ECS272 Final Project Report
InterAxis: Steering Scatterplot Axes via Observation-Level Interaction

Yuchu Lei, Tianran Wang

March 2019

1 Goal

For the final project, we implement a visual analytic system called InterAxis based on the work of Kim et al. [1] The goal of this project is using interaction techniques to apply data visualization and analysis over multi-attribute data on a 2D scatterplot. Although we cannot show all attributes of a dataset on a planar at once, each time we can choose a subset among all attributes and then use scatterplot to visualize the dataset from the new aspect. The key point is to alter two axes in the scatterplot in a user-driven manner . InterAxis provides three types of interactions techniques: (1) attribute-level axis selection, (2) data-level axis steering, and (3) attribute-level axis manipulation. They will be discussed in the following section. The task of our group is to implement this system with all functionalities equipped. At the mean time, we will alter some widgets and add additional features to make this system easier to use. Finally, an evaluation will be given.

2 Implementation

Our implementation is based on D3 V5 toolkit. The paper provides two datasets to demonstrate effectiveness of this system: the car dataset and the crime dataset. In our system we only choose the first dataset because it's easy to think about a scenario when we use InterAxis to help us choose a car model. Although the car dataset contains 18 attributes, they depict a car on every aspect like type, size, price, and performance, etc. While for the crime dataset, attributes are less independent from each other because a single dimension may be split into several attributes, like those 4 race-related attributes and 4 age-related ones. Since the car dataset is already in csv format, the data preprocessing procedure is trivial. Here we just rename and remove some attributes after loading the dataset.

