

R Ders Notları

Dr. Busenur Kızılaslan

2020-11-07

Contents

Chapter 1

Ön Bilgi

R ders notları temelde *BSP2043 - Bilgisayar III* ve *BSP2044 - Bilgisayar IV* derslerine kaynaklık etmesi amacıyla tasarlanmış olup konu çerçevesinde kendisini geliştirmek isteyen herkesin faydalanabilmesi hedeflenmiştir.

Kaynak, temel matematik ve ingilizce bilgisine sahip olan herkesin anlayıp uygulayabileceği basitlikte bir anlatıma sahiptir.

Ders dili Türkçe'dir. Bu bakımdan genel anlatımda Türkçe kullanılacaktır. Literatürü rahat takip edebilmeniz, komut karmaşası yaşamamanız ve araştırma sürecinde zengin forum imkanından yararlanabilmeniz adına temel terimler orijinal haliyle kullanılacaktır.

Aynı zamanda referans verilen kaynaklar da orijinal dili ile paylaşılacaktır. Bu nedenle İngilizce eksikliğinizi gidermeniz önerilir.

Kaynaklar;

- Kitaplar

The R Book - Michael J. Crawley

R for Data Science - Hadley Wickham, Garrett Grolemund

Introduction to Data Science - Rafael A. Irizarry

R Programming for Data Science - Roger D. Peng

The Book of R - Tilman M. Davies

- Eğitimler

HarvardX - Data Science: R Basics

HarvardX - Data Science: Visualization

HarvardX - Data Science: Probability

Katkı ve öneriler için: busenur.sarica@marmara.edu.tr

İyi eğlenceler!

1.1 Hakkımda

2012 yılında Mimar Sinan G. S. Üniversitesi istatistik bölümünde lisans öğrenimimi tamamlamamın ardından 2014 yılı itibariyle Marmara Üniversitesi istatistik bölümünde araştırma görevlisi olarak göreve başladım. Eş zamanlı olarak başladığım yüksek lisans öğrenimimi yine Mimar Sinan G. S. Üniversitesi istatistik anabilim dalında tamamlarken artık çalışma alanım bulanık mantık (fuzzy logic) olarak şekillenmişti. Bir buçuk yılda tamamladığım yüksek lisansın ardından farklı bakış açısı kazanma fikriyle yönümü Yıldız Teknik Üniversitesi'ne çevirdim. 2015 yılında istatistik bölümünde başladığım doktora öğrenimimi 2020 yılında yine bulanık mantık ile öngörü üzerine hazırladığım tezim ile tamamladım. Doktora öğrenimim sırasında İstanbul Teknik Üniversitesi endüstri mühendisliği bölümünden de çeşitli dersler alarak alanında uzman hocaların¹ bilgilerinden faydalanma imkanı buldum.

Akademik kariyer hedefiyle başlamadığım öğrenim yaşamımda ufku açık, aydınlık ve birikimli hocalarım yolumu bulmama yardımcı olmuştur. 2018 yılında Polonya'da gerçekleşen International Conference on Trends and Perspectives in Linear Statistical Inference (LinStat'2018) kapsamında sunduğum, doktora tezimin bir kısmından oluşan çalışma ile aldığım Young Scientists Awards ikincilik ödülü de yanlış yolda olmadığımı göstermiştir.

Üzerimde emeği olan herkese teşekkürlerimle!

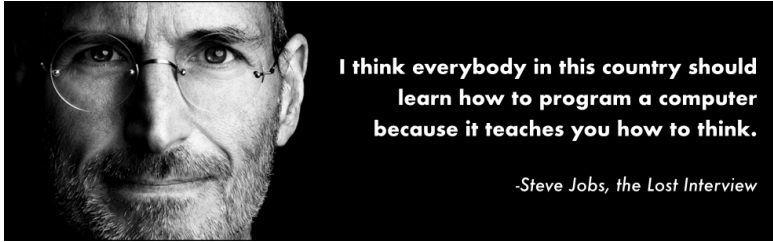
CV, Google Scholar, LinkedIn

¹Prof. Dr. Cengiz Kahraman

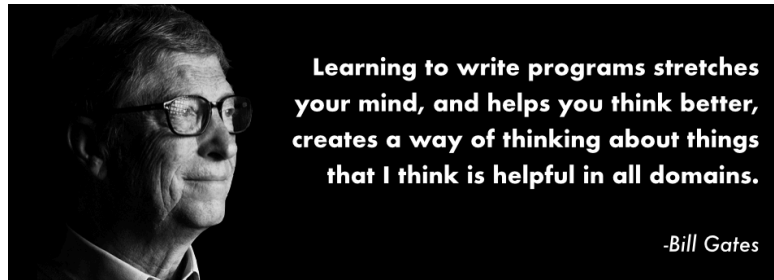
Chapter 2

Motivasyon

‘Bu dersi neden alıyorum?’ sorusuna karşılık olarak alacağınız cevap **‘ilerleyen dönemlerde göreceğiniz derslerin uygulamalarında ihtiyaç duyacaksınız’**dan çok daha fazlası! Herhangi bir programlama diline hakim olmak veriyi anlamlandırabilmek adına zaten önemliyken özellikle R, Python gibi açık kaynak ve hızla gelişmeye devam eden programlama dillerini biliyor olmak sizleri donanımlı kılacaktır.



Programlama bilgisine sahip olmak iş yaşamındaki ihtiyacınızı karşılamasının yanı sıra günlük yaşamdaki problem çözme yeteneğinizin gelişmesine de yardımcı olmaktadır. Programlama yapısını öğrenen kişinin diğer programlama dillerini öğrenmesi kolaylaşmaktadır. İş başvurularında birden fazla programlama dili biliyor olmanın sizi öne çıkaracağı da aşikar. Burada sorulması gereken asıl soru **‘hangi programlama dilini/dillerini öğrenmeliyim?’** olmalıdır. Ders kapsamında R programlama dili ve özellikleri açıklanacaktır.



Chapter 3

Genel Bakış

3.1 R: Nedir?

İstatistiksel analiz ve veri görselleştirme amacıyla geliştirilen R, açık kaynak olup ücretsizdir. R, dizayn olarak var olan iki dilden etkilenmiştir; Becker, Chambers & Wilks'in S programlama dili ve Sussman'ın Scheme programlama dilidir. R, başlangıçta Yeni Zelanda Auckland'daki Auckland Üniversitesi İstatistik Bölümü'nde Ross Ihaka ve Robert Gentleman tarafından yazılmıştır. Ek olarak, büyük bir grup insan, kod ve hata raporları göndererek katkıda bulunmuştur. Tarihçeyi merak edenler *A Brief History of S* (?) kaynağını inceleyebilir.

