CSE 564: Visualization Mini Project #1 [Khurana, Harmanpreet (113262379)]

Video Link

https://drive.google.com/file/d/1NM2KkLM2EHR8e-wtu94lM8XBrygkzmb9/view?usp=sharing

Requirement

This mini project asked to implement three type of Data Visualizations, namely Scatter Plot, Bar Chart and a Histogram with the following properties:

Bar Chart

- For Categorical Variables
- Highlight the bar and display its value on top

Histogram

- For Numerical Variables
- Highlight the bar and display its value on top
- Left drag of mouse should decrease the bin size (increase no. of bins)
- Right drag of mouse should increase the bin size (decrease no. of bins)

Scatter Plot

- Select two variables and plot a graph between them
- Use Radio button to choose the axes among the two variables

Also, a menu should allow user to choose a variable and based on the type of the chosen variable, Bar Chart or Histogram should be displayed. The dataset can be taken from anywhere.

Solution

Data:

NBA 2K21 Player Data was taken from **Kaggle** The data is about the NBA players with the following attributes and their types:

- *Name (Categorical)*
- *Jersey (Categorical)*
- *Team (Categorical)*
- *Position (Categorical)*
- *Country (Categorical)*
- *College (Categorical)*
- *Height (in inches) (Numerical)*
- *Height (in meters) (Numerical)*
- *Weight (in LBS) (Numerical)*
- *Weight (in KG) (Numerical)*
- *Salary (Numerical)*
- *Draft Round (Numerical)*
- *Draft Peak (Numerical)*
- *Rating (Categorical and Numerical)*

- *Draft Year (Categorical and Numerical)*

Some of the data was cleaned using python. For example, the height in inches was given like "6-7" and was converted into a decimal (6 + (7/12)) Here is a screenshot of the attributes and rows (some).

