Name	Sakshi Lonare
Class	BE Computer Engineering (Batch E)
UID	2021300069
Exp No.	7

Aim: Creating Visualizations using D3.js on a Finance Dataset

Objectives:

- To explore and visualize a dataset related to Finance/ Banking/ Insurance/ Credit using D3.js.
- To create basic visualizations (Bar chart, Pie chart, Histogram, Timeline chart, Scatter plot, Bubble plot) to understand data distribution and trends.
- To create advanced visualizations (Word chart, Box and Whisker plot, Violin plot, Regression plot, 3D chart, Jitter) for deeper insights and complex relationships.
- To perform hypothesis testing using the Pearson correlation coefficient to evaluate relationships between numerical variables in the dataset.

Description:

Dataset used is Insurance Dataset available at

https://www.kaggle.com/datasets/andrewmvd/udemy-courses

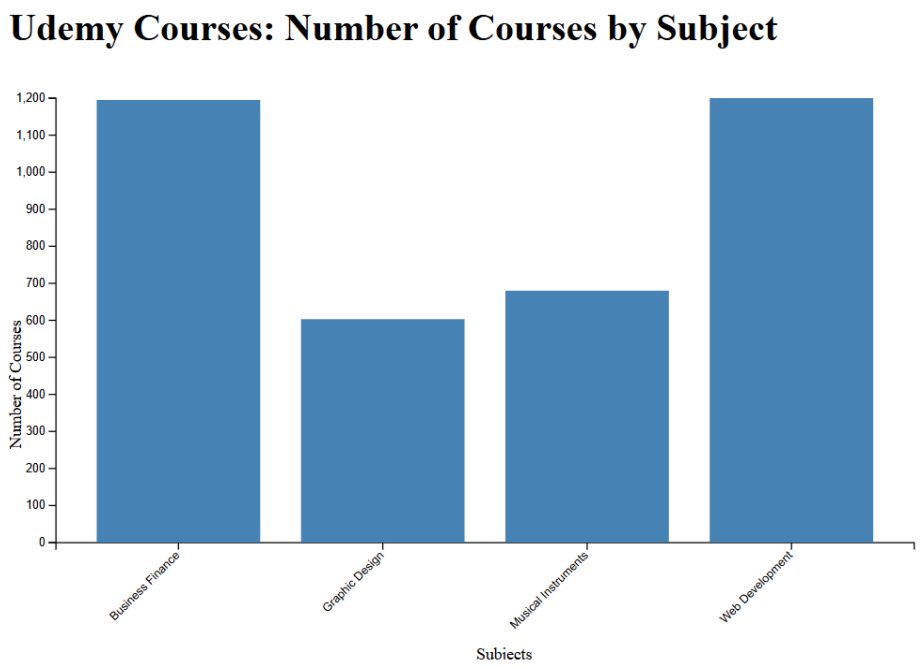
This dataset contains 3.682 records of courses from 4 subjects (Business Finance, Graphic Design, Musical Instruments and Web Design) taken from Udemy.

Udemy is a massive online open course (MOOC) platform that offers both free and paid courses. Anybody can create a course, a business model by which allowed Udemy to have hundreds of thousands of courses.

This version modifies column names, removes empty columns and aggregates everything into a single csv file for ease of use.

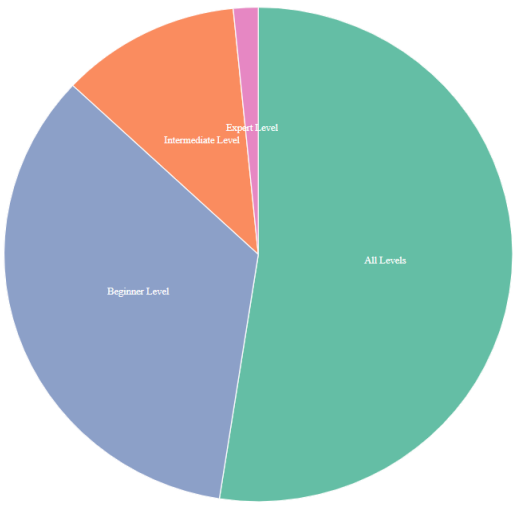
Graphs and Observations:

Bar chart:



