Data Visualization Report

Thomas Vroom
i6291496, t.vroom@student.maastrichtuniversity.nl

I. Chosen dataset

I chose the dataset "Nabijheid voorzieningen; afstand locatie, regionale cijfers" from the CBS as dataset for this project. This dataset contains information on average distances to common facilities such as hospitals, schools, train stations etc. To not make things overly complicated I only used the data from 2023 (the most recent year for which the data is complete), considered the data at municipality-scale (i.e. each municipality is its own spatial area), and used the following subset of columns:

Avg. Distance to Closest {GP, Pharmacy, Hospital, Supermarket, Primary School, High School, Highway, Train Station, Fire Station}

I used the "Wijk- en buurtkaart 2023" as the shape file for this data, also by the CBS. This shape file uses the same municipality codes as the dataset, making the transformation from the original dataset to geospatial data very straightforward. The dataset was preprocessed in the following way:

1) Select the relevant columns from the original dataset.
2) Remove rows that contain NaN (municipalities that do not exist anymore).
3) Merge the shape file and the dataset on the region codes.
4) Remove water areas from the shape file
5) Convert the coordinate system to EPSG:4326.
6) Export the merged shape file as GeoJSON.

The final dataset has 13 columns (features) and 342 rows (spatial areas; one per municipality).

II. Multivariate visualization

The first task is as follows: *"Compare the features of one selected spatial area to a set of other selected spatial areas"*. For the set of spatial areas I selected all municipalities from the province *Limburg*, with the goal of comparing them in a customizable subset of features.

A. Multivariate sketches

The first sketches for solving this task can be seen in Figure 1. If the image is not large enough, you can also find the original scan in the source code under *report/images/*.

B. Best 3 sketches

The first sketch considered can be seen in Figure 2. The **advantages** of this design include that it is the easiest to understand and use out of all my ideas, allows for a lot of customization, and allows for comparison of specific features for specific areas at a glance. The **disadvantages** of this design include that it does not scale well with more areas and / or features, with more of either potentially leading to an overwhelming chart.

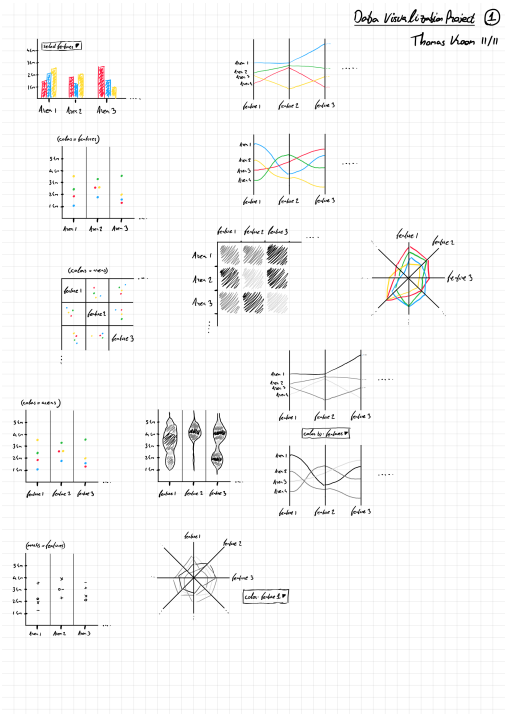

Fig. 1. First sketches for Task 1.

The second sketch considered can be seen in Figure 3. The **advantages** of this design include that it scales well with more areas and columns (up to a certain point), and it offers a lot of flexibility in what to visualize and how to do it. The **disadvantages** of this design include that you cannot find specific areas at a glance, and it is also difficult to compare specific areas.

The third sketch considered can be seen in Figure 4. The **advantages** of this design include that it adds new information (area under the line), scales well with more areas, and offers a lot of flexibility in what to visualize and how to do it. The **disadvantages** of this design include that it does not scale well with more columns, can feel very overwhelming, and is potentially more difficut to code.

In the end I opted for the second design, as it strikes a good balance with the flexibility of the first design, while not being as overwhelming to use as the third design. The disadvantage of it being difficult to compare specific areas should be mitigated in the future by allowing customizable

Fig. 2. The first in-depth sketch for Task 1. This design shows a grouped barchart with a shared y-axis, customizable selection of features through a drop-down menu, and highlightable bars.

