Lesson 1

Need To Visualise Data

Why to Visualise data?

I. 65% of humans are better Visual Learners.Vision is a powerful driver for learning and receiving information.

II. We retain 80% of information seen as compared to 20% of information read.

III. Visual perception is the fastest amongst all senses at 13 ms.

IV. We will forget textual information easily as compared to visual information.

Helps In ?

1. Decision Making
2. Uncovering Patterns and Trends
3. Presenting Arguments/Information and/or Telling a story.

There are cases when the descriptive statistics for different sets of data are the same but that does not mean that the dataset is the same. There could still be fundamental differences in the Nature of the Dataset and plotpoints.

When we plot the datasets, we can see they have different distributions.

Reasons to create visualization can be :

• Visualization compresses large volume of data into easy-to-understand visuals
• Enables business intelligence
• Effective for exploration of the data as well communication of insights from the data
• Discovering answers to questions from the underlying data
• Enabling data aid decision making
• Understanding the data in a context
• Finding hidden patterns in the data
• Presenting an argument or telling a story
• Inspiring or persuading with data story

How Visualization Helps Communicate Complex Information

It Helps in Communicating with Clarity. It Simplifies complex data for all audiences using pre-attentive processing , Reduces cognitive load with intuitive designs, Engages viewers with memorable, actionable insights.

Gestalt principles is a collection of visual perception principles that aid us in understanding how human perception works.

These principles explain how humans tend to interpret structure, logic, and pattern around them.

Purpose : Derived from psychological Research to guide better visual design. **Basis** : Explains how we perceive structure, logic , and patterns in visuals. **Application** : Helps create clear, effective visualisations by distinguishing the subject from its background.

A. Principle of Figure and Ground

1. This states that when we look at a visual, our brain instinctively distinguishes the objects as either they are in foreground or background.
2. Helps to Focus on main elements when you develop charts and put those elements in the foreground.
3. Example : Rubine vase faces.

B. Principle of similarity

1. This states if we group objects together then they appear to be similar. 2. Several Visual elements used to show the similarity among different groups such as shape, size or color.

C. Principle of Focal Point :

1. This states that whatever is different or distinct will be the focus point that catches the viewer's attention.

D. Principle of Continuity :

1. This states that the human brain is more likely to perceive a continuous, smooth path when they visualize a line or a curve, regardless of their shape and color.

E. Principle of Common Region :

1. This states that things that belong to the same closed region are perceived as a group.
2. By creating a boundary, we can create a perception that within that closed region things function similarly.

F. Principle of closure :

1. This states that the human brain tends to find a single known pattern whenever it looks at a complicated, incomplete arrangement of visual elements.

These Principles play an important role while building effective visualisation. One can incorporate these principles in their visualisations to grab the audience's attention for long and immediately.

Lesson 2

Seven Stages of data Analysis

1.**Acquire**: Focuses on gathering data from its source.
2.**Parse**: Structures data to define meaningful components.
3.**Filter**: Eliminates irrelevant details to highlight key data.
4.**Mine**: Analyzes data to reveal underlying trends and insights.
5.**Represent**: Begins with a simple visual to identify complex relationships and patterns.
6.**Refine**: Enhances visual representation to transform data into clear information. 7.**Interact**: Shares analysis with stakeholders, enabling exploration and deeper understanding.

1. **Acquire (Gathering Data)**

* Collect raw data from primary or secondary sources (databases, APIs, sensors, logs).
* Ensure completeness and accuracy of data before moving forward.

2. **Parse (Structuring Data)**

* Organize raw data into structured formats like tables, rows, and columns.
* Define variables and attributes that will be meaningful for analysis.

3. **Filter (Selecting Relevant Data)**

* Remove noise, errors, or irrelevant records that don't serve the analysis goal.
* Focus on subsets of data that align with the intended question or problem.

4. **Mine (Analyzing Data)**

* Apply statistical, machine learning, or computational methods to extract patterns.
* Identify correlations, clusters, or anomalies hidden in the dataset.

5. **Represent (Initial Visualization)**

* Create basic charts or plots (scatter, line, bar) to see initial trends.
* Highlight potential relationships or structures in the data.

6. **Refine (Improving Visuals)**
* Enhance clarity by adjusting scales, colors, labels, and layouts.
* Transform rough visuals into polished, interpretable graphics.

7. **Interact (Exploring & Sharing)**

* Enable users to explore the data interactively (dashboards, filters, drill-downs).
* Share insights with stakeholders to support decision-making.

Principles of Chart Selection

For data visualization to be effective, charts must be :

- appropriate for the data and easily understandable. The main goal is to select a visualization that clearly communicates the key information and insights you want to conclude from your data.

Types of Charts

Charts can be broadly categorized based on the primary message they convey: Comparison, Distribution, Composition, and Relationship.

1. **Comparison Charts**

These charts are used to compare one or more variables to see differences or rank categories.
Use Cases: Comparing product revenues, tracking quarterly sales to find trends, or analyzing monthly footfalls in a store.
Common Chart Type: **Bar Chart**

Column Bar Chart: Best for comparing data across different categories or showing chronological data.

Horizontal Bar Chart: Ideal for comparing variables but can become cluttered with too many items.

Stacked Bar Chart: Used to compare data within a category to itself, often showing percentages.

2. **Distribution Charts**

Distribution charts help identify the normal tendency, range, and presence of outliers in a dataset.
Use Cases : Showing population distribution by age, student grades in an exam, or the heights of children in a class.
Common Chart Types: **Scatter Charts & Histograms**
These are effective for showing data distribution, clustering trends, and spotting anomalies. For instance, a histogram can show how many words are typically used in customer reviews.

3. **Composition Charts**

These charts illustrate how individual parts make up a whole, focusing on the contribution of each component.
Use Cases : Visualizing a country's population by religion, showing company market share, or breaking down sentiments in reviews (e.g., 69% negative, 18% neutral, 13% positive).

Common Chart Types: **Pie charts, stacked charts, and map-based graphs**.

4. **Relationship Charts**
Relationship charts are used to demonstrate the correlation between different variables, which can be positive, negative, or nonexistent.
Use Cases: Analyzing the relationship between marketing expenses and profit, or testing a hypothesis like "Does salary depend on IQ?".
Common Chart Types :

Scatterplot & Line Chart: Show relationships between variables.

Bubble Chart: Shows the relationship between words or concepts, where the bubble's size indicates frequency and color can represent another variable like sentiment.

Word Cloud: Represents the frequency of words in a text, with the size of the word being proportional to its usage.

Main Selection Question is : How can we ensure that the visualization communicates the message correctly to our audience?

Mistakes to avoid :

1. A chart with too much information will create clutter and discourage your audience from seeking insights.
2. Avoid putting a lot of data points in one chart rather use multiple visualizations to convey the story.
3. Always label the axes and provide chart headers. This informs our audience of the context of our analysis.
4. Depending on the data, horizontal, and vertical axes should be scaled appropriately.

Select the chart type wisely so that the audience can easily understand your analysis.

Students are Requested to go through each example pictorially to clearly understand the differences.

Lesson 4

Taxonomy of Data Visualisation Methods

A small number of core visuals are typically used to convey the majority of data stories:

● **Simple Text:** Used when the primary goal is to **focus the audience on a single, key number**. Highlighting, such as using color or bold font, makes the number prominent. ● **Tables and Heatmaps:**

- ○ **Tables** engage the audience's **verbal system**, requiring them to read and look up precise values. Tables should be **data-centric** with minimal distracting borders.
- ○ A **Heatmap** is a table where **color saturation** indicates the relative magnitude or importance of the numbers, guiding the audience's brain to points of interest. ● **Scatter**

Plot (Points): This is useful for showing or exploring the **relationship (correlation) between two numerical variables**.

- **Bubble charts** are an extension that adds a third variable represented by the size of the point.
- **Line Graphs and Slopegraphs:**
  - **Line Graphs** are used to plot **continuous data**, typically to show **trends over time**. Strategic use of color and line types (e.g., dashed, thickness) can add detail to the visualization.
  - A **Slopegraph** is specifically used to plot categorical data, showing the **rate of change between just two time points**.
- **Bar Charts (Vertical, Horizontal, Stacked):** These are intuitive and easy to understand for **comparing categorical values**.
  - **Stacked Bar Charts** illustrate **additive components** (e.g., total sales broken down by product) or show **shares of a whole** (e.g., 100% stacked bar).
- **Waterfall Chart:** This special bar chart demonstrates how an **initial value is incrementally increased and/or decreased** by intermediate values to reach a final total. It's ideal for breaking down the components of a net change.
- **Area Charts (Treemap/Square Map):** These are useful for **analyzing the composition of categories**. In a **Treemap**, a larger area intuitively represents a higher share of the total.

Design Concepts for Data Communication

Data visualization design adheres to the **Product Design Philosophy: "Form follows function"**. This means you must first determine what the audience needs to *do* with the data (**function**) and then create the visual (**form**) to enable that action easily.

Key traditional design concepts applied to data:

1. **Affordances:** These are design cues that suggest how the visual should be used.
   - The goal is to **highlight the most important information** (about 10% of the visual).
   - Achieved by using **Color, Bold font, Underline, and Size** to focus attention, and by **simplifying the visual** (removing clutter).
2. **Accessibility:** The design must be clear for all users.
   - Ensure the visual has an **easy flow**, is **not overcomplicated**, and uses **legible, straightforward language**.
3. **Aesthetics:** Visual appeal is essential for engaging the audience.
   - Focus on being **smart with color**, maintaining **alignment**, and strategically using **white space**.
4. **Acceptance:** Since audiences often resist new formats, gaining buy-in is necessary and requires thoughtful **change management**.
   - Strategies include: articulating the **benefits** of the new design, showing **side-by-side comparisons** with the old design, and engaging **influential users** early to incorporate their feedback.

Lesson 5

Dissecting Model Visuals

The core principle of effective data visualization is **intentional design**, where every element—from color and alignment to text and labels—serves a specific purpose. The goal is to create visuals that clearly guide the viewer to an insight.

Key techniques from the examples include:

- **Emphasize Key Data:** Use pre-attentive attributes like **bold colors**, **thicker lines**, and **clear labels** to draw attention to the most important information, such as "Progress to date" on a line graph.
- **De-emphasize Context:** Use lighter, thinner lines or smaller fonts for secondary information, like comparison data (e.g., last year's results) or footnotes, to keep the focus on the main message.
- **Use Annotations:** Add text directly onto the chart to explain key events, trends, or forecast assumptions.
- **Leverage Layout and Color:**
    - For categories with long names, a **horizontal bar chart** is more readable. ○ In stacked bar charts, use a single,**attention-grabbing color** for the most critical category (like "Missed target").
    - Use **positive and negative bars** to intuitively represent concepts like growth and attrition.
- **Maintain Clarity:** Avoid clutter by labeling only the most essential data points. Ensure axis labels and titles are direct and easy to understand.

Lessons in Storytelling for Data Visualization

Simply showing data isn't enough; it must be communicated effectively through a story to drive action. This involves merging data, visuals, and a narrative to guide your audience from insight to decision-making.

1. Start with Context

Before creating any visual, you must understand your **audience** and your **goal**. Data visualization should be **audience-centric**, not data-centric.