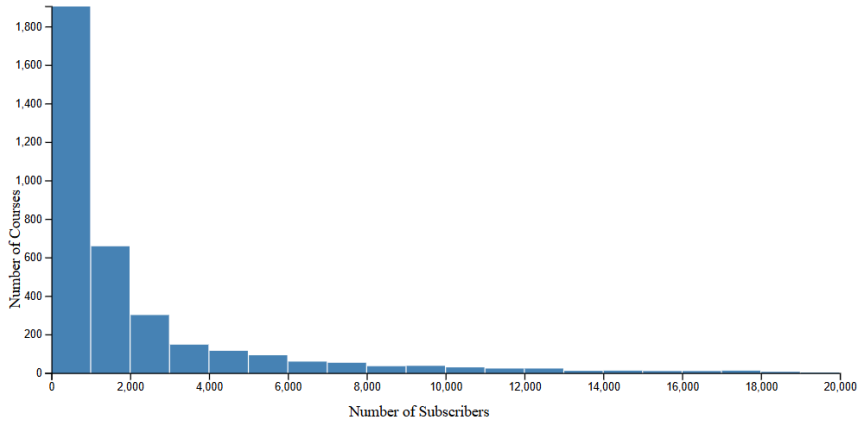
Observation: The bar plot illustrates the number of courses offered by Udemy across four subjects: Business Finance, Graphic Design, Musical Instruments, and Web Development. It's evident that Business Finance and Web Development are the most popular subjects, with a significantly higher number of courses compared to Graphic Design and Musical Instruments. This suggests a greater demand and interest in these two fields on the platform

Pie Chart:



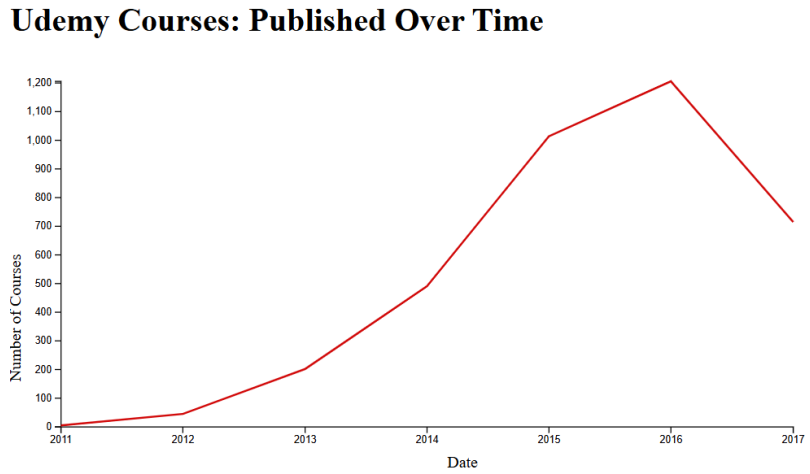
Observation: The pie chart illustrates the distribution of course levels on Udemy. The majority of courses are categorized as "All Levels," indicating a broad appeal to learners of various skill levels. "Beginner Level" courses constitute a significant portion as well, suggesting a strong focus on catering to those new to the subject matter. "Intermediate Level" and "Expert Level" courses make up smaller segments, indicating a more targeted approach for learners with some prior knowledge.

Histogram:



Observation: The histogram displays the distribution of courses based on the number of subscribers. It's evident that a large number of courses have relatively few subscribers, as seen by the tall bars at the left end of the graph. As the number of subscribers increases, the number of courses decreases. This suggests a long-tailed distribution, where a few courses have a very large number of subscribers, while most have a smaller subscriber base.

Line Chat:

