資料視覺化
Data Visualization

期中報告

主題:電視遊戲全球分布分析
姓名:王佑恩
學號:108601205
系級:電機3A
資料來源:https://www.kaggle.com/datasets/gregorut/videogamesales

一、動機與資料分析

1. 選題動機

電視遊戲(又稱電子遊戲)在1970年代開始以商業娛樂媒體的形式出現在市面上，隨著時間演進，逐漸成為人們重要的娛樂項目。近年受疫情影響，遊戲成為居家休閒的最佳活動之一，大眾在遊戲產業中的消費大幅上升，也成功讓電視遊戲產業再次邁入新的高峰。

清明連假時第一次與朋友玩Switch，發現現在的電子遊戲相比於10年前Wii那樣的電子遊戲，在畫質、運行速度與遊玩體驗上，都有明顯的進步。剛好這次在Kaggle上搜尋資料時，看到Video Game Sales這個數據集，便以此當作資料來進行期中報告的資料分析。

2. 數據分析

Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
1	Wii Sports	Wii	2006	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
2	Super Mario	NES	1985	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
3	Mario Kart W	Wii	2008	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
4	Wii Sports R	Wii	2009	Sports	Nintendo	15.75	11.01	3.28	2.96	33
5	Pokemon Re	GB	1996	Role-Playing	Nintendo	11.27	8.89	10.22	1	31.37
6	Tetris	GB	1989	Puzzle	Nintendo	23.2	2.26	4.22	0.58	30.26
7	New Super M	DS	2006	Platform	Nintendo	11.38	9.23	6.5	2.9	30.01
8	Wii Play	Wii	2006	Misc	Nintendo	14.03	9.2	2.93	2.85	29.02
9	New Super M	Wii	2009	Platform	Nintendo	14.59	7.06	4.7	2.26	28.62
10	Duck Hunt	NES	1984	Shooter	Nintendo	26.93	0.63	0.28	0.47	28.31
11	Nintendogs	DS	2005	Simulation	Nintendo	9.07	11	1.93	2.75	24.76
12	Mario Kart D	DS	2005	Racing	Nintendo	9.81	7.57	4.13	1.92	23.42
13	Pokemon Go	GB	1999	Role-Playing	Nintendo	9	6.18	7.2	0.71	23.1

此數據集共有16600筆資料，每筆資料包含11個columns，此數據集中無空值，各個column的介紹如下:

- Rank - 以全球總銷售額進行排名之整體銷售排名
- Name - 遊戲名稱

- Platform - 遊戲發布之平台（如PC、PS4等）
- Year - 遊戲發佈之年份
- Genre - 遊戲類型（如Sport、Racing等）
- Publisher - 遊戲發行商
- NA_Sales - 北美地區銷售額 (in millions)
- EU_Sales - 歐洲地區銷售額 (in millions)
- JP_Sales - 日本地區銷售額 (in millions)
- Other_Sales - 其他地區之銷售額 (in millions)
- Global_Sales - 全球總銷售額

本報告中的資料分析主要針對地區與銷售額之間的關係進行分析，觀察觀察各地區的玩家偏好的遊戲類型差異與分析造成差異的原因。

二、實作與結果展示

在所有columns當中，可以透過各地區銷售額以及遊戲類型來達成分析的目標。實作之程式碼如下所示。

```
// // 期中報告
// 已確認資料無空值

// deal with empty value
const parseNA = string => (string === 'NA' ? undefined : string);
```

因為已確認數據並無空值，所以此步驟主要是放在後續數字處理部分，確保字串格式統一。

```
// 數字處理
// Sales的單位是million
function type(d){
    return{
        Rank: +d.Rank,
        Name: parseNA(d.Name),
        Platform: parseNA(d.Platform),
```

```
14          Year: +d.Year,
15          Genre: parseNA(d.Genre),
16          Publisher: parseNA(d.Publisher),
17          NA_Sales: +d.NA_Sales,
18          EU_Sales: +d.EU_Sales,
19          JP_Sales: +d.JP_Sales,
20          Other_Sales: +d.Other_Sales,
21          Global_Sales: +d.Global_Sales
22        }
23      }
```

透過課堂所教之數字處理方式，將數字的string前面加上'+'，把其轉換成數字格式。剩餘string格式的文字，則維持string。

```
24
25      // Data selection
26      // 主要選擇2010 ~ 2020之間的資料
27      function filterData(data){
28        return data.filter(
29          d => {
30            return(
31              d.Year >= 2010 && d.Year <= 2020 &&
32              d.Name &&
33              d.Platform &&
34              d.Genre &&
35              d.NA_Sales > 0 &&
36              d.EU_Sales > 0 &&
37              d.JP_Sales > 0 &&
```

```
                d.Other_Sales > 0 &&
                d.Global_Sales > 0
            );
        }
    );
}
```

