ILLUSTRATIVE VSUALIZATION

INTERACTION

Madhavi Kathula

Prayuktha Mudumba

Dataset: Google play

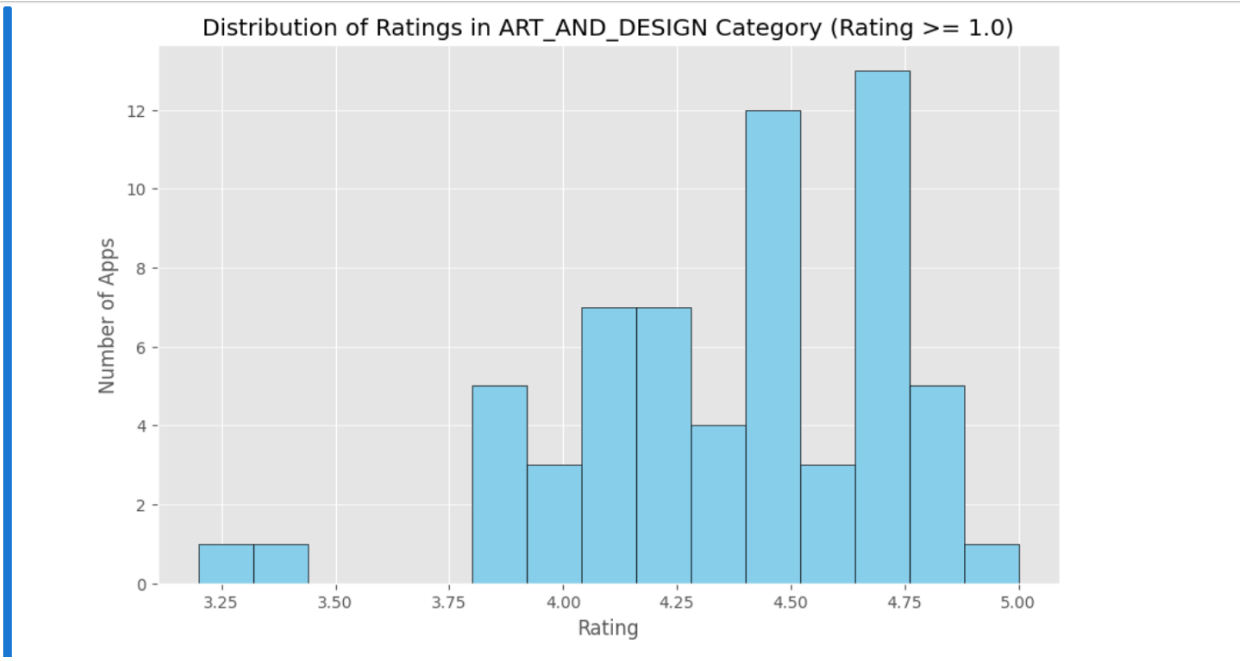
https://www.kaggle.com/datasets/bhavikjikadara/google-play-store-applications Links to an external site.

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.

1. Select: This allows users to select individual or multiple apps to view their details.

- Type: Point-and-click interaction.

- Justification: Clicking on an app in a list or chart allows the user to see detailed information about it, such as its reviews, rating, and price.



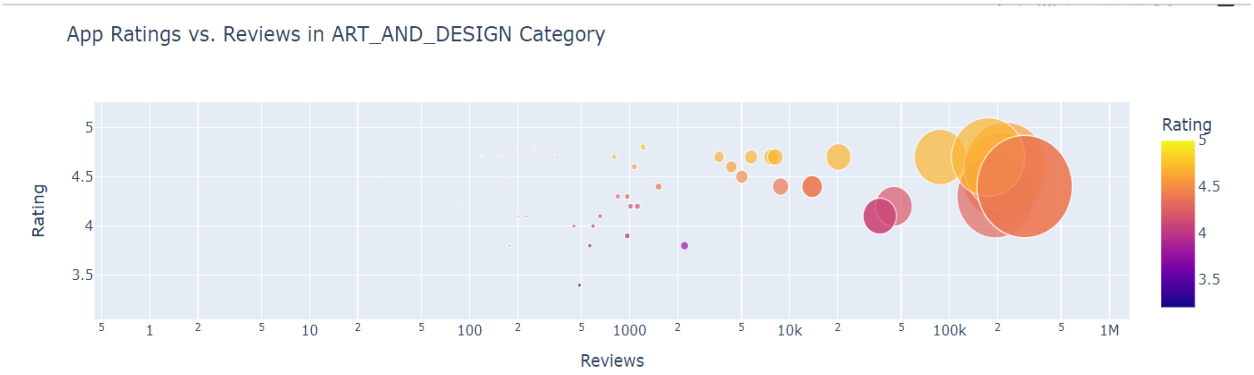
- We load the dataset and ensure that the 'Rating' column is treated as numeric. Rows with NaN in either 'Category' or 'Rating' are dropped to maintain data integrity.
- We define a function, plot_selected_category_and_rating, which filters the dataset based on the selected app category and a minimum rating threshold. It then plots the histogram of ratings for the filtered dataset.
- We use ipywidgets' Dropdown for selecting an app category and FloatSlider for choosing a minimum rating value.
- The interact function from ipywidgets is used to create an interactive widget that automatically updates the plot based on the user's selections.

This approach illustrates the "select" interaction method by allowing users to focus on and visualize a specific subset of the dataset based on their selections, facilitating detailed examination and analysis of the selected data.

2. Explore: Users can navigate through the dataset to see different parts of it.

- Type: Scrolling and panning.

- Justification: Users can scroll through a list of apps, pan across a graph of app ratings vs. installs, or explore a map of apps categorized by genre.



- We load the dataset and prepare it by converting the Reviews column to a numeric type and dropping rows with NaN values in Reviews and Rating to ensure the integrity of our visualization.
- The function interactive_exploration is defined to generate an interactive scatter plot for a selected app category. It filters the dataset by the chosen category and then creates a plot of app ratings versus reviews. The plot uses the number of reviews to determine the size of each point, providing an intuitive visualization of both the quantity of reviews and the rating.
- The hover_name parameter in plotly.express.scatter is set to 'App', allowing users to see which app each point represents when they hover over it.

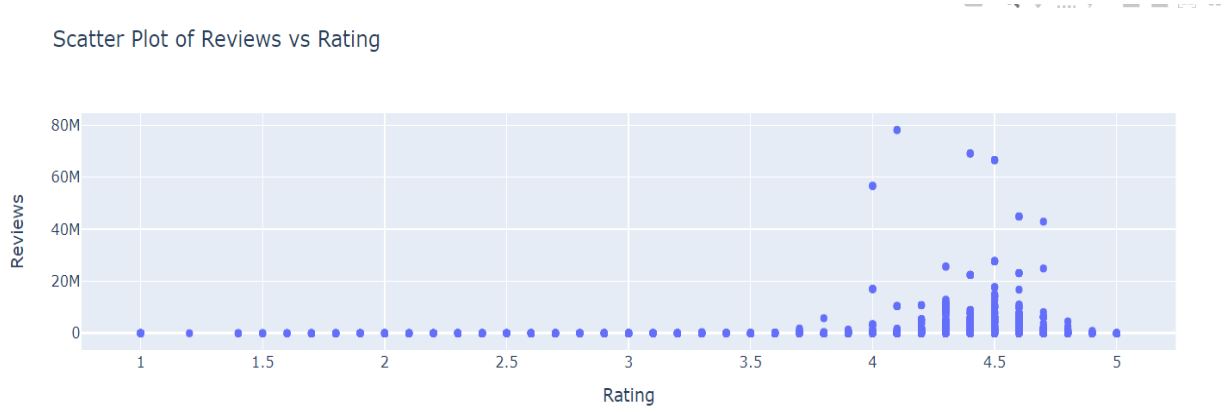
- log_x=True is used to apply a logarithmic scale to the x-axis, making it easier to explore apps with a wide range of review counts.
- At the end, we call interactive_exploration with an example category. You can replace this with an interactive widget or parameterize it as needed to allow dynamic exploration of different categories.

