Austrian Journal of Statistics
2025, Volume 54, 71–99.
http://www.ajs.or.at/
doi:10.17713/ajs.v54i3.2058

Demonstrating the Capabilities of the lionfish Software for Interactive Visualization of Market Segmentation Partitions

Matthias Medl
Institute of Statistics
BOKU University
Vienna

Dianne Cook
Econometrics and Business Statistics
Monash University
Melbourne

Ursula Laa
Institute of Statistics
BOKU University
Vienna

Abstract

Market segmentation partitions multivariate data using some clustering algorithm, resulting in some number of homogeneous clusters of consumers for marketing purposes. Often this type of data has no clear cluster structure, that is, no separations or gaps between clusters of points exist, which is why this is considered partitioning rather than clustering. Understanding the differences between the clusters is typically done by examining single features. However, this can be inconclusive as multiple clusters might share similar characteristics on individual features and the market segmentation partition actually defines the clusters based on different linear constraints on the features. To understand what uniquely characterizes a cluster of customers, examining linear combinations of features may be helpful. This article introduces the R package **lionfish** that provides interactive and dynamic tools to facilitate the exploration and refining of market segmentations. The package integrates tour algorithms that use linear combinations of features to view high-dimensional data, from the **tourr** package, with Python-powered interactivity, allowing manual control, interactive selection, and multiple linked windows, to support revising the cluster memberships based on visual feedback. The focus is on the widely used k-means clustering algorithm, but the tools also support other algorithms. The utility of the software is demonstrated through three example analyses from the domain of market segmentation. The flexible, user-driven approach provided by package **lionfish** offers deeper insights into complex market behaviors, enabling more effective segmentation and enhancing strategic decision-making.

Keywords: interactive graphics, **tourr**, exploratory data analysis, R, Python, clustering, market segmentation.

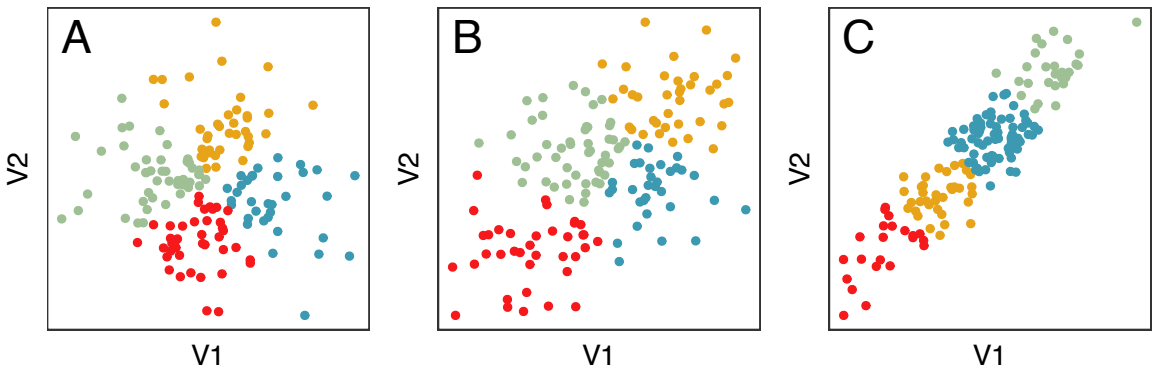

Figure 1: The different ways we might expect a clustering algorithm to partition 2D data with different association structure. If the correlation is high (C), the partitioning happens along the primary direction of the association.

1. Introduction

Clustering algorithms are often used to make large and complex datasets more digestible. For market segmentation, clustering algorithms are used to partition observations into a small number of subsets, by incorporating associations between the features. Market segmentation supports targeted approaches to different subsets of customers based on common traits. Clustering provides a data-driven solution to partitioning customer data into subsets for marketing purposes. Dolnicar, Grün, and Leisch (2018) provide an extensive overview of using clustering algorithms for market segmentation applications, and the advantage of visualization for the interpretation of the resulting clusters was demonstrated in Babakhani, Leisch, and Dolnicar (2019).

A difference between cluster analysis and partitioning is typically the nature of the data. With cluster analysis, we usually envision data that contains separated clusters, and a successful clustering result is one that divides the data based on these gaps. With partitioning, the role of the clustering algorithm is to propose smaller homogeneous subsets, whether there are gaps present or not, with potential for different marketing approaches as done with market segmentation. The shape of the data will affect how it is partitioned. Figure 1 illustrates the different ways that clustering might partition a 2D dataset into four subsets, when the correlation between the two features varies. When the correlation is high (Figure 1C), the partitioning will be along the combination of features that produces the highest variance. With lower correlation (Figure 1B), the clustering partitions the bottom and top, and divides the middle into two parts in the opposite direction. When there is no association, the partitioning is radial like a windmill (Figure 1A).

From the plots of the full 2D data, in each dataset, we can see how the data is divided into four parts. Typically though, the approach is to plot the partition on a single variable, as done in Figure 2. The histograms of the two features, V1, V2, show some relationships between the four subsets. For dataset C, the red cluster has low values on both features and the green cluster has high values for both. Looking more closely, the orange cluster has moderately low values on both and the blue cluster has moderately high values on both. We could infer from this that the partitioning is being done along an equal combination of V1 and V2, but *we cannot see the partition.*

When there are more than two features, histograms of the individual features are still commonly used to display the partitioning results. This means that the analyst likely cannot understand how the partitioning divides the data. All that they can observe is roughly how the individual features relate to the partition, which is useful but inadequate. To understand how the partition is formed in the data we need to view linear combinations of features, and often, focus on specific subsets. Figure 3 illustrates this approach, for the 2D data, where

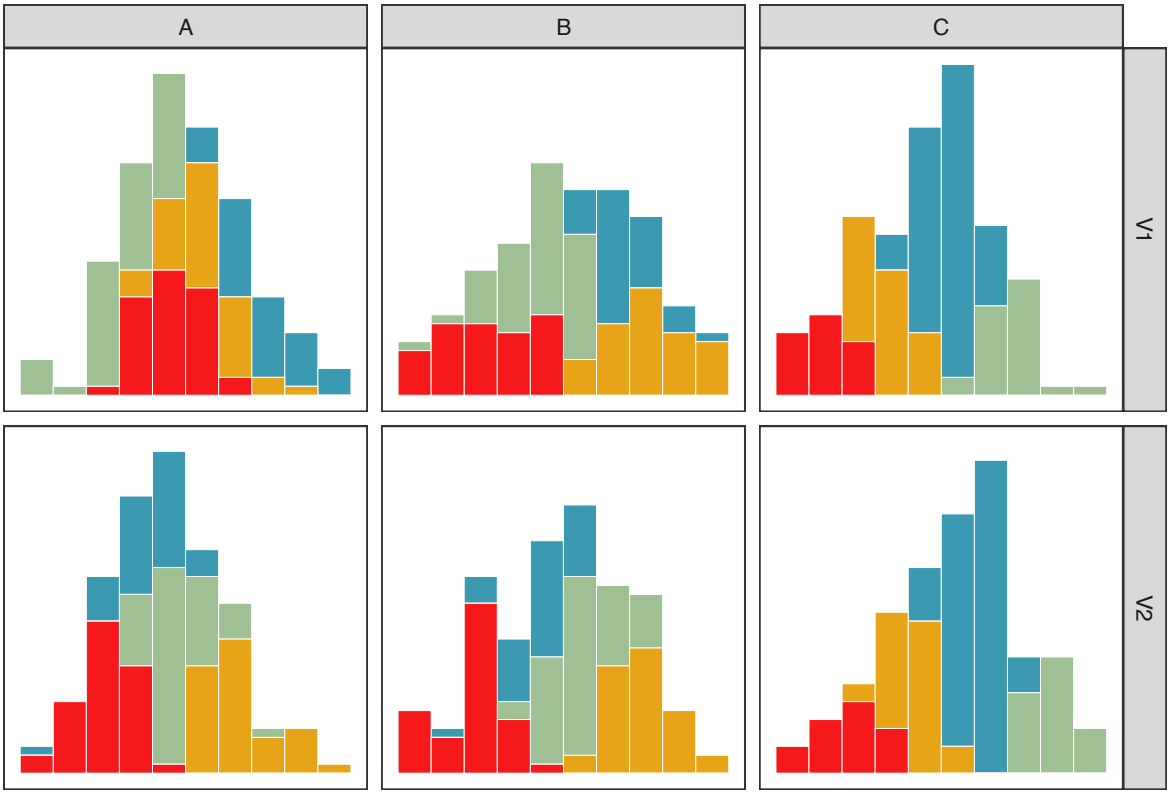

Figure 2: The typical approach to understanding the data partition is plotting individual features using histograms, where the subset is mapped to color. Here, the three datasets displayed in Figure 1 are shown in the columns, and the rows correspond to the two features, V1, V2. Differences between clusters can be seen, and interpreted on a single variable level, such as that the red subset has low values on both V1 and V2 in dataset C. However, how the data is divided cannot be seen.

this is not necessary, but will be useful to obtain a mental model of how this operates in higher dimensions. The partition corresponds to cuts of the data, and is so best viewed using jittered dotplots (using Clarke, Sherrill-Mix, and Dawson (2023)'s **ggbeeswarm** package). For dataset A, we plot V1 where the partitioning of the green and blue clusters happens. Note that the orange and red clusters are faded in the plot, because there is no single linear view where all clusters can be discerned and so one needs to focus on two clusters only. Compare this with the full data plot in Figure 1. The separation between the red and orange clusters could be seen if we plotted V2, and subsetted the data to these two clusters. Similarly, other pairs of clusters would be examined. For dataset B, the linear combination that is roughly a contrast of V1 and V2 shows the distinction between the blue and green clusters, and because there is no single linear combination revealing all clusters we would choose a different linear combination to examine the distinction between other pairs of clusters. In dataset C, all four clusters can be seen when a linear combination constructed from a roughly equal combination of V1 and V2 is used. Here this is actually computed by using the first principal component.

In a realistic, higher-dimensional setting, dedicated approaches for the visualization of cluster analysis results (Leisch 2008) can provide more detailed insights. For example, neighborhood graphs (Leisch 2010) can be used in combination with linear projections, to include information about cluster separation. This is in particular interesting when looking for projections that separate one of the clusters from the rest, as possible when using the methods suggested in Hennig (2004). The resulting view will often show other clusters as overlapping, even when they are clearly separated in the full space.

In general, with data having more than two features, tour methods to view high dimensions can be used to do this visualization. A tour (Asimov 1985, and see Lee, Cook, da Silva,

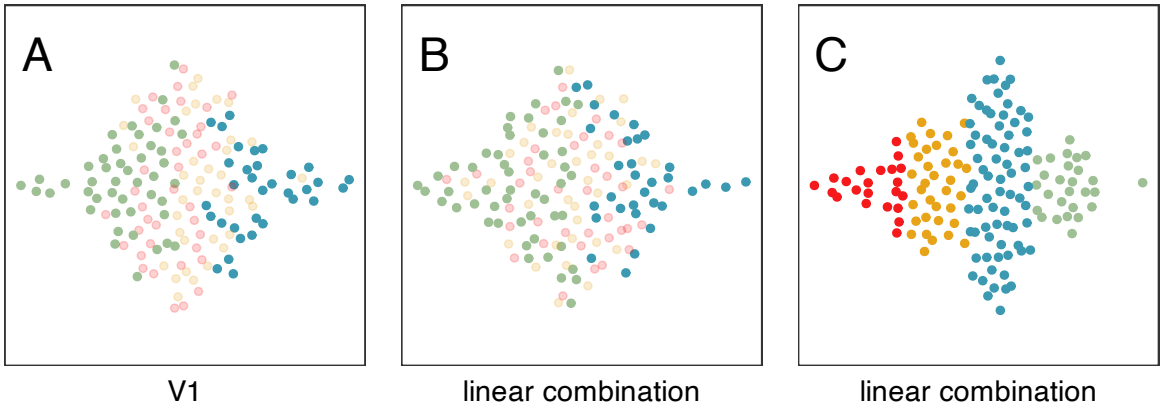

Figure 3: The way to examine partitions in high dimensions is to examine linear combinations, for each of the datasets. The three jittered dotplots show the views of the three datasets (A, B, C) revealing the partitioning. For datasets A and B, the red and orange clusters have been faded out in order to focus on the blue and green subsets, because there is no single linear combination where a separation between all clusters is visible. All four subsets can be seen for dataset C, because there is a linear combination which distinctly splits the clusters.

Laa, Spyrison, Wang, and Zhang 2022) is used to show scatterplots of linear combinations of features and thus provides views like that in Figures 1 and 3 where distinct differences between clusters can be observed. High dimensions are still tricky, and a combination of animations of the linear combinations, and interactive control (Cook and Buja 1997; Laa, Aumann, Cook, and Valencia 2023) over the combinations is important. A scatterplot of a combination of features can be considered to be a projection of the data, and thus like a shadow of a 3D object, some aspects of the data (object) can be obscured. Using slices of the projected data (Laa, Cook, and Valencia 2020) can be a useful addition to projections. This paper illustrates how to do this to better understand partitioning results for multivariate data.

In this article, we will (i) describe the **lionfish** software (available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=lionfish`) and (ii) demonstrate its utility through three example analyses in the domain of market segmentation using k-means clustering algorithms. The paper is organized as follows. Section 2 describes the software and its attributes. The user workflow is explained in Section 3. Illustrative market segmentation case studies using various tourism datasets provided in Dolnicar *et al.* (2018) can be found in Section 4. Sections 5 and 6 discuss the limitations and potential future developments of package **lionfish**.

