Tutorial 8: Data Management and Measurement Error

Jan Vogler (jan.vogler@duke.edu)

October 21, 2016

Today's Agenda

1. Data management I: reading/subsetting data, keeping/deleting variables
2. Data management II: data transformation
3. Data management III: creating new variables
4. Data management IV: useful commands
5. Measurement error

1. Data management I: reading/subsetting data, keeping/deleting variables

R can save its own datafiles. Those will have the format ".Rdata". In order to load such a dataset, you can simply use the load command.

```
setwd("C:/Users/Jan/OneDrive/Documents/GitHub/ps630_lab/ps630_f16/W8/")
load("filename.Rdata")
```

R can also read some other file formats, including .csv and .txt files.

```
### .csv files
csvdata = read.csv("filename.csv", stringsAsFactors = FALSE)
# stringsAsFactors=FALSE is important because otherwise R will load
# character variables as factors, treating them as numerical under the
# surfae, which can lead to complications later on

### .txt files
textdata = read.table("filename.txt", header = TRUE)

# header=TRUE indicates that the first row contains the names of the
# variables (see later for example)
```

Additionally, there are many other data formats (SPSS .sav and Stata .dta files). Some of them require the foreign package.

Use the following command to install it:

install.packages("foreign")

```
library(foreign)
spssdata = read.spss("filename.sav", to.data.frame = TRUE)

library(foreign)
statadata = read.dta("filename.dta")
```

For data files that were saved by the more recent versions of Stata (13 and above), you will need another package called "readstata13". Please use the following command to install it:

install.packages("readstata13")

```
library(readstata13)
read.dta13("filename.dta")
```

Let us now read the LDC dataset.

```
setwd("C:/Users/Jan/OneDrive/Documents/GitHub/ps630_lab/ps630_f16/")
LDC = read.dta("LDC_IO_replication.dta")
```

To delete a variable in a data frame, we simply use the following command:

```
LDC$ecris2 = NULL
```

We can also only keep the data that we actually need for our analysis:

```
LDC = LDC[, c("ctylabel", "date", "polityiv_update2", "gdp_pc_95d", "newtar",
    "l1polity", "l1polity", "l1signed", "l1office", "l1gdp_pc", "l1lnpop", "l1ecris2",
    "l1bpc1", "l1avnewtar", "l1fdi")]
```

The following command allows us to only look at cases on which we have all observations of specific variables.

```
complete = with(LDC, complete.cases(polityiv_update2, gdp_pc_95d))
# The 'with' command allows you to evaluate a file for certain expressions,
# such as complete.cases
LDC = LDC[complete, ]
# Then we subset the file and only take the complete cases
```

This is often a better solution than na.omit because na.omit will remove all rows with missing values, which can result in the loss of too many values

```
LDComit = na.omit(LDC)
# only 383 observations remaining if we use this method
```

Before doing any analysis, you should inspect your variables closely. Make sure that you are aware of the properties of your most important variables.

```
summary(LDC$polityiv_update2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -10.000  -7.000  -4.000  -1.121   7.000  10.000
```

```
summary(LDC$gdp_pc_95d)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    91.02   380.70   978.00  2320.00  2437.00 44160.00
```

```r
summary(LDC$newtar)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    0.10   11.70   17.80   21.73   27.50  102.20    2014
```

You have learned how to merge data in one of the previous tutorials. But what happens if we want to merge data with different numbers of observations?

```r
a = c("Household A", "Household B")
b = c(2, 2)
data1 = data.frame(a, b)
colnames(data1) = c("Name", "Number of people")

data1
```

```
##          Name Number of people
## 1 Household A                2
## 2 Household B                2
```

```r
c = c("Household A", "Household A", "Household B", "Household B")
d = c("Individual 1", "Individual 2", "Individual 3", "Individual 4")

data2 = data.frame(c, d)
colnames(data2) = c("Name", "Person")

data2
```

```
##          Name       Person
## 1 Household A Individual 1
## 2 Household A Individual 2
## 3 Household B Individual 3
## 4 Household B Individual 4
```

As we can see, the data has different numbers of observations on the household level.

```r
merge(data1, data2, by = ("Name"))
```

```
##          Name Number of people       Person
## 1 Household A                2 Individual 1
## 2 Household A                2 Individual 2
## 3 Household B                2 Individual 3
## 4 Household B                2 Individual 4
```

As we can see, the data will expand.

Merge has an option that can be helpful. If you have two data frames “x” (here: data1) and “y” (here: data2), you can decide to keep all parts of either data frame or both. This is useful because the merge command normally deletes observations that are not

```r
merge(x = data1, y = data2, all = TRUE, all.x = TRUE, all.y = TRUE)
```

```
##          Name Number of people       Person
## 1 Household A                2 Individual 1
## 2 Household A                2 Individual 2
## 3 Household B                2 Individual 3
## 4 Household B                2 Individual 4
```

When merging we can think of our data frames as the "master" and "merging" data frame. Often our master data frame contains the crucial variables of our analysis. We then combine it with additional "merging" data frames. If we consider the variables in our master data frame to be most important for the analysis, it often makes sense to keep all entries of this data frame (by using all.x=TRUE). In many cases, one has to be careful when merging data frames because it might be the case that several observations in the merging data frame match an entry in the master data frame, even though the researcher does not want to match one with multiple observations.

Let's assume we are only interested in a subset of the data. For example, we are only interested in the country Angola. How can we subset the data that we have?

```r
LDC2 = subset(LDC, ctylabel == "Angola")
summary(LDC2)
```

```
##    ctylabel              date      polityiv_update2   gdp_pc_95d
##  Length:14          Min.   :1980   Min.   :-7.000   Min.   :519.6
##  Class :character   1st Qu.:1983   1st Qu.:-7.000   1st Qu.:643.8
##  Mode  :character   Median :1986   Median :-7.000   Median :675.3
##                     Mean   :1988   Mean   :-6.143   Mean   :651.0
##                     3rd Qu.:1990   3rd Qu.:-7.000   3rd Qu.:707.4
##                     Max.   :1999   Max.   :-3.000   Max.   :732.4
##
##      newtar       l1polity        l1polity.1        l1signed
##  Min.   : NA   Min.   :-7.000   Min.   :-7.000   Min.   :0
##  1st Qu.: NA   1st Qu.:-7.000   1st Qu.:-7.000   1st Qu.:0
##  Median : NA   Median :-7.000   Median :-7.000   Median :0
##  Mean   :NaN   Mean   :-6.385   Mean   :-6.385   Mean   :0
##  3rd Qu.: NA   3rd Qu.:-7.000   3rd Qu.:-7.000   3rd Qu.:0
##  Max.   : NA   Max.   :-3.000   Max.   :-3.000   Max.   :0
##  NA's   :14    NA's   :1        NA's   :1
##     l1office         l1gdp_pc        l1lnpop         l1ecris2
##  Min.   : 1.000   Min.   :504.1   Min.   :15.74   Min.   :0.0000
##  1st Qu.: 4.000   1st Qu.:641.4   1st Qu.:15.82   1st Qu.:0.0000
##  Median : 6.000   Median :672.5   Median :15.91   Median :0.0000
##  Mean   : 7.231   Mean   :646.8   Mean   :15.96   Mean   :0.1429
##  3rd Qu.: 9.000   3rd Qu.:713.1   3rd Qu.:16.00   3rd Qu.:0.0000
##  Max.   :18.000   Max.   :732.4   Max.   :16.30   Max.   :1.0000
##  NA's   :1        NA's   :1
##      l1bpc1     l1avnewtar        l1fdi
##  Min.   :1    Min.   : 0.00   Min.   : 1.718
##  1st Qu.:1    1st Qu.:17.05   1st Qu.: 2.421
##  Median :1    Median :24.69   Median : 4.211
##  Mean   :1    Mean   :22.28   Mean   : 8.115
##  3rd Qu.:1    3rd Qu.:28.25   3rd Qu.: 5.726
##  Max.   :1    Max.   :30.52   Max.   :36.361
##  NA's   :12                   NA's   :6
```

```
# We are left with observations for Angola only.  We can also apply this to
# numerical variables and set conditions for subsetting.
```

