

한국기술교육대학교 제2캠퍼스 실학관 902호

『1일차』 : 오후

◆ 훈련과정명 : 데이터 분석과 활용

◆ 훈 련 기 간 : 2023.12.19 - 2023.12.20

1	『3과목』 데이터 분석
2	“
3	”
4	“

『3과목』 데이터 분석

제1장 데이터 분석 개요

제2장 R프로그래밍 기초

제3장 데이터 마트

제4장 통계 분석

제5장 정형 데이터 마이닝



『3과목』 데이터 분석

제1장 데이터 분석 개요



학습목표

- 데이터 처리 프로세스를 이해한다.
- 데이터 분석 기법 중 시각화를 이해한다.
- 데이터 분석 기법 중 공간분석을 이해한다.
- 데이터 분석 기법 중 탐색적 자료 분석을 이해한다.

눈높이 체크

- 데이터를 분석을 위해 데이터 마트를 어떻게 만들까요?
- 데이터 분석 방법 중 시각화를 들어보셨나요?
- 데이터 분석 방법 중 공간분석을 들어보셨나요?
- 데이터 분석 방법 중 탐색적 자료분석을 들어보셨나요?



제1절 데이터 분석 기법 개요

데이터 처리

● 개요

- 데이터 분석은 통계 기반을 두고 있지만, 통계 지식과 복잡한 가정이 상대적으로 적은 실용적인 분야

● 데이터 활용

- 대기업은 데이터웨어하우스(DW)와 데이터마트(DM)를 통해 분석 데이터를 가져와서 사용한다.
- 신규 시스템이나 DW에 포함되지 못한 자료인 경우, 기존 운영시스템(Legacy)이나 스테이징(staging area) 영역과 ODS(Operational data Store)에서 데이터를 가져와서 DW에서 가져온 내용과 결합하여 활용할 수 있다.
- 하지만 운영 시스템에 직접 접근해 데이터를 활용하는 것은 매우 위험한 일이므로 거의 이루어지지 않고 있다.
- 스테이징 영역의 데이터는 운영시스템에서 임시로 저장된 데이터이기 때문에 가급적이면 클린징 영역인 ODS에서 데이터 전처리를 해서 DW나 DM과 결합하여 활용하는 것이 가장 이상적이다.



제1절 데이터 분석 기법 개요

데이터 처리

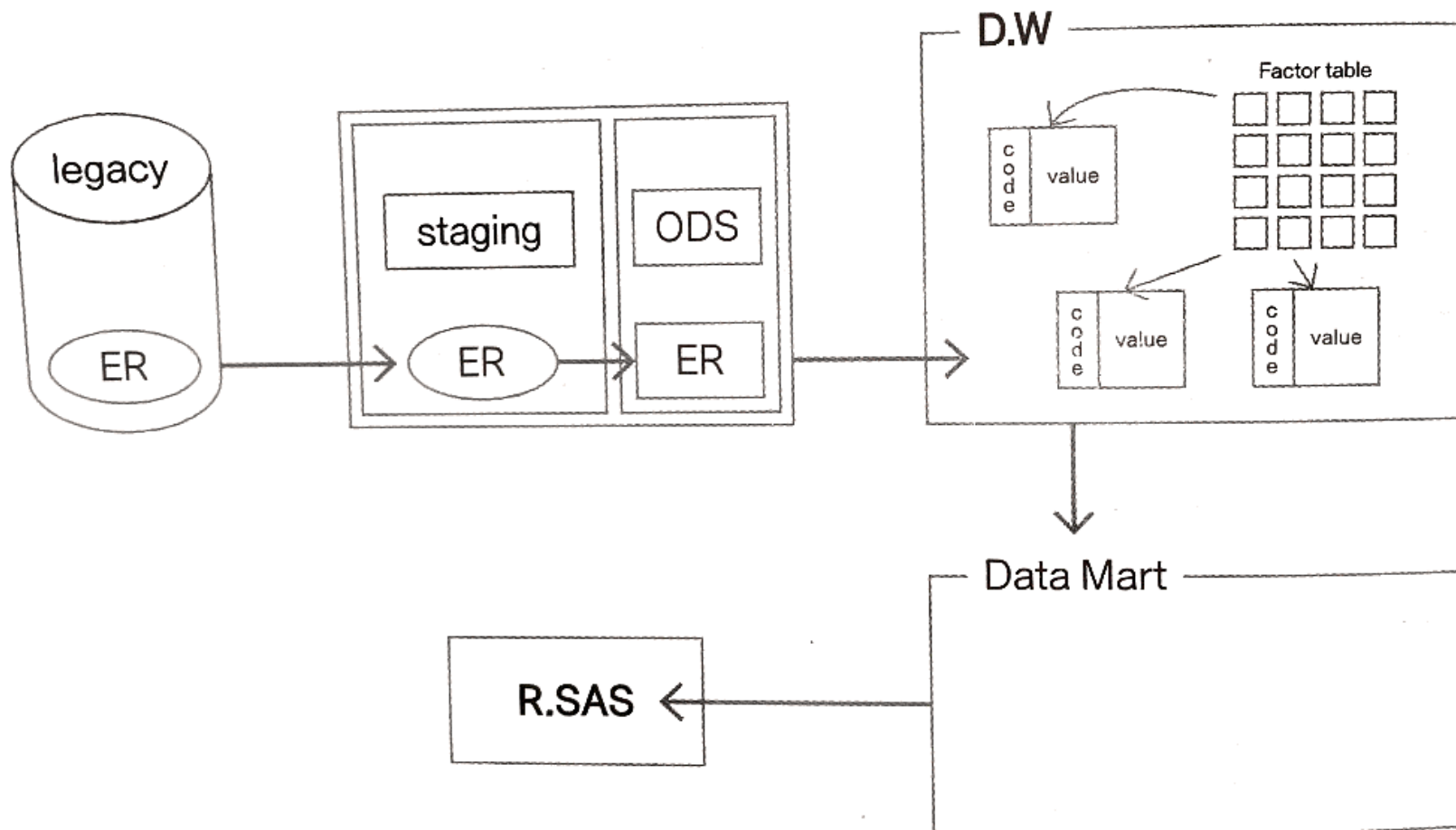
- * 데이터웨어하우스: 기업 내의 의사결정 지원 어플리케이션에 정보 기반을 제공하는 하나의 통합된 데이터 저장 공간. 데이터 주제 지향성, 데이터 통합, 데이터 시계열성(시간의 흐름에 따라 변화하는 값 유지), 데이터 비휘발성
- * 데이터 마트: 재무, 생산, 운영 등과 같이 특정 조직의 특정 업무 분야에 초점을 맞춰 구축



제1절 데이터 분석 기법 개요

데이터 처리

● 구성도





제1절 데이터 분석 기법 개요

데이터 처리

- 최종 데이터를 구조로 가공
- 데이터 마아닝 분류: 분류 값과 입력 변수들을 연관시켜, 인구 통계, 요약 변수, 파생변수 등을 산출
- 정형화된 패턴 처리: 비정형 데이터나 소셜 데이터는 정형화된 패턴으로 처리해야 한다.
 - 1) 비정형데이터: DBMS에 저장됐다가 텍스트 마이닝을 거쳐 데이터 마트와 통합한다.
 - 2) 관계형데이터: DMBS에 저장되어 사회 신경망 분석을 거쳐 분석 결과 통계값이 마트와 통합되어 활용된다.



제1절 데이터 분석 기법 개요

시각화

- 시각화는 가장 낮은 수준의 분석이지만 잘 사용하면 복잡한 분석 보다는 효율적
- 빅데이터 분석에서 시각화는 필수
- 탐색적 분석에서 시각화는 필수
- SNA분석(사회연결망 분석) 시 자주 활용



제1절 데이터 분석 기법 개요

공간분석(GIS)

- 공간분석은 공간적 차원과 관련된 속성들을 시각화하는 분석
- 지도 위에 관련 속성들을 생성하고 크기, 모양, 선 굵기 등으로 구분하여 인사이트를 얻는다.



제1절 데이터 분석 기법 개요

탐색적 자료 분석(EDA)

- 다양한 차원과 값을 조합해가며 특이점이나 의미 있는 사실을 도출하고 분석의 최종 목적을 달성해가는 과정
- 탐색적 데이터 분석과정은 데이터에 포함된 변수의 유형이 어떻게 되는지를 찾아가는 과정
- 데이터의 특징과 내재하는 구조적 관계를 알아내기 위한 기법들의 통칭
- EDA의 4가지 주제: 저항성의 강조, 잔차 계산, 자료 변수의 재표현, 그래프를 통한 현시성
- 탐색적 분석 효율의 예: 2과목 모형 개발 프로세스(KDD, CRSIP-DM 등)에서 언급한 바와 같이 데이터 이해관계(변수의 분포와 특성 파악)와 변수 생성 단계(분석 목적에 맞는 주요한 요약 및 파생 변수 생성) 그리고 변수 선택 단계(목적 변수에 의미 있는 후보 변수 선택)에서 활용되고 있다.



제1절 데이터 분석 기법 개요

통계분석

- 통계: 어떤 현상을 종합적으로 한눈에 알아보기 쉽게 일정한 체계에 따라 숫자와 표, 그림의 형태로 나타낸 것
- 기술통계: 모집단으로부터 표본을 추출하고 표본이 가지고 있는 정보를 쉽게 파악할 수 있도록 데이터를 정리하거나 요약하기 위해 하나의 숫자 또는 그래프 형태로 표현하는 절차
- 추측(추론)통계: 모집단으로부터 추출된 표본의 표본 통계량으로부터 모집단의 특성인 모수에 관해 통계적으로 추론하는 절차
- 활동분야
 - 정부의 경제정책 수립과 평가의 근거자료로 활용(통계청의 실업률, 고용률, 물가지수)
 - 농업(가뭄, 수해 또는 병충해 등에 강한 품종의 개발 및 개량)
 - 의학(의학적 치료 방법의 효과나 신약 개발을 위한 임상실험의 결과 분석)
 - 경영(제품 개발, 품질관리, 시장조사, 영업관리 등에 활용)
 - 스포츠(선수들의 체질 향상 및 개선, 경기 분석과 전략 분석, 선수 평가와 기용 등)



제1절 데이터 분석 기법 개요

데이터마이닝

- 개요: 대용량의 자료로부터 정보를 요약하고 미래에 대한 예측을 목표로 자료에 존재하는 관계, 패턴, 규칙 등을 탐색하고 이를 모형화함으로써 이전에 알려지지 않은 유용한 지식을 추출하는 분석법
- 방법론
 - 데이터베이스에서의 지식탐색
 - 기계학습: 인공신경망, 의사결정나무, 클러스터링, 베이지안분류, SVM 등
 - 패턴인식: 장바구니분석, 연관규칙
- 활용분야
 - 데이터베이스 마케팅
 - 신용평가 및 조기경보시스템
 - 생물정보학
 - 텍스트 마이닝



제1절 데이터 분석 기법 개요

모델링 성능 평가 시 활용하는 평가 기준

- 모델링 성능 평가 시 활용하는 평가 기준
 - 데이터 마이닝에서 활용하는 평가기준: 정확도, 정밀도, 디렉트 레이트, 리프트 등
 - 시뮬레이션에서 활용하는 평가기준: Throughput, Average Waiting Time, Average Queue Length, Time in System

『3과목』 데이터 분석

제2장 R프로그래밍 기초



학습목표

- 데이터 분석 환경을 이해한다.
- 데이터 분석 도구 R의 특성을 이해한다.
- R을 설치하고 GUI를 이해한다.
- R Studio를 설치하고 GUI를 이해한다.

눈높이 체크

- 데이터를 분석을 위해 활용되고 있는 분석 도구에는 어떤 것이 있을까요?
- 최근 빠른 속도로 확산되고 있는 R언어를 아시나요?
- R GUI인 R Studio를 들어보셨나요?

제1절 R소개

R의 탄생

- R은 뉴질랜드 University of Auckland의 Ross Ihaka와 Robert Gentle에 의해 개발된 언어[1993년]
 - R은 통계 소프트웨어 개발과 자료 분석에 널리 사용되고 그래픽 처리 기능이 탁월한 언어
 - R은 GPL[General Public License]하에 배포되는 S프로그래밍 언어의 구현으로 GNU S라고도 함
 - R은 1995년 자유소프트웨어 재단의 GNU 일반 공중사용가서로 인해 무료로 공개됨
 - R은 1988년 소개된 S-PLUS의 무료버전
 - R/S 플랫폼은 통계전문가들의 사실상의 표준 플랫폼
 - 오픈소스임에도 고성능 컴퓨팅 속도와 데이터 처리능력, 각종 소프트웨어 및 구글, 아마존클라우드 서비스와의 API등 성능이 우수하고 연동*호환성이 좋다.
 - Windows뿐만 아니라 Linux나 Mac OS도 지원
 - R은 계산기의 역할을 하며 Java, C프로그램과 연동 가능
- ※ S언어는 1976년 미국의 Bell연구소의 John Chamber가 개발한 통계적인 프로그래밍 언어. S언어에는 무료배포버전인 R과 상용버전인 S-PLUS가 있다.

R기반의 작업환경

- 작업환경은 업무 규모와 본인에게 익숙한 환경이 무엇인지를 기준으로 선택
- 기업환경에서는 64Bbit환경의 듀얼코어, 32GB RAM, 2TB 디스크, 리눅스 운영체제를 추천
- R의 메모리
 - 64bit 유닉스 환경 : 메모리 무제한
 - x86 64bit환경 : 128TB까지 지원
 - 64bit 윈도우 환경 : 8TB까지 지원

제1절 R소개

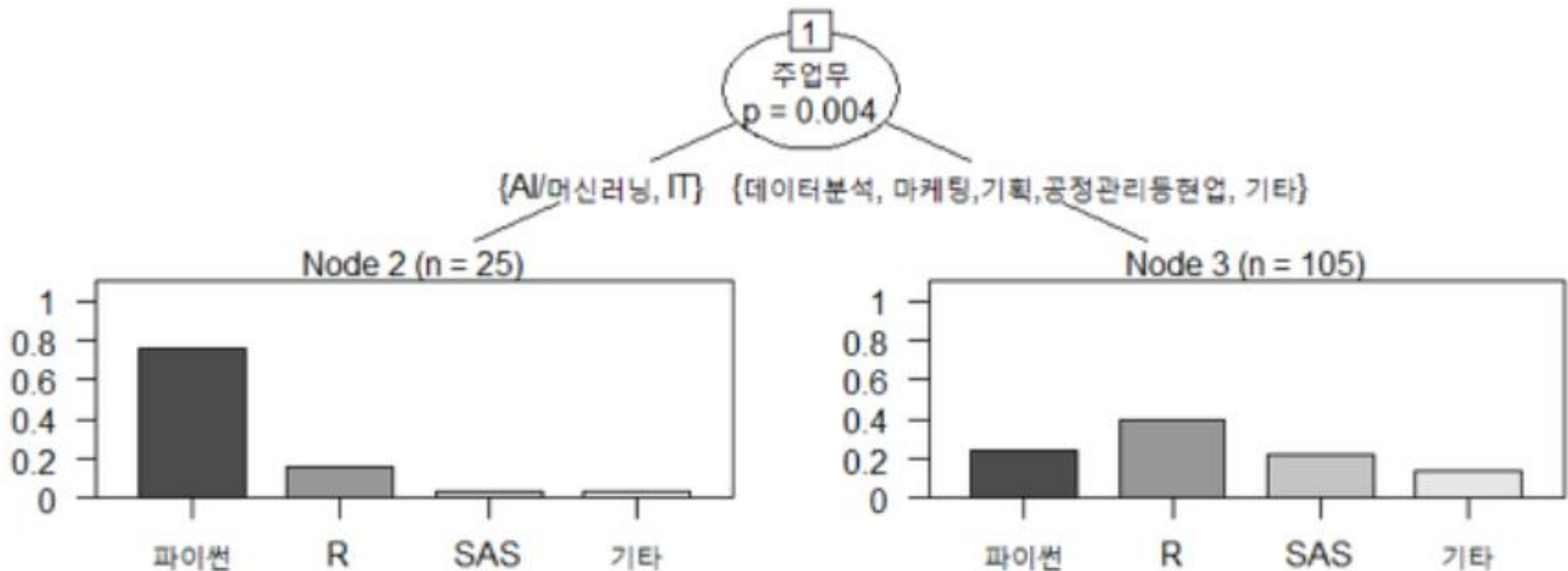
분석도구의 비교

	SAS	SPSS	오픈소스 R
프로그램 비용	유료, 고가	유료, 고가	오픈소스
설치 용량	대용량	대용량	모듈화로 간단
다양한 모듈 지원 및 비용	별도 구매	별도 구매	오픈소스
최근 알고리즘 및 기술 반영	느림	다소느림	매우빠름
학습자료 입수의 편의성	유료 도서 위주	유료 도서 위주	공개 논문 및 자료 많음
질의를 위한 공개 커뮤니티	NA	NA	매우활발

제1절 R소개

분석도구의 비교

- 2019년 구글 서베이를 사용한 온라인 조사 결과를 보면, AI/머신러닝 업무에서 주로 파이썬이 압도적으로 많이 사용되고 있습니다. 그러나 일반 데이터 분석에서는 R을 사용되고 있습니다.