2. Interactive interface with lionfish

One of the aims of this work is to build an interface that allows for interactive exploration of clustered multivariate data. In some combination of the features, as provided by a tour algorithm, one may be able to see the separations between the clusters as illustrated in the 2D example in Figure 1. The grand tour (Asimov 1985) provides a sequence of projections that is essentially an interpolated random walk over all possible projections. It is useful to get an overview of multivariate data. A manual (Cook and Buja 1997) or radial (Laa *et al.* 2023) tour is used to control the contribution of a single (or multiple) features to a projection, typically using it to change the projection from including or excluding the variable. It is useful for assessing the sensitivity of the clustering to particular features. Lee *et al.* (2022) provide an overview of the many different tour algorithms.

While tour animations are best obtained within R (R Core Team 2024) using the **tourr** pack-

age (Wickham, Cook, Hofmann, and Buja 2011), it does not enable the interactivity required for example for a manual tour. Interactive graphics are available when using Javascript, as implemented in the **detourr** package (Hart and Wang 2023). This allows the replay of a recorded tour path with interactive graphics, and can also be linked with additional displays, but lacks capabilities for manual tours. The objectives for our intended interactive interface are to enable:

1. Visualizing the distinctions between clusters based on combinations of features using various tour methods, but most importantly the manual tour.

2. Interactively selecting datapoints (brushing) to refine the cluster solution, as done with spin-and-brush tools (Cook and Swayne 2007).

3. Linking multiple displays to focus on particular clusters, and simplify the problem to better understand the clustering solution.

4. Updating displays based on user selections such as feature selection or cluster selection, or re-scaling.

To integrate user interactions with the capabilities of the **tourr** package an active communication with the interface is required. For example, we may wish to explore the local neighborhood of a projection selected by the user with a local tour animation provided by **tourr**, or we may want to optimize a guided tour path using subsets identified via brushing. Our solution is to use Python (Van Rossum and Drake 2009) for high-performance interactivity, through the packages **Tkinter** (Lundh 1999), **CustomTkinter** (Schimansky 2024), and **matplotlib** (Hunter 2007), with integration to the **tourr** package (Wickham *et al.* 2011) via **reticulate** (Ushey, Allaire, and Tang 2024), a framework that facilitates seamless interoperability between Python and R. Through **matplotlib**, Python provides interactivity on the plots, allowing elements to be directly selected or changed, and low-level bit blit control (for details see Section 2.4), enabling efficient updates and rendering of plots. **CustomTkinter** is a framework that allows users to effortlessly integrate **matplotlib** into a graphical user interface (GUI). An R native approach would be to use **shiny** (Chang, Cheng, Allaire, Sievert, Schloerke, Xie, Allen, McPherson, Dipert, and Borges 2024) for the GUI, but this would not provide interactivity directly on the plots (e.g., Laa *et al.* 2023) or low-level bit blit control.

The interface was implemented in the R package **lionfish** and offers a variety of linked interactive plot types, providing users with the flexibility to visualize their data from multiple perspectives. The ability to navigate through various projections of the displayed tours directly within a GUI enables users to explore different aspects of the dataset. Furthermore, users can initiate new tours directly from the interface. The GUI also supports interactive feature selection, allowing users to specify which subset of features should be visualized in the plots. Once users have identified interesting views or settings, **lionfish** allows them to save the displayed projections, subsets, and plots. This functionality ensures that analysis states can be preserved for further examination or reporting, making the package particularly useful for iterative analysis where findings may need to be revisited or shared with collaborators.

With its high level of interactivity, performance, and ease of use, **lionfish** streamlines the exploration of complex datasets, offering a powerful tool for researchers working with high-dimensional data.

2.1. Overview of the graphical user interface

The **lionfish** GUI can be launched using the function `interactive_tour()`. At minimum, this function requires the data and a list of plot objects to display (typically a 1D or 2D tour, a barchart, scatterplot, etc.). Optionally, the user can define an initial clustering of the data as cluster memberships matching the rows of the data, the layout of the plots, the number of available subsets, the size of the GUI, initially selected features for display, a scaling factor for

Figure 4: Overview of the GUI. A: Sidebar controls; B: Display area; C: Feature selection checkboxes; D: Subset selection with color indicators; E: Frame selection interface; F: Interfaces for adjusting the number of bins of histograms, animating tours and blending out projection axes with a low norm; G: Save and load buttons; H: Interface for starting new tours; I: Metric selection interface.

the projected data and a few minor plotting instructions such as color schemes. If no initial clustering is provided all data will be in one subset, and partitioning can be performed later via manual selection. The `n_subsets` argument fixes the number of available subsets of the GUI upon startup. If `n_subsets` is larger than the number of initially provided subsets, then the extra subsets will be empty, but can be filled via manual selection. This can be useful when a user wants to shift some data of special interest (e.g., outliers) into an extra set or redefine parts of a clustering solution as shown in the case study in Section 4.2. All features in the data are expected to be numeric, so categorical features should be converted prior to using the GUI. If they are not, then the conversion will be automated in the data pre-processing, for example, a variable called "fruit" with categories "banana", "orange", "peach" will be recoded as 1, 2, 3, respectively. The plotting instructions for each display to be shown have to be provided in form of a named list containing a `type` and an `obj`. The `type` element specifies the type of display to generate, such as `"scatter"` for a scatterplot or `"2d_tour"` for a 2D tour. The `obj` element further defines the properties of the chosen display. For example, to create a 2D tour, the user must provide a '`history_array`' object produced by the **tourr** function `save_history`. For a scatterplot, the user needs to provide a character vector specifying the names of the features to be displayed. In this article, the focus is on the use of **lionfish** for exploring k-means clustering results in context of market segmentation, but the package website contains examples of usage for other applications, animated examples showing the interactivity, and details on function arguments.

The GUI is divided into two main sections: a sidebar on the left, which contains a comprehensive set of interactive controls (Figure 4A), and the display area on the right (Figure 4B), where the selected plots are shown. At the top of the sidebar, users can select and deselect features using checkboxes (Figure 4C), thereby controlling which features are displayed in the plots. Below this, a list of the currently loaded subsets of the data is displayed. The subsets can be defined by the user when launching the GUI. In the context of the market segmentation use cases discussed in Section 4, these subsets are clusters. However, depending on the specific application subset labels from other sources may be used, which is why they are referred to as subsets in this section. Each subset has its own checkbox to designate the active subset (Figure 4D).

When datapoints are manually selected in the plots, they will be assigned to the active subset and be colored accordingly. For scatterplots, datapoints can be selected by encircling them directly on the plot while holding down the left mouse button. For barplots, clicking on a specific bar selects the data represented by that bar. The colored boxes next to the subset names indicate the assigned colors for the datapoints. Clicking on these boxes adjusts the transparency of the points, which is helpful for highlighting and comparing subsets. Subsets can also be renamed using the provided text boxes. The `Reset original selection` button allows users to revert the subset selections to their initial state. All plots displayed in the GUI are linked, meaning that changes in one plot will affect the other plots. For instance, if observations are moved from one subset into another subset in one plot then the reassignment will also occur in the other plots.

The frame selection interface (Figure 4E) shows the frames of the currently displayed tours. We can see that "Plot #1" (top left) displays the eighth frame/projection of the displayed tour and "Plot #3" (bottom left) the fifth frame/projection. The menu also enables users to jump between frames/projections of displayed tours by manually typing the desired frame into the respective textbox and pressing the `Update frames` button. One can also move through the projections of tours by pressing the arrow keys.

Below this, there are three additional interfaces (Figure 4F). The "Number of bins of histograms" can be used to adjust the number of bins of displayed histograms – more bins result in higher resolution but slower display updates. The "Animate" interface allows users to animate tours as slideshows. After checking the `Animate` checkbox the tour displays will automatically shift to the next frame/projection after the amount of time specified in the textbox next to the `Animate` checkbox (interval specified as 1 second in Figure 4F). The "Blend out projection threshold" interface offers the functionality to hide projection axes with a norm smaller than a chosen threshold. In Figure 4F the checkbox is not checked, meaning that this option is currently disabled. If it were to be checked, all projection axes would be blended out since the value in the textbox is set to 1 and the norm of the projection axes cannot be larger than 1. A sensible setting for the "Blend out projection threshold" is usually between 0.1 and 0.3. This functionality can be helpful as projection axes with a small norm have little influence on the shown projection, and displaying all of them can be distracting.

Users can also save and load projections and subsets using the respective buttons (Figure 4G). The `Save projections and subsets` button allows users to preserve the current state of their analysis. This not only includes the displayed projections and active tours, but also various minor settings, such as the highlighted subset or the value for the "Number of bins of histograms". These states can be recovered by pressing the `Load projections and subsets` button, which spawns a file browser with which one can select a folder containing previously saved files.

New tours can be started directly from within the GUI (Figure 4H). The options for new tour paths are: a local tour around the currently shown projection and guided tours that search for projections based on the holes, or the linear discriminant analysis (LDA) index. The holes index is sensitive to projections with few points in the center of the projections and the LDA index aims to maximize the distance between the centers of the selected subsets

(Lee, Cook, Klinke, and Lumley 2005; Cook and Swayne 2007). To start new tours, users first have to select the desired type of tour with the dropdown menu and then press the `Run tour` button. The new tours will then be calculated and displayed. The titles of the tour displays (Figure 4B top left and bottom left) will indicate which tours are currently shown. We can see in the titles of the displays that both displays currently display the originally loaded tours. If a user for instance were to initiate local tours, the display titles would change to "Local tour". The `Reset original tour` button can be pressed to return to the originally loaded tours. Additionally, all tour displays allow users to perform a manual tour; by right-clicking and dragging the arrowheads of projection axes, the respective projection is recalculated accordingly. This allows for manual exploration of the data.

At the bottom of the interactive controls users can select different metrics for some plot types, e.g., heatmaps (Figure 4I).

2.2. R/Python interface

The majority of package **lionfish** was written in Python. The R side of the package handles setting up the Python environment where the interactive interface is being run, launching the interactive tour, and generating new tours when initiated through the interface. To incorporate the functionality of the **tourr** package without translating large portions of its code from R to Python, the **reticulate** package was used. This approach allows **lionfish** to automatically benefit from updates to **tourr** and avoids the necessity to re-implement code in Python that was already written in R.

To minimize inefficiencies associated with cross-language communication and to simplify debugging, **tourr** functions were accessed only when necessary and otherwise implemented directly in Python. For instance, the orthonormalization of projection axes via the Gram-Schmidt process, linear transformations of the data with the projection matrices, or scaling of the data with the half range parameter for visualization purposes, were implemented in Python. Implementing these functions directly in **lionfish** improves its stability as future updates of **tourr** or **reticulate** could affect proper interaction between the packages.

2.3. Structure of the Python code

The **customTkinter** class '`InteractiveTourInterface`' represents the central component of the Python code. This class centrally stores attributes related to all plots, such as the dataset, sub-selections, feature selections, and other shared information. Plot-specific data is organized in dictionaries (the Python equivalent of named lists in R), including the display type, construction instructions, tour projections (only in case of tour displays), color schemes for the displayed data, and, where applicable, the selector classes ('`BarSelect`' and '`LassoSelect`') and manual projection manipulation classes ('`DraggableAnnotation1d`' and '`DraggableAnnotation2d`').

The selector classes handle the behavior when users manually select datapoints to move them to the active subset. After a selection is made, the selector class updates the centrally stored sub-selection attribute and ensures all other displays reflect these changes. The manual projection manipulation classes construct the arrows representing the projection axes in the displays and manage the manual adjustment of projections, thus enabling manual tours. Users can right-click and drag the arrowheads to modify the projections, after which the class orthonormalizes the projection axes and updates both the projection and the transformed data accordingly.

2.4. Fast-drawing enhancements

The implementation of bit blitting was crucial to ensuring fast plot updates and providing a smooth user experience. With bit blit, the static elements of the display, such as the outer frames of the plots, are stored as a background image. When a plot is manipulated, only the

affected plot is updated, and within that, only the interactive elements, such as the datapoints and projection axes in a 2D tour, are rendered on top of the background image.

In practice, this means that the background, without the interactive elements, must be captured either during initialization or after major updates. The entire plot is first rendered without the interactive elements, the background is then saved, and finally, the plot is redrawn with the interactive elements blended in. Since this process is relatively slow, full updates are only triggered during initialization or after significant changes, such as modifications to the set of active features.

3. Workflow with the lionfish package

While package **lionfish** can be used with data subsets of any origin, or even without any subsetting at all, we will focus on analyzing cluster solutions in this section. Before launching the **lionfish** interface, the user should have performed a partitioning of their choice, and provide the initial clustering solution to the interface. They can then launch the interface to explore and potentially refine this solution.

3.1. Feature and cluster relationships

A first step is to assess which features are important for the cluster solution. The interface provides different capabilities that support feature selection: tour view and summary heatmaps.

The 1D and 2D tour views can be used to understand the sensitivity of the subsetting to individual features or interactions thereof. Examples of these display types can be seen in Figure 4 where a 2D tour projection is shown on the top left and a 1D tour projection on the bottom left. The arrows shown in the plots indicate the loadings of the linear combination of the projection vector (1D projection) or matrix (2D projection). Short arrows indicate that the loading of a given feature is small, thus the influence of that feature on the projection is small. On the other hand, long arrows indicate a large influence on the projected data. Furthermore, we can identify relationships between the loadings and the data. Looking at Figure 4, we can see in both projections that the blue observations are in the direction that the "alpine skiing" axis points to. We can infer from this that the blue observations generally had a large positive value of "alpine skiing". By clicking and dragging an arrowhead with the right mouse button we can interactively change the loadings of the displayed projections. This functionality is referred to as a manual tour. It can be used to change the influence of the features interactively by turning the data in multi-dimensional space and projecting it into one or two dimensions. This is useful for examining the relationship between the features and the separation between clusters. The ideal starting view is that obtained through a projection pursuit guided tour (Cook, Buja, Cabrera, and Hurley 1995) that was optimized for the separation between the labeled subsets (Lee *et al.* 2005).

