EECS 6893 HW4

Jing Qian (jq2282)

Problem 1

1. Answer these questions in simple words.

1.1

SVG Coordinate Space and Mathematical/Graph Coordinate are both 2-dimensional flat space. They work in the same way except following:

- The (0, 0) coordinates of Mathematical Coordinate Space fall on the bottom left while those of SVG Coordinate Space on the top left.
- The Mathematical Coordinate Space has Y coordinate growing from bottom to top while SVG Coordinate Space has Y coordinate growing from top to bottom.

1.2

In d3.js, enter() and exit() are operations that used in joining an array of data to a D3 selection. If the array is longer than the selection, we use enter() to represent the missing elements. If the array is shorter than the selection, we use exit() to represent elements that need to be removed.

1.3

In SVG, transform functions are used for geometric manipulations on the SVG elements, including translation (movement), rotation, scale, and skew (shear).

A translation moves all the points of an element in the same direction and by the same amount. Translation preserves parallelism, angles and distances.

1.4

The anonymous function a.map(function(d,i){return i+5}) here equals to the following command in Python: [i+5 for i in range(len(a))]. So the return value of the anonymous function is: [5, 6, 7, 8, 9]. 需要描述anonymous function或者map吗？对于定义还是有点迷茫。

2. Modify the sample code to get desired figure.

The screenshots of codes and the output bar-chart are as following:

```html
big_data_hw4_part1.html  ×

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <script type="text/javascript" src='HW4_Q1.js'> </script>
5      <meta charset="UTF-8">
6      <title>Homework 4 Question 1</title>
7  </head>
8
9  <style>body{}.bar-chart{background-color: #006666;}</style>
10 <svg class="bar-chart"></svg>
11
12 <script src="https://d3js.org/d3.v5.min.js"></script>
13 <body>
14 <script>
15     HW4_Q1();
16 </script>
17
18 </body>
19 </html>
```

```
HW4_Q1.js

1   function HW4_Q1(){
2       var data = [80, 100, 56, 120, 180, 30, 40, 120, 160];
3       var svgWidth = 500, svgHeight = 300;
4       // The required padding between bars is 5px.
5       // The label must locate 2px above the middle of each bar.
6
7       var svg = d3.select('svg')
8               .attr("width", svgWidth)
9               .attr("height", svgHeight);
10
11      var barChart = svg.selectAll("rect")
12              .data(data)
13              .enter()
14              .append("rect")
15              .attr("class", "bar")
16              .attr("transform", function(d, i) { return "translate(" + i
                    * 55 + ","+(svgHeight-d)+")"; })
17              .attr('width', 50)
18              .attr('height', svgHeight)
19              .attr("fill", "#C0C0C0");
20
21      var text = svg.selectAll(".textlabel")
22              .data(data)
23              .enter()
24              .append("text")
25              .attr("class", "textlabel")
26              .attr("transform", function(d, i) { return "translate(" + (
                    i * 55+15) + ","+(svgHeight-d-2)+")"; })
27              .style("font-family", "Arial")
28              .style("color", "#333333")
29              .text(function(d){return d;});
30  }
```
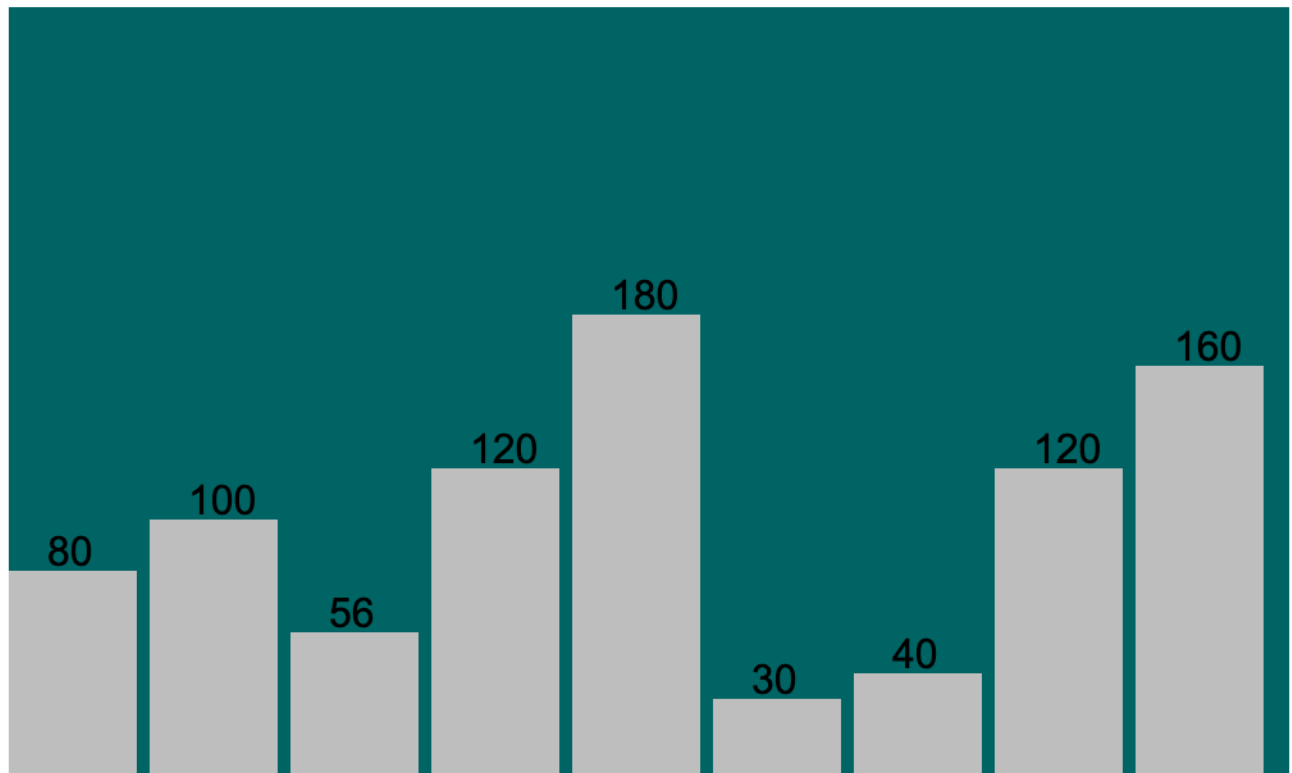
Problem 2

