



5주차: 데이터 가공

ChulSoo Park

School of Computer Engineering & Information Technology

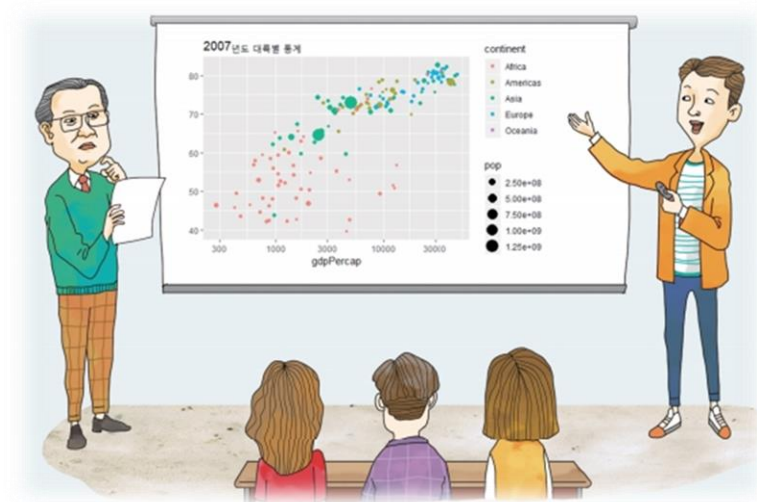
Korea National University of Transportation



05

CHAPTER

데이터 가공



CONTENTS

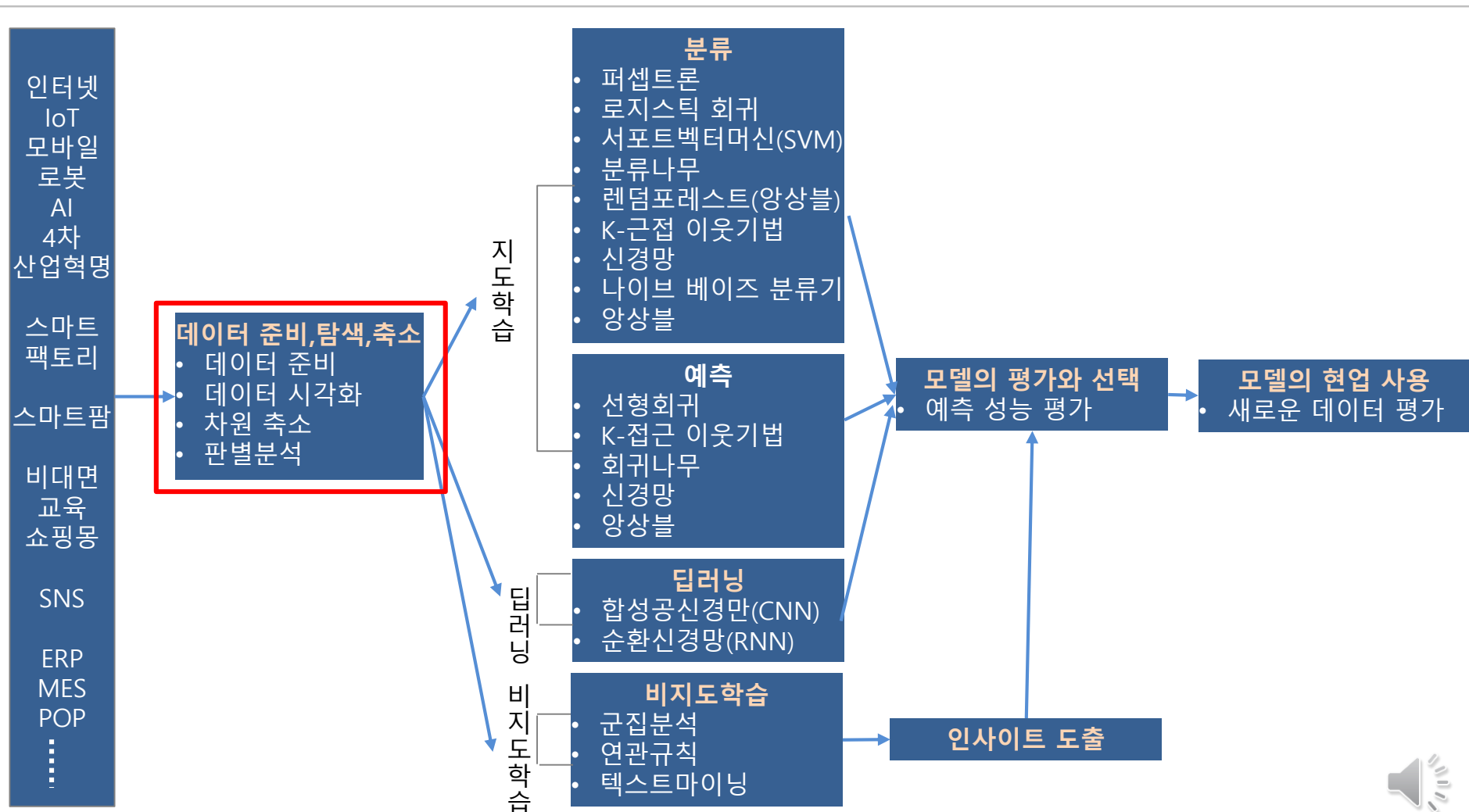
- 5.1 데이터 가공이란?
- 5.2 베이스 R을 이용한 데이터 가공
- 5.3 dplyr 라이브러리를 이용한 데이터 가공
- 5.4 대량의 데이터 가공
- 5.5 데이터 가공 사례 학습
 - 요약



- 잘 정제되고 다듬어진 데이터는 큰 가치가 있지만, 정리되지 않은 데이터는 의미 추출이 어려울 뿐 아니라 잘못된 결론에 이르게 할 수도 있음



■ 데이터 사이언스 전체 Process에서 이번주 교육 위치



5.3 dplyr 라이브러리를 이용한 데이터 가공

- base R의 데이터 가공 기법이 인덱스 기반의 데이터 접근에 기초하고 있다면, dplyr 라이브러리는 `filter` 혹은 `select` 같은 입출력 관계의 함수로 구현함으로써 사용자들이 보다 직관적으로 활용 가능
- 데이터 가공은 가공에 특화된 라이브러리를 사용하는 것이 더 효율적
- 탐색적 데이터 분석과정에서 시각화와 데이터 가공은 매우 긴밀하게 연결되어 시각화를 위한 효율적인 가공 기법도 필요



5.3 dplyr 라이브러리를 이용한 데이터 가공

■ 개요

- dplyr은 R에서 데이터 프레임을 조작하는 패키지
- 주로 데이터 **전처리**를 위해 많이 사용함
- **%>%** 파이프 연산자를 이용하여 체인 형식의 코드 사용가능

■ dplyr의 장단점

• 장점

- 직관적이고 이해하기 쉬운 코드
- 불필요한 객체 생성 없이 사용
- 빠른 처리속도 및 메모리 소모가 적음

• 단점

- 사용 가능한 함수 제한적
- 코드 작성 방식에 있어 호불호 존재



5.3 dplyr 라이브러리를 이용한 데이터 가공

■ gapminder 라이브러리

- 세계 각국의 기대 수명 1인당 국내총생산, 인구 데이터 등을 집계한 gapminder 데이터 셋의 일부를 담고 있음
- 이 데이터는 R을 배우고 통계학을 연습하는데 매우 유용한 기초 자료이고 여러 유형의 데이터가 데이터 프레임 형식으로 저장되어 있어 데이터 과학을 학습하는 우리들에게 매우 좋은 자료이다.




