EECS E6893 Big Data Analytics

Homework 3 Report

Name: Yi Zhang

Uni: yz4130

November 2022

Part I

Problem 1

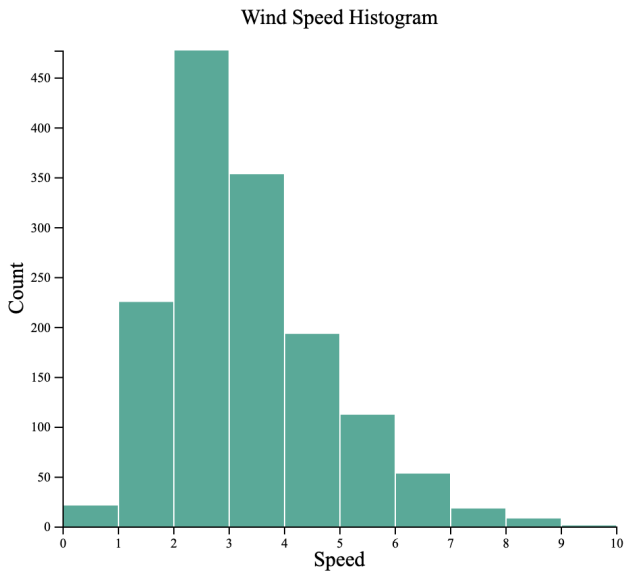
1. A mathematical graph usually places the origin point in the lower left corner of the chart while SVG places the origin in the upper left corner. Thus y-axis is flipped and increases as it goes down.
2. When dealing with transition, *enter()* controls how elements enter the page, and *exit()* controls how they exit the page. If we have more data than elements, the extra data will be bound to elements selected by “enter”. If we have more elements than data, extra elements will belong to the “exit” selection.
3. Transform can define transformations that can be applied to elements as well as their children. Translate is a kind of transformation that moves all the points of an element in the same direction and by the same amount. Translation preserves parallelism, angles, and distances.
4. The anonymous function in d3 is like the lambda function in Python. In the example, the result will be a list like [5, 6, 7, 8, 9] since i represents the index.
5. Outputs will be different. Snippet one will show *“big”* and snippet two will show *“big data assignment 2 submission”*. There is only one element “p” in one but we have more data, without applying the enter function then the program will only use the first data which is big. However, if we use the *“enter”* and *“append”,* all the data will be used and separated with space as we defined the function.

Problem 2

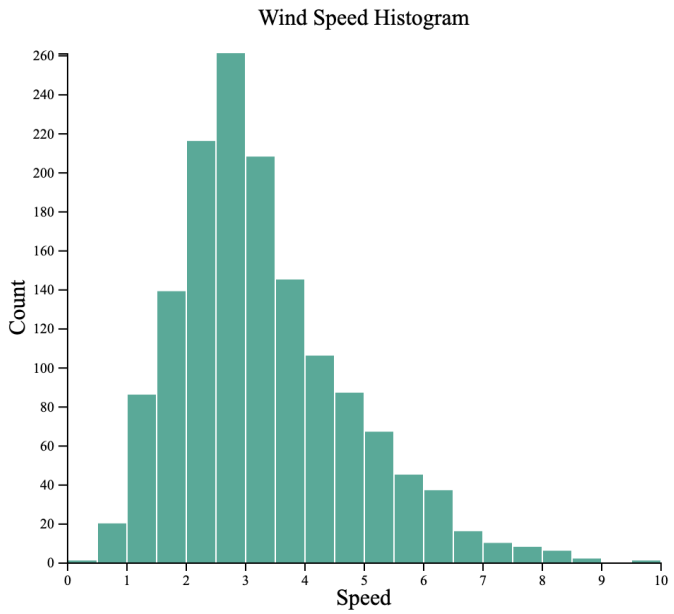
1. The code is as follows

```
d3.csv("seattle-weather.csv",
    function(data) {
        var x = d3.scaleLinear()
            .domain([0, 10])
            .range([0, width]);
        svg.append("g")
            .attr("transform", "translate(0," + height + ")")
            .call(d3.axisBottom(x));
        var histogram = d3.histogram()
            .value(function(d) { return d.wind; })
            .domain(x.domain())
            .thresholds(x.ticks(10));
        var bins = histogram(data);
        var y = d3.scaleLinear()
            .range([height, 0]);
            y.domain([0, d3.max(bins, function(d) { return d.length; })]);
        svg.append("g")
            .call(d3.axisLeft(y));
        svg.selectAll("rect")
            .data(bins)
            .enter()
            .append("rect")
                .attr("x", 1)
                .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
                .attr("width", function(d) { return x(d.x1) - x(d.x0) - 1; })
                .attr("height", function(d) { return height - y(d.length); })
                .style("fill", "#69b3a2");
        svg.append("text")
            .attr("x", (width / 2))
            .attr("y", 0 - (margin.top / 2))
            .attr("text-anchor", "middle")
            .style("font-size", "16px")
            .text("Wind Speed Histogram");
        svg.append("text")
            .attr("transform", "translate(" + (width / 2) + " ," + (height + 30) + ")")
            .style("text-anchor", "middle")
            .style("font-size", "16px")
            .text("Speed");
        svg.append("text")
            .attr("transform", "rotate(-90)")
            .attr("x", -(height / 2))
            .attr("y", -30)
            .style("text-anchor", "middle")
            .style("font-size", "16px")
            .text("Count");
```

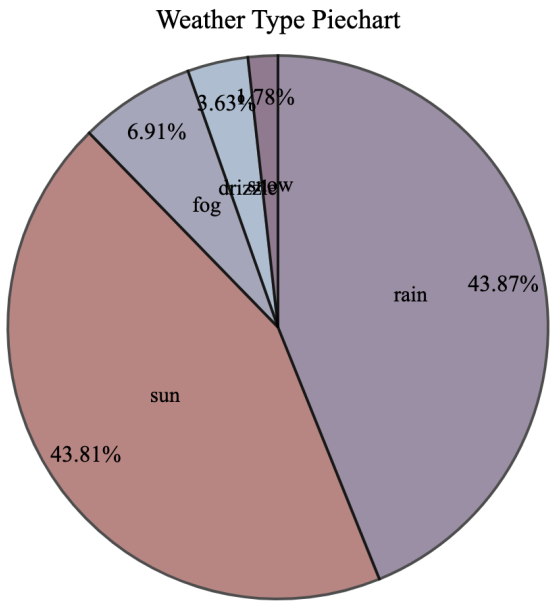
The plot is



2. Change number 10 into 25 and we will get the following plot



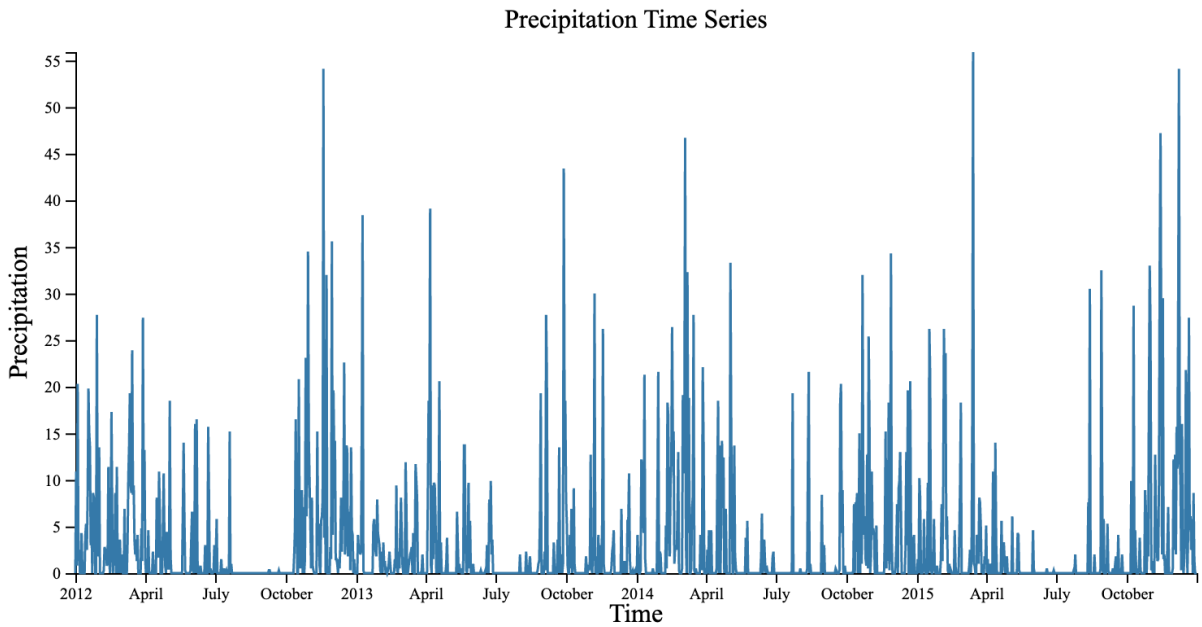
3.