Step 1: Data processing

```python
import numpy as np
import pandas as pd

data = pd.read_csv('hw3-wordcount.csv')
```

```python
time = list(set(data.time))
df = pd.DataFrame(time, columns=['time'])
```

```python
words = ['ai','data','good','movie','spark']
for i in words:
    df[i] = 0
```

```python
for i in range(len(df)):
    for j in words:
        tmp = data[(data.time==df.time[i])&(data.word==j)]
        if not tmp.empty:
            df[j][i] = int(tmp['count'])
df.head()
```

	time	ai	data	good	movie	spark
0	2019-10-31 13:28:50 UTC	10	2	10	244	11
1	2019-10-31 13:29:50 UTC	15	1	4	209	14
2	2019-10-31 13:26:50 UTC	19	4	8	226	14
3	2019-10-31 13:27:50 UTC	13	0	4	178	18
4	2019-10-31 13:25:50 UTC	9	4	3	176	17

wordcount2

Row	time	ai	data	good	movie	spark
1	2019-10-31 13:27:50 UTC	13	0	4	178	18
2	2019-10-31 13:29:50 UTC	15	1	4	209	14
3	2019-10-31 13:28:50 UTC	10	2	10	244	11
4	2019-10-31 13:26:50 UTC	19	4	8	226	14
5	2019-10-31 13:25:50 UTC	9	4	3	176	17
6	2019-10-31 13:30:50 UTC	37	5	9	207	8

Step 2: Create Django project

Name	Date Modified
▼ 📁 hw4_tutorial	Today at 1:36 AM
▼ 📁 hw4_tutorial	Today at 1:34 AM
▶ 📁 __pycache__	Today at 1:43 AM
📄 __init__.py	Today at 1:22 AM
📄 settings.py	Today at 1:32 AM
📄 urls.py	Today at 1:43 AM
📄 view.py	Today at 1:40 AM
📄 wsgi.py	Today at 1:22 AM
▼ 📁 static	Today at 1:40 AM
▶ 📁 css	Today at 1:40 AM
▶ 📁 js	Today at 1:40 AM
▼ 📁 template	Today at 1:33 AM
📄 helloworld.html	Today at 1:34 AM
📄 db.sqlite3	Today at 1:36 AM
📄 manage.py	Today at 1:22 AM

← → C ⓘ 127.0.0.1:8000/hello
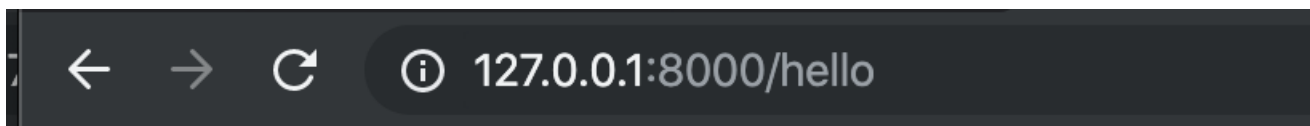
Hello World!

