Data & Visualisation: London’s Crime Map

Link to GitHub:

https://github.com/AndriiArtemenko3/Data-Sensing-Visualisation---London-s-Crime-Map-Choropleth-

Introduction

My data visualisation explores the crime rates in London per borough on an interactive map. The idea is a continuation of my other project (for the HCI unit), where my project was a mobile app with an interactive map of London, with features like crowdsourcing (allowing the users to pin locations where they witnessed a crime or an accident). For this unit, I wanted to move my idea further and create a map which would show the data over time per all Greater London. I had no particular hypothesis in mind that I wanted to prove, instead I wanted to look at:

- if particular boroughs have consistently higher or lower rates vs the average
- if the boroughs that are closer to central areas (= more tourist activity) would show up with the higher rates
- if there are any seasonal changes to the crime rates in the city

I can see my target audience as tourists or people people who are new to London, as there is a lot of misinformation online regarding the crime situation in London, and when I first moved to London 2 years ago, it was quite hard for me to understand what the real picture is like. My project aims to provide a clear, unbiased and evidence-based view on the crime rates in the city.

Data Research & Processing

For accessing the data, I visited the London Datastore Website (https://data.london.gov.uk/dataset/recorded_crime_summary/) - For my project I selected the MPS Borough Level Crime for the past ~ 2 years. It would allow me to

work with a more manageable dataset that I can use for my visualisation, and still spot some important trends.

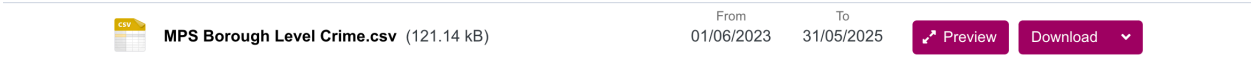

Fig 1. MPS Borough Level Crime.csv - my source data file

The csv table provides the data for all crime categories and the number of crimes reported for each month per borough.

