Using R To Get Value Out Of Public Data

PhD Candidate Marius RADU
PhD Assistant Ioana MUREŞAN
PhD Professor Răzvan NISTOR
Babeş-Bolyai University, Faculty of Economics and Business Administration

ABSTRACT

Public sector information contains great value for the citizens in general. Data stored on computers of public institutions doesn't have value on its own. It has to be processed and analyzed to obtain information, and further on, information should be made available as public good, in order to facilitate its transformation to knowledge. R is a free software programming language, an environment and toolkit of modules addressed to anyone working with statistics. R can ease the road from public data to civic wisdom. This article is a brief review of R capabilities to extract, transform, analyze, and visualize public data. Second part of the article presents an example of a fully-fledged web application written entirely in R. The application uses loosely structured government data about Romanian Auto Park in order to present it in a friendly dashboard.

Key Words: *Open Data, R programming language, Reporting Web Applications*

INTRODUCTION

Data revolution moves forward and public data initiative is part of this movement. Having a strategic and informed view on public data is for everyone's interest. Open data initiatives aims to make communities more functional, sustainable and effective. There are hundreds of specialty forums, blogs and professional groups with discussions about big data, future of NO-SQL, engineering approaches that can cope with the new structure. The public data topic is more and more frequently approached. On the other side we rarely can see applications, results or solutions for real problems using open data for the public good.

Open Data is considered to be included as part of the larger concept of Open Government. In a large understanding open data is both "technically open" and "legally open". Technically open means available in a machine-readable standard format, while legally open means that data is explicitly licensed in a way that permits commercial and non-commercial use and re-use without restrictions [1]United States","source":"ProQuest","event-place":"Washington, United States","abstract":"According to the foundation's

California Open Data Handbook, published in collaboration with Stewards of Change Institute, a national group supporting innovation in human services, data must first be both \"technically open\" and \"legally open.\" If the intended users are developers and programmers, Shaw said, the data should be presented within an application programming interface (API. Between "public data" and "open data" is a thin delimitation and in the present article we will consider this two concepts covering the same thing.

In this paper we will see two main approaches of using R towards open data:

1) The analytical process: For the first approach we will use from EU public data portal a set of open data regarding generation of waste in different countries. Paper will give an example regarding municipal waste in Egypt. The example is created to illustrate R capabilities in respect with open data and not necessary to revel outstanding facts about environment.
2) The application construction to share knowledge from open data. For the second approach we will use open data regarding Romanian Auto Park for 2013. We will use this data from Romanian government data portal to create with few lines of R code a web reporting application

The focus of this article is placed on different ways in which R can help to solve problems with open data, but our intention is not to present in depth a certain case and a problem, neither to present the spectrum of R packages and the complexity of problems that can be solved.

PUBLIC DATA AND THE R POTENTIAL

If we consider only the Romanian public data, there are hundreds of data sets available at Data.gov.ro, or at city data catalogs like http://data.e-primariaclujnapoca.ro/. A big challenge is to make these data add insight or utility to citizen's everyday lives. R can do this with simplicity and no monetary costs. Using R for public data exploration is a meaningful opportunity to tell stories that are relevant to a region and the individuals. R is a tremendous useful tool in many ways when it comes to converting open data into information that people can distill into knowledge and insight.

Behind all positive aspects of open data there are hurdles and risks which complicate the value extraction of the data. Martin and Foulnneau identify seven categories of risks to Open Data initiatives: governance, economic issues, licenses and legal frameworks, data characteristics, metadata, access, and skills [2]United Kingdom","page":"301-XVII","source":"ProQuest","event-place":"Kidmore End, United Kingdom","abstract":"Despite the development of Open Data platforms, the wider deployment of Open Data still faces significant barriers. It requires identifying the obstacles that have prevented e-Government bodies either from implementing an Open Data strategy or from ensuring its sustainability. This paper presents the results of a study carried out between June and November 2012, in which we analyzed three cases of Open Data development through their platforms, in a medium size city (Rennes, France.

On their side, government agencies, municipalities and other public entities should address many challenges related with the infrastructure that make data available to the public.

In general, one of the important barriers to the development of the data centric public goods is the ability of public organizations to store and make use of the data. Saying this we think about organizational and economical abilities not necessary about technical know-how.