The summary heatmaps provide an overview of the cluster compositions relative to the features. An example can be seen in Figure 4 in the top right. Consider the matrix,

$$C = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1p} \\ c_{21} & c_{22} & \dots & c_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ c_{k1} & c_{k2} & \dots & c_{kp} \end{bmatrix},$$

where c_{ij}, $i = 1, \dots, k$ (number of clusters); $j = 1, \dots, p$ (number of features) contains a summary statistic of each feature in each cluster.

Because our examples use binary features, c_{ij} is determined as the number of 1's of feature j in cluster i. There are several ways that these values can be normalized to examine different

aspects: $f^o_{ij} = \frac{c_{ij}}{n}$, $f^c_{ij} = \frac{c_{ij}}{n_i}$ and $f^f_{ij} = \frac{c_{ij}}{n_j}$, where n_i, n_j are the row and column totals. The first, f^o is the overall fraction, where counts are normalized by the overall number of observations n. It gives a quick overview on the overall magnitude of the features. The second, f^c is normalized relative to the size of each cluster (row sum of C) and can be considered the distribution of features in each cluster. It is useful for comparing the composition of each cluster relative to the features. For example, for $p = 4$, $f^c_{1j} = (0.9, 0.2, 0.1, 0.1)$ suggests that high values (1's) of feature 1 distinguish cluster 1, and that the values of the other features are low. In the context of the later examples, this would mean cluster 1 contains tourists that especially engaged in activity 1, but not in activities 2, 3 and 4. This normalization produces what is called the **intra-cluster fraction**. In the case of binary features, it equates to the cluster means.

The last, f^f, is normalized relative to each feature (column sum of C), which can be considered to be the distribution of clusters on each feature. It is useful to examine how features are related to a cluster. Considering the last metric, for example, if $f^f_{i1} = (0.2, 0.7, 0, 0.1, 0)$ $(k = 5)$ would indicate that high values of feature 1 primarily are in cluster 2. In the context of the later examples, this would mean activity 1 is most commonly listed in cluster 2. This normalization produces what is called the **intra-feature fraction**. It also has to be noted that this metric is heavily influenced by the cluster sizes. Consider a feature that is uniformly distributed across all clusters, then the largest cluster would receive the highest intra-feature fraction for that feature.

In this example, while the intra-cluster fraction for cluster 1 and feature 1 is notably high at 0.9, the intra-feature fraction is comparatively low at 0.2. Applied to the subsequent analysis, this indicates that 90% of tourists in cluster 1 participated in activity 1, yet only 20% of all tourists who engaged in activity 1 were part of cluster 1. The majority of individuals participating in activity 1 were members of cluster 2. This discrepancy may be attributed to cluster 1 representing a relatively small subset of tourists, characterized by a pronounced preference for activity 1 and a lack of interest in other activities.

The metrics currently implemented in the heatmap interface, as described in this section, are designed for binary data. To expand the heatmap interface to accommodate ordered categories or numerical scores, one could indicate the cluster and feature means on the edges of the heatmap, with the cluster-specific feature means displayed within each element. This, however, has not been implemented yet.

3.2. Subset selection

The spin-and-brush approach suggests to cluster data manually when using a tour: we run a tour animation, stop when we see a subset of points that are different from the rest of the distribution, brush them, and then continue. Different projections will enable the separation of different subsets, and for well-separated clusters we will be able to recover full cluster solutions in this manner.

A similar approach can be used to refine a partitioning solution. In a *visual analytics* approach we use interactive visualizations to integrate human judgment with statistical and machine learning models (Keim, Kohlhammer, Ellis, and Mansmann 2010) to optimize knowledge extraction from data. Here this is in particular useful to integrate prior knowledge or business interests in a given cluster solution. In the interface we can keep the provided clustering, but separate out new subsets via manual selection, for example after we found a subset of particular interest via a manual tour.

3.3. Reproducibility

Ensuring the reproducibility of data analysis is a fundamental principle in scientific research (see, for example, other contributions in this special issue provided in Peng 2025; Gentleman, Rossini, and Carey 2025). It allows others to verify the validity of the findings and is key to

the integrity of the scientific process. Reproducibility not only builds trust in the research outcomes but also enables the scientific community to build upon existing work. When analyses can be replicated, it can be validated whether the conclusions drawn from the data are robust and not dependent on the specific conditions or idiosyncrasies of the original analyst. Moreover, reproducible research can serve as a foundational building block for subsequent studies, fostering incremental advancements in knowledge. Note that here the focus is on the reproducibility of the interactive data analysis. Reproducibility of the cluster solution can be evaluated using a bootstrap approach (Dolnicar and Leisch 2010).

One challenge in the context of interactive data analysis is that not all steps of the analysis are precisely documented in the form of code, especially when using GUIs where user-driven interactions might not leave a traceable history. This lack of documentation can hinder the ability of others to reproduce the analysis or to understand how specific results were obtained. To mitigate this challenge, it is essential to implement mechanisms that allow users to easily save and share intermediate snapshots of their analyses.

One measure to combat this is to make saving intermediate snapshots of the analysis easy and accessible. Specifically, the `Save projections and subsets` button enables users to take snapshots of their analysis, including visual representations, selected data, and parameter settings. Upon pressing this button, the user can first select, which of the following files they want to save:

- a `.png` file containing the currently displayed graphics;
- `.csv` files that capture the feature and subset selection as well as projections of the tours displayed at the time of the snapshot;
- two `.pkl` files that contain state features of the GUI, allowing for complete recovery of the snapshot.

Then a file browser is triggered, allowing users to specify the destination for saving their snapshot. The saved files provide dual utility.

First, they allow users to fully recover the state of the analysis within the GUI (which requires saving the `.csv` and `.pkl` files). This can be achieved either by using the `Load projections and subsets` button, or by launching a new GUI instance with the `load_interactive_tour()` function. The latter approach, using `load_interactive_tour()`, has the added flexibility of only requiring the original dataset and the directory containing the saved files. This function also allows users to modify display settings, such as adjusting the size of the interactive plots or changing the arrangement of the display grid. In contrast, when loading the saved state directly from within the GUI, it is crucial that the active session was initiated with the same dataset and plot objects that were present at the time of saving. This ensures that the analysis environment is accurately replicated.

Second, the saved `.csv` files provide a way to inspect and further analyze the data outside of the original interface. This opens up opportunities for deeper analysis and extensions of the work.

This level of interactivity and documentation is crucial for reproducibility, as it ensures that even exploratory, interactive data analysis can be retraced and validated by others. Ultimately, these features facilitate a reproducible workflow that balances the flexibility of interactive exploration with the rigor of reproducible research.

4. Applications

In this section, the Austrian Vacation Activities (Dolnicar and Leisch 2003), Australian Vacation Activities (Cliff 2009) and Tourist Risk Taking (Dolnicar 2017) datasets are analyzed

with **lionfish** to illustrate the software's capabilities. It has to be emphasized that these analyses only serve as illustrative case studies and that the application of **lionfish** is not limited to market segmentation. In each example a k-means clustering is used for illustration, primarily because this is what was used in the cited work introducing the data. The examples are either binary or ordinal data which can be considered to be numeric for clustering. Similarly, the choice of number of clusters follows the cited work, and from our observations seems to provide a reasonable partition of each dataset.

4.1. Austrian vacation activities dataset

The Austrian Vacation Activities dataset comprises responses from 2,961 adult tourists who spent their holiday in Austria during the 1997/98 season. Participants were asked to evaluate the importance of 27 different activities during their vacation. The survey categorized responses based on four levels of importance: "totally important", "mostly important", "a bit important", and "not important". The original authors binarized the responses for their analysis: a value of 1 was assigned if the activity was rated as "totally important", and a value of 0 if any of the other categories were selected. The survey was conducted by the Austrian Society for Applied Research in Tourism (ASART) for the Austrian National Tourism Organization (Österreich Werbung).

When working with projections involving binary data, distinct groupings emerge when two features dominate the projection. Figure 5 illustrates this with a scatterplot of alpine skiing and cross-country skiing (Figure 5A), alongside three projections from a grand tour of the data (Figures 5B–D). In these projections, the influence of alpine skiing and cross-country skiing decreases progressively from B to D. When only two binary features are plotted (Figure 5A), only four distinct points are visible. As additional features enter the projections, the data becomes more spread and appears continuous in some projections with many features contributing (Figure 5D).

To gain further insight into the dataset a k-means clustering following the approach described in Dolnicar *et al.* (2018) has been performed. Therefore, the function `stepcclust()` of the R package **flexclust** (Leisch 2006) with $k = 6$ and $nrep = 20$ was used.

Feature selection

The original dataset contains $p = 27$ features. Some of these features are more informative than others, so it may be beneficial to drop some of them. Although this may result in a loss of information, the projections can become difficult to interpret if too many features are plotted. Also, interacting with the projection axes in the **lionfish** GUI can become cumbersome when handling more than ~15 features at once. To counteract this, one can use the feature selection capabilities of **lionfish**, which allow for quick on-the-fly removal and addition of features. An effective and intuitive way to perform feature selection is to use the heatmap display within **lionfish**.

In Figure 6A, where colors show normalized column counts, we can observe the general interests of tourists within each cluster. Some activities are high on almost all clusters (e.g., relaxing, shopping), and some are low on all clusters (e.g., ski touring and horseback riding), and ignoring these can be helpful when assessing the distribution of activities between clusters. When comparing clusters 5 and 6 we can see that alpine skiing is high in both, but cluster 5 tourists also like going to the pool or sauna, but cluster 6 tourists prefer going for walks.

In Figure 6B, we can determine whether tourists selecting a particular feature are equally distributed or if they primarily fall within one or a few clusters. For example, nearly all tourists who visited museums are in cluster 2, and those who used health facilities are primarily attributed to cluster 4. Some activities, such as relaxing, are popular across all clusters.

These heatmaps can help with selecting features to focus on using the tour. Unpopular and universally popular activities can be removed. After performing the feature selection

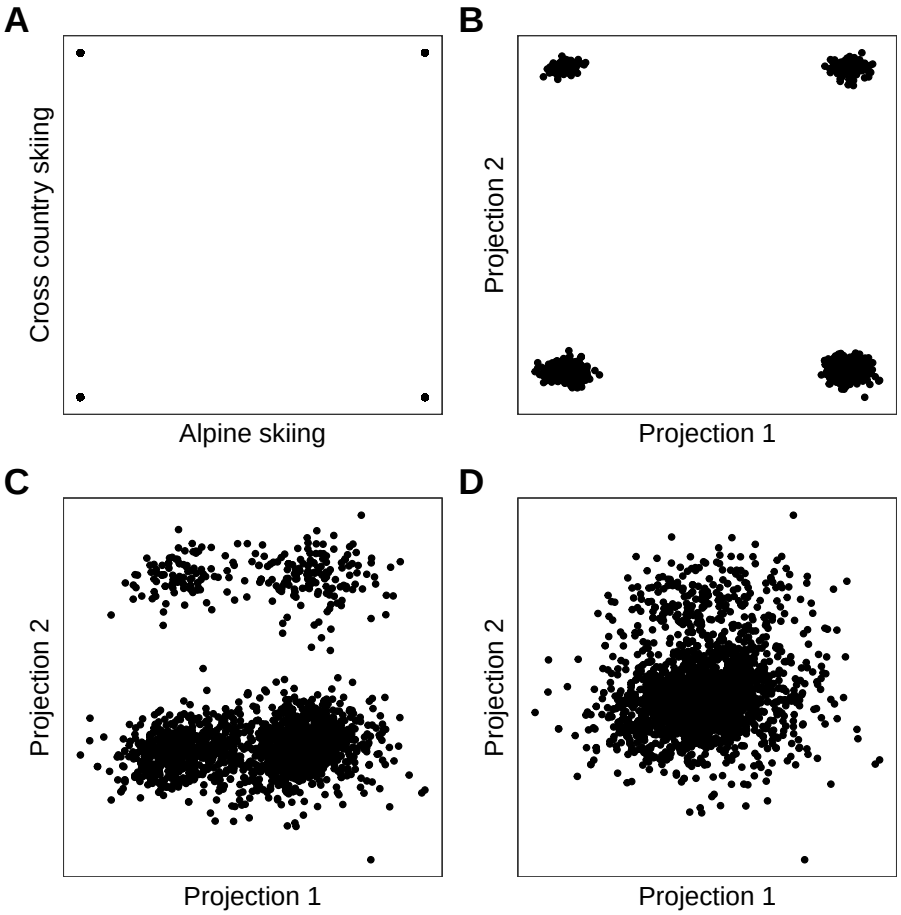

Figure 5: Scatterplot of the features alpine skiing and cross-country skiing (A) and three 2D projections from a grand tour on the Austrian Vacation Activities dataset (B–D). Because the 27 features are binary, one discerns distinct groupings when few features dominate the projection (A–B) but with many binary features the data mostly looks continuous when the loadings of the other features increase (C–D).

by unchecking the corresponding checkboxes in the GUI using this strategy, the following 12 activities remained: alpine skiing, going to a spa, using health facilities, hiking, going for walks, excursions, going out in the evening, going to discos/bars, shopping, sightseeing, museums, and pool/sauna.

We can now repeat the k-means clustering with `stepcclust()` on the reduced dataset. To evaluate the similarity between the two cluster solutions, we can use the **e1071** package's `classAgreement()` function (Meyer, Dimitriadou, Hornik, Weingessel, and Leisch 2024). This function, among other metrics, calculates κ, the percentage of datapoints that are in the same cluster adjusted for chance. The resulting κ value of 0.56 suggests that while there is some overlap between the cluster solutions, they differ substantially. Silhouette plots of both cluster solutions can be seen in Figure 7. By comparing both silhouette plots, we can see that the cluster solution with the reduced dataset results in a clustering of higher quality. Thus, we will continue with the analysis on the reduced dataset with the corresponding cluster solution. We can also see in Figure 7B that cluster 3 is of comparatively low quality.

