데이터 분석(Python / R)

R로 하는 데이터 분석

삼성전자공과대학교 3학년 3학기

R Programming

❑ Data Structure

	Homogeneous	Hetrogeneous
1d	Atomic Vector	List
2d	Matrix	Data frame
nd	Array	

❑ Vector

intV = c(1,2,3);intV

[1] 1 2 3

charV = c(1, "a", 3); charV

[1] "1" "a" "3" > doubleV = c(1, 2, 3.0);

doubleV = c(1, 2, 3.5); doubleV

[1] 1.0 2.0 3.5

booleanV = c(T, F, TRUE); booleanV

[1] TRUE FALSE TRUE

as.numeric(booleanV)

[1] 1 0 1

attr(booleanV, "desc") = "This is boolean Vector"

booleanV

[1] TRUE FALSE TRUE

attr(,"desc")

[1] "This is boolean Vector"

str(booleanV)

atomic [1:3] TRUE FALSE TRUE

- attr(*, "desc")= chr "This is boolean Vector"

❑ List

```
x = list(1:3, "a", c(TRUE, FALSE, TRUE), c(2.3, 5.9));x
[[1]]
[1] 1 2 3
[[2]]
[1] "a"
[[3]]
[1]  TRUE FALSE  TRUE
[[4]]
[1] 2.3 5.9
str(x)
List of 4
 $ : int [1:3] 1 2 3
 $ : chr "a"
 $ : logi [1:3] TRUE FALSE TRUE
 $ : num [1:2] 2.3 5.9
```

❑ Factor

```
x = factor(c("a", "b", "b", "a"));x
[1] a b b a
Levels: a b
class(x)
[1] "factor"
levels(x)
[1] "a" "b“
sex_char = c("m", "m", "m")
sex_factor = factor(sex_char, levels=c("m","f"))
table(sex_char)
sex_char
m
3
table(sex_factor)
sex_factor
m f
3 0
```

❑ Matrix & Array

```
mat = matrix(1:6, ncol = 3, nrow = 2);mat
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
arr = array(1:12, c(2, 3, 2));arr
, , 1
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
, , 2
     [,1] [,2] [,3]
[1,]    7    9   11
[2,]    8   10   12
```

```
length(mat);length(arr)
[1] 6
[1] 12
nrow(mat);nrow(arr)
[1] 2
[1] 2
ncol(mat);ncol(arr)
[1] 3
[1] 3
```

```
rownames(mat) = c("A", "B"); colnames(mat) = c("a", "b", "c"); mat
  a b c
A 1 3 5
B 2 4 6
dimnames(arr) = list(c("A", "B"), c("a", "b", "c"), c("one", "two")); arr
, , one

  a b c
A 1 3 5
B 2 4 6

, , two

  a  b  c
A 7  9 11
B 8 10 12
```

Data Frame

```
df = data.frame(x=1:3, y=c("a", "b", "c"), stringsAsFactors=FALSE);df
  x y
1 1 a
2 2 b
3 3 c
str(df)
'data.frame':  3 obs. of  2 variables:
 $ x: int  1 2 3
 $ y: chr  "a" "b" "c"
class(df)
[1] "data.frame"
is.data.frame(df)
[1] TRUE
```

```
cbind(df, data.frame(z=3:1))
  x y z
1 1 a 3
2 2 b 2
3 3 c 1
rbind(df, data.frame(x=10,y="z"))
   x y
1  1 a
2  2 b
3  3 c
4 10 z
```

```
data.frame(x = 1:3, y = list(1:2, 1:3, 1:4))
Error in data.frame(1:2, 1:3, 1:4, check.names = FALSE,
stringsAsFactors = TRUE) :
  arguments imply differing number of rows: 2, 3, 4
df = data.frame(x = 1:3)
df$y = list(1:2, 1:3, 1:4)
df
  x          y
1 1       1, 2
2 2    1, 2, 3
3 3 1, 2, 3, 4
df = data.frame(x = 1:3, y = I(list(1:2, 1:3, 1:4)))
df
  x          y
1 1       1, 2
2 2    1, 2, 3
3 3 1, 2, 3, 4
```

❑ Subset

```
a = matrix(1:9, nrow = 3); colnames(a) = c("A", "B", "C");a
     A B C
[1,] 1 4 7
[2,] 2 5 8
[3,] 3 6 9
a[1:2,]
     A B C
[1,] 1 4 7
[2,] 2 5 8
a[c(T, F, T), c("B", "A")]
     B A
[1,] 4 1
[2,] 6 3
a[0, -2]
     A C
```

```
a = outer(1:5, 1:5, FUN = "paste", sep = ",");a
     [,1]  [,2]  [,3]  [,4]  [,5]
[1,] "1,1" "1,2" "1,3" "1,4" "1,5"
[2,] "2,1" "2,2" "2,3" "2,4" "2,5"
[3,] "3,1" "3,2" "3,3" "3,4" "3,5"
[4,] "4,1" "4,2" "4,3" "4,4" "4,5"
[5,] "5,1" "5,2" "5,3" "5,4" "5,5"
select = matrix(ncol = 2, byrow = TRUE, c(1,1,3,1,2,4));select
     [,1] [,2]
[1,]    1    1
[2,]    3    1
[3,]    2    4
a[select]
[1] "1,1" "1,2" "3,4"
```

❑ Subset

```
df = data.frame(x = 1:3, y = 3:1, z = letters[1:3]);df
  x y z
1 1 3 a
2 2 2 b
3 3 1 c
df[df$x == 2, ]
  x y z
2 2 2 b
df[c("x", "z")]
  x z
1 1 a
2 2 b
3 3 c
```

```
df[,c("x", "z")]
  x z
1 1 a
2 2 b
3 3 c

str(df["x"])
'data.frame':	3 obs. of  1 variable:
 $ x: int  1 2 3
str(df[,"x"])
 int [1:3] 1 2 3
```

❑ Subset

	Simplifying	Preserving
Vector	x[[1]]	x[1]
List	x[[1]]	x[1]
Factor	x[1:4, drop =T]	x[1:4]
Array	x[1,] / x[, 1]	x[1, , drop = F] / x[, 1, drop = F]
Data frame	x[, 1] / x[[1]]	x[, 1, drop = F] / x[1]

```
a = list(a=1, b=2);a
$a
[1] 1
$b
[1] 2
```

```
a[1]
$a
[1] 1
a[[1]]
[1] 1
```

```
a["a"]
$a
[1] 1
a[["a"]]
[1] 1
```

❑ Out of bound index

operator	index	Atomic	List
[	oob	NA	list(NULL)
[	NA_real_	NA	list(NULL)
[	NULL	x[0]	list(NULL)
[[	oob	Error	Error
[[	NA_real_	Error	NULL
[[	NULL	Error	Error

❑ Assignment

```
x = 1:5;x
[1] 1 2 3 4 5
x[c(2,4)] = c(9,23);x
[1]  1  9  3 23  5
x[-1] = 99;x
[1]  1 99 99 99 99
```

```
df = data.frame(a = c(1, 10, NA));df
   a
1  1
2 10
3 NA
```

```
df$a[df$a < 5] = 0;df
   a
1  0
2 10
3 NA
> df$a
[1]  0 10 NA
```

Function

- Variable Scope - Dynamic loopup

```
f = function() x
f()
x = 15
f()
x = 20
f()
```

where? when?

```
f = function() {
  i = 10
  x
  cat(paste0(i,",",x))

}
codetools::findGlobals(f)
```

external dependencies of function

- Function call with argument

```
f <- function(abcdef, bcde1, bcde2) {
  list(a = abcdef, b1 = bcde1, b2 = bcde2)
}
str(f(1, 2, 3))
str(f(2, 3, abcdef = 1))
str(f(2, 3, a = 1))
str(f(1, 3, b = 1))
```

- Function call with list argument

```
mean(1:10, na.rm = TRUE)
args = list(1:10, na.rm = TRUE)
mean(args)
do.call(mean, args)
```

- Default argument

```
f <- function(a = 1, b = a * 2) {
  c(a, b)
}

f()
f(3)
f(3,5)
```

- Lazy evaluation

```
f <- function(x) {
  10
}
f()

f <- function(x) {
  force(x)
  10
}
f()
```

❑ Function

- Replcement function

```
second <- function(x, value) {
  x[2] <- value
  x
}
x = 1:10
second(x) = 5L

`second<-`<- function(x, value) {
  x[2] <- value
  x
}
x = 1:10
second(x) = 5L;x
```

- on.exit

```
in_dir <- function(dir, code) {
  old <- setwd(dir)  # return old working dir
  on.exit(setwd(old))

  force(code)
}
getwd()
in_dir("/", getwd())
getwd()
```

❑ Functional programming

- Imperative Programming : mutable variables, assignments, control structure(if-then-else, loop, break, continue, return) – C++, Java
- Logic Programming : formal logic – Prolog, Answer set programming(ASP)
- Functional Programming
  - restricted sense : not use imperative programming paradigm
  - wider sense : use function, functions can be values that are produces, consumed, composed.
  - function can be defined anywhere, including side other functions
  - like any other value, they can be passed as parameters to functions and returned as results
  - as for other values, there exists a set operators to compose functions

1959	1975-77	1978	1986	1990	1999	2000	2003	2005	2007
Lisp	ML, FP, Scheme	Smalltalk	Standard ML	Haskell, Erlang	XSLT	OCaml	Scala, XQuery	F#	Clojure