Weather Type Piechart
6.91%
3.63%
1.78%
fog
drizzle
snow
rain
43.87%
sun
43.81%

```
<script>

    var width = 600,
        height = 600,
        margin = 100;

    var radius = Math.min(width, height) / 2 - margin;

    var svg = d3.select("#dataviz")
        .append("svg")
        .attr("width", width)
        .attr("height", height)
        .append("g")
        .attr("transform", "translate(" + width / 2 + "," + height / 2 + ")");

    d3.csv("seattle-weather.csv", function(data) {

        var drizzle = 0,
            fog = 0,
            rain = 0,
            snow = 0,
            sun = 0;

        for (var i = 0; i < data.length; i++) {
            if (data[i].weather == "drizzle") {
                drizzle += 1;
            } else if (data[i].weather == "fog") {
                fog += 1;
            } else if (data[i].weather == "rain") {
                rain += 1;
            } else if (data[i].weather == "snow") {
                snow += 1;
            } else{
                sun += 1;
            }
        }

        var dict = {
            "drizzle": drizzle,
            "fog": fog,
            "rain": rain,
            "snow": snow,
            "sun": sun
        }
        var sum = drizzle + fog + rain + snow + sun

        var color = d3.scaleOrdinal()
            .domain(dict)
            .range(["#98abc5", "#8a89a6", "#7b6888", "#6b486b", "#a05d56"]);

        var percentageFormat = d3.format(".2%");

        var pie = d3.pie()
            .value(function(d) { return d.value; })
        var data_ready = pie(d3.entries(dict))

        var arcGenerator = d3.arc()
            .innerRadius(0)
            .outerRadius(radius)

        var arc = d3.arc()
            .outerRadius(radius + 80)
            .innerRadius(60);

        svg.selectAll('mySlices')
            .data(data_ready)
            .enter()
            .append('path')
                .attr('d', arcGenerator)
                .attr('fill', function(d) { return(color(d.data.key)) })
                .attr("stroke", "black")
                .style("stroke-width", "2px")
                .style("opacity", 0.7)

        svg.selectAll('mySlices')
            .data(data_ready)
            .enter()
            .append('text')
            .text(function(d) { return d.data.key })
            .attr("transform", function(d) { return "translate(" + arcGenerator.centroid(d) + ")" })
            .style("text-anchor", "middle")
            .style("font-size", 16)

        svg.selectAll('mySlices')
            .data(data_ready)
            .enter()
            .append("text")
            .attr("transform", function(d) {
                return "translate(" + arc.centroid(d) + ")";
            })
            .attr("dy", ".35em")
            .style("text-anchor", "middle")
            .text(function(d) {
                return percentageFormat(d.data.value / sum);
            });

        svg.append("text")
            .attr("x", 0)
            .attr("y", -margin - 120)
            .attr("text-anchor", "middle")
            .style("font-size", "20px")
            .text("Weather Type Piechart");

    });
</script>
```

4.

```javascript
d3.csv("seattle-weather.csv",
    function(d) {
        return { date : d3.timeParse("%Y-%m-%d")(d.date), precipitation : d.precipitation }
    },

    function(data) {
        var x = d3.scaleTime()
            .domain(d3.extent(data, function(d) { return d.date; }))
            .range([0, width]);
        svg.append("g")
            .attr("transform", "translate(0," + height + ")")
            .call(d3.axisBottom(x));

        var y = d3.scaleLinear()
            .domain([0, d3.max(data, function(d) { return +d.precipitation; })])
            .range([height, 0]);

        svg.append("g")
            .call(d3.axisLeft(y));

        svg.append("path")
            .datum(data)
            .attr("fill", "none")
            .attr("stroke", "steelblue")
            .attr("stroke-width", 1.5)
            .attr("d", d3.line()
            .x(function(d) { return x(d.date) })
            .y(function(d) { return y(d.precipitation) }));

        svg.append("text")
            .attr("x", (width / 2))
            .attr("y", 0 - (margin.top / 2))
            .attr("text-anchor", "middle")
            .style("font-size", "16px")
            .text("Precipitation Time Series");

        svg.append("text")
            .attr("transform", "translate(" + (width / 2) + " ," + (height + 30) + ")")
            .style("text-anchor", "middle")
            .style("font-size", "16px")
            .text("Time");

        svg.append("text")
            .attr("transform", "rotate(-90)")
            .attr("x", -(height / 2))
            .attr("y", -30)
            .style("text-anchor", "middle")
            .style("font-size", "16px")
            .text("Precipitation");
    }
);
```



5. The observations are
    a. Wind speed mostly is within 2-3 mph and the distribution is right-skewed
    b. Sunny days and rainy days account for almost 90% of possibilities and the city seldom snows
    c. As for the precipitation, Seattle has low precipitation in the summer and a lot in the winter, especially after October
6. After visualization, we have the following

```html
<script src="https://d3js.org/d3.v4.js"></script>
<body>

    <p> Wind speed mostly is within 2-3 mph and the distribution is right-skewed </p>

    <p> Sunny days and rainy days account for almost 90% of possibilities and the city seldom snows </p>

    <p> As for the precipitation, Seattle has low precipitation in the summer and a lot in the winter, especially after October </p>

    <script>
        d3.selectAll("p")
            .data([12, 16, 20])
            .style("font-size", function (d) { return d + "px"; })
            .style("color", "blue")
    </script>

</body>
```

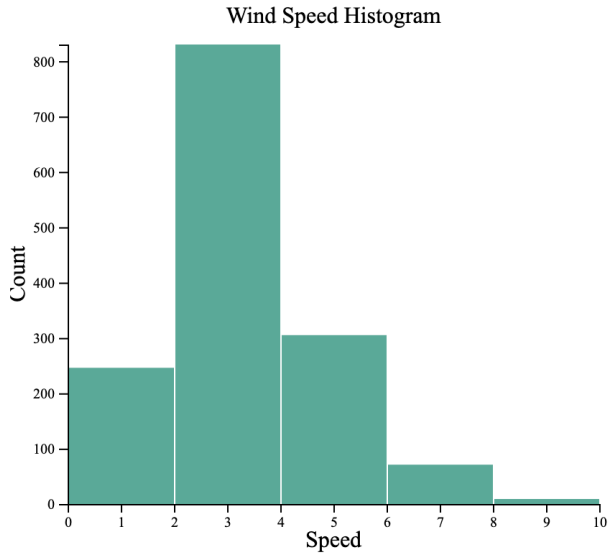
Wind speed mostly is within 2-3 mph and the distribution is right-skewed

Sunny days and rainy days account for almost 90% of possibilities and the city seldom snows

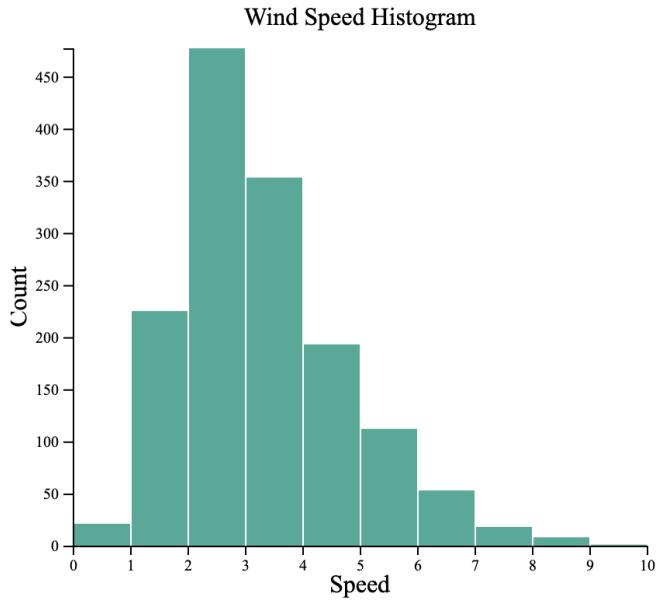
As for the precipitation, Seattle has low precipitation in the summer and a lot in the winter, especially after October