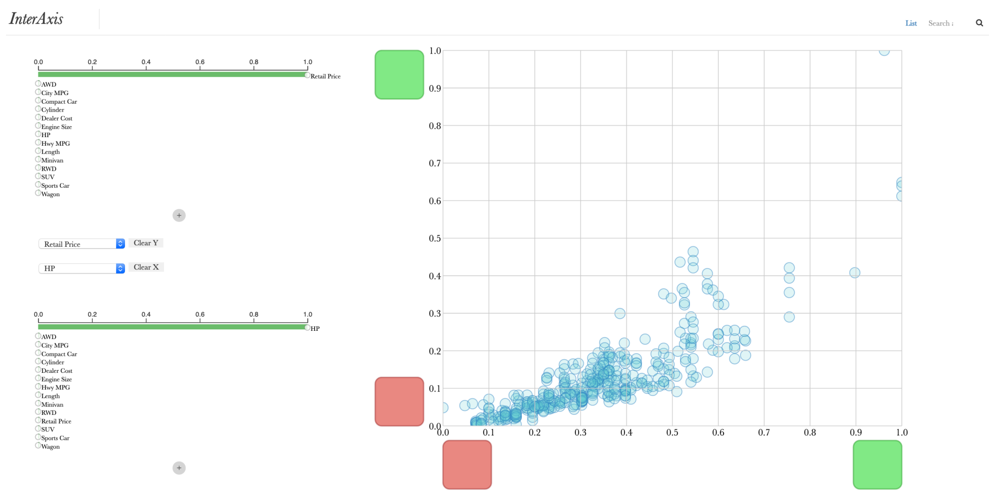


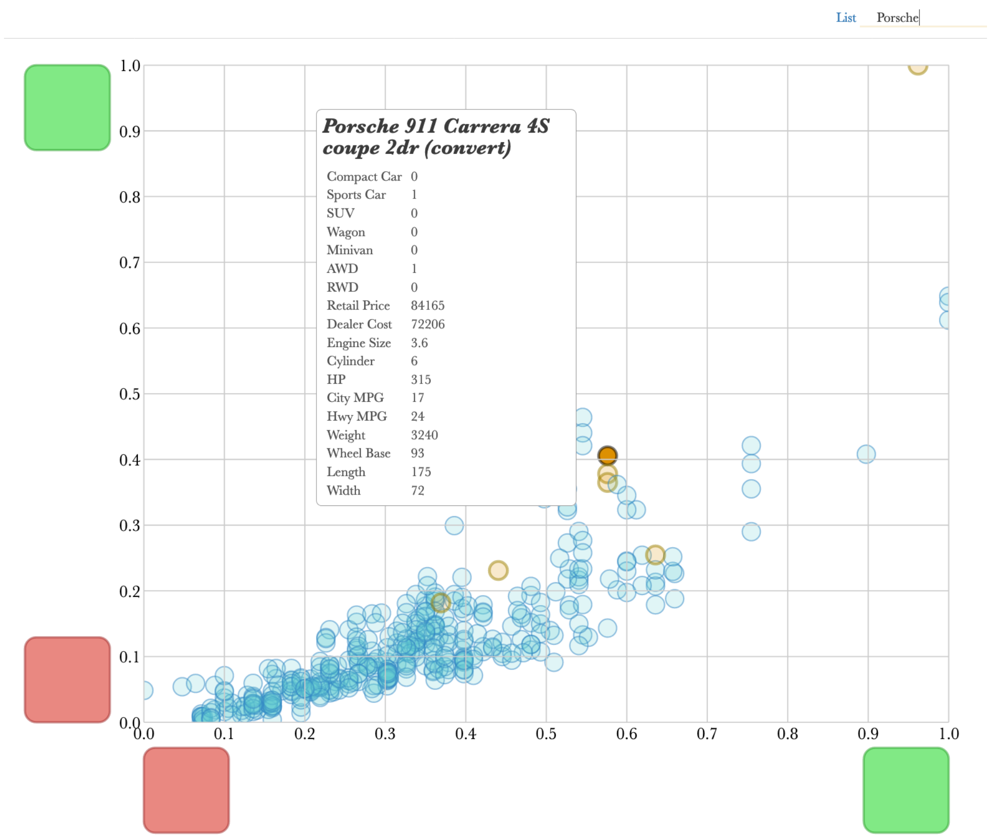
Figure 1: Overview of our implementation.

2.1 Interface Overview

The overview of our system is shown in Figure 1, it contains two parts: on the right there is the scatterplot view with 4 drop zones; while on the left there are two bar charts corresponding to two axes, and a combo box panel in the middle. We change the layout because in the paper's implementation, space left between two drop zones alongside one axis tend to be narrow and sizes of two bar charts are not the same. Here we increase the space utilization by separating components into two parts, so that two bar charts have the same size and the scatterplot tend to be a square.

2.2 Attribute-level Axis Selection

Combo boxes on the left contains all attributes. The default attribute of x axis is Horse Power (HP) and the default attribute of y axis is Retail Price. Each time only one attribute can be applied on one axis and the scatterplot will update correspondingly. Need to mention that in the scatterplot we do not use values of given attributes directly as coordinates, instead, we will first scatter given attribute domains to [0,1] and then use new values as coordinates. This is because domains of different attributes vary a lot and we need to have a normalization procedure for later attribute combination. Panning and zooming operations are allowed in the scatterplot. Also, you can drag ticks on either x or y axis to scale the range of axis. Once you hover on a data point, it will be highlighted and an information label will appear, which shows all attributes of this car model. Instead of hovering on each data point to find a specific car model, here we add a search box on the top right corner. As shown in Figure 2, data points with matched names will be highlighted in orange once you type in characters.



Porsche 911 Carrera 4S coupe 2dr (convert)

Compact Car	0
Sports Car	1
SUV	0
Wagon	0
Minivan	0
AWD	1
RWD	0
Retail Price	84165
Dealer Cost	72206
Engine Size	3.6
Cylinder	6
HP	315
City MPG	17
Hwy MPG	24
Weight	3240
Wheel Base	93
Length	175
Width	72

Figure 2: Searching for data points by name.

2.3 Data-level Axis Steering

The second interaction is to drag data points from the scatterplot into two drop zones at the high- and the low- end of the axis. The idea behind this operation is intuitive: if you like one car, put it in the green drop zone, otherwise put it in the red drop zone. According, a new axis is formed and it will be used to update the scatterplot and the bar chart. In the new scatterplot, data points close to the green point share more similarities with your favorite car, while those close to the red point are probably what you don't like. In the new bar chart, you can get an idea about how much a particular attribute is emphasized or de-emphasized. As shown in Figure 3, after selecting 5 data points into drag zones along x axis, we may conclude that this user probably wants an RWD sports cars most. Instead of using just a single green color like the paper shows, here we also colored low-end drop boxes and points within them with red, which gives users a clear remind that they are of those cars we will never consider. You can have multiple data points within one drop zone and you can also remove one from the drop zone by double click. The scatterplot and the corresponding bar chart will also update. This is an improvement for the original version because the previous one won't update if you deselect all points from drop zones. Moreover, in case you mix one data point with others in the same drop zone before deleting, you can hover on it in the drop zone and then see the highlight on corresponding point in the scatterplot and vice versa. Need to mention that when you click 'Clear X' or 'Clear Y' button, data points will be removed from corresponding drop zones, while drop zones located on the other axis will not be affected.

