Assignment 3

VIKRAMADITYA REDDY VARKALA

Z1973679

Data Visualization(CSCI 627)

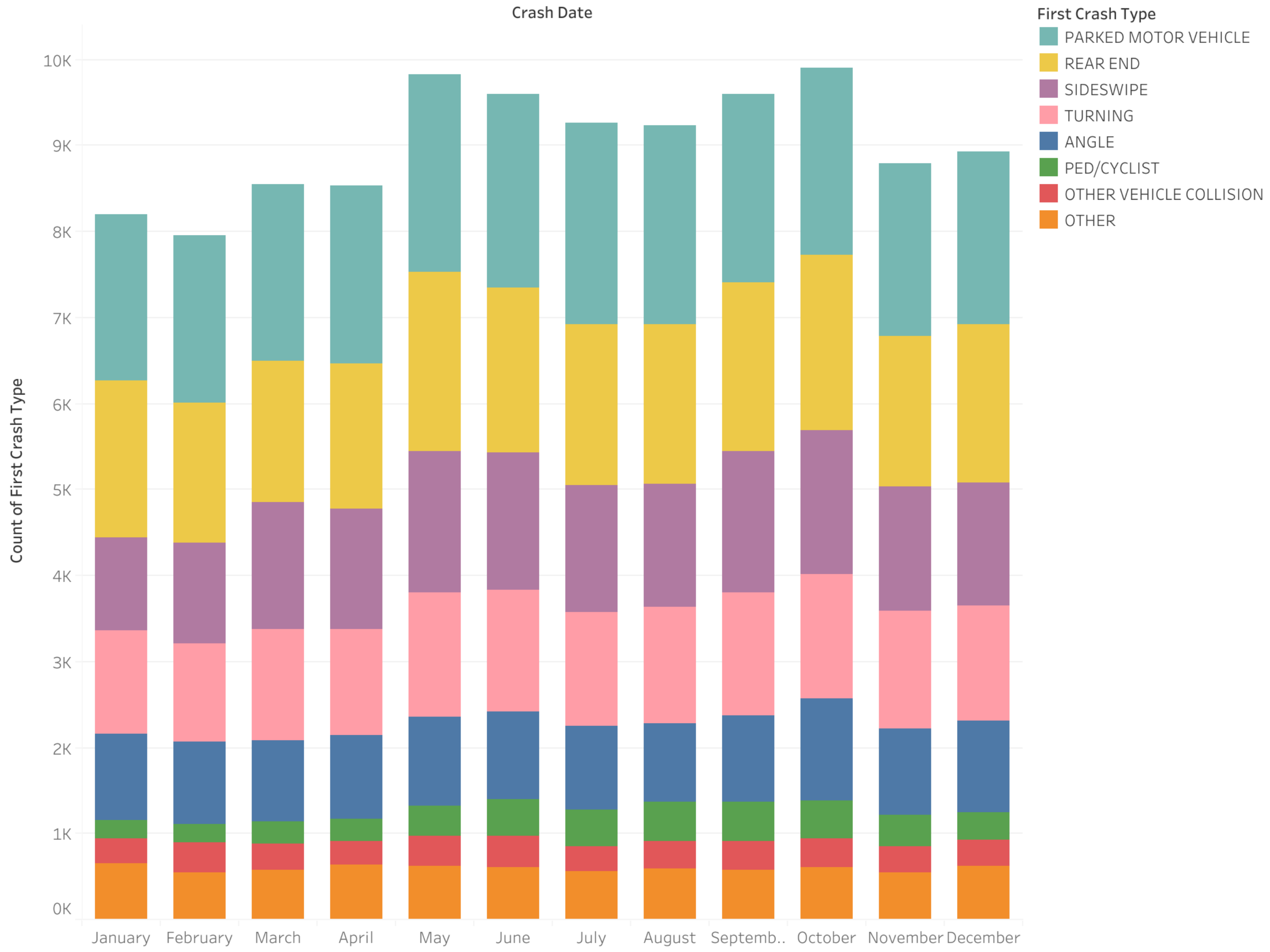
2023-10-11

1. Tableau (25 points)

LINK TO BAR CHART :

https://public.tableau.com/views/A3_Part1_Tableau/Sheet1?:language=en-US&:display_count=n&:origin=viz_share_link

Chicago Traffic Crashes 2022



2. Observable Plot (30 points)

```
data = ▸ Array(108395) [Object, Object, Object, Object, Object, Object, Object, Object, Obje
```

```
monthFormat = ƒ(e)
```

```
groupData = ▶ Array(12) [Array(2), Array(2), Array(2), Array(2), Array(2), Array(2), Array(2
```