표 5-1 gapminder 데이터 프레임의 구성 항목

이름(변수명)	변수형	내용
country	142개 레벨의 범주형	국가명
continent	5개 레벨의 범주형	국가가 속한 대륙
year	int	1952~2007 관측 연도(5년 단위)
lifeExp	num	기대 수명(평균 수명)
pop	int	인구
gdpPercap	num	1인당 국내총생산(물가 상승 반영)



5.3 dplyr 라이브러리를 이용한 데이터 가공

- 기존에 데이터 가공을 data 추출 방법 위해 사용 했던 방법

```
Console C:/RSources/     
> gapminder[gapminder$country=="Korea, Rep."&gapminder$year>1  
970, c("lifeExp","pop")]  
# A tibble: 8 x 2  
  lifeExp      pop  
  <dbl>    <int>  
1    62.6 33505000  
2    64.8 36436000  
3    67.1 39326000  
4    69.8 41622000  
5    72.2 43805450  
6    74.6 46173816  
7    77.0 47969150  
8    78.6 49044790  
> apply(gapminder[gapminder$country=="Korea, Rep.",c("lifeEx  
p","pop","gdpPercap")],2,mean)  
  lifeExp      pop      gdpPercap  
65.001 36499386.333      8217.318  
> |
```



5.3 dplyr 라이브러리를 이용한 데이터 가공

■ 샘플과 속성 추출

- select 함수 이용 : library를 사용해 dplyr 패키지를 부착(메모리에 load해야함)
- 열을 지정할 때 " " 없이 열 이름을 그대로 사용할 수 있어 편리

```
select {dplyr}
```

R Documentation

Subset columns using their names and types

Description

Select (and optionally rename) variables in a data frame, using a concise mini-language that makes it easy to refer to variables by name. `select()` selects all columns from `a` on the left to `f` on the right. You can use predicate functions like [is.numeric](#) to select variables by type.

Usage

```
select(.data, ...)
```

Arguments

- `.data` A data frame, data frame extension (e.g. a tibble), or a lazy data frame (e.g. from dbplyr or dtplyr). See *Methods*, below, for more details.
- `...` [<tidy-select>](#) One or more unquoted expressions separated by commas. Variable names can be used as if they were positions in the data frame, so expressions like `x:y` can be used to select a range of variables.

5.3 dplyr 라이브러리를 이용한 데이터 가공

■ 샘플과 속성 추출

- select 함수 이용 : library를 사용해 dplyr 패키지를 부착(메모리에 load해야함)
- 열을 지정할 때 \$기호나 " " 없이 열 이름을 그대로 사용할 수 있어 편리

```
Console C:/RSources/
> select(gapmaider, country, year, lifeExp)
Error in select(gapmaider, country, year, lifeExp) :
  함수 "select"를 찾을 수 없습니다
> library(dplyr)

다음의 패키지를 부착합니다: 'dplyr'

The following objects are masked from 'dplyr':
  filter, lag

The following objects are masked from 'base':
  intersect, setdiff, setequal, union

> |
```

```
Console C:/RSources/
> select(gapminder, country, year, lifeExp)
# A tibble: 1,704 x 3
  country      year lifeExp
  <fct>      <int>   <dbl>
1 Afghanistan 1952    28.8
2 Afghanistan 1957    30.3
3 Afghanistan 1962    32.0
4 Afghanistan 1967    34.0
5 Afghanistan 1972    36.1
6 Afghanistan 1977    38.4
7 Afghanistan 1982    39.9
8 Afghanistan 1987    40.8
9 Afghanistan 1992    41.7
10 Afghanistan 1997    41.8
# ... with 1,694 more rows
```




5.3 dplyr 라이브러리를 이용한 데이터 가공

■ 샘플과 속성 추출

- 특정 샘플(행)을 추출할 때는 `filter` 함수 사용
- 조건식 구성은 base R과 유사하나 함수 내에서 인덱싱을 위해 데이터 프레임의 이름을 매번 입력하지 않아도 되므로 명령어가 간결

```
Console C:/RSources/   
> filter(gapminder, country=="Korea, Rep.")
```

```
# A tibble: 12 x 6  
  country      continent  
  <fct>        <fct>  
1 Korea, Rep. Asia  
2 Korea, Rep. Asia  
3 Korea, Rep. Asia  
4 Korea, Rep. Asia  
5 Korea, Rep. Asia  
6 Korea, Rep. Asia  
7 Korea, Rep. Asia  
8 Korea, Rep. Asia  
9 Korea, Rep. Asia  
10 Korea, Rep. Asia  
11 Korea, Rep. Asia  
12 Korea, Rep. Asia  
> |
```

```
Console C:/RSources/   
> gapminder[gapminder$country=="Korea, Rep.", c("pop")]
```

```
# A tibble: 12 x 1  
  pop  
  <int>  
1 20947571  
2 22611552  
3 26420307  
4 30131000  
5 33505000  
6 36436000  
7 39326000  
8 41622000  
9 43805450  
10 46173816  
11 47969150  
12 49044790  
> |
```



