuzzy Domain Method

12.1.4.1 Principles of Fuzzy Logic in Image Processing

Fuzzy logic deals with reasoning that is approximate rather than fixed and exact. In the context of image processing, fuzzy logic enables the system to make decisions based on degrees of truth rather than the conventional true or false logic used in classical digital systems. This leads to better adaptability and tolerance to variations in input data [49].

Fuzzy domain enhancement typically involves three main steps:

1. **Fuzzification**
    - In this step, the pixel intensity values of the input image are transformed into fuzzy values using membership functions.
    - Each intensity level is assigned a degree of belonging to different fuzzy sets (e.g., dark, medium, bright).
    - Common membership functions include triangular, trapezoidal, and Gaussian shapes.
2. **Membership Function Modification**
    - Enhancement rules are applied to the fuzzy values to modify the degree of membership.
    - These rules can be based on expert knowledge or adaptive algorithms and aim to improve specific image characteristics such as contrast or brightness.
    - For instance, rules may amplify the membership of brighter pixels in high-intensity fuzzy sets to make bright areas more prominent.
3. **Defuzzification**
    - The adjusted fuzzy values are converted back to crisp pixel intensities.
    - Techniques such as the centroid method or maximum membership principle are used.
    - The resulting image has enhanced features with improved visibility and clarity.

12.1.4.2 Mathematical Representation:

Let $f(x,y)$ be the intensity at pixel (x,y). The fuzzification process maps this intensity to a fuzzy set using a membership function $\mu(f(x,y))$. After enhancement via fuzzy rule application (represented as transformation T), the defuzzified output $g(x,y)$ is obtained:

$$g(x,y) = D[T(\mu(f(x,y)))]$$

Where D represents the defuzzification function.

12.1.4.3 Advantages of Fuzzy Domain Techniques

- **Adaptability**: Can dynamically adjust enhancement based on local image characteristics.
- **Robustness**: Effective in handling images with varying levels of noise or indistinct features.
- **Visual Quality**: Produces images that align more closely with human perception.

12.1.4.4 Limitations

- **Computational Complexity**: Fuzzy systems are generally more resource-intensive than traditional enhancement methods.
- **Parameter Tuning**: Requires careful design and calibration of membership functions and rules.
- **Scalability**: May need adaptation for large-scale or high-resolution image processing tasks.

12.1.4.5 Applications

- **Medical Imaging**: Enhancing soft tissues in MRI or ultrasound images where boundary definitions are subtle.
- **Remote Sensing**: Handling satellite images with mixed terrain categories and fuzzy land-cover boundaries.

- **Underwater Imaging**: Enhancing contrast and feature visibility in low-visibility, noisy environments.

Fuzzy domain methods complement conventional approaches and are increasingly integrated with machine learning systems for intelligent image analysis and interpretation.

12.2 Conclusion:

In conclusion, image enhancement is still the foundation of digital image processing, with broad implications for enhanced visual quality and interpretability of data in many application fields. From diagnostics in medicine to remote sensing, surveillance, underwater imaging, and industrial inspections, improved images support enhanced decision-making and analysis.

The chapter has discussed three primary enhancement areas—spatial, frequency, and fuzzy—each with its strengths and optimal applications:

Spatial domain techniques are simple to use, efficient, and good for real-time processing, but can cause distortion when dealing with complicated images.

Frequency domain techniques offer greater control over certain aspects of an image like textures and patterns, so they are useful for applications where detailed improvement is needed. But they require more computational power and expertise.

Fuzzy domain techniques mimic human thinking and are optimally applicable to situations of uncertainty, small contrast changes, or noisy environments. Their flexibility is paid for with increased computational complexity and the requirement for good rule and function design.

Notably, there is no one technique that is a general-purpose solution. Selection among enhancement methods has to take into consideration the particular image characteristics, the type of noise or artifacts, end-user needs, and application domain.

As imaging systems become more advanced and data volumes increase, hybrid methods that combine several approaches are becoming popular. In addition, AI and ML integration is transforming the domain, allowing systems to make automatic choices between and optimize the enhancement methods based on learned patterns and contextual understanding.

Advancements in the future will probably be directed towards:

i. Context-aware adaptive enhancement driven by AI.
ii. Real-time enhancement for live image streams.
iii. Intelligent noise reduction and feature retention.
iv. Personalized image improvement adapted to personal viewing habits.

In total, image improvement will remain a central facilitator in both academia and business, bridging the gap between raw visual information and useful, actionable knowledge.

Chapter 13

MULTI-POSE GUIDED VIRTUAL TRY-ON SYSTEMS (MPGVTOS)

Gurnoor Singh[1], Sumit Chopra[2] and Gagandeep Singh[3]

[1,2,3]GNA University, Phagwara

13.1 Abstract:

The abstract provides an introduction to the growing significance of Virtual Try-On (VTO) systems in the ever-changing e-commerce environment, more so in the fashion retailing sector. As consumers continue to move towards shopping online, the greatest challenge remains the inability to try on apparel physically, potentially resulting in ambiguity about fit, style, and overall look. Classic VTO solutions try to solve this shortcoming, but they struggle because they are based on fixed poses and little flexibility, necessitating manual adjustments to provide a slightly realistic experience. In an effort to address these limitations, this chapter introduces a new paradigm—Multi-Pose Guided Virtual Try-On System (MPGVTOS). MPGVTOS greatly improves the experience of virtual shopping by providing realistic and dynamic view of garments that self-adapt to varied human poses. This ability not only enhances user interaction but also enhances confidence in buying decisions by better simulating how clothing would look in actual life. For companies, MPGVTOS supports higher conversion rates and lower return rates, which result in higher customer satisfaction and a more sustainable shopping model. In addition, the design of the system is future proof with possibilities of further integration with emerging technologies like virtual reality (VR), footwear visualization, and real-time fabric simulation. Such advancements make MPGVTOS a revolutionary tool in shaping the way people interact with fashion online.