What would you do if you have multiple datafiles that all have a similar name. How can you load this data very efficiently?

Credit to Brett Gall for this chunk of the code.

Let's assume we have 1000 different individuals and their files have similar names. How can we load those files without writing 1000 separate commands for loading the data?

```
setwd("C:/Users/Jan/OneDrive/Documents/GitHub/ps630_lab/ps630_f16/W8/")
ind.files = dir(pattern = "ind\\d+.sav")
# This command allows us to get the names of the datafiles Note that \\d+
# is a so-called 'regular expression' \\ initiates the regular expression
# 'd' stands for a digit + stands for one or more (digit) Google 'regular
# expressions' to learn how to construct more regular expressions

ind.data = lapply(ind.files, read.spss)
# This loads the data via the 'lapply' command Here we apply the command
# 'read.spss' to each element of our list read.spss requires the foreign
# package

names(ind.data) = gsub(".sav", "", ind.files)
# Rename list elements, remove '.sav' and replace it with '' (empty)

names(ind.data)
```

```
## [1] "ind1" "ind2" "ind3" "ind4" "ind5"
```

2. Data management II: data transformation and recoding

We can transform variables in a number of ways. One of the most common ways to transform data is to square it to estimate curvilinear relationships. Let us construct a squared version of the Polity IV Score.

In order to make the square meaningful, we first have to add +10 to all Polity Scores (otherwise negative squares would get the same positive values as the positive square).

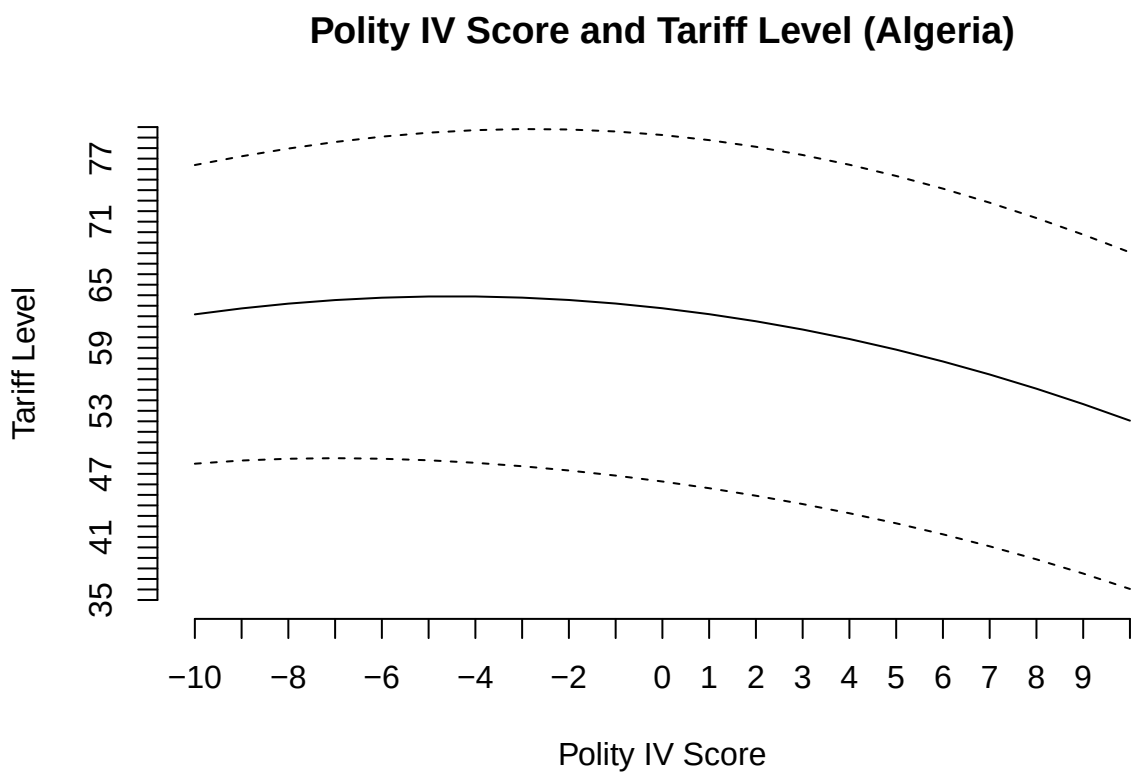
```
setwd("C:/Users/Jan/OneDrive/Documents/GitHub/ps630_lab/ps630_f16/")
LDC = read.dta("LDC_IO_replication.dta")
LDC$polityiv_update2 = LDC$polityiv_update2 + 10
LDC$l1polity = LDC$l1polity + 10
LDC$polityiv_squared = (LDC$polityiv_update2)^2
LDC$l1polity_squared = (LDC$l1polity)^2
```

Let us see whether there is a curvilinear relationship between the Polity IV Score and tariff levels.

```
main_int = lm(newtar ~ l1polity + l1polity_squared + l1signed + l1office + l1gdp_pc +
    l1lnpop + l1ecris2 + l1bpc1 + l1avnewtar + factor(ctylabel) - 1, data = LDC)
# summary(main_int)
```

Interestingly, there appears to be a curvilinear relationship between the two variables! How would we interpret the results of the linear regression? How would we plot this curvilinear relationship?