5.3 dplyr 라이브러리를 이용한 데이터 가공

■ 행/열 단위의 연산

- group_by 함수를 이용하면 데이터 프레임에 포함된 범주형 속성을 활용해 전체 데이터를 그룹으로 분류 가능
- 보통 summarise 함수를 연이어 사용해 그룹별 통계 지표를 한번에 산출

```
Console C:/RSources/
> summarize(gapminder, pop_avg=mean(pop))
# A tibble: 1 x 1
  pop_avg
  <dbl>
1 29601212
> summarize(group_by(gapminder, continent), pop_avg=mean(pop))
# A tibble: 5 x 2
  continent pop_avg
* <fct>      <dbl>
1 Africa    9916003.
2 Americas  24504795.
3 Asia      77038722.
4 Europe    17169765.
5 Oceania   8874672.
> summarize(group_by(gapminder, continent, country), pop_avg=mean(pop))
`summarise()` has grouped output by 'continent'. You can override using the `.groups` argument.
# A tibble: 142 x 3
# Groups:   continent [5]
  continent country      pop_avg
  <fct>      <fct>      <dbl>
1 Africa    Algeria    19875406.
2 Africa    Angola      7309390.
3 Africa    Benin       4017497.
4 Africa    Botswana     971186.
5 Africa    Burkina Faso  7548677.
6 Africa    Burundi     4651608.
7 Africa    Cameroon    9816648.
8 Africa    Central African Republic 2560963
9 Africa    Chad        5329256.
10 Africa   Comoros     361684.
# ... with 132 more rows
```

5.3 dplyr 라이브러리를 이용한 데이터 가공

■ %>% 연산자를 이용한 연속 처리

- %>% 연산자를 사용해 일련의 가공 작업을 연결

Console C:/RSources/

```
> summarize(group_by(gapminder, continent, country), pop_avg=mean(pop))
```

`summarise()` has grouped output by 'continent'. You can override using the `.groups` argument.

A tibble: 142 x 3

Groups: continent [5]

	continent	country	pop_avg
	<fct>	<fct>	<dbl>
1	Africa	Algeria	19875406.
2	Africa	Angola	7309390.
3	Africa	Benin	4017497.
4	Africa	Botswana	971186.
5	Africa	Burkina Faso	7548677.
6	Africa	Burundi	4651608.
7	Africa	Cameroon	9816648.
8	Africa	Central African Republic	2560963.
9	Africa	Chad	5329256.
10	Africa	Comoros	361684.

... with 132 more rows

> |

Console C:/RSources/

```
> gapminder %>% group_by(continent, country) %>% summarise(pop_avg=mean(pop))
```

`summarise()` has grouped output by 'continent'. You can override using the `.groups` argument.

A tibble: 142 x 3

Groups: continent [5]

	continent	country	pop_avg
	<fct>	<fct>	<dbl>
1	Africa	Algeria	19875406.
2	Africa	Angola	7309390.
3	Africa	Benin	4017497.
4	Africa	Botswana	971186.
5	Africa	Burkina Faso	7548677.
6	Africa	Burundi	4651608.
7	Africa	Cameroon	9816648.
8	Africa	Central African Republic	2560963.
9	Africa	Chad	5329256.
10	Africa	Comoros	361684.

... with 132 more rows


> |



5.3 dplyr 라이브러리를 이용한 데이터 가공

- %>% 연산자를 이용한 연속 처리
 - %>% 연산자를 사용해 일련의 가공 작업을 연결

Console C:/RSources/



```
> tmp1=filter(gapminder, country=="Korea, Rep.")  
> tmp2=select(tmp1, country, year, lifeExp)  
> tmp3=apply(tmp2[, c("lifeExp")], 2, mean)  
> tmp3  
lifeExp  
65.001
```

```
> gapminder %>% filter(country=="Korea, Rep.") %>% select(c  
ountry, year, lifeExp) %>% summarise(lifeExp_avg=mean(lifeEx  
p))  
# A tibble: 1 x 1  
  lifeExp_avg  
    <dbl>  
1         65.0  
> |
```



5.4 데이터 가공의 실제 : 방대한 데이터 요약

■ Kaggle에 있는 avocado 데이터 활용

- <https://www.kaggle.com/neuromusic/avocado-prices>
- 캐글(Kaggle) 전 세계 과학자들이 특정 문제의 해결법을 놓고 경쟁을 벌이는 온라인 플랫폼이다. 데이터 과학자들이 기계 학습과 통계학을 기본으로 다양한 전략과 알고리즘을 구사하여 경쟁 모델을 통해 문제 해결 방법을 찾아 가도록 하였다.

The screenshot displays the Kaggle interface for the 'Avocado Prices' dataset. The left sidebar contains navigation links: Home, Compete, Data (selected), Notebooks, Communities, Courses, and More. The main content area features the dataset title 'Avocado Prices' with a subtitle 'Historical data on avocado prices and sales volume in multiple US markets'. Below this, the author 'Justin Kiggins' is listed with a note 'updated 3 years ago (Version 1)'. A horizontal bar shows the dataset's popularity with 'Data' selected, followed by 'Tasks (1)', 'Notebooks (283)', 'Discussion (15)', 'Activity', and 'Metadata'. A 'Download (629 KB)' button and a 'New Notebook' button are visible. The 'Usability' is 9.7, and the 'License' is 'Database: Open Database, Contents: © Original Authors'. The 'Tags' section includes 'food'. The 'Description' section is partially visible, showing a 'Context' heading and a paragraph about Millennials and Avocado Toast.



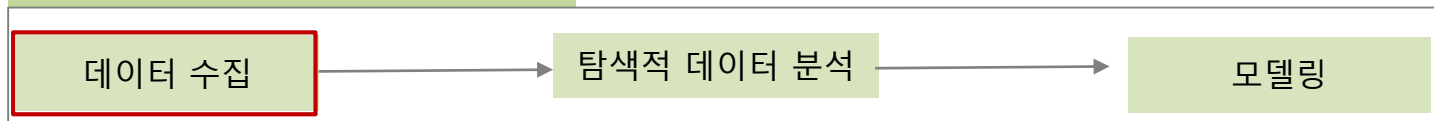
5.4 데이터 가공의 실제 : 방대한 데이터 요약

- 캐글(Kaggle)은 2010년 설립된 빅데이터 솔루션 대회 플랫폼 회사이다. 21 세기의 가장 섹시한 직업으로 데이터 사이언티스트가 꼽힐 만큼[하버드 비즈니스 리뷰] 빅 데이터(Big Data)가 사회 및 기업 환경에서 큰 화두로 떠오르면서 캐글의 규모도 같이 성장하게 되었고, 지난 2017년 3월, 구글은 캐글을 인수하기에 이르렀다.
- 기업 및 단체에서 경품과 상을 걸고 데이터와 해결 과제를 등록하면, 데이터 사이언티스트들이 이를 해결하기 위해 모델을 개발하고 경쟁하게 되는 시스템이다
- 사이트에 들어가 보면, 엄청나게 많은 도전 과제들이 기다리고 있는 것을 확인 할 수 있다. 누구나 이 중 관심있는 과제를 골라서 경쟁에 참여할 수 있다. 다만, 큰 상금이 걸린 과제들은 대부분 난이도가 매우 높다

