🔁 appply

Dr. Kübra Atalay Kabasakal

apply ailesi .huge[]

- apply()
- lapply()
- sapply()
- vapply()
- mapply()
- rapply()
- tapply()

apply() Fonksiyonu

- **apply()** fonksiyonu matris ve dizilerde satır ve sütundaki değerlerin belirlenen fonksiyona göre değerlerini özetlemek için kullanılır.
- **apply()** fonksiyonunun genel kullanımı

```
apply(x, margin, FUN, …)
```

- margin argümanı satır (**1**), sütun (**2**) veya her ikisini (**c(1,2)**) FUN argümanı ise uygulanacak fonksiyonu belirtmektedir.

apply() Fonksiyonu

```r
X = matrix(c(1:9), nr= 3, byrow = T)
X
```

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

```r
#1 satirlar icin,
apply(X, 1, mean) # satir ortalamaları
```

```
[1] 2 5 8
```

```r
#2 sutunlar icin
apply(X, 2, mean) # sutun ortalamaları
```

```
[1] 4 5 6
```

apply() Fonksiyonu

- Ortalaması 50, standart sapması 5 olan normal dağılıma sahip 100 elemanlı "S1" vektöründen 20 satırlı ve 5 sütunlu matrisin oluşturulması

```
set.seed(12)
S1 <- sample(rnorm(10000, 50, 5), 100, replace=TRUE)
Matris1 <- matrix(S1, nrow=20, ncol=5)
```

- `mean()` fonksiyonunun "Matris1" nesnesinin her bir sütununa uygulanarak sütunların ortalamasının alınması

```
apply(Matris1, 2, mean) # Fonksiyonun ikinci girdisi olan 2  sütun elamanla
```

```
[1] 48.20485 52.13701 49.38658 50.61689 48.60479
```

apply() Fonksiyonu

- `summary()` fonksiyonunun "Matris1" nesnesinin her bir sütununa uygulanması

```
1 apply(Matris1, 2, summary)
```

```
           [,1]     [,2]     [,3]     [,4]     [,5]
Min.    39.00080 40.23309 39.04749 39.32974 37.74364
1st Qu. 45.21933 48.44165 45.57123 47.36401 43.71252
Median  49.31295 52.24410 49.49029 51.08794 47.62144
Mean    48.20485 52.13701 49.38658 50.61689 48.60479
3rd Qu. 52.40540 55.97719 52.70180 54.36235 53.32016
Max.    55.24910 63.33272 58.88203 59.93019 60.51715
```

apply() Fonksiyonu

- `summary()` fonksiyonunun "Matris1" nesnesinin her bir satırına uygulanması

```
1 apply(Matris1, 1, summary)
```

```
           [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
[,8]
Min.    45.82396 39.16789 51.63544 40.23309 39.04749 44.81304 39.73637
51.11418
1st Qu. 47.78055 39.32974 52.46878 43.82775 47.16408 47.46234 46.19462
51.96290
Median  48.36804 46.24689 53.43269 47.65095 49.56534 49.64774 49.12984
52.65739
Mean    50.47126 45.82933 54.50679 47.52181 48.65629 52.22224 50.10067
54.92558
3rd Qu. 54.95931 51.70256 56.11501 49.31343 52.65050 59.25790 55.94640
55.56069
Max.    55.42443 52.69959 58.88203 56.58380 54.85404 59.93019 59.49613
63.33272
           [,9]    [,10]    [,11]    [,12]    [,13]    [,14]    [,15]
[,16]
```

apply() Fonksiyonu

- kisisel tanımlı fonksiyon ile kullanılması
- Kullanıcı tanımlı fonksiyonların **apply()** fonksiyonuna uygulanması
- Yazılan **bagil_degiskenlik()** fonksiyonunun "Matris1" nesnesinin her bir sütununa uygulanarak her bir değişkenin bağıl değişkenlik katsayısının hesaplanması

```
1 bagil_degiskenlik <- function(x){
2 (sd(x)/mean(x))*100
3 }
4 apply(Matris1, 2, bagil_degiskenlik)
```

```
[1] 11.24914 10.05771 11.02709 10.59998 12.97312
```