❑ Functional programming

```java
public class Factorial {

    public static long imperativeFactorial(int n){
        assert n > 0 : "n should be greater than 0 ";
        long result = 1;
        for(int i=2;i<=n;i++){
            result *= i;
        }
        return result;
    }

    public static long declarativeFactorial(int n){
        assert n > 0 : "n should be greater than 0 ";
        if(n==1)
            return 1;
        else
            return n * declarativeFactorial(n-1);
    }

}
```

Functional programming

- anonymous function

```
lapply(mtcars, function(x) length(unique(x)))
Filter(function(x) !is.numeric(x), mtcars)
integrate(function(x) sin(x) ^ 2, 0, pi)
```

- closures

```
power = function(exponent) {
  function(x) {
    x ^ exponent
  }
}
square <- power(2)
square(2)
cube <- power(3)
cube(2)
```

- Mutable state

```
new_counter <- function() {
  i <- 0
  function() {
    i <<- i + 1
    i
  }
}
one = new_counter()
one()
one()
```

```
i <- 0
new_counter2 <- function() {
  i <<- i + 1
  i
}
new_counter3 <- function() {
  i <- 0
  function() {
    i <- i + 1
    i
  }
}
one2 = new_counter2()
one2();one2()
one3 = new_counter3()
one3();one3()
```

Functional programming

- Lazy evaluation & closure

```
factory = function (K) {
       function (x) print(K + x)
}
funcs<-list()
for(i in 1:5)
   funcs[[i]]<-factory({cat("evaluating K:",i,"\n"); i})
funcs[[1]](10)

factory = function (K) {
  force(K)
  function (x) print(K + x)
}
funcs<-list()
for(i in 1:5)
  funcs[[i]]<-factory({cat("evaluating K:",i,"\n"); i})
funcs[[1]](10)
```

Functional programming

- List of functions

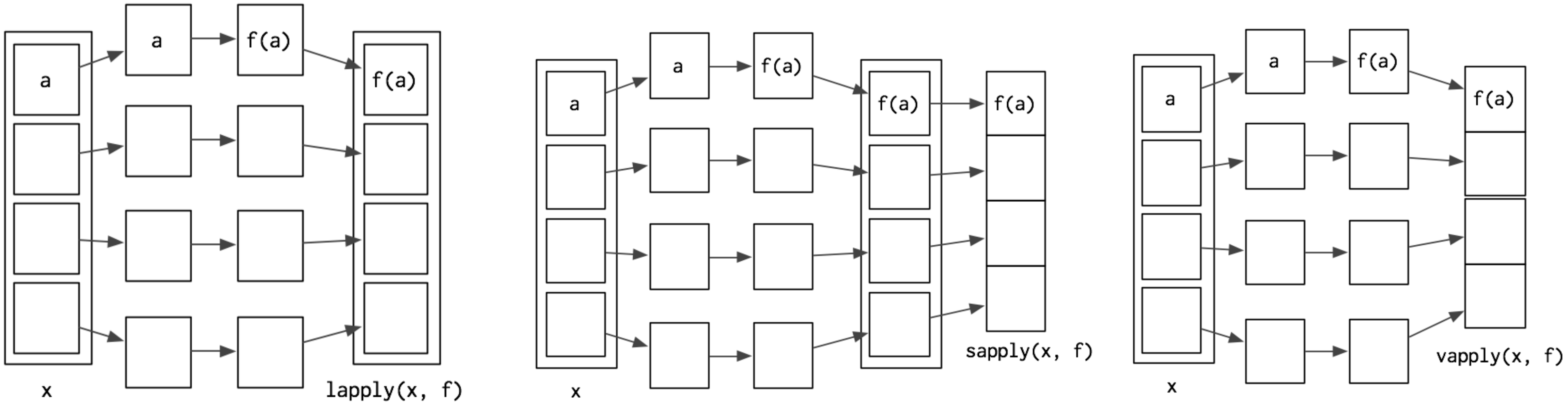
```
compute_mean <- list(
  base = function(x) mean(x),
  sum = function(x) sum(x) / length(x),
  manual = function(x) {
    total <- 0
    n <- length(x)
    for (i in seq_along(x)) {
      total <- total + x[i] / n
    }
    total
  }
)
```

```
compute_mean$base(x)
compute_mean[[2]](x)
compute_mean[["manual"]](x)

lapply(compute_mean, function(f) f(x))
```

ListOfFunctions.R

❑ Functional programming

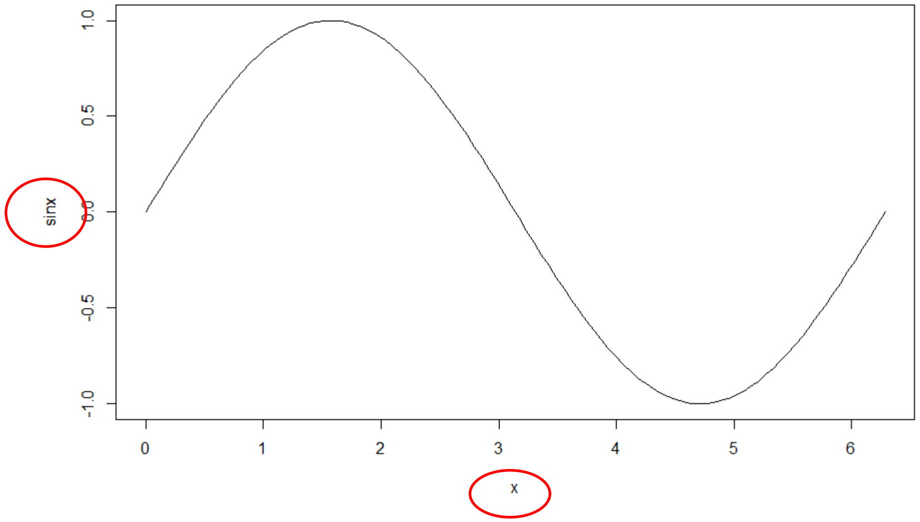
- lapply functions



lapply.R

❑ Non-standard evaluation

```
x = seq(0, 2 * pi, length = 100)
sinx = sin(x)
plot(x, sinx, type = "l")
```



- Capturing expression
  - substitute
  - deparse : char vector
  - library(ggplot2) / library(“ggplot2”)

❑ Performance

- 연산 속도 측정

```
library(microbenchmark)

x <- runif(100)
microbenchmark(
  sqrt(x),
  x ^ 0.5
)
```

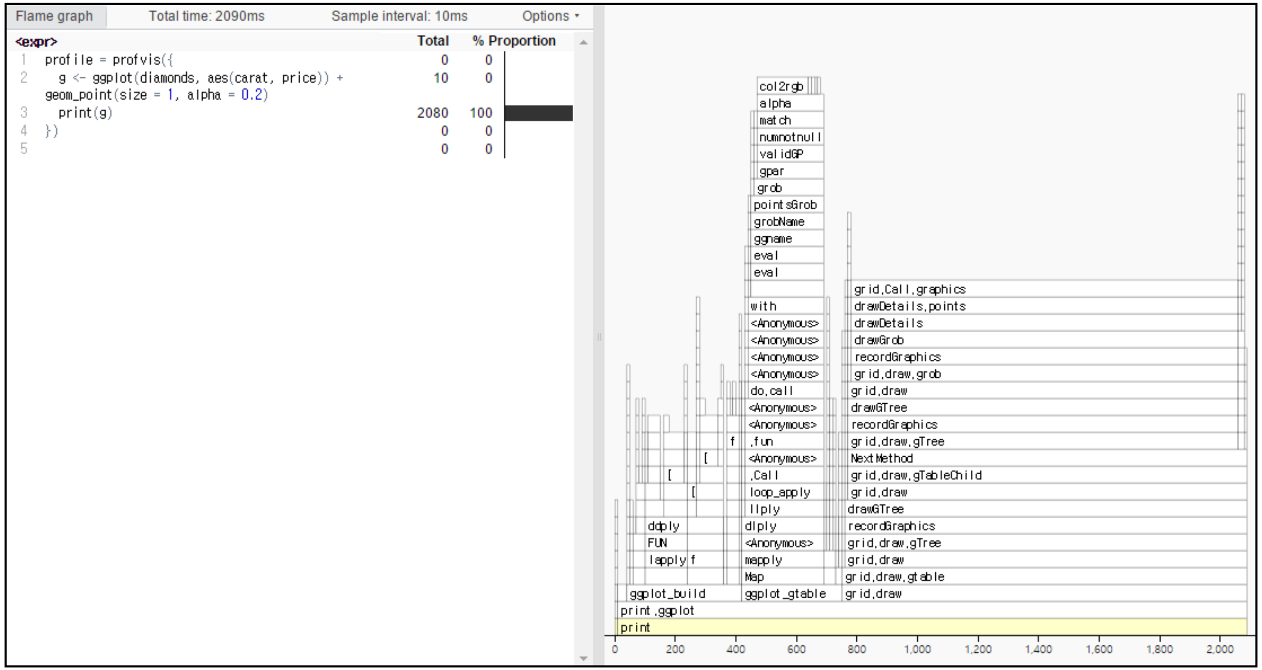
100번 수행한 시간에 대한 통계

- Lazy evaluation

```
f0 <- function() NULL
f1 <- function(a = 1) NULL
f2 <- function(a = 1, b = 1) NULL
f3 <- function(a = 1, b = 2, c = 3) NULL
f4 <- function(a = 1, b = 2, c = 4, d = 4) NULL
f5 <- function(a = 1, b = 2, c = 4, d = 4, e = 5) NULL
microbenchmark(f0(), f1(), f2(), f3(), f4(), f5(), times = 50)

Unit: nanoseconds
 expr min  lq   mean median  uq  max neval cld
 f0()   0   0  28.94    0.0   0  963    50  a
 f1()   0   0  86.76    0.0   1  962    50  a
 f2()   0   0 356.30    0.5 481 9625    50  ab
 f3()   0   0 250.36    1.0 481  963    50  ab
 f4()   0   0 298.70    1.0 482 1925    50  ab
 f5()   0 481 577.66  482.0 962 1925    50   b
```