13.2 Introduction:

The introduction establishes the context of the fashion industry's digital transformation, focusing specifically on the role of Virtual Try-On (VTO) systems. As online shopping becomes increasingly popular, there is a rising demand for tools that bridge the gap between digital and physical shopping experiences. VTOs serve this purpose by enabling customers to visualize how clothes might look on them using technologies like computer vision, augmented reality (AR), and machine learning (ML) [52]. However, many of the current VTO systems are limited in that they only work with static images. They cannot adjust dynamically to a user's varying poses, which significantly reduces the realism and appeal of the experience. This shortfall makes it harder for consumers to make confident purchasing decisions and increases the likelihood of returns. To address these limitations, the Multi-Pose Guided Virtual Try-On System (MPGVTOS) is introduced. MPGVTOS brings a new dimension to VTO by incorporating dynamic pose adaptation, allowing garments to be visualized in sync with various human stances and movements. This improvement not only enhances interactivity and realism but also builds greater confidence in online purchases. Furthermore, the integration of powerful AI technologies such as SnapML and MobileNet with AR development platforms like Lens Studio enables the creation of rich, immersive, and efficient virtual try-on environments. The introduction positions MPGVTOS as a cutting-edge solution capable of redefining the online fashion retail experience, as shown in Fig.13.1 [53].

Fig. 13.1: Shows the fashion industry's digital transformation.

13.3 Background and Related Work:

Over the past few years, there has been increasing research interest in the area of Virtual Try-On (VTO), fueled by breakthroughs in computer vision, machine learning, and 3D graphics. As online consumption has become ubiquitous among millions of shoppers, demand for technology to provide accurate and realistic simulations of the experience of fitting clothes has mushroomed enormously. VTO systems seek to address this requirement by allowing customers to see how clothes would look on their bodies themselves through digital interfaces. Researchers have investigated a number of significant technological methods to make these systems more effective. Among the most significant are photorealistic image synthesis, 3D simulation of garments, and garment warping based on deep learning. Photorealistic image synthesis employs neural networks, for example, Generative Adversarial Networks (GANs), to generate high-resolution, realistic images of users in virtual garments. In contrast, 3D garment simulation is centered on computer simulation of fabric behavior and looks, including folding, stretching, and movement with respect to body motion [55]. Deep learning-based warping of garments, however, is done using algorithms to warp garments in exactly the right way for an individual's body shape and stance, resulting in a more natural and engaging visual outcome.

Salient systems like SPG-VTON (Semantic Prediction Guidance Virtual Try-On Network) and AVTON (Arbitrary Virtual Try-On Network) have brought advanced methods that challenge the limits of what VTO can do. SPG-VTON utilizes semantic prediction maps to recognize and segment body parts so that there is better placement and warping of the clothing on users who are posed in numerous different ways [54]. This enables clothing to be rendered onto a body realistically, even when the subject is not facing the camera in a typical frontal position. AVTON builds on this method by adding mechanisms that retain clothing detail and manage the cross-category garments such as tops, pants, or full-body attire. These systems also use geometric matching modules that align clothes to the user's body shape and limb position, which is needed to create realistic try-on in dynamic scenes. Both SPG-VTON and AVTON are significant advancements in multi-pose virtual try-on functionality.

In spite of these advances, there remain some limitations that prevent these systems from reaching widespread commercial adoption. One of the biggest problems is scalability—most models work well in

controlled environments but tend to struggle with the variability and diversity needed for large-scale deployment. Another constraint is hardware compatibility. Most VTO systems are computationally demanding, using high-end machines and powerful GPUs, which makes them incompatible with use on regular smartphones or low-end consumer devices. In addition, such systems usually cannot cope with real-world variability like unreliable lighting conditions, mixed backgrounds, and non-uniform body shapes or poses. These challenges make the virtual try-on experience less reliable and consistent for end users [56].

One of the more viable but challenging solutions within the domain comes from hybrid approaches, including those presented by Van Duc et al. (2023) [53], that merge methods like head swapping and body reshaping. These systems are able to create highly realistic images by altering facial direction to fit the desired pose and remodeling the body to how varying garments would fit. Yet, the computational needs of such models are very high. They require high-end processors and a lot of memory, which restricts their use and feasibility for general consumer use, particularly on mobile devices.

Aside from technical limitations, there are also significant social and ethical factors that influence the use of VTO systems. Privacy is the primary concern since users may be required to expose private images or enable camera feeds for the applications to function optimally. This open concerns regarding data storage, stakeholders with access to data, and whether the proper consent mechanism is present. Secondly, the cultural setting within which VTO technology is applied largely contributes to the acceptance of users. To some cultures, virtual undressing or body scanning may be considered intrusive or inappropriate, which constrains marketability of such systems in some parts [57].

With such problems in mind, there is a pressing need for next-generation VTO solutions that are context-dependent, lightweight, and user-focused. These systems need to be optimized to run efficiently on a broad spectrum of devices, safeguard user data, be culturally sensitive, and provide high levels of interactivity and realism. The new Multi-Pose Guided Virtual Try-On System (MPGVTOS) is intended to address the gap in these issues through a smart combination of AR, ML, and optimal system design[59].

13.4 Evolution and Impact of Virtual Try-On (VTO) in Fashion E-Commerce:

Evolution of Virtual Try-On (VTO) technology in fashion online retailing has been an evolutionary process driven by technological, market forces, and global retailing behavior changes. The early VTO technologies were basic, based on 2D static overlays of clothing on top of users' uploaded images. They were not depth-enabled, lacked dimensional precision, and were not interactive. Customers would typically experience constraints like clothing misfit, improper size estimations, and unrealistic fit, leading to dissatisfaction and continued use of conventional shopping.

The COVID-19 pandemic was perhaps one of the most critical junctures for VTO, as it significantly changed the way consumers shopped. With physical retail stores having to close or restrict access, contactless shopping solutions became a necessity rather than an option. Virtual try-on technologies quickly developed as a hygienic, safe, and interactive substitute for conventional fitting rooms. This triggered mass investment by retailers and technology firms to simulate the experiential touch of in-store buying using digital media [58].

Studies performed throughout the pandemic years emphatically attest to the beneficial impact of VTO on customer behavior and commercial performance. Research has indicated that VTO systems increase customer confidence by providing a better understanding of product fit and fashion. This online interaction provides a psychological feeling of ownership and fulfillment, which makes the customer feel

more confident in their purchasing decisions. Therefore, VTO technologies help to increase the rates of conversion and decrease the rates of product return, overcoming two of the most recurring issues in e-commerce.

Recent VTO platforms are now integrating state-of-the-art technologies like augmented reality (AR), 3D body scanning, deep learning, and computer vision. They facilitate real-time garment simulation, more accurate capturing of user body measurements, and personalized fitting. For example, AR-based technology can overlay 3D garments on a real-time video capture of the user, creating an interactive preview of how the clothing drapes on the body shown in Fig.13.2.