It is important to note that the silhouette scores were generally quite low, reflecting the lack of clearly separable clusters in the data. This is common in market segmentation analysis, where clusters are often not clearly separable. As a result, clustering and feature selection algorithms may converge to a local optimum, and even if the global optimum is found, it may not necessarily be useful. For instance, a feature selection algorithm might drop features to find a better optimum according to an objective function. However, this can be problematic

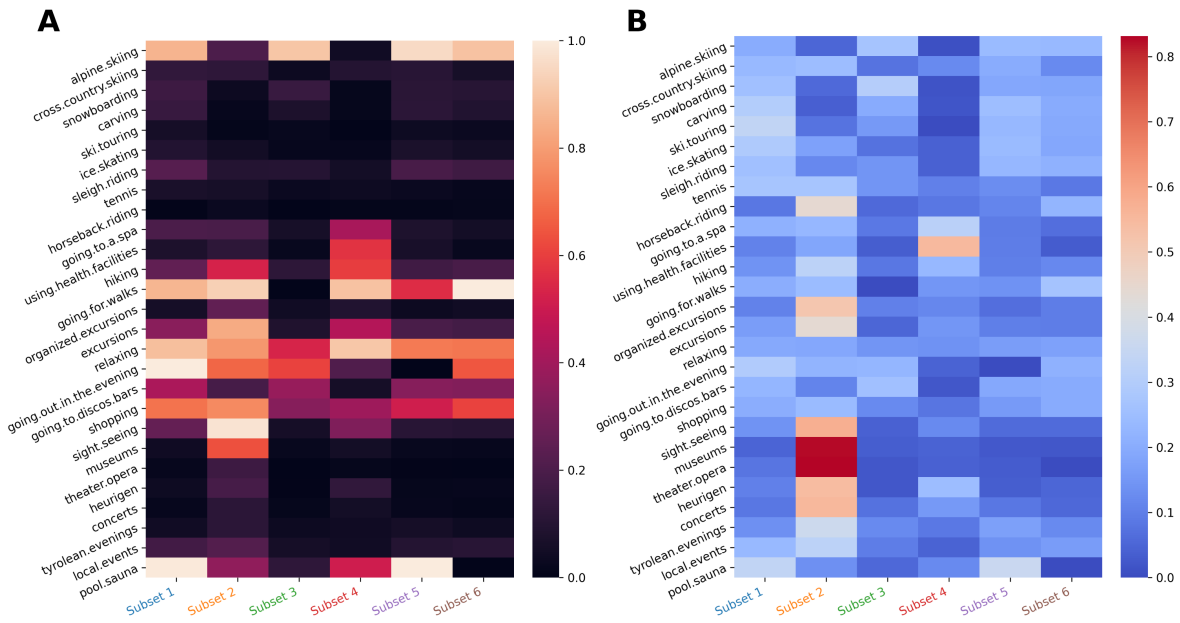

Figure 6: Traditional overview of clusters. Color represents (A) the intra-cluster fraction, and (B) the intra-feature fraction. From A, we can see that cluster 3 tourists like alpine skiing, going out in the evening and going to discos and bars. They also like relaxing, shopping and sightseeing but these are popular among all tourists. From B, we can see the distribution of activities across clusters, e.g., most tourists who use health facilities are found in cluster 4 while tourists going to a pool or sauna are primarily found in clusters 1 and 5.

if an analyst is interested in the influence of the dropped features. Consequently, a purely data-driven analysis can become counterproductive. The primary aim of this analysis is to gain insights into the data and understand the underlying patterns, rather than to identify an objectively optimal clustering configuration.

We can further explore the similarities and differences between the clusters by initializing an `interactive_tour()` with a 2D tour based on the LDA projection pursuit index. By navigating through the tour, we can observe various projections, and when a projection that separates the clusters is found, we highlight each cluster sequentially. The different highlighted clusters can be seen in Figure 8.

This process allows us to visually assess the separation and similarities between the clusters, providing insight into the structure of the dataset. By highlighting each cluster individually, we can evaluate their distinctiveness in different projections. The most influential features shown in Figure 8 are pool/sauna, alpine skiing, museums, going to the spa, going for walks and sightseeing. The projection roughly separates clusters 1 (blue), 2 (orange), and 3 (green) from each other and the other three clusters (red, violet and brown), which appear to be quite similar in the selected projection.

By manually manipulating the projection axes or initiating a local tour, we can gain further insight into the similarities between the different clusters. This interactive exploration allows for a more nuanced understanding of the relationships between clusters and the influence of key features on the separation of the data.

Redefining cluster assignments – learning more about museum goers

There are several reasons why we might want to manually modify a clustering solution. One is to capture observations that do not fit well within their assigned clusters. Another reason is to explore specific features in more detail. The advantage of manual cluster selection is that it preserves most of the original clustering structure, allowing us to adjust specific parts

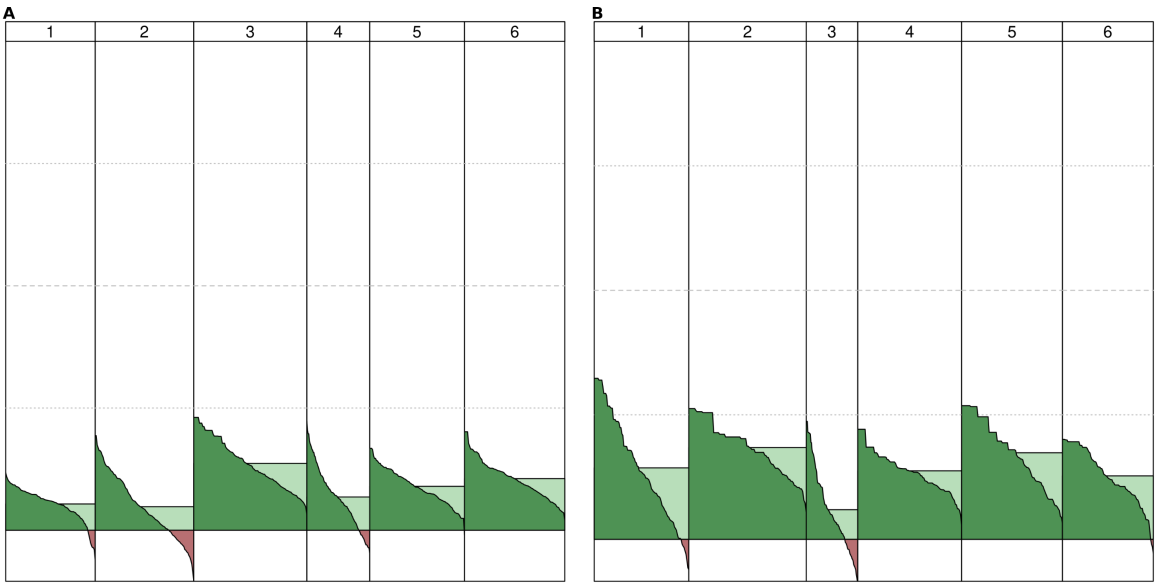

Figure 7: Comparison of silhouette plots of two k-means cluster solutions of the Austrian Vacation Activities dataset with $k = 6$. (A) shows the silhouette plot of the k-means solution of the full dataset and (B) the silhouette plot of the k-means solution of the dataset after manual feature selection. We can see that the cluster solution with the reduced dataset achieved better silhouette scores and that clusters 1 and 3 contain observations with negative silhouette scores.

of the solution without starting from scratch. This approach is particularly useful when we already have a cluster solution that reveals interesting patterns in the data.

In Figure 7B, we observed that clusters 1 and 3 of the reduced dataset contained datapoints that did not fit well into their respective clusters. To further investigate this, we can initialize an `interactive_tour()` with the following components:

- a 2D tour using the LDA projection pursuit index;
- a heatmap showing the intra-cluster fraction;
- a 1D tour with the LDA projection pursuit index; and
- a mosaic plot.

This setup produces the display shown in Figure 9. In the heatmap (top right), it can be seen that both clusters 1 and 3 contain tourists that did not go alpine skiing and that the main difference between them is that tourists in cluster 3 enjoyed going to the spa and health facilities as well as going to the pool, while the ones in cluster 1 did not. Other than that, the clusters were mostly similar. Now we might be interested in the clusters of tourists that enjoy going to museums. Therefore, we can adjust the projection axes so that these axes are elongated and point into different directions. We can see that there is indeed overlap between clusters 1 and 3, as shown in the 2D projection and heatmap in Figure 10. As a next step we can reassign the overlapping section to a new cluster – cluster 7 (pink). By selecting the checkbox for cluster 7 and manually selecting the region of overlap, we can form a new cluster, which is visualized in Figure 11.

In Figure 11, we can observe slight behavioral differences between tourists in clusters 1 (blue) and 7 (pink). Both the 2D projection and the heatmap indicate that, tourists in cluster 7 all enjoyed both museums and sightseeing, whereas most tourists in cluster 1 engaged in sightseeing but showed no interest in museums. Instead, we can take from the heatmap in Figure 11 that participants in cluster 1 exhibited a greater preference for hiking. Despite this,

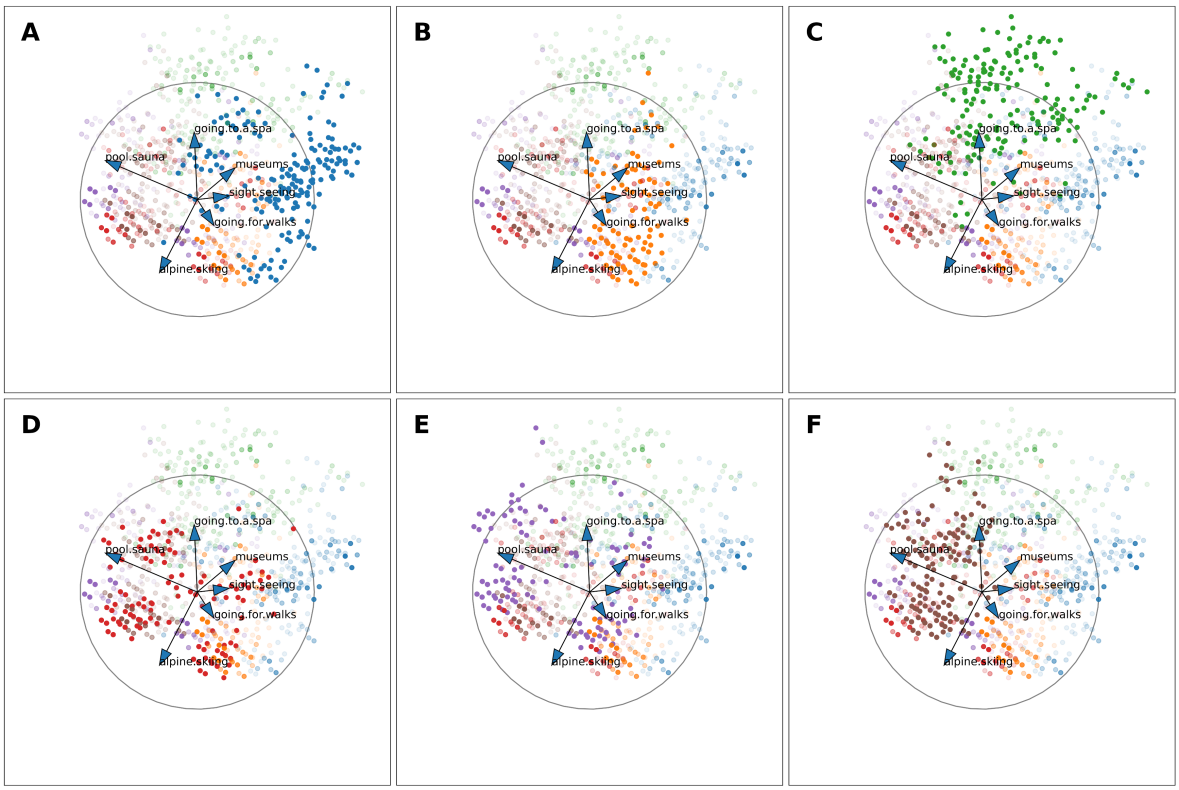

Figure 8: Display of a projection of the Austrian Vacation Activities dataset with the six clusters of the k-means cluster solution highlighted in different colors. Projection axes with a norm < 0.1 are not shown as these axes have little influence on the projection, but reduce clarity of the plot. The colors indicate which clusters the highlighted observations belong to with cluster 1 being blue (A), cluster 2 being orange (B), cluster 3 being green (C), cluster 4 being red (D), cluster 5 being violet (E) and cluster 6 being brown (F). Some datapoints appear to be highlighted always, which occurs due to the overlap of many datapoints in one spot. We can see that the projection separates some clusters well, however, also that there is considerable overlap of clusters 4, 5 and 6.

tourists in both clusters generally shared similar interests. This insight could be valuable for enhancing museum marketing strategies. While clusters 1 and 7 have overlapping interests, it appears that current marketing efforts may not effectively reach tourists in cluster 1. By increasing targeted marketing at hiking trails, popular excursion destinations, and shopping centers, it may be possible to attract more interest in museums from tourists in cluster 1.

4.2. Australian vacation activities dataset

The second dataset, the Australian Vacation Activities dataset, includes responses from 1,003 adult Australians who were surveyed through a permission-based internet panel. The survey was conducted in 2007. Participants were asked whether they engaged in 44 specific vacation activities during their most recent vacation within Australia. Similar to the Austrian Vacation Activities dataset, responses were binarized: a value of 1 indicates that the participant took part in the activity, while a value of 0 signifies they did not. Surveys where participants claimed they partook in more than 40 activities or no activity at all were removed as they are considered faulty.