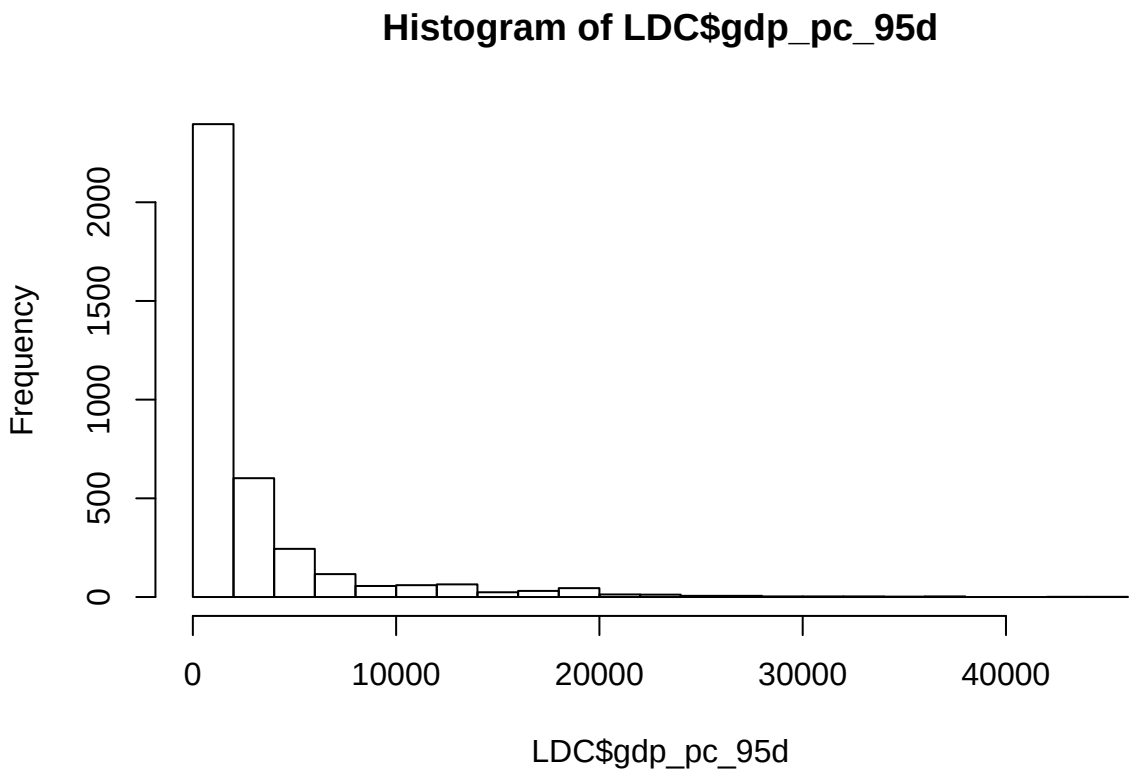
```r
nd <- data.frame(l1polity = seq(0, 20, by = 1), l1polity_squared = seq(0, 20,
    by = 1)^2, l1signed = rep(0.1511, 21), l1office = rep(8.431, 21), l1gdp_pc = rep(2888,
    21), l1lnpop = rep(15.1, 21), l1ecris2 = rep(0.0641, 21), l1bpc1 = rep(0.5909,
    21), l1avnewtar = rep(14.91, 21), ctylabel = rep("Algeria", 21))

pred.p1 <- predict(main_int, type = "response", se.fit = TRUE, newdata = nd)
pred.table <- cbind(pred.p1$fit, pred.p1$se.fit)

fit <- pred.p1$fit
low <- pred.p1$fit - 2 * pred.p1$se.fit
high <- pred.p1$fit + 2 * pred.p1$se.fit
cis <- cbind(fit, low, high)

plot(pred.p1$fit, type = "l", ylim = c(35, 80), main = "Polity IV Score and Tariff Level (Algeria)",
    xlab = "Polity IV Score", ylab = "Tariff Level", axes = FALSE)
axis(1, at = seq(1, 21), labels = seq(-10, 10, 1))
axis(2, at = seq(35, 80), labels = seq(35, 80))
matlines(cis[, c(2, 3)], lty = 2, col = "black")
```



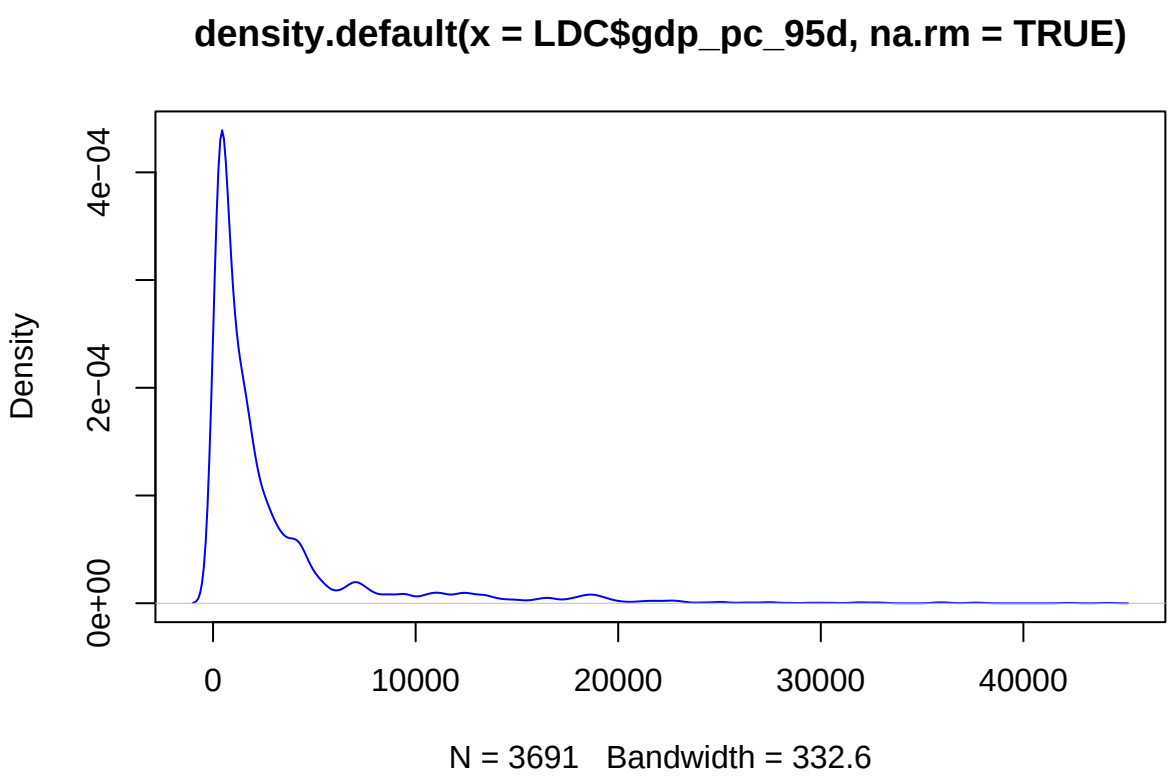
This relationship looks slightly different than the relationship estimated by Milner and Kubota. How is it different?

Moreover, another frequently used way to transform data is to take the natural logarithm. In many cases, researchers do this because the distribution of the data is skewed to the right. The goal is to reduce the skewness.

