Data management by using R: big data clinical research series

Zhongheng Zhang

Department of Critical Care Medicine, Jinhua Municipal Central Hospital, Jinhua Hospital of Zhejiang University, Jinhua 321000, China

Correspondence to: Zhongheng Zhang, MMed. 351#, Mingyue Road, Jinhua 321000, China. Email: zh_zhang1984@hotmail.com.

Author's introduction: Zhongheng Zhang, MMed. Department of Critical Care Medicine, Jinhua Municipal Central Hospital, Jinhua Hospital of Zhejiang University. Dr. Zhongheng Zhang is a fellow physician of the Jinhua Municipal Central Hospital. He graduated from School of Medicine, Zhejiang University in 2009, receiving Master Degree. He has published more than 35 academic papers (science citation indexed) that have been cited for over 200 times. He has been appointed as reviewer for 10 journals, including *Journal of Cardiovascular Medicine*, *Hemodialysis International*, *Journal of Translational Medicine*, *Critical Care*, *International Journal of Clinical Practice*, *Journal of Critical Care*. His major research interests include hemodynamic monitoring in sepsis and septic shock, delirium, and outcome study for critically ill patients. He is experienced in data management and statistical analysis by using R and STATA, big data exploration, systematic review and meta-analysis.

Zhongheng Zhang, MMed.

Abstract: Electronic medical record (EMR) system has been widely used in clinical practice. Instead of traditional record system by hand writing and recording, the EMR makes big data clinical research feasible. The most important feature of big data research is its real-world setting. Furthermore, big data research can provide all aspects of information related to healthcare. However, big data research requires some skills on data management, which however, is always lacking in the curriculum of medical education. This greatly hinders doctors from testing their clinical hypothesis by using EMR. To make ends meet, a series of articles introducing data management techniques are put forward to guide clinicians to big data clinical research. The present educational article firstly introduces some basic knowledge on R language, followed by some data management skills on creating new variables, recoding variables and renaming variables. These are very basic skills and may be used in every project of big data research.

Keywords: Big-data clinical trial; electronic medical record (EMR); R language

Submitted Oct 12, 2015. Accepted for publication Nov 11, 2015.

doi: 10.3978/j.issn.2305-5839.2015.11.26

View this article at: http://dx.doi.org/10.3978/j.issn.2305-5839.2015.11.26

Introduction

Electronic medical record (EMR) system has already been widely used in most hospitals in China, and it can serve as a potential reservoir to provide resources for clinical research (1-3). EMR can be regarded as a form of big data because the data volume of EMR is ever expanding (4). All information of a patient, from outpatient medication to inpatient management, can be easily extracted from established database. Furthermore, hospital admissions and outpatient visits can be linked together by using unique patient identification number. However, clinicians are always experts in clinical practice but lack necessary skills in the management big data. They are usually overwhelmed by the complexity of the structure of EMR data. As a result, many research questions cannot be answered by using the readily available big data. To conduct a prospective experimental studies or observational studies is usually time consuming and even impossible for the extremely busy clinicians. To make ends meet, a series of articles introducing data management skills are put forward to guide clinicians to big data clinical research.

R

R is not a name of software, but it is a language and environment for data management, graphic plotting and statistical analysis (5,6). R is freely available and is an open source environment that is supported by world research community. Thousands of statistical and graphing packages are available for use, and the package pool is ever expanding. One attractive feature of R is its graphing capability, allowing for nearly any customizations of figures (7). R can be downloaded from Comprehensive R Archive Network (CRAN) at http://cran.r-project.org. Users can easily complete the installation following guidance on the website. After installation, R console should be launched where one can input commends for data analysis. In the following sections, I assume that users have already been familiar with the preparations of R. This section will focus on creating and recoding variables, renaming variable. Although these techniques look simple, they must be used in each project of big data exploration.

For the ease of reading, I denote codes that can be executed in R console with the beginning symbol ">".

Working example

A data frame is created to include original variables such as PaO_2, FiO_2, Glasgow coma scale (GCS), mean arterial pressure (MAP), bilirubin, platelet count, creatinine and urine output. The simulated dataset was used for illustration purpose, and it has no practical meaning.