透過此Data Selection之function，選擇2010~2020年之間的數據。此外，為了避免銷售額有錯誤資料(負數)的狀況，且銷售額為0之值並非此報告需要關注的項目，所以我只篩選出銷售額大於0的數據。

```

// 依照遊戲類型(Genre)進行分類
// 將各類型的NA_Sales, EU_Sales, JP_Sales, Other_Sales, Global_Sales分別加總
function prepareData(data, state){
    console.log(data, state);
    if (state == 0){
        const dataMap = d3.rollup(
            data,
            v => d3.sum(v, leaf => leaf.NA_Sales),
            d => d.Genre
        );

        const dataArray = Array.from(dataMap, d => ({Genre:d[0], NA_Sales:d[1]}));
        return dataArray;
    }
    else if (state == 1){
        const dataMap1 = d3.rollup(
```

```
61              data,
62              v => d3.sum(v, leaf => leaf.EU_Sales),
63              d => d.Genre
64          );
65
66          const dataArray1 = Array.from(dataMap1, d => ({Genre:d[0], EU_Sales:d[1]}));
67          return dataArray1;
68      }
69      else if (state == 2){
70          const dataMap2 = d3.rollup(
71              data,
72              v => d3.sum(v, leaf => leaf.JP_Sales),
73              d => d.Genre
74          );
75
76          const dataArray2 = Array.from(dataMap2, d => ({Genre:d[0], JP_Sales:d[1]}));
77          return dataArray2;
78      }
79      else if (state == 3){
80          const dataMap3 = d3.rollup(
81              data,
82              v => d3.sum(v, leaf => leaf.Other_Sales),
83              d => d.Genre
84          );
85
```

```
86          const dataArray3 = Array.from(dataMap3, d => ({Genre:d[0], Other_Sales:d[1]}));
87          return dataArray3;
88      }
89      else{
90          const dataMap4 = d3.rollup(
91              data,
92              v => d3.sum(v, leaf => leaf.Global_Sales),
93              d => d.Genre
94          );
95
96          const dataArray4 = Array.from(dataMap4, d => ({Genre:d[0], Global_Sales:d[1]}));
97          return dataArray4;
98      }
99
100 }
```

因為與課程所使用的數據一樣皆是統計金額，所以在rollup()函式中，也是使用d3.sum()函式將金額加總。與課程內容不同的地方在於，此function使用類似”state machine”的運作方式，多設定一個function接收的值為”state”，來分別進行不同地須銷售額的rollup以及生成array like。

此撰寫方式的優點在於，為了達成目的，不用再多寫很多類似的function。雖然有違一般function盡量回傳一個數據的原則，但我認為這樣可以增加程式的readability。

```
101
102 // 刻度顯示
103 function formatTicks(d){
104     return (d3.format('~s')(d) + 'mil')
105 }
```

