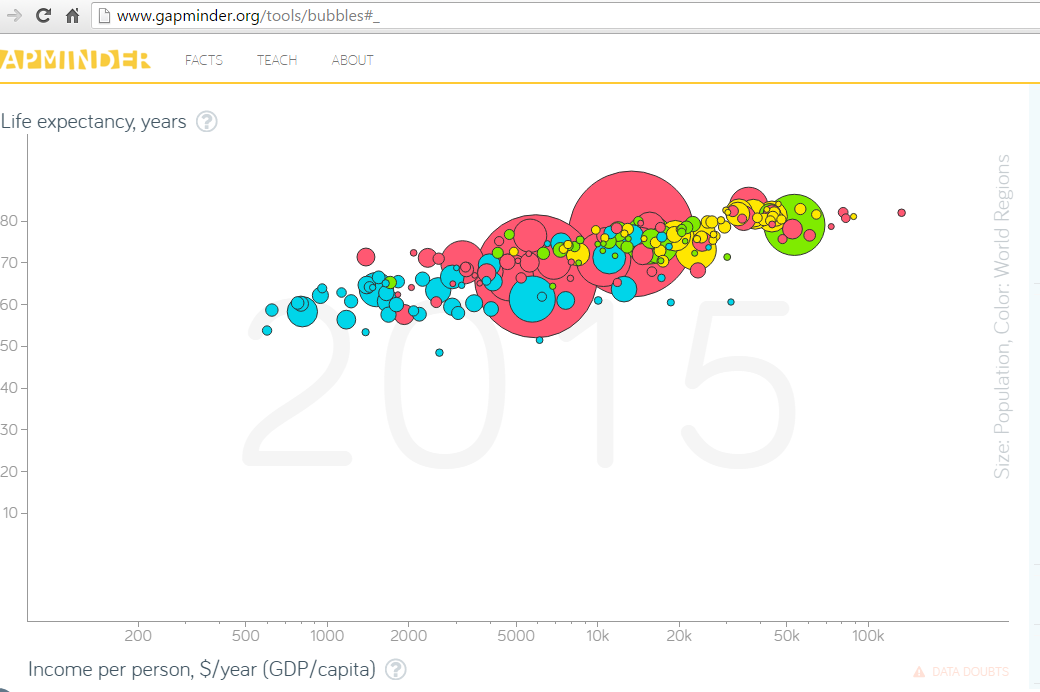
## HU Extension Assignment 09 E63 Big Data Analytics

### Handed out: 04/01/2016 Due by 11:30PM EST on Friday, 04/08/2016

Please, describe every step of your work and present all intermediate and final results in a Word document. Please, copy past text version of all essential command and snippets of results into the Word document with explanations of the purpose of those commands. We cannot retype text that is in JPG images. Please, always submit a separate copy of the original, working scripts and/or class files you used. Sometimes we need to run your code and retyping is too costly. Please include in your MS Word document only relevant portions of the console output or output files. Sometime either console output or the result file is too long and including it into the MS Word document makes that document too hard to read. PLEASE DO NOT EMBED files into your MS Word document. For issues and comments visit the class Discussion Board. You are not obliged to use Java or Eclipse. You are welcome to use any language and any IDE of your choice.

**Problem 1)** Public site GapMinder.org presents many excellent visualizations of data about the World. One such presentation (<http://www.gapminder.org/tools/bubbles#_>) displays average life expectancy in year as a function of average income per person in countries of the world. Countries are represented as circles in different colors depending on their continent, e.g. countries in Europe are yellow, countries in Asia, red, etc. Every country is presented by a circle of area proportional to its population. Radius of the circle is therefore proportional to the square root of population. As the cursor hovers over each country, its name appears over its circle. Data presented in this graph can be found in various Excel files provided by the same site (<http://www.gapminder.org/data/)>. We extracted some files which we believe contain data used in the graph bellow:



Please recreate above graph using D3 or any similar technology of your choice**.** You do not need all data present in provided Excel files. Select most recent data for every country. If you know what you are doing, keep your data in files on your OS or in a database of your choice. Otherwise, copy relevant data directly into your “HTML/JavaScript” code. Please note that the horizontal axis is logarithmic.