R의 주요 특징

- 오픈 소스 프로그램
- 오픈 소스이므로 무료로 사용이 가능해 질의를 위한 사용자 커뮤니티가 매우 활발하다.
 - 구글 검색만으로도 충분히 답을 얻을 수 있을 정도로 커뮤니티가 활성화 되어 있다.
 - 하지만 다양한 사용자들을 통해 다양한 의견이 나오므로 적절한 해결책을 찾기 위해서는 시간과 노력이 필요하다.
- 다양한 기능을 지원하는 많은 패키지가 수시로 업데이트되므로 최신 알고리즘 패키지를 통해 활용하기 쉽다.
 - 다른 사용자가 자신이 제작한 패키지 업로드하면 누구나 다운 가능
 - 패키지 업데이트를 통한 우수한 확장성

R의 주요 특징

- 그래픽 및 성능 & 시스템 데이터 저장 방식
- 프로그래밍이나 그래픽 측면 등 대부분의 주요 특징들에서 상용 프로그램과 대등하거나 월등하다.
- 각 세션 사이마다 시스템에 데이터셋을 저장하므로 매번 데이터를 로딩할 필요가 없고 명령어 스토리도 저장 가능
 - 메모리에 데이터를 불러들여 작동하는 인메모리[InMemory]방식이라 빠른 속도로 데이터 처리가 가능
 - 메모리의 크기에 따라 분석할 수 있는 데이터의 양이 결정된다.
 - 인메모리(Inmemory) 방식이라 하둡의 분산프로세싱 프레임워크인 MapReduce방식을 적용하기 용이
 - 구글, 페이스북, 아마존 등이 빅데이터 분석에 R을 활용하는 이유

R의 주요 특징

- 객체지향언어이며 함수형 언어
 - 통계기능 뿐만 아니라 일반 프로그래밍언어처럼 자동화하거나 새로운 함수를 생성해 사용 가능
 - 객체지향 언어이므로 R은 추정계수, 표준윌차, 잔차 등 결과값을 객체에 저장해 필요한 부분을 호출해 사용 가능
- SAS, SPSS의 경우, 회귀분석 시 화면에 결과가 산더미로 나오게 된다. 따라서 분석결과를 활용하기 위해서는 추가로 프로그래밍하거나 별도의 작업이 필요
- 함수형 언어이므로 기존에 사용한 함수들을 활용함으로 프로그램이 더욱 깔끔하고 단축된 코드를 만든다.
 - 함수형 언어이므로 적절한 함수를 적용해 프로그래밍하면 매우 빠른 코드로 실행속도를 가진다.
 - 함수형 언어이므로 함수를 활용해 프로그래밍함으로 단순한 코드로 디버깅 노력이 감소한다.[디버깅이 쉽다]
 - 함수형 언어이므로 다른 프로그래밍 언어들에 비해 병렬 프로그래밍으로 전환이 용이하다.

R의 주요 특징

- 강력한 시각화[그래프] 기능
 - 단순한 코딩만으로도 2D, 3D그래픽, 지도, GIS, 동적 그래프를 지원해 빅데이터의 시각화가 용이하다.
- 모든 운영체제에서 사용 가능
 - Windows, Mac, Linux 운영체제에서 모두 사용가능
- 데이터 핸들링 기능
 - 텍스트, CSV, 엑셀, SAS, SPSS, DB, Stata 등의 다양한 데이터를 읽어오는 기능
 - 벡터, 행렬, 배열, 데이터 프레임, 리스트 등 다양한 형태의 데이터 구조 지원[다양한 형태의 분석 가능]
 - 수정, 삭제, 정렬, 합치기 등의 데이터 핸들링을 위한 기능

학습목표

- R GUI를 실행하여 프로그래밍을 할 수 있다.
- R GUI의 환경설정을 조정하고 편리한 기능들을 숙지한다.
- R 패키지를 이해하고 CRAN을 통해 다운로드하고 실행할 수 있다.
- R 파일을 실행하고 배치작업을 할 수 있다.
- 변수에 값을 할당하고 변수를 삭제할 수 있다.
- 기본적인 통계량을 계산할 수 있다.
- R에서의 연산자를 확인하고 우선순위를 이해할 수 있다.
- 함수의 생성 방법을 이해하고 활용할 수 있다.

눈높이 체크

- R GUI와 R studio를 통해 R 프로그래밍을 구동해 보신 적이 있습니까?
- R GUI의 환경을 설정해 보신 적이 있습니까?
- R 패키지를 다운받고 실행할 수 있습니까?
- R로 만들어진 프로그램을 실행하고 배치 작업을 경험해 보았습니까?
- 통계분석을 위해 변수들을 생성하고 삭제할 수 있습니까?
- 기본적으로 통계분석에서 활용되는 통계량과 계산 방법을 알고 계십니까?
- R 프로그램의 대입연산자, 사칙연산자, 비교연산자를 이해하십니까?
- R 프로그램에서 함수를 직접 생성하고 활용할 수 있다는 것을 이해하십니까?

통계 패키지 R

- GUI[Graphical User Interface]
 - 사용자가 컴퓨터와 정보를 교환할 때, 그래픽을 통해 작업할 수 있는 환경
 - 마우스 등을 이용해 화면에 있는 메뉴를 선택하여 작업할 수 있다.

R 설치

- <https://cran.r-project.org/bin/windows/base/>

Download R-4.3.2 for Windows

cran.r-project.org/bin/windows/base/

R-4.3.2 for Windows

[Download R-4.3.2 for Windows](#) (79 megabytes, 64 bit)

[README on the Windows binary distribution](#)

[New features in this version](#)

This build requires UCRT, which is part of Windows since Windows 10 and Windows Server 2016. On older systems, UCRT has to be installed manually from [here](#).

If you want to double-check that the package you have downloaded matches the package distributed by CRAN, you can compare the [md5sum](#) of the .exe to the [fingerprint](#) on the master server.

Frequently asked questions

- [Does R run under my version of Windows?](#)
- [How do I update packages in my previous version of R?](#)

Please see the [R FAQ](#) for general information about R and the [R Windows FAQ](#) for Windows-specific information.

Other builds

- Patches to this release are incorporated in the [r-patched_snapshot_build](#).
- A build of the development version (which will eventually become the next major release of R) is available in the [r-devel_snapshot_build](#).
- [Previous releases](#)

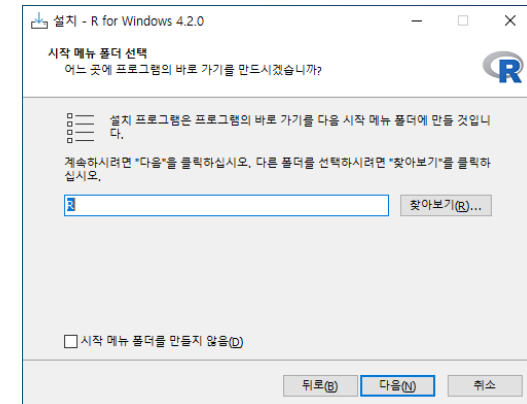
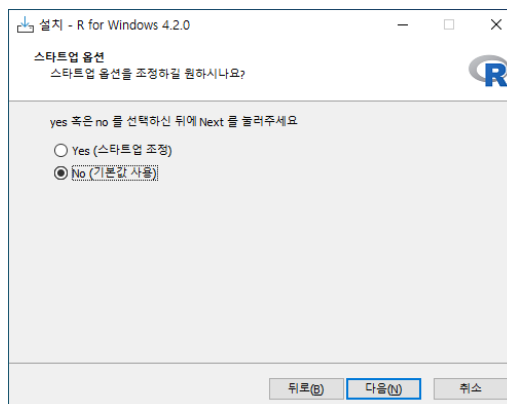
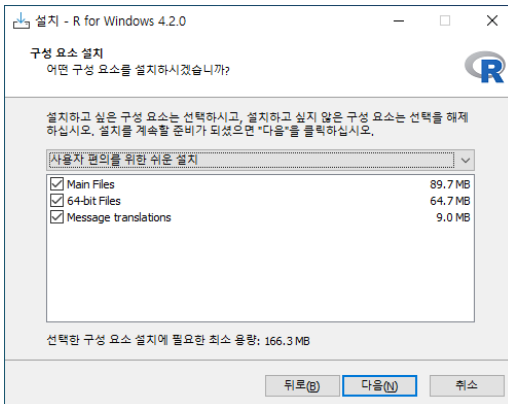
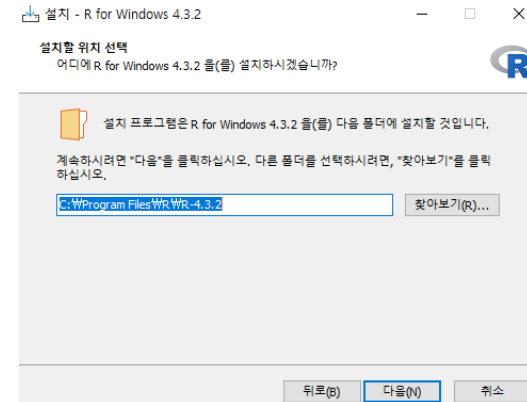
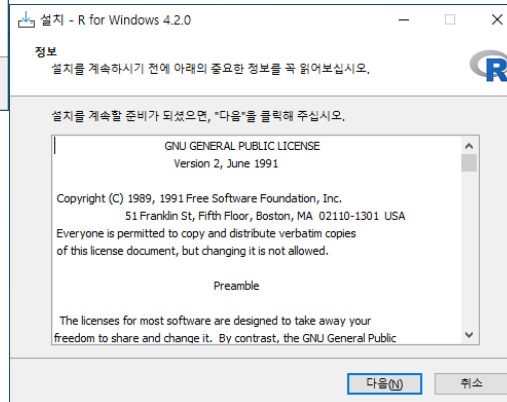
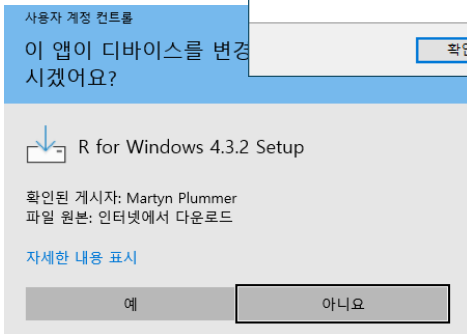
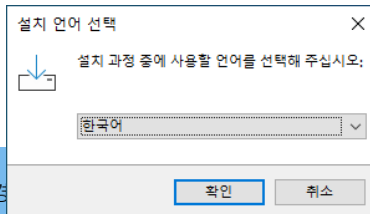
Note to webmasters: A stable link which will redirect to the current Windows binary release is [<CRAN MIRROR>/bin/windows/base/release.html](https://cran.r-project.org/bin/windows/base/release.html).

Last change: 2023-10-31

제2절 R기초

R 설치

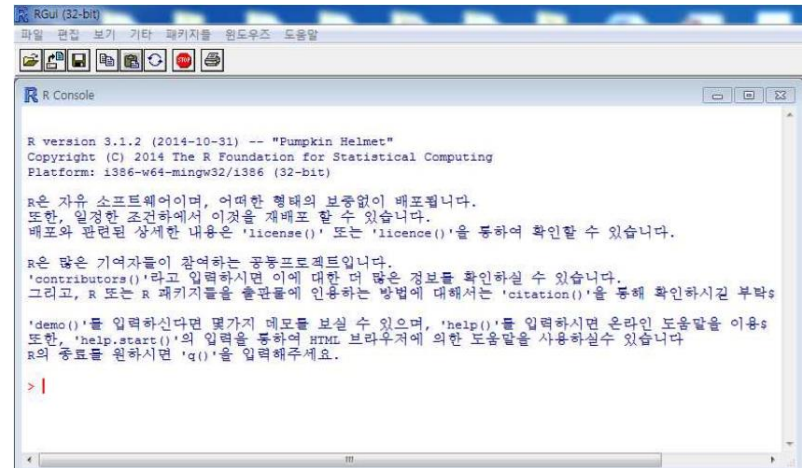
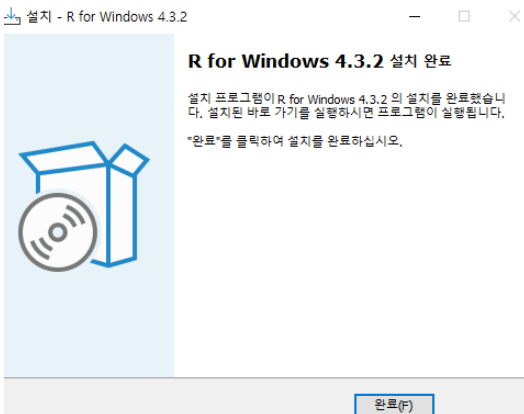
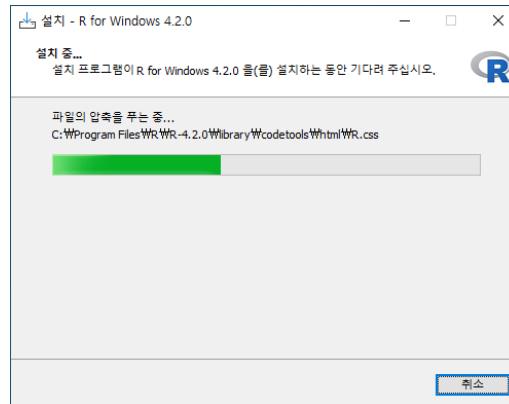
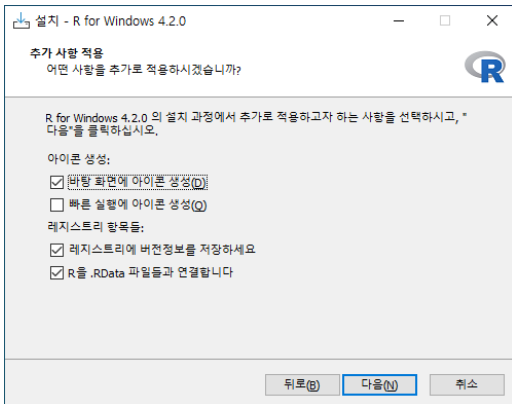
- <https://cran.r-project.org/bin/windows/base/>



