Sardar Patel Institute of Technology

Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058-India

(Autonomous College Affiliated to University of Mumbai)

ADVANCED DATA VISUALIZATION
EXPERIMENT 7

NAME: Bhagya Bijlaney
ROLL NO: 2021700010
BATCH: L

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.

DATASET:
https://www.kaggle.com/datasets/zhijinzhai/loandata

Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Loan Data Visualization</title>
    <script src="https://d3js.org/d3.v7.min.js"></script>
    <style>
        body { font-family: Arial, sans-serif; }
        .chart { margin-bottom: 50px; }
    </style>
</head>
<body>
    <h1>Loan Data Visualization</h1>

    <div id="barChart" class="chart">
        <h2>Loan Status Distribution</h2>
    </div>

    <div id="pieChart" class="chart">
        <h2>Gender Distribution</h2>
    </div>

    <div id="histogram" class="chart">
        <h2>Principal Amount Distribution</h2>
    </div>

    <div id="scatterPlot" class="chart">
        <h2>Age vs Principal Amount</h2>
    </div>

    <div id="timelineChart" class="chart">
        <h2>Loan Amount Over Time</h2>
    </div>



    <div id="boxPlot" class="chart">
        <h2>Principal Distribution by Education</h2>
    </div>



    <script>
    // Load the JSON data
```

```
    d3.json("csvjson.json").then(function(loanData) {
        // Convert string dates to Date objects and ensure numeric values
        loanData.forEach(d => {
            d.effective_date = new Date(d.effective_date);
            d.due_date = new Date(d.due_date);
            d.paid_off_time = d.paid_off_time ? new Date(d.paid_off_time)
: null;
            d.Principal = +d.Principal;
            d.terms = +d.terms;
            d.age = +d.age;
            d.past_due_days = d.past_due_days ? +d.past_due_days : null;
        });

        // Call the functions to create the charts
        createBarChart(loanData);
        createPieChart(loanData);
        createHistogram(loanData);
        createScatterPlot(loanData);
        createTimelineChart(loanData);

        createBoxPlot(loanData);
        createViolinPlot(loanData);

        // Call the new functions
        createRegressionPlot(loanData);
        create3DChart(loanData);
        createJitterPlot(loanData);
        performHypothesisTesting(loanData);
    }).catch(function(error) {
        console.log("Error loading the JSON file:", error);
    });


    // Add these new functions after your existing chart functions but
before the d3.json() call

// Regression Plot: Age vs Principal Amount
function createRegressionPlot(data) {
    const width = 400;
    const height = 300;
```

```
    const margin = {top: 20, right: 20, bottom: 30, left: 40};

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

    const x = d3.scaleLinear().range([0, width]);
    const y = d3.scaleLinear().range([height, 0]);

    x.domain(d3.extent(data, d => d.age));
    y.domain(d3.extent(data, d => d.Principal));

    svg.selectAll("circle")
        .data(data)
        .enter().append("circle")
        .attr("cx", d => x(d.age))
        .attr("cy", d => y(d.Principal))
        .attr("r", 3)
        .attr("fill", "steelblue");

    // Calculate regression line
    const regression = d3.regressionLinear()
        .x(d => d.age)
        .y(d => d.Principal);

    const regressionLine = regression(data);

    svg.append("line")
        .attr("x1", x(regressionLine[0][0]))
        .attr("y1", y(regressionLine[0][1]))
        .attr("x2", x(regressionLine[1][0]))
        .attr("y2", y(regressionLine[1][1]))
        .attr("stroke", "red");