For this, I first assembled the data in the following way:

I took the file *countries of the world.xls* and *indicator life\_expectancy\_at\_birth\_1800-2050.xlsx*

I created a new file called *assignment9\_data.csv* from it with the fields:

Country,Continent,Life expectancy in 2015,Per capita income,Population

The fields Country, Continent, Population and Per capita income are taken from *countries of the world.xls*

(for Per capita income I referred to the excel column GDP $ per capita)

The field Life expectancy in 2015 is taken from the excel sheet indicator *life\_expectancy\_at\_birth\_1800-2050.xlsx*

This is a sample of assignment9\_data.csv I came up with:

Country,Continent,Life expectancy in 2015,Per capita income,Population

Afghanistan,ASIA,61.726,700,31056997

Albania,EUROPE,77.807,4500,3581655

Algeria,AFRICA,71.246,6000,32930091

Angola,AFRICA,52.698,1900,57794

Then I used D3 javascript.

And I wrote the following html code for this problem:

<!DOCTYPE html>

<html lang=*"en"*>

<head>

<meta charset=*"utf-8"*>

<meta name=*"author"* content=*"Rohan Pulekar"*>

<meta name=*"description"*

content=*"This html file is for Assignment9 Problem1 of e63 course (Big Data Analutics) at Harvard Extension School"*>

<title>Assignment9 Problem1</title>

<script type=*"text/javascript"* src=*"../js/d3/d3.js"*></script>

<style type=*"text/css"*>

*.dot* {

stroke: *#000*;

}

**div***.tooltip* {

position: *absolute*;

text-align: *center*;

width: *200px*;

height: *80px*;

padding: *2px*; : 18 px sans-serif;

border: *20px*;

background: *#d1c2f0*;

border-radius: *8px*;

pointer-events: *none*;

}

</style>

</head>

<body>

<script type=*"text/javascript"*>

// create svg margins

**var** svgMargin = {

top : 20,

right : 20,

bottom : 30,

left : 40

}, svgWidth = 1360 - svgMargin.left - svgMargin.right, svgHeight = 500

- svgMargin.top - svgMargin.bottom;

// define x axis

**var** xValue = **function**(d) {

**return** d["Per capita income"];

};

**var** xScale = d3.scale.log().range([ 0, svgWidth ]); // x axis scale is logarithmic scale

**var** xMap = **function**(d) {

**return** xScale(xValue(d));

};

**var** xAxis = d3.svg.axis().scale(xScale).orient("bottom").ticks(10,

",.1s").tickSize(6, 1);

// define y axis

**var** yValue = **function**(d) {

**return** d["Life expectancy in 2015"];

};

**var** yScale = d3.scale.linear().range([ svgHeight, 0 ]); // y axis has a linear scale

**var** yMap = **function**(d) {

**return** yScale(yValue(d));

};

**var** yAxis = d3.svg.axis().scale(yScale).orient("left").tickSize(6, 1);

// create a function that will give a different color for each continent

**var** cValue = **function**(d) {

**return** d["Continent"];

};

// create an ordinal scale of colors

**var** color = d3.scale.category10();

// add the graph canvas to the body of the webpage

**var** svg = d3.select("body").append("svg").attr("width",

svgWidth + svgMargin.left + svgMargin.right).attr("height",

svgHeight + svgMargin.top + svgMargin.bottom).append("g").attr(

"transform",

"translate(" + svgMargin.left + "," + svgMargin.top + ")");

// Define div for the country circle tooltip

**var** div = d3.select("body").append("div").attr("class", "tooltip")

.style("opacity", 0);

// load data from the below mentioned csv file

d3.csv("../input\_files/assignment9\_data.csv", **function**(error, data) {

// change from string to number format the per capita income and life expectancy fields

data.forEach(**function**(d) {

d["Per capita income"] = +d["Per capita income"];

d["Life expectancy in 2015"] = +d["Life expectancy in 2015"];