제2절 R기초

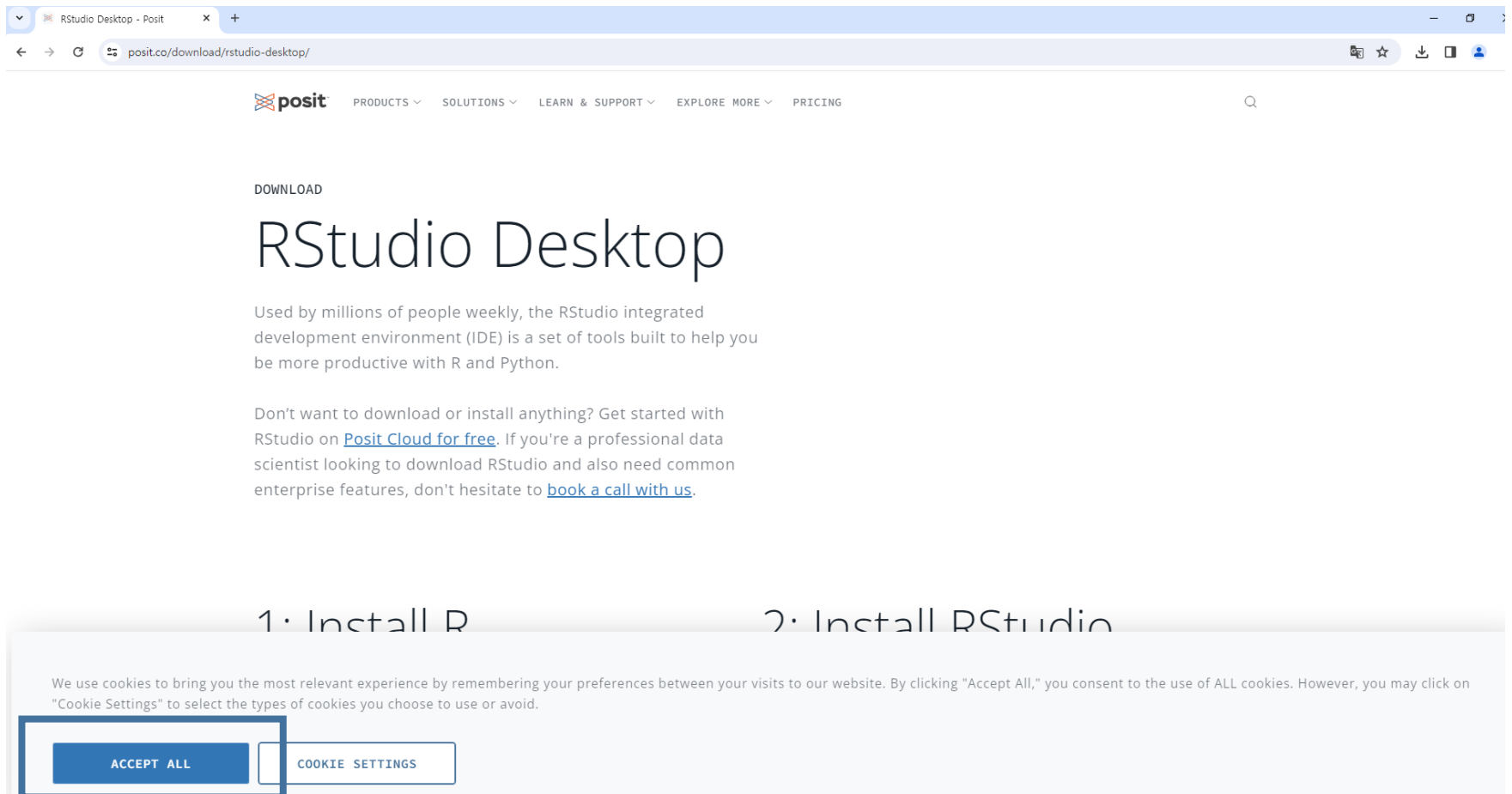
R 설치

- <https://cran.r-project.org/bin/windows/base/>



RStudio 설치

- <https://www.rstudio.com/products/rstudio/download/>

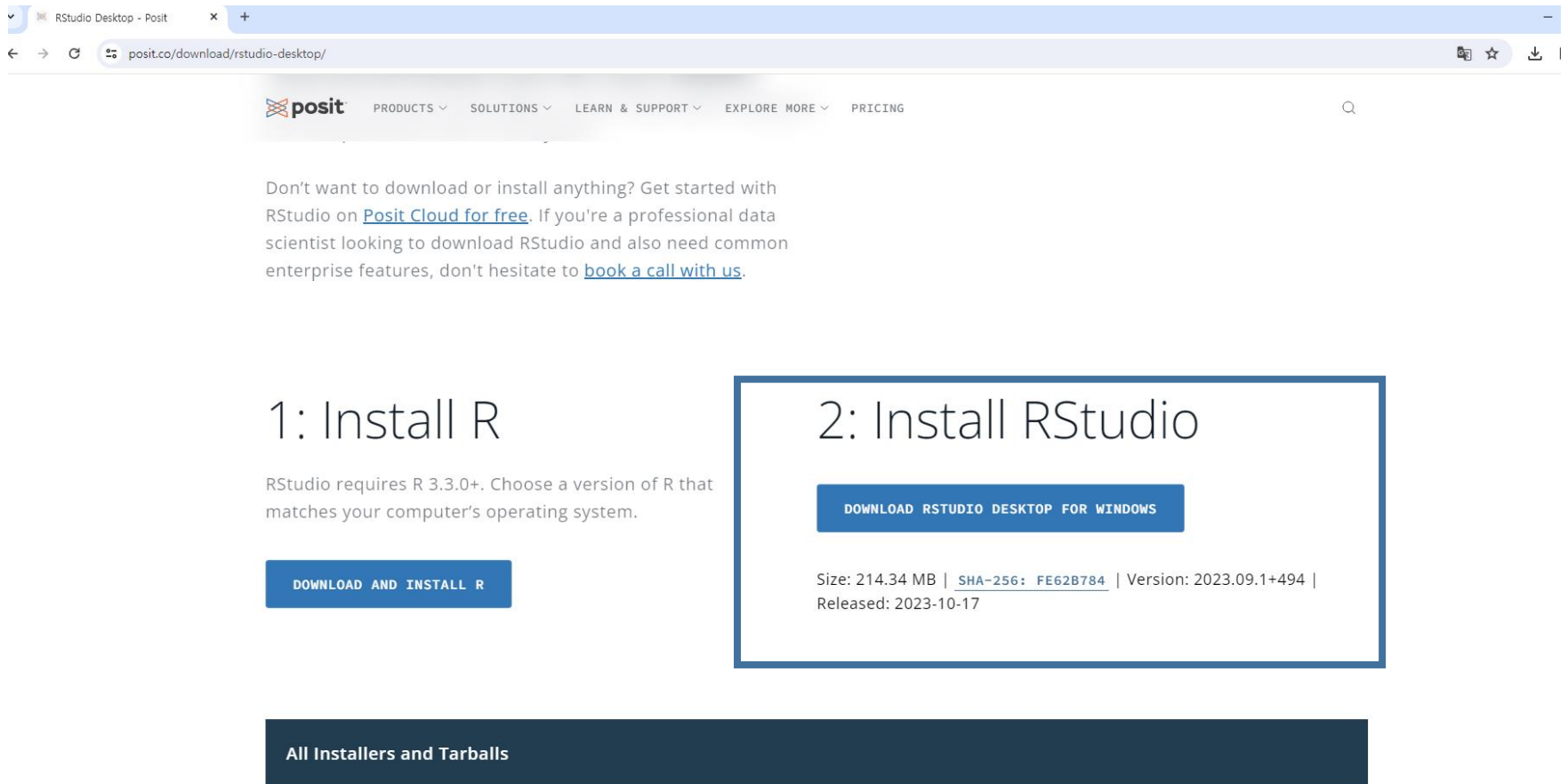


The screenshot shows a web browser window with the URL <https://www.rstudio.com/products/rstudio/download/>. The page is titled "RStudio Desktop" and describes it as an integrated development environment (IDE) for R and Python. It includes a navigation bar with links to PRODUCTS, SOLUTIONS, LEARN & SUPPORT, EXPLORE MORE, and PRICING. The main content area features a "DOWNLOAD" section with the heading "RStudio Desktop" and a description: "Used by millions of people weekly, the RStudio integrated development environment (IDE) is a set of tools built to help you be more productive with R and Python." Below this, it says: "Don't want to download or install anything? Get started with RStudio on [Posit Cloud for free](#). If you're a professional data scientist looking to download RStudio and also need common enterprise features, don't hesitate to [book a call with us](#)."

At the bottom of the page, there are two tabs: "1. Install R" and "2. Install RStudio". Below these tabs is a cookie consent banner that reads: "We use cookies to bring you the most relevant experience by remembering your preferences between your visits to our website. By clicking 'Accept All,' you consent to the use of ALL cookies. However, you may click on 'Cookie Settings' to select the types of cookies you choose to use or avoid." The banner contains two buttons: "ACCEPT ALL" and "COOKIE SETTINGS".

RStudio 설치

- <https://www.rstudio.com/products/rstudio/download/>



The screenshot shows a web browser window with the URL posit.co/download/rstudio-desktop/. The page features the Posit logo and navigation links: PRODUCTS, SOLUTIONS, LEARN & SUPPORT, EXPLORE MORE, and PRICING. A search icon is also present. The main content area includes a paragraph about downloading RStudio on Posit Cloud for free, with links for professional data scientists and enterprise features. Below this, there are two main sections: '1: Install R' and '2: Install RStudio'. The '1: Install R' section has a button labeled 'DOWNLOAD AND INSTALL R'. The '2: Install RStudio' section is highlighted with a blue border and contains a button labeled 'DOWNLOAD RSTUDIO DESKTOP FOR WINDOWS'. Below this button, the size (214.34 MB), SHA-256 hash (FE62B784), version (2023.09.1+494), and release date (2023-10-17) are listed. At the bottom, there is a dark blue bar with the text 'All Installers and Tarballs'.

Don't want to download or install anything? Get started with RStudio on [Posit Cloud for free](#). If you're a professional data scientist looking to download RStudio and also need common enterprise features, don't hesitate to [book a call with us](#).

1: Install R

RStudio requires R 3.3.0+. Choose a version of R that matches your computer's operating system.

[DOWNLOAD AND INSTALL R](#)

2: Install RStudio

[DOWNLOAD RSTUDIO DESKTOP FOR WINDOWS](#)

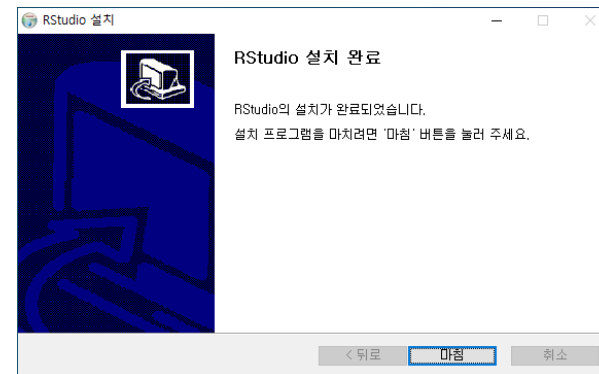
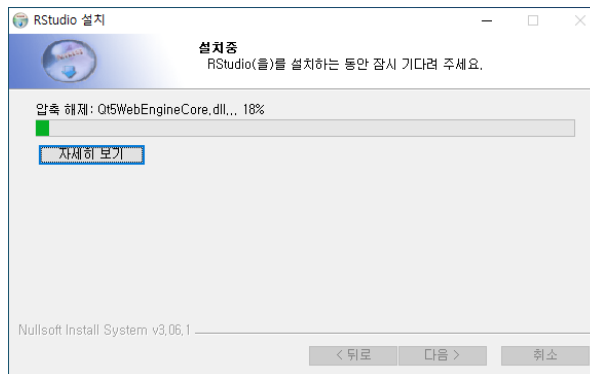
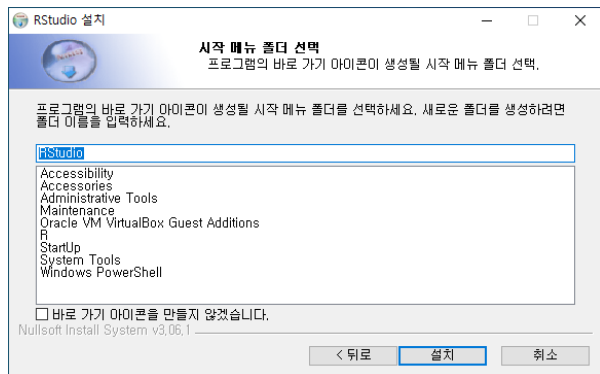
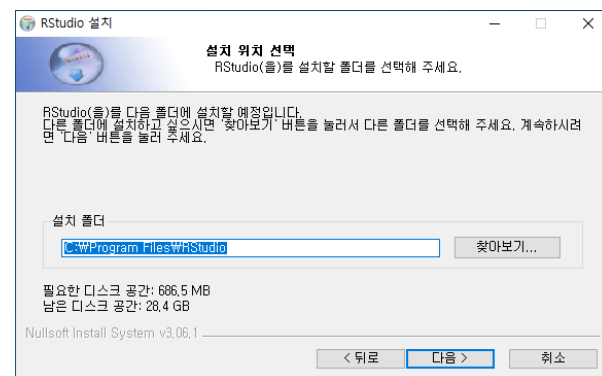
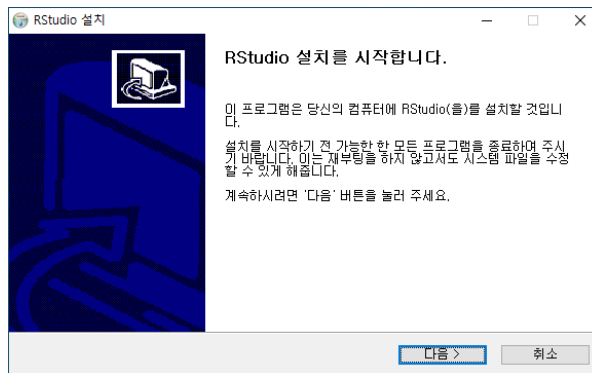
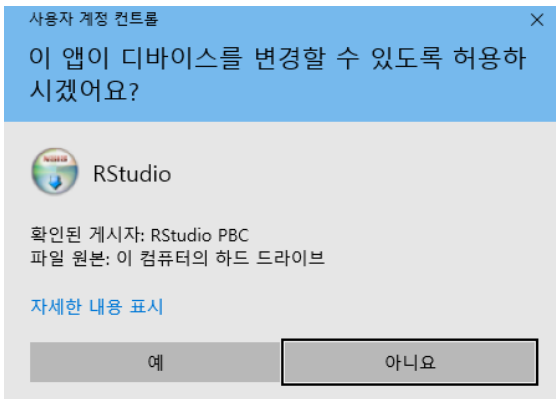
Size: 214.34 MB | [SHA-256: FE62B784](#) | Version: 2023.09.1+494 | Released: 2023-10-17

[All Installers and Tarballs](#)

제2절 R기초

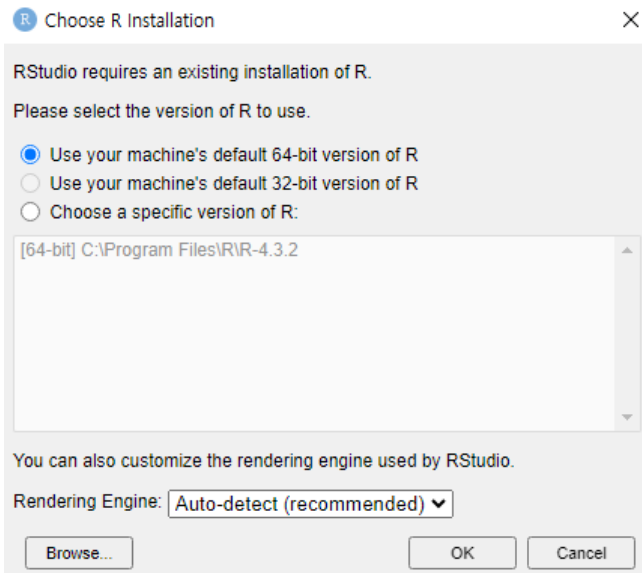
RStudio 설치

- <https://www.rstudio.com/products/rstudio/download/>



RStudio 설치

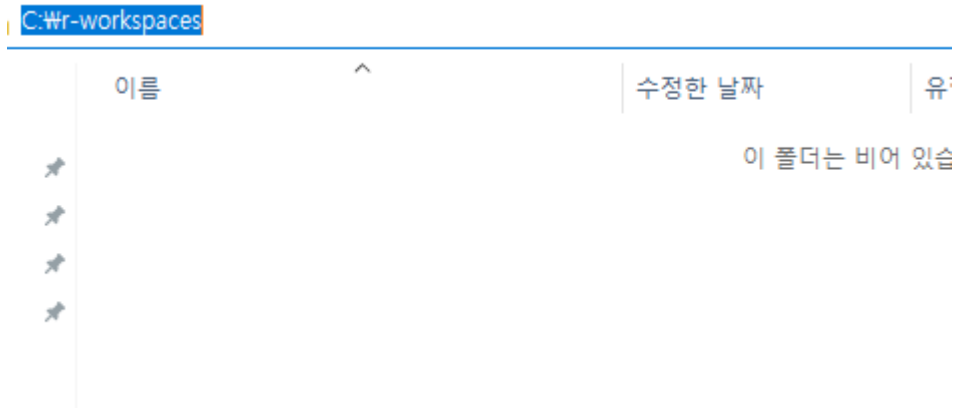
● Rstudio 실행 화면



```
> R.version
platform x86_64-w64-mingw32
arch      x86_64
os        mingw32
system    x86_64, mingw32
status
major     3
minor     5.1
year      2018
month     07
day       02
svn rev   74947
language  R
version.string R version 3.5.1 (2018-07-02)
nickname  Feather Spray
> |
```