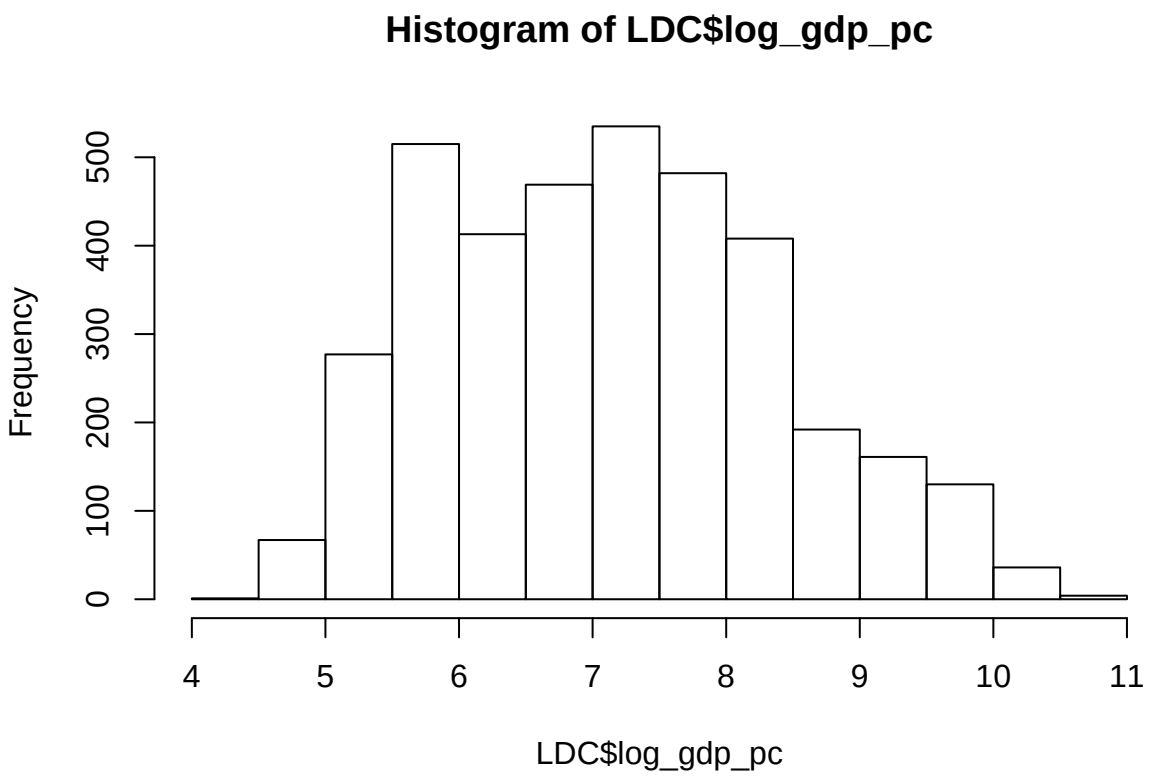
```r
hist(LDC$gdp_pc_95d, breaks = 21)
```



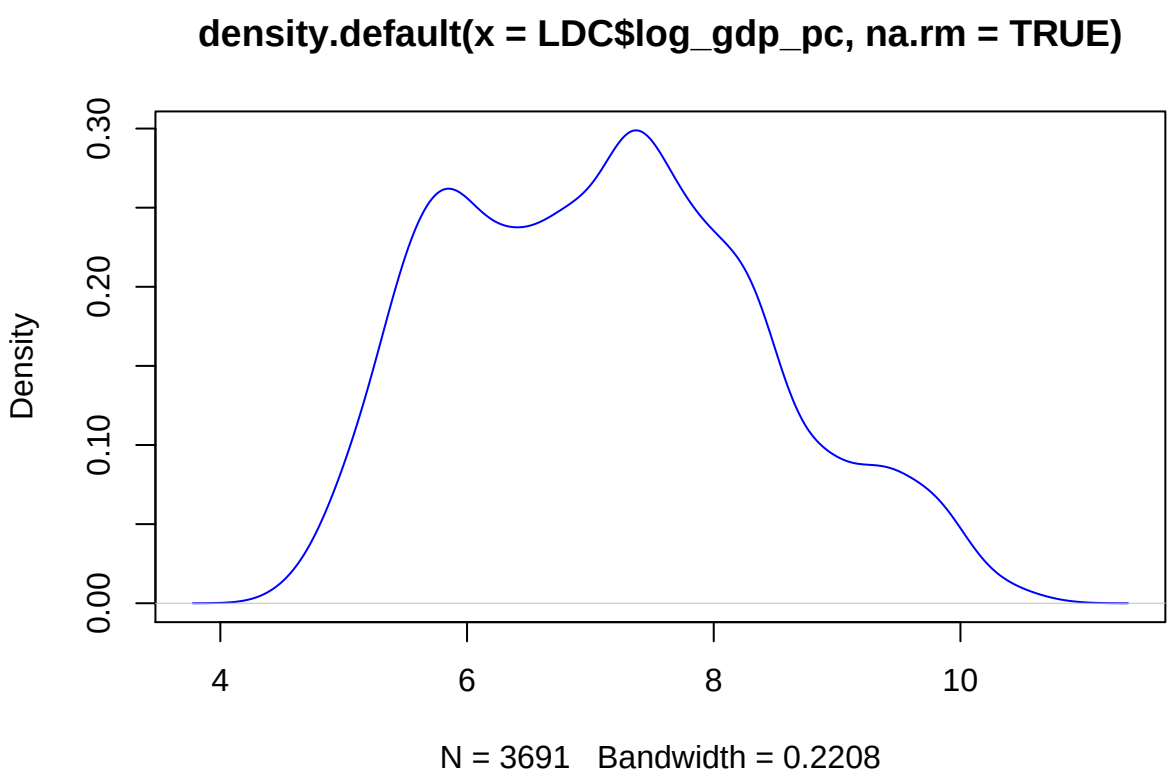

```r
dens = density(LDC$gdp_pc_95d, na.rm = TRUE)
plot(dens, col = "blue")
```




```
LDC$log_gdp_pc = log(LDC$gdp_pc_95d)

hist(LDC$log_gdp_pc, breaks = 21)
```




```
dens = density(LDC$log_gdp_pc, na.rm = TRUE)
plot(dens, col = "blue")
```




```
### We have reduced the skewness and enforced a distribution that is closer to
### a unimodal distribution
```

What do we do if the data is not in the format in which we want it to be?