Fig. 13.2: The interactive buttons on top left and right helps user to switch between the clothing with a hand-free experience

Even with such advancements, seamless realism and usability remain challenging. One of the key limitations is pose variability—most systems find it difficult to properly fit the clothing to different body poses, resulting in distortion or unrealistic fitting when the users move. Another area that tends to be underdeveloped is facial integration and personalization, which can affect the perceived level of realism, especially in fashion categories such as eyewear, hats, or upper clothing. Furthermore, true digital capturing of fabric qualities, including texture, drape, stretch, and transparency, still poses a technological challenge given variability in material performance under contrasting conditions of light and motion.

These issues serve to highlight the necessity for ongoing innovation and optimization in VTO systems. Improving these areas will not only enhance virtual try-on experience realism and user satisfaction but will also increase brand credibility, lower logistical costs associated with returns and aid the fashion industry's sustainability objectives by reducing product return waste[59].

13.5 What are Generative Adversarial Networks (GANs)?

Generative Adversarial Networks (GANs) is a type of machine learning models suited for unsupervised learning applications, invented by Ian Goodfellow et al. in 2014. GANs are formed by two neural networks — the Generator (G) and Discriminator (D) — both trained together at once in game-theoretic form. Both the networks race against each other, pushing each other to optimize in the course of time, given in Fig.13.3.

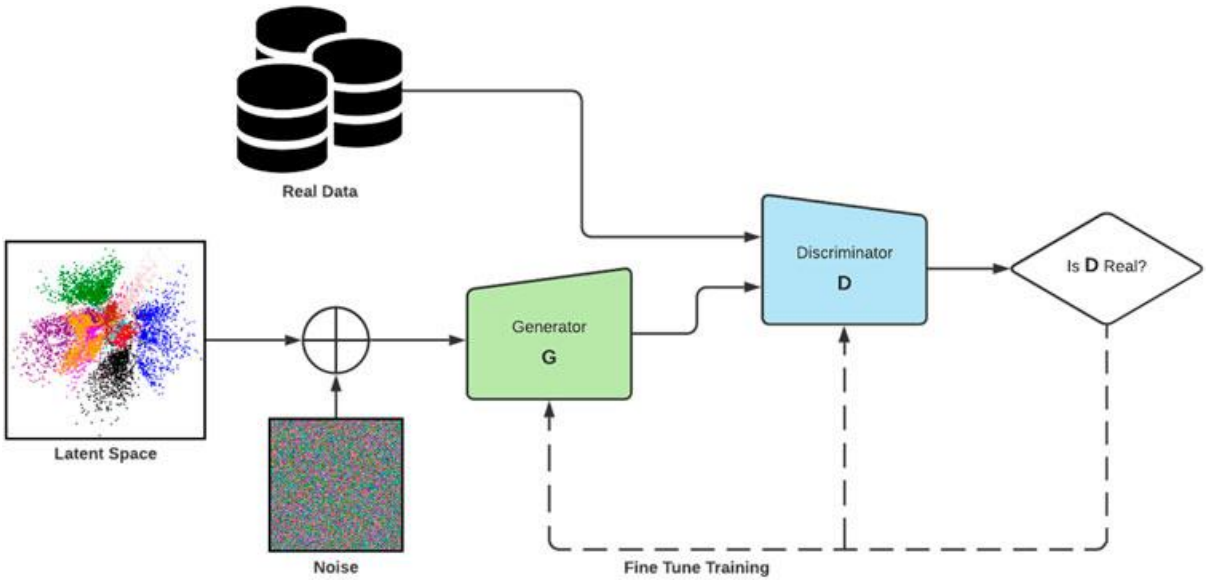


Fig. 13.3: GANs representation

1. **Generator (G)**:
   - Takes a random noise vector z (typically from a Gaussian or uniform distribution) and transforms it into synthetic data resembling the target distribution (e.g., realistic images).
   - Objective: **Fool the Discriminator** into believing that the generated data is real.
2. **Discriminator (D)**:
   - Takes an input and classifies it as **real (from the actual dataset)** or **fake (from the Generator)**.
   - Objective: **Correctly distinguish** between real and generated data.

13.5.1 Training Process (Adversarial Game):

- The Generator tries to **minimize** the probability of the Discriminator identifying its output as fake.
- The Discriminator tries to **maximize** the probability of correctly classifying real and fake data.
- This results in a **minimax optimization problem**:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$

The ideal outcome is when the Generator produces such high-quality outputs that the Discriminator cannot distinguish them from real data, i.e., outputs are indistinguishable from real samples.

13.5.2 Use of GANs in 3D Garment Simulation and Fashion AI:

3D garment simulation involves replicating the **visual appearance, fit, texture, and physics** of clothing in a digital space. Traditional methods rely on physics-based simulation, which can be computationally expensive. GANs offer a **data-driven alternative** to generate high-quality, realistic garments with faster computation given in Fig.13.4 [52].

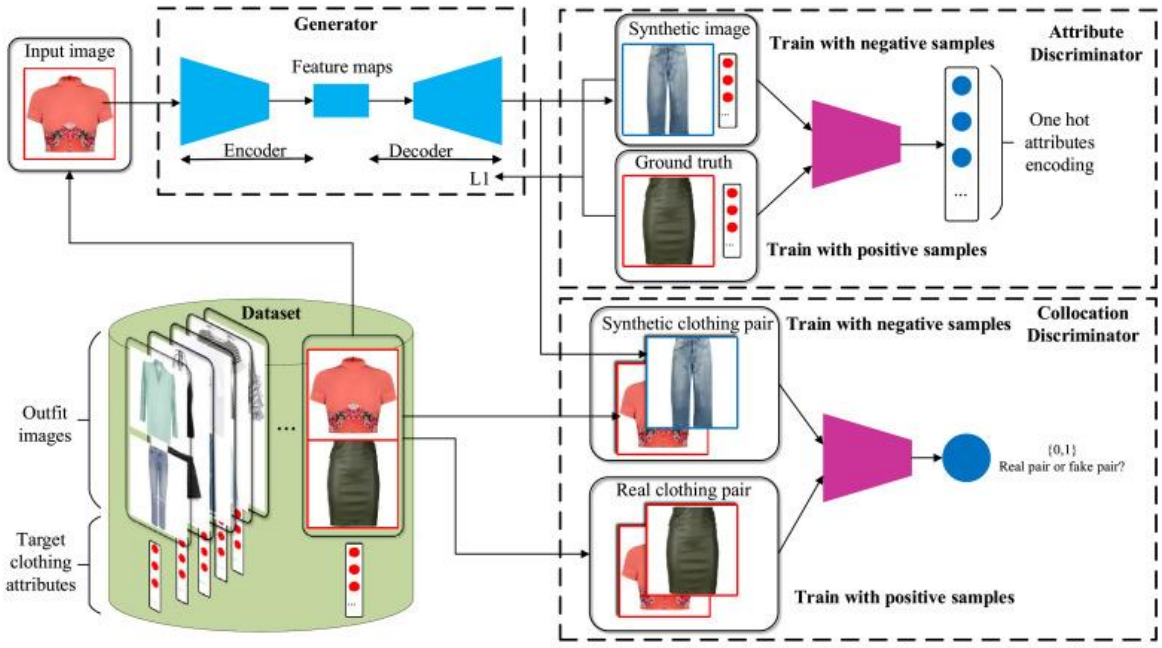


Fig. 13.4: Visualization of GANs in 3D Garment Simulation

Key Applications:

