Requirements Evidence Document

The document was written by Callum Hayden, Mark Schmieg and Ridwan Mukhtar.

This document shows evidence of the requirements achieved in this project. It will comprise images, explanations, and in some cases, code snippets. Should this document prove to be insufficient evidence, we make ourselves fully available to the marker to demo the application and discuss any insufficiency, however, during this time due to social distancing this may only be possible through the use of a skype call.

Assumptions made:

This is a list of assumptions we made based on ambiguity of the requirement.

- We assume the hierarchical layout should be made after steps one to four are fully complete and the agglomerative table is empty.
- We assume that by highlighting multiple points of associated data in a chart, we also highlight one, hence by completing requirement eight, seven is also completed.

All layouts were created by the group with the exception of the tree and agglomerative layouts where we used Professor Mike Chantler's implementation of them. This was done so we could learn about the D3 library and how to use it.

Gateway Requirements

G1

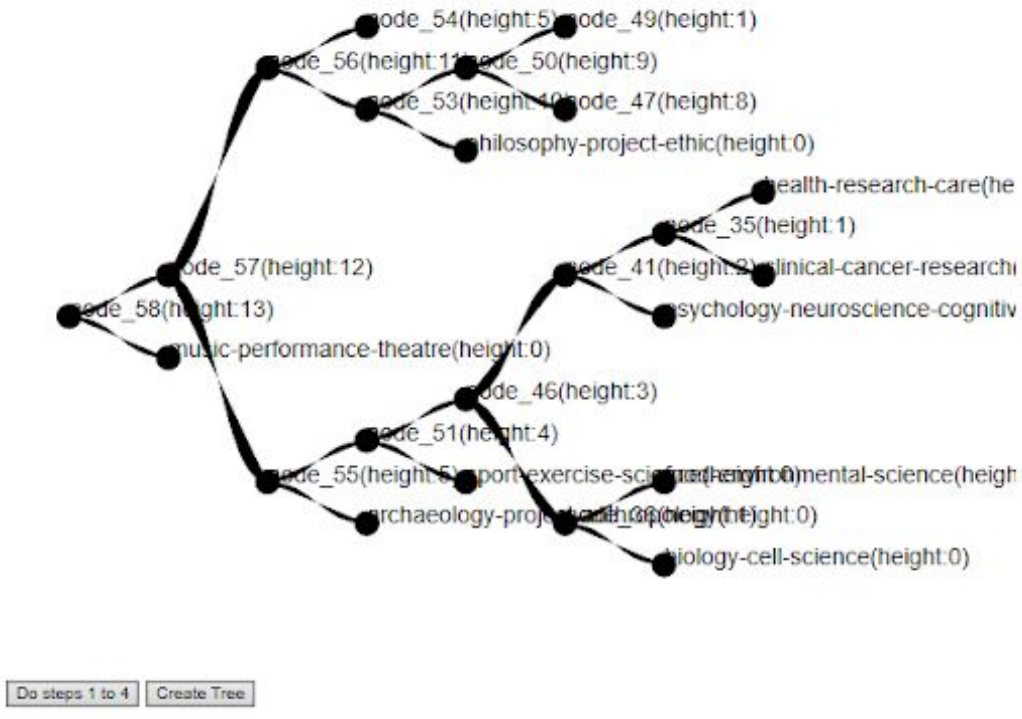
Evidence of use of GitLab by group showing incremental development through regular commits and commit messages.

Evidence of the use of GitLab can be found attached in commit_history.pdf.

G2

Use of D3 hierarchical display to illustrate agglomerative clustering of topic-to-topic similarity.
This interaction works by going through the entire process of the agglomerative clustering (doing step 1 to 4 till the very end) and clicking the create tree button to see the topic-topic similarity in a hierarchical format (as a tree).

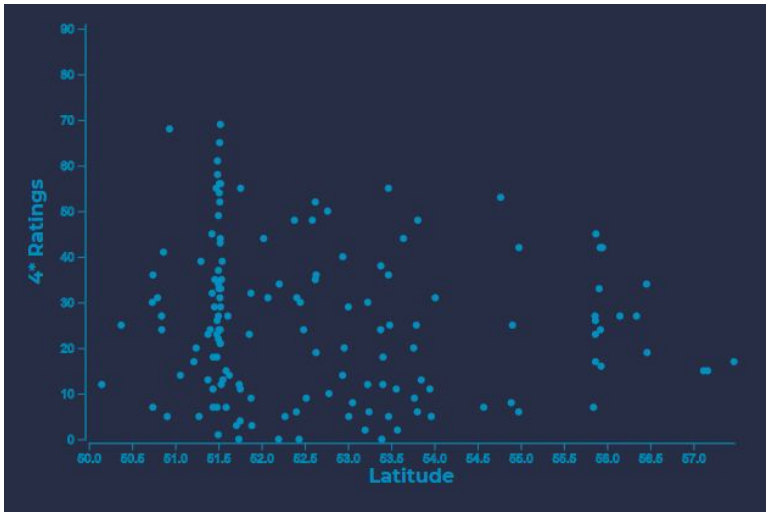
The figure below showcases the tree and the topic-to-topic similarity in a tree format once the entire stepping process is finished