	因為在本數據中金額是以million為單位，所以將刻度的數字後面，加上”mil”字串。
106	
107	// 繪製svg
108	function setupCanvas(Data_NA, Data_EU, Data_JP, Data_Other, Data_Global, state){
109	console.log(state);
110	const svg_width = 600;
111	const svg_height = 500;
112	const chart_margin = {top:80,right:40,bottom:40,left:80};
113	const chart_width = svg_width - (chart_margin.left + chart_margin.right);
114	const chart_height = svg_height - (chart_margin.top + chart_margin.bottom);
115	const this_svg = d3.select('.bar-chart-container').append('svg')
116	.attr('width', svg_width).attr('height',svg_height)
117	.append('g')
118	.attr('transform',`translate(${chart_margin.left},${chart_margin.top})`);
119	
120	if (state == 0){
121	// Scale
122	const xMax = d3.max(Data_NA, d=>d.NA_Sales);
123	const xScale = d3.scaleLinear([0,xMax],[0, chart_width]);
124	
125	// 垂直空間的分配 - 平均分布給各種類
126	const yScale = d3.scaleBand()
127	.domain(Data_NA.map(d => d.Genre))
128	.rangeRound([0, chart_height])
129	.paddingInner(0.25);

```
130
131         //Draw bars
132         const bars = this_svg.selectAll('.bar')
133                         .data(Data_NA)
134                         .enter()
135                         .append('rect')
136                         .attr('class','bar')
137                         .attr('x',0)
138                         .attr('y',d=>yScale(d.Genre))
139                         .attr('width',d=>xScale(d.NA_Sales))
140                         .attr('height',yScale.bandwidth())
141                         .style('fill', 'dodgerblue')
142
143         // Draw header
144         const header = this_svg.append('g').attr('class','bar-header')
145             .attr('transform',`translate(0,${-chart_margin.top/2})`)
146             .append('text');
147         header.append('tspan').text('Total sales by genre in North America');
148         header.append('tspan').text('Years:2010 - 2020')
149               .attr('x',0).attr('y',20).style('font-size','0.8em').style('fill','#555');
150
151         // 設定tickSize
152         const xAxis = d3.axisTop(xScale)
153                         .tickFormat(formatTicks)
154                         .tickSizeInner(-chart_height)
```

```
155                     .tickSizeOuter(0);
156         const xAxisDraw = this_svg.append('g').attr('class','x axis').call(xAxis);
157
158         const yAxis = d3.axisLeft(yScale).tickSize(0);
159         const yAxisDraw = this_svg.append('g').attr('class','y axis').call(yAxis);
160         yAxisDraw.selectAll('text').attr('dx','-0.6em');
161       }
162       else if (state == 1){
163         // Scale
164         const xMax = d3.max(Data_EU, d=>d.EU_Sales);
165         const xScale = d3.scaleLinear([0,xMax],[0, chart_width]);
166
167         // 垂直空間的分配 - 平均分布給各種類
168         const yScale = d3.scaleBand()
169                     .domain(Data_EU.map(d => d.Genre))
170                     .rangeRound([0, chart_height])
171                     .paddingInner(0.25);
172
173         //Draw bars
174         const bars = this_svg.selectAll('.bar')
175                       .data(Data_EU)
176                       .enter()
177                       .append('rect')
178                       .attr('class','bar')
179                       .attr('x',0)
```

```
                    .attr('y',d=>yScale(d.Genre))
                    .attr('width',d=>xScale(d.EU_Sales))
                    .attr('height',yScale.bandwidth())
                    .style('fill', 'lightgreen')

    // Draw header
    const header = this_svg.append('g').attr('class','bar-header')
      .attr('transform',`translate(0,${-chart_margin.top/2})`)
      .append('text');
    header.append('tspan').text('Total sales by genre in Europe');
    header.append('tspan').text('Years:2010 - 2020')
       .attr('x',0).attr('y',20).style('font-size','0.8em').style('fill','#555');

    // 設定tickSize
    const xAxis = d3.axisTop(xScale)
                .tickFormat(formatTicks)
                .tickSizeInner(-chart_height)
                .tickSizeOuter(0);
    const xAxisDraw = this_svg.append('g').attr('class','x axis').call(xAxis);

    const yAxis = d3.axisLeft(yScale).tickSize(0);
    const yAxisDraw = this_svg.append('g').attr('class','y axis').call(yAxis);
    yAxisDraw.selectAll('text').attr('dx','-0.6em');
  }
  else if (state == 2){
```

```
205         // Scale
206         const xMax = d3.max(Data_JP, d=>d.JP_Sales);
207         const xScale = d3.scaleLinear([0,xMax],[0, chart_width]);
208
209         // 垂直空間的分配 - 平均分布給各種類
210         const yScale = d3.scaleBand()
211                     .domain(Data_JP.map(d => d.Genre))
212                     .rangeRound([0, chart_height])
213                     .paddingInner(0.25);
214
215         //Draw bars
216         const bars = this_svg.selectAll('.bar')
217                         .data(Data_JP)
218                         .enter()
219                         .append('rect')
220                         .attr('class','bar')
