R tidyverse 마스터 클래스

1강 - `tidyverse` 패키지와 친해지기

슬기로운통계생활

Issac Lee

코스 훑어보기.

Big picture가 중요합니다.

R tidyverse 마스터 클래스

수강 대상

- 슬기로운 통계생활 기초 R 강의 수강생
  - 무료입니다. 듣고 오세요!

강의 내용

- tidyverse를 사용한 **데이터 전처리**와 **시각화** 기술 마스터
- 수강레벨 - **중급**

왜 슬통 tidyverse 강의인가?

`tidyverse` 전문 클래스

- 시장의 모든 강의들은 기본 R 코드 강의로 시작함.
- 오롯이 `tidyverse`만을 위한 전문 강의

최신 패키지 반영

- 2021년까지 업데이트 된 최신 함수 모두 반영
    - 프로그래밍 언어는 변합니다.
- `tidyverse`와 궁합이 좋은 여러 패키지들을 자연스럽게 공부
- `tidyverse` 입문 강의가 아닌 **마스터** 강의

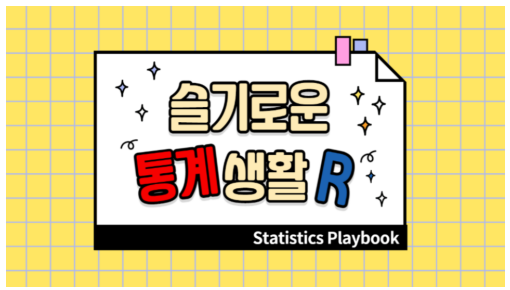
어디서 들을 수 있나요?

매주 수요일 밤 10시 Youtube 라이브!

- 전체 강의 라이브 공개

다시보기

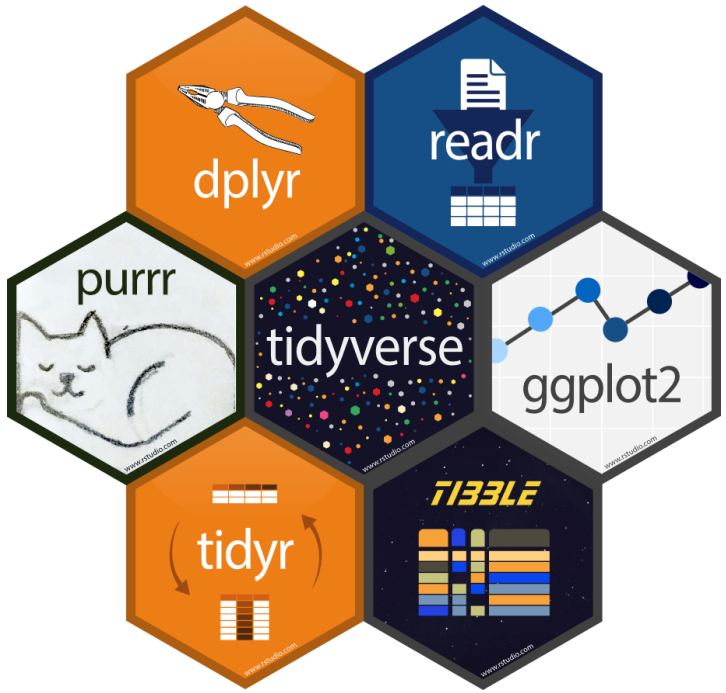
- Udemy 강의 (평생 소장)
  - 가격: $ 59.99
- 슬통갱
  - 멤버십 전용 영상
  - 강의 홈페이지 제공 예정



