1차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- R에 대해 알 수 있다.
- 빅데이터에 대해 알 수 있다.

학습 목차

- 커리큘럼 소개
- R이 무엇인가요?
- 빅데이터란?

01.

커리큘럼 소개

01 커리큘럼 소개

01 목표

- 비전공자 대상 데이터 시각화 기술 습득 (R 프로그래밍)
- 600분 과정

01 커리큘럼 소개

02 구성

① R 소개 및 설치
② 기본 문법 학습
③ 데이터 시각화 학습

02.

R이 무엇인가요?

02 R이란? – R언어의 History

Ross Ihaka

Robert Gentleman

1993

- Ross Ihaka, Robert Gentleman 공동 개발
- Univ. of Auckland

1995

- 오픈소스 소프트웨어 형식 개발 착수

1997

- R core 팀 결성

2000

- Stable 버전(v1.0.0) 첫 출시

2021. 8

- v.4.1.1 서비스 중

02 R이란? – 왜 만들어졌을까?

전산 통계학(computational Statistics)을 위한 프로그래밍 언어

02 R이란? – 왜 만들어졌을까?

전산 통계학(computational Statistics)을 위한 프로그래밍 언어

고전 통계학

전산 통계학

02 R이란? – 왜 만들어졌을까?

전산 통계학(computational Statistics)을 위한 프로그래밍 언어

화상처리(Graphics)

- 시각화

02 R이란? – 왜 만들어졌을까?

전산 통계학(computational Statistics)을 위한 프로그래밍 언어

화상처리(Graphics)

- 시각화

```
sepal_length,sepal_width,petal_length,petal_width,species
5.1,3.5,1.4,0.2,setosa
4.9,3.0,1.4,0.2,setosa
4.7,3.2,1.3,0.2,setosa
4.6,3.1,1.5,0.2,setosa
5.0,3.6,1.4,0.2,setosa
5.4,3.9,1.7,0.4,setosa
4.6,3.4,1.4,0.3,setosa
5.0,3.4,1.5,0.2,setosa
4.4,2.9,1.4,0.2,setosa
4.9,3.1,1.5,0.1,setosa
5.4,3.7,1.5,0.2,setosa
4.8,3.4,1.6,0.2,setosa
4.8,3.0,1.4,0.1,setosa
4.3,3.0,1.1,0.1,setosa
5.8,4.0,1.2,0.2,setosa
5.7,4.4,1.5,0.4,setosa
5.4,3.9,1.3,0.4,setosa
5.1,3.5,1.4,0.3,setosa
5.7,3.8,1.7,0.3,setosa
5.1,3.8,1.5,0.3,setosa
5.4,3.4,1.7,0.2,setosa
```

Raw 데이터 셋

02 R이란? – 왜 만들어졌을까?

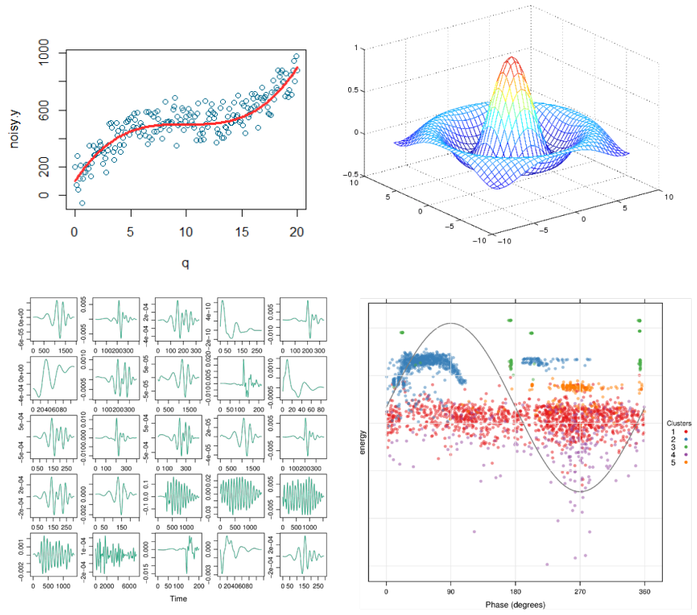
전산 통계학(computational Statistics)을 위한 프로그래밍 언어

화상처리(Graphics)

- 시각화

```
sepal_length,sepal_width,petal_length,petal_width,species
5.1,3.5,1.4,0.2,setosa
4.9,3.0,1.4,0.2,setosa
4.7,3.2,1.3,0.2,setosa
4.6,3.1,1.5,0.2,setosa
5.0,3.6,1.4,0.2,setosa
5.4,3.9,1.7,0.4,setosa
4.6,3.4,1.4,0.3,setosa
5.0,3.4,1.5,0.2,setosa
4.4,2.9,1.4,0.2,setosa
4.9,3.1,1.5,0.1,setosa
5.4,3.7,1.5,0.2,setosa
4.8,3.4,1.6,0.2,setosa
4.8,3.0,1.4,0.1,setosa
4.3,3.0,1.1,0.1,setosa
5.8,4.0,1.2,0.2,setosa
5.7,4.4,1.5,0.4,setosa
5.4,3.9,1.3,0.4,setosa
5.1,3.5,1.4,0.3,setosa
5.7,3.8,1.7,0.3,setosa
5.1,3.8,1.5,0.3,setosa
5.4,3.4,1.7,0.2,setosa
```

Raw 데이터 셋



R기반 시각화

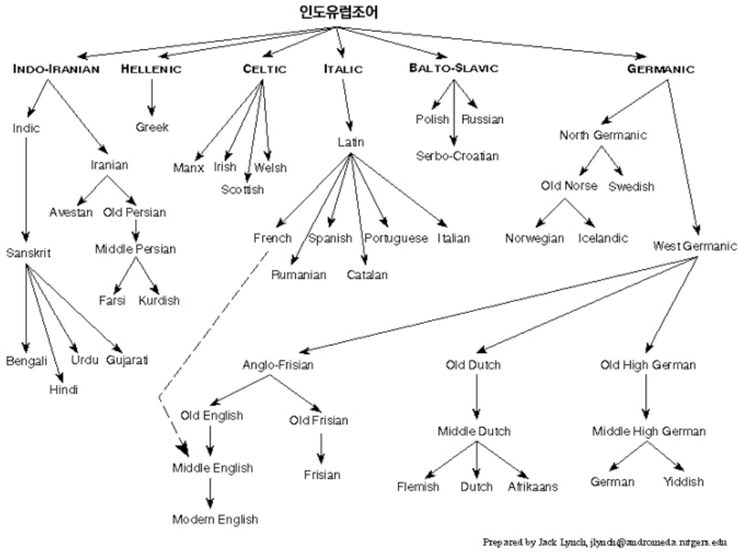
02 R이란? – 왜 만들어졌을까?

전산 통계학(computational Statistics)을 위한 프로그래밍 언어

화상처리(Graphics)

- 시각화

C, Fortran으로 작성된 함수형 프로그래밍 언어



03 빅데이터란? - 데이터의 시대

- ~2000년까지 생산된 정보

  → 20,000,000,000,000,000,000 bite = 10TB 하드디스크 2억개

위의 정보 보다 최근 1주일간 생산된 정보가 더 많다!

실제로 사물인터넷 시대의 데이터 발생량에 대해 언급된 자료가 있습니다. KT경제경영연구소 등에 따르면 인류가 파피루스든, 종이든 기록을 남기기 시작한 이래로 2000년대 초반까지 생산된 정보의 총량은 약 20엑사바이트에 달하는 것으로 추정된다고 합니다. 엑사바이트는 1 다음에 0이 18개가 붙는 단위입니다. 조 단위를 넘어 100경을 일컫습니다. 그러니까 20엑사바이트는 2000경바이트를 의미합니다. 그런데 본격적인 디지털 시대의 진입으로 2010년대에 들어서는 매일 3엑사바이트의 정보량이 생산된다는 견해가 주류라고 합니다. 인류가 2000년 넘게 걸려 생산해 낸 정보의 양을 2010년대에는 1주일 만에 만들어낸다는 겁니다. 어마어마하죠.

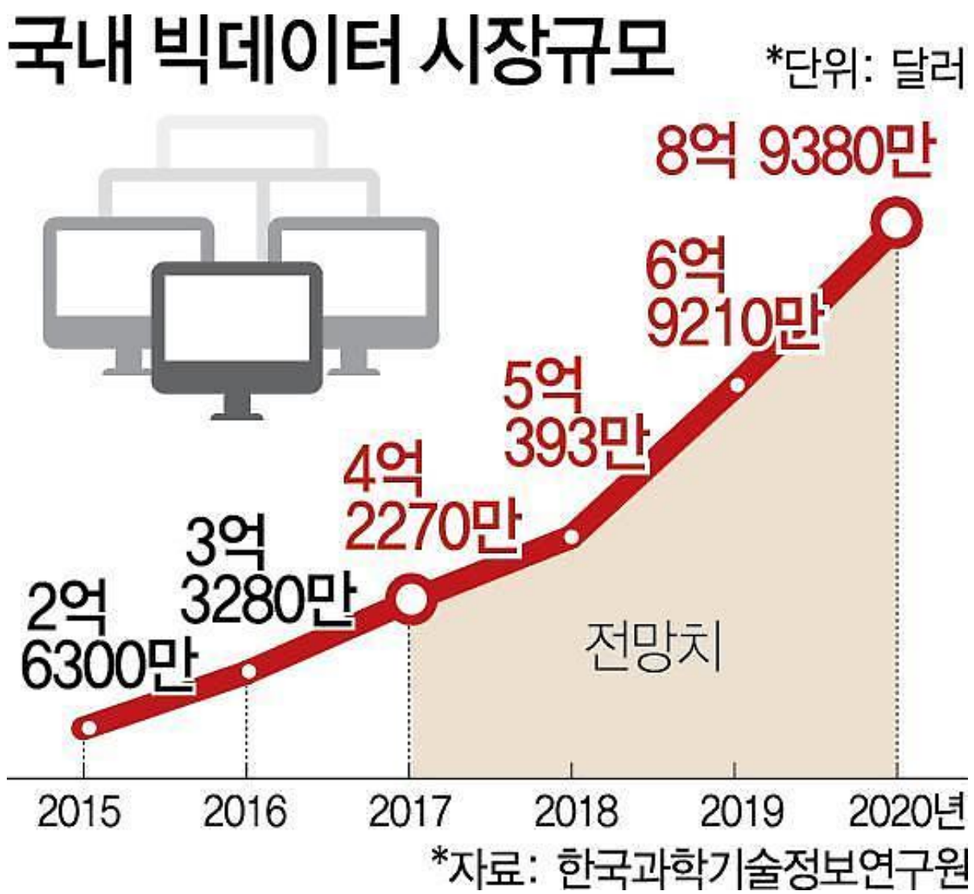
출처: 교육부 블로그, 2017

03.

빅데이터란?

03 빅데이터란? – 데이터의 시대

- 데이터 자체를 매매



03 빅데이터란? – 데이터의 시대

- 데이터 자체를 매매

국내 데이터산업 시장 규모 올해 20조 돌파

| 지난해 19조2736억원…종사자는 36만6천여명

컴퓨팅 | 입력 :2021/03/16 12:00 수정: 2021/03/16 14:49

정부, 2025년까지 데이터 시장 43조원·일자리 90만개 창출

과기부, 인공지능 국가전략 1년 성과 및 계획 발표…AI·SW 핵심인재 10만명 양성

2020.11.25 과학기술정보통신부

03 빅데이터란? – 데이터의 시대

- 사업에 데이터를 활용

- 예시
  - ➢ 넷플릭스 영화 추천
  - ➢ 유튜브 영상 추천
  - ➢ 테슬라 자율주행

03 빅데이터란? – 데이터의 시대

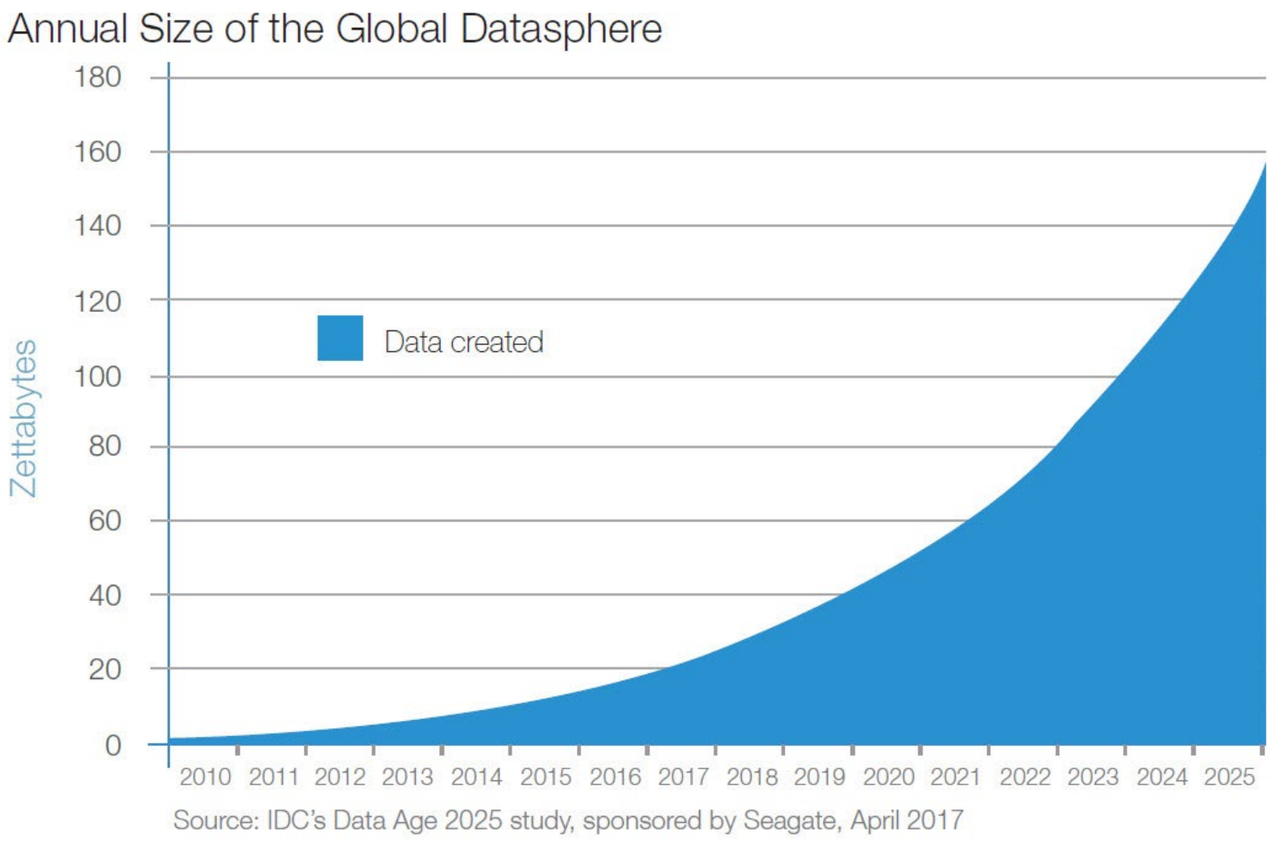
- 사업에 데이터를 활용

- 예시
  - ➢ 넷플릭스 영화 추천
  - ➢ 유튜브 영상 추천
  - ➢ 테슬라 자율주행

- 데이터 수집 중요성 ↑
  - ➢ 플랫폼의 시대

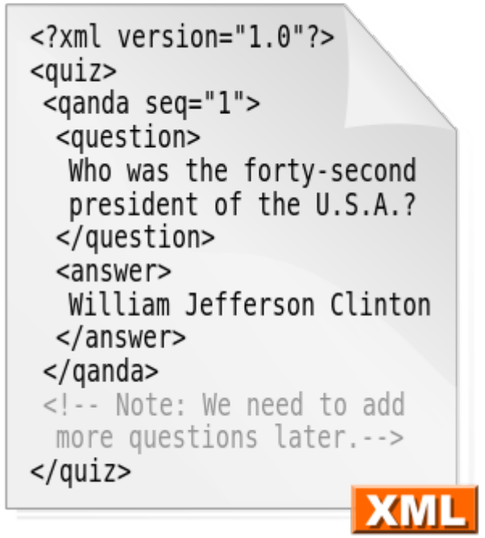
03 빅데이터란? – 빅데이터의 특징: 3V

- **Volume(데이터의 양)**
  - ➢ **물리적 데이터의 양**



03 빅데이터란? – 빅데이터의 특징: 3V

- Volume(데이터의 양)
  - ➢ 물리적 데이터의 양
- Variety(데이터 다양성)
  - ➢ 정형 데이터(표, Excel)
  - ➢ 준정형 데이터(XML, JSON)
  - ➢ 비정형 데이터
    - ▪ 사진, 비디오, 오디오
    - ▪ 위치 정보, 생체 기록 정보, 검색 로그



03 빅데이터란? – 빅데이터의 특징: 3V

- Volume(데이터의 양)
  - 물리적 데이터의 양
- Variety(데이터 다양성)
  - 정형 데이터(표, Excel)
  - 준정형 데이터(XML, JSON)
  - 비정형 데이터
    - 사진, 비디오, 오디오
    - 위치 정보, 생체 기록 정보, 검색 로그
- Velocity(데이터 속도)
  - 빠른 생성
  - 빠른 처리
    - 분석
    - 시각화

03 빅데이터란? – 빅데이터의 특징: 5V, 7V

- 5V
  - Veracity(데이터 진실성)
    - 데이터를 신뢰할 수 있는가?
  - Value(데이터 가치)
    - 가치를 창출할 수 있는 데이터인가?

- 7V
  - Validity(데이터 정확성)
    - 데이터의 품질이 좋은가?
  - Volatility(데이터 휘발성)
    - 오랜 기간 활용될 수 있는 데이터인가?

이번 차시 정리

R이란?

- 전산통계학을 위한 프로그래밍 언어
- 데이터 통계처리 및 시각화에 특화

빅데이터란?

- 빅데이터의 시대
  - ➢ 현 시대의 핵심 자원
- 3V
  - ➢ Volume, Variety, Velocity
- 5V, 7V

1차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

2차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 유사 분석 프로그램에 대해 알 수 있다.
- R의 장점과 단점에 대해 알 수 있다.

학습 목차

- 유사 분석 프로그램
- R의 장점
- R의 단점

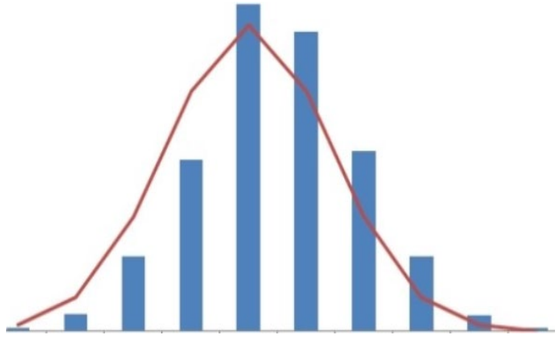
01.

유사 분석 프로그램

01 유사 분석 프로그램 – 왜 알아야 하는가?

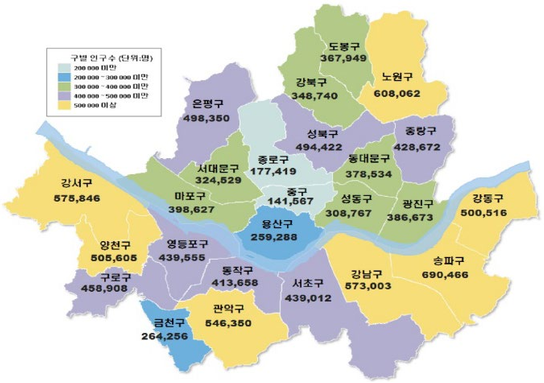
01 R이 모든 경우에서 최고인가?

통계량	수식
표본 평균	$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$
표본 분산	$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X})^2$
표본 표준 편차	$\sqrt{S^2}$



02 R은 유용한 데이터 분석 수단 중 하나

- 분석 프로그램 별 장단점 존재
  - ➢ 상황에 따라 유연하게 선택할 수 있는 소양 필요

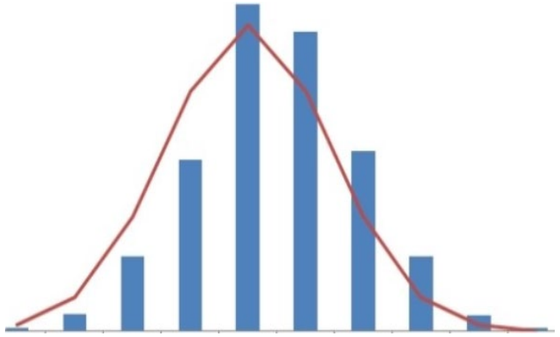


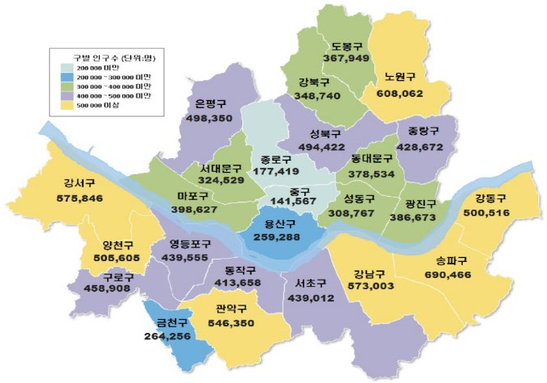
01 유사 분석 프로그램 – 왜 알아야 하는가?

01 ROI 모든 경우에서 최고인가?

- 목표 결과물은 다양함
  - ➢ 평균, 분산
  - ➢ 분포표
  - ➢ 서울시 구 별 데이터
- 데이터 개수,형태에 따라 시각화 난도 급변
- 결과물 산출까지 걸리는 시간

통계량	수식
표본 평균	$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$
표본 분산	$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X})^2$
표본 표준 편차	$\sqrt{S^2}$





01 유사 분석 프로그램 – Excel

표 형태의 분석 프로그램

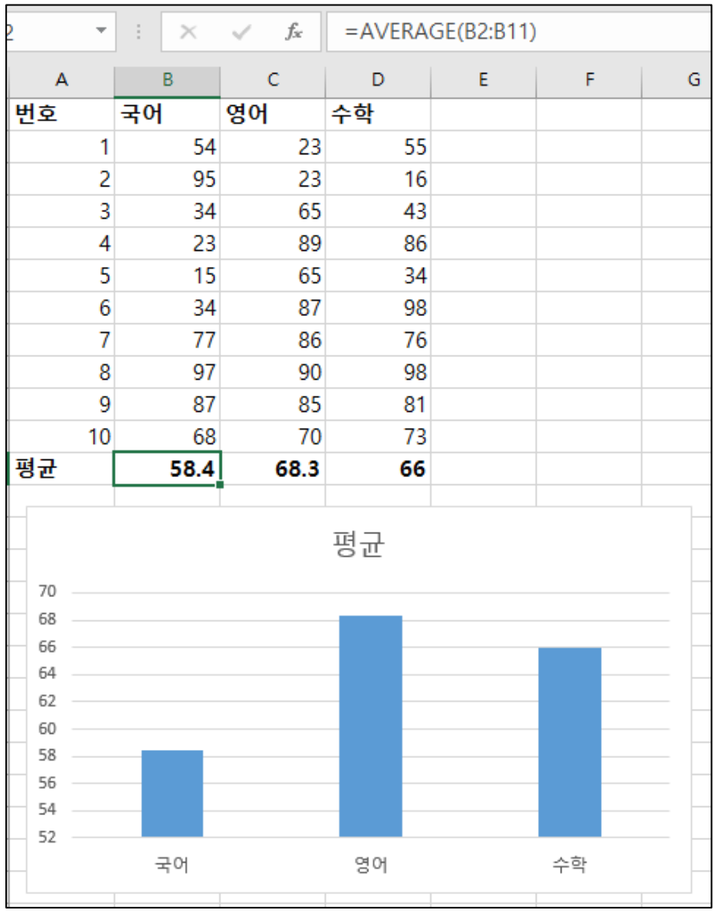
Excel

	A	B	C	D
1	번호	국어	영어	수학
2	1	54	23	55
3	2	95	23	16
4	3	34	65	43
5	4	23	89	86
6	5	15	65	34
7	6	34	87	98
8	7	77	86	76
9	8	97	90	98
10	9	87	85	81
11	10	68	70	73

01 유사 분석 프로그램 – Excel

표 형태의 분석 프로그램

- 장점
  - ➢ 메뉴 기반
    - ▪ 프로그래밍 지식 필요 X
    - ▪ 손쉽게 사용가능
  - ➢ 다양한 매크로 함수 제공
  - ➢ 다양한 셀 단위 시각화 기능
    - ▪ 조건부 서식, 그래프, 원 차트
  - ➢ 익숙한 MS Office 문서 기반



번호	국어	영어	수학
1	54	23	55
2	95	23	16
3	34	65	43
4	23	89	86
5	15	65	34
6	34	87	98
7	77	86	76
8	97	90	98
9	87	85	81
10	68	70	73
평균	58.4	68.3	66

01 유사 분석 프로그램 – Excel

표 형태의 분석 프로그램

- 단점
  - ➢ 고급 통계, 전문적인 시각화 작업 어려움
  - ➢ 다양한 기능 한꺼번에 적용 힘듦
  - ➢ 반복 작업에 비효율적

01 유사 분석 프로그램 - Python

범용 프로그래밍 언어

PYTHON

01 유사 분석 프로그램 – Python

범용 프로그래밍 언어

- 장점
  - ➢ 매우 뛰어난 확장성
    - ▪ 응용 프로그램 개발
    - ▪ 웹/앱 개발
    - ▪ 인공지능 연구
  - ➢ 강력한 IDE
    - ▪ Jupyter notebook
  - ➢ 무료

01 유사 분석 프로그램 – Python

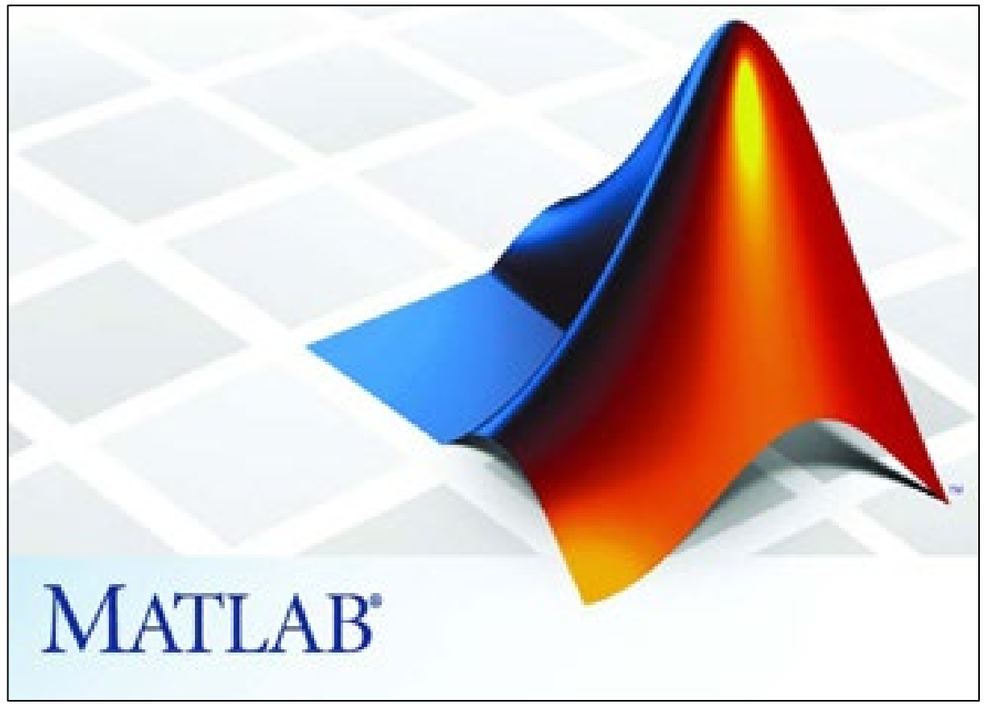
범용 프로그래밍 언어

- 단점
  - ➢ 디버깅 난이도 높음
  - ➢ 특정 데이터 형식 시각화 기능 미흡 (Eg. Network graph)

01 유사 분석 프로그램 - Matlab

공학용 프로그램 패키지

- Mat(rix) lab



01 유사 분석 프로그램 – Matlab

공학용 프로그램 패키지

- 장점
  - 수식 계산에 탁월
  - 간편성
    - 방대한 매크로 함수
  - 시각화 성능 우수

- 단점
  - 유료
  - 통계분석 기능 미흡

01 유사 분석 프로그램 – 그 외의 프로그램들

SQL

- 데이터 베이스 관리 프로그램

Tableau

- 데이터 스토리텔링 특화

Power BI

- Microsoft Office 기반 툴

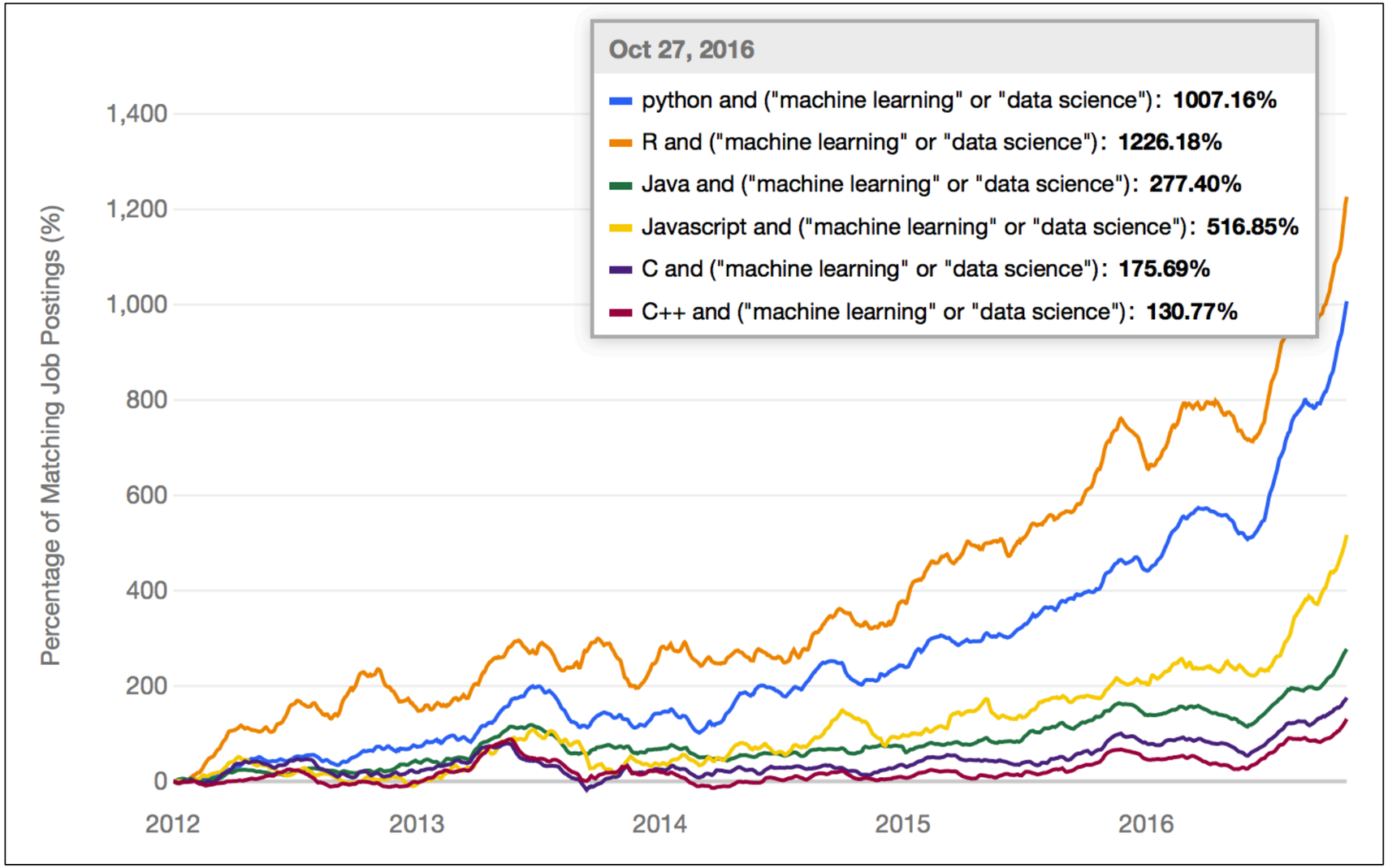
Google Analytics

- 웹 로그 데이터 분석 툴
  - ➢ 사이트 방문자 통계, 방문 경로 분석 등

02.

R의 장점

02 R의 장점


Oct 27, 2016
python and ("machine learning" or "data science"): 1007.16%
R and ("machine learning" or "data science"): 1226.18%
Java and ("machine learning" or "data science"): 277.40%
Javascript and ("machine learning" or "data science"): 516.85%
C and ("machine learning" or "data science"): 175.69%
C++ and ("machine learning" or "data science"): 130.77%
Percentage of Matching Job Postings (%)
1,400
1,200
1,000
800
600
400
200
0
2012
2013
2014
2015
2016

02 R의 장점 – 언어적 측면

통계 계산용 패키지 매우 우수

- dplyr, tidyr, stringr, lubridate

강력한 시각화 패키지 제공

- ggplot2, rgl, htmlwidgets

뛰어난 확장성

- 초기에는 확장성 좋지않았으나, 다양한 패키지 등장
- 웹기반 분석결과 리포팅
  - ggvis, shiny
- 인공지능 연구
  - lme4/nlme, randomForest, caret, deepnet

02 R의 장점 - 사용자 측면

유저 커뮤니티 활성화

- 대부분의 유저들이 유사한 목표를 가짐
- 디버깅 용이

상대적으로 학습난이도 낮음

다양한 OS 지원

- Windows, Mac, Linux

무료

03.

R의 단점

02 R의 단점

느린 속도

- 범용 프로그래밍 언어보다 처리 속도 느림

한정된 사용성

- 대규모 IT 서비스 개발에 접목이 어려움
  - Eg. R은 게임개발에 부적합

보안 기능 없음

02 R VS. Python

공통점

- 무료
  - Open source
  - 다양한 패키지 지속적 업데이트 중
- 활발한 커뮤니티
- 우수한 IDE

차이점

- 통계분석 + 시각화 vs 범용
- 확장성: Python 우세
- 통계분석, 시각화: R 우세
- 디버깅: R 우세
- 데이터 처리까지 필요한 공부량: R 우세

이번 차시 정리

R과 유사한 데이터분석 프로그램

- Excel / Python / Matlab / Etc.

R의 장단점 분석

- R은 데이터 통계분석과 시각화에 최적화

R vs. Python

- R은 데이터 통계분석과 시각화에 최적화

→ 상황과 목적에 알맞은 데이터 분석 프로그램 선택 능력 증진 필요

2차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

3차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- R 활용 가능 분야에 대해 알 수 있다.

학습 목차

- R 활용 가능 분야

01.

R 활용 가능 분야

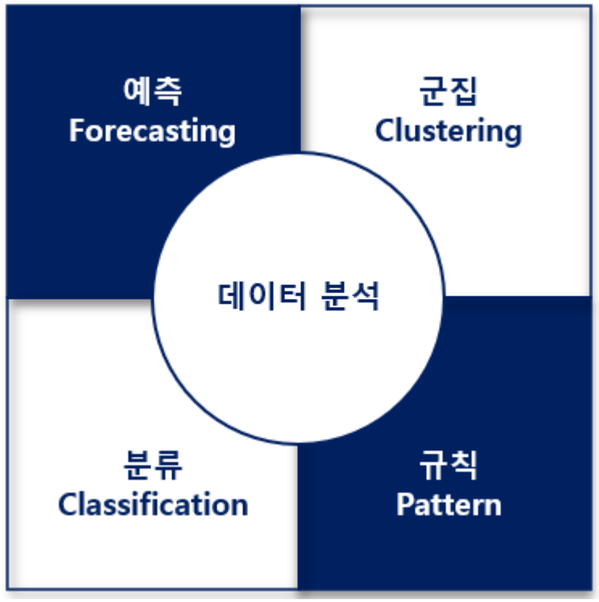
01 R 활용 가능 분야 – 데이터 마이닝

데이터 마이닝?

- Data + Mining = “데이터”로 부터 정보를 “채굴”

데이터 마이닝의 목적

- 데이터 모집단의 패턴을 찾아
  새로운 데이터를 예측, 군집화, 분류



01 R 활용 가능 분야 – 데이터 마이닝

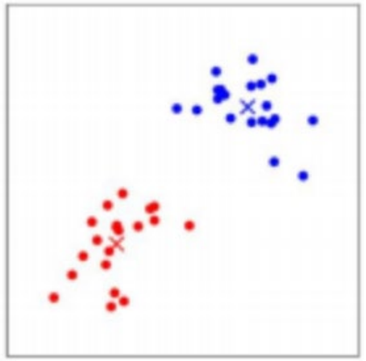
데이터 마이닝?

- Data + Mining = “데이터”로 부터 정보를 “채굴”

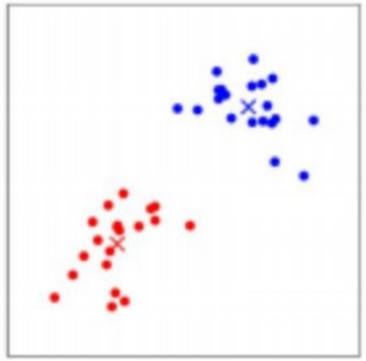
데이터 마이닝의 목적

- 데이터 모집단의 패턴을 찾아
  새로운 데이터를 예측, 군집화, 분류

군집화



분류



예측

01 R 활용 가능 분야 – 텍스트 마이닝

- 웹 등에서 방대한 텍스트 정보 크롤링
- 단어 빈도순으로 가중치 주어 시각화

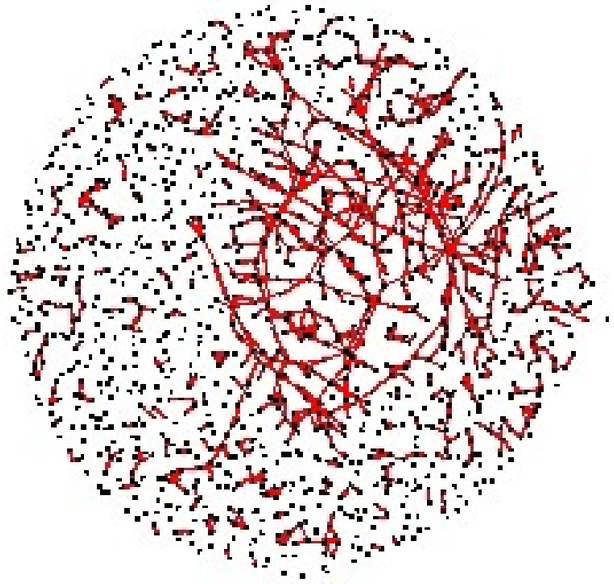
01 R 활용 가능 분야 – 텍스트 마이닝

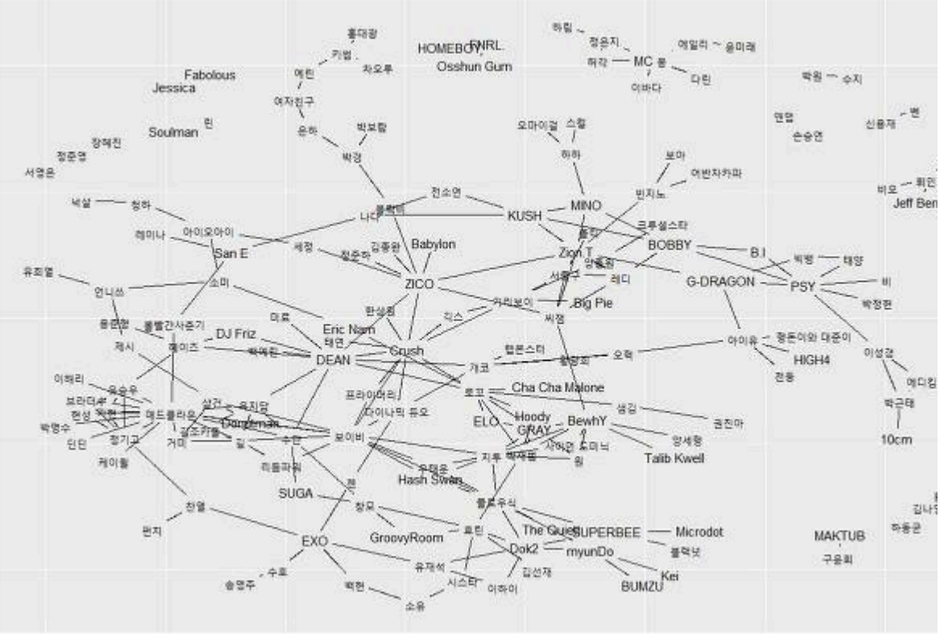
- 웹 등에서 방대한 텍스트 정보 크롤링
- 단어 빈도순으로 가중치 주어 시각화

01 R 활용 가능 분야 – 소셜 네트워크 분석

- 사람 간 관계 시각화
  - ➢ SNS 팔로우, 친구추가 데이터 사용

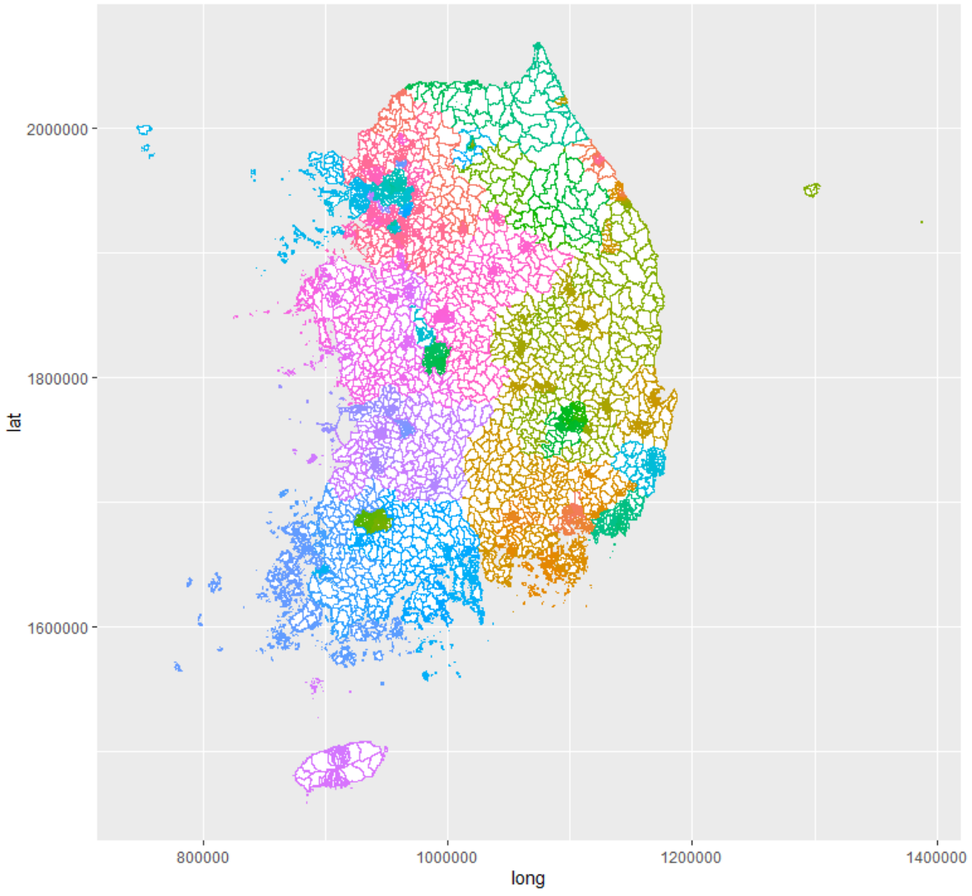
- 연예인 피처링 관계도 분석





01 R 활용 가능 분야 – 소셜 네트워크 분석

- **행정구열 별 정보 시각화**



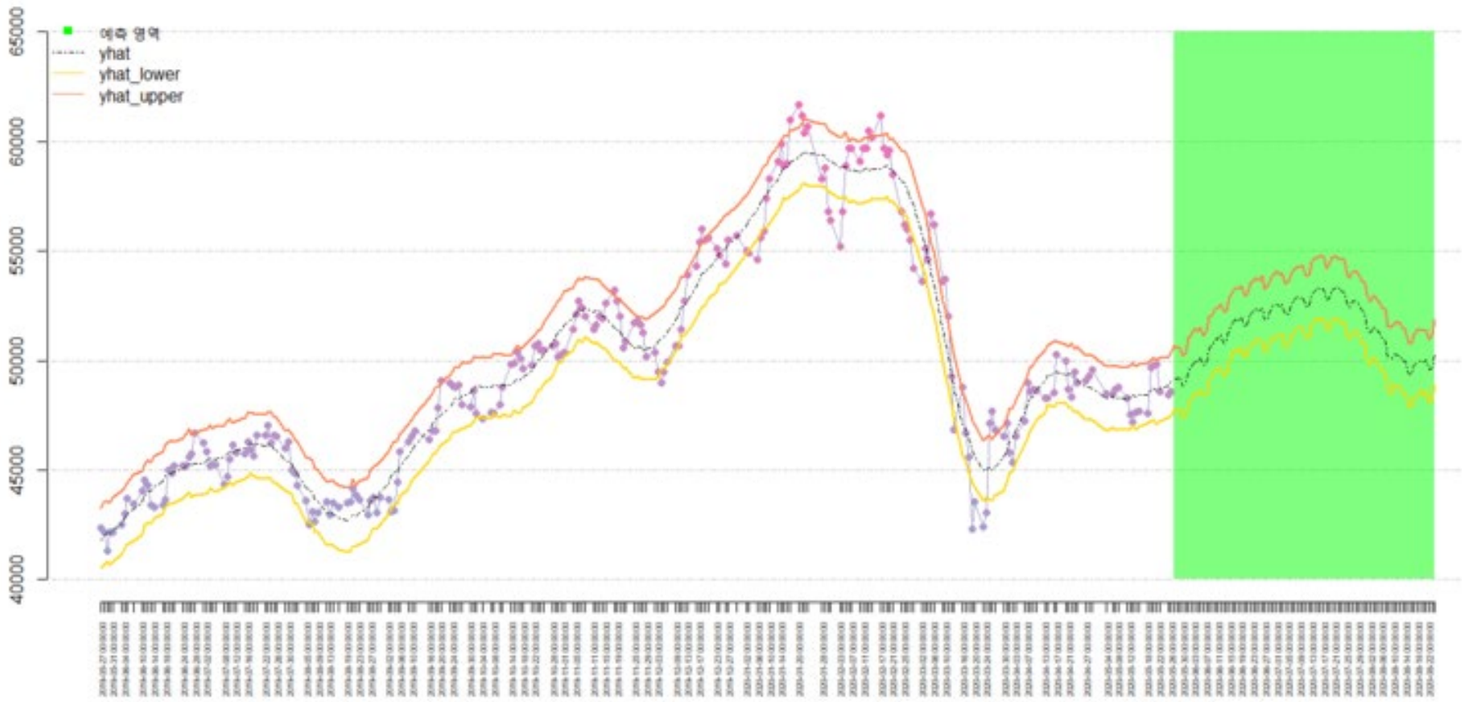
01 R 활용 가능 분야 – 소셜 네트워크 분석

- 서울시 구별 정보 시각화

01 R 활용 가능 분야 – 주식 분석

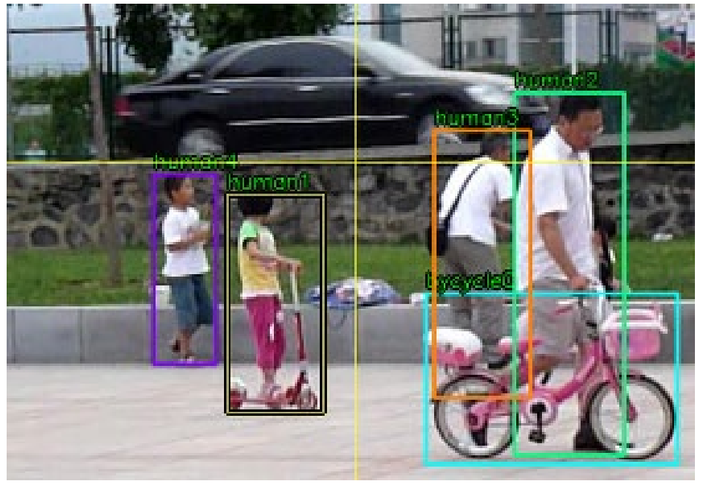
- 증권사 등에서 제공하는 API 사용하여 데이터 수집

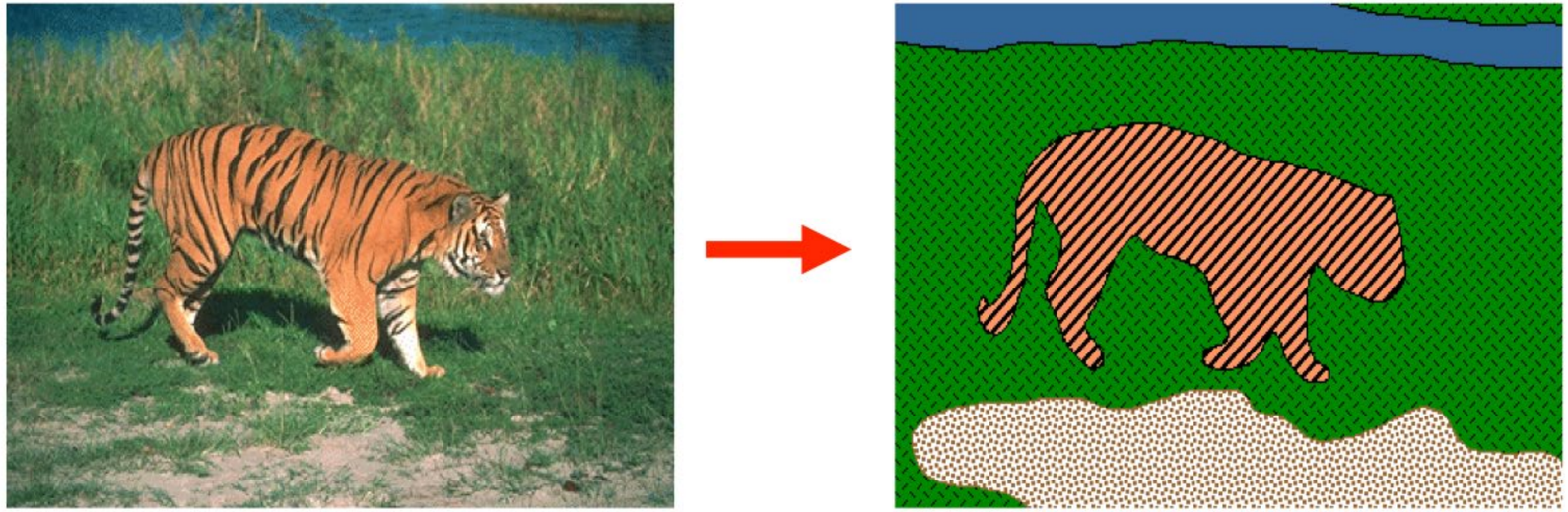
- 통계 패키지를 사용한 주가 예측



01 R 활용 가능 분야 - 이미지 분석

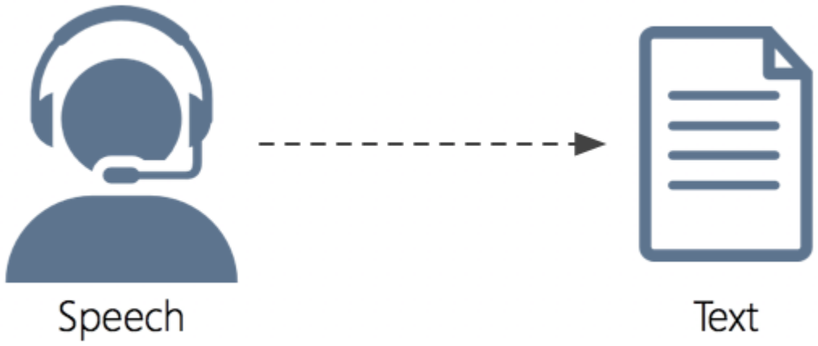
- 이미지 라벨링
- 이미지 구획 나누기

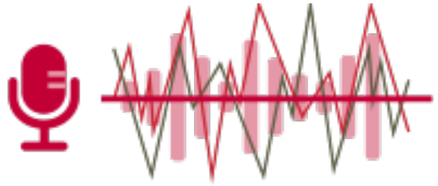




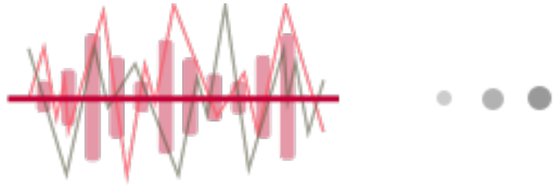
01 R 활용 가능 분야 - 사운드 분석

- 아날로그 소리 데이터를 디지털 벡터 데이터로 변환
- SST(Speech To Text)
- 음성 품질 향상





원본(음성+잡음)



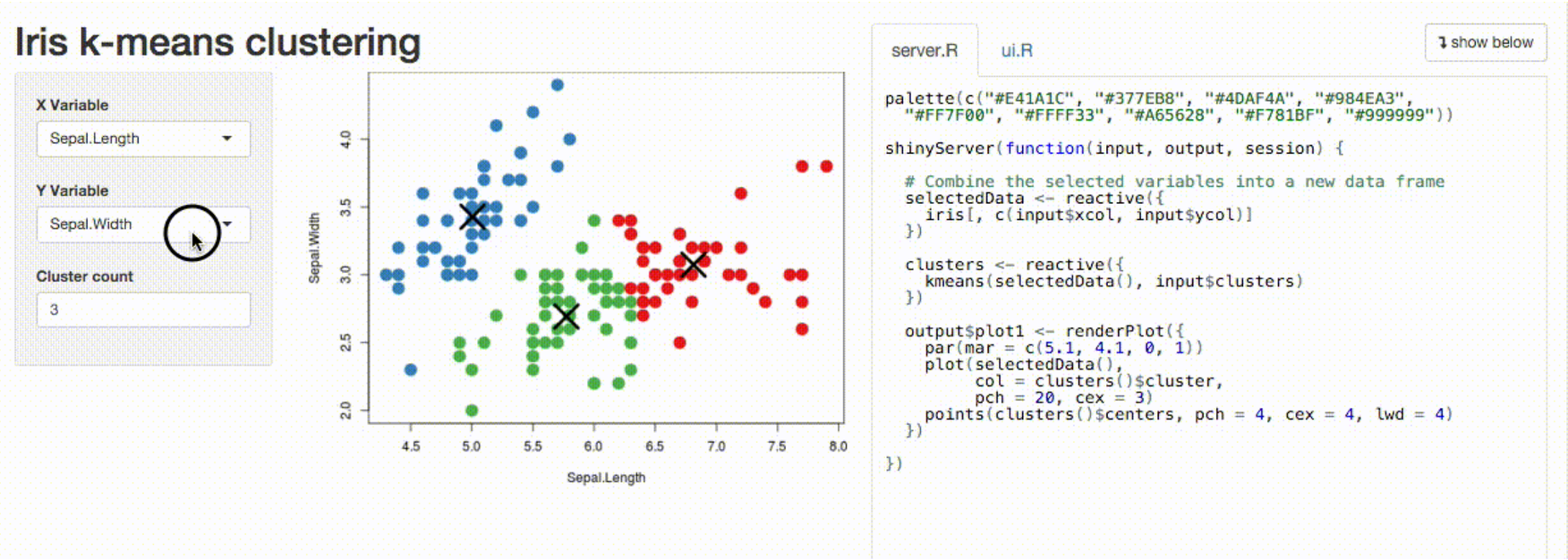
잡음 제거

깨끗한 음성

01 R 활용 가능 분야 – 웹 애플리케이션 개발

- R 언어만을 사용하여 웹에서 데이터 분석 내용 시각화 가능
  - ➢ 데모 시연 용으로 주로 사용 됨

- 이외에도 논문, 발표 슬라이드 등 제작 가능



이번 차시 정리

데이터 마이닝

R 활용 가능 분야

- 텍스트 마이닝
- 소셜 네트워크 분석
- 지도 시각화
- 주식 분석
- 이미지 분석
- 사운드 분석
- 웹 애플리케이션 개발

3차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

4차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 데이터 처리과정에 대해 알 수 있다.
- R의 역할에 대해 알 수 있다.

학습 목차

- 데이터 처리과정
- R의 역할

01.

데이터 처리과정

01 데이터 처리과정 – 개요

- **5단계 과정으로 정의**

문제 정의	데이터 수집	전처리	분석	결과 정리
• 가설 설정 • 분석방향 기획 • 방법론 검토	• 수동 수집 • 자동 수집 • 데이터 파악	• 결측치 처리 • 이상치 처리 • 속성 선별	• 통계 분석 • 시각화	• 분석 결과 해석 • 보고서 정리 • 데모 시연

- **예제 상황을 통해 데이터 처리과정을 알아본다**

01 데이터 처리과정 – 문제정의

- **문제 상황**
  - ➢ **대형마트 특가 상품과 함께 판매할 아이템 선정**

데이터 마이닝?

특가 상품과 함께 결재 되는 아이템 로그를 수집/분석하면 유의미한 정보를 얻을 수 있다

분석방향 기획

데이터 관리팀에 데이터 요청 → 데이터 전처리 SI 요청 or 자체 인력 사용 → 분석 → 보고서 작성

방법론 검토

후보 아이템 히스토그램 시각화, 피어슨 상관계수 기반 분석

문제 정의

- 가설 설정
- 분석방향 기획
- 방법론 검토

01 데이터 처리과정 – 데이터 수집

수동 수집

- 직접 데이터 수집
  - ➢ 설문조사, 관찰 등
- 데이터셋 구매

자동 수집

- 프로그래밍을 통한 자동화
  - ➢ 크롤링, 센서기반 데이터 자동 수집

데이터 파악

- 데이터 규모
  - ➢ 11x5 size
- 데이터 속성 값
  - ➢ 이름, 전화번호, 나이, 특가 제품 구매 유무, 구매한 다른 제품

데이터 수집

- 수동 수집
- 자동 수집
- 데이터 파악

이름	전화번호	나이	특가 제품 구매 유무	구매한 다른 제품
홍xx	0101234xxxx	55	Y	우유
김xx	010xxxxxxxx	23	Y	라면
이xx	010xxxxxxxx	44	Y	과자, 빵
박xx	010xxxxxxxx	53	Y	
신xx	010xxxxxxxx	32	N	우유
이xx	010xxxxxxxx	16	N	대형 스포츠세단
남궁xx	010xxxxxxxx	19	Y	휴지
남xx	010xxxxxxxx	20		우유
배xx	010xxxxxxxx	21	Y	빵
노xx	010xxxxxxxx	24	Y	우유
김xx	010xxxxxxxx	43	N	방향제

01 데이터 처리과정 – 데이터 전처리

Data in the real world is dirty

- 데이터 클리닝 필요

전처리

- 결측치 처리
- 이상치 처리
- 속성 선별

결측치 처리

- Missing value
- 값이 없는 경우 처리

이상치 처리

- Error value
- 잘못된 값 처리

속성 선별

- 분석에 필요한 속성만을 선별하는 작업

제거

이름	전화번호	나이	특가 제품 구매 유무	구매한 다른 제품
홍xx	0101234xxxx	55	Y	우유
김xx	010xxxxxxxx	23	Y	라면
이xx	010xxxxxxxx	44	Y	과자, 빵
박xx	010xxxxxxxx	53	Y	
신xx	010xxxxxxxx	32	N	우유
이xx	010xxxxxxxx	16	N	대형 스포츠세단
남궁xx	010xxxxxxxx	19	Y	휴지
남xx	010xxxxxxxx	20		우유
배xx	010xxxxxxxx	21	Y	빵
노xx	010xxxxxxxx	24	Y	우유
김xx	010xxxxxxxx	43	N	방향제

01 데이터 처리과정 – 데이터 전처리

Data in the real world is dirty

- 데이터 클리닝 필요

결측치 처리

- Missing value
- 값이 없는 경우 처리

이상치 처리

- Error value
- 잘못된 값 처리

속성 선별

- 분석에 필요한 속성만을 선별하는 작업

전처리

- 결측치 처리
- 이상치 처리
- 속성 선별

전처리 된 데이터

특가 제품 구매 유무	구매한 다른 제품
Y	우유
Y	라면
Y	과자, 빵
Y	
N	우유
Y	휴지
Y	빵
Y	우유
N	방향제

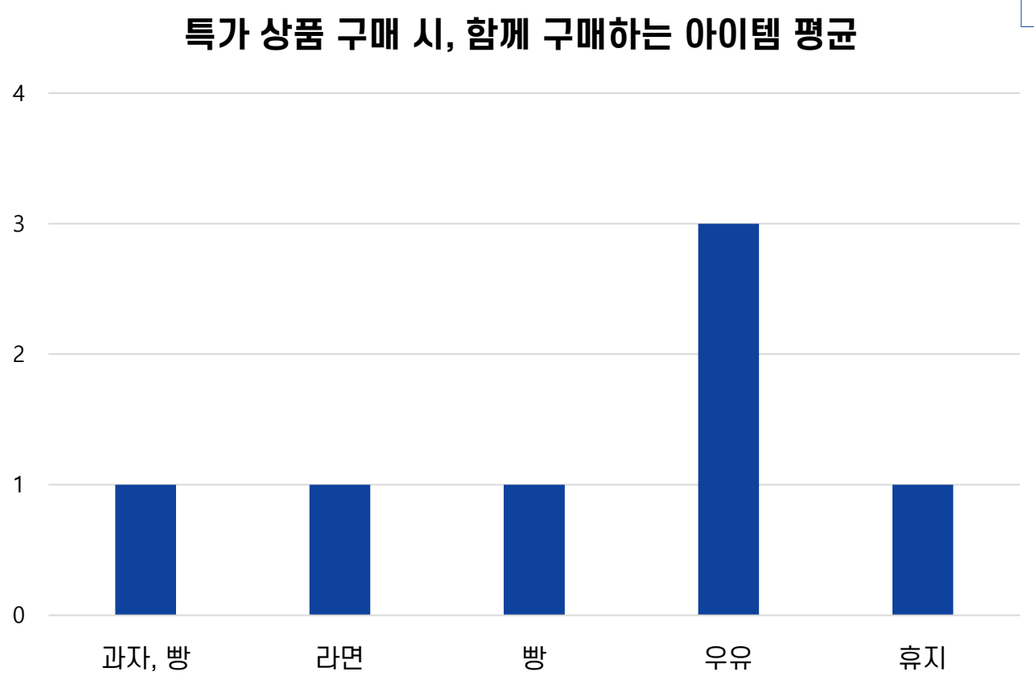
01 데이터 처리과정 – 데이터 분석

통계분석

- 통계학에 기반한 분석법
- Eg. 상관계수 추출

시각화

- 히스토그램
- Box plot
- 원 도표
- 그래프
- Etc.



분석

- 통계 분석
- 시각화

01 데이터 처리과정 – 결과 정리

분석 결과 해석

- 통계, 시각화 자료를 기반으로 가설 성립 여부 예측

보고서 정리

- 데이터 처리 과정 요약 및 분석 결과 기반한 주장
  - ➢ "… 데이터 수집하여 … 통계, 시각화 분석 결과…. 결론적으로, 우유를 함께 파는게 적합하다고 판단 됨."

데모 시연

결과 정리

- 분석 결과 해석
- 보고서 정리
- 데모 시연

02.

R의 역할

02 R의 역할

- **문제 정의를 제외한 모든 과정에서 사용 가능**

문제 정의

- 가설 설정
- 분석방향 기획
- 방법론 검토

데이터 수집

- 수동 수집
- 자동 수집
- 데이터 파악

- 웹 크롤링 패키지 有
- 데이터 정렬 용이
- 학습용 테스트셋 다수 제공

전처리

- 결측치 처리
- 이상치 처리
- 속성 선별

- 전처리를 위한 자료구조 제공
- 강력한 전처리 패키지 지원

분석

- 통계 분석
- 시각화

- 통계 분석에 특화된 언어
- 강력한 시각화 패키지 지원

결과 정리

- 분석 결과 해석
- 보고서 정리
- 데모 시연

- 논문 제작 기능 有
- 웹 기반 데모 제작 기능 有
- 발표 스크립트 제작 기능 有

이번 차시 정리

데이터 처리과정

문제 정의 → 데이터 수집 → 전처리 → 분석 → 결과 정리

R의 역할

- 데이터 수집
- 전처리
- 분석
- 결과 정리

4차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

5차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- R을 설치할 수 있다.
- R Studio를 설치할 수 있다.

학습 목차

- R의 설치
- R Studio의 설치

01.

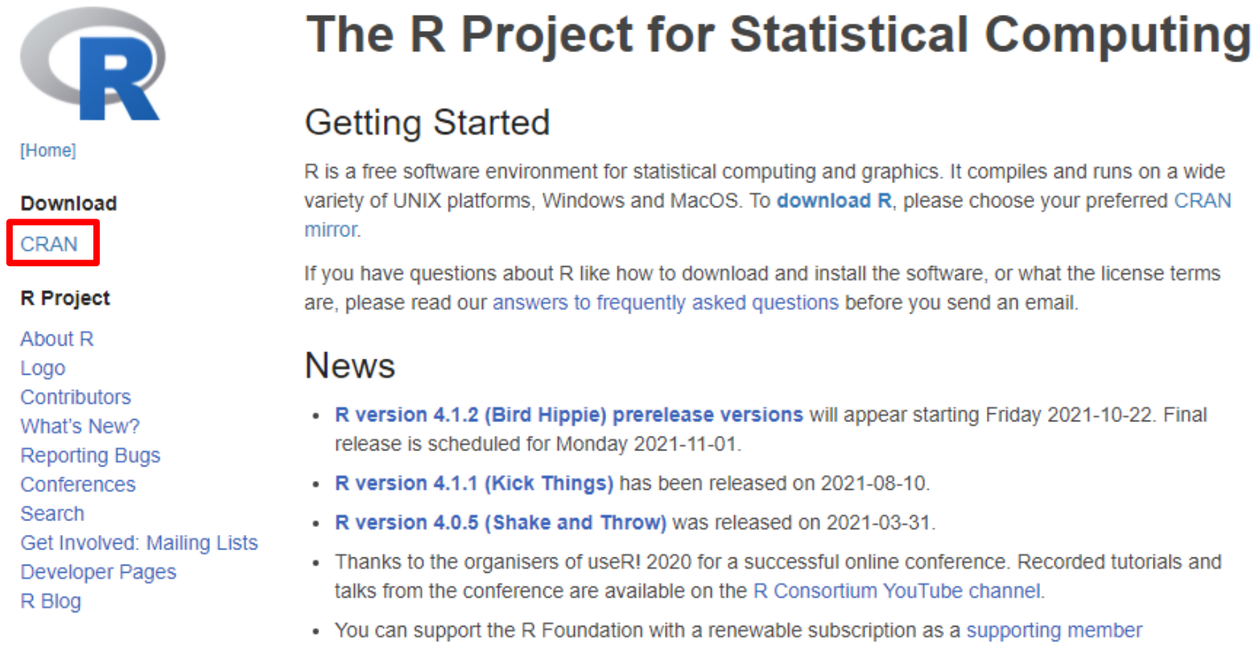
R 설치

01 R 설치 – 홈페이지 진입

1. R Project 홈페이지 접속

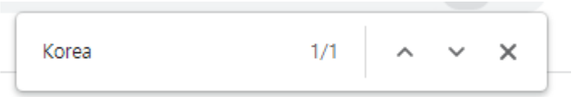
- https://www.r-project.org/

2. 왼쪽 상단 “CRAN” 버튼 클릭



01 R 설치 – CRAN

3. Ctrl + F 눌러 탐색창 연 후 “Korea” 입력



4. Korea 영역 내부 아무 링크 선택하여 다운로드 페이지 진입

https://ftp.yz.yamagata-u.ac.jp/pub/cran/

Korea

https://ftp.harukasan.org/CRAN/

https://cran.yu.ac.kr/

https://cran.seoul.go.kr/

https://cran.biodisk.org/

Malaysia

https://mirrors.upm.edu.my/CRAN/

Mexico

https://cran.itam.mx/

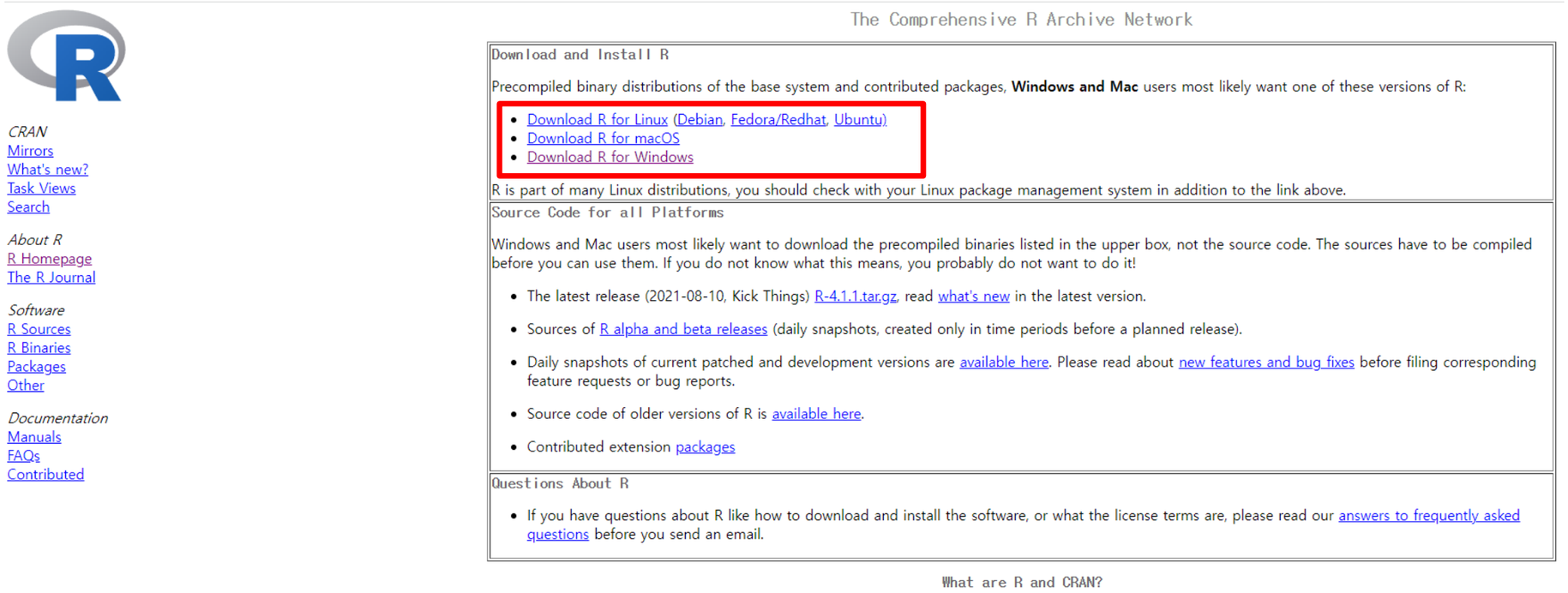
https://www.est.colpos.mx/

Morocco

01 R 설치 – R 다운로드 페이지 진입

5. 중앙 상단, 자신 OS에 알맞은 링크 클릭

- 강의 자료에서는 Windows 선택함



CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux (Debian, Fedora/Redhat, Ubuntu)
- Download R for macOS
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2021-08-10, Kick Things) R-4.1.1.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

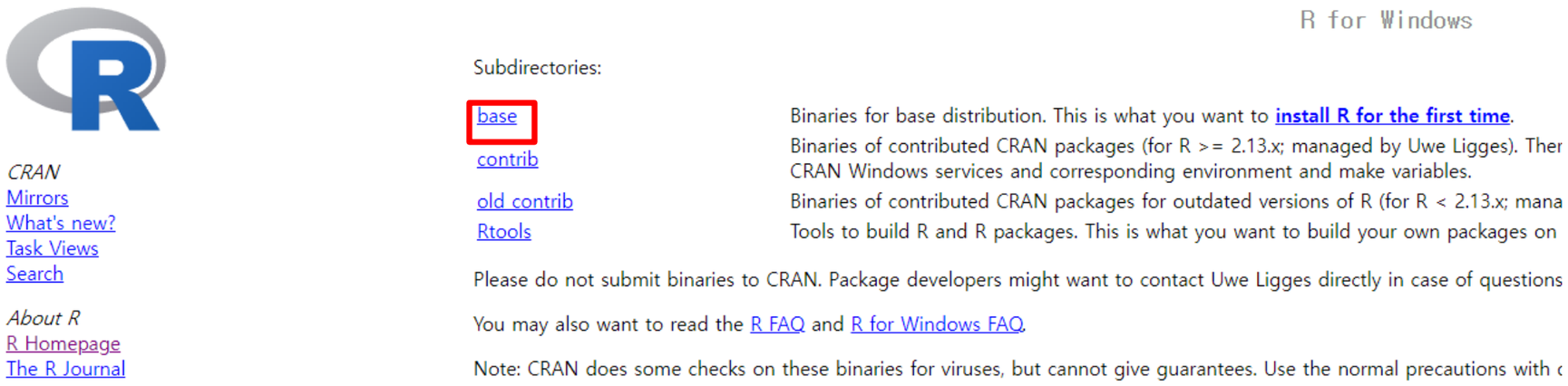
Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

What are R and CRAN?

01 R 설치 – R 설치파일 다운로드

6. “base” 링크 클릭



01 R 설치 – R 설치파일 다운로드

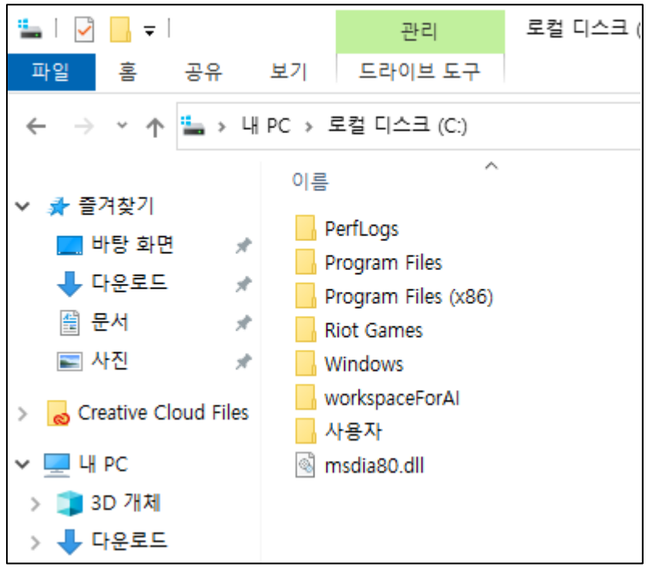
7. 중앙 상단, “Download R x.x.x for XXX” 클릭

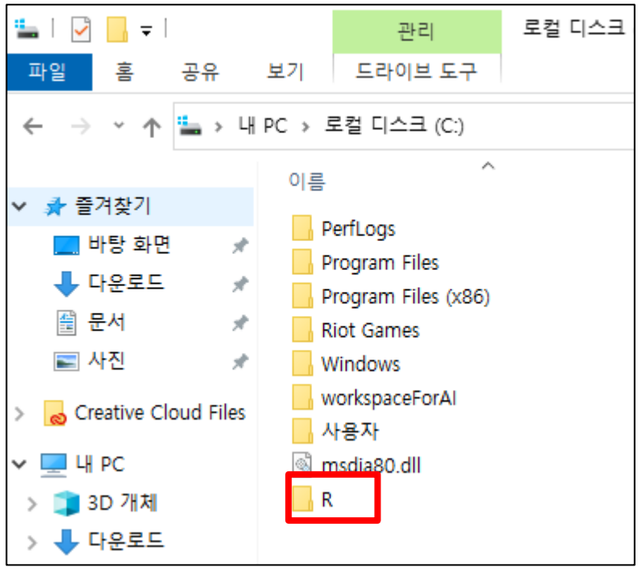
- 설치 파일 자동 다운로드 됨
- 다운로드에 수십 분 소요

01 R 설치 – 작업 경로 설정

8. R 설치용 폴더 생성

- 파일 탐색기 열기 → C드라이브(C:₩) 진입 → 마우스 우클릭 → 새 폴더 생성 → 이름 “R”로 변경
- 정확히 따르지 않을 시, 환경설정이 올바르게 진행되지 않을 수 있음

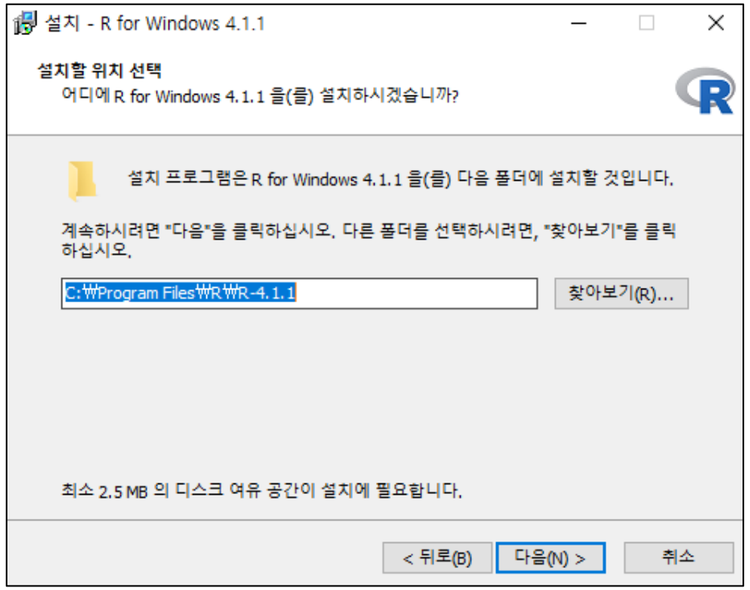


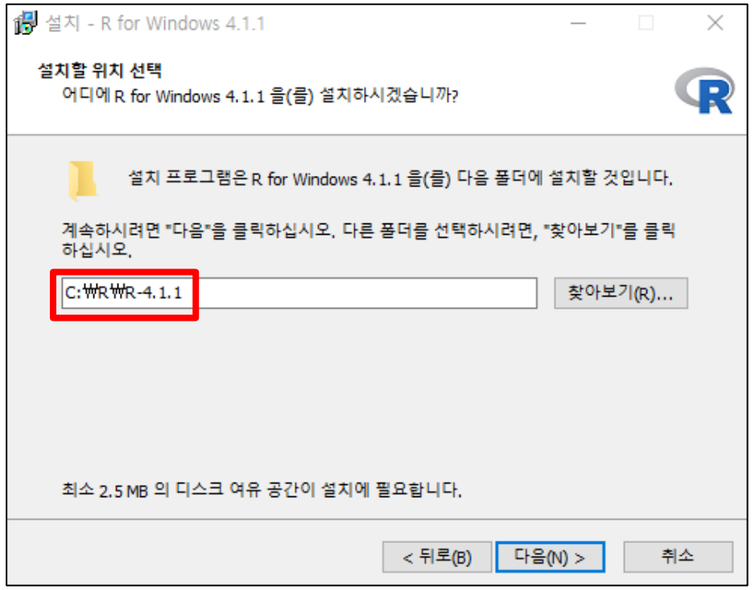


01 R 설치 – 설치 파일 실행

8. R 설치파일 실행

- “다음”을 눌러 진행
- “설치할 위치 선택” 창이 나오면 경로를 이전에 만든 R 폴더로 변경
  - C:₩R₩R-x.x.x





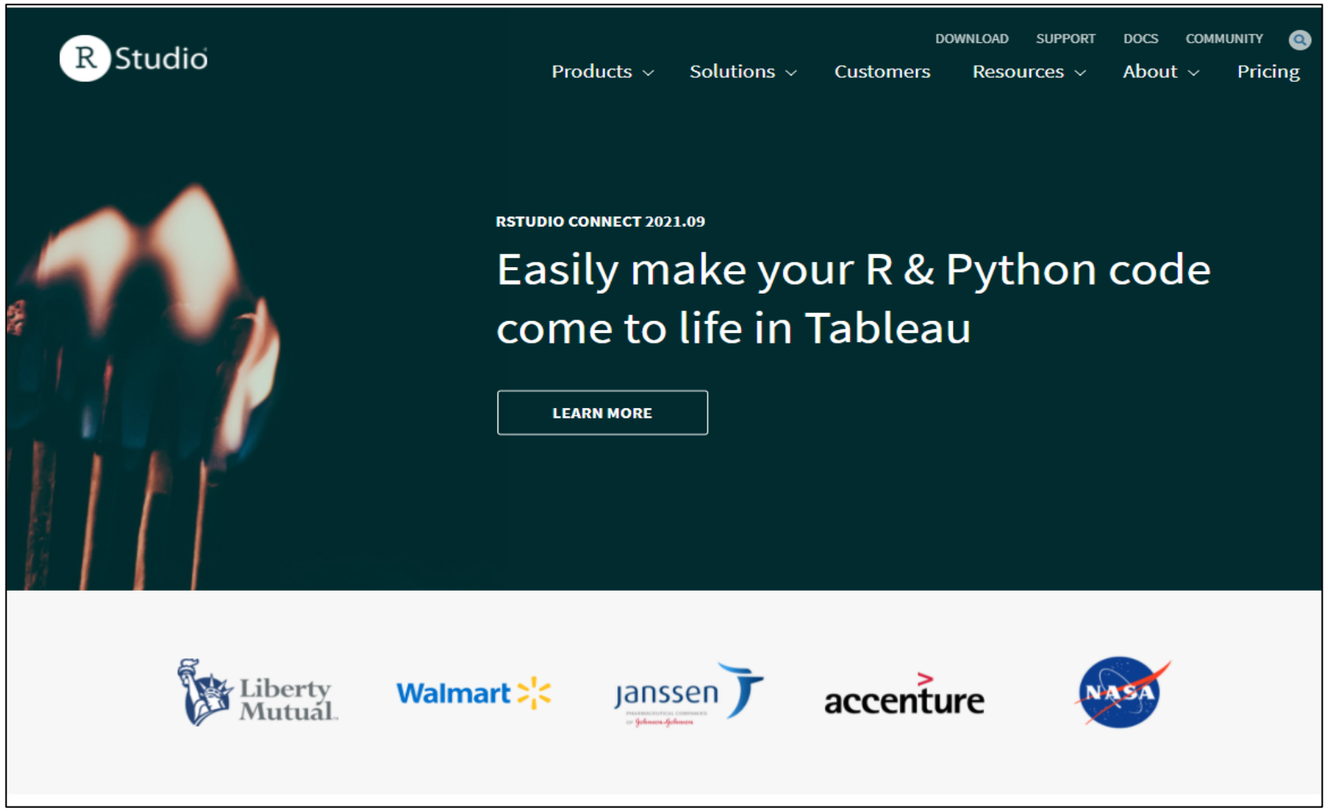
02.