MajorText	MinorText	BoroughName	202306	202307	202308	202309	202310	202311	202312	202401	202402	202403	202404	202405	202406	202407	202408	202409	202410	202411	202412	202501	202502
ARSON AND CRIMINAL DAMAGE	ARSON	Barking and Dagenham	2	5	7	4	4	3	4	4	5	6	3	8	2	3	10	9	5	7	9	7	8
ARSON AND CRIMINAL DAMAGE	CRIMINAL DAMAGE	Barking and Dagenham	122	131	114	128	82	88	125	126	124	133	112	102	105	132	114	79	103	96	89	104	89
BURGLARY	BURGLARY - RESIDENTIAL	Barking and Dagenham	0	0	0	0	0	0	0	0	2	50	52	42	37	37	46	49	62	51	83	71	48
BURGLARY	BURGLARY BUSINESS AND COMMUNITY	Barking and Dagenham	28	25	34	39	26	19	20	31	21	30	23	28	33	21	26	31	29	23	21	16	14
BURGLARY	BURGLARY IN A DWELLING	Barking and Dagenham	63	48	75	86	63	64	53	67	47	0	0	0	0	0	0	0	0	0	0	0	0
BURGLARY	BURGLARY NON-DWELLING	Barking and Dagenham	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0
DRUG OFFENCES	POSSESSION OF DRUGS	Barking and Dagenham	91	129	83	83	58	73	45	71	51	64	59	58	62	48	51	81	68	71	51	81	50
DRUG OFFENCES	TRAFFICKING OF DRUGS	Barking and Dagenham	41	22	15	19	25	23	19	28	18	42	21	29	32	22	17	40	44	76	46	30	37
FRAUD AND FORGERY	FRAUD AND FORGERY	Barking and Dagenham	0	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0
MISCELLANEOUS CRIMES AGAINST SOCIETY	MISC CRIMES AGAINST SOCIETY	Barking and Dagenham	22	26	15	27	22	23	23	27	25	16	20	29	17	27	24	27	29	27	23	24	16
POSSESSION OF WEAPONS	POSSESSION OF WEAPONS	Barking and Dagenham	25	19	21	15	9	16	16	9	13	8	10	10	4	9	8	17	8	12	6	10	6
PUBLIC ORDER OFFENCES	OTHER OFFENCES PUBLIC ORDER	Barking and Dagenham	9	12	11	14	11	8	12	7	13	12	24	13	15	18	14	21	18	12	20	12	10
PUBLIC ORDER OFFENCES	PUBLIC FEAR ALARM OR DISTRESS	Barking and Dagenham	74	68	103	104	62	90	65	69	92	68	65	76	56	68	61	78	66	55	44	54	38
PUBLIC ORDER OFFENCES	RACE OR RELIGIOUS AGG PUBLIC FEAR	Barking and Dagenham	32	42	19	23	24	18	26	13	13	21	22	23	21	29	41	37	29	16	22	13	17
PUBLIC ORDER OFFENCES	VIOLENT DISORDER	Barking and Dagenham	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0
ROBBERY	ROBBERY OF BUSINESS PROPERTY	Barking and Dagenham	2	3	9	11	9	16	9	4	9	10	8	10	15	9	18	15	19	17	8	10	15
ROBBERY	ROBBERY OF PERSONAL PROPERTY	Barking and Dagenham	76	86	83	60	32	48	46	46	73	101	63	43	51	46	35	59	93	67	38	32	72
SEXUAL OFFENCES	OTHER SEXUAL OFFENCES	Barking and Dagenham	43	45	21	27	28	17	32	29	18	36	35	38	26	43	26	32	34	34	33	24	29
SEXUAL OFFENCES	RAPE	Barking and Dagenham	20	16	20	19	12	13	29	13	21	18	34	23	22	16	23	21	27	35	27	17	28
THEFT	BICYCLE THEFT	Barking and Dagenham	11	17	15	12	10	9	10	5	6	4	9	8	10	4	9	13	9	16	5	5	2
THEFT	OTHER THEFT	Barking and Dagenham	209	188	177	182	167	157	179	160	136	160	144	165	173	159	124	138	153	151	163	134	115
THEFT	SHOPLIFTING	Barking and Dagenham	104	126	125	95	106	106	89	102	99	111	117	131	137	195	167	165	155	158	87	110	94
THEFT	THEFT FROM THE PERSON	Barking and Dagenham	53	62	34	52	62	52	66	50	63	77	64	74	74	54	44	53	74	68	73	70	53
VEHICLE OFFENCES	AGGRAVATED VEHICLE TAKING	Barking and Dagenham	3	2	3	3	0	2	3	1	2	3	2	1	2	2	1	1	0	1	0	2	1
VEHICLE OFFENCES	INTERFERING WITH A MOTOR VEHICLE	Barking and Dagenham	17	23	18	33	25	15	31	29	27	25	17	15	20	22	16	16	30	37	13	22	13

Fig 2. Unfiltered Dataset

I filtered the data into a new dataset table that has three columns only: Borough's name, Total Number of Crimes (per month), and Month. To find the total number of crimes per borough per given month I summed all the crime numbers for all the categories per given month. I did it by hand in excel using 'sum' formula and then I transferred the data into the new file.

crime_data

BoroughName	Crimes	Month
Barking and Dagenham	1898	2023-06
Barking and Dagenham	1897	2023-07
Barking and Dagenham	1818	2023-08
Barking and Dagenham	1884	2023-09
Barking and Dagenham	1497	2023-10
Barking and Dagenham	1576	2023-11
Barking and Dagenham	1698	2023-12
Barking and Dagenham	1678	2024-01
Barking and Dagenham	1635	2024-02
Barking and Dagenham	1766	2024-03
Barking and Dagenham	1575	2024-04
Barking and Dagenham	1714	2024-05

Fig 3. Filtered Dataset

SVG Processing

P.S. My first iteration was done with GeoJSON, but later I discarded that prototype, as SVG is better for this scope of project (easier to integrate, can edit in graphic editors + I have more experience with them).

