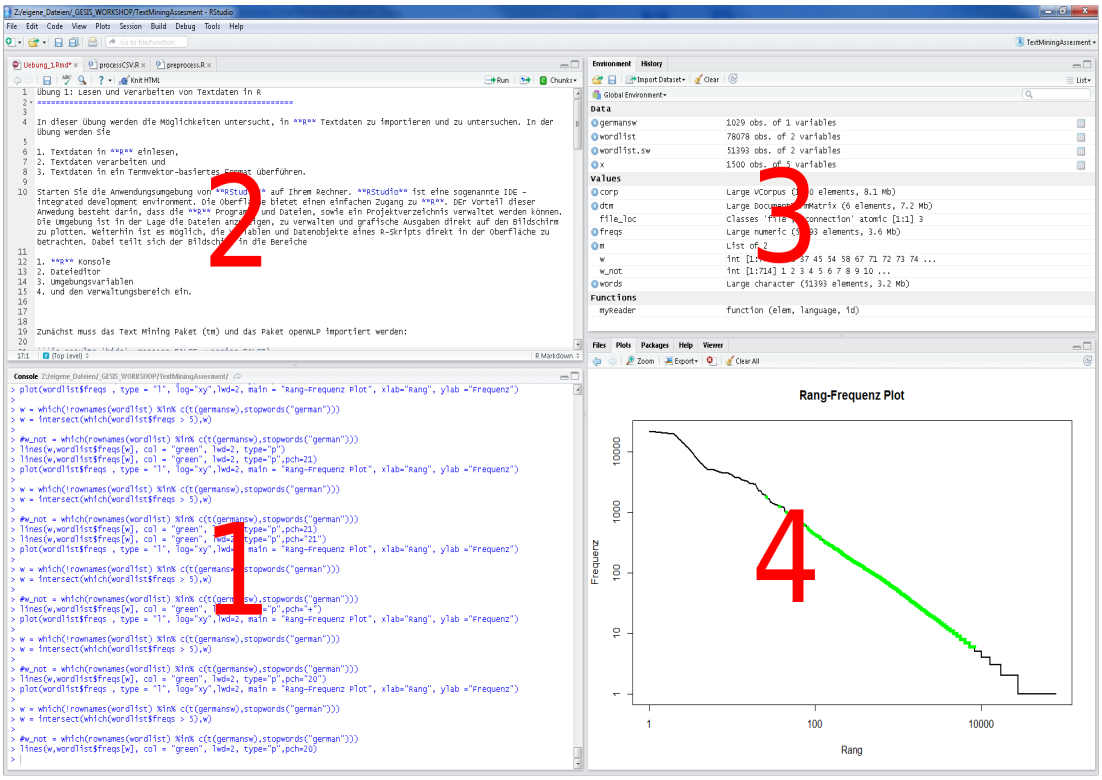
Intro RStudio & R

8 10 2020

Basiert auf Vorarbeiten von Andreas Niekler und Gregor Wiedemann. Wiedemann, Gregor; Niekler, Andreas (2017): Hands-on: A five day text mining course for humanists and social scientists in R. Proceedings of the 1st Workshop on Teaching NLP for Digital Humanities (Teach4DH@GSCL 2017), Berlin.

Start **RStudio** on your computer. **RStudio** is a so-called IDE - Integrated Development Environment. The interface provides easy access to **R**. The advantage of this application is that **R** programs and files as well as a project directory can be managed easily. The environment is capable of editing and running program code, viewing outputs and rendering graphics. Furthermore, it is possible to view variables and data objects of an R-script directly in the interface. The screen divides into the areas

1. **R** console
2. File editor
3. Environment variables
4. Management panes (File browser, plots, help display and R packages).



