

EECS 6893: Big Data Analytics

HW3

Yutao Zhou UNI: yz4359

Part I

Problem 1

1.1.1

In SVG Coordinate origin is the top left of the screen and the x value increase as we move right and the y value increase as we move down the screen. In Mathematical/Graph Coordinate Space the origin is the center, the x value increases as we move right, and the y value increases as we move up.

1.1.2

Enter and exit are two functions in d3.js. Enter is used to add data into DOM element when data have more entry than DOM element. For example, when DOM have element '1','2' and our data have element '1','2','3', we could use enter to add element '3' into DOM element.

Exit is used to remove DOM element when data is shorter than DOM element. For example, when DOM have element '1','2','3' and our data have element '1','2', we could use exit to remove element '3' from DOM element.

1.1.3

Transform is an attribute in SVG that contains a bunch of transformation that we could apply to an element. In other words, transform apply the same transformation to all of the points of an element. Translate is a simpecific transformation in the Transform attribute. Translate is used to move an element by x and y, where x is distance move horizontally and y is distance move verticallys.

1.1.4

The return value of this anonymous function is a = [5, 6, 7, 8, 9]

1.1.5

The output of the 2 code snippets will be different. In the snippet II the enter and append will add data into the body with a space between each of them(the entire data list will be print out where there will be a space between each elements). Where in snippet I the first element will been used to replace the body(only the first element will been print out).

Problem 2

```

1  <!DOCTYPE html>
2  <meta charset="utf-8">
3  <html>
4      <head>
5          <script src="https://d3js.org/d3.v4.js"></script>
6          <link rel="stylesheet" href="1.2.5.css">
7      </head>
8
9      <body>
10
11         <script>
12
13             d3.csv("https://raw.githubusercontent.com/vega/vega/main/docs/data/seattle-weather.csv", function(data) {
14                 time = (data.map(function(d) { return d3.timeParse("%Y-%m-%d")(d.date)}))
15
16                 ##### Start of 1.2.1 histogram with bin 10 #####
17
18                 var marginHis10 = {top: 20, right: 50, bottom: 50, left: 60},
19                 widthHis10 = 460 - marginHis10.left - marginHis10.right,
20                 heightHis10 = 400 - marginHis10.top - marginHis10.bottom;
21
22                 var svg = d3.select("body")
23                     .append("svg")
24                         .attr("width", widthHis10 + marginHis10.left + marginHis10.right)
25                         .attr("height", heightHis10 + marginHis10.top + marginHis10.bottom)
26                         .append("g")
27                             .attr("transform",
28                                 "translate(" + marginHis10.left + "," + marginHis10.top + ")");
29
30                     svg.append("text")
31                         .attr("x", widthHis10 / 2)
32                         .attr("y", -8)
33                         .attr("text-anchor", "middle")
34                         .style("font-size", "16px")
35                         .style("font-weight", "bold")
36                         .text("Histogram With Bin Of 10");
37
38                     svg.append("circle").attr("cx",widthHis10 - 110).attr("cy",10).attr("r", 6).style("fill", "#69b3a2")
39                     svg.append("text").attr("x", widthHis10 - 100).attr("y", 10).text("Wind Speed").style("font-size", "15px").attr("alignment-baseline","middle")
40
41                     var x = d3.scaleLinear()
42                         .domain([0,10])
43                         .range([0, widthHis10]);
44                     svg.append("g")
45                         .attr("transform", "translate(0," + heightHis10 + ")")
46                         .call(d3.axisBottom(x));
47                     svg.append("text")
48                         .attr("transform",
49                             "translate(" + (widthHis10/2) + ", " +
50                                 (heightHis10 + marginHis10.top + 20) + ")")
51                         .style("text-anchor", "middle")
52                         .text("Wind Speed");
53
54                     var histogram = d3.histogram()
55                         .value(function(d) { return d.wind; })
56                         .domain(x.domain());

```

```

57      .thresholds(x.ticks(10));
58
59  var bins = histogram(data);
60
61  var y = d3.scaleLinear()
62    .domain([0, d3.max(bins, function(d) { return d.length; })])
63    .range([heightHis10, 0]);
64
65  svg.append("g")
66    .text("Number of days")
67    .call(d3.axisLeft(y));
68
69  svg.append("text")
70    .attr("transform", "rotate(-90)")
71    .attr("y", 0 - marginHis10.left)
72    .attr("x", 0 - (heightHis10 / 2))
73    .attr("dy", "1em")
74    .style("text-anchor", "middle")
75    .text("Number of days");
76
77  svg.selectAll("rect")
78    .data(bins)
79    .enter()
80    .append("rect")
81      .attr("x", 1)
82      .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
83      .attr("width", function(d) { return x(d.x1) - x(d.x0) - 1; })
84      .attr("height", function(d) { return heightHis10 - y(d.length); })
85      .style("fill", "#69b3a2");
86
87 //#### Start of 1.2.2 histogram with bin 25 #####
88
89 var marginHis25 = {top: 20, right: 50, bottom: 50, left: 60},
90 widthHis25 = 460 - marginHis25.left - marginHis25.right,
91 heightHis25 = 400 - marginHis25.top - marginHis25.bottom;
92
93 var svg = d3.select("body")
94   .append("svg")
95   .attr("width", widthHis25 + marginHis25.left + marginHis25.right)
96   .attr("height", heightHis25 + marginHis25.top + marginHis25.bottom)
97   .append("g")
98     .attr("transform",
99       "translate(" + marginHis25.left + "," + marginHis25.top + ")");
100
101 svg.append("text")
102   .attr("x", widthHis25 / 2)
103   .attr("y", -8)
104   .attr("text-anchor", "middle")
105   .style("font-size", "16px")
106   .style("font-weight", "bold")
107   .text("Histogram With Bin Of 25");
108
109 svg.append("circle").attr("cx",widthHis10 - 110).attr("cy",10).attr("r", 6).style("fill", "#69b3a2");
110
111 var x = d3.scaleLinear()
112   .domain([0,10])
113   .range([0, widthHis25]);

```

```

113     svg.append("g")
114       .attr("transform", "translate(0," + heightHis25 + ")")
115       .call(d3.axisBottom(x));
116   svg.append("text")
117     .attr("transform",
118       "translate(" + (widthHis25/2) + " , " +
119       (heightHis25 + marginHis25.top + 20) + ")");
120     .style("text-anchor", "middle")
121     .text("Wind Speed");
122
123   var histogram = d3.histogram()
124     .value(function(d) { return d.wind; })
125     .domain(x.domain())
126     .thresholds(x.ticks(25));
127
128   var bins = histogram(data);
129
130   var y = d3.scaleLinear()
131     .domain([0, d3.max(bins, function(d) { return d.length; })])
132     .range([heightHis25, 0]);
133   svg.append("g")
134     .text("Number of days")
135     .call(d3.axisLeft(y));
136   svg.append("text")
137     .attr("transform", "rotate(-90)")
138     .attr("y", 0 - marginHis25.left)
139     .attr("x", 0 - (heightHis25 / 2))
140     .attr("dy", "1em")
141     .style("text-anchor", "middle")
142     .text("Number of days");
143
144   svg.selectAll("rect")
145     .data(bins)
146     .enter()
147     .append("rect")
148     .attr("x", 1)
149     .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
150     .attr("width", function(d) { return x(d.x1) - x(d.x0) -1 ; })
151     .attr("height", function(d) { return heightHis25 - y(d.length); })
152     .style("fill", "#69b3a2");
153
154 //#### Start of 1.2.3 pie chart ####
155
156 var widthPie = 400,
157   heightPie = 400,
158   radius = (Math.min(widthPie, heightPie)-50) / 3;
159
160 var color = d3.scaleOrdinal(d3.schemeCategory10);
161
162 var arc = d3.arc()
163   .outerRadius(radius - 10)
164   .innerRadius(0);
165
166 var pie = d3.pie()
167   .sort(null)
168   .value(function(d) {
169     return d.value;