});

// add buffer to data domain for clarity of the graph

xScale.domain([ Math.pow(10, 2), Math.pow(10, 5) ]);

yScale.domain([ d3.min(data, yValue) - 20,

d3.max(data, yValue) + 10 ]);

// draw x axis

svg.append("g").attr("class", "x axis").attr("transform",

"translate(0," + svgHeight + ")").call(xAxis)

.append("text").attr("class", "label").attr("x", svgWidth)

.attr("y", -6).style("text-anchor", "end").text(

"Per capita income");

// draw y axis

svg.append("g").attr("class", "y axis").call(yAxis).append("text")

.attr("class", "label").attr("transform", "rotate(-90)")

.attr("y", 6).attr("dy", ".71em").style("text-anchor",

"end").text("Life expectancy in 2015");

// draw a circle for each country

**var** circles = svg.selectAll(".dot").data(data).enter().append(

"circle").attr("class", "dot");

// set radius of the circle to a number thats proprtional to the square root population

circles.attr("r", **function**(d) {

**return** getRadius(d["Population"]);

});

// set a sort fuunction so that countries with lower population/radius appear ahead of countries with higher population

circles.sort(**function**(a, b) {

**return** d3.ascending(a["Population"], b["Population"]);

});

// set x and y cordinates for center of the circle

circles.attr("cx", xMap).attr("cy", yMap)

// fill the circle with color corresponding to the continent in which country resides

circles.style("fill", **function**(d) {

**return** color(cValue(d));

})

// set mouseover and mouseout events for the circle

circles.on(

"mouseover",

**function**(d) {

// get the circle element on which is mouse is over

**var** circleElement = d3.select(**this**)[0][0];

// set the fill color to light purple

d3.select(**this**).style("fill", "#d1c2f0");

// make the tooltip for the circle visible

div.transition().duration(200).style("opacity", .9);

// set the text and coordinates of the tooltip

div.html(getDisplayLabelForCountry(d)).style(

"left",

(circleElement.getAttribute("cx")

- circleElement.getAttribute("r") - 50)

+ "px").style(

"top",

(circleElement.getAttribute("cy")

- circleElement.getAttribute("r") - 50)

+ "px");

}).on("mouseout", **function**(d) {

// set the fill color to the one for the continent in which this country is located

d3.select(**this**).style("fill", color(cValue(d)));

// make the tooltip for the circle disappear

div.transition().duration(500).style("opacity", 0);

});

// draw legends

**var** legend = svg.selectAll(".legend").data(color.domain()).enter()

.append("g").attr("class", "legend").attr("transform",

**function**(d, i) {

**return** "translate(0," + i \* 20 + ")";

});

// draw legend colored rectangles

legend.append("rect").attr("x", svgWidth - 18).attr("width", 18)

.attr("height", 18).style("fill", color);

// draw legend text

legend.append("text").attr("x", svgWidth - 24).attr("y", 9).attr(

"dy", ".35em").style("text-anchor", "end").text(

**function**(d) {

**return** d;

});

});

// this function gets population and gives corresponding radius. As you can see from the formula, radius given back is proportional to the square root of population

**function** getRadius(population) {

**return** Math.sqrt(population) / 600;

}

// create label for the country on which user has mouse over

**function** getDisplayLabelForCountry(d) {

**var** displayLabelForCountry = d["Country"] + "<br>Income:"

+ d["Per capita income"] + "<br>Life expectancy:"

+ d["Life expectancy in 2015"] + "<br>Population:"

+ Math.round(d["Population"] / 1000000) + " million";