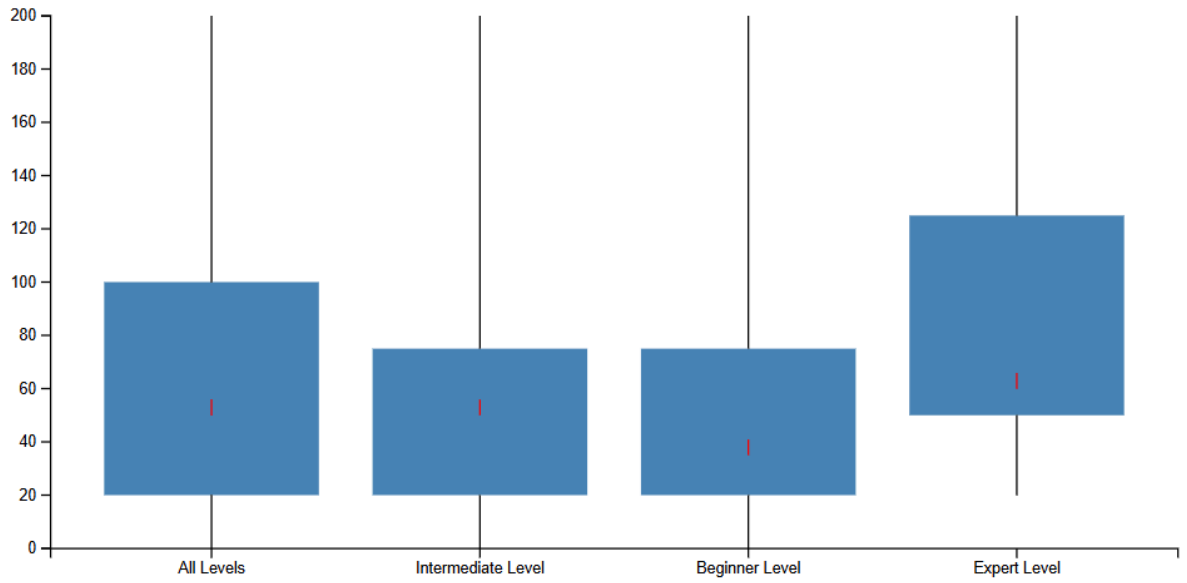


Observation: The line chart illustrates the number of Udemy courses published over time from 2011 to 2017. It shows a clear upward trend, indicating that the platform's course offerings have grown significantly during this period. The growth rate seems to have accelerated in the later years, with a steeper increase in the number of courses published.

Word Chart:

Observation: The word cloud primarily focuses on financial and trading-related topics, highlighting the popularity of these subjects on the platform. Words like "trading," "finance," "investment," and "stock" dominate the visualization, indicating a strong emphasis on these areas.

Box Plot:



Observation: The box plot illustrates the distribution of prices across different course levels on Udemy. It appears that "All Levels" courses tend to have a higher median price compared to the other levels.

Code in D3.js:

Bar Plot:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Udemy Courses Bar Chart</title>
  <script src="https://d3js.org/d3.v7.min.js"></script>
  <script src="d3ADV.js" defer></script>
  <style>
    .bar {
      fill: steelblue;
    }
    .bar:hover {
```

```
      fill: orange;
    }
    .axis-label {
      font-size: 14px;
    }
  </style>
</head>
<body>
  <h1>Udemy Courses: Number of Courses by Subject</h1>
  <div id="chart"></div>
</body>
</html>
const margin = {top: 20, right: 30, bottom: 100, left: 40},
      width = 800 - margin.left - margin.right,
      height = 500 - margin.top - margin.bottom;

const svg = d3.select("#chart")
  .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('udemy_courses.csv').then(data => {

  const subjectCounts = d3.rollup(data, v => v.length, d => d.subject);

  const subjectData = Array.from(subjectCounts, ([subject, count]) =>
({subject, count}));
```

```
const x = d3.scaleBand()
  .domain(subjectData.map(d => d.subject))
  .range([0, width])
  .padding(0.2);

const y = d3.scaleLinear()
  .domain([0, d3.max(subjectData, d => d.count)])
  .nice()
  .range([height, 0]);

svg.append("g")
  .attr("transform", `translate(0,${height})`)
  .call(d3.axisBottom(x))
  .selectAll("text")
    .attr("transform", "translate(-10,0)rotate(-45)")
    .style("text-anchor", "end");

// Add the y-axis
svg.append("g")
  .call(d3.axisLeft(y));

// Create the bars
svg.selectAll(".bar")
  .data(subjectData)
  .enter()
  .append("rect")
    .attr("class", "bar")
    .attr("x", d => x(d.subject))
```