R 스튜디오 설치

01 R 스튜디오 설치 – 홈페이지 진입

1. R Studio 홈페이지 진입

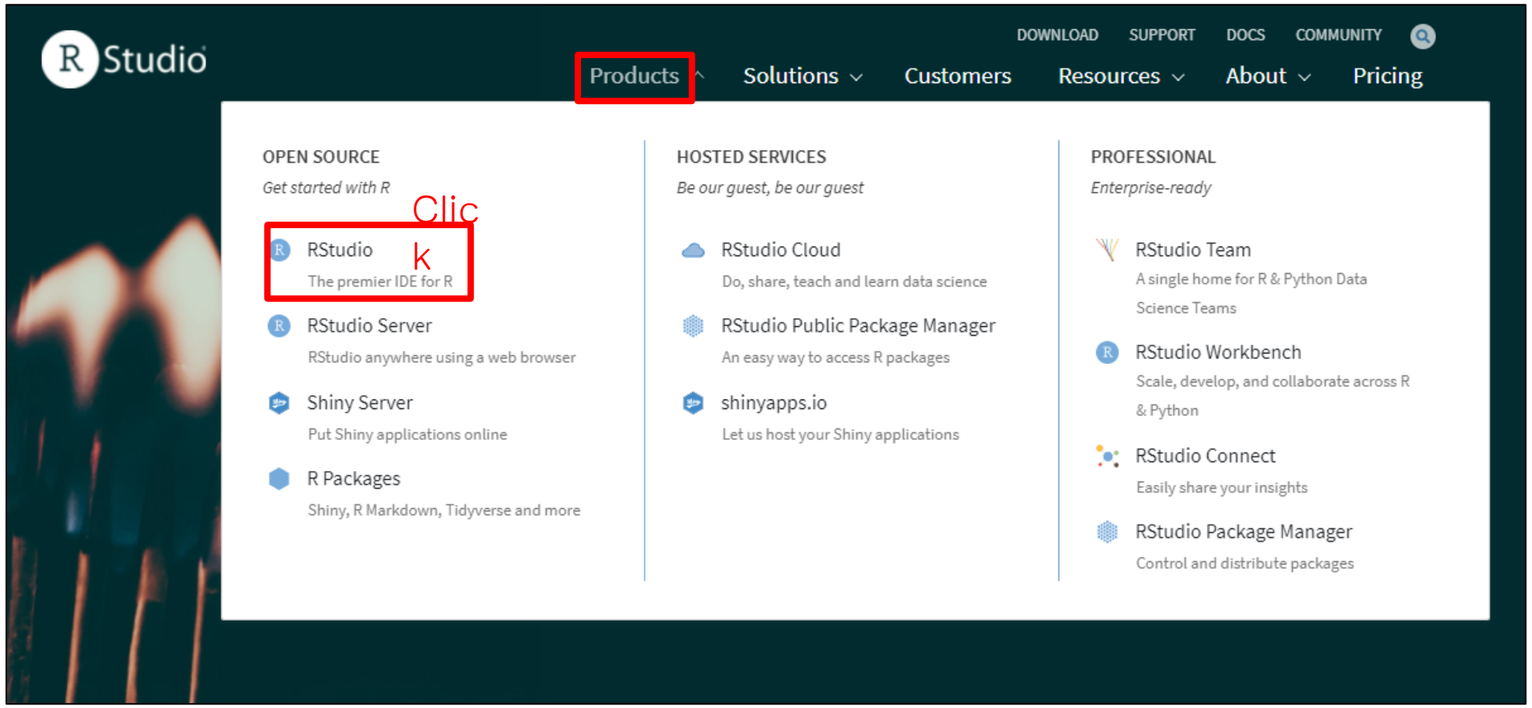
- https://www.rstudio.com



01 R 스튜디오 설치 - 다운로드 페이지 진입(1)

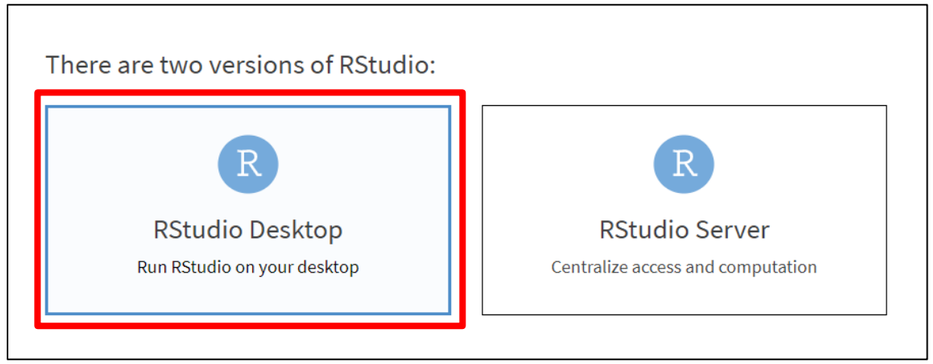
2. 다운로드 페이지 진입

- 중앙 상단 “Products”에 마우스 올려놓은 후
- 좌측 상단 Rstudio 클릭



01 R 스튜디오 설치 – 다운로드 페이지 진입(2)

3. 페이지 약간 내린 후 “Rstudio Desktop” 버전 버튼 클릭



01 R 스튜디오 설치 - 다운로드 페이지 진입(2)

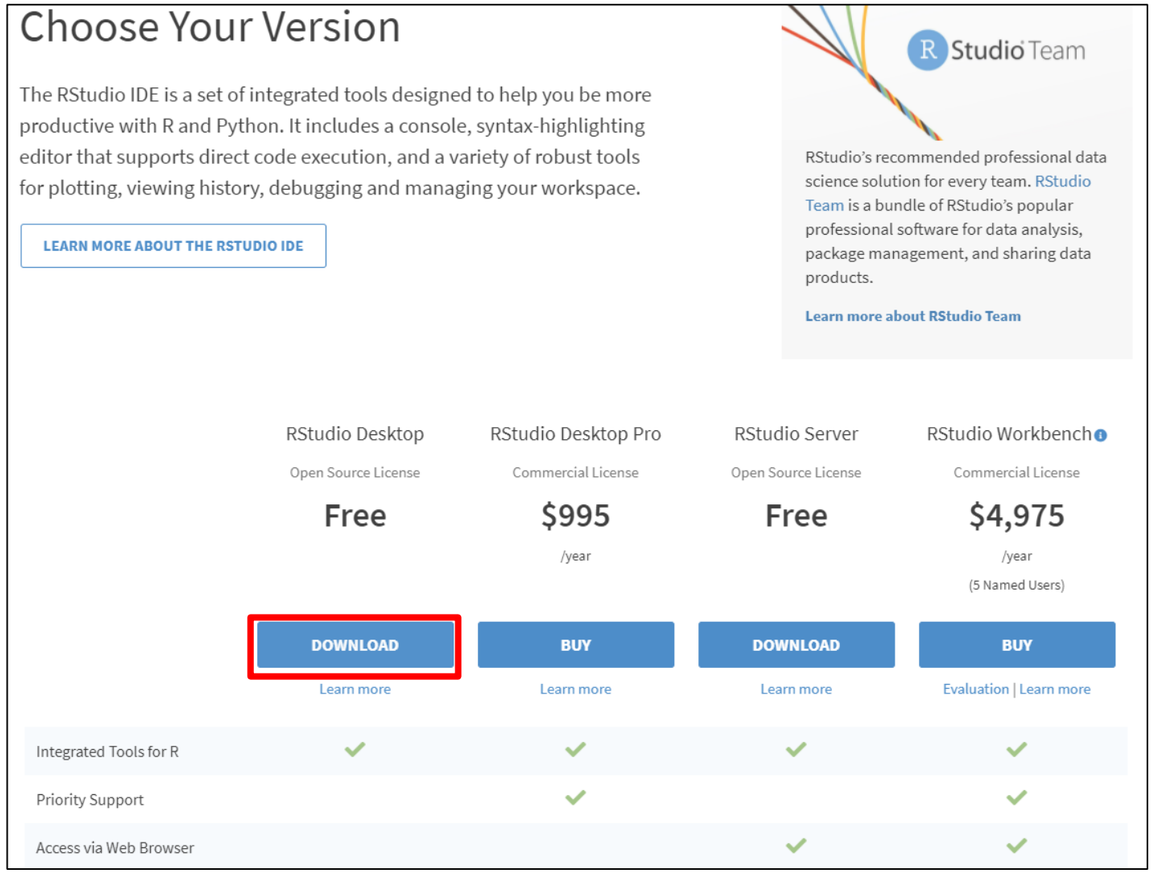
4. “DOWNLOAD RSTUDIO DESKTOP” 버튼 클릭

RStudio Desktop

	Open Source Edition	RStudio Desktop Pro
Overview	• Access RStudio locally • Syntax highlighting, code completion, and smart indentation • Execute R code directly from the source editor • Quickly jump to function definitions • View content changes in real-time with the Visual Markdown Editor • Easily manage multiple working directories using projects • Integrated R help and documentation • Interactive debugger to diagnose and fix errors • Extensive package development tools	All of the features of open source; plus: • A commercial license for organizations not able to use AGPL software • Access to priority support • RStudio Professional Drivers • Connect directly to your RStudio Workbench instance remotely
Support	Community forums only	• Priority Email Support • 8 hour response during business hours (ET)
License	AGPL v3	RStudio License Agreement
Pricing	Free	$995/year
	DOWNLOAD RSTUDIO DESKTOP	DOWNLOAD FREE RSTUDIO DESKTOP PRO TRIAL

01 R 스튜디오 설치 - 다운로드 페이지 진입(3)

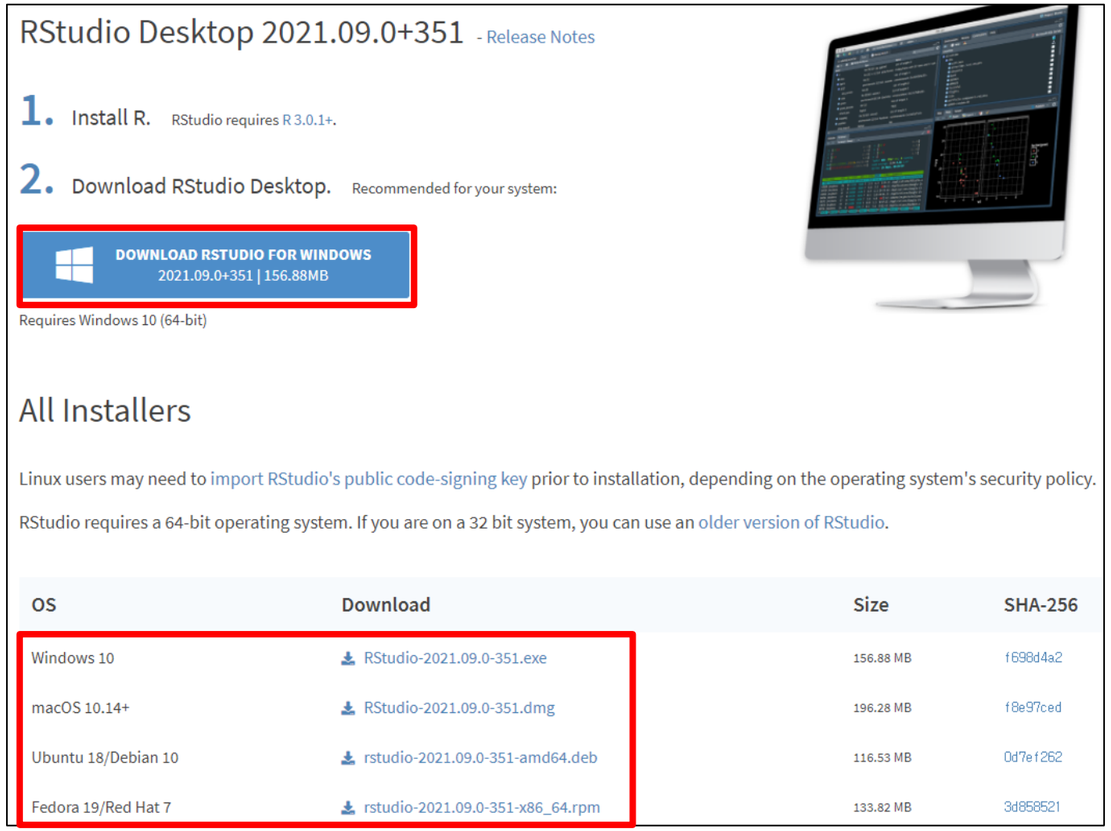
5. “RStudio Desktop” 버전 “DOWNLOAD” 버튼 클릭



01 R 스튜디오 설치 - 설치 파일 다운로드

6. 자신 OS에 알맞은 설치파일 다운로드

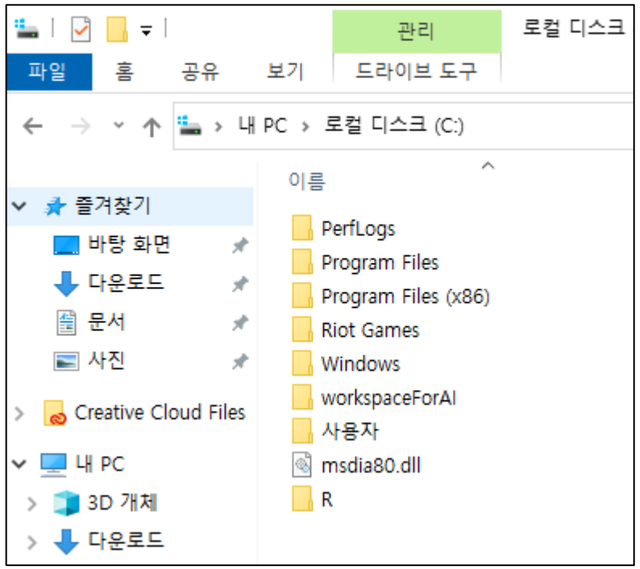
- 반드시 R 설치가 완료된 후 진행 해야함

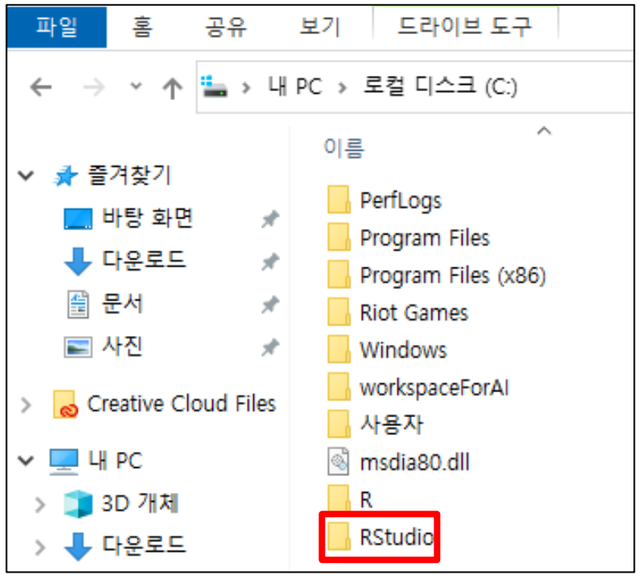


01 R 스튜디오 설치 – 작업 경로 설정

7. RStudio 설치용 폴더 생성

- 파일 탐색기 열기 → C드라이브(C:₩) 진입 → 마우스 우클릭 → 새 폴더 생성 → 이름 “RStudio”로 변경

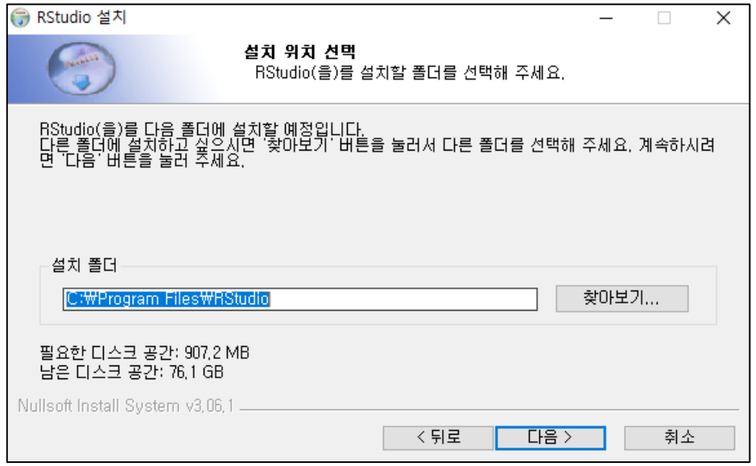


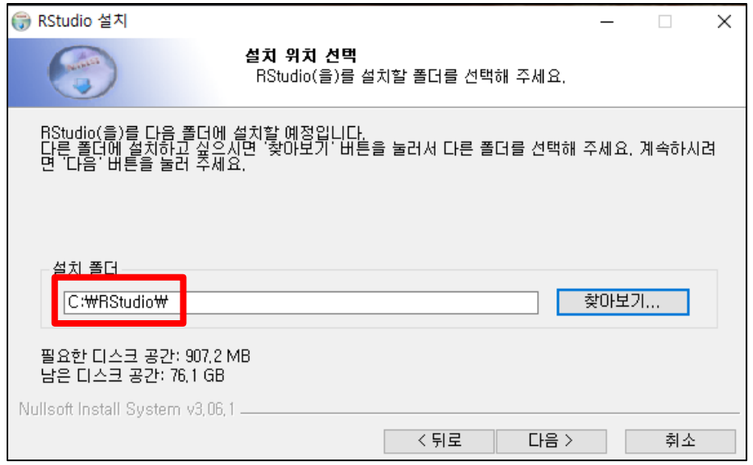


01 R 스튜디오 설치 - 설치 파일 실행

8. Rstudio 설치파일 실행

- "다음"을 눌러 진행
- "설치할 위치 선택" 창이 나오면 경로를 RStudio 폴더로 변경
  - C:₩Rstudio

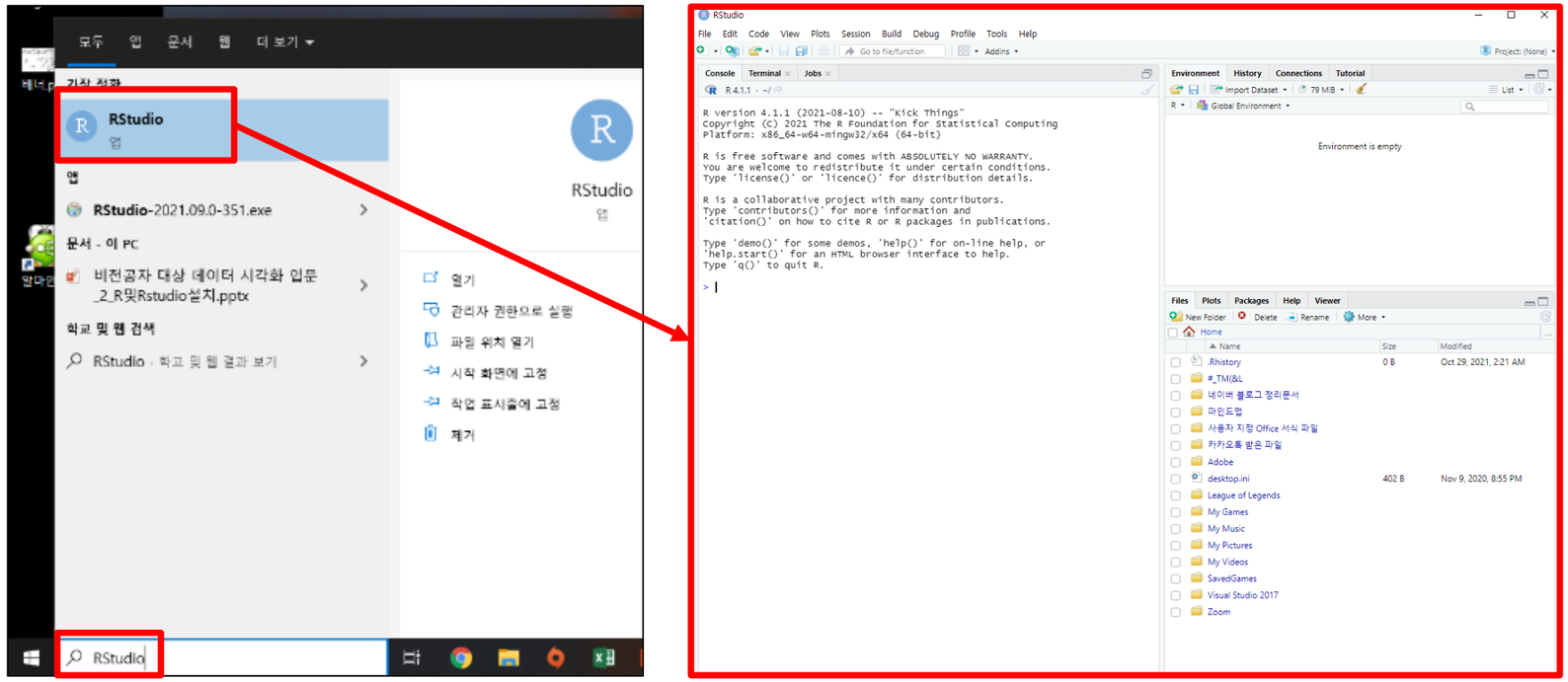




01 R 스튜디오 설치 - 설치 확인

9. 설치 확인

- 윈도우 버튼 → Rstudio 검색 → 앱 실행
- Rstudio 정상 실행 확인



이번 차시 정리

R 설치

- https://www.r-project.org/
- CRAN

RStudio 설치

- https://www.rstudio.com

5차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

6차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- RStudio 구성요소를 알 수 있다.
- R의 Console창을 알 수 있다.

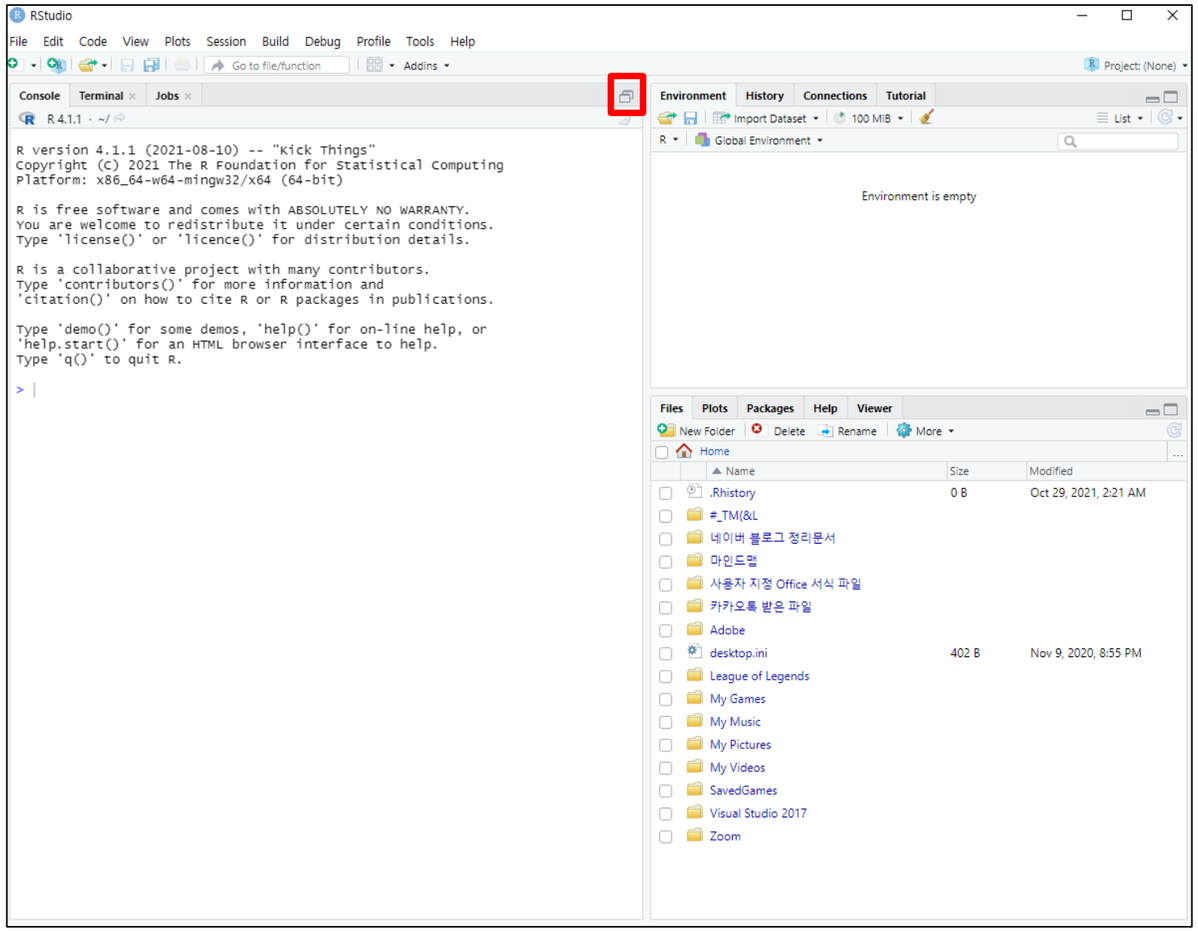
학습 목차

- Rstudio 구성요소
- R 맛보기 – Console 창

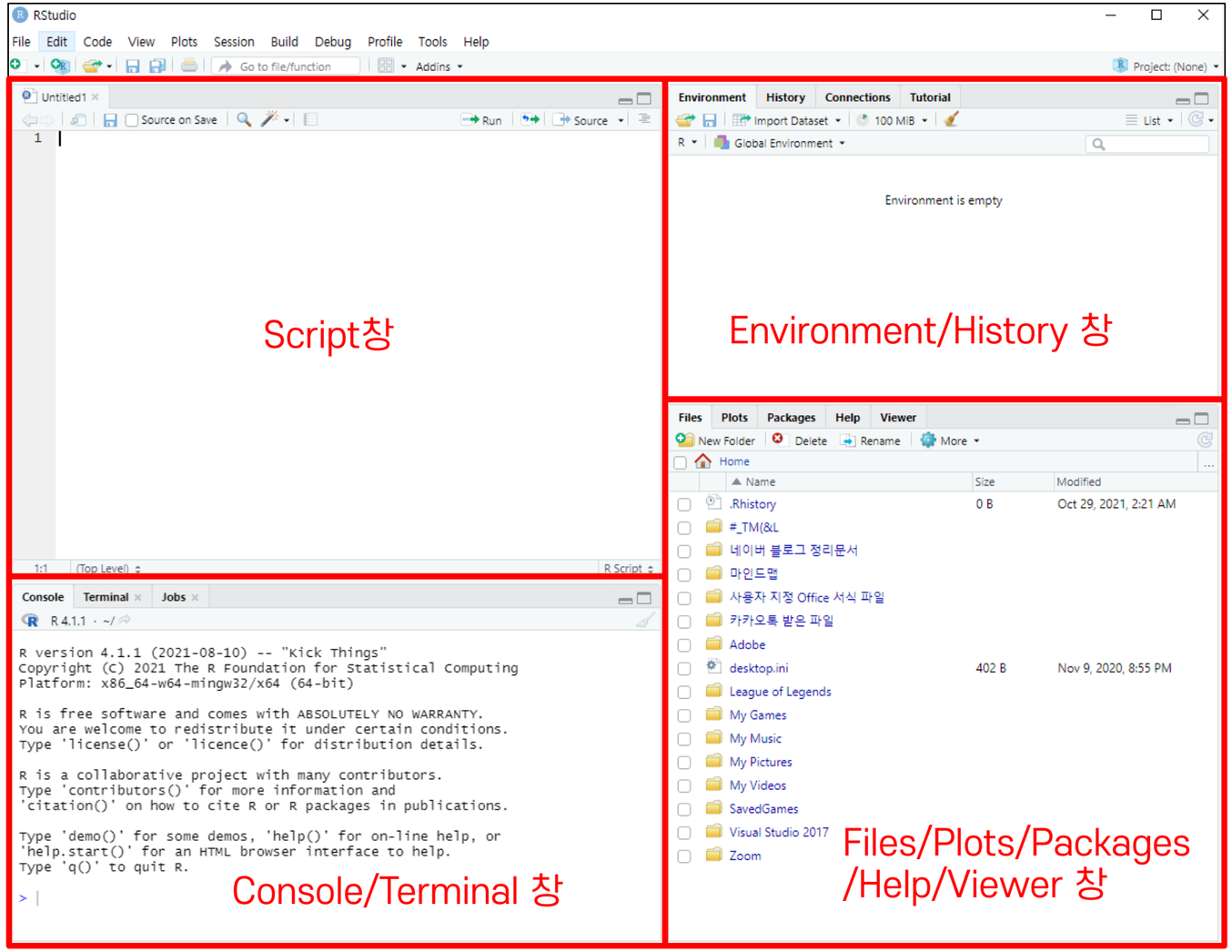
01. RStudio 구성요소

01 RStudio 구성요소 – 최초 실행 화면

1. 빨간 박스 클릭하여 Script창 열기



01 RStudio 구성요소 – 구성요소 오버뷰

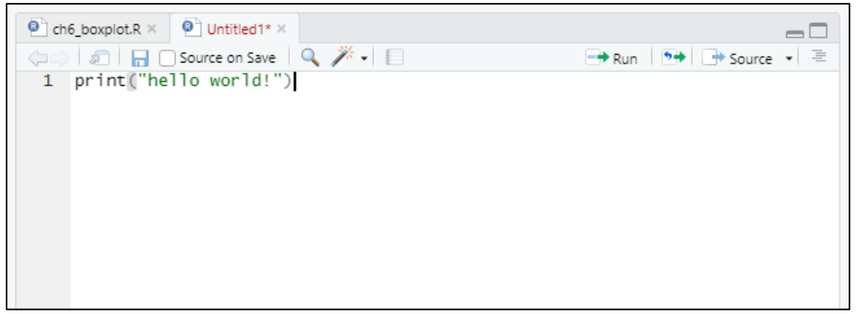


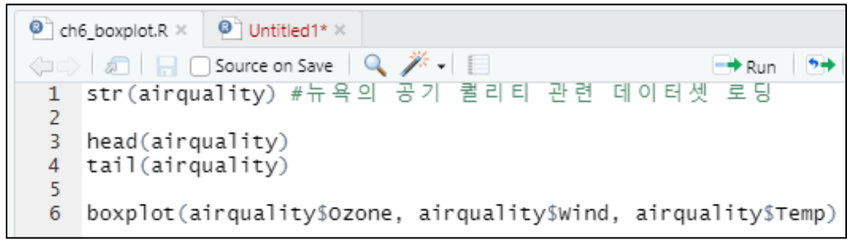
01 RStudio 구성요소 – Script 창

- **R 코드를 작성하는 창**
  - ➢ 작성한 코드를 파일로 저장 및 불러오기 가능
  - ➢ “.R” 확장자 사용

- **함수에 대한 자동완성 기능 제공**

- **Run 버튼 눌러 코드 실행**

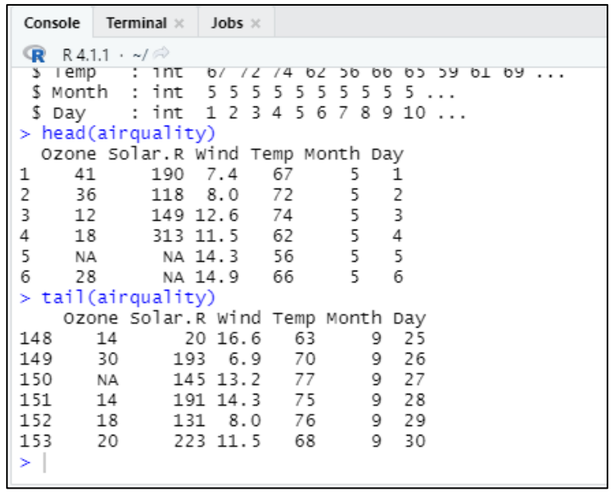




01 RStudio 구성요소 – Console/Terminal 창

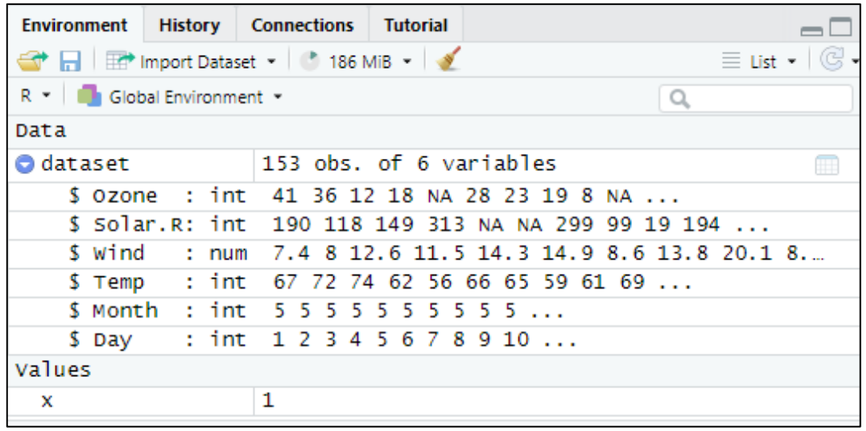
- **Console 창**
  - ➢ 간단한 코드 작성 및 실행
  - ➢ 코드 실행 결과 확인
  - ➢ 오류 확인

- **Terminal 창**
  - ➢ CMD 창과 동일
  - ➢ 커맨드라인 통한 OS 조작

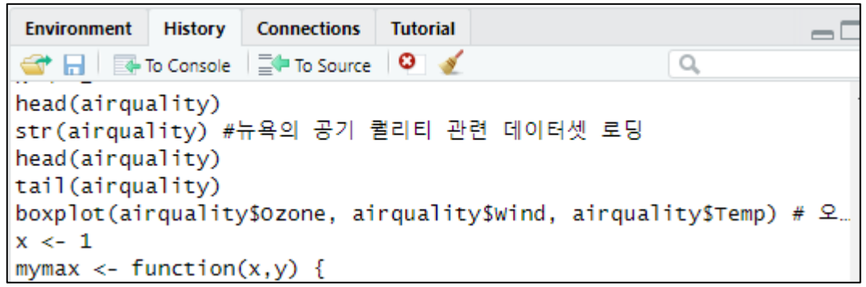


01 RStudio 구성요소 – Environment/History 창

- **Environment 창**
  - ➢ 로딩 된 데이터 셋 표시
  - ➢ 선언 된 변수/함수 표시



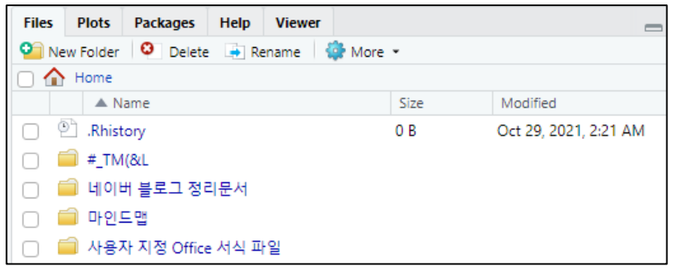
- **History 창**
  - ➢ 실행한 명령어(코드) 로깅



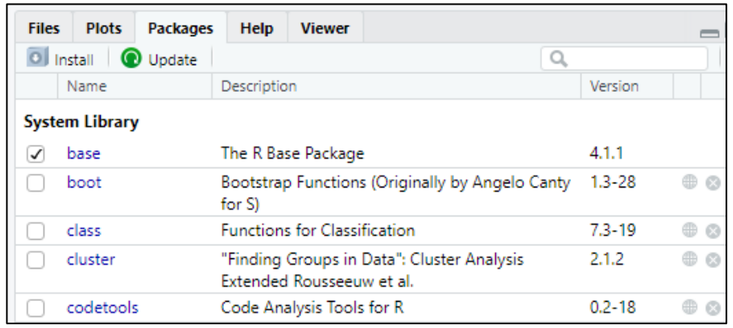
- **Connections 창**
  - ➢ DB 서버와 연결 관리

01 RStudio 구성요소 – File/Plots/Packages 창

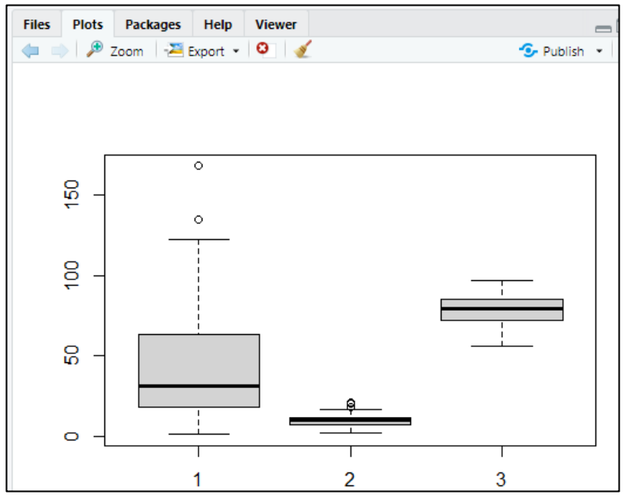
- **Files 창**
  - ➢ 파일 탐색기와 동일



- **Packages 창**
  - ➢ 패키지 관리



- **Plots 창**
  - ➢ 그래프 그린 결과물 출력



01 RStudio 구성요소 – Helper/Viewer 창

- **Helper 창**
  - ➢ 도움말

- **Viewer 창**
  - ➢ 웹 관련 THML코드 실행 결과물 출력 창

02.

R 맛보기

02 R 맛보기 – Console 창 사용해보기

- **기본적인 계산 기능**
  - ➢ 사칙연산
  - ➢ 제곱

```
Console  Terminal ×  Jobs ×
R 4.1.1 · ~/
>
> 1
[1] 1
> 10
[1] 10
> 20
[1] 20
> 1+5
[1] 6
> 4-2
[1] 2
> 10 * 2
[1] 20
> 3/5
[1] 0.6
> 10/2
[1] 5
```

- **문자열 출력 기능**
  - ➢ Print(“xxx”)
    - ▪ 따옴표(“”) 안에 적은 문자열 출력

```
Console  Terminal ×  Jobs ×
R 4.1.1 · ~/
>
>
> print("Hello world!")
[1] "Hello world!"
>
```

- **데이터 셋 로드**
  - ➢ 내장된 데이터셋 로딩
  - ➢ head #상위 6개 데이터 출력
  - ➢ tail #하위 6개 데이터 출력

```
Console  Terminal ×  Jobs ×
R 4.1.1 · ~/
>
>
>
> airquality
    Ozone Solar.R Wind Temp Month Day
1      41     190  7.4   67     5   1
2      36     118  8.0   72     5   2
3      12     149 12.6   74     5   3
4      18     313 11.5   62     5   4
5      NA      NA 14.3   56     5   5
6      28      NA 14.9   66     5   6
7      23     299  8.6   65     5   7
8      19      99 13.8   59     5   8
9       8      19 20.1   61     5   9
10     NA     194  8.6   69     5  10
11      7      NA  6.9   74     5  11
12     16     256  9.7   69     5  12
13     11     290  9.2   66     5  13
14     14     274 10.9   68     5  14
15     18      65 13.2   58     5  15
16     14     334 11.5   64     5  16
17     34     307 12.0   66     5  17
```

02 R 맛보기 – Quiz

- **문제**
  - Console 창 만을 사용
  - “airquality” 데이터 셋 상위 6개 데이터의 “Temp” 값 평균을 print함수 사용하여 출력

02 R 맛보기 – Quiz

- **문제**
  - ➢ Console 창 만을 사용
  - ➢ “airquality” 데이터 셋 상위 6개 데이터의 “Temp” 값 평균을 print함수 사용하여 출력

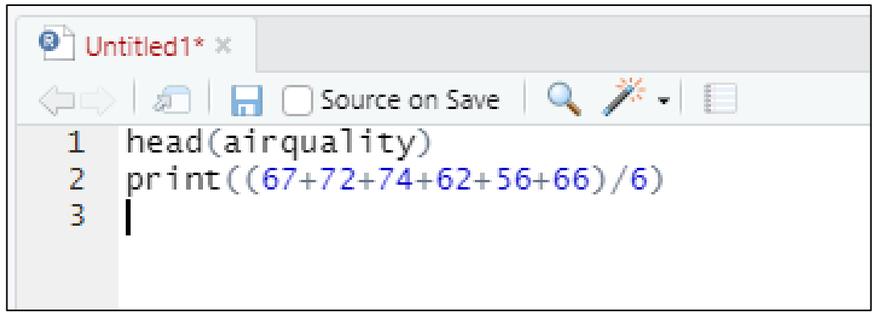
- **답**

1. head(airquality) #Temp값 눈으로 확인
2. print((67+72+74+62+56+66)/6)
   1) 66.16667

```
> head(airquality)
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
> print((67+72+74+62+56+66)/6)
[1] 66.16667
>
```

02 R 맛보기 – Script 실행해보기

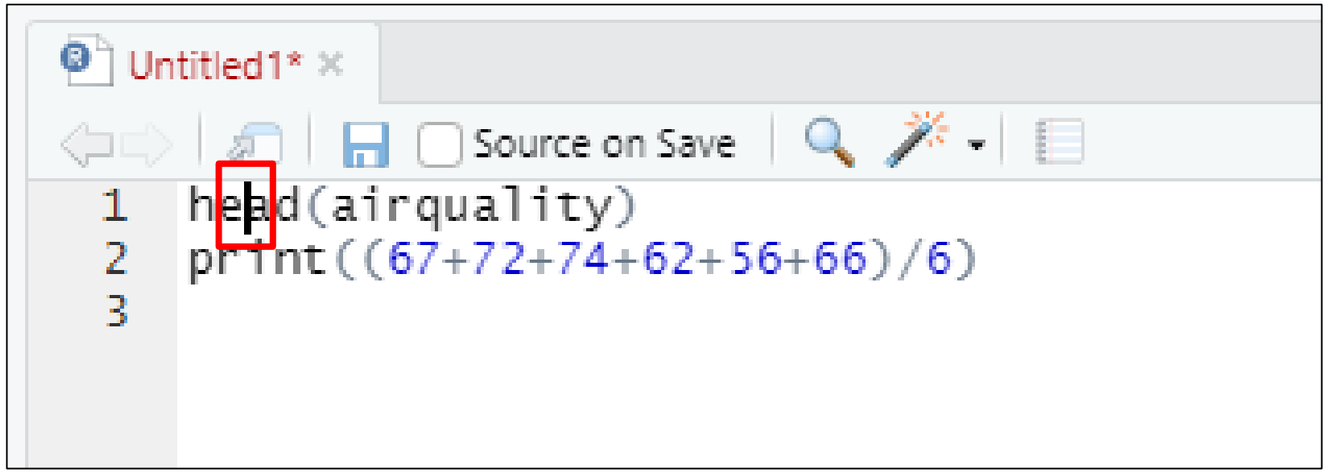
- Console 창에 적었던 코드 스크립트 창에 복사 붙여넣기



02 R 맛보기 – Script 실행해보기

- Console 창에 적었던 코드 스크립트 창에 복사 붙여넣기

- 마우스 커서 첫번째 줄에 두기

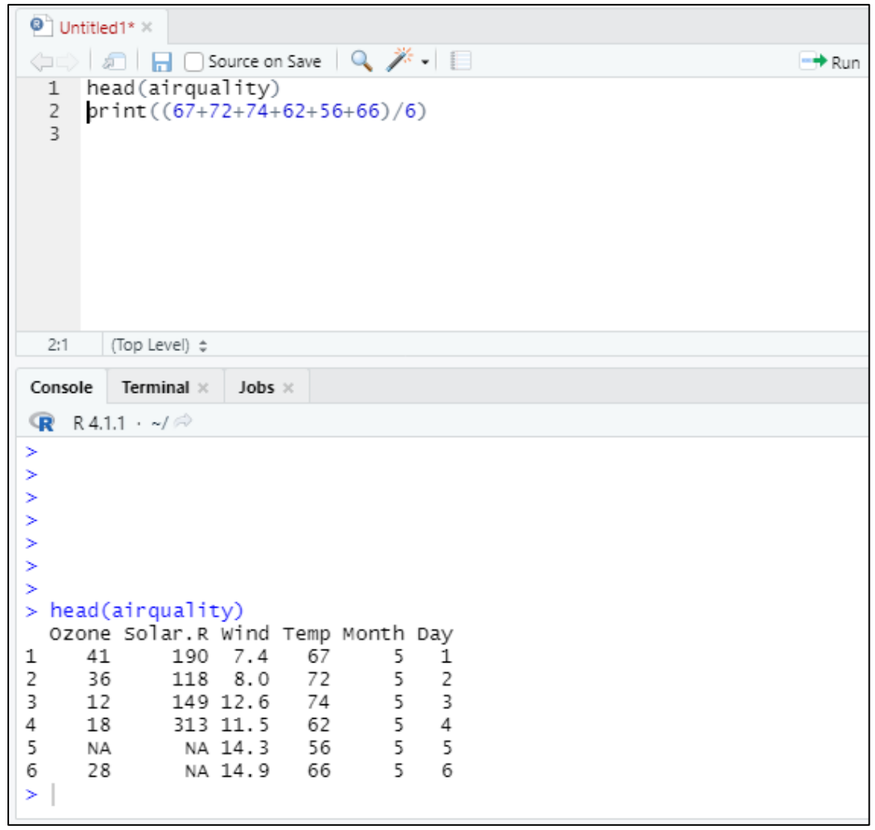


02 R 맛보기 – Script 실행해보기

- **Console 창에 적었던 코드 스크립트 창에 복사 붙여넣기**

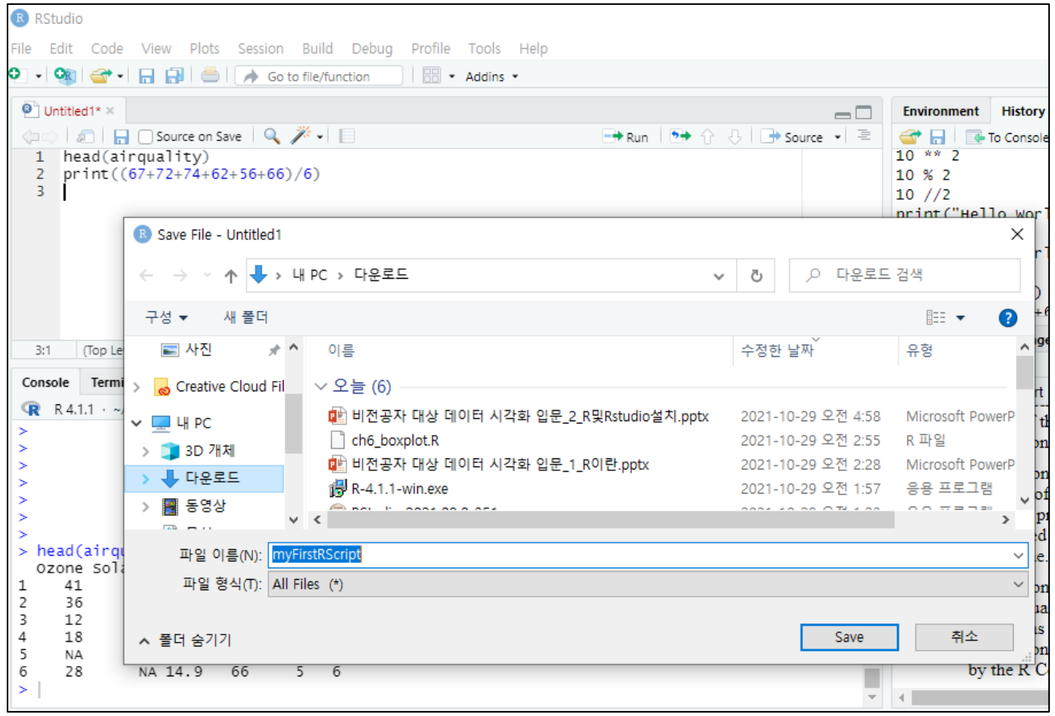
- **마우스 커서 첫번째 줄에 두기**

- **Run(Run)버튼 눌러 실행**
  - ➢ Console 창에 1행 실행 결과 출력
  - ➢ 마우스 커서 다음 줄로 이동



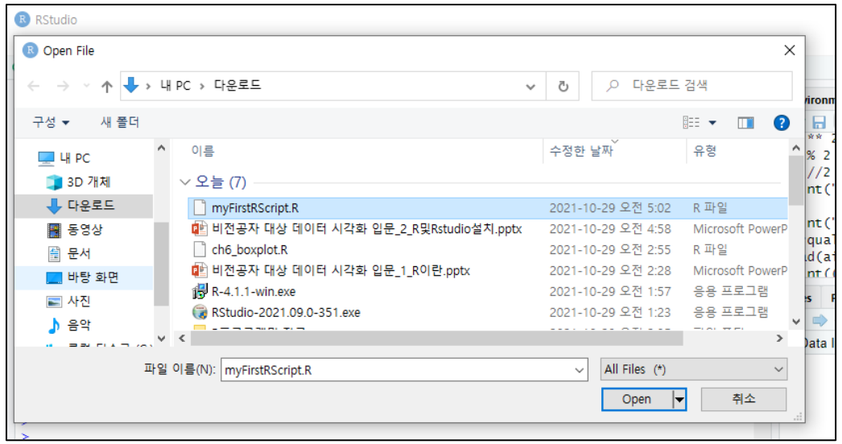
02 R 맛보기 – Script 저장/불러오기

- **Script 저장하기**
  - ➢ File탭 -> save 클릭 (단축키: Cntl+S)
  - ➢ File 저장 다이얼로그 생성되며 파일이름 수정하여 저장 가능



02 R 맛보기 – Script 저장/불러오기

- **Script 저장하기**
  - File탭 → save 클릭 (단축키: Ctrl+S)
  - File 저장 다이얼로그 생성되며 파일이름 수정하여 저장 가능
- **Script 불러오기**
  - File 탭 → open File… 클릭 (단축키: Ctrl+O)
  - ".R" 확장자를 가지는 파일 선택하여 불러오기 가능



이번 차시 정리

Rstudio 구성요소

- Script 창
- Console 창
- Environment/History 창
- Files/Plots/Packages 창

이번 차시 정리

R 맛보기

- 사칙연산
- 문자열 출력
  - Print
- 데이터셋 로딩
- Script 실행/저장/불러오기

6차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

7차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- Working directory를 설정할 수 있다.
- 프로젝트를 생성할 수 있다.

학습 목차

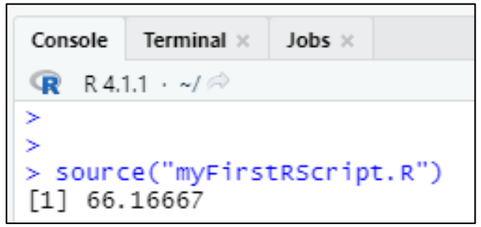
- Working directory 설정
- 프로젝트 생성

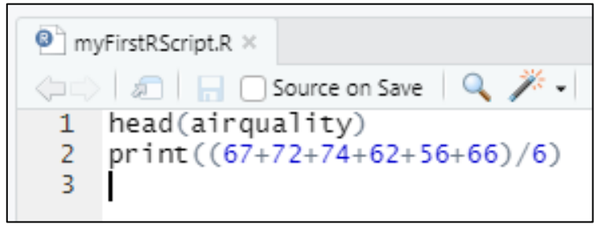
01.

Working directory 설정

01 Working directory – 내장 함수 source

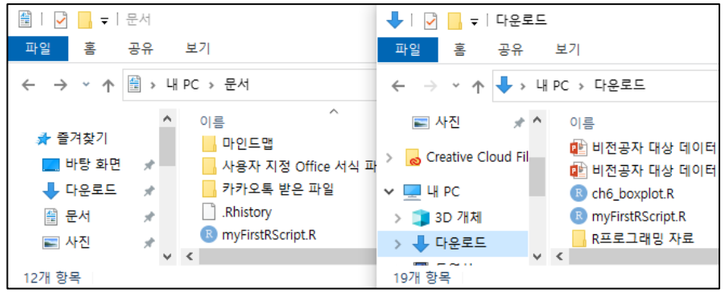
- **source 함수: 만들어진 스크립트(.R) 파일의 내용을 import**
  - ➢ 스크립트 모든 행을 run하는 것과 동일





- **절대 경로 사용**
  - ➢ source("C:₩₩Users₩₩guest₩₩Download₩₩myFirstRScript.R")

- **상대 경로 사용**
  - ➢ source("myFirstRScript.R")
  - ➢ 서로 다른 폴더에 같은 이름을 가진 파일 있다면?
    - ▪ Working director를 기준으로 함



01 Working directory – Working directory 란?

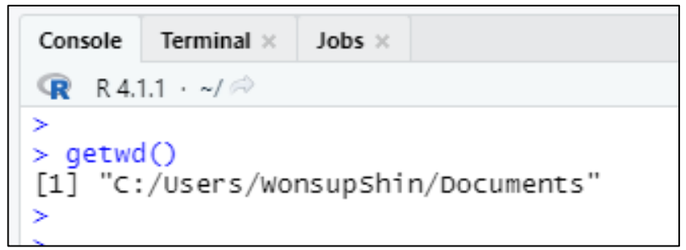
- **작업을 위한 폴더**

- **데이터셋, 파일 등 불러오기/저장 작업 수행하는 기본 폴더**

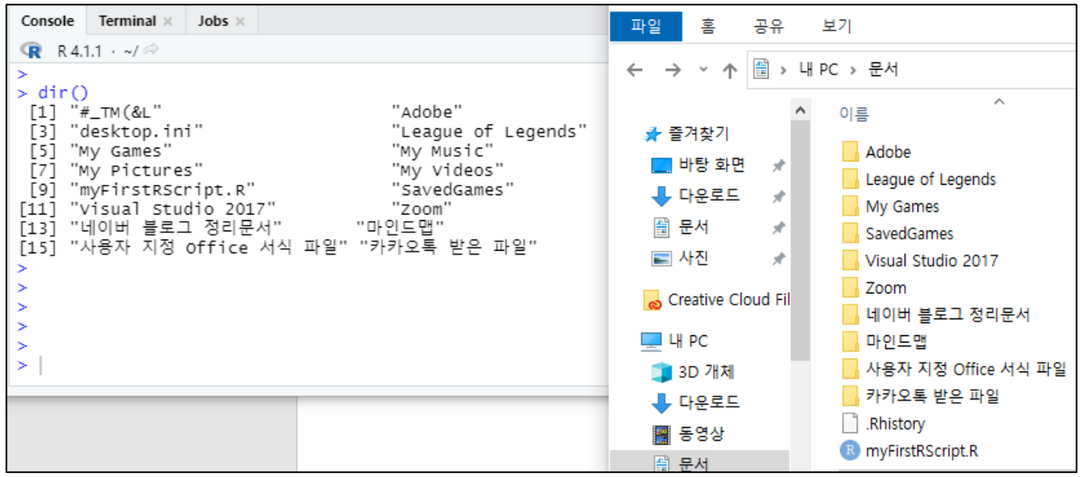
- **프로그래밍에 필요한 파일 관리 용이**
  - ➢ 새 폴더를 만들어 Working directory로 지정 후, R 관련 모든 파일을 한 폴더에 관리

01 Working directory – Working directory 확인

- **현재 Working directory 경로 확인 방법**
  - getwd( )

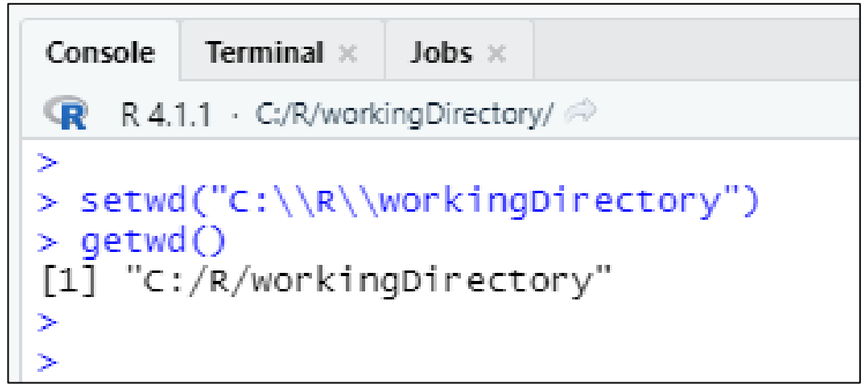


- **Working directory 내부 파일 확인 방법**
  - dir( )



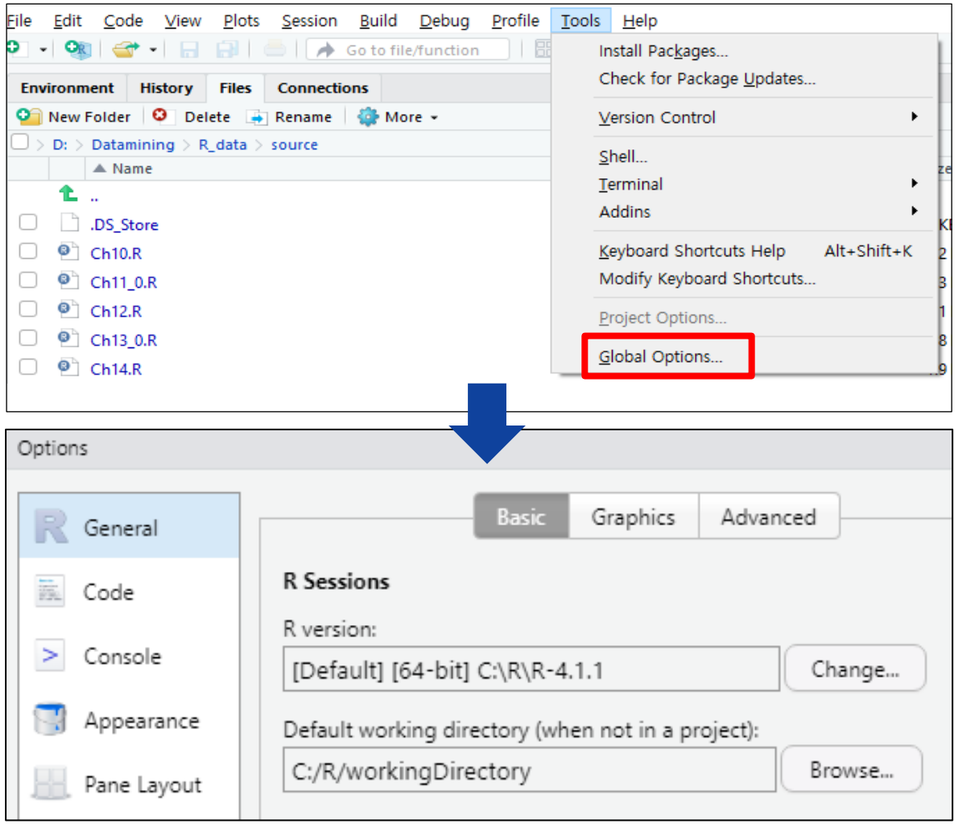
01 Working directory – Working directory 수정

- **커맨드를 사용한 수정**
  - setwd( )
  - 순서
    1. 임의의 새 폴더 생성 (기존 폴더도 가능)
    2. setwd 커맨드 사용하여 working directory 지정
  - RStudio 재시작시 유지 안됨



01 Working directory – Working directory 수정

- **기본 Working directory 수정**
  - ➢ 순서
    1. Tools → Global Options..
    2. Default working directory 수정
  - ➢ RStudio 재시작 시 유지 안됨



01 Working directory – Working directory 실습

1. C:₩R 경로에 “workingDirectory” 폴더 생성

2. Console 창 사용하여 working directory로 지정
   - 기본 working directory 위와 동일하게 수정

3. myFirstRScript.R 파일 working directory로 이동

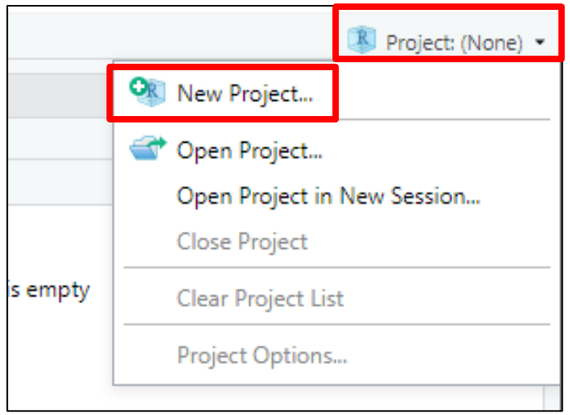
4. source 함수 사용하여 working directory에 존재하는 myFirstRScript.R 파일 import
   - 절대경로 사용
   - 상대경로 사용

02.

프로젝트 생성

02 프로젝트 생성 – 생성 과정

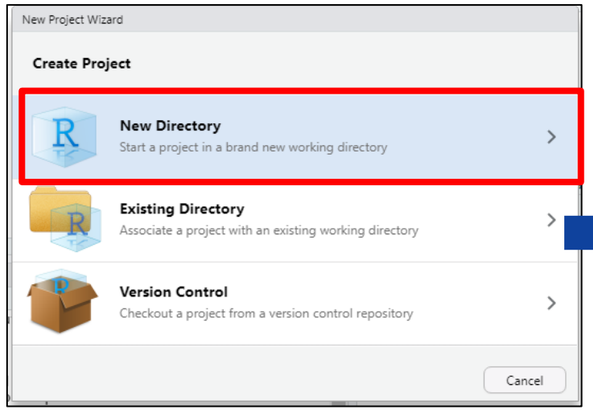
1. 우측 상단 프로젝트 버튼 클릭

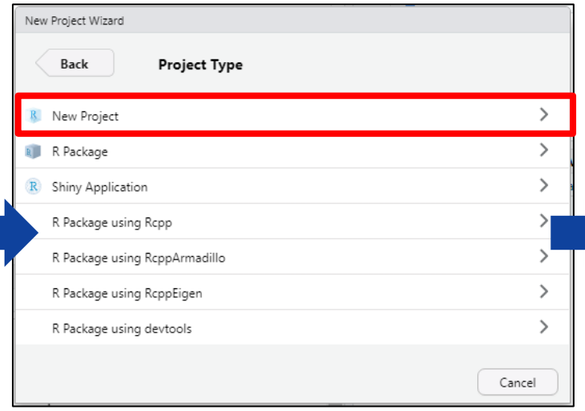


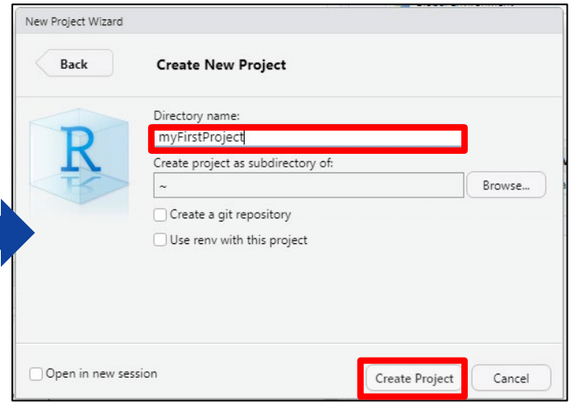
2. New Project… 클릭

3. New Directory 클릭

➢ 프로젝트명 적고 Create Project 클릭



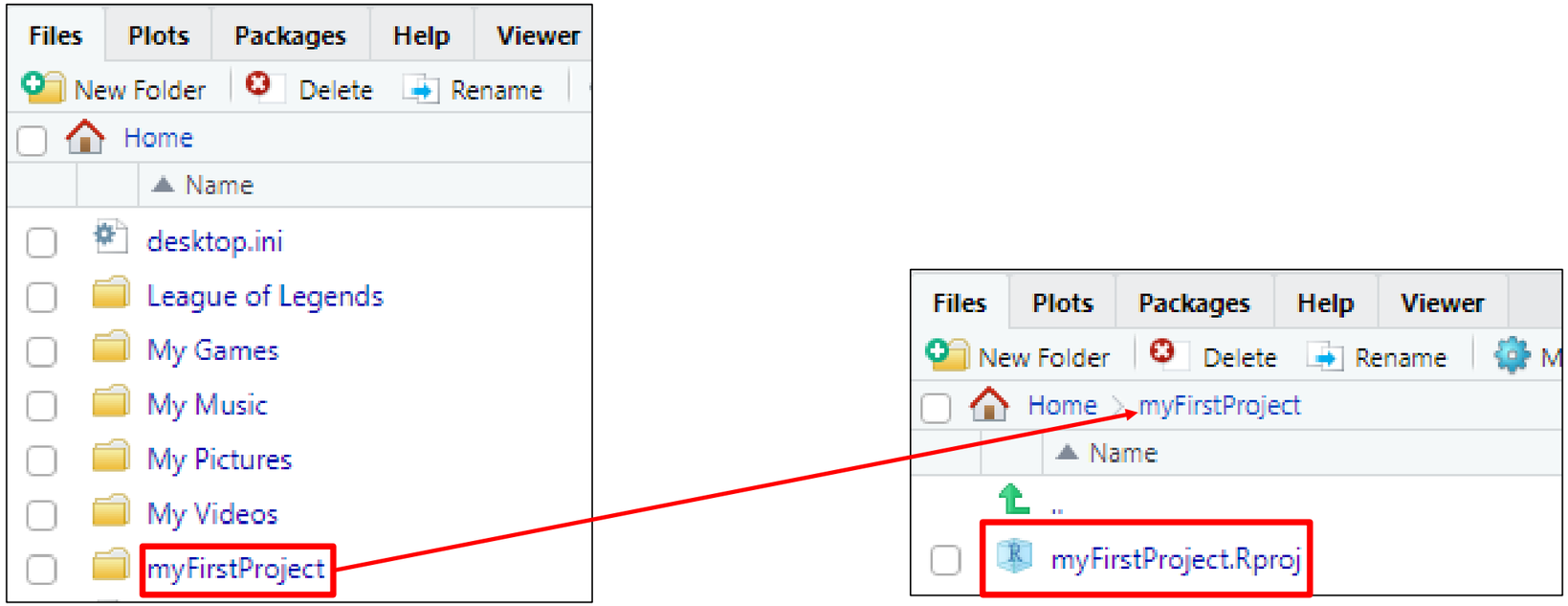




02 프로젝트 생성 – 생성 화면

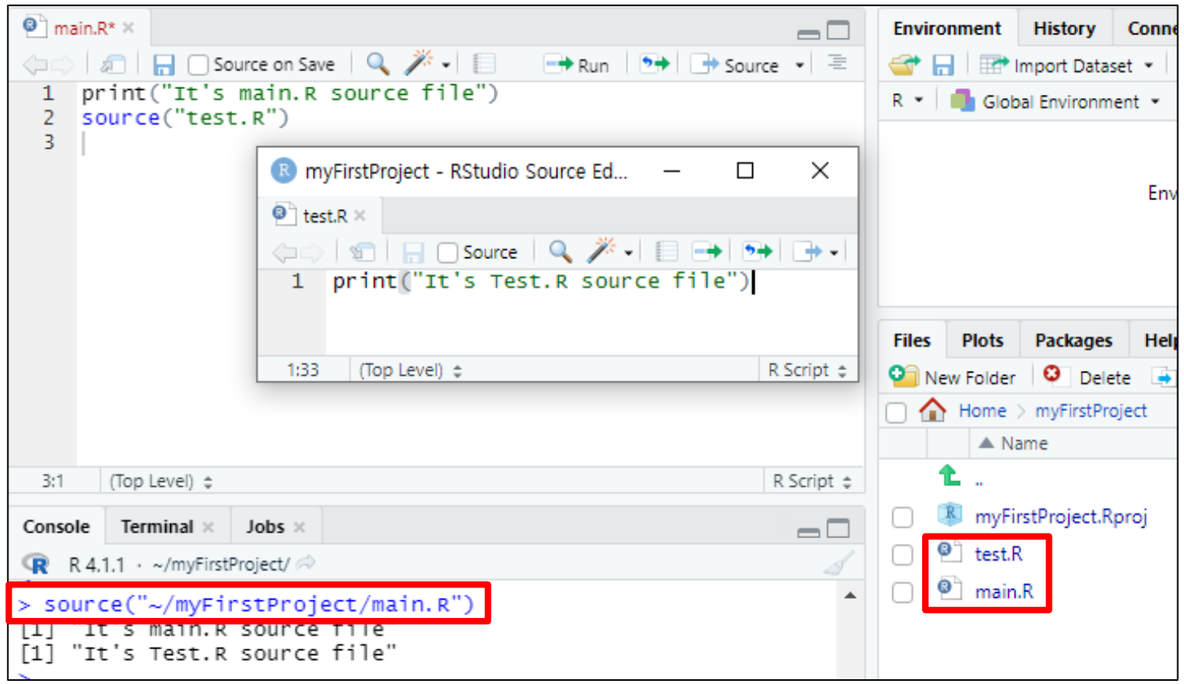
- 프로젝트명 폴더가 생김

- 폴더 내부에는 프로젝트명.Rpoj 파일 존재



02 프로젝트 생성 – 프로젝트 생성 장점

- 프로젝트 폴더가 Working directory로 자동 설정됨

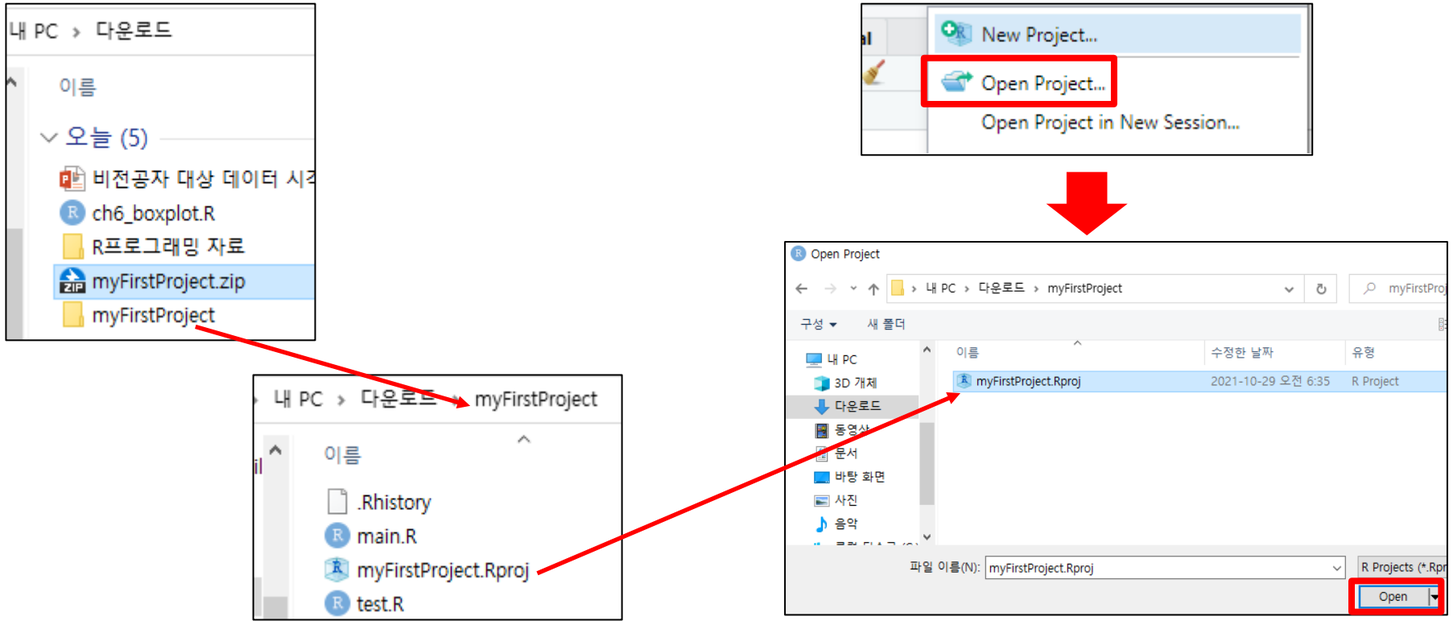


- "프로젝트 오픈" 기능을 통해서 RStudio 설치된 다른 PC에서도 동일한 환경 구성 가능

02 프로젝트 생성 - 프로젝트 생성 장점

- **RStudio 설치된 다른 PC에서도 동일한 작업 환경 쉽게 구성 가능**

1. 기존 프로젝트 압축 하여 다른 PC에 전송
2. 압축 해제
3. Rstudio 실행 → Open Project → 압축 해제 폴더 안 .Rproj 파일 선택



이번 차시 정리

Working Directory

- 데이터, 파일 불러오기/저장 기본 경로로 사용되는 디렉토리
- 관련 함수
  - Source, getwd, dir
- 수정 방법
  - setwd
  - Rstudio 기본 working directory 수정

7차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

8차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 유용한 RStudio 환경을 설정할 수 있다.
- 유용한 단축키를 알 수 있다.

학습 목차

- 유용한 RStudio 환경 설정
- 유용한 단축키

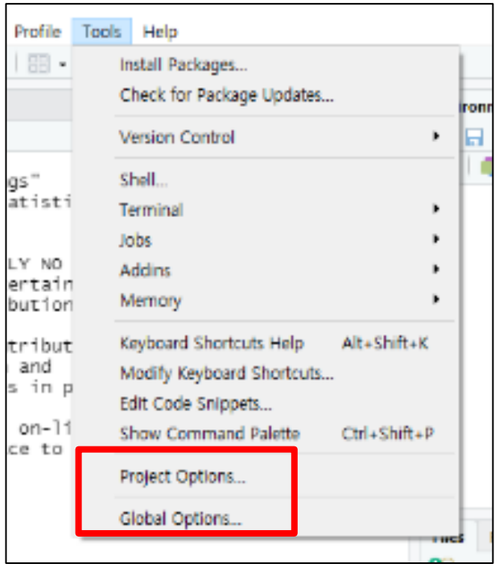
01.

유용한 환경설정

01 유용한 환경설정 – 환경 설정 범위

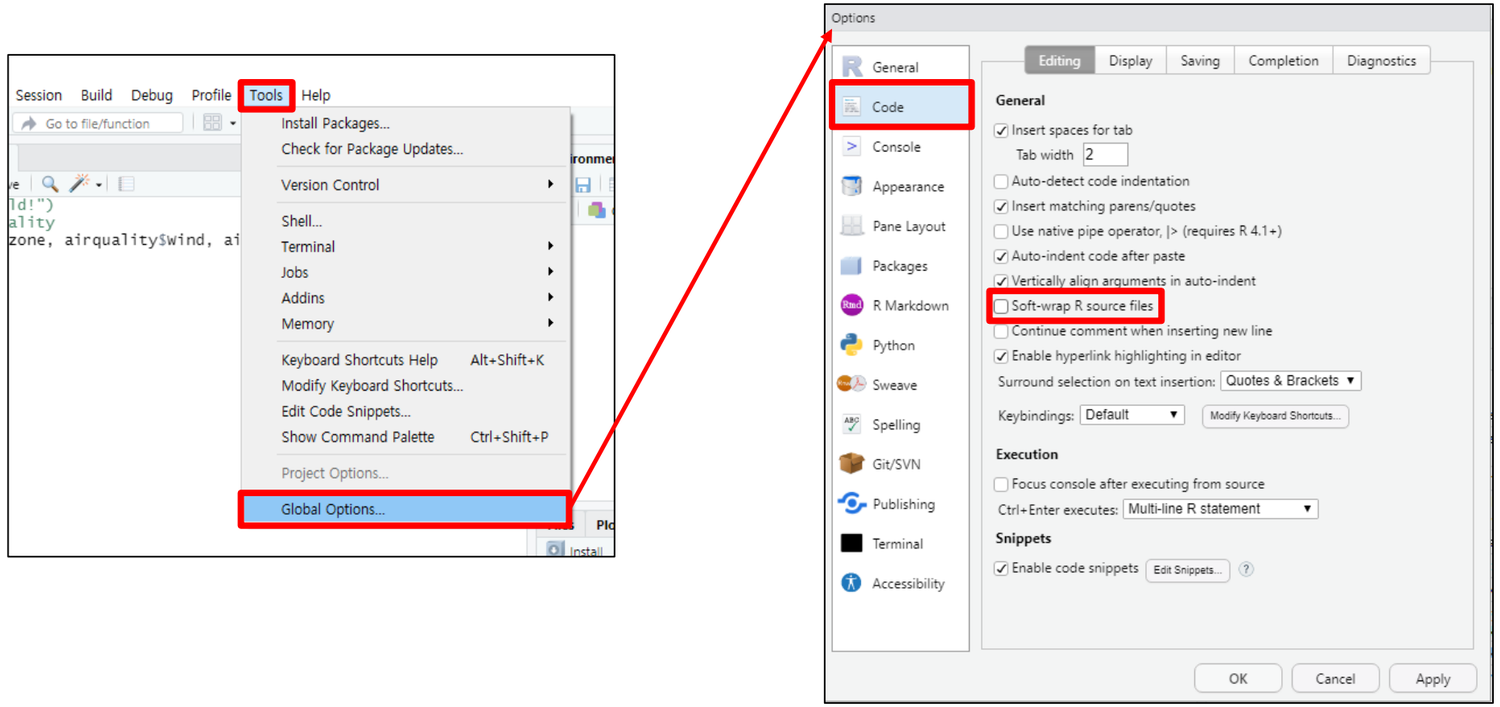
- **Global 환경 설정**
  - Global Options
  - Rstudio 환경 내에서 실행되는 모든 프로젝트에 적용

- **Project 환경 설정**
  - Project Options
  - 프로젝트 별로 다르게 환경 설정 가능
  - 프로젝트에만 설정 사항 반영
  - 프로젝트가 없는 경우 환경 설정 불가능



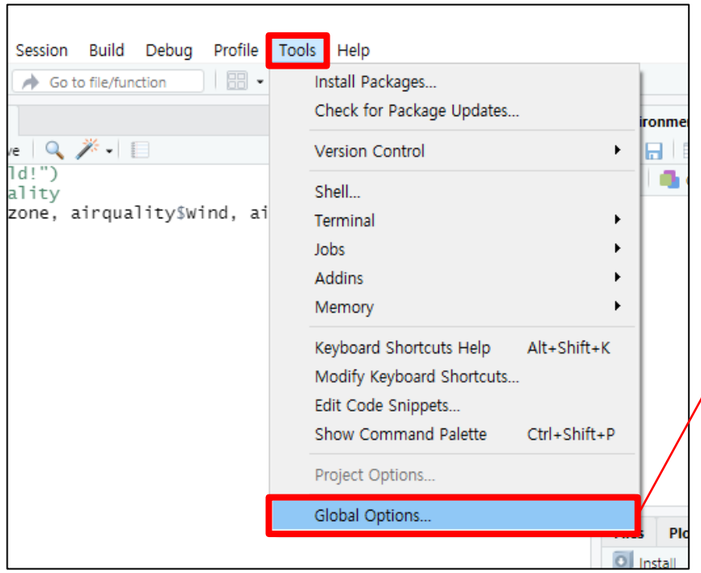
01 유용한 환경설정 – Soft Wrap 기능

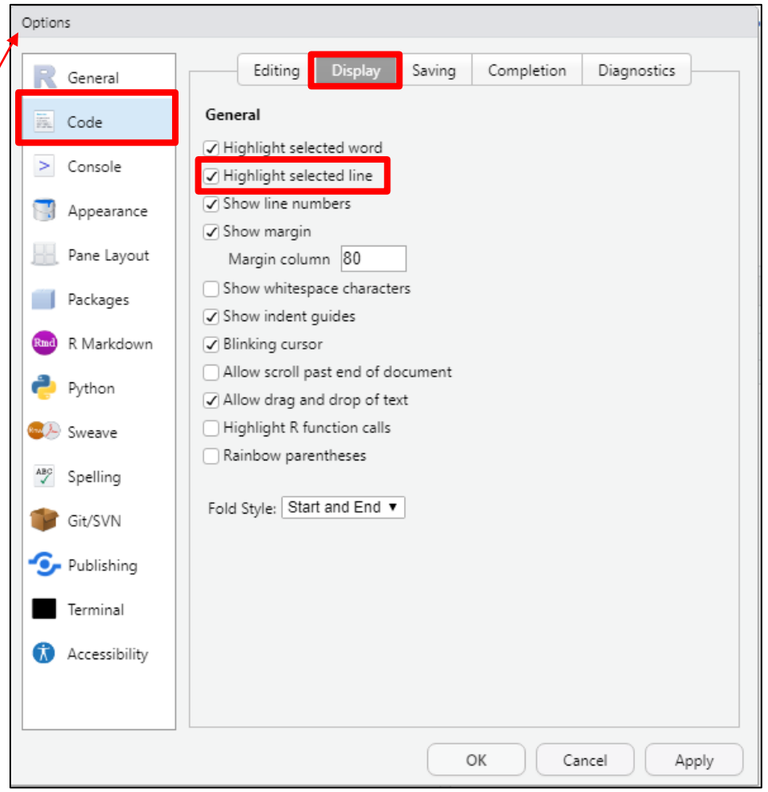
- **자동 줄바꿈 기능**
  - ➢ Source 창에서 코드가 길어질 경우 자동 줄 바꿈
  - ➢ Tools → Global Options → Code → “Soft-wrap R source“ 체크박스 클릭



01 유용한 환경설정 – Line 하이라이트 기능

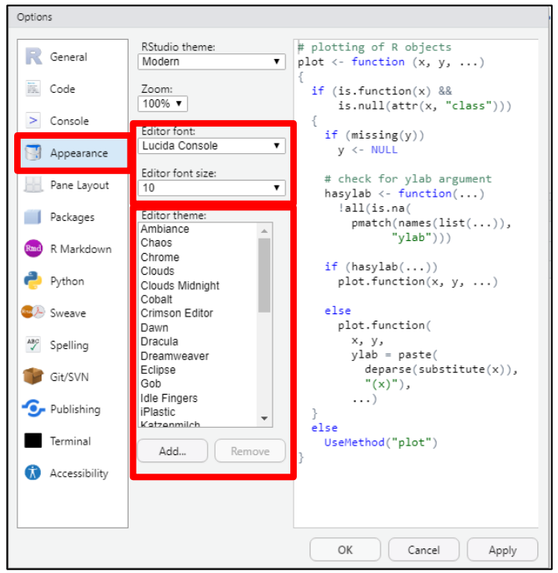
- **Source 창에서 선택된 행 하이라이트 기능**
  - ➢ Tools → Global Options → Code → Display 탭 → "Highlight selected line" 체크박스 클릭





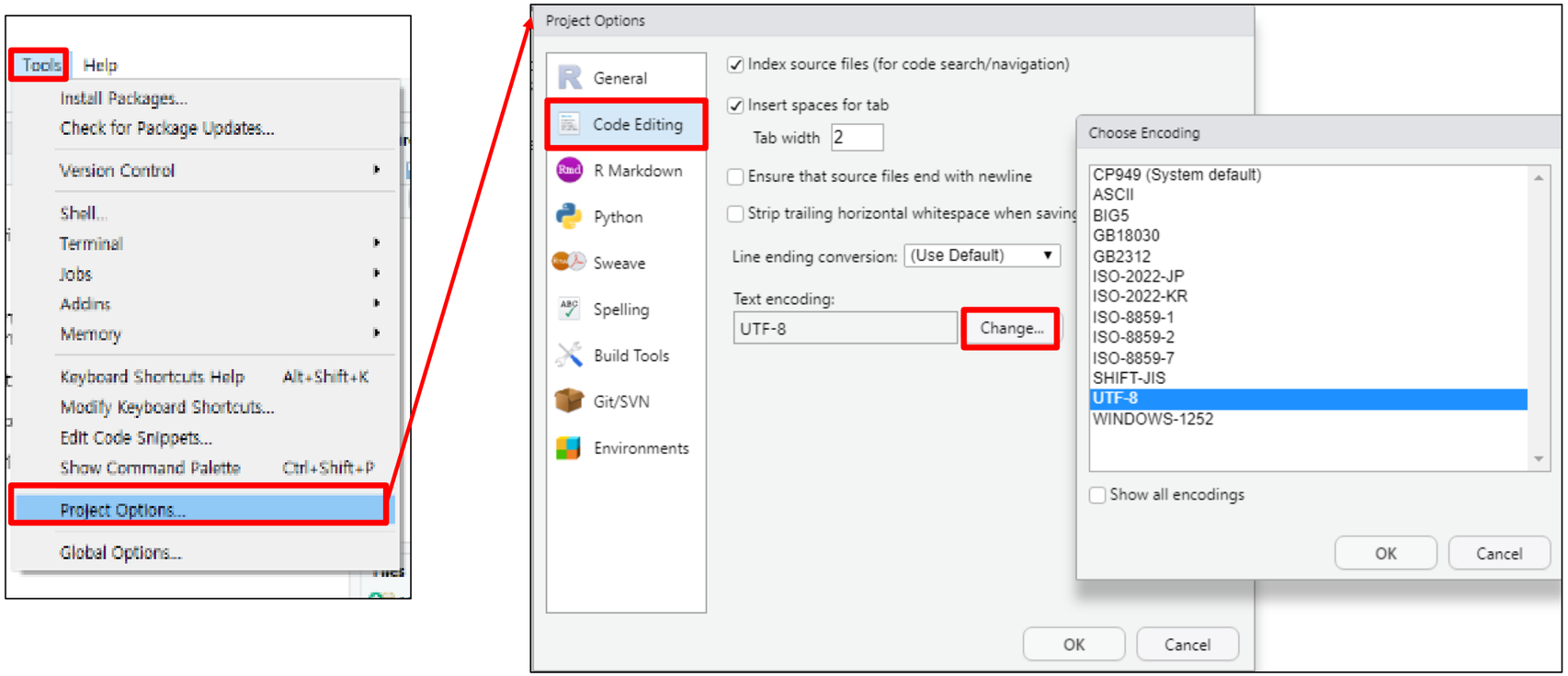
01 유용한 환경설정 – 폰트 및 배경 설정

- **폰트 및 source 창 배경 커스터마이징 가능**
  - ➢ 폰트 타입 및 크기 변경 가능
  - ➢ 다양한 배경 탬플릿 제공
    - ▪ 밝은 배경: Chrome, Dawn, Eclipse, etc..
    - ▪ 어두운 배경: Ambiance, Chaos, Dracula, Cobalt, etc..
  - ➢ Tools → Global Options → Appearance → 폰트, 배경 값 조정



01 유용한 환경설정 – 인코딩 방식 설정

- **텍스트 인코딩 방식 변경 기능**
  - ➢ 한국어 깨지는 경우 확인 해야할 설정
  - ➢ Tools → Project Options → Code Editing → Text encoding → UTF-8로 변경



02.