    svg.append("g")
        .attr("transform", `translate(0,${height})`)
        .call(d3.axisBottom(x).ticks(5));
```

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

    svg.append("text")
        .attr("x", width / 2)
        .attr("y", height + margin.bottom)
        .style("text-anchor", "middle")
        .text("Age");

    svg.append("text")
        .attr("transform", "rotate(-90)")
        .attr("y", 0 - margin.left)
        .attr("x", 0 - (height / 2))
        .attr("dy", "1em")
        .style("text-anchor", "middle")
        .text("Principal Amount");
}

// 3D Chart: Age, Principal Amount, and Term
function create3DChart(data) {
    const width = 500;
    const height = 500;
    const margin = {top: 20, right: 20, bottom: 30, left: 40};

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

    const x = d3.scaleLinear().range([0, width]);
    const y = d3.scaleLinear().range([height, 0]);
    const z = d3.scaleLinear().range([2, 20]);
    const color = d3.scaleLinear().range(["blue", "red"]);

    x.domain(d3.extent(data, d => d.age));
    y.domain(d3.extent(data, d => d.Principal));
    z.domain(d3.extent(data, d => d.terms));
```

```
    color.domain(d3.extent(data, d => d.terms));

    svg.selectAll("circle")
        .data(data)
        .enter().append("circle")
        .attr("cx", d => x(d.age))
        .attr("cy", d => y(d.Principal))
        .attr("r", d => z(d.terms))
        .attr("fill", d => color(d.terms))
        .attr("opacity", 0.7);

    svg.append("g")
        .attr("transform", `translate(0,${height})`)
        .call(d3.axisBottom(x).ticks(5));

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

    svg.append("text")
        .attr("x", width / 2)
        .attr("y", height + margin.bottom)
        .style("text-anchor", "middle")
        .text("Age");

    svg.append("text")
        .attr("transform", "rotate(-90)")
        .attr("y", 0 - margin.left)
        .attr("x", 0 - (height / 2))
        .attr("dy", "1em")
        .style("text-anchor", "middle")
        .text("Principal Amount");
}

// Jitter Plot: Loan Status vs Principal Amount
function createJitterPlot(data) {
    const width = 400;
    const height = 300;
    const margin = {top: 20, right: 20, bottom: 30, left: 40};

    const svg = d3.select("#jitterPlot")
```

```
        .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})`);

    const x = d3.scaleBand().range([0, width]).padding(0.1);
    const y = d3.scaleLinear().range([height, 0]);

    x.domain(data.map(d => d.loan_status));
    y.domain([0, d3.max(data, d => d.Principal)]);

    svg.selectAll("circle")
        .data(data)
        .enter().append("circle")
        .attr("cx", d => x(d.loan_status) + x.bandwidth() / 2 +
(Math.random() - 0.5) * x.bandwidth() * 0.8)
        .attr("cy", d => y(d.Principal))
        .attr("r", 3)
        .attr("fill", "steelblue");

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

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

    svg.append("text")
        .attr("x", width / 2)
        .attr("y", height + margin.bottom)
        .style("text-anchor", "middle")
        .text("Loan Status");

    svg.append("text")
        .attr("transform", "rotate(-90)")
        .attr("y", 0 - margin.left)
        .attr("x", 0 - (height / 2))
        .attr("dy", "1em")
        .style("text-anchor", "middle")
```

```
        .text("Principal Amount");
}

// Hypothesis Testing: Pearson Correlation Coefficient
function performHypothesisTesting(data) {
    // Step 1: Formulate hypothesis
    console.log("Hypothesis: There is a positive correlation between
customer age and loan amount.");

    // Step 2: Calculate Pearson correlation coefficient
    const age = data.map(d => d.age);
    const principal = data.map(d => d.Principal);

    const meanAge = d3.mean(age);
    const meanPrincipal = d3.mean(principal);

    const numerator = d3.sum(age.map((a, i) => (a - meanAge) *
(principal[i] - meanPrincipal)));
    const denominator = Math.sqrt(
        d3.sum(age.map(a => Math.pow(a - meanAge, 2))) *
        d3.sum(principal.map(p => Math.pow(p - meanPrincipal, 2)))
    );

    const correlationCoefficient = numerator / denominator;

    // Step 3: Interpret results
    console.log(`Pearson Correlation Coefficient:
${correlationCoefficient.toFixed(4)}`);
    if (correlationCoefficient > 0.7) {
        console.log("There is a strong positive correlation between
customer age and loan amount.");
    } else if (correlationCoefficient > 0.3) {
        console.log("There is a moderate positive correlation between
customer age and loan amount.");
    } else if (correlationCoefficient > 0) {
        console.log("There is a weak positive correlation between customer
age and loan amount.");
    } else if (correlationCoefficient < 0) {
        console.log("There is a negative correlation between customer age
and loan amount.");
```

```
    } else {
        console.log("There is no correlation between customer age and loan amount.");
    }
}

// Modify your existing d3.json() call to include these new functions:


    function createBoxPlot(data) {
        const width = 500;
        const height = 400;
        const margin = {top: 20, right: 20, bottom: 30, left: 40};

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

        const educationGroups = d3.group(data, d => d.education);

        const x = d3.scaleBand()
            .range([0, width])
            .domain(Array.from(educationGroups.keys()))
            .padding(0.1);

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

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

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

        const boxWidth = 50;
```

```
    educationGroups.forEach((group, key) => {
        const values = group.map(d => d.Principal).sort(d3.ascending);
        const q1 = d3.quantile(values, 0.25);
        const median = d3.quantile(values, 0.5);
        const q3 = d3.quantile(values, 0.75);
        const interQuantileRange = q3 - q1;
        const min = q1 - 1.5 * interQuantileRange;
        const max = q3 + 1.5 * interQuantileRange;

        svg.append("line")
            .attr("x1", x(key))
            .attr("x2", x(key))
            .attr("y1", y(min))
            .attr("y2", y(max))
            .attr("stroke", "black");

        svg.append("rect")
            .attr("x", x(key) - boxWidth / 2)
            .attr("y", y(q3))
            .attr("height", y(q1) - y(q3))
            .attr("width", boxWidth)
            .attr("stroke", "black")
            .attr("fill", "#69b3a2");

        svg.selectAll("myViolin")
            .data([group])
            .enter()
            .append("g")
            .attr("transform", `translate(${x(key)},0)`)
            .append("path")
            .datum(d => d.map(g => g.Principal))
            .attr("d", d3.area()
                .x0(0)
                .x1(function(d) { return d * 2; })
                .y(y)
                .curve(d3.curveCatmullRom)
            )
            .attr("stroke", "none")
            .attr("fill", "#69b3a2")
```

```
                .attr("opacity", 0.6);

            svg.append("line")
                .attr("x1", x(key) - boxWidth / 2)
                .attr("x2", x(key) + boxWidth / 2)
                .attr("y1", y(median))
                .attr("y2", y(median))
                .attr("stroke", "black");
        });
    }



    // Bar Chart: Loan Status Distribution
    function createBarChart(data) {
        const width = 400;
        const height = 300;
        const margin = {top: 20, right: 20, bottom: 30, left: 40};

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

        const statusCounts = d3.rollup(data, v => v.length, d =>
d.loan_status);
        const chartData = Array.from(statusCounts, ([key, value]) =>
({status: key, count: value}));

        const x = d3.scaleBand()
            .range([0, width])
            .padding(0.1);
        const y = d3.scaleLinear()
            .range([height, 0]);

        x.domain(chartData.map(d => d.status));
        y.domain([0, d3.max(chartData, d => d.count)]);
```

```
        svg.selectAll(".bar")
            .data(chartData)
            .enter().append("rect")
            .attr("class", "bar")
            .attr("x", d => x(d.status))
            .attr("width", x.bandwidth())
            .attr("y", d => y(d.count))
            .attr("height", d => height - y(d.count))
            .attr("fill", "steelblue");

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

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

    // Pie Chart: Gender Distribution
    function createPieChart(data) {
        const width = 400;
        const height = 400;
        const radius = Math.min(width, height) / 2;

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

        const genderCounts = d3.rollup(data, v => v.length, d =>
d.Gender);
        const chartData = Array.from(genderCounts, ([key, value]) =>
({gender: key, count: value}));

        const color = d3.scaleOrdinal()
            .domain(chartData.map(d => d.gender))
            .range(d3.schemeCategory10);
```