Feature selection

At first, hierarchical clustering using the Ward2 algorithm (Murtagh and Legendre 2014) and the Jaccard index was applied to the features. The resulting dendrogram is shown in

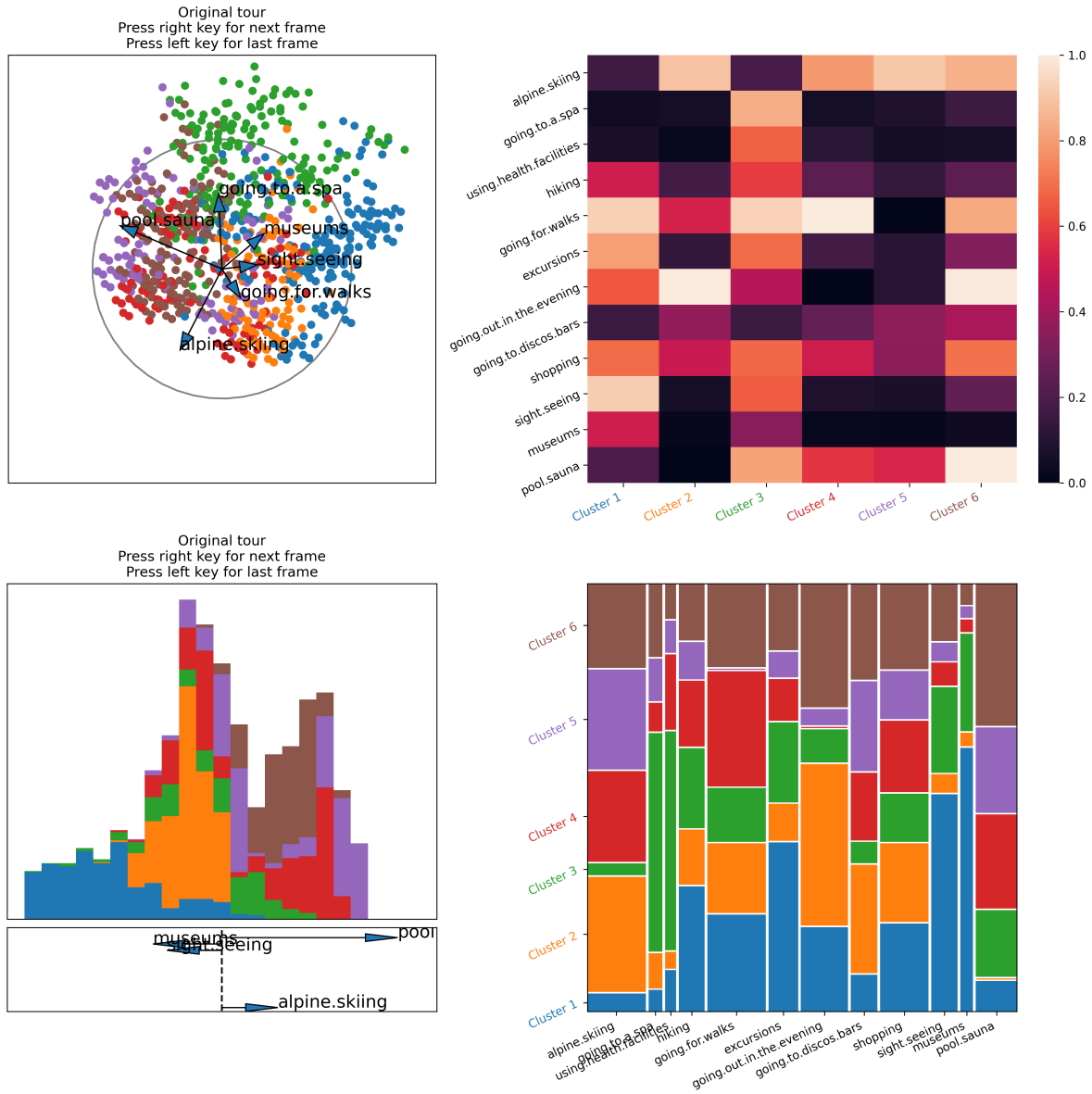

Figure 9: Interactive tour GUI loaded with multiple plots showing different aspects of the k-means solution of the Austrian Vacation Activities dataset. Projection axes with a norm < 0.1 are not shown as these axes have little influence on the projection, but reduce clarity of the plot. Top left: 2D tour with the linear discriminant analysis projection pursuit index. Top right: heatmap with the intra-cluster fraction. Bottom left: 1D tour with the linear discriminant analysis projection pursuit index. Bottom right: mosaic plot. Tourists in both clusters 1 and 3 did not participate in skiing a lot, but tourists in cluster 3 were much more interested in going to the pool, spa and health facilities compared to cluster 1.

Figure 12. Based on this clustering, $k = 15$ clusters were identified, and generally, only one representative feature from each cluster, which was considered informative, was selected for further analysis. Clusters containing unpopular activities, such as “Adventure”, which only had 42 participants, were discarded. The cluster containing popular features like “Beach”, “Swimming”, “ScenicWalks”, “Markets”, “Sightseeing”, “Friends”, “Pubs”, “BBQ”, “Shopping”, “Eating”, “EatingHigh”, “Movies”, and “Relaxing” was treated differently. Multiple features from this cluster were retained to preserve as much information as possible. After feature selection, the observations were clustered using k-means with $k = 6$.

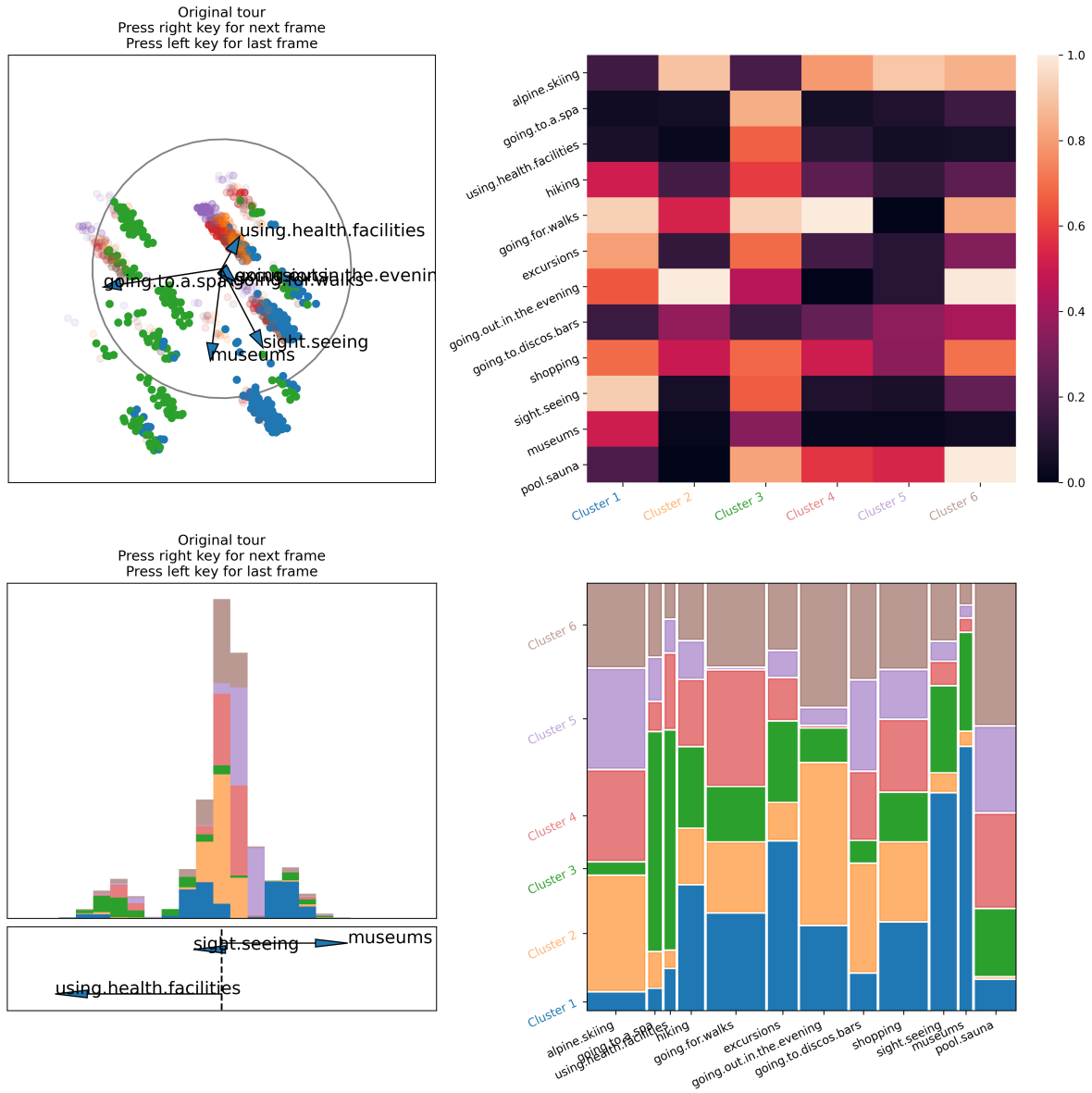

Figure 10: Interactive tour GUI loaded with multiple plots showing different aspects of the k-means solution of the Austrian Vacation Activities dataset, with manually adjusted projections. Projection axes with a norm < 0.1 are not shown as these axes have little influence on the projection, but reduce clarity of the plot. Top left: 2D tour with a manually adjusted projection. Top right: heatmap with the intra-cluster fraction. Bottom left: 1D tour with a manually adjusted projection. Bottom right: mosaic plot. Changing the projection axes of "going to a spa", "museums", "sightseeing" and "using health facilities" reveals the preferences and overlap of clusters 1 and 3.

Segmentation of tourists traveling without friends

In the heatmap displaying the intra-cluster fraction shown in Figure 13 (bottom right), we observe that clusters 1, 2, and 6 tend to prefer traveling without friends. We can assume tourists in these clusters prefer traveling alone, as couples or with their family. Additional features to further divide these clusters might be interesting for future surveys. As a tourist agency, we might be interested in targeting these travelers more effectively. However, clusters 2 and 6 also include individuals who enjoy spending time with their friends. To further explore the dataset, we can launch an `interactive_tour()` with the same configuration as in the previous example. To achieve better separation between the clusters, we can skip to the last frames of a 2D tour optimized for the LDA index.

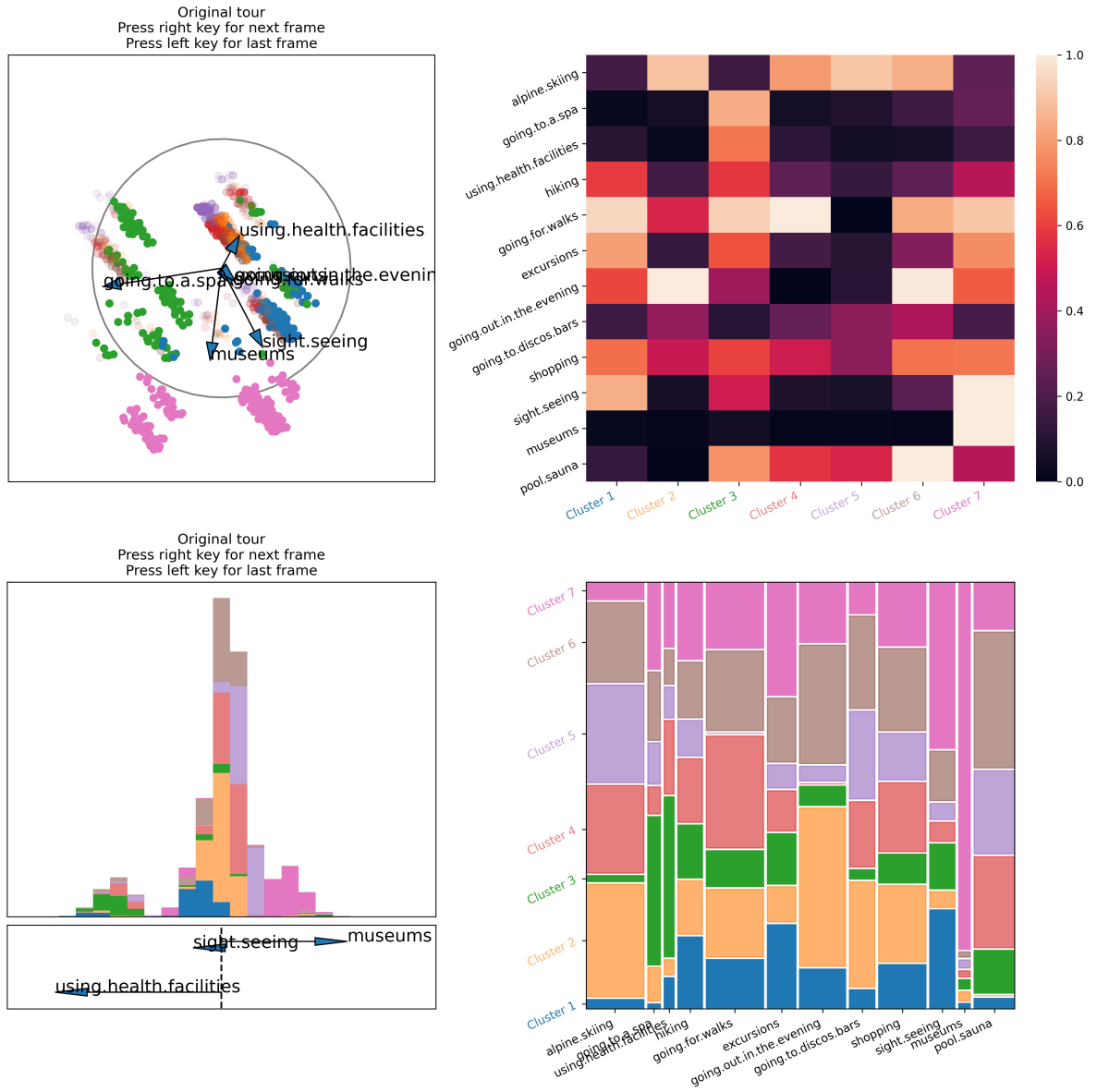

Figure 11: Interactive tour GUI loaded with multiple plots showing different aspects of the k-means solution of the Austrian Vacation Activities dataset, after new clusters have been selected manually. Projection axes with a norm < 0.1 are not shown as these axes have little influence on the projection, but reduce clarity of the plot. Top left: 2D tour with a manually adjusted projection. Top right: heatmap with the intra-cluster fraction. Bottom left: 1D tour with a manually adjusted projection. Bottom right: mosaic plot. We can see that now almost all museum-goers are in the new manually selected cluster 7 and what their preferences are compared to the other clusters.