1997 ortalarından itibaren çekirdek bir yapı (*The R Core Team*) yönetimi sürdürmektedir.

Resmi internet adresi

Yükleme adresi

The R Journal: R kullanıcıları için hakemli ve açık erişimli dergi

Kitaplar: R kullanıcıları için çeşitli alanlarda/dillerde kaynak kitaplar

Faydalı adres1

Faydalı adres2

Hangi komutun ne işe yaradığını anımsayamıyorsanız veya bir hata ile karşılaştıysanız GOOGLE kullanın.

Programlama öğrenmenin en iyi yolu denemek ve hata yapmaktır.

3.1.1 RStudio

R üzerinde doğrudan çalışabilir veya bir grafik ara yüzü olan RStudio'nun zengin özelliklerinden faydalanma imkanından yararlanabilirsiniz. Uygulama kolaylığı sağlayan bir entegre geliştirme ortamı (*integrated development environment (IDE)*) olan RStudio, Windows, Mac ve Linux ile çalışabilir. RStudio'yu kullanışlı kılan birçok özellik mevcuttur, bunlardan birkaçı şu şekilde sıralanabilir.

- Script
 - Kodlama geçmişi, güçlü grafiksel altyapı
 - Cheatsheetler
 - Değişken ve fonksiyon tamamlama özelliği
-

3.2 R: Neden?

Veri analizi için kullanılabilecek SAS, SPSS, Excel, MATLAB gibi birçok yazılım mevcutken neden R kullanıyoruz?

- Ücretsiz (open source)
- Geniş kullanım kitlesi
 - Dünyada 2 milyondan fazla kullanıcıya sahip
 - Sürekli gelişmeye devam eden yapısı
 - Geniş forum ağı
- Uygulama ve kullanım kolaylığı
- Grafik ve görsel üretimindeki başarısı
- Paket kullanım imkanı
- Raporlama kolaylığı ve RMarkdown sayesinde kolay sunum

R aynı zamanda bazı dezavantajlara da sahiptir.

- Güncelleneme gereksinimi
 - Uzmanlaşmanın diğer programlara göre biraz daha zor olması
-

3.3 R: Help

R'da karşılaşacağınız problemler için menüde yer alan **Help** kısmını kullanabilir veya Stackoverflow gibi forumlardan faydalanabilirsiniz. Windows, Mac işletim sistemleri ve genel sorular için üç farklı FAQ (Frequently Asked Questions) kısmı mevcuttur.

Kullanacağınız fonksiyonun ismini biliyorsanız ? kullanarak yine help içeriğinden faydalanabilirsiniz.

```
?sum
```

Kullanacağınız fonksiyonun ismini biliyor fakat hangi pakette yer aldığını bilmiyorsanız bu noktada **find** komutu size yardımcı olacaktır.

```
find("sum")
```

```
## [1] "package:base"
```

Kullanacağınız fonksiyon ile ilgili örnek araştırmak isterseniz **example** komutu işinizi görecektir.

```
example(sum)
```

```
##
## sum> ## Pass a vector to sum, and it will add the elements together.
## sum> sum(1:5)
## [1] 15
##
## sum> ## Pass several numbers to sum, and it also adds the elements.
## sum> sum(1, 2, 3, 4, 5)
## [1] 15
##
## sum> ## In fact, you can pass vectors into several arguments, and everything gets added.
## sum> sum(1:2, 3:5)
## [1] 15
##
## sum> ## If there are missing values, the sum is unknown, i.e., also missing, ....
## sum> sum(1:5, NA)
## [1] NA
##
## sum> ## ... unless we exclude missing values explicitly:
## sum> sum(1:5, NA, na.rm = TRUE)
## [1] 15
```

3.4 Paketler

R programlama dili ile kendi döngülerinizi oluşturabilir, özgün kodlarınızı yazabilirsiniz, aynı zamanda paketler yardımı ile yazılmış olan zengin hazır kod içeriğinden de faydalanabilirsiniz (internet bağlantısı gerektirir). Bu zengin içeriğe hazırladığınız paketler ile dahil olma imkanınız da mevcut.

Paket indirme konusunda problem yaşıyorsanız R'ı yönetici olarak çalıştırmayı deneyin.

Paket yükleme işleminin hızlı olabilmesi için size en yakın mirror seçimini yapmalısınız.

Paketler kullanım kolaylığı ve zamandan kazanç sağlamakla birlikte kullanılmadan önce içeriğin dikkatlice incelenmesi önemlidir. Tanımlamaları iyi anlaşılmadan kullanılan paketlerle yanlış sonuçlar elde edilmesi kaçınılmazdır.

Analyse & Explore



The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying philosophy and common APIs.

[Project Site Link >](#)



ggplot2 is an enhanced data visualization package for R. Create stunning multi-layered graphics with ease.

[Project Site Link >](#)



dplyr is the next iteration of plyr, focussing on only data frames. dplyr is faster and has a more consistent API.

[Project GitHub Link >](#)



tidyr makes it easy to “tidy” your data. Tidy data is data that’s easy to work with: it’s easy to munge (with dplyr), visualise (with ggplot2 or ggvis) and model (with R’s hundreds of modelling packages).

[Project Paper Link >](#)



purrr enhances R’s functional programming (FP) toolkit by providing a complete and consistent set of tools for working with functions and vectors.

[Project Site Link >](#)



A consistent, simple and easy-to-use set of wrappers around the fantastic ‘stringi’ package.

[Project Site Link >](#)

Communicate & Interact



Shiny makes it incredibly easy to build interactive web applications with R. Shiny has automatic “reactive” binding between inputs and outputs and extensive pre-built widgets.

[Project Site Link >](#)



rmarkdown lets you insert R code into a markdown document. R then generates a final document, in a wide variety of formats, that replaces the R code with its results.

[Project Site Link >](#)



Use flexdashboard to publish groups of related data visualizations as a dashboard.

[Project Site Link >](#)

Model & Predict



TensorFlow™ is an open-source software library for Machine Intelligence. The R interface to TensorFlow lets you work productively using the high-level Keras and Estimator APIs and the core TensorFlow API.

[Project Site Link >](#)



The tidymodels framework is a collection of packages for modeling and machine learning using tidyverse principles.

