

# Seoul Bika Data Visualization

Luca Venerando Greco<sup>1</sup>

<sup>1</sup>University of Luxembourg

luca.greco.002@student.uni.lu

## 1. Introduction

This project focuses on developing interactive data visualizations using React and D3.js to represent multivariate data from a bike-sharing dataset. The main objective is to implement two synchronized visualizations: a scatterplot with 2D-Brush interaction and a second custom visualization (star coordinate plot) with brush interaction, allowing users to select and highlight data points across both views simultaneously.

This report will explore all the design choices that have been made toward the creation of the final visualization, as well as critical insights about the implementation itself.

### 1.1. The Dataset

The dataset has been provided as a `.csv` file containing data on bike rental in Seoul, it consists of more than eight thousand data samples, each containing thirteen columns of which eleven are numerical features and three are categorical.

One of these features is particularly tricky to work with: `Date`; since it is encoded using a `dd/mm/yyyy` date format, it was necessary to first parse it, treat it as an integer (converting it into the unix epoch time) and then format it back using `d3.scaleTime` for a nice visualization. No other information has been altered.

### 1.2. Categorical feature encoding

#### 1.2.1. Seasonal Colors

Since both visualizations share the need to define a way to represent the three categorical feature of the dataset, a common representation has been adopted. In particular the `season` attribute has been encoded using a colormap vaguely resembling that of the seasons represented.





Season	Color (Hex Code)	Color Preview
Spring	#abdda4	
Summer	#d7191c	
Autumn	#fdae61	
Winter	#2b83ba	

Table 1. Seasonal Colors

The colors used in this project were selected with ColorBrewer[1].

#### 1.2.2. Holidays and Non-Functioning Days

To represent the categorical features related to holidays and non-functioning days, different SVG icons have been employed. The following conventions were adopted:

- A **star SVG** represents a holiday.
- A **circle SVG** represents a regular day (non-holiday).
- For non-functioning days, a **gear SVG** is overlaid on top of the star or circle to indicate the day is not operational regardless of its holiday status.

This encoding ensures that users can quickly distinguish between holidays, regular days, and non-functioning days in the visualization.

Icon	Meaning
★	Holiday
●	Regular Day
☆	Non-Functioning Holiday
⦿	Non-Functioning Regular Day

**Table 2. Icons used for Holidays and Non-Functioning Days**

## 2. The two visualizations

### 2.1. Scatter Plot

The first visualization is a classical scatter plot, given the presence of more than two categories of data, a smooth transition allows the user to select two of the eleven numerical features to display along the two axes, and the transition lets the user follow the path of the clusters of points as they move around the visualization.

### 2.2. Star Coordinate Plot

The second visualization chosen is the star coordinate plot[3]. This visualization was chosen because the dataset presented has a medium amount of numerical features and the few categorical had a really nice encoding, therefore this visualization gives the possibility to address the main issue with a standard scatter plot, the inability to accommodate more than two or three dimensions. It is important to notice however that we are at the limit of how many features this visualization can handle (the paper makes an example with eight feature and we are using ten), differently from a parallel coordinates visualization plot, which can scale indefinitely provided enough horizontal space.

## 3. Interaction is the key

Interactions are always a really nice to have when visualizing data, but when dealing with this high amount of information, interaction becomes a need. In this section we are going to briefly explore the various interactions implemented in the visualization, most of these interactions are commonly shared by the two plots.

### 3.1. Brushing

One of the requirements for the visualization was to implement two synchronized brushes in the two different visualizations in order to highlight the selected points in both visualizations at the same time. To do so a redux slice has been used, providing the `selectedItems` state and the `updateSelectedItem` reducer. This shared information allows both visualization to gray-out unselected items, making selected items stand out.

#### 3.1.1. Optimizations

Given the very high amount of points, present in both visualization, using a classical approach to search for selected items is unfeasible, for this reason the use of `d3-quad-tree`[2] has been employed.