1. **Realistic Garment Generation**:
    - GANs can learn from real garment datasets and generate new garment shapes, designs, and styles in 3D.
    - This allows for **rapid prototyping** and **creative design exploration**.
2. **Virtual Try-On Systems**:
    - GAN-based models can map 2D clothing images onto 3D human models, simulating how the garment would look on a person.
    - They help in **realistic fitting** and **draping**, adjusting the garment according to the body pose, shape, and movement.
3. **Fabric Behavior Prediction**:
    - GANs can model how different fabrics fold, wrinkle, and move in 3D space.
    - This allows designers to **simulate dynamic cloth behavior** without explicitly coding physics rules.
4. **Texture and Style Transfer**:
    - Using techniques like **StyleGAN**, textures (e.g., patterns, prints) from one garment can be transferred to another.
    - GANs help in **high-resolution texture mapping** and **pattern synthesis** for enhanced visual realism.
5. **Data Augmentation**:
    - GANs generate synthetic data to augment existing datasets for training other deep learning models.
    - This is especially useful in scenarios with **limited labeled 3D garment data**.

6. **Physics-Guided GANs**:
    - Recent advances incorporate **physics priors** into GAN training to ensure the generated garments obey realistic motion and material constraints (e.g., stretching, gravity, collisions).
    - These hybrid models balance **visual realism and physical plausibility**.

13.6 System Architecture of MPGVTOS:

The architecture of the Multi-Pose Guided Virtual Try-On System (MPGVTOS) is a multi-level framework that combines machine learning, computer vision, and augmented reality technology to provide an end-to-end interactive try-on experience. The architecture is optimized to provide real-time response, pose flexibility, and photorealistic visual representation of garments even on mobile devices. It has various fundamental components, each of which plays a distinctive role in the virtual try-on process [63].

A. Augmented Reality Framework: Lens Studio Integration:

The Augmented Reality (AR) Infrastructure of the Multi-Pose Guided Virtual Try-On System (MPGVTOS) is constructed on top of Lens Studio, a mature and developer-oriented platform created by Snap Inc. for building rich AR experiences. Lens Studio offers an extensive framework for incorporating machine learning models, 3D content, gesture control, and physics simulations into desktop and mobile apps. Its feature set makes it well-fitted for fashion retail applications that require real-time response and visual realism [64].

Lens Studio is an AR development platform that allows creators to develop and distribute interactive "lenses" — AR filters and experiences — that are mostly designed for use on Snapchat. It has support for both 2D and 3D asset rendering, object tracking, and gesture detection, and it can be used with SnapML, which supports machine learning models to be directly inserted into AR experiences.

For virtual try-on systems, Lens Studio is beneficial because:

i. It facilitates mobile light deployment.
ii. It has integrated real-time rendering to support interactive clothing visualization.
iii. It provides a straightforward scripting platform via JavaScript and graphical node-based programming.

Lens Studio provides real-time tracking and rendering capabilities, making it suitable for mobile AR applications where responsiveness and realism are critical.

B. Machine Learning Integration: SnapML and ONNX Models:

The machine learning that drives MPGVTOS is fueled by SnapML (Snap Machine Learning), which allows for the embedding of trained ML models into the AR world. The framework accommodates model types such as ONNX (Open Neural Network Exchange), TensorFlow Lite, and Core ML, providing flexibility and platform portability.

Some of the main functionalities driven by machine learning are:

i. **Object detection (through CenterNet):** Finds the user's body, limbs, and face in the frame for precise garment alignment.
ii. **Pose estimation:** Recognizes user pose to allow dynamic adjustment of clothing.
iii. **Facial recognition (through Viola-Jones):** Provides garment fit and continuity, particularly for accessories or upper-body garments.
iv. **Gesture classification:** Recognizes and interprets hand gestures for user interaction.

These models are trained and optimized for lightweight performance to provide seamless operation on mobile devices.

C. Model Development Pipeline:

The training of ML models for MPGVTOS is done through a systematic training pipeline:

i. Dataset Preparation:

1. A portion of the COCO dataset is utilized, with particular emphasis on human figures, poses, and types of garments.
2. Data augmentation methods (scaling, rotation, flipping) enhance robustness.

ii. Model Training:

1. MobileNet v2 is used as the backbone neural network for feature extraction because it is efficient and accurate for mobile use.
2. CenterNet architecture is employed for center point detection of objects and size and pose estimation from central points, minimizing computation while preserving accuracy.

iii. Model Export and Conversion:

1. Trained models are exported into ONNX format for SnapML compatibility.
2. Models are then imported into Lens Studio and associated with the AR interaction logic.

D. 3D Garment Design and Simulation:

Clothing assets are created in Blender, an open-source powerful 3D modeling tool. The garments are:

1. Sculpted to fit real-world measurements and fabric behaviors.
2. Textured with realistic fabric patterns and colors.
3. Exported in FBX format, which is AR deployable using Lens Studio.

Advanced physics-based parameters, like gravity, motion dynamics, and collision matrices, are set up to create realistic draping and garment response to body movement.

E. Interaction Design and Gesture Control:

For better usability, MPGVTOS supports hands-free interaction means:

1. Hand-tracking modules recognize when users point towards AR interface buttons.
2. Dwell time detection (i.e., holding the hand over the button for 2–3 seconds) is utilized as a trigger to toggle through outfits or accessories.
3. This design avoids touch-based inputs, making it more accessible and immersive.

F. Real-Time Rendering and Feedback Loop:

The system processes frames in **14–17 milliseconds**, enabling real-time interaction even on mid-range smartphones. The performance loop includes:

1. **Image capture and analysis**.
2. **Pose and face detection**.
3. **Garment mesh alignment**.
4. **Visual rendering and user feedback**.

This loop is continuously updated to ensure that the garments remain anchored correctly to the user's movements and adapt to pose changes fluidly.