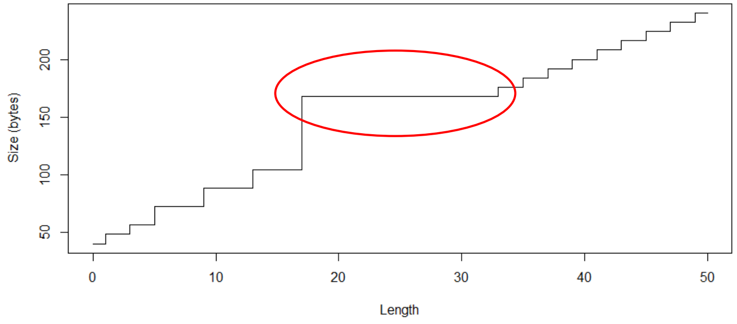
❑ Code profiling

```
devtools::install_github("rstudio/profvis")
library(profvis)
library(ggplot2)

profile = profvis({
  g <- ggplot(diamonds, aes(carat, price)) +
geom_point(size = 1, alpha = 0.2)
  print(g)
})
profile
```



Memory

```
library(pryr)
object_size(1:10)
sizes = sapply(0:50, function(n) object_size(seq_len(n)))
plot(0:50, sizes, xlab = "Length", ylab = "Size (bytes)", type = "s")
```




```
mem_used()
92.8 MB
mem_change(v <- list(1:1e8, 1:1e8, 1:1e8))
1.2 GB
mem_used()
1.29 GB
rm(v)
mem_used()
93.1 MB
```

- Memory profiling

```
devtools::install_github("hadley/lineprof")
library(lineprof)
profile = lineprof(f())
shine(profile)
```

❑ Data read

- file

```
df= read.table("http://www.ats.ucla.edu/stat/data/test.txt", header = T)
is.data.frame(df)
head(df)
?read.table

table.fixed = read.fwf("http://www.ats.ucla.edu/stat/data/test_fixed.txt", width = c(8, 1, 3, 1, 1, 1))
is.data.frame(table.fixed)
head(table.fixed)
```

- RDBMS

```
library(RJDBC)
drv = JDBC("com.mysql.jdbc.Driver",
          "/etc/jdbc/mysql-connector-java-3.1.14-bin.jar",
          identifier.quote="`")
conn = dbConnect(drv, "jdbc:mysql://localhost/test", "user", "pwd")
df = dbReadTable(conn, "iris")
df = dbGetQuery(conn,  "SELECT * FROM iris”)
```

❑ Data read

- HIVE

```
options( java.parameters = "-Xmx2g" )
library(rJava)
library(RJDBC)

cp = c("/usr/hdp/current/hive-client/lib/hive-jdbc.jar",  "/usr/hdp/current/hadoop-client/hadoop-ommon.jar")
.jinit(classpath=cp)
drv = JDBC("org.apache.hive.jdbc.HiveDriver", "/usr/hdp/current/hive-client/lib/hive-jdbc.jar",  identifier.quote="`")
conn = dbConnect(drv, "jdbc:hive2://servername:10000/demo", "user", "password")
df <- dbGetQuery(conn, "show databases")
```

- hdfs file (Spark)

```
Sys.setenv(SPARK_HOME=”/home/shige/bin/spark”)
.libPaths(c(file.path(Sys.getenv(“SPARK_HOME”), “R”, “lib”), .libPaths()))
library(SparkR)
sc = sparkR.init(master = "local[*]", sparkEnvir = list(spark.driver.memory="2g"))
sqlContext = sparkRSQL.init(sc)
df = read.df(sqlContext, “hdfs://namenode:port/xxx/yyy.parquet", "parquet")
```

Data Manipulation

❑ tidyr & dplyr package

```
library(tidyr)
library(dplyr)
```

❖ tidyr

- gather()
- spread()
- separate()
- unite()

❖ dplyr

- select()
- filter()
- group_by()
- summarise()
- arrange()
- join()
- mutate()

❖ %>% 연산자

```
a <- filter(data, variable == numeric_value)
b <- summarise(a, Total = sum(variable))
c <- arrange(b, desc(Total))
```

```
arrange(
        summarize(
            filter(data, variable == numeric_value),
            Total = sum(variable)
        ),
    desc(Total)
 )
```

```
data %>%
        filter(variable == “value”) %>%
        summarise(Total = sum(variable)) %>%
        arrange(desc(Total))
```

❑ data example

```
library(nycflights13)

str(flights)
str(weather)
str(planes)
str(airports)
```

```
str(flights)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':      336776 obs. of  16 variables:
 $ year     : int  2013 2013 2013 2013 2013 2013 2013 2013 2013 2013 ...
 $ month    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ day      : int  1 1 1 1 1 1 1 1 1 1 ...
 $ dep_time : int  517 533 542 544 554 554 555 557 557 558 ...
 $ dep_delay: num  2 4 2 -1 -6 -4 -5 -3 -3 -2 ...
 $ arr_time : int  830 850 923 1004 812 740 913 709 838 753 ...
 $ arr_delay: num  11 20 33 -18 -25 12 19 -14 -8 8 ...
 $ carrier  : chr  "UA" "UA" "AA" "B6" ...
 $ tailnum  : chr  "N14228" "N24211" "N619AA" "N804JB" ...
 $ flight   : int  1545 1714 1141 725 461 1696 507 5708 79 301 ...
 $ origin   : chr  "EWR" "LGA" "JFK" "JFK" ...
 $ dest     : chr  "IAH" "IAH" "MIA" "BQN" ...
 $ air_time : num  227 227 160 183 116 150 158 53 140 138 ...
 $ distance : num  1400 1416 1089 1576 762 ...
 $ hour     : num  5 5 5 5 5 5 5 5 5 5 ...
 $ minute   : num  17 33 42 44 54 54 55 57 57 58 ...
```

❑ gather()

```
flight_delay = flights[c("tailnum","arr_delay", "dep_delay")]
flight_delay = flight_delay[sample(nrow(flight_delay), 5), ]
flight_delay
delay_gather = flight_delay %>% gather(delay, time, arr_delay:dep_delay)
delay_gather
```

❖ flight_delay

	tailnum	arr_delay	dep_delay
1	N434UA	-29	-4
2	N520JB	9	26
3	N538UA	19	-2
4	N934XJ	1	10
5	N744P	-12	-7

	tailnum	delay	time
1	N434UA	arr_delay	-29
2	N520JB	arr_delay	9
3	N538UA	arr_delay	19
4	N934XJ	arr_delay	1
5	N744P	arr_delay	-12
6	N434UA	dep_delay	-4
7	N520JB	dep_delay	26
8	N538UA	dep_delay	-2
9	N934XJ	dep_delay	10
10	N744P	dep_delay	-7

❖ delay_gather

❑ spread()

```
head(delay_gather,10)
flight_return <- delay_gather %>% spread(delay, time)
head(flight_return)
```

❖ delay_gather

	tailnum	delay	time
1	N434UA	arr_delay	-29
2	N520JB	arr_delay	9
3	N538UA	arr_delay	19
4	N934XJ	arr_delay	1
5	N744P	arr_delay	-12
6	N434UA	dep_delay	-4
7	N520JB	dep_delay	26
8	N538UA	dep_delay	-2
9	N934XJ	dep_delay	10
10	N744P	dep_delay	-7

❖ flight_return

	tailnum	arr_delay	dep_delay
1	N434UA	-29	-4
2	N520JB	9	26
3	N538UA	19	-2
4	N934XJ	1	10
5	N744P	-12	-7

❑ seperate()

```
head(airport)
name_seperate <- airports %>% separate(name, c("prefix", "suffix"))
head(name_seperate)
```

❖ airport

```
  faa                          name      lat       lon  alt tz dst
1 04G             Lansdowne Airport 41.13047 -80.61958 1044 -5   A
2 06A Moton Field Municipal Airport 32.46057 -85.68003  264 -5   A
3 06C           Schaumburg Regional 41.98934 -88.10124  801 -6   A
4 06N               Randall Airport 41.43191 -74.39156  523 -5   A
5 09J         Jekyll Island Airport 31.07447 -81.42778   11 -4   A
6 0A9 Elizabethton Municipal Airport 36.37122 -82.17342 1593 -4   A
```

❖ name_seperate

```
  faa       prefix     suffix      lat       lon  alt tz dst
1 04G    Lansdowne    Airport 41.13047 -80.61958 1044 -5   A
2 06A        Moton      Field 32.46057 -85.68003  264 -5   A
3 06C   Schaumburg   Regional 41.98934 -88.10124  801 -6   A
4 06N      Randall    Airport 41.43191 -74.39156  523 -5   A
5 09J       Jekyll     Island 31.07447 -81.42778   11 -4   A
6 0A9 Elizabethton  Municipal 36.37122 -82.17342 1593 -4   A
```