유용한 단축키

02 유용한 단축키

- **Ctrl + S : 파일 저장**
- **Ctrl + O : 파일 불러오기**
- **Ctrl + Shift + N: 새 소스파일**
- **Ctrl + Shift + C: 코드 주석화**
  - ➢ 범위 지정하여 여러 줄 한번에 주석화 가능

Source on Save

```
1 print("Hello World!")
2 dataset <- airquality
3 print(airquality$Ozone, airquality$Wind, airquality$Temp)
```

Ctrl + Shift + C

Source on Save

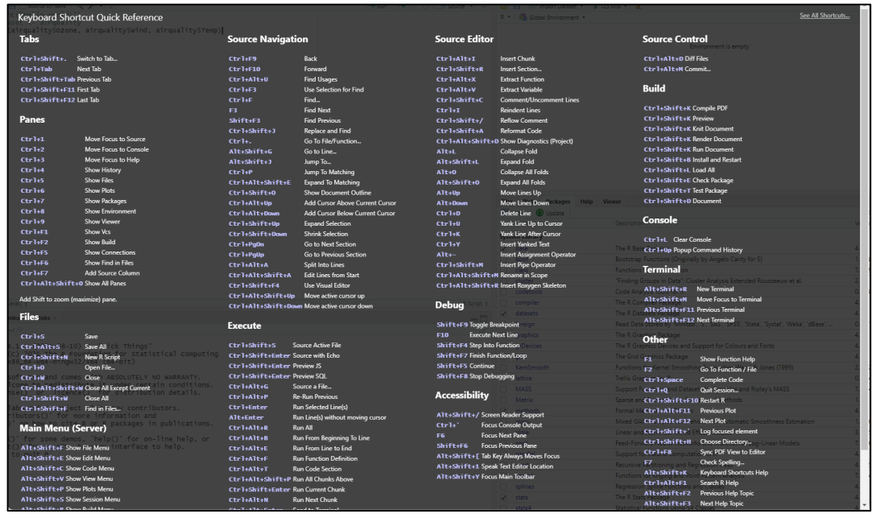
```
1 # print("Hello World!")
2 # dataset <- airquality
3 print(airquality$Ozone, airquality$Wind, airquality$Temp)
```

02 유용한 단축키

- **Ctrl + 1~9: 화면이동 단축키**
  - ➢ 숫자 별 지정된 창으로 커서 이동
  - ➢ 1: Source 창, 2: Console창, 5: Files 창, 6: Plots 창, 8: Environment 창, etc.

- **Ctrl + Shift + Enter**
  - ➢ 소스코드 실행
    - ▪ 현재 작성중인 Source 파일에 source 함수 적용

- **Shift + Alt + K**
  - ➢ 모든 단축키 확인



이번 차시 정리

환경설정 범위

- Global option
- Project option

유용한 환경설정

- Soft wrap
- Line 하이라이트
- 폰트 및 배경 설정
- 인코딩 방식 설정

유용한 단축키

- Shift + Alt + K

8차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

9차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- R의 기본 데이터 타입을 알 수 있다.

학습 목차

- R의 기본 데이터 타입

01. 데이터 타입

01 데이터 타입 – Object, Class

- Object(객체)
  - ➢ R이 다루는 가장 기초적인 구조
    - ▪ Atomic of R
    - ▪ 한국어의 “형태소”와 유사
  - ➢ 데이터 값은 객체에 담김

RStudio	실세계
객체	데이터
값: 0.5 Class: numeric	0.5

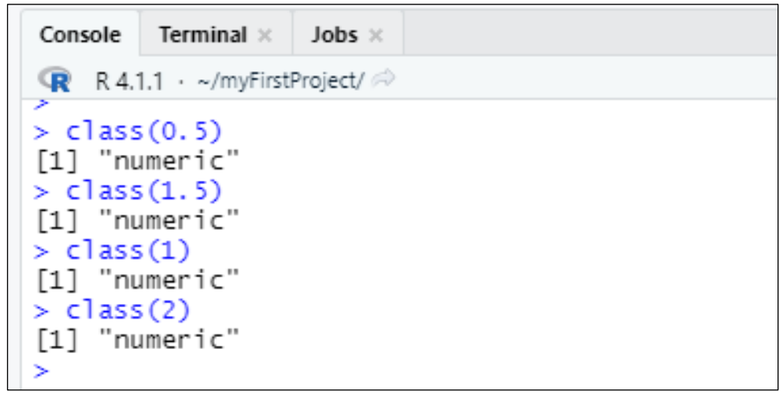
- Class
  - ➢ 객체의 타입
  - ➢ Class 확인: class() 함수 사용
    - ▪ Eg.) class(0.5)

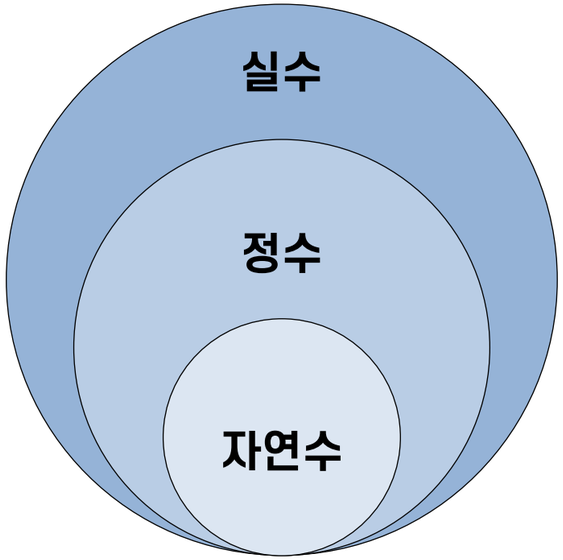
```
> class(0.5)
[1] "numeric"
```

- R은 데이터를 처리할 때 class에 따라 다르게 분류 하여 처리

01 데이터 타입 – Numeric

- 수치형 타입 (Numeric)
  - ➢ 숫자를 나타내는 데이터 타입
    - ▪ 실수 형 타입
  - ➢ Eg.) 0.5, 1.5, 1, 2





[수 의 체계]

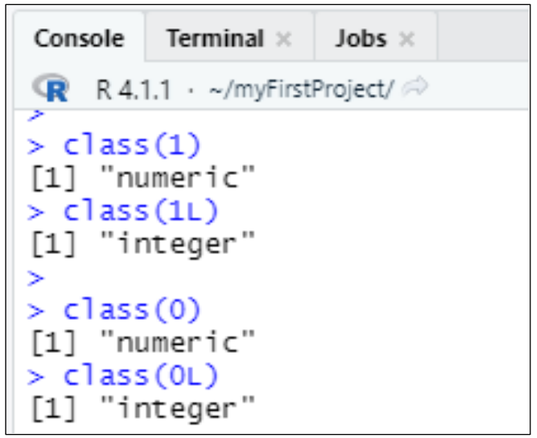
- Quiz. 0의 데이터 타입은?

01 데이터 타입 – Interger

- 정수형 타입
  - 정수 값 만을 나타내는 데이터 타입
  - 메모리 관리에 효율적
  - Eg.) 1, 0, 10, 20, -15

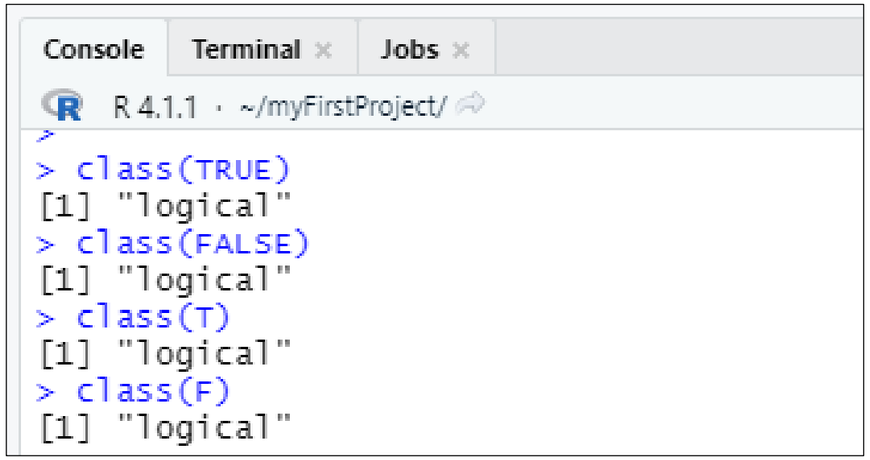
- R에서 숫자는 기본적으로 Numeric 타입으로 선언 됨
  - 객체 타입을 Integer로 지정하려면 값 뒤에 ‘L’ 기입

- [Quiz] Console 창에 ‘class(1.1L)’ 입력하였을 때 결과는?



01 데이터 타입 – Logical

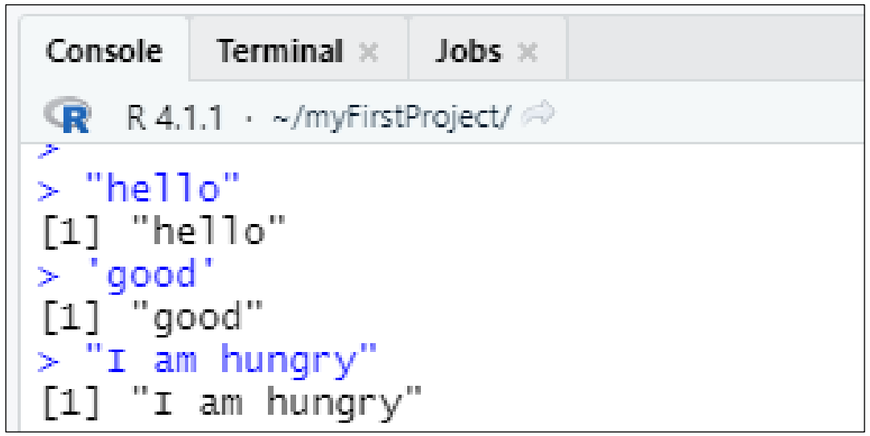
- 논리형 타입
  - ➢ 참 거짓을 나타내는 데이터 타입
  - ➢ TRUE, FALSE
  - ➢ T, F



- [Quiz] Console 창에 ‘class(True)’ 입력하였을 때 결과는?

01 데이터 타입 – Character

- 문자형 타입
  - ➢ 문자 또는 문자열을 나타내는 데이터 타입
  - ➢ 따옴표(‘’) 또는 쌍따옴표(“”)로 감싸는 형태로 표현
    - ▪ Eg.) ‘hello’, ‘good’, “I am hungry”



- [Quiz] Console 창에 ‘class(“1”)’ 입력하였을 때 결과는?

01 데이터 타입 – Complex

- 복소수형 타입 (Complex)
  - 복소수 값을 나타내는 데이터 타입
  - 실수값+허수값(i) 형태로 표현
    - Eg.) 1+1i, 5i
  - 실세계 데이터에서는 찾아보기 힘듦

Console Terminal Jobs

R 4.1.1 · ~/myFirstProject/

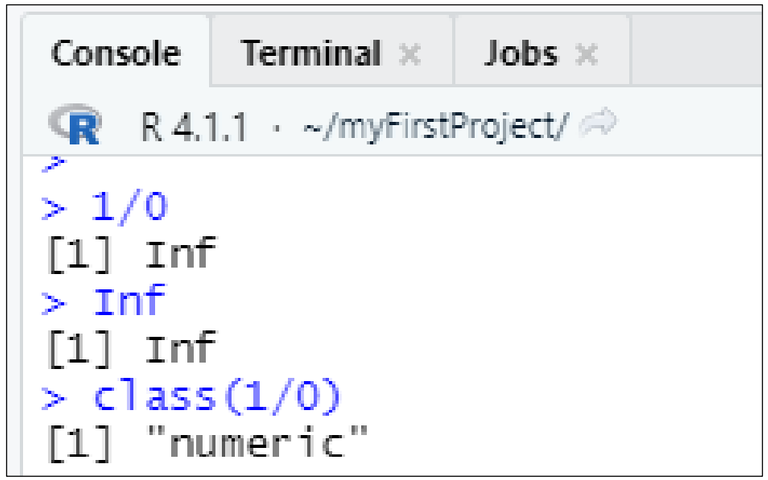
```
>
> class(1+1i)
[1] "complex"
> class(5i)
[1] "complex"
>
```

02.

특별한 데이터 객체

02 특별한 데이터 객체 – Inf

- Inf (Infinity)
  - ➢ 무한히 큰 값을 나타내는 데이터 객체
  - ➢ Eg. 1/0
  - ➢ 일반적인 계산에 사용가능



- [Quiz] '1/Inf'의 결과값은?

02 특별한 데이터 객체 – Na, NaN

- NA (Not Available)
  - ➢ 결측값(Missing value)를 나타내는 데이터 객체
    - ▪ 현실 데이터 중 에는 결측값이 존재하는 경우가 흔함
  - ➢ NA 처리를 위한 다양한 함수 존재

- NaN (Not a Number)
  - ➢ 수치값으로 표현할 수 없는 값을 나타내는 데이터 객체
  - ➢ Eg.) 0/0
  - ➢ 결측값으로 처리됨

실세계

나이
23
52

RStudio

나이
23
Na
52

이번 차시 정리

Object(객체), Class

- 객체는 R언어의 기본구성요소
- Class는 객체의 타입을 나타냄

기본 데이터 타입

- Numeric
- Integer
- Logical
- Character

이번 차시 정리

특별한 데이터 객체

- Inf
- NA
- NaN

9차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

10,11
차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 데이터의 종류를 알 수 있다
- R 자료구조를 알 수 있다.

학습 목차

- 실 세계의 데이터셋
- 데이터의 종류
- R 자료구조

01.

실 세계의 데이터셋

01 실 세계의 데이터셋 – 자료구조 왜 알아야하나?

- 데이터셋(Dataset): 데이터들의 집합, 모음

- 매우 다양한 종류의 데이터셋들이 존재
  - 서울 하루 간 시간 별 날씨
  - 학교 과목별 시험 성적
  - CCTV 영상
  - 병원 건강검진 결과 리스트
  - Etc.

- 데이터 양이 아주 많음
  - 빠른 정보 추출을 위해 효율적인 데이터 탐색, 관리 중요
  - 특정한 규칙으로 컴퓨터 상에 표현 = 자료구조

02.

데이터의 종류

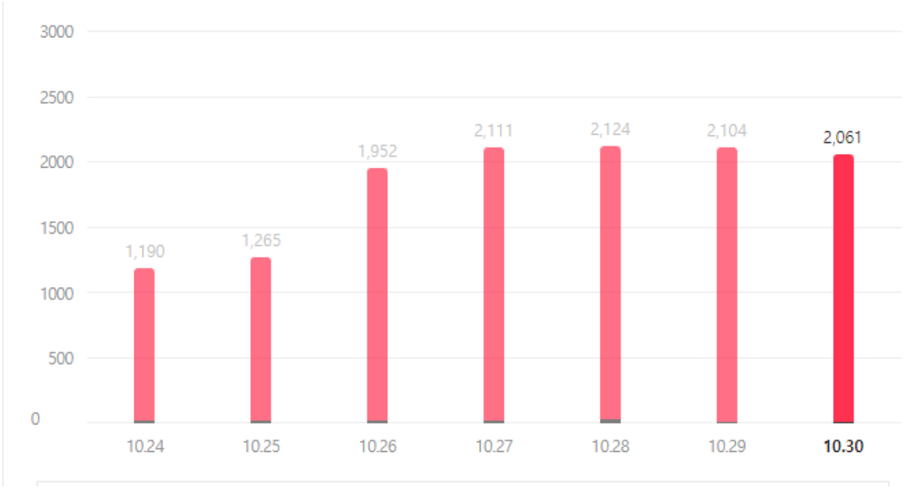
02 데이터 종류 – 요약

- 데이터 차원에 따른 구분
  - 1차원
  - 2차원
  - N차원
- 데이터 구성에 따른 구분
  - 단일형
  - 다중형
- 데이터 값 연속성에 따른 구분
  - 범주형
  - 수치형

02 데이터 종류(차원) - 1차원 구조

- 단일 주제 데이터들을 모아 놓은 구조

- 예시
  - 2021년 전세계 국가의 GDP
  - 하루간 기온
  - 특정지역 강우량
  - 연도별 비행기 탑승객 수

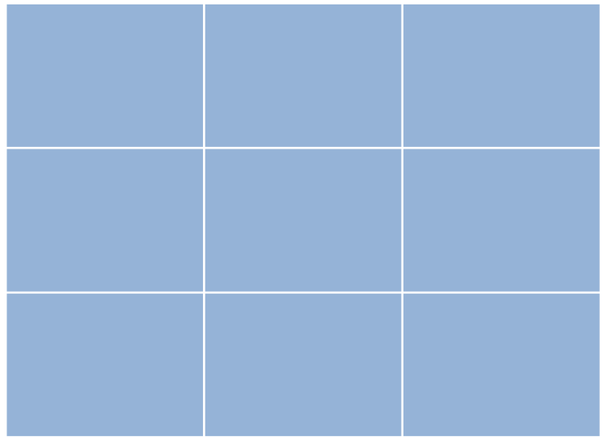


[코로나 일일확진자 데이터]

02 데이터 종류(차원) - 2차원 구조

- 복수 주제 데이터들을 모아 놓은 구조

- 예시
  - 연도별 전세계 국가의 GDP
  - 언어/수학/영어 모의고사점수
  - 병원 건강검진 결과

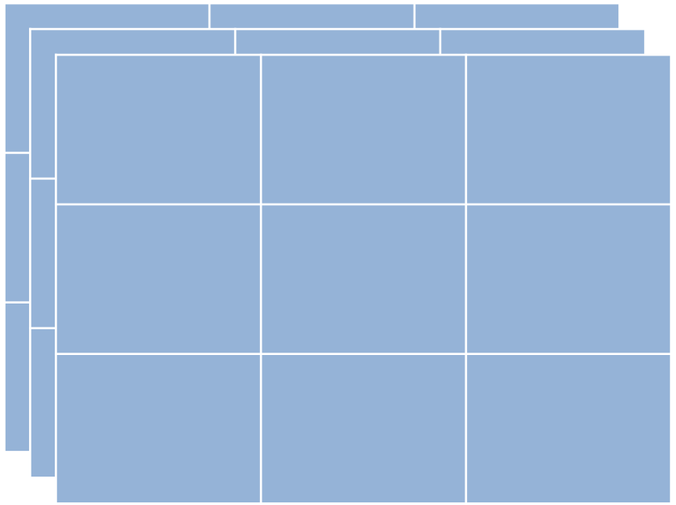


- 가장 활발히 사용되는 데이터 종류
  - 시각화에 가장 적합
  - 사람의 눈은 2차원을 인식하기 때문

02 데이터 종류(차원) - N차원 구조

- N-1차원 데이터들을 모아 놓은 구조

- 예시
  - ➢ 월별 언어/수학/영어 모의고사점수 (3차원)
    - ▪ + 스트레스지수 (4차원)
  - ➢ 연도별 환자 건강검진 결과 (3차원)



02 데이터 종류(구성) – 단일형, 다중형

단일형

- 한가지 타입의 데이터로만 구성된 데이터
- Eg.) 학생 별 나이, 몸무게, 키 데이터

[단일형 데이터]

02 데이터 종류(구성) - 단일형, 다중형

다중형

- 여러가지 타입의 데이터로 구성된 데이터
- Eg.) 학생 별 나이, 이름, 안경 착용 여부 데이터

[다중형 데이터]

02 데이터 종류(값) – 범주형, 수치형

범주형

- 데이터 값이 이산적
- 보통 논리값 또는 문자로 표현
- 산술 연산 불가능

성별	남, 여, 남, 남, 여, 여, 여, …
색상	노랑, 검정, 파랑, 검정, 검정, 노랑, …
사이즈	M, L, S, M, L, XL, L, L, …

[범주형 데이터셋 예시 (의류 발주 기록)]

02 데이터 종류(값) – 범주형, 수치형

수치형

- 데이터 값이 연속적
- 수치로 표현
- 산술 연산 가능

- **수치형 값은 범주형으로 표현 가능**
  - **Eg.) 온도**
    - **(0~10): 추움**
    - **(10~25): 보통**
    - **(25~35): 더움**

온도	**22.5, 21.4, 21.7, 22.3, 22.1, 22.0, …**
습도	**30, 15, 16, 23.4, 70, 88, 80, 76, …**

[수치형 데이터셋 예시 (온도 습도 기록)]

03.

R 자료구조

03 R 자료구조 – 개요

- 차원 및 데이터 구성에 따라 구조 구분

구분	1차원	2차원	N차원
단일형	벡터	행렬	배열
다중형	리스트	데이터 프레임	-

03 R 자료구조 – 벡터(Vector)

- 가장 기본적인 자료 구조
- 데이터 타입이 동일한 값들의 모음
- 생성 방법
  - c() 함수 사용
    - Concatenate의 약어

Console Terminal Jobs

R 4.1.1 · ~/test/

```
>
> c(0.5, 0.6, 1, -5)  #numeric
[1]  0.5  0.6  1.0 -5.0
> c(TRUE, FALSE, T ,F) #logical
[1]  TRUE FALSE  TRUE FALSE
> c("a", "b", "c") #character
[1] "a" "b" "c"
> c(1:10) #integer
 [1]  1  2  3  4  5  6  7  8  9 10
```

03 R 자료구조 – 벡터(Vector)

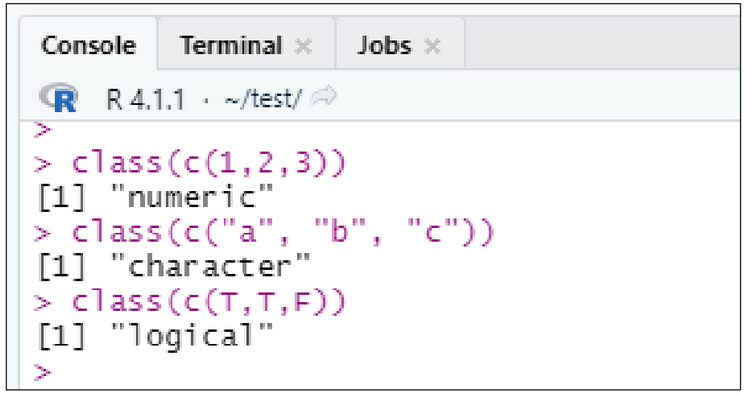
- 가장 기본적인 자료구조
  - 1차원 구조
  - 데이터 타입이 동일한 값들의 모음

- 생성 방법
  - c() 함수 사용
  - vector() 함수 사용
    - 사용방법: vector("데이터 타입", length = 벡터 크기)
    - 실제로 값을 넣지 않았기 때문에 기본값으로 초기화

```
Console   Terminal ×   Jobs ×
R 4.1.1 · ~/test/
>
> vector("character", length = 5)
[1] "" "" "" "" ""
> vector("numeric", length = 5)
[1] 0 0 0 0 0
> vector("logical", length = 5)
[1] FALSE FALSE FALSE FALSE FALSE
```

03 R 자료구조 – 벡터(Vector)

- [Quiz] 벡터의 데이터 타입은 무엇일까?
  - ➢ class(c(1, 2, 3)) 실행 결과는?
  - ➢ class(c("a", "b", "c")) 실행 결과는?
  - ➢ class(c(T, T, F)) 실행 결과는?



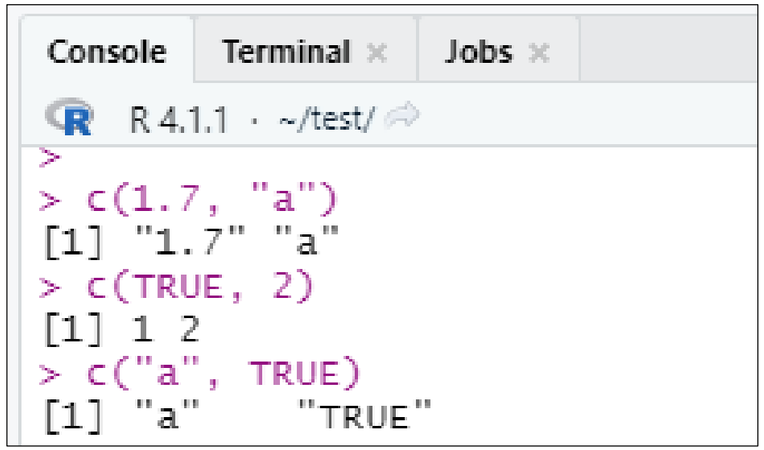
→ 10, T, "a"같은 단순 값 또한 모두 vector 자료구조로 표현 됨

Vector는 가장 기본적인 데이터 자료구조

Eg. 10 == c(10) # 숫자 10을 포함하는 크기가 1인 numeric vector

03 R 자료구조 – 벡터(Vector)

- [Quiz] 벡터에 서로 다른 데이터 타입을 넣는다면 어떻게 될까?
  - c(1.7, “a”)의 결과값은?
  - c(TRUE, 2)의 결과값은?
  - c(“a”, TRUE)의 결과값은?



→ 자동 형변환 일어남

03 R 자료구조 – 형변환

- 데이터 타입이 다른 타입으로 변경되는 것

- 자동 형변환
  - ➢ 컴퓨터에 의해 자동으로 형변환이 일어남
    - ▪ c(1.7, “a”) # vector는 같은 데이터타입만 존재해야 하므로 변환
    - ▪ 1L/2L # 결과값인 0.5는 정수가 아니므로 변환
    - ▪ 1L + 2L # 형변환 일어나지 않음

- 자동 형변환 우선 순위
  - ➢ Character > Numeric > Logical

03 R 자료구조 – 행렬(Matrix)

- Vector의 집합
  - 단일형, 2차원 구조
  - 행(row), 열(col)로 구성

- 생성 방법
  - matrix() 함수 사용
    - 사용 방법: matrix(nrow=행 크기 , ncol= 열 크기)

```
> matrix(nrow=2, ncol=3) #2x3 크기 행렬 생성
     [,1] [,2] [,3]
[1,]   NA   NA   NA
[2,]   NA   NA   NA
```

03 R 자료구조 – 행렬(Matrix)

- Vector의 집합
  - 단일형, 2차원 구조
  - 행(row), 열(col)로 구성

- 생성 방법
  - matrix() 함수 사용
  - Vector값 사용하여 matrix 생성 가능
    - 사용 방법: matrix(vector, nrow=행 크기 , ncol= 열 크기)

```
> matrix( c(1,2,3,4,5,6) ,nrow=2, ncol=3)
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
> matrix( c(1,2,3,4,5,6) ,nrow=3, ncol=2) #3x2 크기 행렬 생성
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
```

03 R 자료구조 – 행렬(Matrix)

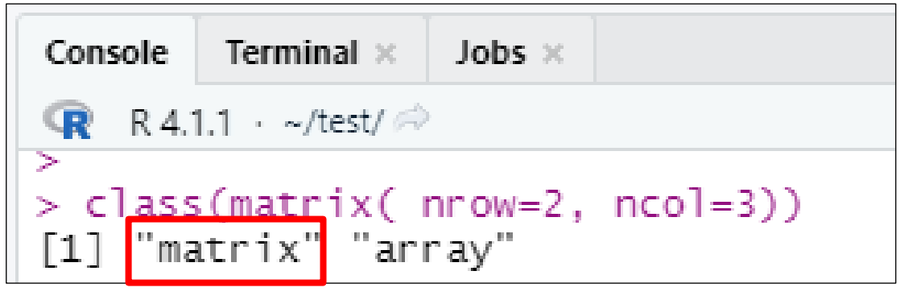
- Vector의 집합
  - 단일형, 2차원 구조
  - 행(row), 열(col)로 구성

- 생성 방법
  - matrix() 함수 사용
  - Vector값 사용하여 matrix 생성 가능
  - byrow 옵션
    - 데이터를 “행“부터 채움
    - 기본값: FALSE

```
> matrix( c(1,2,3,4,5,6) ,nrow=3, ncol=2, byrow=T)
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
```

03 R 자료구조 – 행렬(Matrix)

- [Quiz] 행렬의 데이터 타입은 무엇일까?
  - ➢ class(matrix(nrow=2, ncol=3)) 실행 결과는?



Matrix 객체

Value:
c(1,2,3,4,5,6)
Class: matrix
dim: c(2,3)

→ 행/열 크기를 담는 dim 속성을 추가로 가짐

- ▪ Matrix에서 dim은 크기가 2인 vector

03 R 자료구조 – 배열(Array)

- dim 속성 크기가 2이상인 자료구조
- 생성 방법
  - ➢ array() 함수 사용
    - array(dim=(2, 2, 3))
    - Vector값 사용하여 array 생성 가능
  - ➢ Matrix == dim 크기가 2인 array

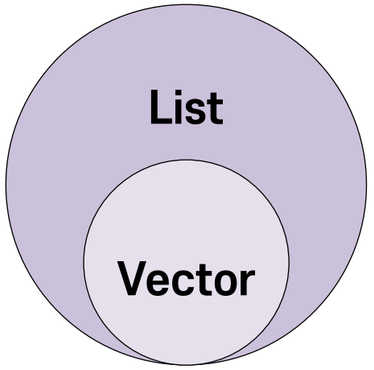
```
> array(dim=c(2,2,3))
, , 1

     [,1] [,2]
[1,]   NA   NA
[2,]   NA   NA

, , 2

     [,1] [,2]
[1,]   NA   NA
[2,]   NA   NA

, , 3

     [,1] [,2]
[1,]   NA   NA
[2,]   NA   NA
```

```
> array(c(1,2,3,4,5,6,7,8,9,10,11,12),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
```

03 R 자료구조 – 리스트(List)

- 서로 다른 데이터 타입이 존재하는 값들의 모음
- 현실 세계에 아주 흔하게 존재하는 데이터 구조
  - ➢ Eg. 신상명세 데이터
    - ▪ 이름, 나이, 키
- 생성 방법
  - ➢ list() 함수사용

```
Console   Terminal ×   Jobs ×
R 4.1.1 · ~/test/
> list(1,T, "hi", 2L)
[[1]]
[1] 1

[[2]]
[1] TRUE

[[3]]
[1] "hi"

[[4]]
[1] 2
```



[Vector, List 상관관계]

03 R 자료구조 – 리스트(List)

- Quiz. List에 다른 자료구조(vector 등)을 넣을 수 있을까?
  - ➢ list(c(1,2,3), 2, T) 결과값은?
  - ➢ list(list(1,T,”a”), 1) 결과값은?
  - ➢ list(matrix(nrow=2, ncol=3), “a”, 5) 결과값은?

Console Terminal Jobs

R 4.1.1 · ~/test/

```
> list(list(1, T,"a"), c(1,2,3), matrix(nrow=2, ncol=3))
[[1]]
[[1]][[1]]
[1] 1

[[1]][[2]]
[1] TRUE

[[1]][[3]]
[1] "a"


[[2]]
[1] 1 2 3

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

→ Vector, matrix, list등의 자료구조 포함 가능

03 R 자료구조 – 데이터 프레임

- List의 집합
  - 다중형, 2차원 구조
  - 행(row), 열(col)로 구성
    - 각 행은 list 취급
    - 각 열은 vector 취급
  - 열 별로 다른 데이터 타입 저장 가능

03 R 자료구조 – 데이터 프레임

- 데이터셋 파일 로딩 시, 주로 데이터 프레임 구조사용
  - Table, excel, csv 형식 데이터

- 생성 방법
  - data.frame() 함수 사용
    - 사용 방법: data.frame(vector, vector, …)

03 R 자료구조 – 데이터 프레임

- 생성 방법
  - ➢ data.frame() 함수 사용
    - ▪ 사용 방법: data.frame(vector, vector, …)
    - ▪ 각 vector는 column값으로 삽입 됨

```
> data.frame(c(1,2,3,4,5), c("A", "B", "C", "A", "B"), c(T, T,F, T, T))
  c.1..2..3..4..5. c..A....B....C....A....B.. c.T..T..F..T..T.
1                1                         A            TRUE
2                2                         B            TRUE
3                3                         C           FALSE
4                4                         A            TRUE
5                5                         B            TRUE
```

03 R 자료구조 – 데이터 프레임

- 실 데이터셋 예시: 학점 및 합격 여부 데이터셋
  - ➢ 변수: 객체를 담을 수 있는 틀

```
> ID <- c(1,2,3,4,5)
> Grade <- c("A", "B", "C", "A", "B")
> Pass <- c(T, T,F, T, T)
> data.frame(ID, Grade, Pass)
  ID Grade  Pass
1  1     A  TRUE
2  2     B  TRUE
3  3     C FALSE
4  4     A  TRUE
5  5     B  TRUE
```

이번 차시 정리

구분 기준에 따른 데이터 종류 이해

- 데이터 차원에 따른 구분
- 데이터 구성에 따른 구분
- 데이터 값에 따른 구분

R 데이터 자료구조 소개 및 선언 방법

- 단일형
- 다중형

10,11
차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

12차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 데이터 타입을 알 수 있다.
- 데이터 종류를 알 수 있다.
- 데이터 자료구조를 알 수 있다.

학습 목차

- Quiz
- 실습

01. Quiz

01 Quiz – 데이터 타입

- Q1. R 언어가 다루는 가장 기초적인 구조는 ( )다

  ① Class ② Vector ③ Numeric ④ Object

- Q2. 데이터 타입을 확인하기 위해서는 ( ) 함수를 사용한다

  class() ② array() ③ matrix() ④ print()

01 Quiz – 데이터 타입

- Q1. R 언어가 다루는 가장 기초적인 구조는 (Object)다

  ① Class ② Vector ③ Numeric ④ Object

- Q2. 데이터 타입을 확인하기 위해서는 (class()) 함수를 사용한다

  ① class() ② array() ③ matrix() ④ print()

01 Quiz – 데이터 타입

- Q3. 기본적인 데이터 타입으로는 수치형, 정수형, 논리형 (　　),
  복소수형 등이 있다.

  ① 단일형　② 문자형　③ 자료형　④ 범주형

- Q4. 결측치를 나타내는 데이터 타입은 (　　)다.

  ① NA　② Na　③ NaN　④ Inf

- Q5. 수치적으로 표현할 수 없는 값을 나타내는 데이터 타입은 (　　)다.

  ① NA　② Na　③ NaN　④ Inf

01 Quiz – 데이터 타입

- Q3. 기본적인 데이터 타입으로는 수치형, 정수형, 논리형 (문자형),
  복소수형 등이 있다.

  ① 단일형 ② 문자형 ③ 자료형 ④ 범주형

- Q4. 결측치를 나타내는 데이터 타입은 (NA)다.

  ① NA ② Na ③ NaN ④ Inf

- Q5. 수치적으로 표현할 수 없는 값을 나타내는 데이터 타입은 (NaN)다.

  ① NA ② Na ③ NaN ④ Inf

01 Quiz – 데이터 종류

- Q6. 데이터들의 집합 또는 모음은 (　　)이라 부른다.

  ① 데이터셋　② 데이터　③ 자료　④ 테이블

- Q7. (　　) 데이터는 한가지 데이터 타입으로만 구성된다.

  ① 중첩형　② 다중형　③ 단일형　④ 사각형

- Q8. 범주형 데이터는 수치형 데이터로 변환 가능하다.

  ① 맞다　② 아니다

01 Quiz – 데이터 종류

- Q6. 데이터들의 집합 또는 모음은 (데이터셋)이라 부른다.

  ① 데이터셋 ② 데이터 ③ 자료 ④ 테이블

- Q7. (단일형) 데이터는 한가지 데이터 타입으로만 구성된다.

  ① 중첩형 ② 다중형 ③ 단일형 ④ 사각형

- Q8. 범주형 데이터는 수치형 데이터로 변환 가능하다.

  ① 맞다 ② 아니다

01 Quiz – 데이터 자료구조

- Q9. 가장 기본적인 데이터 자료구조는 (　　)이다.

  ① Data frame　② Matrix　③ Vector　④ List　⑤ Array

- Q10. 다중형 데이터 자료구조이며, 현실 세계에 흔하게 존재하는 데이터 구조는 (　　)이다.

  ① Data frame　② Matrix　③ Vector　④ List　⑤ Array

- Q11. 단일형, 2차원 데이터 자료구조는 (　　)이다.

  ① Data frame　② Matrix　③ Vector　④ List　⑤ Array

01 Quiz – 데이터 자료구조

- Q9. 가장 기본적인 데이터 자료구조는 (Vector)이다.

  ① Data frame ② Matrix ③ Vector ④ List ⑤ Array

- Q10. 다중형 데이터 자료구조이며, 현실 세계에 흔하게 존재하는 데이터 구조는 (List)이다.

  ① Data frame ② Matrix ③ Vector ④ List ⑤ Array

- Q11. 단일형, 2차원 데이터 자료구조는 (Matrix, Array)이다.

  ① Data frame ② Matrix ③ Vector ④ List ⑤ Array

02.
실습

02 실습 - 은행 대출상품 마케팅

- 데이터 설명: R 은행이 대출상품 마케팅에 사용하기 위하여 다양한 상담매체 사용하여 고객들에게 대출 권유한 결과를 데이터셋화 하였다. 그 내용은 아래와 같다.

나이 (age)	직종 (job)	결혼 유무 (marital)	은행 잔고 (balance)	상담 매체 (campaign)	대출 신청 하였는가? (y)
30	무직	결혼	1787	휴대폰	FALSE
33	서비스	결혼	4789	휴대폰	FALSE
35	관리직	미혼	1350	휴대폰	FALSE
30	관리직	결혼	1476	Unknown	FALSE
68	은퇴	사별	4189	유선전화	TRUE
33	관리직	결혼	3935	휴대폰	TRUE

참조: https://archive.ics.uci.edu/ml/datasets/bank+marketing

02 실습 - 데이터 분석

나이 (age)	직종 (job)	결혼 유무 (marital)	은행 잔고 (balance)	상담 매체 (campaign)	대출 신청 하였는가? (y)
30	무직	결혼	1787	휴대폰	FALSE
33	서비스	결혼	4789	휴대폰	FALSE
35	관리직	미혼	1350	휴대폰	FALSE
30	관리직	결혼	1476	Unknown	FALSE
68	은퇴	사별	4189	유선전화	TRUE
33	관리직	결혼	3935	휴대폰	TRUE

- 각 컬럼 별 데이터 타입을 설명하세요.

Age : Numeric (Integer)

Job : Character

Marital : Character

Balance : Numeric

Campaign : Character

Y : Logical

02 실습 - 데이터 분석

나이 (age)	직종 (job)	결혼 유무 (marital)	은행 잔고 (balance)	상담 매체 (campaign)	대출 신청 하였는가? (y)
30	무직	결혼	1787	휴대폰	FALSE
33	서비스	결혼	4789	휴대폰	FALSE
35	관리직	미혼	1350	휴대폰	FALSE
30	관리직	결혼	1476	Unknown	FALSE
68	은퇴	사별	4189	유선전화	TRUE
33	관리직	결혼	3935	휴대폰	TRUE

- 각 컬럼 별 데이터가 범주형 인지 수치형인지 설명하세요.

Age : 수치형/범주형

Job : 범주형

Marital : 범주형

Balance : 수치형

Campaign : 범주형

Y : 범주형

02 실습 - 자료구조 실습

나이 (age)	직종 (job)	결혼 유무 (marital)	은행 잔고 (balance)	상담 매체 (campaign)	대출 신청 하였는가? (y)
30	무직	결혼	1787	휴대폰	FALSE
33	서비스	결혼	4789	휴대폰	FALSE
35	관리직	미혼	1350	휴대폰	FALSE
30	관리직	결혼	1476	Unknown	FALSE
68	은퇴	사별	4189	유선전화	TRUE
33	관리직	결혼	3935	휴대폰	TRUE

- **위의 데이터를 R에서 데이터 프레임으로 표현하세요**
  - Rstudio 소스 창에서 코딩을 진행하세요
  - 변수를 사용하세요 (37p 참고)
    - 각 컬럼의 변수 명은 컬럼 영문 명을 사용하세요
  - 완성된 데이터 프레임은 print 함수로 출력하세요

02 실습 - 자료구조 실습 ①

- Rstudio 실행하여 Source창 생성

- 컬럼 별 벡터 생성

```
> age <- c(30, 33, 35, 30, 68, 33) # 나이 벡터 생성
> job <- c("무직", "서비스", "관리직", "관리직", "은퇴", "관리직") # 직업 벡터 생성
> marital <- c("결혼", "결혼", "미혼", "결혼", "사별", "결혼") # 결혼 유무 벡터 생성
> balance <- c(1787, 4789, 1350, 1476, 4189, 3935) # 은행 잔고 벡터 생성
> campaign <- c("휴대폰", "휴대폰", "휴대폰", "Unknown", "유선전화", "휴대폰") # 상담 매체 벡터 생성
> y <- c(F, F, F, F, T, T) # 상담 결과 벡터 생성
```

- 컬럼 별 벡터 생성

```
> result <- data.frame(age, job, marital, balance, campaign, y) #데이터 프레임 생성
> print(result) #출력
```

02 실습 - 자료구조 실습 ②

- **Source 코드 실행 및 결과 확인**
  - ➢ **Source 창에서 코딩 진행 후 source창 우측 상단 Source 버튼 클릭**
  - ➢ **Console창 결과 확인**

```
> source("~/.active-rstudio-document", encoding = 'UTF-8')
  age    job marital balance campaign     y
1  30   무직    결혼    1787   휴대폰 FALSE
2  33 서비스    결혼    4789   휴대폰 FALSE
3  35 관리직    미혼    1350   휴대폰 FALSE
4  30 관리직    결혼    1476  Unknown FALSE
5  68   은퇴    사별    4189 유선전화  TRUE
6  33 관리직    결혼    3935   휴대폰  TRUE
```

- **작성된 소스코드 예시**

```r
#12 차시 R 데이터 자료구조 실습 코드
age <- c(30, 33, 35, 30, 68, 33) # 나이 벡터 생성
job <- c("무직", "서비스", "관리직", "관리직", "은퇴", "관리직") # 직업 벡터 생성
marital <- c("결혼", "결혼", "미혼", "결혼", "사별", "결혼") # 결혼 유무 벡터 생성
balance <- c(1787, 4789, 1350, 1476, 4189, 3935) # 은행 잔고 벡터 생성
campaign <- c("휴대폰", "휴대폰", "휴대폰", "Unknown", "유선전화", "휴대폰") # 상담 매체 벡터 생성
y <- c(F, F, F, F, T, T) # 상담 결과 벡터 생성

result <- data.frame(age, job, marital, balance, campaign, y)
print(result)

```

12차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

13차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 변수를 이해할 수 있다.
- 변수 작명 규칙을 알 수 있다.

학습 목차

- 변수
- 변수 작명 규칙

01.
변수란?

01 변수란? – 변수의 개념

- **변할 수 있는 값**
  - ‘x+5=9‘ , ‘x-4=5’ 에서, 변수 x는 방정식에 따라 다른 값을 가짐

- **프로그래밍에서의 변수**
  - 데이터를 저장하기 위해 이름을 할당 받은 메모리 공간
  - 프로그램 내에서 어떤 값을 저장해 놓을 수 있는 보관함

- **R에서의 변수란?**
  - 객체(Object)를 저장해 놓는 보관함

01 변수란? – 사용 방법

- **변수 사용 방법**

a ← 10

  - ‘a’라는 변수를 선언
  - ‘←’ 연산자 사용하여 ’10’ 이라는 수치형 값 할당

- **선언과 할당**
  - 선언: 변수 명을 등록하여 프로그램에 변수의 존재를 알리는 과정
  - 할당: 변수에 값을 저장하는 과정

01 변수란? – 변수 초기화

- **초기화란?**
  - 최초로 값을 저장하는 행위
  - a ← 10 : ‘a’라는 변수 선언과 동시에 10으로 변수를 초기화

- **변수 선언만 하기**
  - NULL 값 사용

a ← NULL

  - 할당할 값이 미정인 상태에서 변수 선언만 하고싶은 경우 사용

```
> a <- NULL # a 변수 선언만 하고 싶은 경우
> a
NULL
> a <- 10  # 후에 할당할 값이 있다면 언제든 할당 가능
> a
[1] 10
```

01 변수란? - 변수 할당 1

- **변수는 연산의 결과를 할당 가능**

```
a <- 10 + 20        # 30
b <- a - 5          # 25 , 변수도 산술 연산에 사용 가능 (데이터 타입이 맞을 시)
c <- c(1, 2, 3) + 5  # c(6, 7, 8) , 벡터에 대한 산술 연산 가능
d <- (10 > 5)        # TRUE
e <- b + d          # 산술 연산에서 TRUE는 1
```

- **변수는 하나의 객체만 저장 가능**

```
a <- 5, 10 # 에러 발생
a <- c(5, 10) # vector는 하나의 객체이기 때문에 저장 가능
```

01 변수란? - 변수 할당 2

- 특별한 데이터 타입 할당가능

```
a <- Inf
b <- 2/0     #Inf
b2 <- -2/0  #-Inf

c <- NA

d <- NaN
e <- 0/0     #NaN
```

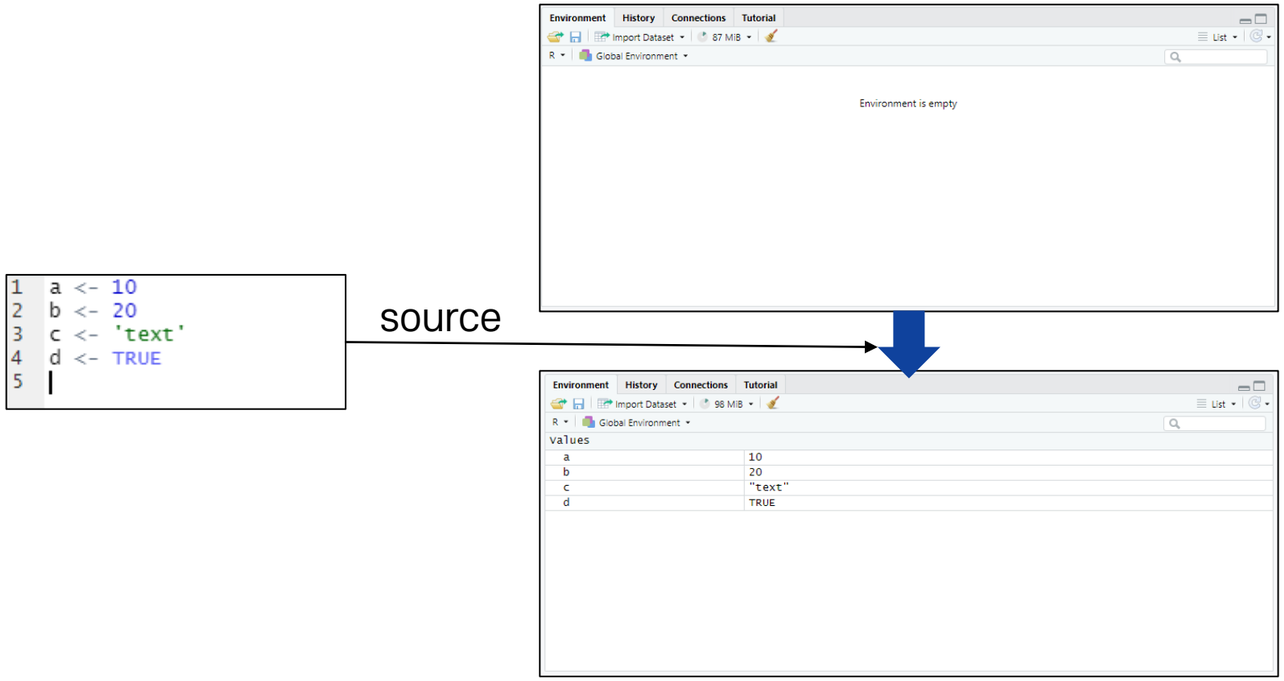
- 한번에 여러 변수 선언 또는 할당 불가능

```
# 에러 발생
a, b <- 5, 10


# 정상 동작
a <- 5
b <- 10
```

01 변수란? – 선언한 변수 확인하기

- **Print 함수 또는 Console 창에서 변수 명 입력하여 확인**

- **Rstudio의 Environment 창**
  - ➢ 선언한 변수 및 할당된 값 간편하게 확인가능



02.

변수 작명 규칙

02 변수 작명 규칙 – 변수 작명 규칙 1

- **첫 글자에는 문자 또는 마침표(.) 만 사용가능**
  - 숫자, 밑줄('_')으로 변수 명 시작 불가능

- **두번째 글자 부터는 숫자, 밑줄 사용가능**
  - 특수문자는 사용할 수 없음

```
a ← 10        # 가능
.a ← NULL     # 가능
a1 ← 10       # 가능
_a ← 10       # 에러 발생
1a ← 10       # 에러 발생
```

02 변수 작명 규칙 – 변수 작명 규칙 1

- **알파벳 대문자와 소문자는 별개의 문자 취급**

```
a <- 10
A <- 20
print(a)      #10
print(A)      #20
```

- **변수명 중간에 빈 칸 넣을 수 없음**

```
a bd <- 10      #에러 발생
a_bd <- 20      # 정상 동작. 빈 칸과 같은 효과를 내고 싶다면 밑줄 사용 권장
```

- **한국어 변수 사용가능**
  - ➢ 인코딩 이슈 등 발생할 수 있기 때문에 권장하지 않음

```
변수 <- 10      # 정상 동작
print(변수)      # 10
```

이번 차시 정리

변수란?

- 변수: 데이터를 저장하는 보관함
- 변수 선언, 할당
- 변수 초기화

선언한 변수 및 할당 값 확인

- Print 함수
- Console 창에서 타이핑
- Rstudio Environment 창

변수 작명 규칙 학습

13차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

14차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 함수에 대해 이해할 수 있다.
- 유용한 내장 함수를 알 수 있다.

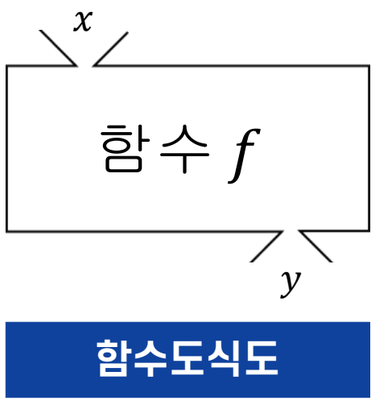
학습 목차

- 함수
- 유용한 내장 함수

01.
함수란?

01 함수란? – 함수의 개념

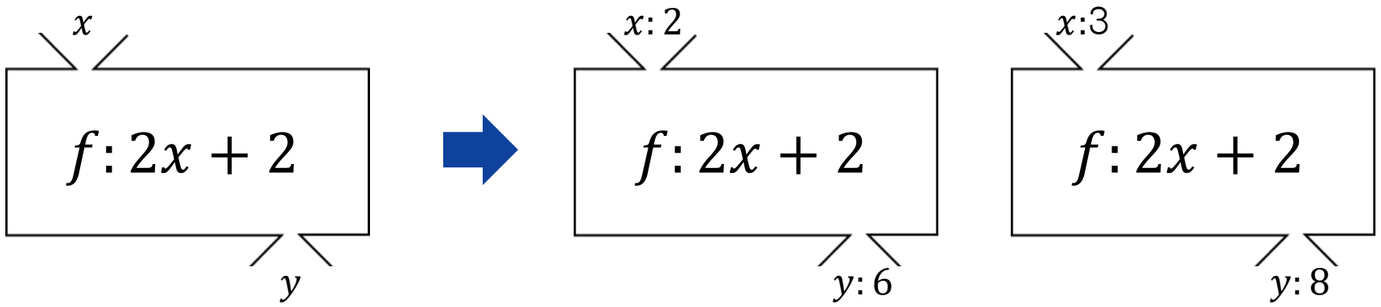
- **기본적인 함수형 $y=f(x)$**
  - 구성
    - x : 입력 변수
    - f : 함수
    - $f(x)$: 함수에 입력 값을 적용
    - y : 결과 변수



- **함수란?**
  - 입력 값을 넣고 실행 → 결과 값 리턴 하는 장치

01 함수란? – R의 함수

- **함수는 실행 내용에 대한 정의가 필요**
  - Eg.) f:2x+2
  - 어떠한 입력 값이 오던지 정의된 내용대로 실행하여 결과 값 리턴

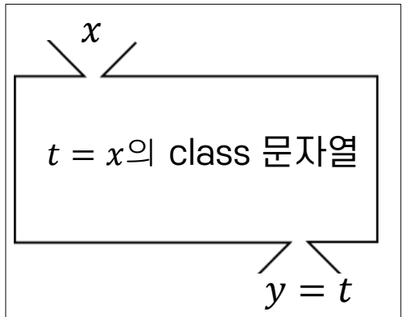


- **R에서의 함수**
  - 모든 객체 입출력 값으로 사용 가능
    - 수학의 함수와 다른 부분

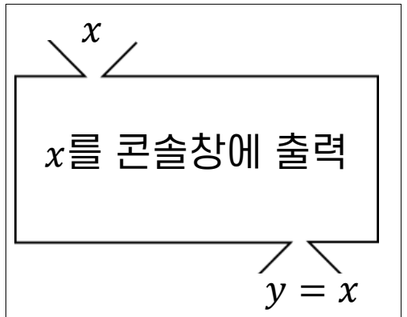
01 함수란? – R의 함수

- **R에서의 함수**
  - ➢ 모든 객체 입출력 값으로 사용 가능
    - ▪ 수학의 함수와 다른 부분

```
a <- TRUE
b <- class(a)    # class함수. a에 대한 데이터 타입을 문자열로 리턴
c <- print(a)    # print 함수. 입력받은 a를 콘솔 창에 출력(기능). 문자열이 아닌 변수 값을 그대로 리턴함
class(b)         # "character"
class(c)         # "logical"
y <- sqrt(25)    # sqrt 함수. 입력받은 값의 제곱근 리턴
y2 <- mean(c(1,2,3,4,5)) #mean 함수. 입력받은 값의 평균 리턴
```



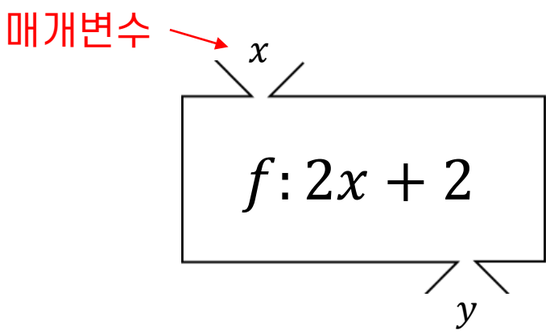
class 함수 도식도



print 함수 도식도

01 함수란? – 매개변수

- **매개변수(Parameter, Argument)**
  - ➢ 함수의 입력 변수를 부르는 명칭
  - ➢ 프로그래밍 용어



- **매개변수의 입력**
  - ➢ 함수의 정의에 알맞은 매개변수 입력 필요
  - ➢ 정의에 맞지않는 매개변수 입력 시, 오작동 또는 에러 발생

```
mean(c(1, 2, 3, 4, 5))          # mean 함수: 평균값 리턴. 결과: 3
mean(c("1", "2", "3"))          # 결과: NA
```

```
> mean(c("1", "2", "3"))
[1] NA
Warning message:
In mean.default(c("1", "2", "3")) :
  인자가 수치형 또는 논리형이 아니므로 NA를 반환합니다
```

01 함수란? – 매개변수 형식

- **함수의 정확한 형태**
  - ➢ Mean함수는 3개의 매개변수 존재
  - ➢ Mean 함수 매개변수 명: x, trim, na.rm

```
mean(x, …)

# S3 method for default
mean(x, trim = 0, na.rm = FALSE, …)
```

Arguments

x	An R object. Currently there are methods for numeric/logical vectors and date, date-time and time interval objects. Complex vectors are allowed for `trim = 0`, only.
trim	the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of trim outside that range are taken as the nearest endpoint.
na.rm	a logical value indicating whether `NA` values should be stripped before the computation proceeds.

01 함수란? - 매개변수 형식

- **함수의 정확한 형태**
  - Mean함수는 3개의 매개변수 존재
  - Mean 함수 매개변수 명: x, trim, na.rm

- **함수의 정확한 사용**
  - 매개변수 명과 매개변수값을 모두 적는 것이 정식 문법
  - "매개변수 명 = 매개변수 값"형태로 입력

```
mean( x=c(1, 2, 3, 4, 5), trim=0, na.rm=FALSE )     # 정식 문법. 결과는 mean(c(1, 2, 3, 4, 5)) 와 동일
```

01 함수란? - 매개변수 형식

- **매개 변수명 생략**
  - ➢ 매개변수 값을 정의 순서대로 입력 하는 경우, 매개변수명 생략 가능

01 함수란? – 매개변수 형식

- **매개 변수값 생략**
  - ➢ 매개변수는 필수 매개변수와 선택적 매개변수로 구분
    - ▪ 기본값은 함수 선언 시 정의 됨
    - ▪ 기본값 지정 X: 필수 매개변수
    - ▪ 기본값 지정 O: 선택적 매개변수
  - ➢ 선택적 매개변수는 생략 가능
    - ▪ 생략 되는 경우 기본값 사용

```
mean(x, …)

# S3 method for default
mean(x, trim = 0, na.rm = FALSE, …)
```

기본값
(Default value)

```
# 모두 정상작동하는 코드
mean( x=c(1, 2, 3, 4, 5), trim=0, na.rm=FALSE )   # 정식 문법
mean( x=c(1, 2, 3, 4, 5), trim=0)                  # 선택적 매개변수 na.rm 생략
mean( x=c(1, 2, 3, 4, 5))                          # 선택적 매개변수 모두 생략
mean( c(1, 2, 3, 4, 5), 0, FALSE)                  # 매개변수 명 생략
mean( c(1, 2, 3, 4, 5))                            # 매개변수 명 및 선택적 매개변수 생략
```

02.

유용한 내장 함수

02 유용한 내장 함수 – 내장 함수란?

- **내장 함수**
  - 프로그램에서 기본적으로 정의되어 제공하는 함수
  - Eg.) print, class, sqrt, mean

- **사용자 정의 함수**
  - 사용자가 직접 정의하여 만들어내는 함수

```

myFunc <- function(x) {
  return (2*x + 2)
}
a <- myFunc(2)
print(a)                #6
print(myFunc(3))        #8
```

02 유용한 내장 함수 – 강제 형변환 함수

- **As 함수**
  - 데이터 형식을 강제로 변환 시킴
  - 데이터 우선순위 낮은 타입 → 높은 타입으로 변화는 모두 가능
  - 데이터 우선순위 높은 타입 → 낮은 타입으로 변화는 일부 가능
  - 데이터 우선순위
    - Character > Numeric > Logical

```
#낮은 우선순위 → 높은 우선순위
vec <- c( T, F, FALSE)            #logical 타입 vector 생성
vec2 <-as.numeric(vec)            # c(1, 0, 0). numeric 타입 강제 형변환. T=1, F=0으로 취급
vec3 <-as.logical(vec2)           # c(T, F, F). 다시 logical 타입 변환 가능. 0 초과의 수 =T, 0 이하의
수=F 취급
vec4 <-as.character(vec3)         # c("TRUE", "FALSE", "FALSE")

#높은 우선순위 → 낮은 우선순위
test <- c("char", "10", "TRUE")
testToNum <- as.numeric(test)   # c(NA, 10, NA). 숫자로 이루어진 문자만 형변환 가능
testToLogic <- as.logical(test)   # c(NA, NA, TRUE). 논리형 값과 동일한 문자만 형변환 가능
```

02 유용한 내장 함수 – 데이터 생성 함수

- **seq 함수**
  - ➢ From 부터 by씩 증가하는 length개의 숫자로 이루어진 벡터 생성

```
seq( length=5, from=3, by=2)          # c(3, 5, 7, 9, 11)
```

- **rnorm 함수**
  - ➢ 평균이 mean, 분산이 se 인 정규분포를 따르는 n개의 숫자로 이루어진 벡터 생성

```
rnorm( n=4, mean =0, sd=1 )      # 평균 0, 분산 1인 정규분포 따르는 4개 데이터 생성
```

- **runif 함수**
  - ➢ [min, max] 범위 사이 n개의 난수로 이루어진 벡터 생성

```
runif( n=4, min=1, max=100)      # 최소값 1, 최대값 100 사이의 무작위 값 4개 생성
```

이번 차시 정리

함수란?

- 매개변수를 받아 정의된 내용을 실행하며, 결과 값을 리턴하는 장치
- 매개변수는 특정 경우 생략가능

유용한 내장함수 소개

- 산술 함수
  - ✓ Min, max, sum, prod, factorial, abs
- 데이터 생성함수
  - ✓ Seq, rnom, runif

14차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

15차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 패키지에 대해 알 수 있다.
- 패키지 탐색/설치/사용 방법에 대해 알 수 있다.
- 컴퓨터 간 동일한 패키지 환경 구축 방법에 대해 이해할 수 있다.

학습 목차

- 패키지란?
- 패키지 탐색/설치/사용 방법
- 컴퓨터 간 동일한 패키지 환경 구축 방법

01.

패키지란?

01 패키지란? – 패키지(Package) 개념

- **함수들의 모음**
  - ➢ 유사한 기능을 하는 함수들을 한데 묶어 관리하는 꾸러미

- **R 주요 패키지 소개**

분류	기능	패키지명
로드	DB 처리	RMySQL, RPostgresSQL, RSQLite
	엑셀 처리	XLConnect, xlsx
핸들링	데이터 조작	dplyr
	데이터 레이아웃 변환	reshape2
	문자열 처리	stringr
시각화	그래프	ggplot2
	워드클라우드	wordcloud
	구글 차트	googleVis
모델링	선형 혼합 효과 모형	lme4
	비선형 혼합 효과 모형	nlme
	머신러닝 랜덤 포레스트	randomForest
	범주형 데이터 시각화	vcd
	Lasso, elastic-net 회귀분석	glmnet
	생존분석	survival
	회귀분석 및 분류 모델의 트레이닝	caret
리포트	웹 대시보드	shiny
공간	지리 지도	maps
	구글 지도	ggmap
기타	대용량 텍스트 데이터 처리	data.table
	멀티코어 사용으로 병렬 프로세싱 처리	parallel
	XML 문서 처리	XML
	JSON 데이터 처리	jsonlite
	HTTP Connection	httr

02.

패키지 탐색/설치/사용 방법

02 패키지 탐색 방법 – CRAN 사용

- **R언어에서 사용 가능한 모든 패키지 알파벳순으로 정리**
  - https://cran.r-project.org/web/packages/available_packages_by_name.html

Available CRAN Packages By Name

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

A3	Accurate, Adaptable, and Accessible Error Metrics for Predictive Models
aaSEA	Amino Acid Substitution Effect Analyser
AATtools	Reliability and Scoring Routines for the Approach-Avoidance Task
ABACUS	Apps Based Activities for Communicating and Understanding Statistics
abbyyR	Access to Abbyy Optical Character Recognition (OCR) API
abc	Tools for Approximate Bayesian Computation (ABC)
abc.data	Data Only: Tools for Approximate Bayesian Computation (ABC)
ABC.RAP	Array Based CpG Region Analysis Pipeline
abcADM	Fit Accumulated Damage Models and Estimate Reliability using ABC
ABCanalysis	Computed ABC Analysis
abcdeFBA	ABCDE_FBA: A-Biologist-Can-Do-Everything of Flux Balance Analysis with this package
ABCoptim	Implementation of Artificial Bee Colony (ABC) Optimization
ABCp2	Approximate Bayesian Computational Model for Estimating P2
abcrf	Approximate Bayesian Computation via Random Forests

02 패키지 탐색 방법 - CRAN 사용

- **R언어에서 사용 가능한 모든 패키지 알파벳순으로 정리**
  - https://cran.r-project.org/web/packages/available_packages_by_name.html

- **패키지에 대한 자세한 명세 및 패키지 다운로드 가능**

ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics

A system for 'declaratively' creating graphics, based on "The Grammar of Graphics". You provide the data, tell 'ggplot
it takes care of the details.

Version:	3.3.5
Depends:	R (≥ 3.3)
Imports:	digest, glue, grDevices, grid, gtable (≥ 0.1.1), isoband, MASS, mgcv, rlang (≥ 0.4.10), scales (≥ 0.5.
Suggests:	covr, ragg, dplyr, ggplot2movies, hexbin, Hmisc, interp, knitr, lattice, mapproj, maps, maptools, mu rpart, sf (≥ 0.7-3), svglite (≥ 1.2.0.9001), testthat (≥ 2.1.0), vdiffr (≥ 1.0.0), xml2
Enhances:	sp
Published:	2021-06-25
Author:	Hadley Wickham [aut], Winston Chang [aut], Lionel Henry [aut], Thomas Lin Pedersen Hiroaki Yutani [aut], Dewey Dunnington [aut], RStudio [cph, fnd]
Maintainer:	Thomas Lin Pedersen <thomas.pedersen at rstudio.com>
BugReports:	https://github.com/tidyverse/ggplot2/issues
License:	MIT + file LICENSE
URL:	https://ggplot2.tidyverse.org, https://github.com/tidyverse/ggplot2
NeedsCompilation:	no
Citation:	ggplot2 citation info
Materials:	README NEWS
In views:	Graphics, Phylogenetics, TeachingStatistics
CRAN checks:	ggplot2 results

Documentation:

Reference manual:	ggplot2.pdf
Vignettes:	Extending ggplot2 Using ggplot2 in packages Aesthetic specifications

Downloads:

Package source:	ggplot2_3.3.5.tar.gz
Windows binaries:	r-devel: ggplot2_3.3.5.zip, r-release: ggplot2_3.3.5.zip, r-oldrel: ggplot2_3.3.5.zip
macOS binaries:	r-release (arm64): ggplot2_3.3.5.tgz, r-release (x86_64): ggplot2_3.3.5.tgz, r-oldrel: ggplot2_3.3.5.tgz

02 패키지 탐색 방법 - CRAN 사용

- **R언어에서 사용 가능한 모든 패키지 알파벳순으로 정리**
  - https://cran.r-project.org/web/packages/available_packages_by_name.html

- **패키지에 대한 자세한 명세 및 패키지 다운로드 가능**
- **기능에 따른 패키지 분류 존재**
  - https://cran.r-project.org/web/views/

CRAN Task Views

CRAN task views aim to provide some guidance which packages on CRAN are relevant for tasks related to a certain topic. They give a brief overview of the included packages and can be automatically installed using the ctv package. The views are intended to have a sharp focus so that it is sufficiently clear which packages should be included (or excluded) - and they are *not* meant to endorse the "best" packages for a given task.

- To automatically install the views, the ctv package needs to be installed, e.g., via
  `install.packages("ctv")`
  and then the views can be installed via `install.views` or `update.views` (where the latter only installs those packages are not installed and up-to-date), e.g.,
  `ctv::install.views("Econometrics")`
  `ctv::update.views("Econometrics")`
- The task views are maintained by volunteers. You can help them by suggesting packages that should be included in their task views. The contact e-mail addresses are listed on the individual task view pages.
- For general concerns regarding task views contact the ctv package maintainer.

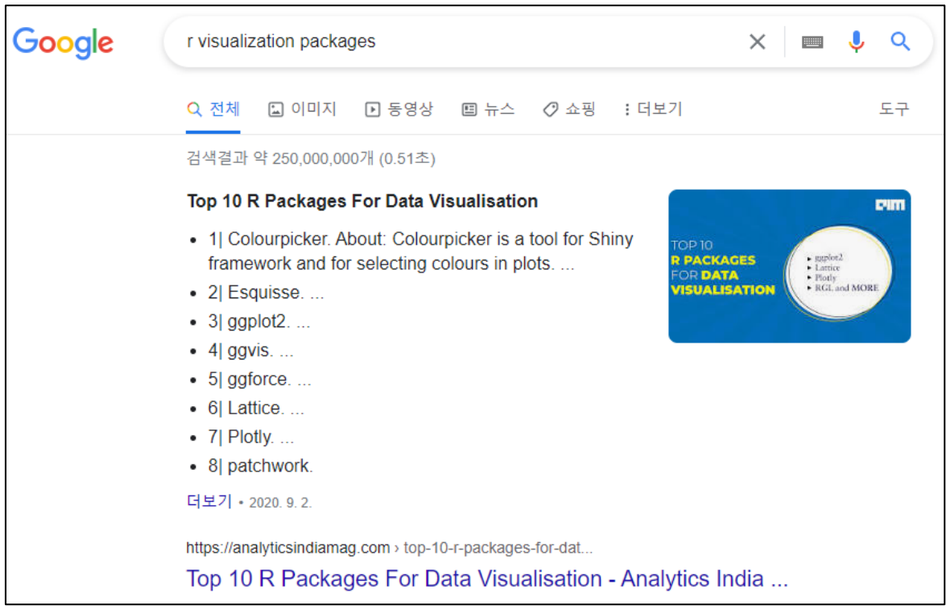
Topics

Bayesian	Bayesian Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
Databases	Databases with R
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
ExtremeValue	Extreme Value Analysis
Finance	Empirical Finance
FunctionalData	Functional Data Analysis

02 패키지 탐색 방법 – 인터넷 키워드 검색

- R 유저간 네트워크 매우 활발

- 검색 엔진의 발전으로 간편한 검색 가능



구글 검색 엔진을 사용한 R 시각화 패키지 탐색

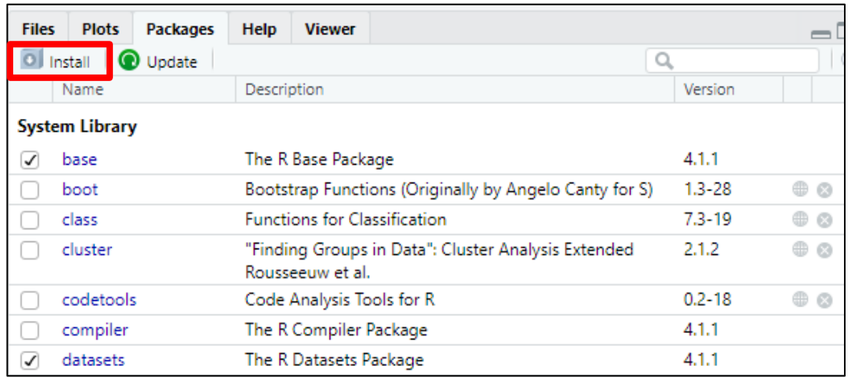
02 패키지 설치 및 사용 – 설치 방법

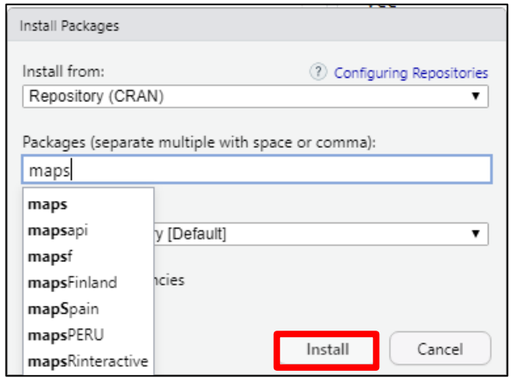
- **함수를 이용하여 설치**
  - ➢ install.packages("패키지 명")

```
> install.packages("maps")
WARNING: Rtools is required to build R packages but is not currently installed. Please
 download and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.1/maps_3.4.0.zip'
Content type 'application/zip' length 3119228 bytes (3.0 MB)
downloaded 3.0 MB

package ‘maps’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\WonsupShin\AppData\Local\Temp\Rtmpcf6pfn\downloaded_packages
>
```

02 패키지 설치 및 사용 – 설치 방법

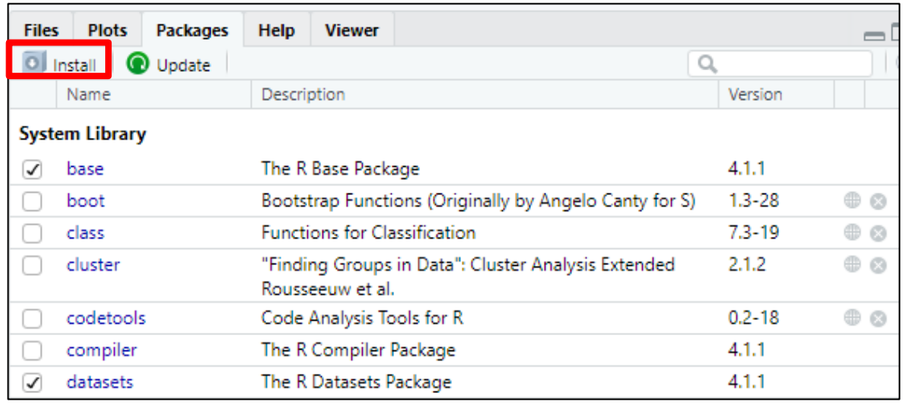
- R 스튜디오 인터페이스 이용하여 설치

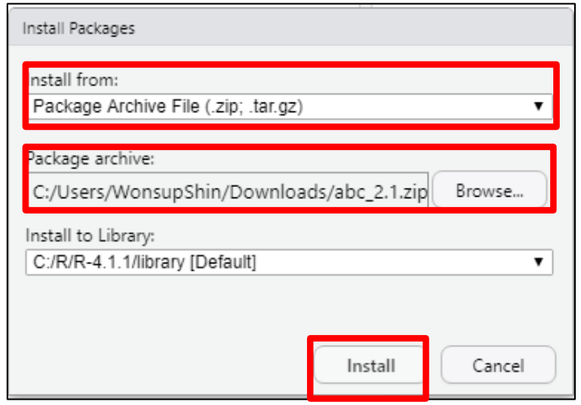




02 패키지 설치 및 사용 – 설치 방법

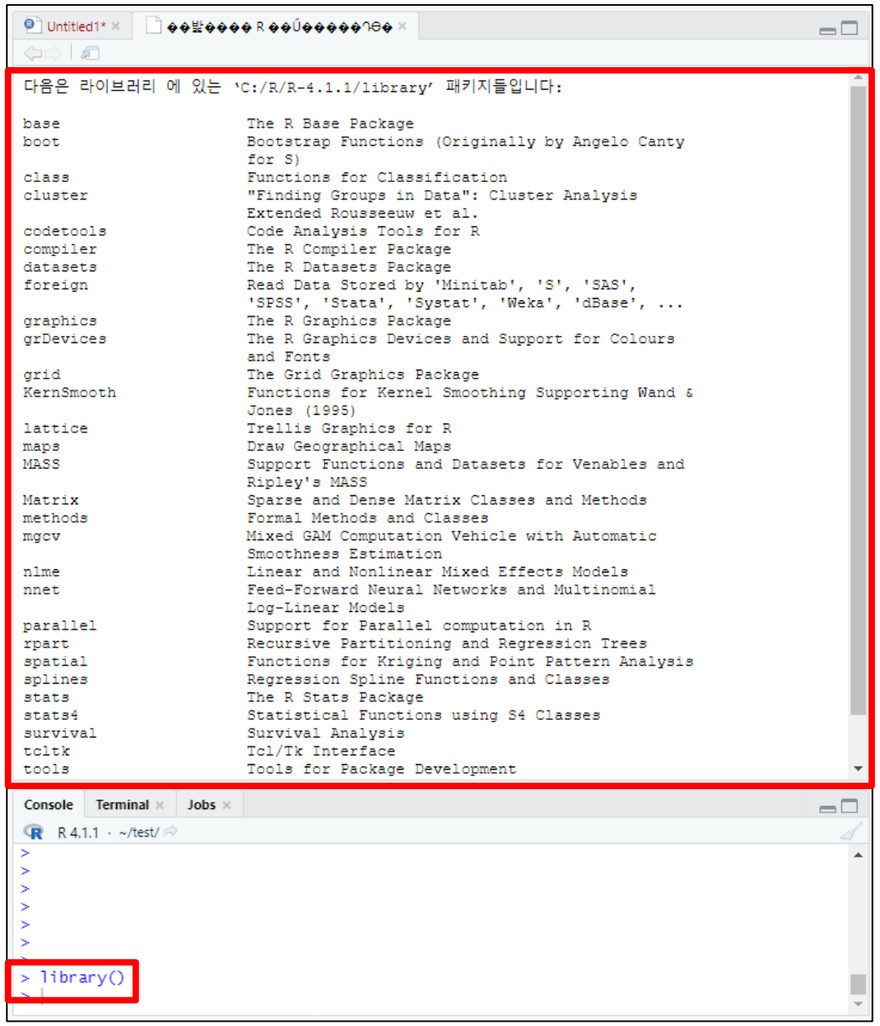
- **R 스튜디오 인터페이스 이용한 오프라인 설치**
  - ➢ 패키지 파일을 가지고 있는 경우 사용





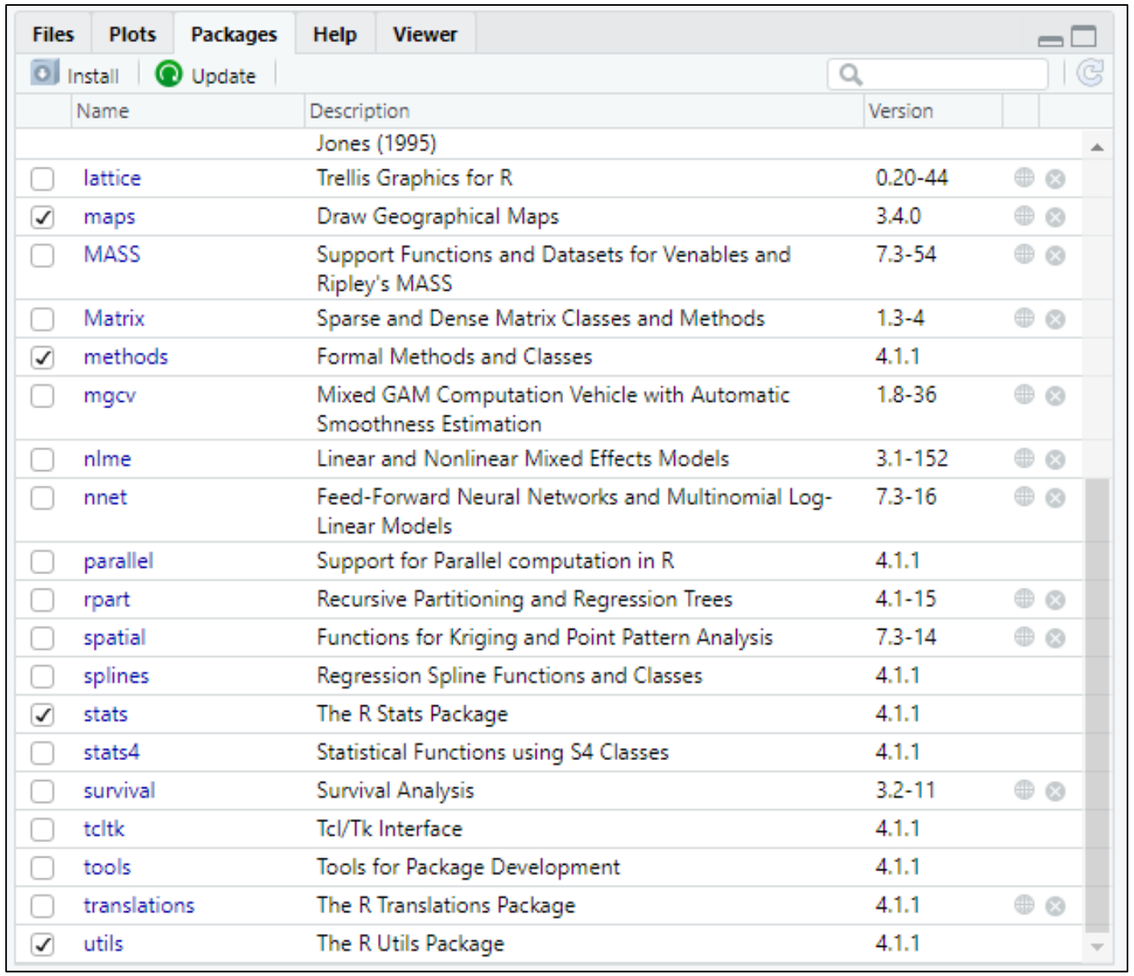
02 패키지 설치 및 사용 – 설치 방법

- **패키지 설치 확인**
  - library()
  - Package 창 활용



02 패키지 설치 및 사용 – 설치 방법

- **패키지 설치 확인**
  - ➢ library( )
    - ▪ Package 창 활용
    - ▪ 설치된 패키지 보여줌
    - ▪ 로딩 된 패키지는 체크박스에 표시



02 패키지 설치 및 사용 – 사용 방법

- **패키지 로딩**
  - ➢ library(패키지명)
    - ▪ Package 창 활용
    - ▪ 설치되지 않은 패키지 로딩 시도 시, 오류 리턴

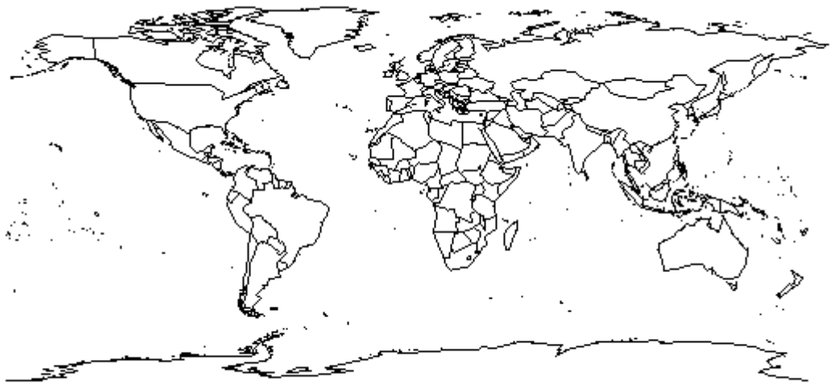
```
> library(maps)
>
```