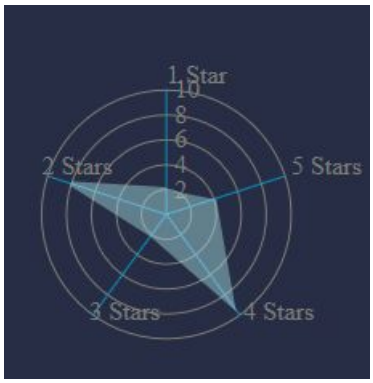


Project Requirements

R1

Use of three different D3 layouts in a single dashboard

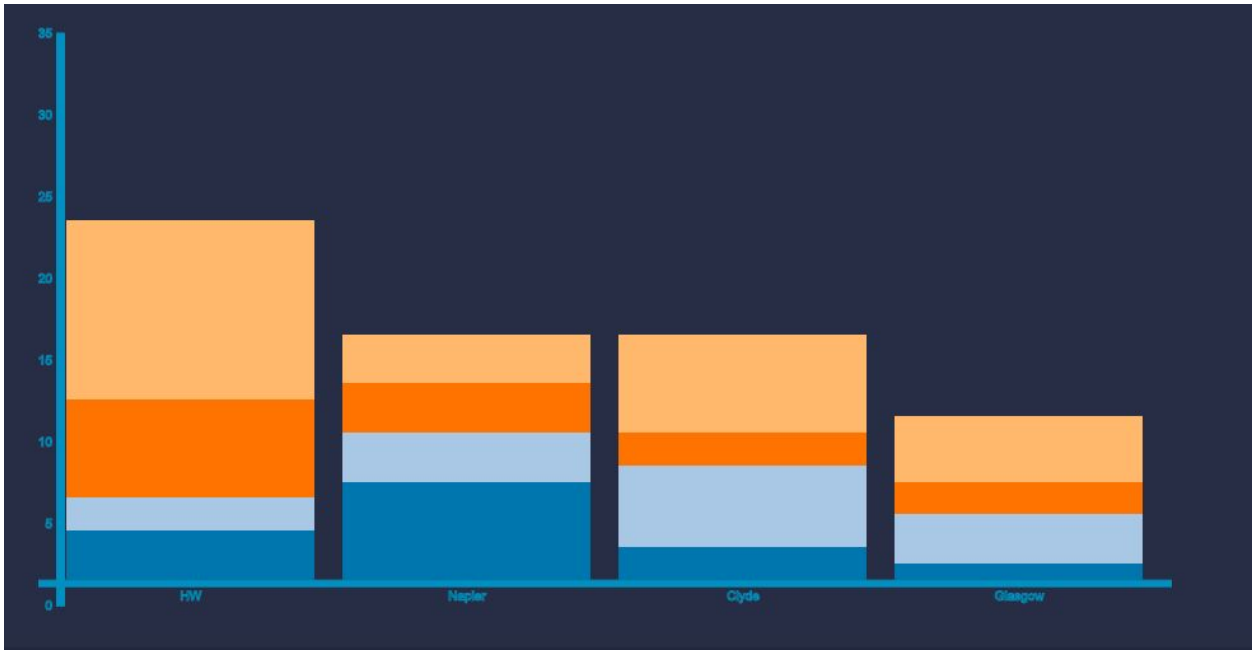


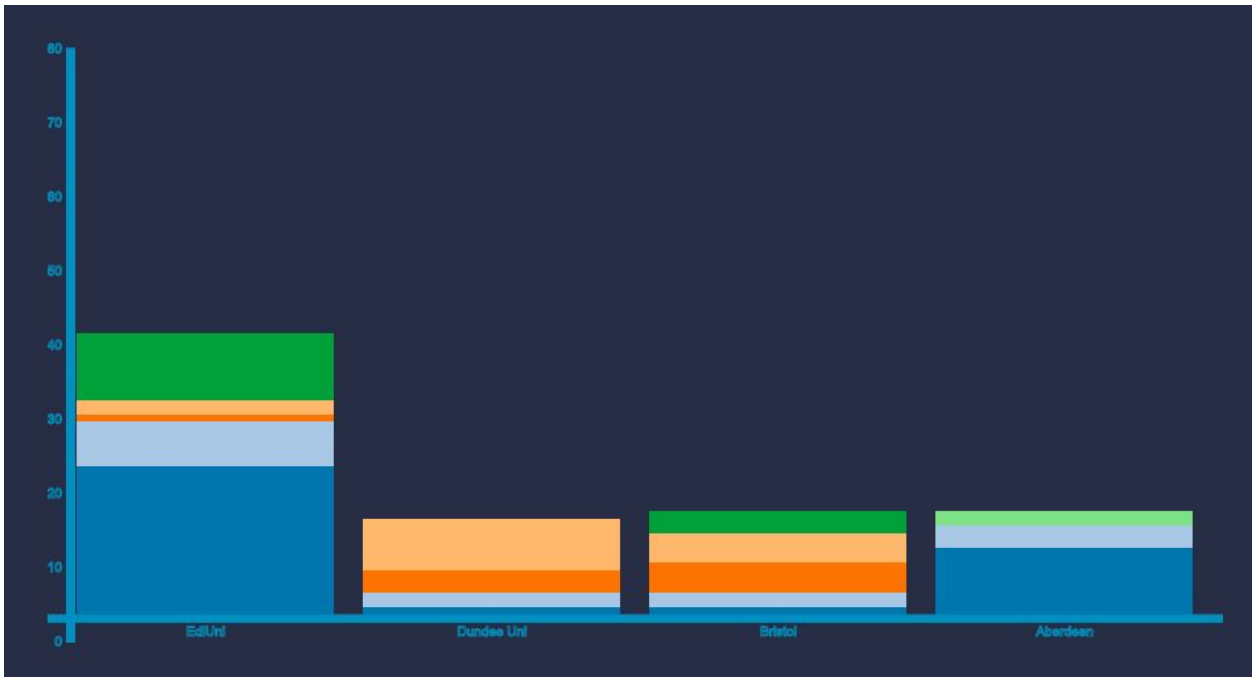


The requirement can be marked as complete due to the use of multiple layouts in the same dashboard, such as the scatter plot, the map and the radar chart. Nevertheless, more than 3 layouts are used and displayed in the project dashboard.

R2

Use of automatic scaling of all axes in a single layout during data update.

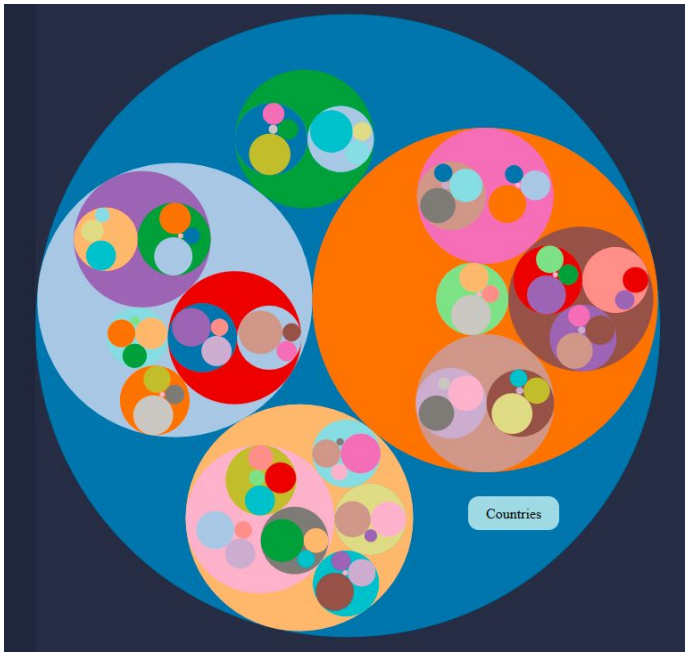




Depending on the height of the stacks in each configuration the y axis as well as the x axis update to match the data's boundaries.