	full_name	rating	jersey	team	position	b_day	height_inInches	height_inMeters	weight_inlbs	weight_inkg	salary	country	draft_year	draft_round	draft_peak	college
0	LeBron James	97	23	Los Angeles Lakers	F	12/30/84	6.75	2.06	250	113.4	37436858	USA	2003	1	1	Tennessee State
1	Kawhi Leonard	97	2	Los Angeles Clippers	F	06/29/91	6.58	2.01	225	102.1	32742000	USA	2011	1	15	San Diego State
2	Giannis Antetokounmpo	96	34	Milwaukee Bucks	F-G	12/06/94	6.08	2.11	242	109.8	25842697	Greece	2013	1	15	Louisville
3	Kevin Durant	96	7	Brooklyn Nets	F	09/29/88	6.08	2.08	230	104.3	37199000	USA	2007	1	2	Texas
4	James Harden	96	13	Houston Rockets	G	08/26/89	6.42	1.96	220	99.8	38199000	USA	2009	1	3	Arizona State
5	Stephen Curry	95	30	Golden State Warriors	G	03/14/88	6.25	1.91	185	83.9	40231758	USA	2009	1	7	Davidson
6	Anthony Davis	94	3	Los Angeles Lakers	F-C	03/11/93	6.08	2.08	222	100.7	27093019	USA	2012	1	1	Kentucky
7	Paul George	93	13	Los Angeles Clippers	F	05/02/90	6.67	2.03	210	95.3	33005556	USA	2010	1	10	Fresno State
8	Damian Lillard	92	0	Portland Trail Blazers	G	07/15/90	6.17	1.88	195	88.5	29802321	USA	2012	1	6	Weber State
9	Joel Embiid	91	21	Philadelphia 76ers	C	03/16/94	7.0	2.13	250	113.4	27504630	Cameroon	2014	1	3	Kansas
10	Kyrie Irving	91	11	Brooklyn Nets	G	03/23/92	6.17	1.88	180	81.6	31742000	Australia	2011	1	1	Duke
11	Nikola Jokic	90	15	Denver Nuggets	C	02/19/95	7.0	2.13	253	114.8	27504630	Serbia	2014	2	41	Arizona
12	Russell Westbrook	90	0	Houston Rockets	G	11/12/88	6.25	1.91	190	86.2	38506482	USA	2008	1	4	UCLA
13	Klay Thompson	89	11	Golden State Warriors	G	02/08/90	6.5	1.98	205	93.0	32742000	USA	2011	1	11	Washington State
14	Karl-Anthony Towns	89	32	Minnesota Timberwolves	F-C	11/15/95	6.08	2.11	248	112.5	27285000	USA	2015	1	1	Kentucky
15	Jimmy Butler	88	22	Miami Heat	G	09/14/89	6.58	2.01	220	99.8	32742000	USA	2011	1	30	Marquette
16	Rudy Gobert	88	27	Utah Jazz	C	06/26/92	7.08	2.16	238	108.0	25258427	France	2013	1	27	UCLA
17	Blake Griffin	88	23	Detroit Pistons	F	03/16/89	6.75	2.06	252	114.3	34449964	USA	2009	1	1	Oklahoma
18	Donovan Mitchell	88	45	Utah Jazz	G	09/07/96	6.08	1.85	211	95.7	3635760	USA	2017	1	13	Louisville
19	Kemba Walker	88	8	Boston Celtics	G	05/08/90	6.0	1.83	172	78.0	32742000	USA	2011	1	9	UConn
20	Luka Doncic	87	77	Dallas Mavericks	G-F	02/28/99	6.58	2.01	227	103.0	7683360	Slovenia	2018	1	3	Fresno State
21	LaMarcus Aldridge	87	12	San Antonio Spurs	F-C	07/19/85	6.08	2.11	246	111.6	26000000	USA	2006	1	2	Texas
22	Bradley Beal	87	3	Washington Wizards	G	06/28/93	6.25	1.91	207	93.9	27093018	USA	2012	1	3	Florida
23	Mike Conley	87	10	Utah Jazz	G	10/11/87	6.08	1.85	185	83.9	32511623	USA	2007	1	4	Ohio State
24	DeMar DeRozan	87	10	San Antonio Spurs	G-F	08/07/89	6.5	1.98	220	99.8	27739975	USA	2009	1	9	USC
25	CJ McCollum	87	3	Portland Trail Blazers	G	09/19/91	6.25	1.91	197	89.4	27556959	USA	2013	1	10	Lehigh
26	Victor Oladipo	87	4	Indiana Pacers	G	05/04/92	6.33	1.93	213	96.6	21000000	USA	2013	1	2	Indiana

Technologies:

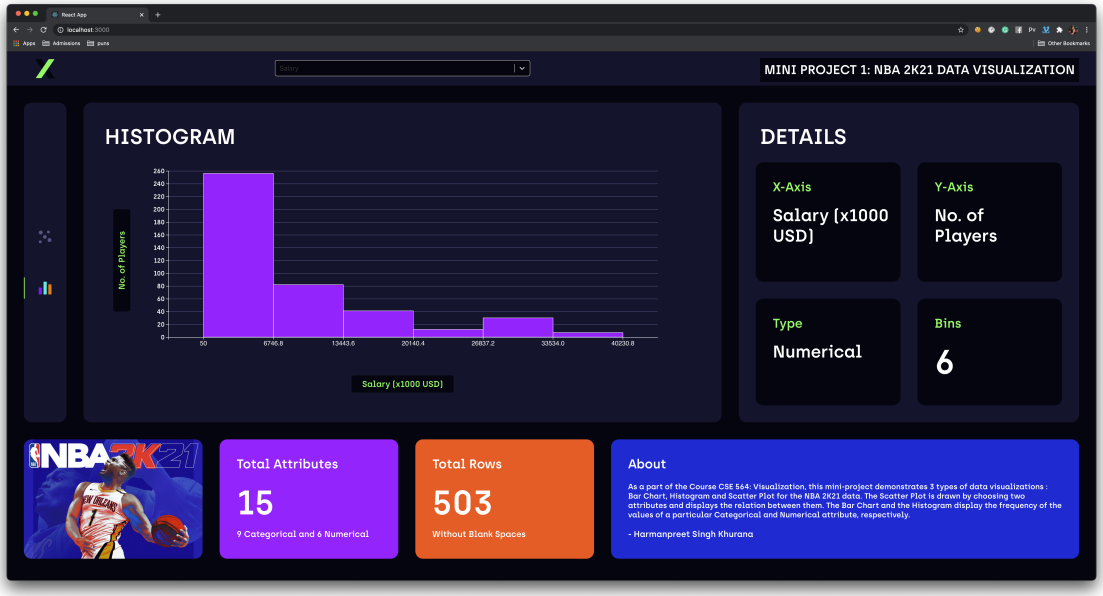
- **d3.js** All the data visualizations drawn in this project have been done via d3.js
- **ReactJS** All the front end tasks have been done using ReactJS
- **python** Python is used to clean some of the data