데이터 분석 도구로서의 R

tidyverse에 대하여

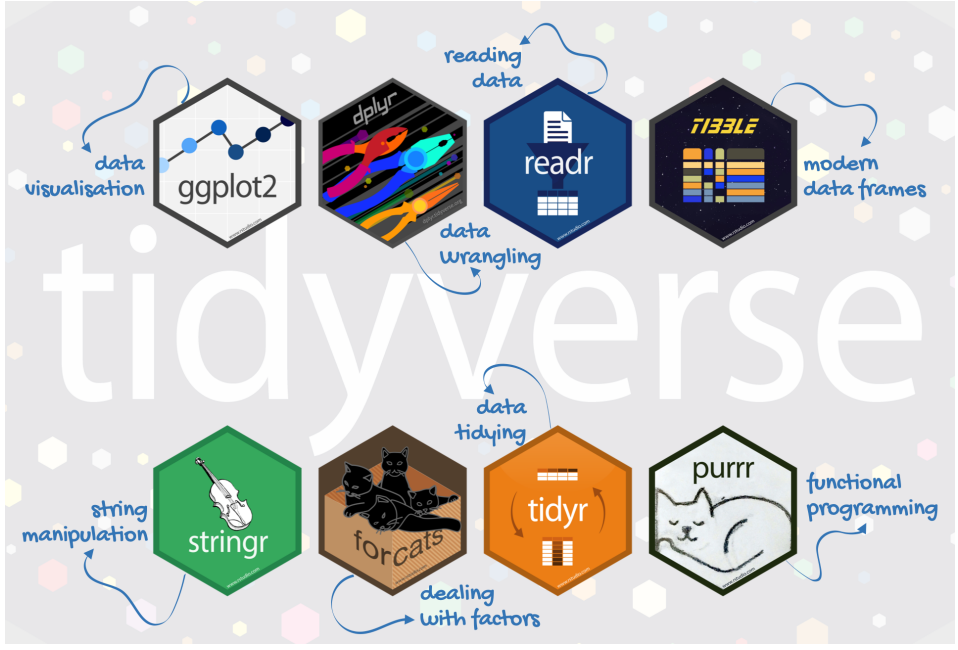
- 여러 데이터 분석에 유용한 패키지들의 모음집
- 공식적인 `tidyverse`의 시작은 2016년
  - 이전 부터 각각 구성원들이 따로따로 사용되고 있었음.
  - `ggplot2` 패키지는 2005년에 만들어짐.



핵심 구성 요소

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

- `ggplot2`: data visualization
- `dplyr`: data wrangling
- `readr`: reading data
- `tibble`: modern data frames
- `stringr`: string manipulation
- `forcats`: dealing with factors
- `tidyr`: data tidying
- `purrr`: functional programming



데이터 맛보기

서울시 생활이동 데이터

서울 열린데이터 광장

- 특정 시점에 서울 안에서 이동, 서울 외부에서 서울로 오고 간 이동
- 통근·통학·쇼핑·여가 등 서울시의 행정수요를 유발하는 모든 이동을 의미
- 2021년 7월 (8시) 이동 데이터
- 자치구 코드 정보
- 학습 데이터 다운로드

데이터 불러오기

```
library(tidyverse)
library(magrittr)
moving_data <- read_csv("./data/seoul_moving_202107_09_hr.csv")
reference_data <- readxl::read_excel("./data/reference.xlsx")
```

데이터 파일 불러오기

- `.csv` 파일의 경우 `read_csv()` 함수를 사용
- `.xlsx` 파일의 경우 `read_excel()` 함수를 사용
  - 차이점은?
  - `::` 의 의미 알고가기!

데이터와 친해지기

기초 탐색

기초 R 강의에서 배운 함수들을 사용해서 데이터 구조를 살펴보자.