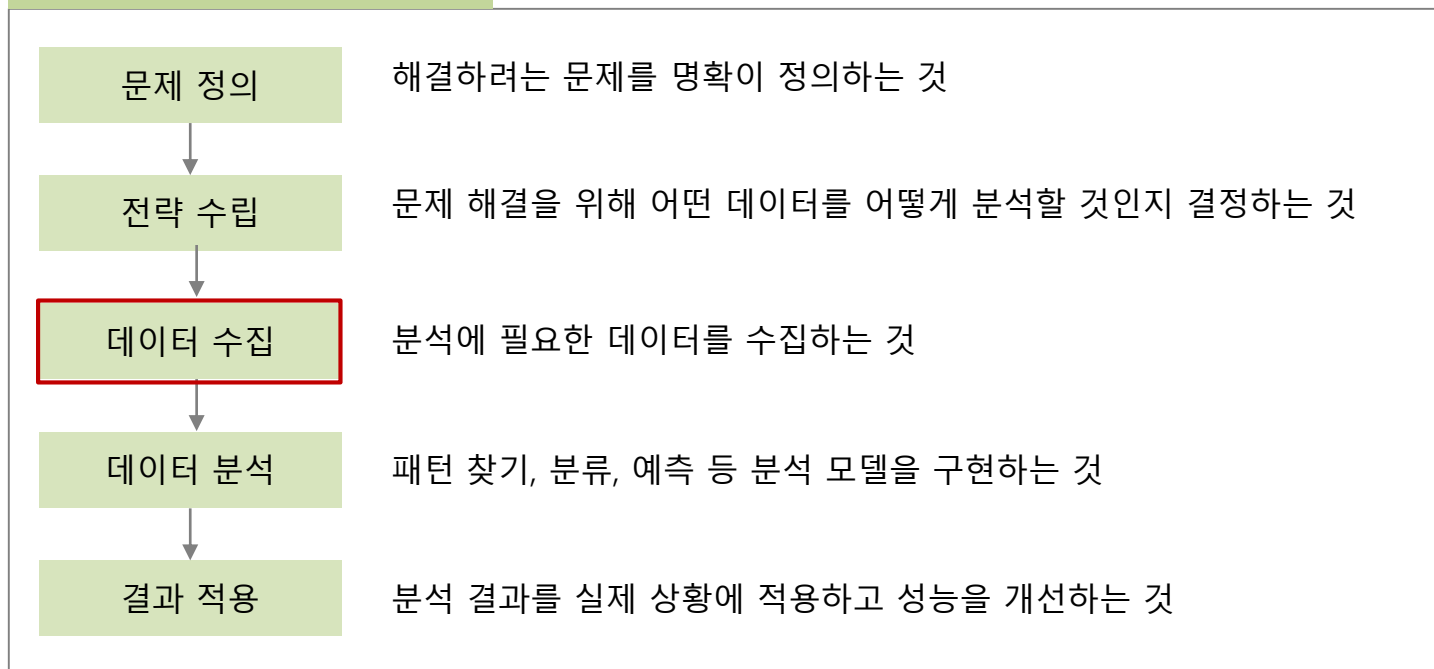


5.4 데이터 가공의 실제 : 방대한 데이터 요약

데이터 과학의 절차(그림 1- 5)