```

```

170     });
171
172     var svg = d3.select("body")
173         .append("svg")
174         .attr("width", widthPie)
175         .attr("height", heightPie)
176         .append("g")
177         .attr("transform", "translate(" + widthPie / 2 + "," + heightPie / 2 + ")");
178
179     svg.append("text")
180         .attr("x", 0)
181         .attr("y", -widthPie / 2 + 30)
182         .attr("text-anchor", "middle")
183         .style("font-size", "16px")
184         .style("font-weight", "bold")
185         .text("Pie Chart of Weather in Seattle");
186
187     //reduce by key; key : weather; value: count
188     var dataPie = d3.nest()
189         .key(function(d) {
190             return d.weather;
191         })
192         .rollup(function(d) {
193             return d3.sum(d, function(g) {
194                 return 1;
195             });
196         })
197         .entries(data);
198
199     //total count for percentage calculation
200     var total = d3.sum(dataPie, function(d) {
201         return d.value;
202     });
203
204     //percentage calculation
205     dataPie.forEach(function(d) {
206         d.percentage = (d.value / total * 100).toFixed(2);
207     });
208
209     var counter = 0
210     var c = ["rgb(31, 119, 180)", "rgb(255, 127, 14)", "rgb(44, 160, 44)", "rgb(214, 39, 40)", "rgb(148, 103, 189)"]
211     var name = ["drizzle", "rain", "sun", "snow", "fog"]
212     dataPie.forEach(function(d) {
213         svg.append("circle").attr("cx", widthPie / 2 - 110).attr("cy", -widthPie / 2 + 50 + counter * 15).attr("r", 6).style("fill", c[counter])
214         svg.append("text").attr("x", widthPie / 2 - 100).attr("y", -widthPie / 2 + 50 + counter * 15).text(name[counter]).style("font-size", "15px").attr("alignment-baseline", "middle")
215         counter = counter + 1
216     });
217
218     var g = svg.selectAll(".arc")
219         .data( pie(dataPie) )
220         .enter()
221         .append("g")
222         .attr("class", "arc");
223
224     g.append("path")
225         .attr("d", arc)

```

```

225   .style("fill", function(d) {
226     return color(d.data.key);
227   });
228
229   g.append("text")
230     .attr("transform", function(d) {
231       var _d = arc.centroid(d);
232       _d[0] *= 2.7;
233       _d[1] *= 2.7;
234       return "translate(" + _d + ")"
235     })
236     .attr("dy", ".35em")
237     .style("text-anchor", "middle")
238     .text(function(d) {
239       return d.data.key;
240     });
241
242   g.append("text")
243     .attr("transform", function(d) {
244       var _d = arc.centroid(d);
245       _d[0] *= 2.7;
246       _d[1] *= 2.7;
247       return "translate(" + _d + ")"
248     })
249     .attr("dy", "2.0em")
250     .style("text-anchor", "middle")
251     .text(function(d) {
252       return d.data.percentage + "%";
253     });
254
255
256 //#### Start of 1.2.4 line graph ####
257
258 var marginLine = {top: 20, right: 50, bottom: 50, left: 60},
259 widthLine = 600 - marginLine.left - marginLine.right,
260 heightLine = 337.5 - marginLine.top - marginLine.bottom;
261
262 var svg = d3.select("body")
263   .append("svg")
264   .attr("width", widthLine + marginLine.left + marginLine.right)
265   .attr("height", heightLine + marginLine.top + marginLine.bottom)
266   .append("g")
267   .attr("transform",
268     "translate(" + marginLine.left + "," + marginLine.top + ")");
269
270 svg.append("text")
271   .attr("x", widthLine / 2)
272   .attr("y", -8)
273   .attr("text-anchor", "middle")
274   .style("font-size", "16px")
275   .style("font-weight", "bold")
276   .text("Line Graph of Precipitation vs Date");
277
278
279 svg.append("line").attr("x1", widthHis10 - 110).attr("y1", 10).attr("x2", widthHis10 - 80).attr("y2", 10).style("stroke", "steelblue").style("stroke-width", 3)
280 svg.append("text").attr("x", widthHis10 - 70).attr("y", 10).text("Precipitation").style("font-size", "15px").attr("
```

```

280     alignment-baseline", "middle")
281
282     var x = d3.scaleTime()
283         .domain(d3.extent(data, function(d) { return d3.timeParse("%Y-%m-%d")(d.date); }))
284         .range([ 0, widthLine ]);
285     svg.append("g")
286         .attr("transform", "translate(0," + heightLine + ")");
287         .call(d3.axisBottom(x));
288
289     svg.append("text")
290         .attr("transform",
291             "translate(" + (widthLine/2) + ", " +
292             (heightLine + marginLine.top + 20) + ")");
293         .style("text-anchor", "middle")
294         .text("Date");
295
296     var y = d3.scaleLinear()
297         .domain([0, d3.max(data, function(d) { return +d.precipitation; })])
298         .range([ heightLine, 0 ]);
299
300     svg.append("g")
301         .call(d3.axisLeft(y));
302
303     svg.append("text")
304         .attr("transform", "rotate(-90)");
305         .attr("y", 0 - marginLine.left)
306         .attr("x", 0 - (heightLine / 2))
307         .attr("dy", "1em")
308         .style("text-anchor", "middle")
309         .text("Precipitation");
310
311     svg.append("path")
312         .datum(data)
313         .attr("fill", "none")
314         .attr("stroke", "steelblue")
315         .attr("stroke-width", 1.5)
316         .attr("d", d3.line()
317             .x(function(d) { return x(d3.timeParse("%Y-%m-%d")(d.date)) })
318             .y(function(d) { return y(d.precipitation) })
319 );
320
321 </script>
322
323 <p>
324 As we can see, the wind speed in Seattle is 1-4 most of the time. The distribution is mainly binomial. About 44% of the time is sunny
325 and another 44% of the time is raining. So, most of the time Seattle is either sunny or rainy. Most of the precipitation happens in
326 spring and summer. Adding up all of the observations, we could conclude that Seattle is a city that has the characteristics of both
327 the Mediterranean zone and oceanic zone.
328 </p>
329 </body>
330
331 </html>

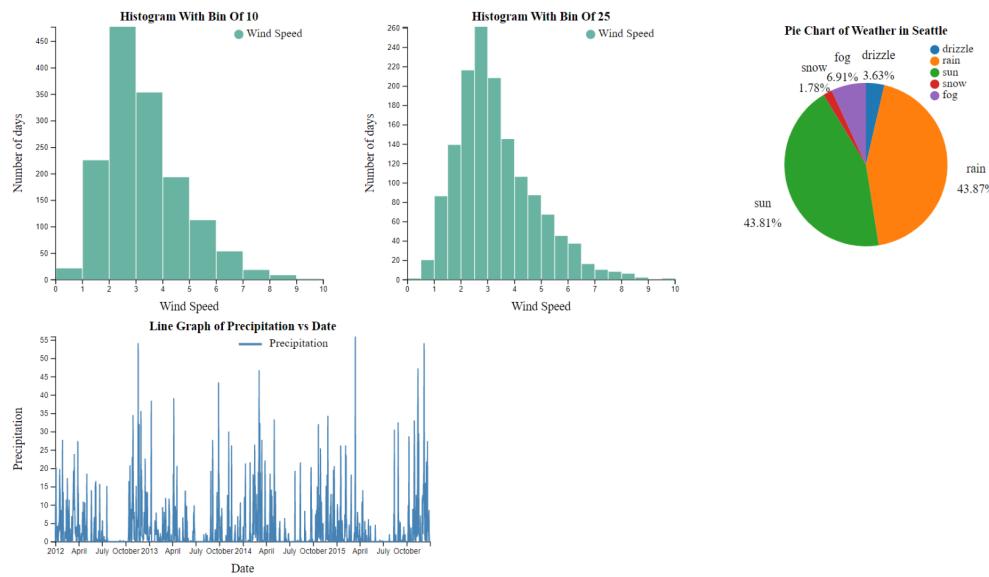
```

```
p {
```

```
    font-family: Arial, Helvetica, Sans Serif;  
    font-size: xx-large;  
    color: LightSkyBlue;  
    text-indent: 3%;
```

```
}
```

As we can see, the wind speed in Seattle is 1-4 most of the time. The distribution is mainly binomial. About 44% of the time is sunny and another 44% of the time is raining. So, most of the time Seattle is either sunny or rainy. Most of the precipitation happens in spring and summer. Adding up all of the observations, we could conclude that Seattle is a city that has the characteristics of both the Mediterranean zone and oceanic zone.