작업 폴더 만들기

- C:\Wr-workspaces

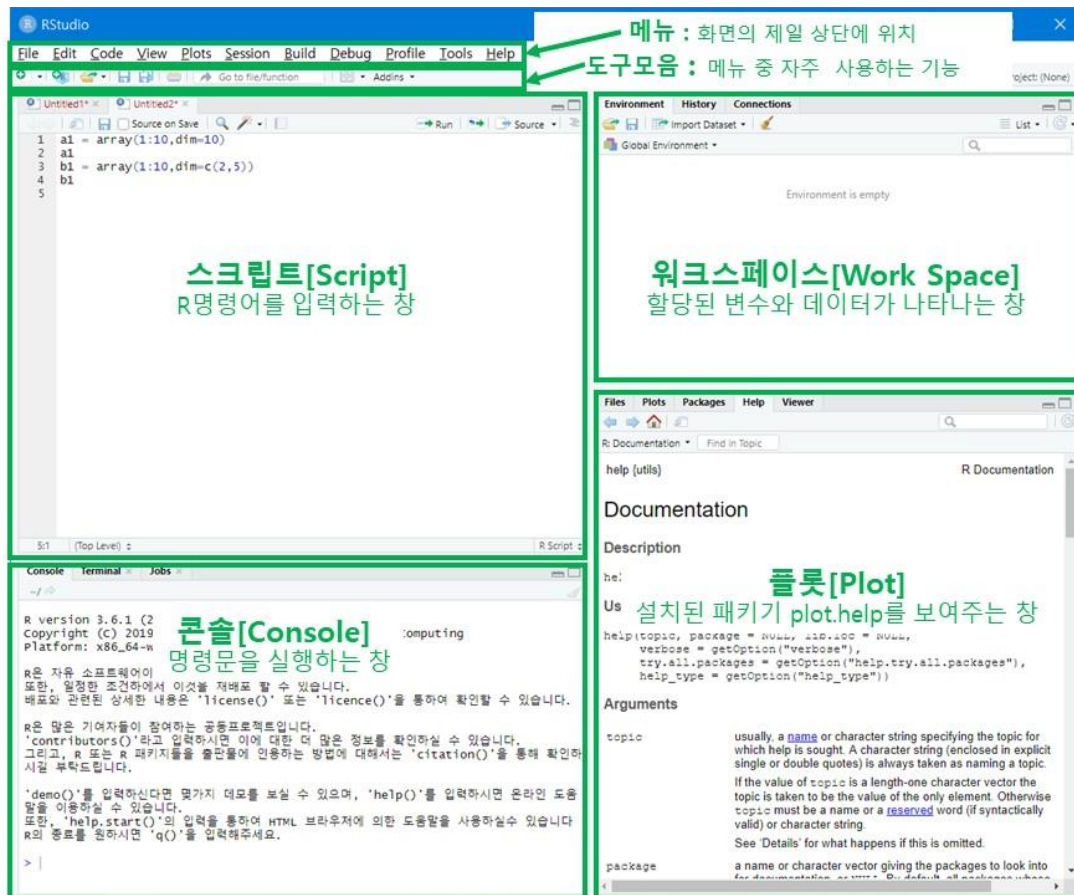


제2절 R기초

통계 패키지 R

● GUI[Graphical User Interface]의 종류

① R STUDIO 구성 화면



통계 패키지 R

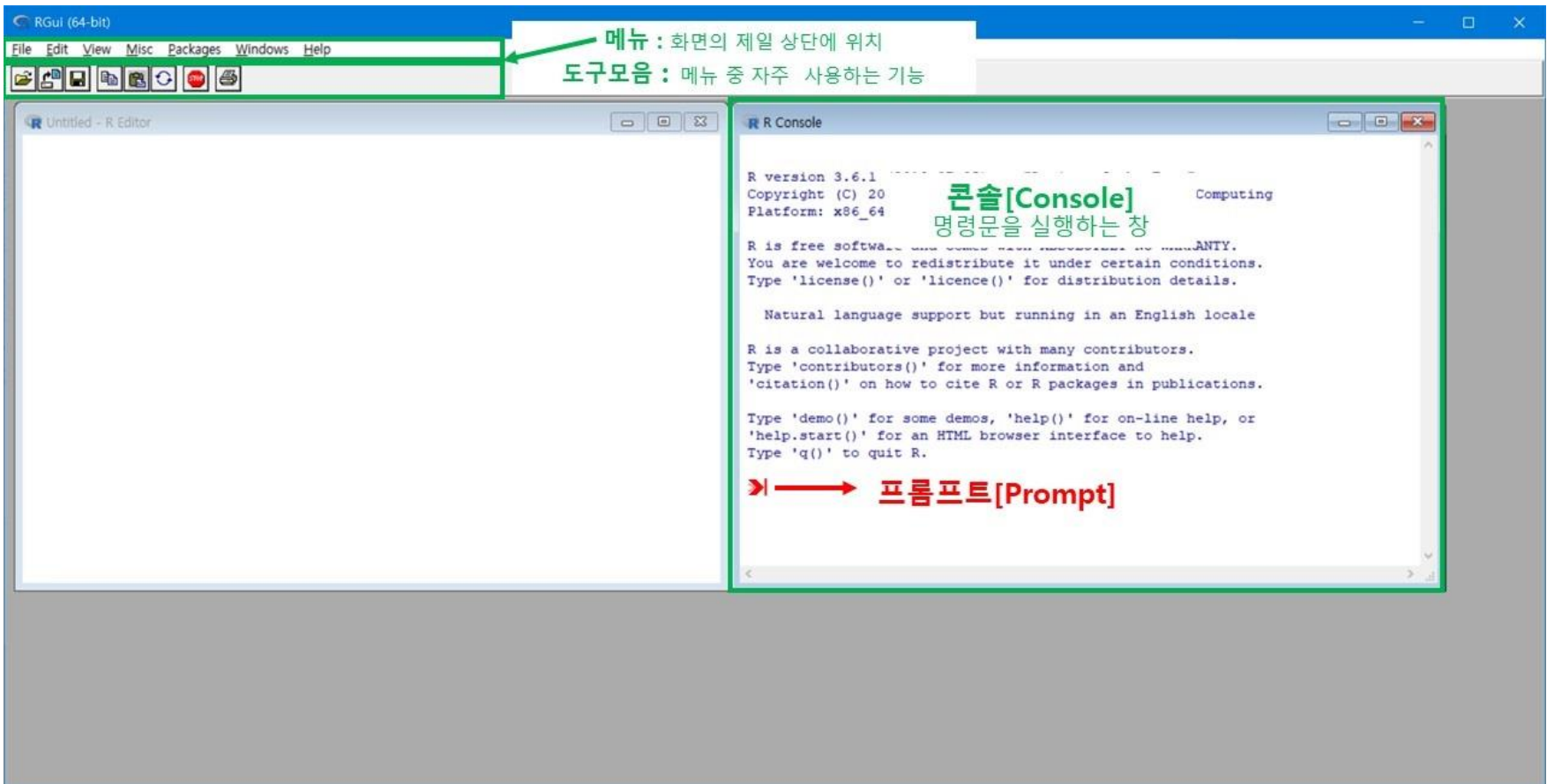
● GUI[Graphical User Interface]의 종류

① R STUDIO 구성 화면

- 프로그래밍 언어 R을 위한 자유-오픈소스 통합개발환경[IDE]으로 다양한 운영체제를 지원하는 오픈소스
- 메모리에 변수가 어떻게 되어 있는지, 타입이 무엇인지 확인 가능
- 스크립트 관리와 도큐멘테이션이 편리
- 코딩을 해야하는 부담이 있으나 스크립트용 프로그래밍으로 어렵지 않게 자동화가 가능
- Rattle[래틀]은 GUI가 패키지와 긴밀하게 결합되어 있어 정해진 기능만 사용 가능해 업그레이드가 제대로 되지 않으면 통합성에 문제가 발생할 수 있다.
- ※ Rattle은 GUI환경에서보다 편리하게 사용할 수 있도록 도와주는 패키지 (컴퓨터 언어에 익숙하지 않은 초보자들이 별다른 사전 지식 없이도 R을 이용할 수 있게 해줌)
- 스크립트 창에서 작성된 코드는 [Ctrl + Enter]키를 누르면 한 줄씩 실행 됨
- → 작성된 코드 중 일부만 실행시키고 싶다면 그 부분만 드래그한 후 [Ctrl + Enter]키 누름

통계 패키지 R

- GUI[Graphical User Interface]의 종류
 - ② R GUI



통계 패키지 R

● GUI[Graphical User Interface]의 종류

② R GUI

- 메뉴, 도구모음, 콘솔(Console) 등으로 구성
- 콘솔(Console) : R의 입력된 명령어가 실행되는 창
- 프롬프트(Prompt) : 콘솔 내의 ">> " 로 명령을 내리면 컴퓨터가 알아서 찾아줌
 - 프롬프트에 R의 문법에 맞게 명령어 입력 후 Enter를 치면 해당 명령어가 실행됨
- → 문법에 맞지 않는 명령어 실행 시 오류 메시지
- → 결과에 문제가 발생할 가능성이 있으면 경고메시지
- → 수치 연산 등의 결과는 콘솔 화면에 제시하며 그래프 작성시 별도의 화면이 실행되어 작성
- 도움말을 실행하면 인터넷 브라우저 화면이 실행되어 R의 사용방법에 대한 내용 확인 가능

통계 패키지 R

- GUI[Graphical User Interface]의 종류
 - ③ 그 외에도 Microsoft Visual Studio, R Commander, Atom, ConTEXT, Eclipse, Emacs, Kate, Notepad++, Architect, WinEdt, Tinn-R, TextMate, Sublime Text 등의 GUI 종류가 있다.

패키지[Package]

- R함수, 데이터, 코드, 문서 등을 묶은 것 [R함수와 데이터 및 컴파일된 코드의 모임]
- R이 설치되면 자동으로 stats, graphics, grDevices, utils, datasets, methods, base의 패키지가 설치되어 통계와 그래프에 대한 기본 기능을 제공
- 자동으로 설치된 패키지를 제공하지 못하는 기능들은 해당 기능을 제공하는 새로운 패키지를 설치해 사용
- R의 패키지를 많이 알고 특히 패키지에서 제공하는 함수를 많이 아는 것이 실력
- <https://cran.r-project.org/>에서 약 12,000개의 패키지들을 제공 [주제별로 분류해 놓음]
 - → 관심있는 주제를 클릭하면 관련 패키지들의 목록과 해당 패키지에 대한 설명이 있음
- 통계청의 "통그라미"라는 무료 프로그램에 막대, 줄기, 원 등 다양한 그래프가 있음

패키지[Package]

- 패키지 설치
 - 새로운 패키지를 설치하기 위해서는 기본적으로 인터넷이 연결되어 있어야 함
 - R의 패키지 제공 서버와 패키지를 설치할 컴퓨터 간에 통신이 되어야 패키지 설치 가능
 - 다른 컴퓨터에 다운로드된 패키지를 USB나 외장하드에 저장해서 복사하면 인터넷 연결 없이도 사용가능
 - 패키지 설치하는 동일 컴퓨터에서 한 번만 하면 됨
 - 패키지를 재설치하는 경우는 패키지가 삭제되었거나 패키지가 업데이트된 경우임
 - 패키지를 설치할 때는 `install.packages("패키지명")` 함수를 사용
 - `ex>> install.packages(c("패키지명1", "패키지명2"))`

패키지[Package]

```
>> >> setwd("C:/DEV/r-workspaces")  
C:/DEV/r-workspaces/library
```

```
C:\Program Files\R\R-4.3.2\etc\ Rprofile.site
```

```
# Things you might want to change
```

```
# options(papersize="a4")  
# options(editor="notepad")  
# options(pager="internal")
```

```
# set the default help type  
# options(help_type="text")  
# options(help_type="html")
```

```
# set a site library  
# .Library.site <- file.path(chartr("\\", "/", R.home()), "site-library")
```

```
# set a CRAN mirror  
# local({r <- getOption("repos")  
#   r["CRAN"] <- "http://my.local.cran"  
#   options(repos=r)})
```

```
# Give a fortune cookie, but only to interactive sessions  
# (This would need the fortunes package to be installed.)  
# if (interactive())  
#   fortunes::fortune()
```

```
# 추가한 부분
```

```
.libPaths("C:/DEV/r-workspaces/library")
```

패키지[Package]

- 패키지 로딩
 - 패키지가 설치되었다고 패키지에 있는 함수나 기능들을 바로 사용할 수 없음
 - 하드디스크: R을 설치하거나 업데이트를 통해 설치
 - 웹: 2014년 CRAN. 저장소에는 약 5000개 의 유용한 패키지가 자동 설치

```
>> >> install.packages("AID")
```

- 패키지 도움말
- 다운로드된 AID 패키지의 help 다큐먼트를 보여준다.

```
>> >> library(help=AID)
```

- 웹을 통한 AID 패키지의 다큐먼트가 보여준다.

```
>> >> help(package=AID)
```



● 패키지 확인

패키지 'AID'에 대한 정보

설명:

```

Package:      AID
Type:         Package
Title:        Box-Cox Power Transformation
Version:      2.7
Date:         2022-06-13
Depends:      R (>= 3.2.0)
Imports:      MASS, graphics, Rcpp, RcppEigen, RcppParallel, RcppShiny, RcppStat, RcppUnit, RcppVisualC, RcppVisualC64, RcppVisualC7, RcppVisualC8, RcppVisualC9, RcppVisualC10, RcppVisualC11, RcppVisualC12, RcppVisualC13, RcppVisualC14, RcppVisualC15, RcppVisualC16, RcppVisualC17, RcppVisualC18, RcppVisualC19, RcppVisualC20, RcppVisualC21, RcppVisualC22, RcppVisualC23, RcppVisualC24, RcppVisualC25, RcppVisualC26, RcppVisualC27, RcppVisualC28, RcppVisualC29, RcppVisualC30, RcppVisualC31, RcppVisualC32, RcppVisualC33, RcppVisualC34, RcppVisualC35, RcppVisualC36, RcppVisualC37, RcppVisualC38, RcppVisualC39, RcppVisualC40, RcppVisualC41, RcppVisualC42, RcppVisualC43, RcppVisualC44, RcppVisualC45, RcppVisualC46, RcppVisualC47, RcppVisualC48, RcppVisualC49, RcppVisualC50, RcppVisualC51, RcppVisualC52, RcppVisualC53, RcppVisualC54, RcppVisualC55, RcppVisualC56, RcppVisualC57, RcppVisualC58, RcppVisualC59, RcppVisualC60, RcppVisualC61, RcppVisualC62, RcppVisualC63, RcppVisualC64, RcppVisualC65, RcppVisualC66, RcppVisualC67, RcppVisualC68, RcppVisualC69, RcppVisualC70, RcppVisualC71, RcppVisualC72, RcppVisualC73, RcppVisualC74, RcppVisualC75, RcppVisualC76, RcppVisualC77, RcppVisualC78, RcppVisualC79, RcppVisualC80, RcppVisualC81, RcppVisualC82, RcppVisualC83, RcppVisualC84, RcppVisualC85, RcppVisualC86, RcppVisualC87, RcppVisualC88, RcppVisualC89, RcppVisualC90, RcppVisualC91, RcppVisualC92, RcppVisualC93, RcppVisualC94, RcppVisualC95, RcppVisualC96, RcppVisualC97, RcppVisualC98, RcppVisualC99, RcppVisualC100

```

Environment is empty

Console Terminal × Background Jobs ×

```

R version 4.2.1 (2022-06-23 ucrt) -- "Funny-Looking Kid"
Copyright (C) 2022 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (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.

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.

> library(help=AID)
> help(package=AID)
>

```

Files Plots Packages Help Viewer Presentation

R: Box-Cox Power Transformation Find in Topic

Box-Cox Power Transformation

Documentation for package 'AID' version 2.7

- [DESCRIPTION file](#)

Help Pages

AID-package	Box-Cox Power Transformation
AADI	Average Annual Daily Traffic Data
boxcoxfr	Box-Cox Transformation for One-Way ANOVA
boxcoxlm	Box-Cox Transformation for Linear Models

패키지[Package]

- 패키지 업데이트
 - 패키지는 버전이 비정기적으로 업데이트되므로 매번 직접해야 한다.
 - 이미 설치된 패키지 중에서 업데이트된 것을 적용할 때는 `update.packages("패키지명")` 함수 사용
- 패키지 삭제하기
 - 설치된 패키지를 삭제하기 위해서는 `remove.package("패키지명")` 함수 사용

프로그램 파일 실행

코드	기능 / 내용
<code>source('파일명')</code>	스크립트로 프로그래밍된 파일 실행하기 오른쪽 방향키
<code>sink(file, append, split)</code> 함수	R 코드 실행결과를 특정 파일에서 출력 file: 출력할 파일명 append: 파일에 결과를 덮어쓰거나 추가해서 출력(디폴트: False / 덮어쓰기) split: 출력파일에만 출력하거나 콘솔창에 출력(디폴트: False / 파일에만 실행결과 출력)
<code>pdf()</code> 함수	그래픽 출력을 .pdf로 지정[ex] <code>pdf("A_out.pdf")</code>
<code>dev.off()</code> 함수	프로그램 파일 닫기

제2절 R기초

프로그램 파일 실행

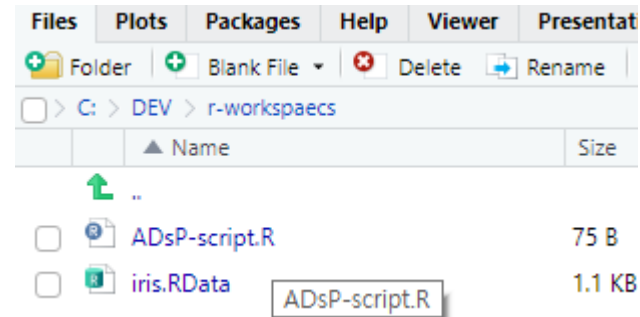
- 작업 폴더 생성
- C:\DEV\r-workspaces
- 사용자 정의 함수 저장

```
SUM <- function(a,b){  
  result <- a+b  
  return(result)  
}
```

- 많은 함수를 작성한 경우에는, 이 함수들을 하나의 R 파일로 저장한 후에 필요할 때 마다 `source()` 함수를 이용해서 불러오는 것이 좋다.
- 한글이 포함된 경우, 인코딩 오류를 방지하기 위해 다음의 옵션을 사용하면 된다.

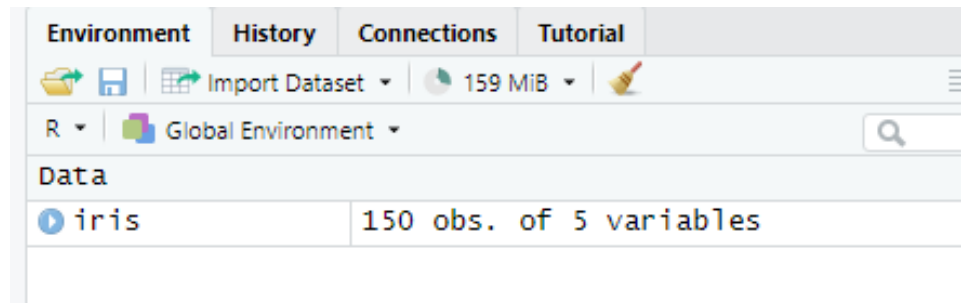
```
=>> encoding="utf-8"  
source('ADsP-script.R',encoding='utf-8')
```

```
>> setwd("C:/DEV/r-workspaces")  
>> source('ADsP-script.R',encoding='utf-8')  
>> SUM(3,5)  
[1] 8  
>>
```



프로그램 파일 실행

- RData 저장
- #iris.RData 사제 후 명령어 실행
- # datasets 패키지 불러오기
>> library(datasets)
- # iris 데이터셋 불러오기
>> data(iris)
- # iris 데이터 확인하기
>> head(iris) # 처음 6개의 행 출력
>> save(iris, file = "iris.RData")
 - RData 읽기
>> load(file="iris.RData")

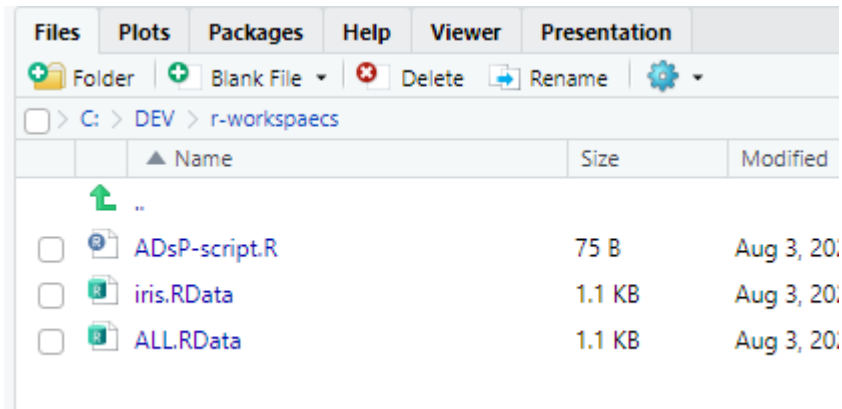


The screenshot shows the RStudio Environment pane. At the top, there are tabs for 'Environment', 'History', 'Connections', and 'Tutorial'. Below the tabs, there is a toolbar with icons for file operations and a search bar. The main area of the pane is titled 'Data' and contains a table with one row: 'iris' with a value of '150 obs. of 5 variables'.

