Project 01: Data Visualization with Python

CSC10108 - Data Visualization

March 8, 2025

Contents

1 Project Objectives

This project aims to equip students with the foundational skills necessary for effective data visualization using Python. By learning and leveraging Python libraries such as Matplotlib, Seaborn, and Pandas to create a wide range of static, interactive, and animated visualizations, the students explore the hidden relationship in your focusing dataset in a specific domain.

2 Notes and Constraints

You must follow these constraints when doing this project:

- Work without a report will not be graded.
- Members who do not contribute to the project will not receive points.
- Reference sources (if any) need to be fully recorded in the report in the References section. Note that it is necessary to distinguish between referencing and plagiarism.
- Individuals or groups that commit cheating and dishonesty will receive 0 points in the course.
- The submission must compressed by using zip. We DO NOT ALLOW other types. If the size is $>$ 20MB, upload it to an external storage service such as Google Drive or OneDrive, and then submit the link. Last but not least, please keep the link public for at least 2 years.

3 Tasks

In this lab, you will leverage Kaggle to identify and acquire a publicly available dataset aligned with a topic of interest selected by your student group. The chosen dataset must be structured in a tabular format, containing a minimum of **five distinct data columns** and **at least 3,000 rows** to ensure sufficient complexity and depth for visualization tasks.

3.1 Task 1: Data collection

Your group is tasked with collecting a dataset and articulating the reasoning behind its selection. This involves providing a detailed narrative that justifies your choice and outlines the dataset's relevance to your project goals. Address the following points in your submission:

- **Context and Motivation**: Describe the central theme or overarching story that drove your group to select this topic. What real-world issue, question, or curiosity does it address, and why is it compelling for data visualization?
- **Dataset Topic and Sources**: Clearly define the primary subject of your dataset and identify the sources used to compile it. For example, is it sourced from government records, surveys, or sensor data?
- **Data Construction Methods**: Explain how the dataset was created. What methods or processes (e.g., manual entry, automated scraping, API collection) were employed to gather and organize the data?
- **Legal and Ethical Use**: Confirm the suitability of the dataset for educational purposes. Is it publicly available under an appropriate license (e.g., Creative Commons, Open Data)? Are there any restrictions or ethical considerations to note?

3.2 Task 2: Data Cleaning and Pre-Processing

Building on skills from prior courses such as *Introduction to Data Science* and *Programming for Data Science*, you will clean and preprocess your dataset to prepare it for visualization. This step ensures the data is accurate, consistent, and optimized for analysis. Explore and document the following aspects:

- **Row-Level Interpretation**: What does each row represent (e.g., an individual, an event, a time point)? Are all rows consistent in meaning, or do variations exist that require reconciliation?
- **Column Definitions**: Provide a clear explanation of what each column represents (e.g., age, timestamp, category). Ensure their roles in the dataset are well-defined.
- **Data Types**: Identify the current data type of each column (e.g., integer, float, string, datetime). Highlight any columns with inappropriate types for visualization or analysis (e.g., numbers stored as strings) and propose corrections.
- **Value Distribution**: Analyze the distribution of values in each column. For numeric columns, describe their range, central tendency (mean, median), and spread (e.g., standard deviation). For categorical columns, list the unique categories and their frequencies.

- **Preprocessing Needs**: Assess whether preprocessing is required (e.g., handling missing values, removing duplicates, normalizing data). If so, detail the specific techniques you will apply, such as imputation, outlier removal, or encoding categorical variables.