Basic Operations

```
1 + 2      # addition
```

```
## [1] 3
```

```
sqrt(9)    # square root function
```

```
## [1] 3
```

```
x <- 1     # assignment
1:10       # sequence
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
# install.packages("quanteda")
library("quanteda")
require("quanteda")

# help
help(require)
?require
apropos("nova")
```

```
## [1] "anova"             "manova"            "power.anova.test"
## [4] "stat.anova"        "summary.manova"
```

```
# set your working directory
setwd("~")
```

Data types

```
x <- 10.5
typeof(x)
```

```
## [1] "double"
```

```
class(x)
```

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

```
as.integer(x)
```

```
## [1] 10
```

```
is.integer(x)
```

```
## [1] FALSE
```

```
is.integer(as.integer(x))
```

```
## [1] TRUE
```

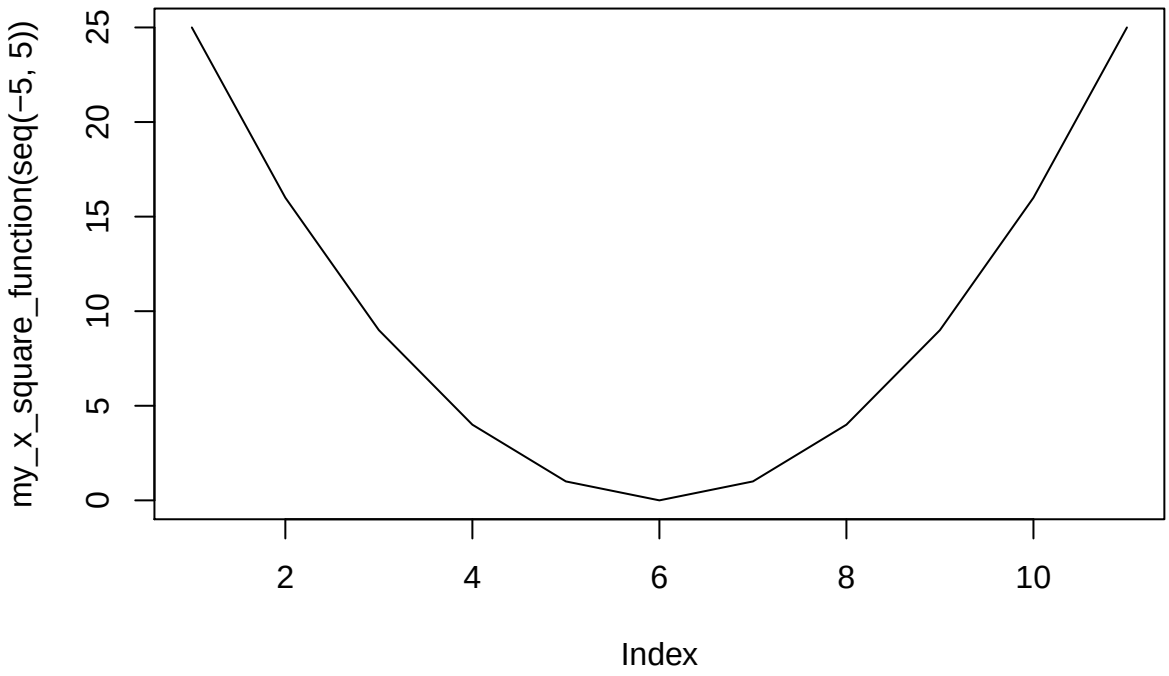
```
x <- "3.14"
typeof(x)
```

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

```r
x <- as.double(x)
1:3 == c(1, 2, 3)
```

```
## [1] TRUE TRUE TRUE
```

```r
# functions
my_x_square_function <- function(x) {
  result <- x ^ 2
  return(result)
}
plot(my_x_square_function(seq(-5,5)), type="l")
```



Vector and matrix

Vectors contain several values of the same type. They are formed with the operator `c(...)`:

```r
v <- c(3.14, 2.71, 8.11)
v
```

```
## [1] 3.14 2.71 8.11
```

```r
typeof(v)
```

```
## [1] "double"
```

```r
length(v)
```

```
## [1] 3
```

```
sum(v)
```

```
## [1] 13.96
```

Frequently used types of vectors can be created with functions: e.g. `seq(start, end, step)` creates a sequence:

```
seq(1, 10, 3)
```

```
## [1]  1  4  7 10
```