- **Who** is your audience (e.g., executives, analysts)?
- **What** do you want them to know or do?
- **How** will they consume the information (e.g., live presentation, email)?

2. Choose the Right Visual

Select a chart type that best matches your message:

- **Line charts** are ideal for showing trends over time.
- **Bar charts** work well for comparing categorical data.
- **Scatter plots** are used to reveal correlations between two variables.

3. Design for Attention

Use a clear **visual hierarchy** to direct your audience's focus to what matters most.

● **Eliminate "Chartjunk":** Remove any distracting elements like 3D effects or unnecessary gridlines. If a design element doesn't add value, it detracts. ● **Use Preattentive Attributes:** Strategically use **color**, **size**, and **position** to make key data "pop" instantly. For example, place the most critical information at the top-left of your visual.

Lesson 6

Taxonomy of Data Visualization Methods

Data visualization methods can be categorized based on their primary communication purpose. While there are many purposes, this lesson focuses on two key areas: comparing categories and assessing hierarchies. Other purposes include showing changes over time, plotting connections and relationships, and mapping geo-spatial data.

I. **Comparing Categories**

This group of visualizations is designed for making comparisons between the relative and absolute sizes of categorical values.

Gantt Chart (Floating Bar)

● **Purpose**: This chart helps to show the range of quantitative values for a given category.
● **Visual Representation**: It uses a horizontal bar that stretches from the lowest value to the highest value for each category. This is effective for comparing the span or duration of different items.

Sankey Diagram

● **Purpose**: Sankey diagrams are primarily used to convey the idea of flow. ● **Visual Representation**: They portray constituent quantities as they move across a number of "stages". The ongoing associations between stages are represented by connecting bands.
● **How it Works**: The width of the connecting links is directly proportional to the quantity of the flow from one stage to another.
● **Use Case**: They are especially useful for visualizing situations where elements transform, divide, or combine over key events.

Small Multiples

● **Purpose**: This technique facilitates efficient and effective comparisons by using a multi-panel display of small, individual chart elements.
● **How it Works**: Small multiples leverage our ability to rapidly scan across a grid (or "trellis") of similar charts to spot patterns and anomalies.
● **Use Case**: They are very effective for comparing categories that have a broad range of values and for showing snapshots of events that change over time.

II. **Assessing Hierarchies and Part-to-Whole Relationships**

This category of visualizations provides a breakdown of categorical values, showing their relationship to a larger population or as constituent elements of a hierarchical structure.

Circle Packing Diagram

- **Purpose**: This method shows part-to-whole relationships by packing constituent circles (the parts) into an overall circular area (the whole).
- **Visual Representation**: Each individual circle represents a different category. The size of each circle is proportional to its associated quantitative value.
- **Enhancements**: Other visual variables, such as color and position, can be used to add further layers of meaning to the display.

Bubble Hierarchy

- **Purpose**: A bubble hierarchy is specifically used to portray organization and structure through a hierarchical display.
- **Visual Representation**: It uses circles to represent the constituent parts of the hierarchy, such as different departments.
- **How it Works**: The circles are sized according to their quantitative value (e.g., budget, staff count) and are colored to visually distinguish the different departments or categories.

Lesson 7

I. **Showing Changes Over Time**

This is one of the most common types of data analysis, focusing on visualizing trends and patterns within a specific timeframe. Common charts for this purpose include the **Line Chart** and the **Area Chart**.

Stacked Area Chart

- **Purpose**: This chart provides a compositional view, showing how the parts of a whole change over time.
- **Visuals**: It consists of stacked layers of area charts, each differentiated by color. It can display either absolute numbers or percentage aggregates. The height of each colored stack at any point along the time axis represents its quantitative value.

Steam Chart

- **Purpose**: This chart's main function is to highlight peaks and troughs in data over time. It's well-suited for showing "ebb and flow" stories due to its organic feel.
- **Visuals**: It layers multiple data series as "streams" of area, where the height of an individual stream represents its value at that point in time. Unlike a stacked area chart, a steam chart has no baseline x-axis, meaning there is no concept of negative or positive values—only aggregates.

Flow Map

- **Purpose**: A flow map visualizes the movement of a quantitative value as it is transformed over time and/or space.
- **Features**: A famous example is the chart showing Napoleon's army marching to Russia in 1812. These maps can preserve geographical accuracy without needing full map detail. They are often enhanced with other data layers, such as a separate line chart below showing temperature, to add more context to the story.

II. Plotting Connections and Relationships

These visualizations are used to assess the associations, distributions, and patterns that exist within multivariate datasets. They are often complex and are primarily used for exploratory analysis.

Scatter Plot Matrix

- **Purpose**: This method uses a grid of scatter plots to compare every variable in a dataset against every other variable. It leverages the eye's ability to rapidly spot patterns and correlations across multiple views of the same chart type.

Radial Network (Chord Diagram)

- **Purpose**: This diagram shows complex relationships between different categorical values, moving beyond the restrictions of a simple x-y axis.
- **Visuals**: The key feature is the connections (or chords) between different components around the circle. The thickness and color of these connections can be used to represent additional layers of detail, such as the strength of the relationship.
- **A Note of Caution**: The length of a connecting line can be misleading, as it is simply a by-product of how the categories are arranged around the circle and does not represent a value.

Network Diagram (Force-Directed)

- **Purpose**: These diagrams facilitate the exploration of complex data structures by showing relationships, connections, and logical organization. The goal is to enable the viewer to see patterns like clusters, gaps, dominant nodes, and sparse connections. ● **Appearance**: They can often look daunting and cluttered due to their complexity.

III. Mapping Geo-spatial Data

These methods are used to visualize datasets that have geographic properties.

Choropleth Map

- **How it Works**: This map colors different geographic units (like states or counties) based on a quantitative value, using a sequential or diverging color scheme to show intensity. ●

Shortcoming: A major flaw is that populations are not uniformly distributed. This can create a distorting effect where large geographic areas get more visual prominence, even if they are not proportionately representative of the underlying data.

Isarithmic Map (Contour Map)

- **Purpose**: This map type is designed to overcome the flaws associated with the choropleth map.
- **How it Works**: It combines color hue, saturation, and darkness to represent the density of a population or value. Algorithms are then applied to smooth the representation, creating an elegant contour effect that shows data density independent of geographic boundaries.

Network Connection Map

- **Purpose**: This map facilitates the exploration of complex geographical connections. It is essentially a network diagram laid over geographic coordinates.
- **How it Works**: It joins related locations with lines, forming a pattern that helps discover hubs, clusters, overlaps, and gaps.
- **A Unique Feature**: When a dataset is exhaustive enough, the sheer number of connection lines can form an image of the world map on its own, making the underlying map layer unnecessary.

Dataset

Given Directly as Excel.

(Assume this occupies cells `A1:G13`, with `Month` in `A1` and `Visible Series` in `G1`.)

Worked example (FULL STEPS): Create a Line Chart comparing series A, B, C, D across months

Goal: make a clean, presentation-ready line chart that shows the monthly trends of A, B, C and D.

Steps (Excel desktop ribbon steps)

1. **Paste the dataset** into a new worksheet (A1:G13). Make sure `A1` = `Month`, `B1` = `A`, `C1` = `B`, `D1` = `C`, `E1` = `D`, etc.

2. **Convert to an Excel Table** (optional but helpful): select `A1:E13` (Month + A–D) → **Insert** tab → **Table** → OK. This makes ranges dynamic.

3. **Select data for the line chart**: click and drag to select `A1:E13` (Month + columns A, B, C, D).
   Note: We intentionally exclude the large-magnitude `Visible Series` from this example so the small A–D lines remain comparable.

4. **Insert the chart**: with that range selected → **Insert** tab → **Charts** group → click **Line Chart** (2-D Line) → choose **Line with Markers**.

5. **Place and resize** the chart on the sheet; give it breathing room.

6. **Chart Title**: Click default title → type `Monthly trends – A, B, C, D`.

7. **Axis Titles**: Chart Tools (or Chart Design) → **Add Chart Element** → **Axis Titles** → **Primary Horizontal** and **Primary Vertical**.
   - Horizontal: `Month`
   - Vertical: `Value`

8. **Format X-axis**: right-click horizontal axis → **Format Axis** → ensure categories are treated as text (Category Axis). If Excel tries to treat months as date serials, set axis type to **Text axis**.

9. **Add markers & smooth lines (optional)**: Right-click a data series → **Format Data Series** → Marker options → choose marker type. To smooth lines: **Line** → check **Smoothed line** (if you like).

10. **Legend**: Position the legend where appropriate — Chart Design → **Add Chart Element** → **Legend** → e.g. **Top** or **Right**.

11. **Gridlines and background**: remove heavy gridlines if you want a clean look — Chart Design → **Add Chart Element** → **Gridlines** → uncheck Major Gridlines. For presentation: Format Plot Area → Fill: No fill.

12. **Final polish**: select each series → Format Data Series → increase line width to 2.25 pt for readability; add data labels only if necessary (too many labels will clutter). Save the workbook.

What to submit for this worked example: the chart embedded in the worksheet (one sheet named `LineChart_Solution`) + short note (1–2 lines) describing one insight you see (e.g., “Series A peaks in May and Nov; series C is relatively flat.”).

Student Tasks (one per chart) — use the same workbook

For each task below: **(a)** create the requested chart or table in a new sheet named as suggested, **(b)** take a screenshot or save the workbook with the chart embedded, and **(c)** write a 1–2 line observation/insight under the chart.

Task 1 — Scatter Plot (`Sheet: Scatter`)

What to plot: Investigate relationship between `Invisible Series` (X) and `Visible Series` (Y).

Steps / hints:

- Create two columns with X = `Invisible Series` and Y = `Visible Series` (`F2:F13` and `G2:G13`).
- Insert → Charts → **Scatter** → **Scatter with only Markers**.
- Add labels: each marker should show the month (use data labels and manually edit or use XY with data labels).
  Deliverable: scatter plot and one-sentence interpretation (e.g., is there a visible correlation?).

Task 2 — Vertical Bar (Clustered Column) (`Sheet: VerticalBar`)

What to plot: `Visible Series` per Month as vertical columns.

Steps / hints:

- Select `A1:A13` (Months) and `G1:G13` (Visible Series).
- Insert → **Column Chart** → **Clustered Column**.
- Optional: add data labels above columns.
  Deliverable: chart + note (e.g., highest/lowest months).

Task 3 — Horizontal Bar (`Sheet: HorizontalBar`)

What to plot: Total of A+B+C+D per month (create a helper column `Total AtoD = SUM(B2:E2)` then use that).
Steps / hints:

- In `H2` insert `=SUM(B2:E2)` and fill down to `H13`.
- Select `A1:A13` and `H1:H13` → Insert → **Bar Chart** → **Clustered Bar** (horizontal bars).
  Deliverable: horizontal bar chart and 1-line insight.

Task 4 — Table (visual) (`Sheet: FormattedTable`)

What to make: A well-formatted table of the full dataset with conditional formatting.
Steps / hints:

- Select the whole dataset `A1:G13` → Insert → **Table** (Ctrl+T).
- Use Table Styles → pick a clean style; turn on Banded Rows and Filters.
- Add **Conditional Formatting** to the Visible Series column: Home → Conditional Formatting → **Data Bars** and also apply **Color Scale** to A–D to create a mini heatmap inside a table.
  Deliverable: formatted table and 1–2 lines describing how conditional formatting helps spot patterns.