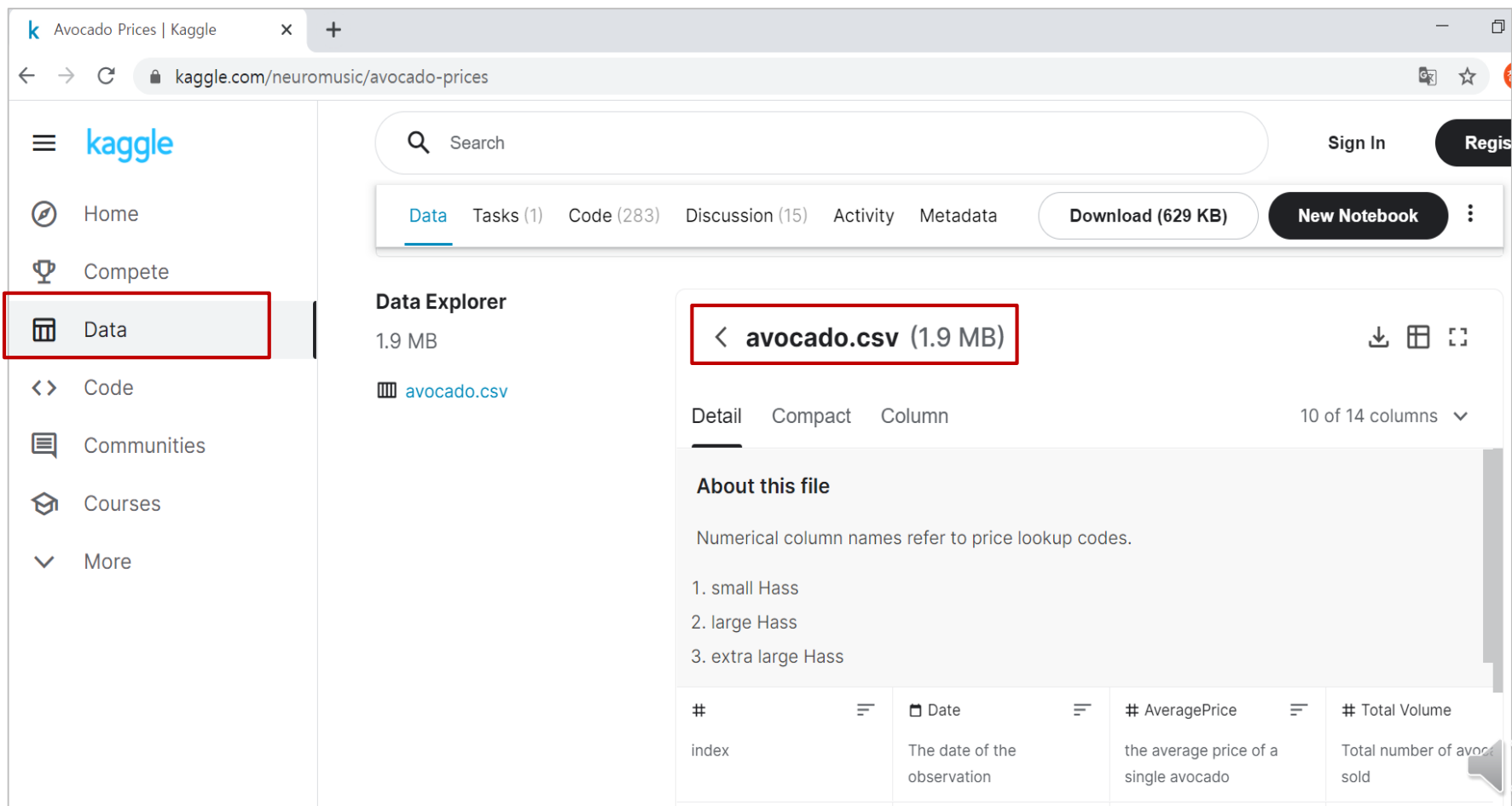


데이터 사이언스 프로세스



5.4 데이터 가공의 실제 : 방대한 데이터 요약

<https://www.kaggle.com/neuromusic/avocado-prices> 에서 avocado.csv 다운로드



The screenshot shows the Kaggle website interface for the 'avocado-prices' dataset. The left sidebar has the 'Data' tab selected. The main content area displays the dataset details, including the file name 'avocado.csv (1.9 MB)' and a table of columns.

Data Explorer
1.9 MB
avocado.csv

< avocado.csv (1.9 MB)

Download (629 KB) New Notebook

Search

Home Compete Data Code Communities Courses More

Sign In Register

Tasks (1) Code (283) Discussion (15) Activity Metadata

About this file

Numerical column names refer to price lookup codes.

1. small Hass
2. large Hass
3. extra large Hass

#	Date	AveragePrice	Total Volume
index	The date of the observation	the average price of a single avocado	Total number of avocados sold

5.4 데이터 가공의 실제 : 방대한 데이터 요약

■ 메모장으로 data 확인



avocado.csv - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
,Date,AveragePrice,Total Volume,4046,4225,4770,Total Bags,Small Bags,Large Bags,XLarge Bags,type,year,region
0,2015-12-27,1.33,64236.62,1036.74,54454.85,48.16,8696.87,8603.62,93.25,0.0,conventional,2015,Albany
1,2015-12-20,1.35,54876.98,674.28,44638.81,58.33,9505.56,9408.07,97.49,0.0,conventional,2015,Albany
2,2015-12-13,0.93,118220.22,794.7,109149.67,130.5,8145.35,8042.21,103.14,0.0,conventional,2015,Albany
3,2015-12-06,1.08,78992.15,1132.0,71976.41,72.58,5811.16,5677.4,133.76,0.0,conventional,2015,Albany
4,2015-11-29,1.28,51039.6,941.48,43838.39,75.78,6183.95,5986.26,197.69,0.0,conventional,2015,Albany
5,2015-11-22,1.26,55979.78,1184.27,48067.99,43.61,6683.91,6556.47,127.44,0.0,conventional,2015,Albany
6,2015-11-15,0.99,83453.76,1368.92,73672.72,93.26,8318.86,8196.81,122.05,0.0,conventional,2015,Albany
7,2015-11-08,0.98,109428.33,703.75,101815.36,80.0,6829.22,6266.85,562.37,0.0,conventional,2015,Albany
8,2015-11-01,1.02,99811.42,1022.15,87315.57,85.34,11388.36,11104.53,283.83,0.0,conventional,2015,Albany
9,2015-10-25,1.07,74338.76,842.4,64757.44,113.0,8625.92,8061.47,564.45,0.0,conventional,2015,Albany
10,2015-10-18,1.12,84843.44,924.86,75595.85,117.07,8205.66,7877.86,327.8,0.0,conventional,2015,Albany
11,2015-10-11,1.28,64489.17,1582.03,52677.92,105.32,10123.9,9866.27,257.63,0.0,conventional,2015,Albany
12,2015-10-04,1.31,61007.1,2268.32,49880.67,101.36,8756.75,8379.98,376.77,0.0,conventional,2015,Albany
13,2015-09-27,0.99,106803.39,1204.88,99409.21,154.84,6034.46,5888.87,145.59,0.0,conventional,2015,Albany
14,2015-09-20,1.33,69759.01,1028.03,59313.12,150.5,9267.36,8489.1,778.26,0.0,conventional,2015,Albany
15,2015-09-13,1.28,76111.27,985.73,65696.86,142.0,9286.68,8665.19,621.49,0.0,conventional,2015,Albany
16,2015-09-06,1.11,99172.96,879.45,90062.62,240.79,7990.1,7762.87,227.23,0.0,conventional,2015,Albany
17,2015-08-30,1.07,105693.84,689.01,94362.67,335.43,10306.73,10218.93,87.8,0.0,conventional,2015,Albany
18,2015-08-23,1.34,79992.09,733.16,67933.79,444.78,10880.36,10745.79,134.57,0.0,conventional,2015,Albany
```



5.4 데이터 가공의 실제 : 방대한 데이터 요약

- 다운로드 data R에서 read 및 데이터 프레임 구성 확인(str)

```
Console C:/RSources/
> avocado <- read.csv("c:/rdata/avocado.csv", header=TRUE, sep=",")
> str(avocado)
'data.frame': 18249 obs. of 14 variables:
 $ X      : int  0 1 2 3 4 5 6 7 8 9 ...
 $ Date   : chr  "2015-12-27" "2015-12-20" "2015-12-13" "2015-12-06" ...
 $ AveragePrice: num  1.33 1.35 0.93 1.08 1.28 1.26 0.99 0.98 1.02 1.07 ...
 $ Total.Volume: num  64237 54877 118220 78992 51040 ...
 $ X4046    : num  1037 674 795 1132 941 ...
 $ X4225    : num  54455 44639 109150 71976 43838 ...
 $ X4770    : num  48.2 58.3 130.5 72.6 75.8 ...
 $ Total.Bags : num  8697 9506 8145 5811 6184 ...
 $ Small.Bags : num  8604 9408 8042 5677 5986 ...
 $ Large.Bags : num  93.2 97.5 103.1 133.8 197.7 ...
 $ XLarge.Bags : num  0 0 0 0 0 0 0 0 0 0 ...
 $ type     : chr  "conventional" "conventional" "conventional" "conventional" ...
 $ year     : int  2015 2015 2015 2015 2015 2015 2015 2015 2015 2015 ...
 $ region   : chr  "Albany" "Albany" "Albany" "Albany" ...
> |
```

avocado.csv - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
,Date,AveragePrice>Total Volume,4046,4225,4770>Total Bags,Small Bags,Large Bags,XLarge Bags,type,year,region
0,2015-12-27,1.33,64236.62,1036.74,54454.85,48.16,8696.87,8603.62,93.25,0.0,conventional,2015,Albany
1,2015-12-20,1.35,54876.98,674.28,44638.81,58.33,9505.56,9408.07,97.49,0.0,conventional,2015,Albany
2,2015-12-13,0.93,118220.22,794.7,109149.67,130.5,8145.35,8042.21,103.14,0.0,conventional,2015,Albany
```