`rep` repeats a value or a vector:

```
rep(1, 10)
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1
```

```
rep(1:3, 3)
```

```
## [1] 1 2 3 1 2 3 1 2 3
```

Logical operations produce a vector of logical values:

```
v < 5
```

```
## [1]  TRUE  TRUE FALSE
```

That can also be used for indexing:

```
v[v<5]
```

```
## [1] 3.14 2.71
```

Matrices can be generated from vectors in rows (`rbind`) or columns (`cbind`):

```
rbind(1:3, 4:6)
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
```

```
cbind(1:3, 4:6)
```

```
##      [,1] [,2]
## [1,]    1    4
## [2,]    2    5
## [3,]    3    6
```

`ncol` and `nrow` give the dimensions of matrices:

```
m <- cbind(1:3, 4:6)
nrow(m)
```

```
## [1] 3
```

```
ncol(m)
```

```
## [1] 2
```

The elements of vectors can have *names* besides values:

```
v <- 1:5
names(v) <- c("eins", "zwei", "drei", "vier", "fünf")
v
```

```
## eins zwei drei vier fünf
##    1    2    3    4    5
```

The function `sort` sorts a vector, while `order` returns the indices in the order of the sorted values:

```
v <- c(2,3,1)
sort(v)
```

```
## [1] 1 2 3
```

```
order(v)
```

```
## [1] 3 1 2
```

```
v[order(v)]
```

```
## [1] 1 2 3
```

To apply any function element by element to a whole vector, we use the function `sapply`. Since most algebraic functions can be applied directly to vectors, `sapply` makes most sense if the result of the operation is not a scalar value:

```
v <- c(3, 7, 2)
sapply(v, function(x) c(rep(1, x), rep(0, max(v)-x)))
```

```
##      [,1] [,2] [,3]
## [1,]    1    1    1
## [2,]    1    1    1
## [3,]    1    1    0
## [4,]    0    1    0
## [5,]    0    1    0
## [6,]    0    1    0
## [7,]    0    1    0
```

Lists and tables

Lists contain named elements (scalars, vectors or matrices) of possibly different types and lengths:

```
x <- list(apples = 1:3, pears = (c(3, 7, 2, 6, 3) > 5), plums = 17)
x
```

```
## $apples
## [1] 1 2 3
##
## $pears
## [1] FALSE  TRUE FALSE  TRUE FALSE
##
## $plums
## [1] 17
```

With the operator `$` we access the individual elements of a list:

```
x$apples
```

```
## [1] 1 2 3
```

```
x$pears
```

```
## [1] FALSE  TRUE FALSE  TRUE FALSE
```

`lapply` applies a function to the list and returns a list of results:

```
lapply(x, typeof)
```

```
## $apples
## [1] "integer"
##
## $pears
## [1] "logical"
##
## $plums
## [1] "double"
```

Tables ('data.frame') are lists of vectors of equal length (but possibly of different types):

```
v <- c(3, 8, 2)
df <- data.frame(word=c("der", "die", "das"), freq=v, rel_freq=v/sum(v))
df
```

```
##   word freq  rel_freq
## 1  der    3 0.2307692
## 2  die    8 0.6153846
## 3  das    2 0.1538462
```

```
df$rel_freq
```

```
## [1] 0.2307692 0.6153846 0.1538462
```

```
options(stringsAsFactors = F)

myvector <- c(1, 2, 3)
names(myvector) <- c("one", "two", "three")
print(myvector)
```

```
##   one   two three
##     1     2     3
```

```
print(myvector[1:2])
```

```
## one two
##   1   2
```

```
print(myvector[-1])
```

```
##   two three
##     2     3
```

```
sum(myvector)
```

```
## [1] 6
```

```
mean(myvector)
```

```
## [1] 2
```

```
mymatrix <- matrix(0, nrow=3, ncol=4)
rownames(mymatrix) <- c("one", "two", "three")
colnames(mymatrix) <- c("house", "sun", "tree", ".")
mymatrix[, 1] <- 12
mymatrix[, "sun"] <- 4
mymatrix[3, 4] <- 5
```

```r
mymatrix[2, 3:4] <- 9
colSums(mymatrix)
```

```
## house   sun  tree     .
##    36    12     9    14
```

```r
rowSums(mymatrix)
```

```
##   one   two three
##    16    34    21
```

```r
colMeans(mymatrix)
```

```
##     house       sun      tree         .
## 12.000000  4.000000  3.000000  4.666667
```

```r
mydatdaframe <- data.frame(v = c(1, 2, 3), c = as.character(myvector), n = c("one", "two", "three"))
mydatdaframe$v
```

```
## [1] 1 2 3
```

```r
mydatdaframe$c
```

```
## [1] "1" "2" "3"
```

```r
mydatdaframe[, "c"]
```

```
## [1] "1" "2" "3"
```

```r
mydatdaframe[1, ]
```

```
##   v c   n
## 1 1 1 one
```

```r
#rowSums(mydatdaframe)
```