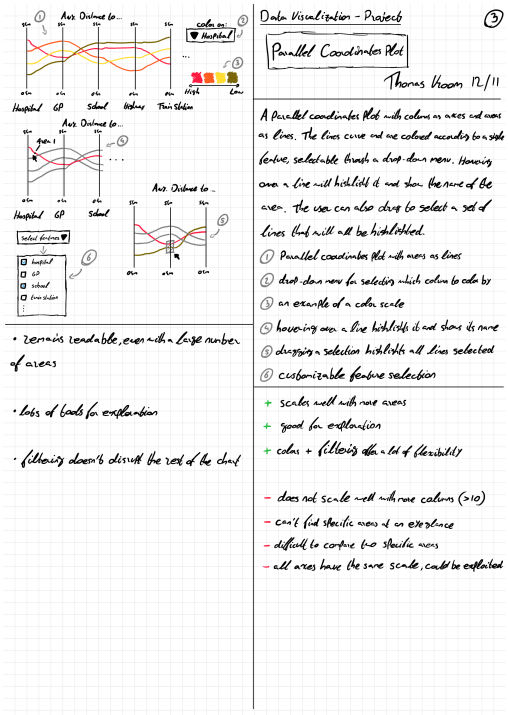

Fig. 3. The second in-depth sketch for Task 1. This design shows a parallel coordinates plot with curved lines, highlightable areas, customizable color-encoding through a drop-down menu, and customizable selection of features through a drop-down menu.

area selection, and since parallel coordinate plots are quite common in multivariate visualization there should be enough resources online to realize the design in practice.

C. Implementation

The full source code can be found in the attached .zip file. As described in there, I used existing code from the following sources: [1] [2] [3].

D. Performing task 1

Screenshots of the visualization can be found in Figures 5 and 6. Figure 5 shows the parallel coordinates plot with 9 axes for the 9 different features. For ease of comparison, all axes share the same units and scale. The color-encoding of the lines can be customized by a drop-down menu in the top left. Hovering over a line highlights it and shows the name of the area in the form of a tooltip (see Figure 6). Multiple areas can be selected at the same time by brushing the axes, and checkbox at the top left of the screen allow for the filtering of features.

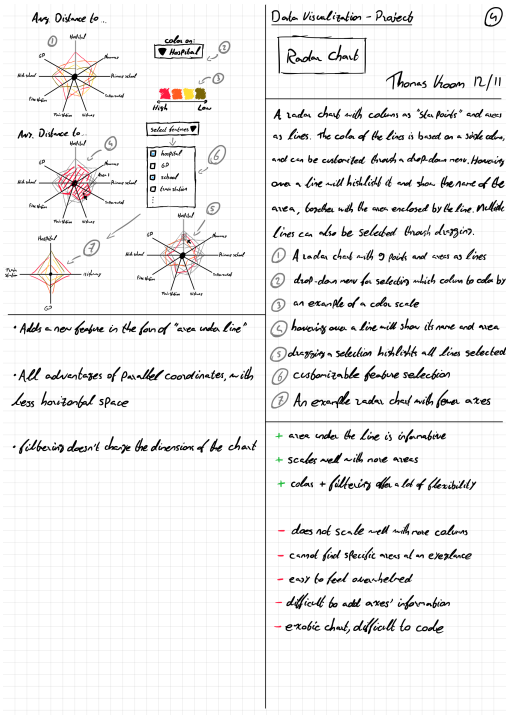

Fig. 4. The third in-depth sketch for Task 1. This design shows a radar chart with areas as lines, customizable selection of features through a drop-down menu, customizable color-encoding through a drop-down menu, and highlightable lines with colored-in areas.

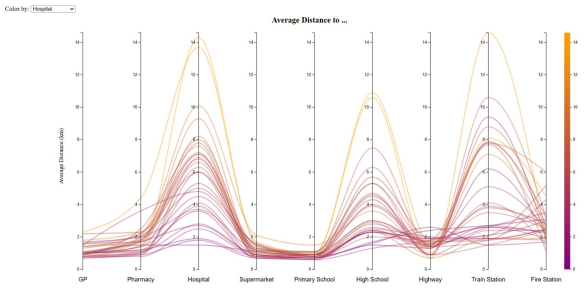

Fig. 5. A screenshot of the implemented visualization for Task 1.