Aside of business tools for data analysis there are plenty of other open-source tools aimed to cope with data analytics in general, and which can be used for public data acquisition, cleaning, analysis, modeling and visualization. To mention just few of them I would choose: Jasper Reports and Pentaho for reporting, Tableau Public, Google Charts and Google Fusion Tables for data visualization, Octave and Python with Numpy, Scipy, scikit-learn for statistics, machine learning and analytical needs. In this open-source software toolbox R can be consider a powerful tool suitable for the entire stack of statistical and analytical problems.

Governments have a large amount of data with unknown value, until we attempt to find its value. Considering the technical efforts, from a high level point of view, open data should be available in three forms:

1. As application programming interface (API) and formats that allow a user to query and subset data such as json or xml. These are for are developers and programmers;
2. As downloadable files, structured standardized data in highly utilized machine readable formats (csv, kml, xml, and even xls) for researchers, journalists and students;
3. Ready presented for citizens who are looking for information.

DATA-DRIVEN JOURNALISM – AN EXAMPLE OF USING R WITH OPEN DATA

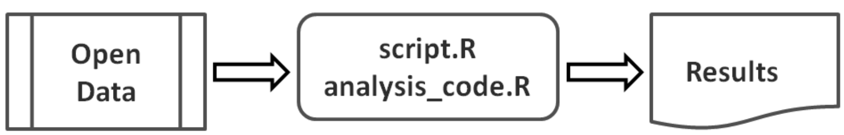
Tim Berners-Lee the inventor of the World Wide Web considers that analyzing data is the future for journalists. Public data in this context is the substance that can support information about politics efficiency, community management, or city evolution. Further on, Tim Berners-Lee asked an interesting and intriguing question: "Who's really going to hold the government, or anyone else, accountable?"[3].

Journalists today have a larger free toolbox available to find stories. Many journalist are already accustomed to use readymade stats, databases, spreadsheets, moreover seasoned journalists use powerful scripts written in languages like Python, Ruby and R for scraping data from the web. Amy Schmitz Weiss from San Diego State University observes that we are now entering in the age of the "Digital Media Data Guru" – this guru is a person with a hybrid of computer science and journalism skills who is able to "do it all" in the newsroom [4].

Trained data journalists can use R to analyze huge datasets that extend the limits of Excel, for instance, a table with a million rows. R is often used as scripting language for file management but especially for data extract-transform-load (ETL) processes. A researcher or journalist can run any script files like in Figure 1 bellow or they can run simple command lines to process streams of data using R scripts. An example on Linux OS is:

cat mathscoresInputFile.csv | Rscript -e 'quantile(as.numeric(readLines("stdin")))' >> resultsOutputFile.csv

Figure 1



Web scraping or web data extraction is a software technique of extracting information from websites. For example a tech savvy journalist may find useful to use an R script like this one bellow to scrape on www.monster.ie website for jobs of interest containing the words "R" and "journalism". Data regarding job descriptions obtained with this method can be analyzed further to depict more frequent requested skills or to find semantic relationship between words used in descriptions. R has an entire set of packages for text mining and analysis e.g. ***tm, RWeka, etc***

```
rm(list=ls())

library(XML)
library(plyr)
require(RCurl)

setwd("E:/WORK_2014/ScrapingStuff")
urls<-c("http://jobsearch.monster.ie/jobs/?q=jounalism-R&cy=ie",
   "http://jobsearch.monster.ie/jobs/?q=jounalism-R&pg=2&cy=ie")

for ( u in urls ) {
   web_page<-readLines(u)
   # Pull out the appropriate line
   jobs_lines <- web_page[grep("slJobTitle", web_page)]

   jobs_lines <- jobs_lines[grep(".aspx", jobs_lines)]

   code<-strsplit(as.character(jobs_lines),'\"',fixed=TRUE)
   vect1<-matrix(NA, ncol=2, nrow=length(jobs_lines))
   for(i in seq(length(jobs_lines))) {
    vect1[i,1] <-code[[i]][15]
    vect1[i,2] <-code[[i]][14]
   }
   vect1<-as.data.frame(vect1)
   vect1
   write.table(vect1, "jobsJournalismandR.tab", append = TRUE,
   sep="\t", row.names = FALSE, col.names = FALSE)
}
```

Open data is fast approachable using R programming language. Why R? Because:

- R is free distributed under terms of the GNU General Public License version 2.
- R has large core statistical analysis toolkit and access to powerful and cutting-edge analytics libraries.
- R is a language and analysis is done by writing functions and scripts. R is an interactive language, it promotes experimentation and exploration.
- R has powerful graphics and data visualization capabilities.
- R has a large community of users and developers.

INFORMED POLITICAL DECISIONS AND BUSINESS VALUE FROM PUBLIC DATA

Politicians and local officials can also benefit from open data. The data that were locked away in department's office desks have become available and accessible. Open data becomes the root of an information platform for viewing the city more holistically and making more informed decisions based on more information [5]. European public funds and projects would be better managed using information from public data. Public data could help in coordination and decisions process at high level for policy makers and also at the low level at individual projects' implementations. For example an agricultural project or an infrastructure project would be better planned based on public data which in present is not really available. We do not consider this is not feasible or data is not available at all, but simplicity of access, proximity, political openness are still only desiderates and not real certainties.

In the near future data for the public good tend to be driven by an eclectic community of media, nonprofits and academics focused on delivering information in different forms to the communities. Public access to government data creates economic and business value and encourages entrepreneurship. Socio-economic census information, traffic patterns, and bus schedules are good data sources for applications and content development, but these do not make an open government [6]NY","genre":"SSRN Scholarly Paper","source":"papers.ssrn.com","event-place":"Rochester, NY","abstract":""Open government" used to carry a hard political edge: it referred to politically sensitive disclosures of government information. The phrase was first used in the 1950s, in the debates leading up to passage of the Freedom of Information Act. But over the last few years, that traditional meaning has blurred, and has shifted toward technology. Open technologies involve sharing data over the Internet, and all kinds of governments can use them, for all kinds of reasons. Recent public policies have stretched the label "open government" to reach any public sector use of these technologies. Thus, "open government data" might refer to data that makes the government as a whole more open (that is, more accountable to the public. There are weekly hackathons around the world that produce a number of useful tools online, including open data tools. To mention few of these tools we would just say Scraper Wiki, Google Refine, mapping and format converters like IssueMa and Copypastemap.

Open sources are often very useful business purposes because they can be easily accessible, inexpensive, quickly accessed and voluminous in availability. Marketers more and more rely on open sources of data in developing strategic plans and tactics [7]. GPS data and weather data usage are just two success cases we want to mention in this sense. Moreover it is hard to imagine in the long run a major achievement with open data without business involvement and support from business partnerships.

A SIMPLE DATA REPORTING PIPELINE – THE FIRST APPROACH TO USE R WITH OPEN DATA

Starting working with R for data analysis might be frustrating and difficult. There are many isolated tutorials on web but the heterogeneity and sparse distribution of thousands of R packages make learning process challenging.

A user of statistical packages tends to run a reduced set of procedures for a specific type of analysis. This analyst might wonder why should learn the R language rather than using a package that provides friendly menus. The answer is still debatable. A statistical package is friendly but often a pricey tool. The main downside of statistical packages is the 'black box' nature of it. With statistical packages analysts can set up the analysis with all the parameters and options that they need; after they run the procedure, the resulting output may be long and verbose. Later they will pull out only the data needed.

The main limitations of statistical packages come together with menus and embedded constraints and assumptions. For example different from a statistical package in R we can change the '*tol'* argument in QR function. This argument controls whether QR decomposition of a matrix will return a value or not for a column depending on whether the column has been judged to be linearly dependent. The R paradigm is very different. With R a researcher has more freedom, flexibility and also more responsibility. An analyst who is using R can go straight to the elements of interests, but in the same time he should be careful to methodologies used, statistical assumptions that come in the process.

PROGRAMMING PARADIGMS IN R

R borrows features from both functional programming languages (Lisp, Scheme) and object oriented programming languages (C++)

```
printHello <- function(name){
    print(paste(“Hello, “, name))
}
```

R has as a system for object-orientation: S3 and S4 are (i.e. built in) approaches for OO programming in R. There are still open debates related with the OOP system robustness in R language.