R3

Use of datum highlighting in which hovering over a datum provides additional information via a tooltip.



In the project dashboard, when hovering over the sunburst or the pack layout a tooltip will appear close to where the mouse is located on the layout. The tooltip in both layouts displays the name of the circle hovered upon in the case of the pack layout and it displays the name of the section hovered over in the sunburst.

R4

Use of D3 transitions to highlight new data.

```
        var enterSelection = selection
            .enter().append("g")
            .attr("class", "layer")
            .style("fill", function (d, i) { return color(i); })

        enterSelection.selectAll("rect")
            .data(function (d) { return d; })
            .enter().append("rect")
            .transition()
            .duration(1000)
            .delay(1000)
            .attr("x", function (d) { return xScale(d.data.key); })
            .attr("y", function (d) { return yScale(d[1]); })
            .attr("height", function (d) { return yScale(d[0]) - yScale(d[1]); })
            .attr("width", xScale.bandwidth());
```

This requirement is completed by having an enterSelection part in the code. It first gives a color to the bars coming into the bar chart, then it uses a transition to ease the new data into the graph. A transition with new data coming into the stacked bar chart can be seen when using the drop down menu in the dashboard.

R5

Use of D3 transitions to highlight updating data.

```
var updateSelection = selection
    .attr("class", "layer")
    .style("fill", function (d, i) { return color(i); });

updateSelection.selectAll("rect")
    .data(function (d) { return d; })
    .transition()
    .duration(1000)
    .delay(1000)
        .attr("x", function (d) { return xScale(d.data.key); })
        .attr("y", function (d) { return yScale(d[1]); })
        .attr("height", function (d) { return yScale(d[0]) - yScale(d[1]); })
        .attr("width", xScale.bandwidth());
```

This requirement is satisfied by the stacked bar chart. As it is hard to show transitions in an image, we have provided code instead. This code is the update selection for the stacked bar chart. It shows the use of transitions during the update selection and is called when data is being updated. It applies a transition to help highlight the data that is being changed.

R6

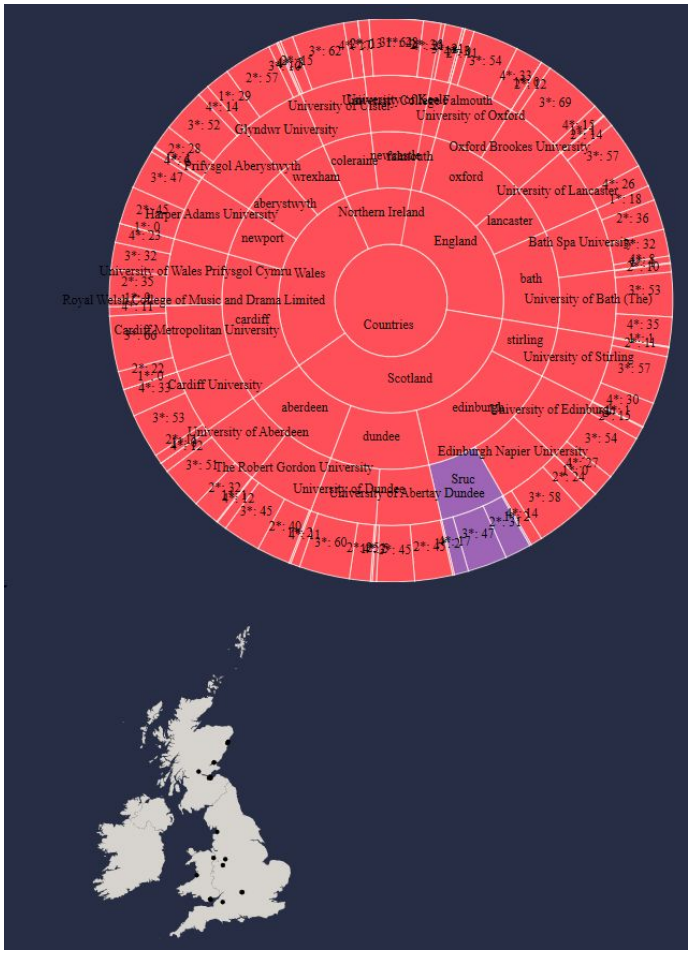
Use of D3 transitions to highlight exiting data.

```
selection.exit()
    .classed("exitSelection", true)
    .transition()
    .duration(500)
    .remove();
```

This requirement is satisfied by the stacked bar chart. As it is hard to show transitions in an image, we have provided code instead. This code is the exit selection for the stacked bar chart. It shows the use of transitions during the exit selection, which deals with the data being removed from (exiting) the chart.

R7

Use of cross-layout brushing in which mousing over a data point in one chart highlights associated data in another chart.



For this requirement an interaction was created between a map and a sunburst. Hovering over any point (university) on the map would highlight a section of the sunburst corresponding to the point on the map.

R8

Use of cross-layout brushing in which moussing over one data point in one chart highlights multiple associated data points in another chart.

By completing R7, we also completed R8, because hovering over one point on the map highlights a section of the sunburst which contains the name of the point of the map itself and its children. We assumed that highlighting multiple associated data points would automatically count as also highlighting one associated data.

R9

Significant use of data not provided by the course.