Task 5 — Stacked Vertical Bar (`Sheet: StackedVertical`)

What to plot: Stack series A, B, C, D per month (so each month's total is the vertical stack).
Steps / hints:

- Select `A1:E13` (Month + A–D) → Insert → **Column** → **Stacked Column**.
- Add legend, axis titles, and data labels (show either totals or individual segment values).
  Deliverable: stacked column chart + comment on which component contributes most overall.

Task 6 — Stacked Horizontal Bar (`Sheet: StackedHorizontal`)

What to plot: Same data as Task 5 but horizontally (stacked bars).
Steps / hints:

- Select the same range → Insert → **Bar** → **Stacked Bar**.

- Consider sorting months by total (optional) for clearer story.
  Deliverable: chart + short observation.

Task 7 — Heatmap (`Sheet: Heatmap`)

What to make: Heatmap of values in A–D across months (a matrix).
Steps / hints:

- Select `B2:E13` (A–D numeric grid).

  - Home → Conditional Formatting → **Color Scales** → choose a two- or three-color scale.

- Optional: add borders, freeze top row, and add a small legend text explaining color mapping.
  Deliverable: heatmap and short interpretation (e.g., months with highest concentration).

Task 8 — Slopegraph (`Sheet: Slopegraph`)

What to plot: Show change from **Jan** → **Dec** for the four series A, B, C, D (one line per series).
Why these columns: slopegraph compares two timepoints; using Jan vs Dec shows net change across the year for the four series A–D.

Steps / hints (simple method):

Create a small helper table:

Series | Jan | Dec
A | 6 | 7
B | 4 | 6
C | 2 | 3
D | 1 | 4

1.
2. Select this 3×5 range (headers included) → Insert → **Line Chart** (2-D Line) — Excel will

plot one line per Series across two x-points (`Jan`, `Dec`).

3. Format: remove markers if cluttered, add labels at both ends for each series (manually place text boxes or use data labels), thin lines, emphasize start/end labels.

4. Alternatively, use a scatter + lines approach if you want to control x-axis spacing.

Deliverable: slopegraph showing the four series and a one-line observation (who gained/lost most).

Task 9 — Waterfall (`Sheet: Waterfall`)

What to plot: Build a **waterfall** of the `Visible Series` across months to show how the `Visible Series` changes month-to-month (or show cumulative). Use the `Invisible Series` helper if you want to construct manually.

Two ways (pick one):

- **Built-in Waterfall (easy)**: Select `A1:G13` or just `A1:A13` & `G1:G13` → Insert → **Insert Waterfall, Funnel, Stock, Surface or Radar Chart** → choose **Waterfall**. Tell Excel which points are totals if needed.

- **Manual stacked-column method (teaches helper-series):**

    1. Use `Invisible Series` as the bottom stack and `Visible Series` as the top stack: select `A1:F13` (Month + Invisible + Visible) → Insert → **Stacked Column**.

    2. Format the `Invisible Series` fill = **No fill** (so it becomes invisible), leaving only the visible columns offset to the correct base.

    3. Color positive and negative segments as needed. Add data labels.

Deliverable: waterfall chart and short explanation (e.g., months that contribute largest positive/negative steps).

Task 10 — Square-area chart (Treemap) (`Sheet: Treemap`)

What to plot: A “square area” representation of the `Visible Series` by month (i.e., **Treemap**).

Steps / hints:

- Select `A1:A13` and `G1:G13` (Month + Visible Series).

- Insert → **Insert Hierarchy Chart** → **Treemap**.
- Add a title and legend. Treemap will show rectangles sized by the Visible Series value.
  Deliverable: treemap + one insight (which months dominate area).

Extra / Bonus (advanced polish)

- Put **Visible Series** on a **secondary axis** if you combine it with A–D in the same chart (Format Data Series → Plot on Secondary Axis).

- Add **sparklines** next to month rows: Insert → Sparklines → Line.

- Create a **PivotTable + PivotChart** that shows A–D sums by quarter (if you add a Quarter helper column).

- Add **interactive slicer** (Insert → Slicer) after converting to Table to filter by quarter or other groups.

FINALLY,

- **For each task** include: the chart/table embedded and a 1–2 line observation written in a text box or cell underneath.

Effective Visuals Design: Key Concepts

This lesson focuses on how to design effective visuals by managing cognitive load and reducing clutter. The main goal is to present information in a way that is easy for the audience to understand and process.

Cognitive Load

Cognitive load refers to the mental effort required to learn new information. When presenting data, you are using the audience's mental processing power.

- **Goal:** The aim is to make **optimal use** of this cognitive load to deliver information effectively.
- **Problem:** Unnecessary visual elements, known as **clutter**, increase cognitive load and make it harder for the audience to understand the main message.

Clutter in Data Visualization

Clutter consists of visual elements that take up space but do not add to the understanding of the information. It complicates the visual and hinders effective communication.

Categories and Types of Clutter

Main Categories of Clutter

Visuals can be broken down into five main components where clutter can occur:

1. **Frame:** The background and borders of the chart.
2. **Graphical objects:** The bars, lines, or points used to represent data.
3. **Axes:** The reference lines for measurement.
4. **Labels:** All text elements, including titles, axis labels, and legends.
5. **Overlay objects:** Explanatory elements like callout boxes that are placed over the chart.

Other Common Types of Clutter

- **Lack of Visual Order:** When elements are not logically or neatly arranged, the visual can feel chaotic. Strategic use of color and alignment can bring order. For example, in a bar chart showing survey feedback, highlighting the most important result with a dark color and making other bars a muted grey creates a clear focal point.
- **Poor Alignment:** Text and graphical elements should be aligned to create a clean, professional look. Avoid center-aligning text; left- or right-aligning text is typically easier to read.
- **Ineffective Use of White Space:** White space is the empty area around elements. Just like pauses in a speech, strategic use of white space can draw attention to the important parts of your visual.
- **Non-strategic Use of Contrast:** Contrast helps direct your audience's attention. However, if you try to make too many things stand out (e.g., using many different colors and shapes in a scatter plot), nothing will stand out. A better approach is to use a single color, like blue, for your data and grey for competitors to make your performance easy to see.

Step-by-Step Guide to Decluttering a Chart

Let's use an example of a line graph showing "Ticket Volume Received" versus "Ticket Volume Processed" over a year to illustrate how to declutter.

The Goal: Simplify the visual to clearly show the relationship between tickets received and processed, especially during months when team members left, to help decide if new hires are needed.

1. **Remove the Chart Border:** A chart border is usually unnecessary and adds visual noise. The human brain automatically sees the chart elements as a cohesive whole (this is the Gestalt principle of closure). White space can be used to separate the chart from other page elements.
2. **Remove Gridlines:** Gridlines can also be distracting and are often not needed. If some guidance is necessary to read the values, make the gridlines a very light, muted color, like a faint grey, so they don't compete with the data.
3. **Remove Data Markers:** Data markers (the dots or squares on a line graph) are often redundant because the lines already show the data's path. Only include them if they serve a specific purpose.
4. **Clean Up Axis Labels:**
    - **Y-Axis (Vertical):** Remove unnecessary precision. For example, if the values are whole numbers, change labels like "150.00" to "150".
    - **X-Axis (Horizontal):** Abbreviate labels for readability. Change "January," "February," etc., to "Jan," "Feb" to avoid diagonal text, which is hard to read. 5. **Label Data Directly:** Instead of using a separate legend, place the labels ("Received" and "Processed") directly next to the corresponding lines. This makes it easier for the audience, as they don't have to look back and forth between the legend and the data. 6. **Use Consistent Color:** Match the color of the text label to the color of the data line it describes. For example, make the "Received" line and its label blue, and the "Processed" line and its label red. This reinforces the connection between the labels and the data.

The Role of Memory in Data Visualization

Our ability to process visual information is tied to three types of memory, each with a specific role. Understanding them helps in designing effective visuals.

- **Iconic Memory:** This is your ultra-fast, unconscious visual sensory memory. It lasts for a **fraction of a second** and has a very large but brief capacity. Its purpose is to rapidly scan the environment for changes. In design, this is where **pre-attentive attributes** like strong colors and contrasts grab a viewer's initial attention.
- **Short-Term Memory:** This is your conscious "working" memory, holding information for active use for a **few seconds**. Its capacity is famously limited to about **four chunks of information**. For designers, this means it's crucial to **avoid visual overload** and simplify elements so the audience isn't overwhelmed.
- **Long-Term Memory:** This is where we store patterns, concepts, and experiences for retrieval, from minutes to a lifetime. It has a **virtually unlimited capacity** and stores both visual and verbal information. To aid recall and transfer information to long-term memory, designs should **combine visuals with words**.

Pre-attentive Attributes: The Key to Guiding Attention

Pre-attentive attributes are visual properties that our brains process in an instant, without conscious effort. By strategically using them, you can guide your audience to see what you want them to see before they even realize it. These attributes reduce the cognitive load, making

information easier and faster to process.

Examples of pre-attentive attributes include:

- **Form:** Orientation, shape, line length, line width, size, curvature
- **Color:** Hue (the color itself) and intensity (saturation)
- **Position:** Spatial position, enclosure, added marks
- **Motion**

Using Pre-attentive Attributes in Text

Without visual cues, a block of text forces a reader to read everything to find the important parts. By adding pre-attentive attributes, you create a **visual hierarchy** that makes the text scannable and directs attention instantly.

For example, consider this feedback: "Your products are the best in class. Replacement parts are shipped when needed. Bev in the billing office was quick to resolve a billing issue."

- **No Attributes:** You have to read the entire text to understand the key points.
- **With Attributes:** Using **bold**, a different color, or a larger font for key phrases like "**best in class**," "**shipped when needed**," and "**quick to resolve**" makes them pop, creating a summary that can be understood at a glance. Some attributes, like color and size, are more powerful attention-grabbers than subtler ones like italics.

Using Pre-attentive Attributes in Graphs

Graphs without pre-attentive attributes are useful for **exploratory analysis** (where you are looking for insights) but not for **explanatory analysis** (where you are communicating insights to others).

To make a graph explanatory, you must guide your audience.

1. **Original Graph:** A simple bar chart showing the "Top 10 design concerns" with all bars in the same color requires the viewer to read every line to understand the data. 2. **Adding Color:** By changing the color of bars for concerns that occur "10 or more times per 1,000," you immediately draw attention to the most frequent issues.
3. **Creating a Story:** To tell a more specific story, you can use color and intensity to highlight only the noise-related concerns ("Tires make excessive noise," "Engine makes abnormal noise," "Excessive wind noise"). Pairing this with text annotations next to the highlighted bars provides context and drives a specific insight.

Another example is a line chart showing "Tickets Received" versus "Tickets Processed." By de-cluttering the graph and using color to emphasize the "Processed" line and adding data labels only at key points (like the lowest point), you can draw attention to a productivity gap, making a clear case for a decision like hiring more staff.

The Strategic Use of Size

Size is a powerful pre-attentive attribute because it implies importance; larger elements grab more attention. Therefore, it's critical that the visual weight you give to an element matches its actual importance.