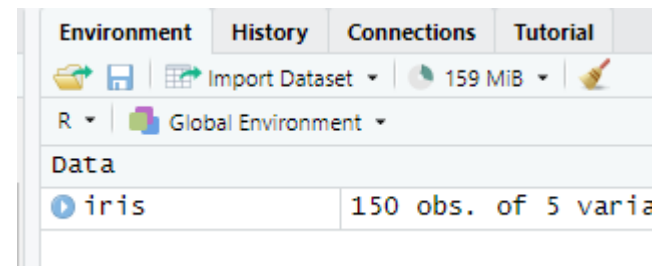
Environment	History	Connections	Tutorial
Import Dataset 159 MiB			
R Global Environment			
Data			
iris	150 obs. of 5 variables		

프로그램 파일 실행

- 작업공간의 모든 객체 저장 및 읽기
 - 하나의 데이터 뿐만 아니라, 모든 작업 내역을 그대로 저장할 수 있다. `save()` 대신 `save.image()`를 사용하면 된다.
 - ## 저장
 - `>> save.image(file = "ALL.RData")`



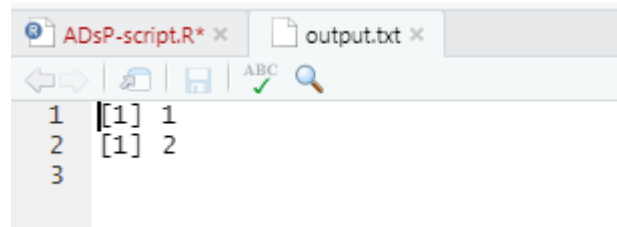
- ## 읽기
- `>> load("ALL.RData")`



프로그램 파일 실행

- `sink()` 함수는 편집 창이나 콘솔 창에서 실행한 R 코드의 결과를 콘솔 창에 출력하는 대신 외부 파일로 출력합니다. 분석한 결과값만 출력해서 정리하고 싶을 때 유용하게 사용할 수 있습니다. `sink("만들 파일이름")`으로 파일을 만든 후 코드를 실행하여 결과 출력 기록 작업을 수행하고, `sink()`로 파일 기록을 마칩니다. 파일은 역시 프로젝트의 워킹 디렉터리에 만들어집니다.

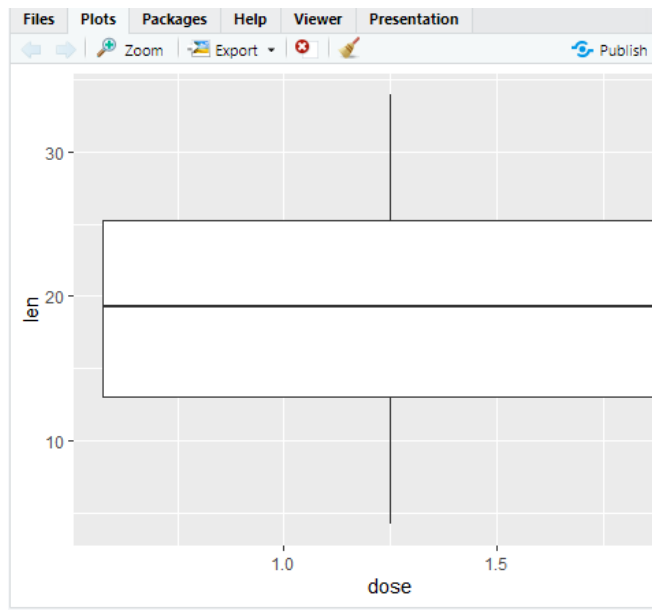
```
>> sink("output.txt")    # 출력을 output.txt에 기록 시작
>> x <- 1
>> y <- 2
>> x
>> y
>> x + y-----실행 결과를 output.txt에 기록
>> sink()                # 기록 마칩
```



프로그램 파일 실행

- ggplot2 혹은 R plot 여러개 동시에 pdf로 저장하기

```
>> library(ggplot2) #ggplot2 불러오기  
>> p <- ggplot(ToothGrowth, aes(x=dose, y=len)) + geom_boxplot()  
>> p
```



프로그램 파일 실행

- ggplot2 혹은 R plot 여러개 동시에 pdf로 저장하기

```
>> #코드를 이용하여 그래프 파일 저장하기  
>> pdf("myplot.pdf") #pdf로 출력하기
```

myplot.pdf 0 B Aug 3, 2022, 8:11 PM

```
>> p <- ggplot(ToothGrowth, aes(x=dose, y=len)) + geom_boxplot() #그래프 그리기  
>> print(p) #그림 파일로 출력하기
```

Warning message:

Continuous x aesthetic -- did you forget aes(group=...)?

```
>> dev.off() #R과 그림파일 연결 끊기
```

RStudioGD

2

```
>>
```

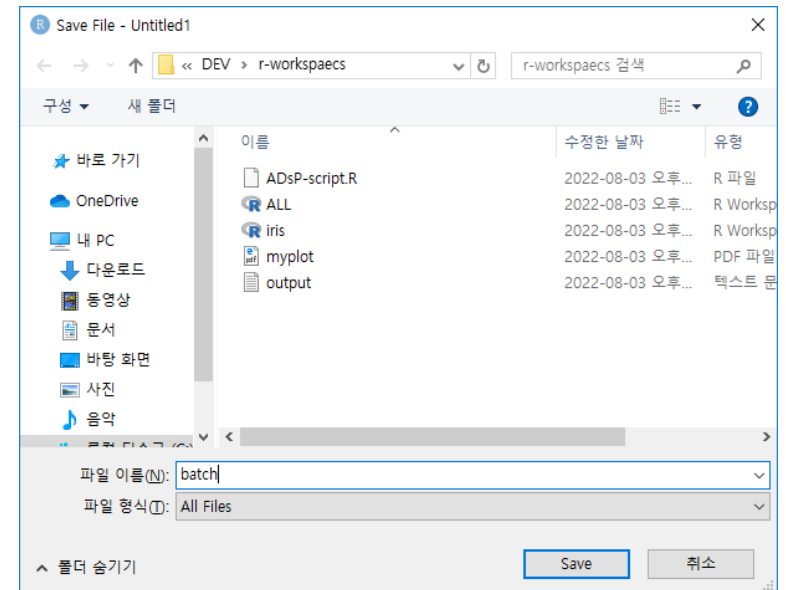
myplot.pdf 4.2 KB Aug 3, 2022, 8:13 PM

제2절 R기초

프로그램 파일 실행

- 배치 모드 기능
- 샘플 예제

```
pdf("xh.pdf")  
hist(rnorm(100))  
dev.off()
```



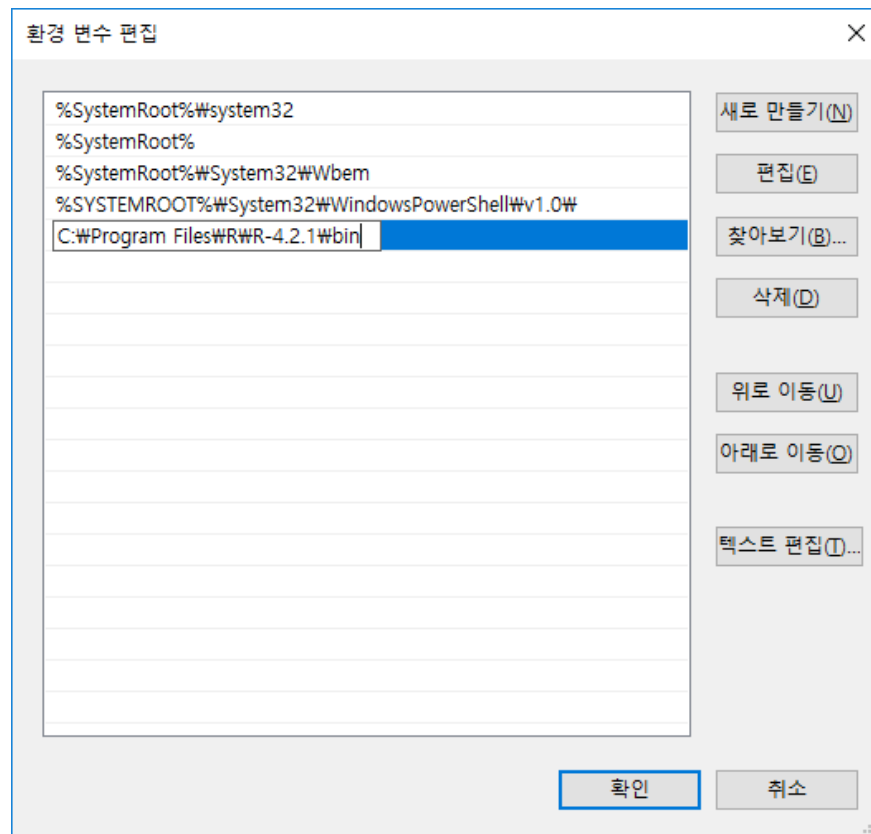
batch.R

42 B

Aug 3, 2022, 8:19 PM

프로그램 파일 실행

- 배치 모드 기능
- 배치 모드: 배치모드 방식은 사용자와 인터렉션이 필요하지 않은 방식으로 매일 돌아가야 하는 시스템에서 프로세스를 자동화할 때 유용하다.
- Path 지정



제2절 R기초

프로그램 파일 실행

- 배치파일 실행: 윈도우 창의 batch.R 실행파일이 있는 위치에서 R CMD BATCH batch.R 실행

The screenshot shows a Windows File Explorer window with the address bar set to 'C:\DEV\workspace'. The left sidebar shows the 'workspace' folder selected. The main pane displays a list of files and folders:

이름	수정된 날짜	유형	크기
ADsP-script.R	2022-08-03 오후...	R Workspace	3KB
ALL	2022-08-03 오후...	R 파일	1KB
batch.R	2022-08-03 오후...	R Workspace	2KB
batch.Rout	2022-08-03 오후...	R 파일	1KB
iris	2022-08-03 오후...	R 파일	1KB
myplot			
output			
xh			

Overlaid on the bottom right is a Command Prompt window titled 'C:\Windows\System32\cmd.exe'. It displays the following text:

```
Microsoft Windows [Version 10.0.14393]  
(c) 2016 Microsoft Corporation. All rights reserved.  
  
C:\DEV\workspace>R CMD BATCH batch.R  
  
C:\DEV\workspace>
```

In the bottom left corner, a portion of a histogram is visible, showing a frequency distribution with a peak around 10-12.

변수와 벡터 생성

- R에서 벡터, 행렬, 배열 및 요인은 데이터를 저장하고 조작하는 데 사용되는 다양한 데이터 구조입니다. 각각의 차이점을 설명해 보겠습니다.
- 벡터(Vector):
 - R에서 가장 기본적인 데이터 구조로, 단일 유형의 요소들이 모여 있는 일련의 데이터를 나타냅니다.
 - 모든 요소는 같은 데이터 유형이어야 합니다 (숫자, 문자열, 논리 값 등).
 - 예를 들어, `c(1, 2, 3, 4, 5)`는 숫자형 벡터이고, `c("apple", "banana", "orange")`는 문자열 벡터입니다.
- 행렬(Matrix):
 - 2차원 데이터 구조로, 동일한 데이터 유형의 요소로 구성된 행과 열로 이루어집니다.
 - `matrix()` 함수를 사용하여 생성하며, `rbind()`나 `cbind()` 등을 통해 행과 열을 합칠 수 있습니다.
 - 예를 들어, 2x3 크기의 행렬은 다음과 같이 생성할 수 있습니다:
`matrix(1:6, nrow = 2, ncol = 3).`

변수와 벡터 생성

● 배열(Array):

- 벡터나 행렬의 일반화된 형태로, 다차원 데이터 구조를 표현합니다.
- 모든 요소는 동일한 데이터 유형이어야 합니다.
- `array()` 함수를 사용하여 생성하며, 차원(dimension) 파라미터를 통해 다차원 배열을 만들 수 있습니다.
- 예를 들어, 2x3x4 크기의 배열은 다음과 같이 생성할 수 있습니다: `array(1:24, dim = c(2, 3, 4))`.

● 요인(Factor):

- 범주형(categorical) 데이터를 나타내는 데 사용됩니다.
- 범주(category)들을 정의하고, 해당 데이터가 어떤 범주에 속하는지를 저장합니다.
- 주어진 범주들의 수와 각 범주의 레벨(level)을 정의하여 생성됩니다.
- 예를 들어, `factor(c("male", "female", "male", "male"), levels = c("male", "female"))`는 "male"과 "female"이라는 범주를 갖는 요인입니다.

변수와 벡터 생성

- R 데이터 유형과 객체
- 숫자: integer, double
- 논리값: True(T), False(F)
- 문자: "a", "abc"

제2절 R기초

R코딩스타일

1. 구글의 R 스타일 가이드(<https://goo.gl/rAQXnt>)
2. 해들리 위컴의 스타일 가이드(<https://goo.gl/kWjlhw>)

← → google.github.io/styleguide/Rguide.html

styleguide

Google's R Style Guide

R is a high-level programming language used primarily for statistical computing and graphics. The goal of the R Programming Style is to make our R code easier to read, share, and verify.

The Google R Style Guide is a fork of the [Tidyverse Style Guide](#) by Hadley Wickham [license](#). Google modifications were developed in collaboration with the internal R user community. The rest of this document explains Google's primary differences with the Tidyverse guide, and why these differences exist.

Syntax

Naming conventions

Google prefers identifying functions with `BigCamelCase` to clea

```
# Good
DoNothing <- function() {
  return(invisible(NULL))
}
```

The names of private functions should begin with a dot. This he

```
# Good
..DoNothingPrivately <- function() {
```

← → 주의 요람 | adv-r.had.co.nz/Style.html

Advanced R by Hadley Wickham

Table of contents

Want a physical copy of the second edition of this material? [Buy a book from Amazon!](#)

Contents

Notation and naming
Syntax
Organisation

You're reading the first edition of *Advanced R*; for the second edition, the style guide has been replaced by [the tidyverse style guide](#).

Style guide

Good coding style is like using correct punctuation. You can manage without it, but it sure makes things easier to read. As with styles of punctuation, there are many possible variations. The following guide describes the style that I use (in this book and elsewhere). It is based on Google's [R style guide](#), with a few tweaks. You don't have to use my style, but you really should use a consistent style.

Good style is important because while your code only has one author, it'll usually have multiple readers. This is especially true when you're writing code with others. In that case, it's a good idea to agree on a common style up-front. Since no style is strictly better than another, working with others may mean that you'll need to sacrifice some preferred aspects of your style.

The `formatR` package, by Yihui Xie, makes it easier to clean up poorly formatted code. It can't do everything, but it can quickly get your code from terrible to pretty good. Make sure to read [the introduction](#) before using it.

Notation and naming

File names

File names should be meaningful and end in `.R`.

```
# Good
fit-models.R
utility-functions.R
```

R 기초 중에 기초

● 출력하기

- `print()`: 출력 형식을 지정할 필요 없음, 한 번에 하나의 객체만 출력
- `cat()`: 여러 항목을 묶어서 연결된 결과로 출력, 복합적 데이터 구조(행렬, list 등)을 출력할 수 없음
- 커맨드 프롬프트에 변수나 표현식을 입력
- [ex] `print(a)`, `print("A","B","C")`

● 변수에 값 할당하기(대입 연산자)

- `<-`, `<<-`, `=`, `->>`

● 변수 목록보기

- `ls()`, `ls.str()`

R 기초 중에 기초

```
> ls()
[1] "a"          "b"          "df_cols"    "df_rows"    "df1"        "df2"
"hflights_df" "mtcars"
[9] "name"       "num"        "student"    "student2"    "student3"    "x"
"y"
> ls.str()
a :  num [1:32] 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
b :  num [1:32] 110 110 93 110 175 105 245 62 95 123 ...
df_cols : 'data.frame': 5 obs. of 4 variables:
 $ x...1: int 1 2 3 4 5
 $ y...2: num 2.423 -1.835 0.229 -0.451 -1.992
 $ x...3: int 6 7 8 9 10
 $ y...4: num 0.399 -0.192 -0.158 -1.328 0.207
df_rows : 'data.frame': 10 obs. of 2 variables:
```

R 기초 중에 기초

● 변수 삭제하기

- `rm()`
- `rm(list=ls())`: 모든 변수를 삭제할 때 사용

● 벡터 생성하기

- `c()`: 벡터의 원소 중 하나라도 문자가 있으면 모든 원소의 모드는 문자형으로 변환됨

● R함수 정의하기

- `fuction(매개변수 1, 매겨변수2,,,,,배개 변수 n){expr1, expr2,,,,exprn}`
- 지역변수: 단순히 값을 대입하기만 하면 지역변수로 생성되고, 함수가 종료되면 지역변수는 삭제됨
- 조건문: `if`문 / 반복문 `for`, `while`, `repeat` 문 / 전역 변수: `<<-` 를 사용하여 전역 변수로 지정할 수 있지만 추천하지 않음

R 프로그램 소개

- 데이터 할당

- `a<-1, a=1`

- 화면 프린트

- `a, print(a) ==>>` 프린트 함수

- 결합

- `x<-c(1,2,3,4) / x<-c(6.25,3.14,5.18) / x<-c("fee","fie","fun") / x<-c(x,y,z)`
- C 함수는 문자, 숫자, 논리 값, 변수를 모두 결합 가능하며 벡터와 데이터 셋을 생성 가능

R 프로그램 소개

• 스칼라(단일값)

```
> a <- 3
> a
[1] 3
> print(a)
[1] 3
> b <- 4.5
> c <- a + b
> print(c)
[1] 7.5
```

• 벡터(다중값)

```
> z = c(1, 2, 3)
> print(z)
[1] 1 2 3
> mean(z)
[1] 2
```

• NA 확인

```
> one <- 100
> two <- 75
> three <- 80
> four <- NA
> is.na(one)
[1] FALSE
> is.na(four)
[1] TRUE
```

스칼라(Scalar)

- 스칼라(Scalar)란 단일 차원의 값을 뜻하는 것으로 숫자 1, 2, 3, ...을 예로 들 수 있다. R에서 데이터 타입의 기본은 벡터(Vector)이므로 스칼라 데이터는 길이가 1인 벡터와 같은 것으로 볼 수 있다.
- 숫자(Numeric) : 정수, 부동소수 등
- NA(Not Available) : 데이터 값이 없음을 의미, 결측값
- NULL : NULL 객체를 뜻하며, 데이터 유형과 길이가 "0"인 비어있는 값
- NaN(not a Number) : 연산이 잘못된 입력을 받아 값이 정상적인 숫자가 아님을 의미
- 문자열 : 문자에 대한 데이터 타입
- 진릿값 : TRUE/T는 참값을, FALSE/F는 거짓값을 의미
- 요인(Factor) : 범주형(Categorical) 데이터를 표현하는 데이터 타입, 요인이 가지는 값의 목록을 수준(level)이라고 함

벡터(Vector)

- 벡터(Vector)는 배열의 개념으로, 한 가지 데이터 타입의 데이터를 저장할 수 있다. 예를 들어, 숫자만 저장하는 배열, 문자열만 저장하는 배열이 벡터에 해당한다.