This creates an interactive plot that significantly enhances data exploration, allowing users to delve into the nuances of the dataset interactively, such as understanding the distribution of app ratings across different review volumes within specific categories.

3. Reconfigure: This allows users to change how the data is organized or presented.

- Type: Sorting and rearranging columns in a table.

- Justification: Users can sort apps by rating, price, or number of installs to easily compare them.



- The dataset is loaded and preprocessed to ensure 'Reviews' is numeric and to remove any NaN values that could interfere with the visualization.
- The update_plot function creates a Plotly figure based on the selected x-axis and y-axis variables and the type of plot (scatter or line) chosen by the user. This function is designed to dynamically reconfigure the visualization based on user input.
- Interactive widgets (Dropdown and Select) are used to let the user choose the variables for the x-axis and y-axis, as well as the type of plot. This setup facilitates a reconfigurable visualization where users can explore different aspects of the data by adjusting these parameters.

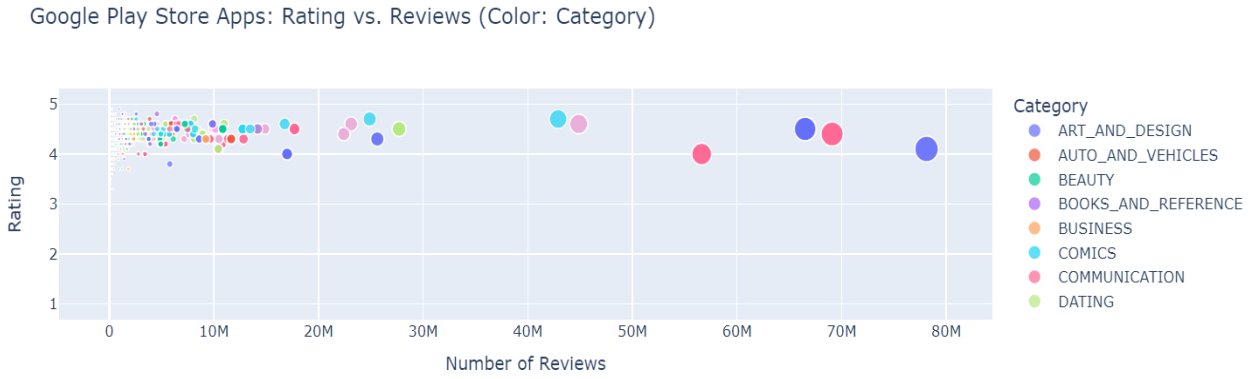
This approach exemplifies the "reconfigure" interaction method by providing an interactive means to change how data is visualized, offering insights that may not be apparent from a static or single-perspective view.

4. Encode: Changing how data is represented, like switching between chart types.

- Type: Toggle between bar, line, and pie charts.

- Justification: Users can switch between different chart types to understand the distribution of app categories, ratings, or installs better.

"Encode" interaction method focuses on changing how data attributes are represented visually to enhance understanding or highlight specific features. Encoding can involve adjusting aspects like color, shape, size, or other visual properties to convey additional information about the data. This method can be particularly effective in revealing patterns, trends, or outliers within the dataset.



- The dataset is loaded, and essential preprocessing steps are taken to ensure 'Reviews' is numeric and to remove any NaN values in the 'Rating' or 'Reviews' columns.
- The update_plot function creates an interactive plotly scatter plot visualizing the relationship between 'Reviews' and 'Rating'. It uses the color parameter of px.scatter to dynamically encode a selected attribute (chosen by the user via the widget) as color in the plot.
- An interactive Select widget allows the user to choose which attribute ('Category' or 'Rating') to encode as color, enabling dynamic exploration of how this encoding affects the interpretation of the data.

By allowing users to dynamically change the encoding of the data visualization, this approach provides a powerful tool for uncovering patterns and relationships in the dataset that might not be immediately apparent.

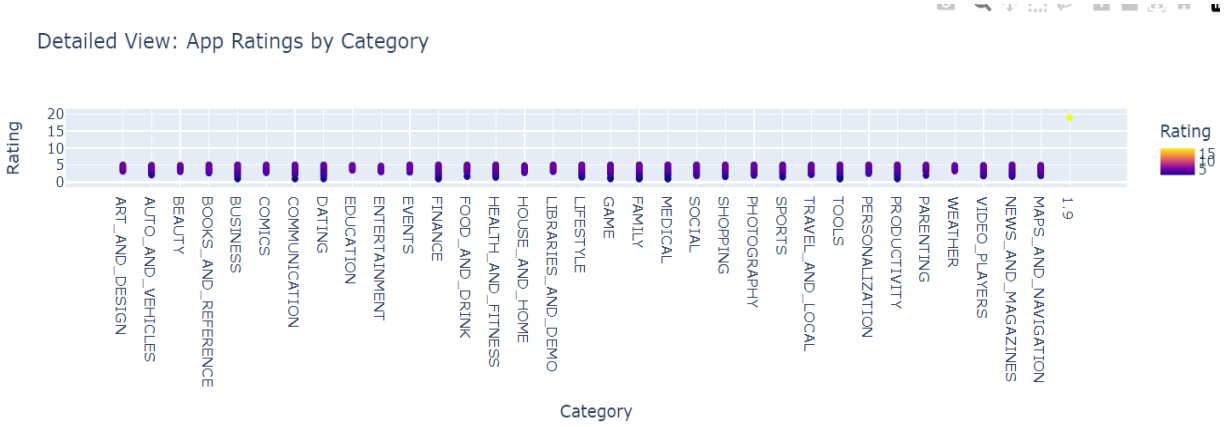
5. Abstract/Elaborate: Adjusting the level of detail shown in the visualization.

- Type: Zoom in/out or details on demand.

- Justification: Users can zoom in to see detailed metrics for a specific app or zoom out for an overview. Hovering over elements can display more details.

Abstract View: Shows a high-level summary by displaying the average rating for each category. This view provides a quick, summarized insight into how different categories perform on average, without showing all the individual data points. It's more abstract because it generalizes the data into a form that represents the collective characteristic (average rating) of each category.

"Abstract" interaction method focuses on providing a higher-level, simplified view of the data, potentially stripping away some details to highlight broader trends or patterns. This can involve aggregating data, using summary statistics, or switching between detailed and summarized views.