A real-world example from Google involved a dashboard where the data that was easiest to get was given the most space. This accidentally made less critical data look more important, which could have led to poor decisions. The design was corrected to give **equal visual weight to equally important elements**, reinforcing the principle that good design must be intentional.

Lesson 11:

1. Designing Visuals

- **Visualization Best Practices**
  - Focus on clarity and effectiveness when presenting data.
  - Emphasize the use of appropriate visual encodings.
- **Edward Tufte's Visual Encoding**
  - A guide for translating data into meaningful visuals.
- **Conversion of Data into Visualizations**
  - Visuals help draw insights from raw data.
  - Important to use proper encoding to effectively communicate the data.

2. Data Types

Understanding the different types of data is crucial for choosing the right visualization approach.

- **Nominal Data**:
  - Qualitative; represents categories (e.g., Gender, Blood Group).
  - Can only be compared using equality or inequality (e.g., `=`, `!=`).
- **Ordinal Data**:
  - Data from an ordered set (e.g., Customer Reviews).
  - Allows comparisons like `=`, `!=`, `>`, `<`.
- **Interval Data**:
  - Fixed interval sets (e.g., IQ level, temperature in Celsius).
  - Comparisons with `=`, `!=`, `>`, `<`, and subtraction are meaningful.
- **Ratio Data**:
  - All arithmetic operations are meaningful (e.g., Revenue, Sales).
  - Supports comparisons and operations like `=`, `!=`, `>`, `<`, `+`, , `%`.

3. Effectiveness of Visual Encodings

How humans decode graphs and patterns plays a key role in understanding visualizations.

- **Three Visual Operations** (Cleveland, 1994):
    1. **Detection**: Recognizing patterns or values.
    2. **Assembly**: Grouping geometric objects to understand relationships.
    3. **Estimation**: Assessing magnitude or comparisons.
- **Gestalt Laws**: Help in grouping objects effectively, such as the Law of Continuity for linking related data points.

4. Types of Visual Encodings

Different encoding techniques are used for visualizing data, with varying levels of effectiveness.

- **Position on Common Scale**: Most effective for all types of comparisons (Discrimination, Ranking, Estimation).
- **Position on Identical but Non-Aligned Scales**: Effective, but less so than a common scale. ● **Length**: Very effective when comparing against a common axis.
- **Angle**: Useful for ranking or estimation, especially with changes over time. ● **Volume / Density / Saturation**: Useful for all types of comparison but less effective than other methods.
- **Color**: Best for discrimination, not suitable for ranking or estimation beyond simple categories.

5. Data Encoding Techniques

Different encoding strategies for presenting data visually:

- **Color**:
    - ○ Use to distinguish data, but avoid more than six colors in a single chart.
    - ○ Can be effective for discrimination but not for ranking or estimation.
- **Volume**:
    - ○ Effective for discrimination and ranking.
    - ○ Allows comparison and estimation of differences.
- **Angle**:
    - ○ Best used to indicate rate of change rather than absolute values.
- **Length and Position**:
    - ○ Very effective for visualizing differences, rankings, and comparisons.

6. Expressiveness vs Effectiveness

- **Expressiveness**: How well the visualization shows all facts in the data.

- **Effectiveness**: How easily the viewer perceives the data through the visual representation.

7. References

- **Books for Further Reading**:
  - *Data Visualization – Storytelling Using Data* by Sharad Sringeswara, Purvi Tiwari, U. Dinesh Kumar.
  - *Storytelling with Data* by Cole Nussbaumer Knaflic.
  - *Information Dashboard Design* by Stephen Few.
  - *Data Visualization: A Successful Design Process* by Andy Kirk.

Lesson 12:

1. Designing Visuals

- **Visualization Best Practices**
  - Importance of selecting the right chart type for different types of data.

- **Visualization of Structured Data**
  - Understanding the importance of visualizing structured datasets effectively.

- **Choosing the Right Chart for the Data**
  - Selecting appropriate charts to draw meaningful inferences from data.

2. Understanding Variable Types

- **Categorical Data**: Data that represents categories (e.g., Gender, Race). ● **Binary Data**: Data with two categories (e.g., Pass/Fail, Buy/Did not buy). ● **Ordinal Data**: Data with a ranking order (e.g., High, Medium, Low).
- **Numerical Continuous Data**: Data that is measured on a continuous scale (e.g., Revenue). ● **Numerical Discrete Data**: Countable data (e.g., Number of customers, vehicles).

3. Univariate Analysis

- **Retail E-commerce Data**:
  - Example from the UCI repository of online shoppers.

- **Types of Visuals for Univariate Analysis**:

  - **Bar Chart**: To show categorical data.
  - **Big Number**: Highlight key figures.
  - **Pie/Donut Chart**: To represent proportions in a circular format.

4. Univariate Analysis – Extended

- **Understand Relationships**:
    - Preferred shopping days, types of visitors, and region-wise analysis.

5. Univariate Analysis – Distribution

- **Histogram**: Represents the frequency distribution of a dataset.
- **Frequency Distribution**: Distribution of data points across different ranges.
    - Example: **Bounce Rates**: Bounce rates for websites are between 0.000 and 0.050, with a few outliers.
    - **Conversion Rate**: Typically occurs within the first 28 minutes of the initial click (Nielsen, 2005).

6. Univariate Analysis – Box Plot

- **Box Plot**: A visualization for distribution at quartiles.
    - **Inter-Quartile Range (IQR)**: The difference between the lower and upper quartiles. ○ **Whiskers**: Extend 1.5 times the IQR. Points outside whiskers are outliers. ○ **Data Symmetry**: Shows the distribution around the median, and helps detect skewness.
- **Business Insight**: Lower bounce rates lead to better business conversions.

7. Multivariate Analysis

- **Data with Two or More Attributes**: Used to visualize relationships between multiple variables.
    - **Types of Multivariate Analysis**:
        - **Comparison**: Stacked bar charts, box charts.
        - **Relationships**: Scatter plots, heat maps, parallel coordinates.
        - **Trends**: Line charts, dual-axis charts.

8. Multivariate Analysis - Comparison

- **Example**: Comparison between conversion rates on weekdays vs. weekends.
    - ○ Weekdays: 15% conversion.
    - ○ Weekends: 17% conversion.
- **Customer Visits**: More customers visit administration pages compared to informational pages.
    - ○ Customers visiting administration pages are more likely to make a purchase.

9. Multivariate Analysis - Relationships

- **Scatter Plot**: Displays the relationship between two variables.
    - ○ **Correlation vs Causation**: Not all relationships are valid; confounding or hidden variables can influence the results.
    - ○ **Encoding**: Use color, hue, shapes, and size to represent additional variables. ●

Example: Higher exit and bounce rates are related to low page values. ● **Joint Plot**:

Shows distributions of both X and Y variables in one plot.

- **Heat Map**: Visualizes the correlation between variables.
- **Parallel Coordinates Chart**: Displays relationships between multiple variables on parallel axes.
    - ○ Care must be taken to scale the vertical axes for accurate interpretation. ○ **Business Insight**: Page values differ for buyers and non-buyers, but there is no significant difference in exit or bounce rates.

10. Multivariate Analysis - Trends

- **Line Charts**: Show trends over time or continuous variables.
- **Dual-Axis Charts**: Example: Showing how visitors increase sales as the number of visitors increases.

Lesson 13:

1. Designing Visuals

- **Visualization Best Practices**: Effective ways to visualize data, particularly text data, and the importance of choosing the right visualizations.

- **Visualization of Text Data**: Challenges and techniques for visualizing textual content such as documents, emails, and conversations.

2. Challenges in Visualizing Text Data

- **Preprocessing Steps**:

    ○ **Case Normalization**: Converting text to either all uppercase or lowercase.
    ○ **Number Handling**: Removing or converting numbers to words.
    ○ **Punctuation Removal**: Removing special characters (e.g., !/@#$%^&<>?*").
    ○ **Tokenization**: Breaking text into smaller tokens (e.g., words or phrases). ○ **Stop Words Removal**: Common words (e.g., "the", "a", "and") are removed as they do not add value to text analysis.
    ○ **Stemming**: Reducing words to their root form (e.g., “running” becomes “run”).
    ○ **Lemmatization**: More detailed morphological analysis for capturing the essence of a word.

3. Visualizing Text Data

- **Word Clouds**: Visual representation of the most frequent words in a text (e.g., representing the main characters of a story).

- **Bar Charts**: Show the frequency of specific words (e.g., how often “Lion King” appears in the text).

- **Sentiment Scores**: Classifying sentiments into categories (positive, negative, neutral) based on sentiment analysis scores ranging from -1 to +1.

4. Visualizing Text Data with Joint Plot (Textblob)

- **Polarity**: Measures the sentiment of the text on a scale from -1 (negative) to +1 (positive).
- **Subjectivity**: Measures how subjective or objective the text is, ranging from 0 (objective) to 1 (subjective).

- **Joint Plot**: Combines scatter and distribution plots to visualize the relationship between Polarity and Subjectivity of text.

5. Visualizing Conversations

- **Timeline**: Visualizing the temporal sequence of conversations.
- **People (Who)**: Visualizing the participants in conversations.
- **Information Flow (Who - When)**: Understanding the interaction between people and time.
- **Network (Who - Who)**: Mapping relationships between participants in conversations through a network graph.
- **Content (What)**: Visualizing the content of conversations.
- **Enron Email Dataset**: Example dataset used for visualizing conversations and analyzing patterns.

6. Visualizing Conversations – Network Graph

- **Nodes and Edges**: Nodes represent entities (people), and edges represent the connections between them (e.g., email exchanges).
- **Network Visualization**: Helps understand the structure and flow of communication between participants in a network.

7. Word Embedding

- **Semantic Relationship**: Words with similar meanings often occur in similar linguistic contexts.
    - Example: “King” and “Man” are semantically similar, just as “Queen” and “Woman” are.
- **Word Embedding**: Visualizing the relationships between words based on their context in the text.

8. Topic Modelling

- **Latent Dirichlet Allocation (LDA)**: A machine learning method for extracting topics from a corpus of text.

- **Latent Semantic Analysis (LSA)**: Another method for extracting topics by analyzing the relationships between words in a document.

- **Topic Visualization**:

    - **Bubble Charts**: Show the distance between topics.
    - **Bars**: Represent the frequency of words in topics.

9. t-SNE Clustering

- **t-Distributed Stochastic Neighbor Embedding**: A technique used to reduce high-dimensional data into two-dimensional space.

- **Clustering with t-SNE**: Helps visualize clusters of similar documents or topics by placing them in a 2D scatter plot.

- **Document Representation**: Each document is represented by a colored rectangle indicating its assigned topic and the dominant word-topic associations.

Lesson 14:

What is a Dashboard?

A **dashboard** is a visual tool that consolidates key information (e.g., KPIs) on a single screen to facilitate easy monitoring and quick decision-making.

Why Do Different Dashboard Types Exist?

Dashboards are tailored to different audiences (executives, analysts, operators) based on factors such as time sensitivity, interactivity, and the types of decisions that need to be supported.

Characteristics of Dashboards

1. **Visual Summarization**: Condenses complex data into easy-to-understand visuals for quick comprehension.

2. **Single-Screen Display**: Ensures all information is visible on one screen without the need to scroll or click.

3. **Real-Time or Near Real-Time**: Dashboards should display data in real-time or near real-time for quick responses to anomalies.