221                         .attr('x',0)
222                         .attr('y',d=>yScale(d.Genre))
223                         .attr('width',d=>xScale(d.JP_Sales))
224                         .attr('height',yScale.bandwidth())
225                         .style('fill', 'orange')
226
227         // Draw header
228         const header = this_svg.append('g').attr('class','bar-header')
229             .attr('transform',`translate(0,${-chart_margin.top/2})`)
```

```
230             .append('text');
231         header.append('tspan').text('Total sales by genre in Japan');
232         header.append('tspan').text('Years:2010 - 2020')
233             .attr('x',0).attr('y',20).style('font-size','0.8em').style('fill','#555');
234
235         // 設定tickSize
236         const xAxis = d3.axisTop(xScale)
237                 .tickFormat(formatTicks)
238                 .tickSizeInner(-chart_height)
239                 .tickSizeOuter(0);
240         const xAxisDraw = this_svg.append('g').attr('class','x axis').call(xAxis);
241
242         const yAxis = d3.axisLeft(yScale).tickSize(0);
243         const yAxisDraw = this_svg.append('g').attr('class','y axis').call(yAxis);
244         yAxisDraw.selectAll('text').attr('dx','-0.6em');
245     }
246     else if (state == 3){
247         // Scale
248         const xMax = d3.max(Data_Other, d=>d.Other_Sales);
249         const xScale = d3.scaleLinear([0,xMax],[0, chart_width]);
250
251         // 垂直空間的分配 - 平均分布給各種類
252         const yScale = d3.scaleBand()
253                 .domain(Data_Other.map(d => d.Genre))
254                 .rangeRound([0, chart_height])
```

```
255                     .paddingInner(0.25);
256
257         //Draw bars
258         const bars = this_svg.selectAll('.bar')
259                     .data(Data_Other)
260                     .enter()
261                     .append('rect')
262                     .attr('class','bar')
263                     .attr('x',0)
264                     .attr('y',d=>yScale(d.Genre))
265                     .attr('width',d=>xScale(d.Other_Sales))
266                     .attr('height',yScale.bandwidth())
267                     .style('fill', 'salmon')
268
269         // Draw header
270         const header = this_svg.append('g').attr('class','bar-header')
271             .attr('transform',`translate(0,${-chart_margin.top/2})`)
272             .append('text');
273         header.append('tspan').text('Total sales by genre in the rest of the world');
274         header.append('tspan').text('Years:2010 - 2020')
275             .attr('x',0).attr('y',20).style('font-size','0.8em').style('fill','#555');
276
277         // 設定tickSize
278         const xAxis = d3.axisTop(xScale)
279                     .tickFormat(formatTicks)
```

```
                    .tickSizeInner(-chart_height)
                    .tickSizeOuter(0);
    const xAxisDraw = this_svg.append('g').attr('class','x axis').call(xAxis);

    const yAxis = d3.axisLeft(yScale).tickSize(0);
    const yAxisDraw = this_svg.append('g').attr('class','y axis').call(yAxis);
    yAxisDraw.selectAll('text').attr('dx','-0.6em');
  }
  else{
    // Scale
    const xMax = d3.max(Data_Global, d=>d.Global_Sales);
    const xScale = d3.scaleLinear([0,xMax],[0, chart_width]);

    // 垂直空間的分配 - 平均分布給各種類
    const yScale = d3.scaleBand()
                    .domain(Data_Global.map(d => d.Genre))
                    .rangeRound([0, chart_height])
                    .paddingInner(0.25);

    //Draw bars
    const bars = this_svg.selectAll('.bar')
                    .data(Data_Global)
                    .enter()
                    .append('rect')
                    .attr('class','bar')
```

```
305                         .attr('x',0)
306                         .attr('y',d=>yScale(d.Genre))
307                         .attr('width',d=>xScale(d.Global_Sales))
308                         .attr('height',yScale.bandwidth())
309                         .style('fill', 'maroon')
310
311         // Draw header
312         const header = this_svg.append('g').attr('class','bar-header')
313             .attr('transform',`translate(0,${-chart_margin.top/2})`)
314             .append('text');
315         header.append('tspan').text('Total sales by genre in worldwide');
316         header.append('tspan').text('Years:2010 - 2020')
317             .attr('x',0).attr('y',20).style('font-size','0.8em').style('fill','#555');
318
319         // 設定tickSize
320         const xAxis = d3.axisTop(xScale)
321                     .tickFormat(formatTicks)
322                     .tickSizeInner(-chart_height)
323                     .tickSizeOuter(0);
324         const xAxisDraw = this_svg.append('g').attr('class','x axis').call(xAxis);
325
326         const yAxis = d3.axisLeft(yScale).tickSize(0);
327         const yAxisDraw = this_svg.append('g').attr('class','y axis').call(yAxis);
328         yAxisDraw.selectAll('text').attr('dx','-0.6em');
329     }
```