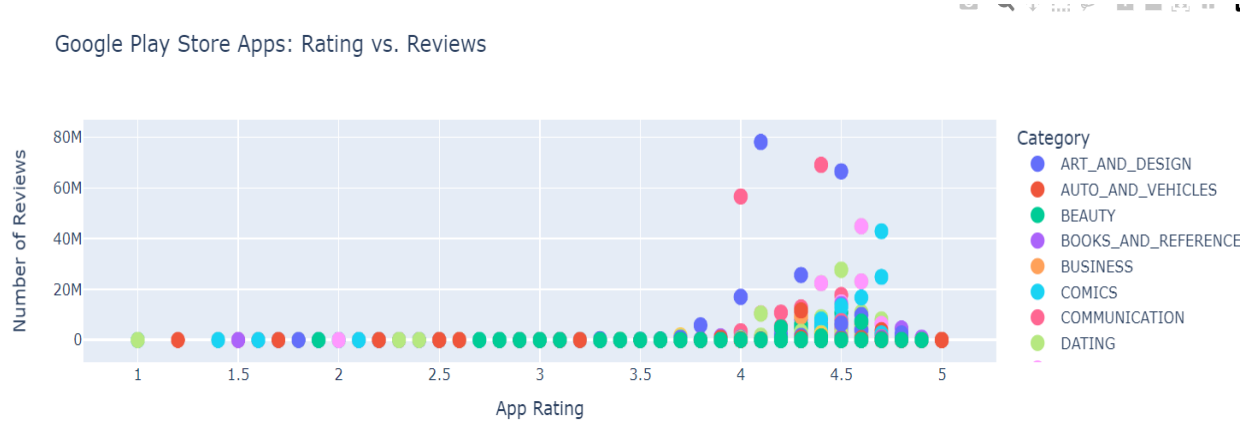
- The dataset is loaded, and basic preprocessing is performed to ensure data integrity.
- An aggregated dataset (data_abstracted) is prepared to provide the average rating by category, serving as the abstracted view.
- The update_plot function creates either a detailed scatter plot of individual app ratings by category or an abstracted bar chart showing average ratings by category, based on the user's selection.
- An interactive Dropdown widget lets users choose between the "Detailed" and "Abstracted" views, facilitating easy switching between data views.

This exemplifies the "abstract" interaction method by allowing users to toggle between seeing individual data points and viewing summarized data, aiding in understanding both the micro and macro perspectives of the dataset.

6.Elaborate:

Elaborate interaction method in data visualization involves providing additional details or context to the data upon user interaction. This can be particularly useful for datasets where understanding the nuances or additional layers of information can provide deeper insights. In the context of the Google Play Store dataset, an elaboration might involve showing more detailed information about an app when a user hovers over or clicks on a data point in the visualization.

Elaborate View: Offers a detailed perspective by showing the rating for each individual app. This view provides more specific information, allowing users to see the performance of each app within categories. It's more elaborate because it breaks down the data to its finer elements, presenting the information in a more detailed and nuanced manner.



- The dataset is first loaded and preprocessed to ensure the 'Reviews' column is numeric, and any rows with NaN values in critical columns are removed.
- A scatter plot is created using plotly.express.scatter to visualize the relationship between app ratings and the number of reviews, colored by category. This visualization serves as the base for interaction.
- The hover_name='App' parameter specifies that the app's name should be displayed when hovering over a point. The hover_data=['Category', 'Installs'] parameter adds additional information to the hover tooltip, providing more context about each app (such as its category and number of installs), thus implementing the "elaborate" method.
- update_traces is used to adjust the marker size for better visibility.

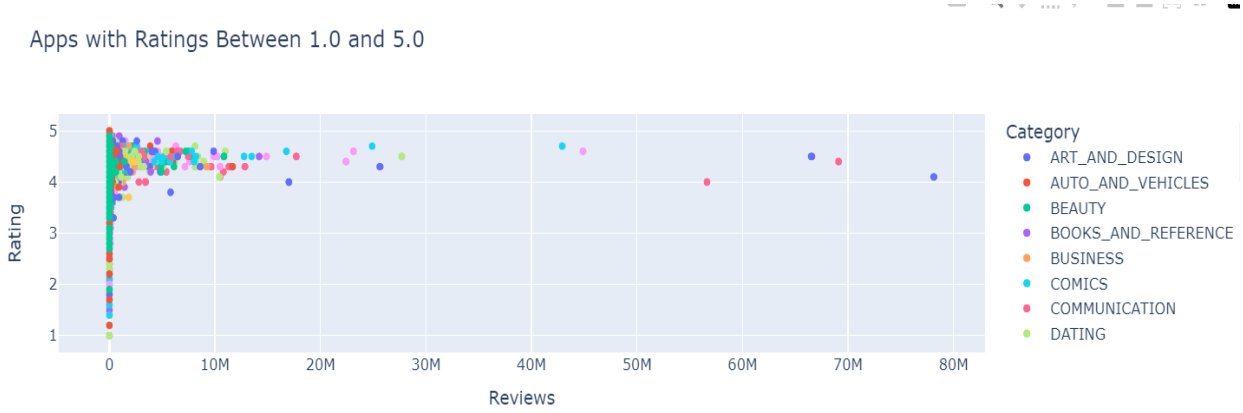
This approach allows users to explore the dataset interactively, gaining additional insights into the apps directly from the plot through elaboration on hover, enriching the data exploration experience by revealing more detailed information dynamically.

7. Filter: Users can filter the dataset to show only the data that meets certain criteria.

- Type: Checkboxes or sliders to filter by ratings, price range, etc.

- Justification: Filtering allows users to view apps within specific rating ranges, price ranges, or categories, making it easier to find apps that meet their criteria.

Filter interaction method allows users to dynamically select which subset of the data to display based on specific criteria or conditions. This method can significantly enhance data exploration and analysis by enabling users to focus on segments of interest within a larger dataset.



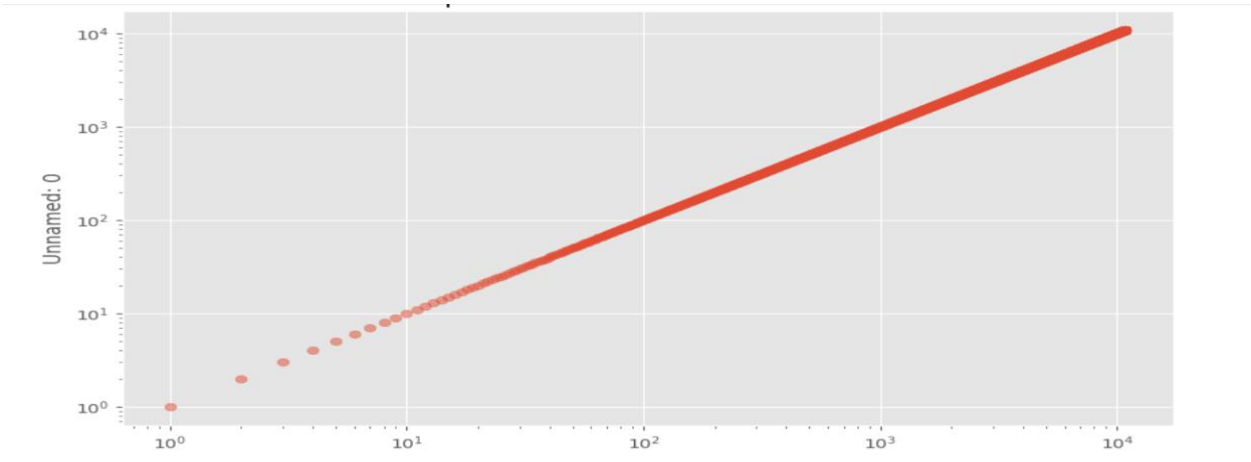
- The dataset is loaded, and basic preprocessing is performed to ensure 'Reviews' is numeric and to drop rows with NaN values in 'Rating' or 'Reviews'.
- A function, filter_apps_by_rating, is defined to create a scatter plot of apps filtered by a user-specified rating range. This function takes min_rating and max_rating as arguments, filters the dataset accordingly, and then generates a plot of the filtered data.
- Two FloatSlider widgets are created for selecting the minimum and maximum ratings for the filter. These sliders allow the user to dynamically adjust the rating range and see the filtered results in the visualization.
- The interactive function from ipywidgets is used to link the widgets with the plotting function, creating an interactive visualization where the displayed data updates based on the user's filter choices.