For my map of London boroughs I used a vector .svg file which I got from the Adobe Stock (I have a student subscription). Then I moved the svg to figma, to edit it and rename the layers.

Fig. 4. Original svg file

In figma, I removed the names of the boroughs on the map (i will show them later via D3), and other vector elements like the Thames, and borders of the boroughs. I also simplified the structure of the svg by removing the groups and leaving just the vector elements, for better control via D3. I renamed all the vectors to match the names of the boroughs in my .csv table.

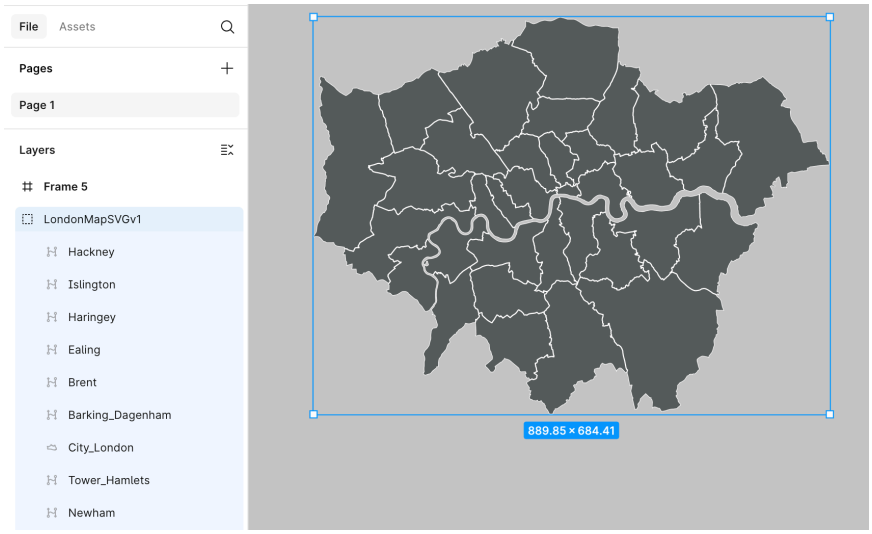

Fig 5. Processed SVG

Early Design & Testing

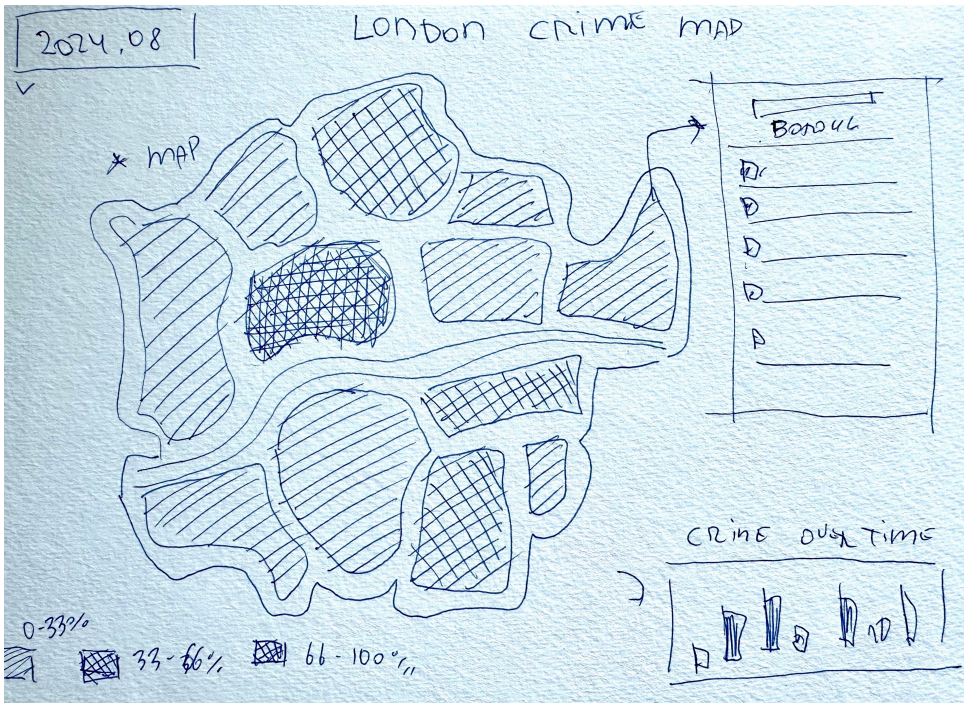

Fig 6. My initial sketch of the layout

After sketching out the layout, I started building my first visualisation prototypes in VSC using my pre-made components (svg and csv). I started by simply displaying my svg and changing appearance of one of the boroughs via d3.append functionality. This way I could test if my svg works with d3.js initially.

Fig 7. Displaying the svg and manipulating it via d3

Then I loaded my csv as an array and I run the console command to see if it loaded correctly.

```
Live reload enabled.                                    index.html:40