On the other hand R is a strongly functional language. David Springate offers a performance benchmark example on his blog [8].

```
# Get all even numbers up to 200000
# C style vector allocation:
x <- c()
for(i in 1:200000){
  if(i %% 2 == 0)f
  x <- c(x, i)
}
## user system elapsed
## 9.86 0.00 9.88

# FP style vectorised operation
a <- 1:200000
x <- a[a %% 2 == 0]
## user system elapsed
## 0.01 0.00 0.01
```

Regarding parallel programming, in R it is possible to do concurrent programming, for example running more functions in the same time with while-loops concurrently. The ***snow, Rmpi***, and ***pvm*** packages support these aspects across computers and also on a multi-CPU or multi-core computers. Starting with R 2.14.0, the ***parallel*** package has bundled parts of snow and multi-core in the basic R distribution.

Further we will present a brief data analysis process using R and open data source from the **European data portal**. We will try to enhance the R programming environment capabilities to get the data, explore, model and communicate the results in a meaningful way.

SETTING-UP THE R WORKING ENVIRONMENT

The working process with R is highly interactive. The analysts run a command for each desired granular output. For example, below we prepare the working environment with few commands to clean memory, set working directory, and evaluate its content.

```
rm(list=ls())
#set working directory
setwd("D:/WORK_2014/Articol_Revista_de_Statistica/RContent/DataAnalysisPipeLine")
list.files() # list files from directory

#### Example - Using R to collect archived data from Public Data
Collection
#### This example regards Generation of waste
#### by sector data from EU data portal
```

R has the main advantage of the community on CRAN with over 2500 packages. Nothing will compare with this in the near future, not even a commercial application like SPSS, Matlab or SAS. R and its packages are written primarily in C and Fortran, although it is being extended through other languages. Here is the way we install and use packages:

```
#install.packages("R.utils")
#install.packages("ggplot2")
library("R.utils") #     Instrumental package to work with
archives
library("ggplot2") #   Tremendous package for charts
```

GETTING THE DATA WITH R

R can get archived data (zip, tar.gz etc.) and read a wide spread formats (csv, xls, tab etc). R can easily get data from API's, in web data formats (json, xml, html etc). R is a good scripting language and it can take as input streaming data from big data storages systems like Hadoop. R has capabilities to interact with entire spectrum of databases from relational to no-sql storage systems (Postgres, MongoDB etc.).

```
#   Collect and Mange Public data from URI:
#   http://ec.europa.eu/eurostat/product?code=med_en22
temp <- tempfile() ## use temporary environment to create a temp.
file
download.file("http://epp.eurostat.ec.europa.eu/NavTree_prod/
everybody/BulkDownloadListing?file=data/med_en22.tsv.gz",temp)
dataf <- gunzip(temp, "temp.tab")  ## unzip the temporary file;
R provides full spectrum on file operations
dataf <- read.csv(dataf, sep="\t")   ## read the data from
structured file, Tab Separated Values in this case
unlink(temp)             ## Remove the temp file via unlink()
if (file.exists("temp.tab")) file.remove("temp.tab") ## Remove
temporary file
```

EXPLORING THE DATA WITH R

With few lines of R command code we can generate fast descriptive statistics related with the working data.

```
# These four functions bellow can give an overview of the file structure and content
dim(dataf); str(dataf); summary(dataf)
View(dataf) # a quick view of the small data set
```

R has a very terse syntax. From the very beginning R was designed as a language specific for data processing. Its data structures as data.frame, matrix, and lists make data crunching, manipulation and transformation very efficient.

```
# Data processing phase implies cleaning and manipulating only data of interest
datafII<-dataf[36,-1] #Select the: Thousands of tones of municipal waste cross time in Egypt

# gsub2 - this is an instrumental local created function, used for cleaning data step
# R is a functional programming language; functions can be considered objects in R; here we create a function to be used locally
gsub2 <- function(pattern, replacement, x, ...) {
 for(i in 1:length(pattern))
  x <- gsub(pattern[i], replacement[i], x, ...)
 x
}


# using the gsub2
x<-gsub(„X“,““,names(datafII)); x
y<-as.numeric(gsub2(c(«:  «,»  «),c(«»,»»),as.matrix(datafII)));
y
```