4. **Customization & Interactivity**: Allows users to filter data and create role-based views tailored to their needs.

5. **Contextual Relevance**: Data presented should be directly relevant to the user's goals and objectives.

6. **Visual Design Principles**: Utilizes pre-attentive attributes (e.g., color, size, position) and clear layout to guide the user's attention and make data easy to understand.

Additional Resources

- **Example**: Greenhouse Gas Emissions Data (IMF)

- **Additional IMF Data**: NGDP Growth Rates by Country

Lesson 15

Types of Data Used in Dashboards

1. **Categorical**: Data that represents categories such as department, region, or product.
2. **Quantitative**: Numerical data such as sales, calls handled, or uptime. 3. **Time-Series**: Data showing trends over time (e.g., hours, days, or months). 4. **Comparative**: Data comparing actual performance against targets or year-over-year data.
5. **Status Indicators**: Alerts, thresholds, and exceptions that signal specific conditions.

Focus: Prioritize actionable metrics that can drive decision-making.

Identifying Dashboard Data Types

- **Categorical Data**: For example, services like visa and passport information.
- **Quantitative Data**: Numerical data such as citizen services numbers. ● **Time-Series Data**: Trends such as visa issuance over time.
- **Comparative Data**: For example, text boxes showing a percentage decrease in data.
- **Status Indicators**: For example, a 3-day visa issuance time threshold.

Performance Dashboards

Performance dashboards focus on:

- **Strategy Execution**: Tracking progress against strategic goals.
- **Monitoring**: Observing performance in real time.
- **Collaborative**: Enabling teamwork and shared insights.
- **Corrections**: Making adjustments based on real-time feedback.
- **Objective Transparency**: Providing clear insights into goals and performance.

Framework: The performance dashboard framework is based on the book *Performance Dashboards: Measuring, Monitoring, and Managing Your Business* by Eckerson.

Performance Dashboard Layers

- **Graphical Abstracted Data**: Graphs, charts, and symbols for quick insights.
- **Summarized Dimensional Data**: Dimensions and hierarchies for analysis (slice/dice).
- **Detailed Operational Data**: In-depth data for operational decision-making.
- **Reporting**: Operational reports and planning updates such as forecasts and models.

Examples of Performance Dashboards

- **Vahan Dashboard**: An example of a performance dashboard displaying vehicle data.

Mistakes in Dashboard Design (Stephen Few)

1. Exceeding the boundaries of a single screen.
2. Supplying inadequate context for the data.
3. Displaying excessive detail or precision.
4. Choosing a deficient measure.
5. Choosing inappropriate display media.
6. Designing unattractive visual displays.
7. Introducing meaningless variety.
8. Using poorly designed display media.
9. Encoding quantitative data inaccurately.
10. Poor arrangement of data.
11. Ineffective or absent highlighting of important data.

12. Cluttering the display with unnecessary decoration.
13. Misusing or overusing color.

Lesson 16

Design Goals

- **Edward Tufte's Data-Ink Ratio Principle**: Maximize the data-ink ratio, meaning every pixel of ink on a graphic should present new information. Non-data pixels (such as background or decoration) should be minimized.

- **Reduce Non-Data Pixels**: Eliminate unnecessary elements and de-emphasize the remaining ones to focus on the data.

- **Enhance Data Pixels**: Remove unnecessary data elements and highlight the most important data for clarity.

Example of Reducing Non-Value Adding Elements:

- 3D diagrams, grid lines in bar graphs, unnecessary decoration, color gradients, and irrelevant background.

Visual Design Process

- **Top-Left and Center Areas**: These areas receive the most visual attention due to natural reading patterns and the way human vision works.

    - **Emphasize critical information** in these prime areas.
    - Avoid placing non-essential items (e.g., logos) in these areas.

Use of Visual Attributes to Enhance Designs

1. **Color Intensity**: Saturated colors are perceived as more important.
2. **Size**: Larger elements are perceived as more important.
3. **Line Width**: Thicker lines stand out more and signify importance.

Design Goals for Dashboards

- **Simple Presentation**: Display a large amount of data in a small space without clutter.
- **Effectiveness and Expressiveness**: The design should clearly communicate information.
- **Space Efficiency**: Dashboards should still be effective even when scaled to smaller spaces.

Example: OECD Life Expectancy Dashboard: OECD Gender Dashboard

Categorization of Design Visuals

- **Graphs**: Line, bar, area graphs.
- **Images**: Photographs and illustrations.
- **Icons**: Represent on/off, right/wrong, increase/decrease.
- **Text**: Highlight key elements.
- **Organizers**: Small multiples for comparative views.
- **Tables**: Used for precise data presentation.
- **Spatial Maps**: To show geographic relationships.

Example: OECD Life Expectancy Dashboard

Lesson 17

Challenges in Dashboard Data

1. **Data Latency**: The delay between the event and the update in the dashboard.
2. **Data Inconsistency**: Conflicting sources and definitions of data.
3. **Missing Data**: Incomplete records or gaps in the data system.
4. **Overload**: Too much data leading to cognitive fatigue and difficulty in decision-making.

Granularity of Dashboards:

- **High-Level (Strategic)**: Aggregated KPIs, quarterly trends.
- **Mid-Level (Tactical)**: Weekly performance, campaign results.
- **Low-Level (Operational)**: Real-time transactions, incident counts.

Tailor the granularity of the dashboard to the user's needs.

Data Context : Making Metrics Meaningful

- **Numbers Alone Aren't Enough**: Show actual vs. target using benchmarks, goals, or past periods.
- **Tools to Provide Context**:
    - Reference lines in charts.
    - Status icons .
    - Bullet charts and gauges.

Insight: Context transforms numbers into meaningful narratives.

Criteria for Dashboards

Criteria	Strategic	Analytical	Operational	Tactical
Update Frequency	Periodic	On-deman d	Real-time	Daily/weekly
Interactivity	Low	High	Medium	Medium
Audience	Executives	Analysts	Operators	Managers
Key Focus	KPIs/Trends	Exploration	Alerts/Status	Action Plans

Types of Dashboards

1. **Strategic Dashboards**
    - **Purpose**: High-level monitoring of KPIs and long-term goals.
    - **Users**: Executives and senior leaders.
    - **Features**: Show overall business health; updated periodically (daily/weekly).
    - **Example Widgets**: Revenue trend, profit margins, customer churn.
2. **Analytical Dashboards**
    - **Purpose**: Enable exploration of data, uncover trends, and perform diagnostics.
    - **Users**: Analysts, data scientists, business intelligence teams.
    - **Features**: Interactive filters, drill-downs, complex visualizations (heat maps, scatter plots).
    - **Common Uses**: Forecasting, root cause analysis, segmentation studies.
3. **Operational Dashboards**
    - **Purpose**: Monitor real-time or near real-time operations.
    - **Users**: Managers, supervisors, front-line teams.
    - **Features**: Highly time-sensitive, auto-refreshing, alert-based or threshold-driven.

- **Example**: System outage alerts, order fulfillment status.

4. **Tactical Dashboards (Hybrid)**

   - **Purpose**: Guide mid-term decision-making by combining strategic and operational elements.
   - **Users**: Mid-level managers, department heads.
   - **Features**: Blend of summary KPIs and detailed performance metrics, supporting short- to medium-term planning and adjustments.
   - **Example**: Marketing campaign performance dashboard.

Dashboard for Usability

- **Organize the Information**: Support the meaning and use of data through visual clarity.
- **Consistency**: Use consistent colors and design elements for easy interpretation.
- **Aesthetically Pleasing**: Design with acceptance and accessibility in mind. ● **User-Driven Narrative**: Include drill-down and interactivity features. ● **Usability Testing**: Prototype design and test usability.

Dashboard Utilities

- **Low or No-Code Tools**:
  - Tableau
  - Power BI
  - Google Data Studio
- **Coding-Based Tools**:
  - Python
  - Dash (Plotly)
  - Voila (Jupyter notebooks)
  - R Packages
  - Shiny (R Studio)
  - Flexdashboard

1. The Power and Science of Storytelling

Storytelling is described as one of the oldest professions and remains a critical tool for communication in modern business contexts.

- **Why it Matters:**
  - **Retention:** Humans are naturally "addicted" to stories. We retain stories significantly better than raw data.
  - **Persuasion:** According to the Harvard Business Review (2020), storytelling allows leaders to effectively persuade teams, clients, and audiences.

    - **Example:** People can easily recall detailed plot points of their favorite movie but struggle to remember details from their company's last annual report.
- **The Science Behind It:**
    - **Cognitive Value:** A structured story aids memory recall.
    - **Brain Activation:** Stories activate various centers of the brain, making the content more immersive.
    - **Jerome Bruner's Theory:** Facts are **22 times more memorable** when they are wrapped in a narrative.
    - **Emotional Connection:** A compelling story creates an emotional relation, which helps the audience remember the information practically.

2. Visual Storytelling in Action

Visual storytelling combines data with narrative to shape public perception and understanding.

- **Real-World Example (U.S. Elections):**
    - In the 2000 U.S. election, media outlets utilized color-coded maps (Red for Republican, Blue for Democrat) to visualize results.
    - These data-infused visuals did not just display numbers; they explained outcomes, shifted perceptions, and helped build a national narrative regarding the political landscape.

3. Frameworks for Effective Presentations

The lecture outlines specific models to ensure presentations are impactful rather than cluttered.

A. Presentation Zen (Garr Reynolds)

This approach emphasizes simplicity and meaning.

- **Restraint:** Eliminate all unnecessary elements from the slide.
- **Simplicity:** Leverage whitespace and stick to **one idea per slide**.
- **Naturalness:** Deliver the presentation with an authentic voice and maintain good eye contact.

B. The SUCCESS Model (Heath Brothers)

To make ideas "stick," presentations should follow the SUCCESS acronym:

- **S**imple
- **U**nexpected
- **C**oncrete
- **C**redible
- **E**motional
- **S**toried
    - **Outcome:** Using this model makes presentations engaging, focused, and memorable.

C. Pecha Kucha (The Art of Visual Brevity)

A Japanese presentation style designed to prevent data overload.

- **The Format:** 20 slides shown for 20 seconds each.
- **Total Time:** Approximately 6 minutes and 40 seconds.
- **Philosophy:** "Talk less, show more." It forces concise communication with strong visual support rather than text-heavy slides.

4. "Death by Presentation": A Case Study on Failure

"Death by Presentation" refers to disengagement caused by long, cluttered, and bullet-heavy slides. Overusing fonts and text leads to audience confusion.

Case Study: NASA Columbia Shuttle Disaster (2003)

A misleading PowerPoint presentation is partly attributed to the communication failure regarding the shuttle's risk.

- **The Flaws in the Slide:**
  - **Clutter:** The slide was filled with dense bullet points, multiple font sizes, and long sentences.
  - **Vague Language:** The words "Significant" and "Significantly" were used repeatedly but did not refer to "statistical significance" in a technical sense, leading to ambiguity.
  - **Poor Hierarchy:** The most critical safety warning—that the flight condition was significantly outside the test database—was written in the **smallest font** at the bottom of the slide.
  - **Misleading Title:** The title "Review of Test Data Indicates Conservatism" downplayed the actual danger.
- **How it Should Have Been Fixed:**
  - **Better Headline:** A direct headline like "Foam strike is 600x stronger than tested safety limit".
  - **Visual Hierarchy:** Prioritizing risk information visually so it isn't buried. ○ **Clear Visuals:** Using images to show potential damage scenarios rather than just text.