[Project Site Link >](#)



Sparklyr provides bindings to Spark's distributed machine learning library. Together with sparklyr's dplyr interface, you can easily create and tune machine learning workflows on Spark, orchestrated entirely within R.

[Project Site Link >](#)

Connect & Integrate



Sparklyr is an R interface to Apache Spark, a fast and general engine for big data processing. This package connects to local and remote Apache Spark clusters, a 'dplyr' compatible back-end, and an interface to Spark's ML algorithms.

[Project Site Link >](#)



Plumber enables you to convert your existing R code into web APIs by merely adding a couple of special comments.

[Project Site Link >](#)



The reticulate package provides a comprehensive set of tools for interoperability between Python and R.

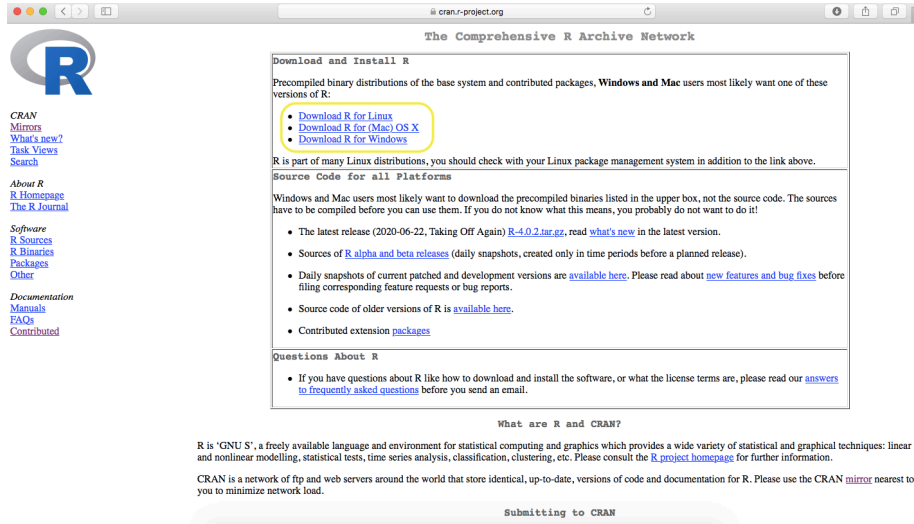
[Project Site Link >](#)

Chapter 4

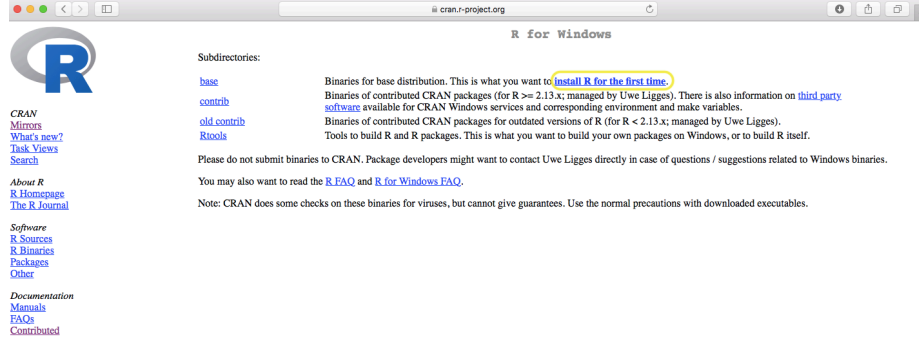
Yükleme ve Tanışma

4.1 Yükleme

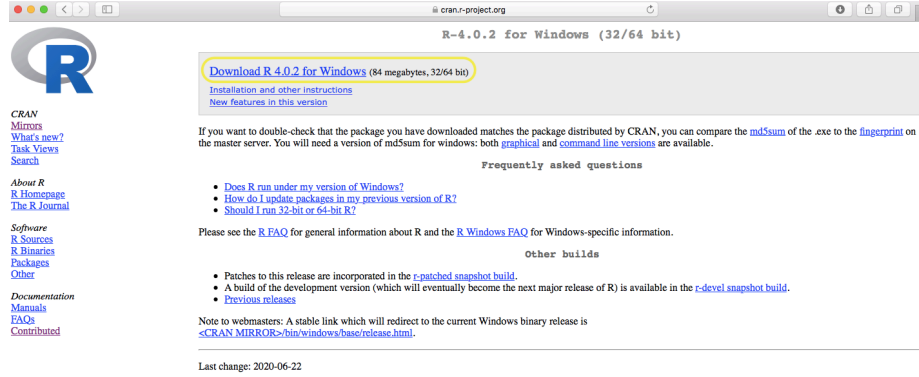
R'in güncel versiyonun indirilmesi için CRAN (The Comprehensive R Archive Network) sayfası ziyaret edilmelidir. Bilgisayarımızdaki mevcut işletim sistemi için uygun olan versiyon indirilmelidir.



Uygun işletim sistemi seçildikten sonra `install R for the first time` tıklayarak temel versiyon seçilmelidir.



Gelen ekranda seçtiğiniz işletim sistemine uygun olan güncel R versiyonu görülecektir. **Download** işlemi başlatılır ve uygun seçenekler dahilinde indirme işlemi ve kurulum tamamlanır.



R yükleme işlemi tamamlandıktan sonra RStudio kurulum işlemine başlanır. RStudio sayfasına gidilir, bilgisayarınızdaki işletim sistemine uygun olan RStudio Desktop versiyonu indirilir ve kurulur.

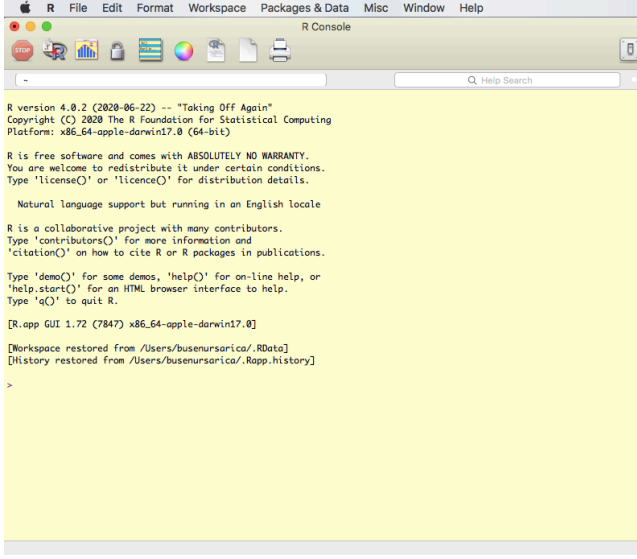
Yükleme destek videosu (R-Windows)