330	}
此function參照課程ppt之繪製svg方式，進行bar chart的繪製，其中也使用了類似”state machine”的運作方式，將多張圖透過不同的參數state繪製出來。但這樣的寫法有點冗長，後續「問題發現與解決」部分會說明為何採用此寫法。	
331	
332	// Main
333	function operate(games){
334	const gameFilter = filterData(games);
335	// Data分類以及sorting
336	const Data_NA = prepareData(gameFilter, 0).sort(
337	(a, b) => {
338	return d3.descending(a.NA_Sales, b.NA_Sales);
339	}
340	);
341	const Data_EU = prepareData(gameFilter, 1).sort(
342	(a, b) => {
343	return d3.descending(a.EU_Sales, b.EU_Sales);
344	}
345	);
346	const Data_JP = prepareData(gameFilter, 2).sort(
347	(a, b) => {
348	return d3.descending(a.JP_Sales, b.JP_Sales);
349	}
350	);
351	const Data_Other = prepareData(gameFilter, 3).sort(
352	(a, b) => {

```
353             return d3.descending(a.Other_Sales, b.Other_Sales);
354         }
355     );
356     const Data_Global = prepareData(gameFilter, 4).sort(
357         (a, b) => {
358             return d3.descending(a.Global_Sales, b.Global_Sales);
359         }
360     );
361
362     // console.log(Data_NA);
363     // console.log(Data_EU);
364     // console.log(Data_JP);
365     // console.log(Data_Other);
366     // console.log(Data_Global);
367     // debugger;
368     // const state = 0;
369     setupCanvas(Data_NA, Data_EU, Data_JP, Data_Other, Data_Global, 0);
370     setupCanvas(Data_NA, Data_EU, Data_JP, Data_Other, Data_Global, 1);
371     setupCanvas(Data_NA, Data_EU, Data_JP, Data_Other, Data_Global, 2);
372     setupCanvas(Data_NA, Data_EU, Data_JP, Data_Other, Data_Global, 3);
373     setupCanvas(Data_NA, Data_EU, Data_JP, Data_Other, Data_Global, 4);
374 }
```