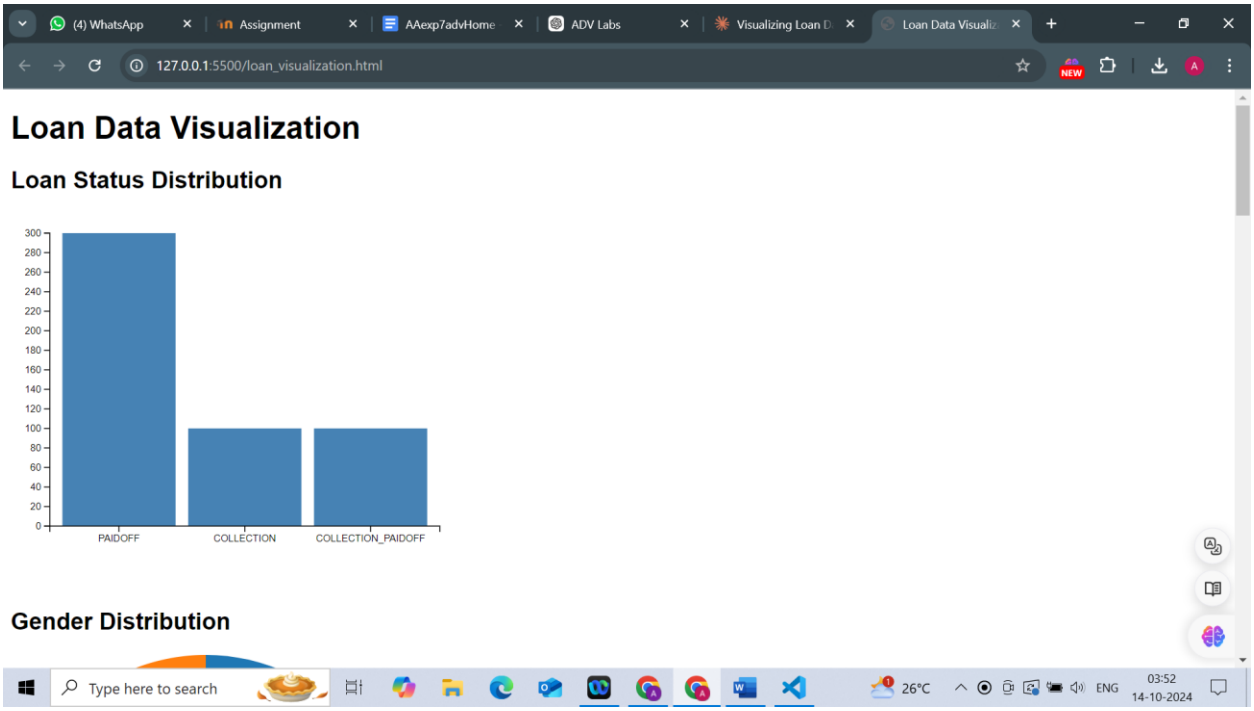
```
        const pie = d3.pie()
            .value(d => d.count);

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

        const arcs = svg.selectAll("arc")
            .data(pie(chartData))
            .enter()
            .append("g");

        arcs.append("path")
            .attr("d", arc)
            .attr("fill", d => color(d.data.gender));

        arcs.append("text")
            .attr("transform", d => `translate(${arc.centroid(d)})`)
            .attr("text-anchor", "middle")
            .text(d => d.data.gender);
    }

    // Histogram: Principal Amount Distribution
    function createHistogram(data) {
        const width = 400;
        const height = 300;
        const margin = {top: 20, right: 20, bottom: 30, left: 40};

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

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

        const y = d3.scaleLinear()
            .range([height, 0]);
```

