MANAJEMEN DATA FRAME DALAM R

Objek Data dalam R

Vector	1 dimensi	Semua elemen memiliki tipe data yg sama Tipe data: numeric, character logical, factor
Matrix	2 dimensi	
Array	2 atau lebih dimensi	
Data frame	2 dimens	table-like data object allowing different data types for different columns
List	Koleksi dari objek data, setiap elemen list adalah objek data	

Manipulating Data Objects

Menciptakan	c(), rep(), seq() numeric(),character(),factor(),logical() matrix(), array(), data.frame(), list()		
Menyimpan Memanggil	save(), load() (*internal data file*)	Importing Exporting	read.table(), write.table() (*external data file*)
Converting	as.numeric(),as.character(),as.factor(),as.logical() as.matrix(), as.array(), as.data.frame(), as.list()		
Naming Indexing Selecting	names(), clonames(), rownames() [i] (*for vector*), [i, j] (*for matrix and data frame*), [i, j, k, …] (*for array*), [[k]] (*for list*) *j, k, can be integer or character or logical index*		
Combining	c(), paste(), cbind(), rbind(), merge()		
Sorting	sort(), order()		
Misc.	class(), length(), dim(), nrow(), ncol() ……		

Materi:

- Menciptakan variabel baru dalam dataframe
- Subsetting data
- Sorting data
- Recoding data
- Merging data
- Reshaping data

Menciptakan Variabel Baru dalam Data Frame

- Lakukan seperti menciptakan vektor dengan menggunakan index atau operator seleksi

  – dt$baru <- ekspresi

Subsetting Data

- Hal penting untuk Subsetting data adalah membuat vektor logical seperti yg diinginkan.
- Harus dapat menterjemahkan idea rumit ke dalam vektor logic
- Fungsi yang digunakan :

  ==, !=, >, >=, <, <=, %in%, duplicated,

  is.na, is.null, is.numeric, dll…

Ilustrasi Subsetting Data

```
> a
   gender v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
1       f  8  9  5  9  9  9  5  9 11   8
2       f 15  8  9  5 10  8 10 12  9  15
3       f  8 13 12  6  7  9 14 12 12   9
4       m 14  9  8 NA 11 10  6 11 11   8
5       m  8  2  9 16  8 10  8  9  8   9
6       f 10  6 10  9 10 10  9  7  7  11
7       m  9  9  7 13  9 12 10  9 11   7
8       m  9 10  8 10  5 10  6 12  6   8
9       f 10 13  6  7 10 12 13  9  6   9
10      m 12 10  9  8 17  7  9 10  7   7
```

Original dataset, a

Misalkan diinginkan dataset baru yang memenuhi females di atas 7 pada v1 tetapi di bawah 10 pada v10, dan males jika non-missing pada v4

```
> myindex <- (a$gender=='f' & a$v1>7 & a$v10<10) | (a$gender=='m' & !is.na(a$v4))
> a.new <- a[myindex,]
> a.new
   gender v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
1       f  8  9  5  9  9  9  5  9 11   8
3       f  8 13 12  6  7  9 14 12 12   9
5       m  8  2  9 16  8 10  8  9  8   9
7       m  9  9  7 13  9 12 10  9 11   7
8       m  9 10  8 10  5 10  6 12  6   8
9       f 10 13  6  7 10 12 13  9  6   9
10      m 12 10  9  8 17  7  9 10  7   7
```

Sorting Data

- Langkah 1 – Buat vektor numerik yang terurut dari data yang akan diurutkan
- Langkah 2 – Gunakan vektor ini sebagai index
- Fungsi yang berguna: order(), sort(), which(), rev(), unique()

Ilustrasi Sorting Data

Dataframe:

```
> a
   gender v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
1       f  7  7 12 10 12  9  7  9 13   8
2       m  7 11 14  4  4 11  8  9 11   8
3       f  9  8  8  8 12  6  8 10 13   4
4       m 10  9  7  9  4 10 12  9 10   8
5       f  8  9 11  7  9 10 11  9 15  10
6       f  5 10  8  6  9  8  8 12  3   7
7       f  7 11  9  6 11 12  9 11 11  10
8       m  7 10  6  9 13  5  9 10  9  11
9       m  7 10 10  8  6  7  8  8  8  11
10      m 12  6 13  3  7 11  8 10  8   4
```

Katakan akan diurutkan berdasarkan gender kemudian v1 jika male dan v2 apabila female secara ascending?