Step 3. Finish the code

Modify the view.py:

```python
SQL = "select * from datasetHW3.wordcount2 limit 8"
df = pandas_gbq.read_gbq(SQL)
#print(df)
data = {}

res = []
words = ['ai','data','good','movie','spark']
time = sorted(list(df.time))
for i in time:
    tmp = {}
    tmp['Time'] = str(i)[11:16]
    tmp['count'] = {}
    for j in words:
        tmp['count'][j] = int(df[df.time==i][j])
    res.append(tmp)
data['data'] = res
#print(data['data'])
return render(request, 'dashboard.html', data)
```

Modify the dashboard.js

```javascript
function segColor(c) {
    cmap = {ai: "#4753CC", data: "#828499", good: "#73C9FF", movie: "#CC6E47"
        , spark: '#FFD4B3'};
    return cmap[c];/* TO FINISH */
}

// compute total for each state.
fData.forEach(function (d) {
    d.total = d['count']['ai']+d['count']['data']+d['count']['good']+d['count'
        ]['movie']+d['count']['spark']; /* TO FINISH */
})
```

```
//create the rectangles.
bars.append("rect")
    .attr("x", function (d,i) {return i*x.rangeBand()*1.1+10;}/* TO
    FINISH */)
    .attr("y", hGDim.b/* TO FINISH */)
    .attr("width", x.rangeBand())
    .attr("height", function (d) {
        return hGDim.h - y(d[1]);
    })
    .attr('fill', barColor)
    .on("mouseover", mouseover)
    .on("mouseout", mouseout);

//Create the frequency labels ABOVE the rectangles.
bars.append("text").text(function (d) {
    return d3.format(",")(d[1])
})
    .attr("x", function (d) {
        return x(d[0])/* TO FINISH */ + x.rangeBand() / 2;
    })
    .attr("y", hGDim.b+5/* TO FINISH */)
    .attr("text-anchor", "middle");
```

```
// transition the height and color of rectangles.
bars.select("rect").transition().duration(500)
    .attr("y", function (d) {
        return y(d[1]);
    }/* TO FINISH */)
    .attr("height", function (d) {
        return hGDim.h - y(d[1]);
    })
    .attr("fill", color);
```

```
// calculate total count by segment for all state.
var tF = ['ai', 'data', 'good', 'movie', 'spark'].map(function (d) {
    return {
        type: d, count: d3.sum(fData.map(function (t) {
            return d.total/* TO FINISH */;
        }))
    };
});
```

Result:

Question 2 - Dashboard
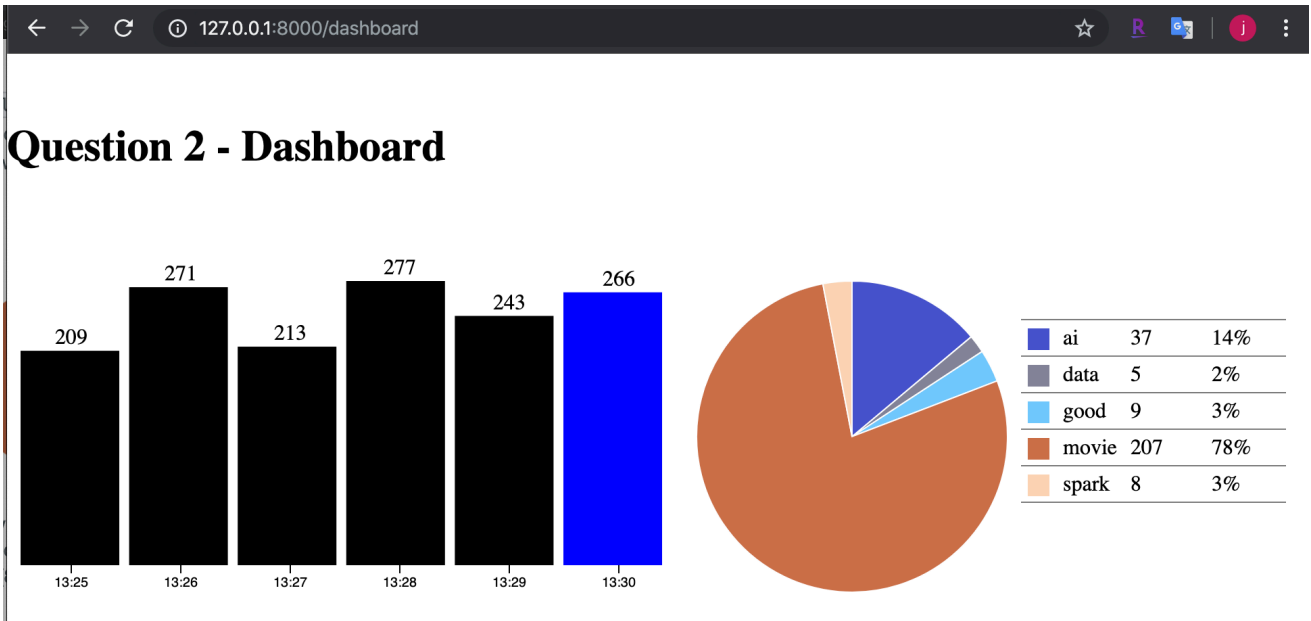
Problem 3

Based on the edge.csv and node.csv provided by TA, the view.py is modified as following:

```python
SQL1 = 'select * from datasetHW3.node'
df1 = pandas_gbq.read_gbq(SQL1)
#print("node preview: ", df1[:5])
node = []
for i in range(len(df1)):
    tmp = {}
    tmp['node'] = int(df1['node'][i])
    node.append(tmp)
#print("node,", node[:5])

SQL2 = 'select * from datasetHW3.edge'
df2 = pandas_gbq.read_gbq(SQL2)
#print("edge preview: ", df2[:5])
edge = []
visited = set()
for i in range(len(df2)):
    tmp = {}
    pair = (int(df2['source'][i]), int(df2['target'][i]))
    if pair not in visited:
        visited.add(pair)
        tmp['source'] = pair[0]
        tmp['target'] = pair[1]
        edge.append(tmp)
print("len of edge: ", len(edge))
data = {}
data['n'] = node
data['e'] = edge
```

Modified connection.js:

```python
SQL = "select * from datasetHW3.wordcount2 limit 8"
df = pandas_gbq.read_gbq(SQL)
print(df)
data = {}

res = []
words = ['ai','data','good','movie','spark']
time = sorted(list(df.time))
for i in time:
    tmp = {}
    tmp['Time'] = str(i)[11:16]
    tmp['count'] = {}
    for j in words:
        tmp['count'][j] = int(df[df.time==i][j])
    res.append(tmp)
data['data'] = res
print(data['data'])
return render(request, 'dashboard.html', data)
```

```javascript
var svg = d3.select("body")
            .append("svg")
            .attr("width", width/* TO FINISH */)
            .attr("height", height/* TO FINISH */);
```

```javascript
var svg_edges = svg.selectAll("line")
                   .data(edges/* TO FINISH */)
                   .enter()
                   .append("line"/* TO FINISH */)
                   .style("stroke","#ccc")
                   .style("stroke-width",1);
```

```javascript
var svg_nodes = svg.selectAll("circle")
                   .data(nodes/* TO FINISH */)
                   .enter()
                   .append("circle"/* TO FINISH */)
                   .attr("r",20)
                   .style("fill", function(d,i){return color(i);}/* TO FINISH
                   */)
                   .call(force.drag);
```

```
var svg_texts = svg.selectAll("text")
                   .data(nodes)
                   .enter()
                   .append("text")
                   .style("fill", "black")
                   .attr("dx", 20)
                   .attr("dy", 8)
                   .text(function(d){return d.node;}/* TO FINISH */ );


force.on("tick", function(){
    svg_edges.attr("x1", function(d){return d.source.x;}/* TO FINISH */)
             .attr("y1", function(d){return d.source.y;}/* TO FINISH */)
             .attr("x2", function(d){return d.target.x;}/* TO FINISH */)
             .attr("y2", function(d){return d.target.y;}/* TO FINISH */);
```

Result: