Report on US crimes for last 10 years

Preeti Tripathi
School of Information Technology
Illinois State University
ptripa2@ilstu.edu

October 22, 2014

1 Introduction

RStudio is a free and open source integrated development environment (IDE) for R. An R is a programming language for statistical computing and graphics.

1.1 Introduction to Quandl

Quandl is data platform covering millions of time-series datasets from hundreds of sources. Quandl covers all the numerical data in world in different formats. In this paper the data is collected in the csv file format from the following Quandl data source:

$https://www.quandl.com/FBI_UCR/USCRIME_STATE_ILLINOIS$

1.2 Introduction to data

In this tutorial we are taking raw data of all the crimes reported from the year 1965 to 2010. Though we will only be using last 10 years records by scrubing the data from raw data. Also we will only take three columns including the population, violent crimes total and murders and nonnegligent menslaughters.

2 Reading in the data

Reading data means we will read data from csv file in R. This can be done by uploading csv file to R using Upload option in the right hand corner box. Also following are the libraries used in R. For example ggplot2 is used for plotting graph from the clean data.

```
> library(ggplot2)
> library(XML)
> library(reshape2)
> # Upload csv file to R using upload option
```

```
> # Code for reading the csv file
> UScrimes <- read.csv(file="crimes.csv", header=TRUE, sep=",")
```

Here the **read.csv** command reads the csv file in R **file = ""**– In double quotes the name of the file is written which is to be read.

1) header=TRUE: You can choose whether or not to have a header by specifying the optional argument *header=TRUE or header=FALSE* to the read.csv function.

This is important if you have a header but lack row names, because R's guess is based on the fact that the header line has one less entry than the next row

2) sep: the field separator character. Values on each line of the file are separated by this character. If sep = "" (the default) the separator is âĂŸwhite spaceâĂŹ, that is one or more spaces,tabs, newlines or carriage returns. But here it is separated by comma(,).

Use of head command Now since we have read data in R, we want to see how the data looks like in R. So we use *head* command to observe first few rows and all the columns of data.

```
> head (UScrimes)
```

```
        Year Population Violent.Crimes.Total
1 2010-12-31   12830632                 55835
2 2009-12-31   12910409                 64185
3 2008-12-31   12842954                 67840
4 2007-12-31   12852548                 68528
5 2006-12-31   12831970                 69498
6 2005-12-31   12765427                 70496
  Murders.and.Nonnegligent.Manslaughters Forible.Rapes Robberies
1                                    706          3033     20054
2                                    773          3901     22923
3                                    790          4104     24067
4                                    752          4103     23100
5                                    780          4078     23782
6                                    770          4313     23255
  Aggravated.Assaults Property.Crimes.Total Burglaries Larceny.Thefts
1               32042                343989      75399         239794
2               36588                353347      77850         248821
3               38879                381247      78968         269553
4               40573                377322      75524         267911
5               40858                387478      77259         272578
6               42158                394670      77635         277662
  Motor.Vehicle.Thefts Violent.Crime.Rate
1                28796              435.2
2                26676              497.2
3                32726              528.2
4                33887              533.2
```

```
5                37641              541.6
6                39373              552.0
  Murder.and.Nonnegligent.Manslaughter.Rate Forible.Rape.Rate Robbery.Rate
1                                       5.5              23.6        156.3
2                                       6.0              30.2        177.6
3                                       6.2              32.0        187.4
4                                       5.9              31.9        179.7
5                                       6.1              31.8        185.3
6                                       6.0              33.8        182.2
  Aggravated.Assault.Rate Property.Crime.Rate Burglary.Rate Larceny.Theft.Rate
1                   249.7              2681.0         587.6              1868.9
2                   283.4              2736.9         603.0              1927.3
3                   302.7              2968.5         614.9              2098.8
4                   315.7              2935.8         587.6              2084.5
5                   318.4              3019.6         602.1              2124.2
6                   330.3              3092.0         608.2              2175.0
  Motor.Vehicle.Theft.Rate
1                    224.4
2                    206.6
3                    254.8
4                    263.7
5                    293.3
6                    308.0
```

Above is the raw data from which we have to obtain clean data

3 Cleaning the data

Now the next step is to extract the required data from the raw data. The process includes scrubing and cleaning of data.

```
> # We use the following command to clean the data from csv file
> part1<-UScrimes[1:10 ,1:4]
```

Here in *part1* which is name of the file we are storing *1 to 10 rows and 1 to 4 columns*

As we can see in above tables we have inconvenient long column names. The command below allows to change the column names.

```
> colnames(part1) <- c("Year", "Population", "Violent crimes","Murders")
> head(part1,10)

         Year Population Violent crimes Murders
1  2010-12-31   12830632          55835     706
2  2009-12-31   12910409          64185     773
3  2008-12-31   12842954          67840     790
```

```
4  2007-12-31   12852548          68528     752
5  2006-12-31   12831970          69498     780
6  2005-12-31   12765427          70496     770
7  2004-12-31   12712016          69365     780
8  2003-12-31   12649087          70376     895
9  2002-12-31   12586447          75759     961
10 2001-12-31   12520227          79270     982
```