Problem 3

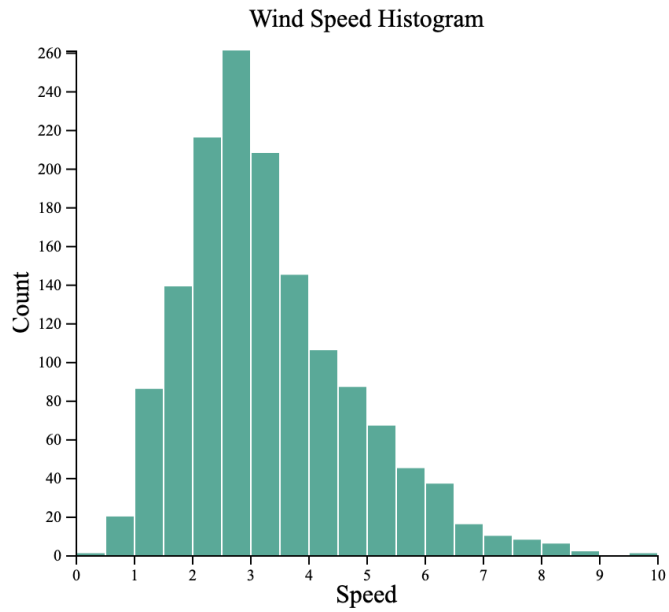
1. Following the sample code from d3 website, we re-plot the histogram



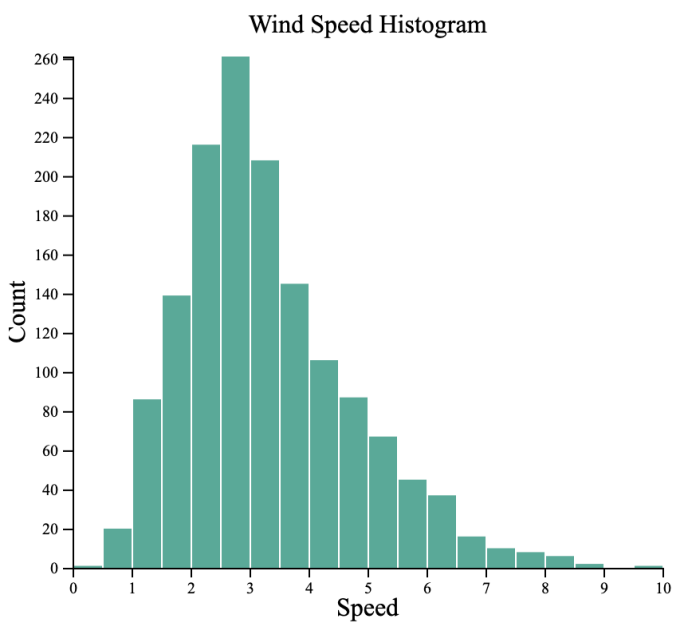
bins 5



bins 10



bins 15



bins 20

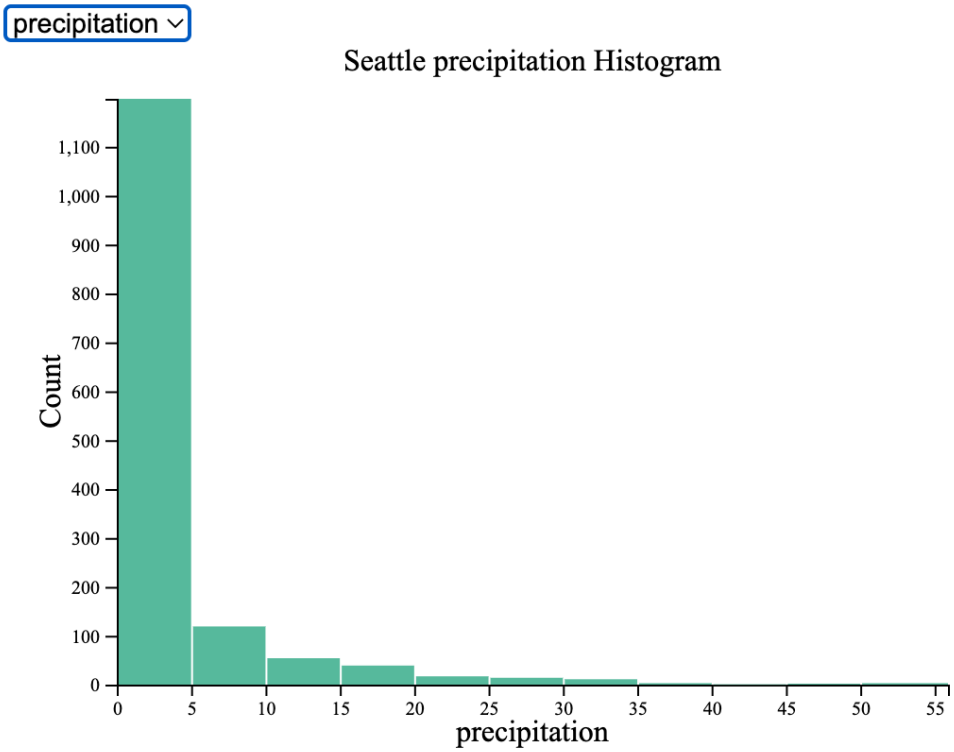
```html
<p>
    <label># bins</label>
    <input type="number" min="1" max="100" step="30" value="20" id="nBin">
</p>

<script>

    var margin = {top: 30, right: 50, bottom: 50, left: 40},
        width = 460 - margin.left - margin.right,
        height = 400 - margin.top - margin.bottom;

    var svg = d3.select("#my_dataviz")
        .append("svg")
        .attr("width", width + margin.left + margin.right)
        .attr("height", height + margin.top + margin.bottom)
        .append("g")
        .attr("transform",
            "translate(" + margin.left + "," + margin.top + ")");

    d3.csv("seattle-weather.csv", function(data) {
        var x = d3.scaleLinear()
            .domain([0, 10])
            .range([0, width]);
        svg.append("g")
            .attr("transform", "translate(0," + height + ")")
            .call(d3.axisBottom(x));

        var y = d3.scaleLinear()
            .range([height, 0]);
        var yAxis = svg.append("g")

        function update(nBin) {
            var histogram = d3.histogram()
                .value(function(d) { return d.wind; })
                .domain(x.domain())
                .thresholds(x.ticks(nBin));

            var bins = histogram(data);

            y.domain([0, d3.max(bins, function(d) { return d.length; })]);
            yAxis
                .transition()
                .duration(1000)
                .call(d3.axisLeft(y));

            var u = svg.selectAll("rect")
                .data(bins)

            u
            .enter()
            .append("rect") // Add a new rect for each new elements
            .merge(u) // get the already existing elements as well
            .transition() // and apply changes to all of them
            .duration(1000)
            .attr("x", 1)
            .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
            .attr("width", function(d) { return x(d.x1) - x(d.x0) -1 ; })
            .attr("height", function(d) { return height - y(d.length); })
            .style("fill", "#69b3a2")

            u
            .exit()
            .remove()
        }

        update(20)

        svg.append("text")
            .attr("x", (width / 2))
            .attr("y", 0 - (margin.top / 2))
            .attr("text-anchor", "middle")
            .style("font-size", "16px")
            .text("Wind Speed Histogram");
        svg.append("text")
            .attr("transform", "translate(" + (width / 2) + " ," + (height + 30) + ")")
            .style("text-anchor", "middle")
            .style("font-size", "16px")
            .text("Speed");
        svg.append("text")
            .attr("transform", "rotate(-90)")
            .attr("x", -(height / 2))
            .attr("y", -30)
            .style("text-anchor", "middle")
            .style("font-size", "16px")
            .text("Count");

        // Listen to the button -> update if user change it
        d3.select("#nBin").on("input", function() {
            update(+this.value);
        });


    });
</script>
```

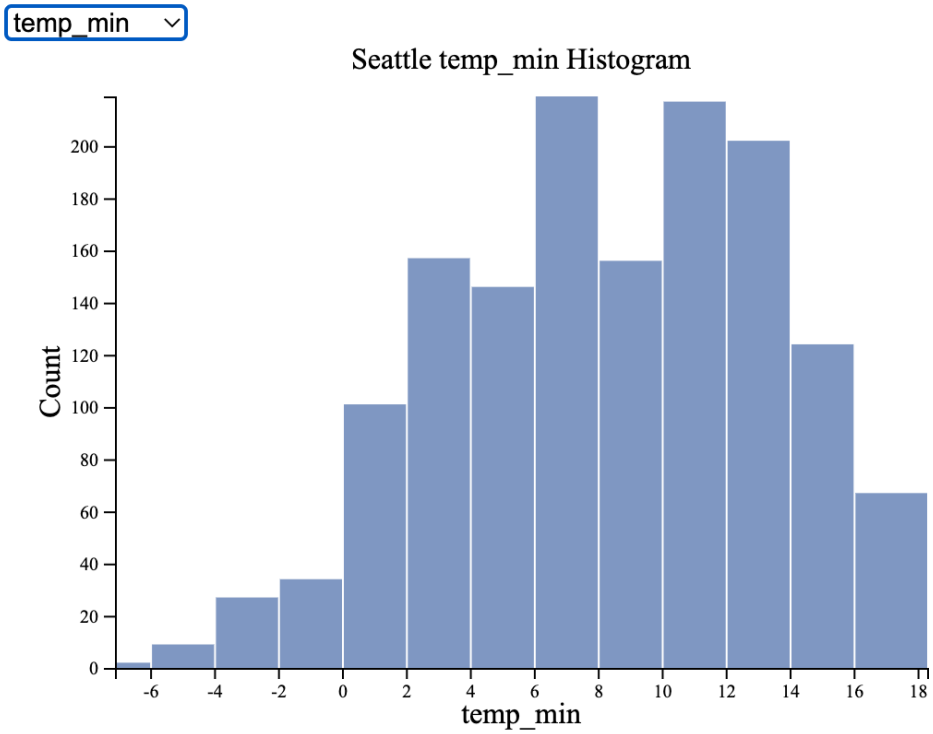
2. After adding the dropdown button, the page looks like this (Code is too long so I cut them into pieces)