```
> library(notInstalledPackage)
Error in library(notInstalledPackage) :
  'notInstalledPackage'이라고 불리는 패키지가 없습니다
```

02 패키지 설치 및 사용 – 사용 방법

- **정상 로딩 시, 패키지 내부에 저장된 함수 사용 가능**
  - maps 패키지 안에는 map( ) 함수 존재

```
library(maps)
map()
```



02 패키지 관련 함수 – 패키지 관리 함수

- **install.packages("패키지 이름")**
  - 해당 패키지 설치 함수
- **library(help= 패키지 이름)**
  - 패키지에 대한 정보 출력
- **update.packages("패키지 이름")**
  - 해당 패키지를 최신 버전으로 업데이트
  - update.packages( )
    - 모든 패키지를 업데이트
- **remove.packages("패키지 이름")**
  - 해당 패키지를 삭제

02 패키지 관련 함수 – 패키지 관리 함수

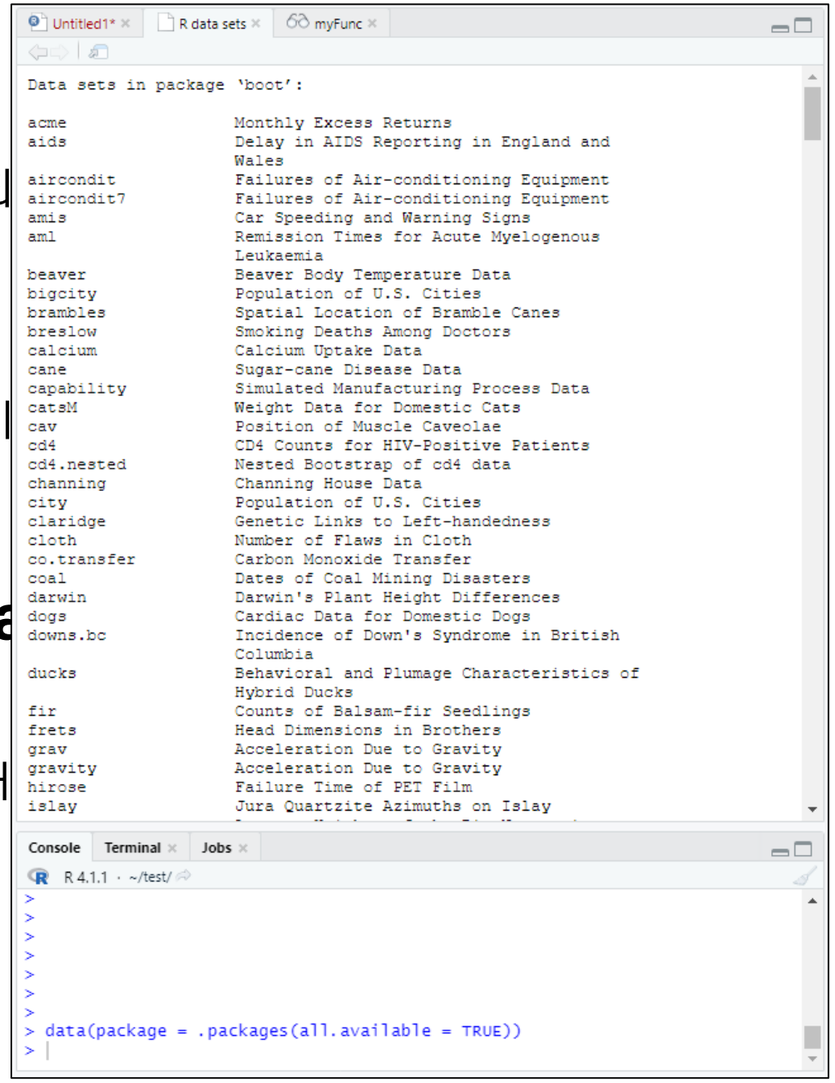
- **패키지 이름::패키지 함수( )**
  - library( ) 함수를 사용하지 않고, 특정 패키지의 특정 함수 사용
  - Eg.) maps::map( )
  - 여러 패키지를 사용하는 경우 함수명이 겹치는 경우가 종종 발생
    - 위와 같이 사용시 함수가 어떤 패키지 것인지 쉽게 구분 가능

- **data(package = .packages(all.available = TRUE))**
  - 설치된 패키지의 모든 내장 데이터 보는 기능
  - 별도의 외부 데이터 수집 필요없이 내장 데이터 사용하여 빠르게 테스트 가능

02 패키지 관련 함수 – 패키지 관리 함수

- **패키지 이름::패키지 함수( )**
  - library( ) 함수를 사용하지 않고, 특정 패키지의
  - Eg.) maps::map( )
  - 여러 패키지를 사용하는 경우 함수명이 겹치는
    - 위와 같이 사용시 함수가 어떤 패키지 것인지

- **data(package = .packages(all.a**
  - 설치된 패키지의 모든 내장 데이터 보는 기능
  - 별도의 외부 데이터 수집 필요없이 내장 데이터



이번 차시 정리

패키지란?

- 유사한 기능을 하는 함수들의 모음

패키지 탐색 방법

- CRAN
- 인터넷 키워드 검색

이번 차시 정리

패키지 설치/사용 방법

- 설치
  - ➢ install.packages("패키지 명")
  - ➢ Rstudio 인터페이스 사용
- 사용
  - ➢ libarary(패키지명)
- 패키지 관련 함수

15차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

16차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 변수에 대해 알 수 있다.
- 함수에 대해 알 수 있다.
- 패키지에 대해 알 수 있다.

학습 목차

- 변수
- 함수
- 패키지

01. Quiz

01 Quiz. 변수 – 학습내용 되돌아보기

- Q1. 변수는 객체를 담는 보관함 역할을 하며 ( ) 연산자를 통해 값을 할당한다.

  ① <- ② -> ③ << ④ >>

- Q2. 변수를 선언만 하고싶은 경우, ( ) 값으로 초기화한다.

  ① NaN ② NULL ③ NA ④ 0

- Q3. 변수는 한번에, 두 개 이상의 값으로 초기화 할 수 있다.

  ① True ② False

01 Quiz. 변수 – 학습내용 되돌아보기

- Q1. 변수는 객체를 담는 보관함 역할을 하며 (<-) 연산자를 통해 값을 할당한다.

  ① <- ② -> ③ << ④ >>

- Q2. 변수를 선언만 하고싶은 경우, (NULL) 값으로 초기화한다.

  ① NaN ② NULL ③ NA ④ 0

- Q3. 변수는 한번에, 두 개 이상의 값으로 초기화 할 수 있다.

  ① True ② False

01 Quiz. 변수 – 학습내용 되돌아보기

- Q4. 변수 선언 시, 숫자 또는 밑줄('_')로 변수 명 시작 하는 것은 불가능하다.

  ① True ② False

- Q5. 변수 선언 시, 변수 명 두번째 글자부터는 특수문자 사용이 가능하다.

  ① True ② False

- Q6. 변수 선언 시, 알파벳 대문자와 소문자는 구분하지 않는다.

  ① True ② False

01 Quiz. 변수 – 학습내용 되돌아보기

- Q4. 변수 선언 시, 숫자 또는 밑줄('_')로 변수 명 시작 하는 것은 불가능하다.

  ① True ② False

- Q5. 변수 선언 시, 변수 명 두번째 글자부터는 특수문자 사용이 가능하다.

  ① True ② False

- Q6. 변수 선언 시, 알파벳 대문자와 소문자는 구분하지 않는다.

  ① True ② False

01 Quiz. 변수 – 학습내용 되돌아보기

- Q7. 함수는 (　　)를 입력받아 정의된 내용을 실행하며,
  결과 값을 리턴하는 장치이다.

  ① 맴버변수　② 지역변수　③ 매개변수　④ 전역변수

- Q8. 프로그램에서 기본적으로 정의되어 제공하는 함수를
  (　　)라 부른다

  ① 사용자 정의 함수　② 기본함수　③ 외장함수　④ 내장함수

- Q9. 데이터 타입 강제 형 변환 해주는 함수 명은 (　　) 이다.

  ① as　② seq　③ convert　④ runif

01 Quiz. 변수 – 학습내용 되돌아보기

- Q7. 함수는 (　매개변수　)를 입력받아 정의된 내용을 실행하며,
  결과 값을 리턴하는 장치이다.

  ① 맴버변수　② 지역변수　③ 매개변수　④ 전역변수

- Q8. 프로그램에서 기본적으로 정의되어 제공하는 함수를
  (　내장함수　)라 부른다

  ① 사용자 정의 함수　② 기본함수　③ 외장함수　④ 내장함수

- Q9. 데이터 타입 강제 형 변환 해주는 함수 명은 (　as　) 이다.

  ① as　② seq　③ convert　④ runif

01 Quiz. 변수 – 학습내용 되돌아보기

- Q10. 함수의 매개변수는 필수적 매개변수와 선택적 매개변수로 나뉘며, 필수적 매개변수는 생략가능하다.

  ① True ② False

- Q11. 내장 함수외에 사용자가 직접 함수를 만들 수 있으며, 사용자가 직접 만든 함수는 '사용자 정의 함수' 라고 부른다.

  ① True ② False

01 Quiz. 변수 – 학습내용 되돌아보기

- Q10. 함수의 매개변수는 필수적 매개변수와 선택적 매개변수로 나뉘며, 필수적 매개변수는 생략가능하다.

  ① True ② False

- Q11. 내장 함수외에 사용자가 직접 함수를 만들 수 있으며, 사용자가 직접 만든 함수는 '사용자 정의 함수' 라고 부른다.

  ① True ② False

02.
실습

01 Quiz. 변수 – 학습내용 되돌아보기

- **R은행은 이전에 고객들에게 대출 권유한 결과를 데이터셋화 하였었다.**
  - 12차시 자료 참고

- **상황 설명**
  해당 데이터 셋에서 간단한 통계정보를 추출하고, 데이터셋을 좀더 명료하게 수정하려 한다.

나이 (age)	직종 (job)	결혼 유무 (marital)	은행 잔고 (balance)	상담 매체 (campaign)	대출 신청 하였는가? (y)
30	무직	결혼	1787	휴대폰	FALSE
33	서비스	결혼	4789	휴대폰	FALSE
35	관리직	미혼	1350	휴대폰	FALSE
30	관리직	결혼	1476	Unknown	FALSE
68	은퇴	사별	4189	유선전화	TRUE
33	관리직	결혼	3935	휴대폰	TRUE

01 Quiz. 변수 – 학습내용 되돌아보기

- **데이터셋 준비**
  - 12차시 자료 참고

```
age <- c(30, 33, 35, 30, 68, 33)                                          # 나이 벡터 생성
job <- c("무직", "서비스", "관리직", "관리직", "은퇴", "관리직")                # 직업 벡터 생성
marital <- c("결혼", "결혼", "미혼", "결혼", "사별", "결혼")                   # 결혼 유무 벡터 생성
balance <- c(1787, 4789, 1350, 1476, 4189, 3935)                          # 은행 잔고 벡터 생성
campaign <- c("휴대폰", "휴대폰", "휴대폰", "Unknown", "유선전화", "휴대폰")    # 상담 매체 벡터 생성
y <- c(F, F, F, F, T, T)                                                  # 상담 결과 벡터 생성
result <- data.frame(age, job, marital, balance, campaign, y)             #데이터 프레임 생성
print(result)                                                             #출력
```

```
> source("~/.active-rstudio-document", encoding = 'UTF-8')
  age    job marital balance campaign     y
1  30   무직    결혼    1787   휴대폰 FALSE
2  33 서비스    결혼    4789   휴대폰 FALSE
3  35 관리직    미혼    1350   휴대폰 FALSE
4  30 관리직    결혼    1476  Unknown FALSE
5  68   은퇴    사별    4189 유선전화  TRUE
6  33 관리직    결혼    3935   휴대폰  TRUE
```

01 Quiz. 변수 – 학습내용 되돌아보기

```
age <- c(30, 33, 35, 30, 68, 33)                                  # 나이 벡터 생성
job <- c("무직", "서비스", "관리직", "관리직", "은퇴", "관리직")          # 직업 벡터 생성
marital <- c("결혼", "결혼", "미혼", "결혼", "사별", "결혼")              # 결혼 유무 벡터 생성
balance <- c(1787, 4789, 1350, 1476, 4189, 3935)                  # 은행 잔고 벡터 생성
campaign <- c("휴대폰", "휴대폰", "휴대폰", "Unknown", "유선전화", "휴대폰")  # 상담 매체 벡터 생성
y <- c(F, F, F, F, T, T)                                          # 상담 결과 벡터 생성
result <- data.frame(age, job, marital, balance, campaign, y)     #데이터 프레임 생성
print(result)                                                     #출력
```

- **Q. 고객들의 평균 나이와 평균 은행 잔고를 각각 'age_mean', 'balance_mean' 변수에 할당하고 print함수 사용하여 출력하세요.**
  - mean 함수 사용

01 Quiz. 변수 – 학습내용 되돌아보기

```
age <- c(30, 33, 35, 30, 68, 33)                                  # 나이 벡터 생성
job <- c("무직", "서비스", "관리직", "관리직", "은퇴", "관리직")          # 직업 벡터 생성
marital <- c("결혼", "결혼", "미혼", "결혼", "사별", "결혼")              # 결혼 유무 벡터 생성
balance <- c(1787, 4789, 1350, 1476, 4189, 3935)                  # 은행 잔고 벡터 생성
campaign <- c("휴대폰", "휴대폰", "휴대폰", "Unknown", "유선전화", "휴대폰")  # 상담 매체 벡터 생성
y <- c(F, F, F, F, T, T)                                          # 상담 결과 벡터 생성
result <- data.frame(age, job, marital, balance, campaign, y)     #데이터 프레임 생성
print(result)                                                     #출력
```

- **Q. 고객들의 평균 나이와 평균 은행 잔고를 각각 'age_mean', 'balance_mean' 변수에 할당하고 print함수 사용하여 출력하세요.**
  - mean 함수 사용

```
age_mean <- mean(age)
balance_mean <- mean(balance)
print(age_mean)
print(balance_mean)
```

```
> print(age_mean)
[1] 38.16667
> print(balance_mean)
[1] 2921
```

01 Quiz. 변수 – 학습내용 되돌아보기

```
age <- c(30, 33, 35, 30, 68, 33)                                  # 나이 벡터 생성
job <- c("무직", "서비스", "관리직", "관리직", "은퇴", "관리직")          # 직업 벡터 생성
marital <- c("결혼", "결혼", "미혼", "결혼", "사별", "결혼")              # 결혼 유무 벡터 생성
balance <- c(1787, 4789, 1350, 1476, 4189, 3935)                  # 은행 잔고 벡터 생성
campaign <- c("휴대폰", "휴대폰", "휴대폰", "Unknown", "유선전화", "휴대폰")  # 상담 매체 벡터 생성
y <- c(F, F, F, F, T, T)                                          # 상담 결과 벡터 생성
result <- data.frame(age, job, marital, balance, campaign, y)     #데이터 프레임 생성
print(result)                                                     #출력
```

- **Q. 아래는 대출을 신청한 사람 수를 계산하는 코드입니다.**
  주석을 참고하여 빈칸을 채우세요.

```
yToNum <- [ ? ]                  # c(0,0,0,0,1,1) ,상담 결과 벡터를 numeric 형식으로 강제 형 변환
yCount <- sum(yToNum)            # sum: 벡터 값들의 총합 리턴
print(yCount)
```

01 Quiz. 변수 – 학습내용 되돌아보기

```
age <- c(30, 33, 35, 30, 68, 33)                                  # 나이 벡터 생성
job <- c("무직", "서비스", "관리직", "관리직", "은퇴", "관리직")          # 직업 벡터 생성
marital <- c("결혼", "결혼", "미혼", "결혼", "사별", "결혼")              # 결혼 유무 벡터 생성
balance <- c(1787, 4789, 1350, 1476, 4189, 3935)                  # 은행 잔고 벡터 생성
campaign <- c("휴대폰", "휴대폰", "휴대폰", "Unknown", "유선전화", "휴대폰")  # 상담 매체 벡터 생성
y <- c(F, F, F, F, T, T)                                          # 상담 결과 벡터 생성
result <- data.frame(age, job, marital, balance, campaign, y)     #데이터 프레임 생성
print(result)                                                     #출력
```

- **Q. 아래는 대출을 신청한 사람 수를 계산하는 코드입니다.**
  주석을 참고하여 빈칸을 채우세요.

```
yToNum <- as.numeric(y)        # c(0,0,0,0,1,1) ,상담 결과 벡터를 numeric 형식으로 강제 형 변환
yCount <- sum(yToNum)          # sum: 벡터 값들의 총합 리턴
print(yCount)
```

➢ logical 값을 numeric으로 형변환 시, True는 1로, False는 0으로 변환 됨

01 Quiz. 변수 – 학습내용 되돌아보기

- **Q. 다음 수정 사항에 따라 데이터값을 변경하세요.**

[수 정 사 항]

- 1. 문자 데이터 수정은 stringr 패키지의 함수 사용할 것
- 2. job 컬럼의 “은퇴” 값은 “무직”으로 대체 할 것
- 3. marital 컬럼은 logical 값으로 관리 할 것
  - ➢ “결혼” 값은 T로, 나머지 값은 모두 F로 관리
- 4. 수정된 결과물은 result_new 변수에 할당 후 출력 할 것

나이 (age)	직종 (job)	결혼 유무 (marital)	은행 잔고 (balance)	상담 매체 (campaign)	대출 신청 하였는가? (y)
30	무직	TRUE	1787	휴대폰	FALSE
33	서비스	TRUE	4789	휴대폰	FALSE
35	관리직	FALSE	1350	휴대폰	FALSE
30	관리직	TRUE	1476	Unknown	FALSE
68	무직	FALSE	4189	유선전화	TRUE
33	관리직	TRUE	3935	휴대폰	TRUE

01 Quiz. 변수 – 학습내용 되돌아보기

- **stringr 패키지 설치 및 로딩**

```
install.packages("stringr")        #설치
library(stringr)                   #로딩
```

- **stringr 패키지내 문자열 replace관련 함수 탐색**
  - ➢ CRAN 내 document 참조

stringformattr	Dynamic String Formatting
stringi	Character String Processing Facilities
stringr	Simple, Consistent Wrappers for Common String Operations
stringx	Drop-in Replacements for Base String Functions Powered by 'stringi'

Documentation:

Reference manual: stringr.pdf

Vignettes: Regular expressions
Introduction to stringr

str_replace *Replace matched patterns in a string.*

Description

Vectorised over `string`, `pattern` and `replacement`.

Usage

```
str_replace(string, pattern, replacement)

str_replace_all(string, pattern, replacement)
```

Arguments

`string`	Input vector. Either a character vector, or something coercible t
`pattern`	Pattern to look for. The default interpretation is a regular expression, as described in search-regex. Control options with `regex()`. Match a fixed string (i.e. by comparing only bytes), using `fi` fast, but approximate. Generally, for matching human text, you which respects character matching rules for the specified locale.
`replacement`	A character vector of replacements. Should be either length o

[분석 결과]

str_replace(string, pattern, replacement)
: string 중에서 pattern과 동일한 문자를 replacement 문자로 변경

str_replace 함수 사용하기로 결정

01 Quiz. 변수 – 학습내용 되돌아보기

- **job 컬럼 수정**

```
job_new <- str_replace(job, "은퇴", "무직")      # "은퇴" 값을 "무직"으로 변경
print(job_new)                                   # 결과 확인
```

```
> job_new
[1] "무직"    "서비스" "관리직" "관리직" "무직"    "관리직"
```

- **marital 컬럼 수정**

```
marital_new <- str_replace(marital, "결혼", "T")   # "결혼" 값을 "T"로 변경
print(marital_new)
marital_logical <- as.logical(marital_new)         # marital_new 벡터 logical로 강제 형변환
print(marital_logical)                             # character -> logical로 형변환 시, "T", "F"외의 값은 NA로 변환 됨
marital_logical[is.na(marital_logical)] = FALSE    # NA값을 F로 변환
print(marital_logical)
```

```
> marital_new <- str_replace(marital, "결혼", "T") # "결혼" 값을 "T"로 변경
> print(marital_new)
[1] "T"    "T"    "미혼" "T"    "사별" "T"
> marital_logical <- as.logical(marital_new)       # marital_new 벡터 logical로 강제 형변환
> print(marital_logical)
[1] TRUE TRUE   NA TRUE   NA TRUE
> marital_logical[is.na(marital_logical)] = F
> print(marital_logical)
[1]  TRUE  TRUE FALSE  TRUE FALSE  TRUE
```

01 Quiz. 변수 – 학습내용 되돌아보기

- **result_new 변수에 수정된 데이터 프레임 할당 및 출력**

```
result_new <- data.frame(age, job_new, marital_logical, balance, campaign, y)
print(result_new) #출력
```

```
> result_new <- data.frame(age, job_new, marital_logical, balance, campaign, y)
> print(result_new) #출력
  age job_new marital_logical balance campaign     y
1  30     무직            TRUE    1787   휴대폰 FALSE
2  33   서비스            TRUE    4789   휴대폰 FALSE
3  35   관리직           FALSE    1350   휴대폰 FALSE
4  30   관리직            TRUE    1476  Unknown FALSE
5  68     무직           FALSE    4189 유선전화  TRUE
6  33   관리직            TRUE    3935   휴대폰  TRUE
>
```

16차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

17차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- R 연산자를 알 수 있다.
- 산술 함수를 알 수 있다.
- 문자열 관련 함수를 알 수 있다.

학습 목차

- R 연산자
- 산술 함수
- 문자열 관련 함수

01.

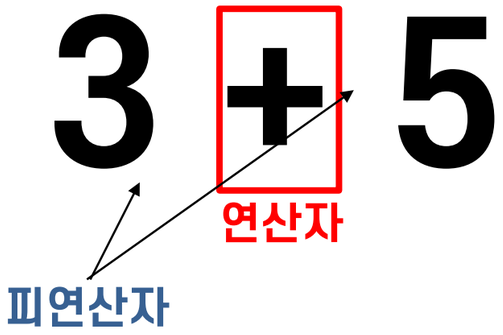
R 연산자의 이해

01 R 연산자의 이해 – 연산자란?

- 연산
  - ➢ 프로그램에서 데이터를 처리하여 결과를 산출하는 과정

- 연산자
  - ➢ 연산을 수행하기 위해 사용되는 문자

- 피연산자
  - ➢ 연산을 당하는 대상
  - ➢ R에서는 객체에 해당



01 R 연산자의 이해 – 산술 연산자

- 프로그램이 산술적인 연산을 하도록 하는 연산자

- 산술 연산자 종류
  - 7 가지

```
a <- 3+5          # '+'  더하기 연산자
b <- 3-5          # '-'  빼기 연산자
c <- 3*5          # '*'  곱하기 연산자
d <- 3/5          # '/'  나누기 연산자
e <- 3^5          # '^'  제곱 연산자
f <- 3%%5         # '%%' 나머지 연산자
g <- 3%/%5        # '%/%' 몫 연산자 (나머지 버리는 나누기)
```

01 R 연산자의 이해 – 산술 연산자

```
a <- 3+5          # '+'  더하기 연산자
b <- 3-5          # '-'   빼기 연산자
c <- 3*5          # '*'   곱하기 연산자
d <- 3/5          # '/'   나누기 연산자
e <- 3^5          # '^'   제곱 연산자
f <- 3%%5         # '%%'  나머지 연산자
g <- 3%/%5        # '%/%' 몫 연산자 (나머지 버리는 나누기)
```

- **[Quiz] 위의 코드를 실습 해본 후 아래 질문에 대답하시오.**
  - **Q1. 32^7의 값은?**
  - **Q2. 10%%3, 10%/%3의 값은?**
  - **Q3. 346434%%3343의 값은?**
  - **Q4. 346434%/%3343 의 값은?**

01 R 연산자의 이해 – 산술 연산자

```
a <- 3+5          # '+'  더하기 연산자
b <- 3-5          # '-'  빼기 연산자
c <- 3*5          # '*'  곱하기 연산자
d <- 3/5          # '/'  나누기 연산자
e <- 3^5          # '^'  제곱 연산자
f <- 3%%5         # '%%' 나머지 연산자
g <- 3%/%5        # '%/%' 몫 연산자 (나머지 버리는 나누기)
```

- **[Quiz] 위의 코드를 실습 해본 후 아래 질문에 대답하시오.**
  - **Q1. 32^7의 값은?**
  - **Q2. 10%%3, 10%/%3의 값은?**
  - **Q3. 346434%%3343의 값은?**
  - **Q4. 346434%/%3343 의 값은?**
  - **A1. 94,931,877,133**
  - **A2. 1, 3**
  - **A3. 2105**
  - **A4. 103**

01 R 연산자의 이해 – 비교 연산자

- 두 데이터 사이의 크기 비교를 수행하는 연산자
  - 결과는 논리형으로 리턴 됨

- 비교 연산자 종류
  - 6 가지

```
a <- (3 >5)          # ‘a > b’ a는 b보다 큰가?
b <- (3 >= 3)        # ‘a >= b’ a는 b보다 크거나 같은가?
c <- (3 < 5)         # ‘a < b’ , a는 b보다 작은가?
d <- (3 <= 5)        # ‘a <= b’ a는 b보다 작거나 같은가?
e <- (3 == 4)        # ‘a == b’ a는 b와 같은가?
f <- (3 != 4)        # ‘a != b’  a는 b와 다른가?
```

01 R 연산자의 이해 – 비교 연산자

```
a <- (3 >5)          # 'a > b' a는 b보다 큰가?
b <- (3 >= 3)        # 'a >= b' a는 b보다 크거나 같은가?
c <- (3 < 5)         # 'a < b' , a는 b보다 작은가?
d <- (3 <= 5)        # 'a <= b' a는 b보다 작거나 같은가?
e <- (3 == 4)        # 'a == b' a는 b와 같은가?
f <- (3 != 4)        # 'a != b'  a는 b와 다른가?
```

- **[Quiz] 위의 코드를 실습 해본 후 아래 질문에 대답하시오.**
  - **Q1. 위의 결과 값들을 하나의 벡터에 담아 출력하시오.**

```
vec <- ?
print(vec)
```

  - **Q2. '10110001011101011 0 < 11001011000101011' 의 결과는?**

```
> 10110001011101011 0 < 11001011000101011
[1] FALSE
```

01 R 연산자의 이해 – 비교 연산자

```
a <- (3 >5)          # 'a > b' a는 b보다 큰가?
b <- (3 >= 3)        # 'a >= b' a는 b보다 크거나 같은가?
c <- (3 < 5)         # 'a < b' , a는 b보다 작은가?
d <- (3 <= 5)        # 'a <= b' a는 b보다 작거나 같은가?
e <- (3 == 4)        # 'a == b' a는 b와 같은가?
f <- (3 != 4)        # 'a != b'  a는 b와 다른가?
```

- **[Quiz] 위의 코드를 실습 해본 후 아래 질문에 대답하시오.**
  - **Q1. 위의 결과 값들을 하나의 벡터에 담아 출력하시오.**

```
vec <- c(a, b, c, d, e, f)
print(vec)
```

  - **Q2. '10110001011101011 0 < 11001011000101011' 의 결과는?**

```
> 10110001011101011 0 < 11001011000101011
[1] FALSE
```

A2. FALSE

01 R 연산자의 이해 – 논리 연산자

- 값 또는 자료구조 객체에 적용되는 논리기능 수행하는 연산자
  - 결과는 논리형으로 리턴 됨

- 논리 연산자 종류
  - 5 가지

```
a <- c(T, T) & c(F, T)      # '&' 벡터화 된 논리곱(AND)
b <- c(T, T) && c(F, T)     # '&&' 벡터화 되지 않은 논리곱 (첫번째 값만 비교)
c <- c(T, T) | c(F, T)      # '|' , 벡터화 된 논리합(OR)  ( | : Shift + ₩)
d <- c(T, T) || c(F, T)     # '||' 벡터화 되지 않은 논리합 (첫번째 값만 비교)
e <- !TRUE                  # '!' 논리 부정(NOT)
```

01 R 연산자의 이해 – 논리 연산자

```
a <- c(T, T) & c(F, T)      # '&' 벡터화 된 논리곱(AND)
b <- c(T, T) && c(F, T)     # '&&' 벡터화 되지 않은 논리곱 (첫번째 값만 비교)
c <- c(T, T) | c(F, T)      # '|' , 벡터화 된 논리합(OR)  ( | : Shift + ₩)
d <- c(T, T) || c(F, T)     # '||' 벡터화 되지 않은 논리합 (첫번째 값만 비교)
e <- !TRUE                  # '!' 논리 부정(NOT)
```

- **[Quiz] 위의 코드를 실습 해본 후 아래 질문에 대답하시오.**
  - **Q1. 위의 결과 값들을 하나의 리스트에 담아 출력하시오**

```
myList <-  ?
print(myList)
```

01 R 연산자의 이해 – 논리 연산자

```
a <- c(T, T) & c(F, T)      # '&' 벡터화 된 논리곱(AND)
b <- c(T, T) && c(F, T)     # '&&' 벡터화 되지 않은 논리곱 (첫번째 값만 비교)
c <- c(T, T) | c(F, T)      # '|' , 벡터화 된 논리합(OR)  ( | : Shift + ₩)
d <- c(T, T) || c(F, T)     # '||' 벡터화 되지 않은 논리합 (첫번째 값만 비교)
e <- !TRUE                  # '!' 논리 부정(NOT)
```

- **[Quiz] 위의 코드를 실습 해본 후 아래 질문에 대답하시오.**
  - **Q1. 위의 결과 값들을 하나의 리스트에 담아 출력하시오**

```
myList <- list(a, b, c, d, e)
print(myList)
```

02.

산술 함수

02 산술 함수 – Min, Max, Sum

- **Min 함수**
  - ➢ **입력 벡터 중 최소값 출력**

```
min( c(1, 2, 3, 4, 5))          #1
```

- **Max 함수**
  - ➢ **입력 벡터 중 최대값 출력**

```
max( c(1, 2, 3, 4, 5))          #5
```

- **Sum 함수**
  - ➢ **입력 벡터 중 총합 출력**

```
sum( c(1, 2, 3, 4, 5))          #15
```

02 산술 함수 – Prod, Factorial, Abs

- **Prod 함수**
  - ➢ **입력 벡터 중 곱셈 값 출력**

```
prod( c(1, 2, 3, 4, 5))          #120
```

- **Factorial 함수**
  - ➢ **1부터 입력 값 까지의 팩토리얼 값 출력**

```
factorial( 4 )                   #4! = 24
factorial( c(1 2 -3 4) )         # 1 2 NaN 24
```

- **Abs 함수**
  - ➢ **입력 값의 절대값 출력**

```
abs(-3)                          # 3
abs( c(1, -2, 3, -4, 5))         #1 2 3 4 5
```

02 산술 함수 – Mean, Median

- **Mean 함수**
  - ➢ **입력 벡터의 평균 값 출력**

```
mean( c(1, 2, 5, 8, 10))            #5.2
```

- **Median 함수**
  - ➢ **입력 벡터의 중간 값 출력**

```
median( c(1, 2, 5, 8, 10) )          #5
```

02 산술 함수 – Range, Var, Sd

- **Range 함수**
  - ➢ 입력 벡터의 최소값 최대값 출력

```
range( c(1, 2, 3, 4, 5))          #1 5
range( c(10, 2, 4, 4, 77))        #2 77
```

- **Var 함수**
  - ➢ 입력 벡터의 평균에 대한 분산값 출력

```
var(c(1, 2, 5, 8, 10))            #14.7
```

- **Sd 함수**
  - ➢ 입력 벡터의 평균에 대한 표준편차값 출력

```
sd(c(1, 2, 5, 8, 10))             #3.834058
```

03.

문자열 관련 함수

03 문자열 관련 함수 – Paste, rep

- **Paste 함수**
  - ➢ **값들을 하나의 문자열로 합치는 함수**

```
paste(‘안녕’, ‘하세’, ‘요’)                # ‘안녕 하세 요’
paste(‘안녕’, ‘하세’, ‘요’, sep=“”)        # ‘안녕하세요’
paste(‘안녕’, ‘하세’, ‘요’, sep=“/”)       # ‘안녕/하세/요’
paste(c(1, 2, 3))                          # c(‘1', ‘2', ‘3') , 벡터 값들을 개별로 인식
paste(c(1,2,3), c(‘번’, ‘번’, ‘번’))       # c(‘1 번‘, ‘2 번’, ‘3 번’)
```

- **Rep 함수**
  - ➢ **rep(data, n): data로 초기화되는 길이가 n인 벡터 생성**

```
rep(1, 5)          # c(1, 1, 1, 1, 1)
rep(‘a’, 5)        # c(‘a’, ‘a’, ‘a’, ‘a’, ‘a’)
```

03 문자열 관련 함수 – Paste, rep

- **Paste&Rep 함수 활용**

```
index <- c(1,2,3)
len <- length(index)                # index 벡터의 크기 리턴
paste( index, rep('번', len) )      # c('1 번', '2 번', '3 번')
```

04. 실습

04 실습

- **데이터셋 설명**
  - ➢ **항공사별 2021년 9월 김포-제주 편 비행기 실적**

항공사 명	운항(편)	탑승객(수)	화물(톤)
아시아나항공	1,575	249,792	3,097.9
에어부산	481	90,985	516.7
에어프레미아	124	29,238	111.1
에어서울	354	71,213	273.1
제주항공	1,197	203,335	847.1
진에어	793	133,253	763.2
대한항공	1,670	250,895	5,406.1
티웨이항공	859	146,497	597.6

데이터셋 소스:
https://www.airport.co.kr/www/cms/frFlightStatsCon/domesticLineStats.do?MENU_ID=1250#none

04 실습

항공사 명	운항(편)	탑승객(수)	화물(톤)
아시아나항공	1,575	249,792	3,097.9
에어부산	481	90,985	516.7
에어프레미아	124	29,238	111.1
에어서울	354	71,213	273.1
제주항공	1,197	203,335	847.1
진에어	793	133,253	763.2
대한항공	1,670	250,895	5,406.1
티웨이항공	859	146,497	597.6

Q.1 운항/탑승객/화물 별 '총 실적' 및 '평균 실적'을 구하세요.

04 실습

항공사 명	운항(편)	탑승객(수)	화물(톤)
아시아나항공	1,575	249,792	3,097.9
에어부산	481	90,985	516.7
에어프레미아	124	29,238	111.1
에어서울	354	71,213	273.1
제주항공	1,197	203,335	847.1
진에어	793	133,253	763.2
대한항공	1,670	250,895	5,406.1
티웨이항공	859	146,497	597.6

Q.1 운항/탑승객/화물 별 '총 실적' 및 '평균 실적'을 구하세요.

```
#데이터 입력
airline <- c('아시아나항공', '에어부산', '에어프레미아', '에어서울', '제주항공', '진에어', '대한항공', '티웨이항공')
flight <- c(1575, 481, 124, 354, 1197, 793, 1670, 859)
passenger <- c(249792, 90985, 29238, 71213, 203335, 133253, 250895, 146497)
freight <- c(3097.9, 516.7, 111.1, 273.1, 847.1, 763.2, 5406.1, 597.6)

#계산 및 출력
total <- c(sum(flight), sum(passenger), sum(freight))
average <- total / length(flight)                    # length(): 입력된 매개변수의 크기 리턴
print(total)
print(average)
```

04 실습

Q.2 운항이 평균 실적보다 높은 항공사들을 벡터 형태로 출력하세요.

04 실습

Q.2 운항이 평균 실적보다 높은 항공사들을 벡터 형태로 출력하세요.

```
#데이터 입력
airline <- c('아시아나항공', '에어부산', '에어프레미아', '에어서울', '제주항공', '진에어', '대한항공', '티웨이항공')
flight <- c(1575, 481, 124, 354, 1197, 793, 1670, 859)
passenger <- c(249792, 90985, 29238, 71213, 203335, 133253, 250895, 146497)
freight <- c(3097.9, 516.7, 111.1, 273.1, 847.1, 763.2, 5406.1, 597.6)

#계산 및 출력
upperAvg_flight <- flight > mean(flight)  #평균 운항 실적과 항공사별 운항실적 비교
print(upperAvg_flight)                    # c(TRUE, FALSE, FALSE, FALSE, TRUE, FALSE, TRUE, FALSE)
result <- airline[upperAvg_flight]        # airline 벡터 중 TRUE에 해당하는 index 값만 남김
print(result)                             # c('아시아나항공', '제주항공', '대한항공')
```

airline	아시아나항공	에어부산	에어프레미아	에어서울	제주항공	진에어	대한항공	티웨이항공
upperAvg_flight	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE
airline[upperAvg_flight]	아시아나항공				제주항공		대한항공	

이번 차시 정리

R 연산자의 이해

- 산술 연산자
- 비교 연산자
- 논리 연산자

산술 함수 소개

- min, max, range,
- sum, prod, fractical, abs,
- mean, median, var, sd

이번 차시 정리

문자열 관련 함수 소개

- paste
- rep

17차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

18차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 조건문에 대해 알 수 있다.
- 조건문을 이해 할 수 있다.

학습 목차

- 조건문이란?
- 조건문의 이해

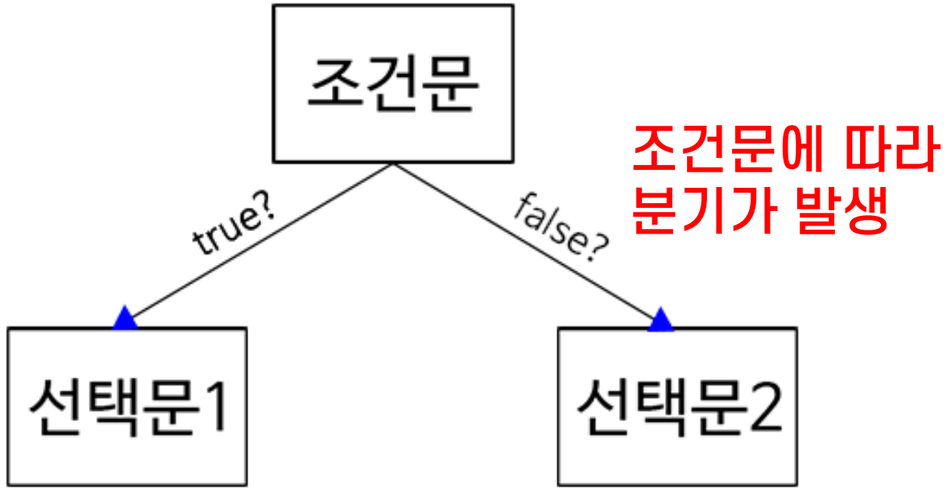
01.

조건문이란?

01 조건문이란?

- 지정한 조건에 해당할 때, 코드를 수행하는 구문

- 조건문 안에 들어가는 값은 반드시 논리값 임



조건문에 따라
분기가 발생

02.
조건문의 이해

02 조건문의 이해 – if ~ else 문

- **문법**

```
if( 조건식 ){
   실행문1        # 조건식이 참일 경우 실행
}
else {
   실행문2        # 조건식이 거짓일 경우 실행
}
```

02 조건문의 이해 – if ~ else 문

- 다중 조건식 사용 가능
  - Else if 구문 사용

```
if( 조건식1 ){
   실행문1
}
else if (조건식2){
   실행문2
}
else if (조건식3){
   실행문3
}….
else{
   실행문4
}
```

02 조건문의 이해 – if ~ else 문

- **예시**

```
score <- scan()   # scan(): 사용자로부터 데이터 입력 받는 함수. 값 입력없이 엔터 누를 시, 입력 종료
passScore <- 50
if( score > passScore ){
   print("합격")
}
else {
   print("불합격")
}
```

02 조건문의 이해 – if ~ else 문

- **다중 조건식 예시**

```
score <- scan()          #1개의 1~100사이 정수 값 만 입력
if( score >= 90 ){
   grade <- 'A'
} else if (score >= 70) {
   grade <- 'B'
} else if (score >= 50) {
   grade <- 'C'
} else {
   grade <- 'F'
}
print( paste("당신의 학점은 :", grade, "입니다"))
```

02 조건문의 이해 – ifelse 문 (삼항연산자)

- **문법**

```
ifelse(조건식, ‘참’인 경우 실행문, ‘거짓’인 경우 실행문)
```

- **예시**

```
score <- c(30, 90, 75, 82)
result <- ifelse( score >= 50, “합격”, “불합격” )
print(result)  # c(“불합격”, “합격”, “합격”, “합격”)
```

- **특징**
  - **벡터 연산 가능**
  - **간단한 조건문에 한하여 가독성 목적으로 사용**
  - **실행문이 복잡한 경우에 적합하지 않음**
  - **다중 조건식에 부적합**

02 조건문의 이해 - switch 문

- **문법**

```
switch(입력 값, 비교값1=실행문1, 비교값2=실행문2, … , 기본 실행문)
```

  - **입력 값 과 비교 값이 같다면 해당하는 실행문 실행**
    - **if(입력 값 == 비교 값){ 실행문 }**
  - **기본 실행문은 입력 값과 같은 비교 값이 없을 때 실행**

- **예시**

```
x <- "age"
z <- “name”
result1 <-switch(x, ‘word’=“hi”, ‘age’ =25, ‘3’=“숫자”, NULL)   # 25
result2 <-switch(y, 'word'="hi", 'age' =25, '3'="숫자", NULL)   # NULL
```

02 조건문의 이해 - switch 문

- 특징
  - ➢ 비교 값은 문자열 형식만 사용가능

```
> switch(x, '1'="hi", 2 ="hello", c="안녕", "몰라")
Error: unexpected '=' in "switch(x, '1'="hi", 2 ="
```

  - ➢ 범주형 데이터 처리해야하는 경우 효율적

02 조건문의 이해 – which 문

- **문법**

```
which( 조건문)
```

➢ **조건에 해당하는 입력 벡터 내부 값의 인덱스(위치) 출력**

- **예시**

```
score <- c(30, 90, 75, 82)
result1 <- which(score == 90)  # 2
result2 <- which(score >= 50)  # c(2, 3, 4)
```

02 조건문의 이해 – which 문

- 특징
  - ➢ 조건에 해당하는 입력 벡터 내부 값의 인덱스(위치) 출력

```
score <- c(30, 90, 75, 82)
result2 <- which(score >= 50)  # c(2, 3, 4)
score[ result2 ]               # c(90, 75, 82)
```

  - ➢ 조건에 따른 실행문 실행 기능 없음

03.

실습

03 실습 - 조건문 실습 1

- Q1. 온도(숫자) 값을 입력 받아 30도 이상이면 “더움”, 20~30도 사이면 “보통”, 10~20도 사이면 “선선”, 0~10도 사이면 “추움”, 0도 이하면 “매우 추움” 을 출력하는 코드를 작성하세요.
  - if ~ else 문을 사용하세요

03 실습 - 조건문 실습 1

- **Q1. 온도(숫자) 값을 입력 받아 30도 이상이면 “더움”, 20~30도 사이면 “보통”, 10~20도 사이면 “선선”, 0~10도 사이면 “추움”, 0도 이하면 “매우 추움” 을 출력하는 코드를 작성하세요.**
  - **if ~ else 문을 사용하세요**

```
t <- scan()
if( t >= 30){
   print(“더움”)
} else if ( t >= 20 ){
   print(“보통”)
} else if (t >= 10 ){
   print(“선선”)
} else if (t >= 0 ){
   print(“추움”)
} else {
   print(“매우 추움”)
}
```

03 실습 - 조건문 실습 2

- Q2. 식사량 설문조사를 진행하기 위하여, 범주형 값을 입력 받을 때마다 수치 값으로 변환하는 프로그램을 작성하려 한다. 아래 질문에 답하시오.
  - “매우 많이”는 100, “많이”는 70, “보통”은 50, “조금”은 30, 그 외는 0으로 수정하는 조건문을 작성하세요.
  - data 변수에 범주형 값이 들어있다고 가정하세요.
  - if~else 문을 사용하세요.

```
data <- “매우 많이” # 가정
```

?

03 실습 - 조건문 실습 2

- **Q2. 식사량 설문조사를 진행하기 위하여, 범주형 값을 입력 받을 때마다 수치 값으로 변환하는 프로그램을 작성하려 한다. 아래 질문에 답하시오.**

```
if(data == "매우 많이"){
  result <- 100
 } else if (data == "많이"){
  result <- 70
 } else if (data == "보통"){
  result <- 50
 } else if (data == "조금"){
  result <- 30
 } else {
  result <- 0
 }
```

03 실습 - 조건문 실습 3

- Q3. 식사량 설문조사를 진행하기 위하여, 범주형 값을 입력 받을 때마다 수치 값으로 변환하는 프로그램을 작성하려 한다. 아래 질문에 답하시오.
  - “매우 많이”는 100, “많이”는 70, “보통”은 50, “조금”은 30, 그 외는 0으로 수정하는 조건문을 작성하세요.
  - data 변수에 범주형 값이 들어있다고 가정하세요.
  - switch 문을 사용하세요.

```
data <- “매우 많이” # 가정
```

?

03 실습 - 조건문 실습 3

- Q3. 식사량 설문조사를 진행하기 위하여, 범주형 값을 입력 받을 때마다 수치 값으로 변환하는 프로그램을 작성하려 한다. 아래 질문에 답하시오.
  - “매우 많이”는 100, “많이”는 70, “보통”은 50, “조금”은 30, 그 외는 0으로 수정하는 조건문을 작성하세요.
  - data 변수에 범주형 값이 들어있다고 가정하세요.
  - switch 문을 사용하세요.

```
data <- “매우 많이” # 가정
result <- switch(data, “매우 많이”=100, “많이”=70, “보통”=50, “조금”=30, 0)
```

03 실습

- **데이터셋 설명**
  - ➢ **항공사별 2021년 9월 김포-제주 편 비행기 실적**

항공사 명	운항(편)	탑승객(수)	화물(톤)
아시아나항공	1,575	249,792	3,097.9
에어부산	481	90,985	516.7
에어프레미아	124	29,238	111.1
에어서울	354	71,213	273.1
제주항공	1,197	203,335	847.1
진에어	793	133,253	763.2
대한항공	1,670	250,895	5,406.1
티웨이항공	859	146,497	597.6

데이터셋 소스:
https://www.airport.co.kr/www/cms/frFlightStatsCon/domesticLineStats.do?MENU_ID=1250#none

03 실습

항공사 명	운항(편)	탑승객(수)	화물(톤)
아시아나항공	1,575	249,792	3,097.9
에어부산	481	90,985	516.7
에어프레미아	124	29,238	111.1
에어서울	354	71,213	273.1
제주항공	1,197	203,335	847.1
진에어	793	133,253	763.2
대한항공	1,670	250,895	5,406.1
티웨이항공	859	146,497	597.6

- **Q4. 운항 횟수가 1000회 이상이면 '우수', 그 이하면 '보통' 값을 가지는 벡터를 생성하세요.**

```
flight <- c(1575, 481, 124, 354, 1197, 793, 1670, 859) # 데이터 입력
```

?

03 실습

항공사 명	운항(편)	탑승객(수)	화물(톤)
아시아나항공	1,575	249,792	3,097.9
에어부산	481	90,985	516.7
에어프레미아	124	29,238	111.1
에어서울	354	71,213	273.1
제주항공	1,197	203,335	847.1
진에어	793	133,253	763.2
대한항공	1,670	250,895	5,406.1
티웨이항공	859	146,497	597.6

- **Q4. 운항 횟수가 1000회 이상이면 '우수', 그 이하면 '보통' 값을 가지는 벡터를 생성하세요.**

```
flight <- c(1575, 481, 124, 354, 1197, 793, 1670, 859) # 데이터 입력
result <- ifelse(flight > 1000, “우수”, “보통”)
print(result)
```

03 실습

- **Q5. 운항이 평균 실적보다 높은 항공사들을 벡터 형태로 출력하세요.**
  - **which 문을 사용하세요.**

```
#데이터 입력
airline <- c('아시아나항공', '에어부산', '에어프레미아', '에어서울', '제주항공', '진에어', '대한항공', '티웨이항공')
flight <- c(1575, 481, 124, 354, 1197, 793, 1670, 859)

#계산 및 출력
```

?

  - **17차시 실습 코드와 비교해보세요.**

03 실습

- **Q5. 운항이 평균 실적보다 높은 항공사들을 벡터 형태로 출력하세요.**
  - **which 문을 사용하세요.**

```
#데이터 입력
airline <- c('아시아나항공', '에어부산', '에어프레미아', '에어서울', '제주항공', '진에어', '대한항공', '티웨이항공')
flight <- c(1575, 481, 124, 354, 1197, 793, 1670, 859)

#계산 및 출력
index <- which(flight > mean(flight))      # c(1, 5, 7)
result <- airline[index]
print(result)                              # c('아시아나항공', '제주항공', '대한항공')
```

  - **17차시 실습 코드와 비교해보세요.**

이번 차시 정리

조건문 이란?

- 지정한 조건에 해당할 때, 코드를 수행하는 구문

R의 조건문 문법 및 특징 이해

- if ~ else 문
- ifelse 문
- switch 문
- which 문

18차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

19차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 반복문에 대해 알 수 있다.
- 반목문을 이해할 수 있다.
- 벡터를 다룰 수 있다.

학습 목차

- 반복문이란?
- 반복문의 이해
- 벡터 다루기

01.

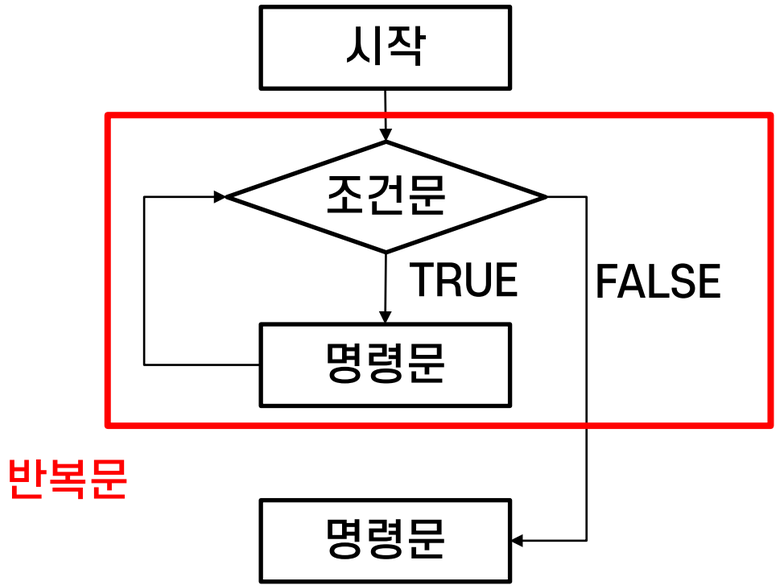
반복문이란?

01 반복문이란?

- 프로그램 내에서 똑같은 명령을 일정 횟수만큼 반복하여 수행하도록 제어하는 명령문

- 조건문과 더불어 가장 많이 사용되는 제어문

- 대부분 조건문과 함께 사용됨



02.
반복문의 이해

02 반복문의 이해 – For 문

- **문법**

```
for( 변수 in 데이터){
    #(변수 사용한) 명령문
}
```

➢ '데이터'에 존재하는 값이 순서대로 '변수'에 할당되어 명령문 실행

➢ '데이터'의 마지막 값까지 반복

- **예시**

```
for( i in c(1,2,3)){
    print(i)
}
```

```
> for( i in c(1,2,3)){
+   print(i)
+ }
[1] 1
[1] 2
[1] 3
```

02 반복문의 이해 – For 문 (예제)

- **예제 1**

```
# sum 직접 구현
data <- c(32, 45, 21, 10, 43)
result <- 0
for( i in data){
    result <- result + i
}
print(result)                                #151
```

- **예제 2**

```
# max 직접 구현
data <- c(32, 45, 21, 10, 43)
max <- 0
for( i in data){
    max <- ifelse( i > max, i, max)
}
print(max)
```

02 반복문의 이해 – While 문

- **문법**

```
while( 조건식){
   #조건이 참인 경우 수행할 명령문
}
```

➢ **조건식이 FALSE라면 반복 종료**

➢ **산술 값 계산 or 위치(index) 값을 사용할 때 주로 사용**

- **예시**

```
i <- 1
while( i < 100){
   i <- 2 * i          # 보통 명령문 안에서 조건식의 변수 값을 조정하여 반복 횟수 조절
   print(i)
}
```

02 반복문의 이해 – While 문 (예제)

- **예제 1**

```
# 무한루프
while( 1 ){
    i <- 2 * i
    if( i > 100) {          # 명령문 안에서 반복 종료 조건 설정
       break                #break : 반복문 안에서 사용시 강제로 반복 종료
    }
}
```

02 반복문의 이해 – While 문 (예제)

- **예제 2**

```
# factorial 직접구현
value <- 5                    # factorial(5) = 1 * 2 * 3 * 4 * 5
result <- 1
i <- 1
while( i <= value){
   result <- i * result       # 보통 명령문 안에서 조건식의 변수 값을 조정하여 반복 횟수 조절
   i <- i +1
}
print(result)                 # 120
```

02 반복문의 이해 - Repeat 문

- **문법**

```
repeat{
    #명령문 (무한반복이 기본)
}
```

- ➢ **가장 간단한 반복문 구조**
- ➢ **잘 사용하지 않음**

02 반복문의 이해 – Repeat 문

- **예시**

```
i <- 1
repeat {
   print( i )       # 반복 내용
   if( i > 9 ) {    # 반복 종료 조건
      break
   }
   i <- i + 1     # 변수 조정
}
```

03.

벡터 다루기

03 벡터 다루기 – 참조, 할당

- 위치(index)값 사용하여 벡터 내부 값 참조/할당 가능
  - ➢ 위치 값은 1부터 시작
  - ➢ 문법: 벡터명[위치값]

Data :	32	45	21	10	43
index	1	2	3	4	5

```
data <- c(32, 45, 21, 10, 43)
data[1]                    #32
data[3]                    #21
data[1] <- 50              # data 벡터 1번 위치에 50 할당
data[1]                    #50
```

03 벡터 다루기 – 참조, 할당

- **다중 위치 참조 가능**

```
data <- c(32, 45, 21, 10, 43)
logic <- c(T, T, F, F, T)
num <- c(1,4,5)
data[logic]                    # c(32, 45, 43) , logic 벡터 값이 TRUE 인 index 값만 리턴
data[num]                      # c(32, 10, 43) , num 벡터 값에 해당하는 index 값만 리턴
```

- **음수 위치값 사용 가능**

```
data <- c(32, 45, 21, 10, 43)
data[ -3 ]                     # c(32, 45, 10, 43) , 해당 자릿수 값만 빼고 리턴
data[ c(-1, -4) ]              # c(34, 21, 43)
```

03 벡터 다루기 – 반복문&벡터

- **반복문 사용하여 벡터 특정 위치 값 참조**

```
 i <- 1
data <- c(32, 45, 21, 10, 43)
while(i < length(data)){
  if( i%%2 == 0 ){                                #짝수 번 째 위치 값만 참조
    print( paste("자리수:", i, "값:", data[i]) )
  }
  i <- i + 1
}
```

03 벡터 다루기 – 반복문&벡터

- **반복문 사용하여 벡터 값 채우기**

```
i <- 1
vec <- c()          #빈 벡터 생성
while(i < 10){
    vec <- c(vec, i)
    i <- i + 1
}
print(vec)          # c(1, 2, 3, 4, 5, 6, 7, 8, 9)
```

04. 실습

04 실습

- **Q1. min함수를 반복문 사용하여 직접 구현해 보세요**

```
# min 직접 구현
data <- c(32, 45, 21, 10, 43)
```

?

04 실습

- **Q1. min함수를 반복문 사용하여 직접 구현해 보세요**

```
# min 직접 구현
data <- c(32, 45, 21, 10, 43)
min <- 99999999                    #충분히 큰 값으로 초기화
 for( i in data){
     min <- ifelse( i < min, i, min)
 }
print(min)
```

04 실습

- **Q2. 숫자형 벡터 데이터가 주어졌을 때, 각 값이 2의 배수라면 TRUE를, 아니라면 FALSE를 출력하는 프로그램을 구현해보세요**

```
data <- c(32, 45, 21, 10, 43)
```

?

04 실습

- **Q2. 숫자형 벡터 데이터가 주어졌을 때, 각 값이 2의 배수라면 TRUE를, 아니라면 FALSE를 출력하는 프로그램을 구현해보세요**

```
data <- c(32, 45, 21, 10, 43)
for( i in data){
    if( i%%2 ==0 ) { #2의 배수라면
        print(TRUE)
    } else {
        print(FALSE)
    }
 }
```

04 실습

- **Q3. 숫자형 벡터 데이터가 주어졌을 때, 홀수 자리 값의 합을 출력하는 프로그램을 구현해보세요**

```
# 홀수번째 값 sum 구현
data <- c(32, 45, 21, 10, 43)
sum <- 0                          # 홀수 자리 합 저장할 변수 초기화
```

?

04 실습

- **Q3. 숫자형 벡터 데이터가 주어졌을 때, 홀수 자리 값의 합을 출력하는 프로그램을 구현해보세요**

```
# 홀수번째 값 sum 구현
data <- c(32, 45, 21, 10, 43)
sum <- 0                          # 홀수 자리 합 저장할 변수 초기화
i <- 1                           # 데이터 인덱스 나타냄
while( i <= length(data)){
    if( i %% 2 == 1 ) {          # 홀수 라면
        sum <- sum + data[i]
    }
    i <- i+1
}
```

04 실습

- **데이터셋 설명**
  - ➢ **항공사별 2021년 9월 김포-제주 편 비행기 실적**

항공사 명	운항(편)	탑승객(수)	화물(톤)
아시아나항공	1,575	249,792	3,097.9
에어부산	481	90,985	516.7
에어프레미아	124	29,238	111.1
에어서울	354	71,213	273.1
제주항공	1,197	203,335	847.1
진에어	793	133,253	763.2
대한항공	1,670	250,895	5,406.1
티웨이항공	859	146,497	597.6

데이터셋 소스:
https://www.airport.co.kr/www/cms/frFlightStatsCon/domesticLineStats.do?MENU_ID=1250#none

04 실습

- Q4. 운항 횟수가 1000회 이상이면 '우수', 그 이하면 '보통' 값을 가지는 벡터를 생성하세요.
  - while문을 사용하세요
  - 결과를 담을 빈 벡터를 미리 생성해놓고 반복문을 돌며 값을 채워넣는 방식으로 코드를 구현하세요

04 실습

```
#데이터 입력
airline <- c('아시아나항공', '에어부산', '에어프레미아', '에어서울', '제주항공', '진에어', '대한항공', '티웨이항공')
flight <- c(1575, 481, 124, 354, 1197, 793, 1670, 859)

result <- c()  #결과 담을 빈 벡터

i <- 1
while(i <=  ?  ) ){          # 데이터 모든 값 한번씩 참조
  if(  ?  > mean(flight)){    # i 번째 항공사의 운항 실적이 평균보다 좋다면
    result <-  ?              # result 벡터에 i 번째 항공사 명 추가
  }
  i <- i + 1
}
print(result)   # c("아시아나항공“, "제주항공“, "대한항공")
```

04 실습

```
#데이터 입력
airline <- c('아시아나항공', '에어부산', '에어프레미아', '에어서울', '제주항공', '진에어', '대한항공', '티웨이항공')
flight <- c(1575, 481, 124, 354, 1197, 793, 1670, 859)

result <- c()  #결과 담을 빈 벡터

i <- 1
while(i <= length(airline) ){          # 데이터 모든 값 한번씩 참조
  if(flight[i] > mean(flight)){         # i 번째 항공사의 운항 실적이 평균보다 좋다면
    result <- c(result, airline[i])      # result 벡터에 i 번째 항공사 명 추가
  }
  i <- i + 1
}
print(result)   # c("아시아나항공“, "제주항공“, "대한항공")
```

- **17, 18차시 실습 코드와 비교해보세요.**

이번 차시 정리

반복문 이란?

- 지정한 조건에 해당할 때, 코드를 수행하는 구문

R의 반복문 문법 및 특징 이해

- for 문
- while 문
- repeat 문

19차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

20차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 사용자 정의 함수 문법에 대해 알 수 있다.

학습 목차

- 사용자 정의 함수 문법
- 실습

01.
사용자 정의 함수

01 사용자 정의 함수 – 문법

- **사용자가 직접 정의하여 만들어내는 함수**
- **문법**

```
함수명 <- function(매개변수1, 매개변수2=초기 값, ...) {
    #실행문
    return(리턴 값)        #생략 가능
}
```

01 사용자 정의 함수 – 문법

- **예시**

```
myFunction <- function(){
    print("Hello!")
}
myFunction()
```

```
mySum <- function(x, y=0) { # 초기(default) 값
    result = x + y
    return(result)
}
sum <- mySum (5, 10)
sum <- mySum(5)          # 선택적 매개변수 제외
```

01 사용자 정의 함수 – 예제

- **예시 1**

```
# 평균, 표준편차, min, max 계산 함수
getStatics <- function( x ) {
    x.mean <- mean(x)
    x.sd <- sd(x)
    x.min <- min(x)
    x.max <- max(x)
    x.summary <- c("mean"=x.mean, "sd"=x.sd, "min"=x.min, max=x.max)  #index별 이
름 설정가능
    return(x.summary)
}
statics <- getStatics(c(1,2,3,4,5))
print(statics)
```

- **벡터에 index별 이름 설정 가능**
  - ➢ **c(이름1=값, 이름2=값,...)**
  - ➢ **'이름'에 적힌 값은 문자열로 자동 형변환되어 취급**

```
     mean       sd      min      max
3.000000 1.581139 1.000000 5.000000
```

01 사용자 정의 함수 – 예제

- **예시 2**

```
# D-day 계산
D_Day <- function( year, month, day ) {
    today <- Sys.Date()                                  # Sys.Date() , 오늘의 날짜 리턴 함수 “yyyy-mm-dd” 형식
    inputDayString <- paste(year, month, day, sep=“-”)   #형변환을 위해 날짜 형식 문자열로 변경
    inputDay <- as.Date(inputDayString)                  # Date 형식으로 형변환
    Ddays <- difftime(today, inputDay)                   # difftime: Date간 차이에 해당하는 일수 리턴
return(as.integer(Ddays))
}
d_day <- D_Day (2022, 01, 01)
print(d_day)
```

- **s <- paste(2022, 1, 1, sep=“-”)**
- **as.Date(s)**

```
> s <- print(paste(2022, 1, 1, sep="-"))
[1] "2022-1-1"
> as.Date(s)
[1] "2022-01-01"
```

- **difftime(today, inputDay)**

```
> difftime(today, inputDay)
Time difference of -59 days
```

01 사용자 정의 함수 – 알아두면 좋은 특성 1

- **매개변수, 함수 안 변수 우선순위**

```
# 매개 변수와, 함수 내부변수 이름이 겹치는 경우
testFunction <- function( x ) {
  x <- 10
  print(x)
}
testFunction(5)                                    #10 , 함수 내부변수 우선
```

- **함수 안에서 사용한 변수 밖에서 사용 불가능**

```
testFunction <- function( x ) {
   y <- 10
   return(x+y)
}
testFunction(5)
print(y)                    # 에러발생. 함수 내부에서 선언한 변수는 밖에서 사용할 수 없음.
```

01 사용자 정의 함수 – 알아두면 좋은 특성 2

- **함수 안 함수 선언 가능**

```
testFunction <- function( x ) {
   square <- function(y) { return(y^y) }          # 매개변수명을 x로 하면 에러발생
   return(square(x))
 }
 testFunction(5)                                  #25
```

➢ **단, 동일한 매개변수 명 사용 불가**

01 사용자 정의 함수 – 알아두면 좋은 특성 2

- **함수를 리턴 가능**

```
makeFunction <- function( x ) {
  power <- function(y) { return(y^x) }
  return(power)
}
square <- makeFunction(2)
cube <- makeFunction(3)
square(5)                    #5^2 = 25
cube(5)                      #5^3 = 125
```

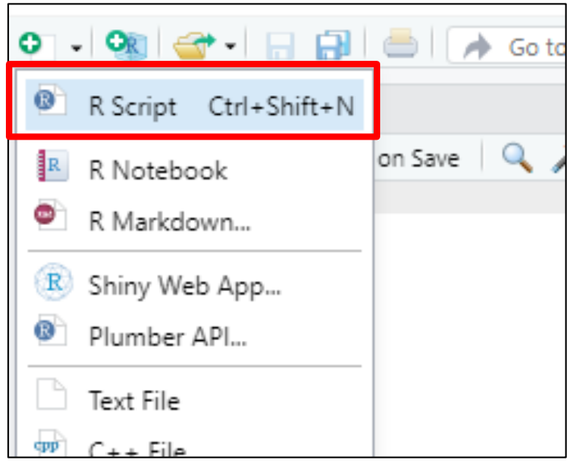
  - **함수를 생산하는 함수**

01 사용자 정의 함수 – 파일에 함수 저장하기

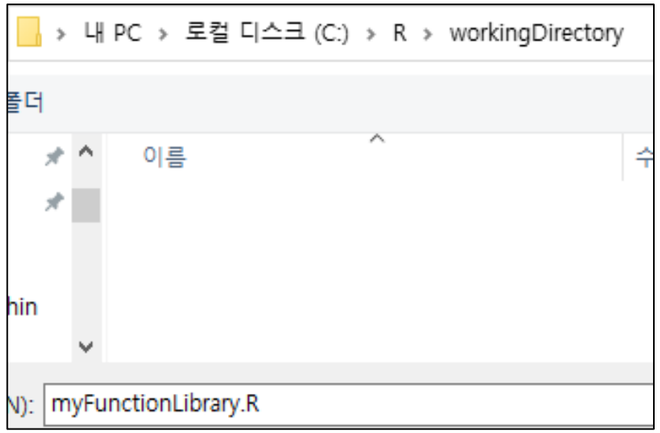
- 별도 파일에 함수 저장
  - 일반적으로 사용자 정의 함수 사용하는 방법
  - 패키지와 유사
  - 함수 외 변수, 데이터 셋 저장가능

01 사용자 정의 함수 – 파일에 함수 저장하기

- 파일 저장 방법
  - ➢ 1. 새 r 스크립트 생성
  - ➢ 2. 함수 작성
  - ➢ 3. r 스크립트 working directory에 저장
    - getwd()

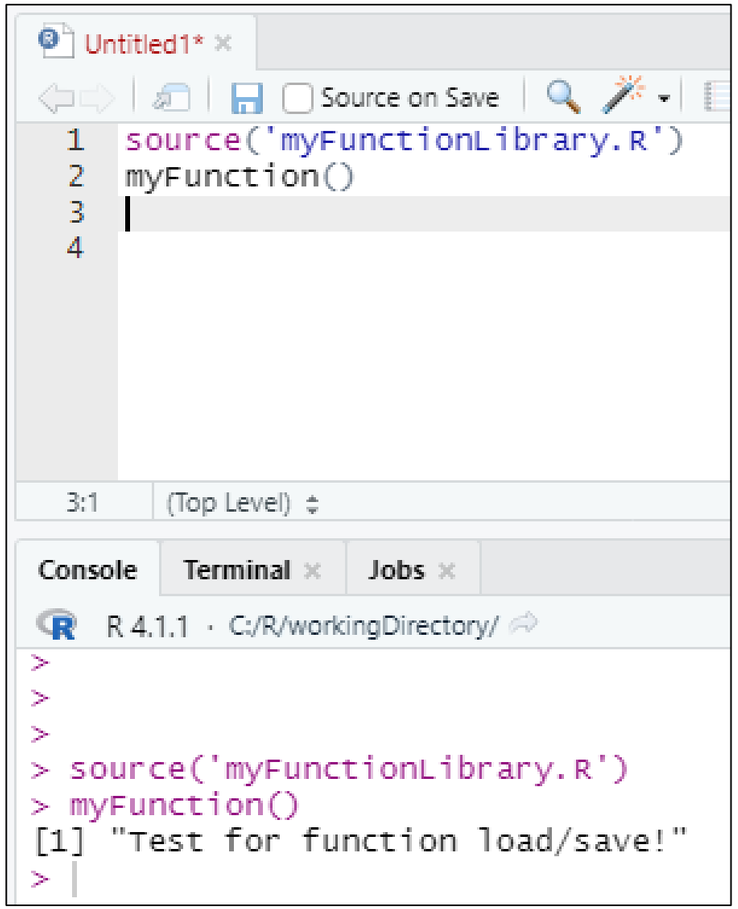



```
myFunction <- function(){
  print("Test for function load/save!")
}
```



01 사용자 정의 함수 – 파일 불러오기

- 1. 새로운 r 스크립트 생성
- 2. source 함수 사용하여 파일 불러오기
  - ➢ source()
- 3. 파일 내부에 저장된 함수 사용



02.
실습

02 실습

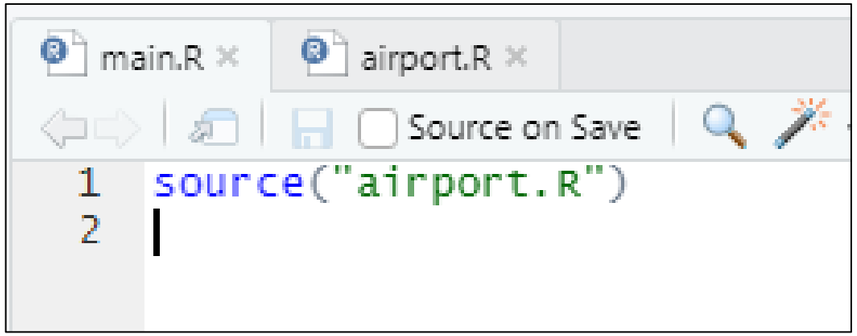
- **데이터셋 설명**
  - ➢ **항공사별 2021년 9월 김포-제주 편 비행기 실적**

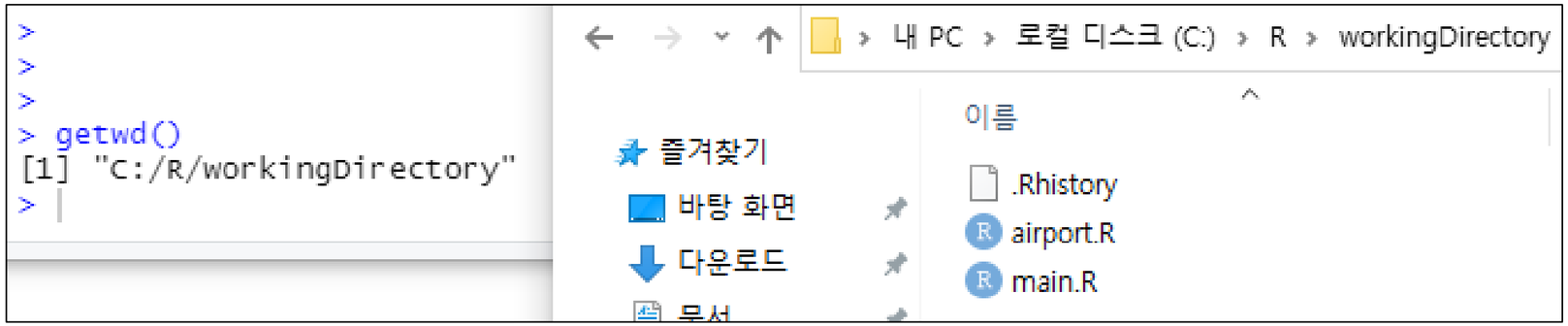
항공사 명	운항(편)	탑승객(수)	화물(톤)
아시아나항공	1,575	249,792	3,097.9
에어부산	481	90,985	516.7
에어프레미아	124	29,238	111.1
에어서울	354	71,213	273.1
제주항공	1,197	203,335	847.1
진에어	793	133,253	763.2
대한항공	1,670	250,895	5,406.1
티웨이항공	859	146,497	597.6

데이터셋 소스:
https://www.airport.co.kr/www/cms/frFlightStatsCon/domesticLineStats.do?MENU_ID=1250#none

02 실습

- Q1. ‘main.R’ 파일과 ‘airport.R’ 파일을 만들고, ‘airport.R’을 ‘main.R’에 로딩하세요.





02 실습

- **Q2. ‘airport.R’ 파일에 아래 양식을 따라 데이터/함수를 추가하세요**

[양 식]

- airline 변수에 항공사 명을 벡터 형식으로 추가
- flight 변수에 운항 실적 데이터를 벡터 형식으로 추가
- passenger 변수에 운항 탑승객 데이터를 벡터 형식으로 추가
- freight 변수에 화물 데이터를 벡터 형식으로 추가
- airport 변수에 전체 데이터셋을 데이터 프레임 형식으로 추가. (컬럼 명은 동일하게 사용)
- 각 변수를 리턴하는 함수 추가. 각 함수명은 ‘get_변수명’ 으로 통일

항공사 명	운항(편)	탑승객(수)	화물(톤)
아시아나항공	1,575	249,792	3,097.9
에어부산	481	90,985	516.7
에어프레미아	124	29,238	111.1
에어서울	354	71,213	273.1
제주항공	1,197	203,335	847.1
진에어	793	133,253	763.2
대한항공	1,670	250,895	5,406.1
티웨이항공	859	146,497	597.6

02 실습

- **Q2. ‘airport.R’ 파일에 아래 양식을 따라 데이터/함수를 추가하세요**

```
#airport.R

# 데이터 벡터 생성
airline <- c(‘아시아나항공’, ‘에어부산’, ‘에어프레미아’, ‘에어서울', '제주항공', '진에어', '대한항공', '티웨이항공')
flight <- c(1575, 481, 124, 354, 1197, 793, 1670, 859)
passenger <- c(249792, 90985, 29238, 71213, 203335, 133253, 250895, 146497)
freight <- c(3097.9, 516.7, 111.1, 273.1, 847.1, 763.2, 5406.1, 597.6)

# 벡터 get 함수 생성
get_airline <- function(){   return(airline) }
get_flight <- function(){   return(flight) }
get_passenger <- function(){   return(passenger) }
get_freight <- function(){   return(freight) }

# 데이터 프레임 생성
airport <- data.frame(‘항공사 명’=)
```

02 실습

- Q3. ‘airport.R’ 파일에 추가된 함수를 사용하여 ‘main.R’ 에서 각 데이터 값들을 로딩 및 확인하세요.
  - ‘get_변수명’ 함수 사용
  - Q2 에서 수정한 airport.R 파일 반드시 저장 후, 실습 진행

02 실습

- **Q3. ‘airport.R’ 파일에 추가된 함수를 사용하여 ‘main.R’ 에서 각 데이터 값들을 로딩 및 확인하세요.**

```
#main.R
source('airport.R', encoding = 'UTF-8')      # 한글 인코딩 이슈 해결 위해 UTF-8 인코딩 지정

# airport.R 사용자 함수 사용
airline <- get_airline()
print(airline)

flight <- get_flight()
print(flight)

passenger <- get_passenger()
print(passenger)

freight <- get_freight()
print(freight)

airport <- get_airport()
print(airport)
```

02 실습

- Q4. 운항이 평균 실적보다 높은 항공사들을 벡터 형태로 출력하는 함수를 ‘airport.R’에 선언한 후 ‘main.R’에서 테스트하세요.
  - 17, 18, 19 차시에서 사용한 코드를 사용하여 구현
  - 각각 함수명은 아래와 같이 지정
    - upperAgvAirline_17
    - upperAgvAirline_18
    - upperAgvAirline_18

02 실습

- **Q4. 운항이 평균 실적보다 높은 항공사들을 벡터 형태로 출력하는 함수를 'airport.R'에 선언한 후 'main.R'에서 테스트하세요.**

```
# airport.R 아래 이어서 작성
upperAgvAirline_17 <- function(){ #연산자를 사용한 방법
  upperAvg_flight <- flight > mean(flight) # c(TRUE, FALSE, FALSE, FALSE, TRUE, FALSE, TRUE, FALSE)
  result <- airline[upperAvg_flight]
  return(result)
}
upperAgvAirline_18 <- function(){ #조건문을 사용한 방법
  index <- which(flight > mean(flight))     # c(1, 5, 7)
  result <- airline[index]
  return(result)
}
upperAgvAirline_19 <- function(){ #반복문을 사용한 방법
  result <- c()
  i <- 1
  while(i <= length(airline) ){                # 데이터 모든 값 한번씩 참조
    if(flight[i] > mean(flight)){               # i 번째 항공사의 운항 실적이 평균보다 좋다면
      result <- c(result, airline[i])           # result 벡터에 i 번째 항공사 명 추가
    }
    i <- i + 1
  }
  return(result)
}
```

02 실습

- **Q4. 운항이 평균 실적보다 높은 항공사들을 벡터 형태로 출력하는 함수를 ‘airport.R’에 선언한 후 ‘main.R’에서 테스트하세요.**

```
# main.R

source('airport.R', encoding = 'UTF-8') #파일 수정이 수정되었다면 새로 로딩해줘야 함

result17 <- upperAgvAirline_17()
result18 <- upperAgvAirline_18()
result19 <- upperAgvAirline_19()
print(result17)
print(result18)
print(result19)
```

이번 차시 정리

사용자 정의 함수란?

- 사용자가 직접 정의하여 만들어내는 함수

사용자 정의 함수 특성

- 매개변수, 함수 안 변수 우선순위
- 함수 안에서 사용한 변수 밖에서 사용 불가능
- 함수 안 함수 선언 가능
- 함수를 리턴 가능

이번 차시 정리

함수 스크립트와 메인 스크립트 분리하여 사용하는 방법

- source()

20차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

21차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 데이터 형식에 대해 알 수 있다.
- 형식별 데이터를 가져올 수 있다.
- 형식별 데이터를 내보낼 수 있다.

학습 목차

- 데이터 형식 소개
- 형식별 데이터 가져오기
- 형식별 데이터 내보내기

01.
데이터 형식 소개

01 데이터 형식

- **컴퓨터가 사용할 수 있는 데이터는 어떤 형식으로 존재할까?**

- **키보드 입력**
  - 키보드로 사용자에게 데이터를 직접 입력 받음
  - scan( )

- **로컬 파일**
  - 로컬 컴퓨터에 저장된 파일
  - 인터넷 연결 유무 상관없이 사용

- **인터넷 파일**
  - 인터넷에 업로드 되어있는 파일
  - 다운로드 필요

02.

형식별 데이터 가져오기

02 데이터 가져오기 – 키보드 입력

- **scan( )**
  - numeric 형식 값만 입력가능