Yükleme destek videosu (R-Mac)

Yükleme destek videosu (RStudio)

4.2 Tanışma

R programlama dilini kullanmak için console erişimi yeterlidir. Console yalnız haliyle kullanılabildiği gibi RStudio aracılığı ile de kullanılabilir.



```
R version 4.0.2 (2020-06-22) -- "Taking Off Again"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin17.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

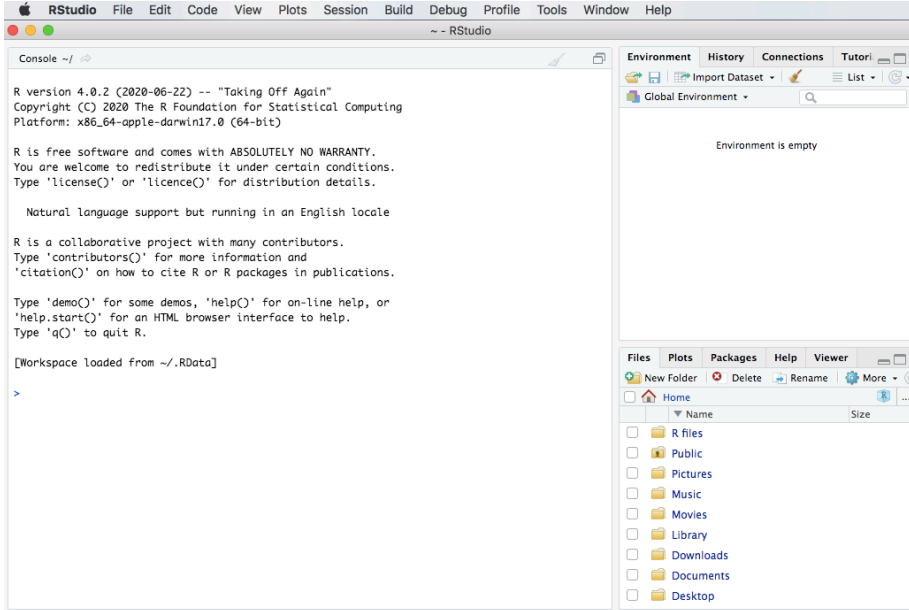
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[R.app GUI 1.72 (7847) x86_64-apple-darwin17.0]

[Workspace restored from /Users/busenursarica/.RData]
[History restored from /Users/busenursarica/.Rapp.history]

>
```

RStudio editor, görsel ve uygulama avantajları sayesinde uygulama kolaylığı sağlamaktadır.

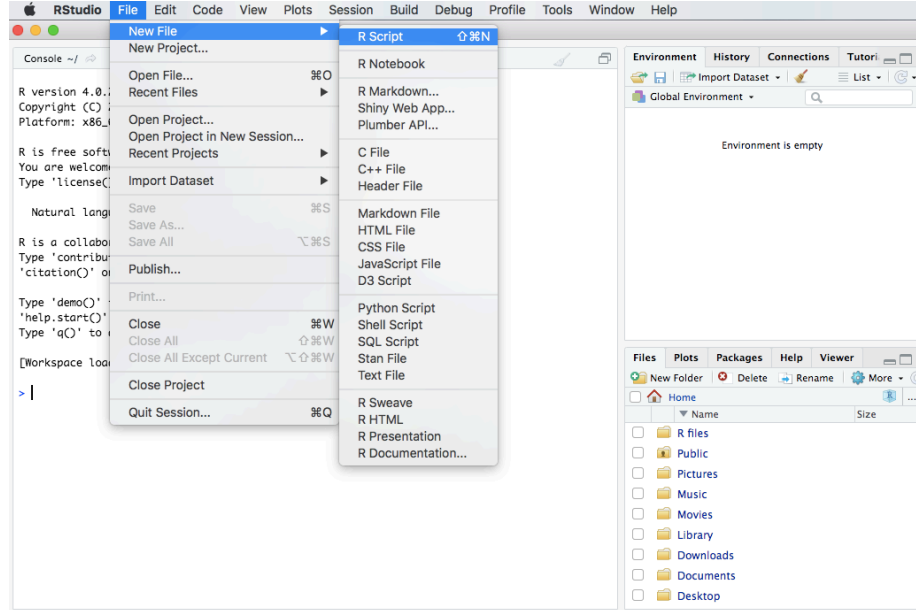


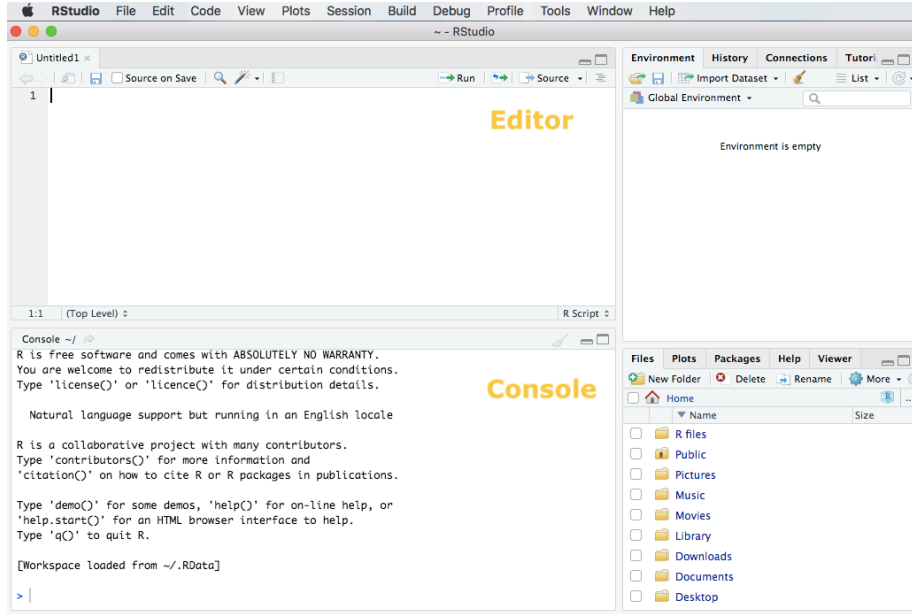
R kullanarak yaptığınız çalışmalarda lütfen referans vermeyi unutmayınız.

```
citation()
```