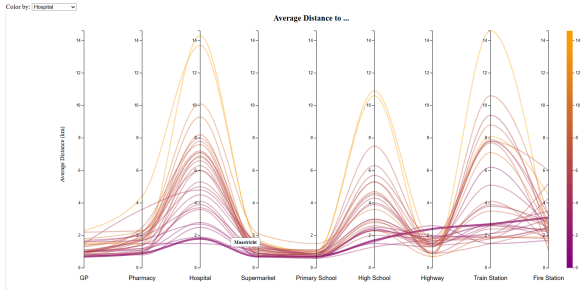

Fig. 6. A screenshot of the implemented visualization for Task 1, highlighting the entry for *Maastricht*.

III. Spatial visualization

The second task is as follows: *"Explore the geospatial distribution of both the absolute and relative feature distributions over the entire spatial region for up to 3 selected features."*.

A. Spatial sketches

The first sketches for solving this task can be seen in Figure 7. If the image is not large enough, you can also find the original scan in the source code under *report/images/*.

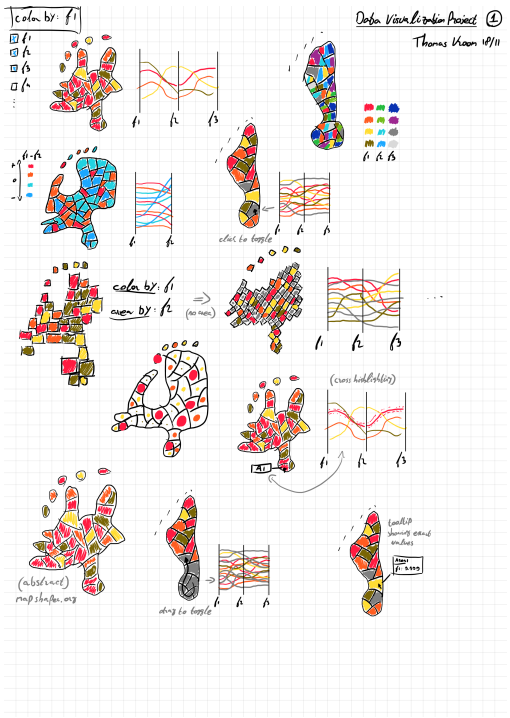

Fig. 7. First sketches for Task 2.

B. Best 3 sketches

The first sketch considered can be seen in Figure 8. The **advantages** of this design include that it is the easiest to understand and use out of all my ideas, and the land area mapping is somewhat relevant since we are plotting distances. The **disadvantages** of this design include that small areas are quite difficult to see, and mapping to land area still adds a bias for larger areas.

The second sketch considered can be seen in Figure 9. The **advantages** of this design include that it can encode more than one feature at a time, and serves as a more compact visualization due to not needing any filtering. The **disadvantages** of this design include that you cannot compare areas feature by feature, and that it still suffers from the same disadvantages as Figure 8.

The third sketch considered can be seen in Figure 10. The **advantages** of this design include that it can encode more than feature at a time, and that it does not suffer from the same area bias as the previous two designs. The **disadvantages** of this design include that it is more complicated to implement, and

Fig. 8. The first in-depth sketch for Task 2. This design shows a chloropleth map for visualizing the relative spatial distribution. Color and features of the parallel coordinates plot can be customized through user interaction, as can the selection / deselection of multiple areas through dragging or brushing. Hovering over areas or lines brings up the exact values of the relevant features. Compared to the other designs this design is not really "unique", it is more so that the other designs build on top of this one.

that the original shape can potentially become very distorted depending on the distributions.

In the end I opted for the first design, as it successfully completes the given task while not adding any additional complexity. The second design will likely lead to very interesting insights, but the drawback of not being able to directly compare areas on a single feature is too big to ignore. The third design covers a lot of the drawbacks of the first design, but with cartograms being so data-dependent it feels like too big of a risk to take not knowing how distorted the original shape will end up being.

C. Implementation

I reused a lot of code from the previous task, only adjusting where things are placed on the screen and how they interact with the other plot. For the spatial visualization, I used existing code from the following source: [4].