CSV loaded  ▸ Array(816)                                app.js:27
```

Fig 8. CSV loaded successfully

At the early stage of development, I also noticed that my color ranges don't show up properly on the map. To find the issue, I colored the all boroughs in blue color, and then by hand I commented out the <mask> section in the svg, to see if it would make a difference, and it did - when I exported my svg from figma, I didn't have a mask there, but it was in the svg's code. So I commented out all the <mask> code from my svg.

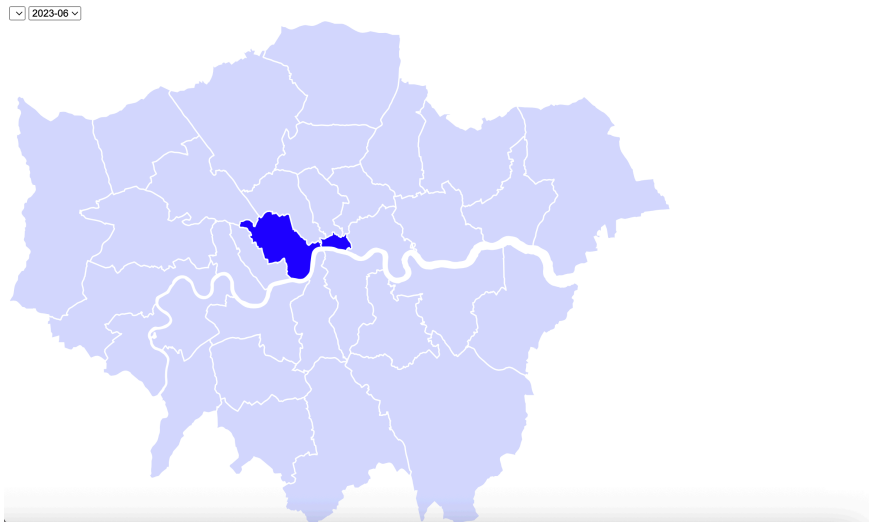

Fig 9. SVG map with one borough's <mask> removed

Main Logic

```
function getColorScale(data) {
    var crimes = data.map(d => +d.Crimes).sort(d3.ascending);
    var thresholds = [0.2, 0.4, 0.6, 0.8].map(p => d3.quantileSorted(crimes, p));
    return d3.scaleThreshold()
      .domain(thresholds)
      .range(["#FFE2E2", "#FFA2A2", "#FB2C36", "#C10007", "#82181A"]); // first went with d
}
```

Fig. 10. Main function for coloring the map

This function is one of the most important in my work, as it defines the percentile groups and color ranges that corresponds to them on the map. For my design I decided to limit the percentile groups to 5, as breaking it down further would make it more difficult to read the map.