More basic functions

```r
sort(mydatdaframe$n)
```

```
## [1] "one"   "three" "two"
```

```r
sort(mydatdaframe$n, decreasing = T)
```

```
## [1] "two"   "three" "one"
```

```r
c(myvector, myvector)
```

```
##   one   two three   one   two three
##     1     2     3     1     2     3
```

```r
is.vector(myvector)
```

```
## [1] TRUE
```

```r
is.matrix(mymatrix)
```

```
## [1] TRUE
```

```r
is.matrix(myvector)
```

```
## [1] FALSE
```

```r
mymatrix / 2
```

```
##       house sun tree   .
## one       6   2  0.0 0.0
## two       6   2  4.5 4.5
## three     6   2  0.0 2.5
```

```r
mymatrix / myvector
```

```
##       house      sun tree        .
## one      12 4.000000  0.0 0.000000
## two       6 2.000000  4.5 4.500000
## three     4 1.333333  0.0 1.666667
```

```r
t(mymatrix)
```

```
##       one two three
## house  12  12    12
## sun     4   4     4
## tree    0   9     0
## .       0   9     5
```

```r
mymatrix %*% t(mymatrix)
```

```
##       one two three
## one   160 160   160
## two   160 322   205
## three 160 205   185
```

```r
cbind(mymatrix, myvector)
```

```
##       house sun tree . myvector
## one      12   4    0 0        1
## two      12   4    9 9        2
## three    12   4    0 5        3
```

```r
# example datasets: data.frame of arrests per USA states
data(USArrests)
View(USArrests)
dim(USArrests)
```

```
## [1] 50  4
```

```r
nrow(USArrests)
```

```
## [1] 50
```

```r
ncol(USArrests)
```

```
## [1] 4
```

```r
length(USArrests)
```

```
## [1] 4
```

```r
max(USArrests)
```

```
## [1] 337
```

```r
which.max(USArrests[, "Murder"])
```

