Interaction

Mounika Devabhaktuni

Nakshatra Reddy Lenkala

1. **Build upon one or multiple visualization methods, and achieve the 8 common basic interaction methods, select, explore, reconfigure, encode, abstract, elaborate, filter, connect. For each method, specify the type and justify briefly why this interaction completes the type. (8 points, 1 point each type). Refer to the slides of interaction.**

   a. **Select:**

      Type: Point Selection Interaction

      Justification: This code allows users to interact with the scatter plot by clicking on individual data points (representing movies or TV shows). Upon clicking, the title of the selected item is displayed, enabling users to identify and explore specific data points interactively.

      Explanation: By implementing point selection interaction, users can directly engage with the data, facilitating targeted analysis and exploration of the dataset. This interactive feature enhances user engagement and understanding of the underlying information represented in the visualization.

      Conclusion:This interaction empowers users to engage directly with the data, fostering a personalized exploration by highlighting movies or TV shows that pique their interest, thereby enabling targeted analysis based on individual selections.

   b. **Explore:**

      Type:Explore(Overview + Detail Interaction)

      Justification: This method qualifies as an Overview + Detail Interaction, providing users with both a broad overview and detailed insights into the distribution of Netflix titles by genre over time.

      Explanation: Users select a genre from the dropdown menu, focusing exploration on a specific category. The resulting bar chart displays the number of titles in the selected genre across release years, revealing trends and patterns.

Conclusion: The combination of genre selection and bar chart visualization enables multifaceted exploration, empowering users to understand both overall trends and genre-specific nuances in Netflix title distribution.

c. Reconfigure:

Type: Reconfigure

Justification: This code allows users to dynamically reconfigure the heatmap based on their preferences. By selecting different options from the dropdown menu, users can reorder the heatmap either by country or total count, offering different perspectives on the data.

Explanation: The dropdown widget serves as a control mechanism, enabling users to choose how they want the data to be organized and displayed. When users make a selection, the heatmap is dynamically updated to reflect the new configuration, providing flexibility and allowing users to explore the data from various angles.

Conclusion:The dynamic reconfiguration of the heatmap based on user preferences offers multiple perspectives on the data, making it easier to identify patterns or anomalies based on the chosen organization criteria.

d. Encode:

Type: Encode

Justification: This code encodes the data regarding the number of Netflix titles by country into two choropleth maps, each with a different color scale. The color of each country on the map represents the count of Netflix titles released in that country.

Explanation: By encoding the count of Netflix titles as the intensity of color on the choropleth maps, users can visually perceive the distribution of titles across different countries. The choice of different color scales (Electric and Picnic) further encodes the data, allowing users to compare the distribution using different color gradients. This encoding facilitates the understanding of the geographical distribution of Netflix titles and the variations in count between countries.

Conclusion: Encoding the number of titles with color intensity on choropleth maps visually communicates the geographical distribution of Netflix content, making comparisons between countries intuitive and immediate.

e. Abstract/Elaborate:

Type: Elaborate

Justification: This code enhances the data visualization of Netflix titles by offering two distinct visualization approaches: a 3D scatter plot using Matplotlib and an interactive scatter plot using Plotly Express. Both methods serve to elucidate the correlation between the release year, duration, and other attributes of Netflix titles.

Explanation: This code provides two visualization methods- a 3D scatter plot using Matplotlib and an interactive scatter plot using Plotly Express. These methods allow users to explore Netflix title data by adjusting granularity and accessing detailed information through hover and zoom features, enhancing understanding of dataset patterns.

Conclusion: Offering both a 3D scatter plot and an interactive scatter plot provides users with flexible ways to delve into the data, from high-level patterns to granular details, enhancing their comprehension of complex relationships.

f. Connect:

Type: Connect

Justification: This code snippet provides a visualization interaction by generating a parallel coordinate plot using Plotly Express. The parallel coordinates plot is an effective way to visualize high-dimensional data and explore relationships between different variables. Users can interact with the plot by hovering over the lines to see specific data points and by adjusting the color scale to represent a continuous variable.