5. Summary of Key Takeaways

● **Storytelling** is a tool for retention and persuasion, not just entertainment. ● **Simplicity** (Presentation Zen) and **Structure** (SUCCESS model) are essential for clarity. ● **Visual Hierarchy** prevents critical information from being lost (as seen in the NASA example).
● **Brevity** (Pecha Kucha) helps avoid the common pitfall of "Death by PowerPoint."

1. The Importance of Business Storytelling

Storytelling in business is not just about presenting data; it is about bridging the gap between logic and emotion to drive decision-making.

- **Logic vs. Emotion:** Even in data-heavy environments, business decisions are often emotion-driven. Storytelling provides the necessary bridge to make data persuasive and clear.
- **Example - The Significant Objects Project:** A project demonstrated that adding a narrative to cheap items (bought for $1.25) allowed them to be sold for significantly higher prices (up to $8,000). This illustrates that narrative adds value and meaning to raw objects or data.
- **Key Insight:** To influence people to make the right decisions with data, you must "get in their head," which is achieved through stories.

2. Defining Data Storytelling

Data storytelling is defined as the art of communicating insights in a way that makes data actionable and memorable. It is a combination of three elements:

1. **Data Visualization:** Making the information visually clear.
2. **Narrative:** The storyline that guides the audience.
3. **Context:** The setting or background that makes the data relevant to the audience.

Historical Example: The 1854 Cholera outbreak map of London's Broad Street is a classic example of using data visualization to tell a story that solved a health crisis.

3. Storytelling Frameworks

To structure a narrative effectively, specific frameworks are often used in persuasive reports and presentations.

- **Monomyth (Hero's Journey):** Involves a journey consisting of a crisis and a resolution guided by a mentor/figure.
- **Story Mountain:** Follows a path of Beginning Conflict Climax Deflation Resolution.
- **Nested Loops:** Involves telling stories within stories, often used to explain transformational changes in business.

4. Types of Narratives in Data Visualization

Narratives are categorized by how much control the author has versus the reader.

A. Author-Driven (Linear)

- **Characteristics:** Linear and static with a strong, prescribed message.
- **Format:** Magazine infographics, static presentation decks, videos.
- **Usage:** When the presenter wants to guide the audience through a specific conclusion without deviation.

B. Reader-Driven (Exploratory)

- **Characteristics:** Interactive and exploratory. The audience discovers their own insights.
- **Format:** Interactive dashboards (e.g., Tableau).
- **Usage:** When the user needs to filter, zoom, or search through data to find answers relevant to them.

C. Hybrid (Interactive Storytelling)

- **Characteristics:** Combines guided structure with interactive elements.
- **Techniques:**
    - **Martini Glass:** Starts with a tight, author-driven narrative (the stem) and opens up to free exploration (the glass).
    - **Drill-down:** Users can click into specific data points for more detail.
    - **Interactive Slideshow:** A slide deck that allows for embedded interaction.

5. The Seven Types of Data Stories

There are seven primary ways to frame a data story. The lecture uses a dataset regarding "Crimes in India (2001-2014)" to illustrate these types.

1. Change Over Time

- **Objective:** To show how values and trends evolve across specific time frames. **Example:** Analyzing crime data from 2001 to 2014 showed an initial downward trend until 2003, followed by a steady increase across all crime groups. However, the story highlights a specific sharp rise in crimes against women during this development phase.

2. Drilling Down

- **Objective:** To move from a general overview to specific details using hierarchies (e.g., Country State City).
- **Example:** Starting with national crime data, one might drill down to state level. Normalizing for population reveals Delhi as the "Crime Capital." Further drilling down into districts identifies that specific areas (like Outer, East, and West districts) contribute to more than 55% of crimes against women in the city.

3. Zooming Out

- **Objective:** To broaden the view to understand the larger context or regional implications.
- **Example:** Zooming out from a single state to look at the whole map of India reveals that three states in the East and North-East regions are among the top 5 contributors for crimes against women.

4. Contrast

- **Objective:** To highlight differences between two or more groups to find disparities. **Example:** Comparing Delhi with neighboring states like Haryana. While Delhi has a

higher overall crime rate (8.4% vs 3.8%), a contrast reveals that Haryana leads specifically in homicide. Furthermore, contrasting demographics shows Haryana has a massive rural population (66.1%) compared to Delhi's small rural population (2.5%), providing context for the crime statistics.

5. Intersection

● **Objective:** To highlight when one category overtakes another (a "crossover" point). ● **Example:** In 2001, homicide rates were higher than crimes against women. The story focuses on the intersection point around 2012, where the order swapped, and crimes against women began contributing more to the total than homicide.

6. Factors

- **Objective:** To check for correlations or identify variables that have the greatest influence on a metric.
- **Example:** Using a waterfall chart to analyze year-over-year changes. It highlights that in 2014, the total number of crimes actually reduced by 6.9% compared to 2013, breaking the previous trend.

7. Outliers

● **Objective:** To identify anomalies or data points that deviate significantly from the norm. ● **Example:** When plotting normalized crime counts, a scatter plot may show a massive cluster of normal data points, with one distinct point far above the rest. In the example data, Delhi appears as a significant outlier specifically in the "Property Stolen" crime group.

I. The Psychology of Visual Perception

Understanding why charts mislead begins with how the human brain processes visual information.

- **System 1 Processing:** Our brains make unconscious, snap judgments based on visuals (Kahneman's System 1).
- **Pre-attentive Attributes:** Charts rely on attributes like color, shape, position, and orientation to catch the eye immediately.
- **Violating Conventions:** If a visual goes against the "natural order" or standard conventions, it deceives the viewer's intuition.
    - *Example:* A chart depicting gun deaths in Florida inverted the y-axis (placing 0 at the top and the maximum value at the bottom). Visually, the line looked like it was going down (decreasing deaths), but statistically, the numbers were rising. This misled viewers into thinking fatalities decreased after the "Stand Your Ground" law was enacted.

II. Axis Manipulation

One of the most common ways to distort data is by manipulating how the axes are presented.

- **Truncated Y-Axis:** Starting the y-axis at a non-zero number (e.g., starting at 34% instead of 0%) makes small differences appear massive.

    - *Example:* A chart showing the expiration of Bush Tax Cuts exaggerated a difference between 35% and 39.6% by cutting off the bottom of the chart, making the increase look catastrophic rather than incremental.
- **Zero-Baseline Omission:** Omitting the zero-baseline exaggerates the visual difference between bar lengths.
- **Inappropriate Zero-Start:** Conversely, sometimes starting at zero *is* misleading if it obscures meaningful shifts, such as in global temperature graphs where small fractional changes have massive impact.
    - *Visual Cue:* Always check the axis scale and starting point to assess "visual honesty".

III. Scaling and Dual Axes

Comparing two different datasets on the same chart can be highly misleading if the scales are not synchronized.

- **Dual Axes Manipulation:** Using two different y-axes (one on the left, one on the right) can manipulate trends, making unrelated scales look related or creating false intersections.
    - *Example:* A chart regarding Planned Parenthood plotted "Abortions" against "Cancer Screening Services." Because the two metrics were on different scales (hundreds of thousands vs. millions) but plotted on the same visual space, the lines crossed to imply a dramatic reversal of priorities, even though the raw numbers told a different story.
- **Recommendation:** Avoid dual-axis charts unless the scales are clearly marked and related.

IV. Logical Fallacies in Data

Visualizations can be technically accurate but logically deceptive.

1. Correlation ≠ Causation

A chart may show that two variables move together, but this does not imply one causes the other.

● **Confounding Variables:** Often, a third, unseen variable drives both metrics. ○ *Example:* Ice cream sales and murder rates rise at the same time. Neither causes the other; both are driven by a third variable: hot weather (Summer). ● **Spurious Correlations:** Statistically linked data that are completely unrelated in reality. ○ *Example:* A graph showing a nearly identical trend line for "Number of people who drowned in a pool" and "Films Nicolas Cage appeared in." The visual correlation is perfect, but there is no logical connection.

2. Simpson’s Paradox

A trend present in individual groups can disappear or reverse when the groups are combined (aggregated).

- **Cause:** This happens when data is aggregated without considering the different sizes of

the groups.

- **Example:** One person might answer fewer questions correctly each day compared to another, but because they answer a higher volume of questions total, their overall percentage might look higher.
- **Strategy:** Always examine subgroup distributions before drawing conclusions from aggregated data.

3. Drill Down Bias

The order in which you filter data can change the insight you derive.

- **Example:** In 2016 US election voter data:
    - Filtering by **Gender** first showed Hillary Clinton ahead.
    - Filtering by **Ethnicity** first revealed that ethnicity was actually the dominant factor driving the trend.
- **Tip:** If child-group distributions do not resemble parent groups, bias is likely present.

V. Design and Selection Errors

- **Cherry-Picking:** The practice of selectively excluding data points that contradict the narrative.
    - *Example:* A chart showing UK National Debt rising sharply might only show data from 1995 to 2016. However, looking at the long-term timeline (starting from 1910) shows that the debt was historically much higher in the 1940s, changing the narrative completely.
- **Decorative Deception (The "Lie Factor"):** When the visual elements (size of bars, bubbles, or slices) do not mathematically match the numbers they represent. ○ *Example:* A pie chart where a slice representing 39.8% is drawn significantly larger than a slice representing 60.2%.
    - *Example:* A chart on Diesel prices where the graphical bars were sized arbitrarily, not proportional to the monetary increase, exaggerating the price hike. ● **Misuse of Color Saturation:** Using colors counter-intuitively can confuse the reader. ○ *Example:* A map of COVID-19 cases used a color palette where the "second highest" case load category was a bright, alarming red, while the "highest" category was a darker, duller maroon. This made the second-highest areas pop out more than the worst-affected areas.

Lecture Notes: Introduction to Matplotlib (with Python & NumPy)

1. What is Matplotlib?

Matplotlib is a powerful plotting library in Python used to create:

Static plots (images)

Animated plots

Interactive visualizations

It works very well with other libraries like **NumPy** and **pandas**, and is widely used for:

Data analysis & exploration

Scientific computing

Creating charts and graphs for reports & presentations

You'll mainly use the submodule `matplotlib.pyplot`, commonly imported as `plt`.

2. Why Use Matplotlib? (Merits)

Some key advantages:

Versatile

Supports line plots, bar charts, scatter plots, histograms, pie charts, etc.

Highly customizable

Control colors, line styles, fonts, figure size, axes, titles, labels, grids, legends, and much more.

Works well with NumPy & pandas

You can directly plot arrays or Series.

Widely used & well-documented

Lots of tutorials, examples, and community support.

3. Basic Setup

Before plotting, we usually import:

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

```
import numpy as np

%matplotlib inline # (Jupyter only) show plots inside the notebook
```

`matplotlib.pyplot` is imported as `plt` by convention.

`numpy` (imported as `np`) is used to generate and manipulate numerical data.

`%matplotlib inline` is a **Jupyter magic command** that makes plots appear directly in the notebook.