### 3.2. Tool tip

While both visualization are very effective in expressing relations between attributes, they lack the capabilities to inspect a single data point in depth. To accommodate for this, a tooltip functionality has been introduced.

When hovering on a point, its opacity will increase and the border will become visible to indicate that the point is on focus, then if the user right-click on the point, the tooltip will appear with all the information regarding the point until the cursor doesn't exit the circle.

The tooltip interaction and the hovering wouldn't be usually possible with the brush interaction, since the overlay of the brush would catch all the events, for this reason the actual event listener is attached to the brush overlay while the binding to the actual point is made through a search in the quadtree.

3.3. Axis Movement in Star Coordinate

The main strength of the star coordinate plot may as well be its biggest weakness. While it is able to visualize almost all the data, it lacks a structure from which extrapolate information. This problem is confronted in the paper, which embraces this issue by proposing the introduction of some interactions to move the axis around, this simple yet brilliant interaction lets the user orient and rescale the axis as they please, giving more or less importance to some feature, aggregating or opposing attributes and so on.

This wide range of possibilities gives the star coordinate plot a personalization potential hardly matched by other visualizations, the very same plot tailored to the need of the user by the user itself.

4. Pros and Cons

Pros	Cons
<ul style="list-style-type: none"><li>• Interactive visualizations (scatterplot and star coordinate plot) provide synchronized brushing for enhanced data exploration.</li><li>• Consistent and intuitive encoding of categorical features using colors and icons.</li><li>• Efficient brushing interaction enabled by D3.js quadtree optimizations for handling large datasets.</li><li>• Customization options in the star coordinate plot, such as axis movement and scaling, offer adaptability.</li><li>• Tooltip functionality allows in-depth inspection of individual data points.</li></ul>	<ul style="list-style-type: none"><li>• Scatterplot is limited to displaying relationships between two features at a time.</li><li>• Star coordinate plot struggles with higher-dimensional datasets, nearing its feature handling limit in this case.</li><li>• Implementation complexities, such as quadtree optimization and brush-event handling, increase development effort.</li><li>• User interface for customizing the star coordinate plot might overwhelm non-expert users.</li><li>• Encoding choices (e.g., icons for holidays) may require explanation for first-time users.</li></ul>

Table 3. Comparison of Pros and Cons for the Seoul Bike Data Visualization Project

5. Conclusions

The project successfully demonstrates how interactive data visualizations can provide powerful tools for exploring and analyzing multivariate datasets. By combining the strengths of a scatter plot and a star coordinate plot, both synchronized through brush interactions, users can effectively navigate and understand the relationships among the different features in the Seoul bike-sharing dataset.

Key design choices, such as the consistent encoding of categorical features, the use of an efficient quadtree structure for brushing, and the implementation of dynamic tooltip functionality, ensure the visualization is both responsive and user-friendly. Furthermore, the ability to customize the star coordinate plot by rearranging and rescaling axes empowers users to adapt the visualization to their specific needs, offering unique insights that static plots cannot provide.

While the star coordinate plot has limitations with higher-dimensional data, this implementation pushes its boundaries effectively. The inclusion of advanced interactivity highlights its potential as a flexible tool for data exploration, capable of revealing patterns and trends that might otherwise remain hidden.

Overall, this project has been an enjoyable and enriching experience, allowing me to explore new concepts and techniques. I particularly enjoyed implementing the star coordinate plot, which I found both fascinating and versatile. I look forward to utilizing this visualization method in future projects.

References

[1] Cynthia A. Brewer and Mark Harrower. *ColorBrewer 2.0*. 2024. URL: <https://colorbrewer2.org/#type=diverging&scheme=Spectral&n=>.

[2] *D3.js Quadtree*. Accessed: 2024-11-21. URL: <https://d3js.org/d3-quadtree>.

[3] Eser Kandogan. “Star Coordinates: A Multi-Dimensional Visualization Technique with Uniform Treatment of Dimensions”. In: *Proceedings of the IEEE Information Visualization Symposium* (Feb. 2001).