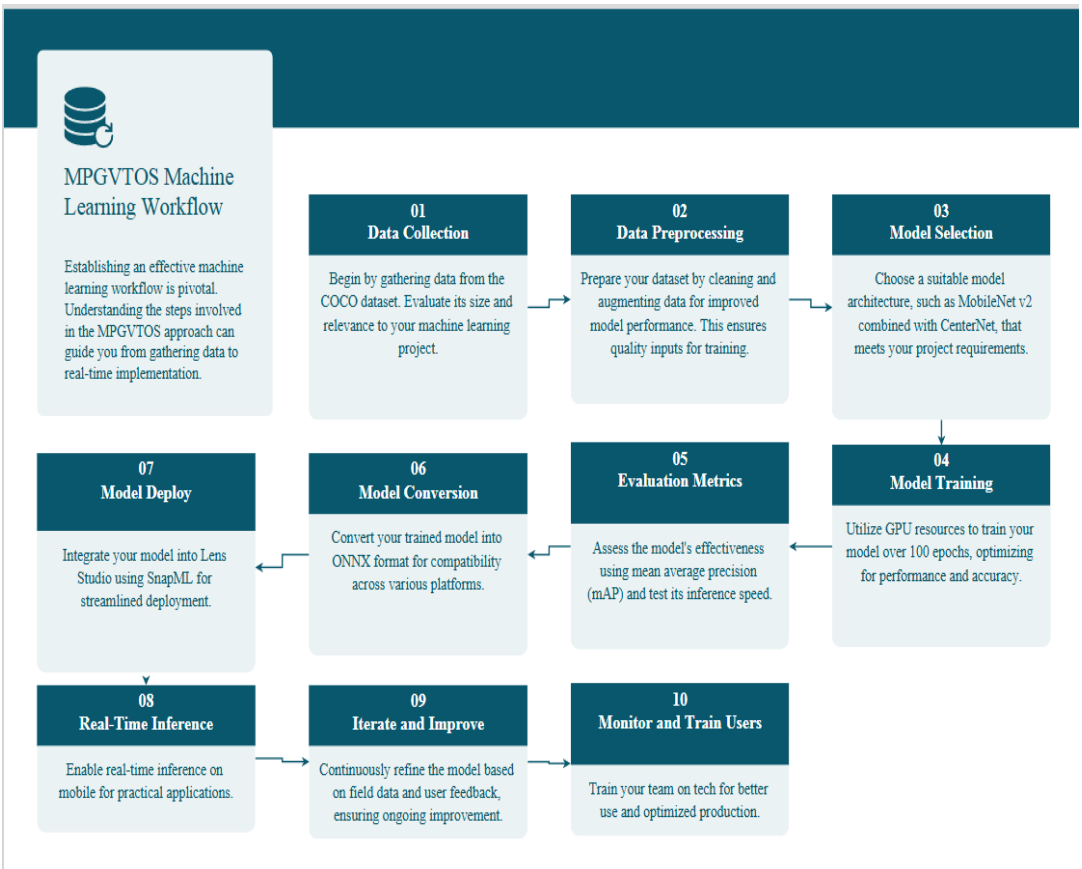
13.7 3D Garment Modeling:

1. **Blender for 3D Modeling:** MPGVTOS uses Blender, a powerful open-source 3D creation tool, to design digital garments. Each item of clothing is modeled from scratch using polygonal meshes

that define the shape, volume, and structure of the garment. This allows precise control over garment dimensions and tailoring [66].

2. **Texturing and Shading:** After the basic structure is created, designers apply realistic fabric textures and colors. UV mapping is used to accurately project 2D textures onto the 3D surface. Blender's shader nodes help simulate material properties such as glossiness, transparency, and softness, making fabrics look authentic (e.g., denim vs. silk).
3. **Rigging for Dynamic Fit:** Rigging involves binding the 3D garment to a skeletal structure (armature) so that it moves naturally with the user's body. Weight painting is used to define how different garment sections respond to joint movements, allowing for natural stretching and bending [67].
4. **Physics-Based Simulation:** Cloth physics are configured to replicate real-world behavior under gravity and motion. These include collision detection with the user's body, draping effects, and interaction between multiple layers of clothing, enhancing realism.
5. **Export to FBX Format:** The final 3D models are exported in the FBX format, which retains all geometry, textures, and rigging information. This format is compatible with AR platforms like Lens Studio, where the garments are rendered in real time on users during the virtual try-on process.

13.8 Machine Learning Workflow:



1. **Dataset Preparation with COCO:** The system leverages a curated subset of the COCO (Common Objects in Context) dataset focused on human figures and wearable items. This rich dataset provides labeled images that help the model learn to identify human anatomy, poses, and clothing types under varied conditions.

2. **Model Architecture: MobileNet v2 + CenterNet:** The core ML architecture combines MobileNet v2 — known for its speed and efficiency on mobile devices — with CenterNet, a pose estimation framework that identifies object centers instead of bounding boxes. This combination provides accurate, low-latency detection of user position and movements.
3. **Training Strategy:** Models are trained using augmented data — including rotated, scaled, and flipped images — to improve generalization and resilience to real-world scenarios. The training process involves multiple epochs and uses GPU acceleration to optimize performance.
4. **Model Conversion to ONNX:** Once trained, models are converted to the ONNX (Open Neural Network Exchange) format. This ensures interoperability with various platforms and seamless integration into **SnapML**, the machine learning engine within Lens Studio.
5. **Real-Time Mobile Inference:** The optimized models support real-time inference, allowing the system to process video frames in 14–17 milliseconds on devices such as smartphones. This enables the garments to respond immediately to user movements, enhancing interactivity and realism in the virtual try-on experience.

13.9 Limitations and Challenges:

Despite the impressive capabilities and innovations introduced by the Multi-Pose Guided Virtual Try-On System (MPGVTOS), several technical, data-driven, and user-experience-related challenges persist. Addressing these limitations is essential to elevate the system to a more scalable, inclusive, and commercially viable solution for the fashion e-commerce industry [69].

1. Self-Occlusions and Misalignment:

Key Issue: One of the most persistent technical challenges in virtual try-on systems is managing self-occlusion—a scenario where parts of the user's body block other parts from the camera's view (e.g., crossed arms, bent knees).

Explanation: In such cases, the system struggles to correctly position the garment, leading to unrealistic warping, misfitting, or portions of the garment appearing in the wrong spatial position. This problem becomes more pronounced in dynamic poses or when users are interacting with the interface using gestures. Accurate pose estimation and garment fitting in such scenarios require advanced depth sensing or multi-view modeling, which are computationally expensive and challenging to implement in real-time mobile systems.