TRANSFORM THE DATA USING R

In most data analysis exercises, preparing the data is more than half of work. The analyst has to find where the data is, has to figure out how to access it, to find the right records, to clean, to filter and transform it before any statistical analysis can be done.

```
# prepare the data structured for plotting
df <- as.data.frame(cbind(x,y))
names(df)<-c("Year", "Quantity")
# R has very fast selection, filtering, merging methods
# also the syntax is very terse, and specific to functional languages
df<-df[!is.na(df[,c(2)]),]

# This line converts the factor levels in numeric values
# here is intentionally enhanced the specificity of main
# data structures in R: data.frames
df$Quantity<-as.numeric(levels(df$Quantity))[df$Quantity]
df$Year<-as.numeric(levels(df$Year))[df$Year]

# Anytime we can interrogate the data to understand the status.
# this is one of the beauty of R environment:
# Interactive environment
head(df)
summary(df)
```

COMMUNICATE AND VISUALIZE THE RESULTS

With tables and plots generated we can learn new things from data and generate useful insights. There are several different graphics systems in R. The oldest one is ***base*** graphics. Base graphics is analogous to drawing on canvas in successive phases. The ***lattice*** and ***ggplot2*** packages provide functions for high-level plots based on grid graphics. Both base and grid graphics are device independent. ***Ggplot2*** provides a unified framework and a set of options and modifiers present in ***base*** graphics. Moreover it is hard to find any visualization method or data wrangling technique that is not already built into R.

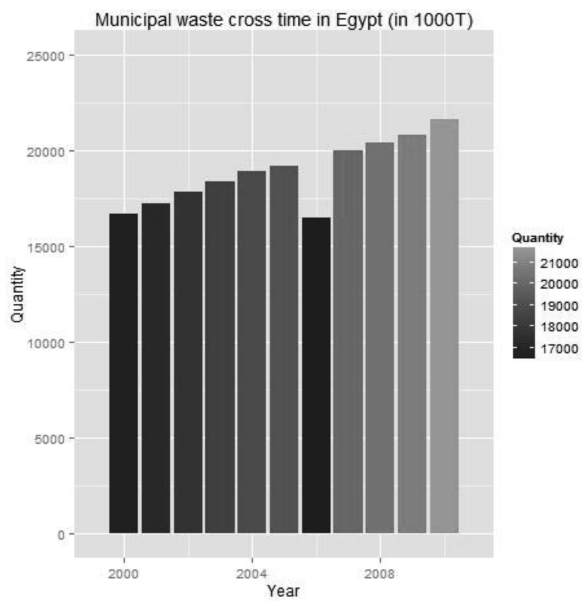
```
#load the package used to save data as MS Excel in a specific sheet
library(xlsx)
write.xlsx(x = df, file = "MunicipalWasteEgypt.xlsx",
      sheetName = "Quantity_by_Year", row.names = FALSE)

# plot the data
p <- ggplot(df, aes(x=Year)) +
 geom_bar(data = df, aes(y=Quantity, fill = Quantity), stat = "identity") +
 scale_x_continuous(expand=c(0.1,0)) +
 ylim(0, 25000) +
 ggtitle("Municipal waste cross time in Egypt (in 1000T)")
p
```

```
#Save the plot to be used later
jpeg("Municipal_waste_cross_time_in_Egypt.jpg")
p
dev.off()
```

Main results from the scripts are presented below. From the chart we can understand the evolution of municipal waste cross time in Egypt. Such kind of charts can support any presentation in a field where we can obtain open data sources.

Chart presenting the Municipal waste cross time in Egypt

Figure 2



Municipal wastes cross time in Egypt

Table 1

Year	Quantity
2010	21632
2009	20800
2008	20400
2007	20000
2006	16500
2005	19200
2004	18900
2003	18400
2002	17800
2001	17200
2000	16700

SOFTWARE APPLICATIONS WITH OPEN DATA – THE SECOND APPROACH TO USE R WITH OPEN DATA

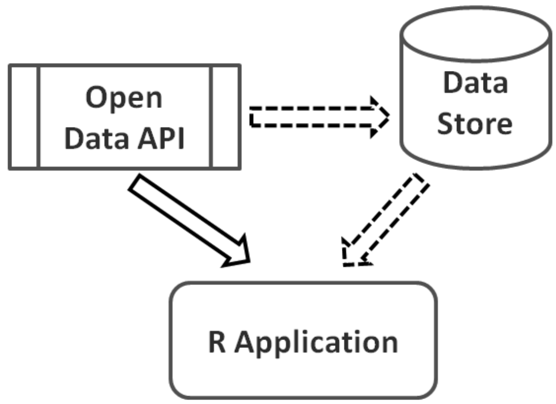
R is to not only for quantitative analysis, but it is used also to construct computer desktop applications and Web applications. The following part of this article presents R capabilities in creating data driven applications with open data and R language.