Since we are particularly interested in tourists who did not spend time with friends, we can extend the projection axis "Friends" outward to separate the data based on that feature. Datapoints in the opposite direction of the "Friends" axis are the tourists we are interested in, which can be seen in Figure 13 on the left side of the top left plot. Subsequently, we can pull all other projection axes in one direction to separate datapoints based on their overall activity level. The resulting projection is shown in Figure 13 (top left). Tourists that fall in the direction of the axes generally engage in more activities compared to those in the opposite direction of the projection axes. This separation is evident, as observations in cluster 6 (brown) are located opposite the axes, and the heatmap (Figure 13, bottom right) shows that they did not engage in many activities. Similarly, we can also see that cluster 3 (green), which contains quite active tourists, is shifted towards the direction of the axes.

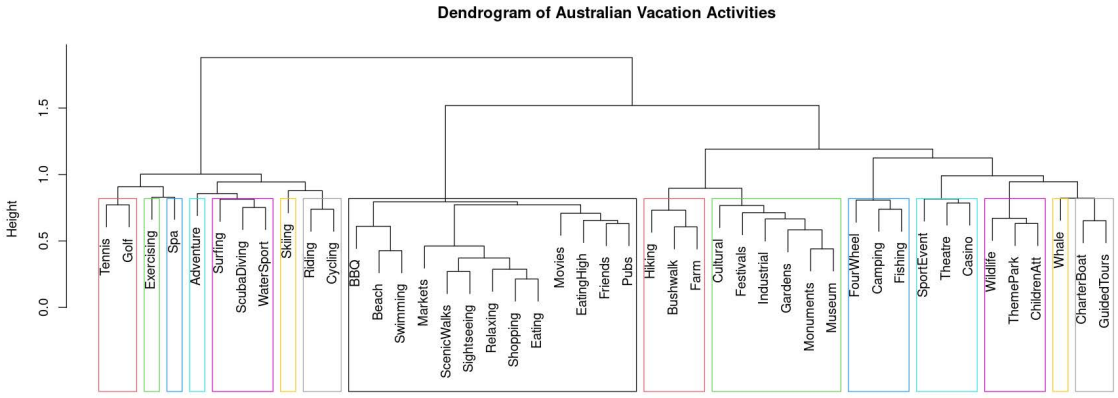

Figure 12: Dendrogram of the features of the Australian Vacation Activities dataset using the Ward2 algorithm with the Jaccard index. Features that were clustered together are marked by the colored boxes. We can see, which activities had similar patterns in tourist interest.

Using this logic, we can manually partition the datapoints on the left into three new clusters: active tourists (more upward – cluster 7), moderately active tourists (center – cluster 8), and largely inactive tourists (bottom – cluster 9). The result of this segmentation can be seen in Figure 14. Since all datapoints in cluster 1 (previously blue) had a value of 0 for "Friends" and were therefore included in the subset of datapoints subject to manual re-selection, no datapoints remain in cluster 1 after the re-selection. As a result, cluster 1 disappears. Analyzing the heatmap (Figure 14 bottom right), we can identify several interesting patterns.

By comparing cluster 3 (green), which contains active tourists who spent time with their friends, with cluster 7 (pink), the active tourists who traveled without their friends, we notice that cluster 7 showed less interest in visiting the casino, theater, and chartering a boat.

We also observe distinctive differences in the interests of the clusters that traveled without friends. According to the heatmap on the bottom right of Figure 14, cluster 7 was interested in relaxing, shopping, sightseeing, wildlife, going to the beach, visiting pubs, and exploring farms. Given that cluster 7 showed a notable interest in going to pubs, we can assume that the solo travelers and couples in this cluster are looking to meet new people. This insight could be leveraged in a marketing campaign by bundling activities that appeal to cluster 7, creating packages tailored to these tourists. Such packages would provide opportunities for them to connect with other travelers who share similar interests while enjoying their preferred activities. Additionally, some solo travelers and couples might prefer experiences or environments without children, allowing for the creation of adult-oriented packages. The remaining subset within cluster 7 consists of families. To better cater to this subsetting, packages can be refined for instance to consider that not all museums or festivals are equally suitable for children, enabling more family-friendly customizations.

Although cluster 8 (gray) was generally less active, almost everyone still engaged in sightseeing. We can see in the heatmap on the bottom right of Figure 14 that for them the focus was on sightseeing, relaxing, shopping, and going to the beach, with much less interest in other activities. It can be assumed that solo travelers in this subset value their time alone. This subset could potentially be targeted more effectively by offering sightseeing options with minimal interaction, such as using a phone app to provide information about interesting locations, rather than relying on a tour guide. For couples in this cluster, the focus may be on enjoying quality time together in a relaxed, low-key environment. Marketing strategies could emphasize romantic sightseeing experiences, such as sunset tours or private beach spots. Additionally, offering couples' packages that include spa treatments, leisurely dining experiences, and personalized itineraries could resonate well with this subset, allowing them

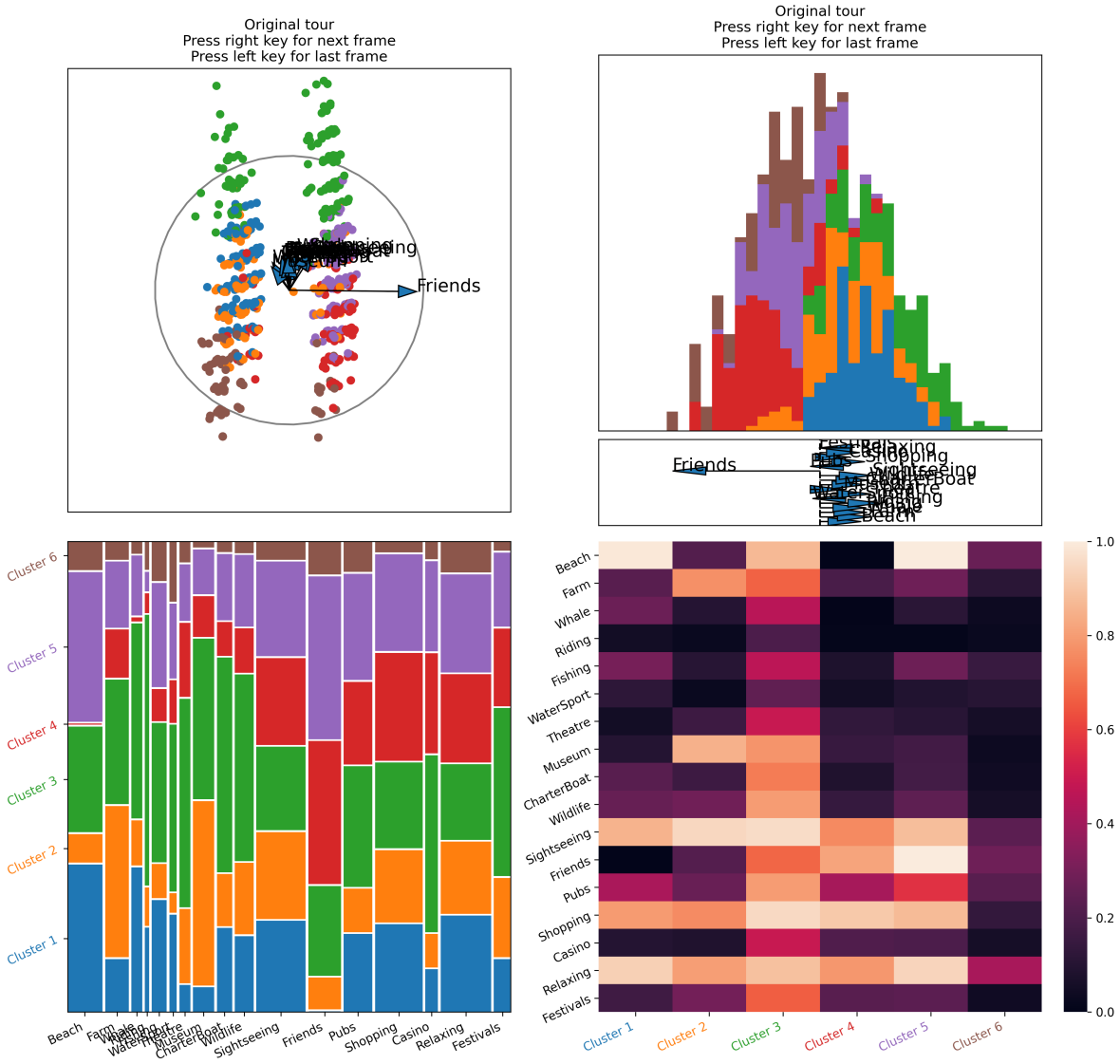

Figure 13: Interactive tour GUI loaded with multiple plots showing different aspects of the k-means solution of the Australian Vacation Activities dataset, with the projection axis of the feature "Friends" being relatively long and pointing into one direction and all the other ones being relatively short and pointing into another one. The projections (top left and right) primarily separate the data based on the feature "Friends" and then based on their general activity level. It is of note that only the relative orientation of the projection axes within the individual displays matters. Top left: 2D tour. Top right: 1D tour. Bottom left: Mosaic plot. Bottom right: Heatmap with the intra-cluster fraction.

to unwind and connect at their own pace. Families in cluster 8 might prioritize activities that balance relaxation and light exploration, particularly in child-friendly settings. Sightseeing tours tailored to families, with engaging and educational content for children, could be a strong fit. Beach outings that offer safe, family-oriented areas or interactive experiences like sandcastle-building competitions could also be appealing. By curating packages that cater to both relaxation and gentle family activities, families can enjoy their vacation with minimal stress.

Finally, since cluster 9 (olive green) was notably inactive, one might infer that they prefer spending much of their time in their accommodation. Consequently, these tourists might be most interested in accommodations that offer well-equipped, comfortable living spaces with

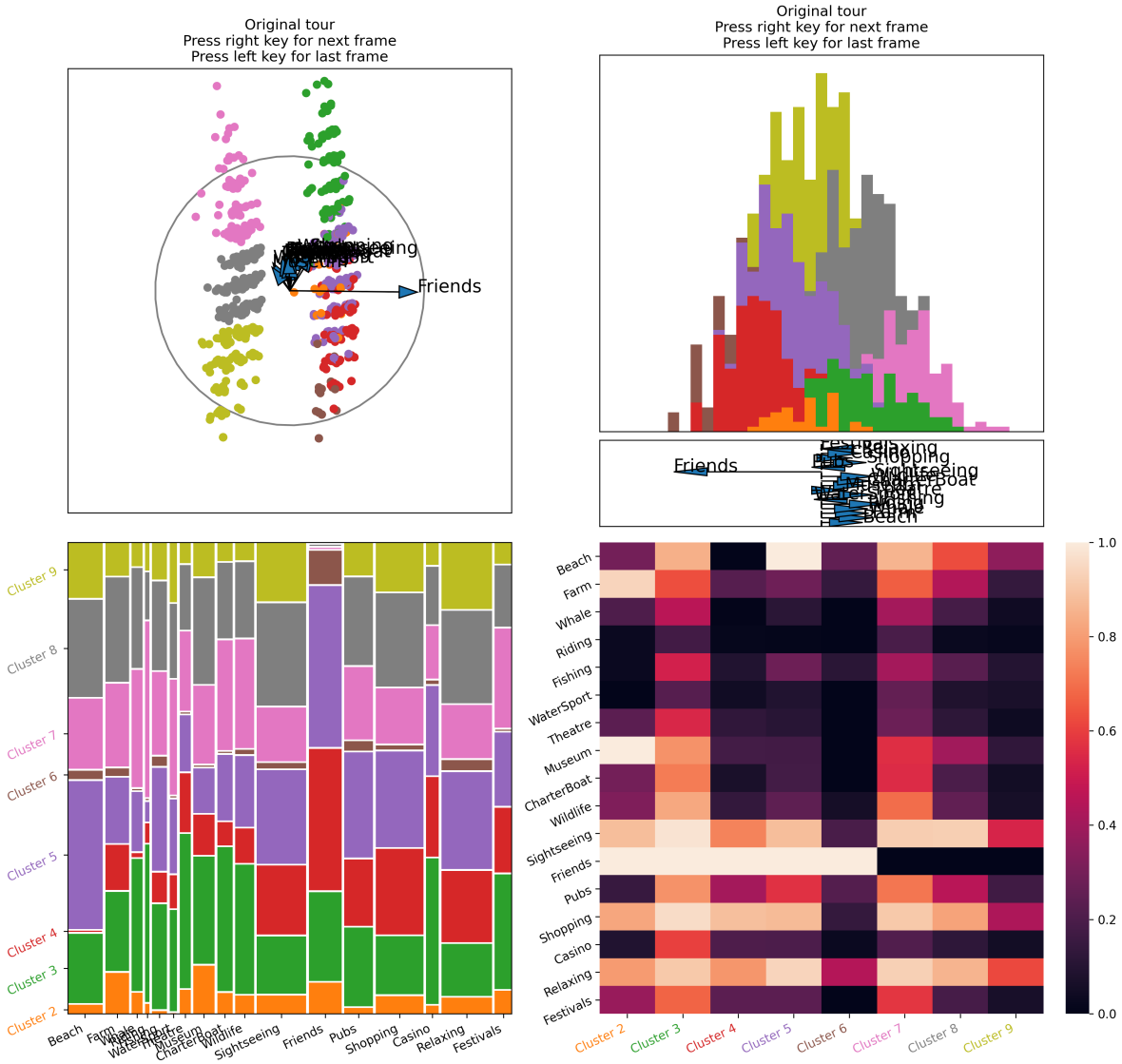

Figure 14: Interactive tour GUI as seen in Figure 13, but after sub-selection of clusters 7 (pink), 8 (gray) and 9 (olive green). Top left: 2D tour. Top right: 1D tour. Bottom left: Mosaic plot. Bottom right: Heatmap with the intra-cluster fraction. We can observe the preferences of three new subsets, which can be interpreted as very active (pink), moderately active (gray) and inactive (olive green) tourists traveling without their friends.

amenities that cater to relaxation and leisure.