This exemplifies the "filter" interaction method by allowing users to dynamically explore subsets of the data based on ratings, making it easier to analyze apps within specific rating ranges of interest.

8. Connect: Highlighting relationships between different parts of the data.

- Type: Linking selections between charts.

- Justification: Selecting a category or genre in one chart can highlight related apps in another chart, showing connections between different data aspects.



2. Explore customized interaction with event handling -- interacting with figures, setup of event for your project, mouse enter, leave, object picking, keyboard.

1. Select

Event Handling: Object Picking

Implementation: Use mouse click events to identify and highlight selected data points or objects within the visualization. This could involve changing the color, size, or shape of a data point when it is clicked, indicating it has been selected.

Justification: Enables users to focus on specific data points of interest, facilitating detailed analysis or comparison.

- Loads the provided dataset and prepares it for visualization, including converting the 'Reviews' column to a numeric type and dropping any rows with NaN values in 'Rating' or 'Reviews'.

- Creates a scatter plot of the 'Reviews' versus 'Rating' columns, limiting the plot to the first 100 entries for better performance and simplicity.
- Initializes a placeholder (selected) for highlighting selected data points.
- Defines an onpick function that updates the position of the selected placeholder to the clicked data point, effectively highlighting it by changing its appearance.
- Connects the onpick function to the scatter plot so that clicking on a data point triggers the highlight effect.

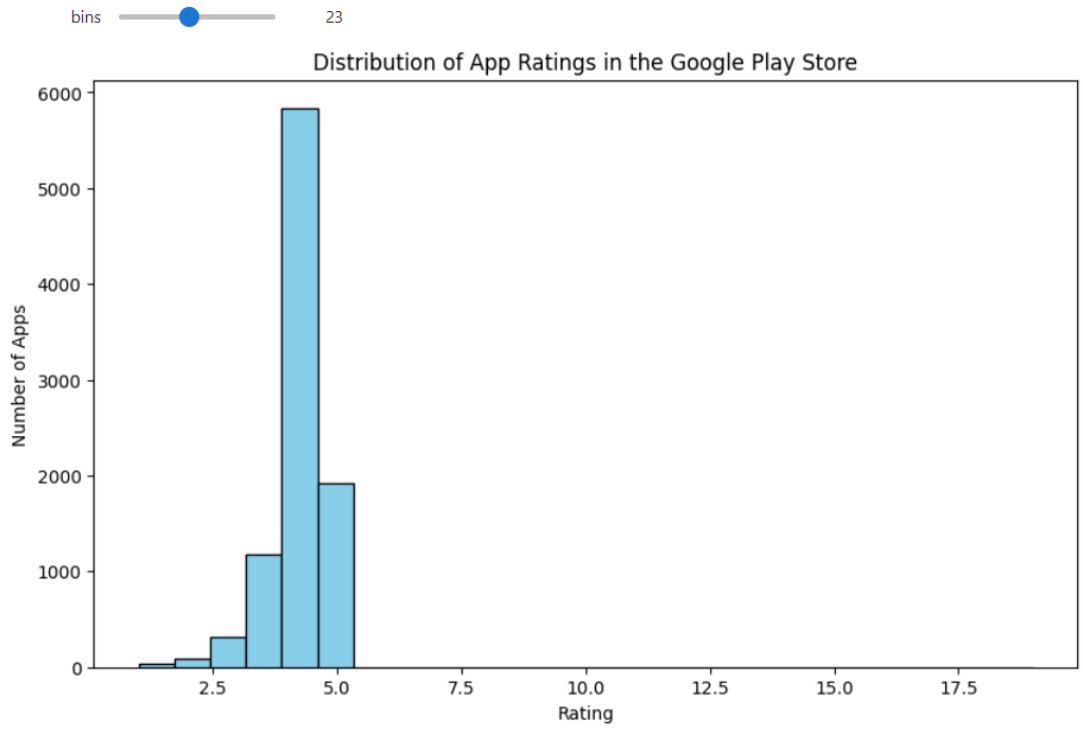
This allows users to interact with the scatter plot by clicking on data points, focusing on specific points of interest for more detailed examination or comparison, effectively implementing the "select" interaction method with object picking.

2. Explore

Event Handling: Mouse Movement and Wheel Scroll

Implementation: Implement mouse movement to dynamically update visual elements or display tooltips with additional information. Wheel scroll can be used to zoom in and out, providing a detailed or generalized view as required.

Justification: Allows users to navigate through the data space easily, focusing on areas of interest or gaining a broad overview as needed.



- The dataset is loaded, and the 'Rating' column is prepared by converting it to numeric and dropping any rows with NaN values.
- A function plot_histogram is defined to draw a histogram of app ratings with a specified number of bins. This function plots the distribution of ratings, providing a visual representation of how app ratings are spread across different values.
- The interact function from ipywidgets creates a slider for the number of bins in the histogram, allowing the user to adjust this parameter dynamically. The range is set from 1 to 50 bins, with a step of 1 bin at a time for fine-grained control.