❑ unite()

```
weather_part = weather[c("Date","Location", "MinTemp", "MaxTemp","Rainfall")]
head(weather_part)
temp_unite <- weather_part %>% unite(Temp, MinTemp, MaxTemp, sep = "/")
head(temp_unite)
```

❖ weather_part

	Date	Location	MinTemp	MaxTemp	Rainfall
1	2007-11-01	Canberra	8.0	24.3	0.0
2	2007-11-02	Canberra	14.0	26.9	3.6
3	2007-11-03	Canberra	13.7	23.4	3.6
4	2007-11-04	Canberra	13.3	15.5	39.8
5	2007-11-05	Canberra	7.6	16.1	2.8
6	2007-11-06	Canberra	6.2	16.9	0.0

	Date	Location	Temp	Rainfall
1	2007-11-01	Canberra	8/24.3	0.0
2	2007-11-02	Canberra	14/26.9	3.6
3	2007-11-03	Canberra	13.7/23.4	3.6
4	2007-11-04	Canberra	13.3/15.5	39.8
5	2007-11-05	Canberra	7.6/16.1	2.8
6	2007-11-06	Canberra	6.2/16.9	0.0

❖ temp_unite

❑ select()

```
head(planes)
planes_part = planes %>% select(tailnum, year, model:speed)
head(planes_part)
planes %>% select(starts_with("t"))
planes %>% select(-manufacturer, -speed)
```

❖ planes

	tailnum	year	type	manufacturer	model	engines	seats	speed	engine
1	N10156	2004	Fixed wing multi engine	EMBRAER	EMB-145XR	2	55	NA	Turbo-fan
2	N102UW	1998	Fixed wing multi engine	AIRBUS INDUSTRIE	A320-214	2	182	NA	Turbo-fan
3	N103US	1999	Fixed wing multi engine	AIRBUS INDUSTRIE	A320-214	2	182	NA	Turbo-fan
4	N104UW	1999	Fixed wing multi engine	AIRBUS INDUSTRIE	A320-214	2	182	NA	Turbo-fan
5	N10575	2002	Fixed wing multi engine	EMBRAER	EMB-145LR	2	55	NA	Turbo-fan
6	N105UW	1999	Fixed wing multi engine	AIRBUS INDUSTRIE	A320-214	2	182	NA	Turbo-fan

	tailnum	year	model	engines	seats	speed
1	N10156	2004	EMB-145XR	2	55	NA
2	N102UW	1998	A320-214	2	182	NA
3	N103US	1999	A320-214	2	182	NA
4	N104UW	1999	A320-214	2	182	NA
5	N10575	2002	EMB-145LR	2	55	NA
6	N105UW	1999	A320-214	2	182	NA

❖ planes_part

❑ filter()

```
summary(planes)
planes_2004 = planes %>% filter(year=='2004', engines > 2)
head(planes_2004)
```

< Less than		!=	Not equal to
> Greater than		%in%	Group membership
== Equal to		is.na	is NA
<= Less than or equal to		!is.na	is not NA
>= Greater than or equal to		&,|,!	Boolean operators

❖ planes

```
     tailnum           year                  engines
Length:3322       Min.   :1956    ...    Min.   :1.000    ...
Class :character  1st Qu.:1997    ...    1st Qu.:2.000    ...
Mode  :character  Median :2001    ...    Median :2.000    ...
                  Mean   :2000    ...    Mean   :1.995    ...
                  3rd Qu.:2005    ...    3rd Qu.:2.000    ...
                  Max.   :2013    ...    Max.   :4.000    ...
                  NA's   :70
```

❖ planes_2004

```
  tailnum year                   type manufacturer      model engines seats speed    engine
1  N854NW 2004 Fixed wing multi engine       AIRBUS A330-223       3   379    NA Turbo-fan
2  N856NW 2004 Fixed wing multi engine       AIRBUS A330-223       3   379    NA Turbo-fan
```

❑ summarise() & group_by()

```
flights %>% summarise(dep_delay_mean=mean(dep_delay),arr_delay_mean=mean(arr_delay))
head(flights);summary(flights)
flights_complete = flights %>% filter(!is.na(dep_delay), !is.na(arr_delay))
summary(flights_complete)
flights_complete %>% summarise(dep_delay_mean=mean(dep_delay),arr_delay_mean=mean(arr_delay))
flights_groupby_summarise = flights_complete %>% group_by(month) %>%
                                             summarise(dep_delay_mean=mean(dep_delay),arr_delay_mean=mean(arr_delay))
flights_groupby_summarise
```

❖ 첫번째 summarise 결과

Source: local data frame [1 x 2]

	dep_delay_mean	arr_delay_mean
1	NA	NA

❖ 두번째 summarise 결과

Source: local data frame [1 x 2]

	dep_delay_mean	arr_delay_mean
1	12.55516	6.895377

Source: local data frame [12 x 3]

	month	dep_delay_mean	arr_delay_mean
1	1	9.985491	6.1299720
2	2	10.760239	5.6130194
3	3	13.164289	5.8075765
4	4	13.849187	11.1760630
5	5	12.891709	3.5215088
6	6	20.725614	16.4813296
7	7	21.522179	16.7113067
8	8	12.570524	6.0406524
9	9	6.630285	-4.0183636
10	10	6.233175	-0.1670627
11	11	5.420340	0.4613474
12	12	16.482161	14.8703553

arrange()

```
flights_groupby_summarise_arrange = flights_groupby_summarise %>% arrange(dep_delay_mean)
flights_groupby_summarise_arrange
flights_groupby_summarise_arrange = flights_groupby_summarise %>% arrange(desc(arr_delay_mean))
flights_groupby_summarise_arrange
```

Source: local data frame [12 x 3]

	month	dep_delay_mean	arr_delay_mean
1	11	5.420340	0.4613474
2	10	6.233175	-0.1670627
3	9	6.630285	-4.0183636
4	1	9.985491	6.1299720
5	2	10.760239	5.6130194
6	8	12.570524	6.0406524
7	5	12.891709	3.5215088
8	3	13.164289	5.8075765
9	4	13.849187	11.1760630
10	12	16.482161	14.8703553
11	6	20.725614	16.4813296
12	7	21.522179	16.7113067

Source: local data frame [12 x 3]

	month	dep_delay_mean	arr_delay_mean
1	7	21.522179	16.7113067
2	6	20.725614	16.4813296
3	12	16.482161	14.8703553
4	4	13.849187	11.1760630
5	1	9.985491	6.1299720
6	8	12.570524	6.0406524
7	3	13.164289	5.8075765
8	2	10.760239	5.6130194
9	5	12.891709	3.5215088
10	11	5.420340	0.4613474
11	10	6.233175	-0.1670627
12	9	6.630285	-4.0183636

❑ join()

```
flights_dest_group = flights %>%  group_by(dest) %>%  filter(!is.na(arr_delay)) %>%
  summarise(arr_delay = mean(arr_delay),  n = n()  ) %>%  arrange(desc(arr_delay))
location = airports %>%  select(dest = faa, name, lat, lon)
flights_join = flights_dest_group %>% left_join(location)
flights_join = flights_dest_group %>% left_join(location, by='dest')
```

```
  dest arr_delay   n
1  CAE  41.76415 106
2  TUL  33.65986 294
3  OKC  30.61905 315
4  JAC  28.09524  21
5  TYS  24.06920 578
6  MSN  20.19604 556
```

```
  dest                           name      lat       lon
1  04G              Lansdowne Airport 41.13047 -80.61958
2  06A  Moton Field Municipal Airport 32.46057 -85.68003
3  06C            Schaumburg Regional 41.98934 -88.10124
4  06N                Randall Airport 41.43191 -74.39156
5  09J          Jekyll Island Airport 31.07447 -81.42778
6  0A9 Elizabethton Municipal Airport 36.37122 -82.17342
```

```
  dest arr_delay   n                   name      lat        lon
1  CAE  41.76415 106  Columbia Metropolitan 33.93883  -81.11953
2  TUL  33.65986 294              Tulsa Intl 36.19839  -95.88811
3  OKC  30.61905 315       Will Rogers World 35.39309  -97.60073
4  JAC  28.09524  21    Jackson Hole Airport 43.60733 -110.73775
5  TYS  24.06920 578          Mc Ghee Tyson 35.81097  -83.99403
6  MSN  20.19604 556 Dane Co Rgnl Truax Fld 43.13986  -89.33751
```

❑ join()

❖ Superheroes

name	alignment	gender	publisher
Magneto	bad	male	Marvel
Storm	good	female	Marvel
Mystique	bad	female	Marvel
Batman	good	male	DC
Joker	bad	male	DC
Catwoman	bad	female	DC
Hellboy	good	male	Dark Horse Comics

❖ Publishers

publisher	founded
DC	1934
Marvel	1939
Image	1992

❖ Superheroes %>% inner_join(Publishers, by=publisher)

name	alignment	gender	publisher	founded
Magneto	bad	male	Marvel	1939
Storm	good	female	Marvel	1939
Mystique	bad	female	Marvel	1939
Batman	good	male	DC	1934
Joker	bad	male	DC	1934
Catwoman	bad	female	DC	1934

❑ join()

❖ Superheroes %>% semi_join(Publishers, by=publisher)

name	alignment	gender	publisher
Batman	good	male	DC
Joker	bad	male	DC
Catwoman	bad	female	DC
Magneto	bad	male	Marvel
Storm	good	female	Marvel
Mystique	bad	female	Marvel