4.3. Tourist risk taking dataset

The final dataset to be analyzed stems from a survey of 563 Australian residents who undertook a holiday of at least four nights in 2015. The respondents were asked about the types of risks they had taken in the past. Six different types of risk were screened: recreational (e.g., rock-climbing, scuba diving), health (e.g., smoking, poor diet, high alcohol consumption), career (e.g., quitting a job without another to go to), financial (e.g., gambling, risky investments), safety (e.g., speeding), and social risks (e.g., standing for election, publicly challenging a rule or decision). The response options for each risk type were on an ordinal scale ranging from 1 to 5: never (1), rarely (2), quite often (3), often (4), and very often (5).

To analyze the data, first a k-means clustering with $k = 5$ was performed. As the dataset comprises only six features, no feature selection was conducted. Subsequently, a 2D tour with

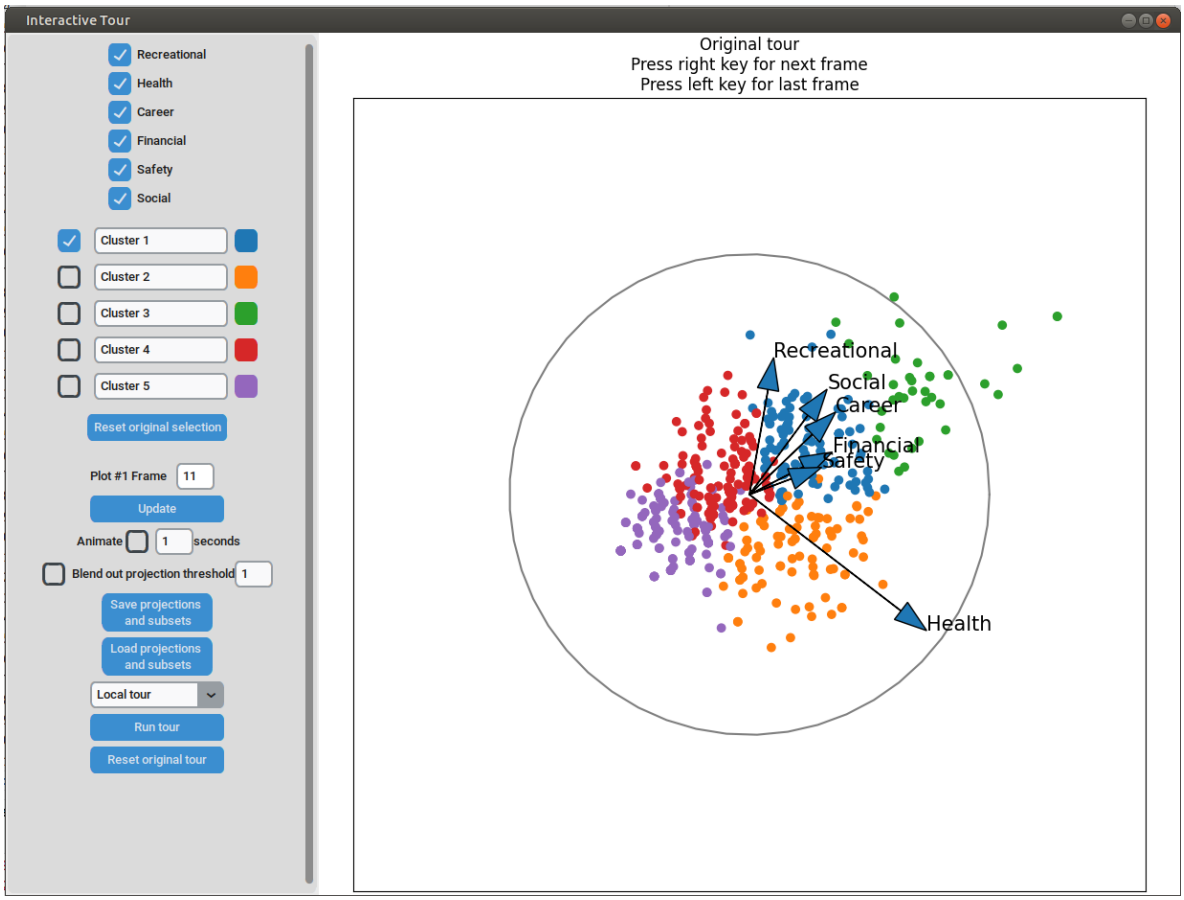

Figure 15: Final frame of a guided tour of the clustered risk data with the LDA index. The color of the datapoints indicates which cluster they belong to with cluster 1 being blue, cluster 2 being orange, cluster 3 being green, cluster 4 being red and cluster 5 being violet. All projection axes have positive values on the displayed x-axis, which can be interpreted as the projection primarily separating the clusters based on their general risk taking behavior. Most clusters are separated nicely, but clusters 4 and 5 overlap.

the LDA index was conducted, and the results were visualized using the **lionfish** package. The final projection of the tour can be seen in Figure 15. In this projection, all projection axes have positive values for the displayed x-axis, indicating that the projection primarily separates the data based on general risk taking behavior. Cluster 5 (violet) is identified as the most risk averse, while cluster 3 (green) is the most risk taking. Individuals in cluster 1 (blue) are also inclined to taking risk, but less so than individuals in cluster 3. Additionally, cluster 2 (orange) is oriented towards the health axis, suggesting that this cluster is more inclined to take health risks. Cluster 4 (red) is generally less risk taking and seems to be more inclined towards taking recreational risk. It can also be seen that there is some overlap between clusters 4 and 5.

Given the overlap between clusters 4 and 5 in Figure 15, we are interested in exploring their differences further. To achieve this, we generate a new tour with the LDA index from within the GUI, utilizing the inbuilt function to redefine the subsets considered in the LDA. This was done by selecting "Guided tour - LDA - regroup" from the dropdown menu and then clicking the "Run tour" button, which spawns an interface for the regrouping. By activating the switches for clusters 1, 2, and 3 in the "Ignore" column of the interface, the tour ignores the data from these clusters, focusing on separating clusters 4 and 5. This interface is shown in Figure 16. Alternatively, we could activate the switches for clusters 1, 2, and 3 not in the "Ignore" column, but in another column, e.g., the one labeled "New subgroup 1". This would result in the tour considering these clusters as one cluster. This configuration would

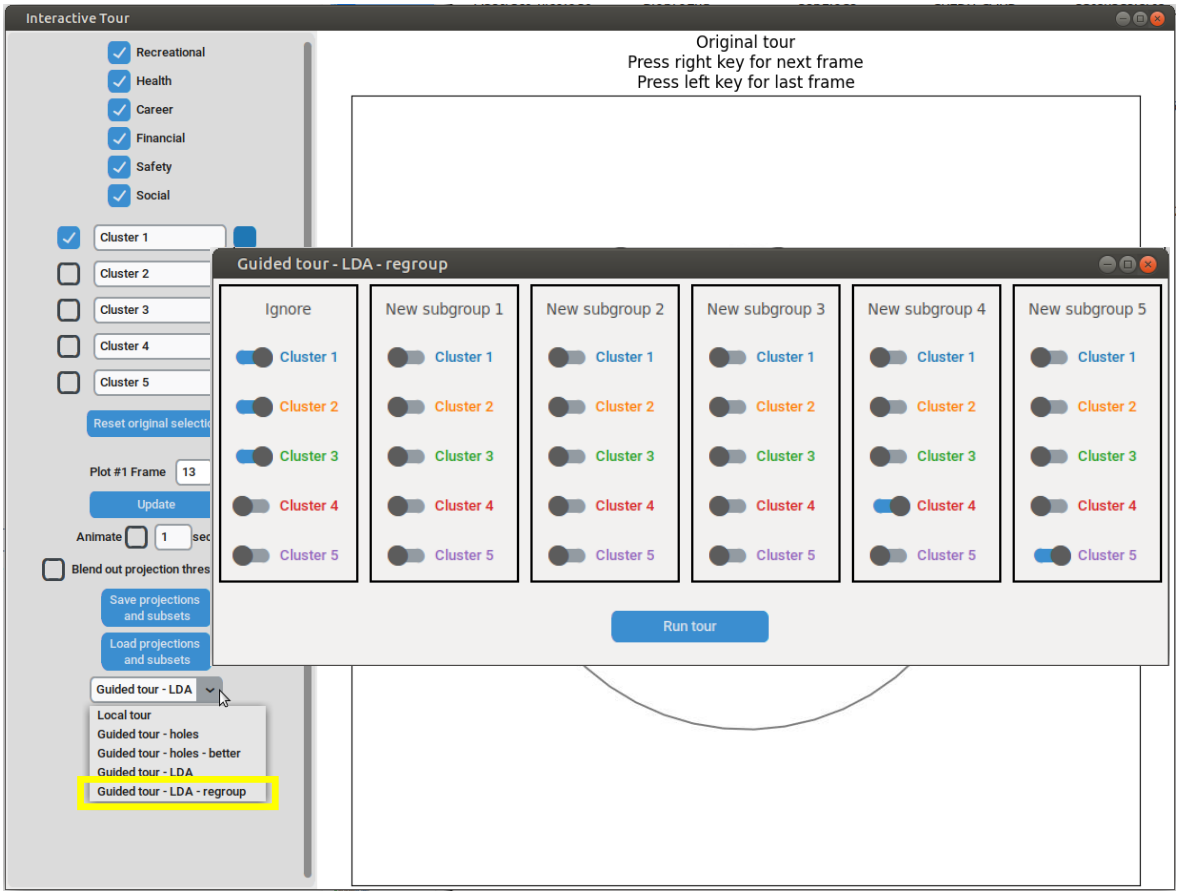

Figure 16: Interface of the "Guided Tour - LDA - regroup" function. The option highlighted by the yellow box indicates which element of the dropdown menu to select to spawn the displayed interface. Within the interface, the user can choose how they want the LDA metric to be computed by activating switches. All clusters selected in one column will be considered as a single cluster when computing the LDA metric. This affects only the calculation of the metric and the resulting projection matrices, not the actual selections. If a switch in the "Ignore" column is activated, the datapoints within that cluster will not be considered for the generation of the guided tour.

essentially try to separate clusters 4 and 5 from each other and all other datapoints. The final projection of the new tour can be seen in Figure 17. In this projection, it is evident that the most defining features for differentiating between the two clusters are recreational and safety risk, and social risk to a smaller extent. Career and financial risk are quite similar for both clusters.

5. Discussion

One might question the necessity of manual exploratory data analysis, considering that it is inherently subjective and relies heavily on intuition. However, in situations like those presented here, where there is no clearly defined or optimal clustering solution or feature selection, manual exploration becomes indispensable. While it is possible to optimize clustering metrics to improve separation between clusters, this alone may not yield conclusions that are useful for practical applications. The lack of clear boundaries and the overlapping nature of clusters in the datasets underscore the limitations of purely automated methods in capturing the complexity and nuance of real-world data.