Problem 3

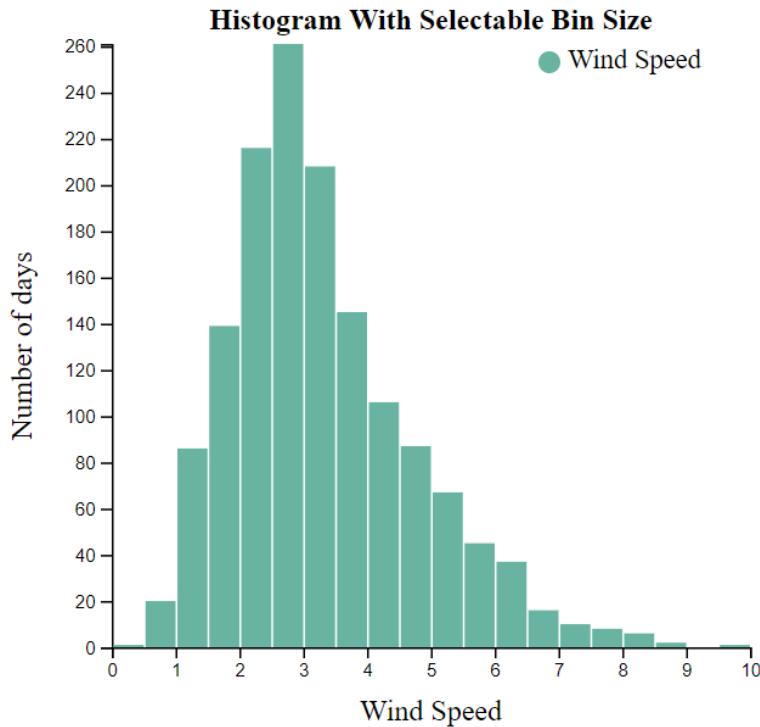
1.3.1

```

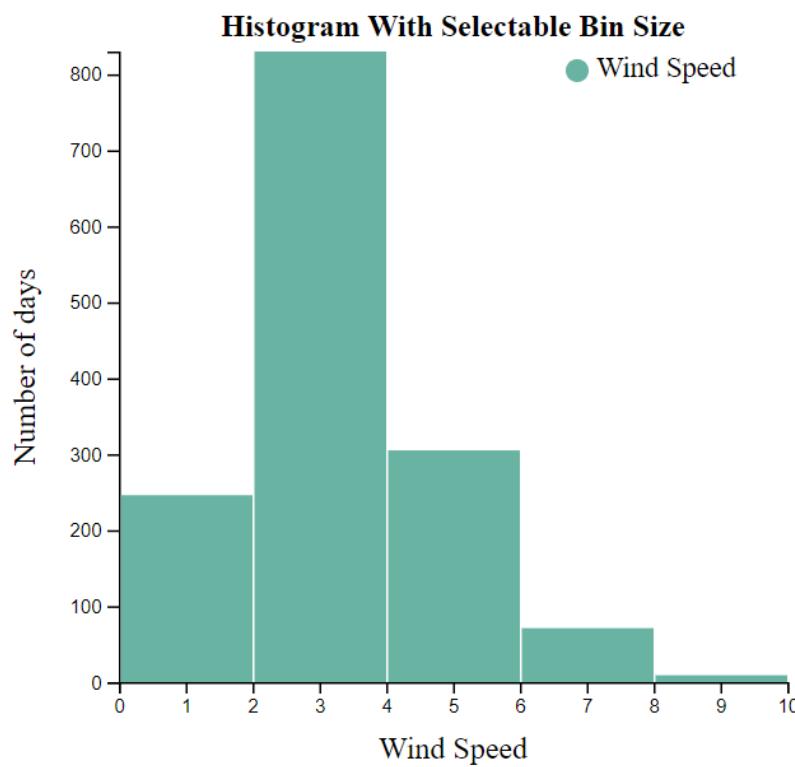
1  <!DOCTYPE html>
2  <meta charset="utf-8">
3  <script src="https://d3js.org/d3.v4.js"></script>
4  <div id="bin_select_histogram"></div>
5
6  <p>
7  <label>number of bins</label>
8  <input type="number" min="5" max="25" step="5" value="15" id="bin_num">
9  </p>
10 <script>
11
12 var margin = {top: 20, right: 50, bottom: 50, left: 60},
13     width = 460 - margin.left - margin.right,
14     height = 400 - margin.top - margin.bottom;
15
16 var svg = d3.select("#bin_select_histogram")
17   .append("svg")
18   .attr("width", width + margin.left + margin.right)
19   .attr("height", height + margin.top + margin.bottom)
20   .append("g")
21   .attr("transform",
22       "translate(" + margin.left + "," + margin.top + ")");
23
24 svg.append("text")
25   .attr("x", width / 2)
26   .attr("y", -8)
27   .attr("text-anchor", "middle")
28   .style("font-size", "16px")
29   .style("font-weight", "bold")
30   .text("Histogram With Selectable Bin Size");
31
32 svg.append("circle").attr("cx", width - 110).attr("cy", 10).attr("r", 6).style("fill", "#69b3a2")
33 svg.append("text").attr("x", width - 100).attr("y", 10).text("Wind Speed").style("font-size", "15px").attr("alignment-baseline", "middle")
34
35 d3.csv("https://raw.githubusercontent.com/vega/vega/main/docs/data/seattle-weather.csv", function(data) {
36   var x = d3.scaleLinear()
37     .domain([0, 10])
38     .range([0, width]);
39   svg.append("g")
40     .attr("transform", "translate(0," + height + ")")
41     .call(d3.axisBottom(x));
42   svg.append("text")
43     .attr("transform",
44         "translate(" + (width/2) + "," +
45             (height + margin.top + 20) + ")"
46     .style("text-anchor", "middle")
47     .text("Wind Speed");
48
49   var y = d3.scaleLinear()
50     .range([height, 0]);
51   var yAxis = svg.append("g")
52     .append("text")
53     .attr("transform", "rotate(-90)")
54     .attr("y", 0 - margin.left)
55     .attr("x", 0 - (height / 2))

```

```
57     .attr("dy", "1em")
58     .style("text-anchor", "middle")
59     .text("Number of days");
60
61     function update(bin_num) {
62
63         var histogram = d3.histogram()
64             .value(function(d) { return d.wind; })
65             .domain(x.domain())
66             .thresholds(x.ticks(bin_num));
67
68         var bins = histogram(data);
69
70         y.domain([0, d3.max(bins, function(d) { return d.length; })]);
71         yAxis
72             .transition()
73             .duration(1000)
74             .call(d3.axisLeft(y));
75
76         var u = svg.selectAll("rect")
77             .data(bins)
78         u
79             .enter()
80             .append("rect")
81             .merge(u)
82             .transition()
83             .duration(1000)
84             .attr("x", 1)
85             .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
86             .attr("width", function(d) { return x(d.x1) - x(d.x0) -1; })
87             .attr("height", function(d) { return height - y(d.length); })
88             .style("fill", "#69b3a2")
89
90         u
91             .exit()
92             .remove()
93
94     }
95
96     update(15)
97
98     d3.select("#bin_num").on("input", function() {
99         update(+this.value);
100    });
101
102 });
103
104
105 </script>
```