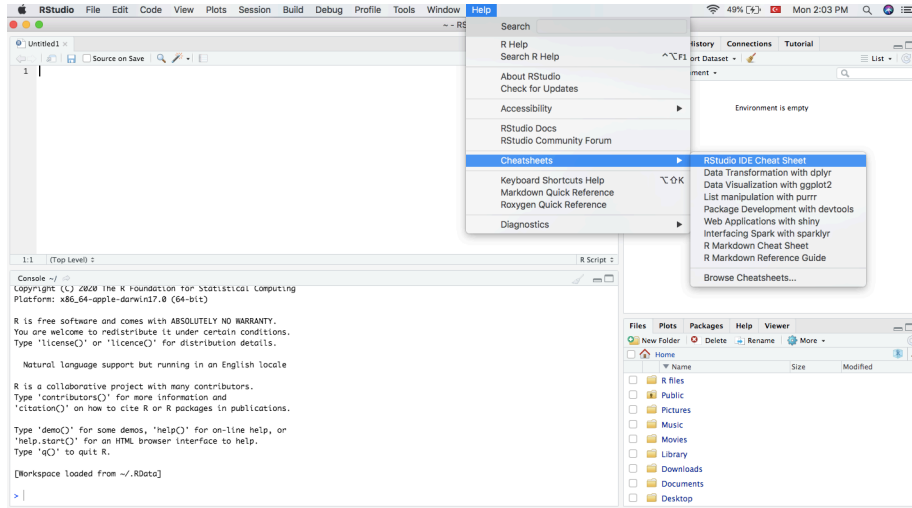
```
##
## To cite R in publications use:
##
## R Core Team (2020). R: A language and environment for statistical
## computing. R Foundation for Statistical Computing, Vienna, Austria.
## URL https://www.R-project.org/.
##
## A BibTeX entry for LaTeX users is
##
## @Manual{,
##   title = {R: A Language and Environment for Statistical Computing},
##   author = {{R Core Team}},
##   organization = {R Foundation for Statistical Computing},
##   address = {Vienna, Austria},
##   year = {2020},
##   url = {https://www.R-project.org/},
## }
##
## We have invested a lot of time and effort in creating R, please cite it
## when using it for data analysis. See also 'citation("pkgname")' for
## citing R packages.
```

Yeni script oluşturmak için;





RStudio cheatsheet yapısı sayesinde bir çok konu başlığında özet bilgiye ulaşmak mümkün.



Chapter 5

Temel Nesneler

Bu bölümde kodlama için ihtiyaç duyacağınız temel yapılar açıklanacak ve uygulamalar ile desteklenecektir. **Farklı uygulamalar ders esnasında eş zamanlı yapılacağından lütfen online dersleri takip ediniz.**

5.1 Aritmetik (Arithmetic)

R, en basit haliyle hesap makinesi olarak kullanılabilir. Toplama +, çıkarma -, çarpma *, bölme / operatörleri ile gerçekleştirilir.

```
5+4
```

```
## [1] 9
```

Birden fazla matematiksel işlem aynı satırda gerçekleştirilebilir.

```
3+4; 6*4; 10-2
```

```
## [1] 7
```

```
## [1] 24
```

```
## [1] 8
```

İşlemler parantez yardımıyla önceliğine göre yazılabilir, yazılmadığı takdirde matematiksel işlem önceliği geçerlidir.

```
10*2-3
```

```
## [1] 17
```

İşlem devam edecek biçimde tanımlanırsa console + simgesi ile devam edecek ve işlem tamamlanana kadar yeni işleme geçmenize engel olacaktır. İşlemi tamamlamalı veya yeni işleme geçmek için **esc** tuşunu kullanmalısınız.

```
10+20+30+
40
```

```
## [1] 100
```

Yapılan işlemler sonucu elde edilen çok büyük veya çok küçük sonuçlar için output exponent olarak verilir.

```
12000*3000
```

```
## [1] 3.6e+07
```

1.3e2 (130 anlamına gelir. e2: ondalık noktasını iki basamak sağa taşı)

1.4e-1 (0.14 anlamına gelir. e-1: ondalık noktasını bir basamak sola taşı)

Uygulamada elde edilen sonucun integer (tamsayı) olması gerekebilir. Bu noktada elde edilen output üste, alta veya 0.5 üzeri ya da altı olma durumuna göre farklı komutlar yardımı ile yuvarlanabilir.

- **floor:** alta yuvarla

```
floor(5.2); floor(5.7)
```

```
## [1] 5
```

```
## [1] 5
```

- **ceiling:** üste yuvarla

```
ceiling(3.2); ceiling(3.8)
```

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

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

- **round:** 0.5 üzeri ise üste, 0.5 altı ise alta yuvarla

```
round(5.6); round(5.3)
```

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

```
## [1] 5
```

Negatif sayılarda komutların nasıl işlediğini inceleyebilirsiniz.

round komutu ile virgülden sonra kaç basamak olması gerektiğini belirterek yuvarlama işlemi yapabilirsiniz.

```
round(1.248,2)
```

```
## [1] 1.25
```

Kullanılabilecek matematiksel fonksiyonlara örnek olarak (?)

5.2 Nesneleri Tanımlama (Assigning Objects)

Temel işlevlerden bir diğeri kullanılacak değişkenlerin tanımlanmasıdır. Değişken için seçilecek isim mümkün olan en kısa haliyle tanımlanarak kavram kargaşası önlenmelidir.

- R, büyük ve küçük harfe duyarlıdır, dolayısıyla tanımlanan B ve b iki farklı değişkeni temsil eder.
- Değişken ismi iki veya daha fazla kelimedenden oluşacaksa kelimeler arasında boşluk yerine nokta kullanılmalıdır. (~~neura-link~~)
- Değişken ismi sayı veya sembol ile başlayamaz. (~~1a~~, ~~&b~~)

Değişken tanımlama işlemi `<-` operatörü ile gerçekleştirilir. Tanımlanan değişken adı ile çağrılmazsa veya `print` komutu kullanılmazsa çıktı yazdırılmaz.

```
x<-3
print(x)
```

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

Sayısal olmayan değer tanımlamaları tırnak içerisinde yapılmalıdır.

```
msg<- "hello world"
```

Tanımlanan değişken veya fonksiyon ile ilgili notlar `#` ile tanımlanır.

```
x.ort<-20 # ortalama değer
```