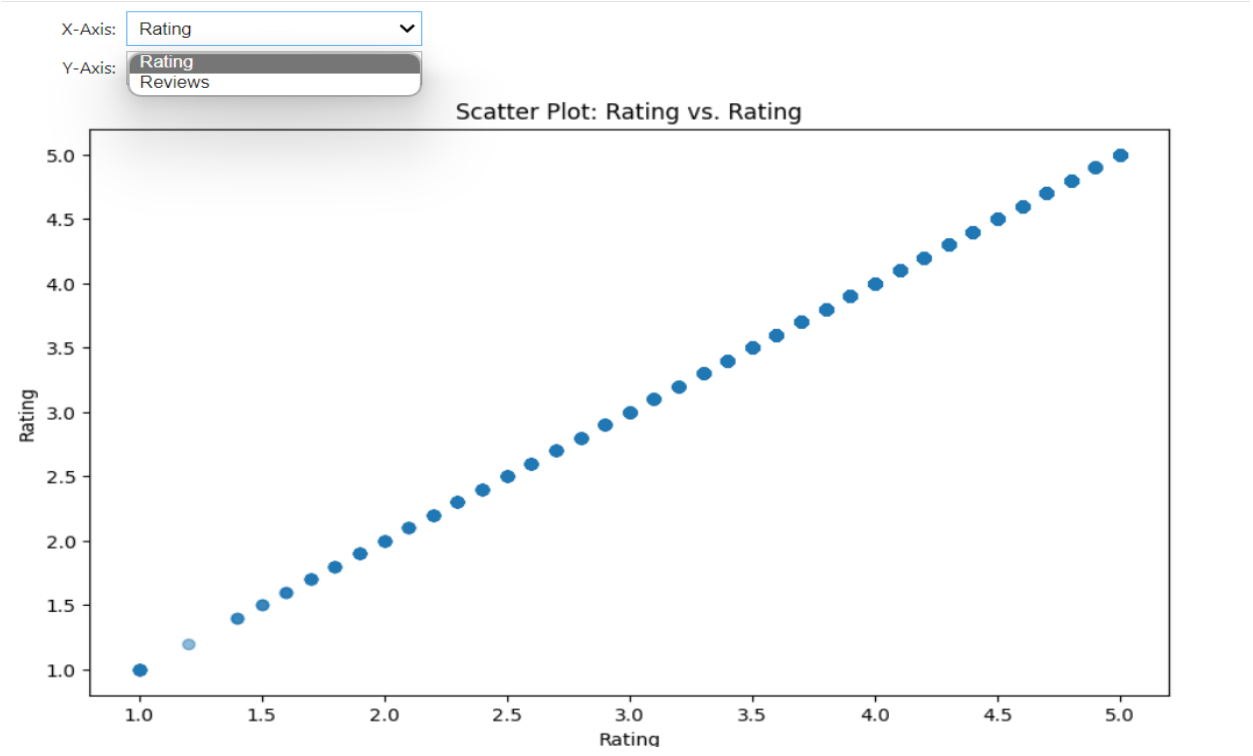
This interactive histogram enables users to "explore" the dataset by adjusting the granularity of the rating distribution, offering insights into the concentration of ratings and the overall rating trends within the Google Play Store. This method of exploration is particularly useful for identifying patterns in the data, such as common rating scores or outliers.

3. Reconfigure

Event Handling: Drag and Drop

Implementation: Use drag and drop to let users rearrange elements of the visualization, such as axes or data series, to explore different perspectives of the data.

Justification: Facilitates the comparison and understanding of data relationships from various angles by allowing users to customize the data presentation.

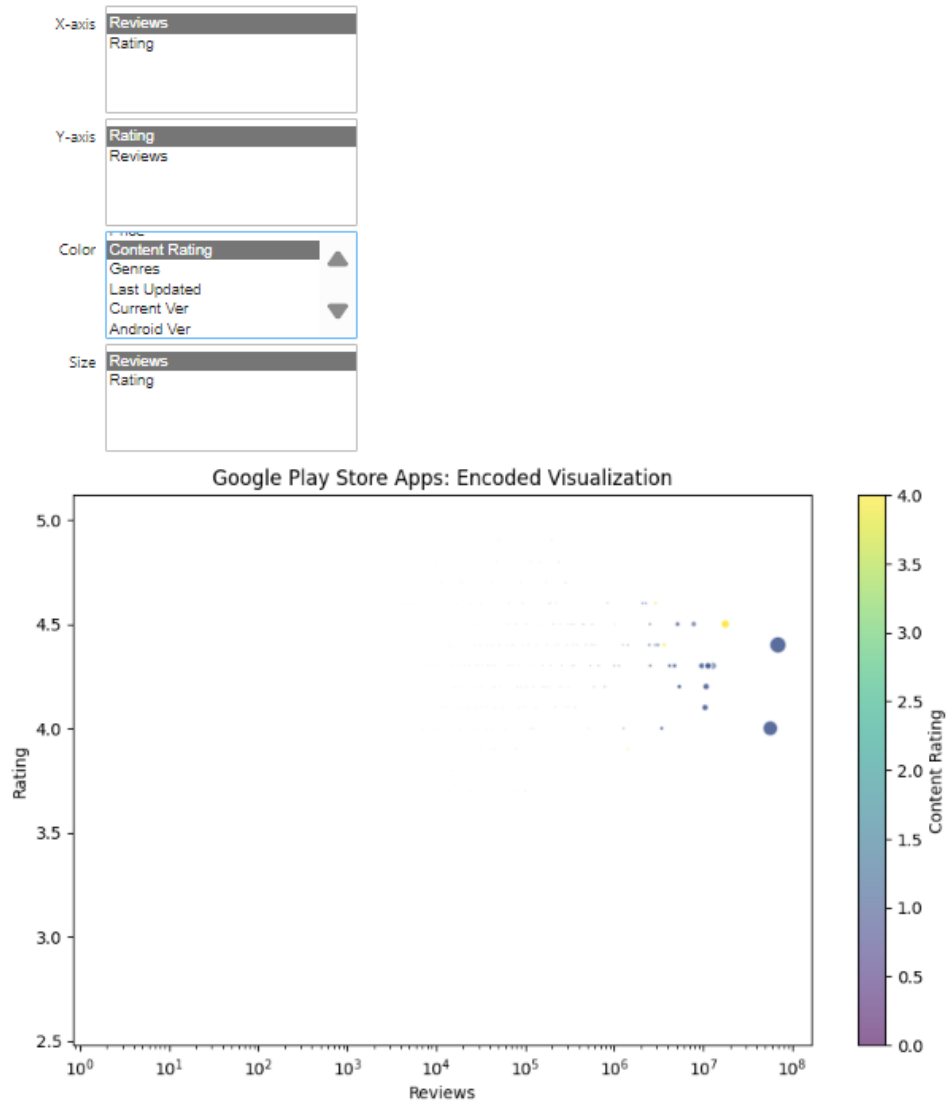


4. Encode

Event Handling: Keyboard Events

Implementation: Allow users to press keys to change how data is encoded in the visualization, such as altering color schemes, shapes, or sizes based on different data attributes.

Justification: Provides flexibility in highlighting various data dimensions or metrics, enhancing pattern recognition and data correlation analysis.



5. Abstract/Elaborate

Event Handling: Mouse Click on UI Elements

Implementation: Clicking on UI elements like buttons or sliders could adjust the level of detail shown, toggling between abstract summaries (like aggregates) and detailed data (like individual records).