Working:

Home Page

The Home page is a dashboard where you can play around with the type graphs.

On the Top, there is a drop down which lets you choose an attribute and based on its type, it will dispay the Bar Chart or a Histogram. On the Left, there is button to switch to the Scatter Plot mode, which is explained in the later section The bottom space defines the data metrics for the project and some "about" information.

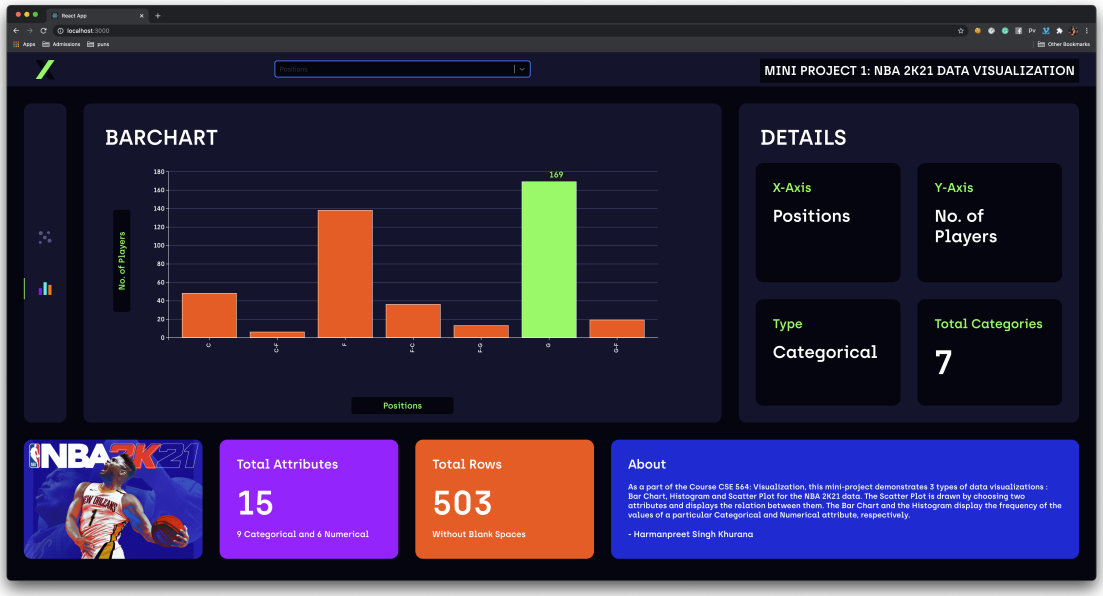


Bar Graph

When a Categorical Variable is chosen, for example, Positions in this case, the Bar graph displays the Positions that the players play at (X-axis) and the number of players in each position (on the Y-axis) **The Green Highlight is done via the mouse pointer and its value is displayed on the top, like asked**. *We can infer from this particular example the most players in the data set play at the G position.*

(The highlight is done in Green Color instead of the red which was asked, to match the theme. I hope that is okay.)