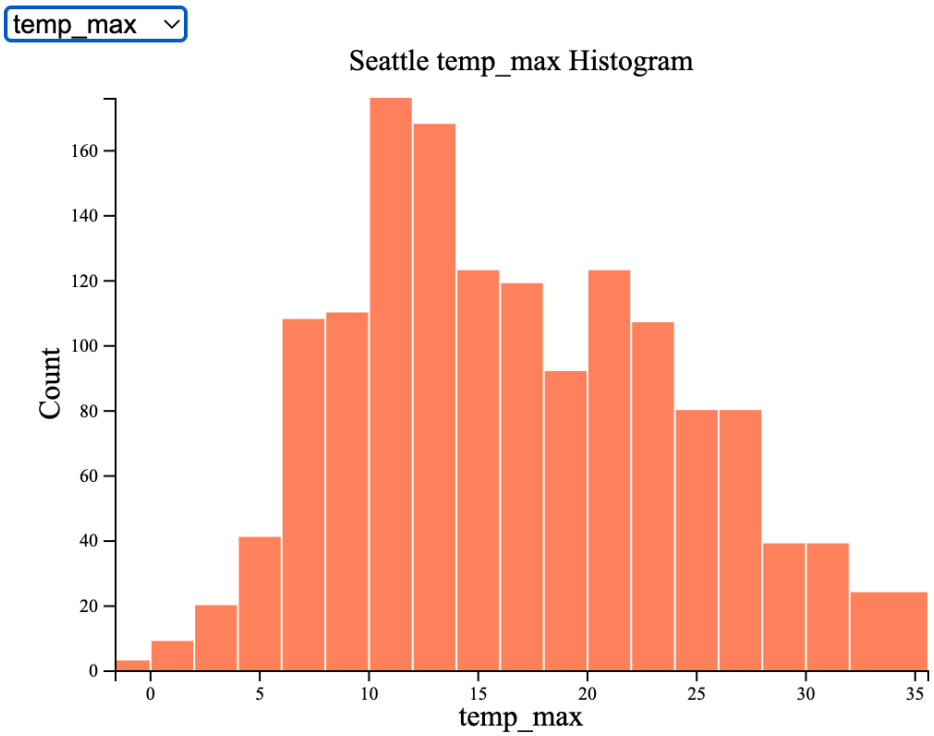
```
<script>

    // set the dimensions and margins of the graph
    var margin = {top: 30, right: 100, bottom: 60, left: 60},
        width = 600 - margin.left - margin.right,
        height = 400 - margin.top - margin.bottom;

    // append the svg object to the body of the page
    var svg = d3.select("#my_dataviz")
        .append("svg")
        .attr("width", width + margin.left + margin.right)
        .attr("height", height + margin.top + margin.bottom)
        .append("g")
        .attr("transform",
            "translate(" + margin.left + "," + margin.top + ")");

        // List of groups
        var allGroup = ["precipitation", "temp_max", "temp_min", "wind"]

        // A color scale: one color for each group
        var myColor = d3.scaleOrdinal()
            .domain(allGroup)
            .range(d3.schemeSet2);
```

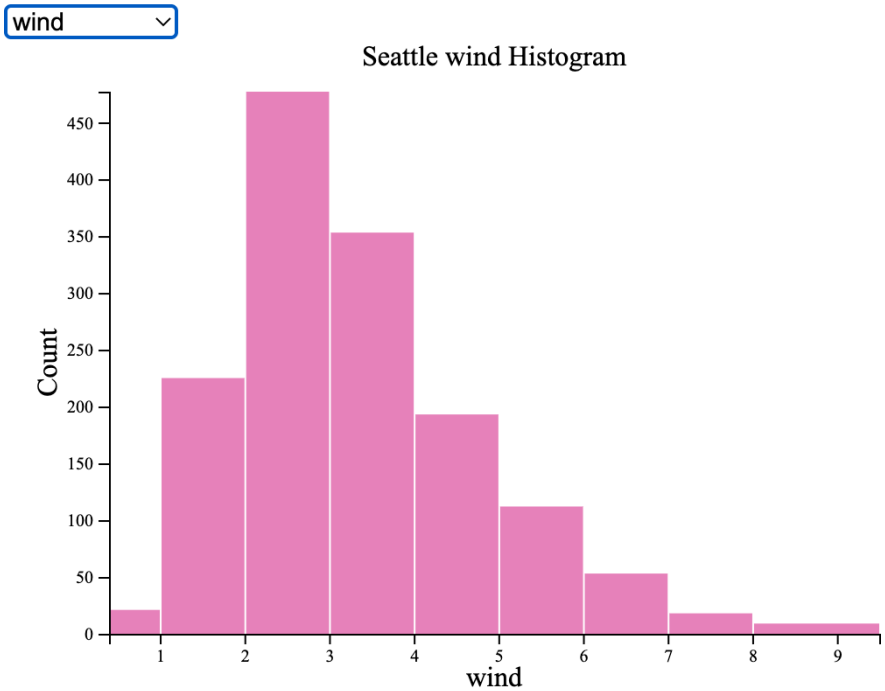
```javascript
// Read the data
d3.csv("seattle-weather.csv", function(data) {

    // add the options to the button
    d3.select("#selectButton")
        .selectAll('myOptions')
        .data(allGroup)
        .enter()
        .append('option')
        .text(function (d) { return d; }) // text showed in the menu
        .attr("value", function (d) { return d; }) // corresponding value returned by the button
        .property("selected", "wind")

    // Add X axis
    var x = d3.scaleLinear()
        .range([0, width])
    var xAxis = svg.append("g")
        .attr("transform", "translate(0," + height + ")")
        .call(d3.axisBottom(x))

    // Add Y axis
    var y = d3.scaleLinear()
        .range([height, 0]);
    var yAxis = svg.append("g")
        .call(d3.axisLeft(y));

    svg.append("text")
        .attr("transform", "rotate(-90)")
        .attr("x", -(height / 2))
        .attr("y", -30)
        .style("text-anchor", "middle")
        .style("font-size", "16px")
        .text("Count");

    var xlabel = svg.append("text")
        .attr("transform", "translate(" + (width / 2) + " ," + (height + 30) + ")")
        .style("text-anchor", "middle")
        .style("font-size", "16px")
        .text("Wind Speed");

    var title = svg.append("text")
        .attr("x", (width / 2))
        .attr("y", 0 - (margin.top / 2))
        .attr("text-anchor", "middle")
        .style("font-size", "16px")
        .text("Seattle Wind Speed Histogram");

```

```javascript
// A function that update the chart
        function update(selectedGroup) {
            x.domain([
                d3.min(data, d => parseFloat(d[selectedGroup])),
                d3.max(data, d => parseFloat(d[selectedGroup]))]
            )

            xAxis
            .transition()
            .duration(1000)
            .attr("transform", "translate(0," + height + ")")
            .call(d3.axisBottom(x))

            var histogram = d3.histogram()
                .value(d => d[selectedGroup])
                .domain(x.domain())
                // .thresholds(x.ticks(nbin))

            var bins = histogram(data)
            y.domain([0, d3.max(bins, d => d.length)])
            yAxis
                .transition()
                .duration(1000)
                .call(d3.axisLeft(y))

            var u = svg
                .selectAll("rect")
                .data(bins)

            // Give these new data to update histogram
            u
            .enter()
            .append("rect")
            .merge(u)
            .transition()
            .duration(1000)
            .attr("x", 1)
            .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
            .attr("width", function(d) { return x(d.x1) - x(d.x0) - 1 })
            .attr("height", function(d) { return y != null ? height - y(d.length) : 0 })
            .style("fill", myColor(selectedGroup));

            u
            .exit()
            .remove()

            xlabel
            .transition()
            .duration(1000)
            .text(selectedGroup)

            title
            .transition()
            .duration(1000)
            // .attr("x", (width / 2))
            // .attr("y", 0 - (margin.top / 2))
            // .attr("text-anchor", "middle")
            // .style("font-size", "16px")
            .text("Seattle " + selectedGroup + " Histogram");
        }

        update("wind")

        // When the button is changed, run the updateChart function
        d3.select("#selectButton").on("change", function() {
            update(this.value)
        })

    })
```


precipitation
Seattle precipitation Histogram
Count
1,100
1,000
900
800
700
600
500
400
300
200
100
0
0
5
10
15
20
25
30
35
40
45
50
55
precipitation


temp_min
Seattle temp_min Histogram
Count
200
180
160
140
120
100
80
60
40
20
0
-6
-4
-2
0
2
4
6
8
10
12
14
16
18
temp_min


temp_max
Seattle temp_max Histogram
Count
0
20
40
60
80
100
120
140
160
0
5
10
15
20
25
30
35
temp_max


wind
Seattle wind Histogram
Count
0
50
100
150
200
250
300
350
400
450
1
2
3
4
5
6
7
8
9
wind