R 프로그램 소개

• 범주형 데이터 저장

```
> gender <- factor("m", c("m","f"))
```

```
> gender
```

```
[1] m  
Levels: m f
```

```
> gender2 <- factor("성별", c("남성","여성"))
```

```
> gender2
```

```
[1] <NA>  
Levels: 남성 여성
```

```
> season <- factor("season", c("spring","summer","fall","winter"))
```

```
> season
```

```
[1] <NA>  
Levels: spring summer fall winter
```

```
> nlevels(season)
```

```
[1] 4
```

```
> levels(season)
```

```
[1] "spring" "summer" "fall" "winter"
```

```
> str(iris)
```

```
'data.frame': 150 obs. of 5 variables:  
 $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...  
 $ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...  
 $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...  
 $ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...  
 $ Species : Factor w/ 3 levels "setosa","versicolor",...: 1 1 1 1 1 1 1 1 1 1 ...
```

```
> nlevels(iris$Species)
```

```
[1] 3
```

```
> levels(iris$Species)
```

```
[1] "setosa" "versicolor" "virginica"
```

- 첫줄의 factor() 는 범주형 변수 값을 표현하기 위해 호출되었다. factor() 에 주어진 첫번째 인자는 'gender' 변수에 저장되는 값으로서 여기서는 m이다. 그리고 이 범주형 변수는 c("m","f")에 표현된 바와같이 2개의 값 m과 f가 가능하다. 다시 말해 gender 에는 m, f 의 2개 값을 갖는 범주에서 m이 선택되어 저장되었다.
- ※ c() 의 표현이 생소한데 이것은 벡터를 표현하는 방식으로 참고로 c() 함수는 합치다라는 뜻의 'Combine'의 머리글자입니다.

R 프로그램 소개

● 수열

- `1:5` / `9:-2` / `seq(from=0, to=20,by=2)` / `seq(from=0, to=20,length.out=5)`
- 콜론, `seq` 함수를 사용하여 시작 값~최종 값까지의 연속 숫자 생성, `seq` 함수는 간격과 결과값의 길이를 제한 가능

```
>> 1:5
[1] 1 2 3 4 5
>> 9:-2
[1] 9 8 7 6 5 4 3 2 1 0 -1 -2
>> seq(from=0, to=20, b=2)
[1] 0 2 4 6 8 10 12 14 16 18 20
>>
```

● 반복

- `rep(1,time=5)` / `rep(1:4, each=2)`, `rep(c,each=2)`
- `rep` 함수는 숫자나 변수의 값들을 `time` 인자에 지정한 횟수만큼 반복

```
>> rep(1,time=5)
[1] 1 1 1 1 1
>> rep(1:4, each=2)
[1] 1 1 2 2 3 3 4 4
```

R 프로그램 소개

● 문자 붙이기

- `A<-paste("a","b","c",sep="-") / paste(A,c("e","f")) / paste(A,10,sep="")`

```
>> A<-paste("a","b","c",sep="-")
```

```
>> paste(A, c("e","f"))
```

```
[1] "a-b-c e" "a-b-c f"
```

```
>> paste(A,10,sep="")
```

```
[1] "a-b-c10"
```

```
>>
```

● 문자열 추출

- `substr("Bigdataanalysis",1,4)`
- `substr(문자열, 시작점, 끝점)`함수는 문자열의 특정 부분을 추출 가능

● 논리 값

- `a<-True / a<-T / b<-False / b<-F (T=True, F=False로 인식)`

R 프로그램 소개

- 논리 연산자

- `==`: 같다 / `!=`: 같지않다 / `<`, `<=`: 작다, 작거나 같다 / `>`, `>=`: 크다, 크거나 같다

- 벡터의 원소 선택하기

- `V[n]`: 선택하고자 하는 자리수 / `V[-n]`: 제외하고자 하는 자리수 / `n`: 원소의 자리수, `V`: 벡터의 이름

R의 연산자

- 산술 연산자[Arithmetic Operator]
 - 덧셈, 뺄셈, 곱셈, 나눗셈 등의 산술식을 구성하는 연산자

연산자	설명	입력내용	결과내용
+	더하기	3+4	7
-	빼기	3-4	-1
*	곱하기	3*4	12
/	나누기	3/4	0.75
, ^	거듭제곱	34, 3^4	81
%%/%	몫	33%%/4	8
%%	나머지	33%%4	1

R의 연산자

- 비교 연산자[Relational Operator]
 - 두 개의 값을 비교하여 맞으면 TRUE, 틀리면, FALSE를 반환하는 연산자

연산자	설명	입력내용	결과내용
>>	크다	3>> 4	FALSE
>> =	크거나 같다.	3>> =4	FALSE
<	작다	3<4	TRUE
<=	작거나 같다.	3<=4	TRUE
==	같다.	3==4	FALSE
!=	같지 않다.	3!=4	TRUE

R의 연산자

- 할당 연산자[Allocation Operator]
 - 어떤 객체의 이름[변수이름, 데이터이름]에 특정한 값을 지정할 때 사용하는 연산자

연산자	설명	입력내용	결과내용
<-, <<-	오른쪽의 값을 왼쪽 이름에 저장	x<-3 , x<<-3	R STUDIO 왼쪽 하단에 x라는 변수를 생성 전부 x의 값이 3임을 알려준다.
=	오른쪽의 값을 왼쪽 이름에 저장	x=3	
->> , ->>>	왼쪽의 값을 오른쪽의 이름에 저장	3->> x, 3->>> x	

R의 연산자

- 논리 연산자[Logical Operator]
- 논리곱(AND), 논리합(OR), 논리부정

연산자	설명	입력내용	결과내용
&	AND	(조건1) & (조건2)	모든 조건이 참일때만
&&	AND	(조건1) && (조건2)	최종적인 결과가 TRUE
	OR	(조건1) (조건2)	조건 중 하나라도 참이면
	OR	(조건1) (조건2)	최종적인 결과가 TRUE
!	부정	!((조건1)==(조건2))	조건1과 조건2가 동일하면 FALSE 조건1과 조건2가 동일하면 TURE

- Vector[벡터]를 사용할 경우 &와 && / |와 ||결과에 차이가 존재

&	데이터가 하나인 경우나 데이터가 두개 이상인 경우
&&	데이터가 하나인 경우에만 가능 → 벡터인 경우, Vector의 첫번째만 작동하고 나머지는 작동하지 않음
	데이터가 하나인 경우나 데이터가 두개 이상인 경우
	데이터가 하나인 경우에만 가능 → 벡터인 경우, Vector의 첫번째만 작동하고 나머지는 작동하지 않음

백터와 연산

연산자 우선순위	뜻	사용법
[[인덱스	a[1]
\$	요소 뽑아내기, 슬롯 뽑아내기	a\$coef
^	지수	5^3
-, +	단항 마이너스와 플러스 부호	-3, +5
:	수열 생성	1:10
%any%	특수 연산자	%%/: 나눗셈 몫, %% 나눗셈 나머지, %*% 행렬의 곱
* /	곱하기, 나누기	3*3, 3/5
+, -	더하기, 빼기	3+2, 3-1
==, !=, <>, <=, >=	비교	3==5
!	논리부정	!(3==5)
&	논리, "and", 단축(short-circuit) "and"	TRUE&TRUE
	논리, "or", 단축(short-circuit) "or"	TRUE TRUE
~	식(formula)	lm(log(brain)~log(body).data=Animals)
->, ->> >>	오른쪽 대입	3->> a
=	대입(오른쪽을 왼쪽으로)	a=4
<-, <<-	대입(오른쪽을 왼쪽으로)	a<-4
?	도움말	?lm

R의 데이터 유형

● 기본적인 데이터 유형 4가지

문자형[Character]	하나의 문자 또는 문자열로 되어 있으며 ' ' 또는 " "로 묶여 있다.
복소수형[Complex]	실수와 허수로 이루어진 복소수
수치형[Numeric]	숫자로 되어 있으며, 정수형[Integer]과 실수형[Double]이 있음
논리형[Logical]	참과 거짓의 논리값으로 TRUE나 FALSE를 가짐

● 특수한 형태의 데이터 유형

Null	존재하지 않는 객체로 지정할 때 사용
NA	Not Available의 약자로 결측치[Missing Value]를 의미
NaN	Not available Number의 약자로 수학적으로 계산이 불가능한 수를 의미 ex>> sqrt(-5) : 음수에 대한 제곱근을 구할 수 없으므로 NaN로 표시됨
Inf	Infinite의 약자로 양의 무한대를 의미 음의 무한대는 "-Inf"로 표기

R의 데이터 유형

● 날짜형 데이터

<code>Sys.Date()</code>	오늘 날짜 표시
<code>Sys.time()</code>	오늘 날짜&시간 표시
<code>date()</code>	오늘 요일&날짜&시간 표시

- 문자형 날짜를 날짜형 데이터로 변경할 때는 `as.Date()`, `strptime()` 사용
- "YYYY-mm-dd"가 날짜형 데이터의 기본형식
- 표준서식이 아닌 문자형의 경우 뒤에서 `format`을 정해줘야 함
- "format = "은 생략해도 무방
- `format()`을 사용하면 원하는 형식으로 출력가능
 - %Y : 4자리 연도
 - %y : 2자리 연도
 - %m : 월
 - %b : 월
 - %d : 일
 - %a : 요일

R의 데이터 유형

● 날짜형 데이터

```
> Sys.Date()
[1] "2023-12-16"
> Sys.time()
[1] "2023-12-16 16:44:58 KST"
> date()
[1] "Sat Dec 16 16:44:59 2023"
> format(Sys.Date(), "%Y")
[1] "2023"
> format(Sys.Date(), "%y")
[1] "23"
> format(Sys.Date(), "%m")
[1] "12"
> format(Sys.Date(), "%b")
[1] "12"
> format(Sys.Date(), "%d")
[1] "16"
> format(Sys.Date(), "%a")
[1] "토"
```

데이터의 유형을 알려주는 명령어

- **mode()함수**
 - 데이터가 가지고 있는 유형을 문자형 형태로 알려주는 함수

문자형 데이터인 경우 "Character" 출력

복소수형 데이터인 경우 "Complex" 출력

수치형 데이터인 경우 "Numeric" 출력

논리형 데이터인 경우 " Logical" 출력

데이터의 유형을 알려주는 명령어

- is. 함수
 - 지정한 데이터 유형인지 아닌지를 알려주는 함수

함수명	설명	결과내용
is.character()	문자형 여부	TRUE or FALSE
is.complex()	복소수형 여부	
is.numeric()	수치형 여부	
is.integer()	정수형 여부	
is.double()	실수형 여부	
is.logical()	논리형 여부	
is.null()	Null 여부	
is.na()	NA 여부	
is.finite()	유한수치 여부	
is.infinite()	무한수치 여부	

데이터 유형의 변경

- as. 함수 사용시 우선순위가 낮은 유형에서 높은 유형으로 강제적으로 변경 가능
- 단, 우선순위가 높은 유형에서 우선순위가 낮은 유형으로의 변환은 일부만 가능

연산자	설명	결과내용
as.character	문자형으로 변환	
as.complex()	복소수형으로 변환	
as.numeric()	수치형으로 변환	
as.integer()	정수형으로 변환	변환되거나 NA
as.double()	실수형으로 변환	
as.logical()	논리형으로 변환	
as.date()	날짜형으로 변환 format 옵션을 통해 형식지정 가능	

R의 기초 함수

<code>data()</code>	데이터셋을 불러들임
<code>summary()</code>	데이터셋 변수 내용을 요약
<code>print()</code>	출력형식을 지정할 필요 없음. 한번에 하나의 객체만 출력
<code>cat()</code>	여러 항목을 묶어서 연결된 결과로 출력. 복합적 데이터 구조(행렬, list 등)를 출력할 수 없음
<code>rm()</code>	변수 삭제하기 * <code>rm(list=is())</code> : 모든 변수를 삭제할 때 사용
<code>lm()</code>	<code>lm(formula,data)</code> 의 형태 * <code>formula</code> 는 주로 <code>y[종속변수]~model[독립변수]</code> 형태로 사용 * <code>data=(data명)</code> 으로 원하는 데이터를 활용
<code>c()</code>	여러 개의 벡터들을 결합하는 함수 문자, 숫자, 논리값, 변수를 모두 결합 가능하며 벡터와 데이터셋을 생성가능 벡터의 원소 중 문자가 하나라도 있을 경우, 모든 원소의 모드는 문자형태로 변환
<code>seq()</code>	수치형에만 적용가능한 벡터 생성 방법
<code>is()</code> / <code>is.str()</code>	변수 목록을 확인
<code>function()</code>	사용자 지정함수[사용자가 원하는 함수를 정의할 수 있다. 인수를 이용한 함수, 인수가 없는 함수를 만들 수 있다.
<code>q()</code>	작업 종료
<code>setwd()</code>	워킹 디렉터리 지정

R의 기초 통계 함수

기 능

비고

mean(변수)

변수의 **평균** 산출

sum(변수)

변수의 **합계** 산출

median(변수)

변수의 **중앙값** 산출

log(변수)

변수의 **로그값** 산출

sd(변수)

변수의 **표준편차** 산출
 $= \text{sqrt}(\text{var}(\text{변수})) = \text{var}(\text{변수})^{(1/2)}$

var(변수)

변수의 **분산** 산출

cov(변수1, 변수2)

변수간의 **공분산** 산출

cor(변수1, 변수2)

변수간 **상관계수** 산출

length(변수)

변수간 길이를 값으로 출력

R 프로그램 주요 키

#[해시기호]	한 줄을 주석[Comment]으로 처리하는 기능 다음 줄에도 주석처리하고 싶을 경우 해당 줄 앞에 #를 단다. 주석은 사용자가 설명을 달아주는 기능을 하므로 따로 문법 검사를 하지 않는다. 해시기호는 스크립트를 짜는 명령들이 어떤 기능을 하는지 설명하고 싶을 때 사용
:[세미콜론]	세미콜론은 하나의 명령어가 끝났음을 알려주는 기능 만약, 한 줄에 하나의 명령 밖에 없다면 세미콜론을 하지 않아도 오류없이 실행됨 단, 하나가 아닌 여러 개의 명령어를 한 줄에 입력할 때 반드시 중간에 세미콜론 사용 * 콜론(:)과 혼동하지 말 것!!
: [콜론]	시작값에서 최종값까지 1씩 변화하는 연속적인 숫자를 생성하는 기능 seq() 함수와 비슷하나 seq()와 달리 간격과 결과값의 길이 제한이 불가능하다.
Enter[엔터]	엔터는 명령어 입력 시, 다음 줄로 이동시켜주는 기능
Ctrl + Enter	스크립트에 작성한 명령어를 실행하는 기능 명령어가 한 줄일 때, 마우스 위치에 상관없이 아무 곳에서 눌러도 명령이 모두 실행됨 두 줄 이상일 땐, 전체 명령어를 블록 잡고 실행 * R STUDIO가 아닌 RGui에서는 Ctrl + R을 눌러야 실행됨
Shift + Enter	명령어가 긴 경우, 명령어 일부를 다음 줄로 넘겨 깔끔하게 표현하는 기능
\n[\ n]	줄바꿈 기호 * cat('평균=', avg, '\ n')
대소문자	R은 대소문자를 구별하므로 만약 명령 실행에 에러가 있다면 대소문자를 헛갈리지 않았는지 확인할 것 폴더이름은 항상 대문자로 설정할 것
[]	Index[인덱스]로 원하는 벡터의 원소 선택 가능 (벡터의 이름)[n] : 지정된 벡터에서 n번째의 원소를 추출 (벡터의 이름)[-n] : 지정된 벡터에서 n번째의 원소를 제외하고 추출 여러개의 원소를 추출하고 싶을 때는 c() 함수를 활용한다.
\$	벡터, 리스트 등에서 원하는 부분을 뽑아내는 함수 결과값은 벡터 matrix에서는 잘 되지 않으며 data.frame에서 주로 사용
?	도움말 함수 앞에 붙일 경우, 함수의 argument들을 알려준다.

R Studio에서만 적용가능한 유용한 단축키

Alt + -	'<-'를 입력할 때 사용하면 편리
Ctrl + 2	Script 창에서 Console창으로 이동
Ctrl + 1	Console창에서 Script창으로 이동

R 프로그래밍 시 자주 하는 실수

기능	기능 / 내용
함수를 불러오고 괄호닫기	function 함수에서의 {}, 함수의 ()
윈도우 파일 경로에서 역슬레시 두 번씩 쓰기	f:\dataedu\r\test.csv => f:\dataedurtest.csv 로 인식함 역슬레시(\)를 두 번 쓰거나, 슬레시(/)를 한번 써야함
<-사이를 붙여쓰기	>> X< -pi Error: object 'X' not found
여러줄 넘어서 식을 계속 넘어갈 때	>> sum<-1+2+3 >> +4+5 [1] 9 >> sum [1] 6
==대신 =을 사용하지 말것	==는 비교연산자 /=은 대입연산자
1:(n+1)대신 1:n+1로 쓰지 말것	>> n<-5; >> 1:n+1 [1] 2 3 4 5 6 >> 1:(n+1) [1] 1 2 3 4 5 6
패키지를 불러오고 library()나 require()을 수행할 것	
2번 써야할 것과 1번 써야 할 것을 혼돈하지 말 것	aList[[a]] vs aList[a] / && vs & / vs 등
인자의 개수를 정확히 사용할 것	>> a<-c(9,10,11); >> mean(a) [1] 10 >> mean(9,10,11) [1] 9

학습목표

- R에서의 다양한 입력과 출력방식을 이해한다.
- 다양한 구조와 형식의 외부데이터를 읽어 들일 수 있다.
- 웹에서 데이터 테이블의 데이터를 읽어 들일 수 있다.
- 복잡한 구조의 데이터 파일을 읽어 들일 수 있다.