The X Axis text has been rotated to fit some other categories which had larger text. The right block gives some details about the chart as shown.



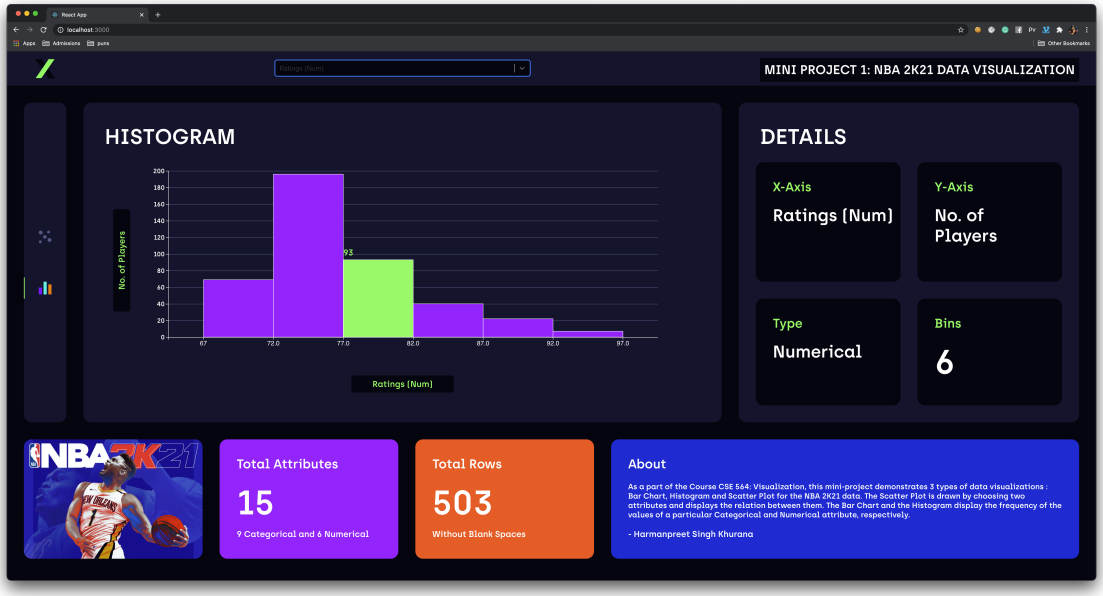
Histogram

Now when a Numerical Variable like Ratings, for example, A Histogram is displayed with the defined number of bins.

On dragging the mouse, the bins increase/decrease in size which is shown in the video. The number of bins text on the right also updates accordingly.

Just like in Bar Chart, the bar here is also highlighted in the same way

We can infer from this particular example that there very few players rated above 92