Part II

Problem 1

1. The first 5 rows of the data set are

mpg	cylinders	displacement	horsepower	weight	acceleration	model-year
18	8	307	130	3504	12	70
15	8	350	165	3693	11.5	70
18	8	318	150	3436	11	70
16	8	304	150	3433	12	70
17	8	302	140	3449	10.5	70

```
<script>
            d3.csv("auto-mpg.csv", function(data) {
                // The table generation function
                function tabulate(data, columns) {
                    var table = d3.select("body").append("table")
                            .attr("style", "margin-left: 250px"),
                        thead = table.append("thead"),
                        tbody = table.append("tbody");

                    // append the header row
                    thead.append("tr")
                        .selectAll("th")
                        .data(columns)
                        .enter()
                        .append("th")
                        .text(function(column) { return column; });

                    // create a row for each object in the data
                    var rows = tbody.selectAll("tr")
                        .data(data)
                        .enter()
                        .append("tr");

                    // create a cell in each row for each column
                    var cells = rows.selectAll("td")
                        .data(function(row) {
                            return columns.map(function(column) {
                                return {column: column, value: row[column]};
                            });
                        })
                        .enter()
                        .append("td")
                        .attr("style", "font-family: Courier")
                        .html(function(d) { return d.value; });

                    return table;
                }

                // render the table
                var columns = [
                    "mpg",
                    "cylinders",
                    "displacement",
                    "horsepower",
                    "weight",
                    "acceleration",
                    "model-year"]
                var peopleTable = tabulate(data, columns);



            });
        </script>
```

2. The results are as follows

year	count
70	29
71	28
72	28
73	40
74	27
75	30
76	34
77	28
78	36
79	29
80	29
81	29
82	31

```
<script>
        d3.csv("auto-mpg.csv", function(data) {
            var dict = {}
            for (var i = 0; i < data.length; i++) {
                if (data[i]["model-year"] in dict ) {
                    dict[data[i]["model-year"]] += 1
                } else {
                    dict[data[i]["model-year"]] = 1
                }
            }

            var table = d3.select("table")
                        .attr("style", "margin-left: 250px"),
            thead = table.append("thead")

            // append the header row
            thead.append("tr")
                .selectAll("th")
                .data(["year", "count"])
                .enter()
                .append("th")
                .text(function(column) { return column; });

            for (var j = 0; j < Object.keys(dict).length; j++) {
                key = Object.keys(dict)[j]
                document.getElementById("target").innerHTML += "<tr><td>" + key + "</td><td>" + dict[key] + "</td></tr>"
            }
```

3. The results are as follows

year	total cylinders
70	196
71	156
72	163
73	255
74	142
75	168
76	192
77	153
78	193
79	169
80	120
81	134
82	130

```
<script>
            d3.csv("auto-mpg.csv", function(data) {
                var dict = {}
                for (var i = 0; i < data.length; i++) {
                    if (data[i]["model-year"] in dict ) {
                        dict[data[i]["model-year"]] += +data[i]["cylinders"]
                    } else {
                        dict[data[i]["model-year"]] = +data[i]["cylinders"]
                    }
                }

                var table = d3.select("table")
                            .attr("style", "margin-left: 250px"),
                thead = table.append("thead")

                // append the header row
                thead.append("tr")
                    .selectAll("th")
                    .data(["year", "total cylinders"])
                    .enter()
                    .append("th")
                    .text(function(column) { return column; });

                for (var j = 0; j < Object.keys(dict).length; j++) {
                    key = Object.keys(dict)[j]
                    document.getElementById("target").innerHTML += "<tr><td>" + key + "</td><td>" + dict[key] + "</td></tr>"
                }
```

4. The results are as follows

acceleration	count
8	1
9	1
10	4
11	7
12	10
13	12
14	16
15	14
16	16
17	14
18	8
19	12
21	5
11.5	7
10.5	1
8.5	2
9.5	2
15.5	21
14.5	23
20.5	3
17.5	4
12.5	8
13.5	15
18.5	5
19.5	6
23.5	1
16.5	13
16.9	4
14.9	7
17.7	3
15.3	3
13.9	2
12.8	3
15.4	4
17.6	4
22.2	2
22.1	1
14.2	3
17.4	2
16.2	4
17.8	2
12.2	2
16.4	9
13.6	2
15.7	4
13.2	6
21.9	1
16.7	3
12.1	1
14.8	3
18.6	4
16.8	2
13.7	2
11.1	1
11.4	2
18.2	5
15.8	7
15.9	2
14.1	1
21.5	1
14.4	5
19.4	3
19.2	3
17.2	2
18.7	2
15.1	2
13.4	2
11.2	1
14.7	5
16.6	3
17.3	5
15.2	3
14.3	2
20.1	2
24.8	1
11.3	1
12.9	2
18.8	1
18.1	1
17.9	1
21.7	1
23.7	1
19.9	1
21.8	1
13.8	2
12.6	2
16.1	1
20.7	1
18.3	1
20.4	1
19.6	2
17.1	1
15.6	1
24.6	1
11.6	1

```
<script>
            d3.csv("auto-mpg.csv", function(data) {
                var dict = {}
                for (var i = 0; i < data.length; i++) {
                    if (data[i]["acceleration"] in dict ) {
                        dict[data[i]["acceleration"]] += 1
                    } else {
                        dict[data[i]["acceleration"]] = 1
                    }
                }

                var table = d3.select("table")
                            .attr("style", "margin-left: 250px"),
                thead = table.append("thead")

                // append the header row
                thead.append("tr")
                    .selectAll("th")
                    .data(["acceleration", "count"])
                    .enter()
                    .append("th")
                    .text(function(column) { return column; });

                for (var j = 0; j < Object.keys(dict).length; j++) {
                    key = Object.keys(dict)[j]
                    document.getElementById("target").innerHTML += "<tr><td>" + key + "</td><td>" + dict[key] + "</td></tr>"
                }
```

Problem 2

The result is as follows
The first is about the function that generates the array we need.

```
<table id="target1", style="float:left"></table>
    <table id="target2", style="float:left"></table>
    <table id="target3", style="float:left"></table>

        <script src="http://d3js.org/d3.v4.js"></script>
        <script>
            var width = 1800,
                height = 600,
                margin = 100;

            var radius = 200;

            var svg = d3.select("#dataviz")
                .append("svg")
                .attr("width", width)
                .attr("height", height)
                .append("g")
                .attr("transform", "translate(" + width / 4 + "," + height / 2 + ")")

            // table function
            function tabulate(dict, header, id) {
                var table = d3.select("#" + id)
                .attr("style", "margin-left: 150px")
                // .append("table")
                thead = table.append("thead")

                // append the header row
                thead.append("tr")
                    .selectAll("th")
                    .data(header)
                    .enter()
                    .append("th")
                    .text(function(column) { return column; })

                for (var j = 0; j < Object.keys(dict).length; j++) {
                    key = Object.keys(dict)[j]
                    document.getElementById(id).innerHTML += "<tr><td>" + key + "</td><td>" + dict[key] + "</td></tr>"
                }
            }

```

Next is the run the function and display the tables.

```javascript
d3.csv("auto-mpg.csv", function(data) {
    // calculate number of cars in each year
    var year_count = {}
    for (var i = 0; i < data.length; i++) {
        if (data[i]["model-year"] in year_count) {
            year_count[data[i]["model-year"]] += 1
        } else {
            year_count[data[i]["model-year"]] = 1
        }
    }

    // calculate total cylinders in each year
    var cylinder = {}
    for (var i = 0; i < data.length; i++) {
        if (data[i]["model-year"] in cylinder) {
            cylinder[data[i]["model-year"]] += +data[i]["cylinders"]
        } else {
            cylinder[data[i]["model-year"]] = +data[i]["cylinders"]
        }
    }

    // calculate number of cars for each acceleration
    var acceleration = {}
    for (var i = 0; i < data.length; i++) {
        if (data[i]["acceleration"] in acceleration) {
            acceleration[data[i]["acceleration"]] += 1
        } else {
            acceleration[data[i]["acceleration"]] = 1
        }
    }

    // display tables
    tabulate(year_count, ["model-year", "count"], "target1")
    tabulate(cylinder, ["model-year", "total cylinders"], "target2")
    tabulate(acceleration, ["acceleration", "count"], "target3")
```

Then we draw the pie chart.