- `dim()`
- `head()`
- `tail()`

tidyverse

- `glimpse()`

데이터 변수 이름 바꾸기

변수 이름 설정 - 한글 코딩은 싫어요!

```
# reference_data
origin_name_reference <- names(reference_da
reference_data <- janitor::clean_names(refe
reference_data %>% names()
```

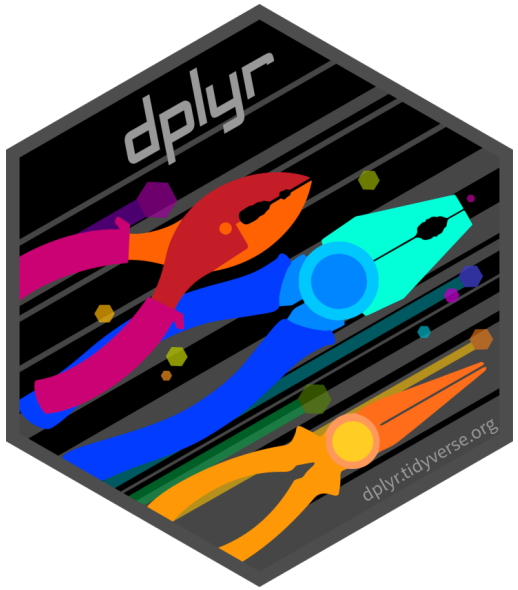
```
## [1] "sido"      "sigungu"   "name"
```

```
# moving_data
origin_name_moving <- names(moving_data)
moving_data <- janitor::clean_names(moving_
moving_data %>% names()
```

```
##  [1] "daesang_yeon_wol"             "yoil
##  [3] "dochagsigan"                  "chul
##  [5] "dochag_sigungu_kodeu"         "seor
##  [7] "nai"                          "idor
##  [9] "pyeong_gyun_idong_sigan_bun" "idor
```

`dplyr` 패키지

기본 동사 학습



`dplyr` 기본 동사 학습하기

Single table 동사들

행(row) 관련 동사들

- `distinct()`
- `filter()`
- `slice()`

열(column) 관련 동사들

- `select()`
- `rename()`
- `mutate()`
- `relocate()`

시도 단위는 몇 개인가?

중복 없는 표본들을 걸러줌.

- `distinct()`
  - `.keep_all=TRUE` 설정으로 딸려있는 데이터 보관하기

```
reference_data %>%
  distinct(sido) %>%
  count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1    17
```

```
reference_data %>%
  distinct(sido, .keep_all = TRU
  dim()
```

```
## [1] 17  4
```

원하는 행들을 걸러(filter)내는 방법

사용 가능 함수들

- 연산자들
  - `==, >, >=, &, |, !`
- 유용한 함수들
  - `is.na()`
  - `between()`, `near()`

```
moving_data %>%
  filter(yoil == "일" &
         seongbyeol == "F")
```

```
moving_data %>%
  filter(chulbal_sigungu_kodeu =
         chulbal_sigungu_kodeu =
```

```
moving_data %>%
  filter(between(nai, 10, 40))
```

원하는 행들을 잘라내는 (slice) 방법

기본 함수인 `head()`의 확장버전

- 기본적으로 인덱싱을 제공
  - 맨 마지막 인덱스: `n()`
- 유용한 함수들
  - `slice_min()`, `slice_max()`, `slice_sample()`, `slice_head()`, `slice_tail()`

```
moving_data %>%
  slice(15:20)

moving_data %>%
  slice_tail(n = 6)

moving_data %>%
  mutate(nai = as.numeric(nai))
  distinct(nai) %>%
  slice_max(nai, n = 5)
```

내맘대로 행을 정렬(arrange)하는 방법

```
df %>%
  arrange(col1, desc(col2))
```

- 정렬기준 우선 순위 순서대로 설정
- 내림차순 `desc()`

```
moving_data %>%
  select(dochagsigan,
         pyeong_gyun_idong_sigar
  arrange(dochagsigan,
          desc(pyeong_gyun_idong
```

```
moving_data %>%
  mutate(yoil = as_factor(yoil)
    lvls_revalue(c("월", "화", "
           "목", "금", "토", "일"
  arrange(yoil)
```