4. Creating a Simple Plot (Sine Wave Example)

4.1 Prepare the Data

Use NumPy to create data points:

```
x = np.linspace(0, 10, 100) # 100 points between 0 and 10
y = np.sin(x) # sine of each x value
```

`np.linspace(start, stop, num)` generates evenly spaced values.

Here, `y` is the sine of `x`, so we will get a nice wave.

4.2 Create the Figure and Axes

```
fig, ax = plt.subplots()
```


`fig` is the **Figure** (the whole canvas).

`ax` is the **Axes** (the actual plot area where the data is drawn).

4.3 Plot the Data

```
ax.plot(x, y, label='Sine')
```

`ax.plot(x, y)` draws a line plot.

`label='Sine'` will be used if we add a legend.

4.4 Customize the Plot

```
ax.set_title("Simple Sine Wave")
ax.set_xlabel("X-Axis")
ax.set_ylabel("Y-Axis")
```

`set_title()` → adds a title at the top.

`set_xlabel()` and `set_ylabel()` → label the axes.

You can also add:

```
ax.grid(True) # add a grid
ax.legend() # show labels for plotted lines
```

4.5 Show the Plot

```
plt.show()
```

This renders the final figure.

What Happens When You Uncomment?

1. **Add the cosine line**


```
ax.plot(x, y2, color='red', linestyle='--', label='Cosine')
```

Adds a second line (cosine) in red with a dashed style.

Now you'll see both sine and cosine on the same axes.

2. **Add the grid**

```
ax.grid(True)
```

Draws a grid in the background.

Helps in reading exact values and understanding the shape.

3. **Add the legend**

```
ax.legend()
```

Shows a small box (usually top-right) labeling each line using the `label=` arguments you provided.

This is a great way to learn: **change one thing at a time and re-run the cell to see the effect.**

6. Saving the Plot

You can save your plot to an image file.

6.1 Basic Save Command

If you have a `fig` object:

```
fig.savefig('my_plot.png')
```

Saves the current figure as **`my_plot.png`** in your working directory.

You can also save as `.jpg` , `.pdf` , `.svg` , etc.

Examples:


```
fig.savefig('sine_wave.jpg')
fig.savefig('sine_and_cosine.pdf')
```

In many environments (like Jupyter), there's also a **save button** in the toolbar, but `savefig()` is more flexible and scriptable.

Lecture 22 Notes : Decorating Graphs with Plot Styles &

Types

1. Colors & Colormaps

Ways to specify color:

- Default cycle: `'C0'` , `'C1'` , ...
- Short codes: `'r'` , `'g'` , `'b'` , `'k'` , ...
- Named: `'skyblue'` , `'salmon'` , `'limegreen'` , ...
- Hex: `'#FF6347'`
- RGB(A): `(0.2, 0.4, 0.6[, alpha])`

```
plt.plot(x, y, color='C1', linewidth=3)
plt.plot(x, y2, color='#FF6347', linewidth=4)
```

Colormaps:

- Good defaults: `'viridis'` , `'plasma'` , `'inferno'` , `'coolwarm'`
- Use with continuous data:

```
plt.scatter(x, y, c=y, cmap='viridis')
plt.colorbar()
```

2. Line & Marker Styles

Lines:

- `linewidth` / `lw`
- `linestyle` / `ls` : `'-'` , `'--'` , `'-.'` , `':'`
- `alpha` for transparency

Markers:

marker='o' , 's' , '^' , 'x' , ...

markersize / ms

```
plt.plot(
 x, y,
 linestyle='--', linewidth=2,
 marker='o', markersize=6
)
```

For scatter:

```
plt.scatter(x, y, s=40, c='C0', alpha=0.7)
```

3. Grids, Spines & Axes

Grid:

```
fig, ax = plt.subplots()
ax.plot(x, y)
ax.grid(True, linestyle=':', alpha=0.5)
```

Spines (borders):

```
for spine in ['top', 'right']:
 ax.spines[spine].set_visible(False)

ax.spines['left'].set_linewidth(1.5)
ax.spines['bottom'].set_color('gray')
```

Axis limits & ticks:

```
ax.set_xlim(0, 10)
ax.set_ylim(-1.5, 1.5)
```

```
ax.set_xticks([0, 2, 4, 6, 8, 10])
ax.set_xticklabels(['0', '2', '4', '6', '8', '10'])
```

4. Titles, Labels, Legends, Annotations

```
ax.set_title('Trigonometric Functions')
ax.set_xlabel('x')
ax.set_ylabel('y')

ax.plot(x, y, label='Sine')
ax.plot(x, y2, label='Cosine')
ax.legend(loc='upper right')
```

Annotation:

```
ax.annotate(
 'Peak',
 xy=(np.pi/2, 1),
 xytext=(2, 1.2),
 arrowprops=dict(arrowstyle='→')
)
```

5. Common Plot Types

Bar:

```
plt.bar(categories, values, color='skyblue')
```

Pie:

```
plt.figure(figsize=(5, 5))
plt.pie(sizes, labels=labels, autopct='%1.1f%%')
```

Histogram:

```
plt.hist(data, bins=20, density=True, alpha=0.7)
```

Box & Violin:

```
plt.boxplot([data1, data2])
plt.violinplot([data1, data2], showmeans=True)
```

Scatter with encodings:

```
plt.scatter(x, y, c=z, s=area, cmap='viridis', alpha=0.8)
plt.colorbar(label='z value')
```

6. Higher-Level APIs

Pandas plotting:

```
ax = df.plot(kind='bar')
ax.set_title('Users by Platform')
```

`df.plot(...)` supports: `line` , `bar` , `hist` , `area` , `hexbin` , etc.

Seaborn (conceptual):

Easier statistical plots, e.g. `sns.boxplot` , `sns.violinplot` , `sns.heatmap` .

7. Subplots / Layouts

```
fig, axes = plt.subplots(1, 2, figsize=(10, 4), sharey=True)

axes[0].scatter(x, y, c=z, cmap='viridis')
axes[0].set_title('Random Scatter')
```

```
axes[1].bar(categories, values, color='skyblue')
for spine in ['top', 'right']:
```



```
 axes[1].spines[spine].set_visible(False)
axes[1].set_title('Category Comparison')

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

`plt.subplots(r, c, ...)` → `fig, axes`

`sharex` , `sharey` to sync axes

`plt.tight_layout()` to reduce overlap

8. Financial & 3D (Brief)

Financial charts (candlesticks):

```
import mplfinance as mpf
mpf.plot(ohlc_df, type='candle', volume=True, style='yahoo')
```

3D surface:

```
from mpl_toolkits.mplot3d import Axes3D

fig = plt.figure(figsize=(6, 5))
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X, Y, Z, cmap='viridis')
ax.set_title('3D Surface')
```

Lecture 23 Notes : Advanced Matplotlib & Data from the Web

1. Data Formats & Reading from URLs

In real projects, data often comes from **remote sources** instead of local files.

Common formats:

- **CSV** (Comma-Separated Values)
  - Text file: one row per line, values separated by commas
  - Read with: `pandas.read_csv(url_or_path)`
- **JSON** (JavaScript Object Notation)
  - Hierarchical / nested structure
  - Read with: `pandas.read_json(url_or_path)`

Key idea: Pandas can read **directly from a URL**, so you don't have to manually download files.

```
import pandas as pd
```

```
csv_url = "https://example.com/data.csv"
df = pd.read_csv(csv_url) # For CSV

json_url = "https://example.com/data.json"
df_json = pd.read_json(json_url) # For JSON
```

2. Standard Imports & Style

For advanced plotting, you typically need:

```
import numpy as np
import pandas as pd
```



```
import matplotlib.pyplot as plt

# Consistent, clean look
plt.style.use('seaborn-v0_8-whitegrid')
```

- `numpy` → numerical arrays, math
- `pandas` → tabular data, time series
- `matplotlib.pyplot` → plotting

A global style (like `seaborn-whitegrid`) makes all plots look nicer by default.

3. Fetching & Plotting Online Weather Data

Example: **Seattle weather** from the Vega datasets repo (online CSV).

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

url =
'https://raw.githubusercontent.com/vega/vega-datasets/main/data/seattle-weather.csv'
```

```
# Read CSV and parse 'date' as a datetime column
df_weather = pd.read_csv(url, parse_dates=['date'])

# Use the date as the index (good for time-series plots)
df_weather.set_index('date', inplace=True)

# Plot daily maximum temperature
plt.figure(figsize=(12, 6))
df_weather['temp_max'].plot(color='crimson')

plt.title('Maximum Daily Temperature in Seattle', fontsize=16)
plt.ylabel('Temperature (°C)')
plt.xlabel('Date')
plt.show()
```


Concepts:

- `parse_dates=['date']` → converts strings to `datetime`.
- `set_index('date')` → makes time-based plots easier.
- Pandas Series `.plot()` integrates nicely with Matplotlib.

4. Multi-Plot Layouts with `plt.subplots`

For multiple related plots in **one figure**, use subplots.

```
fig, axes = plt.subplots(2, 1, figsize=(12, 8), sharex=True)
```

- `fig` → the overall figure (canvas)
- `axes` → array of Axes objects (individual plots)
- `2, 1` → 2 rows, 1 column
- `sharex=True` → share the same x-axis (dates align across plots)

Example: Max temperature and precipitation

```
fig, axes = plt.subplots(2, 1, figsize=(12, 8), sharex=True)

# First subplot: Maximum temperature
axes[0].plot(df_weather.index, df_weather['temp_max'], color='crimson')
axes[0].set_title('Maximum Temperature Trend')
axes[0].set_ylabel('Temp (°C)')

# Second subplot: Daily precipitation
axes[1].plot(df_weather.index, df_weather['precipitation'], color='royalblue')
axes[1].set_title('Daily Precipitation')
axes[1].set_ylabel('Precipitation (mm)')

# Overall figure title and layout
fig.suptitle('Seattle Weather Analysis', fontsize=16)
```



```
plt.tight_layout(rect=[0, 0, 1, 0.96]) # leave space for suptitle
plt.show()
```

Concepts:

- `axes[0]` , `axes[1]` → select specific subplot
- `fig.suptitle()` → title for the **whole figure**
- `plt.tight_layout()` → fixes overlapping labels and titles

Another common pattern (min temperature + wind speed):

```
fig, axes = plt.subplots(2, 1, figsize=(12, 8), sharex=True)

axes[0].plot(df_weather.index, df_weather['temp_min'], color='deepskyblue')
axes[0].set_title('Minimum Daily Temperature')
axes[0].set_ylabel('Temp (°C)')
```

```
axes[1].plot(df_weather.index, df_weather['wind'], color='slategray')
axes[1].set_title('Average Daily Wind Speed')
axes[1].set_ylabel('Speed (m/s)')

fig.suptitle('Seattle Weather Metrics', fontsize=16)
plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()
```

5. Curve Fitting & Extrapolation with NumPy

To visualize **future trends**, you can fit a curve to existing data and extend it.

Key functions:

- `np.polyfit(x, y, deg)` → returns polynomial coefficients (degree `deg`)
- `np.poly1d(coeffs)` → turns coefficients into a callable polynomial function

Lecture 24 Notes

1. What is GeoPandas?

GeoPandas extends pandas to work with **geospatial data**:

- Uses a `GeoDataFrame` (like `DataFrame` + geometry column)
- `geometry` column holds shapes: **points**, **lines**, or **polygons**
- Typical data sources:
  - **Shapefiles** (`.shp`)
  - **GeoJSON** files

Plotting is simple:

```
world = gpd.read_file("countries.geojson")
world.plot() # Matplotlib handles the rendering
```

2. Imports & Installation Considerations

GeoPandas has extra dependencies (e.g., PROJ, GDAL), so installation may be heavier.