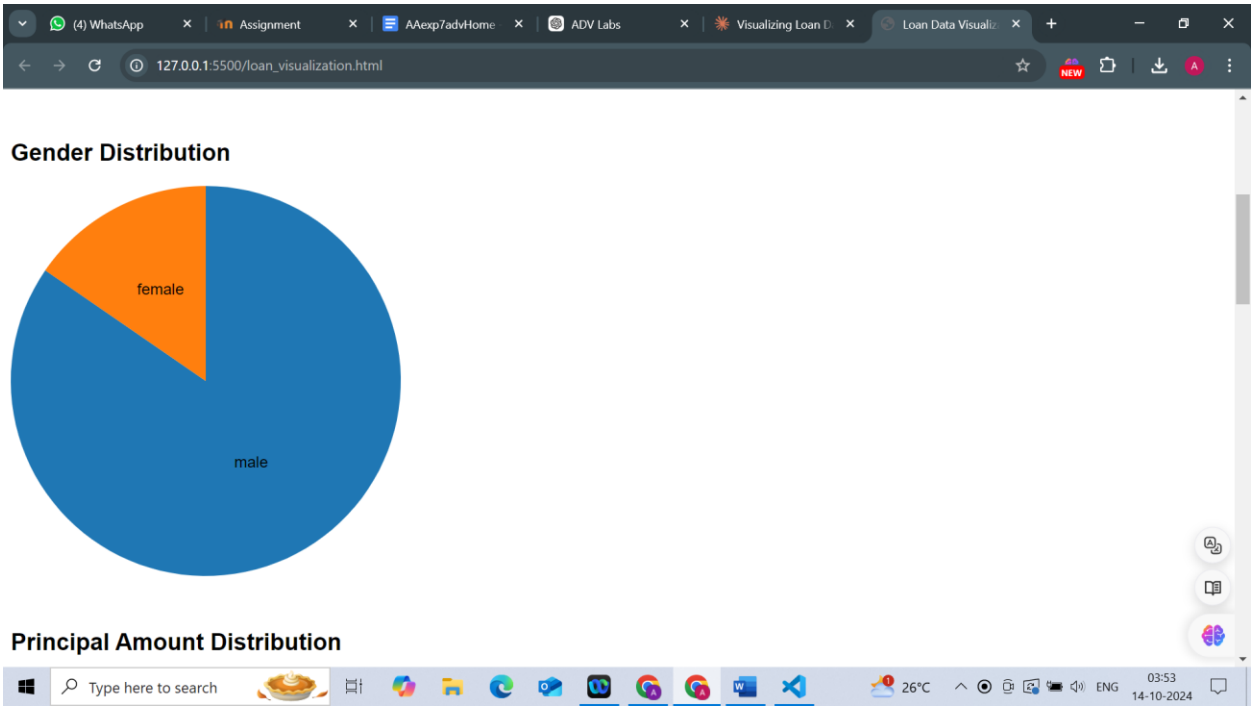
```
        const histogram = d3.histogram()
            .value(d => d.Principal)
            .domain(x.domain())
            .thresholds(x.ticks(20));

        const bins = histogram(data);

        x.domain([d3.min(data, d => d.Principal), d3.max(data, d =>
d.Principal)]);
        y.domain([0, d3.max(bins, d => d.length)]);

        svg.selectAll("rect")
            .data(bins)
            .enter().append("rect")
            .attr("x", d => x(d.x0) + 1)
            .attr("width", d => Math.max(0, x(d.x1) - x(d.x0) - 1))
            .attr("y", d => y(d.length))
            .attr("height", d => height - y(d.length))
            .attr("fill", "steelblue");

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

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

    // Scatter Plot: Age vs Principal Amount
    function createScatterPlot(data) {
        const width = 400;
        const height = 300;
        const margin = {top: 20, right: 20, bottom: 30, left: 40};

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

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

        const y = d3.scaleLinear()
            .range([height, 0]);

        x.domain(d3.extent(data, d => d.age));
        y.domain(d3.extent(data, d => d.Principal));

        svg.selectAll("circle")
            .data(data)
            .enter().append("circle")
            .attr("cx", d => x(d.age))
            .attr("cy", d => y(d.Principal))
            .attr("r", 3)
            .attr("fill", "steelblue");

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

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

    // Timeline Chart: Loan Amount Over Time
    function createTimelineChart(data) {
        const width = 600;
        const height = 300;
        const margin = {top: 20, right: 20, bottom: 30, left: 50};

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