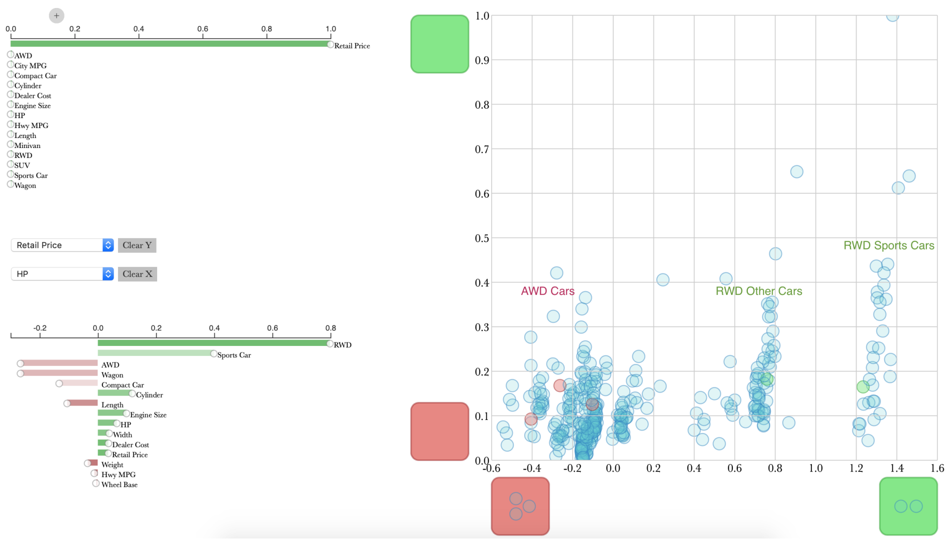


Figure 3: Dragging data points into drag zones.

2.4 Attribute-level Axis Manipulation & Math Behind

Apart from update the bar chart by dragging data points into drag zones, a direct way is to adjust the length of each bar in the interactive bar chart. Our version differs from the one in paper in that: bars with positive values are in green while those with negative values are in red; each bar has a circle at the top as a hint that it is stretchable. Value of each attribute in the bar chart represents its weight in the newly-formed axis. To get the coordinate of each data point alongside the new axis, we need to first calculate a linear combination of all attributes while each one's weight is its value in the bar chart. Then these linear combinations of all data points will be normalized and assigned as their coordinate. The math behind data-level axis steering is the same, first we get a mean data point for red drop zone and that for green drop zone if there are multiple points in each drop zone alongside given axis. Then we calculate the difference between two mean points and scale its domain to unit. Later for every data point, we get its coordinate by subtracting the red mean point and then scattering the remain to the new domain. This principle behind axis steering is illustrated in Section 3.3 of the original paper, which boosts our understanding of how this system works and how should it be implemented.

2.5 Coding Experience

During the implementing process, we found the most difficult part was not how to implement a chart or an interaction, but how to process and manage data. One example is to calculate the coordinate of data points. Each data point has actual values for all attributes, which are irrelevant with its coordinate. What's even worse is that every time an axis is updated, we need to calculate corresponding coordinates for all data points. We found ourselves easily making mistakes on calculating and retrieving data. On the other hand, there are so many interactions in this system and some of them are really hard to implement. For example, we took a long time to figure out how to enable panning and zooming in a plot. However, till now the zooming interaction in our scatterplot is still not smooth.

3 Evaluation

Followed the description and the detail of math calculation in this paper, we implement most features talked in the paper except the saving function because new axis can only be stored in current session, which is of no durability. Besides implementing three main interactions, we also add features we as below: layout optimization, adding red color for those dislike data points, enabling auto update for scatterplot, adding search function, and updating the style of bar charts. Compared with the one in paper, our version has a better UI and is easier to use. However, it still has some problems. Due to the way we select data point and attributes, results may vary a lot and mislead our decisions. For example, we cannot get too many insights if we tag data points that are far from the center as like or dislike. Moreover, if most data points in a dataset are similar to each other while there are still some points with extreme values, we will get a scatterplot with a dense layout in the center. Better visualization techniques can be applied in out system to make it robust to various datasets.

4 Collaboration

In this project, we group collaborate very well and together finished the UI development work. Besides Yuchu Lei implemented the interaction behind bar chart and the search function. While Tianran Wang implemented the interaction in scatterplot.

References

[1] H. Kim, J. Choo, H. Park, and A. Endert. Interaxis: Steering scatterplot axes via observation-level interaction. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):131–140, Jan 2016.