For the color range I first went with D3's pre-built schemes (e.g. d3.schemeBlues[5]), then eventually decided to make my own for better control of the visualisation. I love Tailwind, so I used their color schemes (red → 100,300,500,700,900)

Fig. 11. Custom color range built using tailwind and copied in figma

What makes my map functional just beyond coloring is the dropdown menu. Dropdown menu allows the user to choose a specific month from the list which they want to see on the map.

```
// adding the month selector ( dropdown menu ) for the map
var selector = d3.select("body")
.insert("select", "#svg-container")
.attr("id", "month-selector");
```

Fig. 12. Adding the month selector to D3 with DOM manipulation

```
// Binding the data from the months column to the dropdown menu
selector.selectAll("option")
  .data(uniqueMonths)
  .enter()
  .append("option")
  .attr("value", d => d)
  .text(d => d);
```

Fig. 13. Binding the month data from .csv to the month-selector

```
selector.on("change", (event) => {
  var selected = event.target.value;
  var rowsForMonth = data.filter(row => row.Month === selected);
  var colorScale = getColorScale(rowsForMonth);

  applyChoroplethColoring(rowsForMonth, colorScale);
});
```

Fig. 14. Event listener for the month-selector

This function updates the rendering of the map when the event for month-selector is triggered

```
function applyChoroplethColoring(rows, colorScale) {
  rows.forEach(row => {
    var id = row.BoroughName.replaceAll(" ", "_"); // not strictly necessary for my csv but
    var crimes = +row.Crimes;

    var target = d3.select(`#${id}`);
    if (!target.empty()) {
      var paths = target.selectAll("path");
      paths
        .attr("fill", colorScale(crimes))
        .on("mouseover", function (event) {
          d3.select("#tooltip")
            .style("opacity", 1)
            .html(`<strong>${row.BoroughName}</strong><br/>Crimes: ${crimes}`);
        })
        .on("mousemove", function (event) {
          d3.select("#tooltip")
            .style("left", (event.pageX + 10) + "px")
            .style("top", (event.pageY - 28) + "px");
        })
        .on("mouseout", function () {
          d3.select("#tooltip").style("opacity", 0);
        });
    }
  });
}
```

Fig. 15. Applying color to the Choropleth

Also adding the tooltip and legend are very important in making the visualisation more powerful, readable and accessible without overloading the user with information straight away. I made a tooltip that helps user identify the borough and exact number of crimes when just hovering over it. The legend tells the user immediately what the colors on my choropleth map represent.

```css
#tooltip {
  position: absolute;
  background: white;
  padding: 6px;
  border: 1px solid #aaa;
  border-radius: 4px;
  font-size: 12px;
  pointer-events: none;
  opacity: 0;
  transition: opacity 0.2s ease;
  z-index: 1000;
}

#legend {
  margin-top: 20px;
  font-family: sans-serif;
}

.legend-item {
  display: flex;
  align-items: center;
  margin-bottom: 6px;
}

.color-box {
  display: inline-block;
  width: 20px;
  height: 20px;
  margin-right: 8px;
  border: 1px solid #888;
}
```

Fig 16. CSS classes for tooltip and legend

To further enhance my app and aid in data analysis (spotting trends over time per borough), I made bar graph visualisation of crime rates.

```js
function renderBarChart(data, selectedBorough) {
  var svg = d3.select("#bar-chart");

  clearChart(svg); // important because it prevents axes from duplicating every time I chnage
```

Fig. 17. DOM + cleaning up the bar graph to avoid visual clutter

```js
var boroughData = data.filter(d => d.BoroughName === selectedBorough);
```