눈높이 체크

- 데이터 입력과 출력이 가능한 외부 데이터에 대해 알고 계신가요?
- 데이터의 구조와 형태가 어떤 것이 있는지 알고 계십니까?
- 웹에서 테이블 형태의 데이터, 복잡한 구조의 데이터도 R에서 불러 들일 수 있을까요?

데이터 분석과정

- 분석자가 분석 목적에 맞게 적절한 분석방법론을 선택해서 정확한 분석을 통해 통찰력을 가지고 해석함으로써 분석과정을 마치게 된다.
- 이렇게 데이터를 분석하기 위해서는 분석자가 분석을 위해 설계된 방향으로 데이터를 정확하게 입력 받는 것에서부터 시작될 수 있다.
- 그리고 입력된 데이터는 다양한 전처리 작업을 거쳐 분석이 가능한 형태로 재정리됩니다. 우리는 이것을 데이터 핸들링이라고도 한다.
- 또한 분석된 결과를 이해하기 쉽고 잘 해석될 수 있도록 생산하는 부분이 데이터 출력이라고 할 수 있다. 출력된 결과를 보고서 형태로 정리되어 최종 의사결정자와 고객에게 전달되게 됨으로써 통계 분석과정은 종료된다고 할 수 있다.

R에서의 데이터 입력과 출력

- R에서 처리할 수 있는 데이터 타입은 아래와 같다.
- R에서 다룰 수 있는 파일 타입: Tab-delimited text, Comma-separated text, Excel file, JSON file, HTML/XML file, Database, (Other) Statistical SW file

기능	기능 / 내용
키보드로 데이터를 입력	1) 데이터 양이 적어 직접 입력 - c():combine 함수 2) 데이터 편집기 활용하기: 빈데이터프레임 생성->편집기를 불러와서 편집 & 데이터프레임에 덮어 씌우기
출력할 내용의 자리수 정의	R 부동 소수점 표현: 7자리 표시 print(pi,digits=num) / cat(format(pi,,digits=num),"\n") / options(digits=num)
파일에 출력하기	cat("출력할 내용","변수","\n",file="파일이름",append=T) sink("파일이름") ...출력할내용... sink()
파일 목록보기	list.files() / list.files(recursive=T,all.files=T)

R에서의 데이터 입력과 출력

- R에서 처리할 수 있는 데이터 타입은 아래와 같다.

기능	기능 / 내용
Cannot Open File 해결하기	파일위치: c:\data\sample.txt -> R에서는 c"datasample.txt라고 인식함 [방법1] 역슬래시를 슬래시로 바꾼다. c:/data/sample.txt [방법2] 역슬래시를 두 번 쓴다. c:\\data\\sample.txt
고정자리수 데이터파일 읽기	<code>read.fwf("파일이름",widths=c(w1,w2,...,wn))</code>
테이블로 된 데이터파일 읽기 (변수구분자 포함)	<code>read.table("파일이름",sep="구분자")</code> [주의1] 주소, 이름, 성 등의 텍스트를 요인으로 인식함 <code>read.table("파일이름",sep="구분자",stringsAsFactor=F)</code> [주의2] 결측치를 NA가 아닌 다른 문자로 표현할 때 <code>read.table("파일이름",sep="구분자",na.strings=".")</code> [주의3] 파일의 첫 행을 변수명으로 인식하고자 할 때 <code>read.table("파일이름",sep="구분자",header=T)</code>
CSV데이터파일 읽기 (변수구분자 쉼표)	<code>read.csv("파일이름",header=T)</code> [주의1] 주소, 이름, 성 등의 텍스트를 요인으로 인식함 <code>read.csv("파일이름",header=T,as.is=T)</code>

R에서의 데이터 입력과 출력

- R에서 처리할 수 있는 데이터 타입은 아래와 같다.

기능	기능 / 내용
csv 데이터파일로 출력 (변수구분자 쉼표)	<pre>write.csv(행렬, 또는 데이터 프레임, "파일이름", row.names=F)</pre> <p>[주의1] 1행이 변수명으로 자동인식하지만 변수명이 아닐 경우</p> <pre>write.csv(dfm, "파일이름", col.names=F)</pre> <p>[주의2] 1열에 레코드 번호를 자동 생성하지만 레코드 번호를 생성하지 않을 경우</p> <pre>write.csv(dfm, "파일이름", row.names=F)</pre>
웹에서 데이터파일을 읽어올 때 (변수구분자 쉼표)	<pre>read.csv("http://www.example.com/download/data.csv")</pre> <pre>read.table("http://www.example.com/download/data.txt")</pre> <p>what=numeric(0): 토큰을 숫자로 해석 / what=integer(0): 토큰을 정수로 해석 what=complex(0): 토큰을 복소수로 해석 / what=character(0): 토큰을 문자로 해석 what=logical(0): 토큰을 논리값으로 해석</p>
html에서 테이블 읽어올 때	<pre>library(XML)</pre> <pre>url<-"http://www.example.com/data/table.html"</pre> <pre>t<-readHTMLTable(url)</pre>
복잡한 구조의 파일 (웹테이블) 읽어오 기	<pre>lines<-readLines("a.txt", n=num)</pre> <pre>token<-scan("a.txt", what=numeric(0))</pre> <pre>token<-scan("a.txt", what=list(v1=character(0), v2=numeric(0)))</pre> <pre>token<-scan("a.txt", what=list(v1=character(0), v2=numeric(0), n=num, nlines=num, skip=num, na.strings=list))</pre>

데이터 넣고 꺼내기

- 데이터를 넣고 꺼내는 방법

```
> x <- c("a", "b", "c")
```

```
> print(x)
```

```
[1] "a" "b" "c"
```

```
> x[1:2]
```

```
[1] "a" "b"
```

```
> x[2:3]
```

```
[1] "b" "c"
```

❖ 리스트는 `list(키=값, 키=값, ...)` 형태로 데이터를 나열해 정의한다.

```
> x <- list(name="foo", height=70)
```

```
> x
```

```
$name
```

```
[1] "foo"
```

```
$height
```

```
[1] 70
```

```
> x$name
```

```
[1] "foo"
```

```
> x[[1]]
```

```
[1] "foo"
```

```
>
```

데이터 넣고 꺼내기

- 데이터 프레임은 행렬과 마찬가지로의 모습을 하고 있지만 행렬과 달리 다양한 변수, 관측치(observations), 범주 등을 표현하기 위해 특화되어있습니다.

```
> d <- data.frame(x=c(1, 2, 3, 4, 5), y=c(6,7,8,9,10))
```

```
> d
```

```
  x y  
1 1 6  
2 2 7  
3 3 8  
4 4 9  
5 5 10
```

```
> d$x
```

```
[1] 1 2 3 4 5
```

```
> d$y
```

```
[1] 6 7 8 9 10
```

```
> d[1,]
```

```
  x y  
1 1 6
```

```
> d[,1]
```

```
[1] 1 2 3 4 5
```

- 벡터로 인덱스를 지정하거나, 또는 제외할 행 또는 열을 - 로 표시할 수 있다.

```
> d[c(1,3),2]
```

```
[1] 6 8
```

```
> d[-1,-c(2,3)]
```

```
[1] 2 3 4 5
```

```
> d[-1,-c(2)]
```

```
[1] 2 3 4 5
```

```
>
```



제3절 입력과 출력

데이터 불러오기 - 키보드로 숫자 입력하기

```
> num <- scan()
```

```
1: 1  
2: 2  
3: 3  
4: 4  
5: 5  
6:  
Read 5 items
```

```
> num
```

```
[1] 1 2 3 4 5
```

```
> sum(num)
```

```
[1] 15
```

```
> # 실습: 키보드로 문자 입력하기
```

```
> name <- scan(what = character())
```

```
1: 홍길동  
2: 박문수  
3: 이순신  
4: 황진희  
5:  
Read 4 items
```

```
> name
```

```
[1] "홍길동" "박문수" "이순신" "황진희"
```

데이터 불러오기- read.csv

getwd()

```
[1] "C:/DEV/r-workspaces/data"
```

```
> setwd("C:/DEV/r-workspaces/data")
```

```
> student <- read.table(file = "student.txt")
```

```
> student
```

```
  V1  V2  V3 V4  
1 101 hong 175 65  
2 201 lee 185 85  
3 301 kim 173 60  
4 401 park 180 70
```

```
> names(student) <- c("번호", "이름", "키", "몸무게")
```

```
> student
```

```
번호 이름 키 몸무게  
1 101 hong 175 65  
2 201 lee 185 85  
3 301 kim 173 60  
4 401 park 180 70
```

```
>
```

```
> student2 <- read.csv(file = "student4.txt",sep = ",",na.strings = "-",fileEncoding = "euc-kr")
```

```
> student2
```

```
번호 이름 키 몸무게  
1 101 hong 175 65  
2 201 lee 185 85  
3 301 kim 173 NA  
4 401 park NA 70
```

```
>
```

제3절 입력과 출력

데이터 불러오기- read.csv

```
> install.packages("readxl")
```

Error in install.packages : Updating loaded packages

Restarting R session...

```
> install.packages("readxl")
```

Installing package into '/home/k8s/R/x86_64-pc-linux-gnu-library/3.6'

(as 'lib' is unspecified)

URL 'https://cloud.r-project.org/src/contrib/readxl_1.4.0.tar.gz'을 시도합니다

Content type 'application/x-gzip' length 2089229 bytes (2.0 MB)

=====

downloaded 2.0 MB

* installing *source* package 'readxl' ...

...

* DONE (readxl)

The downloaded source packages are in

'/tmp/Rtmpi6F5Cq/downloaded_packages'

```
>
```

```
> student3 <- read_excel(path = "studentexcel.xlsx",sheet="student")
```

```
> student3
```

A tibble: 5 × 4

학번 이름 성적 평가

<dbl> <chr> <dbl> <chr>

1	101	홍길동	80 B
2	102	이순신	95 A+
3	103	강감찬	78 C+
4	104	유관순	85 B+
5	105	김유신	65 D+

데이터 저장하기

```
> # txt 파일에 저장
> write.table(student, "studentw.txt", row.names = FALSE)
>
> # csv 파일에 저장
> write.csv(student2, "student2w.csv", row.names = F, quote = F)
>
> # excel 파일에 저장
> install.packages("writexl")
Installing package into '/home/k8s/R/x86_64-pc-linux-gnu-library/3.6'
(as 'lib' is unspecified)
URL 'https://cloud.r-project.org/src/contrib/writexl_1.4.0.tar.gz'을 시도합니다
Content type 'application/x-gzip' length 258516 bytes (252 KB)
=====
downloaded 252 KB

...
* DONE (writexl)

The downloaded source packages are in
  '/tmp/Rtmpi6F5Cq/downloaded_packages'
> library(writexl)
> write_xlsx(x=student3, path="student3w.xlsx", col_names = TRUE)
>
```



제4절 데이터 구조와 데이터 프레임

학습목표

- 데이터 구조 중 벡터와 리스트 구조의 차이를 구분할 수 있다.
- 데이터 구조 중 데이터 프레임 구조를 이해한다.
- 그 밖의 R에서 활용 가능한 데이터 구조를 이해한다.
- 벡터/리스트/행렬 구조의 데이터를 다룰 수 있다.

눈높이 체크

- 데이터 구조 중 벡터와 리스트를 알고 계신가요?
- 엑셀의 시트와 SAS의 데이터셋을 다루어 보셨나요?
- 벡터, 스칼라, 행렬, 요인의 정의를 이해하고 계신가요?



제4절 데이터 구조와 데이터 프레임

벡터

- 벡터들은 동질적이다: 한 벡터의 모든 원소는 같은 자료형 또는 같은 모드를 가진다.
- 벡터는 위치로 인덱스 된다: $v[2]$ 는 v 벡터의 2번째 원소이다.
- 벡터는 인덱스를 통해 여러 개의 원소로 구성된 하위 벡터를 반환할 수 있다: $v[c(2,3)]$ 은 v 벡터의 2번째, 3번째로 구성된 하위 벡터이다.
- 벡터 원소들은 이름을 가질 수 있다.

```
> V<-c(10,20,30); names(V)<-c("Moe","Larry","Curly")
```

```
> V["Larry"]
```

```
Larry
```

```
20
```

```
>
```



제4절 데이터 구조와 데이터 프레임

리스트

- 리스트는 이질적이다: 여러 자료형의 원소들이 포함될 수 있다.
- 리스트는 위치로 인덱스 된다: $L[2]$ 는 L리스트의 2번째 원소이다.
- 리스트는 하위 리스트를 추출할 수 있다: $L[c(2,3)]$ 은 L리스트의 2번째, 3번째 원소로 이루어진 하위 리스트이다.
- 리스트의 원소들은 이름을 가질 수 있다: $L[["Moe"]]$ 와 $L\$Moe$ 는 둘 다 Moe라는 이름의 원소를 지칭한다.

제4절 데이터 구조와 데이터 프레임

R에서의 자료 형태

객체	예시	모드
숫자	3.1415	수치형(numeric)
숫자 벡터	c(2,3,4,5,6)	수치형(numeric)
문자열	"Tom"	문자형(character)
문자열 벡터	c("Tom","Yoon","Kim")	문자형(character)
요인	factor(c("A","B","C"))	수치형(numeric)
리스트	list("Tom","Yoon","Kim")	리스트(list)
데이터프레임	data.frame(x=1:3, y=c("Tom","Yoon","Kim"))	리스트(list)
함수	print	함수(function)

데이터 프레임

- 데이터 프레임은 강력하고 유연한 구조. SAS의 데이터셋을 모방해서 만들어진다.
- 데이터 프레임의 리스트의 원소는 벡터 또는 요인이다.
- 그 벡터와 요인은 데이터프레임의 열이다.
- 벡터와 요인은 동일한 길이이다.
- 데이터 프레임은 표 형태의 데이터 구조이며 각 열은 서로 다른 데이터 형식을 가질 수 있다.
- 열에는 이름이 있어야 한다.
- [ex] 데이터 프레임의 원소에 대한 접근 방법: `b[1]`; `b["empno"]` / `b[[i]]`; `b[["empno"]]` / `b$empno`



제4절 데이터 구조와 데이터 프레임

그 밖의 데이터 구조들

객체	설명
단일값	R 원소가 하나인 벡터로 인식 처리 > pi [1] 3.141593 > length(pi) [1] 1
행렬	R에서는 차원을 가진 벡터로 인식 > a<-1:9 > dim(a)<-c(3,3) > a [1,] [2,] [3,] [1,] 1 4 7 [2,] 2 5 8 [3,] 3 6 9



제4절 데이터 구조와 데이터 프레임

그 밖의 데이터 구조들

객체

설명

배열

행렬에 3차원 또는 n차원까지 확장된 상태. 주어진 벡터에 더 많은 차원을 부여하여 배열 생성

```
> b<-1:12
```

```
> dim(b)<-c(2,3,2)
```

```
> b
```

```
,, 1
```

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

```
[1,]  1  3  5
```

```
[2,]  2  4  6
```

```
,, 2
```

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

```
[1,]  7  9 11
```

```
[2,]  8 10 12
```

```
>
```

요인

벡터처럼 생겼지만, R에서는 벡터에 있는 고유값 정보를 얻어내는데, 이 고유값들을 요인의 수준이라고 함.

요인은 두가지 주된 사용처로 범형 변수, 집단 분류가 있다.

제4절 데이터 구조와 데이터 프레임

벡터, 리스트, 행렬 다루기

- 행렬은 R에서 차원을 가진 벡터이며, 텍스트 마이닝과 소셜 네트워크 분석 등에 활용한다.
- 재활용 규칙: 길이가 서로 다른 두 벡터에 대한 연산을 할 때, R은 짧은 벡터의 처음으로 돌아가 연산이 끝날 때까지 원소들을 재활용한다.