Let us look at a dataset that deals with lotteries in different states.

```
setwd("C:/Users/Jan/OneDrive/Documents/GitHub/ps630_lab/ps630_f16/W8/")
prize = read.csv("Lottery Prize Amounts Awarded.csv", stringsAsFactors = FALSE)
tickets = read.csv("Lottery Ticket Sales.csv", stringsAsFactors = FALSE)

# Let's look at the datasets

# Let's get rid of missing values

prize = prize[93:103, ]
tickets = tickets[93:103, ]
```

We want to have the data in a format in which we have both the tickets sold and the prizes awarded for each year and state. How do we get there? The "reshape" package is very useful in this respect.

install.packages("reshape")

```
library(reshape)
prize <- melt(prize, id = c("Year"))
prize$Prize = prize$value
```

```
prize$value = NULL

tickets <- melt(tickets, id = c("Year"))
tickets$TSales = tickets$value
tickets$value = NULL
```

Now let's merge the two dataframes.

```
full = merge(prize, tickets, by = c("Year", "variable"))
View(full)
```

More information on the reshape package can be found here: http://had.co.nz/reshape/

3. Data management III: creating new variables

Often we can create new variables from existing ones. Let us create three categories for a country's wealth. (1) Low income countries, (2) middle income countries, and (3) high income countries.

```
LDC$incomelevel = NA
LDC$incomelevel[LDC$gdp_pc_95d <= 10000] = "Low-income country"
unique(LDC$incomelevel)
```

```
## [1] "Low-income country" NA
```

```
LDC$incomelevel[LDC$gdp_pc_95d > 10000 & LDC$gdp_pc_95d <= 20000] = "Middle-income country"
unique(LDC$incomelevel)
```

```
## [1] "Low-income country"    NA                      "Middle-income country"
```

```
LDC$incomelevel[LDC$gdp_pc_95d > 20000] = "High-income country"
unique(LDC$incomelevel)
```

```
## [1] "Low-income country"    NA                      "Middle-income country"
## [4] "High-income country"
```

```
hist(LDC$incomelevel)
```

```
## Error in hist.default(LDC$incomelevel): 'x' must be numeric
```

```
### Does not work because it is not numeric!
```

If we want a histogram of our data, our argument needs to be numeric. We can do a simple data transformation to turn the variable into a numeric variable.

```
LDC$incomelevel[LDC$incomelevel == "Low-income country"] = 0
LDC$incomelevel[LDC$incomelevel == "Middle-income country"] = 1
LDC$incomelevel[LDC$incomelevel == "High-income country"] = 2

hist(LDC$incomelevel)
```

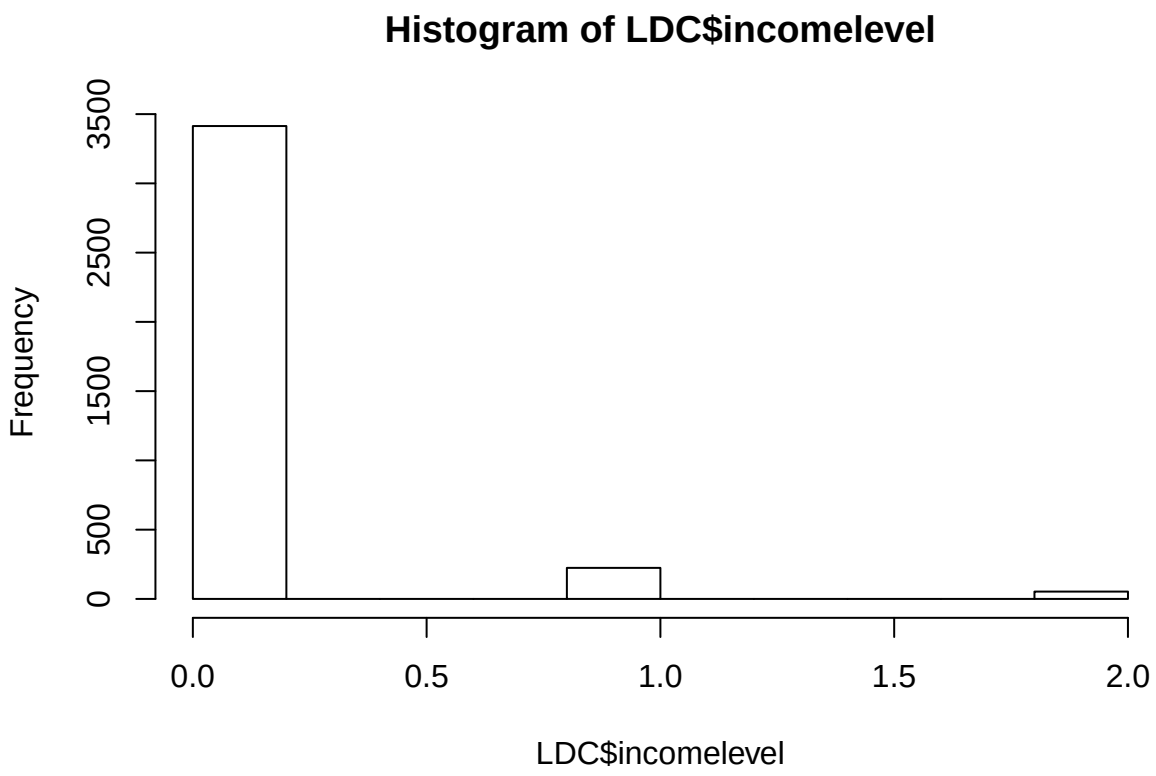
```
## Error in hist.default(LDC$incomelevel): 'x' must be numeric
```

```
### Why does this not work? We just assigned numeric values.
```

We just assigned numeric values to our variable but R still treats this a character variable. What is the reason for this?

In order to change the coding of the variable to numeric, we can use the following command:

```
LDC$incomelevel = as.numeric(LDC$incomelevel)
hist(LDC$incomelevel)
```



As we can see, the vast majority of our sample are low-income countries.

Now let us try to create a new variable that depends on two other variables. We want to have a major economic crises when we have both an economic crisis and a balance-of-payment crisis. How can we do that in R?

```
LDC$major_crisis = NA
LDC$major_crisis[LDC$ecris2 == 1 & LDC$bpcris == 0] <- 0
LDC$major_crisis[LDC$ecris2 == 0 & LDC$bpcris == 1] <- 0
LDC$major_crisis[LDC$ecris2 == 1 & LDC$bpcris == 1] <- 1
```

Let's say you want to create a variable that shows you if the Polity IV Score changed in any given year, with 0 indicating no change and 1 indicating a change. How would we do that?

```
complete = with(LDC, complete.cases(polityiv_update2))
# Necessary because otherwise we will get NAs
LDC = LDC[complete, ]
LDC$politychange = NA
for (i in 2:length(LDC$polityiv_update2)) {
    if (LDC$ctylabel[i] == LDC$ctylabel[i - 1]) {
        if (LDC$polityiv_update2[i] != LDC$polityiv_update2[i - 1]) {
            LDC$politychange[i] = 1
        } else {
            LDC$politychange[i] = 0
        }
    }
}
```

4. Data management IV: useful commands

The following command allows you to aggregate data.