This requirement was completed by sourcing data such as the number of students enrolled at every given university. This was sourced from the HE student enrolments by HE provider and domicile 2017/18 (https://www.hesa.ac.uk/data-and-analysis/sb252/figure-6). This data can be found in:

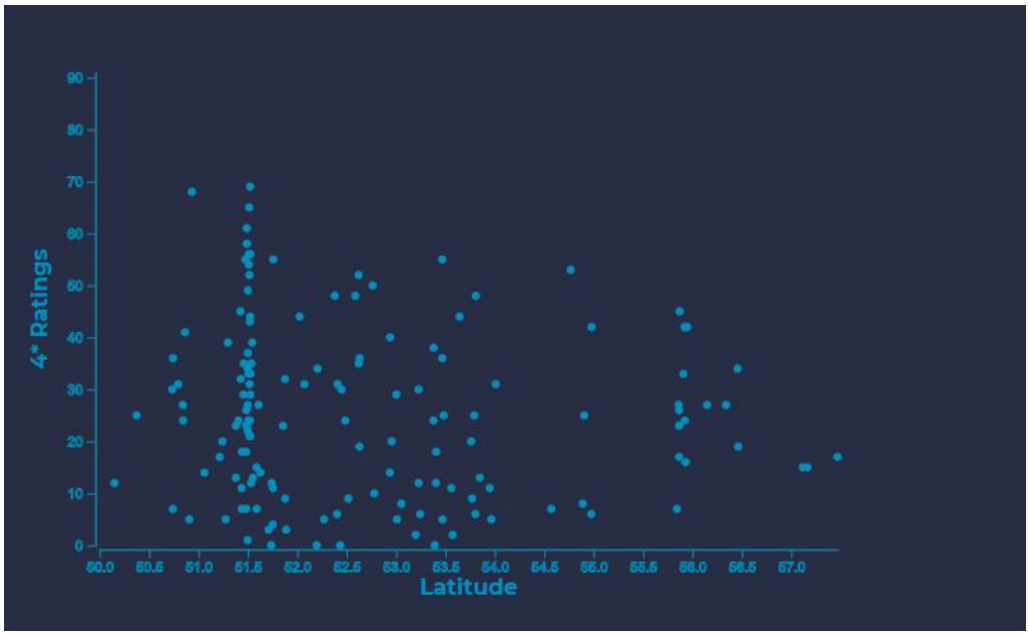
public/data/sb252-figure-6.csv

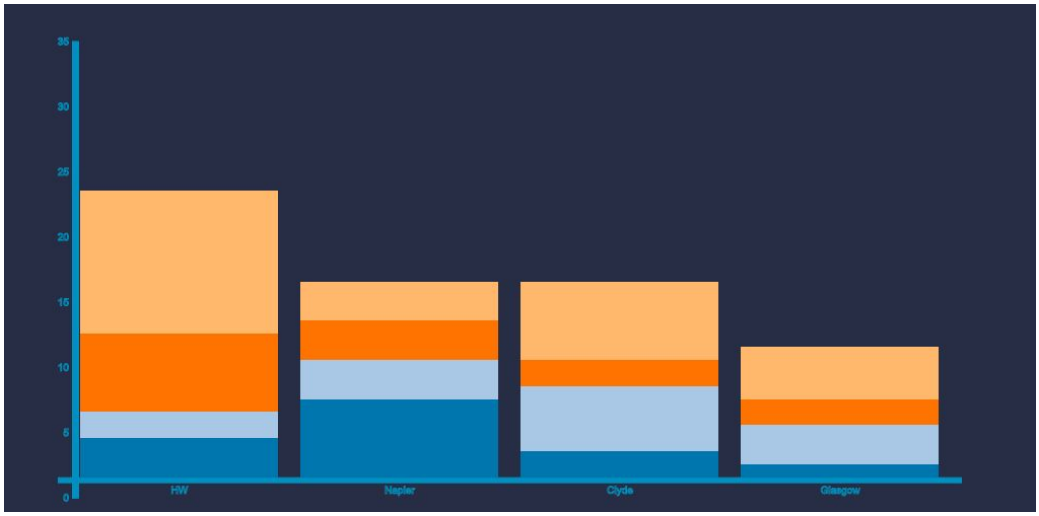
with the license stored in:

public/data/LICENSE.

R10

Use of a scatter plot, a stacked barchart or, a line chart.





As you can see in the 2 images above the project contains a stacked bar chart and a scatter plot.

R11

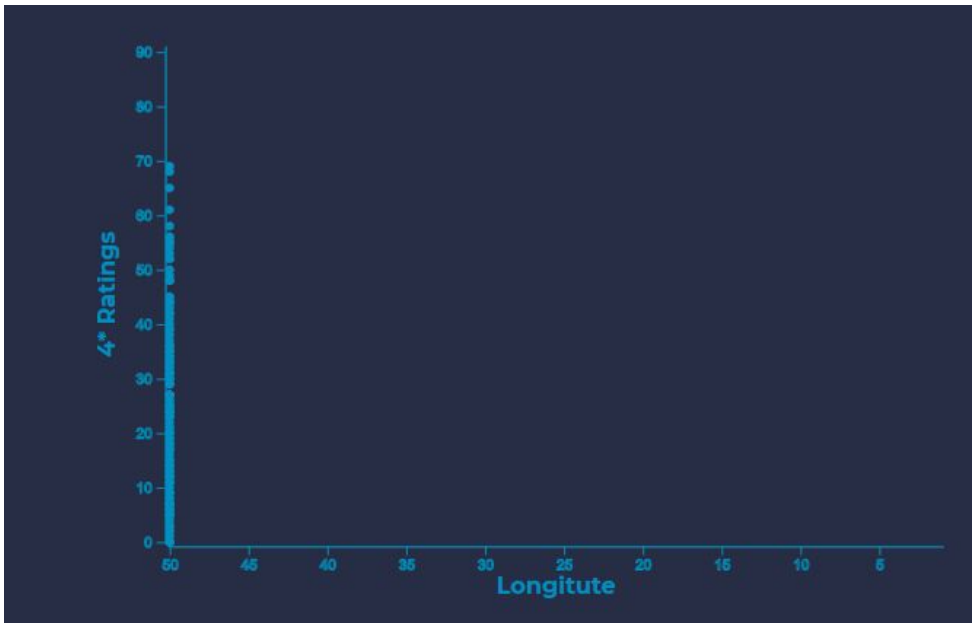
Use of bidirectional interaction between three charts.