2. Limited Dataset Diversity and Scalability:

Key Issue: The current system is trained on a subset of the COCO dataset, which, although comprehensive, may not represent the full variability of real-world clothing styles, human body types, or diverse environmental conditions.

Explanation: The lack of cultural, ethnic, and garment-specific variation in the dataset limits the model's ability to generalize across users with different physiques, clothing preferences, and backgrounds. This directly affects the inclusivity and global adaptability of MPGVTOS. Moreover, extending the training dataset to include multi-layer garments, ethnic wear, oversized apparel, and accessories would require not only large volumes of labeled data but also extensive computational resources and time to retrain the models.

3. High Computational Demands:

Key Issue: Though the models have been optimized for mobile devices, high-fidelity 3D rendering, real-time tracking, and ML inference still demand significant processing power and memory.

Explanation: Many budget or older smartphones may struggle with rendering high-resolution clothing models or performing live pose estimation. The inclusion of advanced features like fabric dynamics, shading effects, and real-time physics simulations can further strain device capabilities, resulting in slower frame rates or thermal throttling. This makes wide-scale deployment across various device types a challenge, unless further model compression or cloud-based computation is implemented.

4. Limited Realism in Fabric Behavior:

Key Issue: While the garments are visually realistic, simulating fabric behavior under motion, gravity, and environmental effects (e.g., wind or collisions) still lacks complete accuracy.

Explanation: Different materials—such as silk, denim, leather, or lace—react uniquely to movement and external forces. Accurately modeling these responses in real time requires physics-based cloth simulation engines that factor in fabric weight, elasticity, and resistance. Most current VTO systems, including MPGVTOS, use simplified physics approximations which can compromise realism, especially when garments are layered or in motion.

5. Gesture-Based UI Limitations:

Key Issue: Although hands-free gesture controls enhance interactivity, they are often sensitive to environmental factors such as lighting conditions, background noise, or camera quality.

Explanation: In low-light environments or when using low-resolution front-facing cameras, hand gestures may not be accurately detected. Additionally, unintentional movements or delays in gesture recognition can lead to poor user experience or frustration. Robust gesture recognition requires sophisticated algorithms that account for spatial and temporal data, potentially increasing the system's complexity and resource usage.

6. Privacy and Ethical Concerns:

Key Issue: The use of facial and body tracking technologies raises privacy concerns among users.

Explanation: Virtual try-on systems that involve real-time video processing and body detection often require access to sensitive visual data. Users may be hesitant to grant camera access or uncomfortable with how their visual data is stored or processed, particularly if the system lacks transparent data handling policies. Additionally, developers must consider the ethical implications of body image representation, ensuring that the system does not inadvertently promote unrealistic beauty standards or bias toward certain body types.

7. Lack of VR Integration:

Key Issue: MPGVTOS currently operates within a 2D or AR interface without integration into fully immersive virtual reality (VR) environments.

Explanation: Although AR provides an interactive experience, VR would offer a more immersive and lifelike try-on experience, enabling users to walk around, inspect garments from multiple angles, and even interact with virtual store environments. However, VR implementation introduces its own challenges, including the need for specialized hardware, motion tracking, and higher system requirements.

13.10 Conclusion:

The advent of Multi-Pose Guided Virtual Try-On Systems (MPGVTOS) represents a breakthrough in the confluence of fashion technology, augmented reality, and artificial intelligence. With e-commerce reshaping the retail industry, consumers increasingly crave experiences that span the divide between physical and digital shopping. MPGVTOS answers this demand directly by allowing users to interact with

garments within a virtual environment in real-time, providing a natural, dynamic, and interactive try-on experience that was heretofore impossible with standard 2D or static VTO systems [70].

This chapter has provided an in-depth introduction to MPGVTOS—its motivations, structure, underlying technologies, and applications. Through the combination of 3D garment modeling via Blender, machine learning-based pose recognition using MobileNet v2 and CenterNet, and real-time AR rendering via Lens Studio, MPGVTOS provides a seamless, highly responsive interface that dynamically adjusts to varied human postures and offers a realistic simulation of fabric behavior and garment fit. The system's use of gesture-based controls and real-time feedback mechanisms further enhances user engagement, making it a promising tool for both online retailers and fashion consumers.

Notably, the system does not merely offer visual gratification. MPGVTOS remedies some of the ongoing issues of online clothing purchases—like size misrepresentation, high return rates, and customer reluctance due to the absence of haptic feedback. By presenting a try-on solution that is both visually engaging and functionally responsive, the system can potentially enhance customer confidence, enhance conversion rates, decrease product returns, and help achieve more sustainable retailing practices [71].

Still, there are challenges to implementing MPGVTOS. These include self-occlusion, computational complexity, limited datasets, and realism of fabric simulation that need to be overcome so that wider use and user acceptance are enabled. Also critical to developing virtual try-on systems as equitable and universally acceptable are concerns regarding privacy, accessibility, and cultural acceptability. In addition, the system's present AR-based deployment, though effective, would be substantially improved upon by integrating with full-immersion virtual reality (VR) environments to create an enriched, more immersive shopping experience.

In the future, the potential of VTO systems such as MPGVTOS depends on further innovation and cross-disciplinary work. By integrating state-of-the-art computer vision, deep learning, and real-time AR/VR technologies with human-centered design, researchers and developers can design next-generation virtual shopping platforms that are not only technologically superior but also inclusive, ethical, and strongly aligned with consumer requirements.

In summary, MPGVTOS is not just a technical product; it is a window into the future of online fashion shopping, where technology facilitates creativity, ease, and confidence in the buying process. With the changing industry, technologies such as MPGVTOS will become pivotal factors in driving the way fashion is experienced, personalized, and consumed in a digitally enabled world.