R-shiny package allows R programmers to transform without much effort their analysis into interactive web applications, accessible by everyone in browsers. Shiny package has embedded prerequisites to build a web application without knowledge of CSS or JavaScript technologies. ***R-shiny*** application allows building full applications for reporting containing controls, sliders, plots tables and summaries. It is designed to work on local port but it has also a server version. Shiny is micro framework which can help statisticians to learn fundamentals of web development. R-shiny is not the single web framework present in R portfolio. There are other packages like ***Rook***, which is a web server interface and an R package in the same time. Rook applications are usually combined with another product ***rApache,*** which is from the same author - Jeffrey Horner**.** This is a framework supporting web application development using the R statistical language and environment and the Apache web server.

An R application can use data sources in two main different forms: 1) from an API; 2) from a data store, database or data source files in repositories.

Diagram presents how an R web application might be sourced with open data

Figure 3



Shiny applications in general contain basically two main R scripts, which are kept within the same folder. They should be named **server.R** and **ui.R.** Besides these two there can be used complementary R scripts to build up further the applications. e.g. **global.R** contains code that has to be run at initiation and is used by the entire application. With `source("codeScript.R")` we can bring into the applications any other R functionalities that we want to have working in the application.

An Example of Data Reporting Web Application on micro framework R-Shiny:

File: **ui.R**

```
library(shiny)

# Define UI for random distribution application
shinyUI(pageWithSidebar(

 # Application title
 headerPanel("Auto Park - Reporting with Public Data"),

 # Sidebar with controls to select the the City and the Detil
related with auto park
  sidebarPanel(
  wellPanel(
   radioButtons("dist", "Report on:",
      list("Cars by Cities" = "cbc",
       "Zoom In" = "parc"))
     ),
```

```
  conditionalPanel(condition = "input.dist=='cbc'",

       wellPanel(
       h4(p(strong("Select the Cities"))),
       selectInput("variableJD1", "City:", jdls),
       selectInput("variableJD2", "City:", jdls),
       selectInput("variableJD3", "City:", jdls),
       selectInput("variableJD4", "City:", jdls)
         )
      ),

  conditionalPanel(condition = "input.dist=='parc'",

       wellPanel(
       h4(p(strong("Select the Detail"))),
       selectInput("variableEL1", "Element:", elem)
         )
      )

     ), # ----close sidebar panel

 # Show a tabset that includes a plot and two table views
 mainPanel(

 conditionalPanel(condition = "input.dist=='cbc'",

      h4(p(strong("Auto Endowment by City"))),

 tabsetPanel(
   tabPanel("City Auto Park", plotOutput("plot",width="1000px",
height="600px")),
   tabPanel("Table Report", tableOutput("table1"))
    )

      ),

 conditionalPanel(condition = "input.dist=='parc'",
      h4(p(strong("Zoom Into Data"))),
      tabsetPanel(
        tabPanel("Zoom Into Auto Park", tableOutput("table2"))
         )
 ))
))
```

File**: server.R**

```
library(shiny)

# Define server logic for random distribution application
shinyServer(function(input, output) {

 output$plot <- renderPlot({

  ls1 <- list(input$variableJD1,input$variableJD2,input$variabl
eJD3,input$variableJD4)
  p<-plotCars(dataf=data1,cityList=ls1)
  print(p)
 })

 # Generate a summary of the data
 output$table1 <- renderTable({
 ls2 <- list(input$variableJD1,input$variableJD2,input$variable
JD3,input$variableJD4)
 dataCarsII(dataf=data1,cityList=ls2)
 })

 # Generate an HTML table view of the data
 output$table2 <- renderTable({
 ls <- list(input$variableJD1,input$variableJD2,input$variableJ
D3,input$variableJD4)
 tableElems(dataf=data1,param=input$variableEL1, cityList=ls)
 })


})
```