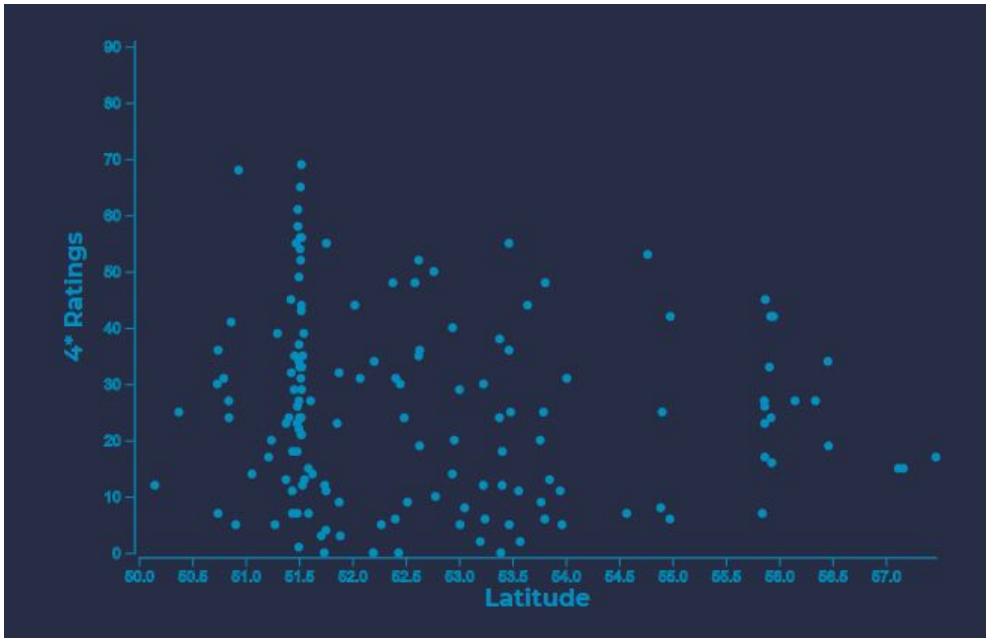
To complete this requirement, we combined the tree, the sunburst and the pack layout together, which all interact with each other. Clicking on one of the three layouts will alter the appearance of all three

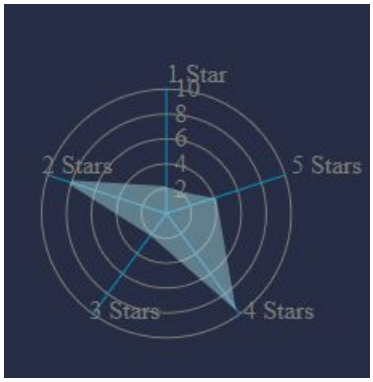
layouts. In the example showcased by the two pictures above, Scotland was clicked in the sunburst in picture 1, which altered the appearance of all three layouts, with the changes shown in picture two. If the pack layout or tree is then clicked all three layouts will change again.

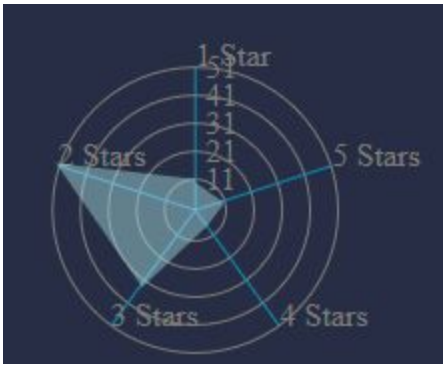
R12

Use of automatic scaling of all axes of a further two layouts during data update.





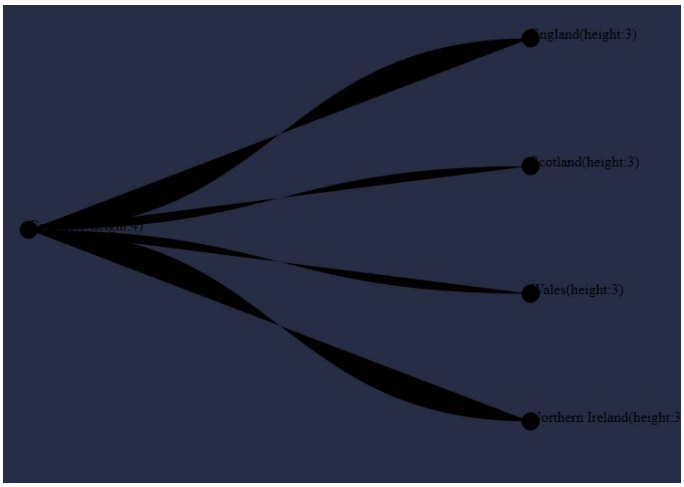


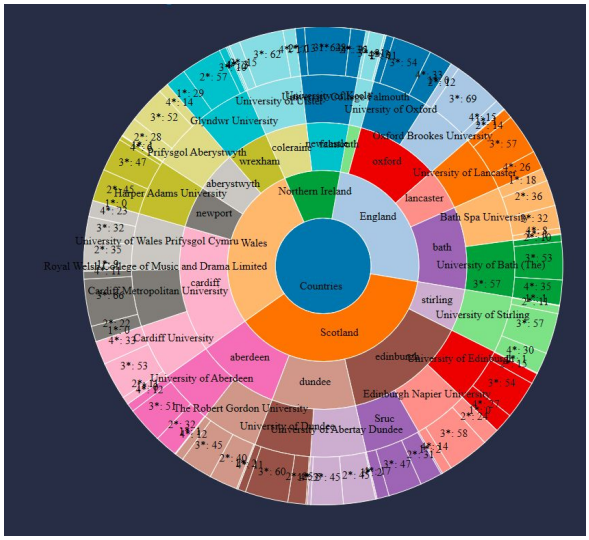


As can be seen in the screenshots of this requirement, 2 further layouts, the scatter plot and the radar graph, employ automatic axes updating when the data is changed. However, to change the axes of the scatter plot, we changed the data in the code to display the scaling, since displaying the 4 star rating in relation to the latitude was the best display view for the scatter plot.

R13

Use D3 hierarchical layout.