The *colnames* helps to rename columns

4 Write new csv file using clean data

As we have clean data now we can write that clean data in a new csv file format by using the write.csv command

```
> # writing new csv file with cleaned data
> write.csv(part1, file = "part1.csv")
```

Here write.csv is the command. part1 is the table containing the data and part1.csv is the file name for the csv file format

5 Data

Different commands are used to see the data in different ways. For example: 1) Class command:- To find out an object's type use the class command

```
> class(part1)
```

```
[1] "data.frame"
```

After running the *class* command we got the result as *data.frame* which means that ***part1 is a dataframe***

Also we can find the object type of each column. For example:

```
> class(part1$Year)
```

```
[1] "factor"
```

The result says it is a factor. Similarly for others such as population:

```
> class(part1$Population)
```

```
[1] "numeric"
```

The object type is numeric

```
> class(part1$Murders)
```

```
[1] "numeric"
```

The object type is numeri:

2) Structure command: It provides information about the structure of some oject

```
> str(part1)
```

```
'data.frame':        10 obs. of  4 variables:
 $ Year          : Factor w/ 51 levels "1960-12-31","1961-12-31",..: 51 50 49 48 47 46 45 44
 $ Population    : num  12830632 12910409 12842954 12852548 12831970 ...
 $ Violent crimes: num  55835 64185 67840 68528 69498 ...
 $ Murders       : num  706 773 790 752 780 770 780 895 961 982
```

In this structure the line says 10 observations of 4 variables which means we have four variables in the data frame which are: 1) Year 2) Population 3) Violent crimes 4) Murders

3) Summary command: summary is a generic function used to produce result summaries of the results of various model fitting functions

```
> summary(part1)
```

```
         Year     Population       Violent crimes     Murders
 2001-12-31:1   Min.   :12520227   Min.   :55835   Min.   :706.0
 2002-12-31:1   1st Qu.:12664819   1st Qu.:68012   1st Qu.:770.8
 2003-12-31:1   Median :12798030   Median :69432   Median :780.0
 2004-12-31:1   Mean   :12750172   Mean   :69115   Mean   :818.9
 2005-12-31:1   3rd Qu.:12840208   3rd Qu.:70466   3rd Qu.:868.8
 2006-12-31:1   Max.   :12910409   Max.   :79270   Max.   :982.0
 (Other)   :4
```

The **summary** function gives the diffrent values for different columns from the new table such as:

Population:*Minimum* 12520227*Maximun* 12910409 *Median* 2798030*Mean* 12750172

Violent crimes:*Minimum* 55835*Maximun* 79270 *Median* 69432*Mean* 69115

Murders: *Minimum* 706 *Maximun* 982 *Median* 780 *Mean* 818.9

5.1 Result

After cleaning the data and performing various commands we get the following *results*:

1.)**Table**– Now we have four columns in our table with new column names namely *Year* which is a *factor*. *Population* with the object type as *nueric*. *Violent crimes* again with the object type as *numeric* and last column as *Murders* with the objet type *numeric*. All the columns have last ten years record that is from **2001 to 2010**.

According to the Population numbers the population has increased every year **except** from the year 2009 to 2010 the population has decreased from 12910409 to 12830632.

Figures of Violent crimes show that they have decreased from 79270 to 69365 in year 2004 but again increased to 70496 in 2005 and then gradually decreased to 55835 in 2010.

In case of Murders the figures are decreasing from 982 to 706 in 2010 though their was a little increase in between 2007 to 2008.

6 Reshaping the data

The melt function comes from the library reshape2.The melt function takes data in wide format and stacks a set of columns into a single column of data.

```
> # Using the melt command for the
> part1.m <- melt(part1)
> part1.m
```

```
         Year       variable    value
1  2010-12-31     Population 12830632
2  2009-12-31     Population 12910409
3  2008-12-31     Population 12842954
4  2007-12-31     Population 12852548
5  2006-12-31     Population 12831970
6  2005-12-31     Population 12765427
7  2004-12-31     Population 12712016
8  2003-12-31     Population 12649087
9  2002-12-31     Population 12586447
10 2001-12-31     Population 12520227
11 2010-12-31 Violent crimes    55835
12 2009-12-31 Violent crimes    64185
13 2008-12-31 Violent crimes    67840
14 2007-12-31 Violent crimes    68528
15 2006-12-31 Violent crimes    69498
16 2005-12-31 Violent crimes    70496
17 2004-12-31 Violent crimes    69365
18 2003-12-31 Violent crimes    70376
19 2002-12-31 Violent crimes    75759
20 2001-12-31 Violent crimes    79270
21 2010-12-31        Murders      706
22 2009-12-31        Murders      773
23 2008-12-31        Murders      790
24 2007-12-31        Murders      752
25 2006-12-31        Murders      780
26 2005-12-31        Murders      770
27 2004-12-31        Murders      780
```

```
28 2003-12-31        Murders      895
29 2002-12-31        Murders      961
30 2001-12-31        Murders      982
```