Example import with a helpful error message:

```
try:
 import geopandas as gpd
 import matplotlib.pyplot as plt
 import requests
except ImportError:
 print("Required packages (geopandas, matplotlib, requests) are not installe
d.")
 print("Install with `pip install geopandas matplotlib requests`")
 raise
```


3. Checking a Remote GeoJSON URL

Before reading a remote GeoJSON file, it’s good to check if the URL is

```
accessible. import requests

def check_url(url):
 try:
 response = requests.head(url)
 if response.status_code == 200:
 return True
 else:
 print(f"URL {url} returned status code {response.status_code}")
```

```
    return False
    except requests.RequestException as e:
    print(f"Error accessing URL {url}: {e}")
    return False
```

4. Loading and Plotting a World Map

Example: load world boundaries from a hosted GeoJSON and plot them.

```
import geopandas as gpd
import matplotlib.pyplot as plt

url = "https://raw.githubusercontent.com/datasets/geo-boundaries-world-110
m/master/countries.geojson"

if not check_url(url):
 raise Exception("GeoJSON URL is not accessible.")

# Read GeoJSON into a GeoDataFrame
world = gpd.read_file(url)

# Basic world map
fig, ax = plt.subplots(1, 1, figsize=(15, 10))
```



```
world.plot(ax=ax, color='lightgray', edgecolor='black')

ax.set_title('World Map (GeoPandas)')
ax.set_xlabel('Longitude')
ax.set_ylabel('Latitude')
plt.show()
```

Concepts:

`gpd.read_file(url)` understands many geospatial formats, including GeoJSON.

Each row is a country polygon with associated attributes (name, etc.).

Plotting uses longitude/latitude coordinates.

5. Overlaying Point Data (e.g., Cities)

You can overlay **points** (cities, sensors, events) on the base map.

```
cities_data = {
 'City': ['New Delhi', 'New York', 'Tokyo'],
 'Latitude': [28.6, 40.7, 35.7],
 'Longitude': [77.2, -74.0, 139.7]
}

# Create GeoDataFrame of city points
gdf_cities = gpd.GeoDataFrame(
 cities_data,
 geometry=gpd.points_from_xy(cities_data['Longitude'], cities_data['Latitud e'])
)

# Plot base map
fig, ax = plt.subplots(1, 1, figsize=(15, 10))
world.plot(ax=ax, color='lightgray', edgecolor='black')

# Overlay cities
gdf_cities.plot(ax=ax, marker='o', color='red', markersize=50, label='Major Cit
```



```
   ies')

   ax.set_title('World Map with Major Cities (GeoPandas)')
   ax.set_xlabel('Longitude')
   ax.set_ylabel('Latitude')
   plt.legend()
   plt.show()
```

Concepts:

`gpd.points_from_xy(longitude, latitude)` → converts coordinates to point geometries.

Multiple `GeoDataFrame` s can be plotted on the **same Axes** (`ax=ax`).

Symbolization (marker, color, size) controls how points appear.

6. Error Handling & Alternative Sources

The code also demonstrates:

Using `try/except` to catch errors when loading remote GeoJSON.

Printing **alternative URLs** if the first one fails.

This pattern is useful whenever your data source may be unavailable.

Lesson 25 Notes:

1. Matplotlib vs Seaborn :Core Idea

Matplotlib

Low-level, very flexible.

Works with arrays/lists.

You handle most stats and styling manually.

Seaborn

Built on top of Matplotlib.

Works directly with **pandas DataFrames**.

Designed for **statistical plots** (aggregates, CIs, regression).

Better-looking defaults (colors, grids, styles).

2. Basic Syntax Difference

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

# Matplotlib
plt.scatter(df['study_hours'], df['exam_score'])
plt.title('Matplotlib')
plt.xlabel('Study Hours')
plt.ylabel('Exam Score')
plt.show()

# Seaborn
sns.scatterplot(data=df, x='study_hours', y='exam_score')
plt.title('Seaborn')
plt.xlabel('Study Hours')
```



```
plt.ylabel('Exam Score')
plt.show()
```

Matplotlib: pass arrays/Series.

Seaborn: pass `data=` and column names.

3. Aesthetics & Defaults

Matplotlib:

```
plt.hist(df['exam_score'], edgecolor='black')
```

Seaborn:

```
sns.histplot(data=df, x='exam_score')
```

Seaborn automatically applies nicer styles (grid, colors, etc.).

4. Statistical Aggregation

Matplotlib (manual mean per category):

```
course_means = df.groupby('course')['exam_score'].mean()
plt.bar(course_means.index, course_means.values)
```

Seaborn (automatic mean + CI):

```
sns.barplot(data=df, x='course', y='exam_score')
```

barplot computes mean and shows confidence intervals by default.

5. Combining Seaborn + Matplotlib


```
sns.boxplot(data=df, x='course', y='exam_score')
plt.title('Exam Scores by Course')
plt.axhline(df['exam_score'].mean(), color='r', linestyle='--')
plt.show()
```

Seaborn draws main visualization.

Matplotlib adds titles, lines, annotations.

6. Example with Built-in Seaborn Data (fmri)

Simple line plot:

```
fmri = sns.load_dataset("fmri")

sns.lineplot(data=fmri, x="timepoint", y="signal")
plt.title("fMRI Signal Over Time")
plt.show()
```

Add extra dimensions:

```
sns.lineplot(
 data=fmri,
 x="timepoint", y="signal",
 hue="region", style="event"
)
```

Facet by region:

```
sns.relplot(
 data=fmri,
 x="timepoint", y="signal",
 hue="event", col="region",
 kind="line"
)
```


Lesson 26 Notes:

Setup

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

sns.set_theme(style="darkgrid") # nice default look
```

1. Visualizing Statistical Relationships with relplot

1.1 relplot : figure-level wrapper

`sns.relplot()` = figure-level function built on **FacetGrid** .

Uses axes-level functions under the hood:

`scatterplot()` → `kind="scatter"` (default)

`lineplot()` → `kind="line"`

Basic scatter:

```
tips = sns.load_dataset("tips")

sns.relplot(
 data=tips,
 x="total_bill",
 y="tip"
)
```

1.2 Semantic mappings: hue , style , size

Color (hue):

```
sns.relplot(
 data=tips,
 x="total_bill", y="tip",
 hue="smoker" # different colors by category
)
```

Marker style:

```
sns.relplot(
 data=tips,
 x="total_bill", y="tip",
 hue="smoker",
 style="smoker" # different markers by category
)
```

Hue + style (two variables):

```
sns.relplot(
 data=tips,
 x="total_bill", y="tip",
 hue="smoker",
 style="time"
)
```

Size:

```
sns.relplot(
 data=tips,
 x="total_bill", y="tip",
 size="size"
)

sns.relplot(
 data=tips,
```

```
 x="total_bill", y="tip",
 size="size", sizes=(15, 200) # control size range
)
```

1.3 Line plots (`kind="line"`), aggregation & uncertainty

For time series / ordered data:

```
dowjones = sns.load_dataset("dowjones")

sns.relplot(
 data=dowjones,
 x="Date", y="Price",
 kind="line"
)
```

By default, multiple observations per x are **aggregated** (e.g., mean).

A shaded band shows **95% confidence interval** of the estimate.

Control estimator & CI:

```
sns.relplot(
 data=tips,
 x="size", y="tip",
 kind="line",
 estimator=np.mean,
 ci=95
)
```

2. Categorical Relationships with `catplot`

`catplot` is a figure-level function that wraps different categorical plot types.

```
sns.catplot(
```

```
 data=tips,
```



```
 x="day", y="total_bill",
 kind="strip" # default
)
```

2.1 Categorical scatter: strip & swarm

Strip plot (jittered points):

```
sns.catplot(
 data=tips,
 x="day", y="total_bill",
 kind="strip", jitter=True
)
```

Swarm plot (non-overlapping points):

```
sns.catplot(
 data=tips,
 x="day", y="total_bill",
 kind="swarm"
)
```

Add hue:

```
sns.catplot(
 data=tips,
 x="day", y="total_bill",
 hue="sex",
 kind="swarm"
)
```

Control category order / swap axes:

```
sns.catplot(
 data=tips,
 x="smoker", y="tip",
```



```
 order=["No", "Yes"]
)

sns.catplot(
 data=tips,
 x="total_bill", y="day",
 hue="time",
 kind="swarm"
)
```

2.2 Categorical distributions: box / boxen / violin

Boxplot:

```
sns.catplot(
 data=tips,
 x="day", y="total_bill",
 kind="box"
)

sns.catplot(
 data=tips,
 x="day", y="total_bill",
 hue="smoker",
 kind="box"
)
```

Boxenplot (for larger datasets):

```
diamonds = sns.load_dataset("diamonds")
```

```
sns.catplot(
 data=diamonds.sort_values("color"),
 x="color", y="price",
```



```
 kind="boxen"
)
```

Violin plot (distribution shape):

```
sns.catplot(
 data=tips,
 x="day", y="total_bill",
 kind="violin"
)
```

2.3 Categorical estimates: bar, count, point

Bar plot (mean + CI):

```
titanic = sns.load_dataset("titanic")

sns.catplot(
 data=titanic,
 x="sex", y="survived",
 hue="class",
 kind="bar"
)
```

Count plot (frequency):

```
sns.catplot(
 data=titanic,
 x="deck",
```

```
 kind="count"
)

sns.catplot(
 data=titanic,
 y="deck",
 hue="class",
```



```
 kind="count",
 palette="pastel",
 edgecolor=".6"
)
```

Point plot (mean + CI, emphasizes slopes / interactions):

```
sns.catplot(
 data=titanic,
 x="sex", y="survived",
 hue="class",
 kind="point"
)
```

3. Distributions with displot , histplot , kdeplot

3.1 1D histograms & binning

```
penguins = sns.load_dataset("penguins")

sns.displot(penguins, x="flipper_length_mm") # default
sns.displot(penguins, x="flipper_length_mm", binwidth=3)
sns.displot(penguins, x="flipper_length_mm", bins=20)
```

Discrete variables:

```
tips = sns.load_dataset("tips")

sns.displot(tips, x="size") # not ideal bins
sns.displot(tips, x="size", discrete=True)
sns.displot(tips, x="day", shrink=.8)
```

3.2 Conditioning on other variables

Multiple subsets via `hue`:


```
sns.displot(
 penguins,
 x="flipper_length_mm",
 hue="species"
)
```

Control how bars are drawn:

```
sns.displot(
 penguins,
 x="flipper_length_mm",
 hue="species",
 multiple="stack" # or "dodge", "fill", "layer"
)

sns.displot(
 penguins,
 x="flipper_length_mm",
 hue="sex",
 multiple="dodge"
)
```

Facet by another variable:

```
sns.displot(
```

```
 penguins,
 x="flipper_length_mm",
 col="sex"
)
```

3.3 Normalized histograms & statistics

```
sns.displot(
 penguins,
```