Continuous variables are created by assuming that they have a normal distribution, whereas categorical variables are created by assuming they have binomial distribution.

```
>pao2<-round(rnorm(100, mean = 300, sd = 30))
>fio2<-round(rnorm(100, mean = 0.5, sd = 0.1),2)
>gcs<-round(rnorm(100, mean = 80, sd = 20)/10)
>map<-round(rnorm(100, mean = 65, sd = 15))
>dop<- round(rnorm(100, mean = 10, sd = 3),1)
>dob<- rbinom(100, 1, 0.4)
>epi<- round(rnorm(100, mean = 10, sd = 3)/100,2)
>nor<- round(rnorm(100, mean = 10, sd = 3)/100,2)
>bilirubin<-round(rnorm(100, mean = 80, sd = 20),1)
>platelet<-round(rnorm(100, mean = 180, sd = 50))
>cr<-round(rnorm(100, mean = 150, sd = 34))
>uo<-rnorm(100, mean = 1000, sd = 500)
>uo<-round(ifelse(uo>0,uo,-uo))
```

The function rnorm() is used to randomly generate variables with mean and standard deviation (sd) arguments. Because generated observations may have decimal places which is not in line with the real world situation, round() function was used to obtain integers (e.g., this function was applied to variable uo and GCS). Ifelse() function was used to convert negative variables into positive ones. rbinom() function is used to generate categorical variables with binomial distribution. In the calculation of SOFA score, we only need to know whether dobutamine is used or not. However, these steps generated separate vectors that are stored in R workplace and we need to combine them into a data frame.

```
>data<-data.frame(pao2,fio2,gcs,map,dop,
dob,epi,nor,bilirubin,platelet,cr,uo)
>head(data,8)
```

The results are shown in *Table 1*. There are 100 observations in the dataset but only the first 8 are displayed. The first row is the variable name and the first column is the number of observations. This dataset illustrates a typical example that we can extract from EMR. We will use it for the illustration of several basic R functions in the following sections.

Table 1 Simulated dataset for illustration (with the first 8 observations displayed)

No.	PaO_2	FiO_2	GCS	MAP	Dop	Dob	Epi	Nor	Bilirubin	Platelet	Cr	Uo
1	326	0.63	9	66	14.6	0	0.11	0.09	70.6	246	164	1144
2	308	0.51	8	59	10.4	0	0.10	0.10	67.8	138	196	578
3	274	0.66	7	42	12.6	1	0.13	0.10	91.2	128	123	1629
4	291	0.57	11	69	6.2	1	0.12	0.09	80.7	94	158	619
5	269	0.50	6	67	11.6	0	0.08	0.11	39.7	383	86	989
6	328	0.54	11	61	7.5	1	0.07	0.12	60.9	190	129	39
7	322	0.61	8	64	7.9	1	0.07	0.05	60.6	182	117	1129
8	298	0.40	5	68	5.4	1	0.06	0.08	72.9	92	141	450

GCS, Glasgow coma scale; MAP, mean arterial pressure; Dop, dopamine; Dob, dobutamine; Epi, epinephrine; Nor, norepinephrine; Cr, creatinine; Uo, urine output.

Creating new variable

It is common to create new variables in data analysis. These variables are called secondary variable, to distinguish them from original variable that can be extracted directly from EMR. In clinical practice, the most widely used secondary variable is varieties of scores. Particularly in critical care medicine, there are numerous risk stratification scores that can be calculated from original physiological and laboratory variables. Sequential Organ Failure Assessment (SOFA) score is one of such example and I would like to illustrate how it can be calculated from original variables.

SOFA score is used to determine the extent of organ functions. It is based on six different scores for respiratory, hepatic, cardiovascular, coagulation, neurological and renal systems (*Table 2*) (8). The SOFA score equals the sum of scores of each organ system, and it simplifies multi-dimension parameters into one dimension.

First, we calculate scores for each organ system. Because there are five categories for each system, we use the ifelse() function.

```
>data$respiratory<-ifelse(data$pao2/data$fio2>=400,0,
                          ifelse(data$pao2/ data$fio2>=300,1,
                           ifelse(data$pao2/ data$fio2>=200,2,
                            ifelse(data$pao2/ data$fio2>=100,3,4))))
>data$neuro<-ifelse(data$gcs >14,0,
                    ifelse(data$gcs >=13,1,
                     ifelse(data$gcs >=10,2,
                      ifelse(data$gcs >=6,3,4))))
>data$liver<-ifelse(data$ bilirubin <20,0,
                    ifelse(data$ bilirubin <=32,1,
                     ifelse(data$ bilirubin <=101,2,
                      ifelse(data$ bilirubin<=204,3,4))))
>data$coagulation<-ifelse(data$ platelet >=150,0,
                          ifelse(data$ platelet >=100,1,
                           ifelse(data$ platelet >=50,2,
                            ifelse(data$ platelet >=20,3,4))))
```

The renal score is a little more complex than others because it encompasses serum creatinine and urine output. The "or" relationship means that for an individual patient we used the maximum scores derived from either urine output or creatinine. Therefore, we first calculate scores for urine output or creatinine, respectively; then the maximum score was used as the final renal score.