Explanation: This code snippet generates a parallel coordinate plot using Plotly Express, facilitating visualization interaction. The plot visually represents relationships between content type, release decade, and rating in Netflix data. Users can hover over lines to view specific data points and adjust color scales for deeper insights. Overall, it offers an interactive exploration of dataset characteristics and relationships.

Conclusion: The parallel coordinates plot connects multiple dimensions of the Netflix dataset, allowing users to explore and understand the intricate relationships between content type, release decade, and ratings, facilitating a multifaceted view of the data landscape.

g. **Filter**

Type: Filter

Justification: This code enables users to filter Netflix titles by genre and release year, offering a tailored exploration experience.

Explanation: Users can select a genre from the dropdown menu and specify a release year range using the slider, refining the dataset for focused analysis.

Conclusion: By providing flexible filtering options, users can extract meaningful insights from the data, enhancing exploration efficiency and user experience.

2. **Explore customized interaction with event handling -- interacting with figures, setup of event for your project, mouse enter, leave, object picking, keyboard. For each method, specify the type and justify briefly why this interaction completes the type. (6 points, 1 point each type). Refer to the link: https://matplotlib.org/stable/users/explain/figure/index.html#**

a. **Interacting with figures**

Plotly Express Interactive Plot: The second code snippet uses Plotly Express to ***generate an interactive scatter plot*** of Netflix titles' release year versus duration. Users can hover over data points to view additional information about each title, providing an interactive exploration experience that enhances understanding and engagement.

b. **Setup of event for your project**

Graphical User Interface (GUI) Interaction: The first code snippet utilizes Tkinter to create a GUI window with a button. When ***the button is clicked, it triggers an event to display a random Netflix title,*** completing the GUI interaction type by providing a responsive interface for user interaction.

c. **Mouse enter & leave**

Matplotlib Mouse Handling: In the third code snippet, Matplotlib is used to create a plot where users can trigger events by hovering over the axes or the figure. ***Different events, such as entering or leaving the axes or figure,*** are handled, demonstrating event-driven programming to make the plot more responsive and interactive.

d. **Object picking**

Matplotlib Event Handling with Pick Event: The fourth code snippet showcases Matplotlib's event handling capability with a pick event. Users can click on data points in the plot, triggering an event to display the number of Netflix titles released in the corresponding year, completing the event-driven programming type by enabling user-triggered interactions.

Matplotlib Event Handling for Interactive Plot Update: The fifth code snippet employs Matplotlib event handling to create an interactive plot. When ***users click on a specific year in the plot, a distribution plot of movie durations for that year is dynamically generated,*** enhancing interactivity and exploration of the dataset.

e. **Keyboard**

Console-based User Input Handling with Keyboard Events: The code snippet above demonstrates console-based user input handling with keyboard events. ***Users can press 'r' to view the distribution of Netflix titles by release year, 'd' to view by duration,*** or 'q' to quit the program. This completion falls under event-driven programming, as it listens for keyboard input events and responds accordingly, providing users with a dynamic way to visualize the dataset.

3. **An algorithm to enhance any of your interaction methods, such as a new method to show a unique statistical feature of the data. Refer to one example paper from our visualization and interaction papers. You can modify and simplify the method. Please describe the original and your method. (3 points) Here are suggestions with Python source codes or wrapper:**

Class-constrained t-SNE: Combining Data Features and Class Probabilities https://github.com/alicelh/class-constrained-t-SNE

Force-directed graph layouts revisited: a new force based on the t-Distribution https://github.com/t-fdp/t-fdp

ManiVault: A Flexible and Extensible Visual Analytics Framework for High-Dimensional Data

The original c-tSNE algorithm provides a more structured and class-aware embedding compared to traditional t-SNE, making it suitable for datasets with labeled classes. However, implementing the original algorithm can be complex and computationally intensive.

To simplify the method for practical use, we can adopt a wrapper approach that integrates existing Python libraries like scikit-learn and modifies the t-SNE embedding process to incorporate class constraints. This wrapper can provide a user-friendly interface for applying

c-tSNE to datasets, allowing users to specify class constraints and visualize the embeddings accordingly.