```
> a <- scan()
1: hi
1:
Error in scan() : scan() expected 'a real', got 'hi'
```

- **scan(what = character( ))**
  - character 형식 값 입력 가능

```
char <- scan(what = character())
print(char)
```

02 데이터 가져오기 – 로컬 파일

- **데이터 파일에 자주 사용 되는 확장자**
- **.txt**
  - ➢ 데이터들을 구분자로 분리하여 저장
  - ➢ 구분자가 여러 개거나, 없는 경우도 존재
- **.csv (Comma Separated Values)**
  - ➢ 구분자가 콤마(,)인 데이터 파일
- **.xsxl**
  - ➢ 엑셀 파일
- **웹 관련 확장자**
  - ➢ xml, json, html 등

student.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기

```
학번;이름;학년;성적
2021432;lim;1;A
2021002;shin;1;B
2020102;park;2;A
2013036;kim;4;C
```

student3.csv - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V)

```
학번,이름,학년,성적
2021432,lim,1,A
2021002,shin,1,B
2020102,park,2,A
2013036,kim,4,C
```

	A	B	C	D	E
1		class	age	sex	survived
2	1	1st class	adults	man	yes
3	2	1st class	adults	man	yes
4	3	1st class	adults	man	yes
5	4	1st class	adults	man	yes
6	5	1st class	adults	man	yes
7	6	1st class	adults	man	yes

02 데이터 가져오기 – read.table

- **테이블(행렬) 형식 파일로부터 데이터 불러오는 함수**
- **문법**

```
read.table(파일명, header=TRUE, sep="", col.names=c(), na.string="")
```

- header
  - 데이터 첫 행에 열 변수명이 있다면 TRUE, 없다면 FALSE
- sep
  - 구분자 지정가능
- col.names
  - 열 변수명 임의 설정가능
  - header가 True라면 굳이 할 필요없음
- na.string
  - 결측치(missing value)를 의미하는 문자가 어떤 것인지 지정

02 데이터 가져오기 – read.table

- **예시1. student.txt 파일 데이터 가져오기**
  - ➢ 분석
    - header 여부: 존재
    - 구분자: 세미콜론(;)
    - 결측치: 없음

student.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기

```
학번;이름;학년;성적
2021432;lim;1;A
2021002;shin;1;B
2020102;park;2;A
2013036;kim;4;C
```

```
getwd()                 #student.txt 파일 다음 경로에 넣기
dir()                   #student.txt 파일 잘 들어가있는지 확인

data <- read.table("student.txt", header=TRUE, sep=";")
class(data)             # data.frame
dim(data)               # c(4 4) , 데이터 크기 확인
print(data)
```

```
> data
     학번 이름 학년 성적
1 2021432  lim    1    A
2 2021002 shin    1    B
3 2020102 park    2    A
4 2013036  kim    4    C
```

02 데이터 가져오기 – read.table

- **예시2. student2.txt 파일 데이터 가져오기**
  - ➢ 분석
    - header 여부: 존재
    - 구분자: 공백(“ ”)
    - 결측치: 있음. 대쉬 기호(-)로 표현

student2.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V

```
학번 이름 학년 성적
2021432 lim 1 A
2021002 shin 1 B
2020102 park 2 A
2013036 kim - C
2014036 lee 1 B
```

```
getwd()                 #student2.txt 파일 다음 경로에 넣기
dir()                   #student2.txt 파일 잘 들어가있는지 확인

data <- read.table(“student2.txt”, sep=“ ”, na.string=‘-’) #header는 TRUE가 기본값이므로 생략가능
class(data)             # data.frame
dim(data)               # c(5 4)
print(data)
```

```
> print(data)
     학번 이름 학년 성적
1 2021432  lim    1    A
2 2021002 shin    1    B
3 2020102 park    2    A
4 2013036  kim   NA    C
5 2014036  lee    1    B
>
```

02 데이터 가져오기 – read.csv

- **csv 형식 파일로부터 데이터 불러오는 함수**
- **문법**

```
read.csv(파일명, header=TRUE, col.names=c(), na.string="")
```

- header
  - 데이터 첫 행에 열 변수명이 있다면 TRUE, 없다면 FALSE
- col.names
  - 열 변수명 임의 설정가능
  - header가 True라면 굳이 할 필요없음
- na.string
  - 결측치(missing value)를 의미하는 문자가 어떤 것인지 지정

02 데이터 가져오기 – read.csv

- **예시. student3.csv 파일 데이터 가져오기**
  - ➢ 분석
    - header 여부: 미존재
    - 결측치: 없음

student3.csv - Windows

파일(F) 편집(E) 서식(O)

```
2019012,song,3,A
2021432,lim,1,A
2021002,shin,1,B
2020102,park,2,A
2013036,kim,4,C
```

```
getwd()                 #student3.csv 파일 다음 경로에 넣기
dir()                   #student3.csv 파일 잘 들어가있는지 확인

data <- read.csv("student3.csv", header=FALSE)
data2 <- read.csv("student3.csv", header=FALSE, col.name=c("학번", "이름", "학년", "성적"))
dim(data)               # c(5,4)
print(data)
print(data2)
```

```
> data
       V1   V2 V3 V4
1 2019012 song  3  A
2 2021432  lim  1  A
3 2021002 shin  1  B
4 2020102 park  2  A
5 2013036  kim  4  C
```

```
> data2
     학번 이름 학년 성적
1 2019012 song    3    A
2 2021432  lim    1    A
3 2021002 shin    1    B
4 2020102 park    2    A
5 2013036  kim    4    C
```

02 데이터 가져오기 – read.excel

- **xlsl, xls 형식 엑셀 파일로부터 데이터 불러오는 readxl 패키지 함수**
  - readxl 패키지 인스톨 및 로드 필요

- **문법**

```
read_excel(파일명, sheet=NULL, range=NULL, col_names=TRUE, col_types=NULL, na="")
```

  - sheet
    - sheet가 여러 개인 경우, sheet 명 지정
    - 지정하지 않으면 첫번째 sheet 데이터 불러옴
  - range
    - 데이터 불러올 범위 지정 가능
  - col_names
    - read.table의 header와 기능 동일
  - col_types
    - read.table의 col.names와 기능 동일
  - na
    - read.table의 na.string와 기능 동일

02 데이터 가져오기 – read.excel

- **예제. student4.xlsx 파일 데이터 가져오기**
  - ➢ 분석
    - 3개의 sheet 존재: Grade, Exam, Homework
    - ‘Grade’, ‘Exam’ sheet에는 결측치 존재
    - 모든 sheet header 존재

	A	B	C	D
1	학번	이름	학년	성적
2	2021432	lim	1	A
3	2021002	shin	1	B
4	2020102	park	2	A
5	2013036	kim		C
6	2014036	lee	1	B
7				

Grade | Exam | Homework

- **Q1. student4.xlsx의 ‘Grade’ sheet 데이터 가져오기**

```
#패키지 설치 및 로딩 (최초 1회만 실행하면 됨)
install.packages("readxl")
library(readxl)

#데이터 로딩
data <- read_excel("student4.xlsx")
class(data)                    # c(“tbl_df”, “tbl”, “data.frame”) , 패키지에서 제공하는 데이터형식 추가됨
print(data)
```

```
> data
# A tibble: 5 x 4
     학번 이름   학년 성적
    <dbl> <chr> <dbl> <chr>
1 2021432 lim       1 A
2 2021002 shin      1 B
3 2020102 park      2 A
4 2013036 kim      NA C
5 2014036 lee       1 B
```

02 데이터 가져오기 – read.excel

- **Q2. student4.xlsx의 'Exam' sheet 데이터 가져오기**

```
exam_data <- read_excel("student4.xlsx", sheet="Exam", na="NULL")
print(exam_data)
```

```
# A tibble: 5 x 3
     학번 `중간 점수` `기말 점수`
    <dbl>       <dbl>       <dbl>
1 2021432          92          99
2 2021002          76          70
3 2020102          85          99
4 2013036          23          NA
5 2014036          68          79
```

	A	B	C	D
1	학번	중간 점수	기말 점수	
2	2021432	92	99	
3	2021002	76	70	
4	2020102	85	99	
5	2013036	23	NULL	
6	2014036	68	79	
7				

Grade | Exam | Homework

02 데이터 가져오기 – read.excel

- **Q3. student4.xlsx의 'Homework' sheet 데이터 가져오기**

```
#1,2 행의 의미없는 값까지 모두 로딩
homework_data <- read_excel("student4.xlsx", sheet="Homework")
print(homework_data)
# cell_rows 함수 사용. 특정 행 범위만 로딩
homework_data2 <- read_excel("student4.xlsx", range=cell_rows(3:8), sheet="Homework")
print(homework_data2)
# row-column 지정 방식 사용. 특정 행/열 범위만 로딩
homework_data3 <- read_excel("student4.xlsx", range="R3C1:R8C4", sheet="Homework")
print(homework_data3)
```

```
> print(homework_data)
# A tibble: 7 x 4
  `R 프로그래밍 수업 과제/출석` ...2  ...3  ...4
  <chr>                         <chr> <chr> <chr>
1 출석 점수는 미반영            NA    NA    NA
2 학번                          과제1 과제2 출석
3 2021432                       10    10    10
4 2021002                       7     7     9
5 2020102                       8     9     6
6 2013036                       10    8     5
7 2014036                       7     9     10
```

```
> print(homework_data2)
# A tibble: 5 x 4
     학번 과제1 과제2  출석
    <dbl> <dbl> <dbl> <dbl>
1 2021432    10    10    10
2 2021002     7     7     9
3 2020102     8     9     6
4 2013036    10     8     5
5 2014036     7     9    10
```

```
> print(homework_data3)
# A tibble: 5 x 4
     학번 과제1 과제2  출석
    <dbl> <dbl> <dbl> <dbl>
1 2021432    10    10    10
2 2021002     7     7     9
3 2020102     8     9     6
4 2013036    10     8     5
5 2014036     7     9    10
```

	A	B	C	D
1	R 프로그래밍 수업 과제/출석			
2	출석 점수는 미반영			
3	학번	과제1	과제2	출석
4	2021432	10	10	10
5	2021002	7	7	9
6	2020102	8	9	6
7	2013036	10	8	5
8	2014036	7	9	10

Grade | Exam | Homework

03.

형식별 데이터 내보내기

03 데이터 내보내기 – write.table

- **문법**

```
write.table(데이터 프레임 변수, 저장할 파일 명, row.name=TRUE, quote = TRUE, sep=“ ”)
```

  - row.name
    - 행 번호 붙힐지 말지 선택
  - quote
    - character 형식 데이터에 쌍따옴표(“’) 붙힐지 말지 선택

- **예시**

```
write.table(data, "save1.txt")
write.table(data, "save2.txt", row.names = FALSE, quote = FALSE)
write.table(data, "save3.txt", row.names = FALSE, quote = FALSE, sep=“;”)
```

03 데이터 내보내기 – write.csv

- **문법**

```
write.csv(데이터 프레임 변수, 저장할 파일 명, row.name=TRUE, quote = TRUE)
```

  - row.name
    - 행 번호 붙힐지 말지 선택
  - quote
    - character 형식 데이터에 쌍따옴표(“’) 붙힐지 말지 선택

- **예시**

```
write.csv(data, "save4.csv")
write.csv(data, "save5.csv", row.names = FALSE, quote = FALSE)
```

03 데이터 내보내기 – write.excel

- “writexl” 패키지 설치 필요

- 문법

```
write_xlsx(데이터 프레임 변수, path =저장할 파일 명)
```

- 예시

```
install.packages("writexl")
library(writexl)

write_xlsx(homework_data2, path ="save6.xlsx")
```

이번 차시 정리

데이터 가져오기

- 키보드 입력
  - scan( )
- read_table( )
  - url 사용하여 웹에서 데이터 다운로드 가능
- read_csv( )
  - url 사용하여 웹에서 데이터 다운로드 가능
- readxl::read_excel( )

이번 차시 정리

데이터 내보내기

- write_table( )
- write_csv( )
- wirtexl::write_xlsx( )

21차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

22차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 데이터 확인 기능을 이해할 수 있다.
- 데이터 프레임 기본 조작할 수 있다.
- 기초적인 데이터 분석을 할 수 있다.

학습 목차

- 데이터 확인 기능 소개 및 이해
- 데이터 프레임 기본 조작
- 기초적인 데이터 분석

01 데이터 확인 – 기본 확인 함수

- **데이터셋 선정**

```
#datasets 패키지 내장 데이터셋 사용
library(datasets)
data <- Orange          # 'Orange' 데이터셋 사용
print(data)
```

```
> print(data)
   Tree  age circumference
1     1  118            30
2     1  484            58
3     1  664            87
4     1 1004           115
5     1 1231           120
6     1 1372           142
7     1 1582           145
8     2  118            33
9     2  484            69
10    2  664           111
11    2 1004           156
12    2 1231           172
13    2 1372           203
```

- **dim 함수**

  ➢ 데이터의 차원 확인

```
dim(data)               # c(35, 3), 3개의 속성을 가진 데이터 35개
```

- **length 함수**

```
length(data)            # 3
length(data$Tree)       # 35
```

01 데이터 확인 – 기본 확인 함수

- **head 함수**
  - ➢ 앞쪽 6개 데이터만 출력

```
head(data)
```

- **tail 함수**
  - ➢ 뒤쪽 6개 데이터만 출력

```
tail(data)
```

```
> head(data)
  Tree  age circumference
1    1  118            30
2    1  484            58
3    1  664            87
4    1 1004           115
5    1 1231           120
6    1 1372           142
> tail(data)
   Tree  age circumference
30    5  484            49
31    5  664            81
32    5 1004           125
33    5 1231           142
34    5 1372           174
35    5 1582           177
```

01 데이터 확인 – 기본 확인 함수

- **str 함수**
  - ➢ class, 크기, 미리보기 값 제공
  - ➢ factor: 범주형 데이터임을 의미하는 문자열 벡터

```
str(data)
```

```
> str(data)
Classes 'nfnGroupedData', 'nfGroupedData', 'groupedData' and 'data.frame':	35 obs. of  3 variables:
 $ Tree         : Ord.factor w/ 5 levels "3"<"1"<"5"<"2"<..: 2 2 2 2 2 2 2 4 4 4 ...
 $ age          : num  118 484 664 1004 1231 ...
 $ circumference: num  30 58 87 115 120 142 145 33 69 111 ...
```

- **names 함수**
  - ➢ 데이터의 헤더 이름 출력

```
names(data)
```

```
> names(data)
[1] "Tree"          "age"           "circumference"
```

01 데이터 확인 – 기본 확인 함수

- **View 함수**
  - ➢ RStudio 뷰어 창에서 데이터 확인

```
View(data)
```

Untitled1* × data ×

Filter

	Tree	age	circumference
1	1	118	30
2	1	484	58
3	1	664	87
4	1	1004	115
5	1	1231	120
6	1	1372	142
7	1	1582	145
8	2	118	33
9	2	484	69
10	2	664	111
11	2	1004	156
12	2	1231	172
13	2	1372	203
14	2	1582	203
15	3	118	30
16	3	484	51
17	3	664	75

Showing 1 to 17 of 35 entries, 3 total columns

01 데이터 확인 – 행/열 추출

- **데이터 특정 행/열 만 추출**
  - data.frame[c(1,2),] : 데이터 프레임의 '1,2행/ 모든 열' 만 추출
  - data.frame[, c(1,2)] : 데이터 프레임의 '모든 행/ 1,2 열' 만 추출
  - data.frame[c(1,2), c(1,2)] : 데이터 프레임의 '1,2 행/ 1,2 열' 만 추출

```
data_someRows <- data[c(1,2), ]        #data의 (1,2행, 모든 열) 출력
data_someCols <- data[ , c(1,2) ]      #data의 (모든 행, 1,2열) 출력
data_someCols2 <- data[  c(1,2) ]
data_someSections <- data[  c(1,2), c(1,2) ] #data의 (1,2행, 1,2열)만 출력
```

```
> data_someRows
  Tree age circumference
1    1 118            30
2    1 484            58
```

```
> data_someCols
   Tree  age
1     1  118
2     1  484
3     1  664
4     1 1004
5     1 1231
6     1 1372
7     1 1582
8     2  118
9     2  484
10    2  664
11    2 1004
12    2 1231
13    2 1372
14    2 1582
15    3  118
16    3  484
17    3  664
```

```
> data_someSections
  Tree age
1    1 118
2    1 484
```

01 데이터 확인 – 행/열 제거

- **데이터 특정 행/열 만 제거**
  - data.frame[c(-1,-2),] : 데이터 프레임의 1,2행 제거
  - data.frame[, c(-2)] : 데이터 프레임의 2열 제거
  - data.frame[c(1,2), c(1,2)] : 데이터 프레임의 '1,2 행/ 1,2 열' 제거
    - 3열의 3~35행 데이터만 남음

```
data_deleteRows <- data[c(-1,-2), ]
data_deleteCols <- data[ , c(-1,-2) ]
data_deleteRowCol <- data[  c(-1,-2), c(-1,-2) ]
```

01 데이터 분석 - 기본 분석

- **mean, var, sd (평균, 분산, 표준편차)**

```
names(data)                    # c("Tree", "age", "circumference")
ageVec <- data$age              # data 데이터 프레임에서 age 컬럼의 벡터만 추출
circVec <- data$circumference # data 데이터 프레임에서 circumference 컬럼의 벡터만 추출
mean(ageVec)                    # 922.1429
mean(circVec)                   # 115.8571
var(ageVec)                     # 241930.7
var(circVec)                    # 3304.891
sd(ageVec)                      # 491.8645
sd(circVec)                     # 57.48818
```

- **table 함수**
  - ➢ 벡터 내에서 데이터 값이 각각 몇 회 등장하였는지 체크
  - ➢ 범주형 데이터 분석에 적합

```
names(data)                    # c("Tree", "age", "circumference")
treeVec <- data$Tree
table(treeVec)
```

```
> table(treeVec)
treeVec
3 1 5 2 4
7 7 7 7 7
```

01 데이터 분석 - 기본 분석

- **median, range(min, max)**

```
# 수치형 값만 분석
ageVec <- data$age
circVec <- data$circumference
median(ageVec)                # 1004
median(circVec)               # 115
range(ageVec)                 # c(118, 1582)
range(circVec)                # c(30 214)
```

- **quantile (사분위 수)**

```
# 수치형 값만 분석
ageVec <- data$age
circVec <- data$circumference
median(ageVec)                # 1004
median(circVec)               # 115
range(ageVec)                 # c(118, 1582)
range(circVec)                # c(30 214)
```

```
> quantile(agevec)
  0%  25%  50%  75% 100%
 118  484 1004 1372 1582
```

```
> quantile(circvec)
   0%   25%   50%   75%  100%
 30.0  65.5 115.0 161.5 214.0
```

01 실습(데이터 확인) – 실데이터 실습

- **Q1. ‘TravelMode.csv‘ 데이터를 로딩하고 데이터를 확인하세요.**
  - ➢ head 함수사용하여 데이터 확인
  - ➢ 데이터 크기 확인

```
data <- read.csv('TravelMode.csv')
head(data)
dim(data)   # (840 10)
```

```
> head(data)
  X individual  mode choice wait vcost travel gcost income size
1 1          1   air     no   69    59    100    70     35    1
2 2          1 train     no   34    31    372    71     35    1
3 3          1   bus     no   35    25    417    70     35    1
4 4          1   car    yes    0    10    180    30     35    1
5 5          2   air     no   64    58     68    68     30    2
6 6          2 train     no   44    31    354    84     30    2
>
```

  - ➢ 각 가정(individual)은 4개의 교통수단(mode) 중 하나를 선택(choice)
  - ➢ 대기시간(wait), 교통비용(vcost), 목적지 도착시간(travel), 소득(income), 가정 구성원 수(size)를 고려하여 산정된 총비용(gcost) 존재

01 실습(데이터 가공) - 실데이터 실습

- **Q2. ‘TravelMode.csv‘ 데이터를 조작하여 실제 여행에 사용한 데이터로만 구성된 데이터셋을 생성하세요.**

```
data <- read.csv(‘TravelMode.csv’)
head(data)
dim(data)   # (840 10)
```

01 실습(데이터 가공) - 실데이터 실습

- **Q2. ‘TravelMode.csv‘ 데이터를 조작하여 실제 여행에 사용한 데이터로만 구성된 데이터셋을 생성하세요.**

- **Q2-1. choice값이 yes인 행으로만 이루어진 데이터셋을 구성**
  - ‘실제 여행에 사용’ == ‘choice 값이 yes'

```
choice <- data$choice
choice_yes_index <- which(choice == "yes")
choice_yes_index
actual_data <- data[choice_yes_index,]  #choice값이 yes인 행과, 그 행들의 모든 열로 이루어진
데이터 프레임
head(actual_data)
```

```
> head(actual_data)
    x individual  mode choice wait vcost travel gcost income size
4   4          1   car    yes    0    10    180    30     35    1
8   8          2   car    yes    0    11    255    50     30    2
12 12          3   car    yes    0    23    720   101     40    1
16 16          4   car    yes    0     5    180    32     70    3
20 20          5   car    yes    0     8    600    99     45    2
22 22          6 train    yes   40    20    345    57     20    1
```

01 실습(데이터 가공) - 실데이터 실습

- **Q2. ‘TravelMode.csv‘ 데이터를 조작하여 실제 여행에 사용한 데이터로만 구성된 데이터셋을 생성하세요.**

- **Q2-2. (optional) x, individual, choice 컬럼은 필요 없으므로 제거**

```
processed_data <- actual_data[,c(-1, -2, -4)] # x, individual, choice 컬럼 제거
head(processed_data)
str(processed_data)        # 생성한 데이터셋 확인
```

```
> head(processed_data)
    mode choice wait vcost travel gcost income size
4    car    yes    0    10    180    30     35    1
8    car    yes    0    11    255    50     30    2
12   car    yes    0    23    720   101     40    1
16   car    yes    0     5    180    32     70    3
20   car    yes    0     8    600    99     45    2
22 train    yes   40    20    345    57     20    1
```

```
> str(processed_data)
'data.frame':   210 obs. of  7 variables:
 $ mode  : chr  "car" "car" "car" "car" ...
 $ wait  : int  0 0 0 0 0 40 45 0 0 0 ...
 $ vcost : int  10 11 23 5 8 20 148 50 17 7 ...
 $ travel: int  180 255 720 180 600 345 115 780 210 210 ...
 $ gcost : int  30 50 101 32 99 57 160 135 40 30 ...
 $ income: int  35 30 40 70 45 20 45 12 40 70 ...
 $ size  : int  1 2 1 3 2 1 1 1 1 2 ...
```

01 실습(데이터 분석) - 실데이터 실습

- **Q3. 가공해서 생성한 데이터셋에서, wait, vcost, travel, gcost, income 컬럼의 평균값을 구하세요**

```
print(mean(processed_data$wait))
print(mean(processed_data$vcost))
print(mean(processed_data$travel))
print(mean(processed_data$gcost))
print(mean(processed_data$income))
```

01 실습(데이터 분석) - 실데이터 실습

- **Q4. 가공하여 생성한 데이터셋에서, mode, size 컬럼은 범주형 데이터의 성격이 강합니다. table 함수를 사용하여 두 컬럼의 값을 분석하세요**

```
table(processed_data$mode)
table(processed_data$size)
```

```
> table(processed_data$mode)

  air   bus   car train
   58    30    59    63
```

```
> table(processed_data$size)

  1   2   3   4   5   6
114  58  20  15   2   1
```

- **Q5. 여행 시, 교통수단으로 버스를 사용한 비율을 구하세요.**

```
mode_table <- table(processed_data$mode)
mode_names <- names(mode_table)              # mode table의 열 이름 벡터화
mode_numeric <- as.numeric(mode_table)       # table -> numeric 형 변환
bus_selected_count <- mode_numeric[which(mode_names == "bus")]  #bus가 선택된 횟수 저장
bus_selected_rate <- bus_selected_count/ sum(mode_numeric)
print(bus_selected_rate)                     # 약 14.3%
```

01 실습(데이터 분석) - 실데이터 실습

- **Q6. 1인 가구의 여행 비율을 구하세요.**
  - Q5 코드 참조

```
size_table <- table(processed_data$size)
size_names <- names(size_table)
size_numeric <- as.numeric(size_table)
solo_selected_count <- size_numeric[which(size_names == "1")]  #1인 가구 수 저장
size_selected_rate <- solo_selected_count/ sum(size_numeric)
print(size_selected_rate)                                      # 약 54.3%
```

이번 차시 정리

데이터 확인

- dim( ), length( )
- head( ), tail( )
- str( ), View( )

데이터 프레임 기본 조작

- 특정 행만 추출/제거
- 특정 열만 추출/제거

이번 차시 정리

기초적인 데이터 분석

- 산술 함수 사용한 분석
- 4분위 수 분석: quantile
- 범주형 데이터 분석: table

22차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

23차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 데이터를 그릴 수 있다.
- 화면 분할을 할 수 있다.

학습 목차

- 데이터 그리기
- 화면 분할

01.

데이터 그리기

01 데이터 그리기 – 산점도, 그래프 그리기

- **plot( ) 함수**
  - 산점도, 그래프 그리기 함수

- **문법**

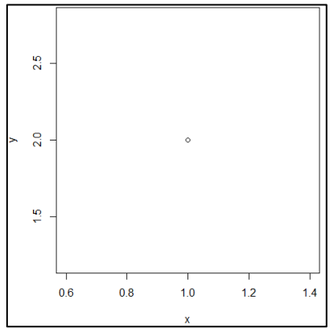
```
plot(x,y, type='p', main=NULL, xlab="", ylab="", xlim=c(), ylim=c())
```

  - x,y: 데이터의 x,y 축 값
  - type: 그래프 타입 (p: 점, l: 선, b: 점&선 등)
  - main: 그래프 제목
  - xlab: x축 이름
  - ylab: y축 이름
  - xlim: x축 값 범위
  - ylim: y축 값 범위
  - 이 외 다양한 매개변수 존재

01 데이터 그리기 – 산점도 그리기

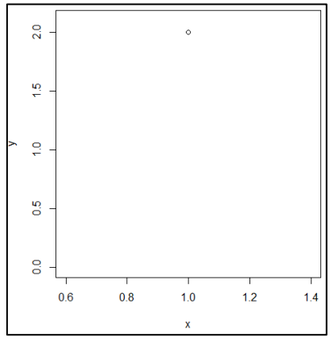
- **예시1**

```
x = 1
y = 2
plot(x,y)
```



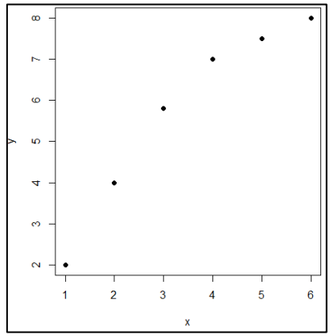
- **예시2**

```
x = 1
y = 2
plot(x,y, ylim=c(0.6,1.4))
```



- **예시3**

```
x = c(1,2,3,4,5,6)
y = c(2,4,5.8,7,7.5,8)
plot(x,y, pch=19)
```

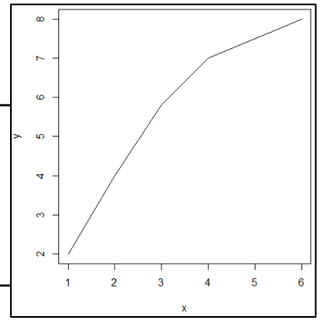


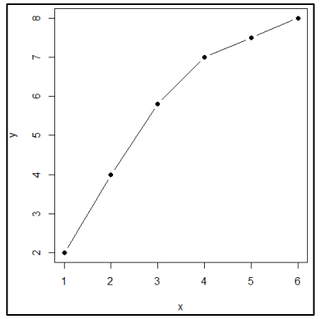
➢ pch: 점의 모양 (숫자: 1~25 , 문자:“*”, “%”, “#”, 등)

01 데이터 그리기 - 그래프 그리기

- **예시1**

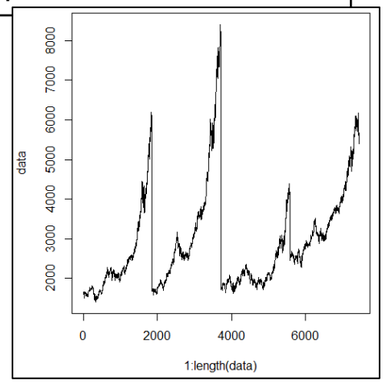
```
x = c(1,2,3,4,5,6)
y = c(2,4,5.8,7,7.5,8)
plot(x,y, type="l", pch=19)
plot(x,y, type="b", pch=19)
```

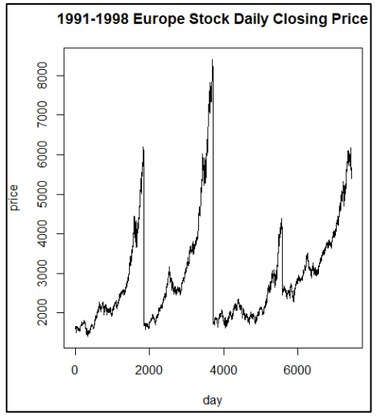




- **예시2**

```
library(datasets)
data <- EuStockMarkets    #1991-1998년간 유럽주식지수 종가 데이터셋
plot(1:length(data),data, type="l")
plot(1:length(data),data, type="l", main="1991-1998 Europe Stock Daily
Closing Price", xlab='day', ylab='price')  #제목, 축이름 추가
```



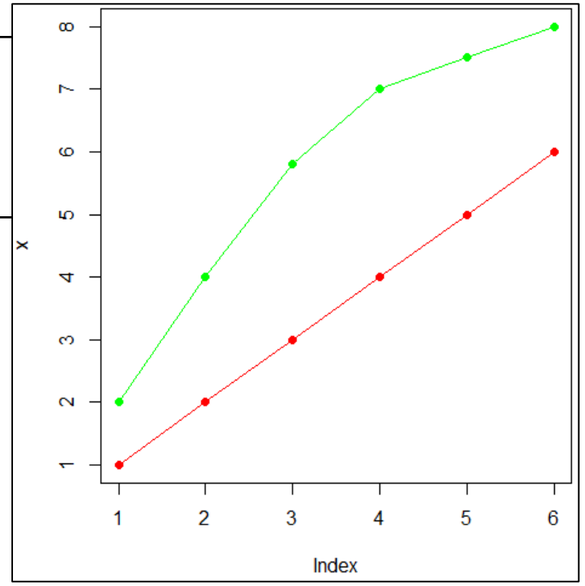


01 데이터 그리기 - 다중 그래프 그리기

- **방법**
  - plot() 함수 사용하여 1개 그래프 그린 후, lines() 함수 사용하여 추가 그래프 그림
  - plot을 통해 그림을 그리고 lines를 통해 plot위에 추가로 그림을 덧칠하는 개념

- **예시**

```
x = c(1,2,3,4,5,6)
y = c(2,4,5.8,7,7.5,8)
plot(x, type="o", pch=19, col="red", xlim=c(1,6), ylim=c(1,8))
lines(y, type="o", pch=19, col="green")
```

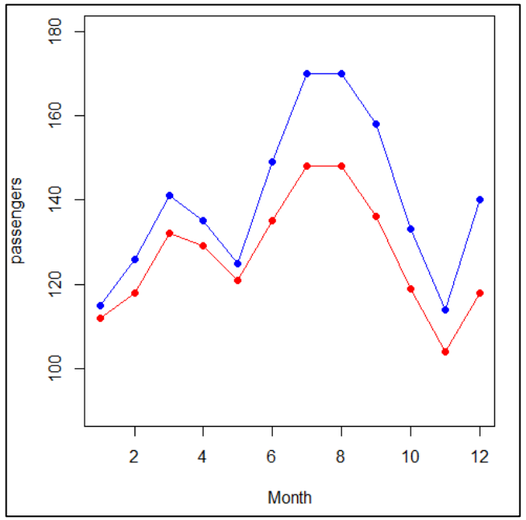


  - col: 그래프 선 색 지정 매개변수

01 데이터 그리기 - 다중 그래프 그리기

- **예제**

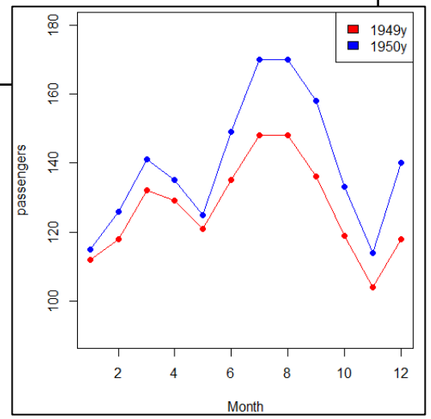
```
library(datasets)
data <- as.numeric(AirPassengers)    #1949년도 1월 - 1960년도 12월까지 월간 향공기 탑승 승객 수
                                       데이터셋. 12x12= 144개 데이터
length(data) #144
y1949 = data[1:12]
y1950 = data[13:24]

plot(y1949, type="o", pch=19, col="red", ylim=c(90,180), xlab="Month", ylab="passengers")
lines(y1950, type="o", pch=19, col="blue")
```



01 데이터 그리기 – 범례 그리기

- **그래프의 라벨을 알려주는 값**
  - legend(위치 값, legend=라벨 값, fill=색상 값)
    - 위치 값: “topright”, “topleft”, “top”, “bottom”, “bottomleft”, “bottomright”, “center”, “left”, “right”
- **예시**

```
data <- as.numeric(AirPassengers)    #1949년도 1월 - 1960년도 12월까지 월간 향공기 탑승 승객 수
y1949 = data[1:12]
y1950 = data[13:24]
plot(y1949, type="o", pch=19, col="red", ylim=c(90,180), xlab="Month", ylab="passengers")
lines(y1950, type="o", pch=19, col="blue")
legend("topright", legend=c("1949y", "1950y"), fill=c("red", "blue"))
```



01 데이터 그리기 – 파이 차트

- **pie( ) 함수**
  - ➢ 원형 차트: 비율을 표현하여 범주형 데이터 시각화에 적합

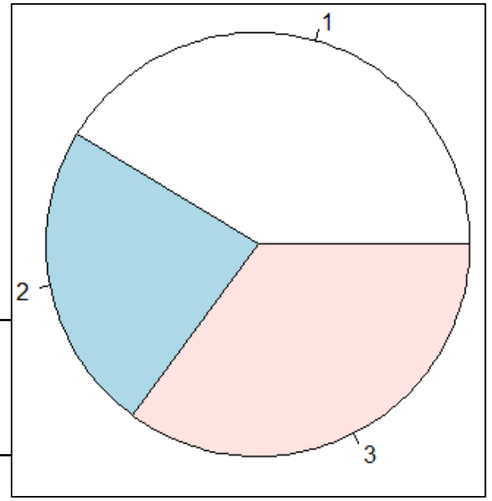
- **pie( ) 함수**

```
pie(data, label="", init.angle=0, radius=1)
```

  - ➢ label: 요소별 이름 등록
  - ➢ init.angle: 기준선 각도
  - ➢ radius: 원의 크기 조절

- **예시**

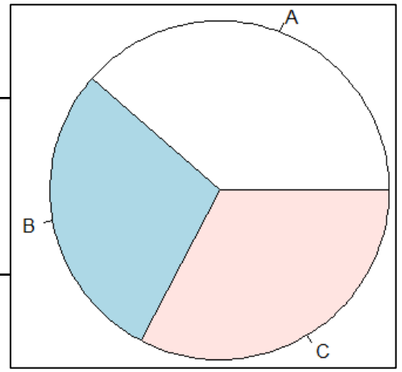
```
data <- c(40, 23, 34)
pie(data)
```



01 데이터 그리기 – 파이 차트

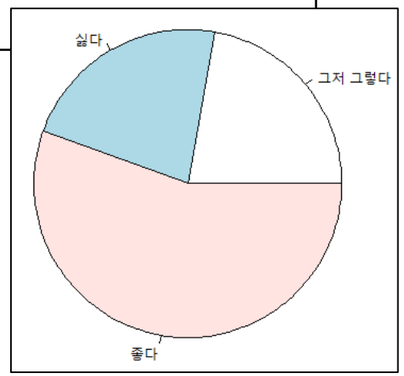
- **예제1**

```
data <- c(40, 23, 34)
label <- c(“A”, “B”, “C”)
pie(data, label=label)
```



- **예제2**

```
#범주형 데이터를 파이차트로 시각화
research <- c("좋다", '그저 그렇다', '좋다',  '좋다',  '좋다',  '좋다', '싫다', '그저 그렇다', '싫다')
result <- table(research)
pie(result)   #table 객체 삽입 시, label이 자체적으로 붙음
```



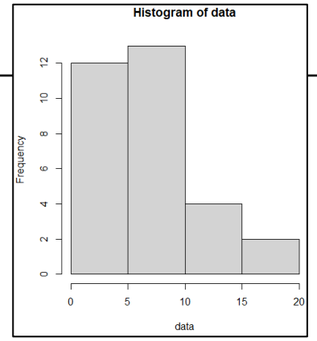
01 데이터 그리기 – 히스토그램

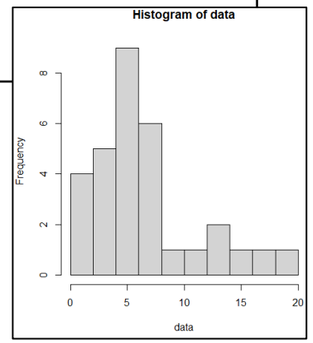
- **hist( ) 함수**
  - ➢ 수치형 데이터를 범주화하여 표현하는데 적합
- **문법**

```
hist(data, breaks=NULL, frep=T)
```

  - ➢ break: 범주 개수
  - ➢ frep: 결과 출력 값 기준. T= 횟수, F= 비율(density)
- **예시**

```
data = c(2,1,4,5.8,6,7,12,13,14,4,5,6,7,6,5,4,7,7.5,8,6,4,7,6,5,9,1,3,2,15,20,17)
hist(data)
hist(data, breaks=10)
```

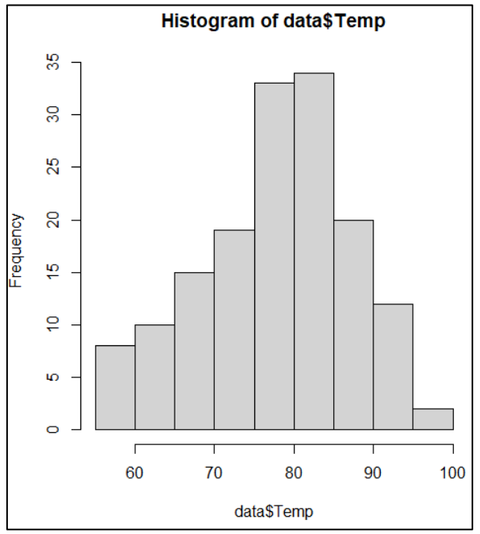


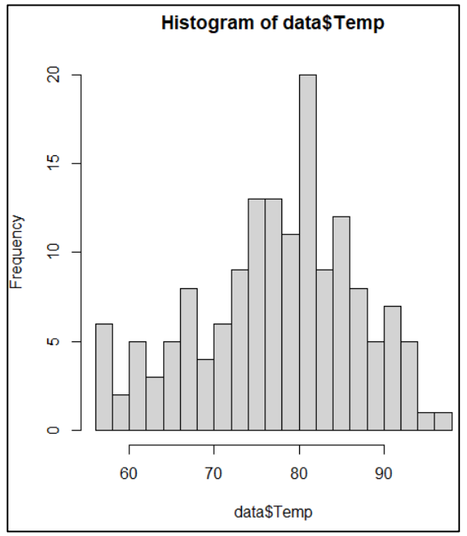


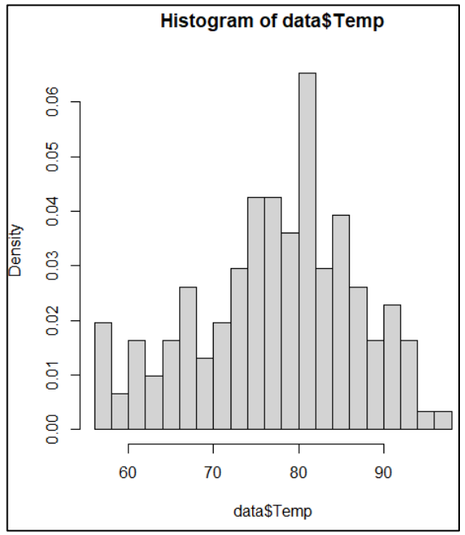
01 데이터 그리기 - 히스토그램

- **예제**

```
data <- airquality          #내장 데이터셋 airquality 사용
head(data)
temp <- data$Temp
range(temp)                 # 56 97
hist(data$Temp)
hist(data$Temp, breaks=20)
hist(data$Temp, breaks=20, freq=F)
```



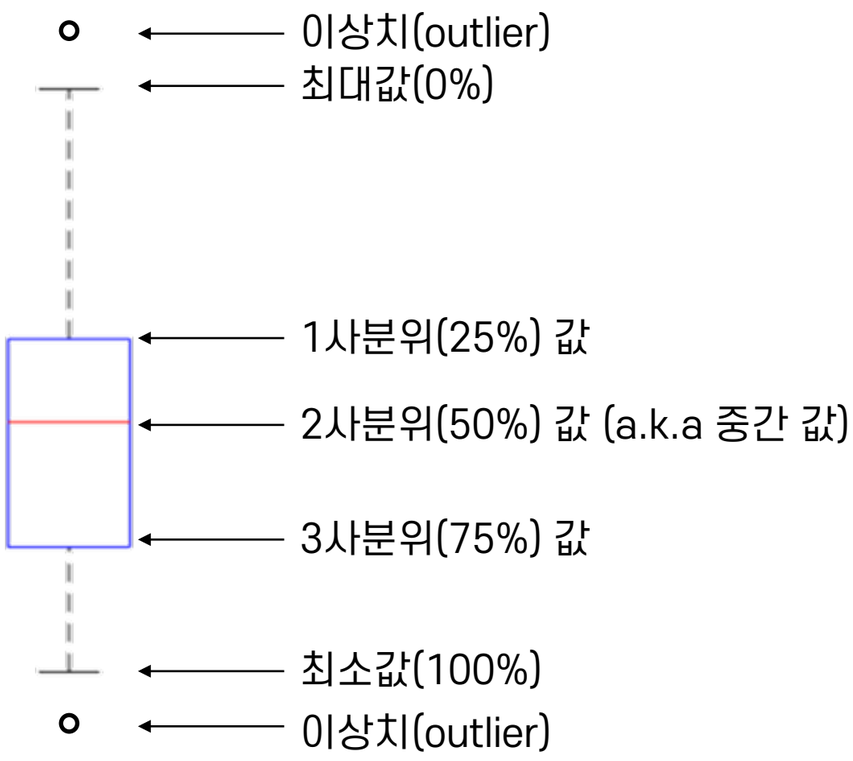




01 데이터 그리기 – 상자그림

- **boxplot( ) 함수**
  - ➢ 수치형 데이터의 분포를 시각화 하는데 적합

- **상자그림이란?**
  - ➢ 5가지 통계수치를 표현
    - ▪ 최대값, 1사분위값, 2사분위값, 3사분위값, 최소값
  - ➢ 이상치 표현
    - ▪ 최대값 보다 매우 크거나, 최소값 보다 매우 작은 값
    - ▪ 잘 못 적힌 값 취급



01 데이터 그리기 – 상자그림

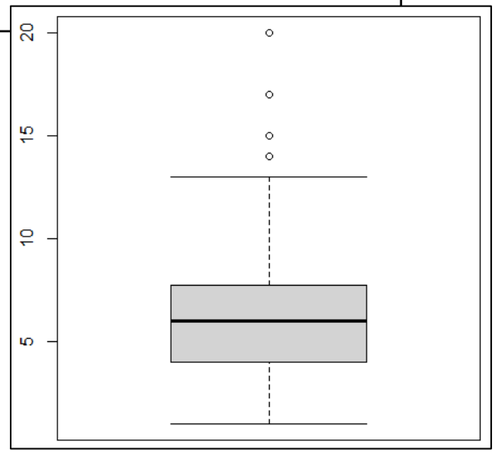
- **hist( ) 함수**

```
boxplot(data, boxwex=NULL, )
```

  - ➢ break: 범주 개수
  - ➢ frep: 결과 출력 값 기준. T= 횟수, F= 비율(density)

- **예시**

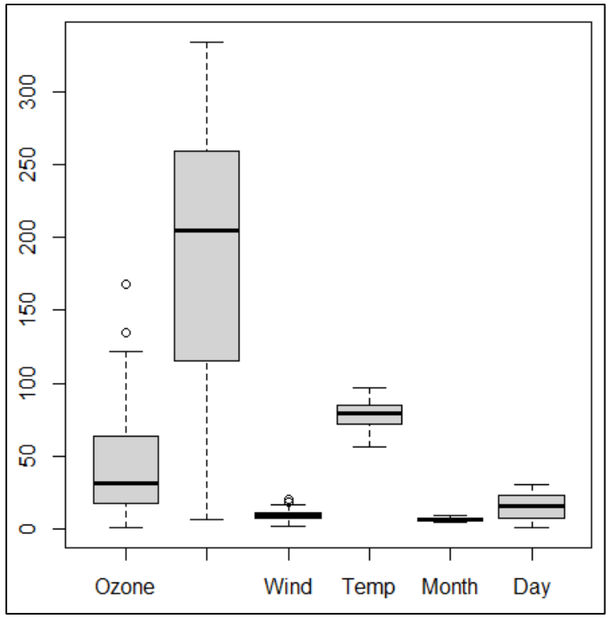
```
data = c(2,1,4,5.8,6,7,12,13,14,4,5,6,7,6,5,4,7,7.5,8,6,4,7,6,5,9,1,3,2,15,20,17)
boxplot(data)
```



01 데이터 그리기 - 상자그림

- **예제**
  - ➢ data.frame 형식 데이터 사용할 시, 컬럼 별 boxplot 그려짐

```
#다중 상자그림
data <- airquality #내장 데이터셋 airquality 사용
head(data)
boxplot(data)
```



02.

화면 분할

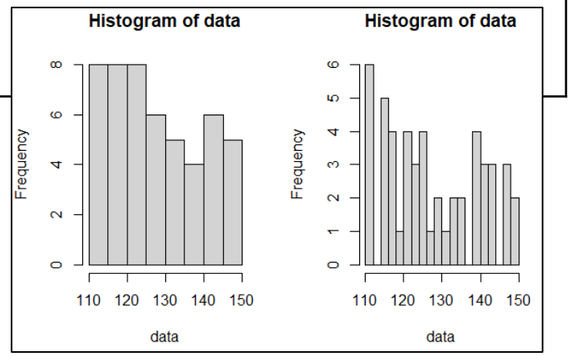
01 데이터 그리기 - 상자그림

- **par( ) 함수**
  - ➢ 한 화면에 여러 그래프를 넣고 싶을 때 사용
- **문법**

```
par(mfrow = c(분할 화면 행 개수, 분할 화면 열 개수))
```

  - ➢ 한 화면에 여러 그래프를 넣고 싶을 때 사용
- **예시**

```
par(mfrow = c(1,2))
data <- sample(110:150, 50) # 110~150 사이 50개 정수 무작위 샘플링
hist(data)
hist(data, breaks=20)
```



이번 차시 정리

다양한 데이터 그리기 방법 이해

- 산점도, 꺾은선 그래프
  - ➢ 다중 그래프
  - ➢ 범례
- 막대 그래프
- 파이 차트
- 히스토그램
- 상자그림(boxplot)

화면 분할 방법 이해

23차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

24차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 현실 데이터를 가져올 수 있다.
- 데이터 확인 및 조작을 할 수 있다.
- 데이터 그리기를 할 수 있다.

학습 목차

- 현실 데이터 가져오기 실습
- 데이터 확인 및 조작 실습
- 데이터 그리기 실습

01 실습

1. 진짜 9월 제주도 그 데이터 엑셀로 읽어와서 뭐 분석하고
   파이차트 그리기 (로딩 특화)

2. 코로나 일일 확진자 데이터 들고와서 그래프그리기
   (로딩 분석 시각화 전반적)

3. 그 오존 있는 데이터 결측치 컬럼 평균값으로 넣는 전처리 랑
   omit하는 전처리 하고 boxplot 결과 보기 (전처리 특화)

4. 22차시 전처리한 데이터 시각화 실습 (시각화 특화)

01 실습 - 제주-김포 비행기 실적

Q1. 2021년 9월 국내 항공사의 '김포공항-제주도'편 비행기 실적 엑셀자료를 다운받아, 항공사별 화물 실적을 파이차트로 그리세요.

- 데이터셋 다운로드 경로: https://www.airport.co.kr/www/cms/frFlightStatsCon/domesticLineStats.do?MENU_ID=1250#none
- 파이차트에는 항공사명과 차지하는 비율값(%)이 라벨로 그려져야 함

01 실습 - 제주-김포 비행기 실적

Q1. 2021년 9월 국내 항공사의 '김포공항-제주도'편 비행기 실적 엑셀자료를 다운받아, 항공사별 화물 실적을 파이차트로 그리세요.

- 데이터셋 다운로드 경로: https://www.airport.co.kr/www/cms/frFlightStatsCon/domesticLineStats.do?MENU_ID=1250#none
- 파이차트에는 항공사명과 차지하는 비율값(%)이 라벨로 그려져야 함

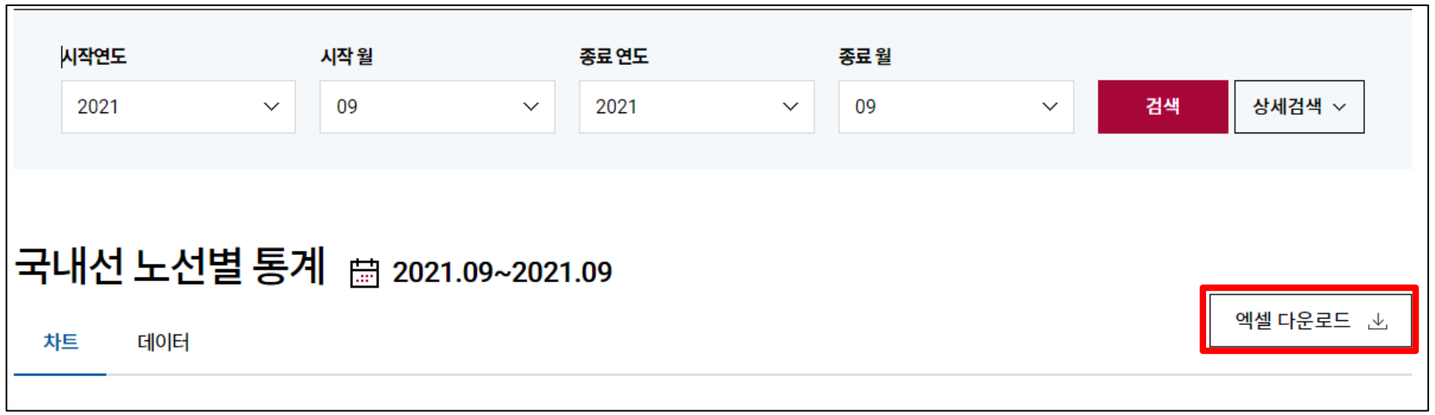
A1-1. 문제 해결 계획 수립

- 슈도코드(pseudo code) 작성

```
#데이터 다운로드
### 데이터 형식 파악. R에서 사용가능한 데이터인지 확인
### 전체 데이터 중 내가 필요한 정보는 어디에 위치하는지 파악
#데이터 로딩
#데이터에서  ' 항공사별 화물 실적 ' 데이터를 추출
# 추출한 데이터를 파이차트로 그림. (단, 항공사명, 비율 라벨 추가해야 함)
```

01 실습 - 제주-김포 비행기 실적

A1-2. 데이터 다운로드



- working director에 “airport.xlsx”이름으로 데이터 저장
- 사용할 수 있는 데이터인가? → O (엑셀 파일)
- 내가 필요한 정보는 어디있는가 →
  - 80~85번째 행
    - ✓ C 열에 ‘항공사 명‘ 데이터 존재
    - ✓ L 열에 '화물 실적‘ 데이터 존재

01 실습 - 제주-김포 비행기 실적

A1-3. 데이터 로딩

```
#엑셀 로딩 패키지 다운로드
install.packages("readxl")
library(readxl)

#데이터 로딩

#C열에 항공명, L열에 화물 실적 데이터 존재
data <- ?# 80행 C열부터 85행 L
열까지의 데이터 로딩
head(data)
```

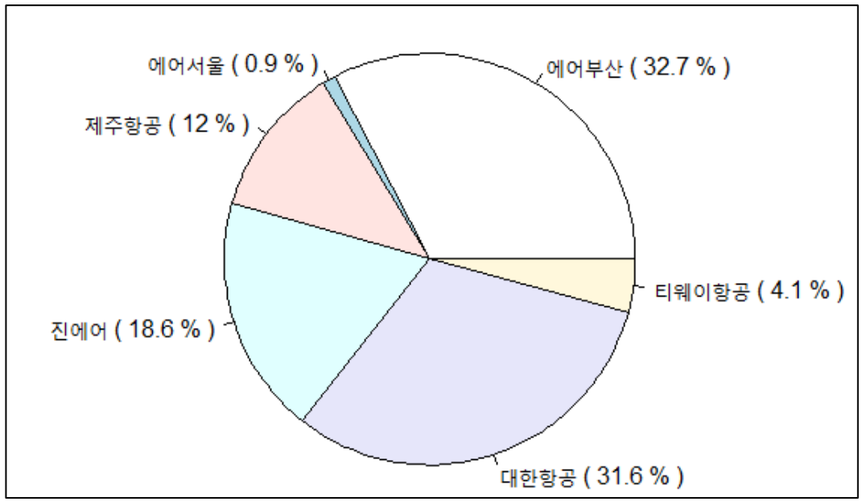
A1-4. 데이터에서 '항공사별 화물 실적' 데이터를 추출

```
#데이터에서  ' 항공사별 화물 실적 ' 데이터를 추출
airportName <- ?  # 항공명
freight <- as.numeric( ? ) # 화물 실적
```

01 실습 - 제주-김포 비행기 실적

A1-5. 라벨 값 생성 및 파이차트 출력

```
# 추출한 데이터를 파이차트로 그림.

#항공사명, 비율 라벨 값 생성
freight_percent<-                ?                # 항공사별 실적 비율 계산
freight_percent <- round(freight_percent, digit=1) #round: 반올림 함수, 소숫점 1째자리에서 반올림
label_data <- paste(airportName, "(", freight_percent,"%", ")")
print(label_data)

#파이차트 출력
                ?
```



01 실습 - 코로나 예방접종 현황

Q2. ‘서울특별시 코로나19 백신 예방접종 현황' 데이터를 다운로드 받으세요. 2021.09.01~2021.09.30일 한달간의 '당일 1차 접종자' 데이터와 '당일 2차 접종자 ' 데이터를 한 그래프 안에 그리세요. 또한, 같은 기간 '1차 접종률‘ 데이터와 '2차 접종률‘ 데이터를 한 그래프안에 그리세요.

- 데이터셋 다운로드 경로: http://data.seoul.go.kr/dataList/OA-20914/S/1/datasetView.do
- 총 두 개의 plot 출력. 각 그래프는 색이 달라야 함.
- 두 plot를 화면분할 하여 출력 하세요

01 실습 - 코로나 예방접종 현황

Q2. '서울특별시 코로나19 백신 예방접종 현황' 데이터를 다운로드 받으세요. 2021.09.01~2021.09.30일 한달간의 '당일 1차 접종자' 데이터와 '당일 2차 접종자 ' 데이터를 한 그래프 안에 그리세요. 또한, 같은 기간 '1차 접종률' 데이터와 '2차 접종률' 데이터를 한 그래프안에 그리세요.

- 데이터셋 다운로드 경로: http://data.seoul.go.kr/dataList/OA-20914/S/1/datasetView.do
- 총 두 개의 plot 출력. 각 그래프는 색이 달라야 함.
- 두 plot를 화면분할 하여 출력 하세요

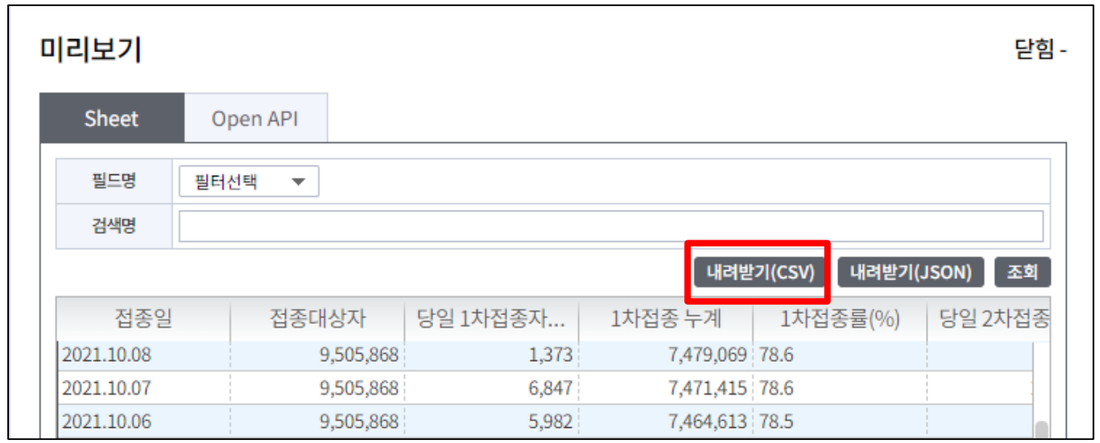
A2-1. 문제 해결 계획 수립

- 슈도코드(pseudo code) 작성

```
#데이터 다운로드
### 데이터 형식 파악. R에서 사용가능한 데이터인지 확인
### 전체 데이터 중 내가 필요한 정보는 어디에 위치하는지 파악
#데이터 로딩
#9월 한달간 ' 당일 1차 접종자 ' , '당일 2차 접종자' 데이터를 추출
# 추출한 데이터를 다중 그래프로 그림
#9월 한달간 ' 1차 접종률 ' , ' 2차 접종률 ' 데이터를 추출
# 추출한 데이터를 다중 그래프로 그림
```

01 실습 - 코로나 예방접종 현황

A2-2. 데이터 다운로드



- working director에 “covid19.xlsx”이름으로 데이터 저장
- 사용할 수 있는 데이터인가? → O (csv 파일)
- 내가 필요한 정보는 어디있는가 →
  - 36~65번째 행
    - ✓ 3,6 열에 ‘1차/2차 일일 접종자’ 데이터 존재
    - ✓ 5,8 열에 ‘1차/2차 접종률’ 데이터 존재
  - 날짜가 역순으로 정렬 되있음. 데이터 위치 값 반전 필요

01 실습 - 코로나 예방접종 현황

A2-3. 데이터 로딩

```
data <- read.csv(                    ?                    ) #header는 없으며, 앞의 35줄을 skip하고
그 다음부터 30줄을 로딩
dim(data)
data <- data[30:1,]                                    #데이터 위아래 뒤집기(reverse)
```

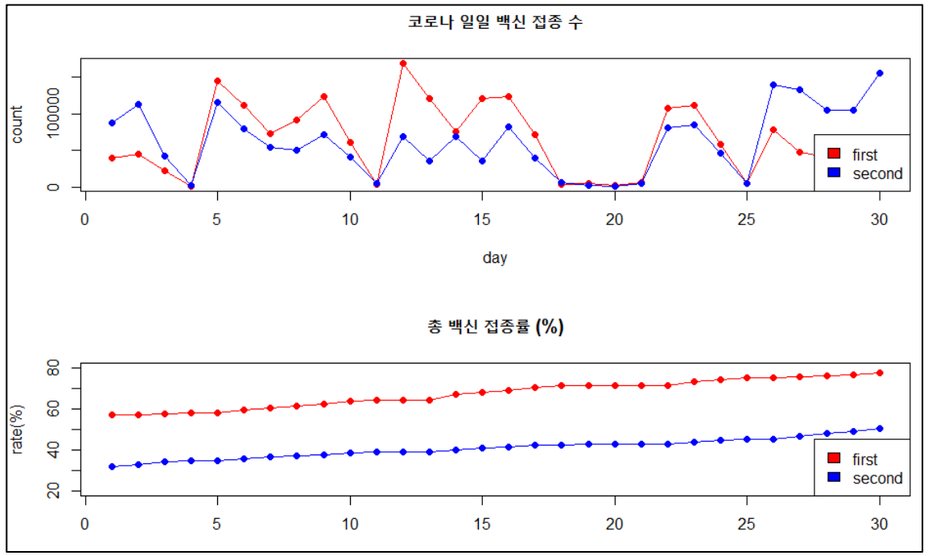
A2-4. 데이터에서 ‘항공사별 화물 실적’ 데이터를 추출

```
#9월 한달간  ' 당일 1차 접종자 ' , '당일 2차 접종자' 데이터를 추출
#9월 한달간  ' 1차 접종률 ' ,  ' 2차 접종률 '  데이터를 추출

first_vaccine <-     ?      #일일 1차 접종자 데이터 추출
second_vaccine <-     ?      #일일 1차 접종자 데이터 추출
first_rate <-     ?      # 1차 접종률 데이터 추출
second_rate <-     ?      # 1차 접종률 데이터 추출
```

01 실습 - 코로나 예방접종 현황

A2-5. 화면 분할 및 그래프 출력

```
#화면분할 출력
par(mfrow = c(2,1))
plot(first_vaccine, type='o', col='red', pch=19, xlab="day", ylab="count", main="코로나 일일 백신 접종 수")
lines(second_vaccine, type='o', col='blue', pch=19)
legend("bottomright", legend=c('first', 'second'), fill=c("red", "blue"))

plot(first_rate, type='o', col='red', pch=19, xlab="day", ylab="rate(%)", ylim=c(20,80), main="총 백신 접종률 (%)")
lines(second_rate, type='o', col='blue', pch=19)
legend("bottomright", legend=c('first', 'second'), fill=c("red", "blue"))
```



01 실습 - 시각화 실습

Q3. 아래는 22차시에 실습하였었던 ' TravelMode.csv' 데이터 전처리 코드입니다. 아래 코드를 실행 한 후, 다음 물음에 답하세요.

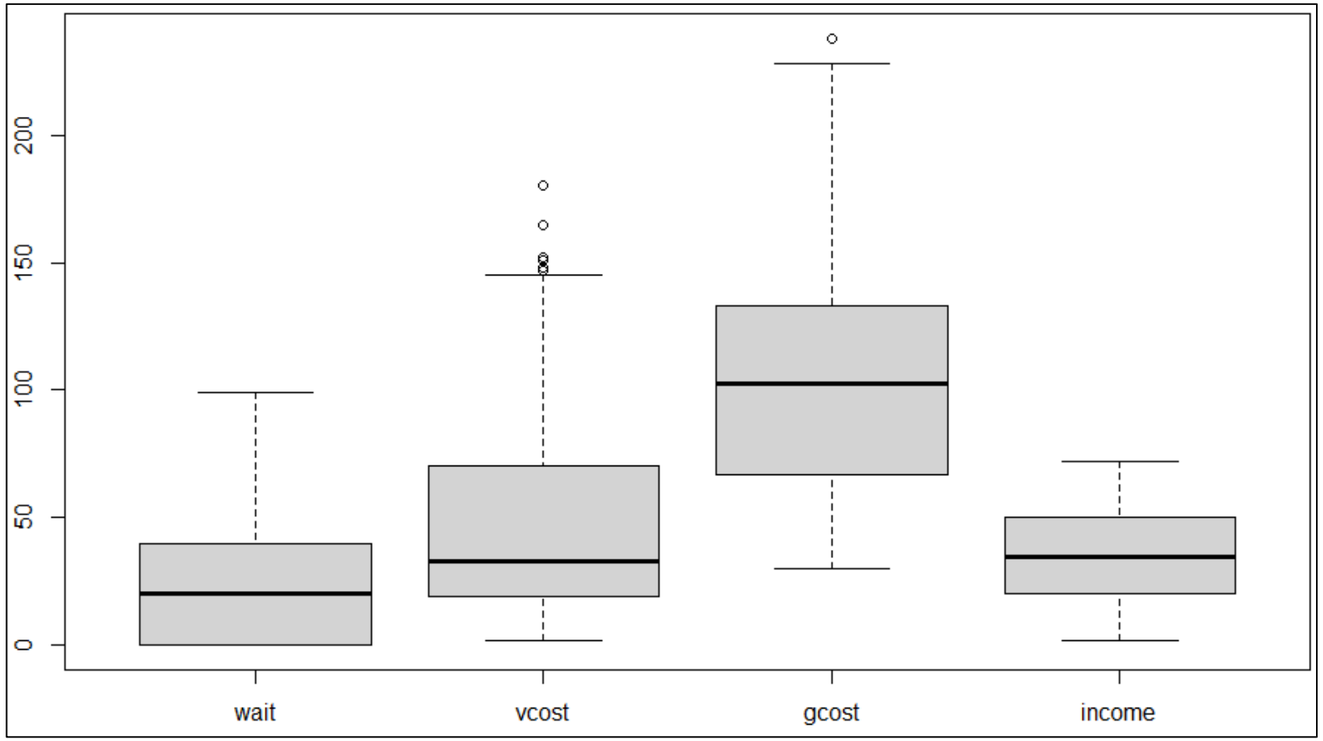
- Q3-1. wait, vcost, gcost, income 컬럼에 대한 다중 박스그림을 그리세요.
- Q3-2. travel 컬럼에 대한 히스토그램을 그리세요.
- Q3-3. 화면 분할을 사용하여, mode, size 컬럼에 대한 파이차트를 한 plot에 그리세요.
- 각 시각화 그림은 자유롭게 꾸미세요. (권장)

```
#TraverMode.csv 데이터셋 전처리
data <- read.csv('TravelMode.csv')
head(data)
dim(data)   # (840 10)
choice <- data$choice
choice_yes_index <- which(choice == "yes")
choice_yes_index
actual_data <- data[choice_yes_index,]  #choice값이 yes인 행과, 그 행들의 모든 열로 이루어진 데이터 프레임
head(actual_data)
processed_data <- actual_data[,c(-1, -2, -4)] # x, individual, choice 칼럼 제거
head(processed_data)
str(processed_data)        # 생성한 데이터셋 확인
```

01 실습 - 시각화 실습

Q3-1. wait, vcost, gcost, income 컬럼에 대한 다중 박스그림을 그리세요.

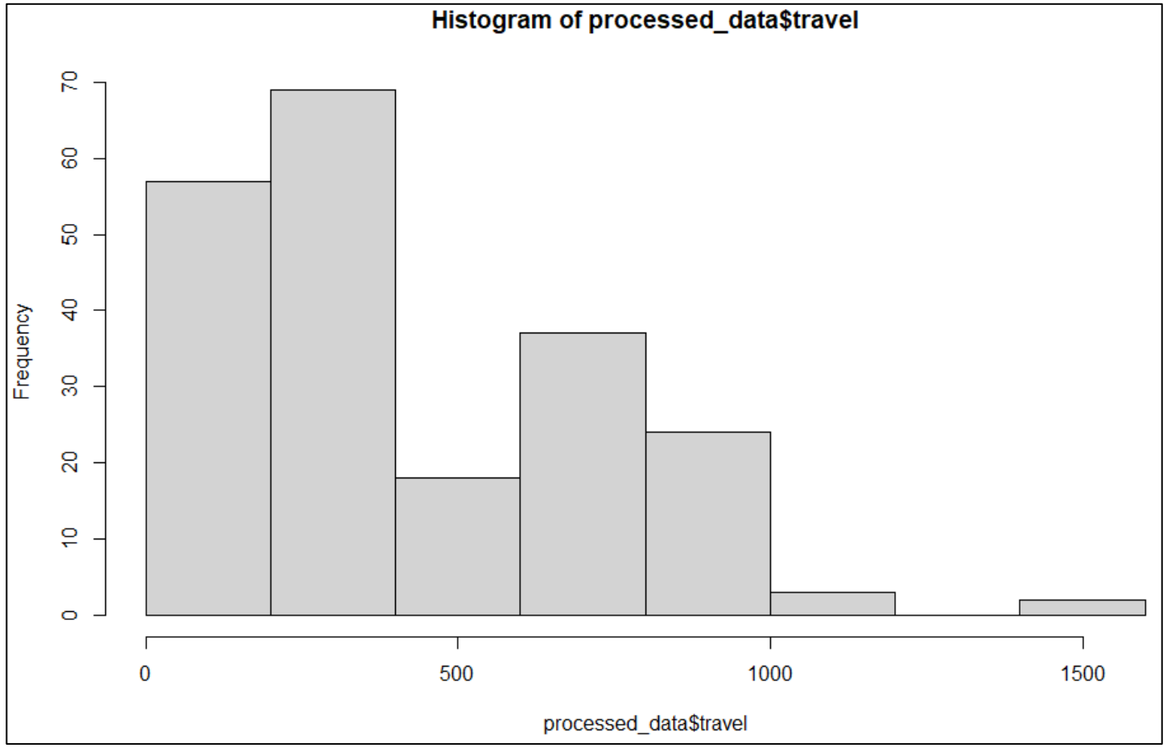
```
#boxplot
boxplot_data <- processed_data[,c(-1, -4, -7)] #mode, travel, size 컬럼 제외
boxplot(boxplot_data)
```



01 실습 - 시각화 실습

Q3-2. travel 컬럼에 대한 히스토그램을 그리세요.

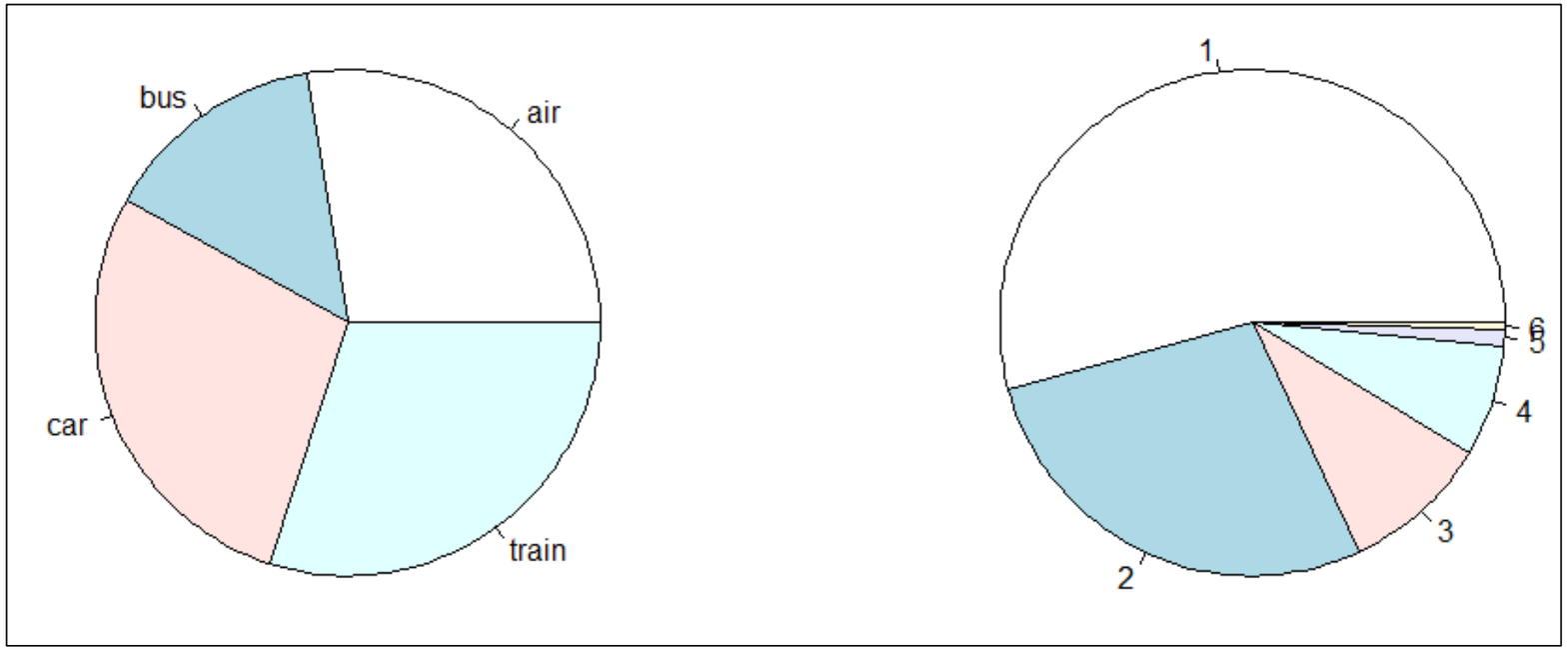
```
#히스토그램
hist(processed_data$travel)
```



01 실습 - 시각화 실습

Q3-3. 화면 분할을 사용하여, mode, size 컬럼에 대한 파이차트를 한 plot에 그리세요.

```
#파이차트
par(mfrow = c(1,2))
pie(table(processed_data$mode))
pie(table(processed_data$size))
```



24차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

25차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

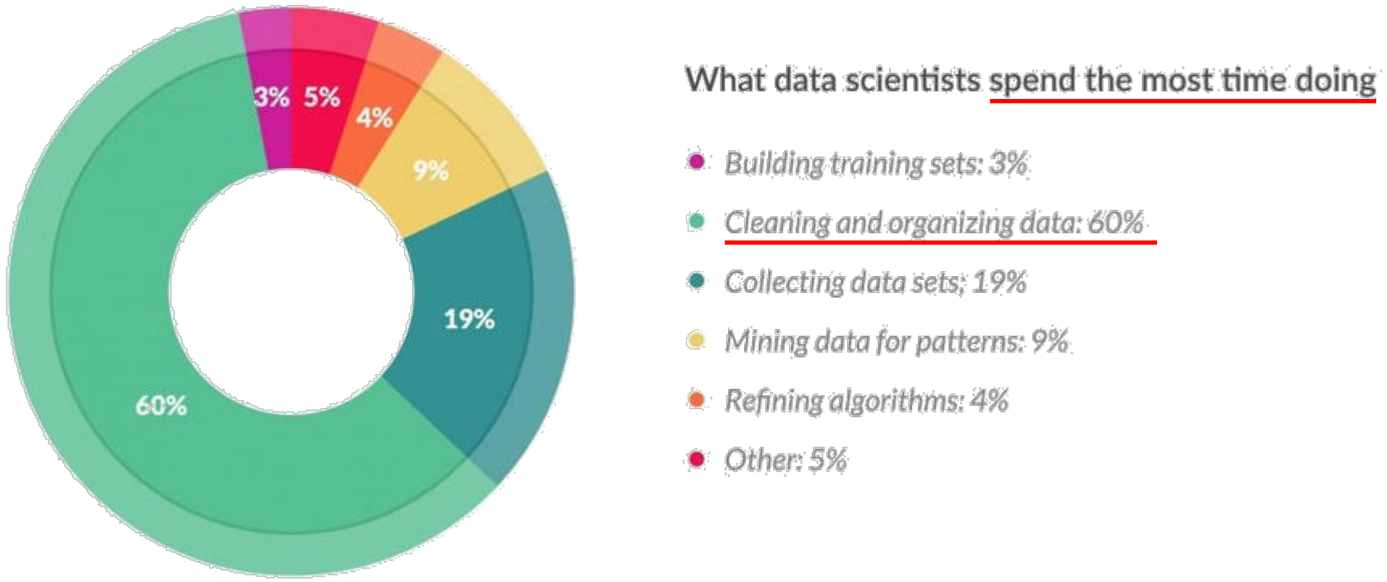
- 파이프 연산자를 이해할 수 있다.
- Dplyr의 간단한 함수를 알 수 있다.

학습 목차

- dplyr 패키지 소개
- 파이프 연산자 이해
- dplyr 간단한 함수 소개

01 dplyr 패키지 소개 – 왜 중요한가요?

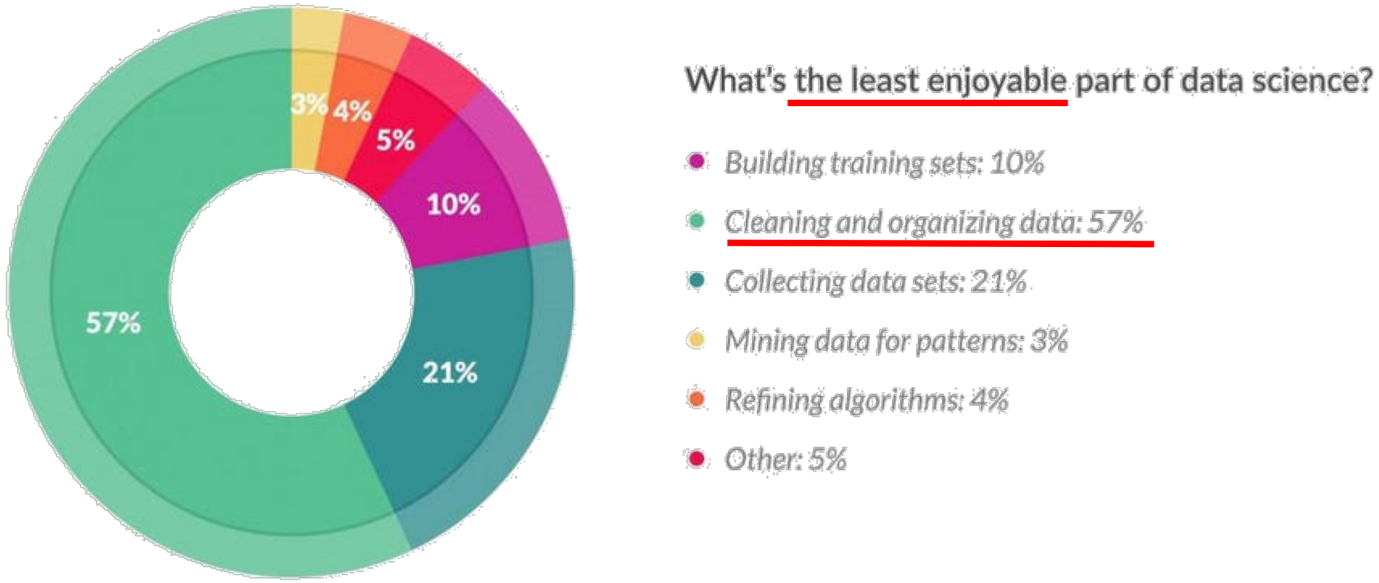
- 현실의 데이터는 더럽다
- 전처리에 많은 시간 사용



출처: https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/?sh=4416c6936f63 (포브스)

01 dplyr 패키지 소개 – 왜 중요한가요?

- 현실의 데이터는 더럽다
- 전처리에 많은 시간 사용 + 힘듦



출처: https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/?sh=4416c6936f63 (포브스)

01 dplyr 패키지 소개 – 왜 중요한가요?

- **현실의 데이터는 더럽다**
- **전처리에 많은 시간 사용 + 힘듦**
- **Why?**
  - ➢ 비교적 단순한 반복 작업이 많기 때문

01 dplyr 패키지 소개 – 왜 중요한가요?

- **전처리를 쉽게해주는 R 패키지**
- **데이터 전처리 시, 반복작업을 줄여줌**
- **처리 속도 빠름**
  - C++기반
  - 이전 버전인 plyr은 R 기반 패키지
    - 느림
- **파이프 연산자 제공**
  - 가독성
  - 메모리 효율성

01 dplyr 패키지 소개 – 설치

- **install.packages() 함수 사용**

```
install.packages("dplyr")   # 패키지 설치
library(dplyr)              # 패키지 로딩
```

02 파이프 연산자 – 소개

- **문법**

```
a <- c(10,22,33)
a %>% mean      # 21.66667.  mean(a)와 동일
```

- 형태: %>%
- 의미: 왼쪽 변수를 오른쪽 함수에 적용
  - ‘f(x)’ 구조를 ‘x %>% f’ 구조로 사용
  - 왼쪽 변수는 오른쪽 함수의 첫번째 파라미터로 적용
- dplyr 패키지에서 제공하는 연산자
- a.k.a 체인 연산자(chain operator)

A를	전달	받아서 B를 처리	결과를 전달	받아서 C를 처리	결과를 전달	받아서 D를 처리
A		B	%>%	C	%>%	D

02 파이프 연산자 – 예시

- 예시1: 데이터의 평균을 구하고, 평균 값 소수점 첫번째 자리에서 반올림

- 기본 R 코드: 값 대입 사용

```
a <- c(10,22,33)
mean_a <- mean(a)                    #앞으로 사용하지 않을 임시변수를 메모리에 할당하여 메모리 손해
round(mean_a, 1)                      #21.7
```

- 기본 R코드: 함수 중첩 사용

```
a <- c(10,22,33)
round(mean(a), 1)
```

- 파이프 연산자(%>%) 사용

```
a <- c(10,22,33)
a %>% mean %>% round(1)
```

02 파이프 연산자 – 예시

- 예시2: a에 대한 $(round(\sqrt{a}+0.2)-1)/5$ 값에 “%” 붙힌 후 출력

- 기본 R 코드: 값 대입 사용

```
a <- c(10,22,33)
sqrt_a <- sqrt(a)
sqrt_a_plus_0.2 <- sqrt_a + 0.2
round_a <- round(sqrt_a_plus_0.2, 1)
round_a_minus_1 = round_a -1
div_a = round_a_minus_1 /5
paste_a <- paste(div_a, "%", sep="")
print(paste_a)        # "0.48%" "0.78%" "0.98%"
```

- 기본 R코드: 함수 중첩 사용

```
a <- c(10,22,33)
print( paste( (round( sqrt(a) + 0.2 , 1) – 1) / 5, "%", sep=""))
```

- 파이프 연산자(%>%) 사용

```
a <- c(10,22,33)
div <- function(x,y){  return(x/y) }   #파이프연산자에서 나눗셈, 곱셈 연산자(/, *) 사용불가
a %>% sqrt() %>% + 0.2 %>% round(1) %>% -1 %>% div(5) %>% paste("%", sep="") %>% print()
```

02 파이프 연산자 – 정리

- **값 조작을 위해 사용하는 함수가 많을 수록 가독성 측면에서 장점**
  - 전처리 과정이 복잡할 수록 코드 작성 및 이해가 편함

- **단축키**
  - Ctrl + Shift + M
  - 단축키로 파이프 연산자 빠르게 타이핑

03 dplyr 함수 소개 – rename 함수

- **데이터 프레임에서 특정 컬럼의 이름 변경**

- **문법**

```
rename(data, 변경 후 컬럼명1=기존 컬럼명1, 변경 후 컬럼명2=기존 컬럼명2, … )
```

- **사용**

```
data <- Orange #Orange 데이터셋 사용
data <- rename(data, TreeIdx=Tree)
```

```
> data
   Tree  age circumference
1     1  118            30
2     1  484            58
```

```
> rename(data, TreeIdx=Tree)
   TreeIdx  age circumference
1        1  118            30
2        1  484            58
```

- **기본 R 문법사용 (colnames() 함수 사용)**

```
data <- Orange
print(colnames(data))                          # "Tree"        "age"         "circumference"
Col_index <- which(colnames(data) == "Tree")   #변경하고 싶은 컬럼의 인덱스값 알아야 함
colnames(data)[Col_index] <- "TreeIdx"         #변경 할 값 대입
```

03 dplyr 함수 소개 – arrange 함수

- **데이터 프레임에서 특정 컬럼을 값을 기준으로 데이터 정렬**

- **문법**

```
arrange(data, 컬럼 이름)          #오름차순이 default
arrange(data, desc(컬럼 이름)) # 내림차순
```

- **사용**

```
data <- Orange #Orange 데이터셋 사용
arrange(data, circumference)
arrange(data, circumference, age) %>% tail() #여러 컬럼을 기준으로 정렬가능
arrange(data, age, circumference) %>% tail() #앞에 나오는 컬럼이 우선순위 더 높음
```

```
> arrange(data, circumference, age) %>% tail() #여러 컬럼을 기준으로 정렬가능
   Tree  age circumference
30    5 1582           177
31    4 1231           179
32    2 1372           203
33    2 1582           203
34    4 1372           209
35    4 1582           214
> arrange(data, age, circumference) %>% tail() #앞에 나오는 컬럼이 우선순위 더 높음
   Tree  age circumference
30    4 1372           209
31    3 1582           140
32    1 1582           145
33    5 1582           177
34    2 1582           203
35    4 1582           214
```

03 dplyr 함수 소개 – distinct 함수

- **데이터 프레임에서 특정 컬럼의 중복 값 제거**

- **문법**

```
distinct(data, 컬럼 이름)
```

- **사용**

```
data <- Orange #Orange 데이터셋 사용
distinct(data, age)
```

```
> distinct(data, age)
   age
1  118
2  484
3  664
4 1004
5 1231
6 1372
7 1582
```

25차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

26차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- dplyr 주요 함수를 이해할 수 있다.

학습 목차

- dplyr 주요 함수 소개 및 이해

01 예제 데이터 생성

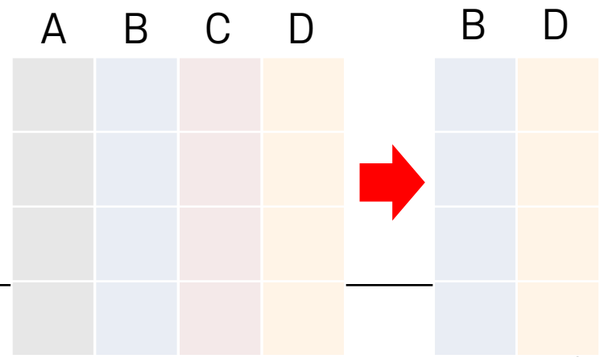
- **임의의 수학, 영어, 과학 성적 데이터 생성**

```
id <- as.character(c(2021001:2021010))
math <- c(100, 54, 36, 76, 54, 94, 15, 6, 34,64)
english <- c(95, 23, 11, 89, 50, 53, 70, 13, 60,90)
science <- c(99, 56, 43, 90, 34, 77, 43, 3, 85,72)
exam <- data.frame(id, math, english, science)
print(exam)
```

```
> exam
        id math english science
1  2021001  100      95      99
2  2021002   54      23      56
3  2021003   36      11      43
4  2021004   76      89      90
5  2021005   54      50      34
6  2021006   94      53      77
7  2021007   15      70      43
8  2021008    6      13       3
9  2021009   34      60      85
10 2021010   64      90      72
```

02 dplyr 주요 함수 소개 – select 함수

- 데이터 프레임에서 원하는 컬럼만 추출



- 문법

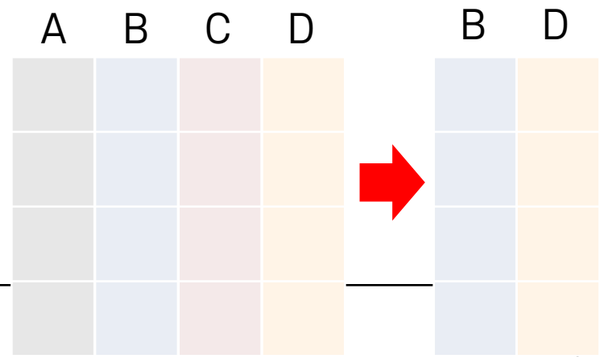
```
select(dataframe, 컬럼명1, 컬럼명2, …)
```

- 예시

```
select(exam, math, science) # math, science 컬럼만 추출
```

- [Quiz] exam 데이터 프레임에서, english, science 컬럼만 추출하세요. 단, science 컬럼이 먼저 출력 되도록 하세요.

02 dplyr 주요 함수 소개 – select 함수



- **데이터 프레임에서 원하는 컬럼만 추출**
- **문법**

```
select(dataframe, 컬럼명1, 컬럼명2, …)
```

- **예시**

```
select(exam, math, science) # math, science 컬럼만 추출
```

- **[Quiz] exam 데이터 프레임에서, english, science 컬럼만 추출하세요. 단, science 컬럼이 먼저 출력 되도록 하세요.**

```
select(exam, science, english)
```

```
> select(exam, science, english)
   science english
1       99      95
2       56      23
3       43      11
4       90      89
5       34      50
6       77      53
7       43      70
8        3      13
9       85      60
10      72      90
```

02 dplyr 주요 함수 소개 – select 함수

- 기본 R 함수와 비교
- 기본 R 함수

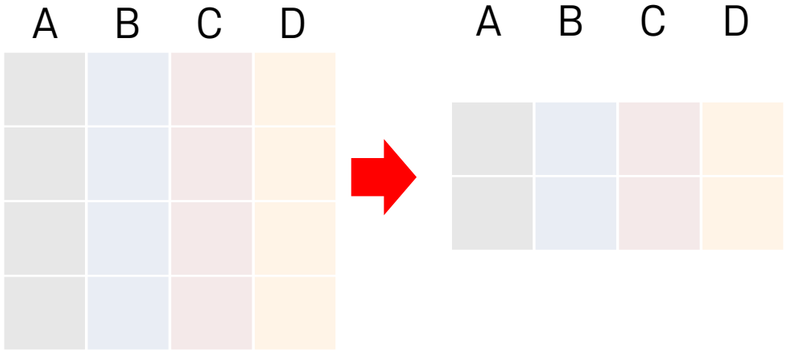
```
exam[, c("math", "science")] # 데이터프레임에서 직접 추출
```

- dplyr 함수

```
select(exam, math, science) # select 함수 이용한 추출
```

02 dplyr 주요 함수 소개 – filter 함수

- 데이터 프레임에서 조건에 충족되는 행만 추출



- 문법

```
filter(dataframe, 조건식)
```

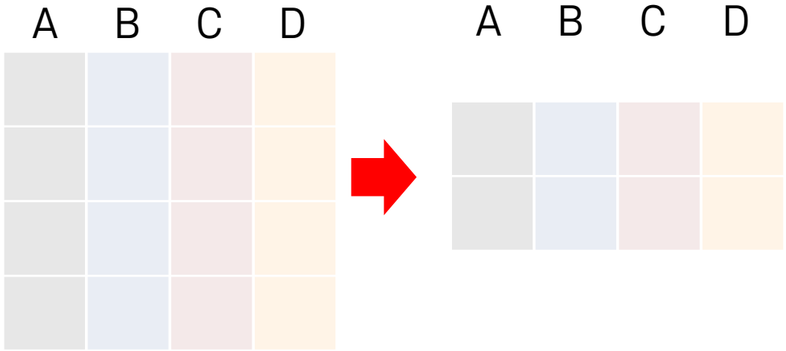
- 예시

```
filter(exam, math > 60)   # 수학 점수가 60점 이상인 학생만 추출
filter(exam, english > 20 & science > 50) # AND(&) 연산자 이용한 추출
filter(exam, english > 20 | science > 50) # OR(|) 연산자 이용한 추출
```

- [Quiz] exam 데이터 프레임에서, 과학점수가 70점 이상이지만 수학점수가 50점 이하인 학생의 아이디를 출력하세요.

02 dplyr 주요 함수 소개 – filter 함수

- 데이터 프레임에서 조건에 충족되는 행만 추출



- 문법

```
filter(dataframe, 조건식)
```

- 예시

```
filter(exam, math > 60)   # 수학 점수가 60점 이상인 학생만 추출
filter(exam, english > 20 & science > 50) # AND(&) 연산자 이용한 추출
filter(exam, english > 20 | science > 50) # OR(|) 연산자 이용한 추출
```

- [Quiz] exam 데이터 프레임에서, 과학점수가 70점 이상이지만 수학점수가 50점 이하인 학생의 아이디를 출력하세요.