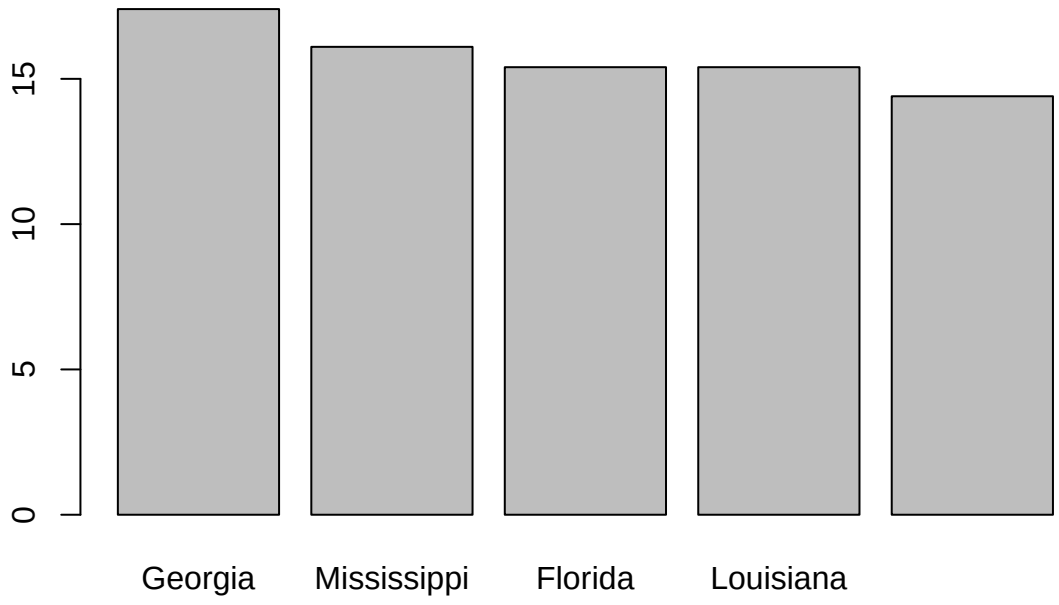
```
## [1] 10
```

```r
o <- order(USArrests[, "Murder"], decreasing = T)
USArrests[o, ]
```

```
##                Murder Assault UrbanPop Rape
## Georgia          17.4     211       60 25.8
## Mississippi      16.1     259       44 17.1
## Florida          15.4     335       80 31.9
## Louisiana        15.4     249       66 22.2
## South Carolina   14.4     279       48 22.5
## Alabama          13.2     236       58 21.2
## Tennessee        13.2     188       59 26.9
## North Carolina   13.0     337       45 16.1
## Texas            12.7     201       80 25.5
## Nevada           12.2     252       81 46.0
## Michigan         12.1     255       74 35.1
## New Mexico       11.4     285       70 32.1
## Maryland         11.3     300       67 27.8
## New York         11.1     254       86 26.1
## Illinois         10.4     249       83 24.0
## Alaska           10.0     263       48 44.5
## Kentucky          9.7     109       52 16.3
## California        9.0     276       91 40.6
## Missouri          9.0     178       70 28.2
## Arkansas          8.8     190       50 19.5
## Virginia          8.5     156       63 20.7
## Arizona           8.1     294       80 31.0
## Colorado          7.9     204       78 38.7
## New Jersey        7.4     159       89 18.8
## Ohio              7.3     120       75 21.4
## Indiana           7.2     113       65 21.0
## Wyoming           6.8     161       60 15.6
## Oklahoma          6.6     151       68 20.0
## Pennsylvania      6.3     106       72 14.9
## Kansas            6.0     115       66 18.0
## Montana           6.0     109       53 16.4
## Delaware          5.9     238       72 15.8
## West Virginia     5.7      81       39  9.3
## Hawaii            5.3      46       83 20.2
## Oregon            4.9     159       67 29.3
## Massachusetts     4.4     149       85 16.3
## Nebraska          4.3     102       62 16.5
## Washington        4.0     145       73 26.2
## South Dakota      3.8      86       45 12.8
## Rhode Island      3.4     174       87  8.3
## Connecticut       3.3     110       77 11.1
## Utah              3.2     120       80 22.9
## Minnesota         2.7      72       66 14.9
## Idaho             2.6     120       54 14.2
## Wisconsin         2.6      53       66 10.8
## Iowa              2.2      56       57 11.3
```

```
## Vermont           2.2      48       32 11.2
## Maine             2.1      83       51  7.8
## New Hampshire     2.1      57       56  9.5
## North Dakota      0.8      45       44  7.3
```