File**: global.R**

```
rm(list=ls())
library("ggplot2")
library("data.table")
library("reshape")
library("RColorBrewer")
############Part II########################

data1 <- read.csv("parcautoTopAll.csv")
long<-dim(data1)[1]; long; head(data1)
data1$Serial <- seq(long)
head(data1)

## This method creates the table for park comparisons cross cities
## It is constructed using ggplot2
dataCars <- function(dataf=data1,cityList=list("B","CJ")){
```

```
 #dataf=data1
 options(warn=-1)
 DT <- data.table(dataf, key = c("Serial"))
      dplt<-as.data.frame(DT[,sum(Numar),by=list(PARC.AUTO.2013,Judet)][Judet      %in%
cityList])
 names(dplt)[3] <- c("Numar")
 dplt<-dplt[dplt$PARC.AUTO.2013!="TOTAL",]
 dplt <- as.data.frame(dplt)
 options(warn=0)
 return(dplt)
}

dataCars()

dataCarsII <- function(dataf=data1,cityList=list("CJ","B")){
 options(warn=-1)
 df <- dataCars(dataf,cityList)
 names(df)[3] <- "value"
 df <- cast(df,PARC.AUTO.2013~Judet,sum)
 return(df)
}

dataCarsII(dataf=data1,cityList=list("CJ","B","DJ"))

## This method creates the chart for park comparisons cross cities
## It is constructed using ggplot2
plotCars <- function(dataf=data1,cityList=list("CJ","B")){
 dplt<- dataCars(dataf,cityList)
 colourCount = length(unique(dplt$PARC.AUTO.2013))
 getPalette = colorRampPalette(brewer.pal(9, "Paired"))
 p <- ggplot(data = dplt, aes(x = PARC.AUTO.2013, y = Numar))
 p <- p + geom_bar(aes(fill= PARC.AUTO.2013), stat="identity")
 p <- p + scale_fill_manual(values = getPalette(colourCount))
 p <- p + facet_grid(Judet~.)
 p <- p + theme(axis.text.x = element_text(angle = 90, hjust = 1))
 p
}

## This method creates the table for charatereisics of park components cross cities
tableElems <- function(dataf=data1,param="Carburant_Benzina", cityList=list("B","CJ")){
 options(warn=-1)
 DT <- data.table(dataf, key = c("Serial"))
 df <- DT[Judet %in% cityList][,sum(noquote(Carburant_Benzina), na.rm=TRUE),by=Judet]
 names(df)[2]<-as.character(param)
 options(warn=0)
 df<-as.data.frame(df)
 return(df)
}
```

```
jdls <- list("Total"="TOTAL",
    "Alba"="AB", "Arges"="AG", "Arad"="AR", "Bucuresti"="B",
"Bacau"="BC", "Bihor"="BH",
     "Bistrita-Nasaud"="BN", "Braila"="BR", "Botosani"="BT",
"Brasov"="BV",
      "Buzau"="BZ", "Cluj"="CJ", "Calarsi"="CL", "Caras-
Severin"="CS",
       "Constanta"="CT", "Covasna"="CV", "Dambovita"="DB",
"Dolj"="DJ",
  "Gorj"="GJ", "Galati"="GL", "Giurgiu"="GR", "Hunedoara"="HD",
  "Harghita"="HR", "Ilfov"="IF", "Ialomita"="IL", "Iasi"="IS",
        "Mehedinti"="MH", "Maramures"="MM", "Mures"="MS",
"Neamt"="NT",
  "Olt"="OT", "Prahova"="PH", "Sibiu"="SB", "Salaj"="SJ",
       "Satu-Mare"="SM", "Suceava"="SV", "Teloerman"="TL",
"Timisoara"="TM",
        "Targoviste"="TR", "Valcea"="VL", "Vrancea"="VN",
"Vaslui"="VS")

elem <- c("Numar"="Numar", "Vechime_0_2"="Vechime_0_2",
"Vechime_3_5"="Vechime_3_5",
   "Vechime_6_10"="Vechime_6_10", "Vechime_11_15"="Vechime_11_
15",
   "Vechime_16_20"=»Vechime_16_20», «Vechime_20plus»=»Vechime_
20plus»,
       «Carburant_Motorina»="Carburant_Motorina", "Carburant_
Benzina"="Carburant_Benzina",
  "Vechime_0_4"="Vechime_0_4", "Vechime_5_8"="Vechime_5_8",
   "Vechime_9_12"="Vechime_9_12", "Vechime_peste_12"="Vechime_
peste_12")
```