We will aggregate the maximum (FUN = max) of FDI by country (l1fdi ~ ctylabel).

```
FDImax=aggregate(formula = l1fdi ~ ctylabel, data = LDC, FUN = max)
FDImax
```

```
##                    ctylabel       l1fdi
## 1                   Albania   4.7695708
## 2                   Algeria   2.7095816
## 3                    Angola  36.3610497
## 4                 Argentina   3.0506372
## 5                   Armenia  11.5032063
## 6                Azerbaijan  28.9507732
## 7                Bangladesh   0.4171754
## 8                   Belarus   0.8701649
## 9                     Benin   2.0329552
## 10                   Bhutan   0.0000000
## 11                  Bolivia  11.4547396
## 12                 Botswana  14.5541086
## 13                   Brazil   4.2229314
## 14                 Bulgaria   5.2014003
## 15              BurkinaFaso   0.9781891
## 16                  Burundi   1.1385435
## 17                 Cambodia   9.4366856
## 18                 Cameroon   4.0056095
## 19  CentralAfricanRepublic   3.2788446
## 20                     Chad   5.1854916
## 21                    Chile   7.4812307
## 22                    China   6.3860941
## 23                 Colombia   5.4056182
## 24                  Comoros   3.8371851
## 25                    Congo  16.4259396
## 26                CostaRica   4.5611410
## 27            Coted'Ivoire   3.4899659
## 28                  Croatia   2.6347528
```

```
## 29               Djibouti   1.1782608
## 30      DominicanRepublic   4.8502913
## 31                Ecuador  10.3575630
## 32                  Egypt   7.1207776
## 33             ElSalvador   9.2755642
## 34       EquatorialGuinea 184.5647125
## 35                Estonia  11.3487129
## 36               Ethiopia   2.7395952
## 37                   Fiji   7.0112934
## 38                  Gabon   8.3263521
## 39                 Gambia   5.7216630
## 40                Georgia   7.4359035
## 41                  Ghana   4.3720012
## 42               Guatemala   4.3011985
## 43                 Guinea   1.3689225
## 44           GuineaBissau   1.4451288
## 45                 Guyana  56.1083908
## 46                  Haiti   1.2406948
## 47               Honduras   2.6956320
## 48                Hungary  10.5387688
## 49                  India   0.8846607
## 50              Indonesia   2.7992134
## 51                   Iran   0.7215830
## 52                Jamaica  11.6016893
## 53                 Jordan   5.0095434
## 54             Kazakhstan   6.0458264
## 55                  Kenya   1.4010590
## 56                  Korea   1.7321507
## 57         KyrgyzRepublic   6.9666777
## 58                   Laos   8.5714207
## 59                 Latvia   9.1518860
## 60                Lesotho  21.6319885
## 61                Liberia  11.2066574
## 62              Lithuania   8.8257484
## 63             Madagascar   0.9018509
## 64                 Malawi   4.1408086
## 65               Malaysia   9.2539558
## 66                   Mali   4.5890121
## 67             Mauritania  10.5607567
## 68              Mauritius   1.6419731
## 69                 Mexico   3.4910583
## 70                Moldova   4.9373941
## 71               Mongolia   2.7545524
## 72                Morocco   1.9184648
## 73             Mozambique   5.7652102
## 74                  Nepal   0.4616630
## 75              Nicaragua   9.7946072
## 76                  Niger   2.8552208
## 77                Nigeria   9.1927996
## 78                   Oman   6.2304463
## 79               Pakistan   1.4411087
## 80                 Panama  15.2318125
## 81          PapuaNewGuinea  10.2237158
## 82               Paraguay   2.4186306
```

```
## 83                   Peru   7.1794763
## 84            Philippines   3.3471756
## 85                 Poland   4.0776300
## 86                Romania   4.9478350
## 87                 Russia   1.5851402
## 88                 Rwanda   1.4561290
## 89                Senegal   4.0890822
## 90             SierraLeone   4.9091926
## 91                 Somalia   6.6817203
## 92             SouthAfrica   2.6254594
## 93               SriLanka   2.9144170
## 94                  Sudan   3.9002509
## 95              Swaziland  13.2583723
## 96                  Syria   1.6254511
## 97             Tajikistan   1.7030628
## 98               Tanzania   2.3532701
## 99               Thailand   6.7666187
## 100                  Togo  11.5931854
## 101       Trinidad&Tobago  18.2224827
## 102               Tunisia   4.3410387
## 103                Turkey   0.6107548
## 104          Turkmenistan   4.8652730
## 105                Uganda   3.1025271
## 106               Ukraine   1.7602383
## 107               Uruguay   3.0336676
## 108             Venezuela   6.4151616
## 109                 Zaire   1.0400436
## 110                Zambia   6.7483664
## 111              Zimbabwe   7.5118356
```