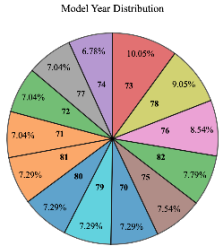
```javascript
var sum = Object.values(year_count).reduce((a, b) => a + b, 0)

// A color scale: one color for each group
var myColor = d3.scaleOrdinal()
    .domain(year_count)
    .range(d3.schemeCategory10);

var percentageFormat = d3.format(".2%");

var pie = d3.pie()
    .value(function(d) { return d.value; })
var data_ready = pie(d3.entries(year_count))

var arcGenerator = d3.arc()
    .innerRadius(0)
    .outerRadius(radius)

var arc = d3.arc()
    .outerRadius(radius + 80)
    .innerRadius(60)

svg.selectAll('mySlices')
.data(data_ready)
.enter()
.append('path')
    .attr('d', arcGenerator)
    .attr('fill', d => myColor(d.data.key))
    .attr("stroke", "black")
    .style("stroke-width", "2px")
    .style("opacity", 0.7)

svg.selectAll('mySlices')
    .data(data_ready)
    .enter()
    .append('text')
    .text(function(d) { return d.data.key })
    .attr("transform", function(d) { return "translate(" + arcGenerator.centroid(d) + ")" })
    .style("text-anchor", "middle")
    .style("font-size", 16)
    .attr("font-weight", 900)

svg.selectAll('mySlices')
    .data(data_ready)
    .enter()
    .append("text")
    .attr("transform", function(d) {
        return "translate(" + arc.centroid(d) + ")";
    })
    .attr("dy", ".35em")
    .style("text-anchor", "middle")
    .text(function(d) {
        return percentageFormat(d.data.value / sum);
    });

svg.append("text")
    .attr("x", 0)
    .attr("y", -margin - 140)
    .attr("text-anchor", "middle")
    .style("font-size", "20px")
    .text("Model Year Distribution");
```

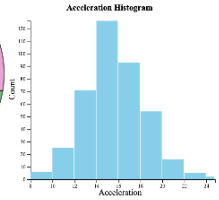
Then we draw the histogram.

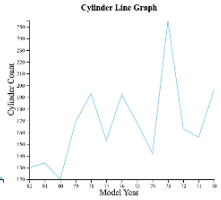
```javascript
var xmin = d3.min(Object.keys(acceleration), d => parseFloat(d))
                    xmax = d3.max(Object.keys(acceleration), d => parseFloat(d))

                var x = d3.scaleLinear()
                    .domain([xmin, xmax])
                    .range([250, 600]);
                svg.append("g")
                    .attr("transform", "translate(0, " + 200 + ")")
                    .call(d3.axisBottom(x));

                var histogram = d3.histogram()
                    .value(function(d) { return d.acceleration; })
                    .domain(x.domain())
                    .thresholds(x.ticks(10));

                var bins = histogram(data)

                var y = d3.scaleLinear()
                .domain([0, d3.max(bins, function(d) { return d.length; })])
                .range([height/3, -100])
                svg.append("g")
                    .attr("transform", "translate(" + 250 + ")")
                    .call(d3.axisLeft(y));

                svg.selectAll("rect")
                    .data(bins)
                    .enter()
                    .append("rect")
                    .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
                    .attr("width", function(d) { return x(d.x1) - x(d.x0) - 1; })
                    .attr("height", function(d) { return (height/3 - y(d.length)); })
                    .style("fill", "skyblue")

                svg.append("text")
                    .attr("x", 400)
                    .attr("y", -120)
                    .attr("text-anchor", "middle")
                    .style("font-size", "16px")
                    .style("font-weight", 600)
                    .text("Acceleration Histogram");

                svg.append("text")
                    .attr("x", 420)
                    .attr("y", 230)
                    .style("text-anchor", "middle")
                    .style("font-size", "16px")
                    .text("Acceleration");

                svg.append("text")
                    .attr("transform", "rotate(-90)")
                    .attr("x", -30)
                    .attr("y", 220)
                    .style("text-anchor", "middle")
                    .style("font-size", "16px")
                    .text("Count");
```

Then we draw the line graph.

```javascript
var x_line = d3.scaleLinear()
                    .domain([
                        d3.max(Object.keys(cylinder), d => parseFloat(d)),
                        d3.min(Object.keys(cylinder), d => parseFloat(d))])
                    .range([650, 1000]);
                    svg.append("g")
                    .attr("transform", "translate(0 " + 200 + ")")
                    .call(d3.axisBottom(x_line));

                var y_line = d3.scaleLinear()
                    .domain([
                        d3.min(Object.values(cylinder), d => d),
                        d3.max(Object.values(cylinder), d => d)])
                    .range([height/3, -100]);

                svg.append("g")
                    .attr("transform", "translate(" + 650 + ")")
                    .call(d3.axisLeft(y_line));

                console.log(Object.keys(cylinder))

                svg.append("path")
                    .datum(d3.entries(cylinder))
                    .attr("fill", "null")
                    .attr("stroke", "skyblue")
                    .attr("stroke-width", 1.5)
                    .attr("d", d3.area()
                    .x(d => x_line(d.key))
                    .y(d => y_line(d.value)))


                svg.append("text")
                    .attr("x", 820)
                    .attr("y", -120)
                    .attr("text-anchor", "middle")
                    .style("font-size", "16px")
                    .style("font-weight", 600)
                    .text("Cylinder Line Graph");

                svg.append("text")
                    .attr("transform", "rotate(-90)")
                    .attr("x", -30)
                    .attr("y", 620)
                    .style("text-anchor", "middle")
                    .style("font-size", "16px")
                    .text("Cylinder Count");

                svg.append("text")
                    .attr("x", 820)
                    .attr("y", 230)
                    .style("text-anchor", "middle")
                    .style("font-size", "16px")
                    .text("Model Year");
```







model-year	count
70	29
71	28
72	28
73	40
74	27
75	30
76	34
77	28
78	36
79	29
80	29
81	29
82	31

model-year	total cylinders
70	196
71	156
72	163
73	255
74	142
75	168
76	192
77	153
78	193
79	169
80	120
81	134
82	130