To run the application we use the following commands:

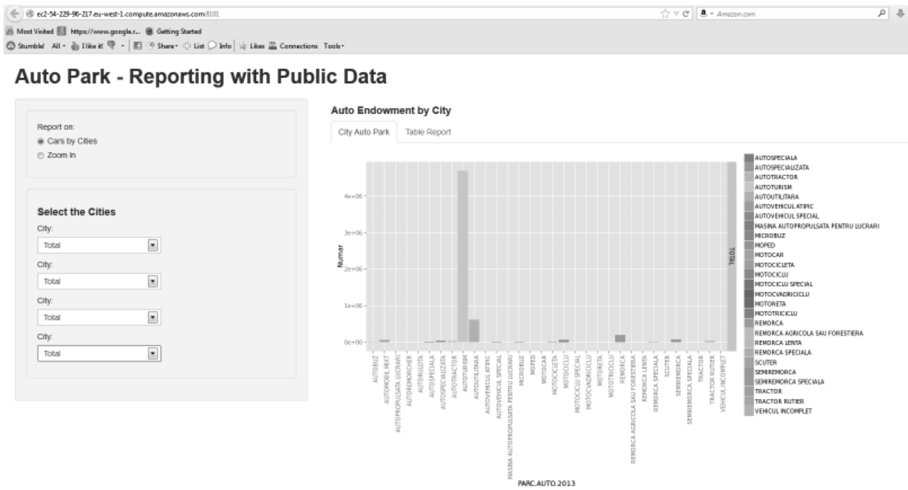
```
library(shiny)
setwd("/home/ubuntu/openapp")
runApp("/home/ubuntu/openapp/rapp", port=8101)
```

The application is running here:
http://ec2-54-229-96-217.eu-west-1.compute.amazonaws.com:8101/

Below a print screen of the main page presents the data chart about car endowment in Romanian Auto Park. Such application can make easier the learning process about socio-economical or political realities using public data.

Print screen of the main page

Figure 4



CONCLUSIONS

Public data has a great potential to generate value for public wealth. Not only for experienced statisticians and analysts but for students, journalists and researchers in general the road from open data to valuable information and insights can be done faster by using R programming language. R language can help not only statisticians but any researcher who has courage to try to solve problems using it. In respect with open data R can come in hand in any analytical step and moreover with R can be built standalone software applications that help extraction of value and wisdom from public data.

References

1. J. Shueh, “Open Data: What Is It and Why Should You Care?,” McClatchy - Tribune Business News, Washington, United States, 17-Mar-2014.
2. S. Martin, M. Foulonneau, S. Turki, and M. Ihadjadene, “Open Data: Barriers, Risks and Opportunities,” in European Conference on e-Government, Kidmore End, United Kingdom, 2013, pp. 301–XVII.
3. C. Arthur, “Analysing data is the future for journalists, says Tim Berners-Lee,” The Guardian, 22-Nov-2010.
4. “At the intersection of journalism, data science, and digital media: How can j-schools prep students for the world they’re headed into?,” Nieman Journalism Lab. .
5. A. Black, “Open Data Movement,” PM Public Manag., vol. 94, no. 6, pp. 22–23, Jul. 2012.
6. H. Yu and D. G. Robinson, “The New Ambiguity of ‘Open Government,’” Social Science Research Network, Rochester, NY, SSRN Scholarly Paper ID 2012489, Feb. 2012.
7. C. S. Fleisher, “Using open source data in developing competitive and marketing intelligence,” Eur. J. Mark., vol. 42, no. 7/8, pp. 852–866, 2008.
8. “DASpringate/samatha,” GitHub. [Online]. Available: https://github.com/DASpringate/samatha. [Accessed: 24-Mar-2014].