In such cases, manual exploration allows analysts to interweave expert knowledge, intuition,

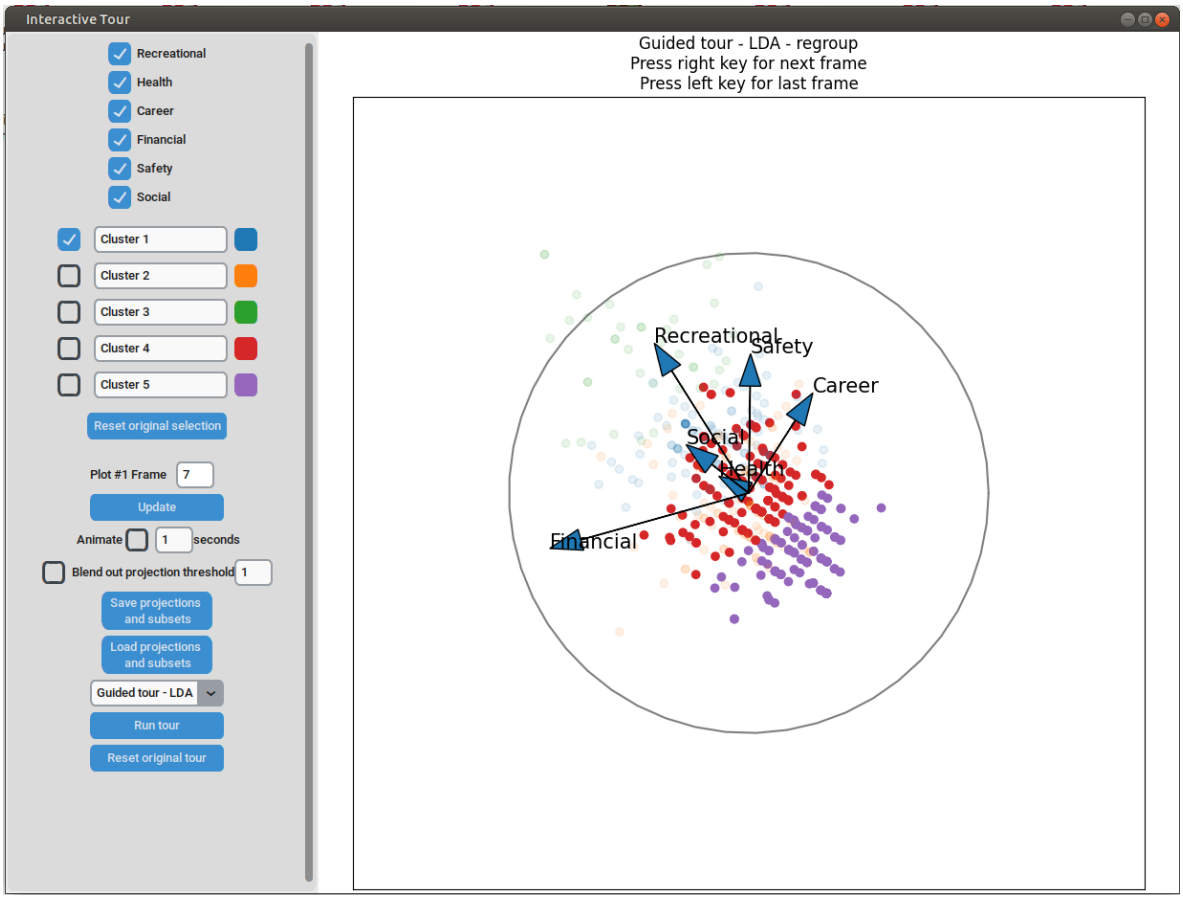

Figure 17: Final frame of the guided tour generated based on the settings shown in Figure 16. Clusters 4 (violet) and 5 (red) are now much better separated compared to the projection shown in Figure 15. The most influential features within the data for differentiating between the clusters were recreational risk and safety risk, with both being higher for cluster 4.

and specific objectives with the initial clustering solution, which serves as the backbone of the analysis. This approach is particularly valuable when no single solution can be deemed "correct". For instance, in the Austrian Vacation Activities dataset, leveraging the interactive GUI for feature selection enabled us to isolate the most informative activities and gain deeper insights into the preferences of tourists who might be interested in visiting museums. This understanding facilitated targeted recommendations for increasing museum attendance by focusing on tourists frequenting hiking trails, excursion spots, and shopping centers. Such insights are challenging to extract through automated optimization alone.

Similarly, in the Australian Vacation Activities dataset, manual exploration allowed us to refine the understanding of tourists traveling without friends by dividing them into three distinct subsets: highly active, moderately active, and largely inactive tourists. This nuanced segmentation, derived from a blend of automated clustering and manual adjustments, offers a deeper understanding of the varied needs and behaviors within this subset, enabling more effective marketing strategies.

The Tourist Risk Taking dataset was used to show how the interactive selection, together with the guided tour, can be used to fully understand the separation of all clusters in terms of the features. There is no single linear projection that can visualize the separation between all clusters, but we can use the interactivity of **lionfish** to understand the separation in two steps. First, we find a projection that can separate most clusters, and in a second step we select those clusters that were overlapping and ask for a projection that can best separate those subsets.

While some plots supported by the **lionfish** package are specifically designed for analyzing

binary survey data, it is important to emphasize that its capabilities extend far beyond this data type. The versatility of tours has been demonstrated across various applications in the past, showcasing their effectiveness in exploring complex, high-dimensional datasets. The interactive GUI enables seamless exploration of both 1D and 2D tours, regardless of the data being analyzed, providing a powerful tool for uncovering patterns and insights across diverse domains.

6. Conclusion

Ultimately, manual data exploration serves as a useful complement to automated methods, providing the flexibility to incorporate context, expert judgment, and specific analytical goals. This approach enables analysts to refine initial results and adapt them to the complexities of real-world scenarios, leading to more nuanced interpretations and actionable insights. The **lionfish** package offers an organized and responsive interactive tool for conducting such analyses, bridging the gap between automated clustering and exploration.

By integrating interactive visualization capabilities, the **lionfish** package empowers users to dynamically engage with their data, making it possible to uncover subtle patterns and relationships that might otherwise remain hidden. This is especially valuable in tackling complex datasets with the mindset that an automated solution needs to be validated. The package has broad applicability across various data types and analytical contexts where clustering is used. The flexibility of setting up the GUI elements from the command line allows it to be tailored for different applications.

Future developments in **lionfish** might include expanding the range of visualization methods and offering additional interactive features, such as enabling the generation of new cluster solutions directly from the GUI. An obvious expansion will be to add support for numeric data to the heatmap interface as described in Section 3. Leveraging the integration of both R and Python, future enhancements could include a seamless combination of algorithms from each language, which broadens the expansion potential of the software. For more complex data shapes, such as those with concavities or non-linear boundaries, the implementation of sliced tours (Laa *et al.* 2020) is recommended to improve exploratory analysis. Streamlining the addition of new plot types could also further enhance the versatility of the package, making it more adaptable for different data visualization needs. This ongoing integration of R and Python exemplifies how the strengths of both languages can be harnessed for the development of new software packages. A further expansion of this would be to develop a **shiny** app (Chang *et al.* 2024) leveraging the package's Python integration.

In summary, package **lionfish** represents a significant advancement in the toolkit of data analysts, offering a novel way to balance automated analysis with human intuition and domain expertise, thereby facilitating a deeper and more comprehensive understanding of complex datasets. We have demonstrated its capabilities by analyzing three different datasets from the domain of market segmentation.

Resources and supplementary materials

The source code and documentation of the software can be found on CRAN at `https://CRAN.R-project.org/package=lionfish` and on Github at `https://mmedl94.github.io/lionfish/`. The documentation features explanations and demonstrations of all implemented plots types as well as additional use cases not shown in this article. Supplementary material including:

- scripts to start the GUI and reproduce the graphics in the paper and

- saved states for different parts of the interactive analyses, so that the results can be reproduced,

can be found at https://github.com/mmedl94/lionfish_article/.

Acknowledgments

The development of the **lionfish** package has been supported by Google Summer of Code 2024.

References

Asimov D (1985). "The Grand Tour: A Tool for Viewing Multidimensional Data." *SIAM Journal of Scientific and Statistical Computing*, **6**(1), 128–143. doi:10.1137/0906011.

Babakhani N, Leisch F, Dolnicar S (2019). "A Good Graph is Worth a Thousand Numbers." *Annals of Tourism Research*, **76**, 338–342. doi:10.1016/j.annals.2018.10.007.

Chang W, Cheng J, Allaire JJ, Sievert C, Schloerke B, Xie Y, Allen J, McPherson J, Dipert A, Borges B (2024). ***shiny**: Web Application Framework for R*. R package version 1.10.0, URL https://shiny.posit.co/.

Clarke E, Sherrill-Mix S, Dawson C (2023). ***ggbeeswarm**: Categorical Scatter (Violin Point) Plots*. R package version 0.7.2, URL https://CRAN.R-project.org/package=ggbeeswarm.

Cliff K (2009). *A Formative Index of Segment Attractiveness: Optimising Segment Selection for Tourism Destinations*. Ph.D. thesis, University of Wollongong. URL https://hdl.handle.net/10779/uow.27663669.v1.

Cook D, Buja A (1997). "Manual Controls for High-Dimensional Data Projections." *Journal of Computational and Graphical Statistics*, **6**(4), 464–480. doi:10.2307/1390747.

Cook D, Buja A, Cabrera J, Hurley C (1995). "Grand Tour and Projection Pursuit." *Journal of Computational and Graphical Statistics*, **4**(3), 155–172. doi:10.2307/1390844.

Cook D, Swayne D (2007). *Interactive and Dynamic Graphics for Data Analysis: With R and **GGobi***, volume 364. Springer-Verlag, New York. doi:10.1007/978-0-387-71762-3.

Dolnicar S (2017). *Peer-to-Peer Accommodation Networks: Pushing the Boundaries*. Goodfellow Publishers. doi:10.23912/9781911396512-3454.

Dolnicar S, Grün B, Leisch F (2018). *Market Segmentation Analysis: Understanding It, Doing It, and Making It Useful*. Springer-Verlag, Singapore. doi:10.1007/978-981-10-8818-6.

Dolnicar S, Leisch F (2003). "Winter Tourist Segments in Austria: Identifying Stable Vacation Styles Using Bagged Clustering Techniques." *Journal of Travel Research*, **41**(3), 281–292. doi:10.1177/0047287502239037.

Dolnicar S, Leisch F (2010). "Evaluation of Structure and Reproducibility of Cluster Solutions Using the Bootstrap." *Marketing Letters*, **21**(1), 83–101. doi:10.1007/s11002-009-9083-4.

Gentleman R, Rossini A, Carey V (2025). "Contributions of Fritz Leisch to Vignettes and Reproducible Research." *Austrian Journal of Statistics*. doi:10.17713/ajs.v54i3.2057.

Hart C, Wang E (2023). "Taking the Scenic Route: Interactive and Performant Tour Animations." *The R Journal*, **15**(2), 307–329. doi:10.32614/rj-2023-052.

Hennig C (2004). "Asymmetric Linear Dimension Reduction for Classification." *Journal of Computational and Graphical Statistics*, **13**(4), 930–945. doi:10.1198/106186004x12740.

Hunter JD (2007). "**matplotlib**: A 2D Graphics Environment." *Computing in Science & Engineering*, **9**(3), 90–95. doi:10.1109/mcse.2007.55.

Keim D, Kohlhammer J, Ellis G, Mansmann F (2010). *Mastering the Information Age: Solving Problems with Visual Analytics*. Eurographics Association. doi:10.2312/14803.

Laa U, Aumann A, Cook D, Valencia G (2023). "New and Simplified Manual Controls for Projection and Slice Tours, with Application to Exploring Classification Boundaries in High Dimensions." *Journal of Computational and Graphical Statistics*, **32**(3), 1229–1236. doi:10.1080/10618600.2023.2206459.

Laa U, Cook D, Valencia G (2020). "A Slice Tour for Finding Hollowness in High-Dimensional Data." *Journal of Computational and Graphical Statistics*, **29**(3), 681–687. doi:10.1080/10618600.2020.1777140.

Lee EK, Cook D, Klinke S, Lumley T (2005). "Projection Pursuit for Exploratory Supervised Classification." *Journal of Computational and Graphical Statistics*, **14**(4), 831–846. doi:10.1198/106186005x77702.

Lee S, Cook D, da Silva N, Laa U, Spyrison N, Wang E, Zhang S (2022). "The State-of-the-Art on Tours for Dynamic Visualization of High-Dimensional Data." *WIREs Computational Statistics*, **14**(4), e1573. doi:10.1002/wics.1573.

Leisch F (2006). "A Toolbox for k-Centroids Cluster Analysis." *Computational Statistics & Data Analysis*, **51**(2), 526–544. doi:10.1016/j.csda.2005.10.006.

Leisch F (2008). "Visualizing Cluster Analysis and Finite Mixture Models." In C houh Chen, W Härdle, A Unwin (eds.), *Handbook of Data Visualization*, pp. 561–587. Springer Berlin Heidelberg, Berlin, Heidelberg. doi:10.1007/978-3-540-33037-0_22.

Leisch F (2010). "Neighborhood Graphs, Stripes and Shadow Plots for Cluster Visualization." *Statistics and Computing*, **20**(4), 457–469. doi:10.1007/s11222-009-9137-8.

Lundh F (1999). "An Introduction to **Tkinter**." URL https://ftp.math.utah.edu/u/ma/hohn/linux/tcl/an-introduction-to-tkinter.pdf.

Meyer D, Dimitriadou E, Hornik K, Weingessel A, Leisch F (2024). ***e1071**: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*. R package version 1.7-16, URL https://CRAN.R-project.org/package=e1071.

Murtagh F, Legendre P (2014). "Ward's Hierarchical Agglomerative Clustering Method: Which Algorithms Implement Ward's Criterion?" *Journal of Classification*, **31**(3), 274–295. doi:10.1007/s00357-014-9161-z.

Peng RD (2025). "Tooling for Reproducible Research: Considerations for the Past and Future of Data Analysis." *Austrian Journal of Statistics*. doi:10.17713/ajs.v54i3.2052.

R Core Team (2024). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Schimansky T (2024). "**CustomTkinter**." URL https://github.com/TomSchimansky/CustomTkinter.

Ushey K, Allaire JJ, Tang Y (2024). ***reticulate**: Interface to 'Python'*. R package version 1.38.0, https://github.com/rstudio/reticulate, URL https://rstudio.github.io/reticulate/.

Van Rossum G, Drake FL (2009). *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA. ISBN 1441412697.

Wickham H, Cook D, Hofmann H, Buja A (2011). "**tourr**: An R Package for Exploring Multivariate Data With Projections." *Journal of Statistical Software*, **40**(2), 1–18. doi:10.18637/jss.v040.i02.

Affiliation:

Matthias Medl
Institute of Statistics
BOKU University Vienna
E-mail: matthias.medl@boku.ac.at

Austrian Journal of Statistics
published by the Austrian Society of Statistics
Volume 54
2025

http://www.ajs.or.at/
http://www.osg.or.at/

Submitted: 2024-10-04
Accepted: 2025-03-31