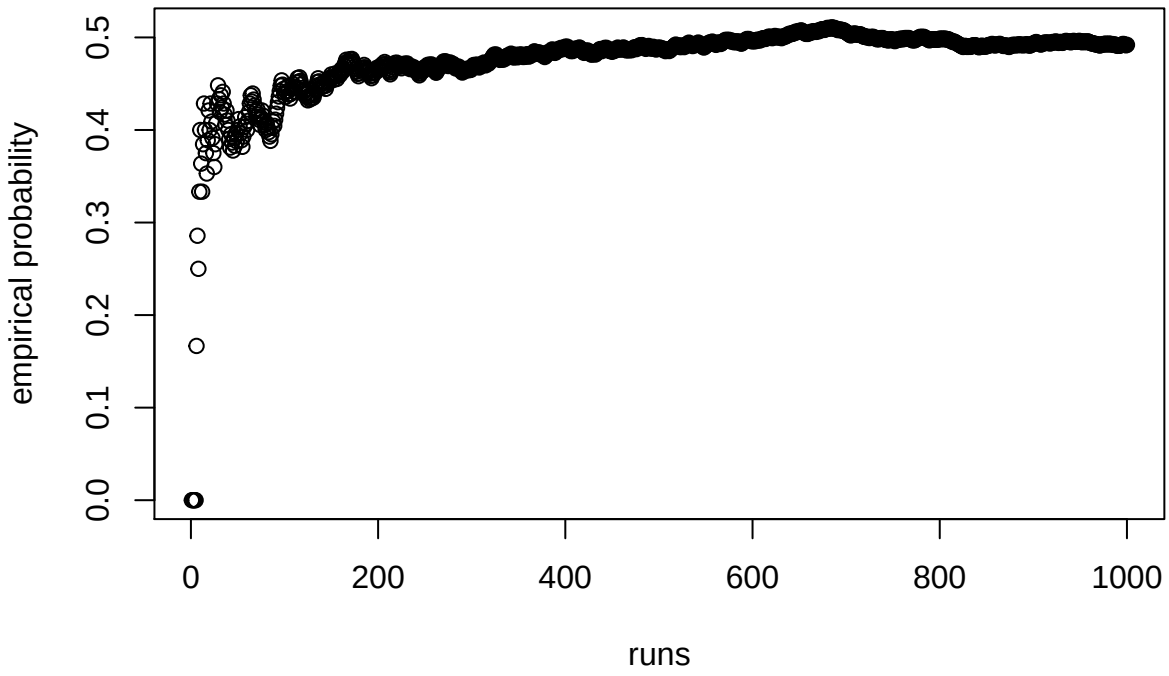
```r
murderRates <- USArrests[o[1:5], "Murder"]
names(murderRates) <- rownames(USArrests)[o[1:5]]
barplot(murderRates)
```



if else statements & for-loops

```r
# check R randomizer
number_of_runs<-1000
over_5<-NULL
below_or_equal_5<-NULL
empirical_probability<-NULL

for (i in 1:number_of_runs){
  random_number <- sample(x = 1:10, size = 1)
  if(random_number >5){
    over_5<-c(over_5,random_number)
  }
  else{
    below_or_equal_5<-c(below_or_equal_5,random_number)
  }
  empirical_probability<-c(empirical_probability,(length(over_5)/i))
}
```

```r
plot(x=1:number_of_runs,y=empirical_probability,xlab = "runs",ylab = "empirical probability")
```