Table 2 Components of Sequential Organ Failure Assessment (SOFA) score

Systems	0	1	2	3	4
Respiratory (PaO_2/FiO_2) (mmHg)	≥400	300-400	200-300	100-200 and mechanical ventilation	<100 and mechanical ventilation
Neurological (GCS)	>14	13-14	10-12	6-9	<6
Cardiovascular (MAP or vasopressor[¶])	MAP ≥70	MAP <70	Dop ≤5 or dob (any dose)	Dop >5 or epi ≤0.1 or nor ≤0.1	Dop >15 or epi >0.1 or nor >0.1
Liver [Bilirubin (μmol/L)]	<20	20-32	33-101	102-204	>204
Coagulation (platelets $\times 10^3/\mu L$)	≥150	100-150	50-100	20-50	<20
Renal [creatinine (μmol/L) or urine output]	<110	110-170	171-229	300-440 (or <500 mL/d)	>440 (or <200 mL/d)

¶, vasopressor drug doses are in μg/kg/min. GCS, Glasgow coma scale; MAP, mean arterial pressure; Dop, dopamine; dob, dobutamine; epi, epinephrine; nor, norepinephrine.

```
>cr.score<-ifelse(data$ cr <110,0,
                  ifelse(data$ cr <=170,1,
                    ifelse(data$ cr <=229,2,
                      ifelse(data$ cr <=440,3,4))))
>uo.score<-ifelse(data$uo>=500,0,ifelse(data$uo >=200,3,4))
>data$renal<-max(cr.score, uo.score)
```

Scores for cardiovascular system comprises five variables (e.g., map, dop, dob, epi and nor). We can apply max() function with more variables as its arguments.

```
>map.score<-ifelse(data$map>=70,0,1)
>dop.score<-ifelse(data$dop<=5,2,ifelse(data$dop<=15,3,4))
>dob.score<-ifelse(data$dob==1,2,0)
>epi.score<-ifelse(data$epi==0,0,ifelse(data$epi <=0.1,3,4))
>nor.score<-ifelse(data$nor==0,0,ifelse(data$nor <=0.1,3,4))
>data$cardio<-max(map.score, dop.score, dob.score,
epi.score, nor.score)
```

Then the total SOFA score can be calculated by summing all individual system scores.

```
>data$sofa.score<- data$cardio+ data$renal+
data$coagulation+ data$liver+ data$neuro+ data$respiratory
>head(data)
```

We can now take a look at the new dataset by using head() function (*Table 3*). The dataset contains scores for each individual system (from variable cardio to coagulation) and the last column is the SOFA score.

Recoding variables

The most commonly used technique in recoding variable is to change a continuous variable into a set of categories. To recode data, one can use R's logical operators (*Table 4*). These logical operators create logical expression and return the value of TRUE or FALSE.

Suppose that we want to make classifications of acute respiratory distress syndrome (ARDS) based on Berlin definition. This definition proposed 3 mutually exclusive categories of ARDS based on degree of hypoxemia: severe (PaO_2/FiO_2 ≤100 mmHg), moderate (100 mmHg < PaO_2/FiO_2 ≤200 mmHg), and mild (200 mmHg < PaO_2/FiO_2 ≤300 mmHg) (9). Then we can recode continuous variable PaO_2/FiO_2 to categorical variable berlin (severe, moderate and mild). First, we create a new variable named 'oxyindex'.

Table 3 New dataset comprising original variables and newly created variables

No.	PaO_2	FiO_2	GCS	MAP	Dop	Dob	Epi	Nor	Bilirubin	Platelet	Cr	Uo	Cardio	Renal	Respiratory	Neuro	Liver	Coagulation	Sofa.score
1	326	0.63	9	66	14.6	0	0.11	0.09	70.6	246	164	1,144	4	4	0	3	2	0	13
2	308	0.51	8	59	10.4	0	0.10	0.10	67.8	138	196	578	4	4	0	3	2	1	14
3	274	0.66	7	42	12.6	1	0.13	0.10	91.2	128	123	1,629	4	4	0	3	2	1	14
4	291	0.57	11	69	6.2	1	0.12	0.09	80.7	94	158	619	4	4	0	2	2	2	14
5	269	0.50	6	67	11.6	0	0.08	0.11	39.7	383	86	989	4	4	0	3	2	0	13
6	328	0.54	11	61	7.5	1	0.07	0.12	60.9	190	129	39	4	4	0	2	2	0	12

GCS, Glasgow coma scale; MAP, mean arterial pressure; Dop, dopamine; Dob, dobutamine; Epi, epinephrine; Nor, norepinephrine; Cr, creatinine; Uo, urine output.

Table 4 Logical operators in R

Operators	Meaning
<	Less than
<=	Less than or equal to
>	More than
>=	More than or equal to
==	Equal to
!=	Not equal to
!a	Not a
a|b	a or b
a&b	a and b
isTRUE(a)	Test if a is true

```
>data$oxyindex<- data$pao2/data$fio2
```

Because patients with oxygen index greater than 500 are less likely to have ARDS, we need to exclude them.

```
>data$oxyindex[data$oxyindex>=500]<-NA
```

This statement assigned null (NA) values to observations with oxygen index greater than 500. The statement 'variable[conditions]<-expression' takes value from expression when the condition is TRUE.