원하는 열을 선택 (select) 하는 방법

사용할 수 있는 옵션들

- 사용가능 연산자들
  - `:`, `!`, `&`, `|`, `c()` 사용 가능
- 편리한 함수들
  - `everything()`, `last_col()`
  - `starts_with()`, `ends_with()`, `contains()`

```
reference_data %>%
  select(sido, full_name) %>%
  head()
```

```
## # A tibble: 6 x 2
##    sido full_name
##   <dbl> <chr>
## 1 11000 서울특별시 종로구
## 2 11000 서울특별시 중구
## 3 11000 서울특별시 용산구
## 4 11000 서울특별시 성동구
## 5 11000 서울특별시 광진구
## 6 11000 서울특별시 동대문구
```

나만의 열을 생성 (mutate) 하는 방법

- new = old 문법

```
mutate(new = old_1 + old_2)
```

- 유용한 옵션

`.keep = c("all", "used", "unused", "none")`

- 궁합 좋은 함수들 (심화공부)
  - `case_when()`, `na_if()`, `coalesce()`, `if_else()`

- 년도 정보만 빼내오기

```
moving_data %<>%
  mutate(year = substr(daesang_y
                       as.integer()) %>
  select(year, everything()) # y
```

- 분(min)을 시간(hr)으로 변환

```
moving_data %<>%
  mutate(idong_sigan_hr =
    pyeong_gyun_idong_sigan_bun
```

시도 정보만 빼내오기

```
reference_data %>%
  mutate(sido_name =
           str_split_fixed(full_name,
                           pattern = " ", 2)[ ,1]) %>%
  select(sido_name) %>%
  distinct()
```

```
## # A tibble: 17 x 1
##    sido_name
##    <chr>
##  1 서울특별시
##  2 부산광역시
##  3 대구광역시
##  4 인천광역시
##  5 광주광역시
##  6 대전광역시
```

열이름 다시 정하기 (rename)

```
df %>% rename(new = old)
df %>% rename_with(function)
```

- `A`을 `a_new`로 바꾸기

```
df %>% rename(a_new = A)
```

- 모든 열이름 대문자로 바꾸기

```
df %>% rename_with(toupper)
```

실습하기

```
moving_data %<>%
  rename(avg_time_min = pyeong_g
         ppl_sum = idong_ingu_ha
```

열의 위치를 재배치 (relocate) 하는 방법

- 열의 위치를 앞으로 땅김

```
df %>% relocate(col)
df %>% select(col, everything())
```

- 특정 위치로 옮길때
  - `.before`, `.after` 옵션
  - `last_col()` 함수
- 변수 타입으로 정렬가능
  - `where(is.character)` 사용

실습하기

- 이동 인구 합 앞으로

```
moving_data %>%
  relocate(idong_ingu_hab)
```

- 문자열 타입 앞으로

```
moving_data %>%
  relocate(where(is.character))
```

다음시간

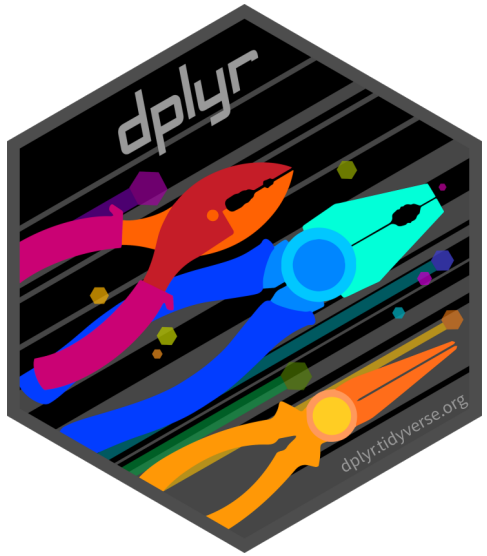
`dplyr` 고급 동사들

Two-table verbs

Make summary

Column-wise operations

Row-wise operations



같이 보면 좋은 책 추천

[1] R for Data Science

- 웹 상에 무료 공개된 책입니다.
- 위 교재의 한글 번역본 R 을 활용한 데이터과학도 있습니다.
- 도서 제목 클릭하셔서 구매하시면 저의 사리사욕을 충당하는데 도움이 됩니다.