此部分為Main function，首先將數據送入fiterData進行Data selection。接著，將篩選好的數據進行分類以及排序，其中使用到課程所介紹的sorting方式，將所選數據呈遞減排序。最後將分類好的5個數據設定不同的state後送入setupCanvas這個function進行svg的繪製。

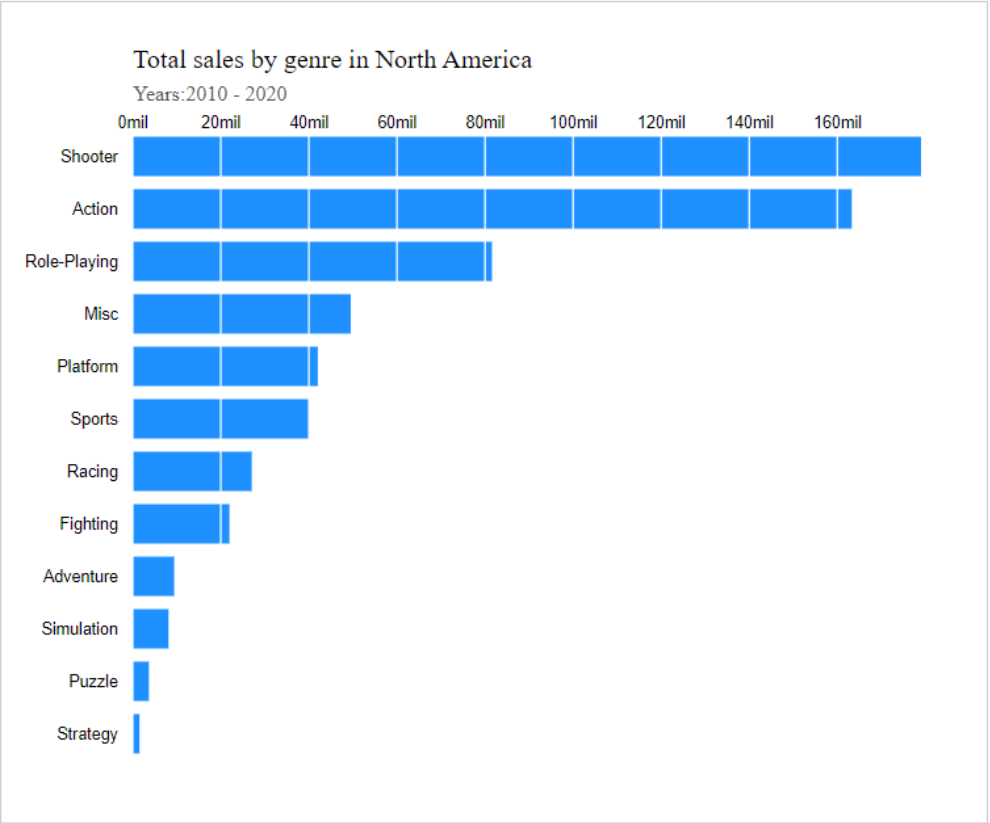
```
375
```

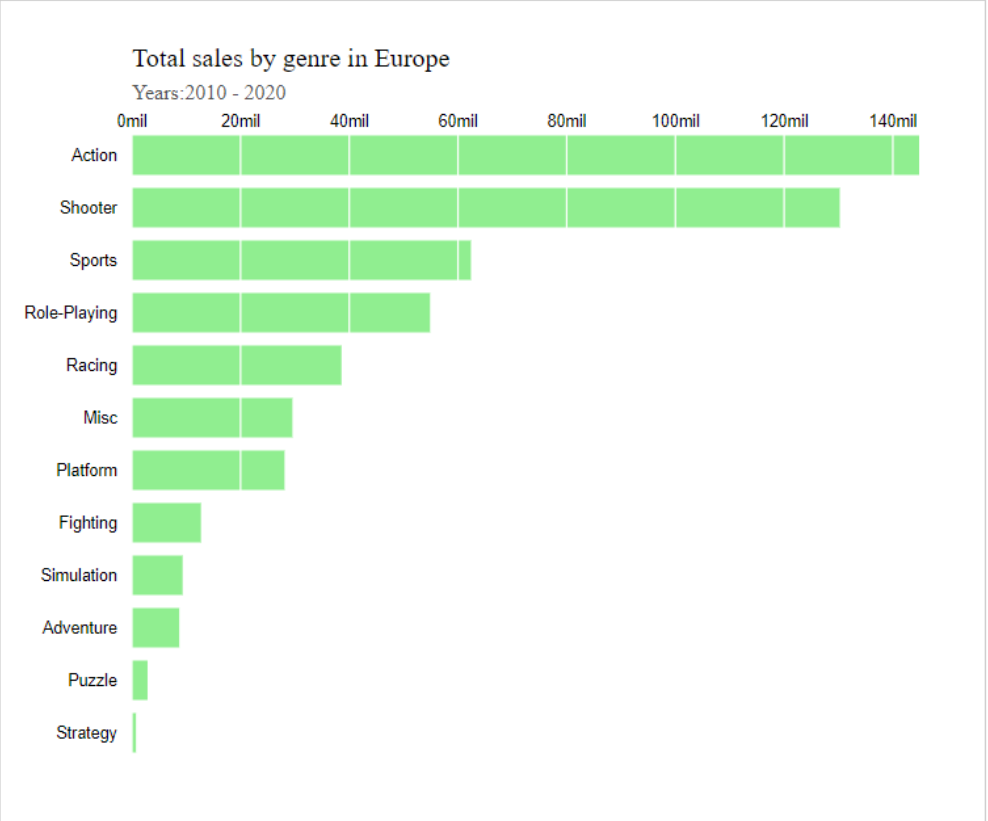
```
// Load Data
d3.csv('data/vgsales.csv', type).then(
    res => {
        operate(res);
        // console.log(res);
        // debugger;    // 要觀察區域變數時，可用debugger進行中斷
    }
);
```