3.3 Task 3: Data visualization

After cleaning the dataset and conducting exploratory data analysis (EDA) to uncover patterns and insights, you will create a series of visualizations to effectively communicate the dataset's attributes and relationships. This task emphasizes the use of diverse chart types to present data in an insightful and visually appealing manner, supporting data-driven conclusions. Address the following requirements:

- **Attribute Selection**: From your EDA, pinpoint the most significant attributes (e.g., variables or features) that merit visualization. These should reflect key trends, patterns, or relationships within the data.
- **Chart Type Selection**: Choose appropriate visualization techniques from your repertoire—such as line charts, bar charts, scatter plots, pie charts, heatmaps, or box plots—to represent these attributes. Each selection should align with the attribute's data type and the story you aim to tell.
- **Justification of Choices**: For each visualization, provide a clear rationale explaining why the chosen chart type is optimal. For example, "A bar chart was selected to compare categorical frequencies due to its clarity in showing discrete differences." If you use multiple chart types for one attribute (e.g., a bar chart and a pie chart), justify the added perspective each offers.
- **Progressive Complexity**: Structure your visualizations to build from simple to complex insights. Start with univariate visualizations (e.g., histograms of single attributes) and progress to multivariate displays (e.g., scatter plots or correlation matrices) that reveal interactions between attributes.
- **Cause-and-Effect Exploration**: Investigate potential cause-and-effect relationships within the data. For instance, if visualizing COVID-19 data, explore whether a rise in infection rates correlates with increased mortality, and select visualizations (e.g., dual-axis line charts) to illustrate this dynamic effectively.

- **Insight Presentation**: After creating each visualization, present the specific insights or conclusions drawn from it. For example, "The scatter plot reveals a strong positive correlation between temperature and ice cream sales, suggesting a seasonal demand pattern." These insights should tie directly to the visual representation and enhance the understanding of the data's story.
- **Breadth of Visualization Types**: While not every relationship must be visualized, aim to incorporate a wide variety of chart types covered in your coursework. Prioritize relevance and insight over exhaustive coverage, ensuring each visualization adds value to your analysis.
- **Innovation and Engagement**: Experiment with creative or advanced visualization techniques—such as interactive plots, treemaps, or geospatial maps—to enhance engagement and uncover deeper insights. These should not only be visually striking but also meaningful in interpreting the data.

3.4 Task 4: Writing report & Present results

In this final task, you will consolidate your work from Tasks 1–3 into a professional report and organize your computational work in Jupyter notebooks for clarity and reproducibility. The goal is to effectively communicate your findings, visualizations, and insights to an audience, while ensuring your process is well-documented and accessible. Address the following requirements:

- **Clear Report in PDF Format**: Compile your project into a polished, well-structured report submitted as a PDF file. The report should include:
  - An introduction outlining your topic, motivation, and objectives.
  - A section for each task (data collection, data cleaning, data visualization), summarizing the methods, findings, and insights.
  - A conclusion that reflects on the overall story told by the data and the key takeaways from your visualizations.
  - Proper formatting with headings, subheadings, and a consistent style (e.g., font size, margins) to ensure readability and professionalism.
- **High-Resolution Charts in the Report**: If you embed all visualizations from Task 3 into the relevant sections of your report. Ensure the following:
  - Charts are exported in high resolution (e.g., 300 DPI) to maintain clarity and detail when viewed or printed.

  - Each chart is accompanied by a caption describing its purpose and a brief explanation of the insight it reveals.
  - Charts are appropriately sized within the document (neither pixelated nor overly large) and labeled with titles, axis labels, and legends as needed for standalone interpretation.

- **Clear Structure of Jupyter Notebooks**: Organize your code and analysis in Jupyter notebooks, splitting the work into separate notebooks for each major task to enhance clarity and modularity. Specifically:
  - Data Collection Notebook: Include code to locate, download, and initially explore the dataset from Kaggle, along with markdown cells explaining your choices and dataset context.
  - Data Cleaning Notebook: Contain all preprocessing steps (e.g., handling missing values, type conversions), with markdown annotations describing each action and its rationale.
  - Data Visualization Notebook: Feature the code for generating all visualizations, with markdown cells justifying chart selections and presenting the insights derived from each.
  - Ensure that each notebook is well-organized with:
    * A title and brief introduction at the top.
    * Logical sections using markdown headings (e.g., `# Section 1: Loading Data`).
    * Concise comments within code cells to explain functionality.
    * Outputs (e.g., charts, summary statistics) displayed inline for immediate review.