**return** displayLabelForCountry;

}

</script>

<div>

<h1>Assignment9 Problem1</h1>

<h3>The above graph is for data of year 2015</h3>

</div>

</body>

</html>

And this is my local directory structure:

rpulekar-m1:WebContent rpulekar$ pwd

/Users/rpulekar/work/big-data-analytics-harvard/eclipse\_workspace\_for\_course/Assignment9/WebContent

rpulekar-m1:WebContent rpulekar$ ls

TA\_samples html input\_files js

rpulekar-m1:WebContent rpulekar$ ls html/

assignment9\_problem1.html assignment9\_problem2.html backup\_code.html data.csv

rpulekar-m1:WebContent rpulekar$

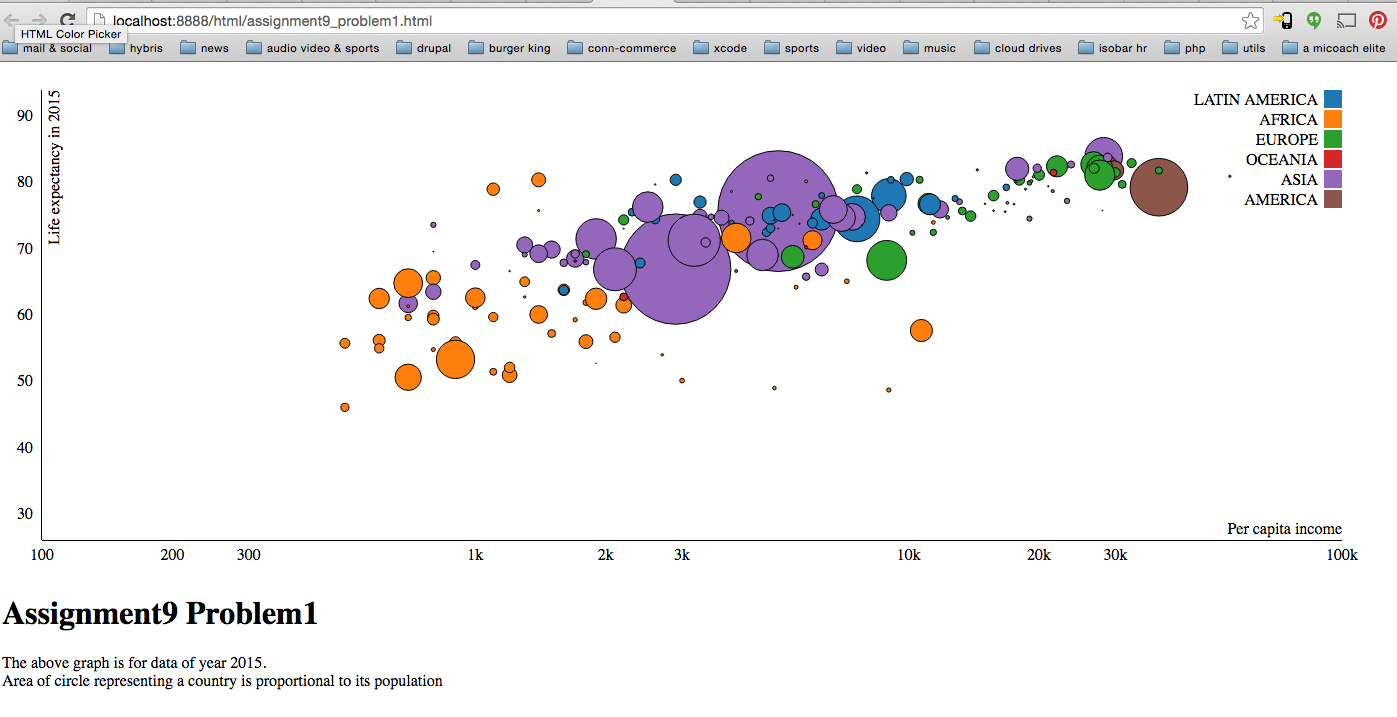
Then I started the python server:

rpulekar-m1:WebContent rpulekar$ python -m SimpleHTTPServer 8888 &

[2] 83950

rpulekar-m1:WebContent rpulekar$ Traceback (most recent call last):

Then I accessed **<http://localhost:8888/html/assignment9_problem1.html>** on my browser and got this graph:

****

On mouse hover, over the particular circle, name of that country alongwith population, income and life expectancy are shown above the circle.