5.4 데이터 가공의 실제 : 방대한 데이터 요약

■ Avocado 데이터 구조 파악

```
Console C:/RSources/
> head(avocado, 6)
```

	X	Date	AveragePrice	Total.Volume	x4046
1	0	2015-12-27	1.33	64236.62	1036.74
2	1	2015-12-20	1.35	54876.98	674.28
3	2	2015-12-13	0.93	118220.22	794.70
4	3	2015-12-06	1.08	78992.15	1132.00
5	4	2015-11-29	1.28	51039.60	941.48
6	5	2015-11-22	1.26	55979.78	1184.27

	x4225	x4770	Total.Bags	Small.Bags	Large.Bags
1	54454.85	48.16	8696.87	8603.62	93.25
2	44638.81	58.33	9505.56	9408.07	97.49
3	109149.67	130.50	8145.35	8042.21	103.14
4	71976.41	72.58	5811.16	5677.40	133.76
5	43838.39	75.78	6183.95	5986.26	197.69
6	48067.99	43.61	6683.91	6556.47	127.44

	XLarge.Bags	type	year	region
1	0	conventional	2015	Albany
2	0	conventional	2015	Albany
3	0	conventional	2015	Albany
4	0	conventional	2015	Albany
5	0	conventional	2015	Albany
6	0	conventional	2015	Albany

- conventional
- organic



5.4 데이터 가공의 실제 : 방대한 데이터 요약

■ 그룹 단위 통계(1)

- 경향 도출을 위해 총 판매량과 평균 가격 속성을 지역에 따라 구분하여 각각 요약
- dplyr 라이브러리의 group_by와 summarize 함수 사용

```
Console C:/RSources/
> avg_data <- avocado %>% group_by(region) %>% summarise
(v_avg=mean(Total.Volume),p_avg=mean(AveragePrice))
> avg_data
# A tibble: 54 x 3
  region          v_avg p_avg
*   <chr>          <dbl> <dbl>
1 Albany         47538.  1.56
2 Atlanta        262145.  1.34
3 BaltimoreWashington 398562.  1.53
4 Boise          42643.  1.35
5 Boston         287793.  1.53
6 BuffaloRochester   67936.  1.52
7 California     3044324.  1.40
8 Charlotte       105194.  1.61
9 Chicago        395569.  1.56
10 CincinnatiDayton 131722.  1.21
# ... with 44 more rows
> |
```



5.4 데이터 가공의 실제 : 방대한 데이터 요약

- 그룹 단위 통계(2) : 지역별 data를 시계열 형식의 연도별 세분화

```
Console C:/RSources/
> (avg_data = avocado %>% group_by(region, year) %>% summarise(v_avg=mean(Total.Volume), p_avg=mean(AveragePrice)))
`summarise()` has grouped output by 'region'. You can override using the `.groups` argument.
# A tibble: 216 x 4
# Groups:   region [54]
  region          year  v_avg p_avg
  <chr>         <int>   <dbl> <dbl>
1 Albany      2015  38749.  1.54
2 Albany      2016  50619.  1.53
3 Albany      2017  49355.  1.64
4 Albany      2018  64249.  1.44
5 Atlanta     2015 223382.  1.38
6 Atlanta     2016 272374.  1.21
7 Atlanta     2017 271841.  1.43
8 Atlanta     2018 342976.  1.29
9 BaltimoreWashington 2015 390823.  1.37
10 BaltimoreWashington 2016 393210.  1.59
# ... with 206 more rows
> |
```



5.4 데이터 가공의 실제 : 방대한 데이터 요약

- 그룹 단위 통계(3) : 지역별 ,년도별 및 유기농 여부를 기준으로 세분화

```
Console C:/RSources/
> avg_data = avocado %>% group_by(region, year, type) %>%
  summarise(v_avg=mean(Total.Volume), p_avg=mean(AveragePrice))
`summarise()` has grouped output by 'region', 'year'. You
can override using the `.groups` argument.
> avg_data
# A tibble: 432 x 5
# Groups:   region, year [216]
   region    year type      v_avg p_avg
  <chr>    <int> <chr>    <dbl> <dbl>
1 Albany   2015 conventional 76209.  1.17
2 Albany   2015 organic    1289.  1.91
3 Albany   2016 conventional 99453.  1.35
4 Albany   2016 organic    1784.  1.72
5 Albany   2017 conventional 95779.  1.53
6 Albany   2017 organic    2931.  1.75
7 Albany   2018 conventional 124161.  1.34
8 Albany   2018 organic    4338.  1.53
9 Atlanta  2015 conventional 440346.  1.05
10 Atlanta 2015 organic    6417.  1.71
# ... with 422 more rows
> |
```



■ 그룹 단위 통계(4)

- 방대한 샘플 데이터로부터 지역, 연도, 유기농 재배 여부를 기준으로 총 판매량과 평균 가격의 요약된 통계를 얻음
- 일반적으로 데이터 가공에 통계적 분석으로 기본적인 데이터의 특성을 파악한다.
- 다음으로 진행하는 것이 시각화이다.
- 이번 장에서는 시각화의 결과만 확인하고 제6장에서 시각화 방법을 학습한 예정임



5.4 데이터 가공의 실제 : 방대한 데이터 요약

■ 시각화

The screenshot shows the Excel 'Text Import Wizard' with three steps. Step 1 is selected, showing the file 'avocado_region_data' in the 'rdata' folder. Step 2 is active, showing the 'Text Import Wizard - 3단계 중 2단계' dialog. The 'Delimited' radio button is selected, and the 'Tab' checkbox under 'List of delimiters' is checked. Step 3 is visible in the background, showing the preview of the imported data.

Text Import Wizard - 3단계 중 2단계

데이터가 구분 기호로 분리됨(으)로 설정되어 있습니다.
데이터 형식이 올바르게 선택되었다면 [다음] 단추를 누르고, 아닐 경우 적절하게 선택하십시오.

원본 데이터 형식
원본 데이터의 파일 유형을 선택하십시오.