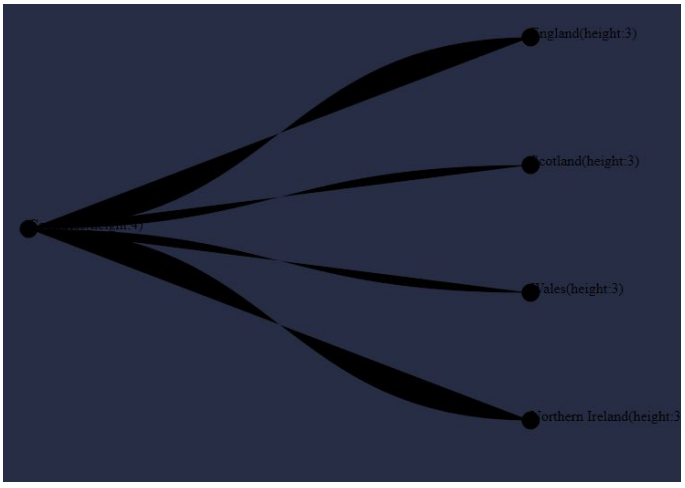




To complete the requirement we have a tree, a sunburst and a pack layout which all have a hierarchical structure.

R14

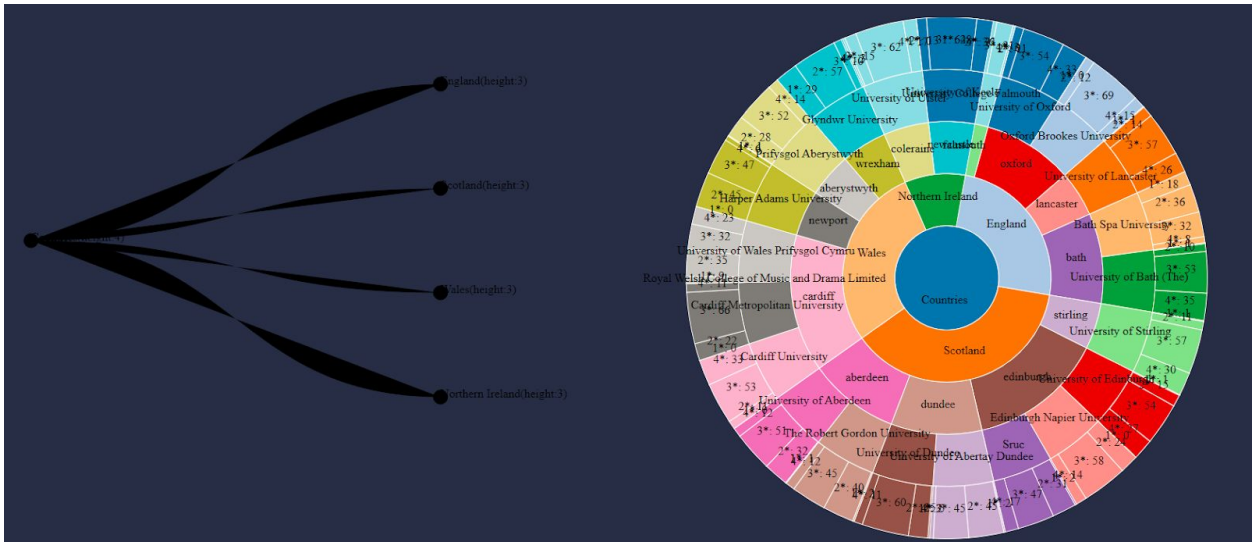
Use of selective reveal in a hierarchical layout display.

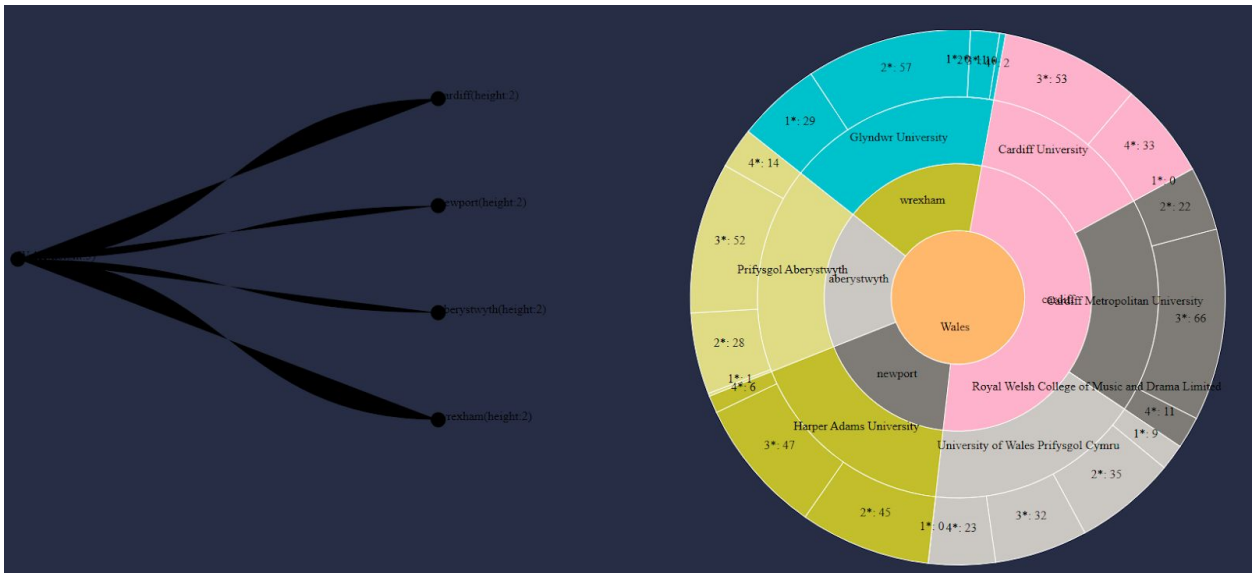


By clicking on a leaf node of the tree, it reveals the next level of the tree with the previous leaf node as the root of the new tree. This enables the user to only see a selected part of the tree, which manages to satisfy this requirement.

R15

Faceted selection interaction between two layouts (in which mouseover or click in one layout results in data being filtered in a second layout).





To count this requirement as completed, we have made an interaction between a tree and a sunburst, where clicking on one node in the tree or section in the sunburst will alter the appearance of both graphs by filtering down to the next level of the hierarchical tree. The example in the two screenshots display this transition, where Wales was clicked in the first picture, which resulted in the second picture being created where Wales is now the center point of the sunburst and the root of the tree.

R16

Use of a map layout that has interaction with another layout.