```
      .attr("y", d => y(d.count))
      .attr("width", x.bandwidth())
      .attr("height", d => height - y(d.count));



  svg.append("text")
    .attr("class", "axis-label")
    .attr("text-anchor", "end")
    .attr("x", width / 2 + margin.left)
    .attr("y", height + margin.bottom)
    .text("Subjects");


  svg.append("text")
    .attr("class", "axis-label")
    .attr("text-anchor", "end")
    .attr("x", -height / 2)
    .attr("y", -margin.left + 10)
    .attr("transform", "rotate(-90)")
    .text("Number of Courses");
});
```

Pie chart

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Udemy Courses Bar Chart</title>
  <script src="https://d3js.org/d3.v7.min.js"></script>
```

```
  <script src="d3ADV.js" defer></script>
  <style>
    .arc text {
      font-size: 12px;
      fill: #fff;
    }
    .arc path {
      stroke: #fff;
    }
  </style>
</head>
<body>
  <h1>Udemy Courses: Distribution by Level</h1>
  <div id="chart"></div>
</body>
</html>
const width = 650,
      height = 650,
      margin = 70;

// Radius for the pie chart
const radius = Math.min(width, height) / 2 - margin;

const svg = d3.select("#chart")
  .append("svg")
    .attr("width", width)
    .attr("height", height)
  .append("g")
    .attr("transform", `translate(${width / 2}, ${height / 2})`);
```

```
// Load the dataset
d3.csv('udemy_courses.csv').then(data => {

  const levelCounts = d3.rollup(data, v => v.length, d => d.level);

  const levelData = Array.from(levelCounts, ([level, count]) =>
({level, count}));

  const color = d3.scaleOrdinal()
    .domain(levelData.map(d => d.level))
    .range(d3.schemeSet2); // Set2 color scheme

  // Pie generator
  const pie = d3.pie()
    .value(d => d.count);

  const arc = d3.arc()
    .innerRadius(0)
    .outerRadius(radius);

  const arcs = svg.selectAll("arc")
    .data(pie(levelData))
    .enter()
    .append("g")
    .attr("class", "arc");

  arcs.append("path")
    .attr("d", arc)
```

```
    .attr("fill", d => color(d.data.level));


  // Add labels to the pie chart
  arcs.append("text")
    .attr("transform", d => `translate(${arc.centroid(d)})`)
    .text(d => d.data.level)
    .style("text-anchor", "middle")
    .style("font-size", "10px")
    .style("fill", "white");
});


<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Udemy Courses Bar Chart</title>
  <script src="https://d3js.org/d3.v7.min.js"></script>
  <script src="d3ADV.js" defer></script>
  <style>
    .bar {
      fill: steelblue;
    }
    .bar:hover {
      fill: orange;
    }
    .axis-label {
      font-size: 14px;
```

```
    }
  </style>
</head>
<body>
  <h1>Udemy Courses: Distribution of Number of Subscribers</h1>
  <div id="chart"></div>
</body>
</html>
const margin = {top: 20, right: 30, bottom: 40, left: 40},
      width = 800 - margin.left - margin.right,
      height = 400 - margin.top - margin.bottom;

const svg = d3.select("#chart")
  .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})`);

// Load the dataset
d3.csv('udemy_courses.csv').then(data => {

  data.forEach(d => {
    d.num_subscribers = +d.num_subscribers; // Convert string to number
  });

  const x = d3.scaleLinear()
    .domain([0, 20000])
    .range([0, width]);
```

```
  const histogram = d3.histogram()
    .value(d => d.num_subscribers)
    .domain(x.domain())
    .thresholds(x.ticks(20));



  const bins = histogram(data);


  const y = d3.scaleLinear()
    .domain([0, d3.max(bins, d => d.length)]) // Number of courses in
each bin
    .range([height, 0]);


  svg.selectAll(".bar")
    .data(bins)
    .enter()
    .append("rect")
      .attr("class", "bar")
      .attr("x", d => x(d.x0))
      .attr("y", d => y(d.length))
      .attr("width", d => x(d.x1) - x(d.x0) - 1)
      .attr("height", d => height - y(d.length));


  svg.append("g")
    .attr("transform", `translate(0,${height})`)
    .call(d3.axisBottom(x));


  svg.append("g")
```

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


  svg.append("text")
    .attr("class", "axis-label")
    .attr("text-anchor", "end")
    .attr("x", width / 2 + margin.left)
    .attr("y", height + margin.bottom)
    .text("Number of Subscribers");


  svg.append("text")
    .attr("class", "axis-label")
    .attr("text-anchor", "end")
    .attr("x", -height / 2)
    .attr("y", -margin.left + 10)
    .attr("transform", "rotate(-90)")
    .text("Number of Courses");
});