4 Limitations

This lab is designed with specific constraints to ensure a focused and accessible learning experience. We ask you to adhere closely to the following guidelines:

- **Programming Environment**: The exercises are intended to be completed within a standard Python programming environment, such as Jupyter Notebooks or a basic IDE (e.g., VS Code, PyCharm). To maintain consistency and emphasize core skills, we encourage the use of simple, widely accessible tools. Please refrain from using advanced, specialized software like Tableau, Power BI, or other GUI-based data visualization platforms for this lab.
- **Permitted Libraries**: You are free to leverage foundational Python libraries commonly used for data analysis and visualization, including NumPy (for numerical operations), Pandas (for

data manipulation), Matplotlib (for basic plotting), and Seaborn (for enhanced statistical visualizations). If you wish to incorporate additional libraries beyond these (e.g., Plotly, SciPy), you must seek prior approval from your instructors to ensure alignment with the lab's objectives.

- **Machine Learning Option**: While integrating simple machine learning techniques (e.g., clustering with K-means or linear regression) can enrich your data exploration and visualization, this is an optional enhancement rather than a requirement. The primary focus remains on effective data visualization, and any use of machine learning should support, not overshadow, this goal.
- **Languages in Report**: Vietnamese
- **AI-Generated Contents**: Must under 30%.

5 Evaluation Criteria

Criteria	Mark
Data collection & pre-process data	5%
Select, interpret, and visualize fields and their hidden relationships.	50%
Derive logical meaning behind each visualized data.	20%
Consider many relationships and many different perspectives.	10%
The report presents a logical and clear layout and format.	15%
There is analysis, visualization with novel charts and drawing of useful information. Use basic machine learning models.	5%
Overall comprehension of the submitted source code.	5%
Total	**110%**

6 Submission Requirements

To ensure a complete and organized submission, you must provide the following components, each adhering to the specified guidelines:

- **Docs Folder**: This folder must contain your project report, submitted in PDF format to guarantee compatibility across platforms and preserve formatting (e.g., charts, fonts). The report should be comprehensive and well-structured, addressing the following key sections:
  - **Group Information**: List your group name and the student IDs of all members to clearly identify your team.
  - **Requirement Fulfillment**: Evaluate how your project meets each task requirement (e.g., data collection, cleaning, visualization). Provide specific examples or evidence, such as the number of rows/columns in your dataset or types of visualizations created.
  - **Member Contributions**: Detail each group member's role and level of contribution (e.g., "Member X handled data preprocessing; Member Y designed visualizations"). Be specific to ensure transparency and fairness in assessment.
  - **Algorithms and Implementation**: Explain the technical approaches used, such as data cleaning techniques (e.g., imputation, normalization) or visualization methods (e.g., Matplotlib bar charts, Seaborn heatmaps). Include concise code snippets or running examples with annotations to illustrate functionality and decision-making.
  - **Presentation Style**: Ensure the report is clear, concise, and visually engaging. Use high-resolution charts, tables, or diagrams where appropriate to support your narrative, and maintain a logical flow with consistent formatting (e.g., headings, bullet points).
- **Sources Folder**: This directory should contain all source code developed for the project, primarily in the form of Jupyter notebooks (.ipynb) and Python scripts (.py). To ensure usability:
  - Organize notebooks by task (e.g., `data_collection.ipynb`, `data_cleaning.ipynb`, `data_visualization.ipynb`) with clear, descriptive filenames.
  - Include markdown cells or comments explaining code logic and outputs.
  - If you use languages other than Python (e.g., R), provide a separate README file with instructions for execution, including required dependencies and runtime environments.
- **Datasets Folder**: This folder should include the dataset(s) used in your project, adhering to the following guidelines:

- For datasets under 100 MB, include the files directly (e.g., .csv, .xlsx). For larger datasets, provide a stable, accessible link to an external source (e.g., Kaggle URL) or cloud storage (e.g., Google Drive, OneDrive), ensuring permissions are set to viewable by anyone with the link.
- Include a brief README file in the folder detailing the dataset's name, source, size, and format, along with instructions for accessing it if hosted externally.