D. Performing Task 2

Screenshots of the visualization can be found in Figures 11, 12 and 13. Figure 11 gives a general overview of the UI. The controls in the top left allow for dynamic coloring of both plots and the checkboxes allow for filtering of the features for the parallel coordinates plot. Hovering over a spacial area highlights it and the corresponding line on the parallel coordinates plot, and vice-versa, and also brings up a tooltip showing

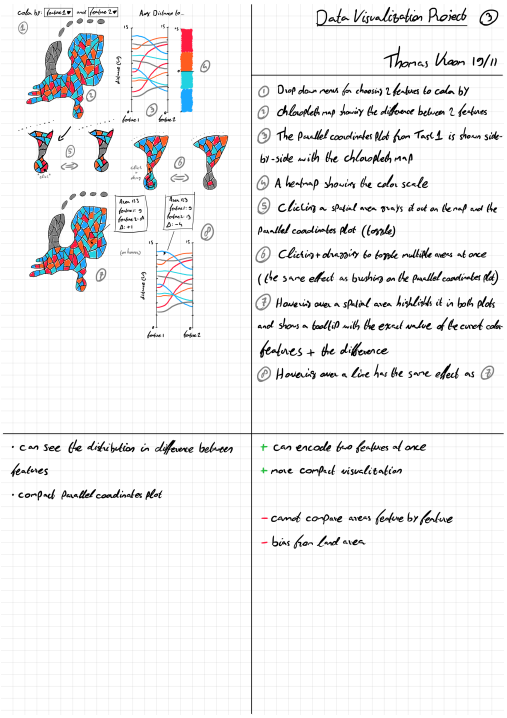

Fig. 9. The second in-depth sketch for Task 2. This design shows a chloropleth map similar to Figure 8, but this map instead shows the *difference* between two features.

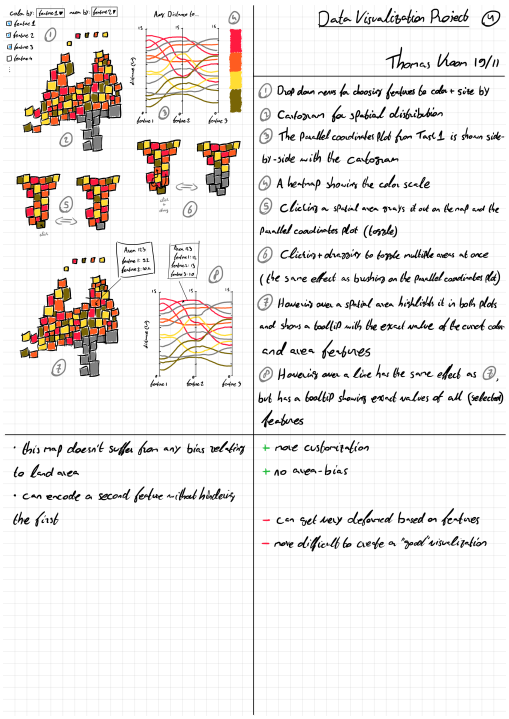

Fig. 10. The third in-depth sketch for Task 2. This design shows a cartogram for visualizing the relative spatial distribution. Both the color as well as the size of the areas can be customized by the user, and it also includes all other user interactions from Figure 8.

the exact value of the area for the feature corresponding to the color (absolute feature distribution). Figure 12 shows how multiple areas can be selected by brushing the axes on the parallel coordinates plot, and how that is also visualized on the chloropleth map, and Figure 13 shows how multiple areas can be selected by clicking and dragging on the chloropleth map, and how that is also visualized on the parallel coordinates plot (relative feature distribution).

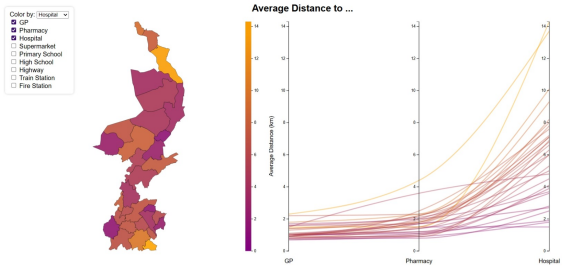

Fig. 11. A screenshot of the implemented visualization for Task 2.