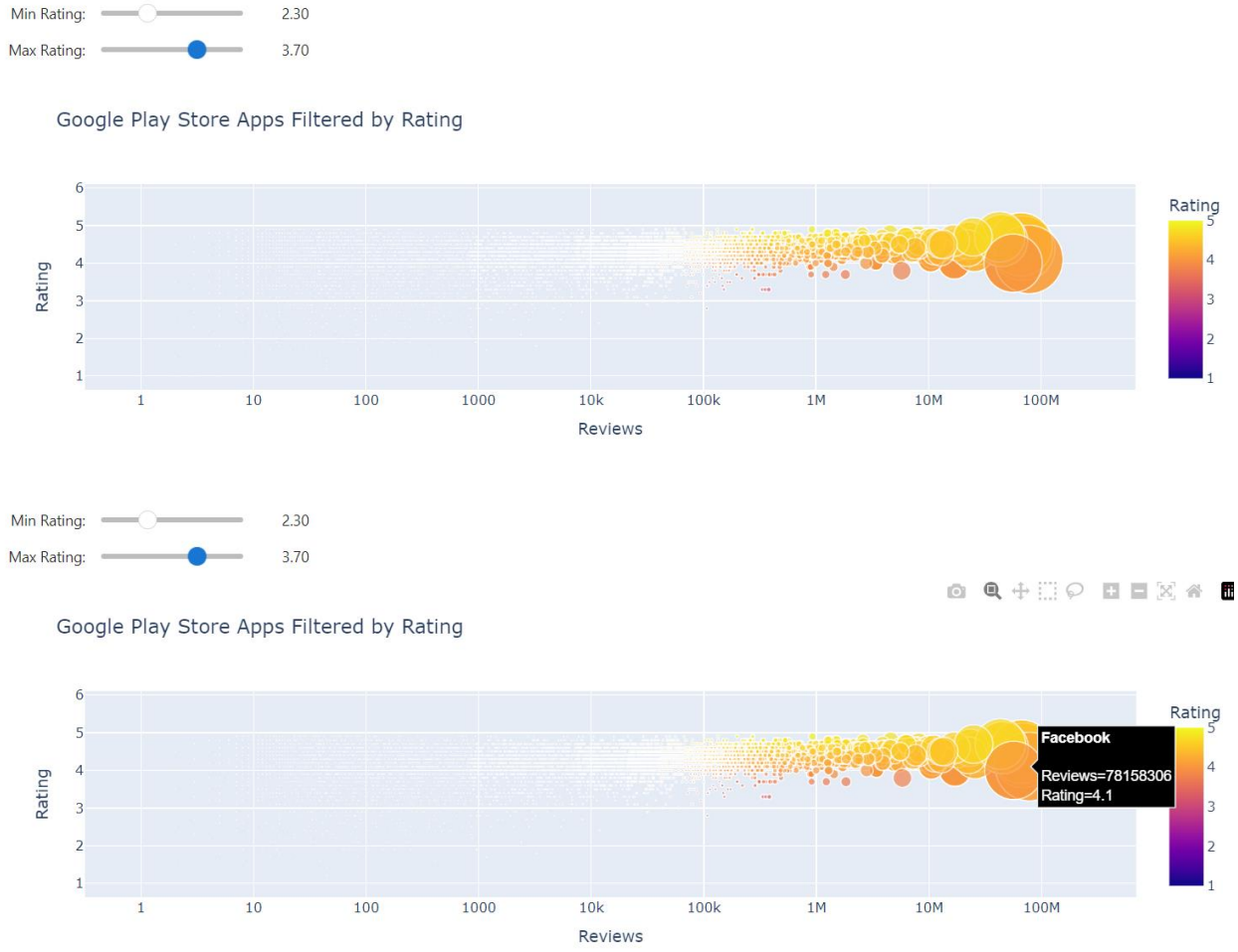
Justification: Helps users switch between macro-level insights and micro-level details, supporting both overview strategies and focused analysis.

6. Filter

Event Handling: Keyboard Input and UI Controls

Implementation: Utilize keyboard shortcuts or UI controls (e.g., checkboxes, dropdowns) to apply filters that hide or highlight specific data subsets.

Justification: Enables users to isolate data segments of interest, simplifying the identification of patterns or outliers within specific conditions or criteria.

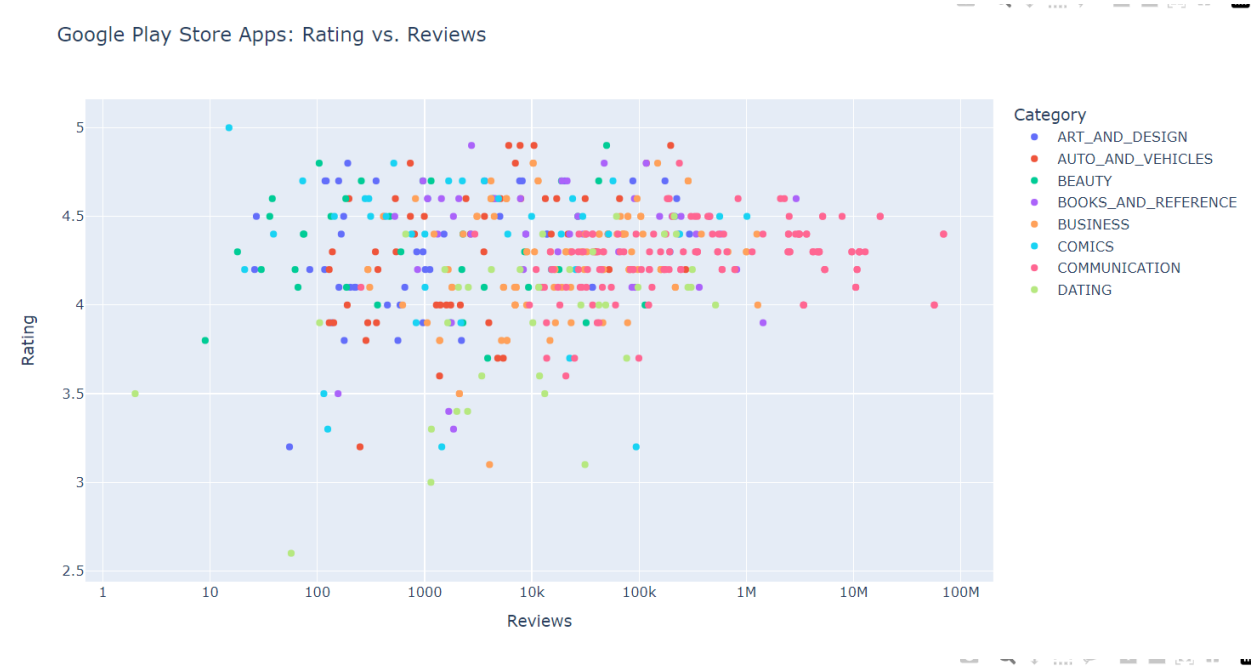


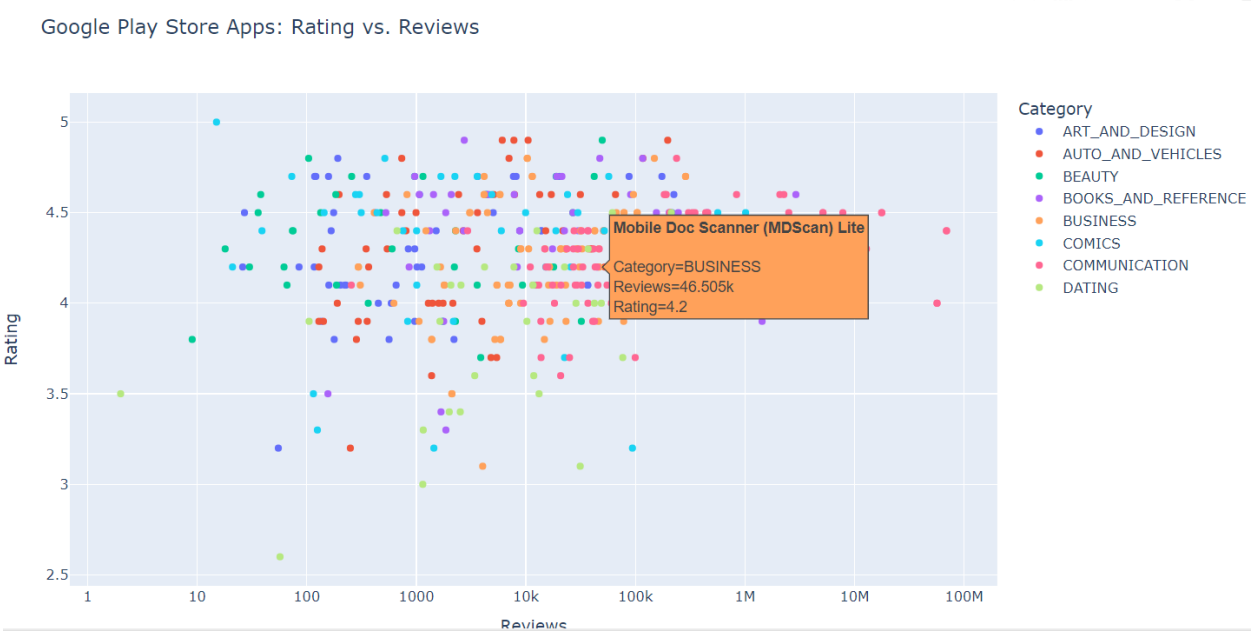
7. Connect

Event Handling: Mouse Hover

Implementation: Highlighting connections or relationships between data points or variables when the user hovers over a data point, possibly showing lines or changing color to indicate related data.