❖ Superheroes %>% left_join(Publishers, by=publisher)

name	alignment	gender	publisher	founded
Magneto	bad	male	Marvel	1939
Storm	good	female	Marvel	1939
Mystique	bad	female	Marvel	1939
Batman	good	male	DC	1934
Joker	bad	male	DC	1934
Catwoman	bad	female	DC	1934
Hellboy	good	male	Dark Horse Comics	NA

❖ Superheroes %>% anti_join(Publishers, by=publisher)

name	alignment	gender	publisher
Hellboy	good	male	Dark Horse Comics

❑ mutate()

```
flights_mutate = flights %>%  select(year, month, day, tailnum, hour, minute) %>%  mutate(time = hour + minute / 60)
flights_mutate_summarise = flights %>% mutate(time = hour + minute / 60) %>%
                                group_by(time) %>%  summarise(arr_delay = mean(arr_delay, na.rm = TRUE), n = n())
```

❖ flights_mutate

	year	month	day	tailnum	hour	minute	time
1	2013	1	1	N14228	5	17	5.283333
2	2013	1	1	N24211	5	33	5.550000
3	2013	1	1	N619AA	5	42	5.700000
4	2013	1	1	N804JB	5	44	5.733333
5	2013	1	1	N668DN	5	54	5.900000
6	2013	1	1	N39463	5	54	5.900000

❖ flights_mutate_summarise

	time	arr_delay	n
1	0.01666667	75.96000	25
2	0.03333333	90.00000	35
3	0.05000000	65.46154	26
4	0.06666667	60.50000	26
5	0.08333333	74.50000	21
6	0.10000000	91.90909	22

❑ data.table package

```
library(data.table)
df = copy(flights)
dt = setDT(df)
```

- large data set
- fast
- clean code

❖ select columns

❖ select rows

❖ group by

❖ add, remove fields

❖ join

❖ fread

DataTable.R

Visualization

❑ ggplot2 package

- ❖ data
- ❖ aesthetic mapping
- ❖ geometric object
- ❖ statistical transformations
- ❖ scales
- ❖ coordinate system
- ❖ position adjustments
- ❖ faceting

http://docs.ggplot2.org/current/

❖ ggplot2에서 지원하지 않는 기능

- 3 차원 그래프 : rgl package
- 그래프 이론 형태의 그래프(node/edges layout) : igraph package
- 대화형 그래프 : ggvis package

❖ ggplot2 구조

```
ggplot(data = <default data set>,
     aes(x = <default x axis variable>,
        y = <default y axis variable>,
        ... <other default aesthetic mappings>),
     ... <other plot defaults>) +

     geom_<geom type>(aes(size = <size variable for this geom>,
                 ... <other aesthetic mappings>),
             data = <data for this point geom>,
             stat = <statistic string or function>,
             position = <position string or function>,
             color = <"fixed color specification">,
             <other arguments, possibly passed to the _stat_ function) +

  scale_<aesthetic>_<type>(name = <"scale label">,
               breaks = <where to put tick marks>,
               labels = <labels for tick marks>,
               ... <other options for the scale>) +

  theme(plot.background = element_rect(fill = "gray"),
      ... <other theme elements>)
```

❑ ggplot2 package

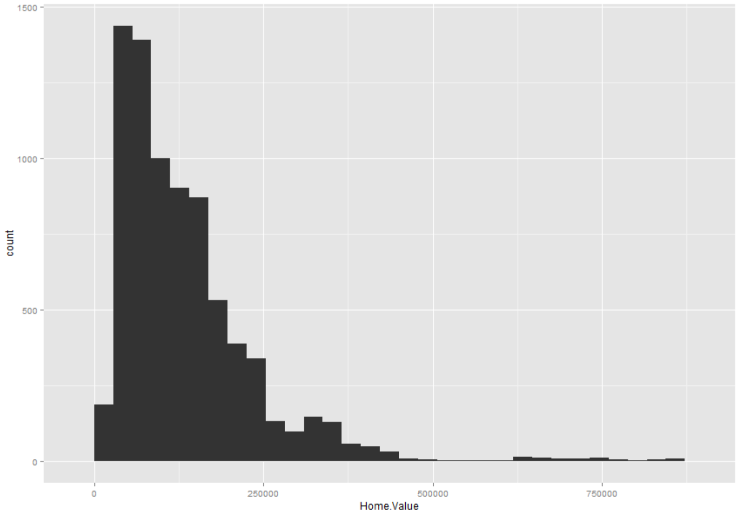
❖ sample data

```
> housing = read.csv("landdata-states.csv")
> str(housing)
'data.frame':	7803 obs. of  9 variables:
 $ State           : Factor w/ 51 levels "AK","AL","AR",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ region          : Factor w/ 4 levels "Midwest","N. East",..: 4 4 4 4 4 4 4 4 4 4 ...
 $ Date            : int  20101 20102 20093 20094 20074 20081 20082 20083 20084 20091 ...
 $ Home.Value      : int  224952 225511 225820 224994 234590 233714 232999 232164 231039 229395 ...
 $ Structure.Cost  : int  160599 160252 163791 161787 155400 157458 160092 162704 164739 165424 ...
 $ Land.Value      : int  64352 65259 62029 63207 79190 76256 72906 69460 66299 63971 ...
 $ Land.Share..Pct.: num  28.6 28.9 27.5 28.1 33.8 32.6 31.3 29.9 28.7 27.9 ...
 $ Home.Price.Index: num  1.48 1.48 1.49 1.48 1.54 ...
 $ Land.Price.Index: num  1.55 1.58 1.49 1.52 1.88 ...
```

❖ histogram

```
> ggplot(housing, aes(x=Home.Value)) + geom_histogram()
stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```

```
> ggplot(housing, aes(x=Home.Value)) + geom_histogram(bins=100)
> ggplot(housing, aes(x=Home.Value)) + geom_histogram(binwidth = 4000)
```



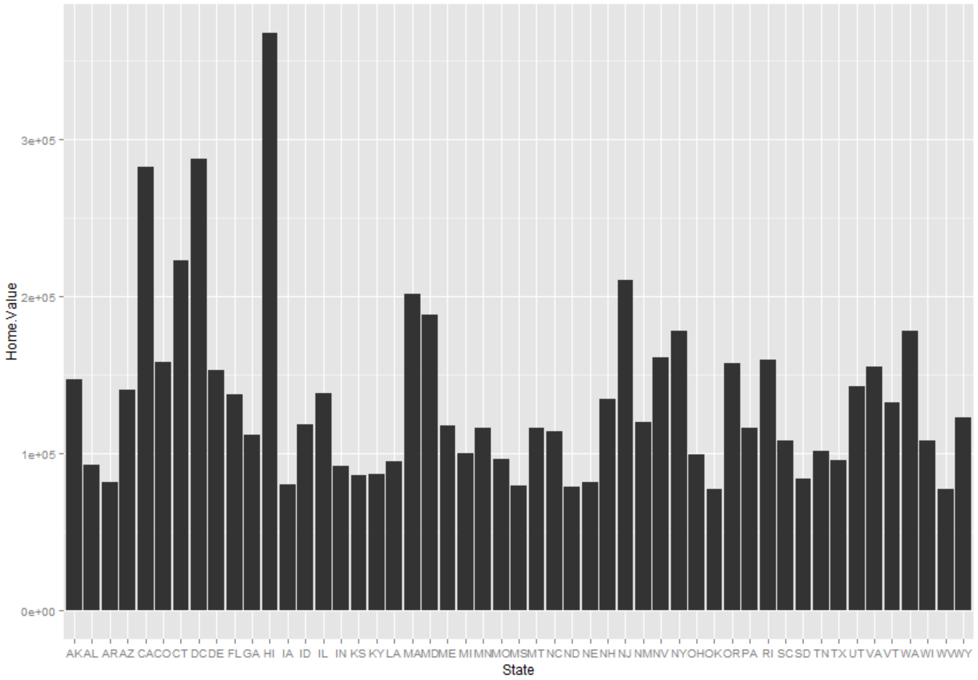
❑ ggplot2 package

❖ Statistical transformation

```
housing.sum = aggregate(housing["Home.Value"], housing["State"], FUN=mean)
head(housing.sum, 10)

ggplot(housing.sum, aes(x=State, y=Home.Value)) + geom_bar()
ggplot(housing.sum, aes(x=State, y=Home.Value)) + geom_bar(stat="identity")
```

❖ housing.sum

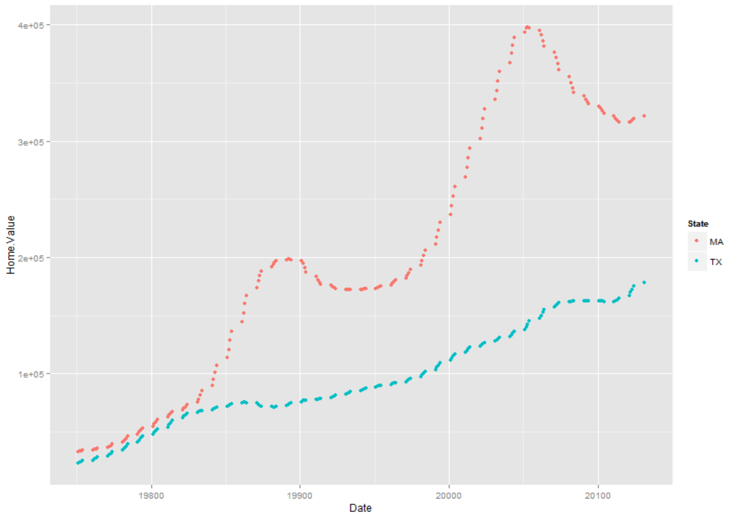
```
   State Home.Value
1     AK  147385.14
2     AL   92545.22
3     AR   82076.84
4     AZ  140755.59
5     CA  282808.08
6     CO  158175.99
7     CT  223063.08
8     DC  287552.56
9     DE  152905.53
10    FL  137842.59
```



ggplot2 package

- scatter plot