Ilustrasi Sorting

- Langkah 1 - Buat vektor numerik yang terurut dari data yang akan diurutkan

```
newvec <- (a$gender=='m')*a$v1 + (a$gender=='f')*a$v2
myord <- order(a$gender, newvec)
```

- Step 2 - Gunakan vektor ini sebagai index

```
a2 <- a[myord,]
```

```
> a2
   gender v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
1       f  7  7 12 10 12  9  7  9 13   8
3       f  9  8  8  8 12  6  8 10 13   4
5       f  8  9 11  7  9 10 11  9 15  10
6       f  5 10  8  6  9  8  8 12  3   7
7       f  7 11  9  6 11 12  9 11 11  10
2       m  7 11 14  4  4 11  8  9 11   8
8       m  7 10  6  9 13  5  9 10  9  11
9       m  7 10 10  8  6  7  8  8  8  11
4       m 10  9  7  9  4 10 12  9 10   8
10      m 12  6 13  3  7 11  8 10  8   4
```

Recoding Data

- Paling umum menggunakan logical:

  dta$grupumur<-**1***(dta$AGE<=30)+**2***(dta$AGE>30)

- Dapat menggunakan fungsi ifelse

  dta$grupumur<-ifelse(dta$AGE %in% 1:30,1,2)

- Menggunakan fungsi recode (ada dalam paket car):

  dta$grupumur = recode(dta$AGE,'1:30=1; else=2')

Merging Data

- Menggunakan rbind atau cbind
- Lebih mudah menggunakan fungsi merge

Ilustrasi Merging Data

```
> a3
   id1 gender v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
1  A.0      f  7  7 12 10 12  9  7  9 13   8
3  B.1      f  9  8  8  8 12  6  8 10 13   4
5  C.0      f  8  9 11  7  9 10 11  9 15  10
6  D.1      f  5 10  8  6  9  8  8 12  3   7
7  E.1      f  7 11  9  6 11 12  9 11 11  10
2  F.0      m  7 11 14  4  4 11  8  9 11   8
8  G.0      m  7 10  6  9 13  5  9 10  9  11
9  H.0      m  7 10 10  8  6  7  8  8  8  11
4  I.0      m 10  9  7  9  4 10 12  9 10   8
10 J.0      m 12  6 13  3  7 11  8 10  8   4
```

```
> a4
  1 2 id2
1 3 2 A.0
2 4 4 B.1
3 1 4 C.1
4 3 3 D.0
5 2 1 E.0
```

```
> merge(a3,a4,by.x=1,by.y=3,all=FALSE)
  id1 gender v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 1 2
1 A.0      f  7  7 12 10 12  9  7  9 13   8 3 2
2 B.1      f  9  8  8  8 12  6  8 10 13   4 4 4
```

```
> merge(a3,a4,by.x=1,by.y=3,all=TRUE)
   id1 gender v1 v2 v3 v4 v5 v6 v7 v8 v9 v10  1  2
1  A.0      f  7  7 12 10 12  9  7  9 13   8  3  2
2  B.1      f  9  8  8  8 12  6  8 10 13   4  4  4
3  C.0      f  8  9 11  7  9 10 11  9 15  10 NA NA
4  C.1   <NA> NA NA NA NA NA NA NA NA NA  NA  1  4
5  D.0   <NA> NA NA NA NA NA NA NA NA NA  NA  3  3
6  D.1      f  5 10  8  6  9  8  8 12  3   7 NA NA
7  E.0   <NA> NA NA NA NA NA NA NA NA NA  NA  2  1
8  E.1      f  7 11  9  6 11 12  9 11 11  10 NA NA
9  F.0      m  7 11 14  4  4 11  8  9 11   8 NA NA
10 G.0      m  7 10  6  9 13  5  9 10  9  11 NA NA
11 H.0      m  7 10 10  8  6  7  8  8  8  11 NA NA
12 I.0      m 10  9  7  9  4 10 12  9 10   8 NA NA
13 J.0      m 12  6 13  3  7 11  8 10  8   4 NA NA
```