Çıktıda basılan `[.]` kaçınıcı gözlemden devam edildiğini gösterir. Örneğin 30 gözleminin `[26]` ifadesinin yardımı ile 26. gözlem olduğunu kolaylıkla söyleyebiliriz. `[.]` ifadeleri asıl seride yer almaz, yalnızca yol gösterici olarak çıktıda gözlenir.

```
x<-5:50
x
```

```
## [1] 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
## [26] 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
```

5.3 Vektörler (Vectors)

Vektör oluşturmak için `c()` operatör kullanılmaktadır. Vektörler

- numeric
- character
- logical
- integer
- complex

yapıları içerebilir. **Vektörler yalnızca aynı yapıda gözlemler içerebilir.**

```
x <- c(0.5, 0.6) # numeric
x <- c(TRUE, FALSE) # logical
x <- c(T, F) # logical
x <- c("a", "b", "c") # character
x <- 9:29 # integer
x <- c(1+0i, 2+4i) # complex
```

T ve F, TRUE ve FALSE'a karşılık kullanılan kısaltma yapılardır.

```
x <- c(T, F) # logical
x
```

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

Aynı zamanda `vector` komutu ile de vektör oluşturabilirsiniz. Vektörü tanımlarken belirlenen içerik yapısına göre oluşturulur.

```
x <- vector("numeric", length = 7)
x
```

```
## [1] 0 0 0 0 0 0 0
```

Complex elemanları içerecek bir vektör oluşturmak istendiğinde;

```
x <- vector("complex", length = 7)
x
```

```
## [1] 0+0i 0+0i 0+0i 0+0i 0+0i 0+0i 0+0i
```

Aynı değişken adı birden fazla tanımlamada kullanılırsa yapılan son tanımlama geçerli olacaktır. Kod yazarken kullandığınız değişken isimlerine ve doğru yazıma dikkat ediniz.

Vektör uzunluğu `length()` komutu ile sorgulanır.

```
x<-c(1:68)
length(x)
```

```
## [1] 68
```

Vektör aynı yapıda gözlemlerden oluşmuyorsa?

Bu durumda tüm gözlemler tek bir yapı olarak algılanır. Herhangi bir değişkenin hangi yapıda gözlem içerdiği `class()` komutu ile sorgulanabilir.


```
y <- c(1.7, "a") # character
class(y)
```

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

```
y <- c(TRUE, 2) # numeric
class(y)
```

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

```
y <- c("a", TRUE) # character
class(y)
```

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

Vektör farklı yapıda gözlemler için verimli kullanılamıyor olabilir ancak bu işlemi gerçekleştirebilen `list` komutu mevcuttur. İlerleyen başlıklarda bu komut detaylandırılacaktır.

Kodlama yaparken sıklıkla kullanılan bir işlem türü de vektör yapısının değiştirilmesidir. Vektör içeriğinin aynı yapıda olması kuralına sadık kalarak tüm vektör içeriği farklı bir yapıya aktarılabilir. Burada `as.numeric`, `as.logical` gibi komutlardan faydalanılır.

```
x <- 0:6
class(x)
```

```
## [1] "integer"
```

`x` vektörünün `integer` yapıda olduğunu gördükten sonra `as.character` komutu ile yeni `x` vektörünü `character` olarak tanımlayabiliriz.

```
x<-as.character(x)
x; class(x)
```

```
## [1] "0" "1" "2" "3" "4" "5" "6"
```

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

Bazı durumlarda R dönüşüm için çözüm üretemez ve `NA` çıktı verir.

```
x <- c("a", "b", "c")
as.numeric(x)
```

```
## Warning: NAs introduced by coercion
```

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

R, eksik gözlemleri **NA (non available)** olarak tanımlar. İmkansız değerleri ise **NaN (not a number)** olarak tanımlar.

Sık kullanılan komutlardan bir diğeri `seq()`, bu fonksiyon sayesinde istediğiniz aralıkta ve artış seviyesinde vektör üretebilirsiniz.

```
x<-1:20  #1'den 20'ye 1'er artan seri
x

## [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
x<-seq(from=5,to=20,by=5) #5'den 20'ye 5'er artan seri
x

## [1]  5 10 15 20
```

Serinin bitiş noktasını belirtmeden de tanımlama yapmak mümkündür, bu durumda serinin uzunluğunun ne olduğun fonksiyonda belirtilmelidir.

```
x<-seq(from=0.04,by=0.01,length=11)
x

## [1] 0.04 0.05 0.06 0.07 0.08 0.09 0.10 0.11 0.12 0.13 0.14
```

Gözlemleri sıralamak için `sort()` komutu kullanılır. `sort()` kodu default olarak küçükten büyüğe sıralama yapar, sıralama yönünü değiştirme işlemi `decreasing` ile gerçekleştirilir.

```
y <- c(4,2,0,9,5,3,10)
sort(y)

## [1]  0  2  3  4  5  9 10
sort(y, decreasing=TRUE) # büyükten küçüğe sıralama

## [1] 10  9  5  4  3  2  0
```

Tekrarlı oralarak işlem yapmak için kullanılan fonksiyon `rep()` şeklinde tanımlanmaktadır.

```
rep(y,times=3)

## [1]  4  2  0  9  5  3 10  4  2  0  9  5  3 10  4  2  0  9  5  3 10
```

Vektörler için kullanışlı fonksiyonlardan bazıları aşağıdaki listede sizlerle paylaşılmıştır (?).

5.4 Matrisler (Matrices) ve Diziler (Arrays)

Matrisler, boyut niteliğine sahip vektörlerdir. Matris yapısında *satır* (*row*) ve *sütun* (*column*) ($r * c$) olmak üzere iki boyut mevcuttur. Matris içeriği de vektörde olduğu gibi tek tip yapıdan oluşmalıdır. m içeriği boş bir matris olmak üzere;

```
m <- matrix(nrow = 2, ncol = 3)
m
```

```
##      [,1] [,2] [,3]
## [1,]   NA   NA   NA
## [2,]   NA   NA   NA
```

Matris boyutu `dim()` komutu ile sorgulanır.

```
dim(m)
```

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

Matris yapısında gözlemler sütun şeklinde sıralanır.

```
m <- matrix(1:6, nrow = 2, ncol = 3)
m
```

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

Elemanların aynı satırdan devam ettiği matris üretmek isteniyorsa `byrow=TRUE` bilgisi eklenmelidir.

```
n<-matrix(1:6,2,3,byrow = TRUE); n
```

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

Vektörler parçalanarak da matris yapısı oluşturabilirler.

```
m <- 1:10 ;m
```

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

```
dim(m) <- c(2, 5) ;m
```