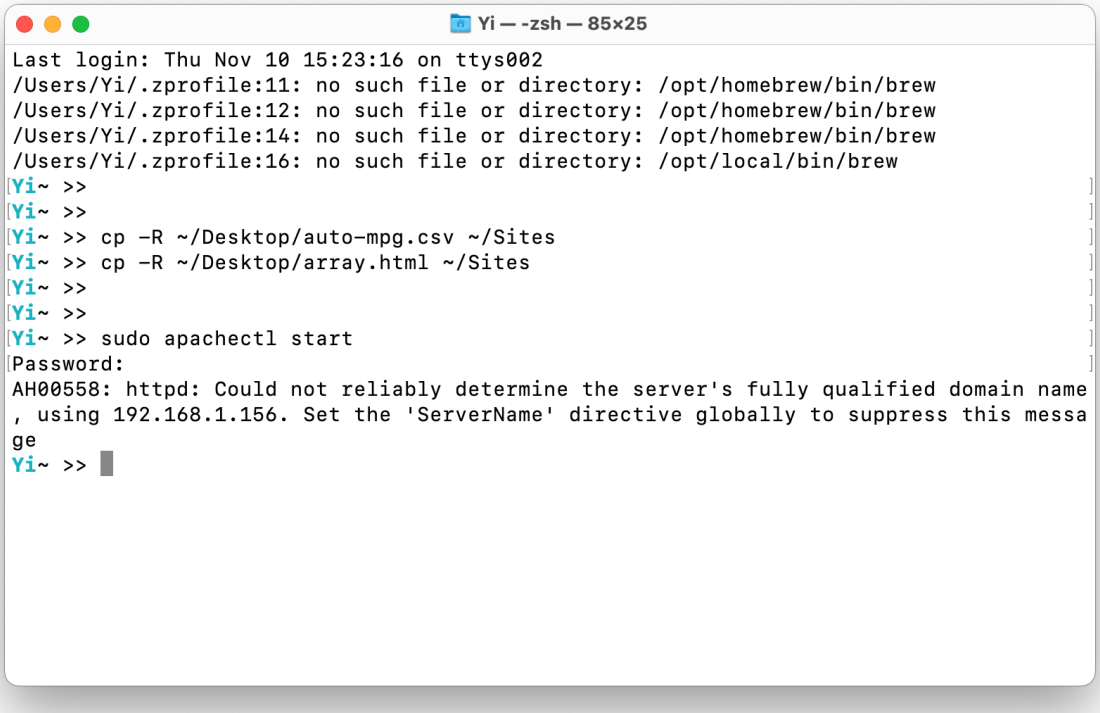
acceleration	count
8	1
9	1
10	4
11	7
12	10
13	12
14	16
15	14
16	16
17	14
18	8
19	12
21	5
11.5	7
10.5	1
8.5	2
9.5	2
15.5	21
14.5	23
20.5	3
17.5	4
12.5	8
13.5	15
18.5	5
19.5	6
23.5	1
16.5	13
16.9	4
14.9	7
17.7	3
15.3	3
13.9	2
12.8	3
15.4	4
17.6	4
22.2	2
22.1	1
14.2	3
17.4	2
16.2	4
17.8	2
12.2	2
16.4	9
13.6	2
15.7	4
13.2	6
21.9	1
16.7	3
12.1	1
14.8	3
18.6	4
16.8	2
13.7	2
11.1	1
11.4	2
18.2	5
15.8	7
15.9	2
14.1	1
21.5	1
14.4	5
19.4	3
19.2	3
17.2	2
18.7	2
15.1	2
13.4	2
11.2	1
14.7	5
16.6	3
17.3	5
15.2	3
14.3	2
20.1	2
24.8	1
11.3	1
12.9	2
18.8	1
18.1	1
17.9	1
21.7	1
23.7	1
19.9	1
21.8	1
13.8	2
12.6	2
16.1	1
20.7	1
18.3	1
20.4	1
19.6	2
17.1	1
15.6	1
24.6	1
11.6	1

Problem 3

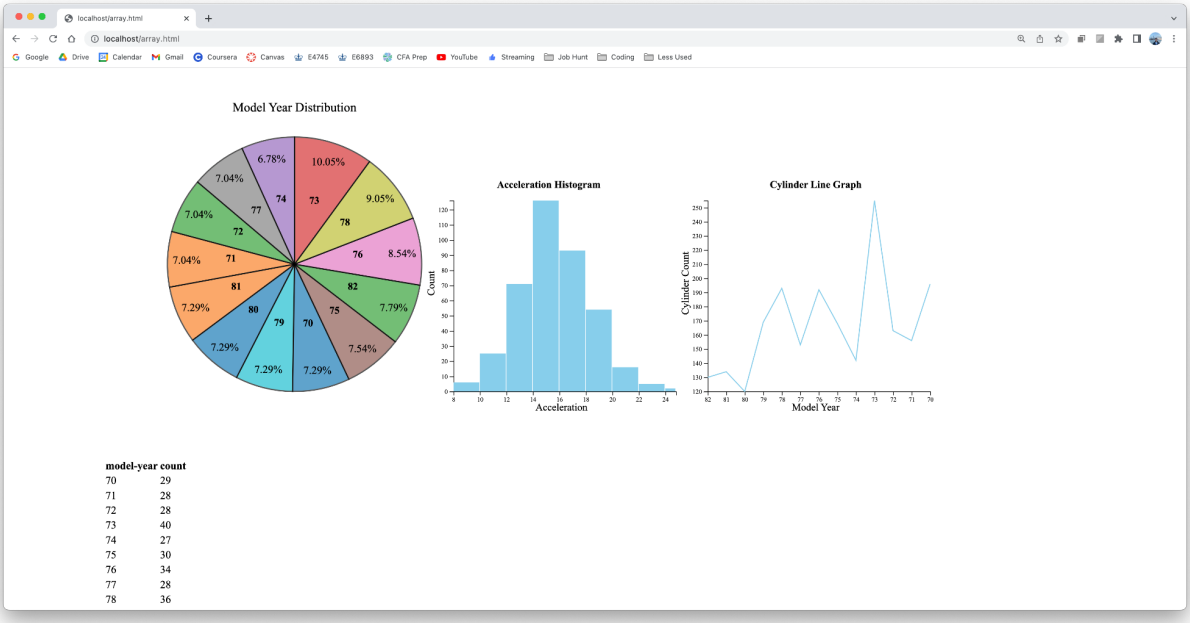
The screenshots are as follows

```
Terminal Saved Output.txt — Edited
Last login: Thu Nov 10 11:24:04 on console
Yi~ >> xcode-select --install
xcode-select: error: command line tools are already installed, use "Software Update" in System Settings to install updates
Yi~ >> /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
==> Checking for `sudo` access (which may request your password)...
Password:
==> This script will install:
/usr/local/bin/brew
/usr/local/share/doc/homebrew
/usr/local/share/man/man1/brew.1
/usr/local/share/zsh/site-functions/_brew
/usr/local/etc/bash_completion.d/brew
/usr/local/Homebrew

Press RETURN/ENTER to continue or any other key to abort:
==> /usr/bin/sudo /usr/sbin/chown -R Yi:admin /usr/local/Homebrew
==> Downloading and installing Homebrew...
remote: Enumerating objects: 1944, done.
remote: Counting objects: 100% (867/867), done.
remote: Total 1944 (delta 867), reused 867 (delta 867), pack-reused 1077
Receiving objects: 100% (1944/1944), 838.62 KiB | 20.45 MiB/s, done.
Resolving deltas: 100% (1388/1388), completed with 329 local objects.
From https://github.com/Homebrew/brew
 * [new branch]          dependabot/bundler/Library/Homebrew/rubocop-1.38.0 -> origin/dependabot/bundler/Library/Homebrew/rubocop-1.38.0
 * [new branch]          issue-templates          -> origin/issue-templates
 * [new branch]          less-agressive-core-tap  -> origin/less-agressive-core-tap
   e23d7b636..72f7880e2  master                   -> origin/master
 * [new tag]             3.6.10                   -> 3.6.10
 * [new tag]             3.6.5                    -> 3.6.5
 * [new tag]             3.6.6                    -> 3.6.6
 * [new tag]             3.6.7                    -> 3.6.7
 * [new tag]             3.6.8                    -> 3.6.8
 * [new tag]             3.6.9                    -> 3.6.9
HEAD is now at 72f7880e2 Merge pull request #14129 from Homebrew/style_more_shell_scripts

Updated 3 taps (microsoft/mssql-release, homebrew/core and homebrew/cask).
==> Installation successful!

==> Homebrew has enabled anonymous aggregate formulae and cask analytics.
Read the analytics documentation (and how to opt-out) here:
  https://docs.brew.sh/Analytics
No analytics data has been sent yet (nor will any be during this install run).

==> Homebrew is run entirely by unpaid volunteers. Please consider donating:
  https://github.com/Homebrew/brew#donations

==> Next steps:
- Run brew help to get started
- Further documentation:
    https://docs.brew.sh

Yi~ >>
Yi~ >>
Yi~ >>
Yi~ >>
Yi~ >>
```

After installing Apache, we modify the html file as suggested and move the problem 2 html file and the dataset to Sites folder and deploy them.



```
Yi — -zsh — 85×25
Last login: Thu Nov 10 15:23:16 on ttys002
/Users/Yi/.zprofile:11: no such file or directory: /opt/homebrew/bin/brew
/Users/Yi/.zprofile:12: no such file or directory: /opt/homebrew/bin/brew
/Users/Yi/.zprofile:14: no such file or directory: /opt/homebrew/bin/brew
/Users/Yi/.zprofile:16: no such file or directory: /opt/local/bin/brew
Yi~ >>
Yi~ >>
Yi~ >> cp -R ~/Desktop/auto-mpg.csv ~/Sites
Yi~ >> cp -R ~/Desktop/array.html ~/Sites
Yi~ >>
Yi~ >>
Yi~ >> sudo apachectl start
Password:
AH00558: httpd: Could not reliably determine the server's fully qualified domain name
, using 192.168.1.156. Set the 'ServerName' directive globally to suppress this messa
ge
Yi~ >>
```