In this image there is a map of the UK and Ireland with data points for different universities. There is also a radar chart which shows the number of one to five stars. When clicking on a university on the map this updates the radar chart to show the data associated with the selected university.

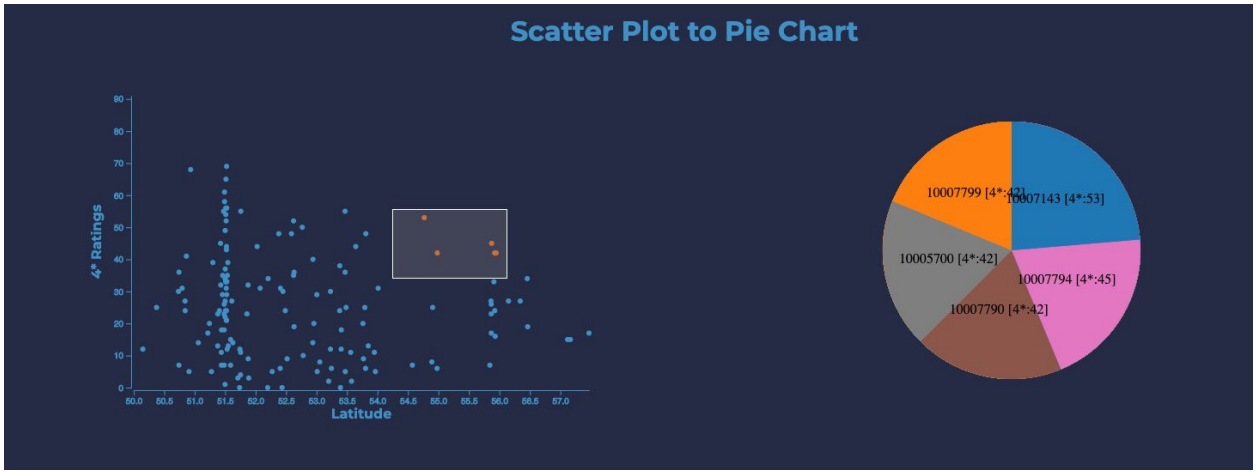
R17

Use of scalar data on a map.

This image shows the use of scalar data, by modifying the size of the point on the map depending on the size of the student population. Essentially, extra data about the quantity of enrolled students at each university was sourced online and used to determine the size of the dots used to represent universities.

R18

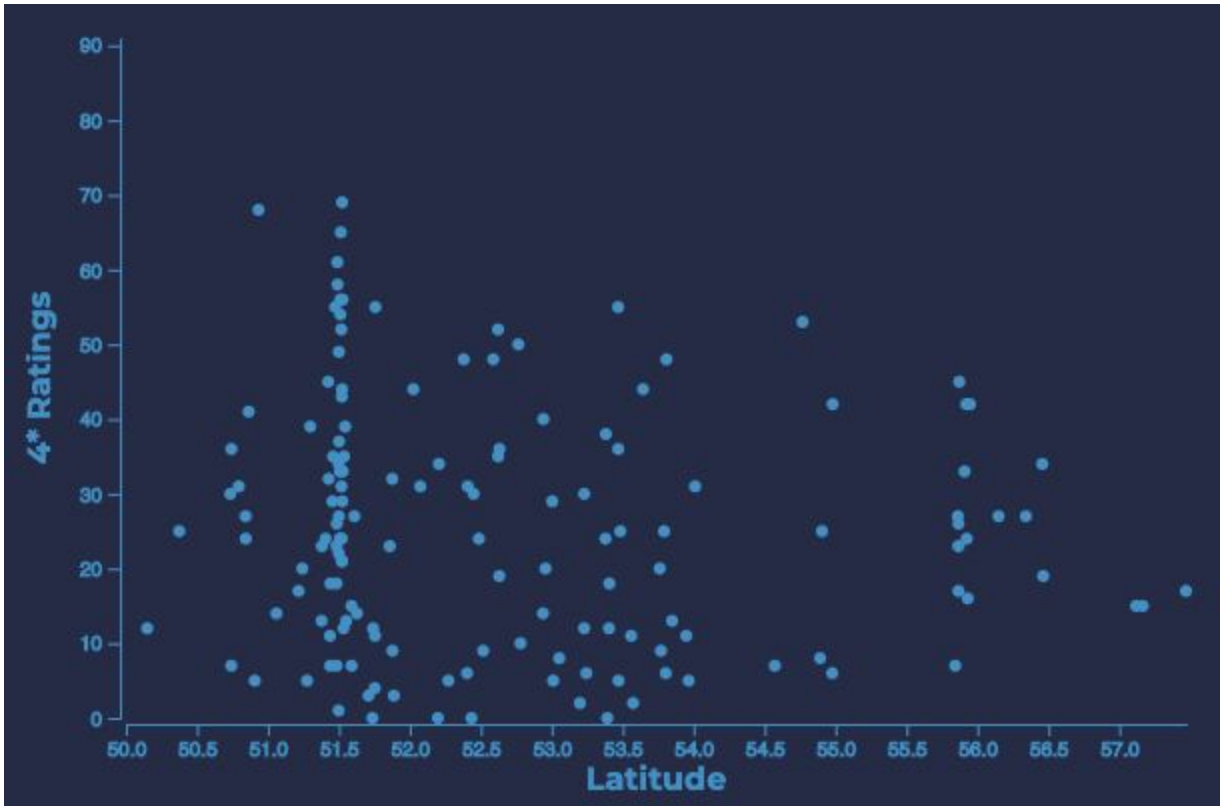
Use of cross-layout brushing in which dragging a rectangle over several data points in one chart highlights multiple associated data points in another chart.



For this requirement we use a scatter plot with brushing that updates a pie chart. The data in the scatter plot shows the number of four star ratings (Y-axis) and the Latitude of the university (X-axis). We enable brushing on this to highlight a rectangle of data points which update the pie chart with the universities selected.