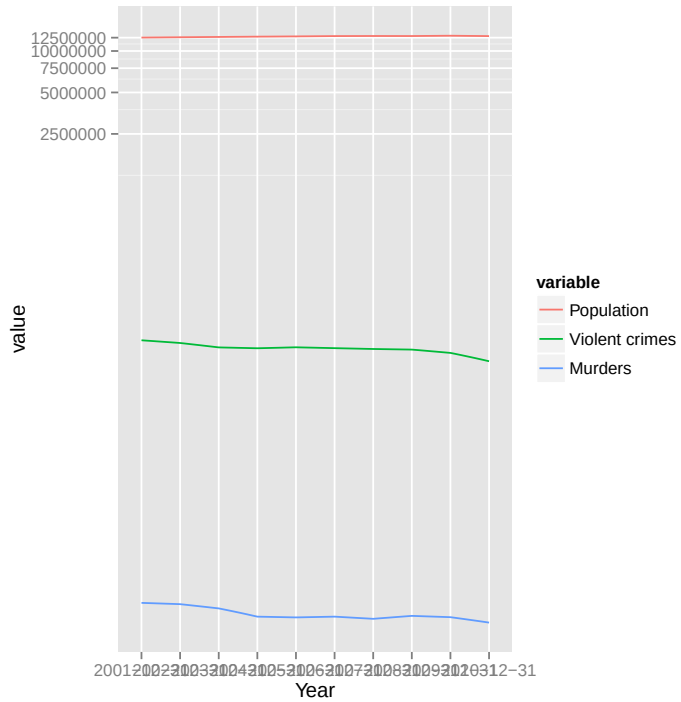
Here melt command is melting the part1 table into part1.m which will be our new table with new table name

Now the part1.m table contains three columns namely Year, Variable and Values. The variable column has three names grouped which are opulation, Violent crimes and Murders and all the figures from these three columns are grouped into value column. We do this step to create the graph or plot.

7 Creating a Plot

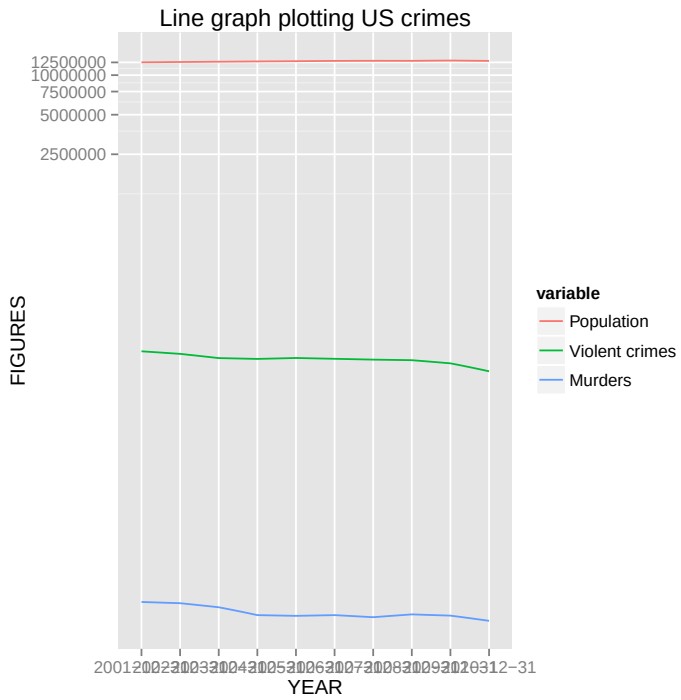
Since we have already imported library ggplot2, we can now use the ggplot command to plot our graph from the melted data.

```
> # Using the ggplot 2 we make a line graph for three variables
> plot<-ggplot(part1.m, aes(x=Year, y=value, group=variable, colour=variable)) +geom_line()+

> print(plot)
```



We can also add labels to graph on X and Y axis as well as give title to graph.

```
> # Now, we want to add a title to the graph and custom legends
> plot1<-ggplot(part1.m, aes(x=Year, y=value, group=variable, colour=variable))+geom_line()+

> print(plot1)
```



In order to increase the width of the plot to 10 we can simply type width = 10 in the (■ fig=true, echo= false, width=10 ■ =) and it will increase the size of plot horizontly

7.1 Result

In the first Graph

The graph shows three coloured lines each for **Population**,**Violent crimes** and **Murders** which were grouped under the cloumn named *variable* in the reshaped table. On the X axis shows Year from 2001 to 2010. and on Y axis is the scale for the figures grouped under the value column of the reshaped table. The population line looks constant due to the large scale used with little variation in values. The violent crimes line in green color shows the value decreasing and the line in blue color named Murders shows small increase and decrease in middle years with final decrease in 2010.

The changes from year 2001 to 2010 explains the population increase within those years while gradual decrease in violent crimes and murders. In conclusion the crime rate has decreased from 2001 to 2010.

In second graph

The second graph merely shows the labeing of X and Y axis with inclusion of title in the graph