Survey of electors per inhabitant in US federal states

```r
# check available meta data sources here:
#   http://search.r-project.org/library/Lock5Data/html/USStates.html
# library provides metadata number for every state
library(Lock5Data)

states <- Lock5Data::USStates

head(states)
```

```
##        State HouseholdIncome Region Population EighthGradeMath HighSchool
## 1    Alabama          43.253      S      4.849           269.2       84.9
## 2     Alaska          70.760      W      0.737           281.6       92.8
## 3    Arizona          49.774      W      6.731           279.7       85.6
## 4   Arkansas          40.768      S      2.966           277.9       87.1
## 5 California          61.094      W     38.803           275.9       84.1
## 6   Colorado          58.433      W      5.356           289.7       89.5
##   College    IQ    GSP Vegetables Fruit Smokers PhysicalActivity Obese
## 1    24.9  95.7 32.615       74.2  54.1    21.5             45.4  32.4
## 2    24.7  99.0 61.156       80.8  60.3    22.6             55.3  28.4
```

```
## 3    25.5  97.4 35.195       76.2  60.5    16.3            51.9  26.8
## 4    22.4  97.5 31.837       72.0  49.5    25.9            41.2  34.6
## 5    31.4  95.5 46.029       82.7  69.6    12.5            56.3  24.1
## 6    37.0 101.6 46.242       80.9  64.3    17.7            60.4  21.3
##   NonWhite HeavyDrinkers Electoral ObamaVote ObamaRomney TwoParents
## 1     30.7           4.3         9     0.384           R       58.7
## 2     33.1           8.2         3     0.408           R       69.6
## 3     20.8           6.3        11     0.446           R       62.7
## 4     21.7           5.0         6     0.369           R       62.0
## 5     37.7           6.4        55     0.602           O       65.3
## 6     15.8           6.7         9     0.515           O       69.9
##   StudentSpending Insured
## 1           8.755    78.8
## 2          18.175    79.8
## 3           7.208    74.7
## 4           9.394    71.7
## 5           9.220    79.7
## 6           8.647    80.0
```

```r
# population
population <- states$Population

# presidential electors per state
electors<-states$Electoral

electors_per_inhabitant<-electors/population

# bind to original dataframe
states <- cbind(states,electors_per_inhabitant)

# sort by number by number of electors per inhabitant

states<-states[order(states$electors_per_inhabitant, decreasing = T),]

states[,c("State","Population","Electoral","electors_per_inhabitant")]
```

```
##             State Population Electoral electors_per_inhabitant
## 50        Wyoming      0.584         3                5.136986
## 45        Vermont      0.627         3                4.784689
## 2          Alaska      0.737         3                4.070556
## 34   North Dakota      0.739         3                4.059540
## 39   Rhode Island      1.055         4                3.791469
## 41   South Dakota      0.853         3                3.516999
## 8        Delaware      0.936         3                3.205128
## 29  New Hampshire      1.327         4                3.014318
## 19          Maine      1.330         4                3.007519
## 26        Montana      1.024         3                2.929688
## 11         Hawaii      1.420         4                2.816901
## 48  West Virginia      1.850         5                2.702703
## 27       Nebraska      1.882         5                2.656748
## 12          Idaho      1.634         4                2.447980
## 31     New Mexico      2.086         5                2.396932
## 28         Nevada      2.839         6                2.113420
## 16         Kansas      2.904         6                2.066116
## 44           Utah      2.943         6                2.038736
```

```
## 4        Arkansas      2.966         6               2.022927
## 24    Mississippi      2.994         6               2.004008
## 7     Connecticut      3.597         7               1.946066
## 15           Iowa      3.107         6               1.931123
## 40 South Carolina      4.832         9               1.862583
## 1         Alabama      4.849         9               1.856053
## 23      Minnesota      5.457        10               1.832509
## 17       Kentucky      4.413         8               1.812826
## 36       Oklahoma      3.878         7               1.805054
## 37         Oregon      3.970         7               1.763224
## 49      Wisconsin      5.758        10               1.736714
## 18      Louisiana      4.650         8               1.720430
## 47     Washington      7.062        12               1.699235
## 6         Colorado      5.356         9               1.680358
## 42      Tennessee      6.549        11               1.679646
## 20       Maryland      5.976        10               1.673360
## 14        Indiana      6.597        11               1.667425
## 25       Missouri      6.064        10               1.649077
## 3         Arizona      6.731        11               1.634230
## 21  Massachusetts      6.745        11               1.630838
## 22       Michigan      9.910        16               1.614531
## 10        Georgia     10.097        16               1.584629
## 30     New Jersey      8.938        14               1.566346
## 38   Pennsylvania     12.787        20               1.564089
## 46       Virginia      8.326        13               1.561374
## 13       Illinois     12.881        20               1.552674
## 35           Ohio     11.594        18               1.552527
## 33 North Carolina      9.944        15               1.508447
## 32       New York     19.746        29               1.468652
## 9         Florida     19.893        29               1.457799
## 5      California     38.803        55               1.417416
## 43          Texas     26.957        38               1.409652
```

```
# what would the outcome have been if each state could send out electors as closely as possible related

# What was the outcome?
obama_electors <- sum(states[which(states$ObamaRomney=="O"),"Electoral"])
obama_electors
```

```
## [1] 329
```

```
romney_electors <- sum(states[which(states$ObamaRomney=="R"),"Electoral"])
romney_electors
```

```
## [1] 206
```

```
electors_overall <- obama_electors+romney_electors
population_overall <- sum(as.numeric(states$Population))


opt_number_of_electors_per_inhabitant <- electors_overall/population_overall

# calculate adjusted electoral values
adjusted_electoral <- states$Population*opt_number_of_electors_per_inhabitant
adjusted_electoral
```

```
##  [1]  0.9819074  1.0542054  1.2391537  1.2425164  1.7738225  1.4341901
##  [7]  1.5737421  2.2311493  2.2361933  1.7217007  2.3875147  3.1104944
## [13]  3.1642976  2.7473232  3.5072927  4.7733480  4.8826356  4.9482082
## [19]  4.9868792  5.0339570  6.0478100  5.2239493  8.1242752  8.1528581
## [25]  9.1751179  7.4197903  6.5202689  6.6749529  9.6812038  7.8182698
## [31] 11.8736820  9.0053017 11.0111503 10.0477377 11.0918550 10.1956964
## [37] 11.3171557 11.3406946 16.6621621 16.9765743 15.0278915 21.4994013
## [43] 13.9989063 21.6574481 19.4935527 16.7193280 33.1999045 33.4470627
## [49] 65.2413599 45.3241074
```