<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Udemy Courses Bar Chart</title>
  <script src="https://d3js.org/d3.v7.min.js"></script>
  <script src="d3ADV.js" defer></script>
  <style>
    .line {
```

```
      fill: none;
      stroke: rgb(212, 10, 10);
      stroke-width: 2px;
    }
  </style>
</head>
<body>
  <h1>Udemy Courses: Published Over Time</h1>
  <div id="chart"></div>
</body>
</html>
const margin = {top: 20, right: 30, bottom: 40, left: 50},
      width = 800 - margin.left - margin.right,
      height = 400 - margin.top - margin.bottom;

const svg = d3.select("#chart")
  .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('udemy_courses.csv').then(data => {

    const parseDate = d3.timeParse("%Y-%m-%dT%H:%M:%SZ");

  data.forEach(d => {
    d.published_timestamp = parseDate(d.published_timestamp);
  });
```

```
  // Group the data by date (month) and count the number of courses per
date
  const coursesByDate = d3.rollup(
    data,
    v => v.length, // Count courses for each date
    d => d3.timeYear(d.published_timestamp) // Group by month
  );

  const timeData = Array.from(coursesByDate, ([date, count]) => ({date,
count}));

  timeData.sort((a, b) => a.date - b.date);



  const x = d3.scaleTime()
    .domain(d3.extent(timeData, d => d.date))
    .range([0, width]);



  const y = d3.scaleLinear()
    .domain([0, d3.max(timeData, d => d.count)])
    .range([height, 0]);

  svg.append("g")
    .attr("transform", `translate(0,${height})`)
    .call(d3.axisBottom(x).ticks(d3.timeYear.every(1)));



  svg.append("g")
```

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



const line = d3.line()
  .x(d => x(d.date))
  .y(d => y(d.count));



svg.append("path")
  .datum(timeData)
  .attr("class", "line")
  .attr("d", line)
  .attr("fill", "none")
  .attr("stroke", "steelblue")
  .attr("stroke-width", 2);


svg.append("text")
  .attr("class", "axis-label")
  .attr("text-anchor", "end")
  .attr("x", width / 2 + margin.left)
  .attr("y", height + margin.bottom)
  .text("Date");


svg.append("text")
  .attr("class", "axis-label")
  .attr("text-anchor", "end")
  .attr("x", -height / 2)
  .attr("y", -margin.left + 10)
  .attr("transform", "rotate(-90)")
```

```
    .text("Number of Courses");
});
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Udemy Courses Bar Chart</title>
  <script src="https://d3js.org/d3.v7.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/d3-cloud/1.2.5/d3.layout.cl
oud.min.js"></script>
  <script src="d3ADV.js" defer></script>
  <style>
    .line {
      fill: none;
      stroke: rgb(212, 10, 10);
      stroke-width: 2px;
    }
  </style>
</head>
<body>
  <div id="word-cloud"></div>
</body>
</html>
const wordCloudWidth = 800;
const wordCloudHeight = 400;

const wordCloudSvg = d3.select("#word-cloud")
```

```
  .append("svg")
    .attr("width", wordCloudWidth)
    .attr("height", wordCloudHeight);

// Load the dataset
d3.csv('udemy_courses.csv').then(data => {
    const communicationCount = d3.rollup(
        data.filter(d => d["course_title"] && d["course_title"].trim()
!== ""),
        v => v.length,
        d => d["course_title"].trim()
    );

    console.log("Subject Count:", communicationCount);

    const wordData = Array.from(communicationCount, ([key, value]) =>
({text: key, size: value}));

    const maxSize = d3.max(wordData, d => d.size);
    const minSize = d3.min(wordData, d => d.size);

    // Create a scale for the font size based on word frequency
    const fontSizeScale = d3.scaleLinear()
                            .domain([minSize, maxSize])
                            .range([10, 100]);

    d3.layout.cloud()
      .size([wordCloudWidth, wordCloudHeight])
      .words(wordData)
      .padding(5)
```

```
      .rotate(() => ~~(Math.random() * 2) * 90)
      .fontSize(d => fontSizeScale(d.size))
      .on("end", draw)
      .start();

    // Draw the word cloud
    function draw(words) {
      d3.select("#word-cloud").select("svg")
        .append("g")
          .attr("transform", `translate(${wordCloudWidth / 2},${wordCloudHeight / 2})`)
        .selectAll("text")
          .data(words)
        .enter().append("text")
          .style("font-size", d => d.size + "px")
          .style("fill", () => "hsl(" + Math.random() * 360 + ",100%,50%)")
          .attr("text-anchor", "middle")
          .attr("transform", d => `translate(${[d.x, d.y]})rotate(${d.rotate})`)
          .text(d => d.text);
    }
}).catch(function(error) {
    console.error("Error loading the CSV file:", error);
});
```