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

Matrisler, satır veya sütunların birleştirilmesi yoluyla da oluşturulabilir. Satırların bir araya getirilmesi için `rbind` komutu kullanılırken, sütunların bir araya getirilmesi için `cbind` komutu kullanılmaktadır.

```
x <- 1:3
y <- 10:12
cbind(x, y)
```

```
##      x  y
## [1,] 1 10
```

```
## [2,] 2 11
## [3,] 3 12
```

```
rbind(x, y)
```

```
##      [,1] [,2] [,3]
## x      1    2    3
## y     10   11   12
```

- `[,4]` anlamı ilgili değişkenin 4. sütunu tüm satırları
- `[2,]` anlamı ilgili değişkenin 2. satırı tüm sütunları

Mevcut yapıyı sorgulamak ve değiştirmek de mümkündür.

```
x<-c(1,2,3,4,5,6,7,8,9);x
```

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

```
is.vector(x)
```

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

```
y<-as.matrix(x);y
```

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

```
class(y)
```

```
## [1] "matrix" "array"
```

Matrisin transpozu `t()` fonksiyonu ile alınır.

```
m1<-matrix(c(1:8),4,2);m1
```

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

```
t(m1)
```

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

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

Matrislerde satır ve sütunlar için isim tanımlama işlemi satır veya sütun tanımlamasına bağlı olarak sırasıyla `rownames()` ve `colnames()` fonksiyonları ile gerçekleştirilir.

```
rownames(m1)<-c("x1","x2","x3","x4")
colnames(m1)<-c("y1","y2"); m1
```

```
##      y1 y2
## x1  1  5
## x2  2  6
## x3  3  7
## x4  4  8
```

Matrislerde işlem kolaylığı sağlamak adına bazı fonksiyonlar tanımlanmıştır. Satır toplam veya ortalama, sütun toplam veya ortalama işlemleri için tanımlı fonksiyonlar aşağıda yer almaktadır.

rowSums(): satır toplam

colSums(): sütun toplam

rowMeans(): satır ortalama

colMeans(): sütun ortalama

```
rowSums(m1)
```

```
## x1 x2 x3 x4
##  6  8 10 12
```

```
colMeans(m1)
```

```
##      y1 y2
## 2.5 6.5
```

Matrislerde çarpma işlemi `%*` operatörü ile gerçekleştirilir.

```
c1<-matrix(c(1,2,3,4),2,2)
c2<-matrix(c(1,3,1,2),2,2); c1; c2
```

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

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

```
c1*c2
```

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

```
## [1,] 1 3
## [2,] 6 8
c1%*%c2
```

```
##      [,1] [,2]
## [1,] 10 7
## [2,] 14 10
```

Herhangi bir matrisin tersini alma işlemi `solve()` komutu ile gerçekleştirilir.

Diziler matrislerdeki satır ve sütune ek olarak bir boyut (h) daha içerir. Birden fazla matrisin yer aldığı $r * c * h$ boyutlu bir yapı olarak düşünülebilir. Dizi içeriği aynı tipte verilerden oluşmalıdır.

```
array(1:12, dim = c(2, 2, 3))
```

```
## , , 1
##
##      [,1] [,2]
## [1,] 1 3
## [2,] 2 4
##
## , , 2
##
##      [,1] [,2]
## [1,] 5 7
## [2,] 6 8
##
## , , 3
##
##      [,1] [,2]
## [1,] 9 11
## [2,] 10 12
```

5.5 Listeler ve Data Frameler (Lists and Data Frames)

Listeler vektörlerin özel bir halidir. Vektörler içeriğinde aynı yapıda eleman bulundurma koşuluna sahipken, listeler için böyle bir koşul yoktur. Özetle, listeler farklı yapıda ve boyutta veri tiplerini içerebilir.

Liste oluşturmak için ihtiyaç duyulacak fonksiyon `list()` dir.

```
x <- list(c(1,2,3), "istatistik", TRUE, 1 + 4i);x
```

```
## [[1]]
## [1] 1 2 3
##
## [[2]]
## [1] "istatistik"
##
## [[3]]
## [1] TRUE
##
## [[4]]
## [1] 1+4i
```

Boş bir liste oluşturmak için `vector()` fonksiyonundan faydalanabilirsiniz.

```
x <- vector("list", length = 3);x
```

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

Data framer de iki boyutlu yapılardır. Sütun içerisinde veri tipi aynı olmalıdır ancak sütunlar arası veri tipi farklılık gösterebilir. Regresyon ve farklı istatistiksel hesaplamalarda data framer sıklıkla kullanılır. Aynı zamanda `as.data.frame()` fonksiyonu yardımıyla matrisler data framelere dönüştürülebilir. Data frame yapısında yer alan değişkenlerin boyutlarının aynı büyüklükte olması gereklidir.

```
x<-data.frame(Gün=c("Pzts","Salı","Çarş","Perş", "Cuma"), Vaka=c(1000,1110,1125,1153,1196));x
```

```
##      Gün Vaka
## 1 Pzts 1000
## 2 Salı 1110
## 3 Çarş 1125
## 4 Perş 1153
## 5 Cuma 1196
```

`data` olarak isimlendirilen data frame yapısındaki veri ile ilgili farklı seçim işlemleri aşağıdaki tabloda belirtilmiştir.

5.6 Sayısal Olmayan Değerler (Non-Numeric Values)

Programlama yaparken sayısal verilerin yanında sayısal olmayan veri tipleri de kullanılmaktadır. Sayısal olmayan veri tipleri;

- Mantık (logical)
- Karakter (character)
- Faktör (factor)

olmak üzere üç başlık altında toplanmaktadır.