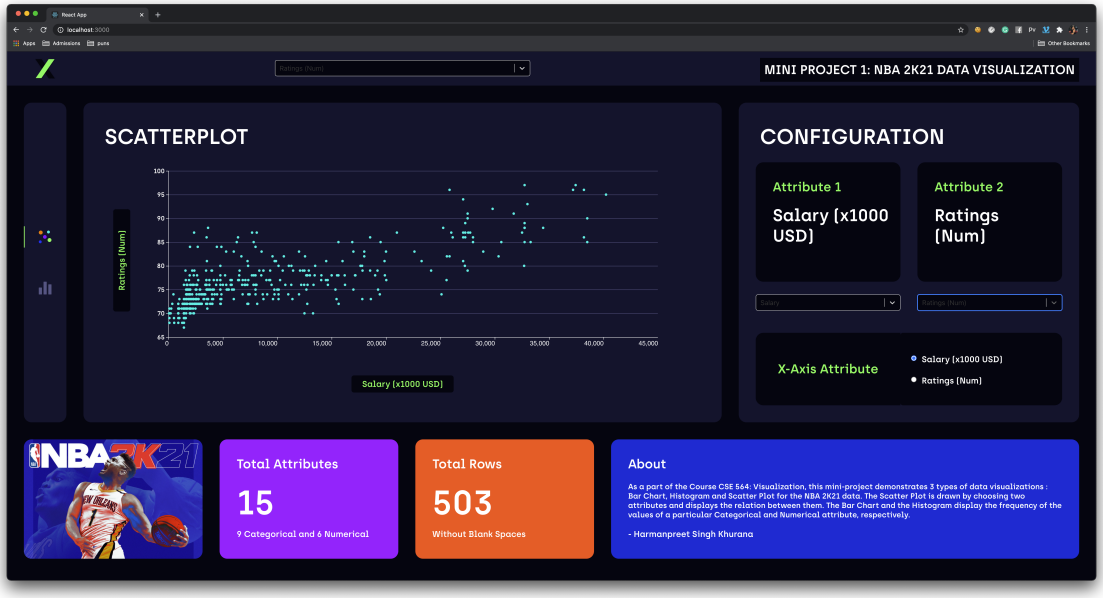
Scatter Plot

On choosing the Scatter Plot button on the left side of the page, a Scatter Plot is displayed. **The menu on the right helps you configure the plot by choosing the two attributes between which the graph is to be plotted. Also, it lets you choose which attribute to be shown on the X axis (other will go to the Y axis automatically) with the help of a radio button and adjusts the chart accordingly**

In the shown example, we drew the plot between Salaries and Ratings of the players.

We can see that players with higher ratings tend to have more salaries (which clearly makes sense). Plus the graph is dense at low salaries/ratings and scattered at the high salary/rating ranges since there are very few players with high ratings (as inferred in the histogram example).

This clearly explains how Data Visualizations help us infer data in an easy way.



Code Snippets

The Axes and the Grid

The X axis and Y axis are drawn with respect to the input data using d3.js

```
var yScale = d3.scaleLinear()
    .domain([0, d3.max(data, function(d, i){return d.val})])
    .range([390,0]).nice();

var xScale = d3.scaleBand()
    .domain(d3.sort(data.map(function(d){return d.key})))
    .range([0, 1150])
    .padding(.2)

const svg = d3.select(this.refs.chart)
    .append("svg")
    .classed("bar-svg", true)
    .attr("width", 1200)
    .attr("height", 550)

var grid = svg.append("g")
    .attr("class", "grid")
    .attr(
        'transform', 'translate(50, 10)'
    )
    .call(d3.axisLeft(yScale)
    .tickSize(-(1150))
    .tickFormat("")
    )

var xAxis = svg.append("g")
```

```
    .classed("xAxis", true)
    .attr(
        'transform', 'translate(50,400)'
    )
    .attr('fill', 'red')
    .call(d3.axisBottom(xScale))
    .selectAll("text")
    .style("text-anchor", "end")
    .attr("dx", "-1em")
    .attr("dy", "-0.5em")
    .attr("transform", "rotate(-90)")

var yAxis = svg.append("g")
    .classed("yAxis", true)
    .attr(
        'transform', 'translate(50, 10)'
    )
    .call(d3.axisLeft(yScale))
```

The Bar Chart

To draw the Bar Chart, we create bars i.e append "rect" in based on the incoming data. The placement of the bars depend on the data and corresponding X and Y scales defined above.

```
var rectGrp = svg.append("g").attr(
        'transform', 'translate(50, 10)'
)
var index = d3.local()
rectGrp.selectAll("rect").data(data).enter()
    .append("rect")
    .attr("width", xScale.bandwidth())
    .attr("height", function(d, i) {
        return 0
    })
    .attr("x", function(d, i){
        index.set(this, i)
        return xScale(d.key)
    })
    .attr("y", function(d, i){
        return 390
    })
    .attr("class", "unselected_bar")
    .on("mouseover", onMouseOver)
    .on("mouseout", onMouseOut)
    .transition()
    .ease(d3.easeLinear)
    .duration(600)
    .delay(function(d, i){
        return i*15;
    })
    .attr("y", function(d, i){
        return yScale(d.val)
```

```
    })
    .attr("height", function(d, i) {
        return 390-yScale(d.val)
    })
```