```
        const x = d3.scaleTime()
            .range([0, width]);

        const y = d3.scaleLinear()
            .range([height, 0]);

        x.domain(d3.extent(data, d => d.effective_date));
        y.domain([0, d3.max(data, d => d.Principal)]);

        svg.append("path")
            .datum(data)
            .attr("fill", "none")
            .attr("stroke", "steelblue")
            .attr("stroke-width", 1.5)
            .attr("d", d3.line()
                .x(d => x(d.effective_date))
                .y(d => y(d.Principal))
            );

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

        svg.append("g")
            .call(d3.axisLeft(y));
    }
    </script>
</body>
</html>
```

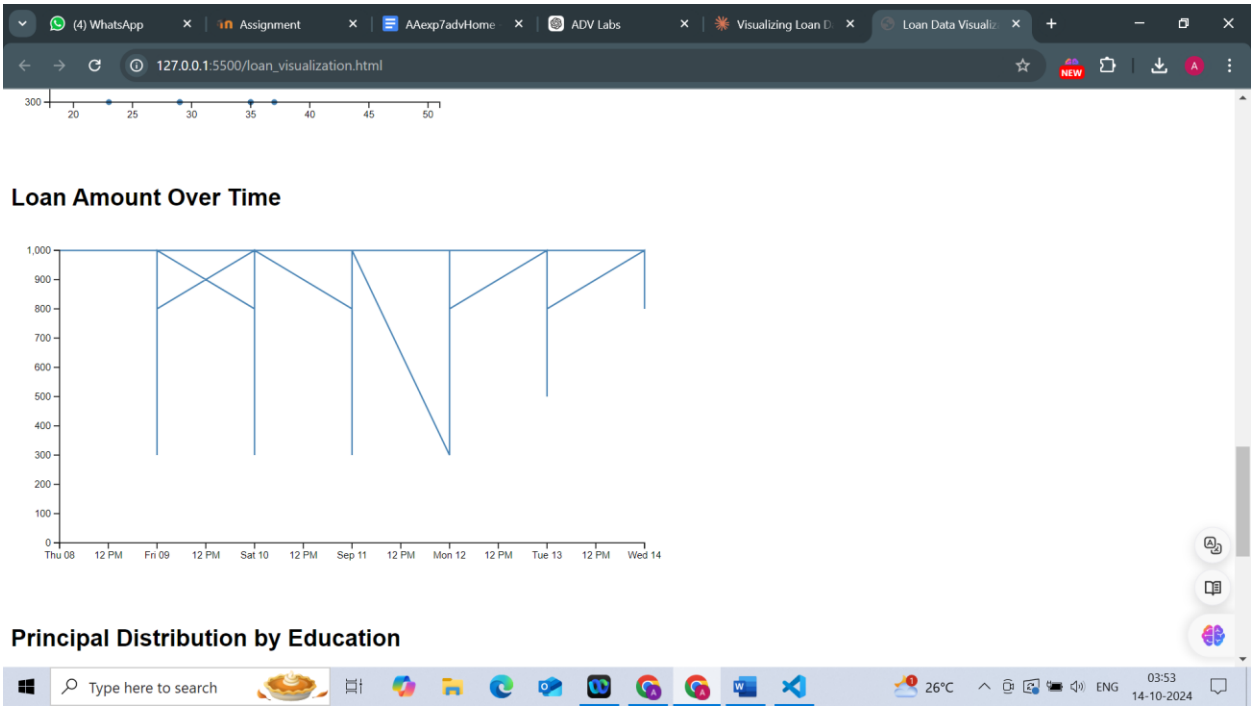
Output:

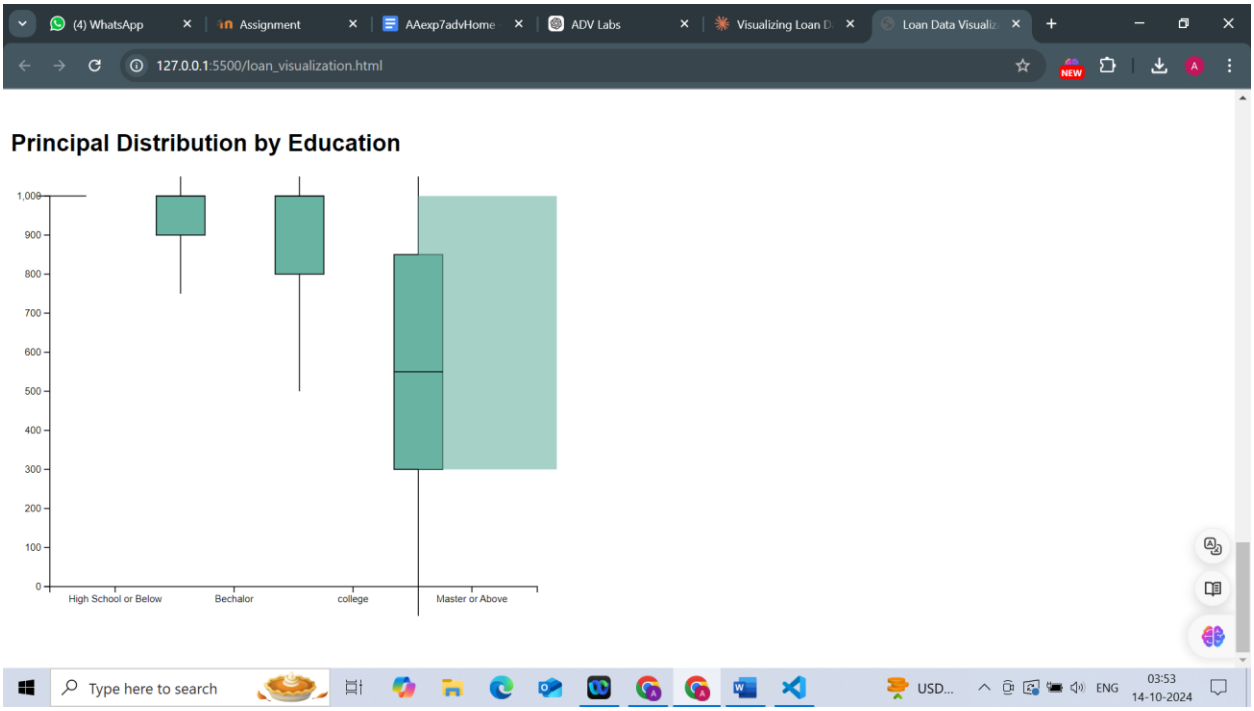


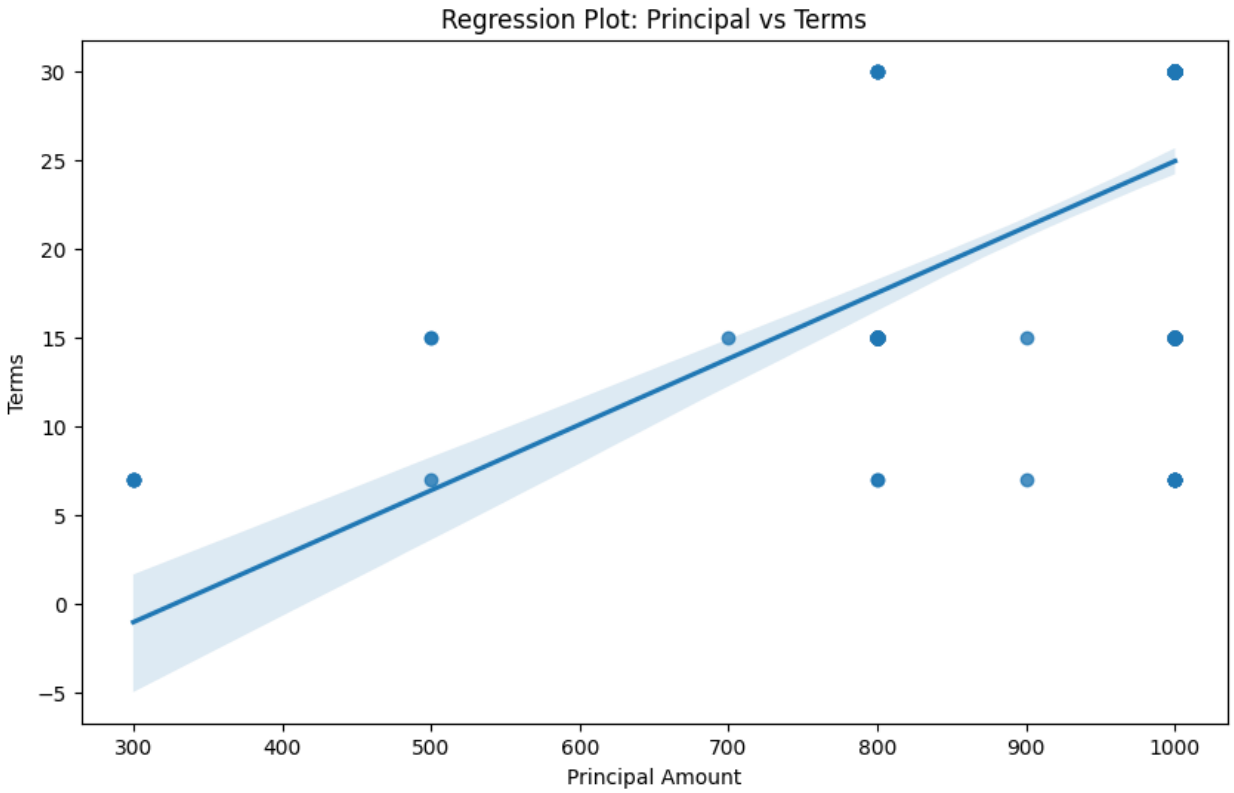


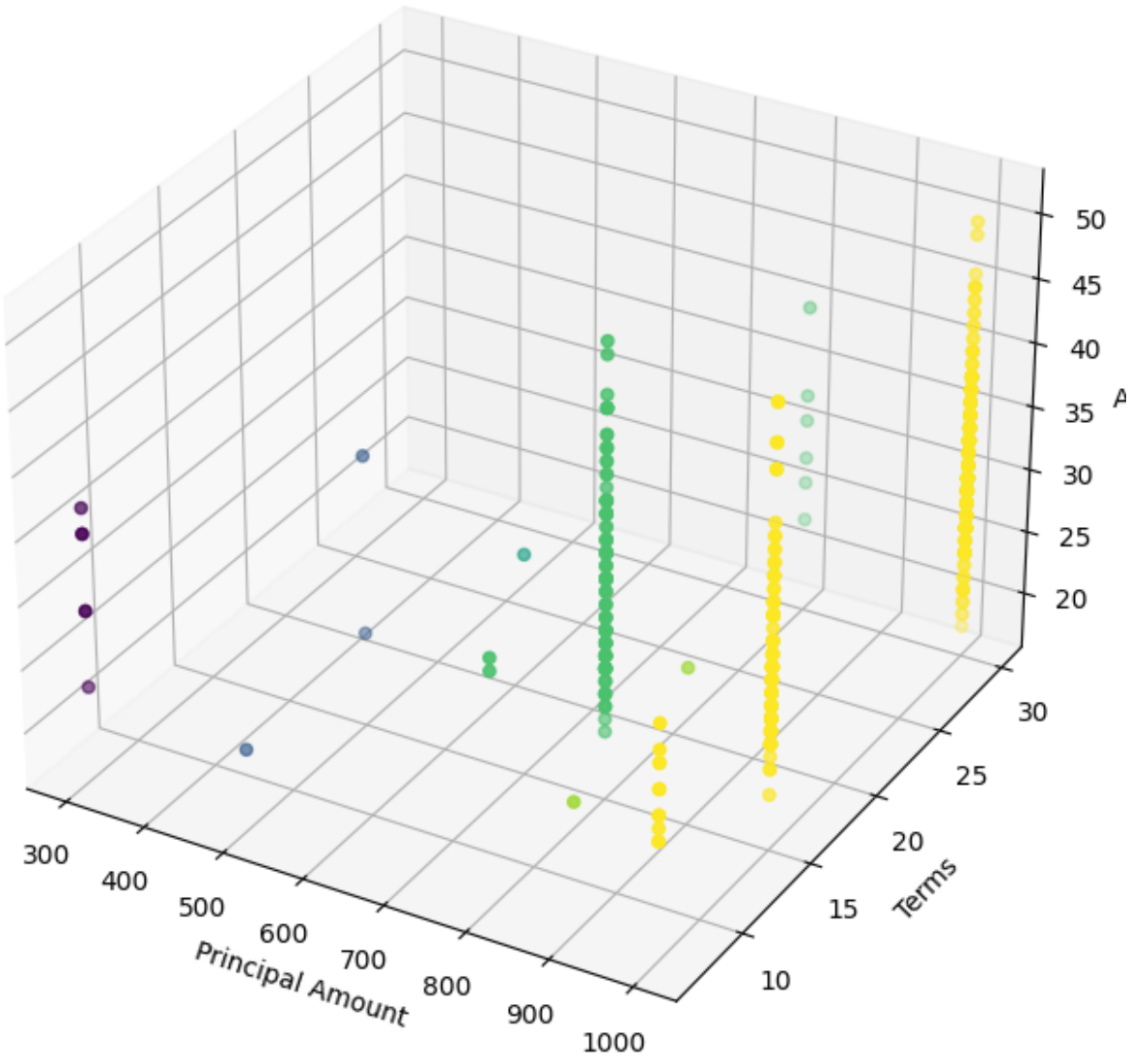
We can see that men tend to get more loans and they obtain loans more easily.