So, basically my input data to the html file is in a csv file on my hard disk.

Horizontal axis (Per capita income) of the graph is logarithmic.

**Problem 2)** Display the graph of population per country as a pie chart. Color countries by the continents and group all countries by the continents. Leave no separation between different countries, however, when you hover over a country change it color to light purple and display its name and population in millions.

For this I used the csv file created in Problem1.

assignment9\_data.csv:

Country,Continent,Life expectancy in 2015,Per capita income,Population

Afghanistan,ASIA,61.726,700,31056997

Albania,EUROPE,77.807,4500,3581655

Algeria,AFRICA,71.246,6000,32930091

Angola,AFRICA,52.698,1900,57794

I have my python server running:

rpulekar-m1:WebContent rpulekar$ python -m SimpleHTTPServer 8888 &

[2] 83950

rpulekar-m1:WebContent rpulekar$ Traceback (most recent call last):

Then I wrote the following html program for this problem:

<!DOCTYPE html>

<html>

<head>

<meta charset=*"UTF-8"*>

<meta name=*"author"* content=*"Rohan Pulekar"*>

<meta name=*"description"*

content=*"This html file is for Assignment9 Problem2 of e63 course (Big Data Analutics) at Harvard Extension School"*>

<title>Assignment9 Problem2</title>

<script type=*"text/javascript"* src=*"../js/d3/d3.js"*></script>

</head>

<body>

<script>

// create dimensions for svg element

**var** svgWidth = 1060, svgHeight = 600, pieRadius = Math.min(svgWidth,

svgHeight) / 2;

// create color map to map each continent to a color

**var** colorMap = **new** Object();

colorMap["ASIA"] = "RED";

colorMap["AFRICA"] = "Yellow";

colorMap["EUROPE"] = "Green";

colorMap["LATIN AMERICA"] = "Orange";

colorMap["OCEANIA"] = "Maroon";

colorMap["AMERICA"] = "SlateGray";

// create an arc generator and give it inner and outer radius

**var** arc = d3.svg.arc().outerRadius(pieRadius - 20).innerRadius(0);

// construct a new pie

**var** pie = d3.layout.pie();

// create the sort order so that all slices belonging to the same continent will be grouped together when pie is being constructed

pie.sort(**function**(a, b) {

**return** d3.descending(a["Continent"], b["Continent"]);

});

// this specified that population should be used while constructing each slice of the pie

pie.value(**function**(d) {

**return** d["Population"];

});

// create the svg element

**var** svg = d3.select("body").append("svg").attr("width", svgWidth).attr(

"height", svgHeight).append("g").attr("transform",

"translate(" + svgWidth / 2 + "," + (svgHeight / 2) + ")");

// read from the below mentioned csv file

d3.csv("../input\_files/assignment9\_data.csv", getPopulation, **function**(

error, data) {

**if** (error)

**throw** error;

// create an arc corresponding to each row in the csv file

**var** arcs = svg.selectAll(".arc").data(pie(data)).enter()

.append("g").attr("class", "arc");

// this is a variable to keep track of the fill color before mouse enters the pie slice

**var** fillColorForCountry;

// create a path for the slice corresponding to current country

arcs.append("path").attr("d", arc).style("fill", **function**(d, i) {

**return** colorMap[d.data["Continent"]];

}).on(

"mouseover",

**function**(d, i) {

// set the fill color to light purple

d3.select(**this**).style("fill", "#c2ade0");

// create label for the country on which user's mouse is over

arcs.append("text").attr("dy", ".5em").style(

"text-anchor", "middle").style("font-size", 25)

.attr("font-family", "Verdana").attr("class",

"selectedCountryLabel").style("fill",

"White").text(

getDisplayLabelForCountry(d));

}).on("mouseout", **function**(d) {

// remove the selected country label

arcs.select(".selectedCountryLabel").remove();