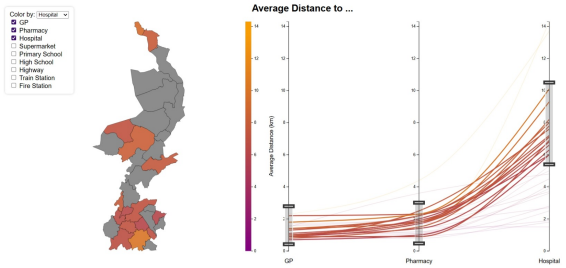

Fig. 12. A screenshot of the implemented visualization for Task 2, showcasing brushing on the parallel coordinates plot.

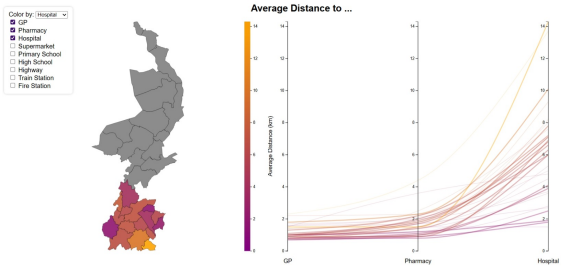

Fig. 13. A screenshot of the implemented visualization for Task 2, showcasing selection of spatial areas.

IV. Linked Clustering

The third task is as follows: *"Explore the similarities between spatial areas."*. For this task I used the k-means clustering algorithm to cluster areas by similatity.

A. Rough Interaction to Multivariate sketches

The rough sketches for integrating clustering with the multivariate visualization can be seen in Figure 14. I ended up going for the additional "Cluster" categorical axis in the parallel coordinates plot. This is easy to integrate with the existing code, intuitive to understand, scalable, and it also has the added bonus of allowing brushing to see elements per cluster.

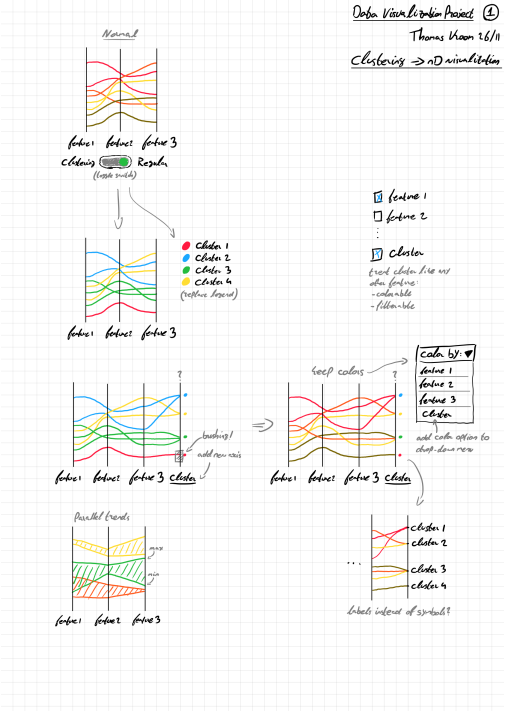

Fig. 14. Rough sketches for integrating clustering with the multivariate visualization.

B. Rough Interaction to Spatial sketches

The rough sketches for integrating clustering with the spatial visualization can be seen in Figure 15. I ended up going for a categorical color map that can be applied to the spatial areas by selecting "Cluster" as option in the drop-down menu for coloring. Similarly to the approach for the multivariate visualization, we are essentially treating the cluster of each element as an additional column, allowing for straightforward integration with the existing visualizations.

C. K-means Interactions

The rough sketches for interacting with the k-means clustering algorithm can also be found in Figure 15. These interactions are pretty limited since I opted for treating the clustering as an additional column, as this approach has the most straightforward integration with the rest of the visualizations and also has the lowest 'mental-interaction cost'. The only