number of bins



number of bins

1.3.2

```
1  <!DOCTYPE html>
2  <meta charset="utf-8">
3  <script src="https://d3js.org/d3.v4.js"></script>
4  <div id="data_select_histogram"></div>
5
6  <body>
7      <label>Data </label>
8      <select id="option">
9          <option value="wind" selected>wind</option>
10         <option value="precipitation">precipitation</option>
11         <option value="temp_max">temp max</option>
12         <option value="temp_min">temp min</option>
13     </select>
14
15     <script>
16         var margin = {top: 20, right: 50, bottom: 50, left: 60},
17             width = 460 - margin.left - margin.right,
18             height = 400 - margin.top - margin.bottom;
19
20         var svg = d3.select("#data_select_histogram")
21             .append("svg")
22             .attr("width", width + margin.left + margin.right)
23             .attr("height", height + margin.top + margin.bottom);
24
25         svg.append("text")
26             .attr("x", width / 2 + 30)
27             .attr("y", 15)
28             .attr("text-anchor", "middle")
29             .style("font-size", "16px")
30             .style("font-weight", "bold")
31             .text("Histogram With Selectable Variable");
32
33         svg.append("circle").attr("cx", width - 80).attr("cy", 30).attr("r", 6).style("fill", "#69b3a2");
34         svg.append("text").attr("x", width - 60).attr("y", 30).text("Wind Speed").style("font-size", "15px").attr("alignment-baseline", "middle");
35
36         var container = svg
37             .append("g")
38             .attr("transform",
39                 "translate(" + margin.left + "," + margin.top + ")");
40
41     function histogram(option) {
42         d3.csv("https://raw.githubusercontent.com/vega/vega/main/docs/data/seattle-weather.csv", function(data) {
43
44             // pick data by option
45             const xFeature = (d) => d[option];
46             const yFeature = (d) => d.length;
47
48             // set up xticks and yticks
49             var x = d3.scaleLinear()
50                 .domain(d3.extent(data, xFeature))
51                 .range([0, width])
52                 .nice();
53
54             // draw bar x axis and x title
55             container.append("g")
56                 .attr("transform",
```

```

57         "translate(" + 0 + " , " + height + ")")  

58     .call(d3.axisBottom(x));  

59  

60     container.append("text")  

61         .attr("transform",  

62             "translate(" + (width/2) + " , " +  

63             (height + margin.top + 20) + ")")  

64         .style("text-anchor", "middle")  

65         .text(option);  

66  

67     var histogram = d3.histogram()  

68         .value(xFeature)  

69         .domain(x.domain())  

70         .thresholds(10);  

71  

72     var bins = histogram(data);  

73  

74     var y = d3.scaleLinear()  

75         .domain([0, d3.max(bins, yFeature)])  

76         .range([height, 0])  

77  

78     // draw bar y axis and y title  

79     container.append("g")  

80         .call(d3.axisLeft(y));  

81  

82     container.append("text")  

83         .attr("transform", "rotate(-90)")  

84         .attr("y", 0 - margin.left)  

85         .attr("x", 0 - (height / 2))  

86         .attr("dy", "1em")  

87         .style("text-anchor", "middle")  

88         .text('Number of days');  

89  

90     //draw bars  

91     bars = container  

92         .append("g").selectAll("rect")  

93         .data(bins)  

94         .enter()  

95         .append("rect")  

96         .attr("height", function(d) { return height - y(yFeature(d)); })  

97         .attr('width', function(d) { return x(d.x1) - x(d.x0) -1 ; })  

98         .attr("x", (d) => x(d.x0) )  

99         .attr("y", (d) => y(yFeature(d)) )  

100        .style("fill", "#69b3a2")  

101    );  

102}  

103}  

104}  

105  

106 d3.select('#option')  

107     .on('change', function (e) {  

108         // clear the container for replotting  

109         container.selectAll('*').remove();  

110         histogram(this.value);  

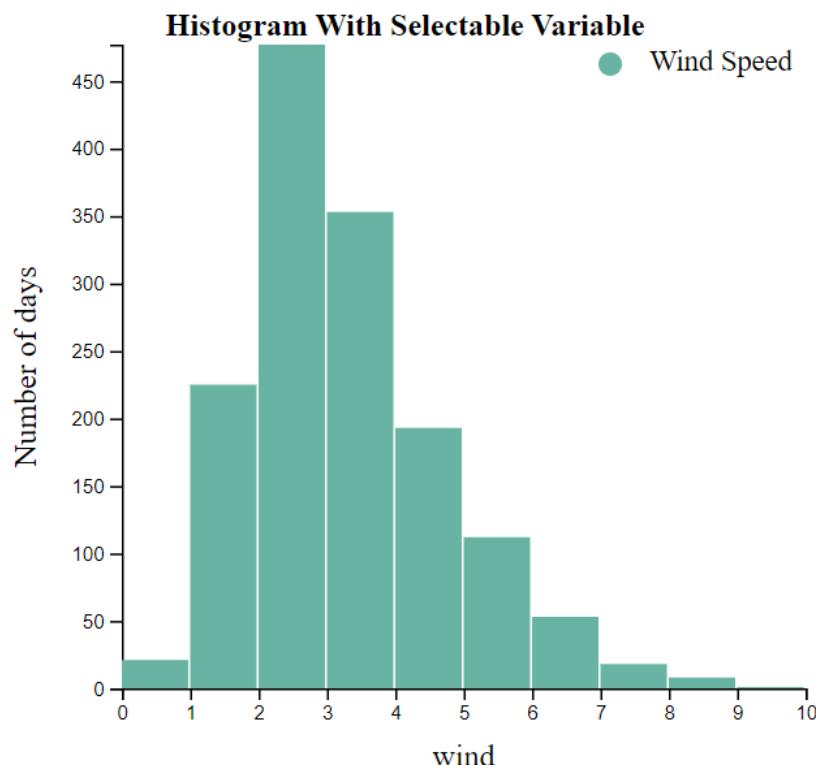
111     });  

112  

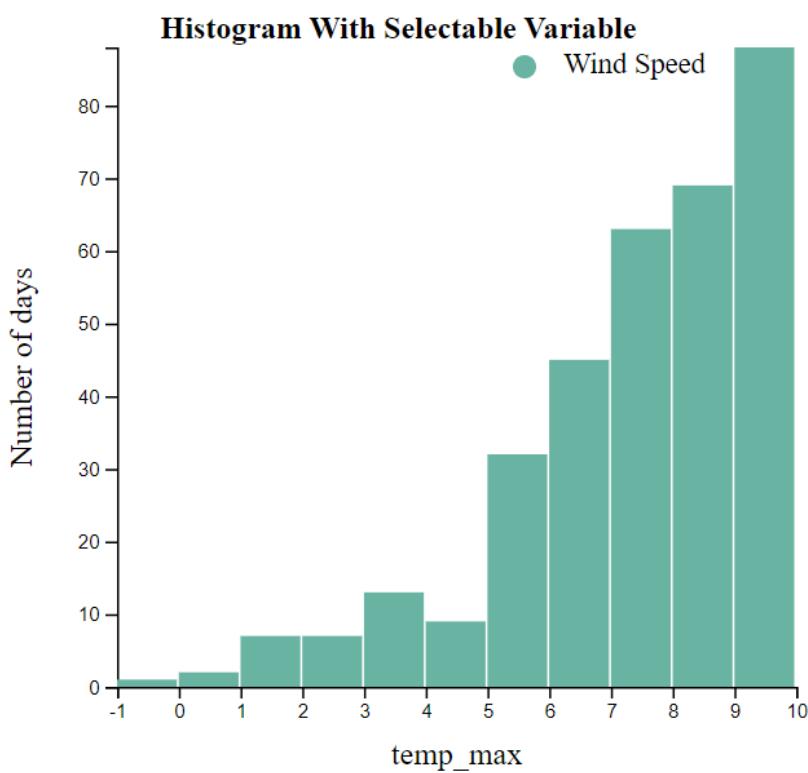
113 histogram('wind')

```

```
114      ...
115      </script>
116    </body>
117  </html>
```



Data ▾



Data ▾

Part II

2.1

```
1 <!DOCTYPE html>
2 <meta charset="utf-8">
3 <html>
4   <head>
5     <script src="https://d3js.org/d3.v4.js"></script>
6   </head>
7
8   <body>
9     <script type = "text/javascript" src="2.1.js"></script>
10  </body>
11 </html>
```

```
1 function displayTable(data, displayRows){
2   if (displayRows==null) {
3     var slicedData = data;
4   }
5   else {var slicedData = data.slice(0, displayRows);}
6
7   var table = d3.select('body').append('table')
8   var thead = table.append('thead')
9   var tbody = table.append('tbody')
10
11   thead.selectAll('th')
12     .data(d3.map(slicedData[0]).keys())
13     .enter()
14     .append('th')
15     .text(function (d) { return d });
16
17   var rows = tbody.selectAll('tr')
18     .data(slicedData)
19     .enter()
20     .append('tr');
21
22   var cells = rows.selectAll('td')
23     .data(function(row) {return d3.map(row).values(); })
24     .enter()
25     .append('td')
26     .text(function (d) { return d });
27
28 }
29
30 d3.csv('https://raw.githubusercontent.com/plotly/datasets/master/auto-mpg.csv',function (data) {
31   //display first 5 rows
32   displayTable(data, 5);
33
34   //groupby model year and count
35   var countByModelYear = d3.nest()
36     .key(function(d) { return d['model-year']; })
37     .rollup(function(dd) {
38       return d3.sum(dd, function(g) {return 1; });
39     })
40     .entries(data)
41     .map(function(data) {
42       return {
43         'model': data.key,
44         'count': data.value,
```

```

45      }
46    });
47
48    displayTable(countByModelYear, null);
49
50    //groupby model year and sum by cylinders
51    var countByModelYearCylinder = d3.nest()
52      .key(function(d) { return d['model-year']; })
53      .rollup(function(dd) {
54        return d3.sum(dd, function(d) {return d.cylinders; });
55      })
56      .entries(data)
57      .map(function(data) {
58        return {
59          'number of cylinders': data.key,
60          'total count': data.value,
61        }
62      });
63
64    displayTable(countByModelYearCylinder, null);
65
66    //groupby acceleration and count
67    var countByAccelerationCount = d3.nest()
68      .key(function(d) { return d.acceleration; })
69      .rollup(function(dd) {
70        return d3.sum(dd, function(d) {return 1; });
71      })
72      .entries(data)
73      .map(function(data) {
74        return {
75          'acceleration': data.key,
76          'count': data.value,
77        }
78      });
79    countByAccelerationCount.sort(function(x, y){
80      return d3.ascending(x.acceleration, y.acceleration);
81    })
82
83    displayTable(countByAccelerationCount, null);
84
85
86  }

```

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

model count

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

		82	130
acceleration count			
79	29	10	4
80	29	10.5	1
81	29	11	7
82	31	11.1	1
number of cylinders total count			11.2
70	196	11.3	1
71	156	11.4	2
72	163	11.5	7
73	255	11.6	1
74	142	12	10
75	168	12.1	1
76	192	12.2	2
77	153	12.5	8
78	193	12.6	2
79	169	12.8	3
80	120		
81	134		

12.9	2	14.9	7	17.1	1
13	12	15	14	17.2	2
13.2	6	15.1	2	17.3	5
13.4	2	15.2	3	17.4	2
13.5	2	15.3	3	17.5	4
13.6	15	15.4	4	17.6	4
13.6	2	15.5	21	17.7	3
13.7	2	15.6	1	17.8	2
13.7	2	15.7	4	17.9	1
13.8	2	15.8	7	18	8
13.9	2	15.9	2	18.1	1
14	16	16	16	18.2	5
14.1	1	16.1	1	18.3	1
14.1	1	16.2	4	18.5	5
14.2	3	16.4	9	18.6	4
14.3	2	16.5	13	18.7	2
14.4	5	16.6	3	18.8	1
14.5	23	16.7	3	19	12
14.5	23	16.8	2	19.2	3
14.7	5	16.9	4	19.4	3
14.8	3	17	14	19.5	6

19.6	2
19.9	1
20.1	2
20.4	1
20.5	3
20.7	1
21	5
21.5	1
21.7	1
21.8	1
21.9	1
22.1	1
22.2	2
23.5	1
23.7	1
24.6	1
24.8	1
8	1
8.5	2
9	1
9.5	2

2.2

```
1  <!DOCTYPE html>
2  <meta charset="utf-8">
3  <html>
4      <head>
5          <script src="https://d3js.org/d3.v4.js"></script>
6      </head>
7
8      <body>
9          <script type = "text/javascript" src="2.2.js"></script>
10     </body>
11 </html>
```

```
1▼ function displayTable(data, displayRows){
2▼     if (displayRows==null) {
3        var slicedData = data;
4    }
5    else {var slicedData = data.slice(0, displayRows);}
6
7    var table = d3.select('body').append('table')
8    var thead = table.append('thead')
9    var tbody = table.append('tbody')
10
11    thead.selectAll('th')
12        .data(d3.map(slicedData[0]).keys())
13        .enter()
14        .append('th')
15        .text(function (d) { return d });
16
17    var rows = tbody.selectAll('tr')
18        .data(slicedData)
19        .enter()
20        .append('tr');
21
22    var cells = rows.selectAll('td')
23        .data(function(row) {return d3.map(row).values(); })
24        .enter()
25        .append('td')
26        .text(function (d) { return d });
27
28}
29
30 d3.csv('https://raw.githubusercontent.com/plotly/datasets/master/auto-mpg.csv',function (data) {
31
32     //#### Start of 2.2.1 pie chart #####
33     //groupby model year and count
34     var pieData = d3.nest()
35         .key(function(d) { return d['model-year'];})
36         .rollup(function(dd) {
37             return d3.sum(dd, function(g) {return 1; });
38         })
39         .entries(data)
40
41     var countByModelYear = pieData
42         .map(function(d) {
43             return {
44                 'Model Year': d.key,
```

```

45     'Count': d.value,
46   });
47 });
48
49 var width = 400,
50 height = 410,
51 radius = (Math.min(width, height)-50) / 3;
52
53 var color = d3.scaleOrdinal(d3.schemeCategory20);
54
55 var arc = d3.arc()
56   .outerRadius(radius - 10)
57   .innerRadius(0);
58
59 var pie = d3.pie()
60   .sort(null)
61   .value(function(d) {
62     return d.value;
63   });
64
65 var svg = d3.select("body")
66   .append("svg")
67   .attr("width", width)
68   .attr("height", height)
69   .append("g")
70   .attr("transform", "translate(" + width / 2 + "," + height / 2 + ")");
71
72 svg.append("text")
73   .attr("x", 0)
74   .attr("y", -width / 2 + 40)
75   .attr("text-anchor", "middle")
76   .style("font-size", "16px")
77   .style("font-weight", "bold")
78   .text("Pie Chart of distribution of the model years");
79
80 var counter = 0
81 var c = ["rgb(31, 119, 180)", "rgb(174, 199, 232)", "rgb(255, 127, 14)", "rgb(255, 187, 120)", "rgb(44, 160, 44)",
82   "rgb(152, 223, 138)", "rgb(214, 39, 48)", "rgb(255, 152, 150)", "rgb(148, 103, 189)", "rgb(197, 176, 213)",
83   "rgb(140, 86, 75)", "rgb(196, 156, 148)", "rgb(227, 119, 194)"]
84 var name = ["70", "71", "72", "73", "74", "75", "76", "77", "78", "79", "80", "81", "82"]
85 pieData.forEach(function(d) {
86   svg.append("circle").attr("cx", width / 2 - 40).attr("cy", -width / 2 + 50 + counter * 15).attr("r", 6).style("fill", c[counter])
87   svg.append("text").attr("x", width / 2 - 30).attr("y", -width / 2 + 50 + counter * 15).text(name[counter])
88   .style("font-size", "15px").attr("alignment-baseline", "middle")
89   counter = counter + 1
90 });
91
92 //total count for percentage calculation
93 var total = d3.sum(pieData, function(d) {
94   return d.value;
95 });
96
97 //percentage calculation
98 pieData.forEach(function(d) {
99   d.percentage = (d.value / total * 100).toFixed(2);
100 });

```

```

101      var g = svg.selectAll(".arc")
102          .data( pie(pieData) )
103          .enter()
104          .append("g")
105          .attr("class", "arc");
106
107      g.append("path")
108          .attr("d", arc)
109          .style("fill", function(d) {
110              return color(d.data.key);
111          });
112
113      g.append("text")
114          .attr("transform", function(d) {
115              var _d = arc.centroid(d);
116              _d[0] *= 2.4;
117              _d[1] *= 2.4;
118              return "translate(" + _d + ")"
119          })
120          .attr("dy", ".35em")
121          .style("text-anchor", "middle")
122          .text(function(d) {
123              return d.data.key;
124          });
125
126
127      g.append("text")
128          .attr("transform", function(d) {
129              var _d = arc.centroid(d);
130              _d[0] *= 2.4;
131              _d[1] *= 2.4;
132              return "translate(" + _d + ")"
133          })
134          .attr("dy", "2.0em")
135          .style("text-anchor", "middle")
136          .text(function(d) {
137              return d.data.percentage + '%';
138          });
139
140
141      displayTable(countByModelYear, null);
142
143
144 //##### Start of 2.2.2 line chart #####
145 //groupby model year and sum by cylinders
146 var lineGraph = d3.nest()
147     .key(function(d) { return d['model-year']; })
148     .rollup(function(dd) {
149         return d3.sum(dd, function(d) {return d.cylinders; });
150     })
151     .entries(data)
152
153 var countByModelYearCylinder = lineGraph
154     .map(function(data) {
155         return {
156             'Model Year': data.key,

```

```

157     'Number of Cylinders': data.value,
158   });
159 });
160
161 var margin = {top: 20, right: 50, bottom: 50, left: 60},
162   width = 600 - margin.left - margin.right,
163   height = 337.5 - margin.top - margin.bottom;
164
165 var svg = d3.select("body")
166   .append("svg")
167   .attr("width", width + margin.left + margin.right)
168   .attr("height", height + margin.top + margin.bottom)
169   .append("g")
170   .attr("transform",
171     "translate(" + margin.left + "," + margin.top + ")");
172
173   svg.append("text")
174     .attr("x", width / 2)
175     .attr("y", -8)
176     .attr("text-anchor", "middle")
177     .style("font-size", "16px")
178     .style("font-weight", "bold")
179     .text("Line Graph of Model Year vs Total Number of Cylinders");
180
181   svg.append("line").attr("x1", width - 110).attr("y1", 10).attr("x2", width - 80).attr("y2", 10).style("stroke", "steelblue")
182     .style("stroke-width", 3)
183   svg.append("text").attr("x", width - 70).attr("y", 10).text("Precipitation").style("font-size", "15px")
184     .attr("alignment-baseline", "middle")
185
186 var parseTime = d3.timeParse("%y");
187 lineGraph.forEach(function(d) {d.key = parseTime(d.key)})
188
189 lineGraph.sort(function(x, y){
190   return d3.descending(x.key, y.key);
191 })
192
193 var x = d3.scaleTime()
194   .domain(d3.extent(lineGraph, function(d) { return d.key; }))
195   .range([width, 0]);
196   svg.append("g")
197     .attr("transform", "translate(0," + height + ")")
198     .call(d3.axisBottom(x));
199   svg.append("text")
200     .attr("transform",
201       "translate(" + (width/2) + " , "
202         +(height + margin.top + 20) + ")")
203     .style("text-anchor", "middle")
204     .text("Model Year");
205
206 var y = d3.scaleLinear()
207   .domain([0, d3.max(lineGraph, function(d) { return +d.value; })])
208   .range([ height, 0 ]);
209   svg.append("g")
210     .call(d3.axisLeft(y));
211   svg.append("text")
212     .attr("transform", "rotate(-90)");

```

```

213         .attr("y", 0 - margin.left)
214         .attr("x", 0 - (height / 2))
215         .attr("dy", "1em")
216         .style("text-anchor", "middle")
217         .text("Total number of Cylinders");
218
219     svg.append("path")
220       .datum(lineGraph)
221       .attr("fill", "none")
222       .attr("stroke", "steelblue")
223       .attr("stroke-width", 1.5)
224       .attr("d", d3.line()
225         .x(function(d) { return x(d.key) })
226         .y(function(d) { return y(d.value) })
227       )
228
229     displayTable(countByModelYearCylinder, null);
230
231     //##### Start of 2.2.3 histogram #####
232     //groupby acceleration and count
233     var histogramData = d3.nest()
234       .key(function(d) { return d.acceleration;})
235       .rollup(function(dd) {
236         return d3.sum(dd, function(d) {return 1; });
237       })
238       .entries(data)
239
240     var countByAccelerationCount = histogramData
241       .map(function(data) {
242         return {
243           'Acceleration': data.key,
244           'Count': data.value,
245         }
246       });
247
248     countByAccelerationCount.sort(function(x, y){
249       return d3.ascending(x.acceleration, y.acceleration);
250     })
251
252     var margin = {top: 20, right: 50, bottom: 50, left: 60},
253       width = 600 - margin.left - margin.right,
254       height = 337.5 - margin.top - margin.bottom;
255
256     var svg = d3.select("body")
257       .append("svg")
258       .attr("width", width + margin.left + margin.right)
259       .attr("height", height + margin.top + margin.bottom)
260       .append("g")
261       .attr("transform",
262         "translate(" + margin.left + "," + margin.top + ")");
263
264     svg.append("text")
265       .attr("x", width / 2)
266       .attr("y", -8)
267       .attr("text-anchor", "middle")
268       .style("font-size", "16px")

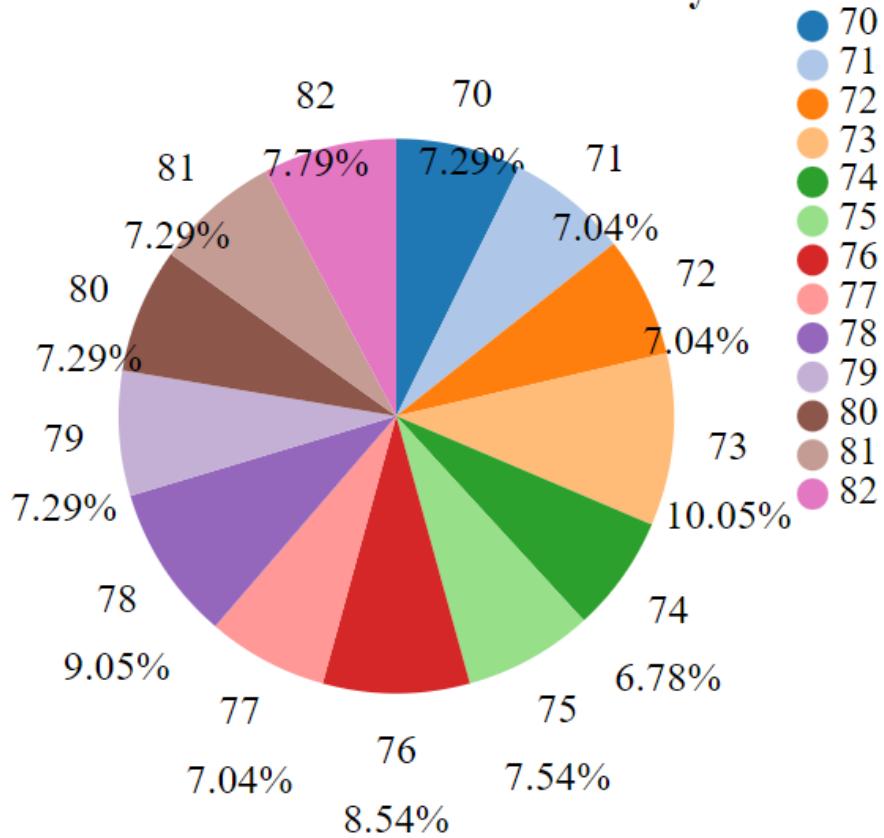
```

```

269         .style("font-weight", "bold")
270         .text("Histogram of Acceleration Distribution With Bin Of 10");
271
272     svg.append("circle").attr("cx",width - 100).attr("cy",10).attr("r", 6).style("fill", "#69b3a2")
273     svg.append("text").attr("x", width - 90).attr("y", 10).text("Wind Speed").style("font-size", "15px")
274         .attr("alignment-baseline", "middle")
275
276     var x = d3.scaleLinear()
277         .domain([6,28])
278         .range([0, width]);
279     svg.append("g")
280         .attr("transform", "translate(0," + height + ")")
281         .call(d3.axisBottom(x));
282     svg.append("text")
283         .attr("transform",
284             "translate(" + (width/2) + " , " +
285             (height + margin.top + 20) + ")");
286         .style("text-anchor", "middle")
287         .text("Acceleration");
288
289     var histogram = d3.histogram()
290         .value(function(d) { return d.acceleration; })
291         .domain(x.domain())
292         .thresholds(x.ticks(10));
293
294     var bins = histogram(data);
295     console.log(bins)
296     var y = d3.scaleLinear()
297         .domain([0, d3.max(bins, function(d) { return d.length; })])
298         .range([height, 0])
299     svg.append("g")
300         .text("Number of days")
301         .call(d3.axisLeft(y));
302     svg.append("text")
303         .attr("transform", "rotate(-90)")
304         .attr("y", 0 - margin.left)
305         .attr("x",0 - (height / 2))
306         .attr("dy", "1em")
307         .style("text-anchor", "middle")
308         .text("Number of cars");
309
310     svg.selectAll("rect")
311         .data(bins)
312         .enter()
313         .append("rect")
314             .attr("x", 1)
315             .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
316             .attr("width", function(d) { return x(d.x1) - x(d.x0) -1 ; })
317             .attr("height", function(d) { return height - y(d.length); })
318             .style("fill", "#69b3a2")
319
320         displayTable(countByAccelerationCount, null);
321
322     }

```

Pie Chart of distribution of the model years



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

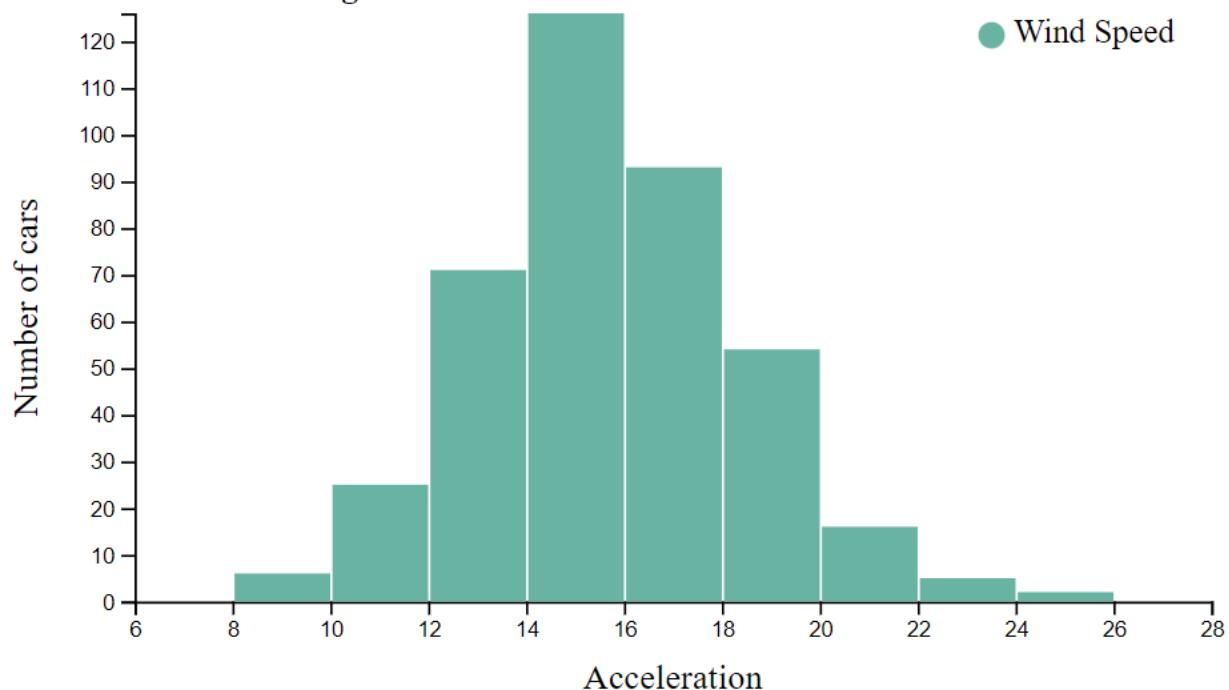
Line Graph of Model Year vs Total Number of Cylinders