// set the fill color back to the color corresponding to continent of the country

d3.select(**this**).style("fill", colorMap[d.data["Continent"]]);

});

});

// this changes population from string to number format

**function** getPopulation(d) {

d["Population"] = +d["Population"];

**return** d;

}

// create label for the country on which user has mouse over

**function** getDisplayLabelForCountry(d) {

**var** displayLabelForCountry = d.data["Country"] + " Population:"

+ Math.round(d.data["Population"] / 1000000) + " million";

**return** displayLabelForCountry;

}

// create legends for the pie chart through javascript

document.write("</br>");

document.write("<table cellpadding='10'><tr>");

**for** ( **var** continent **in** colorMap) {

**if** (colorMap.hasOwnProperty(continent)) {

document.write("<td>");

document

.write("<div style='width:40px;height:10px;border:1px solid #000;background-color:" + colorMap[continent] +"'></div>"

+ continent);

document.write("</td>");

}

}

document.write("</tr></table>");

</script>

<div>

<h2>Assignment9 Problem2</h2>

<span>Pie slice of each country is proportional to its population</span>

</div>

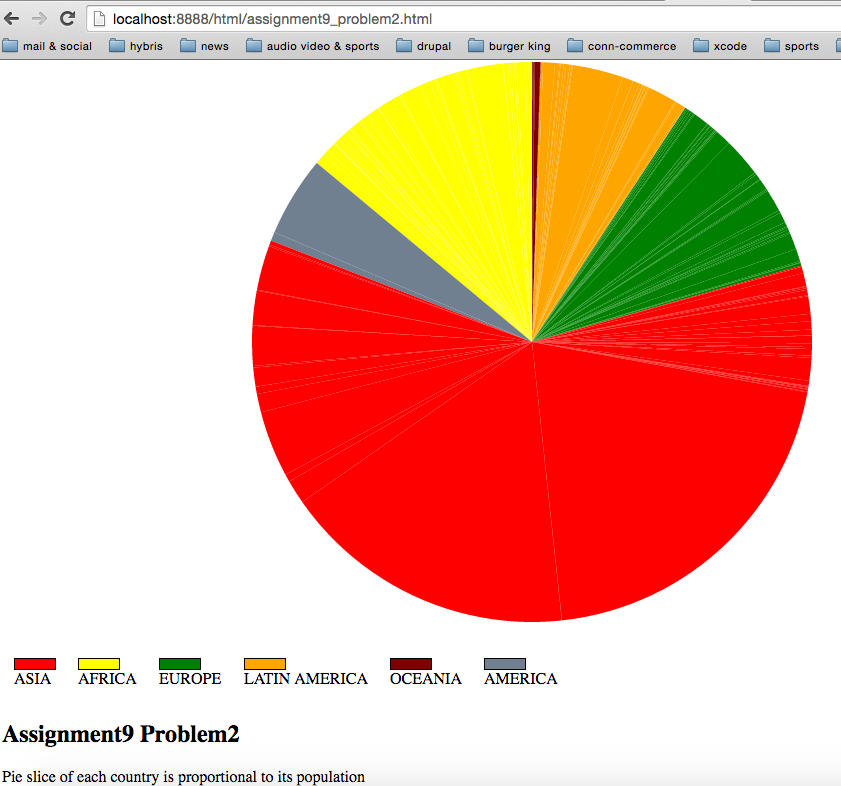
</body>

</html>

I accessed output of this html program on:

<http://localhost:8888/html/assignment9_problem2.html>

This is what I see in the output:



On mouse hover, the particular country slice changes its color to light purple.

And the name and population (in millions) is displayed for that country.

I wanted to use svg to create legends. But was unable to. So, my legends right now are constructed from plain javascript.

I tried to eliminate the dotted blur lines that appear between countries using *stroke-width* property, *stroke* property, but they did not go away.

**For every problem, please provide a working copy of your HTML file including the JavScript (D3) portion*.***

Deliverables:

* assignment9\_RohanPulekar.zip

which has following directory structure