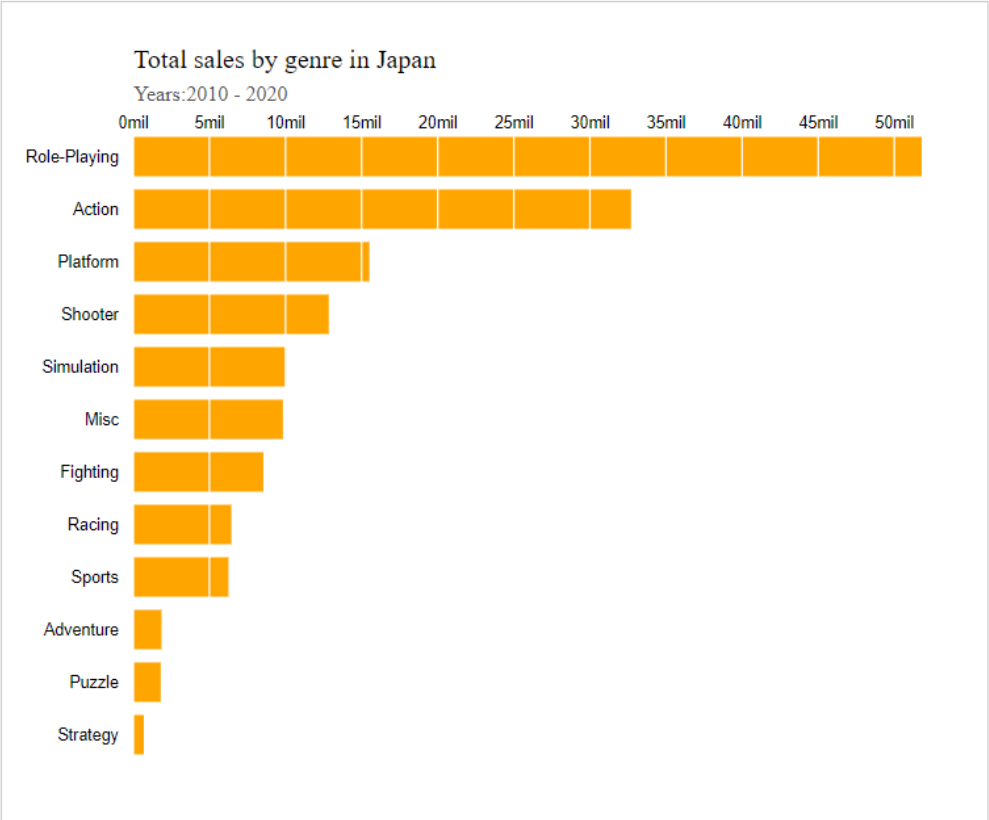
三、結果分析與發現

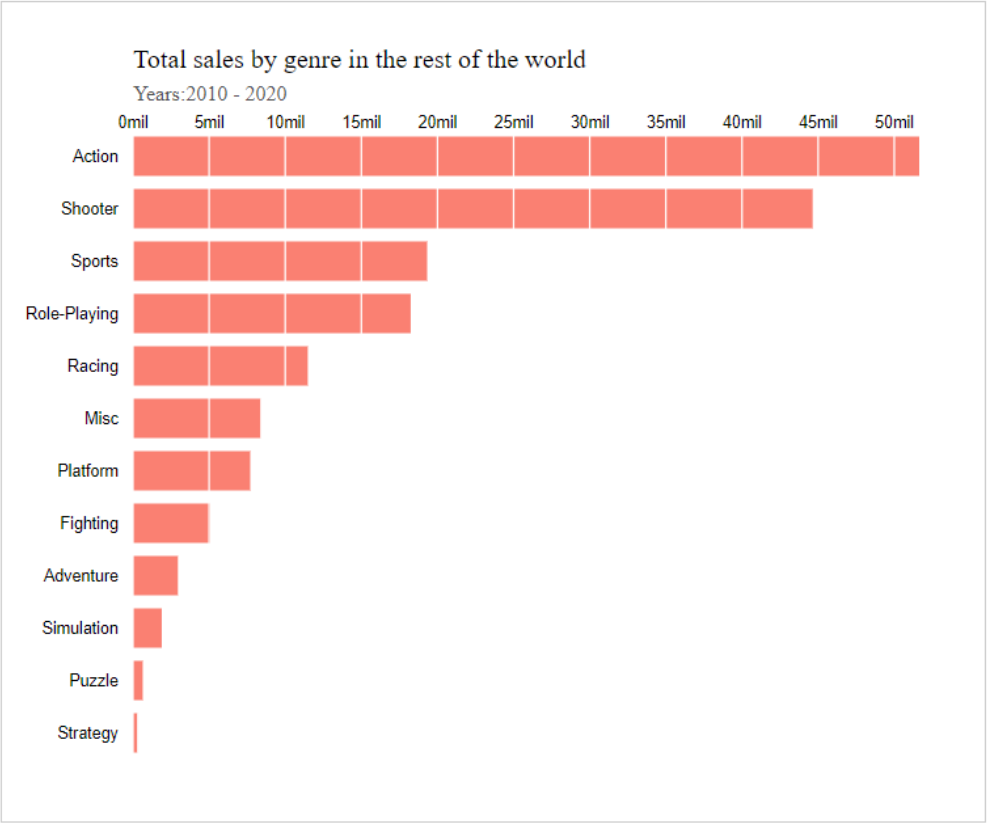
1. **結果展現與分析**

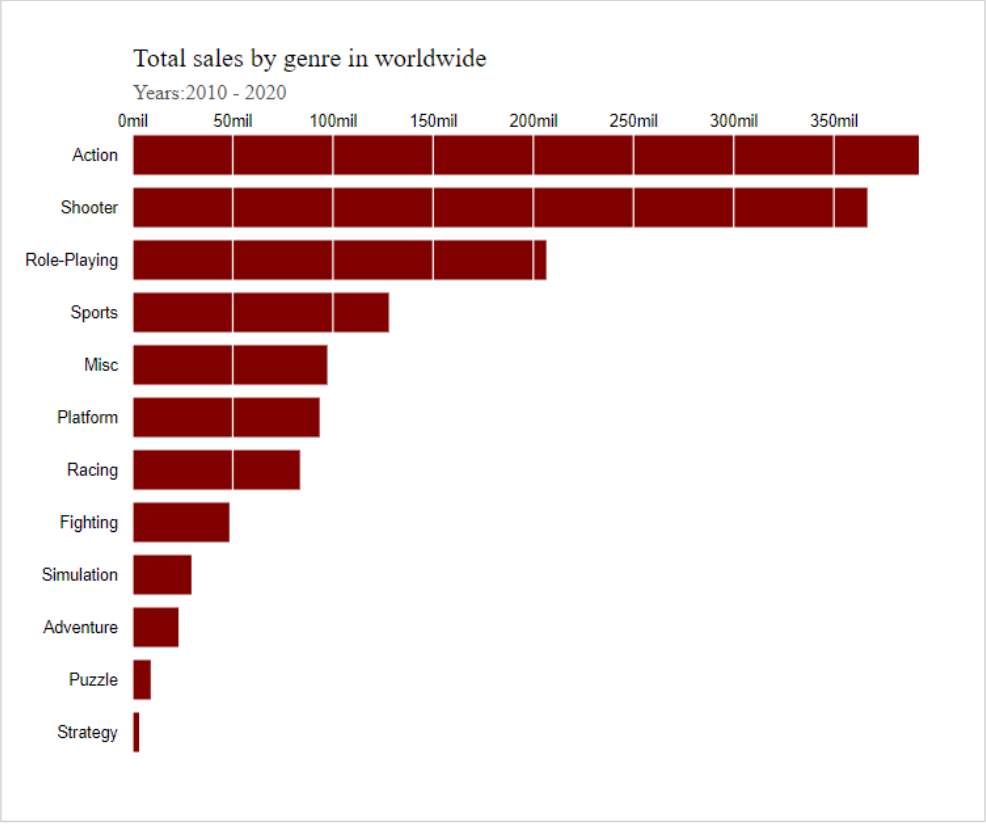
   http://127.0.0.1:5500/midterm/index.html




Total sales by genre in Europe
Years:2010 - 2020
0mil
20mil
40mil
60mil
80mil
100mil
120mil
140mil
Action
Shooter
Sports
Role-Playing
Racing
Misc
Platform
Fighting
Simulation
Adventure
Puzzle
Strategy


Total sales by genre in Japan
Years:2010 - 2020
0mil
5mil
10mil
15mil
20mil
25mil
30mil
35mil
40mil
45mil
50mil
Role-Playing
Action
Platform
Shooter
Simulation
Misc
Fighting
Racing
Sports
Adventure
Puzzle
Strategy





觀察上面圖表可以發現，射擊類遊戲在北美地區最受歡迎，可能的原因是美國法規沒有嚴格的槍械管制，加上好萊塢的電影中有較多使用槍戰的場景。在

這些情況下，大部分玩家還是無法自由使用槍械，所以美國的玩家為了體驗槍械使用，則會透過射擊類遊戲來滿足慾望。

歐洲地區則是動作類遊戲最受歡迎，可能的原因是歐洲有較長的騎士文化，或者是古代戰士、維京人等等歷史。將北美地區與歐洲地區的分析結果做比較，會發現西方玩家大多喜歡射擊類與動作類遊戲。但有趣的是運動類遊戲對於歐洲玩家也是有一定的熱愛程度，反觀北美玩家則對運動類遊戲較不感興趣。

日本地區受歡迎前兩類型的是角色扮演類與動作類遊戲，這與日本蓬勃發展的動漫業有很大的關係，玩家會希望透過遊玩遊戲，體驗自己成為動漫角色，甚至是擁有異能或忍術等的感覺。

2. 還可以分析之項目：

在分析過程中我發現，遊戲玩家除了對遊戲種類有偏好外，對於遊戲發行商也會有自己的喜好。因此可以透過比較某遊戲在哪個地區較受歡迎，並紀錄下地區以及此遊戲的遊戲發行商，將所有遊戲重複這樣的步驟後，就能進行遊戲發行商與地區之間關係的分析，藉此分析哪家遊戲發行商在哪個地區較受歡迎。

四、問題發現與解決

1. rollup()函式使用限制
   - 問題描述：

     發現rollup只能對兩個數據及進行groupby，無法對多組數據同時rollup，導致只能用array把多個rollup完的數據存入，在return此array。但在分割array時，發現分割的語法與python有很大不同。
   - 解決方式：

     多次嘗試後仍無法將資料分割成課堂ppt中return的array like，導致無法以課堂的方式進行後續的sorting以及繪圖。所以最後決定以前述類似state machine的方式進行資料分類。
2. svg繪圖遇到之問題
   - 問題描述：

     在嘗試將setupCanvas這個function簡化，使其只要接收一個數據集以及一個state即可完成繪圖的過程中發現，在設自訂變數時，一定要把自訂變數設定初始值。在不知道如何設定資料型態的情況下，使我無法在程式中呼叫數據集內的資料。
   - 解決方式：

     在現階段初學javascript的情況下，無奈只能將全部的數據集都送入function，並為每個數據都寫一段繪製bar chart的程式碼，才能達到想要的結果。

五、心得

在找資料的過程中，有在網路上看到很多程式大佬做的資料視覺化專案，發現設計精美的分析圖確實能讓人眼睛為之一亮，但呈現的資料讓人容易讀懂也相當重要。我在程式設計的最後一個步驟時，原本想設計讓畫面一次只會顯示一張分析圖，並讓使用者能透過鍵盤按鍵切換不同的bar chart，但多次嘗試d3.select()函式、keydown設置、使用Event.keyCode甚至是清空svg的selectAll("*").remove()語句都無法順利運行。雖然也有試著用後續課堂會使用的按鍵方式想要達到目標，但內部的一些語法卻不知道該如何設定。因為種種原因，所以後來仍然選擇讓視窗上顯示所有的分析圖，期待後續課程我能在理解完老師的講解過後，回頭成功完成此項目標。