Model Year Number of 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

Histogram of Acceleration Distribution With Bin Of 10

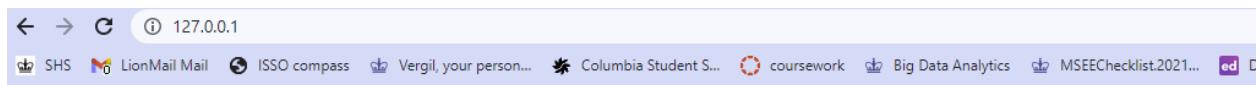


Acceleration Count

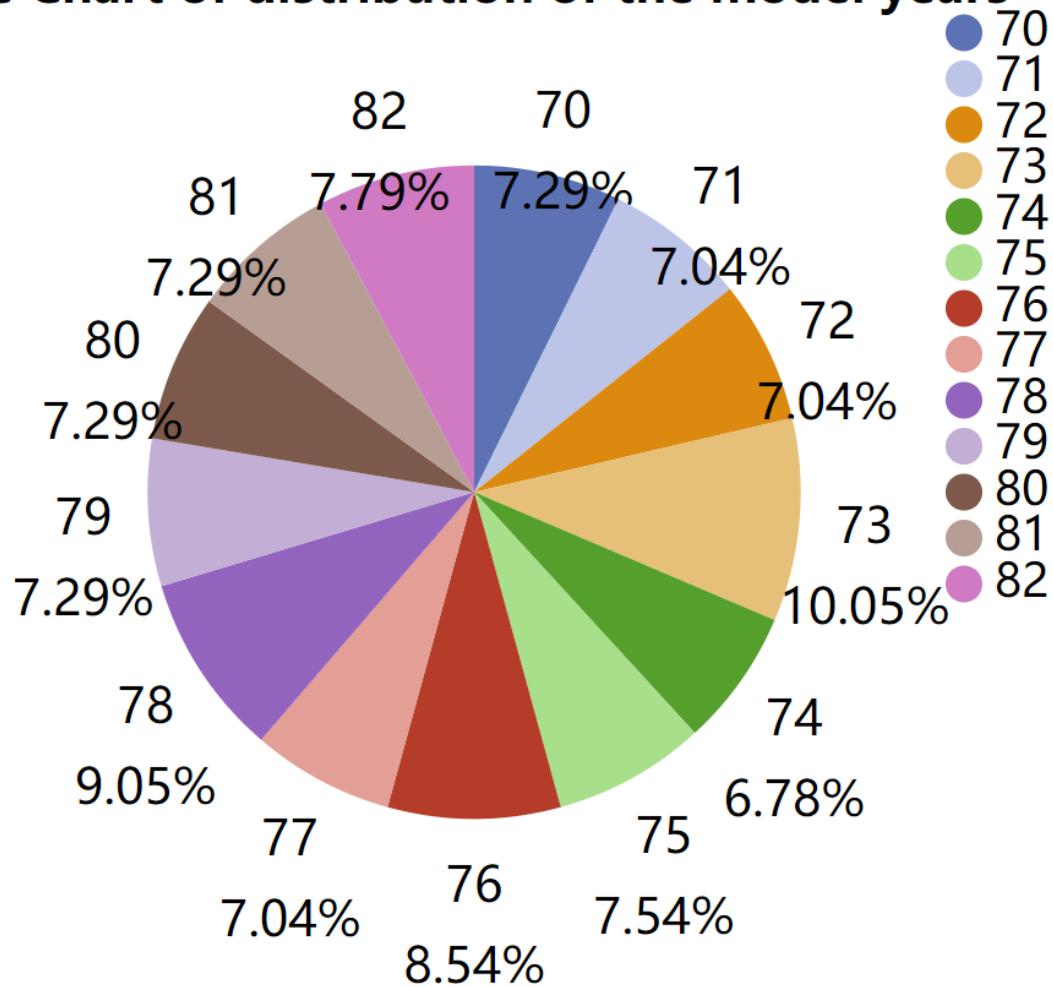
12	10
11.5	7
11	7
10.5	1
10	4
9	1
8.5	2
8	1
9.5	2
15	14
15.5	21
16	16
14.5	23
20.5	3
17.5	4
12.5	8
14	16
13.5	15

		13.7	2		
18.5	5	11.1	1		
19	12	11.4	2		
13	12	18.2	5		
19.5	6	15.8	7		
18	8	15.9	2		
17	14	14.1	1		
23.5	1	21.5	1		
16.5	13	14.4	5		
21	5	19.4	3		
16.9	4	19.2	3		
14.9	7	17.2	2		
17.7	3	18.7	2		
15.3	3	15.1	2		
13.9	2	13.4	2		
12.8	3	11.2	1		
15.4	4	14.7	5		
17.6	4	16.6	3		
22.2	2	17.3	5		
22.1	1	15.2	3		
14.2	3	14.3	2		
17.4	2	20.1	2		
16.2	4	24.8	1	16.1	1
17.8	2	11.3	1	20.7	1
12.2	2	12.9	2	18.3	1
13.6	2	18.8	1	20.4	1
15.7	4	18.1	1	19.6	2
21.9	1	21.7	1	17.1	1
16.7	3	23.7	1	15.6	1
12.1	1	21.8	1	24.6	1
14.8	4	13.8	2	11.6	1
18.6	2	12.6	2		
16.8					

Problem 3



Pie Chart of distribution of the model years



Model Year Count

70	29
71	28

127.0.0.1	304	d...	Other	24...	5 ...		
d3.v4.js	200	sc...	(index)	(... 0 ...			
2.2.js	200	sc...	(index)	(... 0 ...			
auto-mpg.csv	200	xhr	d3.v4.j...	(di... 1 ...			

Part III

```

1  <!DOCTYPE html>
2  <meta charset="utf-8">
3  <html>
4
5  <head>
6      <link rel="stylesheet" href="3.1&3.2.css">
7      <script src="http://d3js.org/d3.v3.js"></script>
8      <script src="3.1&3.2.js"></script>
9  </head>
10
11 <body>
12
13     <script>
14
15         var width = 800,
16             height = 450;
17
18         var color = d3.scale.category20();
19
20         var force = d3.layout.force()
21             .charge(-120)
22             .linkDistance(30)
23             .size([width, height]);
24
25         var svg = d3.select("body").append("svg")
26             .attr("width", width)
27             .attr("height", height);
28
29         svg.append("text")
30             .attr("x", width / 2)
31             .attr("y", 20)
32             .attr("text-anchor", "middle")
33             .style("font-size", "16px")
34             .style("font-weight", "bold")
35             .text("Network Graph For Character In Les Miserables");
36
37         var counter = 0
38         var c = ["rgb(31, 119, 180)", "rgb(174, 199, 232)", "rgb(152, 223, 138)", "rgb(255, 127, 14)", "rgb(44, 160, 44)", "rgb(255, 187, 120)", "rgb(214, 39, 40)", "rgb(148, 103, 189)", "rgb(140, 86, 75)", "rgb(255, 152, 150)", "rgb(197, 176, 213)"]
39         var n = ["Group 1", "Group 2", "Group 3", "Group 4", "Group 5", "Group 6", "Group 7", "Group 8", "Group 9", "Group 10", "Group 11"]
40         d3.range(11).forEach(function(d) {
41             svg.append("circle").attr("cx", 100).attr("cy", 100 + counter * 15).attr("r", 6).style("fill", c[counter])
42             svg.append("text").attr("x", 110).attr("y", 100 + counter * 15).text(n[counter]).style("font-size", "15px").attr("alignment-baseline", "middle")
43             counter = counter + 1
44         });
45
46         var fisheye = d3.fisheye.circular()
47             .radius(50)
48             .distortion(5);
49
50         d3.csv("https://gist.githubusercontent.com/timelyportfolio/5049980/raw/66a239b4aa325c05c7a19bd96bf093632591e809/les_mis.csv", function
51             (data) {
52
53                 graph = { "nodes": [], "links": [] };
54
55                 data.forEach(function (d) {