apply() Fonksiyonu

Adsız (anonymous) fonksiyonlar ile kullanılması

```
1 apply(Matris1, 2, function(x){(sd(x)/mean(x))*100})
```

```
[1] 11.24914 10.05771 11.02709 10.59998 12.97312
```

lapply() Fonksiyonu

- **lapply()** fonksiyonu **apply()** fonksiyonunun listeler (list-apply), vektörler ve veri setleri için özelleşmiş halidir.
- Liste yapısında olduğu için satır veya sütuna ilişkin bir argüman kullanılmaz.
- **lappy()** fonksiyonu veri setlerinde kullanıldığında, sütundaki her bir değişkeni listenin elemanı olarak alır.

lapply() Fonksiyonu

`lappy()` fonksiyonuyla elde edilen çıktılar da liste şeklindedir. `lapply()` fonksiyonunun genel kullanımı

```
lapply(x, FUN, …)
```

```
# liste olusturma
(mylist <- list(a=(1:10), b=(45:77)))
```

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

$b
 [1] 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68
69
[26] 70 71 72 73 74 75 76 77
```

lapply() Fonksiyonu

```r
# liste elemalarını toplama
(resultlapply <- lapply(mylist, sum))
```

```
$a
[1] 55

$b
[1] 2013
```

```r
liste <- list(a=c(1,2,3),b=c(4,5,6))
lapply(liste, function(x){x^(3/2)})
```

```
$a
[1] 1.000000 2.828427 5.196152

$b
[1]  8.00000 11.18034 14.69694
```

sapply() Fonksiyonu

- **sapply()** fonksiyonu **lapply()** fonksiyonu gibi liste, listelerin özelleşmiş hali olan veri setleri ve vektörler üzerinde çalışır.
- **sapply()** fonksiyonunun temel amacı çıktıları basitleştirmektir.
- **lapply()** fonksiyonuyla elde edilen çıktılar liste biçimindeyken **sapply()** fonksiyonuyla elde edilen çıktılar daha çok vektör şeklindedir.

sapply() Fonksiyonu

- **sapply()** fonksiyonu **apply()** fonksiyonuyla benzer çıktılar verir ancak **sapply()** fonksiyonunda margin değerleri bulunmaz.
- `sapply()` fonksiyonunun genel kullanımı

```
1 mylist <- list(a=(1:10), b=(45:77))
2
3 resultsapply <- sapply(mylist, sum)
4 resultsapply
```

```
 a    b
55 2013
```

lapply() Fonksiyonu

```r
# lapply() fonksiyonunun liste veri türüne uygulanması
lapply(list(a=c(1,2,3)), mean) # Liste çıktısı vermektedir.
```

```
$a
[1] 2
```

lapply() Fonksiyonu

```
1 # lapply() fonksiyonunun çıktısının vektöre dönüştürülmesi
2 unlist(lapply(list(a=c(1,2,3)), mean))
```

```
a
2
```

sapply() Fonksiyonu

```r
# sapply() fonksiyonunun liste veri türüne uygulanması
sapply(list(a=c(1,2,3)), mean) # Çıktı adlandırılmış vektör şeklindedir.
```

```
a
2
```

tapply() Fonksiyonu

- **tapply()** fonksiyonun temel görevi verileri belirlenen grup veya faktör değişkenine göre özetlemektir.
- Fonksiyonda bulunan x argümanı vektör, veri seti ve liste şeklindeki nesneleri, index argümanı "x" nesnesinin altboyut, grup veya faktör değişkenini, FUN argümanı ise uygulanacak fonksiyonu belirtir.
- $tapply(x, Index, FUN, \ldots)$

tapply() Fonksiyonu

- **tapply()** liste ve veri seti yapısındaki nesnelere uygulandığında, grup veya faktör değişkenine ilişkin fonksiyon değerlerini fonksiyon türüne gore vektör ya da liste şeklinde verir.
- Eğer **tapply()** içinde kullanılan fonksiyon tek bir değer veriyorsa, çıktı vektör; birden fazla değer veriyorsa, çıktı liste yapısındadır.

tapply() Fonksiyonu