```
ggplot(subset(housing, State %in% c("MA", "TX")), aes(x=Date, y=Home.Value, color=State))+geom_point()
```



ggplot2 package

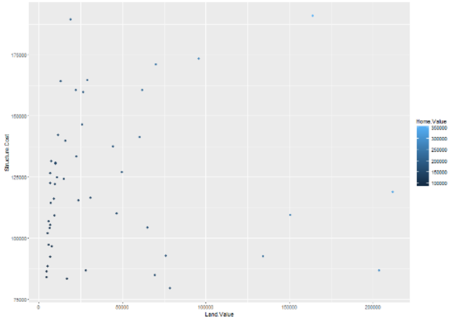
- Aesthetics
  - position(on the x, y axes)
  - color(outside color)
  - fill(inside color)
  - shape(of point)
  - linetype
  - size

- Geometic objects
  - geom_point : scatter plot, dot plot
  - geom_line : time series, trend line
  - geom_boxplot : box plots

help.search("geom_", package = "ggplot2")

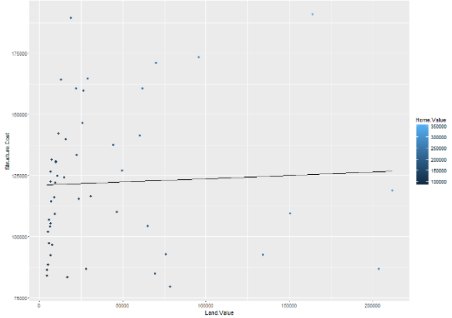
http://docs.ggplot2.org/current/

❑ ggplot2 package



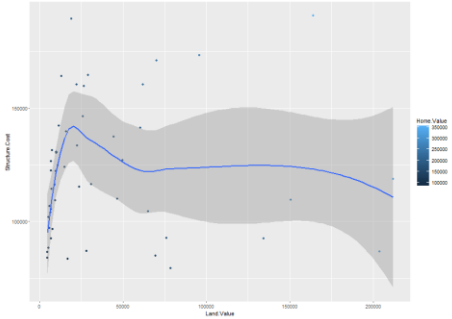
❖ Scatter plot

```
hp2001Q1 = subset(housing, Date == 20011)
p1 = ggplot(hp2001Q1, aes(y = Structure.Cost, x = Land.Value))
(p2 = p1 + geom_point(aes(color = Home.Value)))
```



❖ Prediction line

```
hp2001Q1$pred.SC <- predict(lm(Structure.Cost ~ Land.Value, data = hp2001Q1))
(p3 = p2 + geom_line(aes(y=hp2001Q1$pred.SC)))
```



❖ Smoothers

```
(p4 = p2 + geom_smooth(method=loess))
```

❑ ggplot2 package

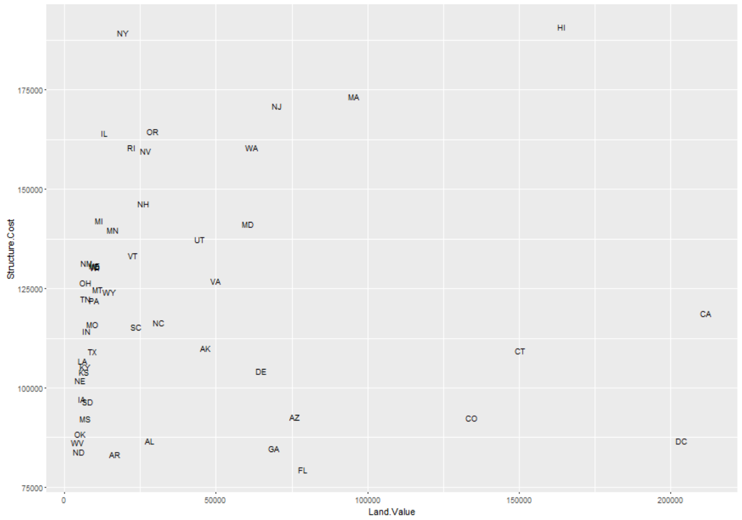
❖ Text

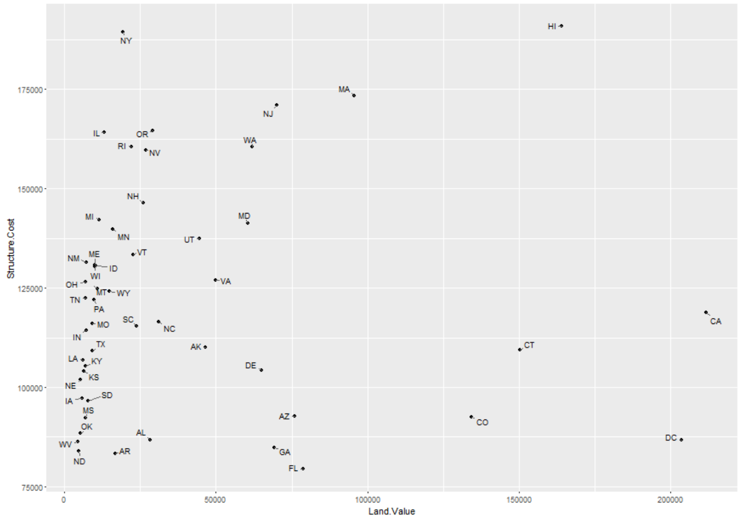
(p5 = p1 + geom_text(aes(label=State), size=3))

❖ 겹침 방지

library(ggrepel)

(p6 = p1 + geom_point() + geom_text_repel(aes(label=State), size = 3))





❑ ggplot2 package

❖ Asethetic mapping vs. Assignment

```
p1 + geom_point(aes())

p1 + geom_point(aes(size=100, color="red"))

p1 + geom_point(aes(), size=2, color="red")                 # 고정 값은 ase() 밖에서 설정

p1 + geom_point(aes(color=Home.Value, shape = region))   # ase() 안에서는 field로 설정
```

<< 실습 >>

실습 데이터 : EconomistData.csv

1. x 축은 CPI, y축은 HDI로 scatter plot
2. 1번 plot 점의 색깔은 파란색으로
3. 점의 색깔을 Region 별로 다르게
4. Region에 의한 CPI boxplot
5. box plot 과 scatter plot overlay

<< 실습 >>

6. 1번 plot에 lm method를 이용하여 smoothing line 추가
7. 1번 plot에 기본 method를 이용하여 smoothing line 추가

❑ ggplot2 package

❖ Scale : 데이터와 aesthetics 간의 mapping 조정

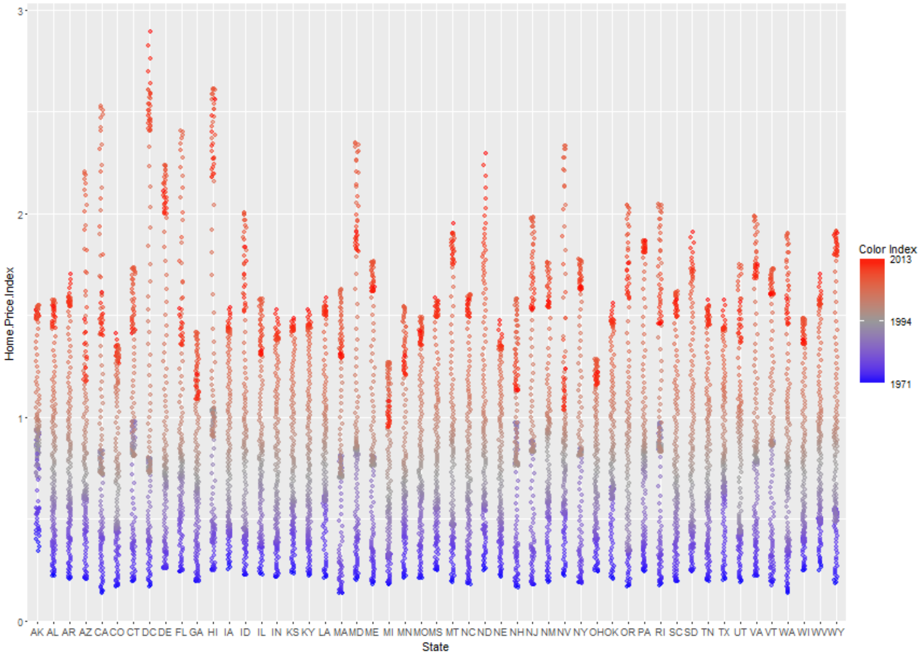
scale_<aestheitc>_<type>

position, color, fill, size, shape, line type

help.search(“scale_”, package = "ggplot2")

http://docs.ggplot2.org/current/

```
(p3 = ggplot(housing, aes(x = State, y = Home.Price.Index)))
(p4 = p3 + geom_point(aes(color=Date), alpha=0.5, size=1.5,
                      position=position_jitter(width=0.25, height=0)))

(p4 +  scale_x_discrete(name="State Abbreviation") +
       scale_color_continuous(name="Color Index",
                 breaks = c(19751, 19941, 20131),
                 labels = c(1971, 1994, 2013),
                 low = "blue", high = "red"))