By providing a simplified implementation of c-tSNE as a Python wrapper, users can easily enhance their interaction methods by incorporating class-aware embeddings into their visualization pipelines. This enhancement enables better exploration and understanding of high-dimensional data, especially in scenarios where class information is crucial for analysis and interpretation.

Algorithm:

Load Data:

Load Netflix dataset.

Fill missing values with 'Unknown'.

Encode categorical 'rating'.

Extract numeric 'duration' values.

Encode 'type' (Movie or TV Show).

Select Features:

Choose 'rating_encoded' and 'duration_numeric' for t-SNE.

Apply t-SNE:

Embed features into 2D space with t-SNE.

Set random state for reproducibility.

Visualize:

Plot scatter with t-SNE results.

Color points by 'rating' with color bar.

Add a secondary y-axis for 'type' color bar.

Include titles, labels, and legends.

Advantages of Introduced Algorithm Over Original Method:

1. Simplicity

2. Computational Efficiency
3. User-Friendly Interface
4. Reproducibility
5. Enhanced Visualization

4. **Explore combined interaction methods by applying all your basic interaction methods, such as exploring first, filtering second, reconfigure, …, and etc. Demonstrate the best 3 insightful knowledge you have obtained with the combined interaction. This will be graded upon how the results will be useful by a domain user. You can specify your domain user, such as a wine specialist for the wine quality dataset, and show how the correlation of wine attributes can help specialists to achieve a task of their interests. (3 points)**

Integrating the insights of domain users and their applications enhances the comprehensive utility of the code provided. This integration not only showcases the versatility of the code in analyzing and visualizing data for strategic decision-making but also highlights its potential to impact various aspects of the entertainment industry, from content creation to audience engagement and academic research. Below are the these insights for domain users:

Understanding Genre Popularity and Evolution Over Time -

For Content Creators and Producers: Recognizing shifts ***in genre popularity helps in aligning content creation with viewer demand***. For instance, noticing a trend towards increased popularity of action movies with PG-13 ratings can prompt creators to develop content that fits this emerging market preference, maximizing audience reach and engagement.

For Streaming Platform Strategists and Curators: Insights into evolving genre trends enable platforms to ***adapt their content libraries to match changing viewer preferences***, ensuring a dynamic content offering that retains and attracts subscribers. Identifying a shift towards more family-oriented content, for example, could lead platforms to invest in a broader selection of PG-rated movies.

Content Strategy Development Based on Rating Distribution -

For Marketing Teams within Entertainment Companies: Understanding how content ***rating distributions within genres change over time can guide targeted marketing strategies***. Knowing that R-rated horror movies are gaining popularity, for example, allows marketing teams to craft campaigns that resonate with adult audiences looking for thrilling content.

For Regulatory Bodies and Content Rating Agencies: Observing trends in content rating distributions aids in assessing whether current rating systems ***reflect societal expectations, informing potential updates to content regulation policies to protect viewers, especially younger audiences,*** from content that may be inappropriate for their age group.

Research and Academic Insights into Media Studies -

For Media Researchers and Academics: The detailed analysis of genre popularity and content ratings over time serves as a valuable dataset for studying cultural and societal trends. This can lead to insights into how ***societal norms and values are reflected in media content,*** contributing to broader academic discussions on media influence and cultural dynamics.

For Streaming Platform Strategists and Curators: Beyond content acquisition and library curation, ***insights from genre and rating trends can inform long-term strategic planning,*** including partnership opportunities with content creators, investment in original content production, and global market expansion strategies tailored to varying cultural preferences.

By combining these insights with the specific applications for domain users, the code's utility is significantly amplified. It provides a foundation for strategic decision-making across the content creation and distribution process, enhances marketing and engagement strategies with targeted insights, informs academic research with rich datasets on cultural trends, and supports policy development for content regulation. This holistic approach ensures that the entertainment industry can navigate the complexities of viewer preferences and market demands more effectively, driving innovation and growth in content creation and distribution.

Roles -

Mounika Devabhaktuni -
Developed visualization methods for interaction methods and enhanced an interaction method using a customized algorithm.

Nakshatra Reddy Lenkala -
Implemented customized interaction through event handling and combined interaction methods to derive insightful findings for practical domain utility.