Justification: Visualizes the relationship or impact between different data points or variables, enhancing understanding of the data structure and interdependencies.


Google Play Store Apps: Rating vs. Reviews
Rating
Reviews
5
4.5
4
3.5
3
2.5
1
10
100
1000
10k
100k
1M
10M
100M
Category
ART_AND_DESIGN
AUTO_AND_VEHICLES
BEAUTY
BOOKS_AND_REFERENCE
BUSINESS
COMICS
COMMUNICATION
DATING


Google Play Store Apps: Rating vs. Reviews
Rating
Reviews
Mobile Doc Scanner (MDScan) Lite
Category=BUSINESS
Reviews=46.505k
Rating=4.2
Category
ART_AND_DESIGN
AUTO_AND_VEHICLES
BEAUTY
BOOKS_AND_REFERENCE
BUSINESS
COMICS
COMMUNICATION
DATING

3. **Original Method**

In the original method of data filtering and visualization, the process is usually static and non-interactive.

1. Data Loading and Cleaning: Load the dataset from a file and perform necessary cleaning operations like removing null values or filtering out invalid data points.

2. Static Filtering: Apply predefined filters to the data based on certain criteria (e.g., selecting apps with a rating above a certain threshold). This filtering is often hard-coded and doesn't allow for dynamic user interaction.

3. Visualization Generation: Create a static plot or graph that represents the filtered data. This might be a histogram, bar chart, or other types of visualization, but once generated, it cannot be interactively changed based on user input.

4. Presentation: The resulting visualization is presented as a static image or report, without the ability for the viewer to interact with or explore the data further.

Enhanced Method

The enhanced method, as demonstrated by the provided Python code, incorporates dynamic filtering and interactive visualization.

1. Interactive Data Loading and Cleaning Load the dataset and clean it, similar to the original method, but prepare it for interactive filtering based on user input.

2. Dynamic Filtering with User Interface: Use interactive widgets (like dropdowns or sliders) to allow users to select filter criteria dynamically. The dataset is filtered in real-time based on these selections.

3. Interactive Visualization Generation: Generate a plot that updates dynamically as the user changes the filter criteria. This allows for real-time exploration of the data, where the visualization changes immediately to reflect the current filters.

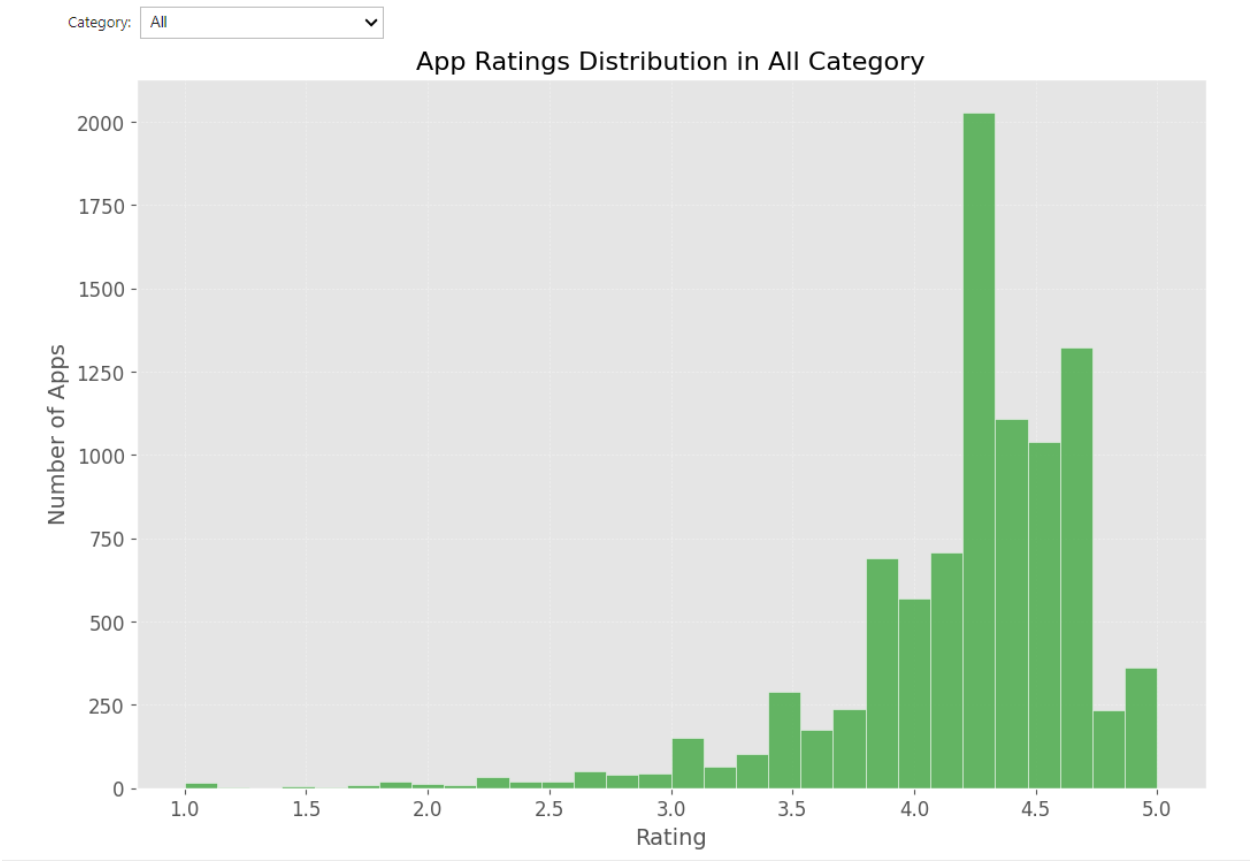
4. Interactive Exploration: Users can interactively explore the data by changing filters, immediately seeing the effects of their changes in the visualization. This interactive process allows for deeper data exploration and insight discovery.