REFERENCES:

1. https://www.tableau.com/visualization/what-is-data-visualization.
2. https://www.geeksforgeeks.org/data-visualization-and-its-importance/.
3. Aigner, W., Miksch, S., Schumann, H., & Tominski, C. (2011). Visualization of Time-Oriented Data. Springer Science & Business Media.
4. Mack C, Su Z, Westreich D.Rockville (MD): Agency for Healthcare Research and Quality (US); 2018 Feb.
5. Bertin, J. (1983). Semiology of Graphics: Diagrams, Networks, Maps. University of Wisconsin Press.
6. Bremer, N., & Wu, S. (2018). Data Sketches. CRC Press.
7. Cairo, A. (2016). The Truthful Art: Data, Charts, and Maps for Communication. New Riders.
8. Cairo, A. (2019). How Charts Lie: Getting Smarter about Visual Information. W. W. Norton & Company.
9. Camoes, J. (2016). Data at Work: Best practices for creating effective charts and information graphics in Microsoft Excel. New Riders.
10. Cederbom, C. (2023). Data Analysis and Visualization: A Complete Guide - 2023 Edition. The Art of Service.
11. Murray, S. (2017). Interactive Data Visualization for the Web (2nd ed.). O'Reilly Media.
12. Cleveland, W. S. (1994). The Elements of Graphing Data. Hobart Press.
13. Dykes, B. (2016). Data Storytelling for Data Management: A Data Quality Approach. Technics Publications.
14. Evergreen, S. D. H. (2017). Effective Data Visualization: The Right Chart for the Right Data. SAGE Publications.
15. Few, S. (2017). Data Visualization for Success: A Step-by-Step Guide to Making Effective Data Visuals. Analytics Press.
16. Healy, K. (2018). Data Visualization: A Practical Introduction. Princeton University Press.
17. Heer, J., Bostock, M., & Ogievetsky, V. (2010). A Tour Through the Visualization Zoo. Communications of the ACM, 53(6), 59-67.
18. Kirk, A. (2016). Data Visualisation: A Handbook for Data Driven Design. SAGE Publications.
19. Kirk, A. (2019). Data Visualisation: A Handbook for Data Driven Design (2nd ed.). SAGE Publications.
20. Munzner, T. (2014). Visualization Analysis and Design. CRC Press.
21. Nussbaumer Knaflic, C. (2015). Storytelling with Data: A Data Visualization Guide for Business Professionals. Wiley.
22. Nussbaumer Knaflic, C. (2019). Storytelling with Data: Let's Practice!. Wiley.
23. O'Dwyer, A. (2021). Data Visualization in Excel: A Guide for Beginners, Intermediates, and Professionals. Routledge.
24. Rahlf, T. (2019). Data Visualisation with R: 111 Examples. Springer.
25. Schwabish, J. (2021). Better Data Visualizations: A Guide for Scholars, Researchers, and Wonks. Columbia University Press.

26. Sedlmair, M., Meyer, M., & Munzner, T. (2016). Design Study Methodology: Reflections from the Trenches and the Stacks. IEEE Transactions on Visualization and Computer Graphics, 18(12), 2431-2440.
27. https://filippovalle.medium.com/the-principle-of-proportional-ink-c8c528d12d4d
28. Tufte, E. R. (2001). The Visual Display of Quantitative Information. Graphics Press.
29. Tufte, E. R. (2020). Seeing with Fresh Eyes: Meaning, Space, Data, Truth. Graphics Press.
30. Ware, C. (2020). Information Visualization: Perception for Design (4th ed.). Morgan Kaufmann.
31. Wickham, H. (2016). ggplot2: Elegant Graphics for Data Analysis. Springer.
32. Wilke, C. O. (2019). Fundamentals of Data Visualization: A Primer on Making Informative and Compelling Figures. O'Reilly Media.
33. Wexler, S., Shaffer, J., & Cotgreave, A. (2017). The Big Book of Dashboards: Visualizing Your Data Using Real-World Business Scenarios. Wiley.
34. Yau, N. (2013). Data Points: Visualization That Means Something. Wiley.
35. Yau, N. (2018). Visualize This: The FlowingData Guide to Design, Visualization, and Statistics. Wiley.
36. Zelazny, G. (2015). The Say It With Charts Complete Toolkit. McGraw-Hill Education.
37. Zelazny, G. (2015). Say It With Charts: The Executive's Guide to Visual Communication. McGraw-Hill Education.
38. Zelazny, G. (2015). Say It With Presentations: How to Design and Deliver Successful Business Presentations. McGraw-Hill Education.
39. Tableau. (n.d.). Tableau Public Documentation. Retrieved from https://public.tableau.com/en-us/s/
40. Microsoft Power BI. (n.d.). Power BI Documentation. Retrieved from https://docs.microsoft.com/en-us/power-bi/
41. Google Flu Trends. (n.d.). Tracking Flu Outbreaks Using Data Visualization. Retrieved from https://www.google.org/flutrends/
42. The New York Times. (2020). Election Data Visualization and Trends. Retrieved from https://www.nytimes.com/interactive/
43. Aaron J, Chew TL (2021) A guide to accurate reporting in digital image processing – Can anyone reproduce your quantitative analysis? J Cell Sci 134:. https://doi.org/10.1242/jcs.254151
44. Bahadır B, Atik OŞ, Kanatlı U, Sarıkaya B (2023) A brief introduction to medical image processing, designing and 3D printing for orthopedic surgeons. Jt Dis Relat Surg 34:451–454. https://doi.org/10.52312/jdrs.2023.57912
45. Gandomi AH (2021) Cyberstalking Victimization Model Using Criminological Theory : A Systematic Literature Review , Taxonomies , Applications , Tools , and Validations
46. Kaur H, Sohi N (2017) A Study for Applications of Histogram in Image Enhancement. Int J Eng Sci 06:59–63. https://doi.org/10.9790/1813-0606015963
47. Ablin R, Sulochana CH, Prabin G (2020) An investigation in satellite images based on image enhancement techniques. Eur J Remote Sens 53:86–94. https://doi.org/10.1080/22797254.2019.1673216
48. Kisekka I, Peddinti SR, Savchik P, et al (2024) Multisite evaluation of microtensiometer and osmotic cell stem water potential sensors in almond orchards. Comput Electron Agric 227:109547. https://doi.org/10.1016/j.compag.2024.109547