☒ 구분 기호로 분리됨(D) - 각 필드가 심표나 탭과 같은 문자로 나누어져 있습니다.
☐ 너비가 일정함(W) - 각 필드가 일정한 너비로 정렬

구분 시작 행(R): 1 원본 파일(Q): 949 : 구분 기호

☐ 내 데이터에 머리글 표시(M)

C:\rdata\avocado_region_data.txt 파일 미리 보기

1	region_v_avg p_avg
2	1 Albany 47537.8697337278 1.56103550295858
3	2 Atlanta 262145.32204142 1.33795857988166
4	3 BaltimoreWashington 398561.89147929 1.53423076923077
5	4 Boise 42642.5673076923 1.34813609467456
6	5 Boston 287792.854526627 1.5308875739645
7	6 BuffaloRochester 67936.3029585799 1.51683431952663

Text Import Wizard - 3단계 중 3단계

데이터의 구분 기호를 설정합니다. 미리 보기 상자에서 적용된 텍스트를 볼 수 있습니다.

구분 기호

☒ 탭(D) ☐ 연속된 구분 기호를 하나로 처리(R)

☐ 세미콜론(M)

☐ 심표(Q)

☐ 공백(S)

☐ 기타(Q):

텍스트 한정자(Q): "

데이터 미리 보기(P)

region_v_avg p_avg
1 Albany 47537.8697337278 1.56103550295858
2 Atlanta 262145.32204142 1.33795857988166
3 BaltimoreWashington 398561.89147929 1.53423076923077
4 Boise 42642.5673076923 1.34813609467456
5 Boston 287792.854526627 1.5308875739645
6 BuffaloRochester 67936.3029585799 1.51683431952663

5.4 데이터 가공의 실제 : 방대한 데이터 요약

■ 시각화

Console C:/RSources/

```
> avg_data <- avocado %>% group_by(region) %>% summarise  
(v_avg=mean(Total.Volume),p_avg=mean(AveragePrice))  
> write.table(avg_data, file="c:/rdata/avocado_region_data.txt",quote = F)
```

NO	region	v_avg	p_avg
1	Albany	47,537.87	1.56
2	Atlanta	262,145.32	1.34
3	BaltimoreWashington	398,561.89	1.53
4	Boise	42,642.57	1.35
5	Boston	287,792.85	1.53
6	BuffaloRochester	67,936.30	1.52
20	Indianapolis	89,536.66	1.31
21	Jacksonville	85,177.53	1.51
22	LasVegas	160,878.42	1.38
47	Southeast	1,820,231.98	1.40
48	Spokane	46,051.11	1.45
49	StLouis	94,890.04	1.43
50	Syracuse	32,374.76	1.52
51	Tampa	195,279.70	1.41
52	TotalUS	17,351,302.31	1.32
53	West	3,215,322.95	1.27
54	WestTexNewMexico	431,408.48	1.26



5.4 데이터 가공의 실제 : 방대한 데이터 요약

■ 시각화

```
Console C:/RSources/
> avg_data = avocado %>% group_by(region, year, type) %>% summarise
(v_avg=mean(Total.Volume), p_avg=mean(AveragePrice))
`summarise()` has grouped output by 'region', 'year'. You can overr
ide using the `.groups` argument.
> avg_data %>% filter(region != "TotalUS") %>% ggplot(aes(year, v_a
vg, col=type)) + geom_line()+facet_wrap(~region)
```

Plot Zoom



5.4 데이터 가공의 실제 : 방대한 데이터 요약

■ 데이터 정렬과 검색(2)

- 정렬과 검색을 통해 데이터를 자세히 관찰 가능
- arrange 함수 : arrange() orders the rows of a data frame by the values of selected columns.
- arrange 함수를 사용해 데이터를 총 판매량의 평균 가격을 기준으로 정렬하면, 판매량 순위는 물론 최대치를 기록한 연도와 지역을 알아낼 수 있음

Console C:/RSources/ ↗

```
> arrange(avg_data, desc(v_avg))
```

```
# A tibble: 432 x 5
```

```
# Groups:   region, year [216]
```

	region <chr>	year <int>	type <chr>	v_avg <dbl>	p_avg <dbl>
1	TotalUS	2018	conventional	42125533.	1.06
2	TotalUS	2016	conventional	34043450.	1.05
3	TotalUS	2017	conventional	33995658.	1.22
4	TotalUS	2015	conventional	31224729.	1.01
5	SouthCentral	2018	conventional	7465557.	0.806
6	west	2018	conventional	7451445.	0.981
7	California	2018	conventional	6786962.	1.08
8	west	2016	conventional	6404892.	0.916
9	west	2017	conventional	6279482.	1.10
10	California	2016	conventional	6105539.	1.05

```
# ... with 422 more rows
```

```
> |
```



■ 데이터 정렬과 검색(2)

- 데이터 셋에 중간 통계 값이 포함된 경우도 있으므로 주의가 필요
- 최댓값의 검색은 max 함수를 사용할 수도 있지만 속성값을 확인할 수 있는 arrange 함수를 사용하는 것이 더 안전

Console C:/RSources/ ➡

```
> avg_data1=avg_data %>%filter(region != "TotalUS")
> avg_data[avg_data$v_avg==max(avg_data1$v_avg), ]
# A tibble: 1 x 5
# Groups:   region, year [1]
  region      year type      v_avg p_avg
  <chr>      <int> <chr>      <dbl> <dbl>
1 SouthCentral 2018 conventional 7465557. 0.806
> |
```



5.4 데이터 가공의 실제 : 방대한 데이터 요약

- 데이터 정렬과 검색(2)
 - 활용 사례



정보 > 가집계 > 주요제품(1)

2004년 01월 31일

[단위 : 천C/S, %]

구 분			당 일 실 적		누 계 실 적						2004년 01월 실 행 목 표	
					당월실적		전년실적		전월실적			
				달성율		달성율		전년비		전월비		전년비
사이다	사이다	병	16.4	1.9	894.1	102.2	806.0	10.9	1,053.9	-15.2	875.0	8.6
		캔	6.4	1.5	386.4	91.6	431.4	-10.4	378.6	2.1	422.0	-2.2
		팩	12.1	1.6	755.7	102.1	809.6	-6.6	633.4	19.3	740.0	-8.6
		계	37.1	1.7	2,177.6	98.0	2,190.4	-0.6	2,242.5	-2.9	2,222.0	1.4
콜라	콜라	병	5.4	2.6	162.0	77.1	225.1	-28.0	245.1	-33.9	210.0	-6.7
		캔	10.5	2.9	326.5	91.5	365.0	-10.5	356.9	-8.5	357.0	-2.2
		팩	16.1	2.7	549.5	92.5	596.2	-7.8	567.9	-3.2	594.0	-0.4
		계	45.6	2.4	1,799.8	93.4	2,027.9	-11.2	2,031.4	-11.4	1,926.0	-5.0
		트위스트	0.4	1.4	17.6	60.8	73.8	-76.1	28.6	-38.3	29.0	-60.7
후레바	후레바	마운틴듀	1.4	1.8	72.4	95.3	66.9	8.3	95.3	-24.0	76.0	13.6
		미린다	4.2	2.7	121.8	79.1	115.6	5.4	186.1	-34.5	154.0	33.2
		밀키스	2.8	1.3	212.0	94.2	206.0	2.9	188.1	12.7	225.0	9.2
탄산	탄산	탄산	91.1	2.0	4,384.5	95.0	4,612.6	-4.9	4,764.8	-8.0	4,613.0	0.0
주스	100%	1.5병	0.2	0.1	289.9	96.6	304.6	-4.8	54.2	435.4	300.0	-1.5
		소병	1.4	1.0	143.8	103.5	147.2	-2.3	121.1	18.8	139.0	-5.6
		오렌지팩	5.7	3.0	179.0	94.2	232.6	-23.1	184.6	-3.1	190.0	-18.3
		직판	2.7	3.6	75.4	100.5	147.6	-48.9	71.0	6.2	75.0	-49.2