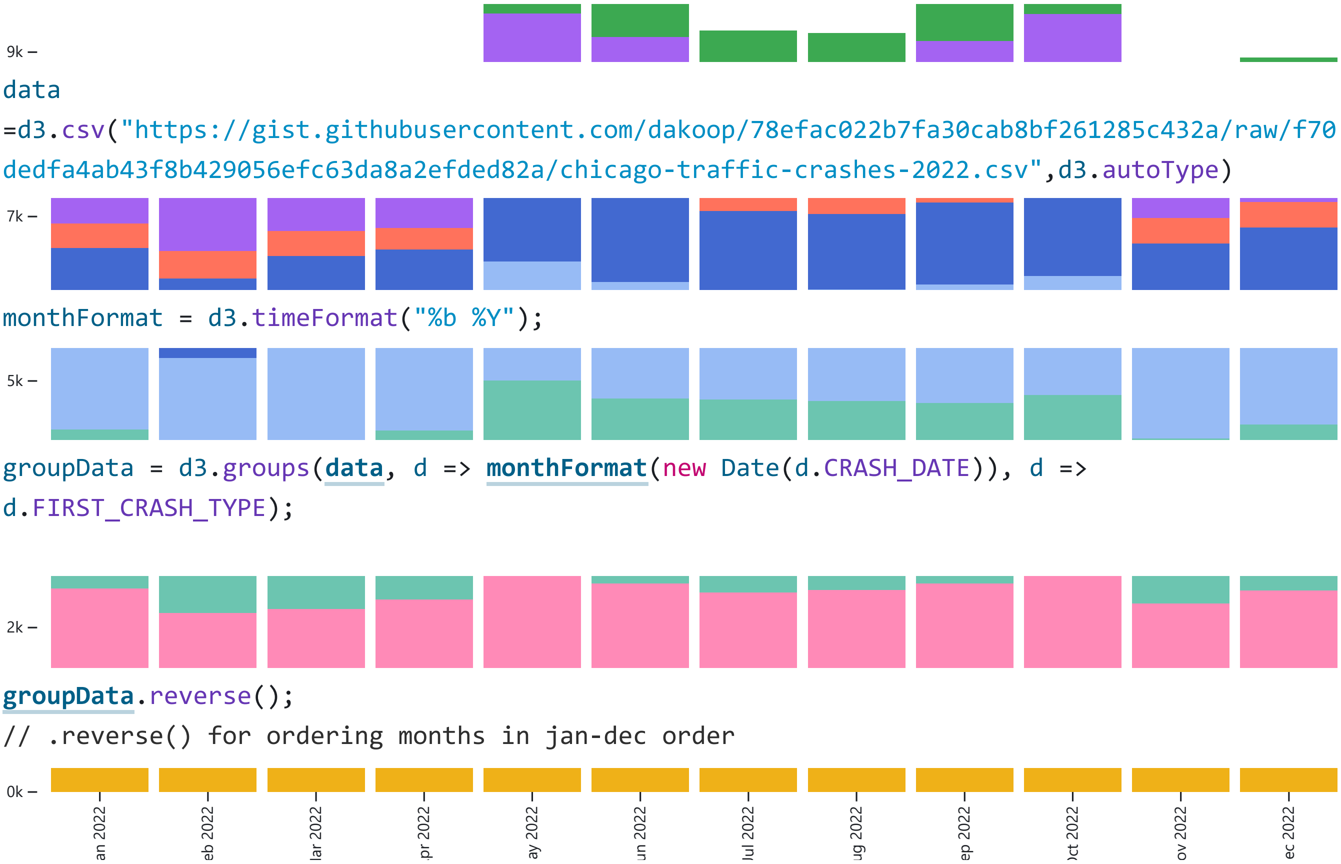
```
▶ Array(12) [Array(2), Array(2), Array(2), Array(2), Array(2), Array(2), Array(2), Array(2),
```

```
plotData = ▶ Array(96) [Object, Object, Object, Object, Object, Object, Object, Object, Obje
```

```
crashes = ▶ Array(8) ["OTHER", "REAR END", "PARKED MOTOR VEHICLE", "TURNING", "ANGLE", "OTHE
```

```
data1 = ▶ Array(96) [Object, Object, Object, Object, Object, Object, Object, Object, Object,
```

🟦 ANGLE 🟨 OTHER 🟥 OTHER VEHICLE COLLISION 🟩 PARKED MOTOR VEHICLE 🟩 PED/CYCLIST 🟪 REAR END 🟪 SIDESWIPE
🟦 TURNING



```
data =d3.csv("https://gist.githubusercontent.com/dakoop/78efac022b7fa30cab8bf261285c432a/raw/f70dedfa4ab43f8b429056efc63da8a2efded82a/chicago-traffic-crashes-2022.csv",d3.autoType)
```

```
monthFormat = d3.timeFormat("%b %Y");
```

```
groupData = d3.groups(data, d => monthFormat(new Date(d.CRASH_DATE)), d => d.FIRST_CRASH_TYPE);
```

```
groupData.reverse();
// .reverse() for ordering months in jan-dec order
```

```
plotData = {
  const output = [];
groupData.forEach(([date, crashGroups]) => {
    crashGroups.forEach(([crashType, entries]) => {
        const totalCount = entries.length;
        output.push({
            date: date,
            type: crashType,
            count: totalCount
        });
    });
});
return output;
}
```

```
crashes = Object.keys(crashesForStack[0]).filter(d => d !== 'Total' && d !== 'Date');
// keys excluding Total and Date
//i have used this later in the assigment
```