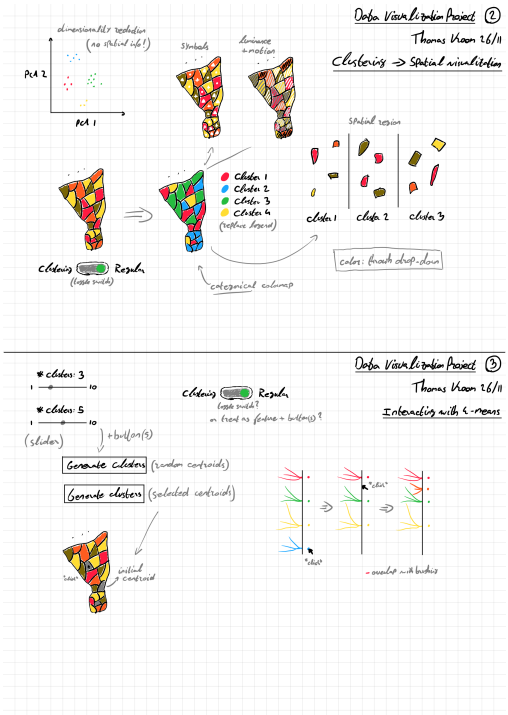

Fig. 15. Rough sketches for integrating clustering with the spatial visualization.

additional interaction with k-means that will be added is a slider for selecting the value of k, and a button for resampling the initial centroids (until this button is pressed, all clusters are cached so you can play around with other interactions without losing the assignment).

D. Implementation

I reused a lot of code from the previous tasks, only modifying things here and there to make the clustering act like an additional column in the dataset (with a categorical axis and color map). For the k-means clustering algorithm I used the following article as a reference, but I wrote the code myself: [5].

E. Resolving Task 3

Screenshots of the visualization can be found in Figures 16, 17 and 18. Figures 16 and 17 show the full visualization with spatial regions and parallel coordinate-lines colored according to k-means with $k = 3$ and $k = 5$ respectively. Note that clustering is performed according to the *current selection* of columns (i.e. only the columns filtered are used in the cluster algorithm). Since clusters are implemented like any other column, previous interactions such as cross-highlighting, filtering, and brushing also work for these cluster visualizations. Figure 18 shows a closer look at the k-means interaction (just below the filtering controls). The current assignment of clusters is cached per value of k, meaning you can freely explore filters and different k values without losing the current assignment. The cache is cleared and clusters are recomputed only upon pressing the "Resample Clusters" button.

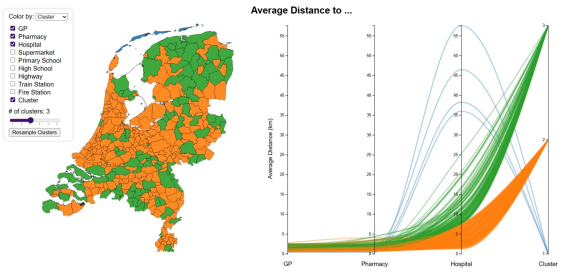

Fig. 16. A screenshot of the spatial and multivariate visualizations colored according to k-means clusters ($k = 3$).

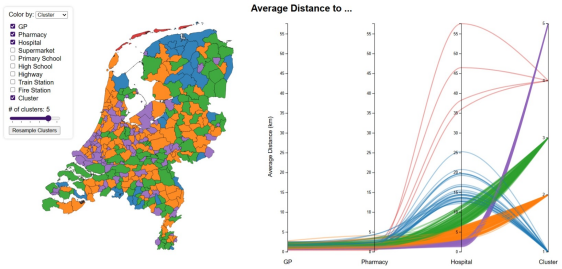

Fig. 17. A screenshot of the spatial and multivariate visualizations colored according to k-means clusters ($k = 5$)

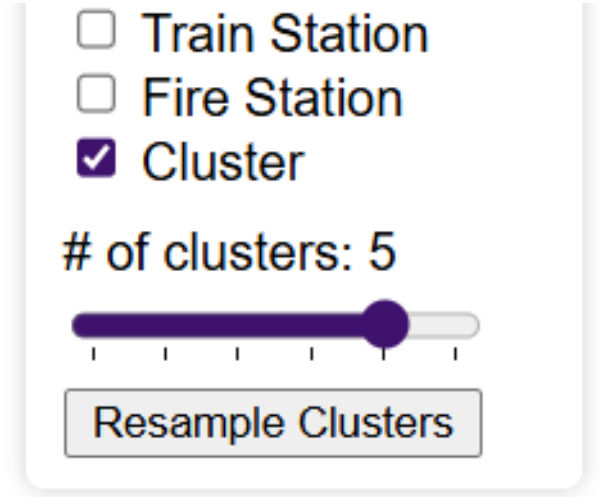

Fig. 18. A closer look at the k-means interaction below the filtering controls.