```r
isim <- c("Ali","Elif","Su","Deniz","Aras","Berk","Can","Ece","Efe","Arda")
boy <- c(160,165,170,155,167,162,169,158,160,164)
kilo <- c(55,55,57,50,48,65,58,62,45,47)
cinsiyet <- c("erkek","kadin","kadin","kadin","erkek",
"erkek","erkek","kadin","erkek","erkek")
cinsiyet <- factor(cinsiyet)
beden <- c("S","M","S","M","S","L","M","L","S","S")
beden <- factor(beden)
# tapply() fonksiyonunun liste veri yapısına uygulanması
Liste <- list(isim=isim,boy=boy,cinsiyet=cinsiyet,beden=beden,kilo=kilo)

tapply(Liste$boy, Liste$cinsiyet, sort)
```

```
$erkek
[1] 160 160 162 164 167 169

$kadin
[1] 155 158 165 170
```

tapply() Fonksiyonu

```
1 tapply(Liste$boy, Liste$cinsiyet, sort, decreasing=TRUE)
```

```
$erkek
[1] 169 167 164 162 160 160

$kadin
[1] 170 165 158 155
```

tapply() Fonksiyonu

```r
isim <- c("Ali","Elif","Su","Deniz","Aras","Berk","Can","Ece","Efe","Arda")
boy <- c(160,165,170,155,167,162,169,158,160,164)
kilo <- c(55,55,57,50,48,65,58,62,45,47)
cinsiyet <- c("erkek","kadin","kadin","kadin","erkek",
"erkek","erkek","kadin","erkek","erkek")
cinsiyet <- factor(cinsiyet)
beden <- c("S","M","S","M","S","L","M","L","S","S")
beden <- factor(beden)
#Once veri seti olusturalım
df <- data.frame(isim,boy,kilo,cinsiyet, beden)
```

tapply() Fonksiyonu

```
1 tapply(df$boy, Liste$cinsiyet, sort)
```

```
$erkek
[1] 160 160 162 164 167 169

$kadin
[1] 155 158 165 170
```

```
1 tapply(df$boy, Liste$cinsiyet, sort, decreasing=TRUE)
```

```
$erkek
[1] 169 167 164 162 160 160

$kadin
[1] 170 165 158 155
```

tapply() Fonksiyonu

```
1 tapply(df$boy, Liste$cinsiyet, mean)
```

```
   erkek    kadin
163.6667 162.0000
```

```
1 tapply(df$boy, Liste$cinsiyet, mean)
```

```
   erkek    kadin
163.6667 162.0000
```

by() Fonksiyonu

```
1 by(df$boy, Liste$cinsiyet, sort)
```

```
Liste$cinsiyet: erkek
[1] 160 160 162 164 167 169
------------------------------------------------------------
Liste$cinsiyet: kadin
[1] 155 158 165 170
```

```
1 by(df$boy, Liste$cinsiyet, sort, decreasing=TRUE)
```

```
Liste$cinsiyet: erkek
[1] 169 167 164 162 160 160
------------------------------------------------------------
Liste$cinsiyet: kadin
[1] 170 165 158 155
```

```
1 by(df$boy, Liste$cinsiyet, mean)
```

```
Liste$cinsiyet: erkek
[1] 163.6667
------------------------------------------------------------
Liste$cinsiyet: kadin
[1] 162
```

```
1 by(df$boy, Liste$cinsiyet, mean)
```

```
Liste$cinsiyet: erkek
[1] 163.6667
------------------------------------------------------------
Liste$cinsiyet: kadin
[1] 162
```

by() Fonksiyonu

```
1 by(df$boy, Liste$cinsiyet, mean)
```

```
Liste$cinsiyet: erkek
[1] 163.6667
------------------------------------------------------------
Liste$cinsiyet: kadin
[1] 162
```

```
1 by(df$boy, Liste$cinsiyet, mean)
```

```
Liste$cinsiyet: erkek
[1] 163.6667
------------------------------------------------------------
Liste$cinsiyet: kadin
[1] 162
```