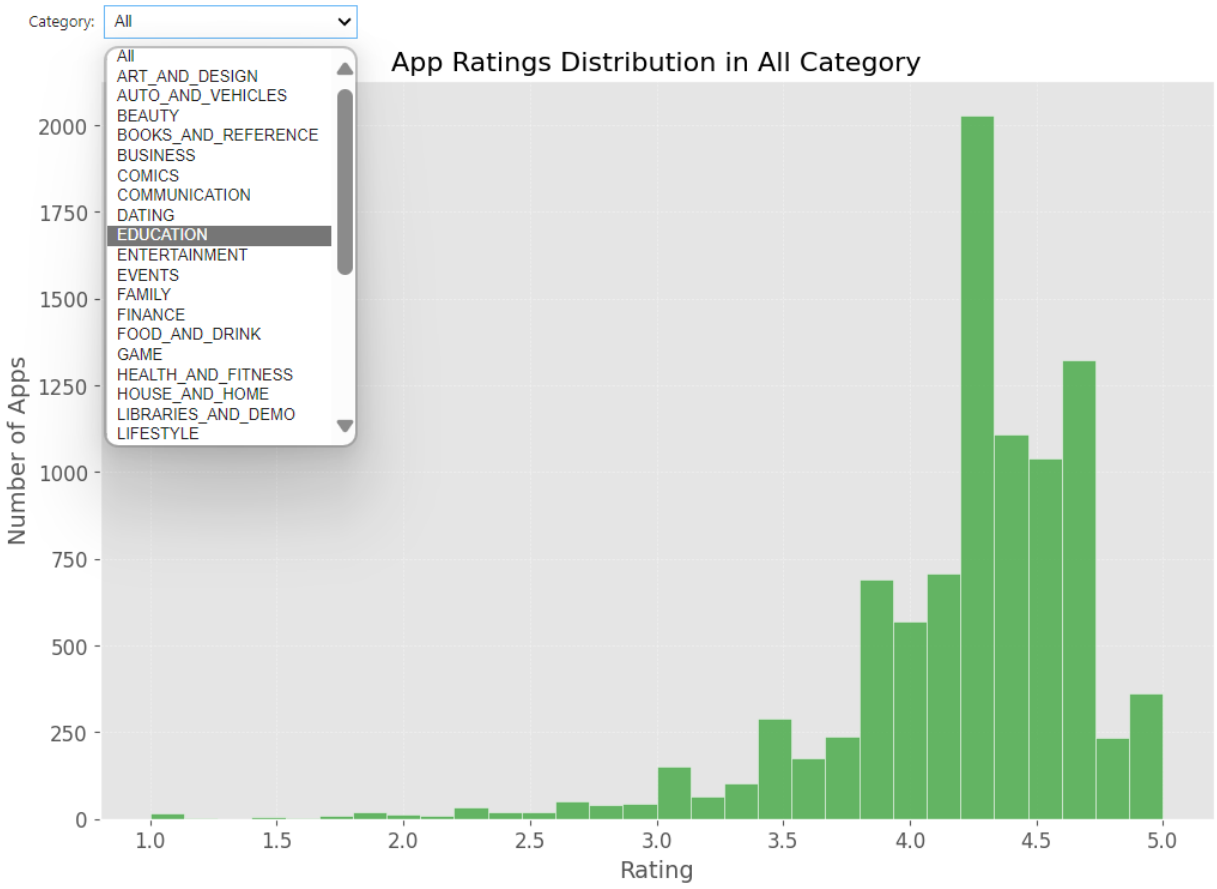
The original method is static, with predefined filters and a fixed visualization, while the enhanced method is dynamic and interactive, allowing users to explore the data in real-time through an interactive interface and immediately see the effects of their filtering choices on the visualization.

Application to Google Play Store Data

Considering the Google Play Store dataset, an enhanced filter interaction method could be a multi-dimensional filter bar that allows users to combine filters for different attributes like Rating, Reviews, and Category. As the user adjusts these filters, the dataset dynamically updates to show only the apps that meet all the selected criteria. This could be visually represented with immediate changes in a graphical representation of the data, such as a scatter plot or histogram, where each app is a point or bar that appears or disappears based on the filter settings.

Output:




Category:
All
All
ART_AND_DESIGN
AUTO_AND_VEHICLES
BEAUTY
BOOKS_AND_REFERENCE
BUSINESS
COMICS
COMMUNICATION
DATING
EDUCATION
ENTERTAINMENT
EVENTS
FAMILY
FINANCE
FOOD_AND_DRINK
GAME
HEALTH_AND_FITNESS
HOUSE_AND_HOME
LIBRARIES_AND_DEMO
LIFESTYLE
App Ratings Distribution in All Category
Number of Apps
2000
1750
1500
1250
1000
750
500
250
0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Rating

4. Domain users are app developers and marketers interested in understanding factors that contribute to an app's success. They aim to use this understanding to guide the development and marketing of their own apps.

Insight 1:

Correlation between ratings, reviews, and app category success when interaction methods are combined.
Combined Interaction Techniques : Investigate + Filter + Reorganize + Encrypt

Scenario and Insight:

Developers are interested in discovering whether the quantity of reviews and ratings and the degree of success in particular app categories are correlated. We can identify trends indicating that successful apps in popular categories (like "Games" and "Family ") typically have a high number of reviews and generally high ratings by navigating the dataset, filtering by categories with the most installs, rearranging the visualization to compare these categories side by side, and visually encoding the number of reviews and ratings.

Usefulness:

This information aids developers in comprehending the significance of user involvement (as demonstrated by reviews) and satisfaction (as demonstrated by ratings) for an app's success.

Insight 2:

The Effect of App Size on Ratings and Downloads in the "Education" Category
Combined Interaction Techniques: Select + Connect + Elaborate

Scenario and Insight:

Given that developers specialize in educational apps, they may be curious about how the size of the app affects ratings and popularity. This is because larger apps may contain more content, but users may be discouraged from downloading them if they are too huge. We find that medium-sized apps not too big to discourage downloads, not too small to be thought to be lacking in content tend to perform better in terms of downloads and ratings. This is because we abstracted the data to average ratings and download numbers by app size within the "Education" category, chose this particular category for in-depth analysis, and connected app size with ratings and download numbers.

Usefulness:

By balancing information richness with download convenience, developers can increase the success of their educational apps by following this insight when optimizing the size of their apps.

Insight 3:

Seasonal Trends in App Updates and Their Impact on Ratings

Combined Interaction Methods: Explore + Filter + Encode + Reconfigure

Scenario and Insight:

App developers may be curious about whether there is a seasonal pattern in the way that app updates affect user ratings. They may speculate that updates that are released at specific periods of the year such as right before significant holidays may receive higher ratings because they encourage users to interact with the app more or because they include seasonal features. We find that updates released in late November and December generally have a positive effect on app ratings by filtering data to only include apps with updates within the last year, encoding the update month as a color in a scatter plot, and investigating the relationship between update times and subsequent rating changes.

Usefulness:

This observation implies that scheduling app upgrades to align with holidays or seasonal events may be a calculated tactic to increase user satisfaction, particularly for applications that can include seasonal features or content.

References:

https://www.kaggle.com/code/zain280/google-play-store-apps-analysis

https://www.kaggle.com/code/martaseidler/google-play-store-apps

https://www.pewresearch.org/internet/2015/11/10/an-analysis-of-apps-in-the-google-play-store/

https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3833926

https://www.ijresm.com/Vol.3_2020/Vol3_Iss3_March20/IJRESM_V3_I3_38.pdf

Role of Madhavi:

Design and implement the visual representations of the data for the task1.

Worked on Developing an algorithm to enhance one of the interaction methods.

Worked on the Documentation.

Role of Payuktha:

Design and implement the visual representations of the data for the task2.

Worked on Developing an algorithm to enhance one of the interaction methods.

Collected all the references related to our dataset.