Reshaping Data

- Membentuk data set baru dengan cara:
  - Long to wide data format
  - Wide to long data format
- Menggunakan fungsi reshape()

Ilustrasi: reshape(): long to wide data formats

```
> df3
   school class ggg      score t2
1       1     9   1  1.3974915  1
2       1    10   1 -0.6842393  2
3       1     9   2  0.9824653  3
4       1    10   2 -0.4250032  4
5       2     9   1  2.0248828  1
6       2    10   1 -0.9013701  2
7       2     9   2 -0.5944581  3
8       2    10   2  0.6564197  4
9       3     9   1 -0.9975693  1
10      3    10   1 -0.5178565  2
11      3     9   2  0.7315830  3
12      3    10   2  1.0372571  4
```

df3 is in ‘long’ format

```
> reshape(df3, idvar=c("school","class"), timevar='ggg',direction="wide")
   school class    score.1 t2.1    score.2 t2.2
1       1     9  1.3974915    1  0.9824653    3
2       1    10 -0.6842393    2 -0.4250032    4
5       2     9  2.0248828    1 -0.5944581    3
6       2    10 -0.9013701    2  0.6564197    4
9       3     9 -0.9975693    1  0.7315830    3
10      3    10 -0.5178565    2  1.0372571    4
```

‘wide’ format

You must use two arguments:

- idvar = 1+ variables in long format identifying rows that are the same individual
- timevar = 1 variable that differentiates multiple records from same individual

Ilustrasi: reshape(): long to wide data formats

```
> df3
   school class ggg      score t2
1       1     9   1  1.3974915  1
2       1    10   1 -0.6842393  2
3       1     9   2  0.9824653  3
4       1    10   2 -0.4250032  4
5       2     9   1  2.0248828  1
6       2    10   1 -0.9013701  2
7       2     9   2 -0.5944581  3
8       2    10   2  0.6564197  4
9       3     9   1 -0.9975693  1
10      3    10   1 -0.5178565  2
11      3     9   2  0.7315830  3
12      3    10   2  1.0372571  4
```

df3 is in 'long' format

```
> reshape(df3, idvar='school', timevar='t2',direction="wide",drop='ggg')
  school class.1    score.1 class.2    score.2 class.3    score.3 class.4    score.4
1      1       9  1.3974915      10 -0.6842393       9  0.9824653      10 -0.4250032
5      2       9  2.0248828      10 -0.9013701       9 -0.5944581      10  0.6564197
9      3       9 -0.9975693      10 -0.5178565       9  0.7315830      10  1.0372571
```

An alternative 'wide' format

- idvar = 1+ variables in long format identifying rows that are the same individual
- timevar = 1 variable that differentiates multiple records from same individual

Ilustrasi: reshape(): wide to long data formats

```
   school class    score.1 t2.1    score.2 t2.2
1       1     9  1.3974915    1  0.9824653    3
2       1    10 -0.6842393    2 -0.4250032    4
5       2     9  2.0248828    1 -0.5944581    3
6       2    10 -0.9013701    2  0.6564197    4
9       3     9 -0.9975693    1  0.7315830    3
10      3    10 -0.5178565    2  1.0372571    4
```

wd is in 'wide' format

```
> reshape(wd,varying=list(c('score.1','score.2'),c('t2.1','t2.2')),direction='long',v.names=c('new1','new2'))
    school class time       new1 new2 id
1.1      1     9    1  1.3974915    1  1
2.1      1    10    1 -0.6842393    2  2
3.1      2     9    1  2.0248828    1  3
4.1      2    10    1 -0.9013701    2  4
5.1      3     9    1 -0.9975693    1  5
6.1      3    10    1 -0.5178565    2  6
1.2      1     9    2  0.9824653    3  1
2.2      1    10    2 -0.4250032    4  2
3.2      2     9    2 -0.5944581    3  3
4.2      2    10    2  0.6564197    4  4
5.2      3     9    2  0.7315830    3  5
6.2      3    10    2  1.0372571    4  6
```

- varying = a LIST of column names in wide format that will be the same columns in the long format
- v.names = a vector of names of the new columns in the long format

TUGAS

- Lakukan merging dua data set A dan B tanpa menggunakan fungsi merge