```
exam %>% filter(science > 70 & math < 50) %>% select(id) %>% as.character() %>% print()
```

02 dplyr 주요 함수 소개 – filter 함수

- **기본 R 함수와 비교**
  - ➢ Quiz. exam 데이터 프레임에서, 과학점수가 70점 이상이지만 수학점수가 50점 이하인 학생의 아이디를 출력하세요.

- **기본 R 함수**

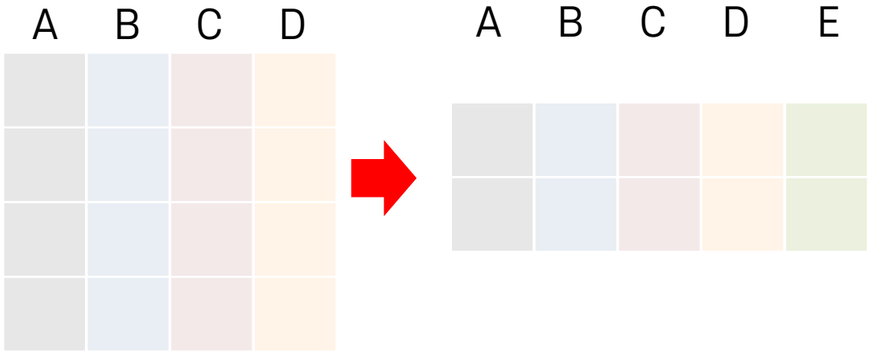
```
selectedIndex <- exam$science>70 & exam$math < 50
print( exam[ selectedIndex, c("id")] ) # 데이터프레임에서 직접 추출
```

- **dplyr 함수**

```
exam %>% filter(science > 70 & math < 50) %>% select(id) %>% as.character() %>% print()
```

02 dplyr 주요 함수 소개 – mutate 함수

- 기존 컬럼값을 사용하여 새로운 컬럼을 생성



- 문법

```
mutate(dataframe, 생성할 컬럼명1 = 수식1, …)
```

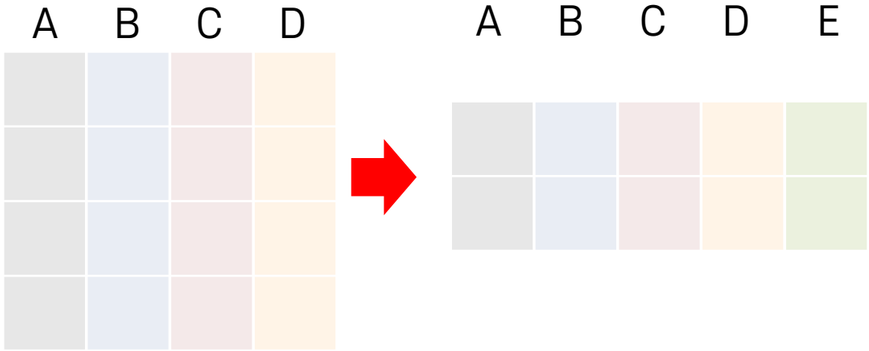
- 예시

```
mutate(exam, average= (math+science+english)/3)
```

- [Quiz] exam 데이터 프레임에 평균점수가 70이상이면 “합격” 값을, 70미만이면 “불합격” 값을 가지는 pass 컬럼을 추가하세요.

02 dplyr 주요 함수 소개 – mutate 함수

- 기존 컬럼값을 사용하여 새로운 컬럼을 생성



- 문법

```
mutate(dataframe, 생성할 컬럼명1 = 수식1, …)
```

- 예시

```
mutate(exam, average= (math+science+english)/3)
```

- [Quiz] exam 데이터 프레임에 평균점수가 70이상이면 “합격” 값을, 70미만이면 “불합격” 값을 가지는 pass 컬럼을 추가하세요.

```
mutate(exam, pass= ifelse( (math+science+english)/3 >= 70, "합격","불합격"))
```

02 dplyr 주요 함수 소개 – mutate 함수

- **기본 R 함수와 비교**
  - ➢ Quiz. exam 데이터 프레임에 평균점수가 70이상이면 “합격” 값을, 70미만이면 “불합격” 값을 가지는 pass 컬럼을 추가하세요.

- **기본 R 함수**

```
exam[,"pass"] <- ifelse( (exam$math + exam$english + exam$science) /3 >= 70, "합격","불합격")
```

- **dplyr 함수**

```
mutate(exam, pass= ifelse( (math+science+english)/3 >= 70, "합격","불합격"))
```

03 예제 데이터 생성 2

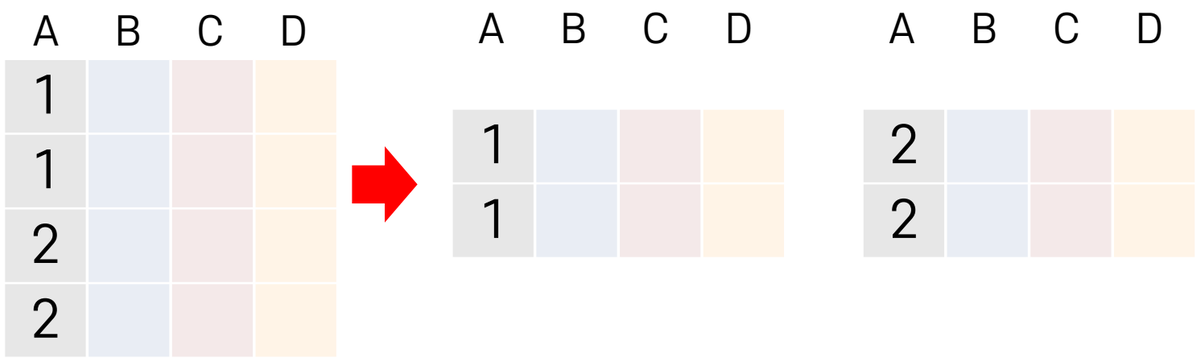
- **임의의 수학, 영어, 과학 성적 데이터 수정**
  - ➢ 기존 값은 중간 고사 점수, 동일한 형식의 기말고사 점수 추가
  - ➢ 3차원 데이터를 2차원으로 표현 한 형태
    - ▪ 10x3x2(중간,기말) → 20x3

```
id <- as.character( rep(c(2021001:2021010), tims=2) )
mid_math <- c(100, 54, 36, 76, 54, 94, 15, 6, 34,64)
final_math <- c(90, 80, 23, 67, 44, 72, 10, 45, 87,55)
math <- c(mid_math, final_math)
mid_english <- c(95, 23, 11, 89, 50, 53, 70, 13, 60,90)
final_english <- c(90, 32, 4, 74, 90, 23, 83, 52, 43,70)
english <- c(mid_english, final_english)
mid_science <- c(99, 56, 43, 90, 34, 77, 43, 3, 85,72)
final_science <- c(100, 79, 25, 65, 63, 75, 73, 66, 50, 83)
science <- c(mid_science, final_science)
examTerm <- rep(c("중간", "기말"), times=c(10,10))
exam2 <- data.frame(id, math, english, science, examTerm)
print(exam2)
```

```
> print(exam)
         id math english science examTerm
1  2021001  100      95      99     중간
2  2021002   54      23      56     중간
3  2021003   36      11      43     중간
4  2021004   76      89      90     중간
5  2021005   54      50      34     중간
6  2021006   94      53      77     중간
7  2021007   15      70      43     중간
8  2021008    6      13       3     중간
9  2021009   34      60      85     중간
10 2021010   64      90      72     중간
11 2021001   90      90     100     기말
12 2021002   80      32      79     기말
13 2021003   23       4      25     기말
14 2021004   67      74      65     기말
15 2021005   44      90      63     기말
16 2021006   72      23      75     기말
17 2021007   10      83      73     기말
18 2021008   45      52      66     기말
19 2021009   87      43      50     기말
20 2021010   55      70      83     기말
```

04 dplyr 주요 함수 소개 – group_by 함수

- **특정 컬럼 값이 같은 데이터들을 같은 집단으로 묶어줌**



- **문법**

```
group_by(exam, 컬럼 명)
```

- **예시**

```
group_by(exam2, id)
```

04 dplyr 주요 함수 소개 – group_by 함수

- **특징**
  - ➢ group_by 만으로는 외관상 따로 달라지는게 없음
  - ➢ ‘grouped_df‘ 형식으로 변경됨
    - ▪ group관련 정보 추가

```
group_by(exam2, id)
```

```
grouped_df [20 x 5] (S3: grouped_df/tbl_df/tbl/data.frame)
 $ id      : chr [1:20] "2021001" "2021002" "2021003" "2021004" ...
 $ math    : num [1:20] 100 54 36 76 54 94 15 6 34 64 ...
 $ english : num [1:20] 95 23 11 89 50 53 70 13 60 90 ...
 $ science : num [1:20] 99 56 43 90 34 77 43 3 85 72 ...
 $ examTerm: chr [1:20] "중간" "중간" "중간" "중간" ...
 - attr(*, "groups")= tibble [10 x 2] (S3: tbl_df/tbl/data.frame)
  ..$ id   : chr [1:10] "2021001" "2021002" "2021003" "2021004" ...
  ..$ .rows: list<int> [1:10]
  .. ..$ : int [1:2] 1 11
  .. ..$ : int [1:2] 2 12
  .. ..$ : int [1:2] 3 13
  .. ..$ : int [1:2] 4 14
  .. ..$ : int [1:2] 5 15
  .. ..$ : int [1:2] 6 16
  .. ..$ : int [1:2] 7 17
  .. ..$ : int [1:2] 8 18
  .. ..$ : int [1:2] 9 19
  .. ..$ : int [1:2] 10 20
  .. ..@ ptype: int(0)
  ..- attr(*, ".drop")= logi TRUE
```

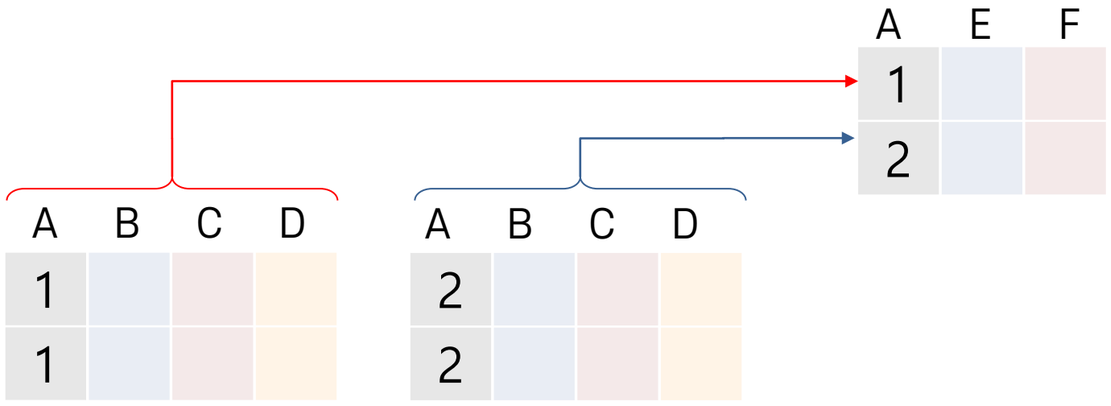
그룹 기준인 id의 범주

id 범주 크기의 list 생성
- list 크기 == group의 개수

같은 그룹인 데이터의 행

04 dplyr 주요 함수 소개 – summarize 함수

- 그룹 단위 동일기능 수행 후, 새로운 데이터 프레임에 수행 결과 통합



- 문법

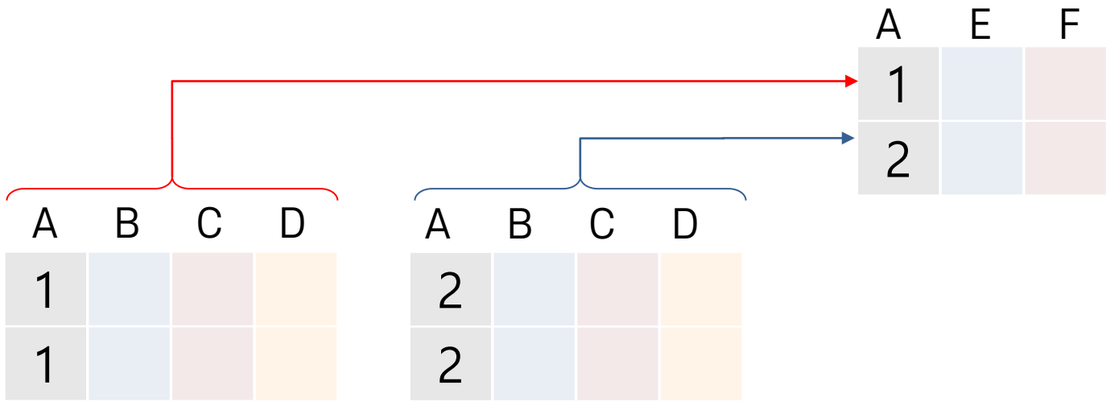
```
summarize(grouped_df, 컬럼 명1=계산 식1, …)
```

- 예시

```
group_exam <- group_by(exam2, id)
#학생 별, 평균 수학, 영어
summarize(group_exam, mathAvg=mean(math), englishAvg=mean(english))
```

04 dplyr 주요 함수 소개 – summarize 함수

- 그룹 단위 동일기능 수행 후, 새로운 데이터 프레임에 수행 결과 통합




```
# A tibble: 10 x 3
   id      mathAvg englishAvg
   <chr>     <dbl>      <dbl>
 1 2021001    95         92.5
 2 2021002    67         27.5
 3 2021003    29.5        7.5
 4 2021004    71.5       81.5
 5 2021005    49         70
 6 2021006    83         38
 7 2021007    12.5       76.5
 8 2021008    25.5       32.5
 9 2021009    60.5       51.5
10 2021010    59.5       80
```

- 문법

```
summarize(grouped_df, 컬럼 명1=계산 식1, …)
```

- 예시

```
group_exam <- group_by(exam2, id)
#학생 별, 평균 수학, 영어
summarize(group_exam, mathAvg=mean(math), englishAvg=mean(english))
```

04 dplyr 주요 함수 소개 – summarize 함수

- **Quiz. 데이터 프레임 exam2의 'examTerm' 단위로 그룹을 묶고, 중간/기말 고사의 평균 수학 점수를 구하세요.**

```
group_exam <- group_by(exam2, examTerm)
#중간/기말 고사 별, 평균 수학 점수
summarize(group_exam, mathAvg=mean(math))
```

26차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

27차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 새로운 데이터셋 탐색 및 분석을 할 수 있다.
- dplyr 기반 데이터 테이블 조작을 할 수 있다.
- 정보 시각화를 할 수 있다.

학습 목차

- 새로운 데이터셋 탐색 및 분석
- dplyr 기반 데이터 테이블 조작
- 정보 시각화

01 데이터셋 소개 – gapminder

- **국가별 경제 수준과 의료 수준 동향을 정리한 데이터셋**
  - 경제적 수준: 일인당 GDP
  - 의료 수준: 평균 기대수명

- **데이터셋 로딩**
  - gapminder 패키지 설치 필요

```
install.packages("gapminder")
library(gapminder)
library(dplyr)      # dplyr 패키지도 로딩
```

01 데이터셋 소개 – gapminder

- **데이터 기본 정보 확인**
  - ➢ 나라별로 12개 데이터 존재

```
#데이터 확인
str(gapminder)
?gapminder
table(gapminder$continent)
table(gapminder$country)
```

01 데이터셋 소개 – gapminder

- **데이터 기본 정보 확인**
  - ➢ 나라별로 12개 데이터 존재

```
#데이터 확인
str(gapminder)
?gapminder
table(gapminder$continent)
table(gapminder$country)
```

```
> str(gapminder)
tibble [1,704 x 6] (S3: tbl_df/tbl/data.frame)
 $ country  : Factor w/ 142 levels "Afghanistan",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ continent: Factor w/ 5 levels "Africa","Americas",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ year     : int [1:1704] 1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...
 $ lifeExp  : num [1:1704] 28.8 30.3 32 34 36.1 ...
 $ pop      : int [1:1704] 8425333 9240934 10267083 11537966 13079460 14880372 12
 $ gdpPercap: num [1:1704] 779 821 853 836 740 ...
```

01 데이터셋 소개 – gapminder

- **데이터 기본 정보 확인**
  - ➢ 나라별로 12개 데이터 존재

```
#데이터 확인
str(gapminder)
?gapminder
table(gapminder$continent)
table(gapminder$country)
```

```
> str(gapminder)
tibble [1,704 x 6] (S3: tbl_df/tbl/data.frame)
 $ country  : Factor w/ 142 levels "Afghanistan",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ continent: Factor w/ 5 levels "Africa","Americas",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ year     : int [1:1704] 1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...
 $ lifeExp  : num [1:1704] 28.8 30.3 32 34 36.1 ...
 $ pop      : int [1:1704] 8425333 9240934 10267083 11537966 13079460 14880372 12
 $ gdpPercap: num [1:1704] 779 821 853 836 740 ...
```

Format

The main data frame `gapminder` has 1704 rows and 6 variables:

country

factor with 142 levels

continent

factor with 5 levels

year

ranges from 1952 to 2007 in increments of 5 years

lifeExp

life expectancy at birth, in years

pop

population

gdpPercap

GDP per capita (US$, inflation-adjusted)

01 데이터셋 소개 – gapminder

- **데이터 기본 정보 확인**
  - ➢ 나라별로 12개 데이터 존재

```
#데이터 확인
str(gapminder)
?gapminder
table(gapminder$continent)
table(gapminder$country)
```

```
> str(gapminder)
tibble [1,704 x 6] (S3: tbl_df/tbl/data.frame)
 $ country  : Factor w/ 142 levels "Afghanistan",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ continent: Factor w/ 5 levels "Africa","Americas",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ year     : int [1:1704] 1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...
 $ lifeExp  : num [1:1704] 28.8 30.3 32 34 36.1 ...
 $ pop      : int [1:1704] 8425333 9240934 10267083 11537966 13079460 14880372 12
 $ gdpPercap: num [1:1704] 779 821 853 836 740 ...
```

```
> table(gapminder$continent)

  Africa Americas     Asia   Europe  Oceania
     624      300      396      360       24
> table(gapminder$country)

             Afghanistan                 Albania                 Algeria                  Angola
                      12                      12                      12                      12
               Argentina               Australia                 Austria                 Bahrain
                      12                      12                      12                      12
              Bangladesh                 Belgium                   Benin                 Bolivia
                      12                      12                      12                      12
  Bosnia and Herzegovina                Botswana                  Brazil                Bulgaria
                      12                      12                      12                      12
            Burkina Faso                 Burundi                Cambodia                Cameroon
                      12                      12                      12                      12
                  Canada Central African Republic                    Chad                   Chile
                      12                      12                      12                      12
```

Format

The main data frame `gapminder` has 1704 rows and 6 variables:

country

factor with 142 levels

continent

factor with 5 levels

year

ranges from 1952 to 2007 in increments of 5 years

lifeExp

life expectancy at birth, in years

pop

population

gdpPercap

GDP per capita (US$, inflation-adjusted)

01 데이터셋 소개 – gapminder

- **데이터 기본 통계값 분석**

```
#대륙별 평균 GDP 계산
mean(pull(gapminder[gapminder$continent == "Africa", "gdpPercap"]))
mean(pull(gapminder[gapminder$continent == "Americas", "gdpPercap"]))
mean(pull(gapminder[gapminder$continent == "Asia", "gdpPercap"]))
mean(pull(gapminder[gapminder$continent == "Europe", "gdpPercap"]))
mean(pull(gapminder[gapminder$continent == "Oceania", "gdpPercap"]))

#한국 평균 GDP 확인
mean(pull(gapminder[gapminder$country == “Korea, Rep.", "gdpPercap"]))
```

➢ pull(): tibble 자료형을 벡터 또는 데이터 프레임 자료형으로 형 변환하는 함수

01 데이터셋 소개 – gapminder

- **데이터 기본 통계값 분석(filter, select 함수 사용)**

```
#대륙별 평균 GDP 계산
gapminder %>% filter(continent == "Africa") %>% select(gdpPercap) %>% pull() %>% mean()
gapminder %>% filter(continent == "Americas") %>% select(gdpPercap) %>% pull() %>% mean()
gapminder %>% filter(continent == "Asia") %>% select(gdpPercap) %>% pull() %>% mean()
gapminder %>% filter(continent == “Europe") %>% select(gdpPercap) %>% pull() %>% mean()
gapminder %>% filter(continent == “Oceania") %>% select(gdpPercap) %>% pull() %>% mean()

#한국 평균 GDP 확인
gapminder %>% filter(country == "Korea, Rep.") %>% select(gdpPercap) %>% pull() %>% mean()
```

➢ pull(): tibble 자료형을 벡터 또는 데이터 프레임 자료형으로 형 변환하는 함수

01 데이터셋 소개 – gapminder

Q1. group_by, summarize 함수를 사용하여 대륙별 평균 GDP 테이블을 생성하세요.

```
#대륙별 평균 GDP 계산
gdp_bycontinents <- gapminder %>% group_by(continent) %>%
                        summarize(mean_gdpPercap=mean(gdpPercap))
```

```
# A tibble: 5 x 2
  continent mean_gdpPercap
  <fct>              <dbl>
1 Africa             2194.
2 Americas           7136.
3 Asia               7902.
4 Europe            14469.
5 Oceania           18622.
```

01 데이터셋 소개 - gapminder

Q2. 아메리카대륙에 대한 데이터 프레임을 구하고, 총 몇 개의 행이 있는지 확인하세요.

01 데이터셋 소개 - gapminder

Q2. 아메리카대륙에 대한 데이터 프레임을 구하고, 총 몇 개의 행이 있는지 확인하세요.

```
America <- gapminder %>% filter(continent == "Americas")
America %>% nrow()                #300
```

01 데이터셋 소개 – gapminder

Q3. 아메리카대륙 인구가 3천만 이상인 데이터 프레임을 구하고, 해당 데이터 프레임의 'country' 컬럼 별 행 개수를 구하세요.

- count 함수 사용
  - count(data, 컬럼 명, sort=F): data에서 해당 컬럼 명의 요소 별 개수 리턴

01 데이터셋 소개 – gapminder

Q3. 아메리카대륙 인구가 3천만 이상인 데이터 프레임을 구하고, 해당 데이터 프레임의 'country' 컬럼 별 행 개수를 구하세요.

- count 함수 사용
  - count(data, 컬럼 명, sort=F): data에서 해당 컬럼 명의 요소 별 개수 리턴

```
America %>% filter(pop >= 30000000) %>% count(country, sort=T)
```

```
# A tibble: 6 x 2
  country           n
  <fct>         <int>
1 Brazil           12
2 Mexico           12
3 United States    12
4 Argentina         5
5 Colombia          5
6 Canada            3
```

01 데이터셋 소개 - gapminder

Q4. Q3에서 확인한 결과, 아메리카대륙 중 55년 내내 인구 3천만명이상인 나라는 3곳입니다. Brazil, Mexico, United States의 년도 별 인구수를 한 그래프에 나타내세요.

- plot의 ylim값은 c(세나라 인구 최소 값, 세나라 인구 최대 값)이 되도록 설정하세요.
- 나라별로 다른 그래프 색을 가지도록 설정하세요.
- 범례를 추가하세요

01 데이터셋 소개 - gapminder

Q4. Q3에서 확인한 결과, 아메리카대륙 중 55년 내내 인구 3천만명이상인 나라는 3곳입니다. Brazil, Mexico, United States의 년도 별 인구수를 한 그래프에 나타내세요.

- plot의 ylim값은 c(세나라 인구 최소 값, 세나라 인구 최대 값)이 되도록 설정하세요.
- 나라별로 다른 그래프 색을 가지도록 설정하세요.
- 범례를 추가하세요

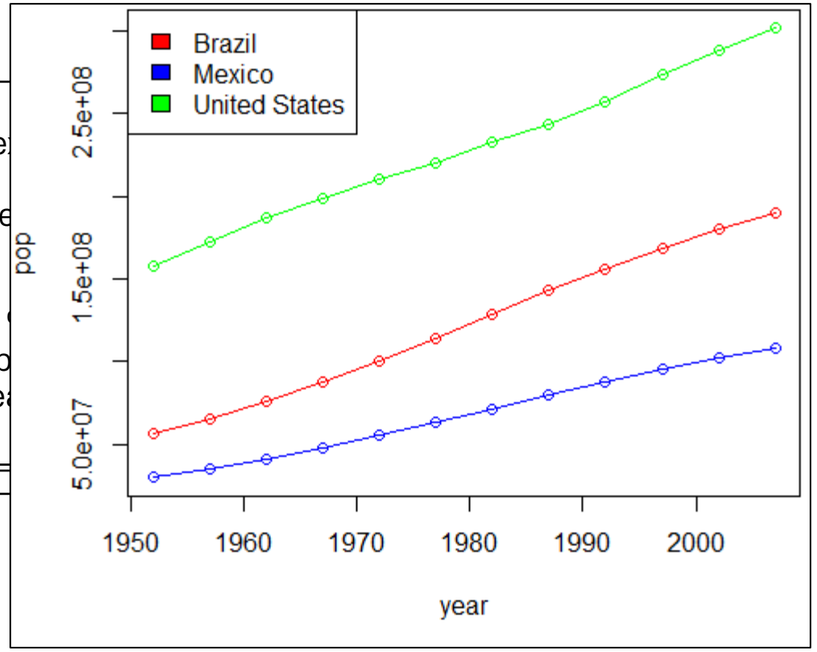
```
#min, max 값 계산
min <- gapminder %>% filter(country == "Brazil" | country == "Mexico" | country == "United States") %>%
select(pop) %>% min()
max <- gapminder %>% filter(country == "Brazil" | country == "Mexico" | country == "United States") %>%
select(pop) %>% max()
#그래프 그리기
gapminder %>% filter(country == "Brazil") %>% select(year, pop) %>% plot(type='o', col="red", ylim=c(min,max))
gapminder %>% filter(country == "Mexico") %>% select(year, pop) %>% lines(type='o', col="blue")
gapminder %>% filter(country == "United States") %>% select(year, pop) %>% lines(type='o', col="green")
#범례 그리기
legend("topleft",legend=c("Brazil", "Mexico", "United States"), fill=c("red", "blue","green"))
```

01 데이터셋 소개 – gapminder

Q4. Q3에서 확인한 결과, 아메리카대륙 중 55년 내내 인구 3천만명이상인 나라는 3곳입니다. Brazil, Mexico, United States의 년도 별 인구수를 한 그래프에 나타내세요.

- plot의 ylim값은 c(세나라 인구 최소 값, 세나라 인구 최대 값)이 되도록 설정하세요.
- 나라별로 다른 그래프 색을 가지도록 설정하세요.
- 범례를 추가하세요

```
#min, max 값 계산
min <- gapminder %>% filter(country == "Brazil" | country == "Me
select(pop) %>% min()
max <- gapminder %>% filter(country == "Brazil" | country == "Me
select(pop) %>% max()
#그래프 그리기
gapminder %>% filter(country == "Brazil") %>% select(year, pop)
gapminder %>% filter(country == "Mexico") %>% select(year, pop
gapminder %>% filter(country == "United States") %>% select(ye
#범례 그리기
legend("topleft",legend=c("Brazil", "Mexico", "United States"), fill=
```



27차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

28차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- dplyr 패키지 관련 Quiz를 풀 수 있다.

학습 목차

- dplyr 패키지 관련 Quiz
- 실습

01 Quiz. dplyr

Q1. 다음 중 dplyr의 장점이 아닌 것은?

① 데이터 전처리를 보다 쉽게 할 수 있도록 도와준다.

② C++ 기반 패키지이며, 처리속도가 빠르다.

③ 가독성 향상에 도움을 주는 파이프 연산자를 제공한다.

④ 다양한 시각화 툴을 제공한다.

Q2. dplyr 함수인 ( ) 함수는 데이터 프레임에서 특정 컬럼의 중복 값을 제거하는 기능을 가진다.

① arrange ② distinct ③ rename ④ count

Q3. dplyr 함수 중 ( ) 함수는 데이터 프레임에서 원하는 컬럼만 추출할 수 하는 기능을 가진다.

① filter ② select ③ group_by ④ summarize ⑤ mutate

01 Quiz. dplyr

Q1. 다음 중 dplyr의 장점이 아닌 것은?

① 데이터 전처리를 보다 쉽게 할 수 있도록 도와준다.

② C++ 기반 패키지이며, 처리속도가 빠르다.

③ 가독성 향상에 도움을 주는 파이프 연산자를 제공한다.

④ 다양한 시각화 툴을 제공한다.

Q2. dplyr 함수인 ( ) 함수는 데이터 프레임에서 특정 컬럼의 중복 값을 제거하는 기능을 가진다.

① arrange ② distinct ③ rename ④ count

Q3. dplyr 함수 중 ( ) 함수는 데이터 프레임에서 원하는 컬럼만 추출할 수 하는 기능을 가진다.

① filter ② select ③ group_by ④ summarize ⑤ mutate

01 Quiz. dplyr

Q4. dplyr 함수인 ( ) 기존에 있는 데이터를 사용하여 새롭게 열 컬럼을 생성하는 기능을 가진다.

① filter ② select ③ group_by ④ summarize ⑤ mutate

Q5. dplyr 함수 중 ( ) 함수는 그룹 단위 동일기능 수행 후, 새로운 데이터 프레임에 수행 결과 통합하는 기능을 가진다.

① filter ② select ③ group_by ④ summary ⑤ mutate

01 Quiz. dplyr

Q4. dplyr 함수인 ( ) 기존에 있는 데이터를 사용하여 새롭게 열 컬럼을 생성하는 기능을 가진다.

① filter ② select ③ group_by ④ summarize ⑤ mutate

Q5. dplyr 함수 중 ( ) 함수는 그룹 단위 동일기능 수행 후, 새로운 데이터 프레임에 수행 결과 통합하는 기능을 가진다.

① filter ② select ③ group_by ④ summary ⑤ mutate

02 실습 – 데이터 확인 – TravelMode 데이터셋

Q1. ‘TravelMode.csv‘ 데이터를 로딩하고 데이터를 확인하세요.

- head 함수사용하여 데이터 확인
- 데이터 크기 확인

```
library(dplyr)

data <- read.csv(‘TravelMode.csv’)
head(data)
dim(data)   # (840 10)
```

```
> head(data)
  X individual  mode choice wait vcost travel gcost income size
1 1          1   air     no   69    59    100    70     35    1
2 2          1 train     no   34    31    372    71     35    1
3 3          1   bus     no   35    25    417    70     35    1
4 4          1   car    yes    0    10    180    30     35    1
5 5          2   air     no   64    58     68    68     30    2
6 6          2 train     no   44    31    354    84     30    2
>
```

- 각 가정(individual)은 4개의 교통수단(mode) 중 하나를 선택(choice)
- 대기시간(wait), 교통비용(vcost), 목적지 도착시간(travel), 소득(income), 가정 구성원 수(size)를 고려하여 산정된 총비용(gcost) 존재

02 실습 – 데이터 확인 – TravelMode 데이터셋

Q2. 교통수단(mode) 별로 총 비용(gcost) 평균 값을 구하세요.

- group_by, summarize 함수를 사용하세요.

02 실습 – 데이터 확인 – TravelMode 데이터셋

Q2. 교통수단(mode) 별로 총 비용(gcost) 평균 값을 구하세요.

- group_by, summarize 함수를 사용하세요.

```
data %>%
  group_by(mode) %>%
  summarize(gcostAgv=mean(gcost))
```

02 실습 – 데이터 확인 – TravelMode 데이터셋

Q2. 교통수단(mode) 별로 총 비용(gcost) 평균 값을 구하세요.

- group_by, summarize 함수를 사용하세요.

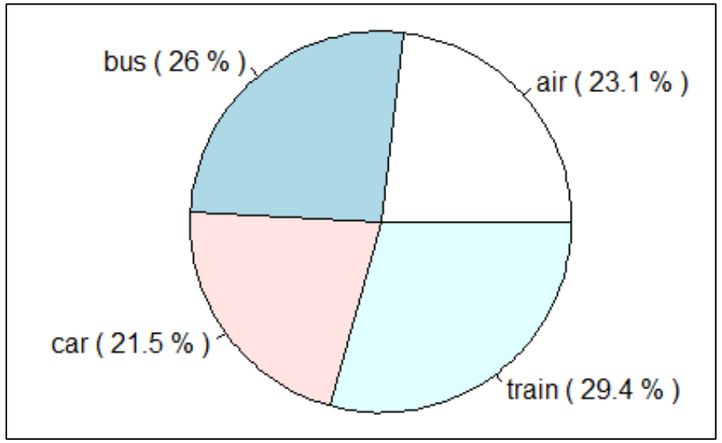
```
data %>%
  group_by(mode) %>%
  summarize(gcostAgv=mean(gcost))
```

```
# A tibble: 4 x 2
  mode  gcostAgv
  <chr>    <dbl>
1 air      103.
2 bus      115.
3 car       95.4
4 train    130.
```

02 실습 – 데이터 확인 – TravelMode 데이터셋

Q3. 교통수단(mode) 별 총 비용(gcost) 평균 값을 파이차트로 그리세요.

- label 값에 각 항목이 차지하는 비율이 들어가도록 그리세요.

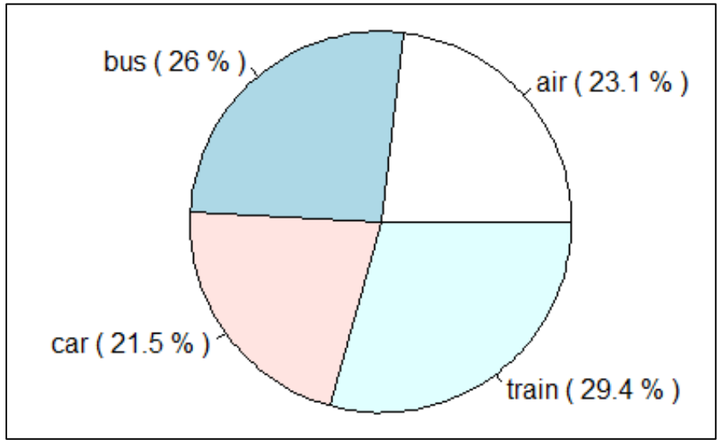


02 실습 – 데이터 확인 – TravelMode 데이터셋

Q3. 교통수단(mode) 별 총 비용(gcost) 평균 값을 파이차트로 그리세요.

- label 값에 각 항목이 차지하는 비율이 들어가도록 그리세요.

```
avg_table <- data %>%  group_by(mode) %>%
                        summarize(gcostAgv=mean(gcost))

avg_table <- avg_table %>%   # avg_table는 Q2의 결과 table 저장한 변수
  as.data.frame()
label <- paste(avg_table$mode, "(", round(avg_table$gcostAgv / sum(avg_table$gcostAgv) *100,1), "% )" )
pie(avg_table$gcostAgv,labels = label)
```



28차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

29차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- ddplot2를 설치할 수 있다.
- ddplot2의 기본 문법을 알 수 있다.
- ddplot2의 기본 함수를 사용할 수 있다.

학습 목차

- ddplot2 소개 및 설치
- ddplot2 기본 문법
- ddplot2 기본 함수

01 ggplot2 패키지 소개 – ggplot2란?

- **데이터 시각화 패키지**
  - 기본 패키지보다 우수한 시각화 기능 제공
- **'gg': Grammar of graphics**
  - 그래프 문법 구조 바탕
  - 일관된 규칙에 따라 그래프 생성
- **우수한 사용성**
  - 빠른 시각화를 위한 quick 함수 제공
  - 그래프 문법 기반 자세한 조정 가능
- **통계 시각화 기능 제공**

01 ggplot2 패키지 소개 – 설치

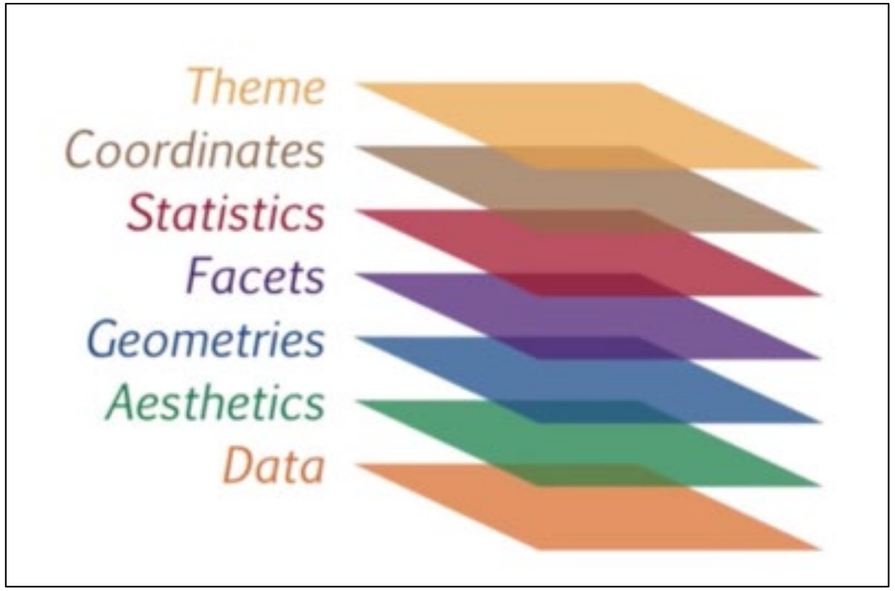
- **install.packages() 함수 사용**

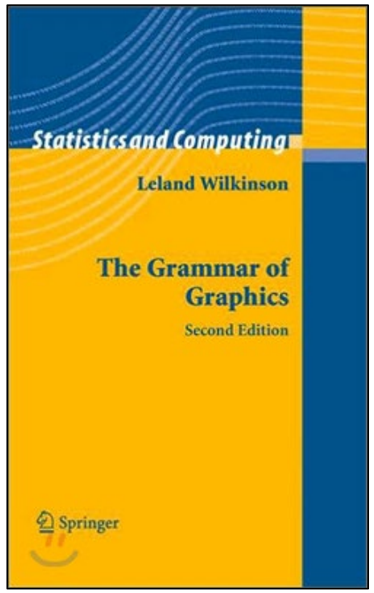
```
install.packages("ggplot2")   # 패키지 설치
library(ggplot2)              # 패키지 로딩
```

02 ggplot2 기본 문법 – Layer 구조

- **Grammar of graphics 저서 기반**
  - ➢ 데이터 시각화는 요소들의 층으로 이루어짐

- **ggplot2 문법 구조**
  - ➢ data layer부터 시작하여 각 층의 요소들을 더해 시각화를 완성하는 방식

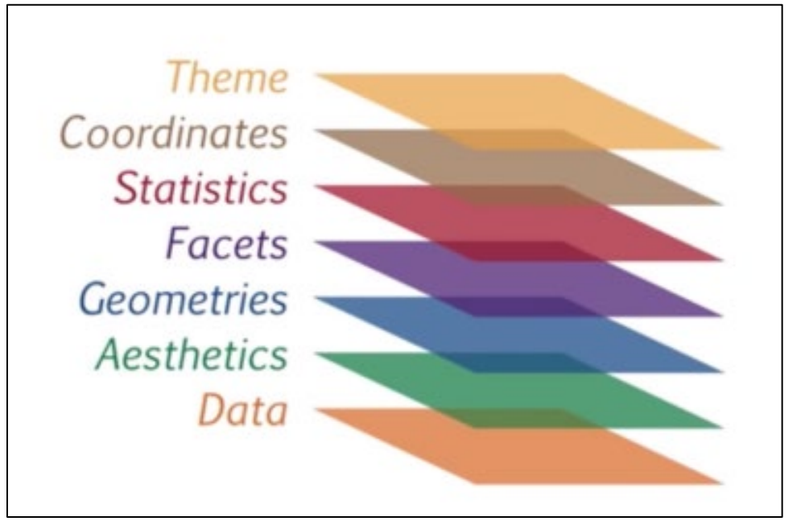




02 ggplot2 기본 문법 – Layer 설명 1

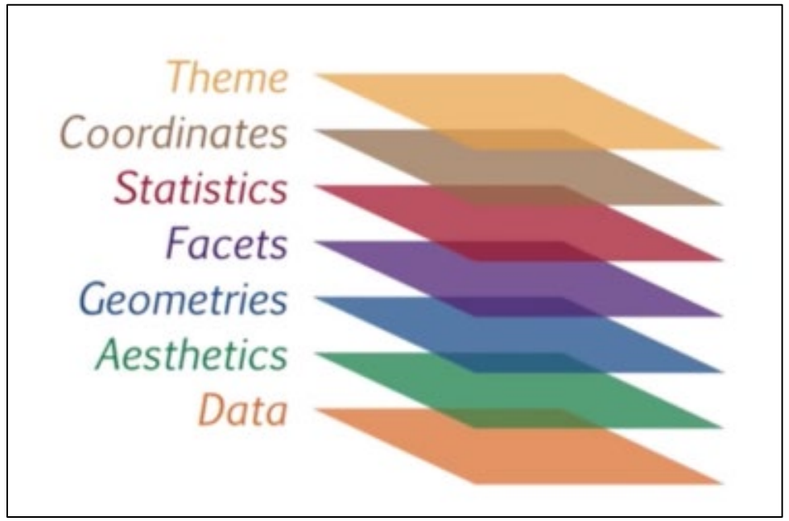
- **Data(데이터) layer**
  - ➢ 시각화를 하려는 데이터를 입력하는 층
  - ➢ data frame 형식 사용

- **Aesthetics(미학적) layer**
  - ➢ 데이터 매핑하고자 하는 스케일 설정
    - ▪ x축 컬럼, y축 컬럼 지정
    - ▪ 색깔
    - ▪ 점/선 모양
    - ▪ 축에 대한 범위
    - ▪ Etc.



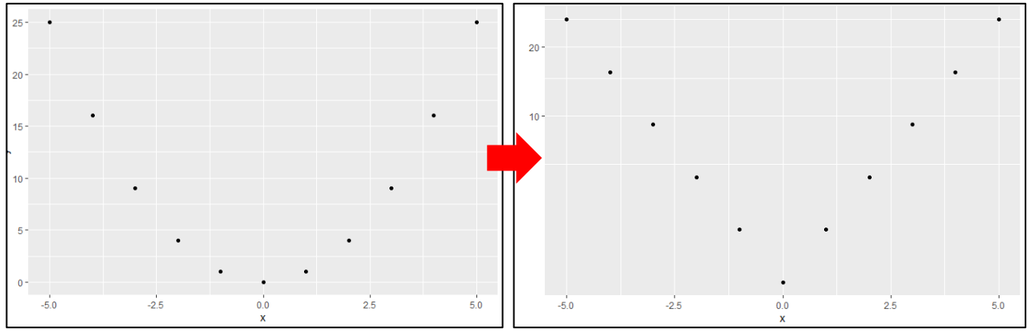
02 ggplot2 기본 문법 – Layer 설명 2

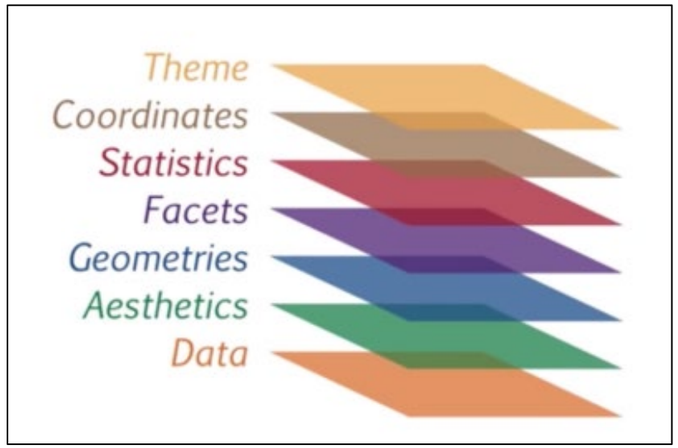
- **Geometries(기하학) layer**
  - ➢ 데이터 사용하는 시각화 요소 정의
    - ▪ 점/선
    - ▪ 히스토그램
    - ▪ 원 차트
    - ▪ 박스그림
    - ▪ Etc.
- **Facets(양상) layer**
  - ➢ 특정 기준에 따라 그래프 나눔
  - ➢ 화면 분할



02 ggplot2 기본 문법 – Layer 설명 3

- **Statistics(통계) layer**
  - ➢ 그려진 그래프에 대한 통계적인 시각화 처리
- **Coordinates(좌표) layer**
  - ➢ 좌표계를 조작
- **Themes(테마) layer**
  - ➢ 데이터가 아닌 추가 꾸미기 요소
    - ▪ 제목
    - ▪ Etc.





03 ggplot2 기본 함수 – qplot과 ggplot

- **qplot()**
  - Quick(q) plot
  - 빠르게 데이터 시각화 할 때 사용
    - 데이터 외의 값을 지정하지 않아도 기본값 사용가능
    - layer별 문법 요소 지정 기능은 존재
  - 기본 시각화 함수(plot)과 유사한 사용법
- **ggplot()**
  - grammar of grapics(gg) plot
  - 그래픽 문법 요소 층 전부, 또는 일부를 직접 쌓아 데이터 시각화
    - '+' 연산자를 사용하여 문법 요소 추가
  - 원하는 모든 형태의 복잡한 시각화 자료 표현 가능

03 ggplot2 기본 함수 – qplot 함수

- **qplot()**
  - Quick(q) plot
  - 빠르게 데이터 시각화 할 때 사용
    - 그래프의 문법 요소 층을 별도로 지정하지 않음
  - 기본 시각화 함수(plot)과 유사한 사용법

- **문법**

```
qplot(data='사용할 데이터 변수명', x='data에서 x축에 사용할 변수', y ='data에서 y축에 사용할 변수')
```

  - 그래픽 문법 요소 층의 2개 층 표현
    - 1층: data layer (데이터 지정)
    - 2층: aesthetics layer (x, y 축 지정)

03 ggplot2 기본 함수 – qplot 함수 – 예시

- **사용 데이터셋**

```
diamond = diamonds #다이아몬드의 품질 값들 및 가격 저장한 데이터셋
str(diamond)          # dplyr 패키지 import 필요
?diamonds             # 다이아몬드 데이터셋 help 페이지 출력
```

Format

A data frame with 53940 rows and 10 variables:

price
: price in US dollars (\$326–\$18,823)

carat
: weight of the diamond (0.2–5.01)

cut
: quality of the cut (Fair, Good, Very Good, Premium, Ideal)

color
: diamond colour, from D (best) to J (worst)

clarity
: a measurement of how clear the diamond is (I1 (worst), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (best))

x
: length in mm (0–10.74)

y
: width in mm (0–58.9)

z
: depth in mm (0–31.8)

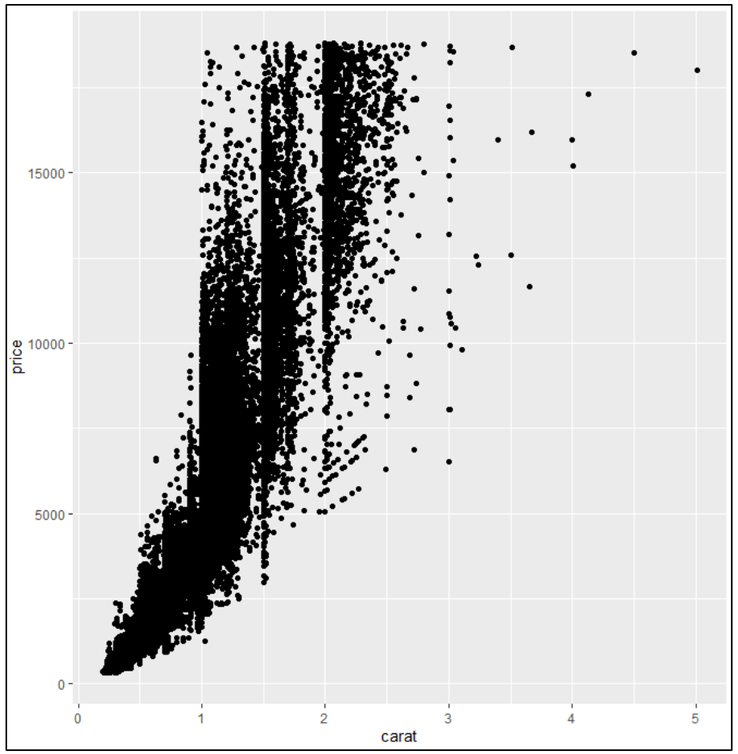
depth
: total depth percentage = z / mean(x, y) = 2 * z / (x + y) (43–79)

table
: width of top of diamond relative to widest point (43–95)

03 ggplot2 기본 함수 – qplot 함수 – 예시

- 예시1

```
qplot(data=diamonds, x=carat, y=price)
```

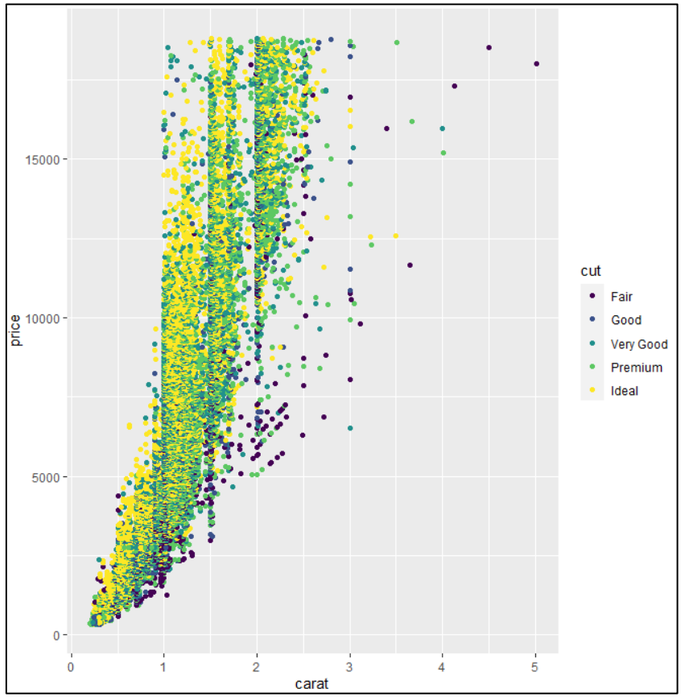


03 ggplot2 기본 함수 – qplot 함수 – 예시

- **예시2**

```
qplot(data=diamonds, x=carat, y=price, aes(col=cut))
```

➢ aes(Aesthetics) 요소 추가 가능

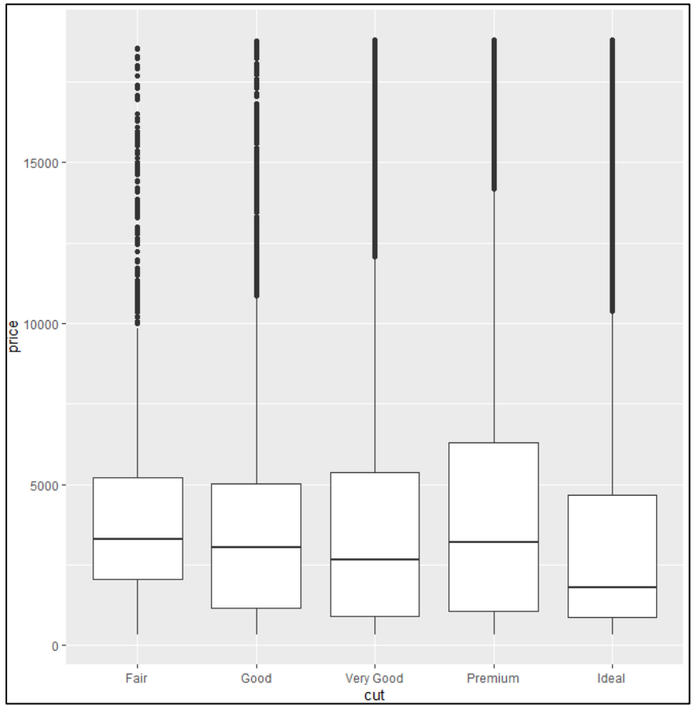


03 ggplot2 기본 함수 – qplot 함수 – 예시

- **예시3**

```
qplot(data=diamonds, x=cut, y=price, geom='boxplot')
```

➢ geom(Geometrics) 요소 추가 가능



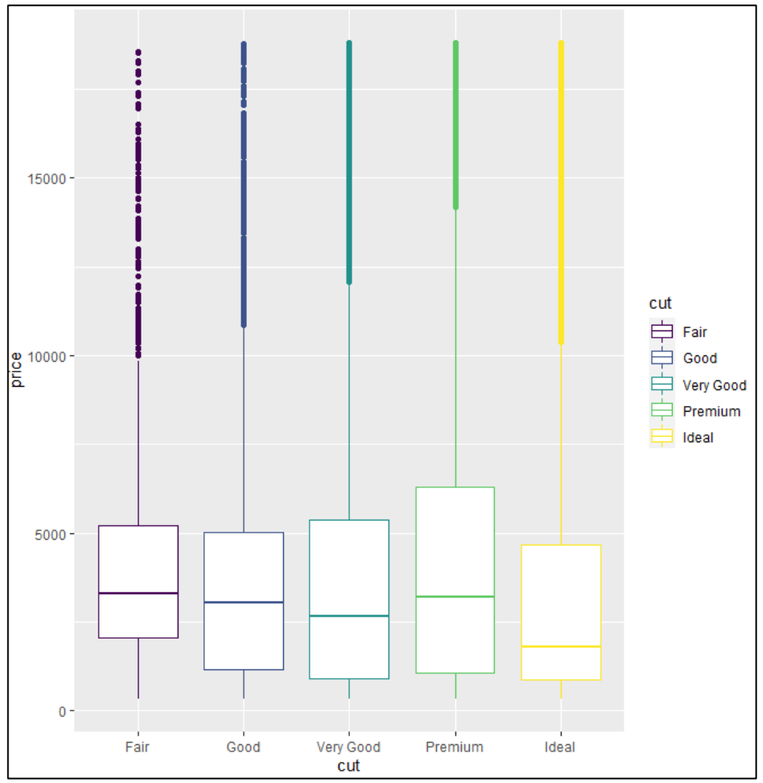
03 ggplot2 기본 함수 – qplot 함수 – 예시

- **예시4**

```
qplot(data=diamonds, x=cut, y=price, aes(col=cut), geom='boxplot')
```

➢ aes(Aesthetics) 요소 추가 가능

➢ geom(Geometrics) 요소 추가 가능



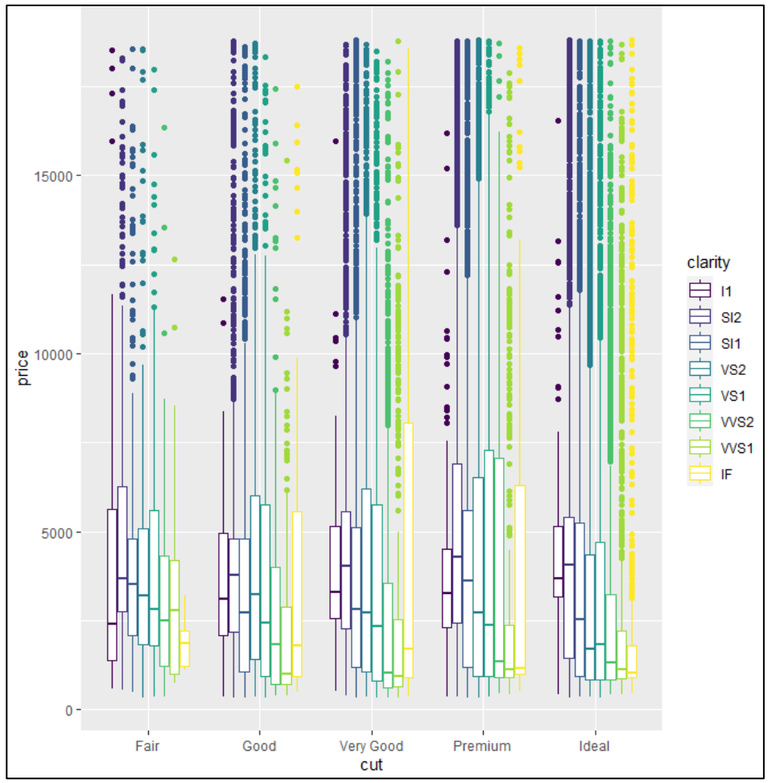
03 ggplot2 기본 함수 – qplot 함수 – 예시

- **예시5**

```
qplot(data=diamonds, x=cut, y=price, aes(col=clarity), geom='boxplot')
```

➢ aes(Aesthetics) 요소 추가 가능

➢ geom(Geometrics) 요소 추가 가능



03 ggplot2 기본 함수 – ggplot 함수

- **ggplot()**
  - grammar of grapics(gg) plot
  - 그래픽 문법 요소 층 전부, 또는 일부를 직접 쌓아 데이터 시각화
    - '+' 연산자를 사용하여 문법 요소 추가
  - 원하는 모든 형태의 복잡한 그래프의 표현 가능

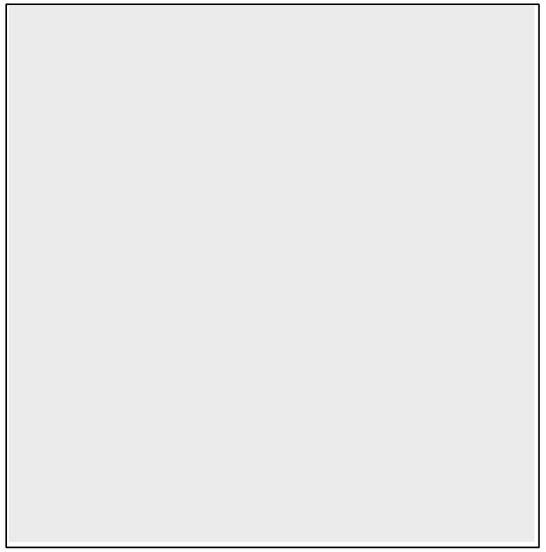
- **문법**
  - ggplot은 문법 요소 층을 쌓을 시작 지점이 되는 함수

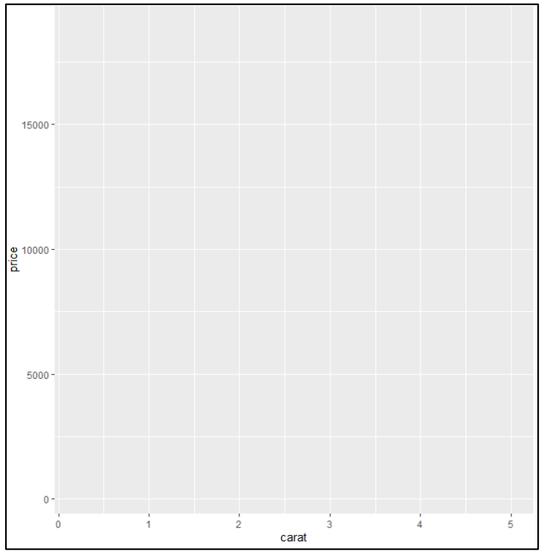
```
ggplot(data='데이터 변수 명', mapping = aes(x='data에서 x축에 사용할 변수', y ='data에서 y축에 사용할 변수', ...))
```

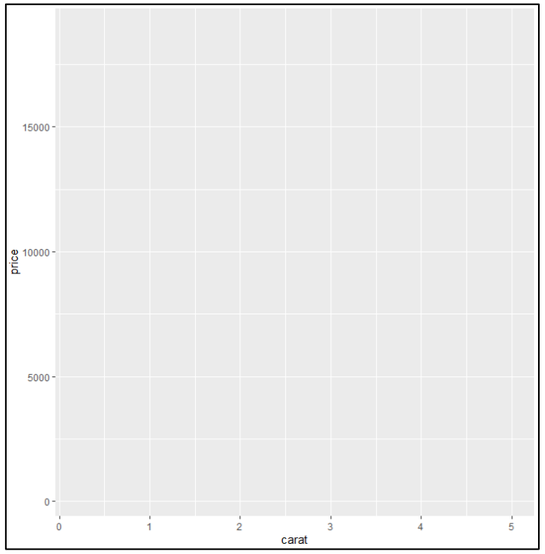
03 ggplot2 기본 함수 – ggplot 함수 – 예시

- **예시1**
  - ➢ 그래픽 틀만 출력 됨

```
ggplot(diamonds)  # 데이터셋으로 diamonds가 지정됨
ggplot(diamonds, aes(x = carat, y = price))  # x축으로 carat, y축으로 price가 지정됨
ggplot(diamonds, aes(x = carat, y = price, color = cut)) # cut에 따라 색이 서로 다르게 지정됨
```



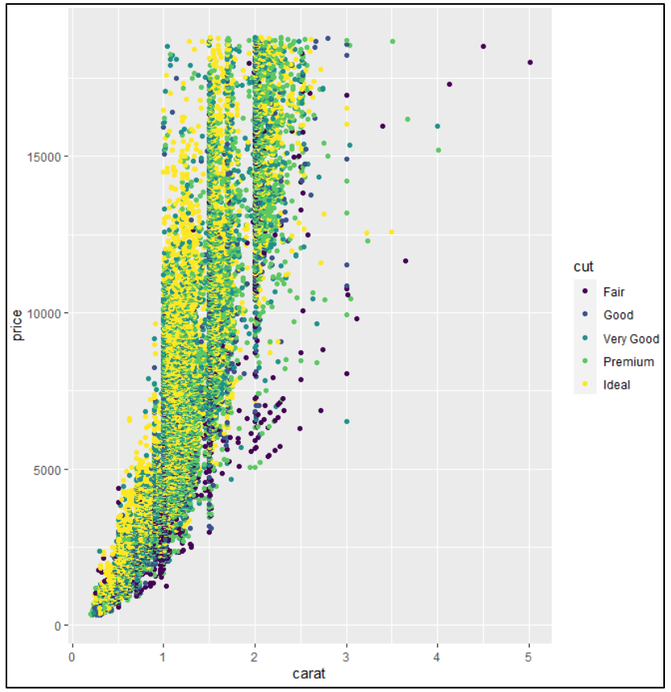




03 ggplot2 기본 함수 – ggplot 함수 – 예시

- **예시2**
  - ➢ 사용할 기하 요소 층을 추가

```
ggplot(diamonds, aes(x = carat, y = price, color = cut)) +
geom_point()
```



29차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

30차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

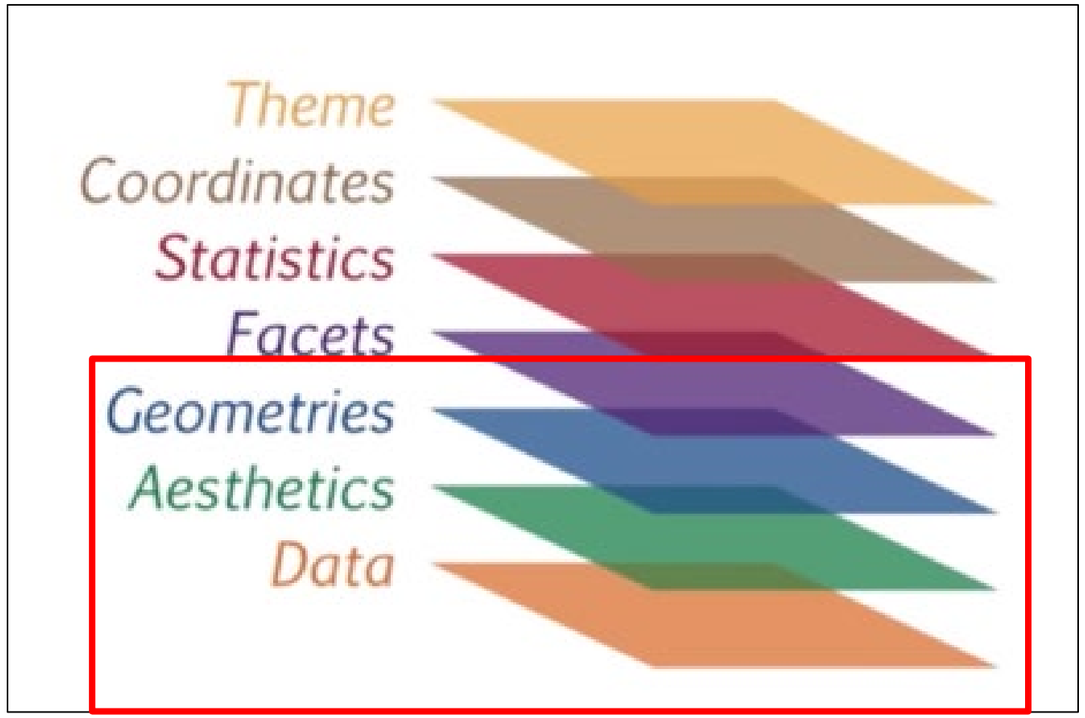
- 산점도 그래프를 그릴 수 있다.
- 선 그래프를 그릴 수 있다.
- 막대그래프를 그릴 수 있다.
- 히스토그램을 그릴 수 있다.
- 상자그림을 그릴 수 있다.

학습 목차

- 산점도 그래프
- 선 그래프
- 막대그래프
- 히스토그램
- 상자그림

01 Layer 구조 – 필수 입력 layer

- **데이터가 시각화 되기 위해서 반드시 정의되어야 하는 layer**
  - 1층(data)
  - 2층(x, y축, 그 외는 옵션)
  - 3층(point, line, boxplot 등.. 최소 1개 이상)



02 산점도 그래프 - 문법

- **geom_point() 함수**

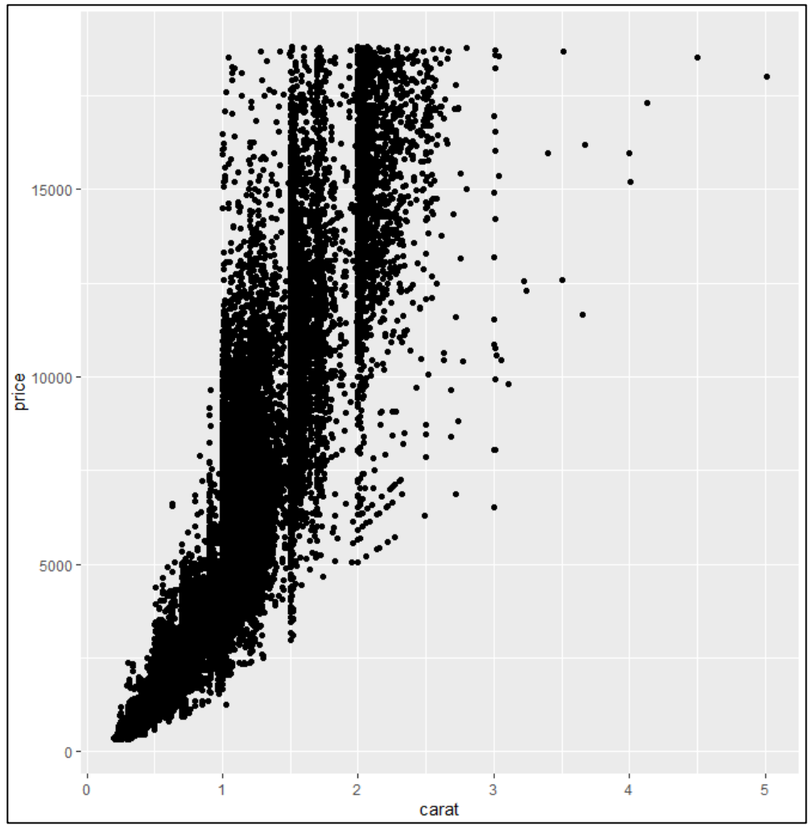
```
ggplot(dataFrame, aes(x=x축, y=y축)) +
geom_point()
```

  - ‘+’ 연산자 사용하여 추가 layer 추가

- **geom_point 주요 매개변수**
  - shape: 점 모양
  - color: 테두리 색
  - fill: 채우기 색
  - size: 점 크기
  - stroke: 테두리 두께

02 산점도 그래프 – 예시1

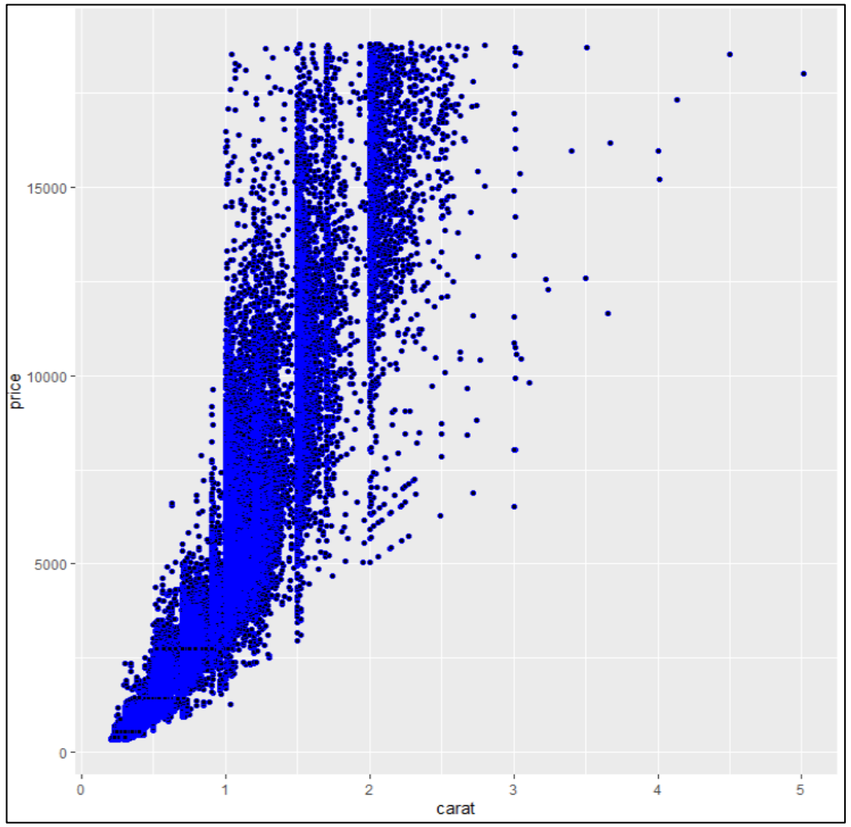
```
ggplot(diamonds, aes(x = carat, y = price)) +
  geom_point()
```



02 산점도 그래프 - 예시2

```
ggplot(diamonds, aes(x = carat, y = price)) +
  geom_point(shape=21, color='blue', fill='black')
```

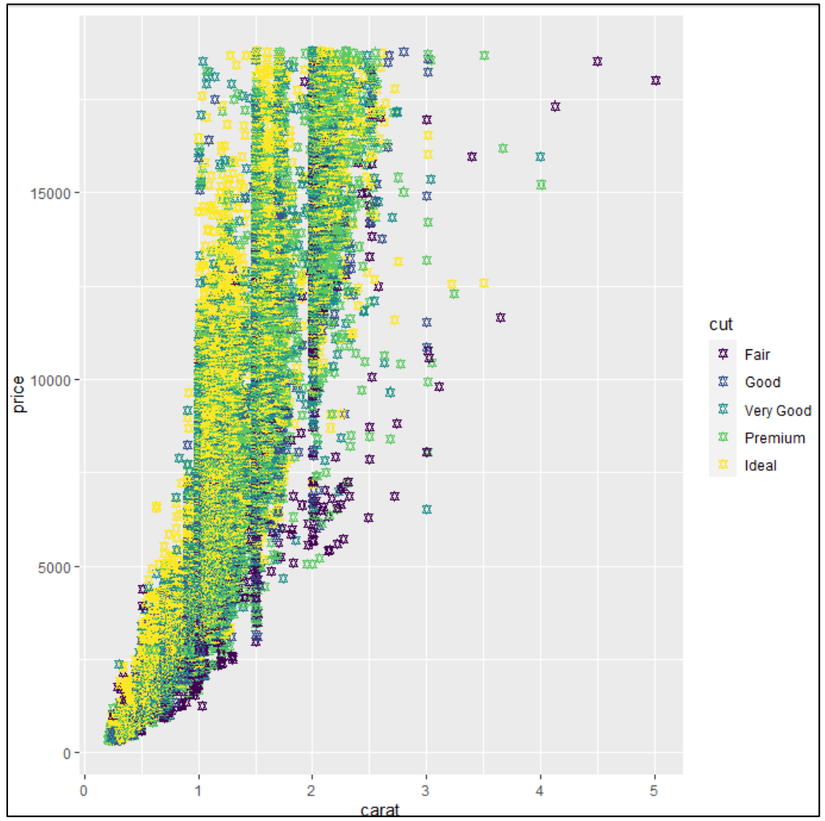
- **모든 점에 동일한 색 지정**



02 산점도 그래프 – 예시3