Fig. 18. Filtering data for selected borough only

```
function getXScale(data) {
  return d3.scaleBand()
    .domain(data.map(d => d.Month))
    .range([40, 580])
    .padding(0.1);
}

function getYScale(data) {
  return d3.scaleLinear()
    .domain([0, d3.max(data, d => +d.Crimes)])
    .range([260, 20]);
}
```

Fig. 19. Defining the dimensions of the bars

```
  svg.append("g")
    .attr("transform", "translate(0,260)")
    .call(d3.axisBottom(x).tickValues(x.domain().filter((d, i) => i % 2 === 0)));

  svg.append("g")
    .attr("transform", "translate(40,0)")
    .call(d3.axisLeft(y));
```

Fig. 20. Adding axes for readability

```
  svg.selectAll("rect")
    .data(boroughData)
    .enter()
    .append("rect")
    .attr("x", d => x(d.Month))
    .attr("y", d => y(+d.Crimes))
    .attr("width", x.bandwidth())
    .attr("height", d => 260 - y(+d.Crimes))
    .attr("fill", "#FB2C36");
```

Fig. 21. Drawing the bars

Final Outcome

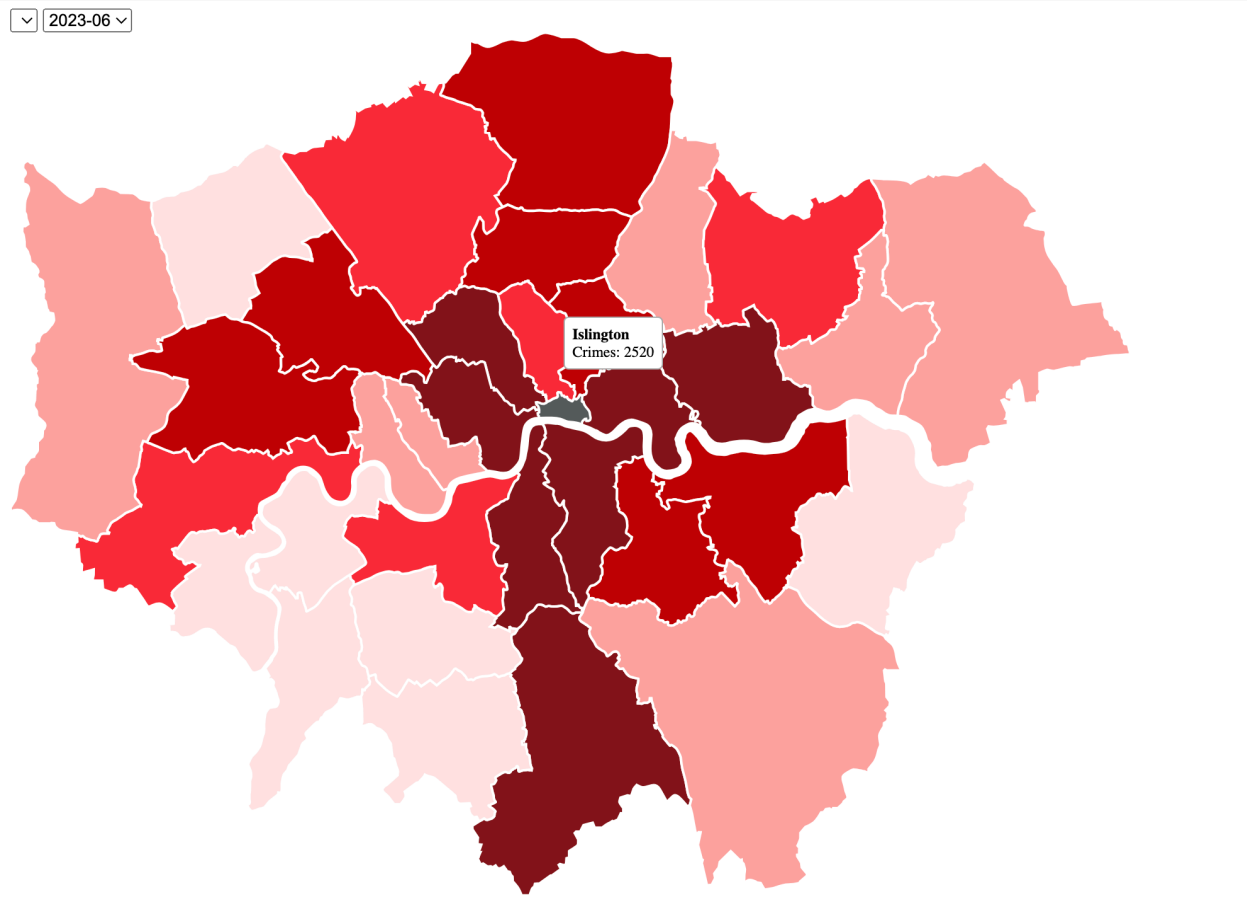

Fig. 22. Web App - Choropleth Map

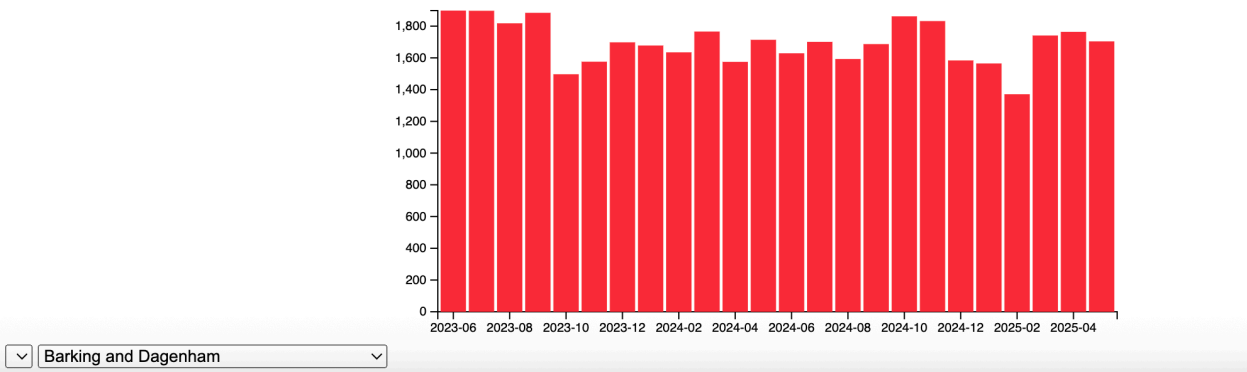

Fig. 23. Web App - Bar Graph

References

ChatGPT use

During the development process, I used OpenAI's ChatGPT as a support tool to assist with:

- Debugging specific errors
- Clarifying D3.js syntax and functionality
- Reviewing general coding logic
- Exploring alternative approaches to implementation

No code was directly copied or pasted from ChatGPT. All development decisions, design choices, and implementation work reflect my own understanding, learning, and effort. ChatGPT was used in a similar manner to how one might consult documentation, forums, or a peer discussion for guidance.

Learning Materials

Bostock, M. and Observable, Inc. (2025) *Threshold scales*. D3.js. Available at: https://d3js.org/d3-scale/threshold (Accessed: 16 June 2025).

Bostock, M. (ca. 2018) *Threshold Choropleth*. Observable. Available at: https://observablehq.com/@d3/threshold-choropleth (Accessed: 16 June 2025).

Mike Bostock (2021) *D3.js in 10 Minutes or Less | ep. 002 – Scales!* YouTube, 27 February 2022. Available at: https://www.youtube.com/watch?v=QrlreXCYC08 (Accessed: 16 June 2025).

Kelleher, C. (c. 2018) *Customizing Axes of a Bar Chart with D3.js*. YouTube, 6 September 2018 . Available at: https://www.youtube.com/watch?v=c3MCROTNN8g (Accessed: 16 June 2025).