Histogram

The drawing of the histogram is very similar to the Bar Chart, but here we have to define the number of bins. The number of bins are dynamic (change on mouse drag but start with some default value). Now, the input data to d3 is updated to take the number of bins into picture and then the X scales and Y scales are created.

```
const data = this.props.data

var bins = this.state.bins;

var binSize = (d3.max(data) - d3.min(data))/bins

var freq = new Array(bins).fill(0)

data.forEach(element => {
    freq[Math.floor((element - d3.min(data))/binSize)]++
});

var ranges = []
var min = d3.min(data)

for (var i = 0; i<bins; i++){
    var end = (+min + +binSize).toFixed(1);
    ranges.push(min)
    min = end
}
ranges.push(min)

var yScale = d3.scaleLinear()
    .domain([0, d3.max(freq)])
    .range([390,0]).nice();

var xScale = d3.scaleBand()
    .domain(ranges)
    .range([0, 1150])
    .padding(0)
```

Highlighting the Bars

The "onmouseover" and "onmouseout" events are used to change the CSS of the bars to highlight/unhighlight and to append the value in the svg

```
function onMouseOver(d, a) {
    d3.select(this).attr('class', 'selected_bar')

    var i = index.get(this)
    rectGrp.append("text")
    .attr("class", "val")
```

```
    .attr("x", function() {
        return xScale(a.key) + (xScale.bandwidth())/2
    })
    .attr("y", function() {
        return yScale(a.val) - 10;
    })
    .text(function(){
        console.log(i)
        return a.val
    })
    .attr("fill", "#82FC6B")
    .attr("font-size", "20")
    .attr("font-family", "Archia")
    .style("backgroud-color", "white")
}

function onMouseOut(d, a) {
    d3.select(this).attr('class', 'unselected_bar')
    d3.selectAll(".val")
    .remove()
    .exit()
}
```

Increasing/Decreasing the Bin Size

The Bin size is to be changes on "mousedown" and then "mousemove". If the mouse moves right, no. of bins are decreased (bin size increases). And when it moves left, no. of bins increase. So we need to maintain the X coordinate of the pointer to find the direction and a bool to only change if the mouse move is done on a clicked mouse

```
const mouseDownFunc = (event) => {
    var mouseDown = true
    var xDown = event.pageX;

    const mousemove = (e) => {
        var oldBins = this.state.bins
        if (mouseDown){
            var xNow = e.pageX
            if (xDown - xNow > 10){
                if (oldBins<=21){
                    this.setState({bins: oldBins+1})
                    xDown = xNow
                }

            }else if(xNow - xDown > 10){
                if (oldBins>2){
                    this.setState({bins: oldBins-1})
                    xDown = xNow
                }
            }
        }

    }
```

```
    d3.select(window)
    .on("mousemove", mousemove)
    .on("mouseup", mouseup)

    event.preventDefault()

    function mouseup() {
        mouseDown = false
    }
}
```

Scatter Plot

This takes two data as input and draws circles instead of rectangles. So, the center of the circle depends on the x and y scales defined.

```
scatter.append("g")
    .selectAll("dot")
    .data(xData)
    .enter()
    .append("circle")
    .attr("cx", function(d, i){return xScale(d)})
    .attr("cy", function(d, i){return yScale(yData[i])})
    .transition()
    .ease(d3.easeLinear)
    .duration(0.5)
    .delay(function(d, i){
        return i*1;
    })
    .attr("r", 3)
    .style("fill", "#01F1E4")
```