```
ggplot(diamonds, aes(x = carat, y = price, col=cut)) +
  geom_point(shape=11)
```

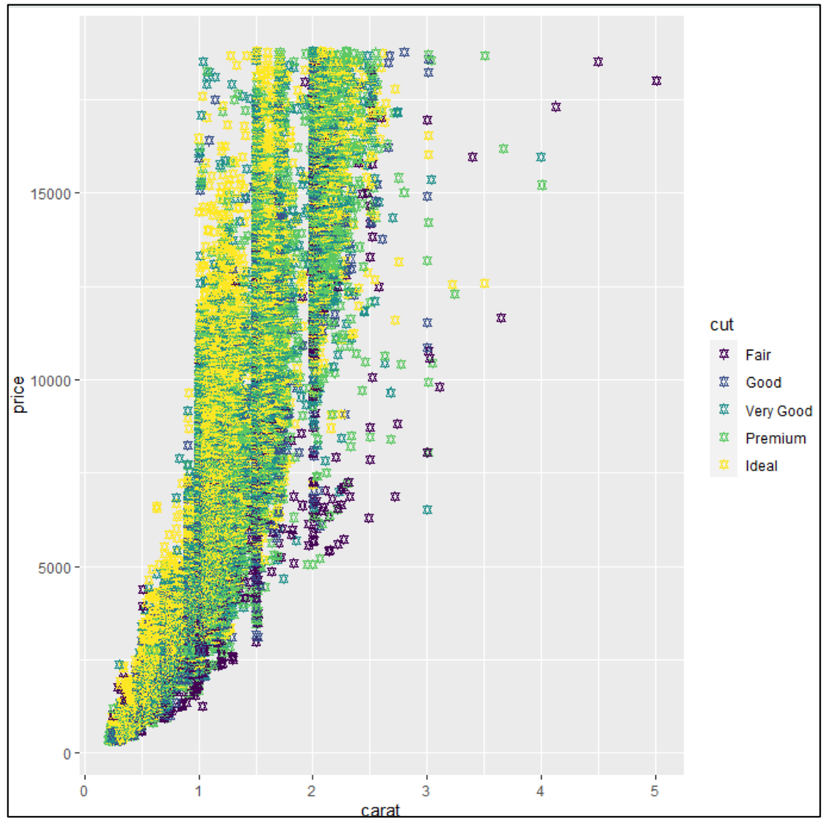
- **범주별 다른 색 지정**



02 산점도 그래프 - 예시4

```
ggplot(diamonds, aes(x = carat, y = price)) +
  geom_point(shape=11, mapping=aes(col=cut))
```

- **범주별 다른 색 지정**
  - ➢ point 그림에만 범주별 색 지정



03 Aesthetics layer – 자주 사용하는 옵션

```
aes(x,y, col, fill))
```

- **x, y**

- **col: 선 색**
  - 점(테두리), 선 등에 사용

- **fill: 색 채우기**
  - 막대 그래프, 히스토그램 등에 사용

04 선 그래프 – 문법

- **geom_line() 함수**

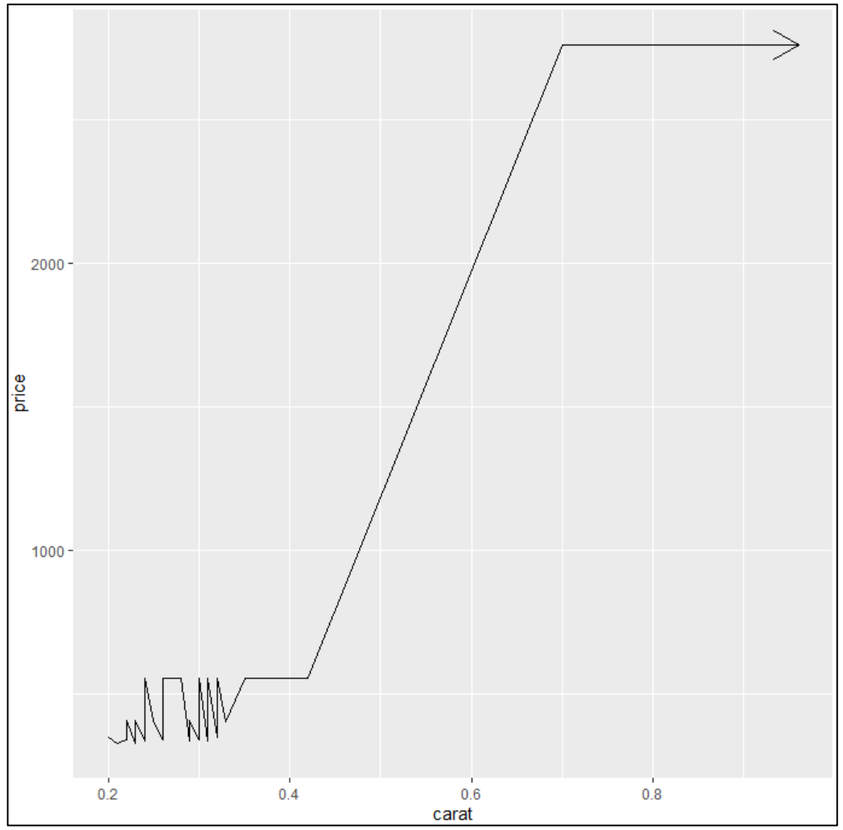
```
ggplot(dataFrame, aes(x=x축, y=y축)) +
geom_line()
```

  - ‘+’ 연산자 사용하여 추가 layer 추가

- **geom_line 주요 매개변수**
  - color: 선 색
  - size: 선 크기
  - arrow: 선 끝에 화살표 생성
    - arrow() 함수 매개변수 값으로 사용

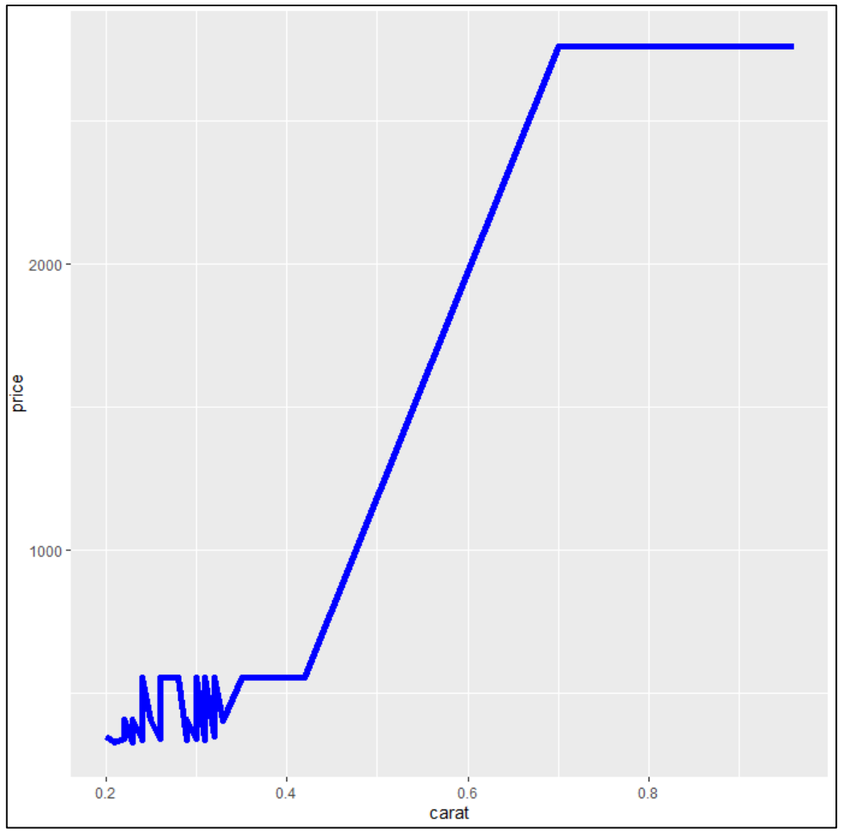
04 선 그래프 – 예시1

```
ggplot(diamonds[1:100,], aes(x = carat, y = price)) +   #100행만 추출
  geom_line(arrow= arrow())
```



04 선 그래프 – 예시2

```
ggplot(diamonds[1:100,], aes(x = carat, y = price)) +   #100행만 추출
  geom_line(color='blue', size=2)
```



05 막대 그래프 – 문법

- **geom_bar() 함수**

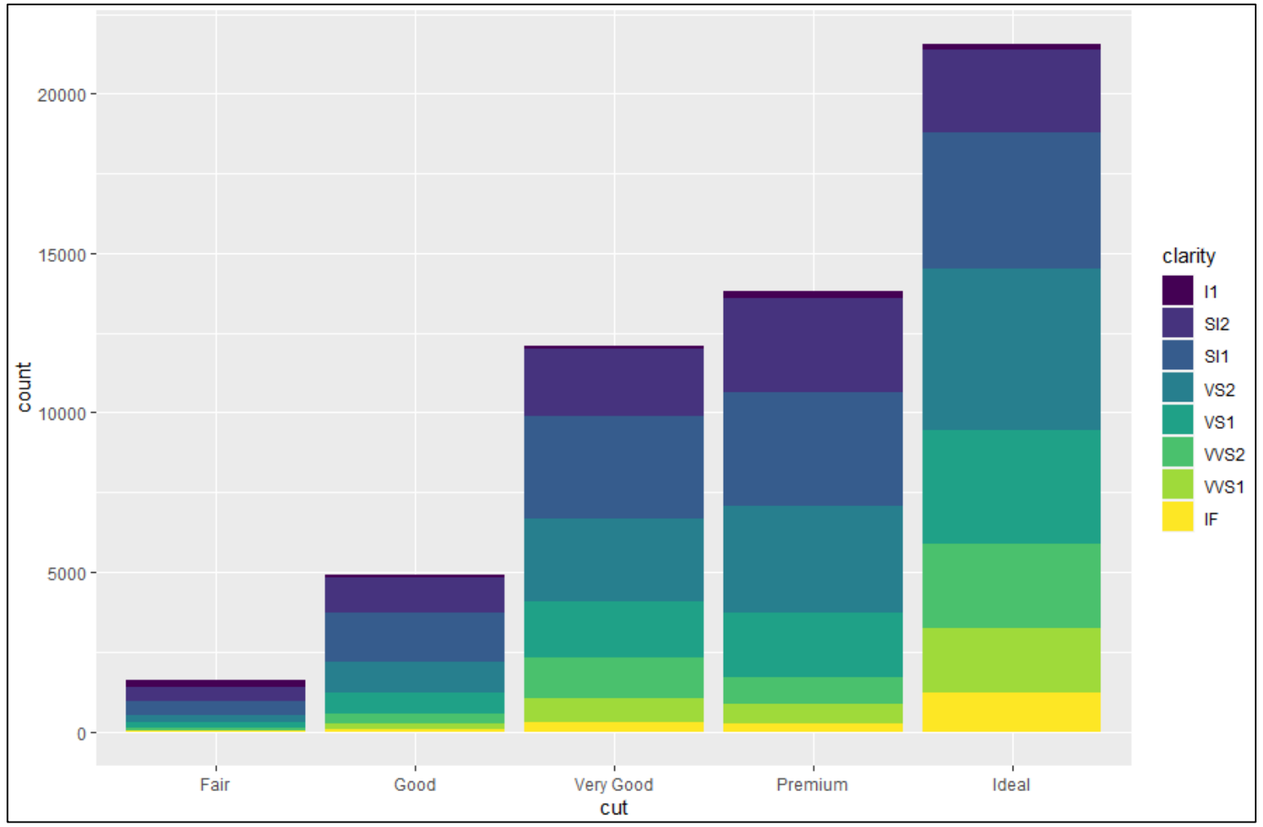
```
ggplot(dataFrame, aes(x=x축)) +
geom_bar()
```

  - ‘+’ 연산자 사용하여 추가 layer 추가

- **geom_bar 주요 매개변수**
  - geom_point 주요 매개변수 참고

05 막대 그래프 – 예시1

```
ggplot(diamonds, aes(x=cut, fill=clarity)) +
  geom_bar()
```



06 히스토그램 – 문법

- **geom_histogram() 함수**

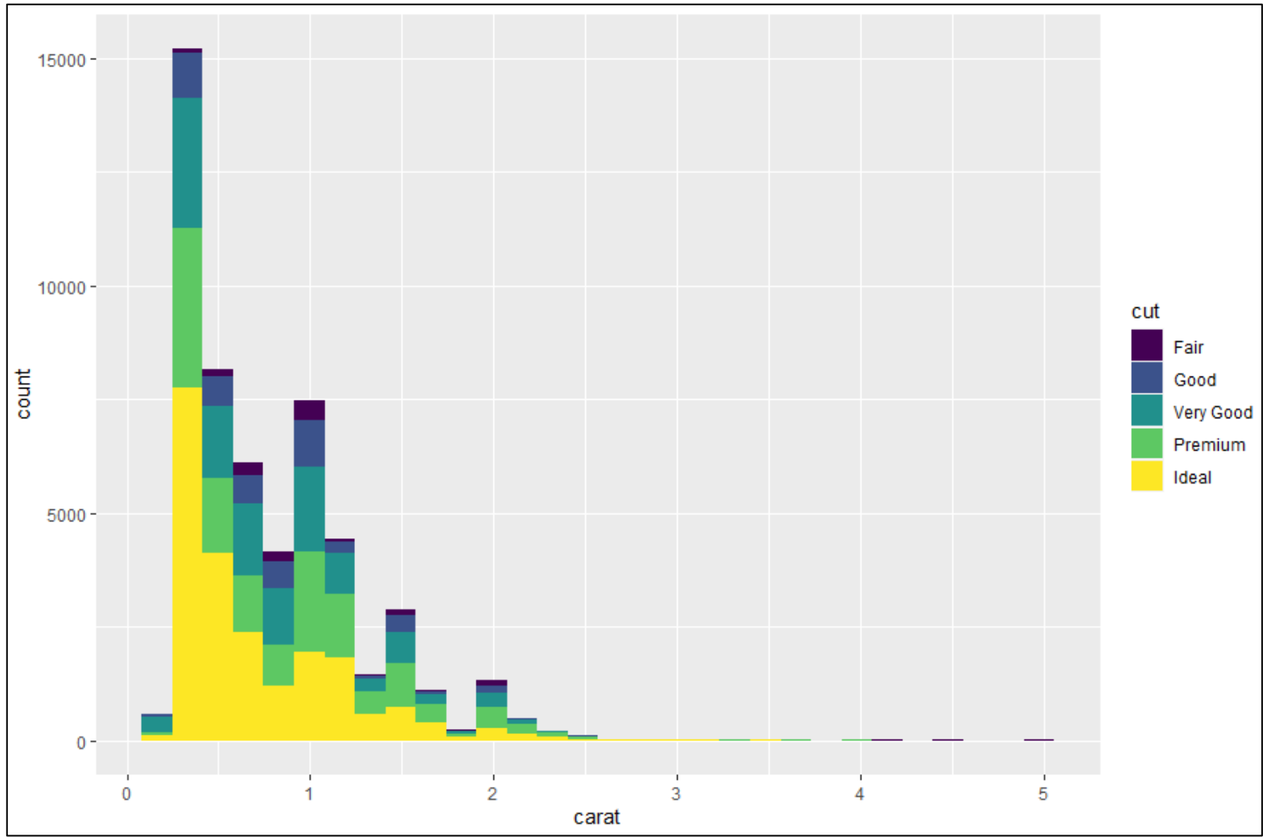
```
ggplot(dataFrame, aes(x=x축)) +
geom_histogram()
```

  - ‘+’ 연산자 사용하여 추가 layer 추가

- **geom_histogram 주요 매개변수**
  - binWidth: 히스토그램 데이터를 나누는 구간의 길이
    - *주의* 히스토그램 바 개수가 아님
  - geom_bar 주요 매개변수 참고

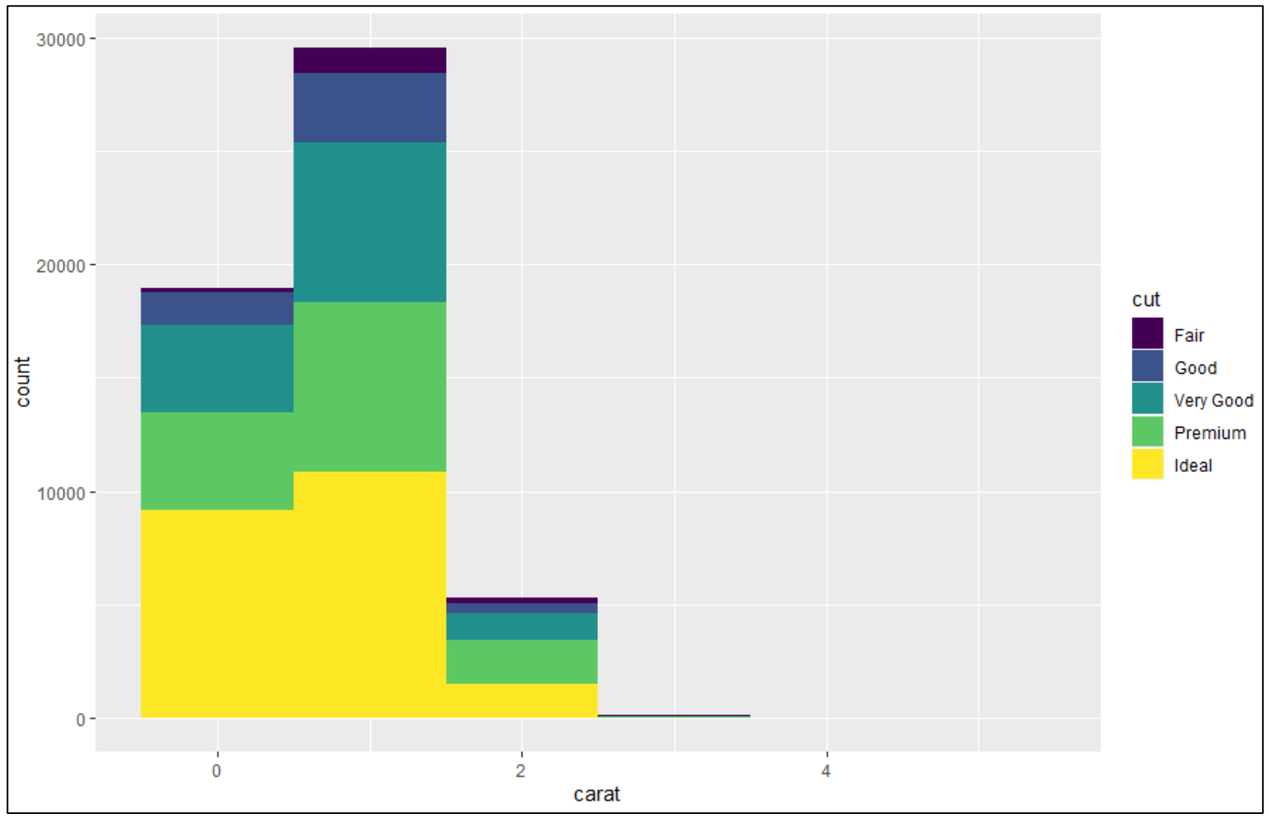
06 히스토그램 – 예시1

```
ggplot(diamonds, aes(x=carat, fill=cut)) +
  geom_histogram()
```



06 히스토그램 – 예시2

```
ggplot(diamonds, aes(x=carat, fill=cut)) +
  geom_histogram(binWidth=1)
```



07 상자그림 - 문법

- **geom_boxplot() 함수**

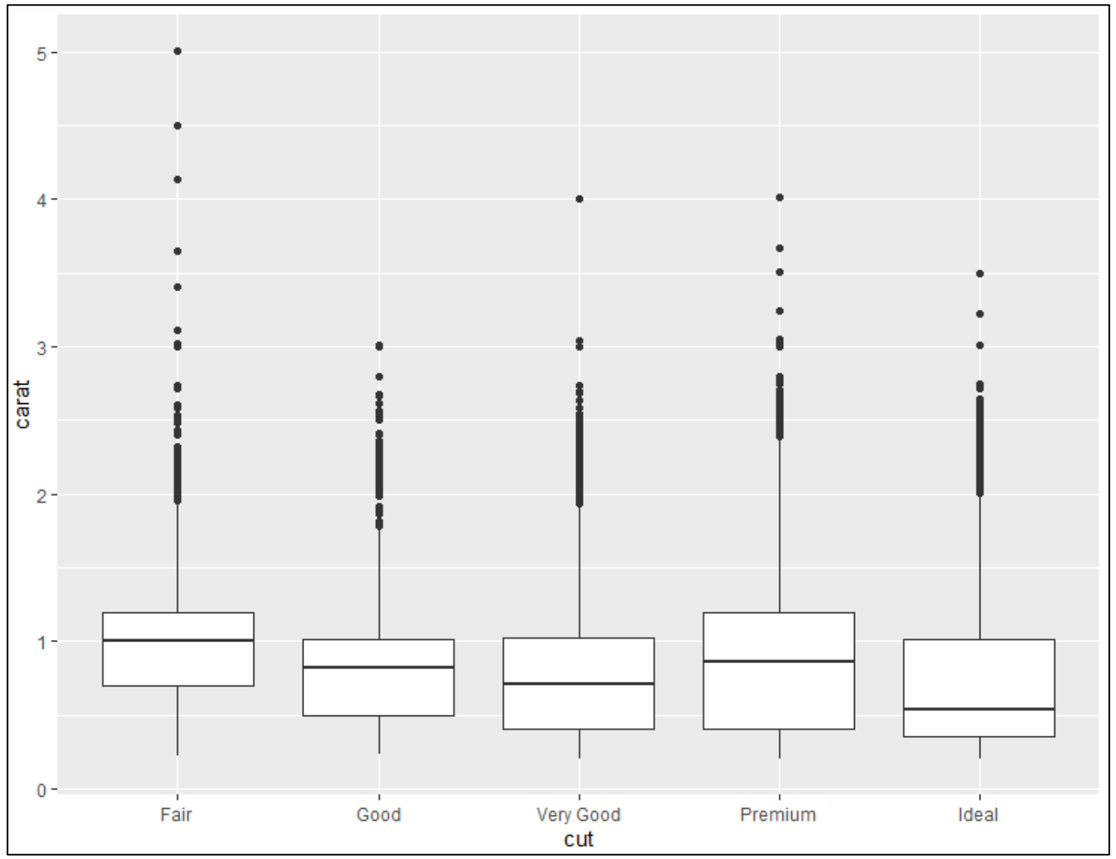
```
ggplot(dataFrame, aes(x=x축, y=y축)) +
geom_boxplot()
```

  - ‘+’ 연산자 사용하여 추가 layer 추가
  - y축 값(수치형)만 입력 가능
  - x축 값(범주형)과 함께 입력 시, 범주 별 데이터의 상자그림 출력가능

- **geom_boxplot 주요 매개변수**
  - geom_bar 주요 매개변수 참고

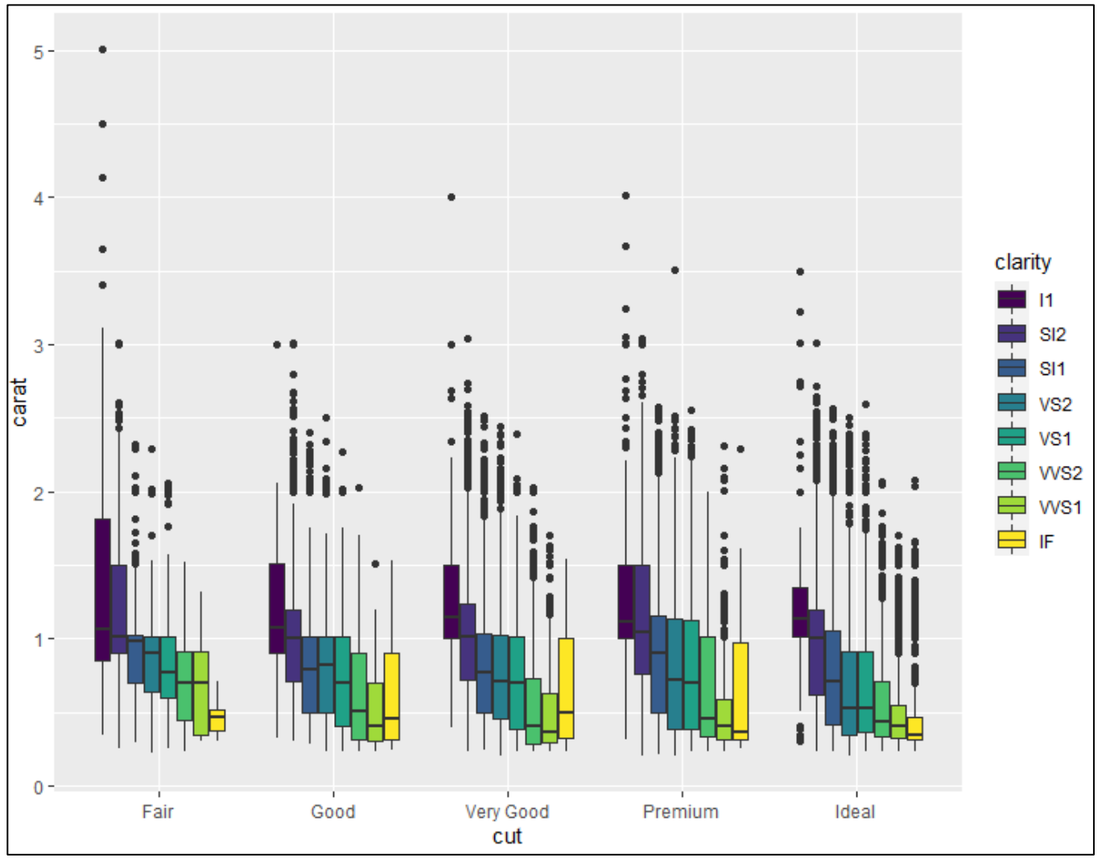
07 상자그림 - 예시1

```
ggplot(diamonds, aes(x=cut, y=carat )) +
  geom_boxplot()
```



07 상자그림 - 예시2

```
ggplot(diamonds, aes(x=cut, y=carat, fill=clarity )) +
  geom_boxplot()
```



30차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

31차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

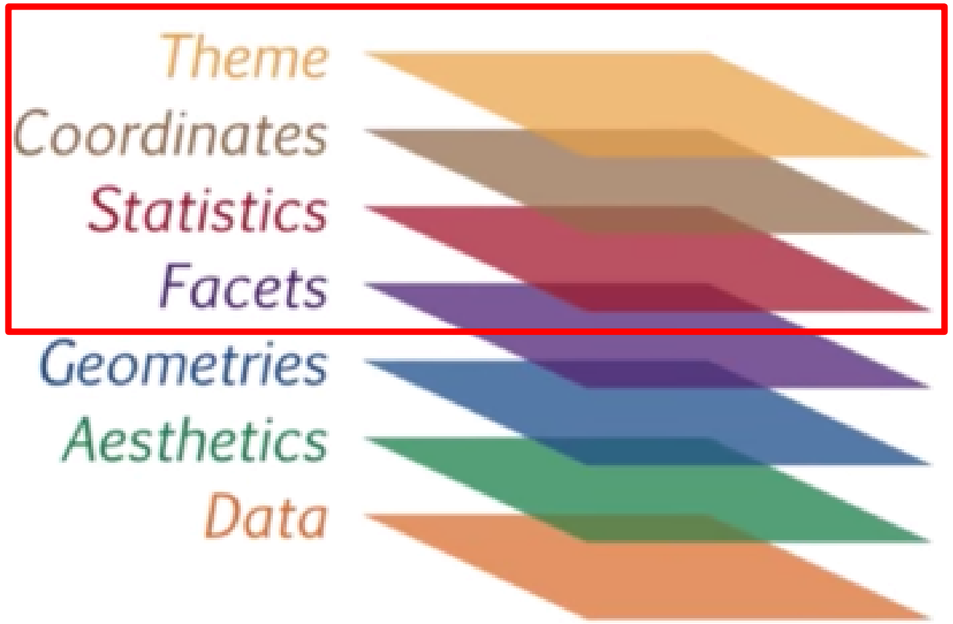
- Facets layer를 이해할 수 있다.
- Statistics layer를 이해할 수 있다.
- Coordinates layer를 이해할 수 있다.
- Theme layer를 이해할 수 있다.

학습 목차

- Facets layer
- Statistics layer
- Coordinates layer
- Theme layer

01 Layer 구조 – 옵션 layer

- **데이터 시각화 완성도를 높여주는 layer**
  - 4층(Facets): 범주 값 별 서브그래프 분석
  - 5층(Statistics): 통계관련 값 시각화
  - 6층(Coordinates): 좌표계 변환
  - 7층(Theme): 배경, 제목 등 추가/변경



02 Facets layer – 설명

- **범주형 컬럼의 변수별 서브 그래프를 각기 다른 패널에 그려주는 층**
  - 일종의 화면 분할 기능
- **문법**
  - '~' 뒤에 서브 그래프를 그릴 범주형 컬럼명 기입

```
ggplot(…) +
  geom_xxxx(..) +
  facet_xxxx(~'범주형 변수')
```

- **주요 함수**
  - facet_wrap()
  - facet_grid()

02 Facets layer – facet_wrap

- **1가지 범주형 컬럼에 대한 서브 그래프 출력 하는 함수**

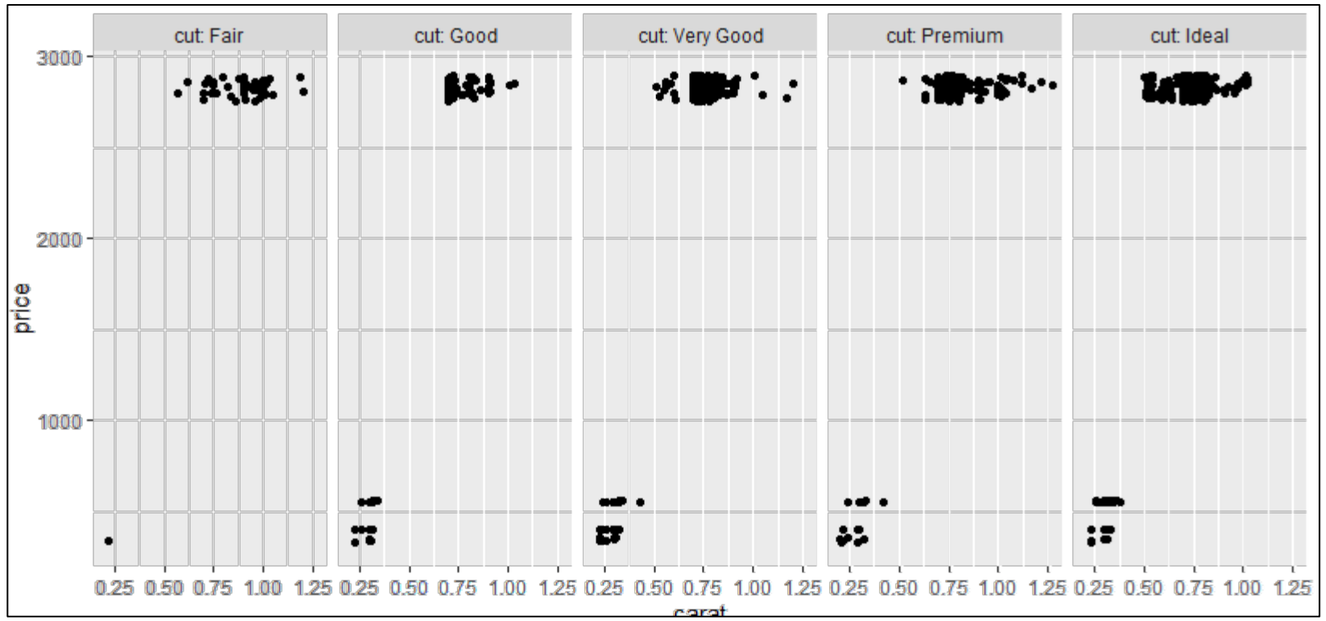
```
ggplot(…) +
  geom_xxxx(..) +
  facet_wrap(~'범주형 컬럼 변수', labeller=label_value, nrow=NULL, ncol=NULL)
```

- **주요 매개변수**
  - labeller: 'label_both' 사용시 컬럼명과 범주값 함께 출력
  - nrow, ncol : 'nrow x ncol' 형태로 서브 그래프 배치

02 Facets layer – facet_wrap – 예시

- 예시

```
ggplot(diamonds[1:1000,], mapping = aes(x=carat, y=price)) +
  geom_point() +
  facet_wrap(~cut, labeller=label_both, nrow = 1, ncol= 5)
```



02 Facets layer – facet_grid

- **2가지 범주형 컬럼에 대한 서브 그래프 출력 하는 함수**

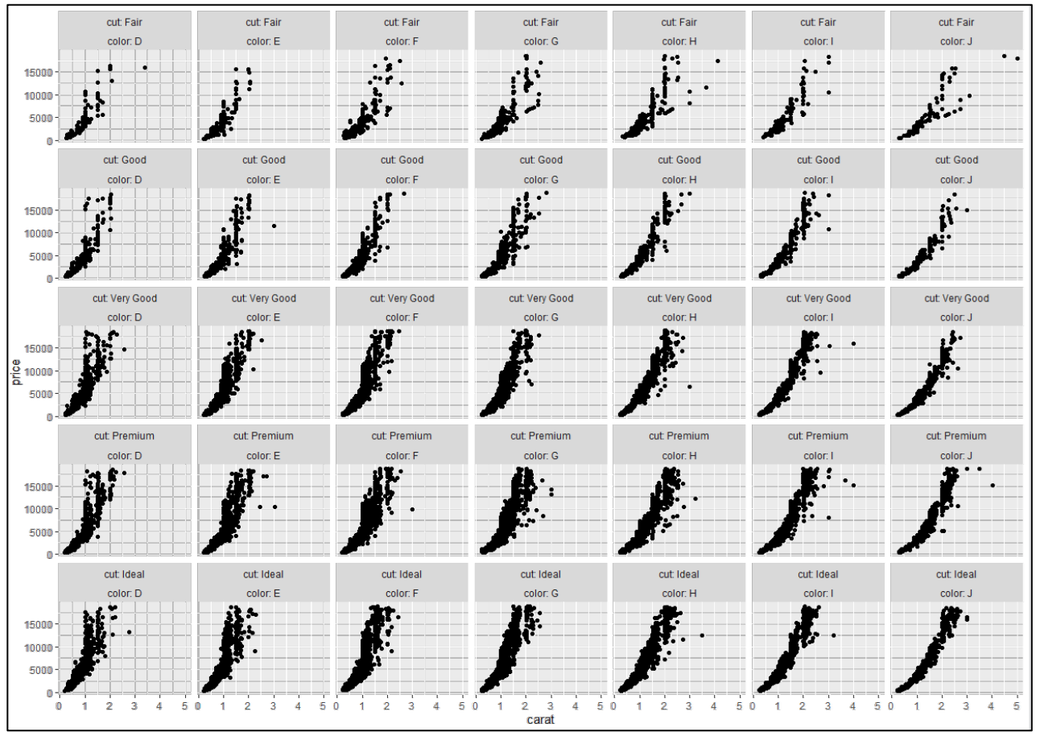
```
ggplot(…) +
  geom_xxxx(..) +
  facet_wrap('범주형 컬럼 변수1'~'범주형 컬럼 변수2', labeller=label_value)
```

- **주요 매개변수**
  - facet_wrap과 동일

02 Facets layer – facet_grid – 예시

- 예시

```
ggplot(diamonds, mapping = aes(x=carat, y=price)) +
  geom_point() +
  facet_wrap(cut~color, labeller=label_both, nrow=5, ncol=7)
```



03 Statistics layer – 설명

- 데이터에 대한 통계값을 시각화 해주는 층

- 주요 함수
  - stat_smooth()
  - stat_summary()

03 Statistics layer – stat_smooth

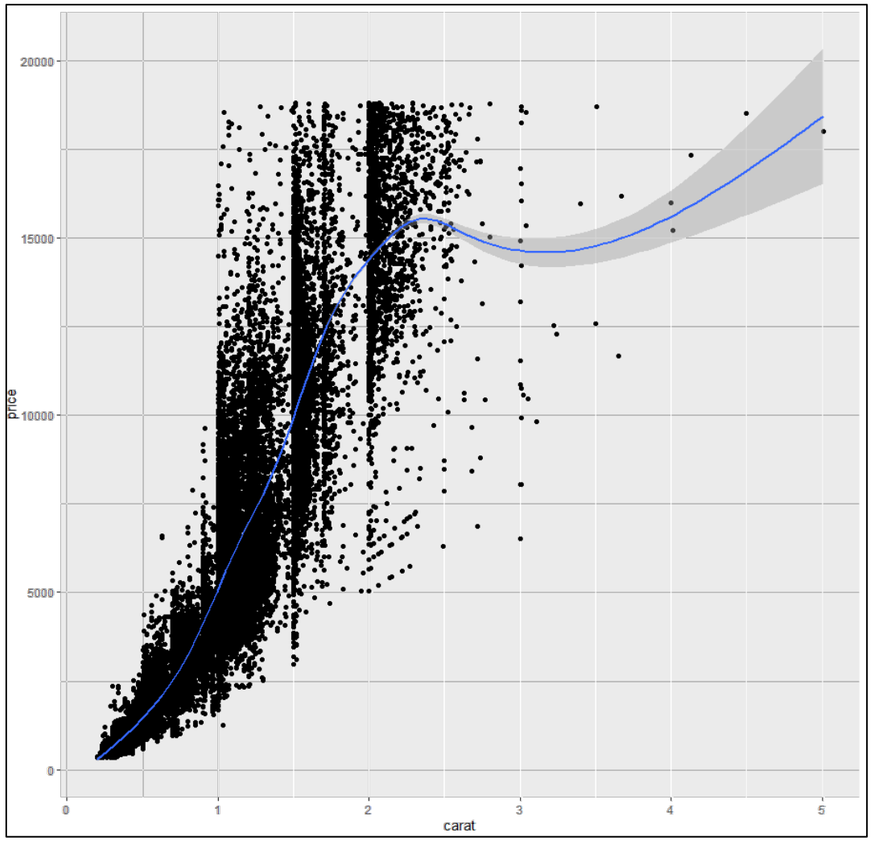
- **데이터에 회귀선을 그리는 함수**

```
ggplot(…) +
  geom_xxxx(..) +
 stat_smooth(level=0.95)
```

- **주요 매개변수**
  - level: 신뢰구간 (0.0~1.0)

- **예시**

```
ggplot(…) +
  geom_xxxx(..) +
 stat_smooth(level=0.95)
```



03 Statistics layer – stat_summary

- **x값에 대한 y값의 간단한 통계값을 그래프에 그려주는 함수**

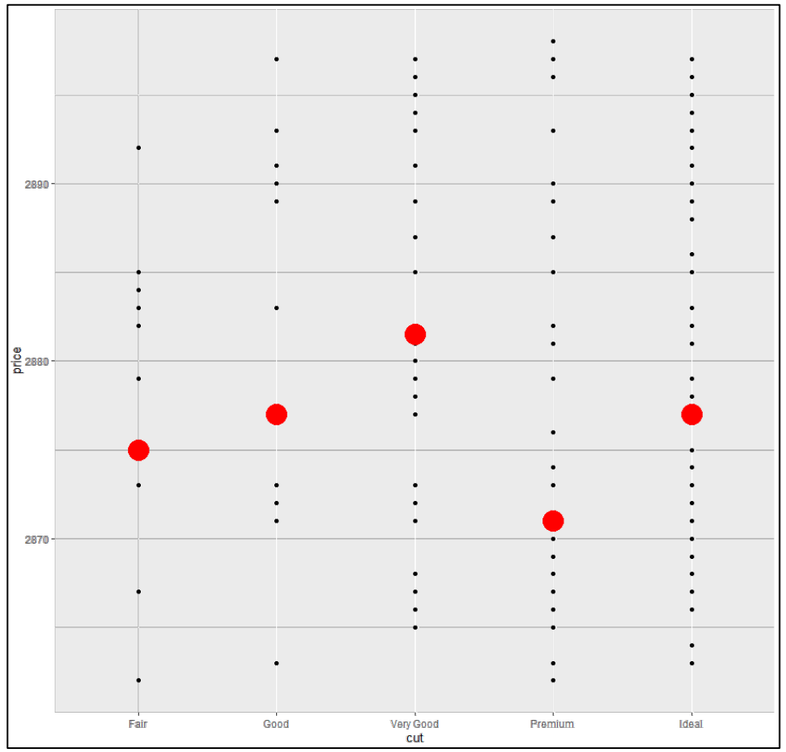
```
ggplot(…) +
  geom_xxxx(..) +
 stat_summary(fun.y = mean, color, size, geom = ‘point’)
```

- **주요 매개변수**
  - fun.y: 구하고 싶은 통계함수
    - mean, min, max, median 등
  - color: geom 색
  - size: geom 크기
  - geom: 통계치를 나타낼 시각화 형태

03 Statistics layer – stat_summary – 예시

- 예시

```
ggplot(diamonds[800:1000,], mapping = aes(x=cut, y=price )) +
  geom_point() +
  stat_summary(fun.y = mean, color = 'red', size=2)                    # 평균값 선택
```



04 Coordinate layer – 설명

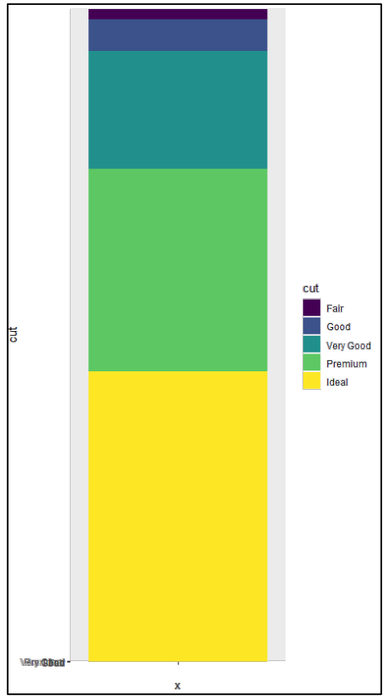
- **x축 또는 y축 정보를 변형하는 시각화 층**

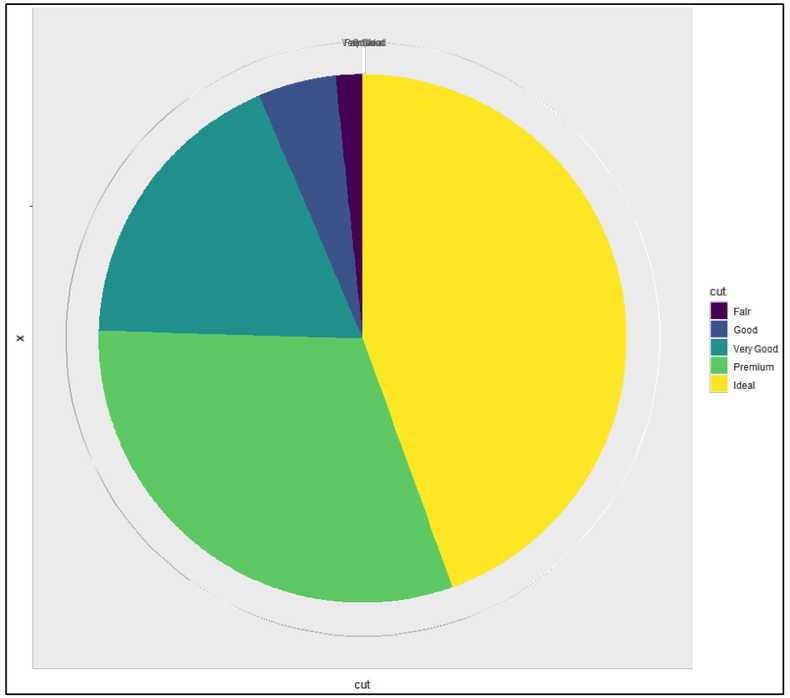
- **주요 함수**
  - coord_cartesian(xlim, ylim)
    - x축, y축 범위 지정
  - coord_flip()
    - x축, y축 반전
  - coord_polar()
    - (x,y) 좌표계를 극좌표계로 변환

04 Coordinate layer – 예시

- **coord_polar() 함수 예시**

```
ggplot(diamonds[1:1000,], mapping = aes(x="", y= cut, fill= cut)) +
  geom_bar(stat='identity',width=1) +
  coord_polar(theta="y")
```





05 Theme layer – 설명

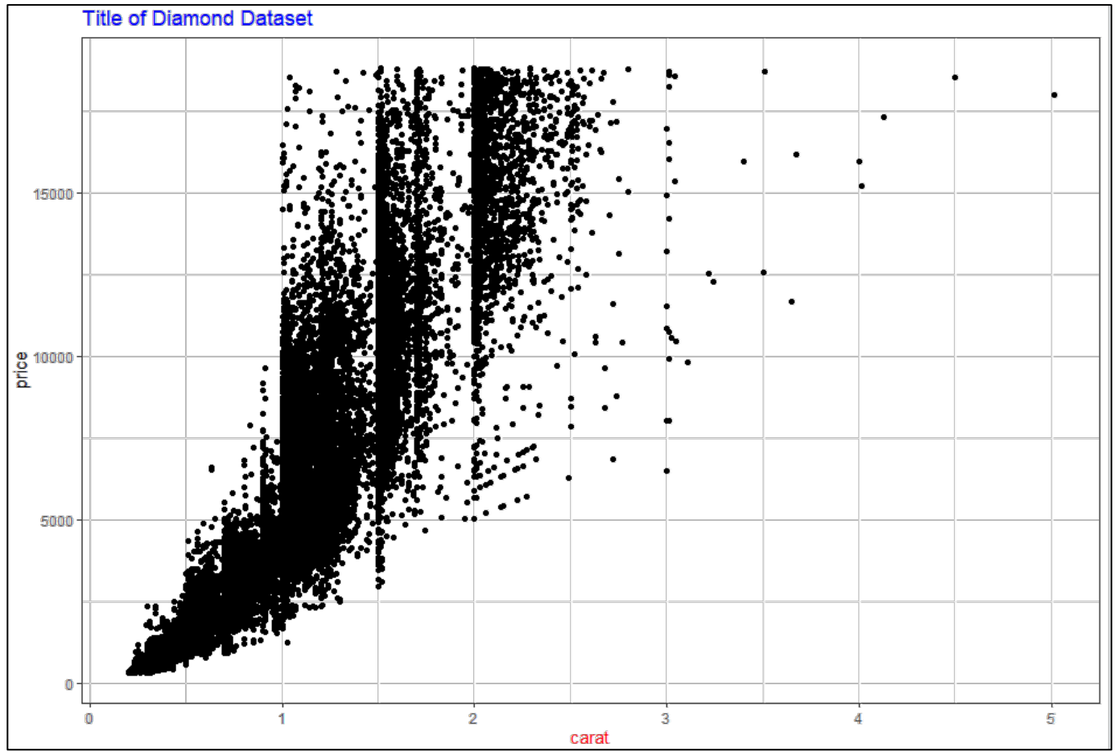
- **데이터와 무관한 시각화 요소를 꾸밀 수 있는 층**

- **주요 함수**
  - ggtitle(“”)
    - 제목 작성
  - theme(…)
    - 제목, 축, 범례, 판넬, 배경, facet 모양 등의 속성 변경
  - theme_gray()
    - 배경을 회색으로 지정
  - theme_bw()
    - 배경을 하얀색으로 지정

Name	Description	Element type
text	All text elements	element_text()
rect	All rectangular elements	element_rect()
line	All line elements	element_line()
axis.line	Lines along axes	element_line()
axis.title	Appearance of both axis labels	element_text()
axis.title.x	X-axis label appearance	element_text()
axis.title.y	Y-axis label appearance	element_text()
axis.text	Appearance of tick labels on both axes	element_text()
axis.text.x	X-axis tick label appearance	element_text()
axis.text.y	Y-axis tick label appearance	element_text()
legend.background	Background of legend	element_rect()
legend.text	Legend item appearance	element_text()
legend.title	Legend title appearance	element_text()
legend.position	Position of the legend	left, "right", "bottom", "top", or c (x, y)
panel.background	Background of plotting area	element_rect()
panel.border	Border around plotting area	element_rect(linetype="dashed")
panel.grid.major	Major grid lines	element_line()
panel.grid.major.x	Major grid lines, vertical	element_line()
panel.grid.major.y	Major grid lines, horizontal	element_line()
panel.grid.minor	Minor grid lines	element_line()
panel.grid.minor.x	Minor grid lines, vertical	element_line()
panel.grid.minor.y	Minor grid lines, horizontal	element_line()
plot.background	Background of the entire plot	element_rect(fill = "white", colour = NA)
plot.title	Title text appearance	element_text()
strip.background	Background of facet labels	element_rect()
strip.text	Text appearance for vertical and horizontal fac et labels	element_text()
strip.text.x	Text appearance for horizontal facet labels	element_text()
strip.text.y	Text appearance for vertical facet labels	element_text()

05 Theme layer – 예시

```
ggplot(diamonds, mapping = aes(x=carat, y=price)) +
  geom_point() +
  ggtitle("Title of Diamond Dataset") +
  theme_bw() +
  theme(axis.title.x = element_text(colour="red"), plot.title = element_text(color="blue"))
```



31차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

32차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 현실데이터를 사용한 ggplot2 기반 데이터 시각화를 할 수 있다.

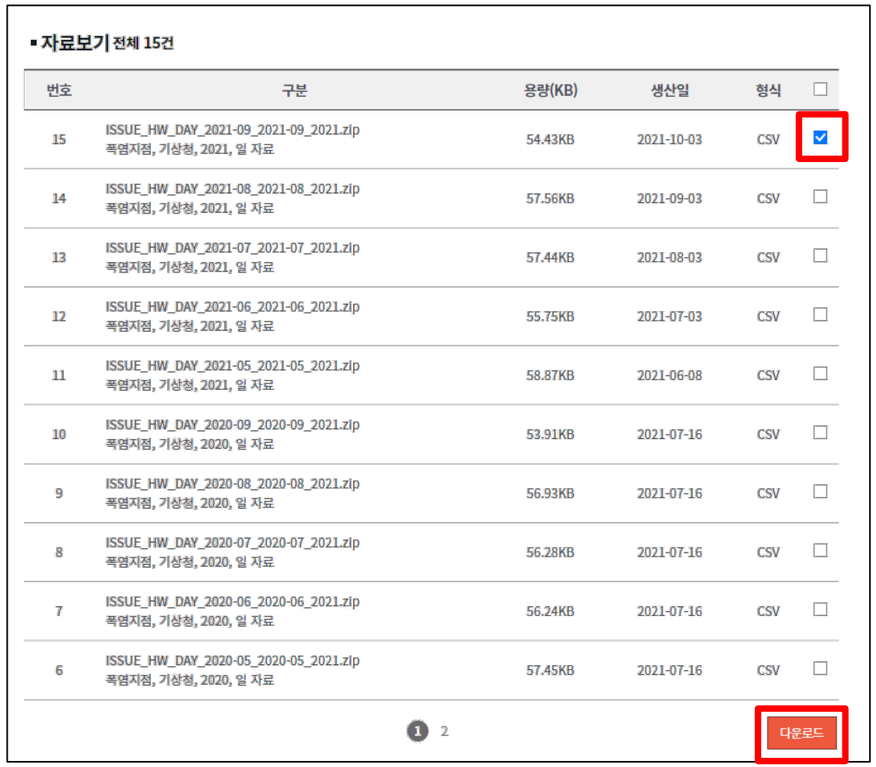
학습 목차

- 현실데이터를 사용한 ggplot2 기반 데이터 시각화

01 실습 - 데이터셋 다운로드

Q1. 기상청 홈페이지에서 2021년 9월 폭염 데이터셋을 다운 받으세요.

- 사이트 링크: https://data.kma.go.kr/data/weatherIssue/slthtList.do?pgmNo=690



▪ 자료보기 전체 15건

번호	구분	용량(KB)	생산일	형식	☐
15	ISSUE_HW_DAY_2021-09_2021-09_2021.zip 폭염지점, 기상청, 2021, 일 자료	54.43KB	2021-10-03	CSV	☑
14	ISSUE_HW_DAY_2021-08_2021-08_2021.zip 폭염지점, 기상청, 2021, 일 자료	57.56KB	2021-09-03	CSV	☐
13	ISSUE_HW_DAY_2021-07_2021-07_2021.zip 폭염지점, 기상청, 2021, 일 자료	57.44KB	2021-08-03	CSV	☐
12	ISSUE_HW_DAY_2021-06_2021-06_2021.zip 폭염지점, 기상청, 2021, 일 자료	55.75KB	2021-07-03	CSV	☐
11	ISSUE_HW_DAY_2021-05_2021-05_2021.zip 폭염지점, 기상청, 2021, 일 자료	58.87KB	2021-06-08	CSV	☐
10	ISSUE_HW_DAY_2020-09_2020-09_2021.zip 폭염지점, 기상청, 2020, 일 자료	53.91KB	2021-07-16	CSV	☐
9	ISSUE_HW_DAY_2020-08_2020-08_2021.zip 폭염지점, 기상청, 2020, 일 자료	56.93KB	2021-07-16	CSV	☐
8	ISSUE_HW_DAY_2020-07_2020-07_2021.zip 폭염지점, 기상청, 2020, 일 자료	56.28KB	2021-07-16	CSV	☐
7	ISSUE_HW_DAY_2020-06_2020-06_2021.zip 폭염지점, 기상청, 2020, 일 자료	56.24KB	2021-07-16	CSV	☐
6	ISSUE_HW_DAY_2020-05_2020-05_2021.zip 폭염지점, 기상청, 2020, 일 자료	57.45KB	2021-07-16	CSV	☐

1 2

다운로드

01 실습 - 데이터셋 확인/로딩

Q1. 기상청 홈페이지에서 2021년 9월 폭염 데이터셋을 다운 받으세요.

- 필요한 컬럼 값: 최고체감온도, 최고기온, 평균상대습도, 폭염영향예보, 자외선지수
- 다운로드 받은 데이터 파일은 xls 형식
- 프로그램상에 데이터 로딩이 안될 경우
  - 엑셀파일을 켬 → 다른 이름으로 저장 → 저장타입 xlsx로 변경 → 변경된 파일로 로딩

01 실습 - 데이터셋 확인/로딩

Q2. 데이터셋을 열어 로딩가능여부를 분석하고, 프로그램상에 데이터를 로딩하세요.

- 필요 데이터 위치
  - 4~12 컬럼
  - 6, 7, 9, 11 컬럼은 필요 없음

```
library(readxl)

data <- read_excel('day.xlsx', col_names = TRUE, range=cell_cols(4:12) )
data <- data[,c(-3, -4, -6, -8)] #필요없는 컬럼 제거
print(data)
```

01 실습 - 데이터셋 전처리

Q3. 주어진 명세를 따라 데이터셋을 전처리하세요

- 1. 각 열 이름의 특수문자 제거
  - strsplit 함수 사용
- 2. 폭염영향예보 컬럼의 결측치 “보통”으로 변경
  - is.na 함수 사용
- 3. 2번 진행 후, 결측치 존재하는 행 제거
  - is.omit 함수 사용

01 실습 - 데이터셋 전처리

Q3. 주어진 명세를 따라 데이터셋을 전처리하세요

```
library(dplyr)

# 열 이름 특수문제 제거
result <- data %>%  names() %>% strsplit(split="[[:punct:]]") #특수문자를 구분자로 사용. '(' 포함
result

new_name <- c()
for(t in result){ #list의 특정 값을 가져올 때는 index문을 사용하여 index 직접 참조
  print(t)
  print(paste("first value:", t[1]))
  new_name <- c(new_name, t[1])
}
print(new_name)

#컬럼 명 변경
names(data) <- new_name

#결측치 값 변경
data$폭염영향예보[is.na(data$폭염영향예보)] <- "보통"  # '폭염영향예보' 컬럼 NA값 "보통"으로 변경
str(data)
data <- na.omit(data)          # NA가 존재하는 행 제거
str(data)                      # 데이터 전처리 완료 결과 확인
```

01 실습 - 데이터셋 시각화

Q4. '최고기온'을 x축, '최고체감온도'를 y축으로 하고, '자외선 지수'에 따라 색이 다른 산점도를 그리세요.

- ggplot2 패키지의 ggplot 함수를 사용하세요

```
#Q4 시각화1. 최고기온=x, 최고체감온도=y, col=자외선지수
ggplot(data, mapping = aes(x=최고기온, y=최고체감온도, col=자외선지수)) +
  geom_point()
```

01 실습 - 데이터셋 시각화

Q5-1. Q4의 결과에 '자외선 지수' 따라 facet을 나누세요

```
#Q5-1. + facet1
ggplot(data, mapping = aes(x=최고기온, y=최고체감온도, col=자외선지수)) +
  geom_point() +
  facet_wrap(~자외선지수)
```

Q5-2. Q4의 결과에 '폭염영향예보' 및 '자외선 지수' 2개의 범주형 값에 따라 facet을 나누세요

- facet의 label은 컬럼명과 값이 모두 보이도록 셋팅하세요

```
#Q5-2. + facet2
ggplot(data, mapping = aes(x=평균상대습도, y=최고체감온도, col=자외선지수)) +
  geom_point() +
  facet_grid(폭염영향예보~자외선지수, labeller=label_both)
```

01 실습 - 데이터셋 시각화

Q6. Q5-1의 결과값에 회귀선을 추가로 그리세요

- 회귀선의 신뢰구간은 99%로 설정하세요

```
#Q6. + statistics
ggplot(data, mapping = aes(x=최고기온, y=최고체감온도, col=자외선지수)) +
  geom_point() +
  facet_wrap(~자외선지수) +
  stat_smooth(level=0.99)
```

01 실습 - 데이터셋 시각화

Q7. Q6의 결과값에 y축 범위가 20부터 33까지만 보이도록 좌표계를 수정하세요.

```
#Q7. + coordinates
ggplot(data, mapping = aes(x=최고기온, y=최고체감온도, col=자외선지수)) +
  geom_point() +
  facet_wrap(~자외선지수) +
  stat_smooth(level=0.99) +
  coord_cartesian(ylim=c(20,33))
```

01 실습 - 데이터셋 시각화

Q8. Q7의 결과값에 “2021년 9월 대한민국 최고기온 대비 최고체감온도” 라는 타이틀을 붙히고, 배경 색을 하얗게 변경하세요.

```
#Q8. + theme
ggplot(data, mapping = aes(x=최고기온, y=최고체감온도, col=자외선지수)) +
  geom_point() +
  facet_wrap(~자외선지수) +
  stat_smooth(level=0.99) +
  coord_cartesian(ylim=c(20,33)) +
  ggtitle("2021년 9월 대한민국 최고기온 대비 최고체감온도") +
  theme_bw()
```

32차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

33차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 영화평 텍스트 마이닝 시각화를 수행할 수 있다.

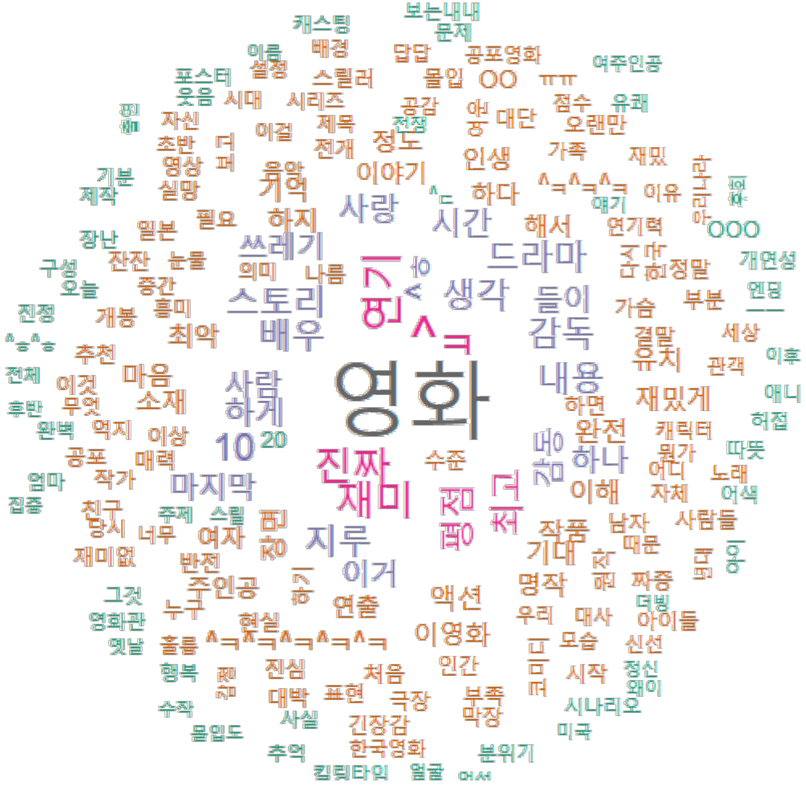
학습 목차

- 영화평 텍스트 마이닝 시각화 실습

01 영화평 텍스트 마이닝 - 목표

- **데이터셋 설명**
  - ‘ratings.txt’ 제공
  - 네이버 영화 리뷰평을 모아 놓은 데이터
  - 약 19만개 데이터

- **Q. 목표**
  - 네이버 리뷰 영화평 중에서 명사만을 추출 및 카운팅하여 CloudForm으로 출력



01 영화평 텍스트 마이닝 – 필요 패키지 설치

- **A1. 실습에 필요한 패키지 설치**
  - rJava
    - 해당 패키지 설치를 위해서는 PC에 JAVA가 설치되어 있어야 함
    - JAVA 설치 링크: https://www.java.com/ko/download/manual.jsp
  - memoise
  - KoNLP

```
install.packages("rJava")
install.packages("memoise")
install.packages("KoNLP",
                 repos = c("https://forkonlp.r-universe.dev",
                           "https://cloud.r-project.org"),
                 INSTALL_opts = c("--no-multiarch")
)

#jdk 깔렸는지 테스트
rJava::.jinit()

#jdk 안깔렸다면 설치
install.packages("multilinguer")
multilinguer::install_jdk()
```

01 영화평 텍스트 마이닝 – 필요 패키지 로딩

- **A2. 실습에 필요한 패키지 로딩**
  - KoNLP
  - dplyr
  - stringr

```
library(KoNLP)
library(dplyr)
library(stringr)

useNIADic() #NIA 사전
```

01 영화평 텍스트 마이닝 - 데이터 전처리

- **A3. 아래 가이드라인에 맞추어 ratings.txt데이터를 로딩 및 전처리**

```
#데이터 로딩
## readLines 함수 사용

#영화 평 으로만 구성된 벡터 생성
## 반복문 사용하여 readLines으로 로딩한 데이터 직접 참조해서 벡터 생성

#영화 평 내 특수문자 제거
## str_replace_all(데이터, '[[:punct:]]',  " ") 함수 사용

#명사 추출
## extractNoun 함수 사용
## extractNoun("나는 집에 가서 밥을 먹는다.") << 예시코드 실행 하여 함수 결과물 대략적 확인 후 진행

#명사 별 등장 횟수를 data frame 형식으로 저장
## list를 vector로 변환하는 함수인 'unlist' 함수 참고
## data frame 형식      1열: 명사, 2열: 등장 횟수

#단어수 2글자 이상인 행만 남김
```

01 영화평 텍스트 마이닝 - 데이터 전처리

- **A3-1. 데이터 로딩 및 영화평 벡터 생성**

```
#데이터 로딩
## readLines 함수 사용
txt <- readLines('ratings.txt', encoding = 'UTF-8')  #한국어 정상적으로 로딩하기 위해 readLines사용

#영화 평 으로만 구성된 벡터 생성
## 반복문 사용하여 readLines으로 로딩한 데이터 직접 참조해서 벡터 생성
result <- strsplit(txt, split='₩t')
data <- vector("character", length(result))
i <- 1
for(item in result) {
  data[i] <- item[2]
  i<- i+1
}
```

01 영화평 텍스트 마이닝 - 데이터 전처리

- **A3-2. 영화 평 내 특수문자 제거 및 명사 추출**

```
#영화 평 내 특수문자 제거
## str_replace_all(데이터, '[[:punct:]]',  " ") 함수 사용
data <- str_replace_all(data, '[[:punct:]]',  " ")  #44행 확인하면 ! 사라짐

#선택 사항. 아래 extractNoun함수는 시간이 많이걸림. 빠르게 결과 보고싶다면 데이터 크기 축소
data <- data[1:10000]

#명사 추출
## extractNoun 함수 사용
## extractNoun("나는 집에 가서 밥을 먹는다.") << 예시코드 실행 하여 함수 결과물 대략적 확인 후 진행
extractNoun("나는 집에 가서 밥을 먹는다.") #명사 추출하기 예시
nouns <- extractNoun(data)  #추가 package 설치할 수 있음
```

01 영화평 텍스트 마이닝 - 데이터 전처리

- **A3-3. 추출한 명사 별 등장 횟수를 데이터 프레임 형식으로 저장**

```
#명사 별 등장 횟수를 data frame 형식으로 저장
## list를 vector로 변환하는 함수인 'unlist' 함수 참고
wordcount <- table(unlist(nouns))

## data frame 형식      1열: 명사, 2열: 등장 횟수
df_word <- as.data.frame(wordcount, stringsAsfactors=F)
str(df_word)
df_word <- rename(df_word, word = Var1, freq = Freq)

#factor에서 character로 자료형 변환
class(df_word$word)
df_word$word <- as.character(df_word$word)

#단어수 2글자 이상인 행만 남김
df_word <- filter(df_word, nchar(word) > 2)

#결과 잘나왔는지 테스트 (등장 횟수 Top 40 추출)
top_40 <- df_word %>% arrange(desc(freq)) %>% head(40)
```

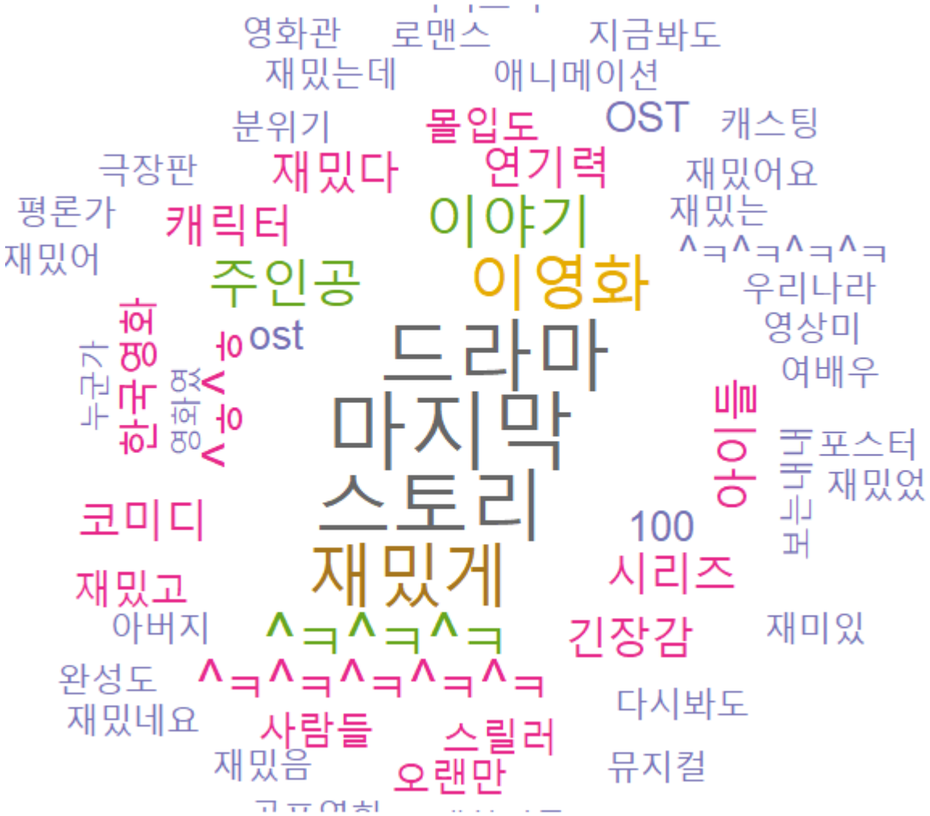
01 영화평 텍스트 마이닝 – wordcloud 시각화

- **A4. wordcloud 패키지 설치 및 로딩하여 시각화**

```
#패키지 설치
install.packages("wordcloud")

# 패키지 로드
library(wordcloud)
library(RColorBrewer)

pal <- brewer.pal(8, "Dark2") # Dark2 색상 목록에서 8개 색상 추출
pal
set.seed(1)

#wordcloud 출력
wordcloud(words = df_word$word,   #단어
          freq = round(sqrt(df_word$freq)),       #빈도 – sqrt 함수 사용하여 차이 줄임
          min.freq = 5,                           #최소 단어 빈도
          max.words = 200,                        #표현 단어 수
          random.order = F,                       #고빈도 단어 중앙 배치
          rot.per = .1,                           #회전 단어 비율
          scale = c(4, 0.5),                      #단어 크기 범위
          colors = pal)                           #색상 목록
```

01 영화평 텍스트 마이닝 - wordcloud 시각화

- **A4. wordcloud 패키지 설치 및 로딩하여 시각화**



33차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

34차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 국정원 트윗 텍스트 마이닝을 할 수 있다.
- 발라드 가사 텍스트 마이닝을 할 수 있다.

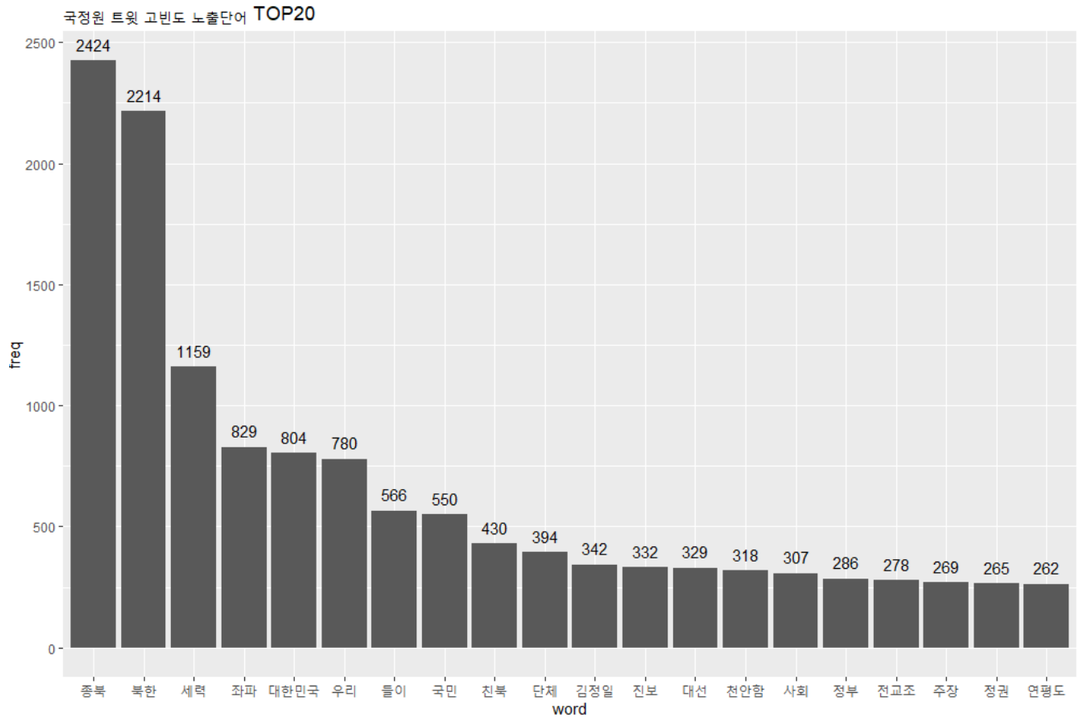
학습 목차

- 국정원 트윗 텍스트 마이닝
- 발라드 가사 텍스트 마이닝

01 트윗 텍스트 마이닝 - 목표

- **데이터셋 설명**
  - 2013년 뉴스타파에서 공개한 데이터셋
  - 국정원이 대선에 개입한 사실을 밝히는 근거자료로 공개

- **Q. 목표**
  - 네이버 리뷰 영화 평 중에서 명사만을 추출 및 카운팅하여 ggplot2 막대그래프로 출력



01 트윗 텍스트 마이닝 – 필요 패키지 로딩

- **A1. 실습에 필요한 패키지 로딩**
  - KoNLP
  - dplyr
  - stringr

```
library(KoNLP)
library(dplyr)
library(stringr)
library(ggplot2)

useNIADic() #NIA 사전
```

- **패키지 설치방법은 33차시 자료 참조**

01 트윗 텍스트 마이닝 - 데이터 로딩

- **A2. 데이터 로딩**

```
twitter <- read.csv("twitter.csv",
                    header = T,
                    stringsAsFactors = F,
                    fileEncoding = "UTF-8")
str(twitter)
```

01 트윗 텍스트 마이닝 - 데이터 전처리

- **A3. 데이터 전처리**

```
#특수문자제거
twitter$내용 <- str_replace_all(twitter$내용, '[[:punct:]]', " ")
head(twitter$내용)

#트윗에서 명사 추출
nouns <- extractNoun(twitter$내용)
wordcount <- table(unlist(nouns))

#데이터 프레임으로 변환
df_word <- as.data.frame(wordcount, stringsAsFactors = F)
df_word <- rename(df_word,
                  word = Var1,
                  freq = Freq)

#1글자 단어 제거
df_word <- filter(df_word, nchar(word) >= 2)

#빈도수 단위 top 20선정
top20 <- df_word %>% arrange(desc(freq)) %>%head(20)
```

01 트윗 텍스트 마이닝 - 데이터 시각화

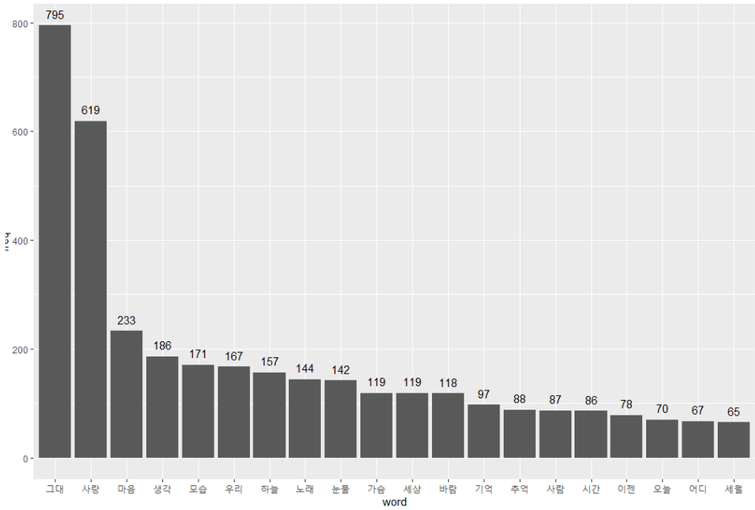
- **A4. 데이터 시각화**
  - scale_x_discrete: x축 좌표값을 범주형 값으로 변경
  - geom_col: x(범주형)별 y(값)을 막대그래프로 그려주는 함수
    - geom_bar 함수는 범주형 벡터를 막대그래프화
  - geom_text: 그래프 내에 텍스트 그려주는 함수
    - 그래프 밖에 text 적을 때는 theme_xxxx

```
#tos20 벡터 빈도별로 정렬한 텍스트 벡터(order) 생성
order <- arrange(top20, desc(freq))$word

ggplot(top20, aes(x = word, y = freq)) +
  geom_col() +
  scale_x_discrete(limit = order) +
  ggtitle("국정원 트윗 고빈도 노출단어 TOP20") +
  geom_text(aes(label = freq), vjust= -0.8)
```

02 발라드 텍스트 마이닝 - 목표

- **데이터셋 설명**
  - ➢ 발라드 가사들을 모아 놓은 데이터셋

- **목표**
  - ➢ 네이버 리뷰 영화 평 중에서 명사만을 추출 및 카운팅하여 ggplot2 막대그래프 및 CloudForm으로 출력



02 발라드 텍스트 마이닝 - 로딩 및 전처리

```
#로딩
txt <- readLines('balad.txt', encoding = 'UTF-8')  #한국어 정상적으로 로딩하기 위해 readLines사용

#데이터 전처리
data <- str_replace_all(txt, '[[:punct:]]',  " ")
data <- data[1:10000]

nouns <- extractNoun(data)  #추가 package 설치할 수 있음
wordcount <- table(unlist(nouns))

df_word <- as.data.frame(wordcount, stringsAsfactors=F)
str(df_word)
df_word <- rename(df_word, word = Var1, freq = Freq)

class(df_word$word)
df_word$word <- as.character(df_word$word)
class(df_word$word)

df_word <- filter(df_word, nchar(word) >= 2) #2글자 이상인 단어만 남김
```

02 발라드 텍스트 마이닝 - 데이터 시각화

```
pal <- brewer.pal(8, "Dark2")
set.seed(20)

#CloudForm 시각화
wordcloud(words = df_word$word,   #단어
          freq = round(sqrt(df_word$freq)),       #빈도 차이 줄임
          min.freq = 5,                    #최소 단어 빈도
          max.words = 200,                         #표현 단어 수
          random.order = F,                        #고빈도 단어 중앙 배치
          rot.per = .1,                            #회전 단어 비율
          scale = c(3, 0.5),                       #단어 크기 범위
          colors = pal)                            #색상 목록

top20 <- df_word %>% arrange(desc(freq)) %>%head(20)
order <- arrange(top20, desc(freq))$word

#막대그래프 시각화
ggplot(top20, aes(x = word, y = freq)) +
  geom_col() +
  scale_x_discrete(limit = order) +
  geom_text(aes(label = freq), vjust= -0.8)
```

34차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

35차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 미국 주 별 강력 범죄율 단계 구분도를 만들 수 있다.
- 인구 분포에 따른 강력 범죄율을 화면 분할을 이용하여 나타낼 수 있다.

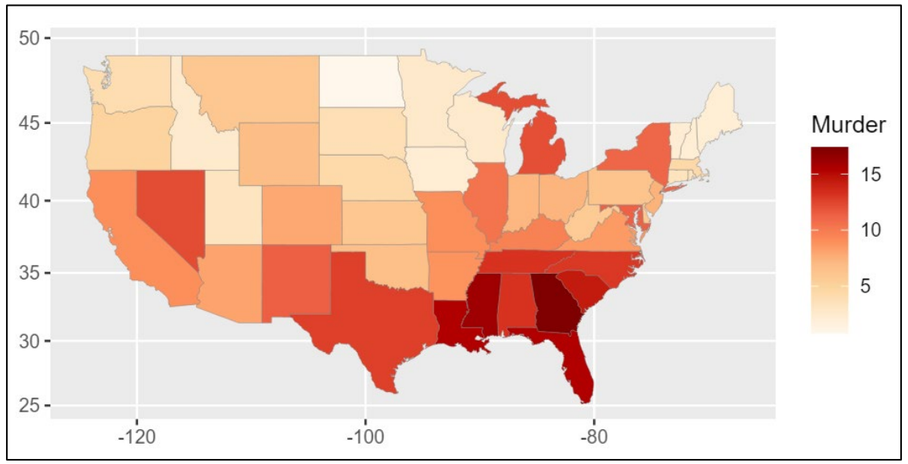
학습 목차

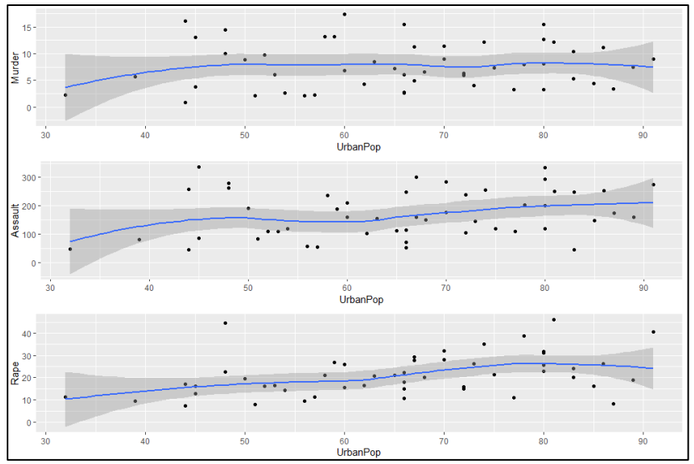
- 미국 주 별 강력 범죄율 단계 구분도 만들기
- 인구 분포에 따른 강력 범죄율을 화면 분할을 이용하여 나타내기

01 미국 강력 범죄율 – 목표

- **데이터셋 설명**
  - ➢ 내장 데이터셋 (USArrests)
  - ➢ 인구 10만명당 살인, 폭행 강간 범죄 발생 수를 주 별로 기록

- **목표**
  - ➢ 미국 지도에 주 별 강력범죄 단계 구분도 그리기
  - ➢ 인구 분포에 따른 강력 범죄율 변화를 화면 분할 사용하여 한 화면에 나타내기





01 미국 강력 범죄율 – 필요 패키지 설치

- **ggiraphExtra 패키지**
  - ggplot 기반 다양한 시각화 모형 제공
- **maps 패키지**
  - 세계 행정단위 지역의 위도, 경도 값 저장
- **gridExtra 패키지**
  - ggplot으로 만든 그래픽 자료 화면 분할 기능 제공

```
#패키지 설치
install.packages("ggiraphExtra")
install.packages('maps')
install.packages("gridExtra")
#패키지 로딩
library(ggiraphExtra)
library(tibble)
library(ggplot2)
library(gridExtra)
```

01 미국 강력 범죄율 – 데이터셋 로딩 및 전처리

- **USArrests 데이터셋 로딩 및 주 이름값 소문자로 변환**
  - tolower (): 영문자를 모두 소문자로 변환

```
#데이터셋 로딩
crime <- rownames_to_column(USArrests, var = 'state')
crime$state <- tolower(crime$state)
print(crime$state)
```

```
> print(crime$state)
 [1] "alabama"        "alaska"         "arizona"        "arkansas"       "california"
 [6] "colorado"       "connecticut"    "delaware"       "florida"        "georgia"
[11] "hawaii"         "idaho"          "illinois"       "indiana"        "iowa"
[16] "kansas"         "kentucky"       "louisiana"      "maine"          "maryland"
[21] "massachusetts"  "michigan"       "minnesota"      "mississippi"    "missouri"
[26] "montana"        "nebraska"       "nevada"         "new hampshire"  "new jersey"
[31] "new mexico"     "new york"       "north carolina" "north dakota"   "ohio"
[36] "oklahoma"       "oregon"         "pennsylvania"   "rhode island"   "south carolina"
[41] "south dakota"   "tennessee"      "texas"          "utah"           "vermont"
[46] "virginia"       "washington"     "west virginia"  "wisconsin"      "wyoming"
```

01 미국 강력 범죄율 – 미국 주 위치 값 받아오기

- **미국 주 별 위도 경도 값 가져옴**
  - map_data : maps 패키지에 존재하는 함수