We can also use other functions, such as “min” or “mean”.

```
FDImin=aggregate(formula = l1fdi ~ ctylabel, data = LDC, FUN = min)
FDImin
```

```
##                  ctylabel         l1fdi
## 1                 Albania   0.000000000
## 2                 Algeria  -2.979254246
## 3                  Angola   1.718133330
## 4               Argentina  -0.017934196
## 5                 Armenia   0.000000000
## 6              Azerbaijan   0.000000000
## 7              Bangladesh  -0.003069585
## 8                 Belarus   0.052252244
## 9                   Benin  -0.412608743
## 10                 Bhutan   0.000000000
## 11                Bolivia  -6.695839405
## 12               Botswana  -9.197874069
## 13                 Brazil   0.124915160
## 14               Bulgaria   0.000000000
## 15             BurkinaFaso  -0.131469905
## 16                Burundi   0.000000000
## 17               Cambodia   1.666512489
## 18               Cameroon  -1.058670759
```

```
## 19  CentralAfricanRepublic  -0.781170487
## 20                    Chad  -0.017000727
## 21                   Chile  -5.115606785
## 22                   China   0.000000000
## 23                Colombia   0.166798070
## 24                 Comoros  -0.518406570
## 25                   Congo   0.000000000
## 26              CostaRica   0.882714272
## 27           Coted'Ivoire  -2.355392218
## 28                 Croatia   0.727679968
## 29                Djibouti   0.000000000
## 30       DominicanRepublic  -0.018158471
## 31                 Ecuador  -0.386101872
## 32                   Egypt   0.000000000
## 33              ElSalvador  -0.049047634
## 34        EquatorialGuinea  -0.288051605
## 35                 Estonia   1.965055346
## 36                Ethiopia  -0.049632955
## 37                    Fiji   0.000000000
## 38                   Gabon  -2.939527273
## 39                  Gambia   0.000000000
## 40                 Georgia   0.000000000
## 41                   Ghana  -0.666676939
## 42               Guatemala   0.289264798
## 43                  Guinea   0.005949384
## 44            GuineaBissau   0.000000000
## 45                  Guyana -21.132360458
## 46                   Haiti  -0.153849840
## 47                Honduras  -0.134078220
## 48                 Hungary   0.000000000
## 49                   India  -0.030314794
## 50               Indonesia  -0.380973965
## 51                    Iran  -0.299713492
## 52                 Jamaica  -1.211688519
## 53                  Jordan  -0.625535965
## 54              Kazakhstan   0.373197615
## 55                   Kenya   0.000000000
## 56                   Korea   0.009868198
## 57          KyrgyzRepublic   0.233004227
## 58                    Laos   0.000000000
## 59                  Latvia   0.455621183
## 60                 Lesotho   0.000000000
## 61                 Liberia  -1.630434752
## 62               Lithuania   0.000000000
## 63              Madagascar  -0.191244096
## 64                  Malawi  -0.121288791
## 65                Malaysia   1.399446726
## 66                    Mali  -0.739642382
## 67              Mauritania -27.235622406
## 68               Mauritius  -0.348367631
## 69                  Mexico   0.235614866
## 70                 Moldova   0.000000000
## 71                Mongolia   1.038361549
## 72                 Morocco  -0.267307460
```

```
## 73              Mozambique  -0.090275630
## 74                   Nepal  -0.040643137
## 75               Nicaragua   0.000000000
## 76                   Niger  -2.173686743
## 77                 Nigeria  -1.211055875
## 78                    Oman  -5.066972256
## 79                Pakistan  -0.056051854
## 80                  Panama  -1.373375416
## 81          PapuaNewGuinea   0.000000000
## 82                Paraguay   0.017125741
## 83                    Peru  -0.983303130
## 84             Philippines  -0.380164444
## 85                  Poland   0.160014689
## 86                 Romania   0.000000000
## 87                  Russia   0.000000000
## 88                  Rwanda  -0.090909094
## 89                 Senegal  -1.459938407
## 90              SierraLeone -16.292877197
## 91                 Somalia  -4.397937298
## 92              SouthAfrica   0.280452609
## 93                 SriLanka  -0.029449204
## 94                   Sudan  -0.048586573
## 95               Swaziland  -2.463035345
## 96                   Syria   0.000000000
## 97              Tajikistan   0.000000000
## 98                Tanzania   0.000000000
## 99                Thailand   0.147821426
## 100                   Togo  -6.965785027
## 101        Trinidad&Tobago  -0.319305092
## 102                Tunisia   0.631348133
## 103                 Turkey   0.018492833
## 104           Turkmenistan   0.000000000
## 105                 Uganda  -0.115410216
## 106                Ukraine   0.000000000
## 107                Uruguay   0.000000000
## 108              Venezuela  -2.359975576
## 109                  Zaire  -1.823154688
## 110                 Zambia -17.049514771
## 111               Zimbabwe  -0.465988696
```

```
FDImean=aggregate(formula = l1fdi ~ ctylabel, data = LDC, FUN = mean)
FDImean
```

```
##                   ctylabel       l1fdi
## 1                  Albania  2.23239709
## 2                  Algeria  0.20353489
## 3                   Angola  8.11528322
## 4                Argentina  0.86324958
## 5                  Armenia  2.83021245
## 6               Azerbaijan 14.43002019
## 7               Bangladesh  0.03238168
## 8                  Belarus  0.33224374
## 9                    Benin  0.49522205
## 10                  Bhutan  0.00000000
```

```
## 11                 Bolivia  1.65282179
## 12                Botswana  2.27415610
## 13                  Brazil  1.00788061
## 14                Bulgaria  0.77953954
## 15              BurkinaFaso  0.26363197
## 16                 Burundi  0.09972449
## 17                Cambodia  4.68731999
## 18                Cameroon  0.73865800
## 19  CentralAfricanRepublic  0.56048093
## 20                    Chad  1.29785930
## 21                   Chile  1.75360696
## 22                   China  2.35426572
## 23                Colombia  1.29944650
## 24                 Comoros  0.46309064
## 25                   Congo  2.67786402
## 26              CostaRica  2.11343176
## 27           Coted'Ivoire  0.98545534
## 28                 Croatia  1.57793431
## 29                Djibouti  0.61929779
## 30       DominicanRepublic  1.86384365
## 31                 Ecuador  2.01944831
## 32                   Egypt  1.73205762
## 33              ElSalvador  0.81510875
## 34        EquatorialGuinea 16.08937686
## 35                 Estonia  5.20413458
## 36                Ethiopia  0.36286171
## 37                    Fiji  2.56215772
## 38                   Gabon  2.22137839
## 39                  Gambia  1.59621079
## 40                 Georgia  2.33365107
## 41                   Ghana  0.85297245
## 42               Guatemala  1.33839263
## 43                  Guinea  0.51626106
## 44            GuineaBissau  0.25497584
## 45                  Guyana  4.31317396
## 46                   Haiti  0.50053612
## 47                Honduras  1.01882552
## 48                 Hungary  1.86709644
## 49                   India  0.14964875
## 50               Indonesia  0.85146475
## 51                    Iran  0.02106030
## 52                 Jamaica  2.33981003
## 53                  Jordan  0.87021897
## 54              Kazakhstan  3.37076003
## 55                   Kenya  0.45298946
## 56                   Korea  0.37839610
## 57          KyrgyzRepublic  3.15429416
## 58                    Laos  2.11829154
## 59                  Latvia  4.45896152
## 60                 Lesotho  2.08957526
## 61                 Liberia  1.89087533
## 62               Lithuania  2.38776219
## 63              Madagascar  0.28297330
## 64                  Malawi  1.00321458
```

```
## 65                Malaysia  4.15192420
## 66                    Mali  0.49155235
## 67              Mauritania  0.19961921
## 68               Mauritius  0.54238837
## 69                  Mexico  1.32955403
## 70                 Moldova  1.62848800
## 71                Mongolia  1.62541984
## 72                 Morocco  0.48580303
## 73              Mozambique  1.07189444
## 74                   Nepal  0.07295209
## 75               Nicaragua  1.73085526
## 76                   Niger  0.63644178
## 77                 Nigeria  2.60831515
## 78                    Oman  1.51846195
## 79                Pakistan  0.43738152
## 80                  Panama  2.28191798
## 81          PapuaNewGuinea  3.09780590
## 82                Paraguay  0.98350743
## 83                    Peru  1.14265594
## 84             Philippines  0.88293511
## 85                  Poland  2.11028122
## 86                 Romania  1.17296024
## 87                  Russia  0.50407697
## 88                  Rwanda  0.63700253
## 89                 Senegal  0.72326921
## 90              SierraLeone  0.14086243
## 91                  Somalia  0.06787166
## 92              SouthAfrica  0.94953346
## 93                 SriLanka  0.70001354
## 94                   Sudan  0.17717631
## 95               Swaziland  4.42905121
## 96                   Syria  0.27601968
## 97              Tajikistan  0.56546844
## 98                Tanzania  0.99261495
## 99                Thailand  1.31779913
## 100                   Togo  1.31465533
## 101        Trinidad&Tobago  5.32565142
## 102                Tunisia  1.86982996
## 103                 Turkey  0.26530363
## 104           Turkmenistan  2.29801540
## 105                 Uganda  0.78800072
## 106                Ukraine  0.66298052
## 107                Uruguay  0.64248856
## 108              Venezuela  0.73593569
## 109                  Zaire  0.06247111
## 110                 Zambia  1.54929367
## 111               Zimbabwe  0.43699174
```