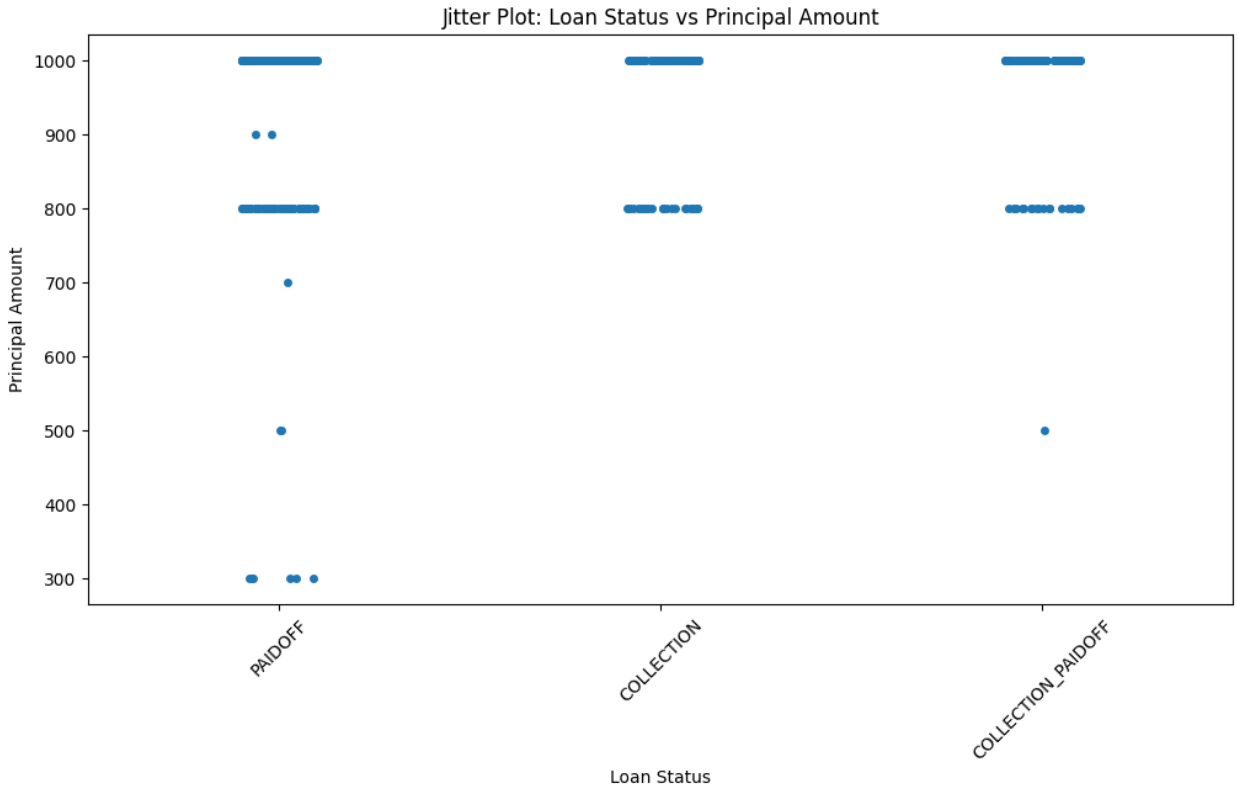

(4) WhatsApp
Assignment
AAexp7advHome
ADV Labs
Visualizing Loan D
Loan Data Visualiz
127.0.0.1:5500/loan_visualization.html
Age vs Principal Amount
1,000
950
900
850
800
750
700
650
600
550
500
450
400
350
300
20
25
30
35
40
45
50
Loan Amount Over Time
1,000
Type here to search
26°C
ENG
03:53
14-10-2024


(4) WhatsApp
Assignment
AAexp7advHome
ADV Labs
Visualizing Loan D
Loan Data Visualiz
127.0.0.1:5500/loan_visualization.html
300
20
25
30
35
40
45
50
Loan Amount Over Time
1,000
900
800
700
600
500
400
300
200
100
0
Thu 08
12 PM
Fri 09
12 PM
Sat 10
12 PM
Sep 11
12 PM
Mon 12
12 PM
Tue 13
12 PM
Wed 14
Principal Distribution by Education
Type here to search
26°C
ENG
03:53
14-10-2024


(4) WhatsApp
Assignment
AAexp7advHome
ADV Labs
Visualizing Loan D
Loan Data Visualiz
127.0.0.1:5500/loan_visualization.html
Principal Distribution by Education
1,000
900
800
700
600
500
400
300
200
100
0
High School or Below
Bechalor
college
Master or Above
Type here to search