49. Zhuang L, Guan Y (2019) Image enhancement using modified histogram and log-exp transformation. Symmetry (Basel) 11:1–17. https://doi.org/10.3390/SYM11081062
50. Browarska N, Kawala-Sterniuk A, Zygarlicki J, et al (2021) Comparison of smoothing filters' influence on quality of data recorded with the emotiv epoc flex brain–computer interface headset during audio stimulation. Brain Sci 11:1–23. https://doi.org/10.3390/brainsci11010098
51. Siddiqi MH, Alhwaiti Y (2022) Signal-to-Noise Ratio Comparison of Several Filters against Phantom Image. J Healthc Eng 2022:. https://doi.org/10.1155/2022/4724342.
52. Batool, R. and Mou, J. (2024) 'A systematic literature review and analysis of try-on technology: Virtual fitting rooms', Data and Information Management, 8(2), p. 100060. doi:10.1016/J.DIM.2023.100060.
53. Van Duc, T. et al. (2023) 'A Hybrid Photorealistic Architecture Based on Generating Facial Features and Body Reshaping for Virtual Try-on Applications', Mendel, 29(2), pp. 97–110. doi:10.13164/mendel.2023.2.097.
54. Feng, Y. and Xie, Q. (2019) 'Privacy Concerns, Perceived Intrusiveness, and Privacy Controls: An Analysis of Virtual Try-On Apps', Journal of Interactive Advertising, 19(1), pp. 43–57. doi:10.1080/15252019.2018.1521317.
55. Ghodhbani, H. et al. (2022) You can try without visiting : a comprehensive survey on virtually try-on outfits. Multimedia Tools and Applications.
56. Goel, P., Mahadevan, K. and Punjani, K.K. (2023) 'Augmented and virtual reality in apparel industry: a bibliometric review and future research agenda', Foresight, 25(2), pp. 167–184. doi:10.1108/FS-10-2021-0202/FULL/XML.
57. Hu, B. et al. (2022) 'SPG-VTON: Semantic Prediction Guidance for Multi-Pose Virtual Try-on', IEEE Transactions on Multimedia, 24(8), pp. 1233–1246. doi:10.1109/TMM.2022.3143712.
58. Hwangbo, H. et al. (2020) 'Effects of 3D Virtual "Try-On" on Online Sales and Customers' Purchasing Experiences', IEEE Access, 8, pp. 189479–189489. doi:10.1109/ACCESS.2020.3023040.
59. Islam, T. et al. (2024) 'Deep Learning in Virtual Try-On: A Comprehensive Survey', IEEE Access, 12(February), pp. 29475–29502. doi:10.1109/ACCESS.2024.3368612.
60. Lagė, A. et al. (2020) 'Comparative study of real and virtual garments appearance and distance ease', Medziagotyra, 26(2), pp. 233–239. doi:10.5755/j01.ms.26.2.22162.
61. Lavoye, V., Tarkiainen, A., et al. (2023) 'More than skin-deep: The influence of presence dimensions on purchase intentions in augmented reality shopping', Journal of Business Research, 169(August). doi:10.1016/j.jbusres.2023.114247.
62. Lavoye, V., Sipilä, J., et al. (2023) 'The emperor's new clothes: self-explorative engagement in virtual try-on service experiences positively impacts brand outcomes', Journal of Services Marketing, 37(10), pp. 1–21. doi:10.1108/JSM-04-2022-0137.
63. Lee, H. and Xu, Y. (2020) 'Classification of virtual fitting room technologies in the fashion industry: from the perspective of consumer experience', International Journal of Fashion Design, Technology and Education, 13(1), pp. 1–10. doi:10.1080/17543266.2019.1657505.
64. Liu, Yu et al. (2024) 'Arbitrary Virtual Try-on Network: Characteristics Preservation and Tradeoff between Body and Clothing', ACM Transactions on Multimedia Computing, Communications and Applications, 20(5), pp. 1–12. doi:10.1145/3636426.

65. Ren, B. et al. (2023) 'Cloth Interactive Transformer for Virtual Try-On', ACM Transactions on Multimedia Computing, Communications and Applications, 20(4). doi:10.1145/3617374.
66. Savadatti, M.B. et al. (2022) 'Theoretical Analysis of Viola-Jones Algorithm Based Image and Live-Feed Facial Recognition', Proceedings - IEEE International Conference on Advances in Computing, Communication and Applied Informatics, ACCAI 2022 [Preprint]. doi:10.1109/ACCAI53970.2022.9752590.
67. Zhang, T. (2019) 'The role of virtual try-on technology in online purchase decision from consumers ' aspect Internet Research Article information :', (March). doi:10.1108/IntR-12-2017-0540.
68. Aamir, M. et al. (2021) 'Efficiently Processing Spatial and Keyword Queries in Indoor Venues'. doi:10.1109/TKDE.2020.2964206.
69. Hu, B. et al. (2022) 'SPG-VTON: Semantic Prediction Guidance for Multi-Pose Virtual Try-on', IEEE Transactions on Multimedia, 24(8), pp. 1233–1246. doi:10.1109/TMM.2022.3143712.
70. Lavoye, V. et al. (2023) 'The emperor's new clothes: self-explorative engagement in virtual try-on service experiences positively impacts brand outcomes', Journal of Services Marketing, 37(10), pp. 1–21. doi:10.1108/JSM-04-2022-0137.
71. Liu, Yu et al. (2024) 'Arbitrary Virtual Try-on Network: Characteristics Preservation and Tradeoff between Body and Clothing', ACM Transactions on Multimedia Computing, Communications and Applications, 20(5), pp. 1–12. doi:10.1145/3636426.

Fundamentals of Data Handling and Visualization

ISBN: 978-93-48620-52-1

About Editors

Dr. Sumit Chopra is working as an Associate Professor in GNA University, Phagwara. He has over 18 years of teaching experience and has published research papers in various international conferences and peer reviewed journals. He is reviewer of various research journals and delivered expert talks in different institutes. He has served as NBA Coordinator for CSE Department of GNA University and got B. Tech (CSE) program accredited for 3 years. With a keen interest in Data Science and artificial Intelligence, he is actively involved in academic research and technical writing.

Navjot Kaur Basra is currently pursuing a Master of Technology (M.Tech) in Computer science and engineering at GNA University. With a keen interest in Data Science and artificial Intelligence she is actively involved in academic research and technical writing. As an editor of this book, Navjot Kaur Basra has contributed by reviewing and refining content to enhance its academic value and readability, aiming to support fellow students and researchers in the field.

Simran is currently pursuing her Master of Technology (M.Tech) in Computer Science and Engineering from GNA University. Passionate about Data Science and Artificial Intelligence, she is deeply engaged in research activities and technical content development. As an editor of this book, Simran has played a vital role in evaluating and polishing the chapters to improve clarity, quality, and academic relevance. Her contributions aim to benefit students, scholars, and researchers working in emerging areas of computer science.

Debjit is cyber security professional with 2+ years of experience in Threat intelligence, Digital forensics, and Incident Handling/Response and is currently pursuing M.Tech in Computer Science and Engineering with Specialization of Cyber Security & Digital Forensics at GNA University. He holds certifications like CEH, CHFI, ECIH and CTIA and has worked with leading security firms and delivered sessions at NSG, Indian Army, and Indian Navy.

Bhumi Publishing

ISBN 978-93-48620-52-1

9 789348 620521