5.4 데이터 가공의 실제 : 방대한 데이터 요약

■ 데이터 정렬과 검색(2)

■ 활용 사례

일자	2018년 실적	2019년		계획 및 동기대비		
		사업계획	실적	계획비	전년비	
01월	7,355	7,477	7,429	99.4%	101.0%	
02월	7,213	7,354	6,997	95.1%	97.0%	
03월	8,140	8,174	7,683	94.0%	94.4%	
4월수량	9,034	8,578	9,467	110.4%	104.8%	
4월금액	10,635	12,967	13,692	105.6%	128.7%	
05월	8,504	8,494				
06월	8,691	8,705				
07월	8,559	8,526				
08월	8,011	8,328				
09월	8,226	8,220				
10월	8,233	8,644				
11월	8,270	8,221				
12월	8,168	7,978				
04월 누계	수량	7,643	7,904	7,471	94.5%	97.7%
	금액	255,490		260,316	0.0%	101.9%
총계	8,176	8,230	7,471	90.8%	91.4%	



■ Date형 데이터의 활용(1)

- Date형 속성은 1개월은 31일, 1년은 12개월로 구성되어 일반 숫자형처럼 처리할 경우 데이터 간의 간격이 일정하지 않아 시각화나 모델링 단계에서 잘못 적용될 수 있으므로 특별 처리 필요
- 속성 하나에 세 가지 속성(연-월-일) 정보를 포함하는 것이므로 적절히 가공하면 활용하면 데이터를 좀 더 세밀하게 분석할 수 있음

```
Console C:/RSources/
> head(avocado,6)
  X Date AveragePrice Total.Volume x4046 x4225
1 0 2015-12-27      1.33    64236.62 1036.74 54454.85
2 1 2015-12-20      1.35    54876.98  674.28 44638.81
3 2 2015-12-13      0.93   118220.22  794.70 109149.67
4 3 2015-12-06      1.08    78992.15 1132.00 71976.41
5 4 2015-11-29      1.28    51039.60  941.48 43838.39
6 5 2015-11-22      1.26    55979.78 1184.27 48067.99
  X4770 Total.Bags Small.Bags Large.Bags XLarge.Bags
1 48.16   8696.87   8603.62      93.25          0
2 58.33   9505.56   9408.07      97.49          0
3 130.50  8145.35   8042.21     103.14          0
4 72.58   5811.16   5677.40     133.76          0
5 75.78   6183.95   5986.26     197.69          0
6 43.61   6683.91   6556.47     127.44          0
  type year region
1 conventional 2015 Albany
2 conventional 2015 Albany
3 conventional 2015 Albany
4 conventional 2015 Albany
5 conventional 2015 Albany
6 conventional 2015 Albany
```



■ Date형 데이터의 활용(2)

- avocado 판매 정보를 이번에는 연도별 평균 대신 월별 평균으로 요약
- Date형 속성인 Date에서 month를 추출하려면 lubridate 라이브러리에서 제공하는 **month 함수** 사용

```
Console C:/RSources/
> library(lubridate)
> avg_data = avocado %>% group_by(region,year, month(Date),type) %
>% summarise(v_avg=mean(Total.Volume),p_avg=mean(AveragePrice))
`summarise()` has grouped output by 'region', 'year', 'month(Date)'. You can override using the `.groups` argument.
> head(avg_data,10)
# A tibble: 10 x 6
# Groups:   region, year, month(Date) [5]
  region year month(Date) type      v_avg p_avg
  <chr>   <int>   <dbl>   <chr>   <dbl> <dbl>
1 Albany  2015      1 conventional 42932.  1.17
2 Albany  2015      1 organic      1198.  1.84
3 Albany  2015      2 conventional 52343.  1.03
4 Albany  2015      2 organic      1334.  1.76
5 Albany  2015      3 conventional 50659.  1.06
6 Albany  2015      3 organic      1444.  1.83
7 Albany  2015      4 conventional 48594.  1.17
8 Albany  2015      4 organic      1402.  1.89
9 Albany  2015      5 conventional 97216.  1.26
10 Albany 2015      5 organic      1836.  1.94
> |
```



■ Date형 데이터의 활용(3)

- avocado 판매 정보를 이번에는 연도별, 월별 평균 대신 일별 평균으로 요약
- Date형 속성인 Date에서 day 를 추출하려면 lubridate 라이브러리에서 제공하는 day 함수

```
Console C:/RSources/
> avg_data = avocado %>% group_by(region,year, month(Date),day(Date), type) %>% summarise(v_avg=mean(Total.Volume),p_avg=mean(Average Price))
`summarise()` has grouped output by 'region', 'year', 'month(Date)', 'day(Date)'. You can override using the `.groups` argument.
> head(avg_data,10)
# A tibble: 10 x 7
# Groups:   region, year, month(Date), day(Date) [5]
  region year `month(Date)` day(Date) type      v_avg p_avg
  <chr>  <int>    <dbl>    <int>  <chr>    <dbl> <dbl>
1 Albany  2015         1         4 convent~ 40873.  1.22
2 Albany  2015         1         4 organic  1374.   1.79
3 Albany  2015         1        11 convent~ 41195.  1.24
4 Albany  2015         1        11 organic  1183.   1.77
5 Albany  2015         1        18 convent~ 44511.  1.17
6 Albany  2015         1        18 organic  1118.   1.93
7 Albany  2015         1        25 convent~ 45148.  1.06
8 Albany  2015         1        25 organic  1116.   1.89
9 Albany  2015         2         1 convent~ 70874.  0.99
10 Albany 2015         2         1 organic  1229.   1.83
> |
```



Thank you