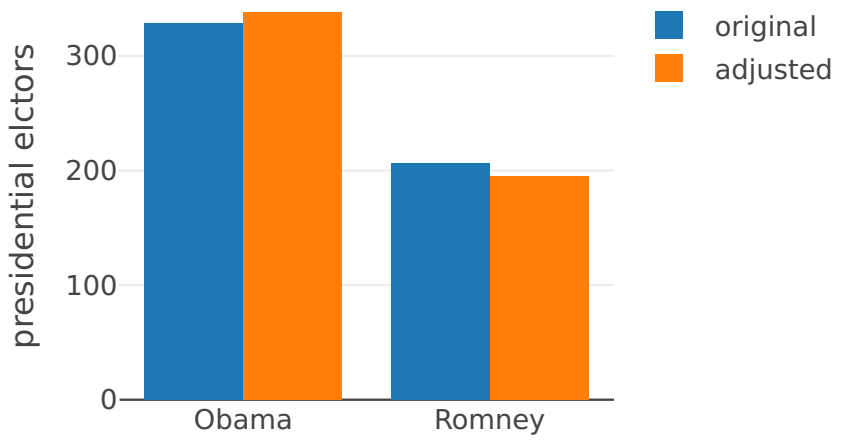
```
adjusted_electoral <- round(adjusted_electoral)

# add to original dataframe
states<-cbind(states,adjusted_electoral)

# check adjusted outcome
adjusted_obama_electors <- sum(states[which(states$ObamaRomney=="O"),"adjusted_electoral"])
adjusted_obama_electors
```

```
## [1] 338
```

```
adjusted_romney_electors <- sum(states[which(states$ObamaRomney=="R"),"adjusted_electoral"])
adjusted_romney_electors
```

```
## [1] 195
```

```
p<-plotly::plot_ly(x=c("Obama","Romney"), y=c(obama_electors,romney_electors),type="bar",name="original
p<-plotly::add_trace(p, y=c(adjusted_obama_electors,adjusted_romney_electors), name= "adjusted")
p<-plotly::layout(p, yaxis=list(title="presidential elctors"),barmode="group")
p
```



correlation betwwen metadata and obama votes?

```
cor(states$ObamaVote,states$HouseholdIncome)
```

```
## [1] 0.514619
```

```
cor(states$ObamaVote,states$IQ)
```

```
## [1] 0.073712
```

```
cor(states$ObamaVote,states$Vegetables)
```

```
## [1] 0.2839672
```