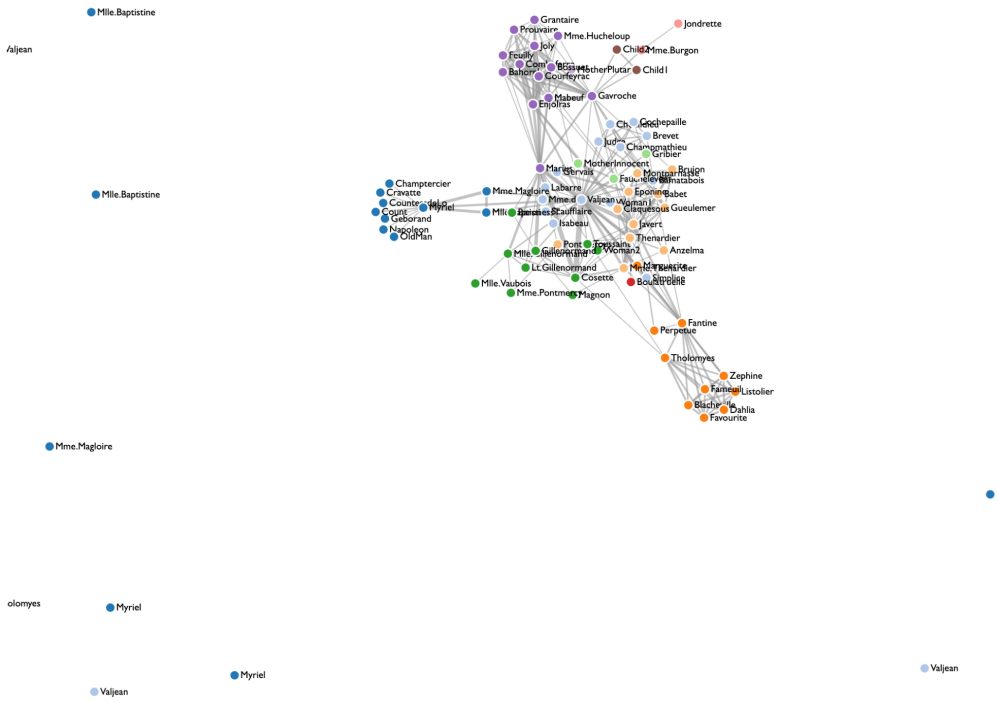
model-year	count
70	29
71	28
72	28
73	40
74	27
75	30
76	34
77	28
78	36

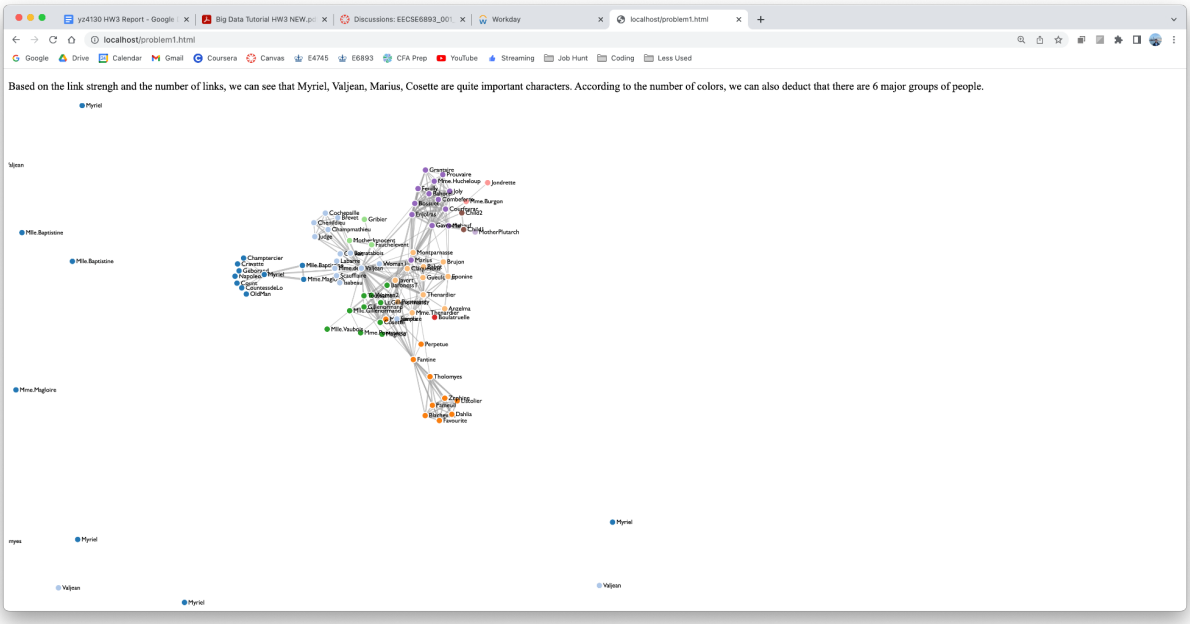
Part III

Problem 1

The codes and the graph are listed below

```html
<!DOCTYPE html>
<meta charset="utf-8">

<!-- Load d3.js you can use ither version if you want -->
<script src="https://d3js.org/d3.v4.js"></script>

<style>
    .links line {
        stroke: #999;
        stroke-opacity: 0.6;
    }

    .nodes circle {
        stroke: #fff;
        stroke-width: 1.5px;
    }

    text {
        font-family: 'Gill Sans', 'Gill Sans MT', Calibri, 'Trebuchet MS', sans-serif;
        font-size: 10px;
    }
    </style>


<svg width="1000" height="800"></svg>


<script>
    var svg = d3.select("svg"),
        width = +svg.attr("width"),
        height = +svg.attr("height")

    var simulation = d3.forceSimulation()
        .force("link", d3.forceLink().id(function(d) { return d.id; }))
        .force("charge", d3.forceManyBody())
        .force("center", d3.forceCenter(width / 2, height / 2));

    var color = d3.scaleOrdinal(d3.schemeCategory20)

    // var fisheye = d3.fisheye.circular()
    //  .radius(200)
    //  .distortion(2)

    d3.csv("https://gist.githubusercontent.com/timelyportfolio/5049980/raw/66a239b4aa325c05c7a19bd96bf093632591e809/les_mis.csv",
    function(data) {

        let graph = {"nodes": [], "links": []}

        // data preprocessing
        data.forEach(function(line) {
            column = Object.entries(line)
            graph.nodes.push({
                'id': column[0][1],
                'group': column[3][1]
            })
            graph.nodes.push({
                'id': column[1][1],
                'group': column[4][1]
            })
            graph.links.push({
                'source': column[0][1],
                'target': column[1][1],
                'value': column[2][1]
            })
        })

        var link = svg.append("g")
            .attr("class", "links")
            .selectAll("line")
            .data(graph.links)
            .enter()
            .append("line")
            .attr("stroke-width", d => Math.sqrt(d.value))

        var node = svg.append("g")
            .attr("class", "nodes")
            .selectAll("g")
            .data(graph.nodes)
            .enter().append("g")

        var circles = node.append("circle")
            .attr("r", 5)
            .attr("fill", function(d) { return color(d.group); });

        // Create a drag handler and append it to the node object instead
        var drag_handler = d3.drag()
            .on("start", dragstarted)
            .on("drag", dragged)
            .on("end", dragended);

        drag_handler(node);

        var lables = node.append("text")
            .text(function(d) {
                return d.id;
            })
            .attr('x', 6)
            .attr('y', 3);

        node.append("title")
            .text(function(d) { return d.id; });

        simulation
            .nodes(graph.nodes)
            .on("tick", ticked)

        simulation.force("link").links(graph.links)

        function ticked() {
            link
                .attr("x1", d => d.source.x)
                .attr("y1", d => d.source.y)
                .attr("x2", d => d.target.x)
                .attr("y2", d => d.target.y)

            node
                .attr("transform", function(d) {
                    return "translate(" + d.x + "," + d.y + ")";
                })
        }

        function dragstarted(d) {
            if (!d3.event.active) simulation.alphaTarget(0.3).restart();
            d.fx = d.x;
            d.fy = d.y;
        }

        function dragged(d) {
            d.fx = d3.event.x;
            d.fy = d3.event.y;
        }

        function dragended(d) {
            if (!d3.event.active) simulation.alphaTarget(0);
            d.fx = null;
            d.fy = null;
        }

        // svg.on("mousemove", function() {
        //  fisheye.focus(d3.mouse(this));

        //  node.each(function(d) { d.fisheye = fisheye(d); })
        //      .attr("cx", function(d) { return d.fisheye.x; })
        //      .attr("cy", function(d) { return d.fisheye.y; })
        //      .attr("r", function(d) { return d.fisheye.z * 4.5; });

        //  link.attr("x1", function(d) { return d.source.fisheye.x; })
        //      .attr("y1", function(d) { return d.source.fisheye.y; })
        //      .attr("x2", function(d) { return d.target.fisheye.x; })
        //      .attr("y2", function(d) { return d.target.fisheye.y; });
        // });


    });
</script>
```



Problem 2

Similarly, we can deploy our graph on the Apache web server and the result looks like the following