<code>a<-seq(1,6)</code>	<code>b<-seq(7,9)</code>	<code>a+b</code>	<code>cbind(a,b)</code>
1	7	8	1 7
2	8	10	2 8
3	9	12	3 9
4		11	4 7
5		13	5 8
6		15	6 9

제4절 데이터 구조와 데이터 프레임

벡터, 리스트, 행렬 다루기

기능	코드 및 비고
벡터에 데이터 추가	<code>v<-c(v,newitem) / v[length(v)+1]<-newitem</code>
벡터에 데이터 삽입	<code>append(vec, newvalues, after=n)</code>
요인 생성	<code>f<-factor(v)</code> , v: 문자열 또는 정수 벡터 <code>f<-factor(v, levels)</code>
여러 벡터를 합쳐 하나의 벡터와 요인으로 만들기	<code>comb<-stack(list(v1=v1,v2=v2,v3=v3))</code>
벡터 내 값 조회	<code>v[c(1,3,5,7)]</code> : 벡터 내 1,3,5,7번째 값 조회 <code>v[-c(2,4)]</code> : 벡터 내 2,4번째 값을 제외하고 조회
리스트	<code>list(숫자, 문자, 함수)</code> : 리스트 함수의 인자로는 숫자, 문자, 함수가 포함
리스트 생성하기	<code>L<-list(x,y,z)</code> <code>L<-list(valuename1=data, valuename2=data, valuename3=data)</code> <code>L<-list(valuename1=vec, valuename2=vec, valuename3=vec)</code>
리스트 원소 선택	<code>L[[n]]</code> : n번째 원소 / <code>L[c(n1,n2,n3...nk)]</code> : 목록

제4절 데이터 구조와 데이터 프레임

벡터, 리스트, 행렬 다루기

기능	코드 및 비고
이름으로 리스트 선택	<code>L[["name"]], L\$name</code>
리스트 원소 제거	<code>L[["name"]]<-NULL</code>
NULL원소 리스트에서 제거	<code>L[sapply(L,is.null)]<-NULL / L[L==o]<-NULL / L[is.na(L)]<-NULL</code>
행렬	<code>matrix(data, 행수, 열수), a<-matrix(data,2,3), d<-matrix(0,4,5), e<-matrix(1:20,4,5)</code> data 대신 숫자를 입력하면 행렬의 값이 동일한 수치값 부여
차원	<code>dim(행렬), dim(a): a행렬의 차원은 2행,3열</code>
대각	<code>diag(행렬), diag(b): b행렬의 대각선 ㄱ뺀 반환</code>
전치	<code>t(행렬), t(a): a행렬의 전치행렬을 반환</code>
역	<code>slove(matrix)</code>

제4절 데이터 구조와 데이터 프레임

벡터, 리스트, 행렬 다루기

기능	코드 및 비고
행렬의 곱	행렬 <code>%%t(행렬)</code> , <code>a%%t(a)</code>
행 이름 부여	<code>rownames(a)<-c("행이름1","행이름2","행이름3")</code>
열 이름 부여	<code>colnames(a)<-c("행이름1","행이름2","행이름3")</code>
행렬의 연산 +,-	<code>f+f</code> , <code>f-f</code> : 행렬간의 덧셈, 뺄셈 / <code>f+1</code> , <code>f-1</code> : 행렬 상수간 덧셈, 뺄셈
행렬의 연산 *	<code>f%%f</code> : 행렬간의 곱 / <code>f*3</code> : 행렬 상수 간의 곱
행렬에서 행, 열 선택하기	<code>vec<-matrix[1,]</code> / <code>vec<-matrix[,3]</code>

제4절 데이터 구조와 데이터 프레임

데이터 프레임

기능	코드 및 비고
데이터프레임 레코드 생성	<code>data.frame(벡터, 벡터, 벡터)</code> : 벡터들로 데이터 셋 생성 <code>new<-data.frame(a=1,b=2,c=3,d="a")</code> 레코드 생성시 숫자, 문자를 함께 사용 가능
열 데이터(변수)로 데이터 프레임 만들기	<code>dfrm<-data.frame(v1,v2,v3,f1,f2)</code> <code>dfrm<-as.data.frame(list.of.vectors)</code>
데이터셋 행 결합	<code>rbind(dfrm1, dfrm2)</code> / <code>newdata <- rbind(data, row)</code> 두 데이터프레임을 행으로 결합
데이터셋 열 결합	<code>cbind(dfrm1, dfrm2)</code> / <code>newdata <- cbind(data, col)</code> 두 데이터프레임을 행으로 결합
데이터 프레임 할당	<code>N=1,000,000</code> <code>dftm<-data.frame(dosage=numeric(N),lab=character(N), response=numeric(N))</code>
데이터프레임 조회	[방법1] <code>dfrm[dfrm\$gender="m"]</code> : 데이터셋 내 성별이 남성만 조회 [방법2] <code>dfrm[dfrm\$변수1>4 & dfrm\$변수2>5, c(변수3, 변수4)]</code> : 데이터셋의 변수1과 변수2의 조건에 만족하는 레코드 변수3과 변수4만 조회 [방법3] <code>dfrm[grep("문자", dfrm\$변수1, ignore.case=T), c("변수3", "변수4")]</code> : 데이터셋의 변수1내 "문자"가 들어있는 케이스들의 변수2, 변수3값을 조회
데이터셋 조회	<code>subset(dfrm, select=변수, subset=변수>조건)</code> 데이터셋의 특정변수의 값이 조건에 맞는 변수셋 조회, <code>subset</code> 은 벡터와 리스트에서도 선택 가능

제4절 데이터 구조와 데이터 프레임

데이터 프레임

기능	코드 및 비고
데이터 선택	<code>lst1[[2]], lst1[2], lst1[2,], lst1[,2]</code> <code>lst1[["name"]], lst1\$name</code> <code>lst1[c("name1", "name2", ..., "namek")]</code>
데이터 병합	<code>merge(df1, df2, by="df1과 df2의 공통 열 이름")</code> : 공통변수로 데이터셋 병합
열 네임 조회	<code>colnames(변수)</code> : 변수의 속성들을 조회
행, 열 선택	<code>subset(dfm, select=열이름)</code> <code>subset(dfm, select=c(열이름1, 열이름2,..., 열이름n))</code> <code>subset(dfm, select=열이름, subset(조건))</code> 열이름에 "" 표시 안함, 조건에 맞는 행의 열 자료만 선택
이름으로 열 제거	<code>subset(dfm, select=-"열이름")</code>
열이름 바꾸기	<code>colnames(dfm)<-newnames</code>
NA 있는행 삭제	<code>NA_cleaning<-na.omit(dfm)</code>
데이터 프레임 두개 합치기	열: <code>cbind_dfm <- cbind(dfm1, dfm2)</code> 행: <code>rbind_dfm<- rbind(dfm1, dfm2)</code> [유의사항1] <code>cbind</code> 행의 개수가 동일해야함-recycling Rule [유의사항2] <code>rbind</code> 열의 개수와 열의 이름이 동일해야 함
두개 데이터 프레임을 동일한 변수 기준으로 합치기	<code>merge(dfm1, dfm2, by="T_name")</code> <code>merge(dfm1, dfm2, by="T_name", all=T)</code>

자료형 데이터 구조 반환

기능	코드
데이터프레임의 내용에 쉽게 접근하기	<code>with(dfm, expr) / attach(dfm) / detach(dfm)</code>
자료형 변환하기	<code>as.character() / as.complex() / as.numeric()</code> 또는 <code>as.double()</code> <code>as.integer() / as.logical()</code>
데이터구조변환하 기	<code>as.data.frame() / as.list() / as.matrix() / as.vector()</code>



제4절 데이터 구조와 데이터 프레임

데이터 구조 변경

기능	코드
벡터->리스트	<code>as.list(vec)</code>
벡터->행렬	1행짜리 행렬: <code>cbind(vec)</code> 또는 <code>as.matrix(vec)</code> 1열짜리 행렬: <code>rbind(vec) / nxm</code> 행렬: <code>matrix(vec, n, m)</code>
벡터->데이터프레임	1열짜리 데이터프레임: <code>as.data.frame(vec)</code> 1행짜리 데이터프레임: <code>as.data.frame(rbind(vec))</code>
리스트->벡터	<code>unlist(lst)</code>
리스트->행렬	1열짜리 행렬: <code>as.matrix(lst) / 1행짜리 행렬: as.matrix(rbind(lst))</code> <code>nxm</code> 행렬: <code>matrix(lst, n, m)</code>
리스트->데이터프레임	목록 원소들이 데이터의 열이면: <code>as.data.frame(lst)</code> 리스트 원소들이 데이터의 행이면: <code>rbind(obs[[1]], obs[[2]])</code>
행렬->벡터	<code>as.vector(mat)</code>
행렬->리스트	<code>as.list(mat)</code>
행렬->데이터프레임	<code>as.data.frame(mat)</code>
데이터프레임->벡터	1열짜리 데이터프레임: <code>dfm[[1]]</code> or <code>dfm[,1]</code> 1행짜리 데이터프레임: <code>dfm[1,]</code>
데이터프레임->리스트	<code>as.list(dfm)</code>
데이터프레임->행렬	<code>as.matrix(dfm)</code>

제4절 데이터 구조와 데이터 프레임

벡터의 기본 연산

기능	코드
벡터 연산	벡터1+벡터2(덧셈연산) / 벡터1-벡터2(뺄셈연산) / 벡터1*벡터2(곱셈연산) / 벡터1^벡터2(승수연산)
함수 적용	sapply(변수, 연산함수) / sapply(a,log)연산 및 적용함수를 통해 변수에 적용
파일 저장	write.csv(변수이름, "파일이름.csv") write.csv(변수이름, file="파일이름.Rdata"): R파일로 저장
파일 읽기	read.csv("파일이름.csv")
파일 불러오기	load("파일 R") / source("a.R"): R파일 불러오기
데이터 삭제	rm(변수): 변수를 메모리에서 삭제 rm(list=ls(all=True)): 모든 변수를 메모리에서 삭제

제4절 데이터 구조와 데이터 프레임

그 외에 간단한 함수

기능	코드
데이터 불러오기	<code>data()</code> : R에 내장된 데이터셋 리스트를 보여줌. / <code>data(데이터셋)</code> : 데이터셋을 불러들임
데이터셋 요약	<code>summary(데이터셋)</code> : 데이터셋 변수 내용을 요약
데이터셋 조회	<code>head(데이터셋)</code> : 6개 레코드까지 데이터 조회
패키지 설치	<code>install.packages("패키지 명")</code>
패키지 불러오기	<code>library("패키지 명")</code>
작업종료	<code>q()</code>
워킹디렉토리 지정	<code>setwd("~/")</code> : R데이터, 파일을 로드하거나 저장할 때 워킹 디렉토리를 저장

제5절 데이터 변형

주요 코드

기능	코드
요인으로 집단 정의	<pre>v<-c(24, 23, 52, 46, 75, 25) / w<-c(87, 86, 92, 84, 77, 66) f<-factor(c("A","A","B","B","C","A"))</pre>
벡터를 여러 집단으로 분할 (벡터의 길이만 같으면 됨)	<pre>groups<-split(v,f) / groups<-split(w,f) groups<-unstack(dta.frame(v,f)) 두 함수 모두 벡터로 된 리스트를 반환</pre>
데이터프레임을 여러집단으로 분할	<pre>MASS 패키지, Cars93 데이터셋 활용 library(MASS) sp<-split(Cars93\$MPG.city, Cars93\$Origin) median(sp[[1]])</pre>
리스트의 각 원소에 함수 적용	<pre>lapply(결과를 리스트 형태로 반환) list<-lapply(l, func) sapply(결과를 벡터 또는 행렬로 반환) vec<-sapply(l,function)</pre>
행렬에 함수 적용	<pre>m<-apply(mat, 1, func) / m<-apply(mat, 2, func)</pre>
데이터프레임에 함수 적용	<pre>dfm<-lapply(dfm, func)dfm<-sapply(dfm, func) dfm<-apply(dfm, func): 데이터프레임이 동질적인 경우만(모든문자, 숫자)활용가능 데이터프레임을 행렬로 변환 후 함수 적용</pre>

주요 코드

기능	코드
대용량 데이터의 함수 적용	<pre>cors <- sapply(dfm, cor, y=targetVariable) mask <- (rank(-abs(cors)) <= 10) best.pred <- dfm[, mask] lm(targetVariable ~ bes.pred)</pre> <p>많은 변수가 있는 데이터에서의 다중회귀 분석</p> <ol style="list-style-type: none"> 1. 데이터프레임에서 타겟 변수를 정함 2. 타겟변수와 상관계수를 구한다. 3. 상관계수가 높은 상위 10개의 변수를 입력변수로 선정 4. 타겟변수와 입력변수로 다중회귀분석을 실시한다.
집단별 함수 적용	<pre>tapply(vec, factor, func)</pre> <p>데이터가 집단(factors)에 속해 있을 때 합계/평균 구하기</p>
행집단 함수 적용	<pre>by(drm, factor, func): 요인별 선형회귀선 구하기 model(dfm, factor,function(df) lm(종속변수~독립변수1+독립변수2+...독립변수 k, data=df))</pre>
병렬 벡터, 리스트들 함수 적용	<pre>mapply(factor, v1, ..., vk) mapply(factor, list1, ..., list k)</pre>

문자열 날짜 다루기

기능	코드
문자열 길이	<code>nchar("단어")</code> : 단어나 문장 또는 벡터 내 원소의 문자열 길이 반환 [주의] <code>length(vec)</code> : 문자열의 길이가 아닌 벡터 길이 반환
문자열 연결	<code>paste("word1", "word2", sep="-")</code> <code>paste("the pi is approximately", pi)paste(vec, "loves me", collapse=", and")</code>
하위문자열 추출	<code>substr("statistics", 1, 4)</code> : 문자열의 1번째에서 4자리 추출
구분자로 문자열 추출	<code>strsplit(문자열, 구분자)</code>
하위 문자열 대체	<code>sub(old, new, string)</code> <code>gsub(old, new, string)</code>
쌍별 조합	<code>mat <- outer(문자열1, 문자열2, paste, sep="")</code>

문자열 날짜 다루기

기능	코드
날짜변환	<code>Sys.Date()</code> : 현재 날짜 반환 / <code>as.Date()</code> : 날짜 객체 반환 <code>format(Sys.Date(), format=%m%d%y)</code>
날짜조회	<code>format(Sys.Date(), "%a")</code> 요일조회 / <code>format(Sys.Date(), "%b")</code> 축약된 월조회 <code>format(Sys.Date(), "%B")</code> 전체 월조회 / <code>format(Sys.Date(), "%d")</code> 두자리 숫자의 일조회 <code>format(Sys.Date(), "%m")</code> 두자리 숫자의 월조회 <code>format(Sys.Date(), "%y")</code> 두자리 숫자의 연도조회 <code>format(Sys.Date(), "%Y")</code> 네자리 숫자의 연도조회
날짜 일부 추출	<code>d <- as.Date("2014-12-25")</code> <code>p <- as.POSIXlt(d)</code> <code>p\$yday</code> <code>start <- as.Date("2014-12-01")</code> <code>end <- as.Date("2014-12-25")</code> <code>seq(from=start, to=end, by=1)</code>

『3과목』 데이터 분석

제2장 R프로그래밍 기초 - QnA





1

01. 다음 중 R의 데이터 구조 중 벡터에 대한 설명으로 적절한 것은?

- ① 벡터는 행과 열을 갖는 $m \times n$ 형태의 직사각형에 데이터를 나열할 데이터 구조이다.
- ② 벡터는 하나의 스칼라 값 또는 하나 이상의 스칼라 원소들을 갖는 단순한 형태의 집합이다.
- ③ 벡터는 행렬과 유사한 2차원 목록 데이터 구조이다.
- ④ 벡터는 숫자로만 구성되어야 한다.



2

02. 다음 중 아래의 프로그램을 통해 생성된 벡터 `xy`에 대한 설명으로 옳지 않은 것은?

```
> x <- c(1:4)
> y <- c("apple","banana","orange")
> xy <- c(x,y)
```

- ① `xy` 는 문자형 벡터이다.
- ② `xy` 의 길이는 7이다.
- ③ `xy[1]+ xy[2]`의 결과는 3이다.
- ④ `xy[5:7]`은 `y`와 동일하다.

`xy`는 문자형 벡터로 문자형은 서로 연산을 할 수 없으므로 출력결과에는 에러가 나타난다.



3

03. 다음 중 나머지 세 개의 명령과 결과가 다른 것은?

- ① `Z=c(1:3, NA)`
`is.na(Z)`
- ② `Z<-c(1:3,NA)`
`is.na(Z)`
- ③ `Z=c(1:3,NA)`
`Z==NA`
- ④ `c(1,1,1,2)==2`

FALSE FALSE FALSE TRUE
NA NA NA NA



4

04. 다음 중 아래 R코드의 결과로 적절한 것은?

```
> s <-c("Monday","Tuesday","Wednesday")  
> substr(s,1,2)
```

- ① "Mo","Tu","We"
- ② "Monday","Tuesday"
- ③ "Mo" "Tu"
- ④ "Monday"

substr() 함수를 사용하여 각 문자열의 첫 두 글자를 추출하는 것



5

05. R에서 데이터 타입이 같지 않은 객체들을 하나의 객체로 묶어놓을 수 있는 자료구조는 어떤 것인가?

- ① 행렬
- ② 배열
- ③ 리스트
- ④ 문자열

- R에서 다양한 자료구조가 있습니다.
- 벡터(Vector):
 - 단일 데이터 유형으로 구성된 1차원 배열입니다.
 - numeric, character, logical 등의 데이터 유형을 포함할 수 있습니다.
 - c() 함수를 사용하여 생성하거나, vector() 함수를 사용하여 초기화합니다.
- 행렬(Matrix):
 - 2차원 배열로, 동일한 데이터 유형의 요소를 갖습니다.
 - matrix() 함수를 사용하여 생성하며, 행과 열의 구조를 가집니다.
- 배열(Array):
 - 행렬의 일반화된 형태로, 다차원 데이터 구조입니다.
 - array() 함수를 사용하여 생성하며, 3차원 이상의 다차원 배열을 만들 수 있습니다.
- 데이터 프레임(Data Frame):
 - 열(column)과 행(row)으로 구성된 2차원 데이터 구조입니다.
 - 각 열은 서로 다른 데이터 유형을 가질 수 있습니다.
 - data.frame() 함수를 사용하여 생성하며, 테이블 형식의 데이터를 다루는 데 사용됩니다.

- **리스트(List):**

- 서로 다른 데이터 유형의 요소들로 구성된 순서가 있는 자료구조입니다.
- 서로 다른 길이의 요소들을 포함할 수 있습니다.
- `list()` 함수를 사용하여 생성하며, 데이터를 그룹화하고 저장하는 데 사용됩니다.

- **요인(Factor):**

- 범주형 데이터를 표현하는 데 사용되는 자료구조입니다.
- 범주(category)와 해당 데이터에서 범주에 대한 레벨(level)을 저장합니다.
- `factor()` 함수를 사용하여 생성하며, 주로 통계 분석에서 범주형 변수를 다룰 때 사용됩니다.



6

06. 다음 중 2019/08/23을 "2019-08-23"으로 나타내는 코드로 올바른 것은?

- ① as.Date('08/23/2019','%m/%d/%Y')
- ② as.Date('08/23/2019','%m/%D/%Y')
- ③ as.Date('08/23/2019','%M/%d/%Y')
- ④ as.Date('08/23/2019','%M/%D/%Y')