5.6.1 Mantıksal Değerler

Mantıksal değerler **TRUE** veya **FALSE** olarak tanımlanabilir. Mantıksal değerler fonksiyonların içinde de sıklıkla kullanılmaktadır. Örneğin **sort** fonksiyonunda sıralamanın artan veya azalan olmasını belirleyen `decreasing=FALSE`, ya da matriste eleman dizilimini belirleyen `byrow=TRUE` gibi. **TRUE** ve **FALSE** kısaltması olarak **T** ve **F** de kullanılabilir.

```
p<-c(T,F,T,T,T,F,T,F);p
```

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

Operatör	Anlamı
==	Eşittir
!=	Eşit değildir
>	Büyüktür
<	Küçüktür
>=	Büyük eşittir
<=	Küçük eşittir

```
1==2
```

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

```
1>2
```

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

```
1!=(2+5)
```

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

```
h<-c(3,2,1,4,1,2,1,-1,0,3)
```

```
m<-c(4,1,2,1,1,0,0,3,0,4)
```

```
length(h)==length(m)
```



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

```
h==m
```

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

İki mantıksal değeri karşılaştırmak için kullanılan operatörler ve sonuçlar

```
x<-c(T,F,F,T)
```

```
y<-c(F,F,T,T)
```

```
x&y
```

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

```
x&&y
```

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

```
x|y
```

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

```
x||y
```

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

Mantıksal değerler ikili yapıları gereği 1 TRUE ve 0 FALSE olarak tanımlanır.

```
TRUE+TRUE
```

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

```
T+T+F+T+T+T+F
```

```
## [1] 5
```

```
1&&1
```

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

```
1||0
```

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

Benzer mantık işlemleri karakter veriler için de uygulanabilir.

```
"alpha"=="alpha"
```

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

```
"alpha"!="beta"
```

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

5.6.2 Karakterler

Diğer bir yaygın kullanılan veri tipi de karakterlerdir.

```
x<-"OpenAI, kâr amacı gütmeyen bir yapay zeka araştırma şirketidir."
x; length(x)
```

```
## [1] "OpenAI, kâr amacı gütmeyen bir yapay zeka araştırma şirketidir."
## [1] 1
```

R, dizeyi tek bir varlık olarak ele alır. Diğer bir deyişle, x, 1 uzunluğunda bir vektördür çünkü R, tek tek sözcükler veya karakterler yerine yalnızca farklı dizelerin toplam sayısını sayar. Tek tek karakterlerin sayısını saymak için `nchar` fonksiyonunu kullanabilirsiniz.

```
nchar(x)
```

```
## [1] 63
```

`cat` ve `paste` yardımıyla birleştirmek istediğiniz argümanları bir araya getirebilirsiniz.

```
gpt3<-c("içerik üretmek için derin öğrenmeyi kullanan",
        "Generative Pre-trained Transformer 3",
        "insanların yazdığı metinlere benzer")
```

```
cat(gpt3[2], "(kısaca GPT-3)", ",", gpt3[3], gpt3[1], "özbağımlı dil modelidir.")
```

```
## Generative Pre-trained Transformer 3 (kısaca GPT-3) , insanların yazdığı metinlere benzer
```

```
paste(gpt3[2], "(kısaca GPT-3)", ",", gpt3[3], gpt3[1], "özbağımlı dil modelidir.")
```

```
## [1] "Generative Pre-trained Transformer 3 (kısaca GPT-3) , insanların yazdığı metinlere benzer"
```

Elde etmeniz gereken boşluk içermeyen bir metinse bu bilgiyi de fonksiyon içerisinde belirterek (`sep=""`) uygun çıktıyı elde edebilirsiniz.

```
paste("bir", "iki", "üç", "dört", sep="")
```

```
## [1] "birikiüçdört"
```

Benzer şekilde boşluklar farklı sembollerle de doldurulabilir.

```
paste("bir", "iki", "üç", "dört", sep="**")
```

```
## [1] "bir**iki**üç**dört"
```

```
paste("bir", "iki", "üç", "dört", sep="--")
```

```
## [1] "bir--iki--üç--dört"
```

Metin içerisinde belli bir kısmı almak istediğinizde `substr` komutu işlemi kolaylıkla gerçekleştirecektir.

```
x<-"OpenAI, kâr amacı gütmeyen bir yapay zeka araştırma şirketidir."
substr(x,start=32 ,stop=41)

## [1] "yapay zeka"
```

5.6.3 Faktörler (Factors)

Faktörler, kategorik verileri temsil etmek için kullanılır ve sıralanmamış veya sıralanmamış olabilir. Kategorik veriler, veri biliminde önemli bir rol oynamaktadır. Bir faktör, her tam sayının bir etikete sahip olduğu bir tamsayı vektörü olarak düşünülebilir.

Faktör nesneleri, `factor()` işlevi ile oluşturulabilir.

```
k<-factor(c("evet","hayır","evet","evet","evet","hayır"));k
```

```
## [1] evet hayır evet evet evet hayır
## Levels: evet hayır
```

```
class(k)
```

```
## [1] "factor"
```

`table()` fonksiyonu ile verideki faktörlerin sıklığını gözlemek mümkündür.

```
table(k)
```

```
## k
## evet hayır
##      4      2
```

Faktör seviyeleri varsayılan yapıda alfabetik olarak sıralanmaktadır, seviyelerin sırasına müdahale etmek `levels` komutu ile mümkündür.

```
k<-factor(c("evet","hayır","evet","evet","evet","hayır"));k # alfabetik
```

```
## [1] evet hayır evet evet evet hayır
## Levels: evet hayır
```

```
k2<-factor(c("evet","hayır","evet","evet","evet","hayır"), levels=c("hayır","evet"));k2
```

```
## [1] evet hayır evet evet evet hayır
## Levels: hayır evet
```

Verideki mevcut faktör seviyelerini yine `levels` komutu ile sorgulayabilirsiniz.

```
aylar<-factor(c("mart","ağustos","ekim","ocak", "nisan","eylül","haziran","temmuz","şubat",
               "mayıs","kasım", "aralık"))
```

```
length(aylar)

## [1] 12
class(aylar)

## [1] "factor"
levels(aylar) #alfabetik

## [1] "ağustos" "aralık" "ekim" "eylül" "haziran" "kasım" "mart"
## [8] "mayıs" "nisan" "ocak" "şubat" "temmuz"
```

5.7 Eksik Gözlemler (Missing Values)

Eksik gözlemler veri setinde NA veya NaN olarak tanımlanmaktadır.

- `is.na()` NA sorgulama için kullanılır.
- `is.nan()` NaN sorgulama için kullanılır.
- NA integer veya character olabilir.
- NaN aynı zamanda NA iken tersi doğru değildir.

NA ve NaN içeren bir seri oluşturup mevcut olma durumunu sorgulamak istersek;

```
x <- c(1, 2, NA, 10, 3, NaN)
is.na(x)

## [1] FALSE FALSE TRUE FALSE FALSE TRUE
is.nan(x)

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

Chapter 6

Import Export İşlemleri

Chapter 7

Proje

Chapter 8

DPLYR

Chapter 9

Apply Ailesi

Chapter 10

Grafikler

Chapter 11

Döngüler

Chapter 12

Fonksiyonlar

Chapter 13

Referans