R19

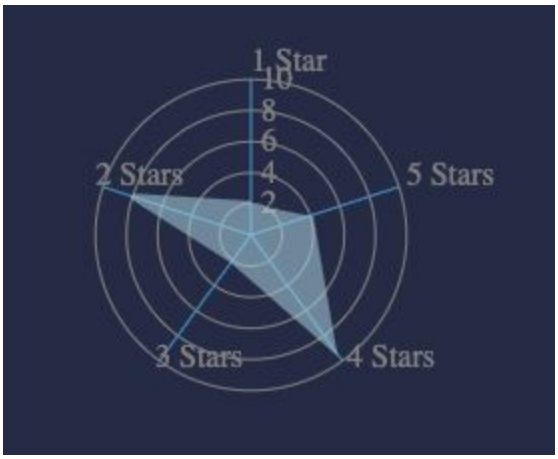
Use of the correlation between university/UOA star ratings and other data.



For this requirement, we opted for checking if there is a correlation between the latitude of a university and the number of 4 star ratings. This can be seen in the screenshot above of a scatter plot showing data points which are universities, with their latitude on the X axis, and the number of four star ratings on the X axis.

R20

Use of a custom shape generator.



A custom shape generator was used to create this chart, sometimes called a radar chart and others called a spider chart.

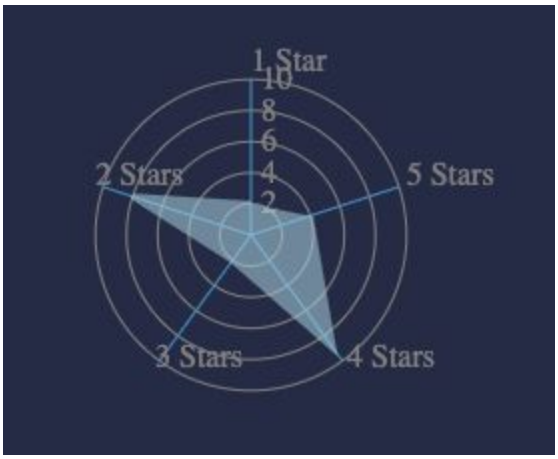
The custom shape drawn by this graph is the shape created when all data points are connected and filled (this can be seen as something similar to a triangle in the accompanying image). Here is a snippet of code showing the path being drawn and another showing the coordinates being calculated.

```
    radarChart.append("path")
        .datum(coordinates)
        .attr("d",line)
        .attr("stroke-width", 3)
        .attr("stroke", color)
        .attr("fill", color)
        .attr("stroke-opacity", 1)
        .attr("opacity", 0.5)
        .attr("class", "radarPath")
```

```
function getPathCoordinates(data_point){
    let coordinates = [];
    console.log("inside helper function")
    for (var i = 0; i < features.length; i++){
        let ft_name = features[i];
        let angle = (Math.PI / 2) + (2 * Math.PI * i / features.length);
        coordinates.push(angleToCoordinate(angle, data_point[ft_name]));
    }
    return coordinates;
}
```

R21

Use of a custom layout generator.



A custom layout generator was used to create this chart. The layout generator creates the circles which show the value of the distance from the center of the layout. It also creates axises, one for each feature (in this scenario star count). This can be seen in the following code snippet.

```
spiderChart
            .attr("width", 300)
            .attr("height", 300)

        ticks.forEach(t =>
            spiderChart.append("circle")
                .attr("cx", 100)
                .attr("cy", 100)
                .attr("fill", "none")
                .attr("stroke", "gray")
                .attr("class", "radar")
                .attr("r", radialScale(t))
        );

        ticks.forEach(t =>
            spiderChart.append("text")
                .attr("x", 105)
                .attr("y", 105 - radialScale(t))
                .text(t.toString())
                .attr("class", "radar")
                .attr("fill", "gray")
        );

        for (var i = 0; i < features.length; i++) {
```

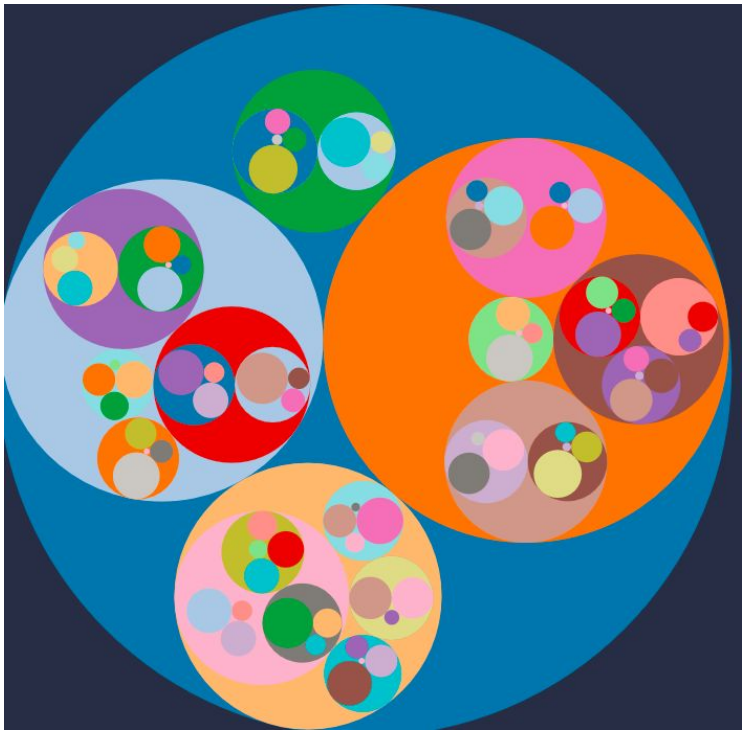
```
        let ft_name = features[i];
        let angle = (Math.PI / 2) + (2 * Math.PI * i / features.length);
        let line_coordinate = angleToCoordinate(angle, getMax(dataset) + 1);
        let label_coordinate = angleToCoordinate(angle, getMax(dataset) + 1.5);

        //draw axis line
        spiderChart.append("line")
            .attr("x1", 100)
            .attr("y1", 100)
            .attr("x2", line_coordinate.x)
            .attr("y2", line_coordinate.y)
            .attr("class", "radar")
            .attr("stroke","gray");

        //draw axis label
        spiderChart.append("text")
            .attr("x", label_coordinate.x)
            .attr("y", label_coordinate.y)
            .attr("fill", "gray")
            .attr("class", "radar")
            .text(ft_name);
```

R22

Use of a clustering analysis.



For the completion of this requirement we decided to make use of the pack layout which is a type of cluster layout, it clusters from universities in different countries together down to the star ratings of each individual university.