USD...
ENG
03:53
14-10-2024


Regression Plot: Principal vs Terms
Terms
Principal Amount
30
25
20
15
10
5
0
−5
300
400
500
600
700
800
900
1000


3D Chart: Principal, Terms, and Age
300
400
500
600
700
800
900
1000
Principal Amount
10
15
20
25
30
Terms
20
25
30
35
40
45
50



Hypothesis Testing: Correlation between Age and Principal Amount
Pearson Correlation Coefficient: -0.0926
P-value: 0.0384
The correlation is statistically significant ($p < 0.05$)
There is a negative correlation between age and loan amount.

Regression Plot (Principal vs Terms)

- **Trend:** There's a slight positive trend, suggesting that as the principal amount increases, the terms tend to increase slightly. This relationship is not very strong.
- **Data Points:** The points are clustered around certain term values (likely discrete values like 7, 15, 30 days), indicating that loan terms are typically standardized.

3D Chart (Principal, Terms, and Age)

- **Distribution:** Most data points cluster around lower principal amounts and shorter terms.
- **Age:** There's no clear pattern between age, principal, and terms. The color variation, representing principal, shows that higher principal loans are scattered across different age groups and terms.
- **Outliers:** There might be some outliers with higher principal amounts and longer terms.

Jitter Plot (Loan Status vs Principal Amount)

- **Loan Status:** The plot shows the distribution of principal amounts for different loan statuses (e.g., PAIDOFF, COLLECTION).
- **PAIDOFF:** Loans that were paid off seem to have a wider range of principal amounts.
- **COLLECTION:** Loans that went into collection appear to be concentrated around lower principal amounts. This suggests that smaller loans might have a higher risk of default.

Conclusion: I deployed a D3.js code and the graphs were obtained on a HTML site seen on local host. I created various basic and advance charts through which I made various observations.