```

```

55      graph.nodes.push({ "character": d.source, "group": +d.groupsource });
56      graph.nodes.push({ "character": d.target, "group": +d.grouptarget });
57
58      graph.links.push({ "source": d.source, "target": d.target, "value": +d.value });
59    });
60
61    var nodesMap = d3.nest()
62      .key(function (d) { return d.character; })
63      .rollup(function (d) { return { "character": d[0].character, "group": d[0].group }; })
64      .map(graph.nodes);
65
66    graph.nodes = d3.keys(d3.nest())
67      .key(function (d) { return d.character; })
68      .map(graph.nodes);
69
70    graph.links.forEach(function (d, i) {
71      graph.links[i].source = graph.nodes.indexOf(graph.links[i].source);
72      graph.links[i].target = graph.nodes.indexOf(graph.links[i].target);
73    });
74
75    graph.nodes.forEach(function (d,i) { graph.nodes[i]={ "character": nodesMap[d].character, "group": nodesMap[d].group }; });
76
77    force
78      .nodes(graph.nodes)
79      .links(graph.links)
80      .start();
81
82    var link = svg.selectAll(".link")
83      .data(graph.links)
84      .enter()
85      .append("line")
86      .attr("class", "link")
87      .style("stroke-width", function (d) { return Math.sqrt(d.value); });
88
89    var node = svg.selectAll(".nodes")
90      .data(graph.nodes)
91      .enter()
92      .append("g")
93      .attr("class", "nodes")
94      .call(force.drag);
95
96    node.append("text")
97      .attr("dy", -10)
98      .attr("dx", -13)
99      .style("font-size", "6px")
100     .style("text-align", "center")
101     .text(d => d.character);
102
103   node.append("circle")
104     .attr("r", d=> 6)
105     .attr("id",d=> "circle"+d.id)
106     .style("stroke", "grey")
107     .style("stroke-opacity",0.3)
108     .style("fill", d => color(d.group));
109
110   svg.on("mousemove", function() {
111     fisheye.focus(d3.mouse(this));

```

```

112
113         node.each(function(d) { d.fisheye = fisheye(d); })
114
115     ▼
116         node.selectAll("circle")
117             .attr("vx", function(d) { return d.fisheye.x; })
118             .attr("vy", function(d) { return d.fisheye.y; })
119             .attr("r", function(d) { return d.fisheye.z * 4.5; });
120
121     ▼
122         node.selectAll("text")
123             .attr("dx", function(d) { return d.fisheye.x - d.x; })
124             .attr("dy", function(d) { return d.fisheye.y - d.y; });
125
126     link
127         .attr("x1", function(d) { return d.source.fisheye.x; })
128         .attr("y1", function(d) { return d.source.fisheye.y; })
129         .attr("x2", function(d) { return d.target.fisheye.x; })
130         .attr("y2", function(d) { return d.target.fisheye.y; });
131
132     ▼
133         force.on("tick", function () {
134             link
135                 .attr("x1", function (d) { return d.source.x; })
136                 .attr("y1", function (d) { return d.source.y; })
137                 .attr("x2", function (d) { return d.target.x; })
138                 .attr("y2", function (d) { return d.target.y; });
139         });
140
141     </script>
142
143     ▼
144     <p>
145         I see there are multiple clusters or groups that each of them are represented in different colors. People in the same group are generally more connected, meaning there are more edges among members in the same group. Usually, there is one or a few people in one group that are responsible for connecting with other groups.
146     </p>
147     </body>
148
149 </html>

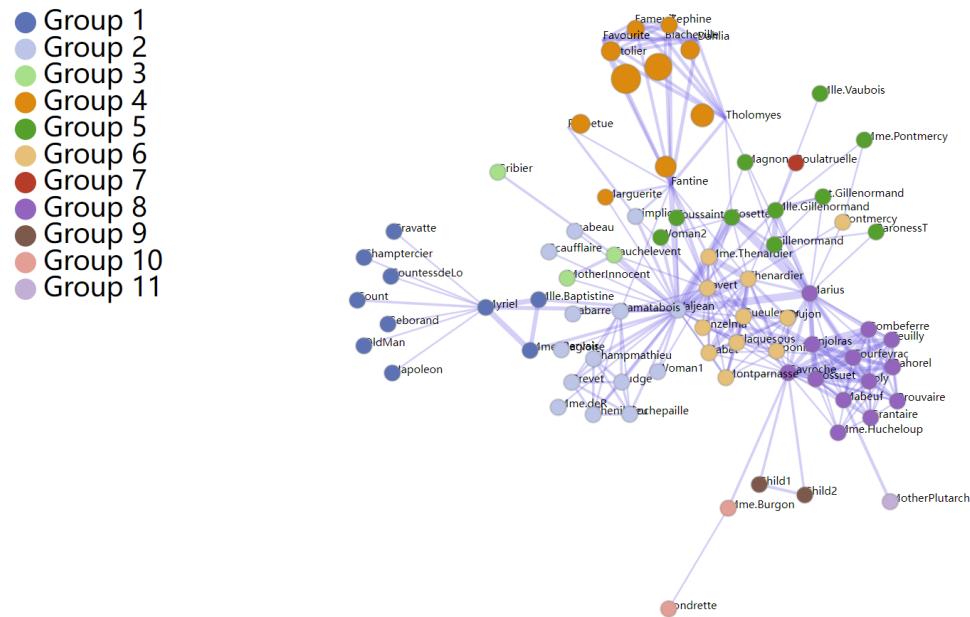
```

```
1  (function() {
2    d3.fisheye = {
3      scale: function(scaleType) {
4        return d3_fisheye_scale(scaleType(), 3, 0);
5      },
6      circular: function() {
7        var radius = 200,
8            distortion = 2,
9            k0,
10           k1,
11           focus = [0, 0];
12
13      function fisheye(d) {
14        var dx = d.x - focus[0],
15            dy = d.y - focus[1],
16            dd = Math.sqrt(dx * dx + dy * dy);
17        if (!dd || dd >= radius) return {x: d.x, y: d.y, z: dd >= radius ? 1 : 10};
18        var k = k0 * (1 - Math.exp(-dd * k1)) / dd * .75 + .25;
19        return {x: focus[0] + dx * k, y: focus[1] + dy * k, z: Math.min(k, 10)};
20      }
21
22      function rescale() {
23        k0 = Math.exp(distortion);
24        k0 = k0 / (k0 - 1) * radius;
25        k1 = distortion / radius;
26        return fisheye;
27      }
28
29      fisheye.radius = function(_) {
30        if (!arguments.length) return radius;
31        radius = +_;
32        return rescale();
33      };
34
35      fisheye.distortion = function(_) {
36        if (!arguments.length) return distortion;
37        distortion = +_;
38        return rescale();
39      };
40
41      fisheye.focus = function(_) {
42        if (!arguments.length) return focus;
43        focus = _;
44        return fisheye;
```

```
45     };
46     return rescale();
47   }
48 };
49 };
50
51 function d3_fisheye_scale(scale, d, a) {
52
53 function fisheye(_) {
54   var x = scale(_),
55     left = x < a,
56     range = d3.extent(scale.range()),
57     min = range[0],
58     max = range[1],
59     m = left ? a - min : max - a;
60   if (m == 0) m = max - min;
61   return (left ? -1 : 1) * m * (d + 1) / (d + (m / Math.abs(x - a))) + a;
62 }
63
64 fisheye.distortion = function(_) {
65   if (!arguments.length) return d;
66   d = +_;
67   return fisheye;
68 };
69
70 fisheye.focus = function(_) {
71   if (!arguments.length) return a;
72   a = +_;
73   return fisheye;
74 };
75
76 fisheye.copy = function() {
77   return d3_fisheye_scale(scale.copy(), d, a);
78 };
79
80 fisheye.nice = scale.nice;
81 fisheye.ticks = scale.ticks;
82 fisheye.tickFormat = scale.tickFormat;
83 return d3.rebind(fisheye, scale, "domain", "range");
84 }
85 })();
```

```
1  body{  
2      font: Arial 12px;  
3      text-align: center;  
4  }  
5  
6  p {  
7      font-family: Arial, Helvetica, Sans Serif;  
8      font-size: 20px;  
9      color: LightSkyBlue;  
10     text-indent: 3%;  
11  }  
12  
13 .node {  
14     stroke: black;  
15     stroke-width: 20px;  
16  }  
17  
18 .link {  
19     stroke: royalblue;  
20     stroke-opacity: 0.3;  
21  }
```

Network Graph For Character In Les Miserables



I see there are multiple clusters or groups that each of them are represented in different colors. People in the same group are generally more connected, meaning there are more edges among members in the same group. Usually, there is one or a few people in one group that are responsible for connecting with other groups.

Name	St...	Ty...	Initiator	Size	Ti...	Waterfall
127.0.0.1	304	d...	Other	24...	2 ...	█
3.1&3.2.js	200	sc...	(index)	(...)	0 ...	█
3.1&3.2.css	200	st...	(index)	(di...)	2 ...	█ █
d3.v3.js	307	sc...	(index)	(di...)	2 ...	█
d3.v3.js	200	sc...	d3.v3.js	(di...)	4 ...	█
les_mis.csv	200	xhr	d3.v3.j...	(di...)	1 ...	█ █