Next we can use the following code to create the berlin variable:

```
>data$berlin[data$oxyindex<=100] <- "severe"
>data$berlin [data$oxyindex >100 &data$oxyindex <= 200]
<- "moderate"
>data$berlin [data$oxyindex >200] <- "mild"
```

Data frame names are used in these codes to ensure the new variables are saved back to the data frame. Alternatively, if you do not want to repeat data frame name, you can use within() function to write the code more compactly.

```
>data <- within(data,{
            berlin <- NA
            berlin[oxyindex<=100] <- "severe"
            berlin[oxyindex>100 & oxyindex<=200] <- "moderate"
            berlin[oxyindex>200] <- "mild"& oxyindex<500})
```

Another very useful function shipped with R is cut(), which is able to convert continuous variable into factor variable.

```
>data$berlin<-cut(data$oxyindex,breaks
=c(500,300,200,100),labels=c("severe","moderate","mild"))
```

The first argument of cut() function is a numeric vector to be converted to a factor variable. The second argument can be a numeric vector containing two or more cutoff points. The last argument labels each level of the factor variable. Note that we do not need to specify NA values to values greater than 500, and the newly created factor variable automatically excludes observations with oxygen index >500.

Renaming variables

When working with big data, you may have hundreds of variables at hand and you need to rename variables to avoid confusion. In our example, if you are not happy with the name oxyindex, you can change it by using names() function.

```
>names(data)[20]<- "oxygen.index "
>names(data)[11]<- "creatinine"
```

As you can see, the names() function extract variable names of a data frame.

```
>names(data)
[1] "pao2"       "fio2"       "gcs"        "map"        "dop"
[6] "dob"        "epi"        "nor"        "bilirubin"  "platelet"
[11] "cr"        "uo"         "cardio"     "renal"      "respiratory"
[16] "neuro"     "liver"      "coagulation" "sofa.score"  "oxyindex"
[21] "berlin"
```

Alternatively, you can use rename() function for the same purpose (10). Rename() function is in the reshape package and you should load it to the working environment first.

```
>install.packages("reshape")
>library(reshape)
>data <- rename(data,
                c(oxyindex="oxygen.index", cr="creatinine")
                )
```

This code looks more compact and it is particularly useful to change a series of variable names.

Summary

The educational article introduces some basic R functions for data management. Differently from other educational material, this article illustrates the R functions in the context of clinical research. The questions discussed were those that have been encountered by the author, and were considered as the most fundamental skills. Creating new variable is to create new variable based on other original variables that can be directly extracted from EMR. The ifelse() function is very useful. In recoding variables, logistical operators are always used. There is also a useful function called cut() which is able to convert continuous variable into factor variables. Renaming variable is applied when there are hundreds of variables and you are not satisfied with its original forms. The principle of naming variable is to make it concise and informative.

Acknowledgements

None.

Footnote

Conflicts of Interest: The author has no conflicts of interest to declare.

References

1. Zhang Z. Big data and clinical research: focusing on the area of critical care medicine in mainland China. Quant Imaging Med Surg 2014;4:426-9.
2. Zhang Z. Big data and clinical research: perspective from a clinician. J Thorac Dis 2014;6:1659-64.
3. Monteith S, Glenn T, Geddes J, et al. Big data are coming to psychiatry: a general introduction. Int J Bipolar Disord 2015;3:21.
4. Potash JB. Electronic medical records: fast track to big data in bipolar disorder. Am J Psychiatry 2015;172:310-1.
5. Kabacoff R. R in Action. Shelter Island: Manning Publications Co., 2011.
6. Lander JP. R for Everyone: Advanced Analytics and Graphics. Boston: Addison-Wesley Professional, 2014.
7. Horton NJ, Kleinman K. Using R for data management, statistical analysis, and graphics. Clermont: CRC Press, 2010.
8. Vincent JL, Moreno R, Takala J, et al. The SOFA (Sepsis-related Organ Failure Assessment) score to describe organ dysfunction/failure. Intensive Care Med 1996;22:707-10.
9. ARDS Definition Task Force, Ranieri VM, Rubenfeld GD, et al. Acute respiratory distress syndrome: the Berlin Definition. JAMA 2012;307:2526-33.
10. Wickham H. Reshaping data with the reshape package. Journal of Statistical Software 2007;21:1-20.

Cite this article as: Zhang Z. Data management by using R: big data clinical research series. Ann Transl Med 2015;3(20):303. doi: 10.3978/j.issn.2305-5839.2015.11.26