```
states_map <- map_data("state")
str(states_map)
View(states_map %>% filter(region == "alabama"))
```

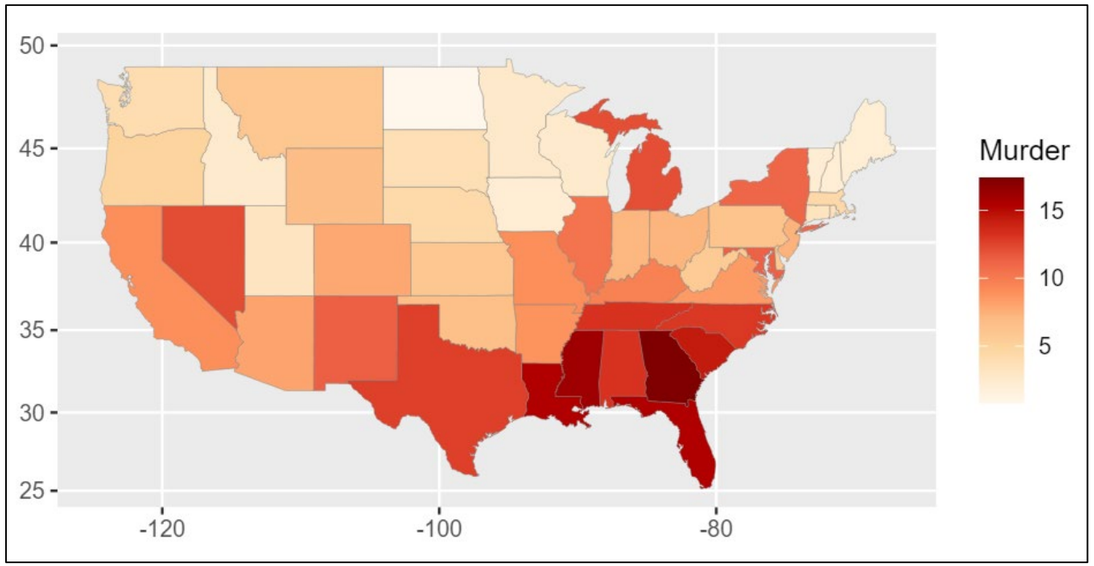
Filter

	long	lat	group	order	region	subregion
1	-87.46201	30.38968	1	1	alabama	NA
2	-87.48493	30.37249	1	2	alabama	NA
3	-87.52503	30.37249	1	3	alabama	NA
4	-87.53076	30.33239	1	4	alabama	NA
5	-87.57087	30.32665	1	5	alabama	NA
6	-87.58806	30.32665	1	6	alabama	NA
7	-87.59379	30.30947	1	7	alabama	NA
8	-87.59379	30.28655	1	8	alabama	NA
9	-87.67400	30.27509	1	9	alabama	NA
10	-87.81152	30.25790	1	10	alabama	NA
11	-87.88026	30.24644	1	11	alabama	NA
12	-87.92037	30.24644	1	12	alabama	NA
13	-87.95475	30.24644	1	13	alabama	NA
14	-88.00632	30.24071	1	14	alabama	NA
15	-88.01778	30.25217	1	15	alabama	NA
16	-88.01205	30.26936	1	16	alabama	NA
17	-87.99486	30.27509	1	17	alabama	NA
18	-87.95475	30.27509	1	18	alabama	NA
19	-87.90318	30.28082	1	19	alabama	NA
20	-87.82870	30.28655	1	20	alabama	NA
21	-87.80006	30.28655	1	21	alabama	NA
22	-87.80006	30.32665	1	22	alabama	NA

01 미국 강력 범죄율 – 단계 구분도 시각화

- **주별 살인율 단계 구분도 시각화**
  - ➢ ggChoropleth: ggiraphExtra에 존재하는 함수
    - ▪ Choropleth : 단계 구분도

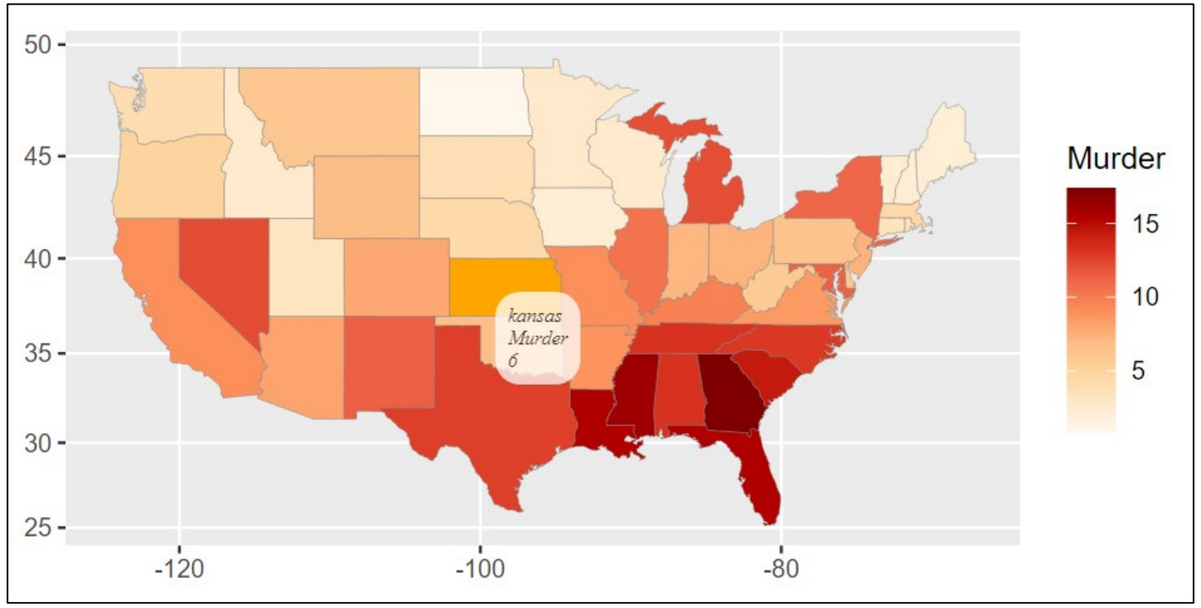
```
ggChoropleth(data = crime,
             aes(fill = Murder,
                 map_id = state),
             map = states_map)
```



01 미국 강력 범죄율 – 인터렉티브 단계 구분도

- **주별 살인율 인터렉티브 단계 구분도**
  - interactive 옵션 T로 변경

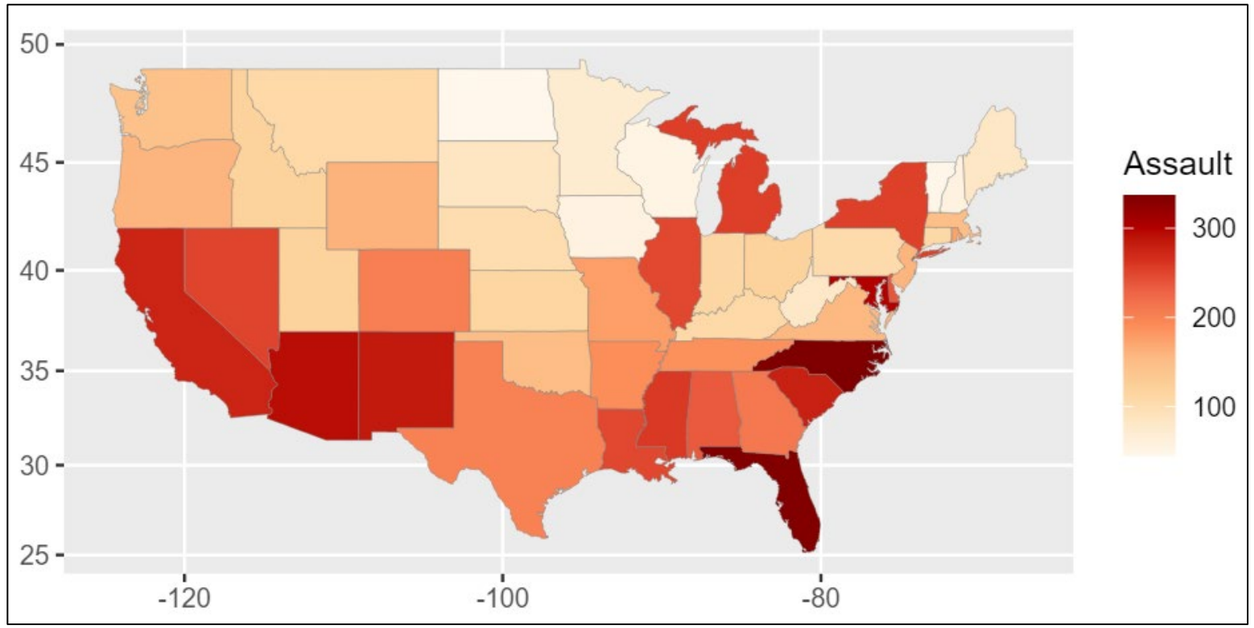
```
ggChoropleth(data = crime,
        aes(fill = Murder,
            map_id = state),
        map = states_map,
        interactive = T)
```



01 미국 강력 범죄율 – 인터렉티브 단계 구분도

- **Q. 주별 폭행율 인터렉티브 단계 구분도를 작성하세요**

```
ggChoropleth(data = crime,
             aes(fill = Assault,
                 map_id = state),
             map = states_map,
             interactive = T)
```



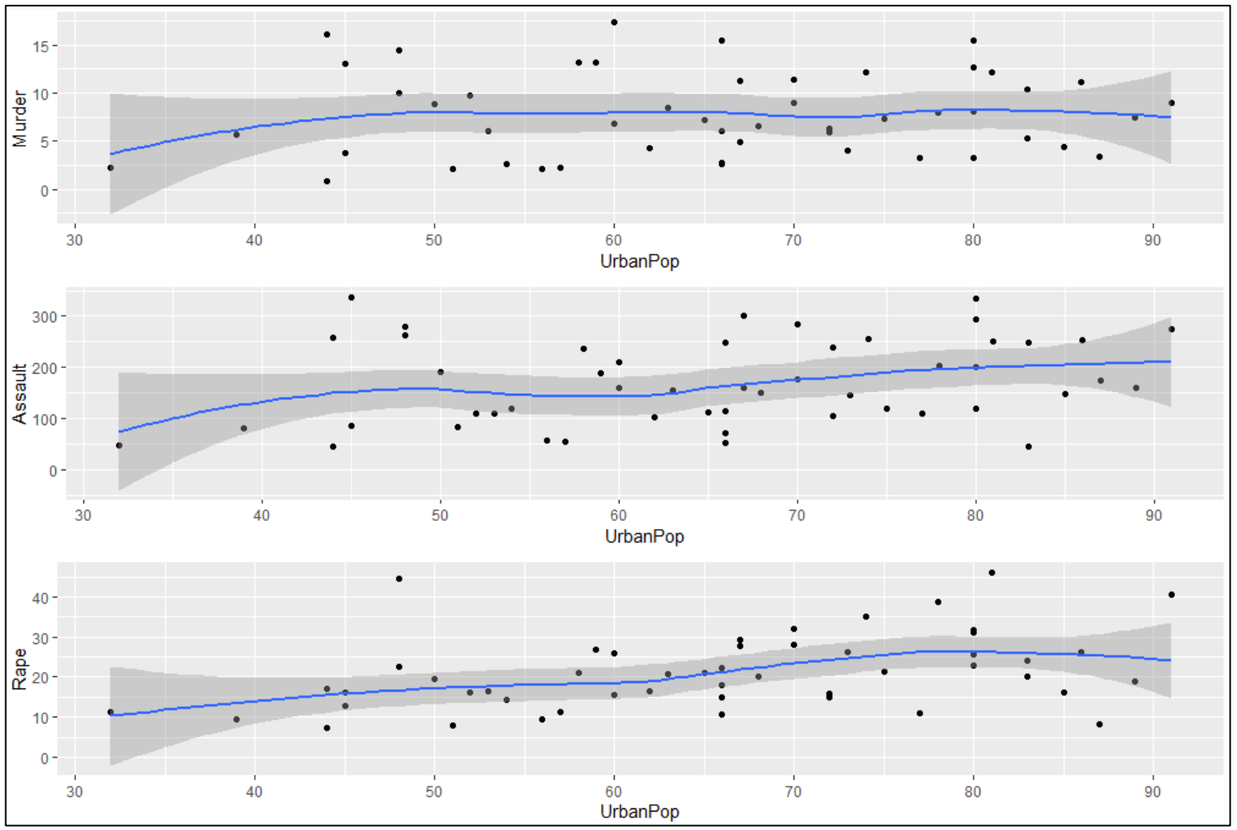
01 미국 강력 범죄율 – 화면 분할 시각화

- **인구수 대비 강력범죄 발생율 화면분할 하여 나타내기**
  - grid.arrange: gridExtra 패키지에 있는 함수

```
#나타내고 싶은 plot을 각각의 변수에 저장
p1 <- ggplot(data= crime, aes(x=UrbanPop, y=Murder))+
  geom_point() +
  stat_smooth(level = 0.9)

p2 <- ggplot(data= crime, aes(x=UrbanPop, y=Assault))+
  geom_point() +
  stat_smooth(level = 0.9)

p3 <- ggplot(data= crime, aes(x=UrbanPop, y=Rape))+
  geom_point() +
  stat_smooth(level = 0.9)


grid.arrange(p1, p2, p3)
```

01 미국 강력 범죄율 – 화면 분할 시각화

- **인구수 대비 강력범죄 발생율 화면분할 하여 나타내기**
  - grid.arrange: gridExtra 패키지에 있는 함수



35차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

36차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- sin 함수를 예측할 수 있다.
- 월별 항공기 이용객 수를 예측할 수 있다.
- 삼성전자 일별 종가를 예측할 수 있다.

학습 목차

- sin 함수 예측
- 월별 항공기 이용객 수 예측
- 삼성전자 일별 종가 예측

01 시계열 데이터 – 목표

- **시계열 데이터란?**
  - 시간 값에 따라 변하는 데이터

- **데이터셋**
  - sin 함수 데이터: 직접 만듦
  - 항공기 이용객 수 데이터: 내장 데이터셋 사용(AirPassengers)
  - 삼성전자 종가 데이터: quantmod 패키지 사용

01 시계열 데이터 – 필요 패키지 설치

- **forecast 패키지**
  - ➢ 시계열 데이터 예측 함수 제공

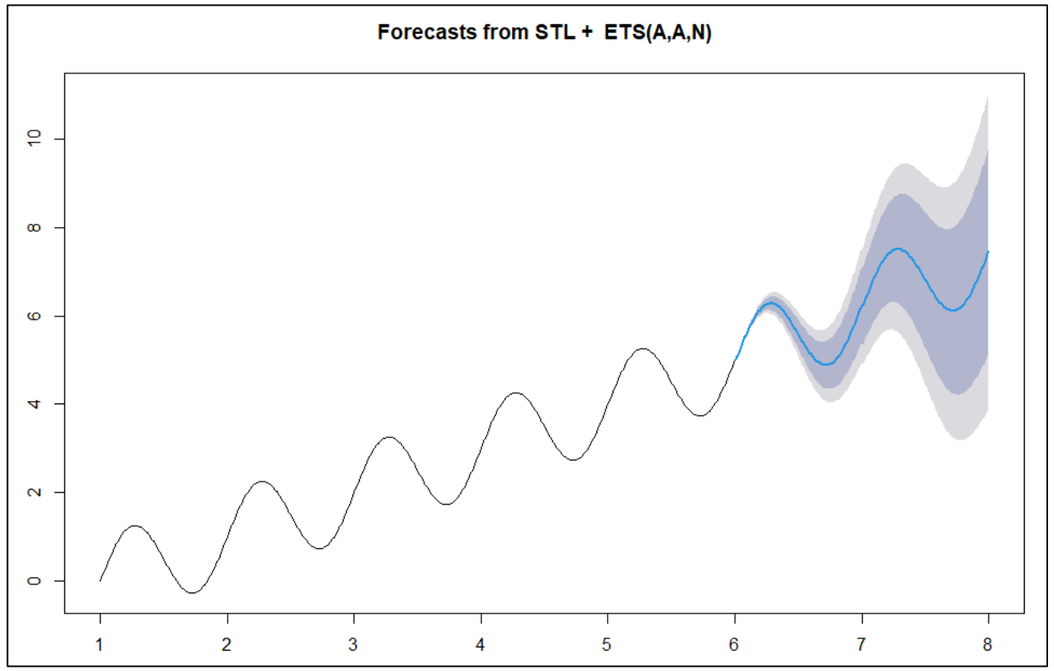
- **quantmod 패키지**
  - ➢ 한국 주가 데이터 받아오는 함수 제공

```
#패키지 설치
install.packages("forecast")
install.packages("quantmod")

#패키지 로딩
library(forecast)
library(quantmod)
```

01 시계열 데이터 - sin 함수 파형 예측

- **forecast 함수 성능 확인 목적**

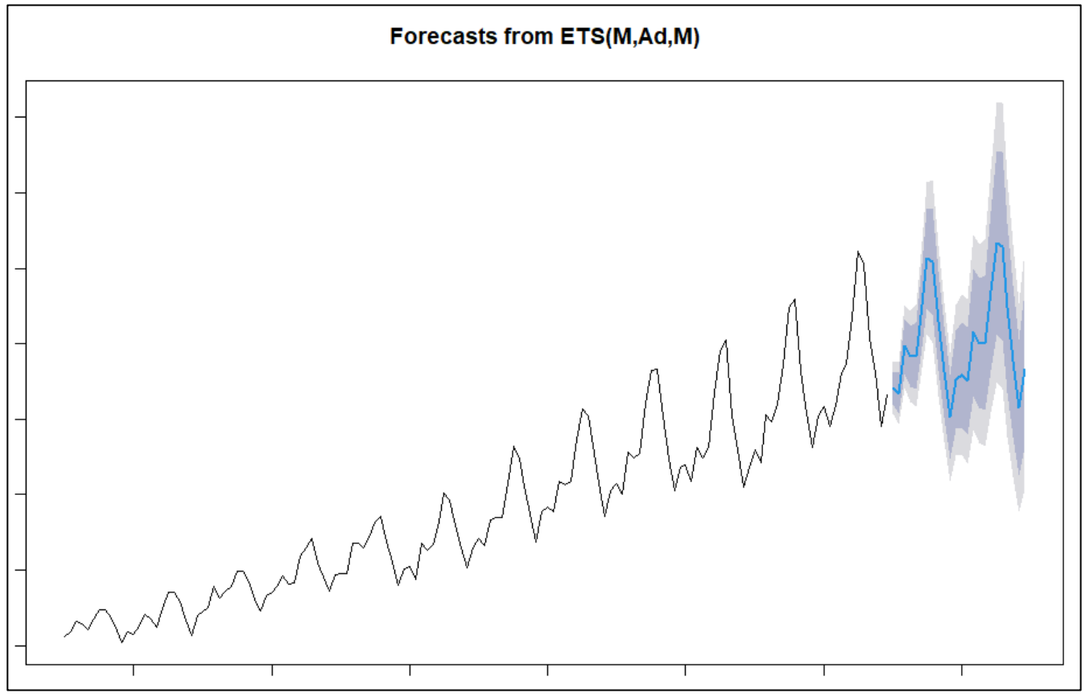
```
#sin 함수 파형 예측
x = seq(0, 5, 0.01)
y = ts(sin(2 * pi * t) + x, frequency= 100)  # ts: time-series 타입
plot(t,y, type= ' l ' )
plot(forecast(y, h=200))                    #h: 예측할 데이터 개수
```



01 시계열 데이터 – 항공기 고객 수 예측

- **forecast 함수 성능 확인 목적**

```
class(AirPassengers) # ts

plot(x= c(1:144), y=AirPassengers, type="l")
plot(forecast(AirPassengers), h=200)
```



02 시계열 데이터 – 삼성전자 종가 예측

- **Q. 삼성전가 종가 예측 시각화 문제**
  - ➢ 예측 모델에는 1월~8월까지의 데이터를 주고 9,10월 종가를 예측하도록 작성
  - ➢ 실제 가격 대비 예측 가격을 그래프로 나타내어 비교

- **A1. 데이터셋 로딩**

```
data_pred = getSymbols('005930.KS',
                from = '2021-01-01', to = '2021-09-01',
                auto.assign = FALSE)
data_real = getSymbols('005930.KS',
                    from = '2021-01-01', to = '2021-10-30',
                    auto.assign = FALSE)
```

02 삼성전자 종가 예측 – 데이터셋 로딩

- **Q. 삼성전가 종가 예측 시각화 문제**
  - ➢ 예측 모델에는 1월~8월까지의 데이터를 주고 9,10월 종가를 예측하도록 작성
  - ➢ 실제 가격 대비 예측 가격을 그래프로 나타내어 비교

- **A1. 데이터셋 로딩**

```
data_pred = getSymbols('005930.KS',                    #삼성전자 주식코드
            from = '2021-01-01', to = '2021-09-01',
            auto.assign = FALSE)
data_real = getSymbols('005930.KS',
              from = '2021-01-01', to = '2021-10-30',
              auto.assign = FALSE)
```

02 삼성전자 종가 예측 – 데이터 셋 전처리

- **전처리 순서**
  - data 는 행 이름을 가지고 있으므로 행 이름 값을 ‘date’ 컬럼으로 이동
  - ‘date’ 컬럼은 Date 타입으로 형 변환

```
names(data_pred) <- c("open", "high", "low", "close", "volume", "adjusted")
names(data_real) <- c("open", "high", "low", "close", "volume", "adjusted")

data_pred <- rownames_to_column(as.data.frame(data_pred), var = 'date')
data_real <- rownames_to_column(as.data.frame(data_real), var = 'date')

data_pred$date <- as.Date(data_pred$date)
data_real$date <- as.Date(data_real$date)
```

02 삼성전자 종가 예측 – 종가 값 예측

- **값 예측 순서**
  - ➢ forecast 함수 data_real 행 수 – data_pred 행 수 값 만큼 데이터 예측
    - ▪ 주식장은 매일 열리지 않음

```
pred_length <- length(data_real$close)-length(data_pred$close)

test <- forecast(data_pred$close, h=)
test <- as.data.frame(test)
length(test$`Point Forecast`)
```

02 삼성전자 종가 예측 – 예측 결과 시각화

- **시각화 순서**
  - data_pred의 종가값과 예측 값을 합쳐서 data_real에 'pred_close'라는 새로운 컬럼으로 등록
  - ggplot 사용하여 선 그래프 출력
    - 예측값에 대한 선은 빨간색으로 지정

```
data_real[,"pred_close"] <- c(data_pred$close, test$`Point Forecast`)
str(data_real)

datebreaks <- seq(as.Date("2021-01-01"), as.Date("2021-10-01"), by="1 month")
ggplot(data=data_real, aes(x=date)) +
  geom_line(aes(y=pred_close, col='red')) +                    # 예측선 먼저 빨간색으로 그린다
  geom_line(aes(y=close)) +
  scale_x_date(breaks=datebreaks) +                            #Theme layer. x축을 date값으로 나타냄
  theme(axis.text.x = element_text(angle=30, hjust=1))         # x축 값 텍스트를 30도 회전
```

36차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

37차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 가설검정을 이해할 수 있다.
- Z-검정을 이해할 수 있다.
- R을 사용하여 Z-검정을 할 수 있다.

학습 목차

- 가설검정의 이해
- Z-검정 이해
- R을 사용한 Z-검정

01 가설검정이란? - 모집단과 표본

- **모집단에 대한 가설을 모집단으로 부터 추출한 표본을 사용하여 검토하는 추론방법**

- **모집단**
  - 정보를 얻고자 하는 관심 대상의 전체집합
  - 예시
    - 목표 정보: 20대 남성 평균 키
    - 모집단: 모든 20대 남성의 키 집합

- **표본**
  - 모집단으로부터 샘플링 하여 얻어지는 결과
  - 모집단의 부분집합

01 가설검정이란? - 가설 설정

- **20대 남성 키 데이터셋 수집**
  - 데이터셋 == 표본

- **가설 설정(문제 설정)**
  - 20대 남성 평균 신장은 175로 알려져있다. 아래 수집한 데이터셋을 보았을 때, 이는 타당한가?

모집단에 대한 가설을 표본을 사용해 검토

20대 남성 평균 신장이 175인지 데이터셋을 사용해 검토

01 가설검정이란? - 귀무 가설과 대립 가설

- **귀무가설**
  - 차이가 없는 경우의 가설
  - 처음부터 버릴 것을 예상하는 가설
  - Eg.) 20대 남성 평균 키는 175가 맞다

- **대립가설**
  - 차이가 있는 경우의 가설
  - 검증하고 싶은 가설
  - Eg.) 20대 남성 평균 키는 175가 아니다

01 가설검정이란? – 판단 원리

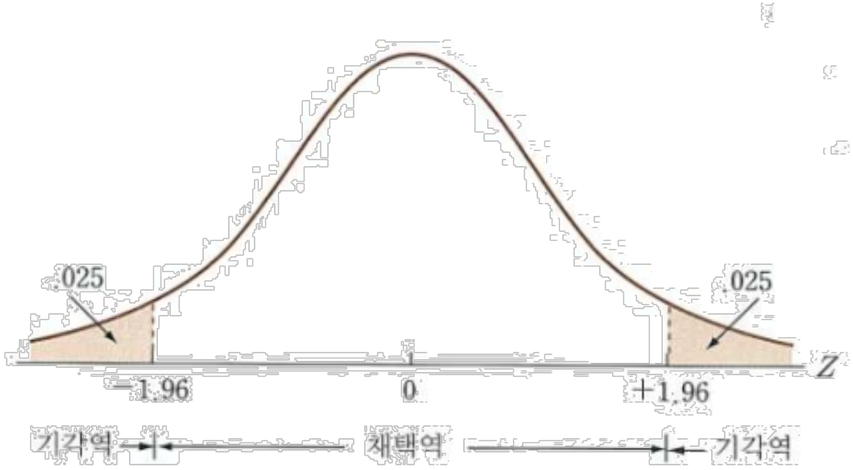
- **원리**
  - 통계량을 계산 한 후, 유의 수준을 넘는지 비교
- **통계량**
  - 판단을 위해서 계산하여 얻는1종 오류가 발생할 확률의 허용한계 (보통 5%) 특정 값
  - z값, t값 등..
  - z값을 사용한다면 z-test, t값을 사용한다면 t-test, 등.. 으로 부름
- **유의수준**
  - 1종 오류가 발생할 확률의 허용한계 (보통 5%)

	귀무가설이 참 (평균 키 175 맞음)	귀무가설이 거짓 (평균 키 175 아님)
귀무가설 기각 X	옳은 결정	2종 오류
귀무가서 기각 O	1종 오류 (원래 맞던 걸 틀렸다 함)	옳은 결정

02 Z-검정 - 검정 방법

- **가정**
  - 모집단은 정규 분포를 따른다고 가정

- **통계량: Z값**
  - $Z = \frac{\bar{X} - \mu}{\sigma / \sqrt{n}}$
    - (μ, σ): 모집단 평균, 표준 편차
    - $\bar{X}$: 표본 평균
    - n: 데이터 개수



- **통계량: Z값**
  - 정규분포에서는 유의 수준에 따른 비교 값이 고정
    - 1% : 2.58, 5% : 1.96, 10%: 1.64
  - 모집단의 평균과 표준편차를 알 때 사용
  - 유의 수준 값 보다 z값이 크다면 귀무가설 기각

02 Z-검정 - 검정 예시

- **문제풀이**
  - 모집단 평균 175, 분산 6 이라 가정
  - 유의수준 5%, 1% 에서 z-검정 여부는?

```
# 모집단 평균 175, 분산 6
data <- read_excel('man_height.xlsx')                    # 데이터 다운로드
z <- (mean(data$키)-175)/(6/sqrt(length(data$키)))  # z값 계산
print(z)                                                 # 2.373464
```

유의수준 5%: 1.96 < 2.373464 = T　　귀무가설 기각 == 남성 평균 키는 175가 아니다

유의수준 1%: 2.56 < 2.373464 = F　　귀무가설 유지 == 남성 평균 키는 175가 맞다

- **결과를 말할 때는 언제나 '유의수준 ~%에서 귀무가설을 기각/유지 한다'라고 붙여 말함**

03 R을 사용한 Z-검정 - 예제

- **직접 계산**

```
library(readxl)
library(ggplot2)
data <- read_excel('man_height.xlsx')

# 모집단 평균 175, 분산 6
# 유의 수준 0.05 ---> 1.96
alpha <- 1.96
data <- read_excel('man_height.xlsx')
#귀무 가설: 남자 키 평균은 175다
#대립 가설: 남자 키 평균은 175가 아니다
z <- (mean(data$키)-175)/(6/sqrt(length(data$키)))
if(abs(z) > 1.96) {
  print("귀무가설 기각. 남자 평균 키는 175가 아니다")
} else {
    print("귀무가설 기각. 남자 평균 키는 175다")
}
```

03 R을 사용한 Z-검정 – 예제

- **z.test 함수 사용**

```
install.packages("BSDA")
library("BSDA")
z.test(x=data, alternative = “two.sided", mu=175, sigma.x=6, conf.level = 0.95) # conf.level = 1 – 유의 수준
```

```
data:  data
z = 2.3735, p-value = 0.01762
alternative hypothesis: true mean is not equal to 175
95 percent confidence interval:
 175.453 179.747
sample estimates:
mean of x
    177.6
```

37차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

01 시계열 데이터 – 목표

- **시계열 데이터란?**
  - 시간 값에 따라 변하는 데이터

- **데이터셋**
  - sin 함수 데이터: 직접 만듦
  - 항공기 이용객 수 데이터: 내장 데이터셋 사용(AirPassengers)
  - 삼성전자 종가 데이터: quantmod 패키지 사용

01 시계열 데이터 – 필요 패키지 설치

- **forecast 패키지**
  - ➢ 시계열 데이터 예측 함수 제공

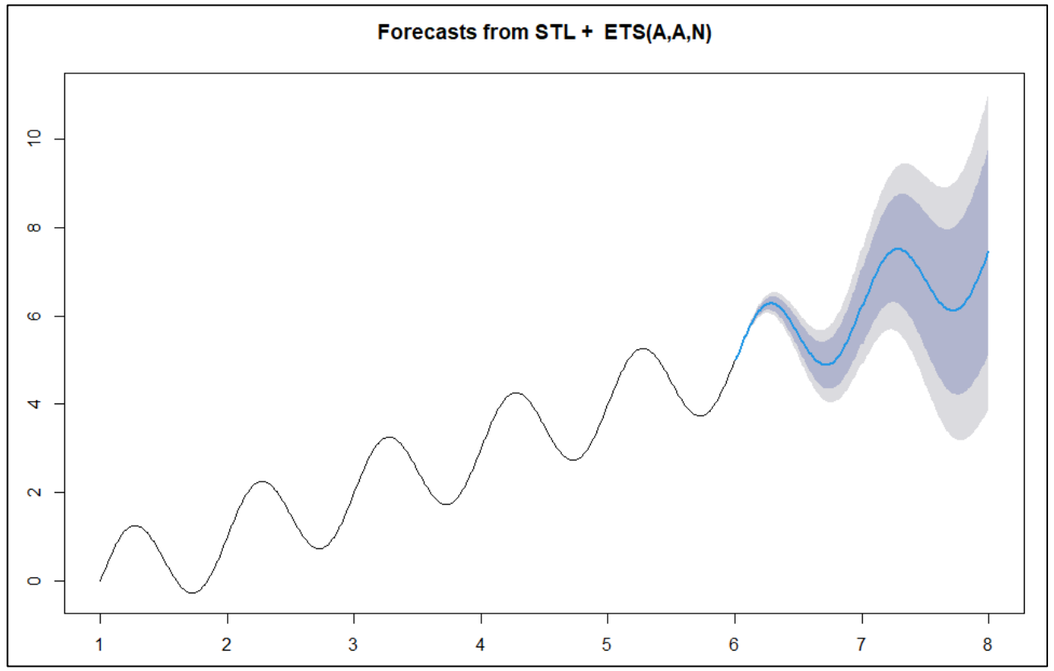
- **quantmod 패키지**
  - ➢ 한국 주가 데이터 받아오는 함수 제공

```
#패키지 설치
install.packages("forecast")
install.packages("quantmod")

#패키지 로딩
library(forecast)
library(quantmod)
```

01 시계열 데이터 - sin 함수 파형 예측

- **forecast 함수 성능 확인 목적**

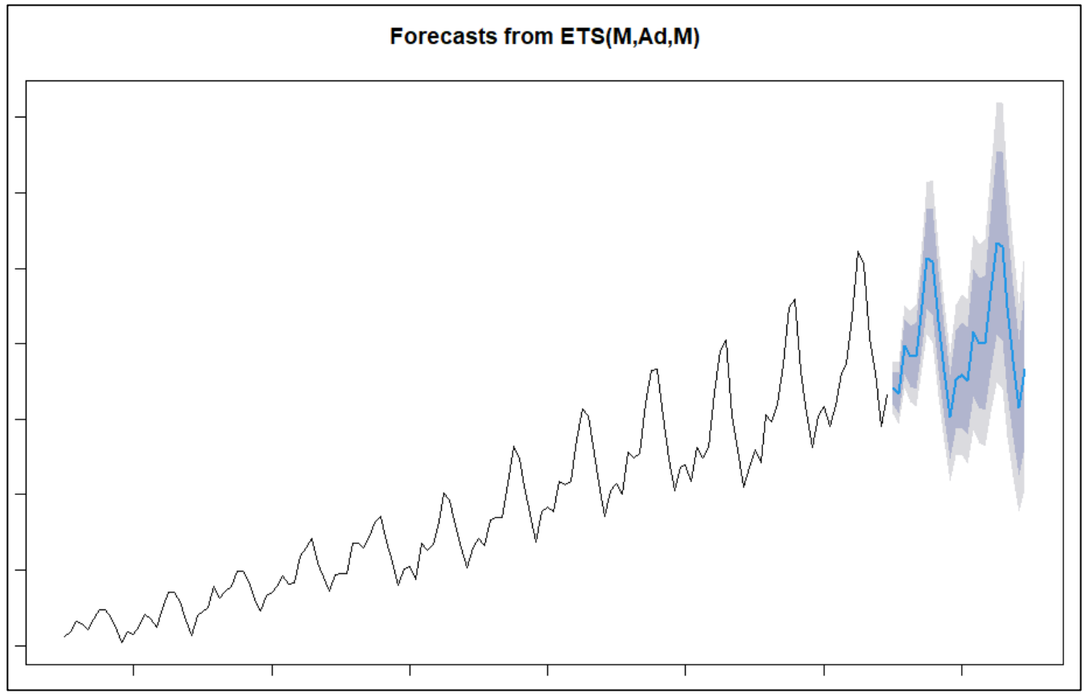
```
#sin 함수 파형 예측
x = seq(0, 5, 0.01)
y = ts(sin(2 * pi * t) + x, frequency= 100)  # ts: time-series 타입
plot(t,y, type= ' l ' )
plot(forecast(y, h=200))                    #h: 예측할 데이터 개수
```



01 시계열 데이터 – 항공기 고객 수 예측

- **forecast 함수 성능 확인 목적**

```
class(AirPassengers) # ts

plot(x= c(1:144), y=AirPassengers, type="l")
plot(forecast(AirPassengers), h=200)
```



02 시계열 데이터 – 삼성전자 종가 예측

- **Q. 삼성전가 종가 예측 시각화 문제**
  - ➢ 예측 모델에는 1월~8월까지의 데이터를 주고 9,10월 종가를 예측하도록 작성
  - ➢ 실제 가격 대비 예측 가격을 그래프로 나타내어 비교

- **A1. 데이터셋 로딩**

```
data_pred = getSymbols('005930.KS',
                from = '2021-01-01', to = '2021-09-01',
                auto.assign = FALSE)
data_real = getSymbols('005930.KS',
                    from = '2021-01-01', to = '2021-10-30',
                    auto.assign = FALSE)
```

02 삼성전자 종가 예측 – 데이터셋 로딩

- **Q. 삼성전가 종가 예측 시각화 문제**
  - ➢ 예측 모델에는 1월~8월까지의 데이터를 주고 9,10월 종가를 예측하도록 작성
  - ➢ 실제 가격 대비 예측 가격을 그래프로 나타내어 비교

- **A1. 데이터셋 로딩**

```
data_pred = getSymbols('005930.KS',                    #삼성전자 주식코드
            from = '2021-01-01', to = '2021-09-01',
            auto.assign = FALSE)
data_real = getSymbols('005930.KS',
              from = '2021-01-01', to = '2021-10-30',
              auto.assign = FALSE)
```

02 삼성전자 종가 예측 – 데이터 셋 전처리

- **전처리 순서**
  - data 는 행 이름을 가지고 있으므로 행 이름 값을 ‘date’ 컬럼으로 이동
  - ‘date’ 컬럼은 Date 타입으로 형 변환

```
names(data_pred) <- c("open", "high", "low", "close", "volume", "adjusted")
names(data_real) <- c("open", "high", "low", "close", "volume", "adjusted")

data_pred <- rownames_to_column(as.data.frame(data_pred), var = 'date')
data_real <- rownames_to_column(as.data.frame(data_real), var = 'date')

data_pred$date <- as.Date(data_pred$date)
data_real$date <- as.Date(data_real$date)
```

02 삼성전자 종가 예측 – 종가 값 예측

- **값 예측 순서**
  - forecast 함수 data_real 행 수 – data_pred 행 수 값 만큼 데이터 예측
    - 주식장은 매일 열리지 않음

```
pred_length <- length(data_real$close)-length(data_pred$close)

test <- forecast(data_pred$close, h=)
test <- as.data.frame(test)
length(test$`Point Forecast`)
```

02 삼성전자 종가 예측 – 예측 결과 시각화

- **시각화 순서**
  - data_pred의 종가값과 예측 값을 합쳐서 data_real에 'pred_close'라는 새로운 컬럼으로 등록
  - ggplot 사용하여 선 그래프 출력
    - 예측값에 대한 선은 빨간색으로 지정

```
data_real[,"pred_close"] <- c(data_pred$close, test$`Point Forecast`)
str(data_real)

datebreaks <- seq(as.Date("2021-01-01"), as.Date("2021-10-01"), by="1 month")
ggplot(data=data_real, aes(x=date)) +
  geom_line(aes(y=pred_close, col='red')) +                    # 예측선 먼저 빨간색으로 그린다
  geom_line(aes(y=close)) +
  scale_x_date(breaks=datebreaks) +                            #Theme layer. x축을 date값으로 나타냄
  theme(axis.text.x = element_text(angle=30, hjust=1))         # x축 값 텍스트를 30도 회전
```

36차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

38차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- t 검정을 할 수 있다.
- 단측 검정과 양측 검정을 할 수 있다.

학습 목차

- t 검정
- 단측 검정과 양측 검정

01 t-검정 - 검정 방법

- **가정**
  - ➢ 모집단은 정규 분포를 따른다고 가정

- **통계량: t값**
  - ➢ $t = \frac{\bar{X} - \mu}{S/\sqrt{n}}$
    - ▪ μ: 모집단 평균
    - ▪ $(\bar{X}, S)$: 표본 평균, 표준편차
    - ▪ n: 데이터 개수

- **t-검정**
  - ➢ 모집단의 표준편차를 모를 때, 표본의 표준편차를 사용해 검정하는 방법
  - ➢ 유의수준이 주어졌을 때, t 값이 t분포표 값보다 크다면 귀무가설 기각

α: 유의수준

df: 자유도 = n-1

df \ α	0.4	0.25	0.1	0.05	0.025	0.01	0.005	0.0025	0.001	0.0005
1	0.325	1.000	3.078	6.314	12.706	31.821	63.657	127.32	318.31	636.62
2	0.289	0.816	1.886	2.920	4.303	6.965	9.925	14.089	22.327	31.599
3	0.277	0.765	1.638	2.353	3.182	4.541	5.841	7.453	10.215	12.924
4	0.271	0.741	1.533	2.132	2.776	3.747	4.604	5.598	7.173	8.610
5	0.267	0.727	1.476	2.015	2.571	3.365	4.032	4.773	5.893	6.869
6	0.265	0.718	1.440	1.943	2.447	3.143	3.707	4.317	5.208	5.959
7	0.263	0.711	1.415	1.895	2.365	2.998	3.499	4.029	4.785	5.408
8	0.262	0.706	1.397	1.860	2.306	2.896	3.355	3.833	4.501	5.041
9	0.261	0.703	1.383	1.833	2.262	2.821	3.250	3.690	4.297	4.781
10	0.260	0.700	1.372	1.812	2.228	2.764	3.169	3.581	4.144	4.587
11	0.260	0.697	1.363	1.796	2.201	2.718	3.106	3.497	4.025	4.437
12	0.259	0.695	1.356	1.782	2.179	2.681	3.055	3.428	3.930	4.318
13	0.259	0.694	1.350	1.771	2.160	2.650	3.012	3.372	3.852	4.221
14	0.258	0.692	1.345	1.761	2.145	2.624	2.977	3.326	3.787	4.140
15	0.258	0.691	1.341	1.753	2.131	2.602	2.947	3.286	3.733	4.073
16	0.258	0.690	1.337	1.746	2.120	2.583	2.921	3.252	3.686	4.015
17	0.257	0.689	1.333	1.740	2.110	2.567	2.898	3.222	3.646	3.965
18	0.257	0.688	1.330	1.734	2.101	2.552	2.878	3.197	3.610	3.922
19	0.257	0.688	1.328	1.729	2.093	2.539	2.861	3.174	3.579	3.883
20	0.257	0.687	1.325	1.725	2.086	2.528	2.845	3.153	3.552	3.850
21	0.257	0.686	1.323	1.721	2.080	2.518	2.831	3.135	3.527	3.819
22	0.256	0.686	1.321	1.717	2.074	2.508	2.819	3.119	3.505	3.792
23	0.256	0.685	1.319	1.714	2.069	2.500	2.807	3.104	3.485	3.768
24	0.256	0.685	1.318	1.711	2.064	2.492	2.797	3.091	3.467	3.745
25	0.256	0.684	1.316	1.708	2.060	2.485	2.787	3.078	3.450	3.725
26	0.256	0.684	1.315	1.706	2.056	2.479	2.779	3.067	3.435	3.707
27	0.256	0.684	1.314	1.703	2.052	2.473	2.771	3.057	3.421	3.690
28	0.256	0.683	1.313	1.701	2.048	2.467	2.763	3.047	3.408	3.674
29	0.256	0.683	1.311	1.699	2.045	2.462	2.756	3.038	3.396	3.659
30	0.256	0.683	1.310	1.697	2.042	2.457	2.750	3.030	3.385	3.646
40	0.255	0.681	1.303	1.684	2.021	2.423	2.704	2.971	3.307	3.551
60	0.254	0.679	1.296	1.671	2.000	2.390	2.660	2.915	3.232	3.460
120	0.254	0.677	1.289	1.658	1.980	2.358	2.617	2.860	3.160	3.373
∞	0.253	0.674	1.282	1.645	1.960	2.326	2.576	2.807	3.090	3.291

t 분포표

01 t-검정 - 검정 예시

- **문제풀이**
  - 문제: 37차시 참고
  - 모집단 평균 175
  - 유의수준 5%에서 t-검정 여부는?

```
# 모집단 평균 175
data <- read_excel('man_height.xlsx')
length(data$키) #30. 즉, 자유도는 29
t <- (mean(data$키)-175)/(sd(data$키)/sqrt(length(data$키)))  # t값 계산
print(t)                                  # 1.921312
```

- **양측검정이기에 $\frac{\alpha}{2}$ 값 사용**

df \ α	0.4	0.25	0.1	0.05	0.025
1	0.325	1.000	3.078	6.314	12.706
2	0.289	0.816	1.886	2.920	4.303
3	0.277	0.765	1.638	2.353	3.182
4	0.271	0.741	1.533	2.132	2.776
5	0.267	0.727	1.476	2.015	2.571
6	0.265	0.718	1.440	1.943	2.447
7	0.263	0.711	1.415	1.895	2.365
8	0.262	0.706	1.397	1.860	2.306
9	0.261	0.703	1.383	1.833	2.262
10	0.260	0.700	1.372	1.812	2.228
11	0.260	0.697	1.363	1.796	2.201
12	0.259	0.695	1.356	1.782	2.179
13	0.259	0.694	1.350	1.771	2.160
14	0.258	0.692	1.345	1.761	2.145
15	0.258	0.691	1.341	1.753	2.131
16	0.258	0.690	1.337	1.746	2.120
17	0.257	0.689	1.333	1.740	2.110
18	0.257	0.688	1.330	1.734	2.101
19	0.257	0.688	1.328	1.729	2.093
20	0.257	0.687	1.325	1.725	2.086
21	0.257	0.686	1.323	1.721	2.080
22	0.256	0.686	1.321	1.717	2.074
23	0.256	0.685	1.319	1.714	2.069
24	0.256	0.685	1.318	1.711	2.064
25	0.256	0.684	1.316	1.708	2.060
26	0.256	0.684	1.315	1.706	2.056
27	0.256	0.684	1.314	1.703	2.052
28	0.256	0.683	1.313	1.701	2.048
29	0.256	0.683	1.311	1.699	2.045
30	0.256	0.683	1.310	1.697	2.042

유의수준 5%: 2.045 < 1.921312 = F　　귀무가설 채용 == 남성 평균 키는 175 이다

02 R을 사용한 t-검정 - 예제

- **직접 계산**

```
library(readxl)
library(ggplot2)
data <- read_excel('man_height.xlsx')

# 모집단 평균 175
# 유의 수준 0.05
# 양측 검정 : TRUE
data <- read_excel('man_height.xlsx')
#귀무 가설: 남자 키 평균은 175다
#대립 가설: 남자 키 평균은 175가 아니다
t <- (mean(data$키)-175)/(sd(data$키)/sqrt(length(data$키)))
tval <- qt(1- 0.05/2,29)              # qt: t분포표 값 얻을 수 있는 함수
print(tval)                           # 2.04523
if(abs(t) > tval) {
  print("귀무가설 기각. 남자 평균 키는 175가 아니다")
} else {
    print("귀무가설 기각. 남자 평균 키는 175다")
}
```

02 R을 사용한 t-검정 – 예제

- **t.test 함수 사용**

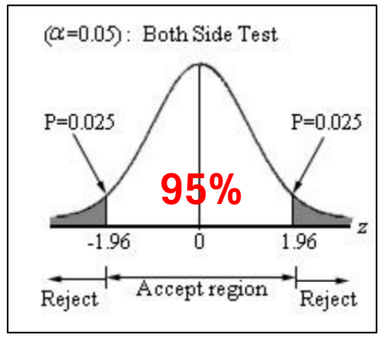
```
install.packages("BSDA")
library("BSDA")
z.test(x=data, alternative = "two.sided", mu=175, sigma.x=6, conf.level = 0.95)
```

  - 유의수준 conf.level에서 귀무가설을 기각

```
        One Sample t-test

data:  data
t = 1.9213, df = 29, p-value = 0.06457
alternative hypothesis: true mean is not equal to 175
95 percent confidence interval:
 174.8323 180.3677
sample estimates:
mean of x
    177.6
```

03 양측 검정, 단측 검정 – 양측 검정이란?

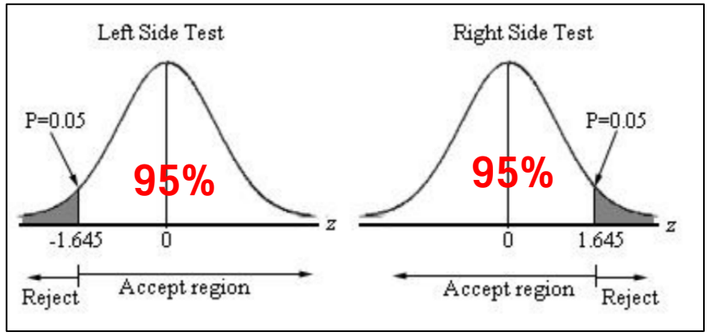
- **양측 검정**
  - ➢ 대립가설이 "같지 않다" 라는 조건으로 주어지는 경우 사용
    - ▪ 귀무가설: $\mu=\mu_0$
    - ▪ 대립가설: $\mu\neq\mu_0$
  - ➢ '같지 않을' 조건은 특정 값 보다 많이 크거나, 많이 작으면 됨
    - ▪ 양쪽 바깥 영역의 밀도가 유의 수준과 같아져야 함
- **통계량을 유의수준 /2 값과 비교해야 함**



[정규 분포에서 유의수준 0.05, 양측 검정]

03 양측 검정, 단측 검정 – 양측 검정이란?

- **단측 검정**
  - ➢ 대립가설이 ”크다”, “작다” 라는 조건으로 주어지는 경우 사용
    - ▪ 귀무가설: $\mu=\mu_0, \mu\geq\mu_0, \mu\leq\mu_0$
    - ▪ 대립가설: $\mu>\mu_0, \mu<\mu_0$
  - ➢ ‘작을’ 조건은 특정 값 보다 많이 작은 경우, 한 방향 뿐
    - ▪ ‘클’ 조건도 동일
    - ▪ 한쪽 바깥 영역의 밀도가 유의 수준과 같아져야 함
- **통계량을 유의수준 값과 비교해야 함**



[정규 분포에서 유의수준 0.05, 단측 검정]

04 코드 예시 – 양측 검정, 단측 검정

- **z.test, t.test 함수의 alternative 매개변수**

```
library("BSDA")

data <- read_excel('man_height.xlsx')
z.test(x=data, alternative = "greater", mu=175, sigma.x=6, conf.level = 0.95)
z.test(x=data, alternative = "less", mu=175, sigma.x=6, conf.level = 0.95)
t.test(x=data, alternative = "two.sided", mu=175, conf.level = 0.95)
```

- **alternative 매개변수 별 의미 확인**

- **z.test, t.test 함수에서 유의 수준을 별도로 조정할 필요는 없음**
  - ➢ 직접계산 시 조정 필요

05 z검정, t검정의 전제

- **두 집단 간의 검정 방법**
  - 3개 이상 집단간의 검정은 다른 방법론 필요

- **아래 3가지 조건을 만족해야 z검정, t검정 사용가능**
  - 1. 독립성
    - 한 집단에서 사용한 표본을 다른 집단에서 사용하지 않음
  - 2. 정규성 (*중요*)
    - 표본 데이터가 정규분포를 따름
    - Shapiro-wilk 검정
  - 3. 등분산성독립성
    - 집단 내 특정 샘플이 선택될 확률은 모두 같음
    - Levene 검정
  - 실데이터셋 검정 시, 위 3가지 조건 선제적으로 분석 필요

38차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

39차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 상관분석에 대해 알 수 있다.
- 상관계수를 이해할 수 있다.
- R기반 상관분석의 방법을 이해할 수 있다.

학습 목차

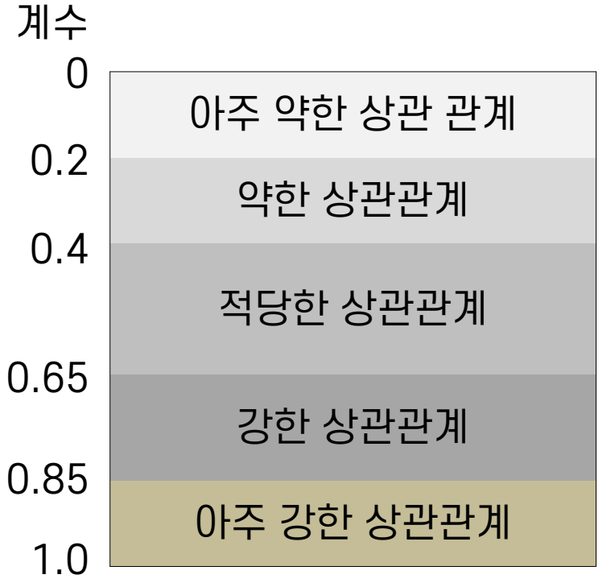
- 상관분석이란?
- 상관계수의 이해
- R기반 상관분석 방법 이해

01 상관분석이란?

- **두 변수 간에 어떤 관계를 가지는지 분석하는 기법**

- **상관계수(Correlation coefficient) 사용**
  - 상관관계의 강도를 나타내는 척도
  - 0~1 사이 값으로 정의

- **상관계수 종류**
  - 피어슨(Pearson) 상관계수
  - 켄달(Kendall) 상관계수
  - 스피어만(Spearman) 상관계수
  - Etc.



02 상관계수 설명 - iris 데이터셋 설명

- **붓꽃 꽃받침과 잎의 길이, 넓이 값과 붓꽃 종류 저장한 데이터셋**
- **데이터셋 구성**
  - 150개 데이터
  - 열 구성
    - Sepal.Length: 꽃받침 길이
    - Sepal.Width: 꽃받침 넓이
    - Petal.Length: 꽃받침 길이
    - Petal.Width: 꽃받침 넓이
    - Species: 3종류(setosa, versicolor, virginica)
- **종류에 따른 길이, 두께 값이 비교적 명확하게 구분 되어있어 상관 분석 예시로 주로 사용**

02 상관계수 설명 - 피어슨 상관계수

- **연속형 변수의 선형적인 상관관계 측정 구성**
  - Eg.) 신장과 몸무게의 상관관계 등
  - 가장 기본적으로 사용되는 상관계수

- **예시1**

```
data <- iris   # iris 데이터셋 사용. 내장 데이터셋
cor(data$Sepal.Length, data$Petal.Width)        #꽃받침 길이와 꽃잎 넓이 변수간의 피어슨 상관계수 측정

# 귀무 가설: '두 변수는 서로 상관 관계가 없다'인 가설 검정 결과도 출력 (상관계수도 출력)
# p-value가 0.05보다 작다면 귀무 가설을 기각. 즉, 5% 유의수준아래 두 변수는 상관 있다
cor.test(data$Sepal.Length, data$Petal.Width)
```

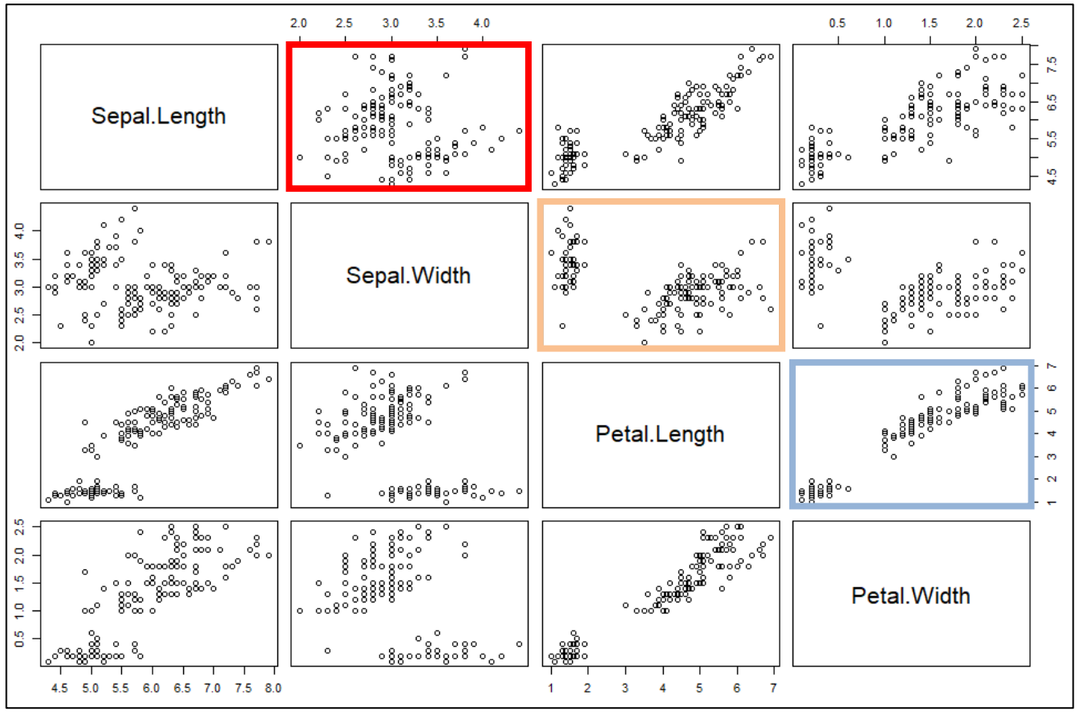
- **예시2**

```
cor(data[,1:4])          # 여러 개 변수 상관분석
pairs(data[,1:4])         # 변수별 산점도 그래프 출력 (시각화)
```

02 상관계수 설명 - 피어슨 상관계수 분석

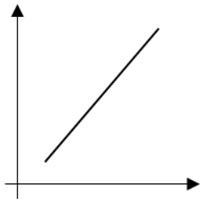
- 예시2 결과

```
> cor(data[,1:4])
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000   -0.1175698    0.8717538   0.8179411
Sepal.Width    -0.1175698    1.0000000   -0.4284401  -0.3661259
Petal.Length    0.8717538   -0.4284401    1.0000000   0.9628654
Petal.Width     0.8179411   -0.3661259    0.9628654   1.0000000
```

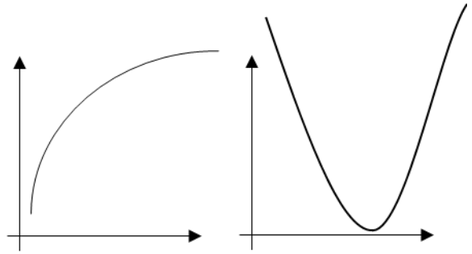


02 상관계수 설명 – 켄달, 스피어만 상관계수

- **순서가 있는(ordered) 변수의 상관관계 측정**
- **순서가 있는 범주형 데이터 분석가능**
  - Eg.) 게임 레벨에 따른 승률, 과학 등수에 따른 수학 등수
    - 한 축의 순서가 올라갈 때, 다른 축의 순서가 올라가는지에 대한 검정
    - 단조 증가, 단조 감수 하는 데이터에 큰 상관관계 부여
  - 당연히, 수치형 데이터 가능
- **비선형적인 상관관계 분석 가능**



[선형]



[비선형]

02 상관계수 설명 – 켄달 상관계수 예시

- **예시1**

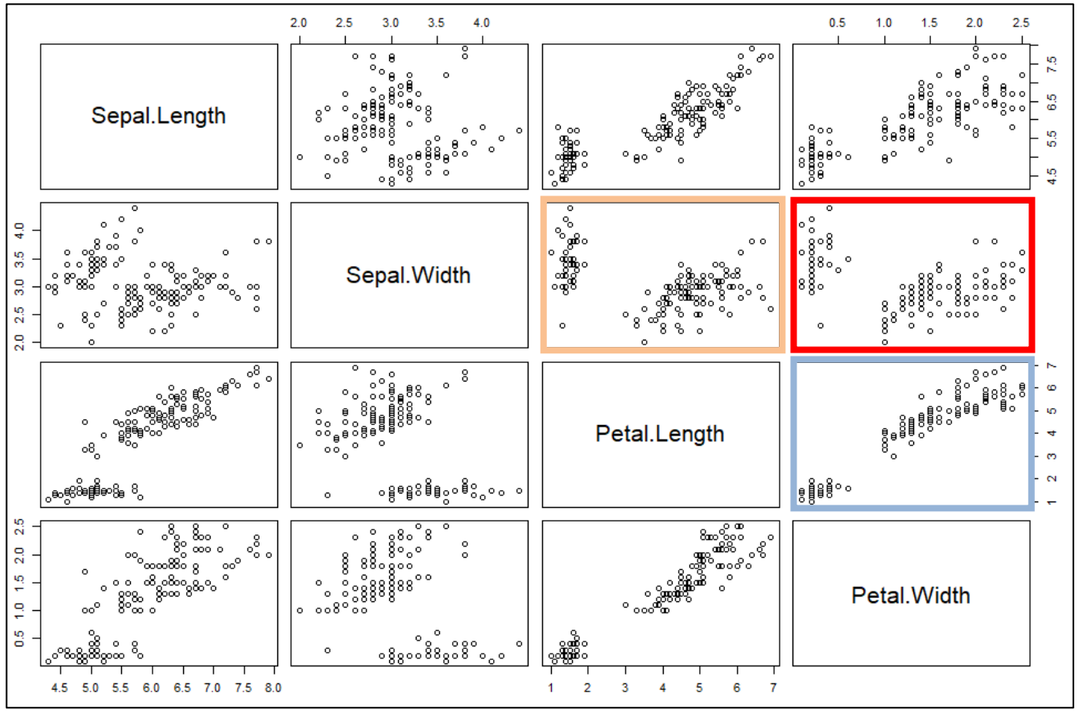
```
cor(data[,1:4], method="kendall")
```

- **예시2**

```
#psych 패키지 사용 – 통계 관련 유용한 함수 다수 제공
install.packages("psych")
library(psych)

#컬럼 별 상관계수 및 컬럼 별 상관관계 가설검정 수치 함께 출력. cor 함수와 cor.test 함수와 동일
corr.test(data[,1:4],
        use = 'complete',
        method = 'kendall',
        adjust ='none')
```

02 상관계수 설명 - 켄달 상관계수 분석

```
> cor(data[,1:4], method="kendall")
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length   1.00000000 -0.07699679    0.7185159   0.6553086
Sepal.Width   -0.07699679  1.00000000   -0.1859944  -0.1571257
Petal.Length   0.71851593 -0.18599442    1.0000000   0.8068907
Petal.Width    0.65530856 -0.15712566    0.8068907   1.0000000
```



02 상관계수 설명 – 스피어만 상관계수 예시

- **예시1**

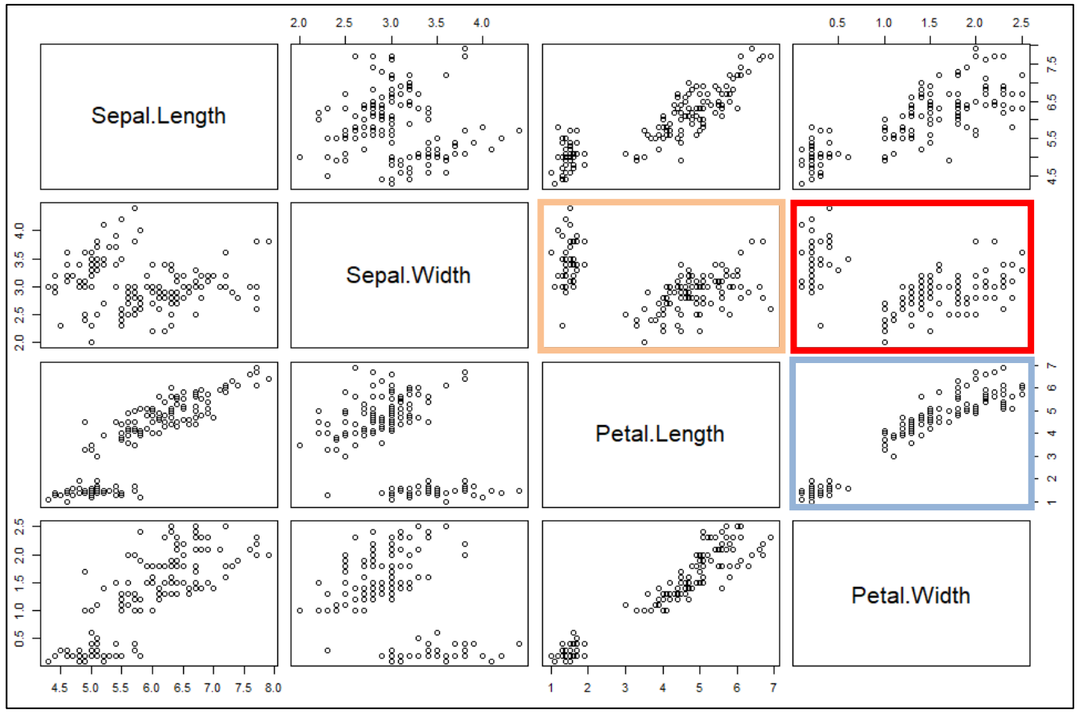
```
cor(data[,1:4], method="spearman")
```

- **예시2**

```
#psych 패키지 사용 – 통계 관련 유용한 함수 다수 제공
install.packages("psych")
library(psych)

#컬럼 별 상관계수 및 컬럼 별 상관관계 가설검정 수치 함께 출력. cor 함수와 cor.test 함수와 동일
corr.test(data[,1:4],
        use = 'complete',
        method = 'spearman',
        adjust ='none')
```

02 상관계수 설명 - 스피어만 상관계수 분석

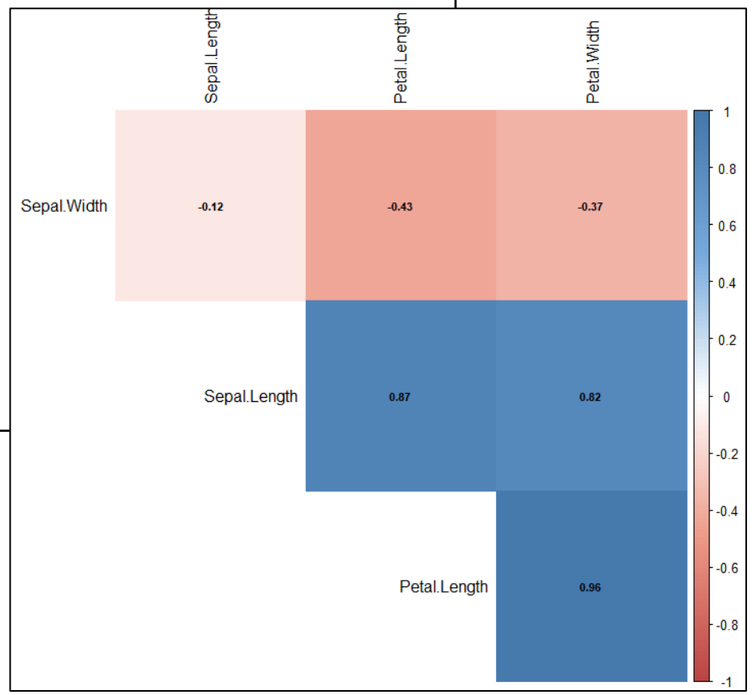
```
> cor(data[,1:4], method="spearman")
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000  -0.1667777    0.8818981    0.8342888
Sepal.Width    -0.1667777   1.0000000   -0.3096351   -0.2890317
Petal.Length    0.8818981  -0.3096351    1.0000000    0.9376668
Petal.Width     0.8342888  -0.2890317    0.9376668    1.0000000
```



03 상관계수 시각화 – corrplot 패키지

- **상관계수 시각화 기능 제공**

```
install.packages("corrplot")
library(corrplot)

col <- colorRampPalette(c("#BB4444", "#EE9988", "#FFFFFF", "#77AADD", "#4477AA"))
M = cor(iris[,1:4])
corrplot(M,
        method = "color",
        col = col(150),
        type = "upper",
        order = "hclust",
        number.cex = .7,
        addCoef.col = "black",
        tl.col = "black",
        tl.srt = 90,
        sig.level = 0.01,
        insig = "blank",
        diag = FALSE)
```



39차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

40차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭

학습 목표

- 와인의 화학 요소와 품질 간의 상관관계를 분석하고 시각화 할 수 있다.
- 와인 평균 알코올 도수에 대한 가설검정을 할 수 있다.

학습 목차

- 와인의 화학 요소와 품질 간의 상관관계 분석 및 시각화
- 와인 평균 알코올 도수에 대한 가설검정

01 와인 품질 데이터셋 – 데이터셋 설명

- **다운로드 경로: https://archive.ics.uci.edu/ml/datasets/Wine+Quality**

- **속성 명세**

1. fixed acidity: 고정된 산미
2. volatile acidity: 휘발성 산미
3. citric acid : 구연산
4. residual sugar: 잔여 당분
5. chlorides: 염화물
6. free sulfur dioxide: 유리 이산화황
7. total sulfur dioxide: 총 이산화황
8. density: 밀도
9. ph: 농도
10. sulphates: 황산염
11. alcohol: 알코올
12. quality: 등급 (0~10 정수)

02 상관관계 분석 실습 - 데이터셋 다운/로딩

Q1-1. 주어진 다운로드 경로에서 ‘winequality-red.csv’, ‘winequality-white.csv’ 파일을 다운로드 받고 로딩하세요.

1. 두 파일의 형식은 같으며, 따라서 다운로드 코드의 중복이 발생합니다.
   다운로드 로딩 과정을 함수화하여 사용하세요.
2. 로딩 한 데이터 셋은 각각 ‘red_wine’, ‘white_wine’ 변수에 저장하세요.

```
myload_data <- function(path){                          # 로딩 과정 함수화
  data <- read.table(path, header=T, sep=';')

  for (col in names(data)){                             # 모든 값들이 문자열로 덮여 있으므로 숫자화
    data[,col] <- as.numeric(data[, col])

  }
  return(data)
}

red_wine <- myload_data('winequality-red.csv')
white_wine <- myload_data('winequality-white.csv')
```

02 상관관계 분석 실습 - 품질과 유관한 속성은?

Q1-2. 데이터의 속성별 상관관계를 분석하려 합니다. 데이터셋의 상관관계표를 시각화 하는 코드를 작성하세요.

1. corrplot::corrplot() 를 사용하세요.
   - 39차시 자료 참조
2. 상관관계표 시각화 과정도 함수화 하세요.
   - 함수 매개변수: data, 상관계수(pearson, kendall, spearman)
3. 생성한 함수를 사용하여 데이터셋 별 pearson, spearman 상관계수 사용한 상관관계표를 시각화하세요.

02 상관관계 분석 실습 - 품질과 유관한 속성은?

Q1-2. 데이터의 속성별 상관관계를 분석하려 합니다. 데이터셋의 상관관계표를 시각화 하는 코드를 작성하세요.

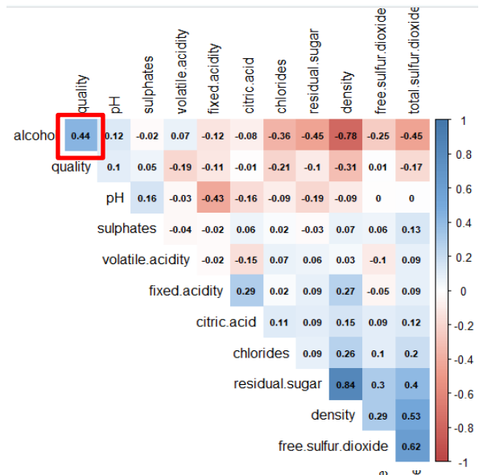
```
library(corrplot)
mydraw_correlation <- function(data, cor_method="pearson"){
  M <- cor(data, method=cor_method)
  names(M) <- names(data)  #데이터 상관관계값 추출하면 컬럼 이름명 사라짐. 재할당
  col <- colorRampPalette(c("#BB4444", "#EE9988", "#FFFFFF", "#77AADD", "#4477AA"))
  corrplot(M,
           method = "color",
           col = col(150),
           type = "upper",
           order = "hclust",
           number.cex = .7,
           addCoef.col = "black",
           tl.col = "black",
           tl.srt = 90,
           sig.level = 0.01,
           insig = "blank",
           diag = FALSE)
}
mydraw_correlation(white_wine) #피어슨 상관관계 비교
mydraw_correlation(red_wine)
mydraw_correlation(white_wine, "spearman") #피어스만 상관관계 비교
mydraw_correlation(red_wine, "spearman")
```

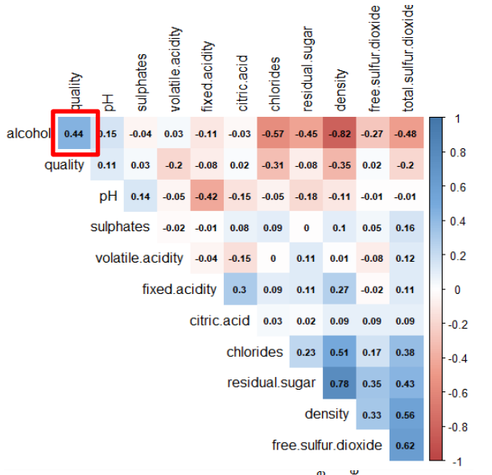
02 상관관계 분석 실습 - 품질과 유관한 속성은?

Pearson

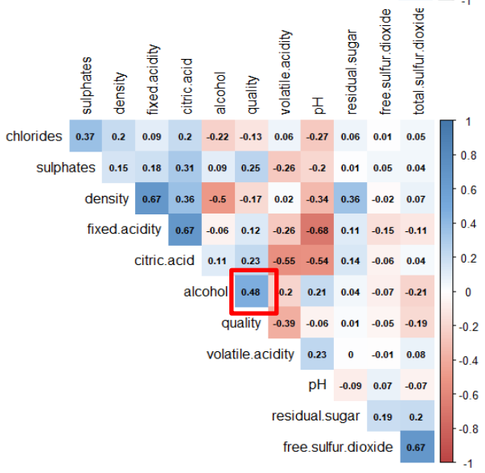
Spearman

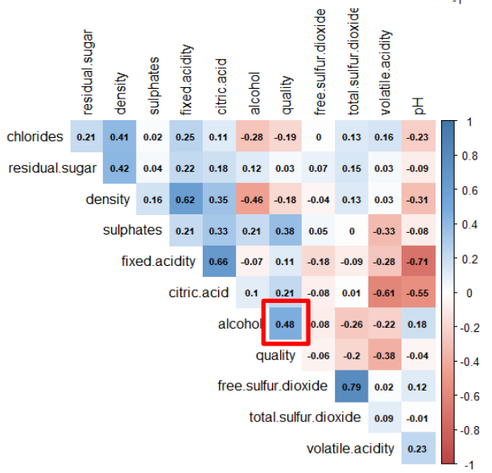
White_wine





Red_wine





02 상관관계 분석 실습 - 변수 별 산점도 그래프

Q1-3. 이번엔 시각적으로 변수간의 상관관계를 확인하려 합니다. 변수 별 산점도 그래프를 출력하세요.

- 데이터셋 별로 각각 출력하세요

```
#잘 안보인다.
pairs(red_wine)
pairs(white_wine)
```

02 상관관계 분석 실습 - 상자그림 그래프

Q1-5. 알코올 도수(alcohol)별 품질(quality)의 상자그림 그래프를 출력하세요.

- ggplot2::ggplot 함수 사용 및 그래픽 문법을 활용하세요

```
#alcohol값과 quality의 값이 양의 관계에 있다는 것이 직관적으로 보인다.
ggplot(red_wine, aes(y=alcohol)) +
  geom_boxplot(aes(col=factor(quality)))
ggplot(white_wine, aes(y=alcohol)) +
  geom_boxplot(aes(col=factor(quality)))
```

02 상관관계 분석 실습 – 품질과 유관한 속성은?

- 실습에 사용하는 데이터는 포르투갈의 'Vinho Verde'에 속하는 와인들을 분석한 결과입니다. Wikipedia에 따르면, 해당 와인들은 평균 알코올 도수가 8.5~14%라고 합니다. 11.5%가 평균 알코올 도수라 생각할 때, 아래 물음에 답하세요.

Q2. 데이터셋 와인들의 알코올 도수 평균 값이 11.25%이라는 가설이 타당한지 가설검정을 통해 밝히세요.

- red, white 와인 데이터셋 각각에 대하여 검정하세요.

02 상관관계 분석 실습 – 정규성 검증

Q2-1. 와인 데이터의 알코올 값이 정규분포를 따르는지 검증하세요.

- shapiro.test() 함수 사용
  - p-value가 유의수준(0.05)보다 크면 정규분포를 따른다는 뜻

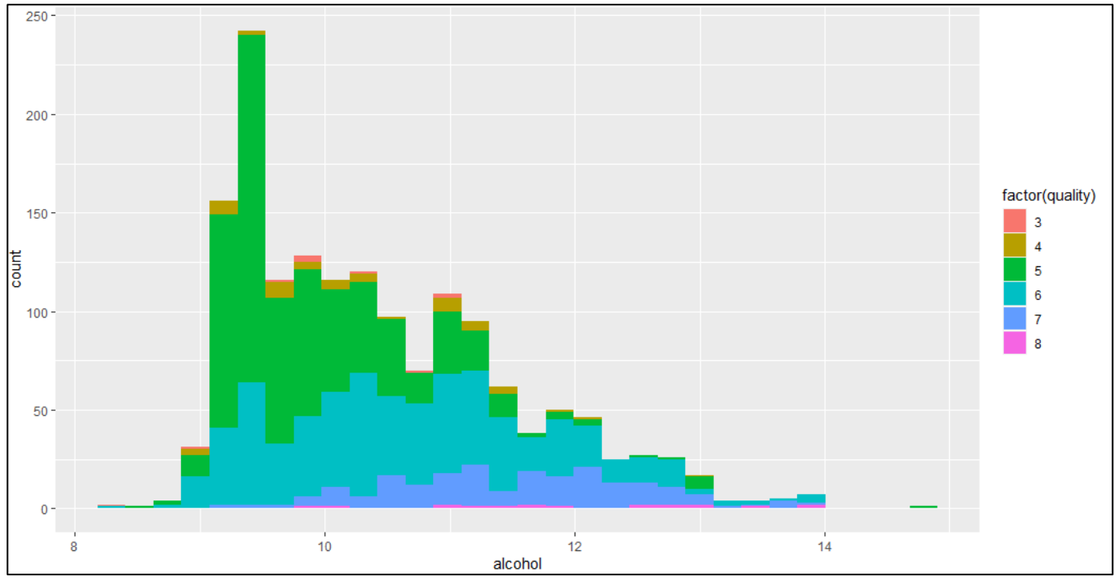
```
shapiro.test(red_wine$alcohol)        # p-value: 2.2/10^16. 귀무가설 기각. 정규분포를 따르지 않는다.
shapiro.test(white_wine$alcohol)      # p-value: 2.2/10^16. 귀무가설 기각. 정규분포를 따르지 않는다.
```

02 상관관계 분석 실습 - 알코올 도수 분포 확인

Q2-2. 알코올 도수에 대한 히스토그램을 그리세요.

- 품질 값 별로 카운팅을 따로 하는 히스토그램을 그리세요.

```
ggplot(red_wine) +
  geom_histogram(aes(x=alcohol, fill=factor(quality)))
```



→ 저품질 와인 (3~6)들의 분포가 낮은 알코올 도수에 지나치게 치우쳐 있음

02 상관관계 분석 실습 - 정규성 검증2

- 고품질 와인으로만 데이터 구성하여 검정해보면 어떨까?

Q2-3. 품질이 7이상인 데이터만 사용하여 정규성 검증을 다시 검증하세요.

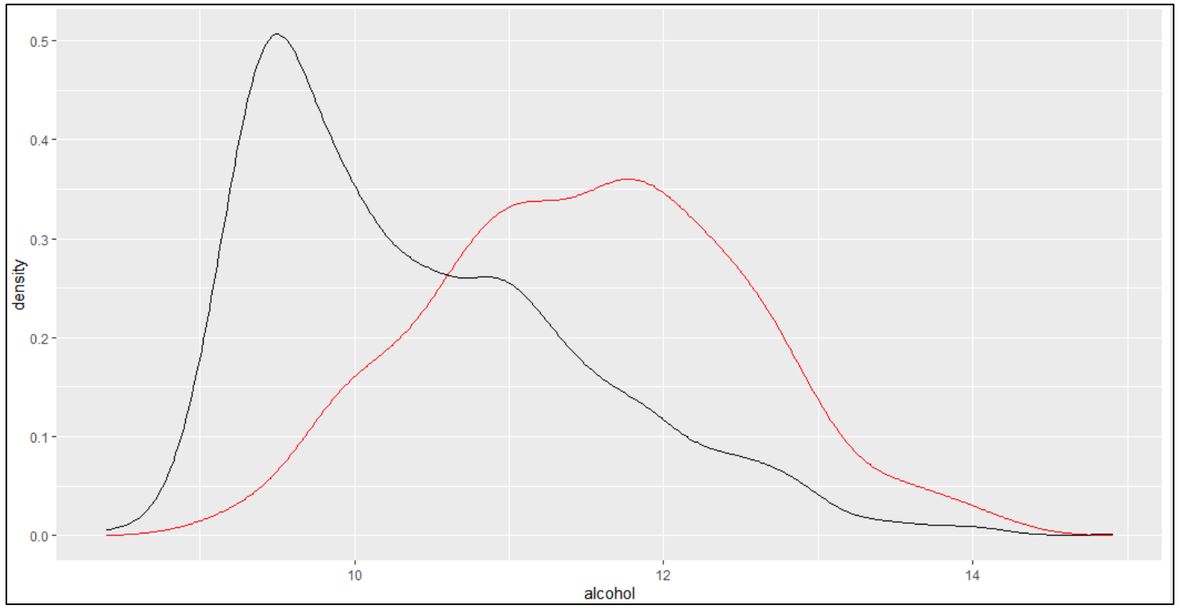
```
library(dplyr)
high_quality_white <- white_wine %>% filter(quality > 6)
shapiro.test(high_quality_white$alcohol)          #p-value: 5.908/10^14. 귀무가설 기각

high_quality_red <- red_wine %>% filter(quality > 6)
shapiro.test(high_quality_red$alcohol)            #p-value:0.2322. 유의확률 0.05에 대하여 귀무가설 채택
```

→ 고품질(7~) 레드와인에 대한 가설검정은 **의미가 있다.**

02 상관관계 분석 실습 - 정규성 검증2

Q2-4. 고품질 레드 와인과 전체 레드와인의 알코올 도수 분포를 시각화 하세요.

```
red_wine %>% filter(quality > 6) %>% ggplot() +
  geom_density(aes(x=alcohol), col='red') +     #전체 레드와인 알코올 분포
  geom_density(data=red_wine, aes(x=alcohol))   #고품질 레드와인 알코올 분포
```



02 상관관계 분석 실습 - 가설 검증

Q2-5. '고품질 레드 와인의 평균 알코올 도수는 11.5%이다' 라는 가설이 타당한지 검증하세요.

```
# 모집단의 표준편차값을 알 수 없기에 t검정 사용
t.test(x=high_quality_red$alcohol, alternative = "two.sided", mu=11.5, conf.level = 0.95)
```

```
data:  high_quality_red$alcohol
t = 0.26637, df = 216, p-value = 0.7902
alternative hypothesis: true mean is not equal to 11.5
95 percent confidence interval:
 11.3845 11.6516
sample estimates:
mean of x
 11.51805
```

- 분석
  - p-value가 유의수준 0.05보다 크므로 귀무가설 채택
  - 즉, 고품질 레드 와인의 평균 알코올 도수는 11.5%이다 라는 가설은 타당하다.

40차시

비전공자를 위한

데이터 시각화 입문(R 프로그래밍)

신 원 섭