(p4 +  scale_color_gradient2(name="Color Index",
                 breaks = c(19751, 19941, 20131),
                 labels = c(1971, 1994, 2013),
                 low = "blue",
                 high = "red",
                 mid = "gray60",
                 midpoint = 19941))
```

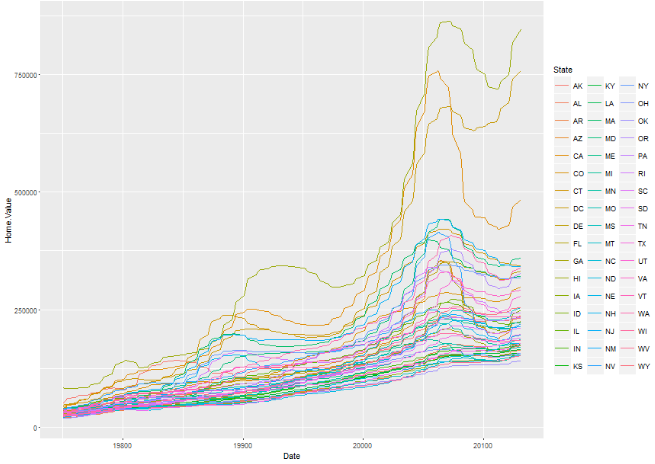


❑ ggplot2 package

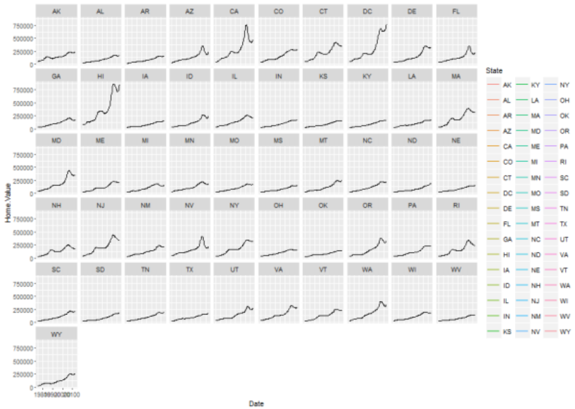
❖Faceting : 데이터셋을 일부를 다른 panel에 표시

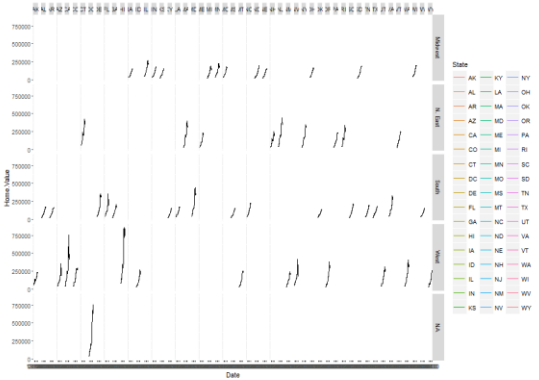
facet_wrap() : 1차원

facet_grid() : 2차원




```
(p5 = ggplot(housing, aes(x = Date, y = Home.Value)) +
        geom_line(aes(color = State)))

(p5 <- p5 + geom_line() +  facet_wrap(~State, ncol = 10))

(p5 + geom_line() +  facet_grid(region~State))
```





ggplot2 package

❖ Theme : 데이터 plot 이외의 다른 요소 설정(축 레이블, 배경, 범례 등)

```
p5 + theme_linedraw()
p5 + theme_light()

# theme 재정의
p5 + theme_minimal()
p5 + theme_minimal()+ theme(text = element_text(color = "turquoise"))


theme_new = theme_bw() +
  theme(plot.background = element_rect(size = 1, color = "blue", fill = "black"),
        text=element_text(size = 12, family = "Arial", color = "ivory"),
        axis.text.y = element_text(colour = "purple"),
        axis.text.x = element_text(colour = "red"),
        panel.background = element_rect(fill = "pink"),
        strip.background = element_rect(fill = "orange"))

p5 + theme_new
```

❑ ggplot2 package

❖ 두개의 변수로 plot 그리기

```
housing.byyear = aggregate(cbind(Home.Value, Land.Value) ~ Date,
                                    data = housing, mean)
head(housing.byyear)

ggplot(housing.byyear, aes(x=Date)) +
        geom_line(aes(y=Home.Value), color="red") +
        geom_line(aes(y=Land.Value), color="blue")
```

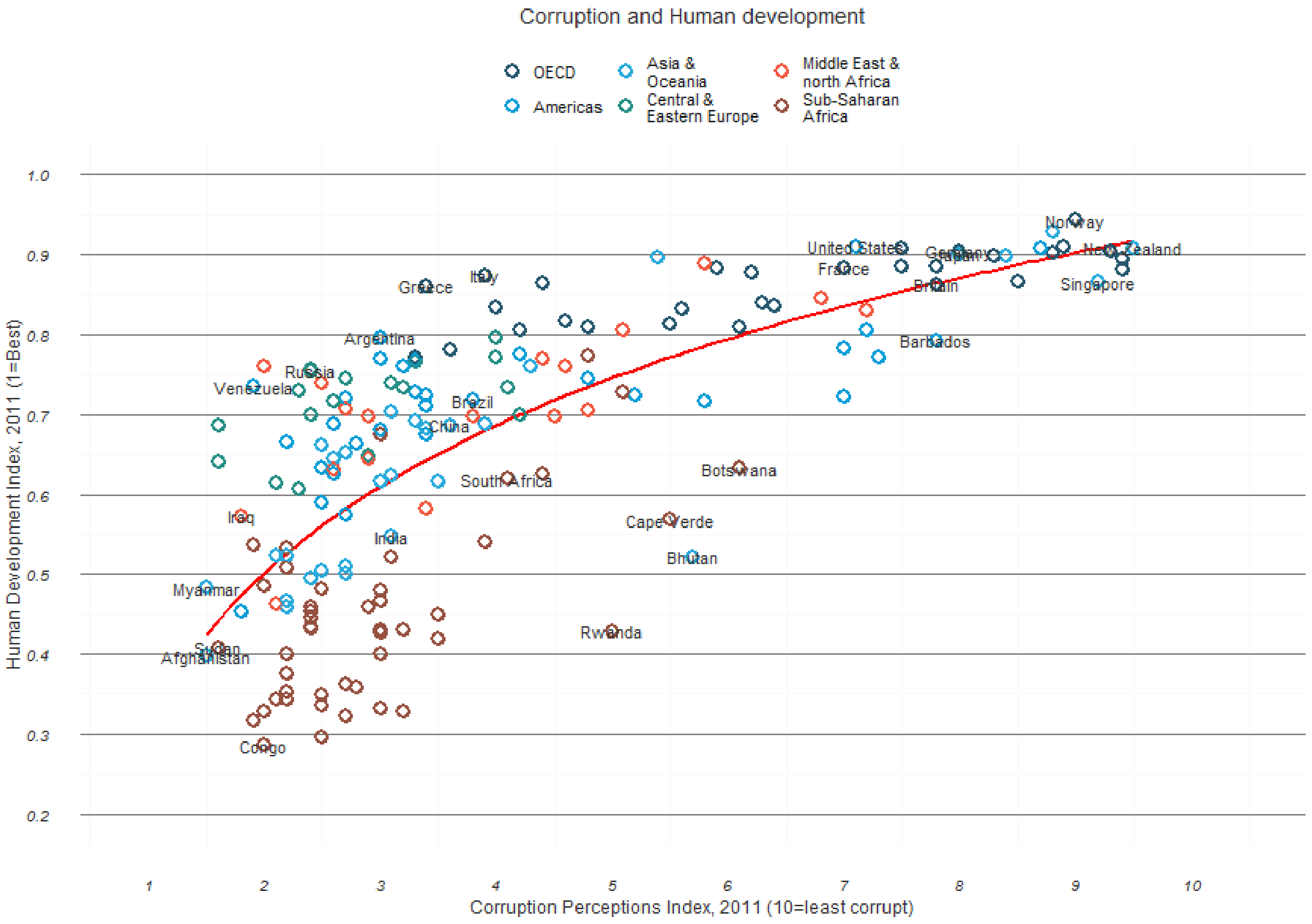
```
library(tidyr)
home.land.byyear = gather(housing.byyear, value = "value",
                                  key = "type",Home.Value, Land.Value)
head(home.land.byyear)