Here are some more useful commands for data management:

```
unique(LDC$ctylabel)
```

```
##   [1] "Turkey"                 "Yugoslavia"
```

```
##   [3] "SouthAfrica"            "Argentina"
##   [5] "Bolivia"                "Brazil"
##   [7] "Chile"                  "Colombia"
##   [9] "CostaRica"              "DominicanRepublic"
##  [11] "Ecuador"                "ElSalvador"
##  [13] "Guatemala"              "Haiti"
##  [15] "Honduras"               "Mexico"
##  [17] "Nicaragua"              "Panama"
##  [19] "Paraguay"               "Peru"
##  [21] "Uruguay"                "Venezuela"
##  [23] "Guyana"                 "Jamaica"
##  [25] "Trinidad&Tobago"        "Bahrain"
##  [27] "Cyprus"                 "Iran"
##  [29] "Iraq"                   "Israel"
##  [31] "Jordan"                 "Kuwait"
##  [33] "Lebanon"                "Oman"
##  [35] "Qatar"                  "SaudiArabia"
##  [37] "Syria"                  "UnitedArabEmirates"
##  [39] "Egypt"                  "Yemen"
##  [41] "RepublicofYemen"        "Afghanistan"
##  [43] "Bangladesh"             "Bhutan"
##  [45] "Myanmar"                "Cambodia"
##  [47] "SriLanka"               "India"
##  [49] "Indonesia"              "Korea"
##  [51] "Laos"                   "Malaysia"
##  [53] "Nepal"                  "Pakistan"
##  [55] "Philippines"            "Singapore"
##  [57] "Thailand"               "Vietnam"
##  [59] "Djibouti"               "Algeria"
##  [61] "Angola"                 "Botswana"
##  [63] "Burundi"                "Cameroon"
##  [65] "CentralAfricanRepublic" "Chad"
##  [67] "Comoros"                "Congo"
##  [69] "Zaire"                  "Benin"
##  [71] "EquatorialGuinea"       "Ethiopia"
##  [73] "Gabon"                  "Gambia"
##  [75] "Ghana"                  "GuineaBissau"
##  [77] "Guinea"                 "Coted'Ivoire"
##  [79] "Kenya"                  "Lesotho"
##  [81] "Liberia"                "Libya"
##  [83] "Madagascar"             "Malawi"
##  [85] "Mali"                   "Mauritania"
##  [87] "Mauritius"              "Morocco"
##  [89] "Mozambique"             "Niger"
##  [91] "Nigeria"                "Zimbabwe"
##  [93] "Rwanda"                 "Senegal"
##  [95] "SierraLeone"            "Somalia"
##  [97] "Namibia"                "Sudan"
##  [99] "Swaziland"              "Tanzania"
## [101] "Togo"                   "Tunisia"
## [103] "Uganda"                 "BurkinaFaso"
## [105] "Zambia"                 "Fiji"
## [107] "PapuaNewGuinea"         "Armenia"
## [109] "Azerbaijan"             "Belarus"
```

```
## [111] "Albania"                "Georgia"
## [113] "Kazakhstan"             "KyrgyzRepublic"
## [115] "Bulgaria"               "Moldova"
## [117] "Russia"                 "Tajikistan"
## [119] "China"                  "Turkmenistan"
## [121] "Ukraine"                "Uzbekistan"
## [123] "Cuba"                   "Czechoslovakia"
## [125] "Estonia"                "Latvia"
## [127] "Hungary"                "Lithuania"
## [129] "Mongolia"               "Croatia"
## [131] "Slovenia"               "Poland"
## [133] "Romania"
```

```
# Displays all of the empirically observed unique values Is this useful for
# continuous variables?

head(LDC$polityiv_update2)
```

```
## [1] 18  8  8 19 19 19
```

```
# Returns the first five values of a vector Also, tail() returns the last
# five values

names(LDC)
```

```
##  [1] "country"          "ctylabel"         "date"
##  [4] "gatt_wto_new"     "aclpn"            "bpc1"
##  [7] "dopen_wacz2"      "ecris2"           "fdignp"
## [10] "gdp_pc_95d"       "l1aclpn"          "l1bpc1"
## [13] "l1ecris2"         "newtar"           "polityiv_update2"
## [16] "signed"           "yrsoffic"         "usheg"
## [19] "l1usheg"          "l1fiveop"         "l1gdp_pc"
## [22] "avsw"             "avnewtar"         "l1avsw"
## [25] "l1avnewtar"       "lnpop"            "l1lnpop"
## [28] "l1office"         "l1partyage2000"   "l1fdi"
## [31] "l1polity"         "l2polity"         "l3polity"
## [34] "l1signed"         "milit2"           "sp2"
## [37] "pers2"            "l1milit2"         "l1sp2"
## [40] "dictator1"        "l1dictator1"      "yr70"
## [43] "yr80"             "l1ssch"           "closedyr"
## [46] "_spline1"         "_spline2"         "_spline3"
## [49] "l1gatt_wto_new"   "polityiv_squared" "l1polity_squared"
## [52] "log_gdp_pc"       "incomelevel"      "major_crisis"
## [55] "politychange"
```

```
# Returns the variable names of a data frame colnames() does the same

class(LDC$ctylabel)
```

```
## [1] "character"
```

```r
# Show the classification of a variable

char_newtar = as.character(LDC$newtar)
# Changes a variable to a character variable

val_newtar = as.numeric(LDC$char_newtar)
# Changes a variable to a numeric variable

quantile(LDC$newtar, p = c(0.1, 0.9), na.rm = T)
```

```
##   10%   90%
##  8.38 40.06
```

```r
# Displays the 10th and 90th percentile of a variable
```

5. Measurement error

Let us discuss measurement error and how it can influence our research results.

In order to do this we will use a dataset that contains teaching ratings of professors.

Description: "TeachingRatings contains data on course evaluations, course characteristics, and professor characteristics for 463 courses for the academic years 2000-2002 at the University of Texas at Austin."

Let us look at some of the variables and think about what kind of measurement error we could observe here.

```r
library(foreign)
tr = read.dta("TeachingRatings.dta")
summary(tr)
```

```
##     minority           age            female         onecredit
##  Min.   :0.0000   Min.   :29.00   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.0000   1st Qu.:42.00   1st Qu.:0.0000   1st Qu.:0.00000
##  Median :0.0000   Median :48.00   Median :0.0000   Median :0.00000
##  Mean   :0.1382   Mean   :48.37   Mean   :0.4212   Mean   :0.05832
##  3rd Qu.:0.0000   3rd Qu.:57.00   3rd Qu.:1.0000   3rd Qu.:0.00000
##  Max.   :1.0000   Max.   :73.00   Max.   :1.0000   Max.   :1.00000
##      beauty          course_eval        intro          nnenglish
##  Min.   :-1.45049   Min.   :2.100   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:-0.65627   1st Qu.:3.600   1st Qu.:0.0000   1st Qu.:0.00000
##  Median :-0.06801   Median :4.000   Median :0.0000   Median :0.00000
##  Mean   : 0.00000   Mean   :3.998   Mean   :0.3391   Mean   :0.06048
##  3rd Qu.: 0.54560   3rd Qu.:4.400   3rd Qu.:1.0000   3rd Qu.:0.00000
##  Max.   : 1.97002   Max.   :5.000   Max.   :1.0000   Max.   :1.00000
```

What problems related to measurement error can you think of?

What are other examples of measurement error that you can think of?