Box Plot

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
```

```
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Udemy Courses Bar Chart</title>
  <script src="https://d3js.org/d3.v7.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/d3-cloud/1.2.5/d3.layout.cl
oud.min.js"></script>
  <script src="d3ADV.js" defer></script>
  <style>
    .line {
      fill: none;
      stroke: rgb(12, 1, 1);
      stroke-width: 2px;
    }
  </style>
</head>
<body>
    <div id="box-plot"></div>
</body>
</html>
const margin = { top: 20, right: 30, bottom: 40, left: 50 },
      width = 800 - margin.left - margin.right,
      height = 400 - margin.top - margin.bottom;

const svg = d3.select("#box-plot")
  .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('udemy_courses.csv').then(data => {
    data.forEach(d => {
        d.price = +d.price;
    });

    const nestedData = d3.group(data, d => d.level);
    const boxPlotData = Array.from(nestedData, ([key, values]) => {
        const prices = values.map(d => d.price);
        return {
            level: key,
            Q1: d3.quantile(prices.sort(d3.ascending), 0.25),
            median: d3.quantile(prices.sort(d3.ascending), 0.5),
            Q3: d3.quantile(prices.sort(d3.ascending), 0.75),
            IQR: d3.quantile(prices.sort(d3.ascending), 0.75) -
d3.quantile(prices.sort(d3.ascending), 0.25),
            min: d3.min(prices),
            max: d3.max(prices),
        };
    });

    // Set up scales
    const x = d3.scaleBand()
        .domain(boxPlotData.map(d => d.level))
        .range([0, width])
        .padding(0.2);

    const y = d3.scaleLinear()
        .domain([0, d3.max(boxPlotData, d => d.max)]) // Adjust max if
necessary
```

```
      .nice()
      .range([height, 0]);

  svg.append("g")
    .attr("transform", `translate(0,${height})`)
    .call(d3.axisBottom(x));

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

  // Create box plots
  svg.selectAll(".box")
    .data(boxPlotData)
    .enter().append("rect")
      .attr("class", "box")
      .attr("x", d => x(d.level))
      .attr("y", d => y(d.Q3))
      .attr("height", d => y(d.Q1) - y(d.Q3))
      .attr("width", x.bandwidth())
      .style("fill", "steelblue");

  // Add median lines
  svg.selectAll(".median")
    .data(boxPlotData)
    .enter().append("line")
      .attr("class", "median")
      .attr("x1", d => x(d.level) + x.bandwidth() / 2)
      .attr("x2", d => x(d.level) + x.bandwidth() / 2)
      .attr("y1", d => y(d.median))
```

```
        .attr("y2", d => y(d.median) - 10) // Length of the median line
        .style("stroke", "red");


    // Add whiskers
    svg.selectAll(".whisker")
      .data(boxPlotData)
      .enter().append("line")
        .attr("class", "whisker")
        .attr("x1", d => x(d.level) + x.bandwidth() / 2)
        .attr("x2", d => x(d.level) + x.bandwidth() / 2)
        .attr("y1", d => y(d.min))
        .attr("y2", d => y(d.Q1))
        .style("stroke", "black");


    svg.selectAll(".whiskerMax")
      .data(boxPlotData)
      .enter().append("line")
        .attr("class", "whiskerMax")
        .attr("x1", d => x(d.level) + x.bandwidth() / 2)
        .attr("x2", d => x(d.level) + x.bandwidth() / 2)
        .attr("y1", d => y(d.Q3))
        .attr("y2", d => y(d.max))
        .style("stroke", "black");
});
```

Conclusion:

Through this experiment, we gained valuable insights into D3.js and its powerful capabilities for data visualization. We explored how to effectively plot various types of graphs, including bar charts, line plots, histograms, and more.