ggplot(home.land.byyear, aes(x=Date, y=value, color=type)) +  geom_line()
```

❑ ggplot2 package

❖ 실습

- 데이터 : EconomistData.csv

1. scatter plot
2. trend line
3. open point
4. labeling
5. 겹침 방지
6. 범례 변경
7. scale 설정 : x, y, color
8. theme 설정

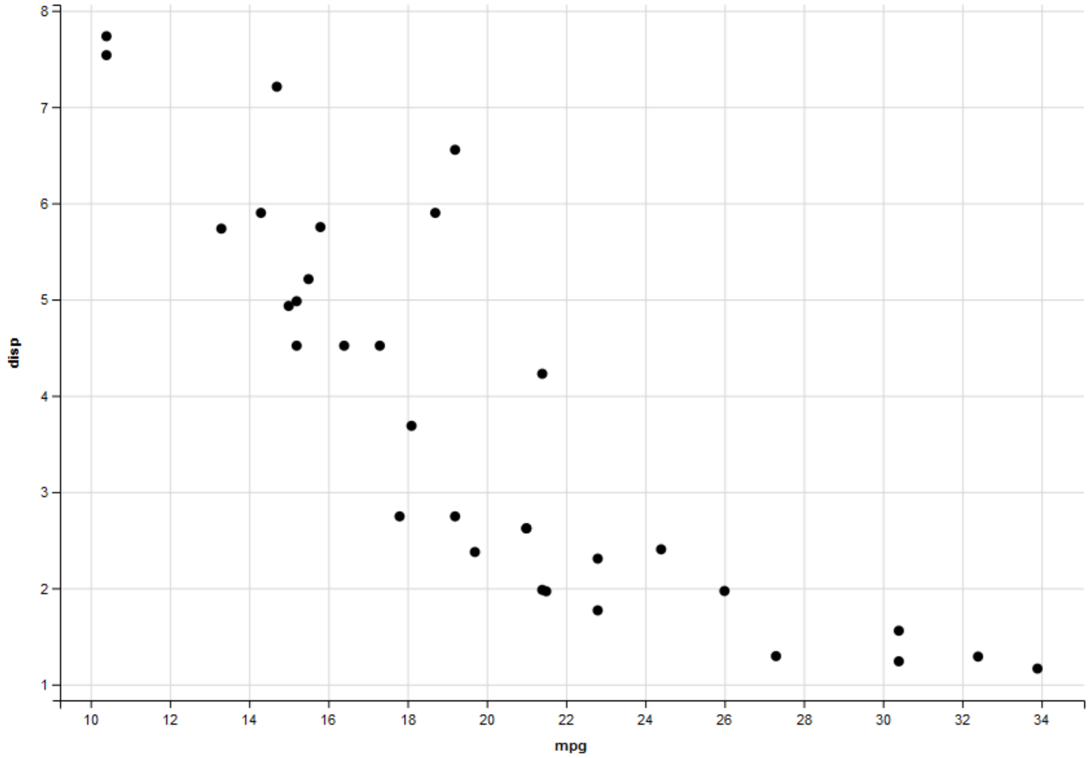


❑ ggvis package

```
library(ggvis)
library(dplyr)
library(shiny)

mtcars %>%  ggvis(x = ~mpg, y = ~disp) %>%
            mutate(disp = disp / 61.0237) %>%
            layer_points()
```

ggvis.R

https://rstudio-pubs-static.s3.amazonaws.com/1704_8f4e918c76cc447fac11113df250e02b.html



Machine Learning with R

❑ caret package

❖ Caret(Classification and regression training)

- data splitting
- pre-processing
- feature selection
- model tuning using resampling
- variable importance estimation

❖ Machine learning algorithm

- Linear discriminant analysis
- Regression
- Naïve Bayes
- Support vector machines
- Classification, regression trees
- Random forests, Boosting

https://topepo.github.io/caret/modelList.html

EDA(Exploratory data analysis)

```
#install.packages("caret", dependencies = c("Depends", "Suggests"))
library(caret)
library(kernlab)
data(spam)
data = spam

dim(data)
str(data)
sapply(data, class)
summary(data)

head(data,10)
levels(data$type)

percentage <- prop.table(table(data$type)) * 100
cbind(freq=table(data$type), percentage=percentage)
```

	freq	percentage
nonspam	2788	60.59552
spam	1813	39.40448

❑ EDA(Exploratory data analysis)

```
features <- data[,1:57]
target <- data[,58]

par(mfrow=c(1,4))

for(i in 1:57) {
  boxplot(features[,i], main=names(data)[i])
}

plot(target)

partFeatures = data[,1:4]
featurePlot(x=partFeatures, y=target, plot="ellipse")
featurePlot(x=partFeatures, y=target, plot="box")
scales = list(x=list(relation="free"), y=list(relation="free"))
featurePlot(x=partFeatures, y=target, plot="density", scales=scales)
```

```
x <- matrix(rnorm(50*5),ncol=5)
y <- factor(rep(c("A", "B"),  25))

# classification
featurePlot(x, y, "ellipse")
featurePlot(x, y, "strip", jitter = TRUE)
featurePlot(x, y, "box")
featurePlot(x, y, "pairs")

# regression
pairs, scatter
```

❑ Data slicing

```
sampling = createDataPartition(y=data$type, p=0.75, list=F)
sampling

trainData = data[sampling,]
testData = data[-sampling,]
dim(trainData);dim(testData)

# folding
set.seed(1234)
training = createFolds(y=data$type, k=10, list=T, returnTrain=T)
set.seed(1234)
testing = createFolds(y=data$type, k=10, list=T, returnTrain=F)

sapply(training, length)
sapply(testing, length)
training[[1]][1:10]
testing[[1]][1:10]
```

```
# resampling
set.seed(1234)
resampling = createResample(y=data$type,
                            times=10, list=T)
sapply(resampling, length)
resampling[[1]][1:10]

# time slices
set.seed(1234)
tme = 1:1000
timeslicing = createTimeSlices(tme,
                     initialWindow=20, horizon=10)
names(timeslicing)
sapply(timeslicing$train, length)
sapply(timeslicing$test, length)
timeslicing$train[[1]]
timeslicing$test[[1]]
```

❑ Training

```
model = train(type~., data=trainData, method="glm")
args(train.default)
```

```
function (x, y, method = "rf", preProcess = NULL, ...,
    weights = NULL,
    metric = ifelse(is.factor(y), "Accuracy", "RMSE"),
    maximize = ifelse(metric %in% c("RMSE", "logLoss"),
                              FALSE, TRUE),
    trControl = trainControl(),
    tuneGrid = NULL, tuneLength = 3)
NULL
```

```
Generalized Linear Model

3451 samples
  57 predictor
   2 classes: 'nonspam', 'spam'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 3451, 3451, 3451, 3451, 3451,
3451, ...
Resampling results

  Accuracy   Kappa      Accuracy SD  Kappa SD
  0.9199084  0.8318034  0.01199406   0.02314435
```

❑ Training

```
control <- trainControl(method = 'repeatedcv',
                        repeats = 5,
                        number = 5,
                        classProbs = T)
model = train(type~., data=trainData, method="glm“,
              trControl = control)
```

❖ method : resampling

- boot - bootstrapping
- boot632 - bootstrapping with adjustment
- cv - cross validation
- repeatedcv - repeated cross validation
- LOOCV - leave one out cross validation

❖ repeats

- subsampling을 반복 하는 회수
- 숫자가 커지면 수행속도 느려짐

❖ number

- boot / cross validation
- 사용할 subsampling 개수

❖ classProbs

- 결과값을 확률로 나타낼 것인가

  분류를 할 것인가

❑ preprocessing

```
ggplot(trainData, aes(x=capitalAve)) + geom_histogram()
mean(trainData$capitalAve)
sd(trainData$capitalAve)    # 값의 편차가 너무 큼

# standarization
capitalAveS = (trainData$capitalAve - mean(trainData$capitalAve)) / sd (trainData$capitalAve)
mean(capitalAveS)
sd(capitalAveS)   # 1

preObj = preProcess(trainData[,-58], method=c("center","scale"))
predict(preObj, trainData[,-58])
capitalAveS = predict(preObj, trainData[,-58])$capitalAve
mean(capitalAveS)
sd(capitalAveS)
par(mfrow=c(1,2))
hist(capitalAveS); qqnorm(capitalAveS)

model = train(type~., data=trainData, method="glm",
              trControl = control, preProcess=c("center","scale"))
```

❑ preprocessing

```
# box-cox transform
preObj = preProcess(trainData[,-58], method=c("BoxCox"))
capitalAveS = predict(preObj, trainData[,-58])$capitalAve
mean(capitalAveS)
sd(capitalAveS)
hist(capitalAveS); qqnorm(capitalAveS)

# missing value(Imputing data)
set.seed(1234)
trainData$capAve = trainData$capitalAve
summary(trainData$capAve)
selectNA = rbinom(dim(trainData)[1], size=1, prob=0.05) == 1
trainData$capAve[selectNA] = NA
summary(trainData$capAve)

preObj = preProcess(trainData[,-58], method=c("knnImpute"))
capAve = predict(preObj, trainData[,-58])$capAve
summary(capAve)
```

❑ preprocessing

method	설명	method	설명
scale	data / sd(data)	zv	zero variance
center	data - mean(data)	nzv	near zero variance
range	값을 [0,1] 사이의 값으로 scaling(normalization)	knnImpute	knn을 이용해서 NA 근처의 값들을 가중평균해서 값을 채움
Box-Cox	치우친 데이터를 정규분포화 (positive value)	bagImpute	Bagged tree model을 통해 NA 값을 예측하여 채움
YeoJohnson	치우친 데이터를 정규분포화 (0, negative 도 가능)	medianImpute	중앙값으로 NA 값을 채움
expoTrans	지수함수를 이용한 정규분포화(positive, negative)		
pca	주성분 분석		
ica	독립성분분석, n.comp 지정해야 함		

❑ Evaluation

```
modelPredict =  predict(model, testData)
confusionMatrix(modelPredict, testData$type)
```

```
Confusion Matrix and Statistics

          Reference
Prediction nonspam spam
   nonspam     653   38
   spam         44  415

               Accuracy : 0.9287
                 95% CI : (0.9123, 0.9429)
    No Information Rate : 0.6061
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.851
 Mcnemar's Test P-Value : 0.5808

            Sensitivity : 0.9369
            Specificity : 0.9161
         Pos Pred Value : 0.9450
         Neg Pred Value : 0.9041
             Prevalence : 0.6061
         Detection Rate : 0.5678
   Detection Prevalence : 0.6009
      Balanced Accuracy : 0.9265

       'Positive' Class : nonspam
```