References:1) https://observablehq.com/@observablehq/plot-stacked-bar-chart

```
//sorts crash types in the stack by comparing the index of crash types.
data1 = plotData.sort((a, b) => crashes.indexOf(a.type) - crashes.indexOf(b.type));
```

3. D3 (30/45 points)

```
crashesMonthType = ▶ Map(12) {11 => Map(8), 10 => Map(8), 9 => Map(8), 8 => Map(8), 7 => Map
```

```
crashesForStack = ▶ Array(12) [Object, Object, Object, Object, Object, Object, Object, Objec
```

```
▶ Array(12) [Object, Object, Object, Object, Object, Object, Object, Object, Object, Object,
```

```
stack = ƒ(i)
```

```
series = ▶ Array(4) [Array(12), Array(12), Array(12), Array(12)]
```

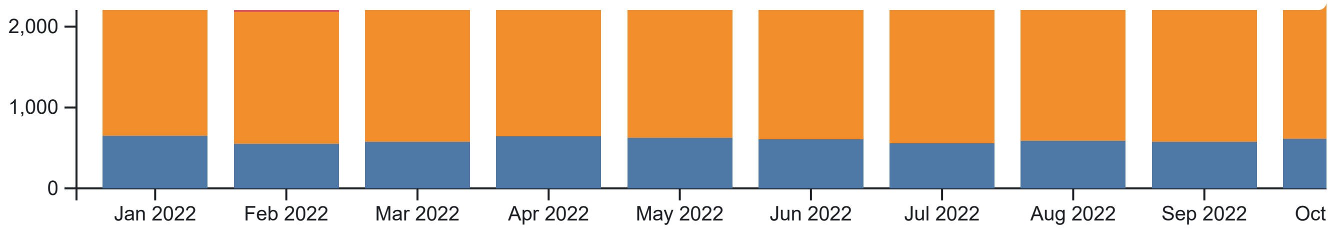
```
height = 400
```

```
width = 800
```

```
import {Swatches} from "@d3/color-legend"
```

```
  //plot stacked barchart
  Plot.plot({
    marginBottom: 60,
    width: 900,
    height: 600,
    x: { label: "Crash Date" ,tickRotate: -90},
    y: { tickFormat: "s", tickSpacing: 40 , label:"Count of First Crash Type"},
    color: { legend: true, label: "First Crash Type" },
    marks: [
      Plot.barY(data1, {
        x: "date",
        y: "count",
        fill: "type",

        sort: {x: null,reverse: true}
    })
  ]
});
```



```
crashesMonthType = d3.rollup(data, v => v.length, d => d.CRASH_DATE.getMonth(), d =>
d.FIRST_CRASH_TYPE);
```

```
crashesForStack = [...crashesMonthType.entries()].map(d =>
({...Object.fromEntries([...d[1].entries()]), "Total": d3.sum(d[1].values()), "Date":
d[0]}))
```

```
crashesForStack.reverse();
//for ordering months in jan-dec order
```

```
series = stack(crashesForStack);
```

```
height= 400
```

```
width = 800
```

```
import {Swatches} from "@d3/color-legend"
```

```
{
  const margin = {top: 20, bottom: 20, left: 80, right: 10};

  const svgElt = d3.create("svg")
    .attr("width", width + margin.left + margin.right)
    .attr("height", height + margin.top + margin.bottom);

  const svg = svgElt.append("g")
    .attr("transform", `translate(${margin.left}, ${margin.top})`);


  // Stacks data
  const stack = d3.stack()
    .keys(crashes)//from part 2 of assignment.
  const series = stack(crashesForStack);
  const monthNames = crashesForStack.map(d => d3.timeFormat("%b %Y")(new Date(2022,
d.Date)));


  //scale
  const xScale = d3.scaleBand()
    .domain(monthNames)
    .range([0, width])
    .padding(0.2);

  const yScale = d3.scaleLinear()
```

```
    .domain([0, d3.max(series, d => d3.max(d, d => d[1]))])
    .range([height, 0]);



  // axes
  const xAxis = d3.axisBottom(xScale);
  svg.append("g")
    .attr("transform", `translate(0,${height})`)
    .call(xAxis);

  const yAxis = d3.axisLeft(yScale);
  svg.append("g").call(yAxis);


  // stacked bar colour.
  const color = d3.scaleOrdinal()
    .domain(crashes)
    .range(d3.schemeTableau10);

  svg.append("g")

  .selectAll("g")
  .data(series)

  .enter()
  .append("g")
    .attr("fill", d => color(d.key))

  .selectAll("rect")
  .data(d => d)

  .enter()
  .append("rect")
    .attr("x", (d, i) => xScale(monthNames[i]))
    .attr("y", d => yScale(d[1]))
    .attr("height", d => yScale(d[0]) - yScale(d[1]))
    .attr("width", xScale.bandwidth());

 return svgElt.node();
}
```